IMPERIAL COLLEGE OF SCIENCE AND TECHNOLOGY

Petroleum Engineering Section

Department of Mineral Resources Engineering

THE USE OF FINITE ELEMENT ANALYSIS IN PETROLEUM RESERVOIR SIMULATION

by

PRINS CHRYSANTHUS CASINADER, B.Sc., ARSM

A thesis submitted for the degree of Doctor of Philosophy of the University of London, and for the Diploma of Membership of Imperial College.

NOVEMBER 1979

THIS THESIS IS DEDICATED

TO

MY PARENTS AND SISTERS

ABSTRACT

This Thesis is concerned with the application of classical Galerkin-type finite elements to the simulation of immiscible two-phase (water/oil) displacements in petroleum reservoirs. The Chapeau (linear), Quadratic and Hermite Cubic elements have been used in 1 Dimension, while their tensor-product equivalents have been employed in the 2 Dimensional studies.

The 1 D results are compared with the analytical (Buckley-Leverett) solution, and an assessment is made of the sensitivity of the finite element solution to such factors as the grid size, the capacity matrix and, most important of all, the character of the mobility data which is reflected in the shape of the fractional mobility (f_w) curve. By using a variety of data sets, it is shown that the solution is good when the f_w curve has a monotonically decreasing slope, but that it is very unreliable when the curve is strongly S-shaped. To ensure numerical stability under all conditions, two possible alternatives have been investigated:

(1) The use of a modified f_w curve, as proposed by Dalen, and

(2) The use of the original f_w curve, together with an artifice such as, either:

    (a) upstream Gauss-point relative permeabilities, or

    (b) a fictitious capillary pressure gradient, as used by Spivak et al.

A method is presented for estimating the latter, based on the mobility data.

The 2-Dimensional studies have been conducted on a quarter five-spot, using a diagonal grid. The pressure distribution compares favourably with Muskat's analytical solution, except in the neighbourhood of wells. The saturation contours generally display the expected features but vary in internal detail, such as front steepness and the degree of oscillation, according to the type of finite element being used.

Finally, the breakthrough oil recoveries are shown to be in reasonably good agreement with those obtained from semi-analytical methods, particularly that of Higgins and Leighton.

ACKNOWLEDGEMENTS

I am deeply grateful to the following who have, either directly or indirectly, contributed towards this project.

- My Supervisor, Mr D C Wilson
- Professor C G Wall of this Department and Mr M Ford of British Gas Corporation
- Mr M H Goldwater and Dr P A Collins of London Research Station, Fulham
- Dr R A Dawe and Dr R J Wright of this Department
- Dr M J M Bernal and Dr C Elliott of the Mathematics Department
- My colleague Mr T C Tan, who so generously supplied the nine-point finite difference model
- British Gas Corporation (R and D) for financing this project
- Mrs E Kitson for typing the manuscript
- Energy Resources Consultants Ltd., for kindly permitting me to use their photocopying facilities

CONTENTS

Page

LIST OF GRAPHS AND MAPS

G27	Five-spot pressure distribution (Muskat)
G28a,b,c	Finite element pressure distribution at t = 30 days
G29a,b,c	Pressure profile along diagonal
G30a,b,c	Pressure profile along side
G31(1 to 10)	5-spot simulation, Chapeau
G32(1 to 10)	5-spot simulation, Quadratics
G33(1 to 10)	5-spot simulation, Hermite Cubics
G34(1 to 10)	5-spot simulation, 9-pt. F.D.
G35	Rate variation in Higgins and Leighton model
G36	Producing water-cut vs time, Chapeau
G37	Producing water-cut vs time, Quadratics
G38	Producing water-cut vs time, Hermite Cubics
G39	Producing water-cut vs time, 9-pt. F.D.
G40	Producing water-cut vs time, Higgins and Leighton

LIST OF TABLES

CHAPTER 1

An Introduction to the methods for analysing the performance of Petroleum Reservoirs

A petroleum reservoir is an underground accumulation of hydrocarbons within the pore spaces of a sedimentary rock. The hydrocarbons are collectively known as petroleum, while the rock which contains them is called reservoir rock.

1.1 Reservoir rocks

These consist largely of sandstones and limestones laid down on the earth's crust millions of years ago by sedimentary processes. Sedimentary rocks possess two important physical properties:

(a) Porosity (ϕ), which is the fraction of the rock volume that is pore space; and

(b) Permeability (K), which is a measure of the rock's ability to transmit fluids.

Thus, when a porous and permeable "reservoir rock", such as sandstone, is overlain and sealed by an impervious "cap rock" (eg shale), and has itself a structure that is capable of accommodating and retaining fluids, this combination of features is called a "trap" or a potential reservoir.

1.2 Reservoir fluids

A mixture of hydrocarbons can exist under subsurface conditions either in the form of free gas, or as a liquid (called oil), or both.

The saline water which often underlies the reservoir is referred to as an aquifer.

The surfaces of the sand grains show varying degrees of attraction to the phases they contain, and this preferential attraction is expressed in terms of the "wettability" of the porous medium. Generally, the water phase is more wetting than the oil phase, which, in turn, is more wetting than the gas, although this sequence is by no means universal. Due to this phenomenon of wettability, the migration of hydrocarbons into a water-wet rock does not result in a complete expulsion, from the pores, of the original water. Instead, the capillary forces arising from this preferential attraction succeed in retaining an appreciable portion of the water in the form of films surrounding the grains. This water is called "interstitial" or "connate" water, and the fraction of pore space it occupies the connate water saturation (S_{wc}).

The compositions of the gas and oil phases (the latter in particular) are very complex, and detailed thermodynamic calculations are difficult to perform, and are rarely used. For practical engineering purposes, however, sufficiently accurate results can be obtained by adopting a highly simplified concept for their compositions. The gas phase is treated as a single component which obeys the imperfect gas law ($PV = nZRT$). The reservoir oil, on the other hand, is regarded as being made up of two components:

(a) Stock tank oil (or dead oil), and

(b) A quantity of gas which is dissolved in this stock tank oil. This is known as solution gas.

For a given gas/oil system, the amount of gas which is soluble

in each stock tank volume of oil is a monotonically increasing function of pressure. The pressure at which all the available gas can be dissolved is called the saturation pressure, or the Bubble Point Pressure (P_b) of the system. Furthermore, due to the swelling effect of the dissolved gas, a unit volume of stock tank oil occupies a larger volume under reservoir conditions, and this volume is termed the "formation volume factor" (B_o).

1.3 A review of reservoir production mechanisms

The principal driving force responsible for the subsurface movement of fluids is the reservoir pressure. Thus, when an outlet is created, pressure gradients are set up within the porous medium which cause the fluids to flow towards the well.

A field may be developed by a variety of mechanisms which differ in the way in which energy is supplied for the extraction of hydrocarbons. The simplest of these is the Primary (or Depletion) mechanism, in which the required energy is derived entirely from within the reservoir by the expansion of its fluids, particularly the solution gas. Depending on the amount of gas dissolved in the oil, recoveries range from a few per cent (if little gas is present) to in excess of 30 per cent if there is a high GOR and gravity segregation. An initial gas cap can result in even higher recovery factors because of the extra energy available for expansion.

A second mechanism occurs if the reservoir is in communication with a reasonably large aquifer which can respond to oil production by providing a substantial influx of water into the oil zone, thereby helping to stabilize the reservoir pressure and displace more oil. This mechanism is known as "natural water drive".

The existence of such an aquifer cannot normally be relied upon. Consequently, it follows that in order to ensure the maintenance of reservoir pressure, fluids must be injected into the reservoir from the surface. Gas and water have been used extensively, water being the more popular because it is both cheap, and easy to obtain. Presently, many oil fields, particularly the undersaturated ones, are subjected to powered water injection (ie waterflooding) early in their lives. The study of multi-phase immiscible flow therefore forms an integral part of reservoir engineering.

1.4 Reservoir performance analysis

As the subject of petroleum engineering developed into a science, the need arose to quantify the behaviour of oil and gas fields. By considering the reservoir to be a "tank", with known initial contents, engineers in the early part of this century were able to formulate "material balance" equations, which related the cumulative volumes of the injected and produced fluids to the change in the contents of the reservoir. Such equations were merely an algebraic statement of the fact that the recovery of hydrocarbons is the combined effect of the drive mechanisms to which the reservoir is subject - namely, the expansion of hydrocarbons and connate water, influx of water from an aquifer, and the displacement by injected fluids. Although these methods (based on the tank model approach) were certainly useful, they provided little in the way of describing what was actually happening within the reservoir. The spatial variation of pressure, and the distribution of fluids, for example, could not be determined except as gross averages over the whole reservoir.

The next step was to consider the dynamics of fluid flow in reservoirs. The theory behind this is simple. It consists of representing the porous medium by a uniform continuum, and setting up a partial differential equation (PDE) in space and time (x,y,t), to express the material balance condition at each point of the flow field.

For single phase flow, the PDE contains a single dependent variable, namely pressure. Furthermore, the occurrence of time as an independent variable is governed by the compressibility of the fluid. Inclusion of the fluid compressibility creates a time-dependent PDE. If, however, for simplifying reasons, the compressibility is omitted, then the PDE will be time-invariant, and involve the spatial co-ordinates only. The latter case (i.e. single phase, incompressible flow) is also known as steady state flow.

Problems involving steady state flow are comparatively easy to solve. 1 Dimensional flow is trivial, while, in 2 Dimensions, the application of methods based on potential theory have enabled solutions to be obtained for a wide range of problems, as is borne out by the work of Muskat (43).

Considerable progress has also been achieved in solving single phase, time-dependent problems, despite their greater complexity. Developments in this area were motivated primarily by the need for understanding and interpreting transient pressure responses occurring in the neighbourhood of wells. The governing PDE is often formulated in radial coordinates (r,t), and, by employing such mathematical tools as integral transforms and Bessel functions, it has been possible to generate solutions in the form of infinite series, which can subsequently be truncated and simplified to yield elegant results. This branch of reservoir engineering is known as "Pressure Analysis".

The analytical methods available for solving two-phase flow equations, however, are comparatively few, and limited in scope. The reasons for this are:

(a) The system is now governed by two simultaneous PDE's, each of which involves the two dependent variables - pressure and saturation.

(b) These differential equations are strongly non-linear in saturation.

The single most important contribution to this field has been the work of Buckley and Leverett (7) who developed an elegant analytical method for solving the incompressible water/oil displacement problem in a 1 D system. In 2 Dimensions, however, the problem is less tractable. The potential theory methods of Muskat could be applied only under very exceptional circumstances, such as when the two phases had constant and equal mobilities. Practising engineers had therefore to be content with adapting the linear Buckley-Leverett model by combining it with whatever laboratory data that was available on 2 D water-floods, in order to meet their specific needs. Such was the general state of affairs until advances in electronics opened up other possibilities.

1.5 Reservoir Simulation

The advent of high speed digital computers in the 1950's ushered in a new branch of mathematics known as numerical analysis, a significant part of which was devoted to obtaining approximate solutions of both ordinary and partial differential equations. With this powerful technique, it became a relatively easy matter to compute,

with sufficient accuracy, the solutions to differential equations which had previously defied analysis. All branches of engineering were able to benefit greatly, and reservoir engineering was no exception. Thus, the application of numerical methods to solve for the pressure and saturation distributions in reservoirs gained immense popularity, and acquired the name of reservoir simulation.

The aim of all numerical models is the same - namely, to divide the region of interest into a mesh, and to compute the (approximate) solution to the differential equation at these mesh points. To do this, the given differential equation is transformed into a difference equation at each point, so that as many simultaneous (algebraic) equations are generated as there are mesh points. The difference equations are such that each mesh point is connected to some or all of the neighbouring points, depending on the type of method used for this transformation. Having set up the complete matrix, the system is then solved by any of several well-known methods (e.g. Gaussian elimination, successive over relaxation, etc.) to determine the unknown values at the nodes. If the differential equation is time-dependent (as is usually the case), the entire solution procedure is repeated over successive time-intervals, for as many time-steps as required.

Numerical simulators are considerably more versatile than analytical models in that they do away with many of the restrictions and simplifying assumptions that characterise the latter. The following features, in particular, can be readily handled by simulators:

(1) Reservoirs having irregular (i.e. non-rectangular) geometry;

(2) Variation in reservoir properties (e.g. porosity, permeability, thickness etc.) over large areas; and

(3) Multi-phase flow.

There are at least two fundamental numerical techniques on which simulators can be based. These are:

(a) Finite Difference Method (FDM), and

(b) Finite Element Method (FEM).

These methods differ principally in the approach used for transforming the continuous differential equation into a discrete difference equation.

1.5.1 Finite Differences

This method is the older and the most widely known numerical technique, and it utilizes the Taylor series to approximate the derivative terms occurring in the differential equation. In conventional finite difference simulators, the reservoir is divided into rectangular blocks, and the unknown variables are taken to be the average pressure and saturation for each block. (These are also called block-centred values). The difference equations for these unknowns turn out to be equivalent to material balance statements for each phase at each of the blocks. Thus, for example, the conservation equation for phase n at block i, takes the form:

$$\sum_j (q_{in})_j + (q_{in})_{I/P} = \left[\begin{array}{l}\text{Rate of accumulation}\\ \text{of phase } n \text{ in}\\ \text{block } i.\end{array}\right], \quad (1.1)$$

where the $(q_{in})j$ are the rates at which phase n is entering block i from its neighbouring blocks j, and $(q_{in})I/P$ is the rate at which the phase is injected into (or produced from) block i, with the convention that the injection term is taken as positive. In the classical 2 D

versions, each block is in communication with its adjacent blocks in the x and y directions (Figure 1.1a), so that each equation is associated with the pressures and saturations in five blocks. While this "five-point difference scheme" has been very popular in reservoir simulation, it has been shown recently (61) that better results can be achieved by using an alternative "nine-point formula" which allows for diagonal communication between blocks (Figure 1.1b).

The application of finite differences to simulation has advanced to the stage that most oil companies now have their own 3 dimensional, 3 phase models, constructed with varying degrees of sophistication. The fundamentals of this technique are described in several texts (Crichlow (15), Peaceman (47), and Aziz and Settari (4)).

Despite their great value in reservoir engineering, finite difference models are not without limitations. The main difficulty is the phenomenon of "numerical dispersion" which leads to a distortion of the saturation distribution, so that the "numerical" saturation profiles often exhibit a considerable amount of spatial smearing, even when, from theoretical analysis, a steep profile (or "front") is known to exist. To control or reduce numerical dispersion, one would have to resort to the costly operation of using a fine mesh. Furthermore, with the use of rectangular blocks, the representation of curved reservoir boundaries and faults becomes difficult, and the nearest approximation of these can be achieved only by the use of a "stepped pattern", as shown in Figure 1.2. These factors have stimulated the search for alternative numerical techniques in recent years, and have thus led to the investigation of Variational (or Finite Element) methods.

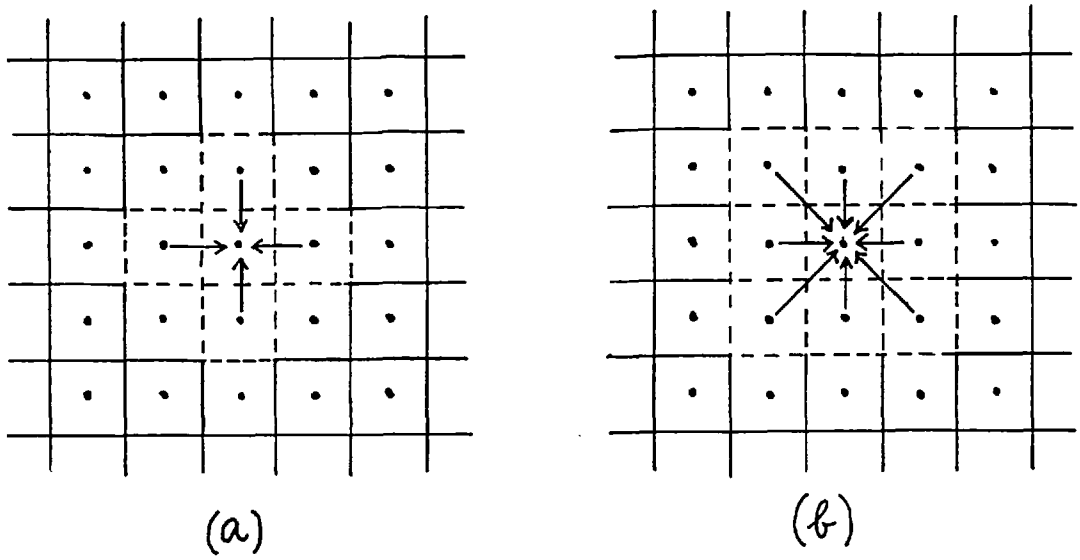


Figure 1.1 Communication between blocks in Finite Difference Simulators.

(a) 5-point scheme (classical)
(b) 9-point scheme

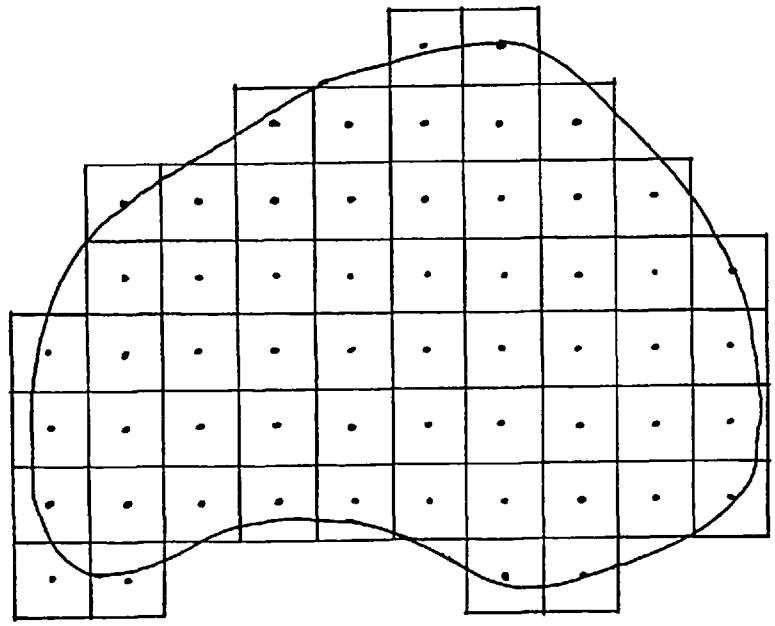


Figure 1.2 "Stepped" pattern for approximating a curved reservoir boundary with a rectangular finite difference grid.

1.5.2 Finite Elements

In this technique, the reservoir is divided into a mesh of finite elements of any desired shape (triangles, rectangles etc.), and the pressure and saturation are approximated by piecewise polynomials over these elements. The unknown parameters are once again taken to be the nodal values, but the transformation from the differential equation to the difference equation is carried out by integration processes (as opposed to the Taylor series methods used in finite differences). The resulting matrices are less sparse than those of finite differences, and hence the solution procedures are correspondingly more sophisticated.

The method, therefore, seemed promising for reservoir simulation from the point of view of accurately representing both the reservoir boundary, as well as the sharp saturation profiles that characterize many waterfloods. Although this technique originated in the field of structural mechanics in the 1960's, its application to fluid mechanics, and to reservoir simulation in particular, took place much later, and nearly all the available literature in the latter area is confined to the past decade.

The object of this thesis is to demonstrate the application of finite elements to two-phase immiscible (water/oil) displacement in reservoirs. The numerical examples that have been considered here are of a simple character - namely, the 1 D linear displacement model, and the 2 D "quarter five-spot" model. Throughout this project, attention has been focussed on the nature of the finite element solution, and the sensitivity of the method to various factors, rather than on the construction of a commercially usable model. It is hoped that the

results presented herein will help to elucidate some of the characteristic features of this method, and explain the diversity of opinion prevailing among the many authors, concerning the feasibility of the finite element method in reservoir simulation.

1.6 Arrangement of Chapters

In Chapter 2, the basic differential equations governing two-phase immiscible flow in porous media are derived from first principles. Chapter 3 describes some of the analytical and semi-analytical methods that are available in the literature (i.e. methods other than simulation), with special emphasis on those which are used later in the thesis for comparing with the finite element results. Chapter 4 presents the theory underlying the finite element method, and, beginning with a broad overview of the field, it proceeds to concentrate on those elements which have actually been employed in this study. The chapter also includes derivations of the appropriate shape functions and matrix equations, together with a discussion of the solution techniques and other relevant features of the models. Chapter 5 describes a program written for contouring 2 D finite element results. This package has been used for generating all the contour maps shown in this thesis. Chapter 6 opens with a summary of the work of other authors in this field, and moves on to discuss the results of this project, and to compare them with suitable analytical results (described in Chapter 3). Chapter 7 concludes the thesis by listing the inferences that can be drawn from these results, and by pointing out some of the areas that still remain to be either explored or developed.

CHAPTER 2

Basic Equations of Two Phase Flow in Reservoirs

The simultaneous flow of two immiscible phases (oil and water) through a porous medium is governed by three basic equations:

(1) a transport equation;

(2) an equation of state; and

(3) a continuity (or mass conservation) equation.

These combined equations (written for each phase), together with the appropriate initial and boundary conditions, have to be solved, analytically and/or numerically, over the whole reservoir.

2.1 Choice of variables

The reservoir engineer is interested primarily in four unknowns (or variables) which jointly determine the flow characteristics of the system. These are:

(1) the oil phase pressure Po ;

(2) the water phase pressure Pw ;

(3) the oil saturation So ; and

(4) the water saturation Sw .

These physical quantities are, however, interrelated.

(a) The saturations satisfy the volumetric identity :

$$S_o + S_w = 1 \; . \qquad (2.1)$$

(b) Since the fluids are immiscible, the pressure in the wetting phase (usually water) differs from that in the non-wetting phase by an

amount known as the capillary pressure, P_c, which is dependent on the fluid saturations. Thus :

$$P_w = P_o - P_c(S_w) \qquad (2.2)$$

It follows therefore that only two of the four parameters are independent, and, consequently, the choice of a suitable pair is left to the discretion of the authors. Some authors have chosen, as their main variables, one pressure and one saturation (e.g, P_o and S_w) [16], [36], while others have used either the two pressures (P_o and P_w) [39], or any linearly independent combinations thereof (e.g $\frac{1}{2}(P_o+P_w)$ and $\frac{1}{2}(P_o-P_w)$) [38]. In the two latter cases, the saturations are calculated from the pressure difference via the capillary pressure relationship (2.2). This approach, however, has the disadvantage that it cannot be applied to a hypothetical situation where capillary pressures are either small or absent altogether (thus implying $P_o=P_w$). Moreover, since the fluid distribution within the reservoir is one of the key items of interest, it seems desirable to include one of the saturations as a direct variable.

In our treatment, therefore, we shall adopt P_o and S_w as the principal unknowns to be solved for. The region of the reservoir to be simulated will be referred to as the domain (Ω), and its boundary will be denoted by ($\partial\Omega$). The basic flow equations will now be presented in a 2-dimensional form, for an areal domain of uniform thickness h.

2.2 Transport equation

By analogy with other transport phenomena, (such as diffusive

heat - and mass - transfer), the flow of a single fluid through a porous medium can be quantified by invoking the concept of a "potential field". The linear relationship which normally exists between the volumetric flux vector and the potential gradient is known as Darcy's law:

$$\underline{u} = -\frac{1}{\mu}\,\underline{K}\,\underline{\nabla}\,\Phi \qquad (2.3)$$

where the potential, Φ, is the pressure, corrected hydrostatically to an arbitrary datum.

$$\Phi = p - \rho g z \qquad (2.4)$$

In these equations:

ρ = average density of fluid (g/cm^3) ;

g = gravitational constant ;

z = depth below datum (cm) ;

μ = fluid viscosity (cp) ;

$\underline{u}$ = volumetric flux ($cm^3/cm^2 S$) ;

and

$$\underline{K} = \begin{bmatrix} K_x & 0 \\ 0 & K_y \end{bmatrix}$$ = permeability tensor (Darcy).

K_x and K_y are the absolute permeabilities of the porous medium in the x and y directions, respectively which (for convenience) are assumed to coincide with the principal axes of the rock. For an isotropic medium, we have:

$$K_x = K_y$$

When two or more fluids flow simultaneously, however, it is found that the motion of each fluid is more impeded than if it alone were present, and that the extent of this departure from Darcy's law is controlled by the fluid saturations. Thus, in order to make allowance for the increased resistance of the porous medium, it is necessary to incorporate into equation (2.3) a saturation dependent factor k_r ($0 \leq k_r \leq 1$) called the relative permeability. Introduced originally in the 1940's, this concept of relative permeability has come to be regarded as the focal point of displacement theory in porous media.

With the above modification, the multi-phase version of Darcy's law takes the form:-

$$\begin{aligned} \underline{u} &= -(k_r/\mu)\,\underline{K}\,\underline{\nabla}\,\Phi \\ &= -\lambda_r \underline{K}\,\underline{\nabla}\,\Phi \end{aligned} \tag{2.5}$$

where λ_r ($= k_r/\mu$) is termed the relative mobility of the fluid. Since the oil and water phases have each their own relative permeability curves (k_{ro} vs S_w and k_{rw} vs S_w), their relative mobilities will also be different, and hence the modified Darcy equation has to be applied individually to the two phases - Thus:

$$\begin{cases} \lambda_{ro} = k_{ro}/\mu_o \\ \lambda_{rw} = k_{rw}/\mu_w \end{cases} \tag{2.6}$$

and so the volumetric fluxes are given by:

$$\underline{u}_o = -\lambda_{ro}\,\underline{K}\,\underline{\nabla}\,\Phi_o \quad , \text{ and} \tag{2.7}$$

$$\underline{u}_w = -\lambda_{rw}\,\underline{K}\,\underline{\nabla}\,\Phi_w \, . \tag{2.8}$$

We can also expand the potential gradients in the two fluids (in terms of the basic variables p_o and S_w) by combining the equations (2.4) and (2.2):

$$\underline{\nabla}\Phi_o = \underline{\nabla}p_o - \rho_o g \underline{\nabla} z \; ; \quad \text{and} \tag{2.9}$$

$$\begin{aligned}\underline{\nabla}\Phi_w &= \underline{\nabla}(p_w - \rho_w g z) \\ &= \underline{\nabla}p_o - \underline{\nabla}p_c - \rho_w g \underline{\nabla} z \\ &= \underline{\nabla}p_o - p_c' \underline{\nabla}S_w - \rho_w g \underline{\nabla} z\end{aligned} \tag{2.10}$$

where $p_c'(=dp_c/dS_w)$ is the slope of the capillary pressure curve with respect to saturation. Finally, substituting equations (2.9) and (2.10) into (2.7) and (2.8) respectively, we obtain the basic transport equations for the oil and water phases:

$$\underline{u}_o = -\lambda_{ro}\underline{k}\left[\underline{\nabla}p_o - \rho_o g \underline{\nabla} z\right], \quad \text{and} \tag{2.11}$$

$$\underline{u}_w = -\lambda_{rw}\underline{k}\left[\underline{\nabla}p_o - p_c'\underline{\nabla}S_w - \rho_w g \underline{\nabla} z\right] \tag{2.12}$$

The relative permeabilities (k_{ro} and k_{rw}) and the capillary pressure, p_c, are, in general, strongly non-linear functions of S_w, and are derived from laboratory experiments. Figure 2.1 shows a schematic plot of these quantities for a typical water-wet reservoir rock.

2.3 Equation of State

Due to the compressibility of the fluids and the variation in pressure over the domain, we need an equation which expresses the volume of the fluid as a function of pressure. For this purpose, it is convenient to use the formation volume factor, B, which is defined as:

$$B = \frac{V_{res}}{V_{sc}} = \frac{\text{volume of a given mass at reservoir conditions}}{\text{volume of the same mass at standard conditions}} \tag{2.13}$$

in which "standard" (or "stock tank") conditions refer to 1 atmosphere and 60^{o}F. B is thus a "normalised" volume, and is used for representing volume changes within the reservoir which, if thermal expansion effects are ignored (as is usually the case), will be attributable entirely to the pressure.

We can therefore write the compressibility as

$$c = -\frac{1}{V}\frac{dV}{dp} = -\frac{1}{B}\frac{dB}{dp}. \tag{2.14}$$

For fluids of constant compressibility, such as water and undersaturated oil, equation (2.14) can be integrated directly to yield:

$$\begin{aligned} B &= B_i e^{c(p_i - p)} \\ &= B_i e^{c\Delta p} \\ &= B_i\left[1 + c\Delta p + \frac{1}{2!}c^2(\Delta p)^2 + \cdots\right], \end{aligned} \tag{2.15}$$

where (p_i, B_i) are arbitrary reference values (at the reservoir temperature), and

$$\Delta p = p_i - p .$$

If, in addition, the compressibility is small (so that $|c\Delta p| << 1$), the expansion (2.15) reduces to:

$$B = B_i(1 + c\Delta p) \tag{2.16}$$

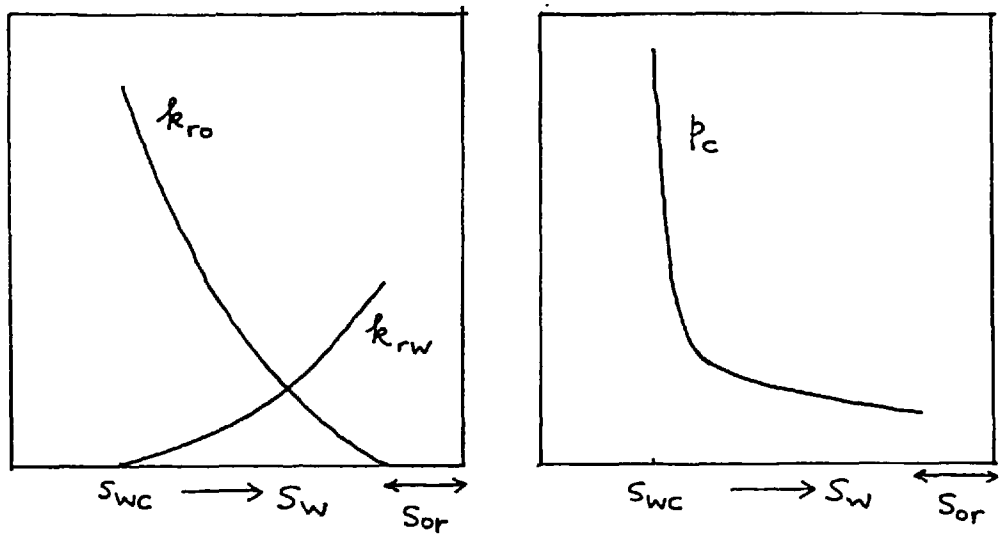


Figure 2.1: Typical relative permeability and capillary pressure curves for a water wet system (schematic).

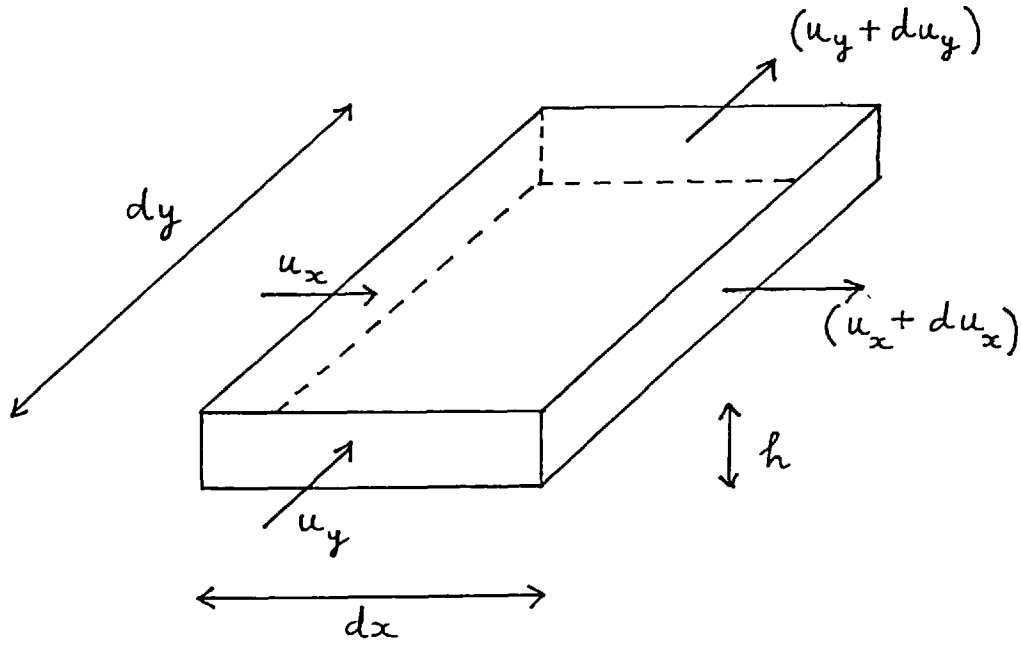


Figure 2.2: A differential element of the domain, used for deriving the continuity equation.

Thus, we ultimately have

$$B_o = B_{oi}(1 + c_o \Delta p) \quad \text{for oil,} \tag{2.17}$$

$$\text{and} \quad B_w = B_{wi}(1 + c_w \Delta p) \quad \text{for water.} \tag{2.18}$$

2.4 Continuity Equation

This is a differential equation which expresses the fact that the mass (or, alternatively, volume at standard conditions) of each phase is conserved throughout the reservoir. It may easily be derived by considering an infinitesimally small rectangular element ($d\Omega$) of dimensions dx, dy and thickness h. (Figure 2.2)

The volume of this element is, clearly,

$$d\Omega = h\,dx\,dy \tag{2.19}$$

Any differential change in the fluid mass (generation or elimination) occurring within this element can be attributed to three causes:

(a) dm_t due to transport (i.e. flux - or velocity gradients)

(b) dm_s due to sources or sinks (e.g. wells)

(c) dm_c due to change in the "contents" of the element (e.g. S_w change etc).

We shall consider these terms one by one.

(a) The Divergence Term

As shown in Figure 2.2, let us suppose that the fluid enters

the element with velocity components $[u_x , u_y]$ and leaves with components $[(u_x + du_x) , (u_y + du_y)]$. Using the definition of the formation volume factor B (equation 2.13), we can convert these velocities to "standard conditions" in the form:

$$\left[\frac{u_x}{B} , \frac{u_y}{B}\right] \text{ and } \left[\left\{\frac{u_x}{B} + d\left(\frac{u_x}{B}\right)\right\}, \left\{\frac{u_y}{B} + d\left(\frac{u_y}{B}\right)\right\}\right]$$

The infinitesimal gain in mass acquired by the element over a small time-interval dt is therefore:

$$dm_t = - h\, d\left(\frac{U_x}{B}\right) dy\, dt \; - \; h\, d\left(\frac{U_y}{B}\right) dx\, dt . \qquad (2.20)$$

But from elementary calculus we know that

$$d\left(\frac{U_x}{B}\right) = \frac{\partial}{\partial x}\left[\left(\frac{U_x}{B}\right)\right] dx \quad \text{and} \quad d\left(\frac{U_y}{B}\right) = \frac{\partial}{\partial y}\left[\left(\frac{U_y}{B}\right)\right] dy \qquad (2.21)$$

So from the equations (2.20 and (2.21), we obtain:

$$dm_t = -\left[\frac{\partial}{\partial x}\left(\frac{u_x}{B}\right) + \frac{\partial}{\partial y}\left(\frac{u_y}{B}\right)\right] h\, dx\, dy\, dt$$

or

$$dm_t = -\underline{\nabla}\cdot\left(\frac{\underline{u}}{B}\right) h\, dx\, dy\, dt ; \qquad (2.22)$$

which is the differential increase in stock tank volume due to the divergence of the flow field.

(b) Source/Sink term

If fluid is being supplied to (or removed from) the element by an external source (or sink) of intensity σ (expressed in standard volume per unit reservoir volume per unit time), then the resultant increase in stock tank volume of the fluid brought about by this

means over a time interval dt is:

$$dm_s = \sigma h \, dx \, dy \, dt, \tag{2.23}$$

where we have adopted the sign convention that σ is positive for injection and negative for production.

(c) Capacity term

The stock tank volumes of fluid that are contained within the element is given by:

$$\phi h \left(\frac{S}{B}\right) dx \, dy .$$

Hence the infinitesimal quantity of mass released inside the element due to a change in its contents over a time interval dt is:

$$dm_c = - \phi h \, d\left(\frac{S}{B}\right) dx \, dy$$

or

$$dm_c = - \phi h \frac{\partial}{\partial t}\left(\frac{S}{B}\right) dx \, dy \, dt. \tag{2.24}$$

Since the law of conservation requires the nett increase in mass to be zero, we have:

$$dm_t + dm_s + dm_c = 0 \tag{2.25}$$

and so, combining equations (2.22), (2.23) and (2.24), we finally obtain:

$$\underline{\nabla} \cdot \left(\frac{\underline{u}}{B}\right) + \phi \frac{\partial}{\partial t}\left(\frac{S}{B}\right) - \sigma = 0 \tag{2.26}$$

which is the basic continuity equation.

Since the numerical models in this thesis are formulated in terms of "reservoir units" rather than "standard units", we shall expand the above equation (2.26) in order to convert it to the appropriate form. Thus:

$$\frac{1}{B}(\underline{\nabla}\cdot\underline{u}) + \underline{\nabla}\left(\frac{1}{B}\right)\cdot\underline{u} + \frac{\phi}{B}\left(\frac{\partial S}{\partial t}\right) + \phi S\frac{\partial\left(\frac{1}{B}\right)}{\partial t} - \sigma = 0. \quad (2.27)$$

Using the definition of compressibility (equation 2.14), the terms involving B can be simplified to yield:

$$\underline{\nabla}\left(\frac{1}{B}\right) = \frac{d\left(\frac{1}{B}\right)}{dp}\underline{\nabla}p = -\frac{1}{B^2}\frac{dB}{dp}\underline{\nabla}p = \frac{c}{B}\underline{\nabla}p; \quad (2.28)$$

and

$$\frac{\partial\left(\frac{1}{B}\right)}{\partial t} = \frac{d\left(\frac{1}{B}\right)}{dp}\frac{\partial p}{\partial t} = -\frac{1}{B^2}\frac{dB}{dp}\frac{\partial p}{\partial t} = \frac{c}{B}\frac{\partial p}{\partial t}. \quad (2.29)$$

Substituting equations (2.28) and (2.29) into (2.27), and multiplying throughout by B, we get :

$$\underline{\nabla}\cdot\underline{u} + c\,\underline{\nabla}p\cdot\underline{u} + \phi\frac{\partial S}{\partial t} + \phi S c\frac{\partial p}{\partial t} - \sigma B = 0, \quad (2.30)$$

which is the equivalent of (2.26) in reservoir units.

In constructing the 1 D numerical models, it was decided to ignore the term c $\underline{\nabla}p \cdot \underline{u}$ (which is the one associated with $\underline{\nabla}\left\{\frac{1}{B}\right\}$ in equation 2.27). The removal of this non-linear term was intended merely to facilitate the programming process, although, in retrospect, its inclusion would have been desirable at least in the form of an

"explicit" (right-hand-side) term. The other term $\phi S_c \left\{\frac{\partial p}{\partial t}\right\}$ (which in the case of single phase flow, governs the rate of pressure change with time) was retained in the continuity equation as the only term which accounts for compressibility effects.

However, the saturation solutions are relatively insensitive to the fluid compressibilities, provided they are small (as is the case with liquids). Therefore, the omission alluded to above has a negligible effect on the saturation profiles, and consequently we shall adopt this simplified continuity equation as the basis for further development.

Hence, the final equations for oil and water are:

$$\underline{\nabla} \cdot \underline{u}_o + \phi \frac{\partial S_o}{\partial t} + \phi S_o c_o \frac{\partial p_o}{\partial t} - \sigma_o B_o = R_o = 0; \quad (2.31)$$

and

$$\underline{\nabla} \cdot \underline{u}_w + \phi \frac{\partial S_w}{\partial t} + \phi S_w c_w \frac{\partial p_o}{\partial t} - \sigma_w B_w = R_w = 0; \quad (2.32)$$

in which a further simplifying approximation can be observed, namely that

$$\frac{\partial p_w}{\partial t} \approx \frac{\partial p_o}{\partial t} .$$

2.5 Treatment of well terms

The nature of the source/sink term, σ, deserves attention since, at any point P (x_o, y_o) of the domain, there are three distinct possibilities:

(1) If no source or sink exists at P, then obviously $\sigma = 0$.

(2) If P belongs to a subregion w over which there is a continuous

(or distributed) flux of material, then σ is a finite quantity, and represents the volumetric intensity of the source. The cumulative rate of injection (or production) of fluid over this subregion can be found by direct integration:

$$q = \int_w \sigma \, d\Omega = h \iint_{\substack{\text{areal} \\ \text{limits} \\ \text{of } w}} \sigma(x,y) \, dx \, dy \qquad (2.33)$$

(3) If P represents a concentrated source (e.g. a well in an areal model) then, in the vicinity is P, the intensity σ assumes a singularity, which can be expressed in terms of the "Dirac Delta" function. Thus, if the well rate is q (standard volumes per unit time), then:

$$\sigma = (q/h)\,\delta[(x-x_o)(y-y_o)]. \qquad (2.34)$$

It is a well known property of the Delta function that its integral over any region which contains the singularity is unity. So, integrating equation (2.34) we obtain:

$$\int_\Omega \sigma \, d\Omega = \left(\frac{q}{h}\right) h \iint \delta[(x-x_o)(y-y_o)]\,dx\,dy = q, \qquad (2.35)$$

which verifies the statement that the integral of the source intensity yields the rate of injection (or production) at the well.

Figure 2.3 illustrates the two types of sources described above.

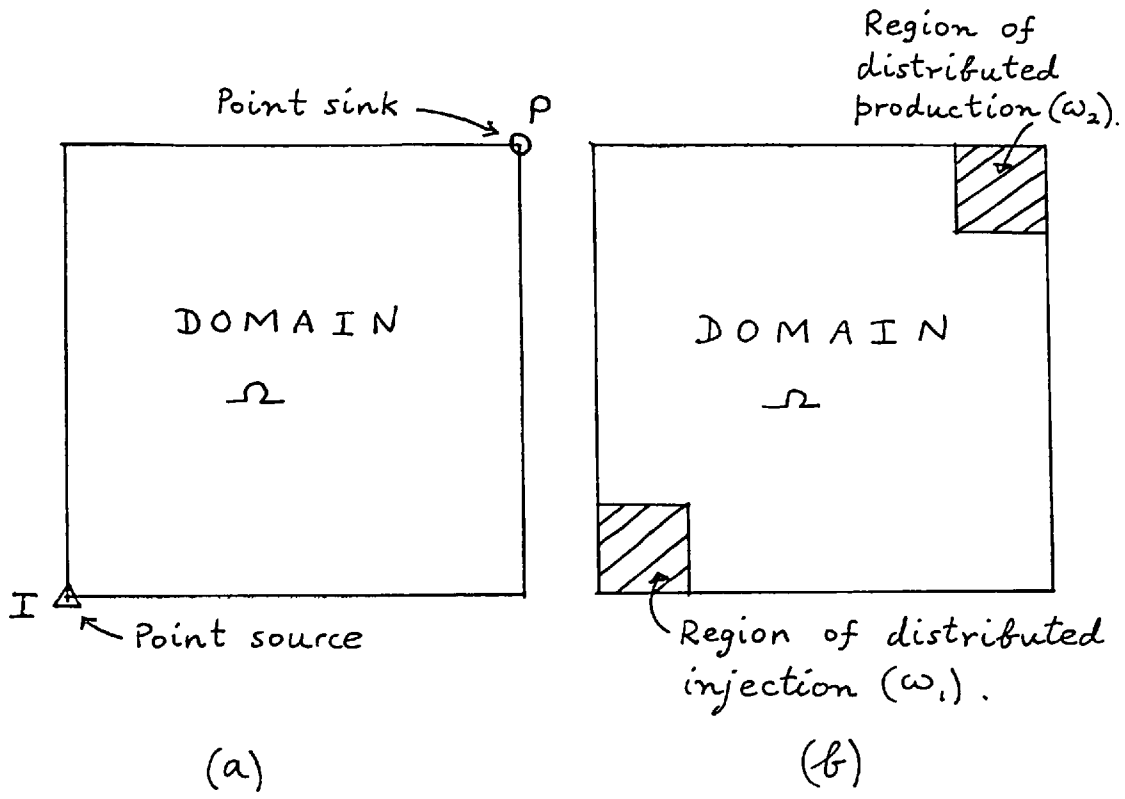


Figure 2.3: Illustration of (a) point source/sink and (b) distributed source/sink configurations, with respect to the five-spot example.

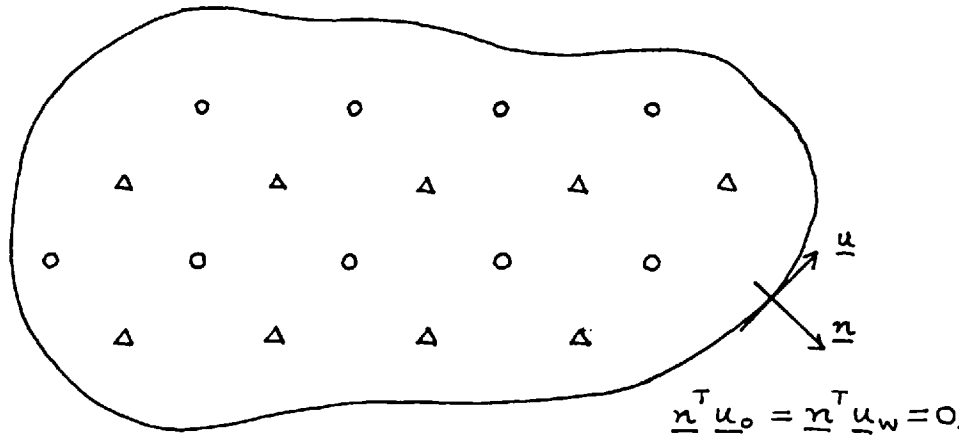


Figure 2.4: Illustration of no flow boundary condition. Boundary velocities are in the tangential direction only. Normal component = 0.

2.6 Boundary conditions

Having defined the fluid sources within the system, we need also to specify the conditions prevailing on the boundary of the domain. The most commonly used boundary condition in reservoir simulation is that of a "closed" system, which requires that, for each phase, the flux velocity normal to the boundary ($\partial\Omega$) be zero. This condition applies everywhere on the boundary, except at points where fluid is either entering or leaving the domain. In vector notation, this means

$$\underline{n}^T \underline{u}_o = \underline{n}^T \underline{u}_w = 0 \quad \text{on} \quad \partial\Omega \tag{2.36}$$

where $\underline{n}$ is the unit outward normal (Figure 2.4).

Physically, this "no flow" boundary condition would be the result of either:

(1) a permeability cut-off, giving $\underline{K} = \underline{0}$ on the boundary.

or

(2) the boundary coinciding with a limiting streamline, as is the case when considering the symmetry element of a pattern flood (such as five-spot).

In the latter situation, we find that, if the reservoir is horizontal, then the boundary condition (2.36), together with the transport equations (2.11) and (2.12), implies:

$$\underline{n}^T \underline{\nabla} p_o = 0 \quad \text{on} \quad \partial\Omega, \tag{2.37}$$

and that if, in addition, capillary pressure exists, then:

$$\underline{n}^T \underline{\nabla} S_w = 0 \qquad \text{on} \qquad \partial\Omega . \tag{2.38}$$

The incorporation of these constraints into the 2D finite element models will be dealt with in Chapter 4.

2.7 Initial conditions

Since the differential equations are time-dependent, it is necessary that the main variables p_o and S_w be known throughout the domain at time t = 0. For horizontal systems, the initialisation procedure is usually straightforward. Thus, for example, if these variables are initially constant over the whole reservoir, we simply have:

$$\begin{cases} p_o = p_i & \text{(initial pressure), and} \qquad (2.39) \\ S_w = S_{wc} & \text{(connate water saturation)} \qquad (2.40) \end{cases}$$

These, in fact, are the initial conditions used for the numerical models, studied in this thesis. Should a vertical dimension exist, however, the pressures and saturations would normally be initialised so as to be consistent with hydrostatic and capillary equilibrium respectively.

Finally, combining all the foregoing sections, we can briefly re-state the problem as follows:

Given:

(1) Reservoir dimensions and properties ;

$$\Delta x, \Delta y, \phi, \underline{K} \text{ etc.}$$

(2) Saturation and pressure dependent data;

k_{ro}, k_{rw}, p_c vs S_w; B_{oi}, B_{wi}, c_o, c_w etc.

(3) The boundary - and initial conditions;

(4) The locations and rates of the injection and production wells; and

(5) The basic equations of transport, state and continuity,

it is required to determine the variables

$p_o(x,y,t)$ and $S_w(x,y,t)$ for all t.

CHAPTER 3

A Selective Review of Analytical Methods for Waterflood Calculations

In this chapter we shall consider some of the models that are available for solving problems associated with displacement. These models are based on analytical or semianalytical techniques (i.e. methods other than "simulation", in the strict sense of the word), and they are presented here to review the development of two-phase flow theory, and with a view to using some to assess the accuracy of the simulation models.

3.1 1 Dimensional moving Boundary model

One of the simplest models for describing two-phase flow in 1D is given by Muskat [43a]. This model is based on a piston-type displacement, and resembles in some ways, the movement of a meniscus through a capillary tube. The derivation given below is similar to that of Dawe [18]. The Muskat model is based on the following assumptions:

(a) Horizontal, immiscible displacement of oil by water in a linear system of length L and cross-sectional area A.

(b) The pressures at the inlet and outlet are constant.

(c) The displacement is characterised by two regions of uniform (but different) saturations, separated by a moving interface. Ahead of the front, the system is at connate water saturation, and conducts oil only (at superficial velocity u_o), while, behind the front, it is at residual oil saturation, so that

water alone flows (at superficial velocity u_w).

(d) The fluids are incompressible, and hence we have

$$u_o = u_w = u = \text{constant}$$

at any point in time.

Let us suppose that, at time t, the "front" is at a pressure P, and that it has advanced a distance x from the inlet end (Figure 3.1). Since we have a constant-velocity single phase flow in each region, it follows, from Darcy's law, that the pressure varies linearly with distance on either side of the front. Thus:

$$u = -\lambda_r k \frac{\partial p}{\partial x} = \lambda_r k \frac{\Delta p}{\Delta x} \tag{3.1}$$

The pressure drops occurring in the water - and oil zones are therefore:

$$\Delta p_w = p_1 - p = \frac{u x}{\lambda_{rw} k} \tag{3.2}$$

and

$$\Delta p_o = p - p_2 = \frac{u(L-x)}{\lambda_{ro} k} . \tag{3.3}$$

Addition of these two equations yields the superficial velocity as a function of the front position, x.

$$u = k \Delta p \Big/ \left[\frac{x}{\lambda_{rw}} + \frac{L-x}{\lambda_{ro}} \right] . \tag{3.4}$$

This velocity can be related to the rate of advance of the front (dx/dt), by performing a material balance calculation. If the front moves a distance dx in time interval dt, then the amount of water received by this newly invaded element is

$$\phi A \left[(1 - S_{or}) - S_{wc} \right] dx ,$$

and this quantity must be supplied by the water flowing in from behind the front, which is

$$u A \, dt .$$

Equating these expressions, we find

$$\frac{dx}{dt} = \frac{u}{\phi (1 - S_{wc} - S_{or})} . \qquad (3.5)$$

Finally, combining equations (3.4) and (3.5) to eliminate u, and integrating the resulting equation from $(x = 0, t = 0)$ to $(x = x, t = t)$, we obtain :

$$\frac{L}{\lambda_{ro}} x + \frac{1}{2} \left[\frac{1}{\lambda_{rw}} - \frac{1}{\lambda_{ro}} \right] x^2 = \frac{k \Delta p}{\phi (1 - S_{wc} - S_{or})} t \qquad (3.6)$$

which is the relationship describing the progress of the flood front with time. Furthermore, if we define the mobility ratio (M) of the displacement as:

$$M = \frac{\text{mobility of displacing fluid}}{\text{mobility of displaced fluid}} = \frac{\lambda_{rw}}{\lambda_{ro}} , \qquad (3.7)$$

and introduce the dimensionless variables

$$X = x/L \qquad (3.8)$$

and $$T = \frac{(\lambda_{ro} k \Delta p / L^2)}{\phi (1 - S_{wc} - S_{or})} \cdot t , \qquad (3.9)$$

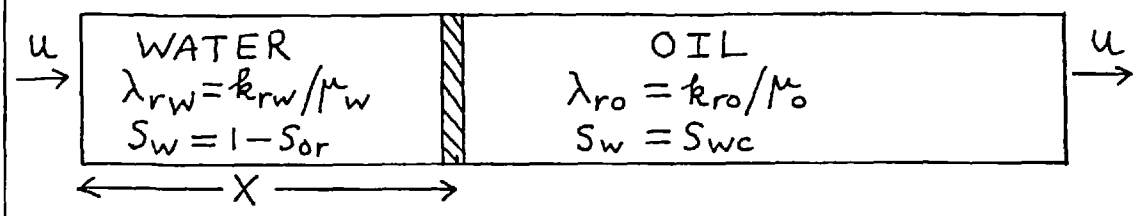


Figure 3.1 The linear, moving-boundary displacement model

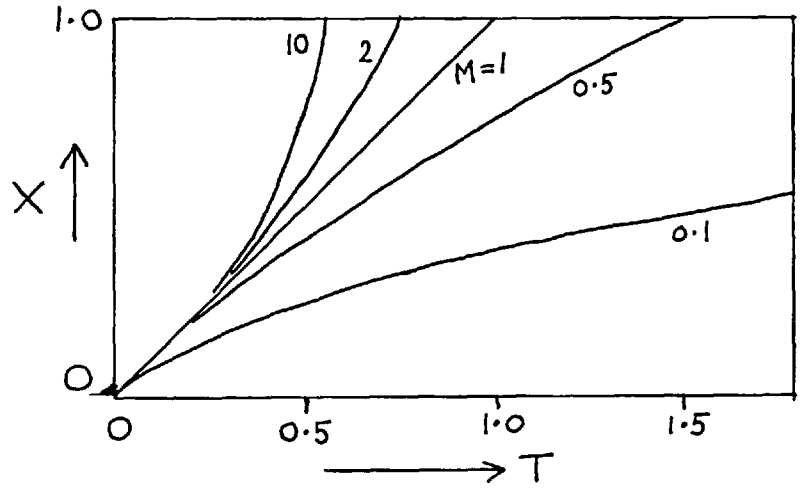


Figure 3.2 Plot of fractional length VS dimensionless time for various mobility ratios. (Moving boundary model)

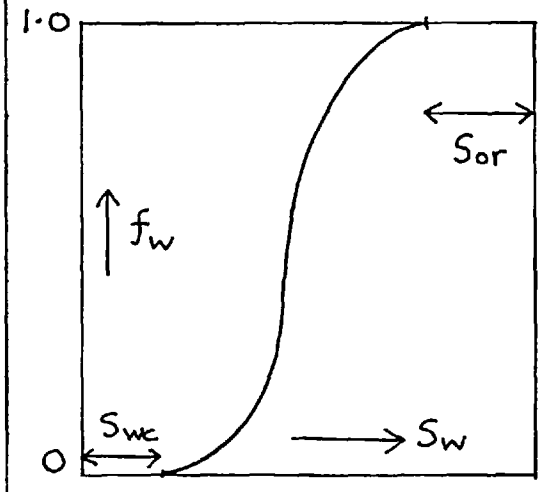


Figure 3.3 Sketch of f_w Vs Sw

$f_w = (\lambda rw/\lambda rt)$

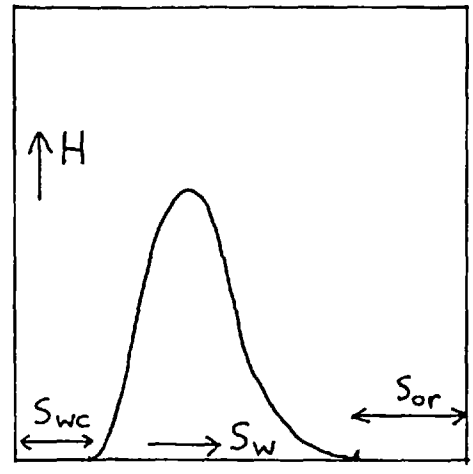


Figure 3.4 Sketch of H vs S_w

$$H = \frac{\lambda ro \ \ \lambda rw}{\lambda rt} \ Pc'$$

we can express equation (3.6) in the normalised form:

$$X + \tfrac{1}{2}\left[\frac{1}{M} - 1\right]X^2 = T \,. \qquad (3.10)$$

Figure 3.2 illustrates these [X vs T] curves for various mobility ratios, from which it can be seen that the flood-front accelerates, or moves with constant velocity, or decelerates, according as $M \gtrless 1$. The dimensionless breakthrough time is obtained by putting X = 1, thus:

$$T_{BT} = \tfrac{1}{2}\left[1 + \frac{1}{M}\right] . \qquad (3.11)$$

From a physical viewpoint, this model is realistic for a miscible flood under a constant pressure differential. Apart from this, however, it is of limited applicability due to the assumptions on which it is based.

3.2 Development of the Classical Theory

The formulation of this theory was a direct consequence of the introduction of the relative permeability concept. The significant feature of this approach was that it allowed the two fluids to flow simultaneously at any point, rather than one behind the other, as in the Muskat model.

3.2.1 Fractional flow equation (Leverett)

The starting point for this theory was the pioneering work of Leverett [37] who derived an expression for the superficial velocity of a fluid as a fraction of the total flow stream velocity. The fractional flow equation for water, for example, may be derived as

follows:-

Addition of the transport equations (2.11) and (2.12) yields.

$$\underline{u}_t = -\lambda_{rt}\underline{k}\,\underline{\nabla}p_o + \lambda_{rw}p_c'\,\underline{k}\,\underline{\nabla}S_w + g(\lambda_{ro}\rho_o + \lambda_{rw}\rho_w)\underline{k}\,\underline{\nabla}z \quad (3.12)$$

where

$$\underline{u}_t = \underline{u}_o + \underline{u}_w \quad \text{and} \quad \lambda_{rt} = \lambda_{ro} + \lambda_{rw}\,.$$

Combining the above equation with (2.12), and eliminating $\underline{\nabla}p_o$, we obtain

$$\underline{u}_w = \left(\frac{\lambda_{rw}}{\lambda_{rt}}\right)\underline{u}_t + \frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}}p_c'\,\underline{k}\,\underline{\nabla}S_w + \frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}}g(\rho_w - \rho_o)\underline{k}\,\underline{\nabla}z. \quad (3.13)$$

There are three important factors in this equation.

(1) Fractional mobility

The fractional mobilities of the two phases are denoted by f_w and f_o respectively, where

$$f_w = \frac{\lambda_{rw}}{\lambda_{rt}} \quad \text{and} \quad f_o = \frac{\lambda_{ro}}{\lambda_{rt}} = 1 - f_w\,. \quad (3.14)$$

substituting for λro and λrw, we obtain the familiar expression for f_w in terms of the relative permeabilities and viscosities:

$$f_w(S_w) = \frac{(k_{rw}/\mu_w)}{(k_{ro}/\mu_o) + (k_{rw}/\mu_w)} \quad (3.15)$$

For moderate viscosity ratios and typical relative permeability curves (such as shown in Figure 2.1a), a plot of f_w vs S_w commonly takes on

the characteristic "S" shaped profile (i.e. possessing an upward concavity in its lower regions), beginning at the point

$$f_w(S_{wc}) = 0$$

and terminating at

$$f_w(1-S_{or}) = 1$$

as shown in Figure 3.3. Although an "S shaped" curve need not always be present, it is encountered in many practical situations, and it cannot be overemphasised that the shape of the f_w curve plays a very important part in determining the nature of the solution - be it analytical or numerical. We shall return to this point later.

(2) Capillary pressure term

This is the second term of equation (3.13)

$$H(S_w) = \frac{\lambda_{ro}\,\lambda_{rw}}{\lambda_{rt}}\,p_c' \quad , \tag{3.16}$$

and its role is clearly to enhance the flow of water relative to the oil, thereby reflecting the preferential attraction of the porous medium to the wetting phase (imbibition) Figure 3.4 shows a sketch of this curve for a water-wet system. It shows that the effect of the capillary pressure term is very pronounced in the lower part of the saturation range.

(3) Gravity term

The contribution of the hydrostatic gradient to the fluid flow

is governed by the term:

$$G(S_w) = \frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}} g(\rho_w - \rho_o) , \qquad (3.17)$$

and thus, the larger the difference in densities, the greater will be the effect of gravity in segregating the two fluids.

The capillary and gravity terms are very similar:

(a) they are both proportional to the harmonic mean of the fluid mobilities $\left[H.M = 2\lambda_{ro}\lambda_{rw}/(\lambda_{ro}+\lambda_{rw})\right]$;

(b) the capillary gradient $(P_c' \underline{\nabla} S_w)$ is analogous to the hydrostatic gradient $(g\Delta\rho\underline{\nabla}z)$. One important difference, however, is the fact that $\underline{\nabla}z$ is a known variable which is independent of time, while $\underline{\nabla}S_w$ is an unknown. Thus capillary effects are more difficult to handle analytically than gravity effects. Rewriting equation (3.13) in 1 dimensional form and defining f_w^* to be the true water-cut (i.e fraction of water in the flow stream) we have

$$f_w^* = u_w / u_t ,$$

$$\text{or} \quad f_w^* = \frac{\lambda_{rw}}{\lambda_{rt}}\left[1 + \frac{\lambda_{ro}k}{u_t}\left\{P_c'\frac{\partial S_w}{\partial x} + g\Delta\rho\frac{\partial z}{\partial x}\right\}\right] , \qquad (3.18)$$

which is the complete fractional flow equation. In many situations, the capillary and gravity effects are considered negligible, and the only significant term is the fractional mobility. Thus, for a horizontal, non-capillary displacement, equation (3.18) reduces to

$$f_w^* \doteq (\lambda_{rw}/\lambda_{rt}) = f_w . \qquad (3.19)$$

3.2.2 Frontal advance equation (Buckley and Leverett)

Buckley and Leverett [7] considered the 1 dimensional incompressible displacement problem, under a constant rate of injection (as opposed to the constant pressure difference used by Muskat). This implies that the total superficial velocity u_t is invariant in space and time (Figure 3.5).

The boundary conditions are:

$$\left.\begin{aligned} u_o &= 0 \\ u_w &= u_t \end{aligned}\right\} \text{ at the inlet } (x=0) \qquad (3.20)$$

and

$$\left.\begin{aligned} u_o &= (1-f_{we})\,u_t \\ u_w &= f_{we}\,u_t \end{aligned}\right\} \text{ at the outlet } (x_e = L). \qquad (3.21)$$

The initial condition is simply

$$x(S_w, 0) = 0 \qquad (S_w > S_{wc}). \qquad (3.22)$$

Beginning with equation (2.32), and assuming zero compressibility, the continuity equation for water (in 1D) is:

$$\frac{\partial u_w}{\partial x} + \phi \frac{\partial S_w}{\partial t} = 0 \qquad (3.23)$$

which, upon substituting the simplified fractional flow equation (3.19) with u_t = constant, becomes:

$$u_t \frac{\partial f_w}{\partial x} + \phi \frac{\partial S_w}{\partial t} = 0. \qquad (3.24)$$

Since f_w is a function of S_w only, we have

$$\left[u_t \frac{df_w}{dS_w}\right] \frac{\partial S_w}{\partial x} + \phi \frac{\partial S_w}{\partial t} = 0 , \qquad (3.25)$$

which is a non-linear hyperbolic partial differential equation in (x,t). It is at this stage that the basic "analytical" and "simulation" approaches part company. The reason for this is that, by introducing the identity

$$\left(\frac{\partial x}{\partial t}\right)_{S_w} = -\left(\frac{\partial S_w}{\partial t}\right)_x \Big/ \left(\frac{\partial S_w}{\partial x}\right)_t , \qquad (3.26)$$

we can transform the above differential equation from the Eulerian system $S_w(x,t)$ to the more convenient Lagrangian system $x(Sw,t)$, from which the analytical solution follows directly. The transformed version of equation (3.25) is

$$\frac{\partial x(S_w,t)}{\partial t} = \frac{u_t}{\phi} \frac{df_w}{dS_w} . \qquad (3.27)$$

As the right hand side of the above equation is a function of S_w only, each saturation advances with a constant velocity which is proportional to the slope of the (f_w vs S_w) curve at that particular saturation. Integrating this velocity w.r.t. time from t=o to t=t, we obtain.

$$x(S_w,t) = \left[\frac{u_t}{\phi} \frac{df_w}{dS_w}\right] t + x(S_w,0) . \qquad (3.28)$$

We now introduce the dimensionless variables

$$X(S_w,t) = x(S_w,t)/L \left[\begin{array}{l}\text{Fractional distance moved by the}\\ \text{saturation Sw in time t}\end{array}\right] \text{ and} \qquad (3.29)$$

$$Q(t) = \frac{u_t t}{L\phi} \left[\begin{array}{l}\text{Cumulative pore volumes of}\\ \text{water injected in time t}\end{array}\right] , \qquad (3.30)$$

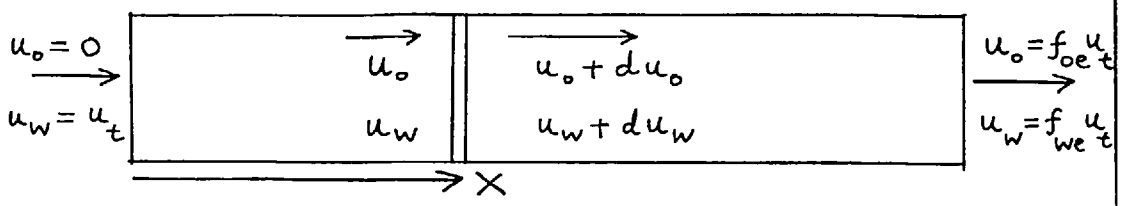
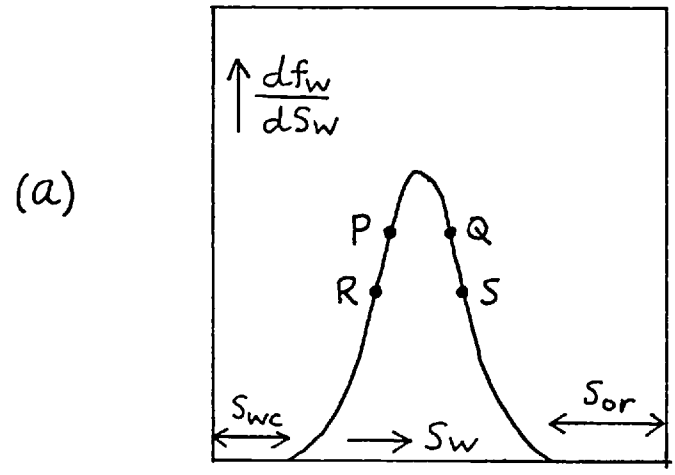


Figure 3.5 The linear, simultaneous-flow model of Buckley and Leverett.



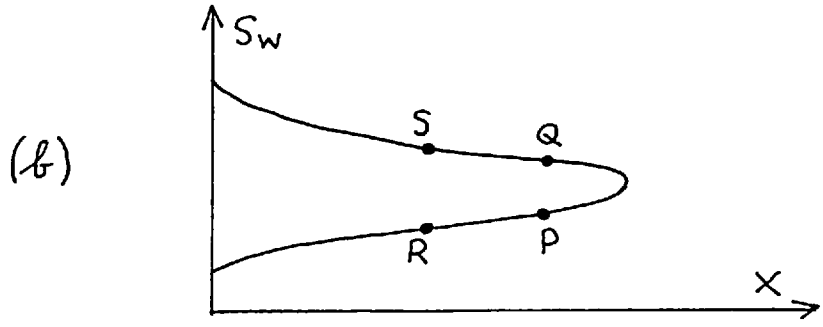


Figure 3.6 (a) [dfw/ds$_w$ vs S$_w$] and (b) [X vs S$_w$] profiles, derived from an originally "S-shaped" fw curve.

and substitute the initial condition (3.22), arriving at the equation

$$X(S_w,t) = Q(t)\,\frac{df_w(S_w)}{dS_w}\ , \qquad (3.31)$$

which is the Buckley-Leverett frontal advance equation. Figure 3.6a shows a schematic plot of (dfw/dSw) vs S_w for an S shaped f_w curve as shown in Figure 3.3, while Figure 3.6b shows the corresponding graph of X vs Sw as implied by equation (3.31).

Ironically for this theory, Figure 3.6b, which was expected to provide the solution to the displacement problem, turned out to be the very centre of a major controversy. Due to the S-shaped nature of the f_w curve, it follows that pairs of saturation points (such as (P,Q), (R,S) etc., in Figure 3.6a) have the same value of (dfw/dSw), and hence also, by equation (3.27), the same velocity of propagation. Consequently, the saturation profile shown in Figure 3.6b exhibits two distinct saturations at each point, which is a physical absurdity.

In their paper, Buckley and Leverett [7] recognised this erroneous situation, and argued that the double valued saturation profile ought to be replaced by one possessing a saturation discontinuity (or "front") from a frontal saturation S_{wf} down to the connate water saturation S_{wc}, and that the magnitude of S_{wf} should be determined by material balance considerations. Since the area enclosed by the saturation profile against the S_w axis is equal to the cummulative pore-volumes of water injected, it follows that the above construction is equivalent to truncating the curve by a vertical line BDF (Figure 3.7b) such that

$$\left.\begin{array}{l}\text{Area ABCDE} = \text{Area ABDFE}\\ \text{or, simply,}\quad \text{Area BCD} = \text{Area DEF}\end{array}\right\}\ . \qquad (3.32)$$

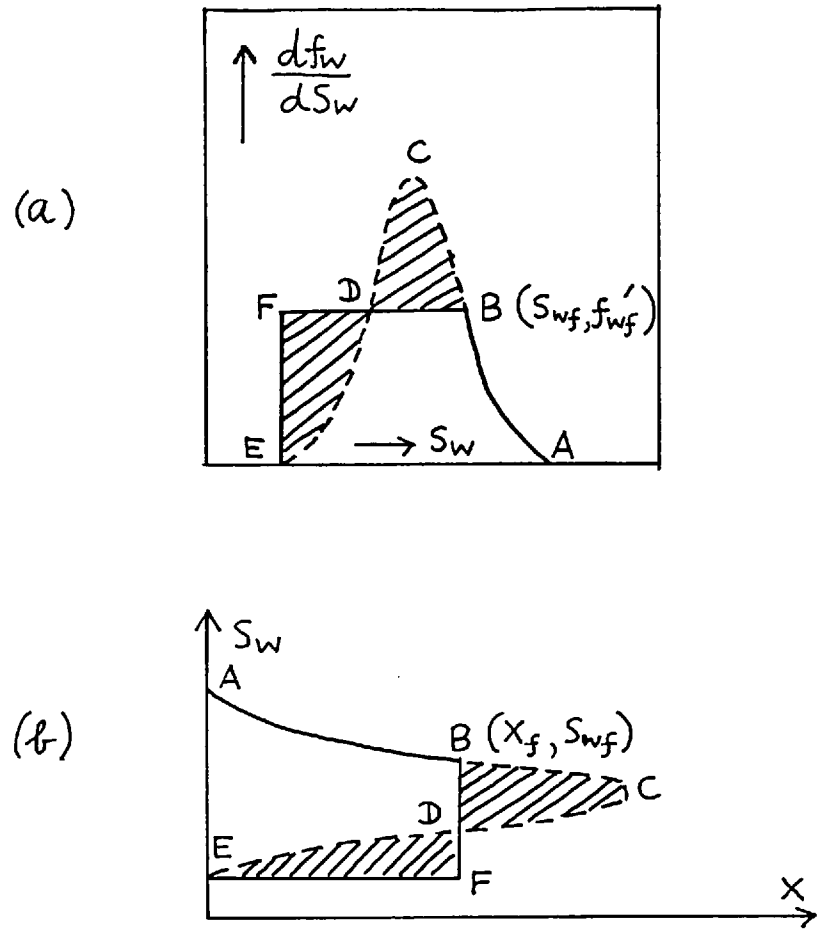


Figure 3.7 Modified curves of (a) (dfw/dsw) and (b) Sw profile, to produce a "front" satisfying the material balance criterion.

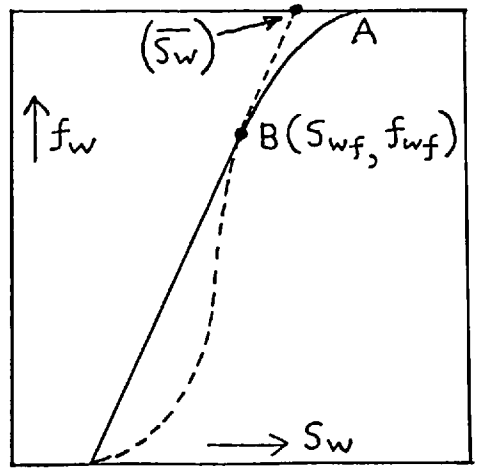


Figure 3.8 Welge's "tangent" construction to determine the "front saturation" S_{wf}

The piston-like profile ABDF is therefore the correct solution to the displacement problem. The magnitude of the saturation discontinuity can be determined by an elegant construction due to Welge [56], which can be derived as follows:

3.2.3 Determination of front saturation (Welge)

Let S_{wf} be the frontal saturation (point B of Figure 3.7a,b), and let X_f be the distance moved by the front in time t, during which Q pore-volumes of water have been injected. The water cut and its derivative at $S_w = S_{wf}$ will be denoted by f_{wf} and f_{wf}' respectively.

The material balance condition (3.32) requires that

$$\int_{S_{wc}}^{S_{wf}} \left[X(S_w) - X_f \right] dS_w = 0 \; . \tag{3.33}$$

But, from the frontal advance equation (3.31), we know that

$$\left.\begin{aligned} X(S_w) &= Q\,(df_w/dS_w) \\ \text{and} \quad X_f &= Q\,f_{wf}' \end{aligned}\right\} \; . \tag{3.34}$$

Hence, substituting these into equation (3.33), and bearing in mind that f_{wf}' is a constant with respect to the integration, we obtain (see Figure 3.7a):

$$Q \int_{S_{wc}}^{S_{wf}} \left[\frac{df_w(S_w)}{dS_w} - f_{wf}' \right] dS_w = 0 \tag{3.35}$$

or,

$$\left[f_w \right]_{S_{wc}}^{S_{wf}} - f_{wf}' \left[S_w \right]_{S_{wc}}^{S_{wf}} = 0 \; ; \tag{3.36}$$

which simplifies to the equation

$$f_{wf}' = \frac{f_{wf}}{S_{wf} - S_{wc}} \; . \tag{3.37}$$

We see therefore that the "front" which satisfies the material balance criterion is determined by the saturation at which the derivative of the f_w curve has the same slope as the chord which joins it to the connate water saturation (Figure 3.8) . Obviously this technique provides a more convenient way of locating Swf than the trial-and-error process of equating the areas. The Buckley-Leverett frontal advance equation (3.31), together with Welge's construction (3.37), is the standard analytical solution used in the petroleum industry for 1D immiscible displacements, namely:

$$X(S_w, t) = \begin{cases} Q f'_{wf} , & S_{wc} \leq S_w \leq S_{wf} \\ Q \dfrac{df_w}{dS_w} , & S_{wf} \leq S_w \leq (1 - S_{or}). \end{cases} \quad (3.38)$$

3.3. Method of characteristics

This is a technique employed in the solution of hyperbolic partial differential equations, and hence is also applicable to the 1D displacement problem described by equation (3.25). In 1959, there appeared a series of papers attempting to analyse and re-interpret the Buckley-Leverett theory in the light of the method of characteristics. The most notable was that of Sheldon et al [52].

Basically, a characteristic is a path (in the plane of the two independent variables) along which the dependent variable takes on a constant value. Not surprisingly, therefore, the characteristic of equation (3.25) turns out to be equation (3.27), which shows that the saturations are propagated along straight-line characteristics in in the (t,x) plane, starting from the x-axis (t=0) which represents the

initial saturations. Sheldon et al, demonstrated that the final solution depends on two factors:

a) the initial saturation distribution, and

b) the shape of the f_w curve.

Let us suppose, for example, that we have a smooth uniformly decreasing "initial saturation distribution" along the x - axis (Figure 3.9a). Then, if the f_w curve has a monotonically decreasing slope (Figure 3.9b), the characteristics will fan outwards (i.e. diverge) into the (t, x) plane, and hence produce smooth solutions (Figure 3.9c). If, however, the f_w curve has an S shape (Figure 3.9d), then the characteristics will intersect in due course, and result in multiple-valued saturations, (Figure 3.9e). By analogy with supersonic flow, the coalescence of characteristics is explained by invoking the concept of a "shock" or discontinuity. The sequence of changes that occur to the saturation profile can be described in three stages.

(1) "Pre-shock" period

Starting with the initial distribution (Figure 3.9a) we can see that, due to the S shaped f_w curve, some of the higher saturations will travel faster than, and eventually catch up with, the lower ones. This tendency is evident from the convergent nature of the characteristics (figure 3.9e), which implies a gradual steepening of the saturation profile (Figure 3.10a).

(2) Shock formation

The process of steepening continues until the first pair of

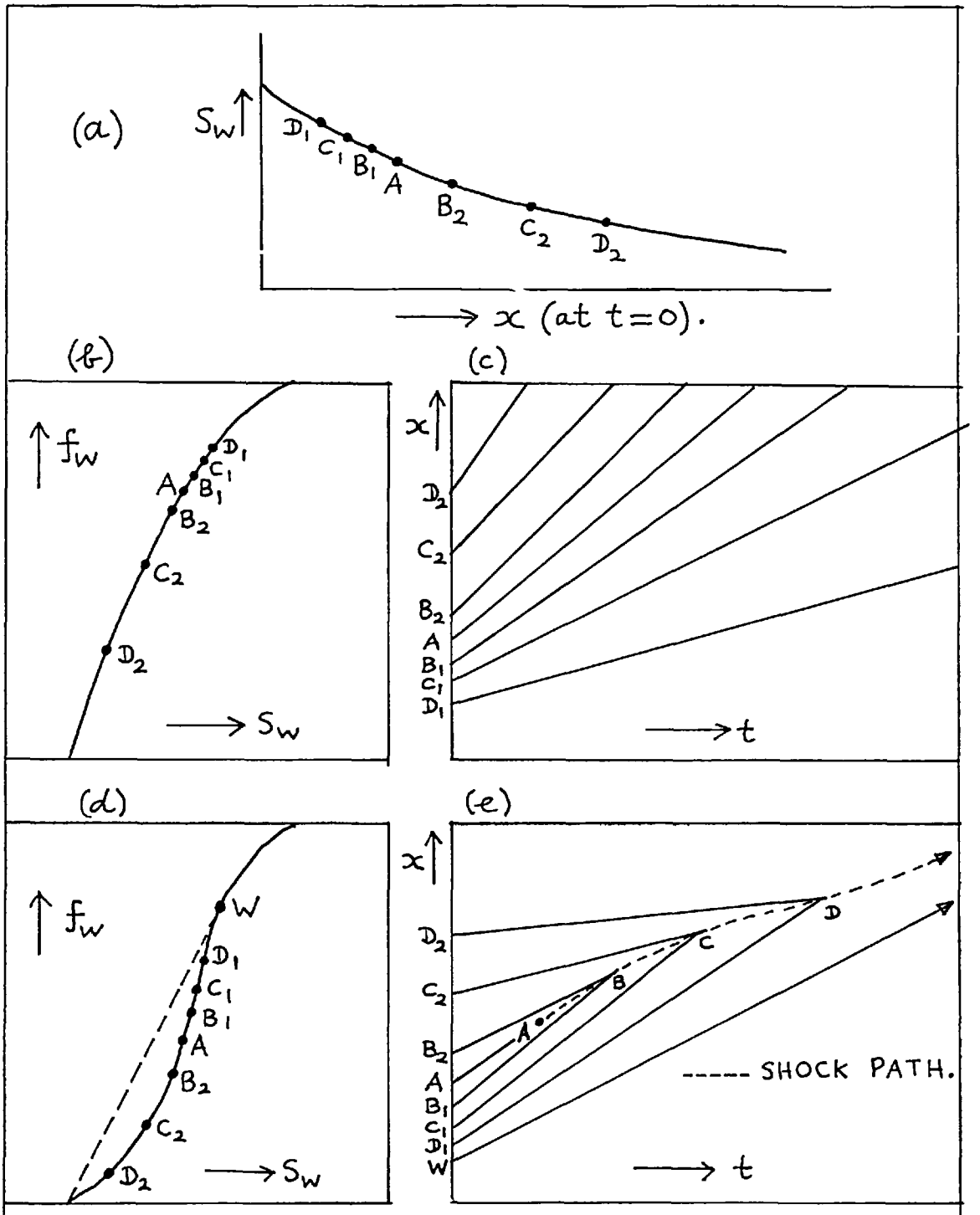


Figure 3.9 Schematic illustration of the method of characteristics. [After Sheldon et al [52]].

(a) Initial saturation distribution;

(b) Smooth fw curve and (c) corresponding characteristics diagram;

(d) "S-shaped" fw curve and (e) corresponding characteristics diagram.

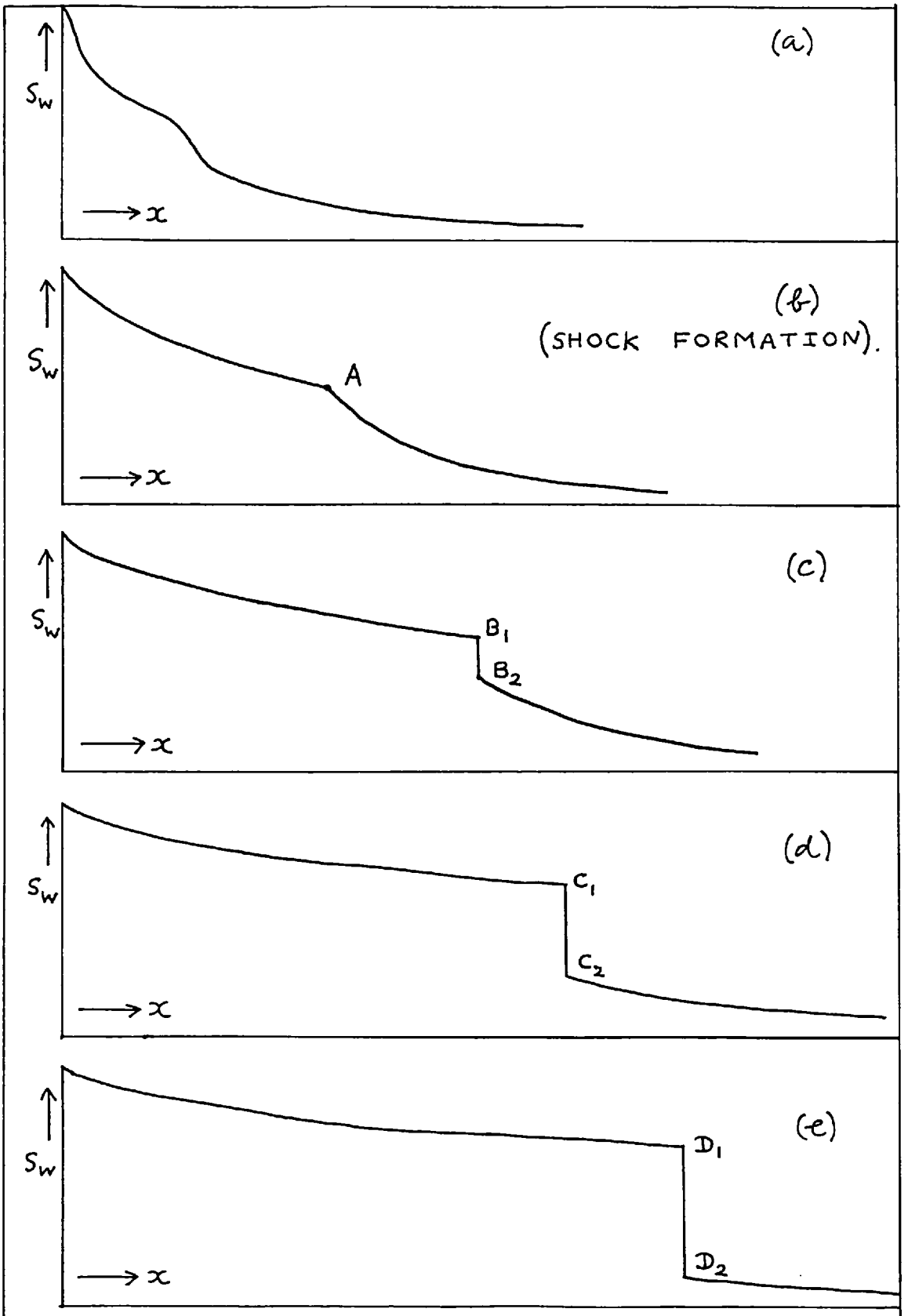


Figure 3.10 Schematic saturation profiles at successive stages during a displacement, to illustrate the initiation growth and propagation of a "shock front", as deduced from the method of characteristics. [cf. Figures 3.9 d, e].

infinitesimally adjacent characteristics meet, thereby initiating a discontinuity in the saturation profile (point A of Figures 3.9e, 3.9d and3.10b). Mathematically, this corresponds to the situation when the saturation gradient $(\partial S_w/\partial x)$ first becomes infinity (or, equivalently, $(\partial x/\partial S_w) = 0$)

(3) "Post - shock" period

This period marks the growth and propagation of the "shock", as the characteristics on either side of A continue to converge in pairs. Figures 3.10c,d,e illustrate the profile at a series of times which denote the arrival of the saturation pairs B_1 B_2, C_1 C_2 and D_1 D_2, respectively (Figure 3.9d). The points of intersection of these characteristics - A,B,C,D.. etc., - constitute what is known as the "shock path" (Figure 3.9e). This path eventually becomes asymptotic to the characteristic corresponding to the Welge tangent W.

Cardwell [8] has presented analytical expressions for determining the locus of the shock path.

The use of characteristics thus provides a general method for handling 1D immiscible displacements with an arbitrary initial saturation distribution. The Buckley-Leverett model is, in fact, a special case of this, in which (due to the initial step-discontinuity (equation 3.22)) the shock is initiated at the very beginning (t=0), and is propagated with a constant magnitude.

3.4 Treatment of Capillary Pressures

Our discussion so far has been confined to the non-capillary

displacement theory which, as we have seen, has proved amenable to analytical methods of solution. The introduction of capillary pressures, however, makes it necessary to combine analytical and numerical methods, in order to solve the differential equation.

Fayers and Sheldon [25] were among the first to consider this subject at length, and they approached it with a two-fold objective:

(a) to determine whether saturation discontinuities (i.e shock fronts) do occur in practice, or whether the capillary forces operating in reservoirs succeed in preventing the formation of these shocks; and

(b) in the case of the latter, to assess the magnitude of the "smearing effect" of the capillary pressures on the Buckley-Leverett frontal solution. The first part of their technique was to derive the Lagrangian version of the displacement equation, which is identical to equation (3.27), with the exception that it is expressed in terms of the dimensionless variables X and Q [equations 3.29 and 3.30], and that it involves the <u>true</u> water-cut f_w^* as opposed to the fractional mobility f_w. Thus:

$$\frac{\partial X}{\partial Q} = \frac{\partial f_w^*}{\partial S_w} \qquad (3.39)$$

in which, if we neglect gravity (but retain capillarity):

$$f_w^* = f_w + \frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}} \frac{k}{u_t} p_c' \frac{\partial S_w}{\partial x} \; ; \qquad (3.40)$$

$$= f_w + C \frac{\partial S_w}{\partial X} \quad , \tag{3.41}$$

where C(Sw) = $\frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}} \frac{k}{u_t L} p_c'$ (3.42)

is known as the capillary function.

Substituting equation (3.41) into (3.39), and remembering that

$$\frac{\partial S_w}{\partial X} = 1 \Big/ \left(\frac{\partial X}{\partial S_w}\right) ,$$

we obtain

$$\frac{\partial X}{\partial Q} = \frac{df_w}{dS_w} + \frac{\partial}{\partial S_w}\left[\frac{C(S_w)}{\left(\frac{\partial X}{\partial S_w}\right)}\right] . \tag{3.43}$$

A comparison with equation (3.27) reveals the additional complication resulting from the insertion of the capillary pressure. To solve this non-linear differential equation, Fayers and Sheldon [25] used a finite difference technique. The saturation range S_{wc} to (1 -Sor) was divided into n fixed intervals (at the nodes Swo to Swn), and the fractional distances (Xo to Xn) moved by these nodal saturations were taken as the unknowns to be solved for, at each time-step.

Consider, for example, the time interval $\Delta Q = Q^{n+1} - Q^n$. To evaluate the average gradient (∂X/∂Sw) over this interval, the X's were interpolated to an intermediate time - level.

$$\overline{X_i} = (1 - a) X_i^n + a X_i^{n+1} \quad , \tag{3.44}$$

where "a" is a time-centering parameter. With this approximation,

the difference equation corresponding to (3.43) at a typical node i, takes the form:

$$(S_{w_{i+\frac{1}{2}}} - S_{w_{i-\frac{1}{2}}}) \frac{X_i^{n+1} - X_i^n}{\Delta Q} = (f_{w_{i+\frac{1}{2}}} - f_{w_{i-\frac{1}{2}}})$$

$$+ \frac{C_{i+\frac{1}{2}}(S_{w_{i+1}} - S_{w_i})}{(\bar{X}_{i+1} - \bar{X}_i)} - \frac{C_{i-\frac{1}{2}}(S_{w_i} - S_{w_{i-1}})}{(\bar{X}_i - \bar{X}_{i-1})} . \quad (3.45)$$

The resulting set of equations, together with the appropriate boundary conditions, were solved by a combination of the Newton Iteration method and the tri-diagonal (Thomas) algorithm. Fayers and Sheldon tested their models on different sets of capillary pressure curves and flow rates, and the main conclusions which they drew from these numerical results were:

(1) The presence of capillary pressure prevents the occurrence of multiple-valued saturations.

(2) The effect of capillarity varies inversely with the total flow rate. At moderate to low flow rates, the capillary pressure tends to smear the "frontal region" quite appreciably, and hence "shocks" are unlikely to occur in such situations. At high flow rates, however, the capillary effect is diminished , and hence the saturation distribution approaches the Buckley-Leverett "shock" profile.

Recently, Cheshire and Curtis [11] have analyzed the capillary displacement problem in 1 dimensional radial co-ordinates through a novel technique. By introducing a dimensionless radial distance for each saturation in the form:

$$X(S_w, t) = \frac{\pi r^2 h \phi}{q_t t} , \quad (3.46)$$

they transformed the original partial differential equation into a non-linear ordinary differential equation of the type:

$$(Y - f_w)\frac{d^2Y}{dS_w^2} = \frac{4\pi h}{q_t} c \frac{dY}{dS_w} \qquad (3.47)$$

where $X = (dY/dS_w)$ [c.f. eqn. 3.46],

and this time-invarient O.D.E. was solved once and for all by a suitable numerical algorithm, to generate saturation profiles for different sets of capillary pressure/rate data. Their results are consistent with capillary theory, and this method is applicable in 2D cartesian models provided that it is restricted to regions close to the injection well, where flow is radial.

3.5 Areal Flood Patterns

In the field, waterflooding is often carried out by deploying the injection and production wells in systematic arrangements, known as "patterns". Basically, these consist of repetitions of regular polygons such that the injectors are located at the vertices (and/or sides), and the producers at the centres. These are termed "normal" patterns, to distinguish them from "inverted" networks which refer to the reverse arrangement. The following are typical flooding networks, and are illustrated in Figure 3.11.

(1) Four-spot (triangle)

(2) Five-spot (square)

(3) Seven-spot (hexagon)

(4) Nine-spot (square)

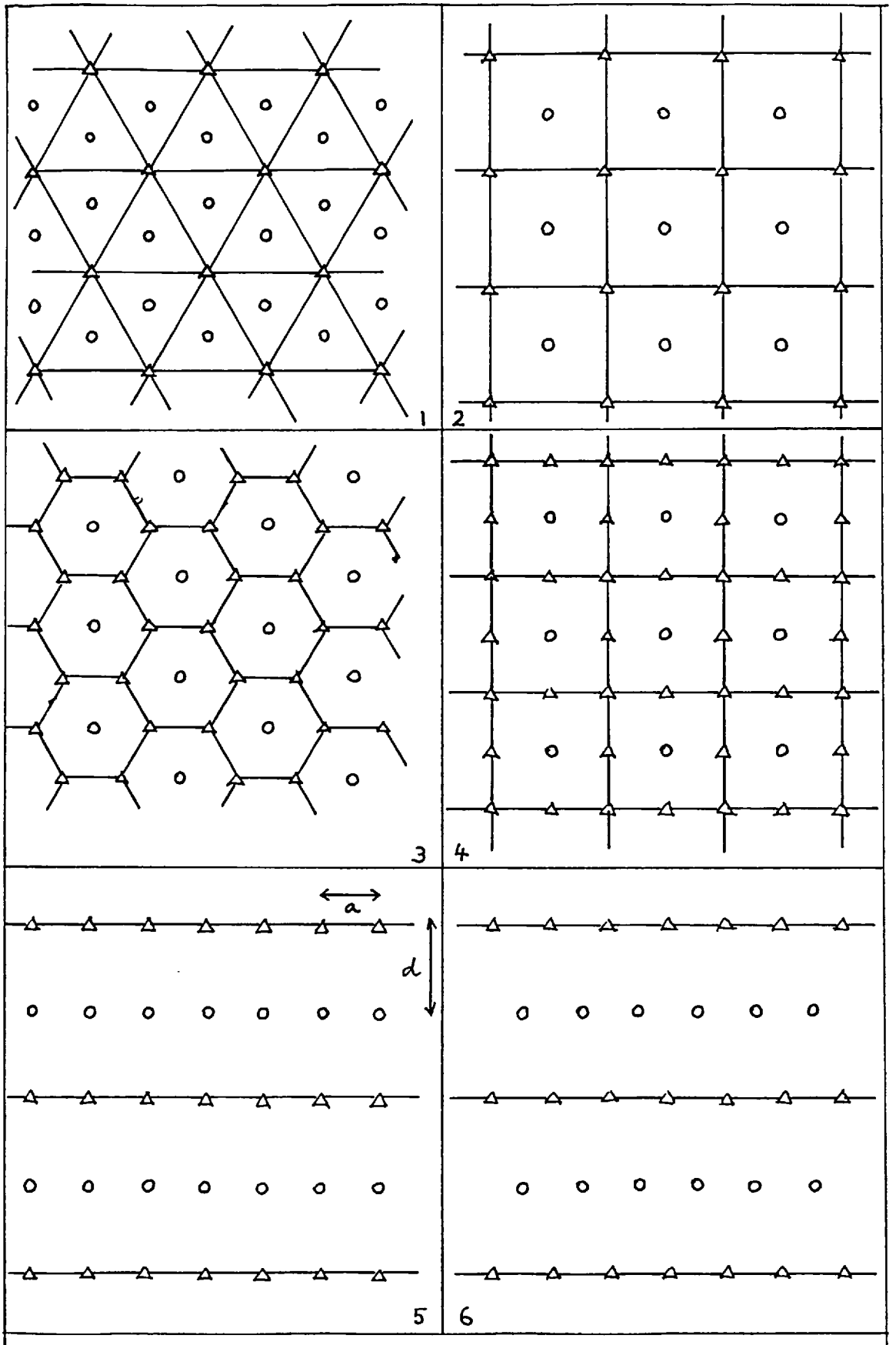


Figure 3.11 Types of flooding networks [Craig [13a]].
(1) Four spot (2) Five spot (3) Seven spot (4) Nine spot
(5) Direct line drive (6) Staggered line drive.

Of these, the five-spot is perhaps the most popular, and widely studied, pattern. Besides these, there are two further types of patterns, consisting of alternate parallel rows of injection and production wells, called "line drives". Depending on whether the injection and production rows are directly above each other, or have an offsetting lateral displacement between them, two varieties can be recognised.

(5) Direct line drive

(6) Staggered line drive (Figure 3.11)

Representing the inter-row distance by "d", and the inter-well spacing along each row by "a", we see that the five-spot is, in fact, a special case of the staggered line drive, with $d=\frac{1}{2}a$. The areal simulations that have been studied in this thesis are restricted to the repeated five-spot pattern only. The rest of this chapter, therefore, will be concerned with methods for calculating the performance of the quarter five-spot symmetry element.

3.6 Pressure distribution inside a single phase 5 -spot.

For single phase incompressible flow, the continuity equation reduces to

$$\underline{\nabla} \cdot \underline{u} = 0 \qquad (3.48)$$

which is simply the Laplace equation, with pressure as the only dependent variable:

$$\underline{\nabla}^2 p = 0 \qquad (3.49)$$

Analytical solutions to this equation can be obtained by representing the reservoir as an infinite continuum, and the injection

and production wells as point sources and sinks respectively. The pressure tends to + ∞ or - ∞ at these points, but is continuous everywhere else. The use of complex variable theory has enabled the solution to be obtained in the form of pairs of functions (known as conjugate functions), of which one represents the pressure itself, and the other the "stream function". The stream-lines denote the paths of motion of the fluid particles, beginning and terminating at point-sources and sinks respectively, or else, at "infinity". Furthermore, as would be expected, these stream-lines are orthogonal to the equi-pressure curves (isobars). We shall now proceed to derive a "series expansion" for the pressure distribution inside a single-phase five-spot, following part of the treatise of Muskat [43].

3.6.1 Single Well (in an infinite reservoir)

Since the flow is radially symmetric about the well, it is convenient to express the differential equation in radial co-ordinates:

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{dp}{dr}\right) = 0 . \qquad (3.50)$$

Also, the volumetric rate of flow across any radius r is a constant (q), and is given by Darcy's law:

$$q = 2\pi r h \frac{k}{\mu}\frac{dp}{dr} . \qquad (3.51)$$

Integrating equation (3.50) w.r.t. r, and inserting (3.51), we obtain

$$r\frac{dp}{dr} = \text{const.} = \frac{q\mu}{2\pi k h} \qquad (3.52)$$

which, on integrating once more, yields:

$$p(r) = \bar{q}\, \ln(r^2) + \text{const.} \qquad (3.53)$$

where $$\bar{q} = \frac{q\mu}{4\pi k h} \qquad (3.54)$$

is the 'rate' of the well.

Figure (3.12) illustrates the isobars and streamlines for this radial case.

3.6.2 Infinite line array of wells

Let us consider an array consisting of an infinite number of production wells each with a constant rate q, placed along the line y=d (i.e. parallel to the x=axis) such that the well spacing is "a", and one of the wells lies on the y-axis (Figure 3.13). The co-ordinates of these wells are therefore:

$$(na, d) \; ; \quad n = -\infty \text{ to } +\infty . \qquad (3.55)$$

The pressure at any point P(x,y) due to this array can be obtained by superimposing the contributions from the individual wells, as implied by equation (3.53) . Thus, we have:

$$p(x,y) = \bar{q} \sum_{-\infty}^{\infty} \ln(r_n^2) + \text{const.} \qquad (3.56)$$

$$= \bar{q} \sum_{-\infty}^{\infty} \ln\left[(y-d)^2 + (x-na)^2\right] + \text{const.} \qquad (3.57)$$

from (3.55). This infinite series can be simplified by a result, shown

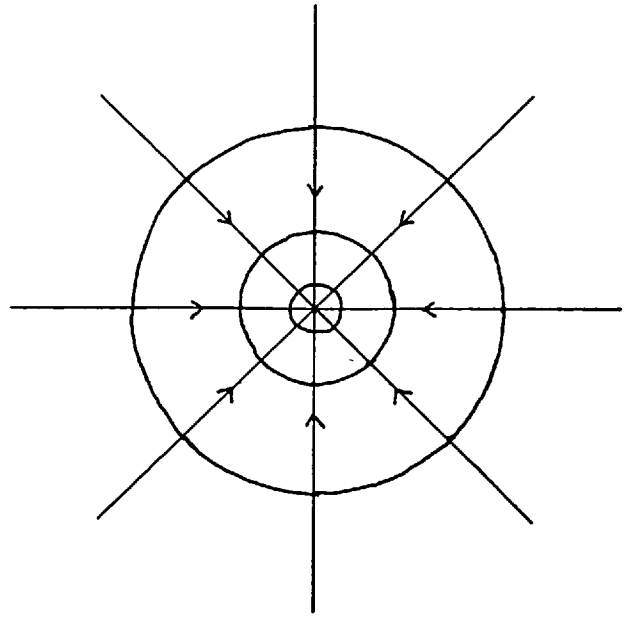


Figure 3.12

Isobars (concentric) and stream-lines (radial) for a single point sink in an infinite medium.

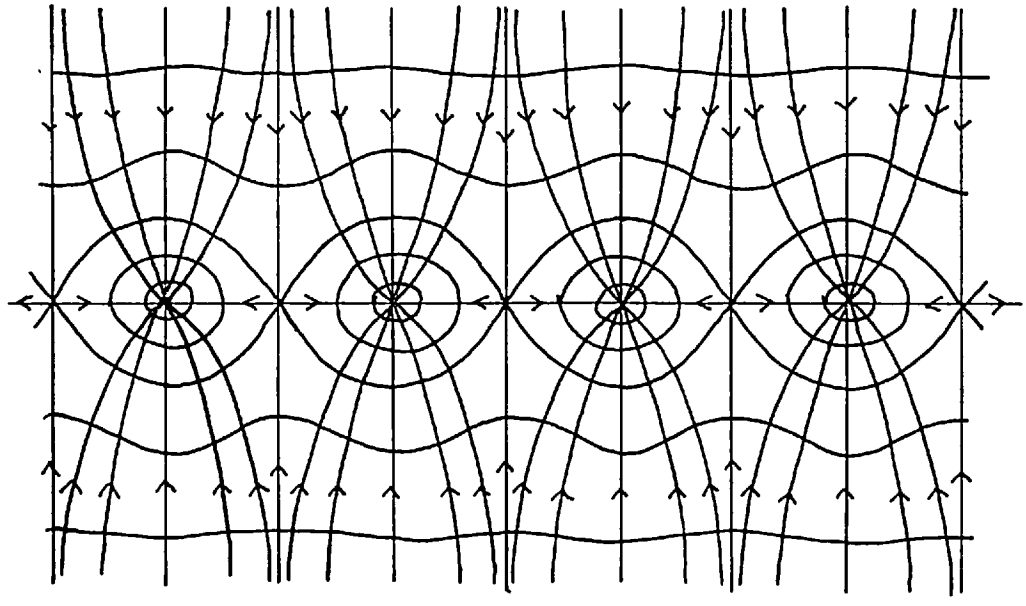


Figure 3.13

Schematic diagram showing isobars and stream-lines for an infinite well array parallel to the x-axis. (After Muskat [43d]).

in Whittaker and Watson [58](and referred to by Muskat [43b]), which we shall assume without proof. Using this result, equation (3.57) becomes:

$$p(x,y) = \bar{q} \ln\left[\cosh\frac{2\pi(y-d)}{a} - \cos\frac{2\pi x}{a}\right] + \text{const.} \quad (3.58)$$

As pointed out by Muskat [43c] this expression displays the following properties which are consistent with physical intuition, and are revealed in the isobar/stream-line diagram (Figure 3.13)

(a) it satisfies the Laplace equation ;

(b) it is symmetrical about the well-array (y=d), since cosh x = cosh (-x) ; and

(c) it has a periodicity "a" in the x-direction, as implied by the cosine term.

Equation (3.58) will be used as the basic expression in the derivation of a series solution for the five-spot.

3.6.3 The Five-spot

As shown in Figure 3.14, the five-spot pattern can be viewed as a combination of two direct line drives, each with a spacing "d" between its arrays and "2d" between the wells in each array. The only difference between the two line drives is that the second is displaced diagonally from the first (i.e. a distance "d" parallel to the x-axis, followed by a further "d" parallel to the y-axis). With the sign convention that well 'rates' are positive or negative according as they are production or injection wells, we may write down the 'rates' and

co-ordinates of the wells belonging to the two sets, as follows:

Set 1

$$\left[(-1)^m \bar{q} \; ; \; \{(na, md) : n = -\infty \text{ to } +\infty\}\right] : m = -\infty \text{ to } +\infty. \quad (3.59)$$

and Set 2

$$\left[(-1)^{m+1} \bar{q} \; ; \; \{(\overline{n+\tfrac{1}{2}}\, a, md) : n = -\infty \text{ to } +\infty\}\right] : m = -\infty \text{ to } +\infty. \quad (3.60)$$

where a=2d, and the indices m and n refer to the rows and wells respectively. Substituting these quantities into equation (3.58), we find that the contributions to the pressure at an arbitrary point P(x,y) due to a typical row "m" chosen from each of these sets, would be, respectively :

$$p_{1m}(x,y) = (-1)^m \bar{q} \left[\ln \left\{ \cosh \frac{2\pi(y-md)}{a} - \cos \frac{2\pi x}{a} \right\} + \text{const.} \right] \quad (3.61)$$

and

$$p_{2m}(x,y) = (-1)^{m+1} \bar{q} \left[\ln \left\{ \cosh \frac{2\pi(y-md)}{a} - \cos \frac{2\pi(x-\frac{1}{2}a)}{a} \right\} + \text{const.} \right]$$

$$= -(-1)^m \bar{q} \left[\ln \left\{ \cosh \frac{2\pi(y-md)}{a} + \cos \frac{2\pi x}{a} \right\} + \text{const.} \right] \quad (3.62)$$

Before summing these terms, however, it is necessary to determine the constants which occur in them, so that the result would be finite, and physically meaningful. Now, intuitively, it is obvious that the influence of a linear well-array (e.g. y = md) at a point P(x,y) will decrease (and eventually vanish) as the array becomes further and further from P. This implies that the logarithmic term must tend to a zero limit as $|m| \to \infty$. But we also know that $\cosh x \to \infty$ as $|x| \to \infty$,

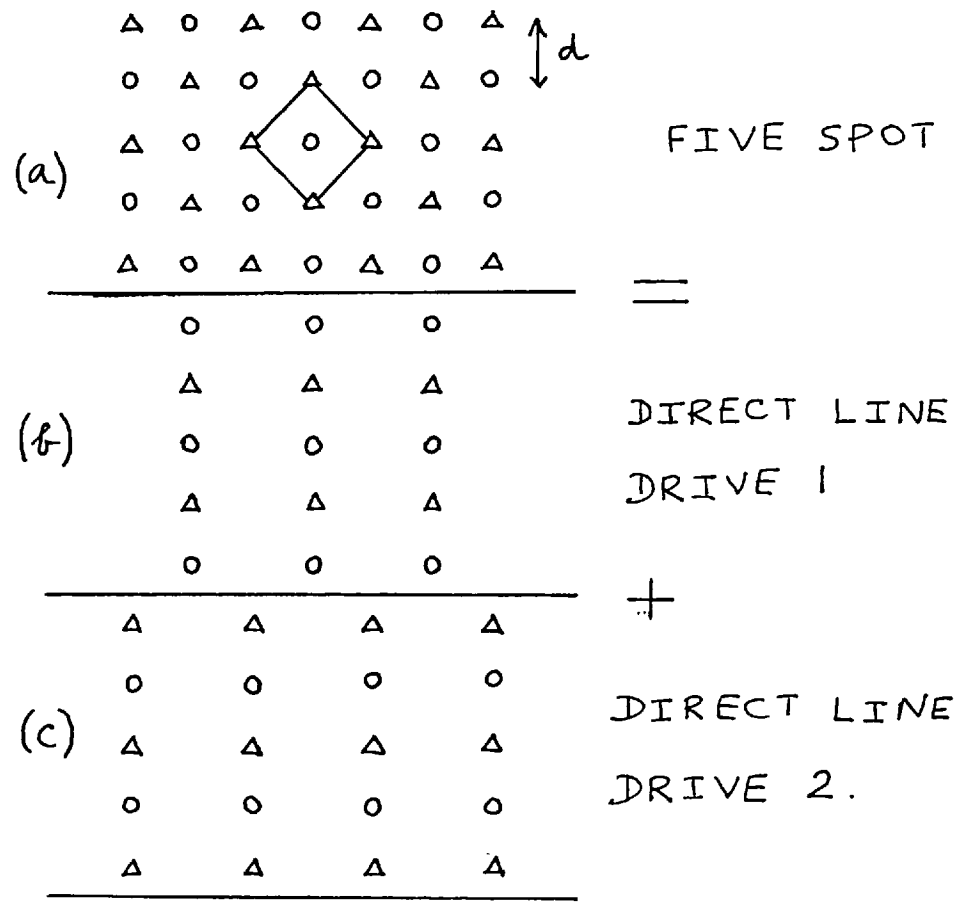


Figure 3.14

The repeated five spot pattern, expressed as a combination of two direct line drives.

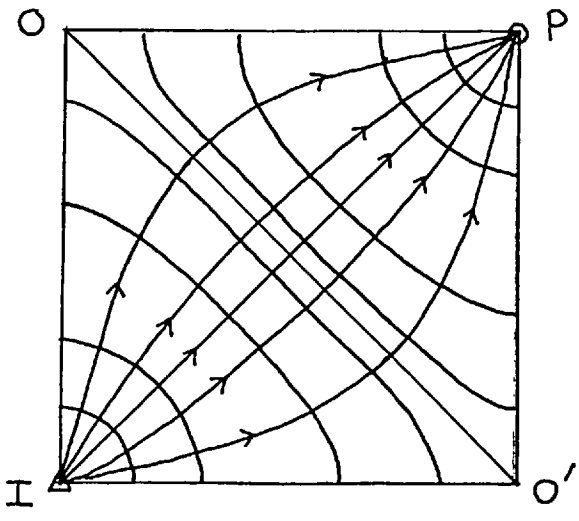


Figure 3.15

Schematic diagram of isobars and stream-lines in a quarter five spot (single phase)

[OO' = Median line]

[After Muskat [43f]]

and, consequently, it follows that the above requirement of a vanishing limit cannot be met unless a suitable "constant" is added to this log term in order to counter the effect of the cosh term. Taking this constant to be of the form log c_m , we may express the above requirement as:

$$\lim_{|m|\to\infty}\left[\ln\left\{\frac{\cosh\frac{2\pi(y-md)}{a} \pm \cos\frac{2\pi x}{a}}{c_m}\right\}\right]=0 \qquad (3.63)$$

or, more simply,

$$\lim_{|m|\to\infty}\left[\frac{\cosh\frac{2\pi(y-md)}{a} \pm \cos\frac{2\pi x}{a}}{c_m}\right]=1. \qquad (3.64)$$

But since:

$$\left.\begin{aligned}&\left|\cos\frac{2\pi x}{a}\right| \le 1 \quad \text{for all } x;\\ &\text{and}\quad \cosh\frac{2\pi(y-md)}{a} \approx \tfrac{1}{2}e^{2\pi|m|d/a} \quad \text{for large } |m|;\end{aligned}\right\} \qquad (3.65)$$

for equation (3.64) to be satisfied, we need

$$c_m = \tfrac{1}{2}e^{2\pi|m|d/a}. \qquad (3.66)$$

Substituting these constants into equations (3.61) and (3.62), and summing the infinite series (with a=2d) we obtain:

$$\begin{aligned}\phi(x,y) &= \sum_{-\infty}^{\infty}\left[\phi_{1m}(x,y)+\phi_{2m}(x,y)\right]\\ &= \bar{q}\sum_{-\infty}^{\infty}(-1)^m \ln\left[\frac{\cosh\frac{\pi(y-md)}{d}-\cos\frac{\pi x}{d}}{\frac{1}{2}e^{\pi|m|}}\right]\\ &\quad -\bar{q}\sum_{-\infty}^{\infty}(-1)^m \ln\left[\frac{\cosh\frac{\pi(y-md)}{d}+\cos\frac{\pi x}{d}}{\frac{1}{2}e^{\pi|m|}}\right].\end{aligned} \qquad (3.67)$$

To simplify the notation, we may remove the terms corresponding to m=o outside the summation, and group the remaining terms in pairs of positive and negative m, thus:

$$m = 0 \ , \ \pm 1 \ , \ \pm 2 \ , \ \pm 3 \ , \ \ldots \ldots \longrightarrow \pm \infty \ .$$

This re-arrangement finally yields:

$$p(x,y) = \bar{q} \ln \left[\frac{\cosh(\pi y/d) - \cos(\pi x/d)}{\cosh(\pi y/d) + \cos(\pi x/d)} \right]$$

$$+ \bar{q} \sum_{1}^{\infty} (-1)^m \ln \frac{4\left[\cosh \frac{\pi(y-md)}{d} - \cos \frac{\pi x}{d}\right]\left[\cosh \frac{\pi(y+md)}{d} - \cos \frac{\pi x}{d}\right]}{e^{2\pi m}}$$

$$- \bar{q} \sum_{1}^{\infty} (-1)^m \ln \frac{4\left[\cosh \frac{\pi(y-md)}{d} + \cos \frac{\pi x}{d}\right]\left[\cosh \frac{\pi(y+md)}{d} + \cos \frac{\pi x}{d}\right]}{e^{2\pi m}} \qquad (3.68)$$

which is the series expansion for the five-spot pressure distribution, as given by Muskat [43e]. A sketch of this is shown in Figure 3.15. Although mathematically this series converges at any point P on the (x,y) plane, yet, from a numerical view-point, only a finite number of terms can be summed, and hence, in order to obtain speedy convergence (i.e. for the least "m"), P should be within the symmetry element surrounding the origin of co-ordinates (Figure 3.14a). This, however, is no restriction because, due to the repetition of the pattern in the (x,y) plane, a knowledge of the pressure distribution in a single symmetry element suffices to determine the pressure at any other point of the domain.

Two interesting corollaries may be deduced from equation (3.68).

(1) Substituting $x = \frac{1}{2}d$, we find that the pressure along the median line ($x = \frac{1}{2}d$) between injection and production wells is zero.

(2) Taking r_w (<<d) as the well radius, Muskat [43g] derived a simple expression for the pressure difference between injection and production wells:

$$\Delta p = p_{inj} - p_{prod}$$
$$= p(0, d - r_w) - p(0, r_w) \quad \text{(Figure 3-14a)}$$

leading to $$\Delta p = \frac{q\mu}{\pi k h}\left[\ln\left(\frac{d}{r_w}\right) - 0.619\right] \qquad (3.69)$$

in Darcy units.

3.7 Breakthrough recovery of five-spot flood

If the fractional area contacted by the displacing fluid is E_A (also known as the areal sweep efficiency), and the fractional recovery of the displaced fluid from the invaded zone is E_D (the displacement efficiency), then the overall recovery efficiency of the flood at breakthrough is clearly the product of these two quantities:

$$(E_R)_{BT} = (E_A)_{BT} \times (E_D)_{BT} \,. \qquad (3.70)$$

With this as the basic equation, we shall now consider some of the methods that have been developed in the literature for estimating $(E_R)_{BT}$, from the point of view of both the moving boundary (piston) concept, and the simultaneous flow concept.

3.7.1 Moving boundary approach (E_D=1).

This is basically an extension of the 1D model described in section 3.1 to 2 dimensions, and it involves the tracking of a moving boundary which separates the displaced fluid from the displacing fluid. The flow in each region is single phase, and the mobility ratio of the flood is as defined by equation (3.7). Clearly, the simplest case arises when the mobilities of the two fluids are the same (i.e. when M = 1).

(a) Unit mobility ratio

Since the mobilities are equal, there will be no discontinuity in the pressure gradients across the interface, and the flow is identical to that of a single phase. Hence, the steady state pressure distribution derived in section 3.6 holds, and it can be used as shown by Muskat [43h] to evaluate the time to breakthrough (t_{BT}), and the recovery efficiency. The occurrence of breakthrough will be governed by the streamline along which the velocities are greatest, which is obviously the straight-line joining the injection and production wells. Let "l" be its length, and "$d\ell$" the differential length along it. Then, by Darcy's law, the true velocity of fluid particles along this stream-line is given by:

$$u = -\frac{k}{\mu\phi}\left(\frac{dp}{d\ell}\right) \qquad (3.71)$$

and so, the time taken to traverse its length is:

$$t_{BT} = \int_0^{\ell} \frac{d\ell}{u} = \frac{\phi\mu}{k}\int_0^{\ell} \frac{d\ell}{(-dp/d\ell)}. \qquad (3.72)$$

Since (from Figure 3.14a) the pore-volume of the five-spot symmetry element is

$$V_P = 2d^2 h\phi \quad ,$$

the breakthrough recovery turns out to be

$$(E_R)_{BT} = \frac{q\, t_{BT}}{V_P} = \frac{q\mu}{2d^2 kh} \int_0^{\ell} \frac{d\ell}{(-dp/d\ell)} \,. \qquad (3.73)$$

Combining this with the pressure equation (3.68) and performing the required integration, Muskat [43i] obtained the result:

$$(E_R)_{BT} = (E_A)_{BT} = 0.72 \qquad (3.74)$$

which has since then been confirmed by several authors as the areal sweep efficiency for a unit mobility five-spot flood. Recently, Morel - Seytonx [41] has demonstrated the elegance of complex variable theory, in the calculation of $(E_A)_{BT}$ etc. for various flood patterns with unit mobility ratio.

(b) Non-unit mobility ratios

In this case, we still have (due to the piston-type displacement): $(E_R)_{BT} = (E_A)_{BT}$

However, because of the unequal mobilities (which give rise to a continuously changing pressure distribution) it becomes almost impossible to track, by purely analytical means, the movement of the interface, and hence to determine the areal sweep efficiency.

Experimental and/or numerical methods had to be resorted to in order that the areal sweep might be correlated with the mobility ratio.

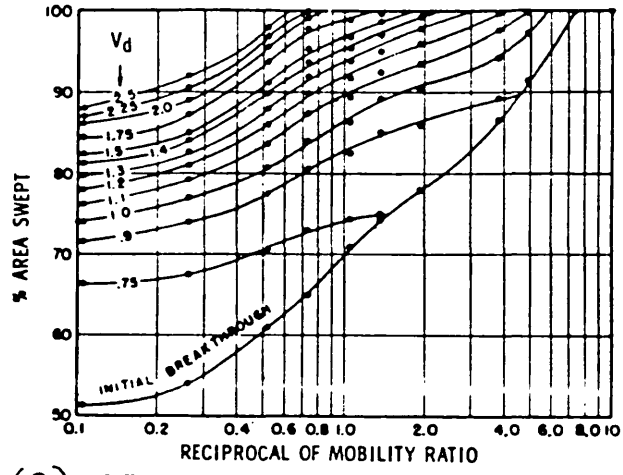
Potentiometric model studies, which had originally been used for unit mobility ratio, were extended by Aronofsky [3] to cover a range of mobility ratios. Slobod and Caudle [53] introduced the X-ray shadowgraph technique and demonstrated its value in determining the areal sweep. Similar experiments were later performed by Dyes et al[23] (using miscible fluids) who published a series of charts showing the areal sweep efficiency (at, and after, breakthrough) as a function of the mobility ratio, for different flood patterns. Figure 3.16a illustrates their results for the five-spot flood, and confirms the following properties of E_A, as would be intuitively expected:

(1) the areal sweep increases as the mobility ratio decreases (i.e. as the displacing fluid becomes less mobile); (2) for a given mobility ratio, the areal sweep continues to increase after breakthrough, as more and more fluid is injected; and (3) the breakthrough areal sweep for unit mobility ratio is approximately 0.70, which agrees well with the theoretical value given by equation (3.74).

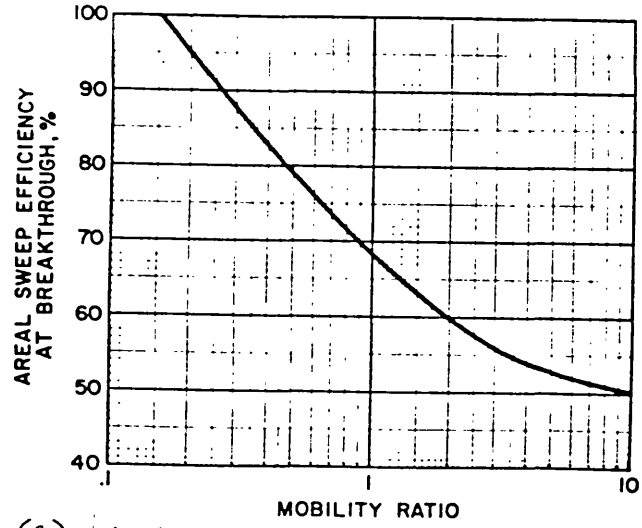
We may also note that, since miscible floods are characterised by a sharp, piston-type, interface, the use of miscible fluids in the above experiments is consistent with the "moving boundary" concept.

3.7.2 Simultaneous immiscible flow

This problem is much more complicated. The recovery at breakthrough is still given by equation (3.70), but the displacement efficiency (E_D) is no longer equal to unity. Instead, it is governed by the average saturation of the displacing fluid in the swept region, which, in turn, has to be calculated from the 1D Buckley-Leverett theory. The main difficulty, however, lies in the estimation of E_A. This is



(a) Effect of mobility ratio on the displaceable volumes injected for the five-spot pattern.



(b) Areal sweep efficiency at water breakthrough, five-spot pattern.

Figure 3.16

Correlation of Areal sweep (E_A)

vs Mobility ratio (M)

(a) Miscible floods (after Dyes et al [23]

(b) Immiscible floods (after Craig et al [14] and Craig [13c]).

because, despite the correlations between E_A and M referred to in the last section, the basic definition (3.7) of the mobility ratio poses an obvious ambiguity. Since the displacement now involves a range of saturations rather than two, fixed, "end-point" saturations, we are faced with the question as to which saturations should be used for the evaluation of the mobilities. For the oil mobility it is reasonable to use the value in the unswept region (i.e. to evaluate it at the connate water saturation Swc), although some oil is still flowing in the invaded zone. On the other hand, the saturation at which the water mobility should be calculated is open to debate, and three particular saturations (representative of the whole range) present themselves as possible candidates. These are:

(1) The maximum water saturation (1-Sor)

(2) The average water saturation ($\bar{Sw}$) in the flooded region.

(3) The frontal water saturation Swf.

Craig et al [14] performed X-ray shadowgraph tests with immiscible fluids, and reported that the use of $\bar{S}_w$ in the mobility ratio definition provided the best agreement between their results and the (E_A vs M) curves obtained by other authors from miscible flood experiments. Figure 3.16b shows the correlation of areal sweep vs mobility ratio, as obtained by Craig et al. In his monograph on waterflooding, Craig [13b] recommends this approach as the "standard" method for calculating the breakthrough recovery from an immiscible five-spot flood. The basic steps, then, are as follows:

(1) From 1D Buckley-Leverett theory, calculate the average water saturation ($\bar{S}_w$) in the flooded region.

(2) Calculate mobility ratio of the flood:

$$M = \lambda_{rw}(\bar{S}_w) / \lambda_{ro}(S_{wc}).$$

(3) Using this M, determine the breakthrough areal sweep $(E_A)_{BT}$ from Figure (3.16b)

(4) Determine displacement efficiency:

$$(E_D)_{BT} = (\bar{S}_w - S_{wc}) / (1 - S_{or} - S_{wc}).$$

(5) Finally, compute the breakthrough recovery $(E_R)_{BT}$ from equation (3.70).

Besides such elementary methods based on experimentally derived correlations, there are more sophisticated ones which, by directly combining the Buckley-Leverett theory with a suitable numerical method, seek to obtain a better representation of the five-spot flooding process. Some examples of these are:

(a) the method of Sheldon and Dougherty [51,21] which tracks the propagation of bands of constant saturation; and, more important:

(b) the stream-channel approach of Higgins and Leighton [30]

3.8 Stream channel method of Higgins and Leighton

In this method, the five-spot (see Figure 3.17a) is divided into four channels, which are isolated from each other by streamlines corresponding to a unit mobility ratio pressure distribution. Since the fluids cannot cross these streamlines, it is possible to perform a Buckley-Leverett analysis separately along each channel. Finally, the performances of the individual channels are added to obtain the cumulative performance of the five-spot.

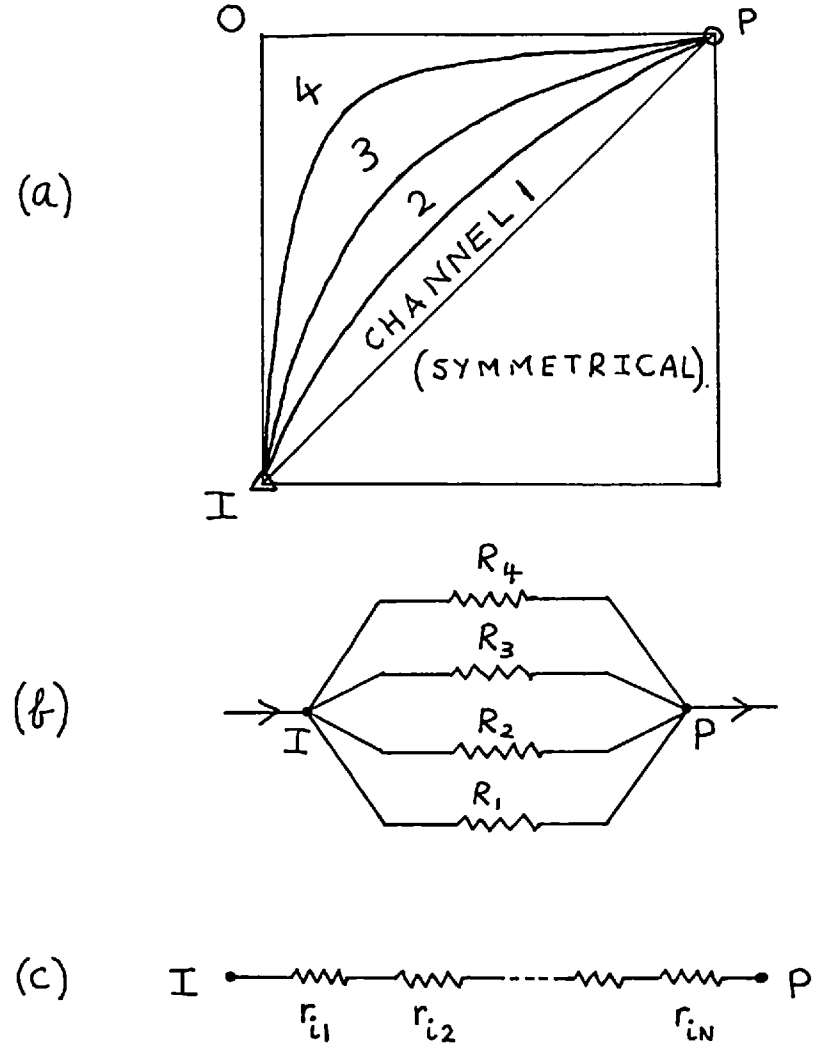


Figure 3.17

(a) Division of the quarter five-spot into stream channels [After Higgins and Leighton [30]]

(b) Electrical analogue of (a);

(c) Electrical analogue of the cells which constitute a typical channel i.

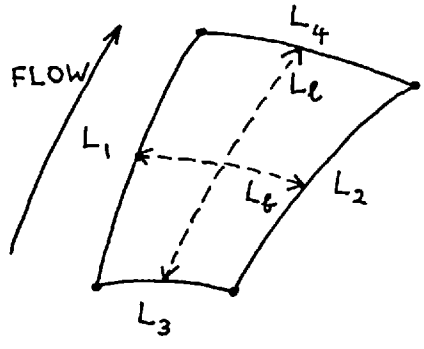


Figure 3.18

Plan view, and dimensions, of a curvilinear cell.

$L_l = \frac{1}{2}(L_1 + L_2)$; $L_b = \frac{1}{2}(L_3 + L_4)$

Shape factor $G = (L_l/L_g)$.

The displacement is assumed to take place under a constant pressure difference between the injection and production wells. Physically, therefore, these stream channels may be compared to four electrical resistances (R_1, R_2, R_3, R_4) arranged in parallel between two points across which a constant voltage difference is maintained. (Figure 3.17b). An important point to note, however, is that, in applying the 1D Buckley-Leverett theory, allowance must be made for the non-uniform cross-section of the channels. Fortunately, this merely involves (as can easily be shown) replacing the "fractional distance X" in the frontal advance equation (3.38), by the "fractional pore-volume (V_p) invaded" by a given saturation. Thus :

$$V_p(s_w, t) = Q \frac{df_w}{ds_w}. \qquad (3.75)$$

This direct proportionality between the pore volume invaded and (dfw/ds_w) is taken advantage of in the present method and, accordingly, all saturation-dependent quantities (such as k_{ro}, k_{rw} etc., including S_w itself) are expressed in terms of f_w', instead of as functions of S_w. This is perfectly valid since f_w' is monotonic in the range: [Swf to (1-Sor)].

3.8.1 Channel representation

Each channel is divided into N cells of equal volume. In terms of our electrical analogue, this division is equivalent to replacing each of the resistances in Figure 3.17b, by N resistances connected in series ($r_1, \ldots, r_n$). (Figure 3.17c). Pressure differences across these cells are calculated by incorporating a shape factor (G) into Darcy's law, so as to take account of the non-rectangular geometry of the cells. If we consider, for example, a typical cell (of unit

thickness, and whose plan view is as shown in Figure 3.18), then, the single phase Darcy equation takes the form:

$$q = kA\frac{\Delta p}{\Delta L} = k\frac{L_b \cdot 1}{L_\ell}\Delta p = k\left(\frac{1}{G}\right)\Delta p \qquad (3.76)$$

where

$$G = \frac{L_\ell}{L_b} = \frac{\frac{1}{2}(L_1 + L_2)}{\frac{1}{2}(L_3 + L_4)} . \qquad (3.77)$$

The shape factors for each of the cells, as well as the volumes of the channels, are tabulated in reference [31].

3.8.2 Analysis of channel performance

The progress of the saturation profile through the channel is calculated in a series of discrete steps. During the pre-breakthrough period, each "calculation step" corresponds to the invasion of a new cell by water, and the consequent arrival of the frontal saturation (Swf) at the cell boundary. Figure (3.19a) illustrates the saturation profiles that prevail at the end of the first three steps of the calculation.

Since the cells have equal volumes, and since volume is proportional to fw´ (equation 3.75), it follows that the cells in the invaded region occupy equal parts of the range of the fw´ axis (Fig. 3 - 19b). This leads to the following observations:

(1) At any given stage, the fw´ axis is divided into as many equal parts as there are flooded cells ;

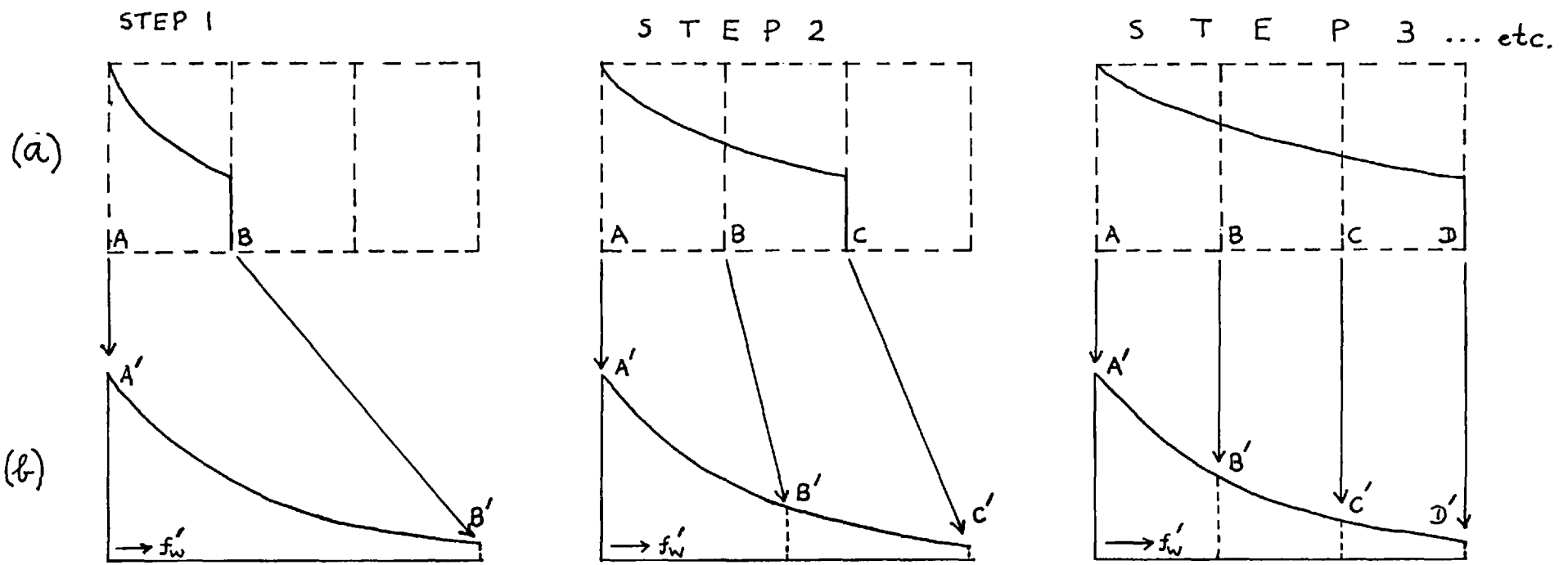


Figure 3.19

Invasion of successive cells in the Higgins and Leighton method.
(a) Saturation profiles; (b) Evaluation of saturation - dependent quantities at inter-cell boundaries.

(2) The volumetric average of a function over any cell is equal to the average obtained by integrating the function w.r.t. fw´ in the appropriate interval;

(3) The saturations (and any Sw dependent parameters) at the cell interfaces (A,B,C etc) can be found by entering the fw´ axis at the points of subdivision (A´,B´,C´ etc), and reading off the ordinates from the appropriate graph (Figure 3.19b).

To determine the performance of a channel, we need to calculate, at each stage, four basic quantities:

(a) Saturation at the producing end (Swe);

(b) Cumulative oil (and/or water) produced (Qo,Qw);

(c) The total flow rate in the channel (q_t); and

(d) The cumulative time (t).

These calculations will now be illustrated for the period before breakthrough.

3.8.3 Calculations for Pre-breakthrough period

Let us consider the performance of a typical channel i (with fractional pore volume V_i), at a stage I, when I cells have been invaded. From 1D Buckley-Leverett theory, we immediately have:

$$(a)\ (S_{we})_{i,I} = S_{wc} \qquad (3.78a)$$

$$(b)\ (Q_o)_{i,I} = V_i(\bar{S}_w - S_{wc})(I/N);\ (Q_w)_{i,I} = 0. \quad [3.78b]$$

(c) By analogy with the electrical model, the total flow rate in the channel depends on the resistances of the individual cells. For cells lying in the flooded zone, the resistances are obtained through 'volumetric averaging". The following sequence of steps outlines the procedure involved.

1. Relative permeability $= k_r$

2. Relative impedance $i_r = 1/k_r$

3. Average relative impedance $\bar{i}_r = \frac{1}{V_p}\int i_r dV_p = \left[\int\left(\frac{df_w'}{k_r}\right)\right] \Big/ \left[\int df_w'\right]$.

4. Average relative permeability:

$$\bar{k}_r = 1/\bar{i}_r = \left[\int df_w'\right] \Big/ \left[\int \frac{df_w'}{k_r}\right]$$

5. Total (two-phase) conductance of an invaded cell:

$$c_a = \frac{K}{G}\left[\frac{\overline{k_{ro}}}{\mu_o} + \frac{\overline{k_{rw}}}{\mu_w}\right]$$

6. Total conductance of uninvaded cell:

$$c_b = \frac{k}{G}\ \frac{k_{ro}(S_{wc})}{\mu_o}$$

7. Using subscripts (i,j) to denote the channel and cell numbers, respectively, we have:

(i) Resistance of invaded part of channel:

$$(R_a)_i = \sum_{j=1}^{I}\left[1/(c_a)_{ij}\right]$$

(ii) Resistance of uninvaded part of channel:

$$(R_b)_i = \sum_{j=I+1}^{N}\left[1/(c_b)_{ij}\right]$$

8. Finally, from (7), the total flow rate through channel i (at stage I) is

$$(q_t)_{i,I} = \frac{\Delta p}{(R_a)_i + (R_\ell)_i} \qquad (3.78c)$$

where Δp is the fixed pressure difference between the injection and production wells. Until breakthrough occurs, we clearly have $q_o = q_t$.

(d) Cumulative time at the end of stage I is given by :

$$t_{i,I} = t_{i,(I-1)} + \frac{\text{oil produced between stages (I-1) and (I)}}{\text{average oil rate during this interval}}$$

$$= t_{i,(I-1)} + \frac{(Q_o)_{i,I} - (Q_o)_{i,(I-1)}}{\frac{1}{2}\left[(q_o)_{i,I} + (q_o)_{i,(I-1)}\right]} . \qquad (3.79)$$

We have thus completed the analysis of channel performance until breakthrough. The analysis for the post-breakthrough period follows along much the same lines, with the exception that it takes into account the changing saturation at the outflow end, and the amount of water that is being produced. Finally, the overall performance of the five-spot is calculated by a linear combination of the individual channel performances.

The chief limitations of the Higgins and Leighton method are as follows:

(1) The channel shapes are based on unit mobility ratio.

(2) The geometry of channels is assumed to be invariant in time.

(3) Since the co-ordinates of the cells have not been published, it is not possible to plot the saturation distribution inside the five-spot.

(4) The method is based on a constant pressure difference between wells, and is hence not strictly valid for constant-rate floods.

Despite these large drawbacks, however, the method has been reported to give reasonably good results in the prediction of waterflood performance. Finally in a recent paper by Albright et al [1], a purely analytical solution to the five-spot problem is indicated. The approach used was to reduce the 2D differential equation to a 1D (Buckley-Leverett) form by means of a co-ordinate transformation involving the isobars and streamlines. Although the method seems promising, more details of the theory would be required before an assessment of its true potential can be made.

CHAPTER 4

Principles of Finite Elements and their Application to Two Phase Flow

Much of the theory on which the Finite Element method is based is now found in text books on the subject. In his work, Zienkiewicz [62] has given what is perhaps the most comprehensive, state-of-the-art exposition of the technique. The emphasis of various authors has naturally tended to be in the direction of their particular fields of interest. Thus, for example, while Zienkiewicz [62] and Desai and Abel [19] have approached the subject from the point of view of structural analysis, Argyris [2] has demonstrated its application to aeronautics, and Connor and Brebbia [12] have dealt with its use in Fluid Mechanics problems. The subject has also received its share of abstract, rigorous treatment in the hands of numerical analysts and mathematicians, of which Oden [45,46] and Rachford [49] are examples.

Beginning with a somewhat generalised description of the finite element method, this chapter will move on to consider, in some detail, the way in which it has been used in this thesis.

4.1 Spatial discretization by finite elements

The starting point for this method is the division of the flow domain into an arbitrary number of discrete (usually triangles or polygons quadrilaterals) known as finite elements (Figure 4.1). The description of these figures is considerably facilitated by a well-known feature of Cartesian geometry-namely, that all triangles and quadrilaterals

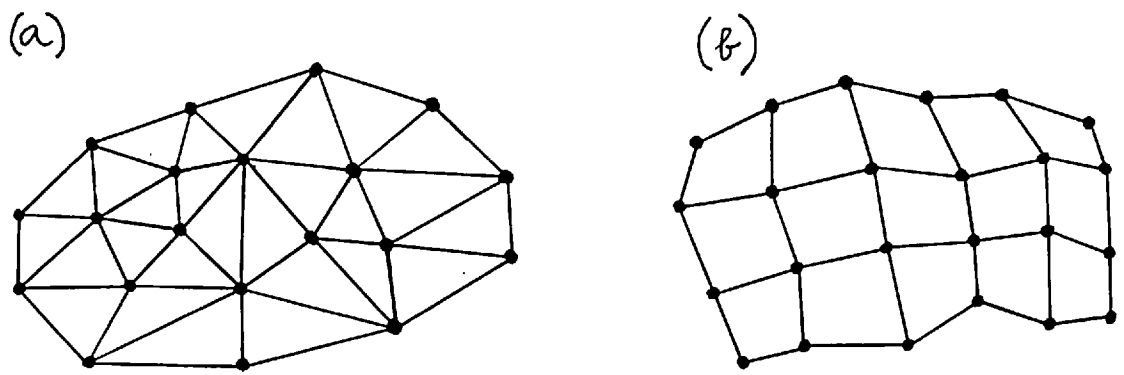


Figure 4.1 Spatial discretization by finite elements (a) triangles and (b) quadrilaterals.

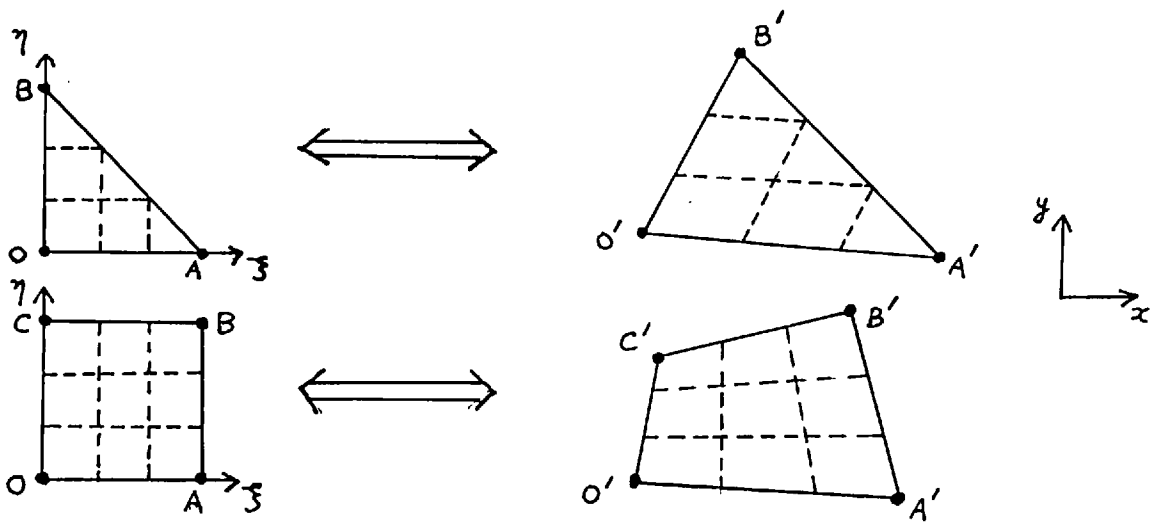


Figure 4.2 Mapping of triangles and quadrilaterals in the (x,y) plane onto standard figures in the (ξ,η) plane.

in the (x,y) plane, whatever their shape or location, can be mapped uniquely onto "standard" invariant figures in another (ξ,η) plane. These standard figures are, respectively:

(1) the right isosceles triangle, with vertices at (0,0), (1,0) and (0,1); and

(2) the unit square, whose corners are at (0,0), (1,0), (1,1) and (0,1).

Figure 4.2 shows an example of these two types, and also illustrates some of the internal trajectories in the (x,y) plane, which become mapped onto lines which are parallel to the ξ and η axes in the standard figures. This "one-to-one" correspondence which exists between the co-ordinates of points belonging to the original figures (x,y), and those of the standard figures (ξ, η), can be expressed in the form:

$$\left.\begin{aligned} x &= x(\xi,\eta) \quad \text{and} \\ y &= y(\xi,\eta) \end{aligned}\right\}. \tag{4.1}$$

The above transformations will be either linear, or, polynomial in ξ and η, according as the bounding sides of the original figures (in the x,y plane) are straight lines or curves. In general, these transformations can be determined without much difficulty, and let us therefore assume for the present that they are known. We can thus regard ξ and η as the normalised coordinates (since $0 \leq (\xi,\eta) \leq 1$) of any point P(x,y) in the (x,y) diagram.

4.2 Approximation of variables

Having discretized the domain by a series of finite elements,

the chief variables (in this case po and Sw) are approximated, over each element, by functions which are of known character, and are defined to be continuous along the inter-element boundaries. Indeed, it is this piecewise continuous nature of the assumed function that imparts to the finite element method (FEM) its distinctive character.

Except in the vicinity of sources and sinks, where the use of sophisticated functions (such as logarithms) may be warranted, it is common practice to choose polynomials (in x and y) to represent the approximating function. However, since (as described in the previous section) the co-ordinates (x,y) are themselves polynomials of the normalised co-ordinates (ξ, η), it is more convenient to write the approximating polynomial directly in terms of ξ and η.

Clearly, a polynomial in two variables (e.g. ξ and η) is simply a linear combination of a finite number of binomial terms (i.e. products of powers of ξ and η) which we shall call modes. So, within a typical finite element, the polynomial approximation can be expressed as:

$$\psi = \sum_{i=1}^{n} c_i f_i(\xi, \eta) = \underline{f}\,\underline{c} \qquad (4.2)$$

where:

ψ is the object variable (Po or Sw) ;

n = number of polynomial modes chosen;

$\underline{f} = [f_1(\xi,\eta), \ldots, f_n(\xi,\eta)]$ is the row vector of modes ; and

$\underline{c} = [c_1, \ldots, c_n]^T$ is the column vector of scalar multipliers.

Figure 4.3 shows a "Pascal triangle" composed of these basic modes, which have been arranged so that, starting with a zero-degree polynomial at

the apex (which corresponds to the term unity), each successive row introduces terms of progressively higher degree.

4.3 Calculation of derivatives

Since the polynomial in equation (4.2) is written in terms of ξ and η, the evaluation of partial derivatives of Ψ with respect to x and y is, in general, not straightforward, involving the Jacobian matrix ($\underline{J}$) of the transformation from the (ξ,η) system to the (x,y) system. To begin with, the first order derivatives ($\partial\Psi/\partial x$) and ($\partial\Psi/\partial y$) can be obtained by a direct application of the chain rule for differentiation:

$$\left.\begin{aligned} \frac{\partial}{\partial \xi} &= \frac{\partial x}{\partial \xi}\frac{\partial}{\partial x} + \frac{\partial y}{\partial \xi}\frac{\partial}{\partial y} \;; \text{ and} \\ \frac{\partial}{\partial \eta} &= \frac{\partial x}{\partial \eta}\frac{\partial}{\partial x} + \frac{\partial y}{\partial \eta}\frac{\partial}{\partial y} \;. \end{aligned}\right\} \qquad (4.3)$$

And so, inverting this pair of equations, we have:

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix} , \qquad (4.4)$$

in which the (2x2) matrix enclosed within brackets will be recognised as the Jacobian matrix [i.e $\underline{J}\,(\xi,\eta)$]. Simplifying the nomenclature, we may write equation (4.4) as:

$$\underline{\nabla}_{xy} = \underline{J}^{-1}\,\underline{\nabla}_{\xi\eta} \;. \qquad (4.5)$$

If we denote the two rows of $\underline{J}^{-1}$ by $\underline{j}^{T}_{x}$ and $\underline{j}^{T}_{y}$ respectively, then we can extract (separately) the partial derivatives w.r.t x and y, in the form:

$$\left.\begin{aligned} \frac{\partial}{\partial x} &= \underline{j}_x^T \, \underline{\nabla}_{\xi\eta} \quad , \quad \text{and} \\ \frac{\partial}{\partial y} &= \underline{j}_y^T \, \underline{\nabla}_{\xi\eta} \end{aligned}\right\} . \tag{4.6}$$

Equation (4.6) can now be used as the basis for generating higher order partial derivatives w.r.t.x and y. Thus, in general, we have :

$$\begin{aligned} \frac{\partial^{\ell+m}}{\partial x^{\ell} \partial y^{m}} \psi &= \left[\underline{j}_x^T \, \underline{\nabla}_{\xi\eta}\right]^{\ell} \left[\underline{j}_y^T \, \underline{\nabla}_{\xi\eta}\right]^{m} \psi \\ &= \mathcal{D}(\xi, \eta, \ell, m)\, \psi \; ; \end{aligned} \tag{4.7}$$

where D is the resulting scalar differential operator. Substituting this into equation (4.2), we have:

$$\frac{\partial^{\ell+m}}{\partial x^{\ell} \partial y^{m}} \psi = \sum_{i=1}^{n} c_i \, \mathcal{D} f_i = \left[\mathcal{D}\underline{f}\right] \underline{c} \quad , \tag{4.8}$$

which is, in fact, the generalised form of (4.2), since the latter corresponds to the special case l=m=0 (i.e. D≡1).

4.4 Degrees of freedom and shape functions

Although equation (4.2) is the basis of our numerical representation of the unknown function, an alternative (though equivalent) version of the same equation is usually preferred, because it is not only more convenient to use, but also permits an easier visualization of the physical quantity in question. Basically, this involves replacing the scalars $[c_i]$ in equation (4.2) by "n" physically meaningful

parameters connected with the object variable Ψ. These parameters are known as degrees of freedom (DOF's), and consist normally of the function value Ψ and/or its derivatives (w.r.t. x and y), defined at selected points of the element called nodes.

It is worth noting that, in general, a node can accommodate more than one DOF (e.g. Ψ and $\partial\Psi/\partial x$). Consequently, the number of nodes belonging to an element will be less than, or equal to, the number of its degrees of freedom (n), depending on whether or not "multi-DOF" nodes are present. In the majority of applications, however, each node has only one DOF associated with it - namely, the function value Ψ at that point.

Turning now to equation (4.2), its alternative version may be derived as follows:

Let $\underline{\psi}_e = \left[\psi_{e_1}, \ldots\ldots, \psi_{e_n}\right]^T$ be the vector

containing the n degrees of freedom of the element. Then, for a typical DOF given by:

$$\psi_{e_i} = \frac{\partial^{\ell_i + m_i}}{\partial x^{\ell_i} \partial y^{m_i}} (\psi) \qquad (4.9)$$

which is situated at the node (ξ_i, η_i), we have, by using equation (4.8) :

$$\psi_{e_i} = \left[\mathcal{D}_i \underline{f}(\xi_i, \eta_i)\right] \underline{c} \,, \qquad (4.10)$$

where $D_i = D(\xi, \eta, l_i, m_i)$ is the differential operator corresponding

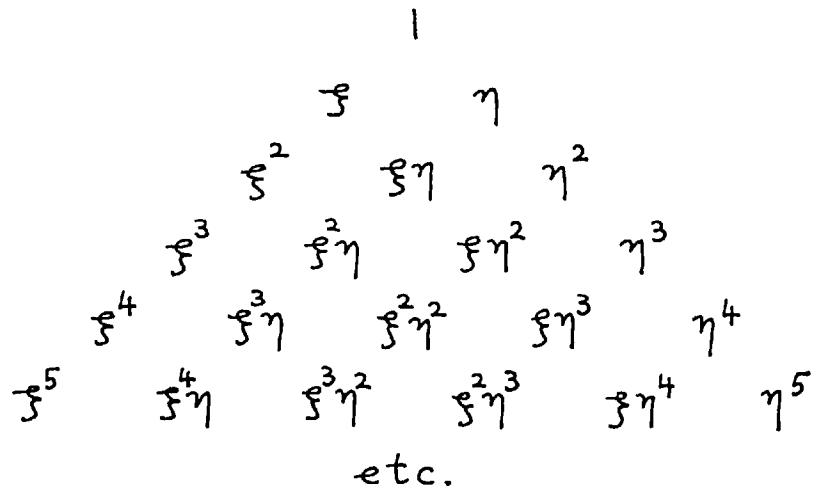


Figure 4.3

A Pascal triangle consisting of polynomial modes.

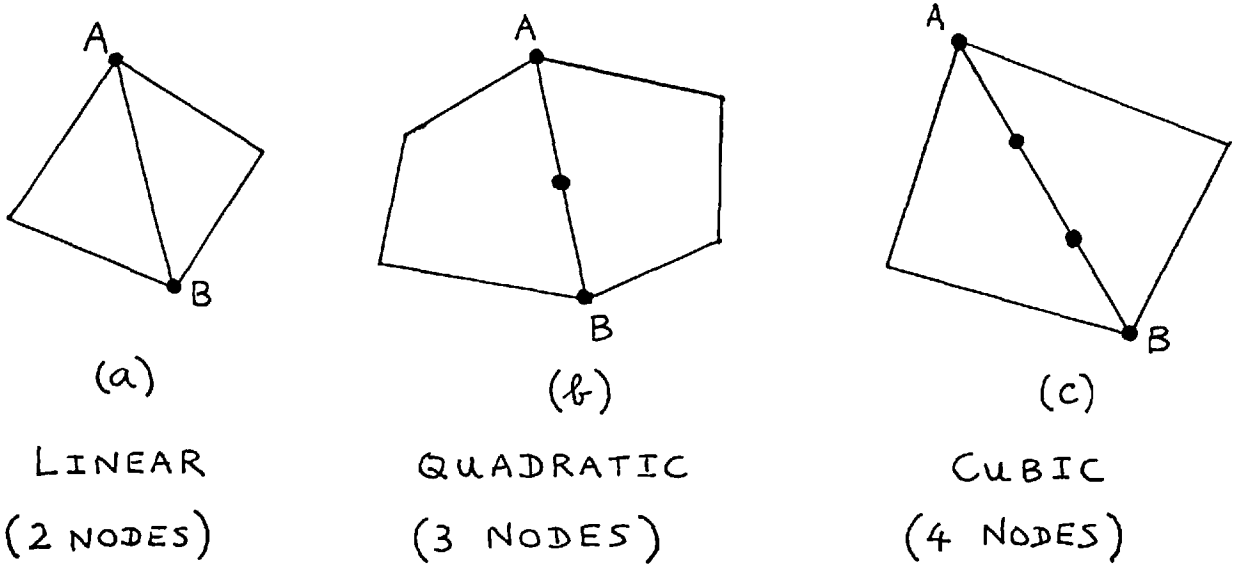


Figure 4.4

Positioning of the nodes to obtain C^0 continuity across inter-element boundary AB.

to Ψe_i, and Di $\underline{f}$ ($\xi i, \eta i$) denotes the value of Di $\underline{f}$ at the node ($\xi i, \eta i$). Collecting together all the DOF's ($i = 1 \rightarrow n$) given by the equation (4.10), we can write (in vector form):

$$\underline{\Psi e} = \underline{F}\,\underline{c} \tag{4.11}$$

where $\underline{F}$ is an (nxn) matrix, whose rows are given implicitly R.H.S. of equation (4.10). $\underline{F}$ can now be inverted to yield:

$$\underline{c} = \underline{F}^{-1}\,\underline{\Psi e} \tag{4.12}$$

and, on substituting $\underline{c}$ into the original equation (4.2), we obtain:

$$\Psi = \left[\underline{f}(\xi,\eta)\,\underline{F}^{-1}\right]\underline{\Psi e} = \underline{Ne}\,\underline{\Psi e}\,. \tag{4.13}$$

In this equation, $\underline{Ne}$ will be recognised as the familiar (1 x n) row-vector containing what are traditionally referred to as the "element shape functions" or "basis functions". Furthermore, since :

$$\underline{Ne} = \underline{f}\,\underline{F}^{-1}, \tag{4.14}$$

it follows that we now have a systematic procedure for calculating the shape functions, given:

(a) the polynomial modes ;

(b) the degrees of freedom ; and

(c) the dimensionless co-ordinates ($\xi i, \eta i$) of the nodes.

Equation (4.13) is exactly equivalent to (4.2), and is the form often encountered in classical finite element analysis. By examining it in some detail, we may deduce the following, more or less intuitive,

properties of shape functions and DOF's.

(1) The number of DOF's in each element equals the number of polynomial modes chosen, and each DOF(i) possesses its own shape function Ne_i.

(2) Application of the differential operator Di to equation (4.13) at node (ξi,ηi) should yield the value of the DOF (Ψei). Thus :

$$\mathcal{D}_i \psi(\xi_i,\eta_i) = \left[\mathcal{D}_i \underline{Ne}(\xi_i,\eta_i)\right] \underline{\psi_e} = \psi_{e_i} , \qquad (4.15)$$

which implies that the term [DN] satisfies the Kronecker Delta Function :

$$\mathcal{D}_i Ne_j(\xi_i,\eta_i) = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{if } j \neq i . \end{cases} \qquad (4.16)$$

This merely re-iterates the well known fact that, for every DOF Ψe_i, the term Di Ne_i (or simply the shape function Ne_i, if $\ell i = m_i = 0$) takes on the value unity at the node (ξi,ηi) where Ψe_i operates, and is zero at all other nodes.

(3) When the function Ψ happens to be constant (=Ψc, say) over the whole element, then the polynomial "approximation" should reduce to an "exact" representation, and so, equation (4.13) should read :

$$\underline{Ne}\ \underline{\psi_e} = \psi_c . \qquad (4.17)$$

However, in such a case, all the derivatives of Ψ are zero. We may therefore remove from the vector Ψe those DOF's (if any) which involve derivatives and set the remaining DOF's equal to the constant Ψc.

So, equation (4.17) now becomes :

$$\left.\begin{aligned} \sum_{i=1}^{n} N_{e_i}\,\psi_c &= \psi_c \\ \text{or, simply} \quad \sum_{i=1}^{n} N_{e_i} &= 1 \end{aligned}\right\} \quad \text{for } l_i = m_i = 0. \qquad (4.18)$$

where the summation (as indicated by the condition $l_i = m_i = 0$) is taken over the non-derivative shape functions (i.e. those corresponding to the function value only).

4.5 Functional continuity and finite element types

Since, by definition, a polynomial is continuous within the region in which it is defined, it follows that, in the context of finite elements, the function value ψ, and all its derivatives, are continuous inside the elements. However, due to the piecewise nature of the polynomial, discontinuities can, and do, arise across the inter-element boundaries (i.e. sides). A finite element approximation is said to possess C^m continuity if all its partial derivatives of order upto, and including, m are continuous across the interfaces (and hence also throughout the domain).

Where partial differential equations of second or lower order are involved, it normally suffices for the function value alone to be continuous at all points, thereby placing the finite element in the C^0 category. However, in some areas of engineering, such as plate bending, where fourth order PDE's occur, it is necessary to ensure the continuity of both the function and its slopes, and hence C^1 finite elements have to be employed. C^0 finite elements are

alternatively known as Lagrangian elements, while C^1 and higher types are called Hermitian elements. The required minimum continuity will thus be dictated by the needs of the problem, but in most cases, it is adequate to use the C^o type elements.

In order to maintain functional continuity between neighbouring elements, it is evidently necessary that the variation of ψ along the interface be controlled by parameters common to both elements. Here, we see yet another advantage which equation (4.13) possesses over (4.2). In the latter, the scalars $[c_i]$ are set up independently for each element, which means that special constraining relationships have to be enforced to obtain the required inter-element continuity. This complication renders the equation very cumbersome to handle. Equation (4.13), by contrast, is ideally suited to this purpose, and provides a very natural way of dealing with continuity requirements. This is so, because it allows a great deal of flexibility both in the choice of the degrees of freedom, and in the positioning of the nodes. We can easily see that, if nodes are placed along inter-element boundaries, then they will :

(a) belong to both elements ; and

(b) exercise direct control over the variation of ψ along this side.

These considerations provide us with the key to determining the number, and locations, of the DOF's that need to be cited on the interface. Suppose that the highest polynomial term that occurs in the approximation is of degree n. Then, as the interface is approached from within the two elements situated on either side of it, the variation of ψ along the interface will converge to two separate (1D) polynomials, both of degree n. This implies that if, ψ is to

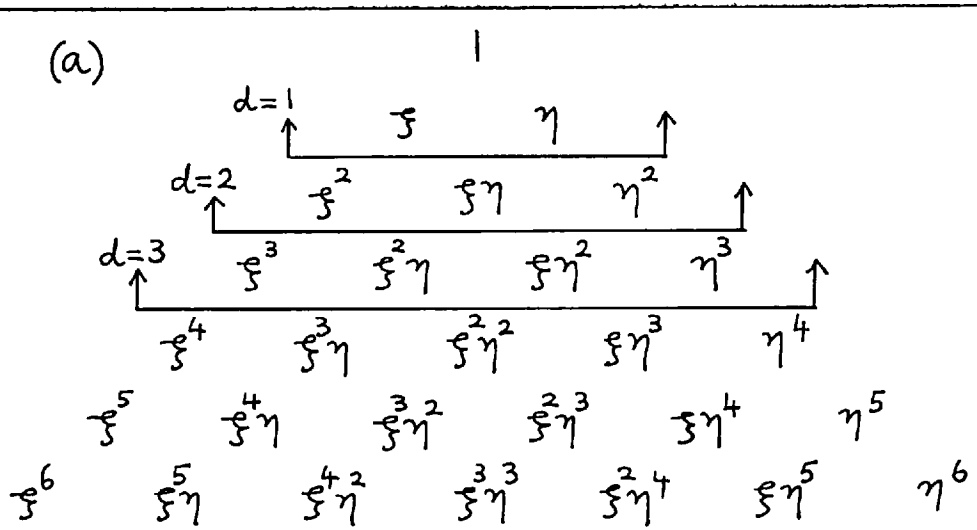
be continuous across the interface, these polynomials must coincide. But, we know that a 1D polynomial of degree n is uniquely determined if it is made to pass through (n + 1) points. Hence it follows that functional continuity will be ensured if (n + 1) DOF's are located along the side. Two of these are positioned at the extremities (i.e. vertices), while the remaining (n - 1) DOF's are distributed regularly along the interface. These arguments hold equally for triangles and quadrilaterals. (Figure 4.4).

Provided that all the finite elements constituting the domain are of the same type (as is normally the case), we can extend the above reasoning, and state that all inter-element boundaries will have the same nodal configuration - a feature which is consistent with the requirements of symmetry. Figures 4.5, 4.6 and 4.7 illustrate the locations of the nodes for some of the common types of finite elements.

The nodes on the element boundary do not, however, preclude the existence of internal nodes which also contribute to the polynomial variation inside the element, without affecting the inter-element continuity. The choice of the nodal system will therefore be determined by the modes we wish to include in the polynomial. On the basis of the choice of polynomial modes, we can distinguish three broad classes of finite elements (Figures 4.5, 4.6, 4.7).

(1) Complete polynomials (triangles)

As can be seen from the Pascal-triangle of Figure 4.5a, a complete polynomial of degree "d" in ξ and η can be constructed by



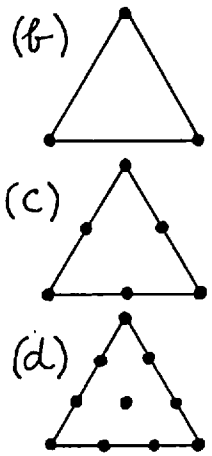
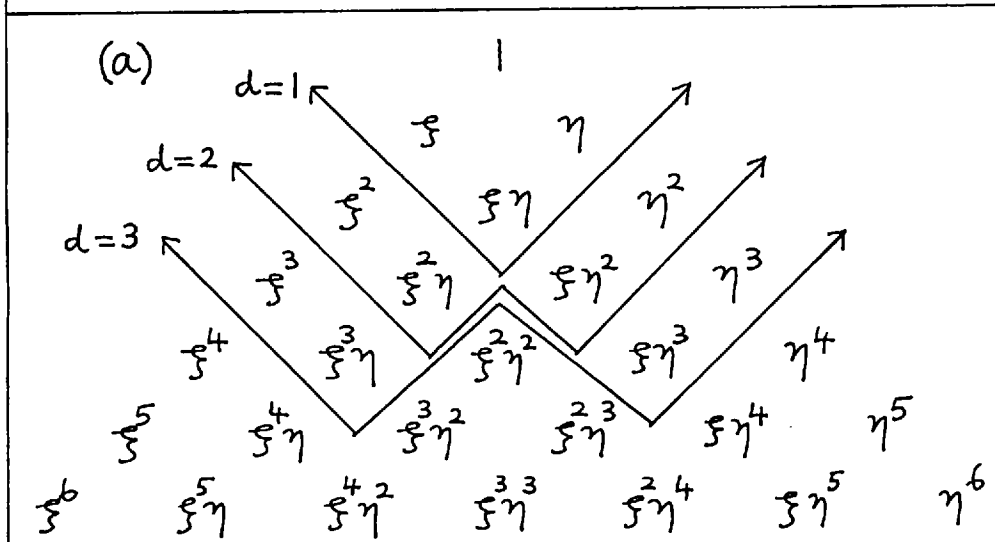


Figure 4.5 Complete polynomials (Triangles); d = degree of polynomial.



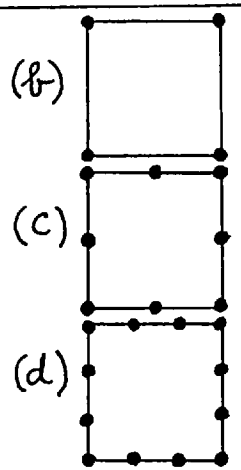


Figure 4.6 Serendipity family (Rectangles); d = degree of polynomial.

taking all the modes that lie on, and above, the (d + 1) th row. The total number of modes, n, is :

$$n = \tfrac{1}{2}(d+1)(d+2) \; ; \qquad (4.19)$$

which is also the number of nodes that the element must possess. Clearly, triangles are the obvious choice for representing complete polynomials in 2D, since the required number of nodes is easily obtained by drawing lines parallel to the three sides, and by placing the nodes at the intersection points. Linear (Chapeau), Quadratic and Cubic triangles are shown by the cases d=1,2 and 3 respectively, in Figure 4.5.

(2) The Serendipity family

These are quadrilateral elements, having the usual configuration of boundary nodes, but with relatively few internal nodes. In fact, the latter do not appear until after the cubic element. For each polynomial (of degree "d"), they contain not only the complete set of modes used by the triangles, but also two additional terms $\xi^d\eta$ and $\xi\eta^d$, so that the number of nodes per element becomes:

$$n = \tfrac{1}{2}(d+1)(d+2) + 2 \; . \qquad (4.20)$$

Thus, although powers of (d + 1) occur in the expansion, the polynomial is complete only to the power "d". Figure 4.6 shows the modes involved, and the corresponding nodal configurations, for this class of finite elements. Of these, the 8-node (quadratic) element, in particular, has enjoyed a wide popularity in structural mechanics and other civil engineering applications. The name "Serendipity" was introduced by Zienkiewicz [62a], to draw attention to the fact that,

in earlier days, when finite element theory was being developed, there was no systematic procedure for deriving the shape functions for this class of elements, and consequently they were derived by trial and error.

(3) Tensor product family

These are quadrilateral elements in which the approximating polynomial, as the title implies is a product of two 1-dimensional polynomials in the ξ and η directions respectively. If the polynomial is of degree "d" in each direction, then the number of DOF's per element is given by:

$$n = (d+1)^2 \tag{4.21}$$

which, for Lagrangian (C^0) elements, will also represent the number of nodes. In each element, the nodes are arranged regularly, row-by-row [(d+1)x(d+1)], and it can be observed that these elements possess more internal nodes than the corresponding serendipity elements. Figure (4.7a) shows the polynomial modes involved, while Figures (4.7b,c and d) illustrate the bilinear (Chapeau), biquadratic and bicubic (C^0) tensor product elements, respectively. Figure 4.7e, however, shows a bicubic element which has C^1 continuity, and therefore belongs to the Hermitian type. We shall now consider briefly the structure of such elements.

Hermitian elements in 1D

To define a polynomial of odd degree (2d+1), we may proceed in two ways. Since the required number of DOF's is (2d+2), we may either

(1) represent the DOF's entirely by function values at (2d+2) nodes,

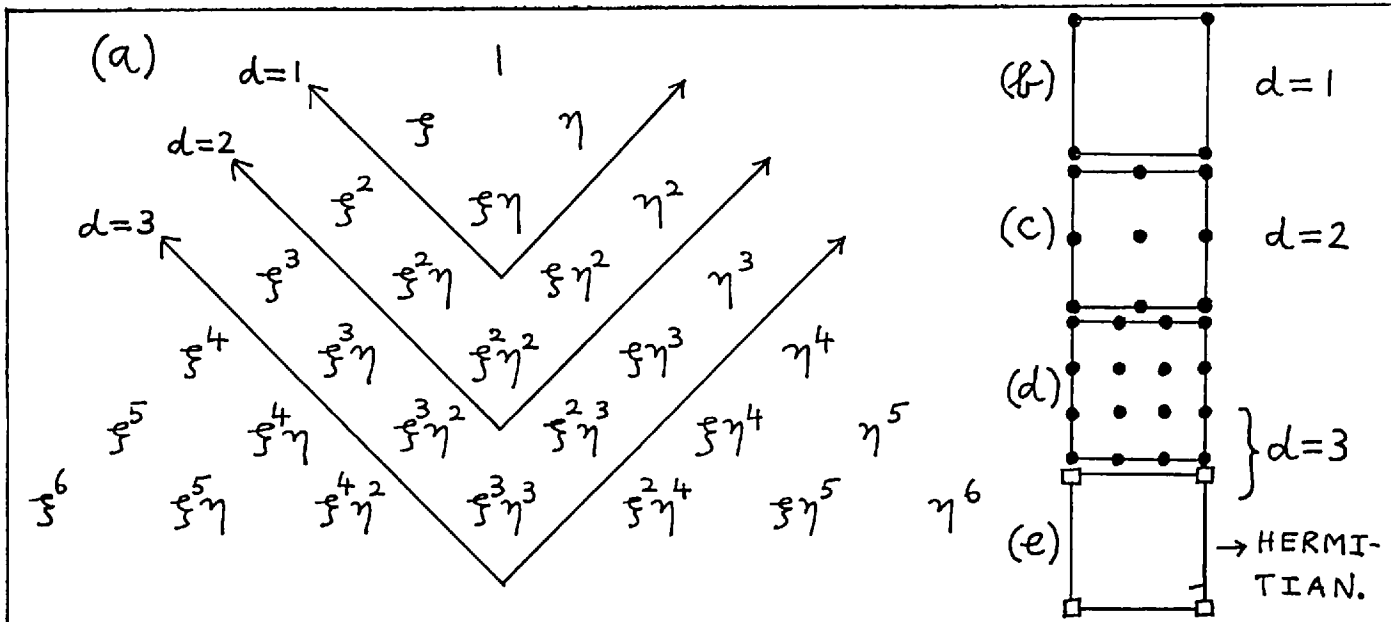


Figure 4.7 Tensor product family (Rectangles); d = degree of polynomial. (After Connor and Brebbia (12a)).

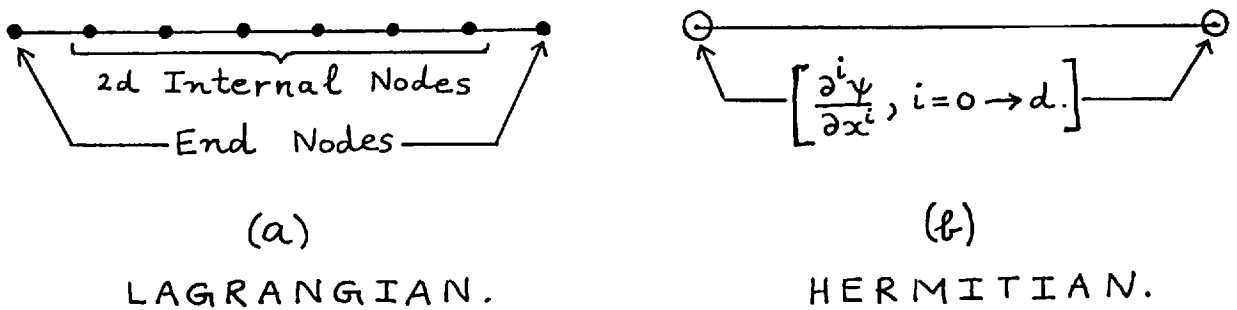


Figure 4.8 Representation of 1D polynomial of degree (2d+1).

placed regularly along the interval; this is the Lagrangian case (Figure 4.8a).

or

(2) use only the two "end nodes", and assign multiple DOF's to them. Each node will thus contain (d + 1) DOF's, which will consist of the function value and the first "d" derivatives:

$$\left[\psi, \left(\frac{\partial \psi}{\partial x}\right), \ldots\ldots, \left(\frac{\partial^d \psi}{\partial x^d}\right)\right] \quad \text{(Figure 4.8b)}.$$

This is the 1D Hermitian element, in which the presence of derivatives at the "end nodes" ensures that the approximating polynomial has C^d continuity throughout the domain.

Hermitian elements in 2D

In 2 dimensions, we shall restrict our consideration to rectangular hermitian elements only (instead of general quadrilaterals), so that the sides of the elements are parallel to the x- and y-co-ordinate axes. By analogy with the 1D hermitian element, only the four "corner nodes" will be present, and at each node i, the DOF's will consist of the tensor products of the 1D DOF's. Thus:

$$\psi_{e_i} = \left[\frac{\partial^\ell}{\partial x^\ell}\right]\left[\frac{\partial^m}{\partial y^m}\right]\psi_i \; ; \quad \ell = 0 \to d \, , \, m = 0 \to d \qquad (4.22a)$$

or, collectively, for the whole element, we can write :

$$\underline{\psi_e} = \left[\left\{\left(\frac{\partial^{\ell+m}\psi}{\partial x^\ell \partial y^m}\right)_i \;\middle|\; 0 \le (\ell, m) \le d\right\}, \; i = 1 \to 4\right], \qquad (4.22b)$$

where i is the index covering the four nodes. Due to the tensor product formulation, all the "single" derivatives:

$$\frac{\partial^{\ell}\psi}{\partial x^{\ell}} : \ \ell = 1 \longrightarrow d, \ \text{and}$$

$$\frac{\partial^{m}\psi}{\partial y^{m}} : \ m = 1 \longrightarrow d$$

will be continuous across the element sides. Furthermore, since these sides are parallel to the x- and y- axes, it follows that the "mixed" derivaties :

$$\frac{\partial^{\ell+m}\psi}{\partial x^{\ell}\partial y^{m}} \qquad (\ell \neq 0, \ m \neq 0)$$

will also be continuous from element to element. We can conclude, therefore, that the tensor product hermitian element of degree (2d + 1) in each variable, provides complete C^d continuity over the whole domain.

We have thus far considered the properties of three classes of finite elements, and it is appropriate at this point to make some comments on the system:

(1) Although it includes most of the "standard" elements, the classification is by no means a comprehensive one; several intermediate or "mixed" elements have been proposed in the literature.

(2) Hermitian elements are also available for triangles, besides the purely rectangular ones considered here. A classic example of this is the 21 DDF quintic triangular element which provides C^1 continuity.

(3) Rectangular hermitian elements are restricted to odd degree polynomials in x and y, while Lagrangian elements can take on any desired polynomial.. Of the variety of elements that have been presented, only a few have been applied to the simulation of waterfloods in the present study. These are as follows:

1 Dimensional models

(1) Linear (Chapeau) 2 nodes , 2 DOF's ;

(2) Quadratics 3 nodes , 3 DOF's ; and

(3) Hermite cubics 2 nodes , 4 DOF's .

2 Dimensional models

(1) Tensor product chapeau (4 nodes , 4 DOF's) Fig. 4.7b

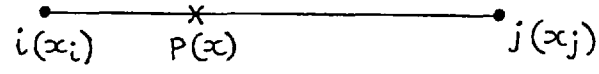
(2) Tensor product quadratics (9 nodes , 9 DOF's) Fig. 4.7c

(3) Tensor product H. cubics (4 nodes ,16 DOF's) Fig. 4.7e

Moreover, since a regular orthogonal grid was used in the simulation, the 20 elements were strictly rectangular in shape (rather than quadrilaterals).

4.6 Calculation of Shape Functions

Using the approach described in section 4.4, the shape functions for the three (1D) elements can now be calculated. Let the terminal nodes of a typical element be i, j respectively, with co-ordinates x_i, x_j relative to the origin.



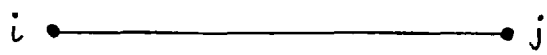
Then, the normalised co-ordinate of a point P(x) is given by

$$\xi = \frac{x - x_i}{x_j - x_i} = \frac{x - x_i}{\Delta x} , \qquad (4.23)$$

and the 1 dimensional version of equation (4.4) is:

$$\frac{\partial}{\partial x} = \frac{1}{\Delta x} \frac{\partial}{\partial \xi} . \qquad (4.24)$$

4.6.1 Linear (Chapeau) Shape functions



Degrees of freedom : $\underline{\psi}_e = [\psi_i \quad \psi_j]^T$

Normalised co-ordinates of nodes : $0 \; ; \; 1$.

Polynomial modes : $\underline{f} = [1 \quad \xi]$

$$\mathcal{D}(\xi, 0)\,\underline{f} = [1 \quad \xi]$$

Substituting the nodal co-ordinates: $\underline{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$

Finally, from equation (4.14),

$$\underline{Ne} = \underline{f}\,\underline{F}^{-1} = [1 \quad \xi]\begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$

i.e.

$$\underline{Ne} = \begin{bmatrix} Ne_{10} \\ Ne_{20} \end{bmatrix}^T = \begin{bmatrix} 1-\xi \\ \xi \end{bmatrix}^T . \qquad (4.25)$$

Also, since $Ne_{10} + Ne_{20} = 1$, we have thus verified equation (4.18) which states that the "non-derivative" shape functions add up to unity.

4.6.2 Quadratic shape functions

Degrees of freedom : $\underline{\psi}_e = [\psi_i \quad \psi_k \quad \psi_j]^T$

Normalised nodal co-ordinates : $0 \; ; \; \frac{1}{2} \; ; \; 1$.

Polynomial modes : $\underline{f} = [1 \quad \xi \quad \xi^2]$

$$\mathcal{D}(\xi,0)\,\underline{f} = \begin{bmatrix} 1 & \xi & \xi^2 \end{bmatrix}$$

Inserting nodal co-ordinates,

$$\underline{F} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} \\ 1 & 1 & 1 \end{bmatrix}$$

And so,

$$\underline{Ne} = \underline{f}\,\underline{F}^{-1} = \begin{bmatrix} 1 & \xi & \xi^2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{bmatrix}$$

i.e.

$$\underline{Ne} = \begin{bmatrix} Ne_{10} \\ Ne_{20} \\ Ne_{30} \end{bmatrix}^T = \begin{bmatrix} (1-\xi)(1-2\xi) \\ 4\xi\,(1-\xi) \\ \xi\,(2\xi-1) \end{bmatrix}^T . \qquad [4.26]$$

Once again, we see that : $Ne_{10} + Ne_{20} + Ne_{30} = 1$.

4.6.3 Hermite cubic shape functions

Degrees of freedom : $\underline{\psi_e} = \left[\psi_i , \left(\frac{\partial \psi}{\partial x}\right)_i , \psi_j , \left(\frac{\partial \psi}{\partial x}\right)_j \right]^T$

Normalised nodal co-ordinates : 0 ; 1 .

Polynomial modes : $\underline{f} = \begin{bmatrix} 1 & \xi & \xi^2 & \xi^3 \end{bmatrix}$

$$\begin{cases} \mathcal{D}(\xi,0)\,\underline{f} = \begin{bmatrix} 1 & \xi & \xi^2 & \xi^3 \end{bmatrix} \\ \mathcal{D}(\xi,1)\,\underline{f} = \frac{1}{\Delta x}\begin{bmatrix} 0 & 1 & 2\xi & 3\xi^2 \end{bmatrix} \end{cases}$$

Substituting the appropriate nodal-co-ordinates for each DOF. :-

$$\underline{F} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\Delta x} & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & \frac{1}{\Delta x} & \frac{2}{\Delta x} & \frac{3}{\Delta x} \end{bmatrix}$$

Hence :

$$\underline{Ne} = \begin{bmatrix} 1 & \xi & \xi^2 & \xi^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \Delta x & 0 & 0 \\ -3 & -2\Delta x & 3 & -\Delta x \\ 2 & \Delta x & -2 & \Delta x \end{bmatrix}$$

i.e.:

$$\underline{Ne} = \begin{bmatrix} Ne_{10} \\ Ne_{11} \\ Ne_{20} \\ Ne_{21} \end{bmatrix}^T = \begin{bmatrix} 1 - 3\xi^2 + 2\xi^3 \\ \xi(1-\xi)^2 \Delta x \\ 3\xi^2 - 2\xi^3 \\ -\xi^2(1-\xi)\Delta x \end{bmatrix}^T \qquad (4.27$$

In this case, we have $Ne_{10} + Ne_{20} = 1$. It is interesting to note the dimensionality of each hermite cubic shape function . Those associated with the function value ψ (i.e. Ne_{10} and Ne_{20}) are found to be dimensionless, while the shape functions Ne_{11} and Ne_{21}, which correspond to the derivative $(\partial\psi/\partial x)$, have the dimensions of length. This feature is to be expected since the product of any shape function and the appropriate DOF should have the dimensions of the basic variable ψ.

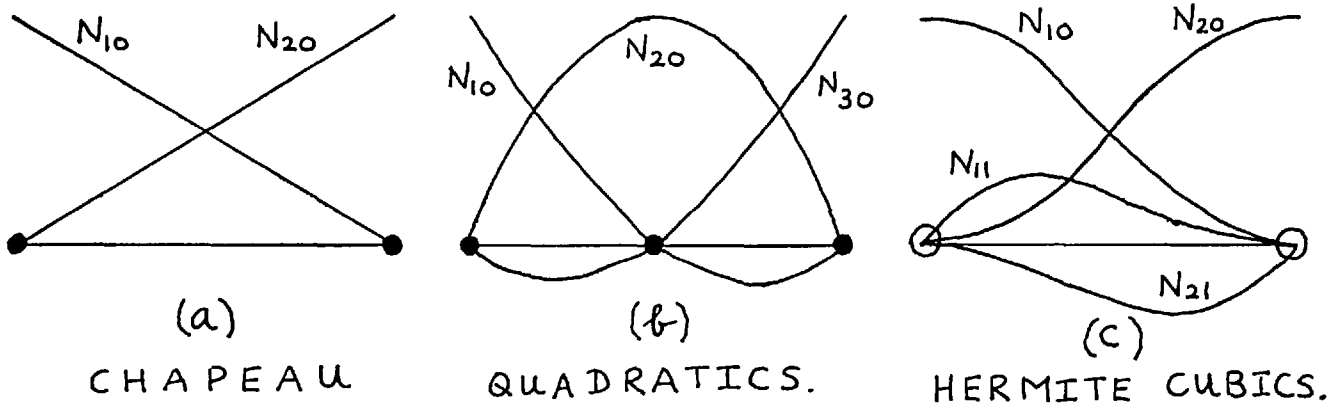


Figure 4.9 Illustration of 1D shape functions.

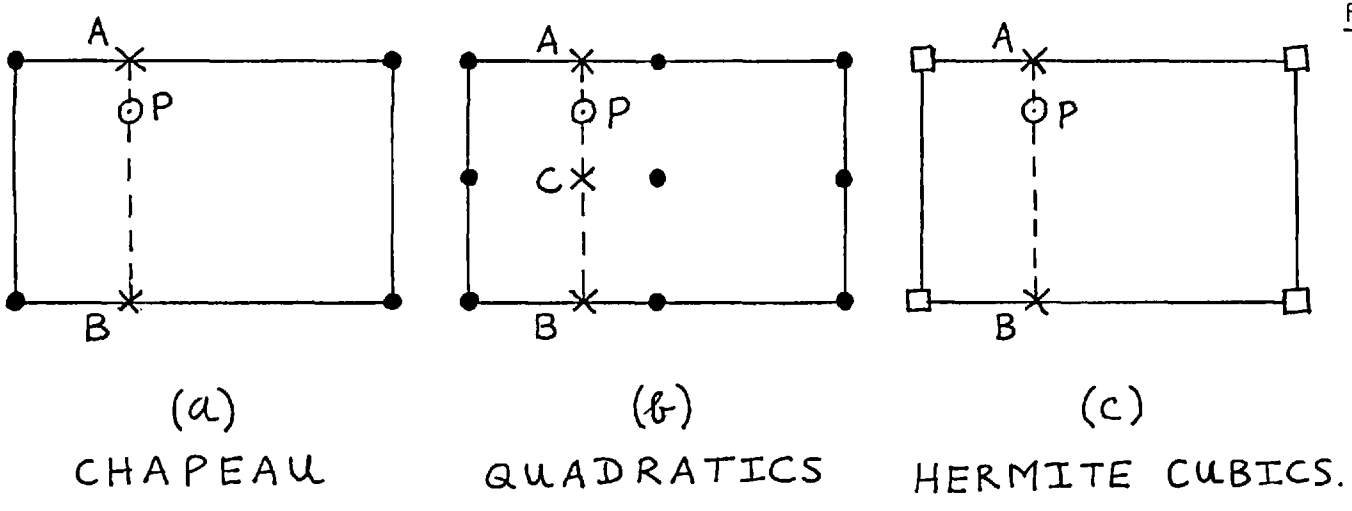


Figure 4.10 Interpolation (at P) within tensor product elements.

The linear, quadratic, and hermite cubic shape functions (as given by equations 4.25, 4.26 and 4.27) are illustrated in Figures 4.9a,b and c respectively.

4.6.4 2D Tensor product shape functions

To derive a general expression for the shape functions of tensor product rectangular elements, we shall adopt the following nomenclature, bearing in mind that the ξ and η directions coincide with the x and y directions respectively:- (i, j)= indices for nodes in the x and y directions; (n_i, n_j)= number of element nodes in the x and y directions; (l, m)= indices of differentiation w.r.t and x and y.

$$\psi_{ijlm} = DOF\left(\frac{\partial^{l+m}\psi}{\partial x^l \partial y^m}\right) \quad at \ node\ (i,j).$$

Ne_{ijlm} = shape function corresponding to ψ_{ijlm}; d = degree of continuity (C^d) of the polynomial. Interpolation of the function value ψ at internal points (such as P, in Figures 10a,b,c) can be carried out in two stages:

(a) along the n_j rows of nodes parallel to the x-axis, to determine the $n_j(d+1)$ "1D, y-directional DOF's" at points such as A,B,C etc.

(b) then, using the above DOF's, interpolate parallel to the y-axis, along the line AB, to obtain the function value ψ at P.

The first set of interpolations can be expressed mathematically as:

$$\left[\left\{\psi_{jm} = \sum_{i=1}^{n_i}\sum_{l=0}^{d} Ne_{il}(\xi)\,\psi_{ijlm}\right\} : \begin{matrix} m=0\rightarrow d; \\ j=1\rightarrow n_j. \end{matrix}\right] \qquad (4.28)$$

The second set, which represents summations along the line AB, can now be written as :

$$\psi(\xi,\eta) = \sum_{j=1}^{n_j} \sum_{m=o}^{d} Ne_{jm}(\eta)\, \psi_{jm} \tag{4.29}$$

which, on substituting equation (4.28), becomes :

$$\psi = \sum_{j=1}^{n_j} \sum_{i=1}^{n_i} \sum_{m=o}^{d} \sum_{\ell=o}^{d} \left[Ne_{i\ell}(\xi)\, Ne_{jm}(\eta) \right] \psi_{ij\ell m} . \tag{4.30}$$

From this equation it follows that:

$$Ne_{ijlm}(\xi,\eta) = Ne_{il}(\xi)\, Ne_{jm}(\eta) \tag{4.31}$$

which is the basic property of the tensor product space - namely, that the two-dimensional shape functions can be expressed as the products of the appropriate 1D shape functions. This feature is also found to be helpful in the evaluation of derivatives :

$$\left.\begin{aligned} \frac{\partial}{\partial x}(Ne_{ij\ell m}) &= \frac{1}{\Delta x}\left[\frac{dNe_{i\ell}(\xi)}{d\xi}\right] Ne_{jm}(\eta), \text{ and} \\ \frac{\partial}{\partial y}(Ne_{ij\ell m}) &= \frac{1}{\Delta y}\, Ne_{i\ell}(\xi)\left[\frac{dNe_{jm}(\eta)}{d\eta}\right] \end{aligned}\right\} . \tag{4.32}$$

These derivative terms are necessary for the calculation of pressure - and saturation gradients during finite element simulations. It is perhaps worth observing that the 1D shape functions are independent of the location of the element, since they are expressed in terms of the local co-ordinates ξ (or η). The only exceptions to this, however, are the shape functions associated with derivative DOF's

(e.g. Ne_{11} and Ne_{21} in equation 4.27) which depend on the lengths (Δx or Δy) of the element sides.

4.6.5 Global notation

The notation of the basic interpolation formula

$$\psi = \underline{Ne}\ \underline{\psi e}$$

(c.f. equation (4.13)), can be changed from an "element" level to a "global" scale, in the form:

$$\psi = \underline{N}\ \underline{\Psi} \tag{4.33}$$

where $\underline{\Psi}$ and $\underline{N}$ are the global vectors containing all the DOF's of the domain, and their associated shape functions, respectively. Since equation (4.33) is identical to equation (4.13), it follows that, at any point P of the domain, the only "active" terms in the array $\underline{N}$ will be the shape functions (Ne) corresponding to the element within which P lies. All the remaining shape functions in $\underline{N}$ will be zero at that point. We shall use this global representation scheme in the derivation of the finite element integral equations which will form the subject of the next two sections. Meanwhile, we note that our chief variables, p_o and Sw, can now be written as :

$$\left.\begin{aligned} p_o &= \underline{N}\ \underline{p_o} \\ \text{and} \quad S_w &= \underline{N}\ \underline{S_w} \end{aligned}\right\}. \tag{4.34}$$

4.7 Weighted residual methods

Having approximated the variables po and Sw by the finite element polynomials (equation 4.34), it now remains to set up a suitable criterion which will result in the best possible solution to

the governing partial differential equations 2.31 and 2.32. Basically the role of such a criterion is to generate a system of simultaneous difference equations, from which the unknown vectors $\underline{po}$ ans $\underline{Sw}$ can be determined.

Since a polynomial function cannot, in general, be an exact solution to a P.D.E, it is clear that the resulting residuals Ro and Rw (in equations 2.31 and 2.32) will be non-zero, at all but a finite number of points. The best solution, therefore, is one which attempts to minimise R, in some cumulative sense, over the whole domain. This is the basis for the class of methods known as "Weighted Residual Methods".

Let n_t be the total number of DOF's in the system. Each DOF ψ_i is assigned a weighting function $W_i(x,y)$, such that these W_i's form a set of n_t linearly independent functions. The weighted residual criterion requires that the product of the residual R with each of the weighting functions W_i, when integrated over the flow domain Ω, should be equal to zero. In other words, the residual is rendered orthogonal to the space of the weighting functions:

$$\int_{\Omega} W_i R \, d\Omega = 0 \; ; \quad i = 1 \longrightarrow n_t . \qquad (4.35)$$

Different types of weighted residual methods have been reported in the literature, of which the following are among the main branches, as summarised by Zienkiewicz [62b].

(1) Point collocation

In this method, the weighting functions are taken to be the Dirac Delta function, centred at the nodes where the DOF's operate. Thus:

$$W_i = \delta\left[(x-x_i)(y-y_i)\right] , \qquad (4.36)$$

and so

$$\int_{\Omega} W_i R \, d\Omega = R = 0 , \quad i = 1 \rightarrow n_t . \qquad (4.37)$$

Point collocation is therefore a criterion which makes the residual vanish at the nodal points.

(2) Sub-domain collocation

The W_i's, in this case, are set equal to unity over a sub-region surrounding the relevant node, and are zero everywhere else.

$$W_i = \begin{cases} 1 & \text{within the sub-region;} \\ 0 & \text{elsewhere;} \end{cases} \qquad (4.38)$$

and the integral relationship is

$$\int_{\Omega} W_i R \, d\Omega = \int_{\text{Sub-region}} R \, d\Omega = 0 \; ; \; i = 1 \rightarrow n_t , \qquad (4.39)$$

which shows that this method has a better "averaging" tendency than the "point collocation" method.

(3) Galerkin

This is perhaps one of the best known methods, in which the weighting functions are made identical to the shape functions Ni. Thus:

$$W_i = N_i , \quad i = 1 \rightarrow n_t ; \qquad (4.40)$$

so that the integral equation reads :

$$\int_{\Omega} N_i R \, d\Omega = 0 , \quad i = 1 \rightarrow n_t .$$

In vector notation, we can write this as:

$$\int_{\Omega} \underline{N}^T R \, d\Omega = \underline{0} \, . \qquad (4.41)$$

This Galerkin criterion will now be used to derive the sets of difference equations for p_o and Sw.

4.8 Generation of Difference equations (Galerkin)

Following Spivak et al [54], we shall employ the sequential solution procedure in which, during each time-step, the pressure vector po is calculated first, and then, using these new (updated) pressures, the saturations Sw are determined. In order to do this, however, the differential equations 2.31 and 2.32 have to be uncoupled, so that po becomes the predominant unknown of the equation. An inspection of this pair of equations reveals that po is associated mainly with the transmissibility term ($\underline{\nabla}.\underline{u}$), whereas Sw plays a major part in the capacity term ($\emptyset \partial s/\partial t$). The latter can be eliminated by adding these equations (2.31 and 2.32) to obtain:

$$R_t = \underline{\nabla} \cdot \underline{u}_t + \phi \left(S_o c_o + S_w c_w \right) \frac{\partial p_o}{\partial t} - \left(\sigma_o B_o + \sigma_w B_w \right) = 0 \qquad (4.42)$$

in which the subscript "t" denotes "total" (=oil + water). Although equation (4.42) still contains terms involving Sw (e.g. those involving compressibilities and relative permeabilities), the main capacity term is now absent. Consequently, we can treat the saturation terms explicitly (i.e. at the old time-level), and regard po as the main variable in this equation.

4.8.1 The pressure equation

Applying the Galerkin method to equation (4.42), we obtain:

$$\int_{\Omega} \underline{N}^T R_t \, d\Omega = \underline{0} \, . \tag{4.43}$$

The three terms involved will be considered individually.

(a) Transmissibility vector

$$\underline{t}_1 = \int_{\Omega} \underline{N}^T (\underline{\nabla} \cdot \underline{u}_t) \, d\Omega \tag{4.44}$$

Let us consider a typical shape function Ni. From elementary calculus, it is known that :

$$N_i (\underline{\nabla} \cdot \underline{u}_t) = \underline{\nabla} \cdot (N_i \, \underline{u}_t) - (\underline{\nabla} N_i)^T \underline{u}_t \, , \tag{4.45}$$

and hence

$$\int_{\Omega} N_i (\underline{\nabla} \cdot \underline{u}_t) \, d\Omega = \int_{\Omega} \underline{\nabla} \cdot (N_i \, \underline{u}_t) \, d\Omega - \int_{\Omega} (\underline{\nabla} N_i)^T \underline{u}_t \, d\Omega \, . \tag{4.46}$$

The first integral on the R.H.S may be transformed by the Divergence (or Gauss) theorem, which states that the integral of the divergence of a vector over a closed domain is equal to the integral of the normal component of the vector taken along the boundary. Thus :

$$\int_{\Omega} \underline{\nabla} \cdot (N_i \, \underline{u}_t) \, d\Omega = \int_{\partial\Omega} N_i \, \underline{n}^T \underline{u}_t \, d(\partial\Omega) \, , \tag{4.47}$$

where $\underline{n}$ is the unit vector along the outward normal. However, due to the "no flow" boundary condition (described in section 2.6), we know that:

$$\underline{n}^T \underline{u}_t = 0 \quad \text{at all points of } \partial\Omega .$$

The value of the integral in equation (4.47) is therefore zero, and equation (4.46) reduces to:

$$\int_{\Omega} N_i \, \underline{\nabla} \cdot \underline{u}_t \, d\Omega = - \int_{\Omega} (\underline{\nabla} N_i)^T \underline{u}_t \, d\Omega \qquad (4.48)$$

And so, collecting all such integrals ($i = 1 \rightarrow n_t$) into vector form, we obtain the simplified version of equation (4.44).

$$\underline{t}_1 = - \int_{\Omega} (\underline{\nabla} \underline{N})^T \underline{u}_t \, d\Omega \, . \qquad (4.49)$$

The value of $\underline{U}_t$ at the new time-level (n+1) is

$$\underline{u}_t^{n+1} = \underline{u}_t^{n} + \Delta(\underline{u}_t) \; ; \qquad (4.50)$$

where the Δ, during this stage, represents the increment with respect to po (since the saturations are held constant). Combining the transport equations 2.11 and 2.12, and ignoring gravity, we obtain:

$$\begin{aligned} \Delta(\underline{u}_t) &= - \lambda_{rt} \, \underline{k} \, \Delta(\underline{\nabla} p_o) \\ &= - \lambda_{rt} \, \underline{k} \, \underline{\nabla} (\Delta p_o) \\ &= - \lambda_{rt} \, \underline{k} (\underline{\nabla} \underline{N}) [\Delta \underline{p}_o] \, . \end{aligned} \qquad (4.51)$$

Finally, substituting equations (4.50) and (4.51) into (4.49), we obtain the complete expression for the transmissibility vector, at time level (n+1) :

$$\underline{t}_1^{n+1} = \left[\int_{\Omega} \lambda_{rt} (\underline{\nabla} \underline{N})^T \underline{k} (\underline{\nabla} \underline{N}) \, d\Omega \right] [\Delta \underline{p}_o] - \int_{\Omega} (\underline{\nabla} \underline{N})^T \underline{u}_t^n \, d\Omega \, . \qquad (4.52)$$

(b) Approximation of time derivative

The temporal gradient of pressure ($\partial po/\partial t$) which occurs in the second term of equation 4.42, is approximated by a finite difference formula, since finite elements, in this study, are used only for spatial

discretization. In order to relate this gradient to the pressures at the beginning and at the end of the time step, we make use of the Mean Value Theorem, which states that the chord-slope between any two points on a continuous and monotonically varying curve is a weighted mean of the two "end slopes" (Figure 4.11). Thus:

$$\frac{p_o^{n+1} - p_o^n}{\Delta t} = (1-\theta)\left(\frac{\partial p_o}{\partial t}\right)^n + \theta\left(\frac{\partial p_o}{\partial t}\right)^{n+1}; \qquad (4.53)$$

where θ is the weighting factor. Depending on the value assigned to θ, three familiar schemes can be recognised :

(1) $\theta = 0$ Forward difference (Explicit)

(2) $\theta = \frac{1}{2}$ Central difference (Crank-Nicholson)

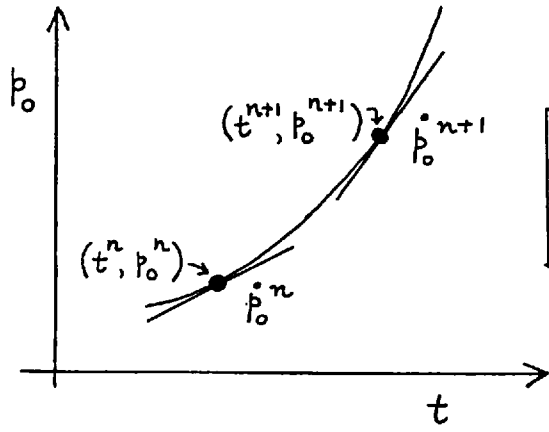
(3) $\theta = 1$ Backward difference (Implicit)

Of these, the explicit scheme is unstable. The central difference formula is the most accurate scheme, although it is known to exhibit some minor temporal oscillations. The Backward difference scheme is stable and smooth, but is thought to be slightly less accurate than the Central difference formula. In the present study, the Backward difference scheme has been used throughout to avoid time oscillations.

So, putting $\theta = 1$ in equation (4.53), we find:

$$\left(\frac{\partial p_o}{\partial t}\right)^{n+1} = \frac{p_o^{n+1} - p_o^n}{\Delta t} = \frac{1}{\Delta t}\,\underline{N}\left[\Delta \underline{p}_o\right] \qquad (4.54)$$

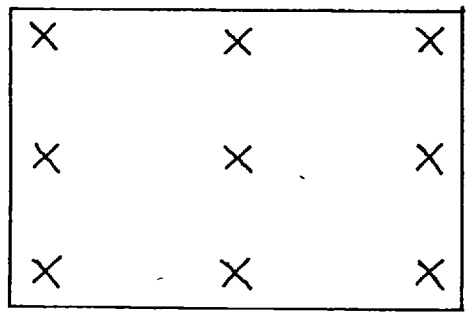
Substituting this into equation 4.42, and performing the Galerkin integration (4.43), we obtain :



$$\frac{p_o^{n+1} - p_o^n}{t^{n+1} - t^n} = (1-\theta)\,\dot{p}_o^n + \theta\,\dot{p}_o^{n+1}$$

Figure 4.11 Approximation of the chord slope by the Mean Value Theorem.

Figure 4.12 Illustration of (3×3) Gausspoint quadrature for a rectangle.



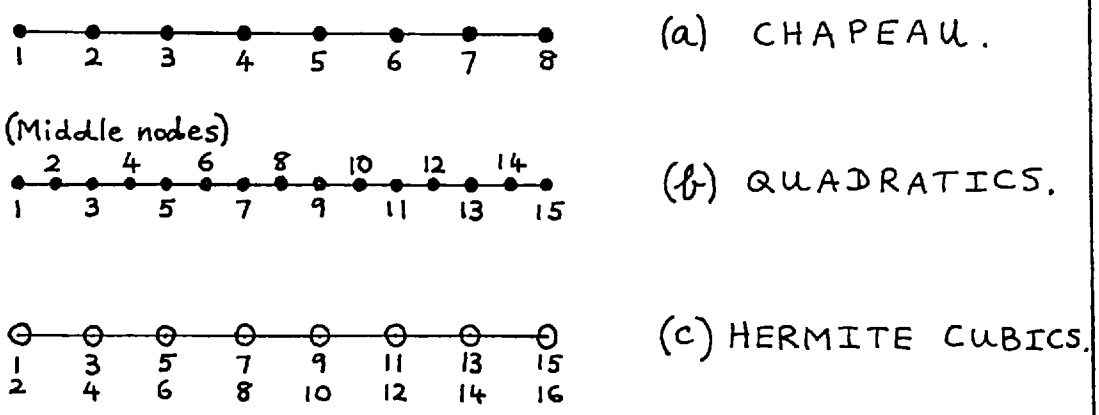


Figure 4.14 Numbering of nodes and/or DOF's in a 1D system consisting of 7 elements.

$$\underline{t}_2^{n+1} = \int_\Omega \underline{N}^T \phi \left(S_o^n c_o + S_w^n c_w \right) \left(\frac{\partial p}{\partial t} \right)^{n+1} d\Omega$$

$$= \left[\int_\Omega \frac{\phi c_t^n}{\Delta t} \underline{N}^T \underline{N} \, d\Omega \right] \left[\Delta \underline{p}_o \right] \qquad (4.55)$$

where C_t^n is the total fluid compressibility at t^n.

(c) Source/sink vector

For the rate terms, the Galerkin procedure yields:

$$\underline{t}_3 = - \int_\Omega \underline{N}^T \left(\sigma_o B_o + \sigma_w B_w \right) d\Omega$$

which , on introducing the point source/sink concept, reduces (as shown in equation 2.35) to:

$$\underline{t}_3^{n+1} = - \left(\underline{q}_o^{n+1} + \underline{q}_w^{n+1} \right) = - \underline{q}_t^{n+1} , \qquad (4.56)$$

where $\underline{qo}$, $\underline{qw}$ and $\underline{qt}$ are the global vectors containing the oil, water and total rates, respectively, expressed at reservoir conditions.

Having thus expanded the various integral terms, we may now write the Galerkin equation 4-43, at time level (n+1), in the form:

$$\underline{t}_1^{n+1} + \underline{t}_2^{n+1} + \underline{t}_3^{n+1} = \underline{0}$$

which, on combining equations 4.52, 4.55 and 4.56, yields the final set of difference equations for the pressure increments $[\Delta \underline{po}]$:

$$\left[\left\{ \int_\Omega \lambda_{rt} \left(\underline{\nabla N} \right)^T \underline{k} \left(\underline{\nabla N} \right) d\Omega \right\} + \left\{ \int_\Omega \frac{\phi c_t}{\Delta t} \underline{N}^T \underline{N} \, d\Omega \right\} \right] \left[\Delta \underline{p}_o \right] \qquad (4.57)$$

$$= \underline{q}_t^{n+1} + \int_\Omega \left(\underline{\nabla N} \right)^T \underline{u}_t^n \, d\Omega \, .$$

This is a matrix equation of the form:

$$\left[\underline{T} + \underline{\varepsilon}_1\right]\left[\Delta \underline{p}_o\right] = \underline{r}_1 , \tag{4.58}$$

where the matrix

$$\underline{T} = \int_{\Omega} \lambda_{rt} (\underline{\nabla N})^T \underline{k} (\underline{\nabla N}) \, d\Omega \tag{4.59}$$

(which we shall call the total transmissibility matrix) plays the important role of determining the pressure distribution during two-phase flow. If the fluids are assumed to be incompressible, then the matrix $\underline{\varepsilon}_1$ disappears, and equation (4.58) simplifies to:

$$\underline{T}\left[\Delta \underline{p}_o\right] = \underline{r}_1 . \tag{4.60}$$

4.8.2 The Saturation equation

The difference equations for the saturation vector $[\Delta \underline{Sw}]$ are obtained by applying the Galerkin procedure to the continuity equation for water (2.32). The steps involved are very similar to those described in the previous section. Since saturation is now the fundamental unknown, all incremental terms (Δ) are evaluated w.r.t.Sw, while keeping the pressure unchanged.

At the new time-level (n+1), the individual vectors which make up the Galerkin equation are as follows:

(1) Water transmissibility vector

$$\underline{t}_1^{n+1} = \int_{\Omega} \underline{N}^T \left(\underline{\nabla} \cdot \underline{u}_w^{n+1}\right) d\Omega = -\int_{\Omega} (\underline{\nabla N})^T \left[\underline{u}_w^n + \Delta \underline{u}_w\right] d\Omega \tag{4.61}$$

Using the transport equation for water (2.12), we find

$$\Delta(\underline{u}_w) = -\Delta(\lambda_{rw})\underline{K}\underline{\nabla}p_o + \underline{K}\,\Delta\left[\lambda_{rw}p_c'\underline{\nabla}S_w\right]$$

$$= \underline{K}\left[-\lambda_{rw}'\underline{\nabla}p_o + \left(\lambda_{rw}p_c'' + \lambda_{rw}'p_c'\right)\underline{\nabla}S_w\right]\Delta S_w \qquad (4.62)$$

$$+ \lambda_{rw}p_c'\underline{K}\,\Delta(\underline{\nabla}S_w)\,.$$

Substituting this into equation (4.61), we obtain:

$$\underline{t}_1^{n+1} = \left[\left\{\int_\Omega (\underline{\nabla}\underline{N})^T\underline{K}\left(\lambda_{rw}'\underline{\nabla}p_o - \left(\lambda_{rw}p_c'' + \lambda_{rw}'p_c'\right)\underline{\nabla}S_w\right)\underline{N}\,d\Omega\right\}\right.$$

$$\left. - \left\{\int_\Omega \lambda_{rw}p_c'(\underline{\nabla}\underline{N})^T\underline{K}(\underline{\nabla}\underline{N})\,d\Omega\right\}\right]\left[\Delta\underline{S}_w\right] - \int_\Omega (\underline{\nabla}\underline{N})^T\underline{u}_w^n\,d\Omega\,, \qquad (4.63)$$

where ' and " denote the first and second derivatives respectively, w.r.t. Sw.

(2) Capacity vector

$$\underline{t}_2^{n+1} = \int_\Omega \underline{N}^T\phi\,\frac{\partial S_w}{\partial t}\,d\Omega = \left[\int_\Omega \frac{\phi}{\Delta t}\,\underline{N}^T\underline{N}\,d\Omega\right]\left[\Delta\underline{S}_w\right]. \qquad (4.64)$$

(3) Compressibility vector

$$\underline{t}_3^{n+1} = \int_\Omega \underline{N}^T\phi S_w c_w\frac{\partial p_o}{\partial t}\,d\Omega$$

$$= \left[\int_\Omega \frac{\phi c_w}{\Delta t}\left(p_o^{n+1} - p_o^n\right)\underline{N}^T\underline{N}\,d\Omega\right]\left[\underline{S}_w^n + \Delta\underline{S}_w\right] \qquad (4.65)$$

(4) Source/sink vector

$$\underline{t}_4^{n+1} = -\int_\Omega \underline{N}^T(\sigma_w B_w)^{n+1}\,d\Omega = -\underline{q}_w^{n+1}\,. \qquad (4.66)$$

Finally, the Galerkin criterion requires that:

$$\int_{\Omega} \underline{N}^T R_w^{n+1} \, d\Omega = \underline{t}_1^{n+1} + \underline{t}_2^{n+1} + \underline{t}_3^{n+1} + \underline{t}_4^{n+1} = \underline{0}$$

and hence, combining equations 4.63 through 4.66, we obtain the set of difference equations for $[\Delta \underline{S}_w]$:

$$\left[\left\{\int_{\Omega} \frac{\phi}{\Delta t} \underline{N}^T \underline{N} \, d\Omega\right\} - \left\{\int_{\Omega} \lambda_{rw} P_c' (\underline{\nabla N})^T \underline{K} (\underline{\nabla N}) \, d\Omega\right\} + \left\{\int_{\Omega} \frac{\phi c_w}{\Delta t} (P_o^{n+1} - P_o^n) \underline{N}^T \underline{N} \, d\Omega\right\}\right.$$

$$\left. + \left\{\int_{\Omega} (\underline{\nabla N})^T \underline{K} \left(\lambda_{rw}' \underline{\nabla} P_o - (\lambda_{rw} P_c'' + \lambda_{rw}' P_c') \underline{\nabla} S_w\right) \underline{N} \, d\Omega\right\}\right] [\Delta \underline{S}_w]$$

$$= \left[\underline{q}_w + \int_{\Omega} (\underline{\nabla N})^T \underline{u}_w^n \, d\Omega\right] - \left[\int_{\Omega} \frac{\phi c_w}{\Delta t} (P_o^{n+1} - P_o^n) \underline{N}^T \underline{N} \, d\Omega\right] [\underline{S}_w^n], \quad (4.67)$$

which basically has the following structure:

$$\left[\underline{C} - \underline{\varepsilon}_1 + \underline{\varepsilon}_2 + \underline{\varepsilon}_3\right] [\Delta \underline{S}_w] = \underline{r}_2 - \underline{\varepsilon}_2 [\underline{S}_w^n] \quad (4.68)$$

where the matrix :

$$\underline{C} = \int_{\Omega} \frac{\phi}{\Delta t} \underline{N}^T \underline{N} \, d\Omega \quad (4.69)$$

is known as the capacity matrix, and is as important to the saturation distribution as the transmissibility matrix $\underline{T}$ (equation 4.59) is to the pressure distribution. Of the remaining minor matrices, $\underline{\varepsilon}_1$ is associated with the capillary pressure gradient; $\underline{\varepsilon}_2$ involves the compressibility; and $\underline{\varepsilon}_3$ is the "chord-slope" matrix which makes the equation implicit in Sw. However, $\underline{\varepsilon}_3$ is also a non-symmetric matrix, and, in all the runs

described in the present study, it has been ignored, and the equation treated explicitly in Sw. [To compensate for this restriction, the time-steps were prevented from becoming too large]. In addition to this, if the fluids are incompressible, then equation (4.68) simplifies to:

$$\left[\underline{C} - \underline{\varepsilon_1}\right]\left[\Delta \underline{S_w}\right] = \underline{r_2} \,. \qquad (4.70)$$

4.8.3 Integration and assembly of matrices

The pressure equation (4.60) and the saturation equation (4.70) are both of the form:

$$\underline{A}\,\underline{x} = \underline{b} \,,$$

where the global matrix $\underline{A}$ and the R.H.S. vector $\underline{b}$ involve integrations, over the whole domain, of matrices (and vectors) associated with shape functions and reservoir/fluid parameters. Since the shape functions provide at least C^0 continuity, and since derivatives higher than first order do not appear in any of the Galerkin equations, it may be concluded that the discontinuities in the integrands across inter-element boundaries are finite, and do not contribute to the final integral. The global integrals can therefore be expressed as the sum of integrals carried out over each element :

$$\int_{\Omega} f \, d\Omega = \sum_{elems} \int_{\Omega_e} f \, d\Omega_e \,. \qquad (4.71)$$

Thus, in the finite element method, the matrices involved are calculated at the element level, and added into the appropriate locations within the global matrix. This twin process of "generation" and "assembly" is repeated until all the elements of the domain have been traversed.

Within each element, the required integrations can be performed either analytically or numerically. The analytical approach is usually restricted to cases where the integrand does not involve saturation terms, and where the shape functions are not too complicated [e.g. the capacity matrices]. Most of the matrices, however, cannot be tackled in this way due to the occurrence of relative permeabilities and similar terms, and hence the common practice in finite element programs is to employ numerical integration schemes. These are formulae which approximate the integral by a weighted sum of the integrand, evaluated at selected points within the range of integration. Thus :

$$\int_0^1 f(\xi)\, d\xi \doteq \sum_{i=1}^{n} W_i f(\xi_i) . \qquad (4.72a)$$

The Gauss-Legendre quadrature, for example, is well suited to rectangular finite elements, and the use of "n" integration points (also called Gauss points) is sufficient to integrate exactly a 1D polynomial of degree (2n-1). These integration points (ξi) and the corresponding weighting factors (Wi) can be found in standard numerical tables.

Applying equation (4.72a) to a 2D rectangular element, we obtain :

$$\int_{\Omega_e} f\, d\Omega_e = h\,\Delta x\,\Delta y \int_0^1\int_0^1 f(\xi,\eta)\, d\xi\, d\eta$$

$$\doteq h\,\Delta x\,\Delta y \sum_{i=1}^{n}\sum_{j=1}^{n} W_i W_j f(\xi_i,\eta_j), \qquad (4.72b)$$

where h, Δx, Δy are the dimensions of the element.

Figure 4.12 illustrates a (3 x 3) Gauss-point scheme.

4.8.4 Solution of matrix equations

Equations (4.60) and (4.70) are solved sequentially, during each time-step, to obtain the pressure and saturation vectors respectively. The steps involved can be summarised as follows:-

Suppose that, at time level n, we know $\underline{po}^n$ and $\underline{Sw}^n$. It is required now to determine $\underline{po}^{n+1}$ and $\underline{Sw}^{n+1}$.

1 Pressure solution

Assemble: $\underline{T}(\underline{S}_w^n)$ and $\underline{r}_1(\underline{P}_o^n, \underline{S}_w^n)$. (see equation 4.60)

Solve: $$\Delta \underline{P}_o = \left[\underline{T}(\underline{S}_w^n)\right]^{-1} \underline{r}_1(\underline{P}_o^n, \underline{S}_w^n). \qquad (4.73)$$

Increment: $$\underline{P}_o^{n+1} = \underline{P}_o^n + \Delta \underline{P}_o \,. \qquad (4.74)$$

These new pressures $\underline{po}^{n+1}$ are now used for setting up the matrices for the saturation equation.

2. Saturation Solution

Assemble:

$$\underline{C}\,, \quad \underline{\varepsilon}_1(\underline{S}_w^n) \text{ and } \underline{r}_2(\underline{P}_o^{n+1}, \underline{S}_w^n) \text{ [see eq. (4.70)]}$$

Solve:

$$\Delta \underline{S}_w = \left[\underline{C} - \underline{\varepsilon}_1(\underline{S}_w^n)\right]^{-1} \left[\underline{r}_2(\underline{P}_o^{n+1}, \underline{S}_w^n)\right] \qquad (4.75)$$

Increment:

$$\underline{S}_w^{n+1} = \underline{S}_w^n + \Delta \underline{S}_w \,. \qquad (4.76)$$

Having thus calculated $\underline{po}^{n+1}$ and $\underline{Sw}^{n+1}$, this procedure is repeated for all the subsequent time-steps.

4.9 Implementation of Boundary conditions

Since, in the present study, the sides of the quarter five-spot domain are parallel to the coordinate axes, it follows that the normal gradients on the boundaries simply reduce to either $(\partial/\partial x)$ or $(\partial/\partial y)$. Let us denote by B.C.1 and B.C.2 the boundary conditions which prevail along the sides parallel to the x and y axes, respectively.

B.C.1

From equation (2.37) we know that the "no flow" boundary condition requires the normal pressure gradient to be zero on the boundary.

Hence $$\frac{\partial p_o}{\partial y} = 0 . \qquad (4.77)$$

However, since this condition holds at every point along the line, we also have:

$$\frac{\partial}{\partial x}\left[\frac{\partial p_o}{\partial y}\right] = \frac{\partial^2 p_o}{\partial x \partial y} = 0 . \qquad (4.78)$$

Equation (2.38) imposes similar constraints on the saturation, and so, combining all of these, we can express B.C.1 in the form :

$$\frac{\partial p_o}{\partial y} = \frac{\partial^2 p_o}{\partial x \partial y} = \frac{\partial S_w}{\partial y} = \frac{\partial^2 S_w}{\partial x \partial y} = 0 . \qquad (4.79)$$

Likewise, B.C.2 is given by

$$\frac{\partial p_o}{\partial x} = \frac{\partial^2 p_o}{\partial x \partial y} = \frac{\partial S_w}{\partial x} = \frac{\partial^2 S_w}{\partial x \partial y} = 0. \qquad (4.80)$$

It may be stated that although B.C.1 and B.C.2 represent "no flow" conditions, they may equally well have been deduced from considerations of symmetry and smoothness across boundaries.

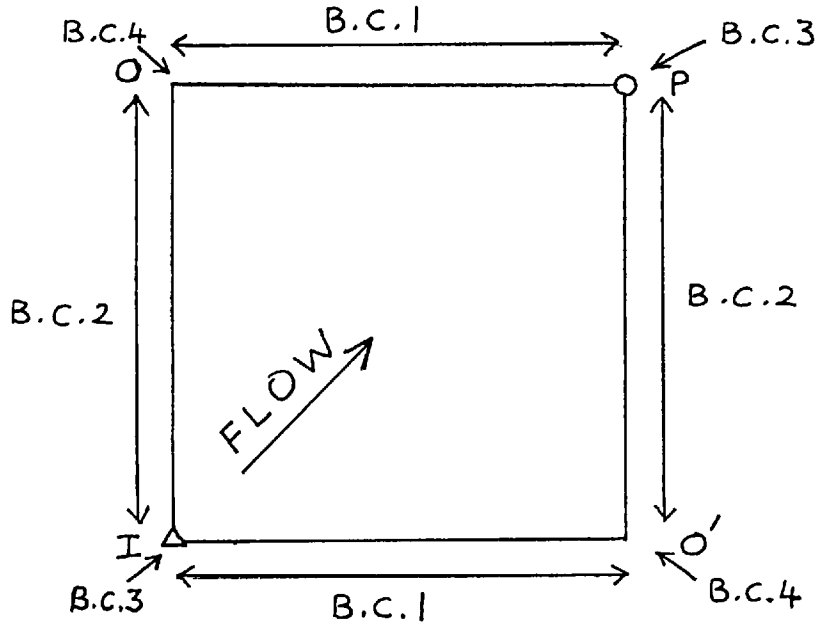
In the Chapeau and Quadratics models, these boundary conditions are implicitly taken care of by the Galerkin formulation [c.f. equations 4.47, 4.48], and hence do not have to be introduced externally. In the Hermite cubics model, however, it was found, after considerable numerical experimentation, that due to the occurrence of these derivatives as degrees of freedom, they must be forced to be zero, as otherwise large oscillations could develop. The enforcement of such a condition means that, for each boundary DOF (ψi) concerned, the difference equation generated by the Galerkin process should be replaced by the identity $\psi \equiv 0$. Following Zienkiewicz [62c] this can be done quite conveniently by adding a very large number (e.g. 10^{98}) to the diagonal term of the relevant equation.

B.C.3

The boundary conditions B.C.1 and B.C.2 are enforced at all the boundary nodes except those where wells are located. At such points, the constraints on the pressure gradients are removed, and hence the boundary condition used at the wells (B.C.3) is simply:

$$\frac{\partial S_w}{\partial x} = \frac{\partial S_w}{\partial y} = \frac{\partial^2 S_w}{\partial x \partial y} = 0. \qquad (4.81)$$

It is possible that even this restriction may be removed by the use of



B.C.1

$$\frac{\partial p_o}{\partial y} = \frac{\partial^2 p_o}{\partial x \partial y} = \frac{\partial S_w}{\partial y} = \frac{\partial^2 S_w}{\partial x \partial y} = 0.$$

B.C.2

$$\frac{\partial p_o}{\partial x} = \frac{\partial^2 p_o}{\partial x \partial y} = \frac{\partial S_w}{\partial x} = \frac{\partial^2 S_w}{\partial x \partial y} = 0.$$

B.C.3

$$\frac{\partial S_w}{\partial x} = \frac{\partial^2 S_w}{\partial x \partial y} = \frac{\partial S_w}{\partial y} = 0.$$

B.C.4

$$p_o = p_i \ (= \text{constant.})$$

Figure 4.13 Boundary conditions used in the simulation of incompressible, two phase flow in a five-spot. (B.C.1, B.C.2 and B.C.3 apply to Hermite Cubics only).

"singular" basis functions at the wells.

B.C.4

If the fluids are assumed to be incompressible (as in the five-spot studies reported here), then it is also necessary to specify a "fixed pressure" boundary condition, since the transmissibility matrix ($\underline{T}$) would otherwise become singular. In the case of the five-spot, one obvious way to do this is to hold the pressure constant (at its initial value) at the two extremities of the median line:

$$p_o = p_i \text{ for all } t, \text{ at points } O \text{ and } O'. \quad (4.82)$$

The various boundary conditions described above are summarised in Figure 4.13. We shall make three comments regarding the use of these constraints:

(1) All four boundary conditions were applied to the Hermite Cubics model, while the Chapeau and the Quadratics models were subject to B.C.4 only.

(2) The combination given here is what has been developed from trial-and-error experiments.

(3) No boundary conditions were applied to the 1D models (where fluid compressibility was taken into account).

4.10 Solution algorithms for the difference equations

In finite element programs it is customary to use "Direct solution techniques" (i.e. Gaussian elimination, or variations thereof) to solve the system of equations. Iterative methods, on the other hand, are of limited scope in this field, largely due to the variability of the

matrix structure. For this reason, some of the popular finite difference algorithms such as Alternating Direction Implicit Procedure (ADIP) and Strongly Implicit Procedure (SIP) are difficult to use in finite element models.

The two most important factors which determine the feasibility of any solution method are:

(a) storage, and

(b) computer time.

Unlike finite difference models, the storage requirement poses a severe problem even in small finite element programs. This is attributable partly to the use of Direct solution methods, and also to the fact that, due to the presence of a large number of DOF's in each element, and the strong interaction between them, the bandwidth of the overall (global) matrix tends to be quite large. This leads to a situation where even relatively small finite element meshes place unusually high demands on the computer's central memory, and the entire matrix cannot normally be accommodated in core. [For example, the semi-bandwidth of a (3x3) element grid is 3 for finite differences, and 23 for Hermite cubic finite elements]. To overcome the storage problem two methods are commonly in use:

(1) Banded solution algorithm

(2) Frontal solution algorithm [Irons [32]].

4.10.1 Comparison of "Banded" and "Frontal" methods

One feature that is common to both methods is that, instead of assembling the global matrix and then solving it (a procedure which

would almost be impossible from the point of view of storage), they combine the processes of assembly and elimination so that, at any given stage of the operation, the size of the "active" matrix required for holding the coefficients is considerably smaller than that of the global matrix. Moreover, the upper-triangular entries created during the elimination phases are written successively onto a disk file. Finally, when all the elements of the domain have been traversed, and the joint process of "assembly" and "elimination" completed, then the algorithm enters the "Back-substitution" phase, and calculates the solution vector by reading off the upper triangular records (in reverse order) from the disk file.

The banded solution method is strongly dependent on the ordering scheme used for the DOF's, since this directly influences the variation of the bandwidth. A node-labelling algorithm (with, preferably, the maximum optimising capability) is therefore necessary for use in conjunction with the Banded solution method.

The Frontal approach, on the other hand, is entirely independent of the nodal sequence, and, in fact, the nodes are given fictitious numbers (aptly termed "nicknames"). The key to its success lies in a preliminary subroutine (known as the "prefrontal routine") which controls the storage requirements by means of a sophisticated housekeeping scheme for the matrix. In particular, it scans the original mesh, element by element, and performs the following tasks at each stage:

(1) seeks out the "inactive" DOF's for elimination;

(2) records the (empty) rows and columns vacated by the outgoing DOF's.

(3) Admits new DOF's as they come in, either by inserting them into these empty locations (if they are available), or else, by

augmenting the "active" matrix to create more space.

The principal advantage which the "frontal" method possesses over the "Banded" method is that it enables changes to the original mesh, such as insertion or deletion of nodes, to be performed without any difficulty.

In the present study, the finite element models have been equipped with Banded matrix algorithms because for the simple geometries studied they are likely to perform faster. These will be described below:

4.10.2 Banded algorithms for 1D finite elements

Although the storage requirement for any 1D model is so small that the entire matrix can be held in central memory, the algorithms developed here are nevertheless based on an alternation of the "assembly" and "elimination" procedures, as is characteristic of the banded solution method.

(a) Algorithm for 1D Chapeau model

Figure (4.14a) shows a 1D system consisting of 7 Chapeau elements, and Figure (4.15a) shows the resulting tri-diagonal global matrix. The normal procedure, therefore, would be to assemble the whole matrix first, and then to solve it by the Thomas Algorithm. By contrast, the banded solution technique proceeds as follows:-

{ Assemble matrix for element 1 ;
{ Eliminate node 1 ;
{ Assemble matrix for element 2 ;
{ Eliminate node 2 ;
etc.

It can thus be seen that, while the global matrix contains (8x3) entries, the "active" matrix of the banded method contains only (2x2). Let $\underline{A}$ (2x2) and $\underline{b}$ (2x1) denote the active matrix and the active R.H.S vector respectively. Suppose, also, that after the assembly of the i^{th} element (which involves nodes i and i+i), the structure of these matrices are as follows:

(1) $$\underline{A} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \qquad \underline{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{4.83}$$

(2) Converting to upper-triangular form (such that the diagonal term is unity), we obtain:

$$\underline{A}' = \begin{bmatrix} 1 & a_1^{-1}a_2 \\ 0 & (a_4 - a_3 a_1^{-1} a_2) \end{bmatrix} \quad \underline{b}' = \begin{bmatrix} a_1^{-1} b_1 \\ (b_2 - a_3 a_1^{-1} b_1) \end{bmatrix} \tag{4.84}$$

(3) The coefficients of the top row can be stored in the form:

$$c_i = a_1^{-1} a_2 \quad \text{and} \quad d_i = a_1^{-1} b_1 \tag{4.85}$$

for use, <u>later</u>, during the back-substitution phase:

$$\gamma_i = d_i - c_i \gamma_{i+1} \tag{4.86}$$

(4) The bottom row is now shifted <u>diagonally upwards</u> to replace the top row, and the remaining entries are all set to zero. The vector $\underline{b}'$ is treated likewise:

$$\underline{A}'' = \begin{bmatrix} (a_4 - a_3 a_1^{-1} a_2) & 0 \\ 0 & 0 \end{bmatrix} \quad \underline{b}'' = \begin{bmatrix} (b_2 - a_3 a_1^{-1} b_1) \\ 0 \end{bmatrix} \tag{4.87}$$

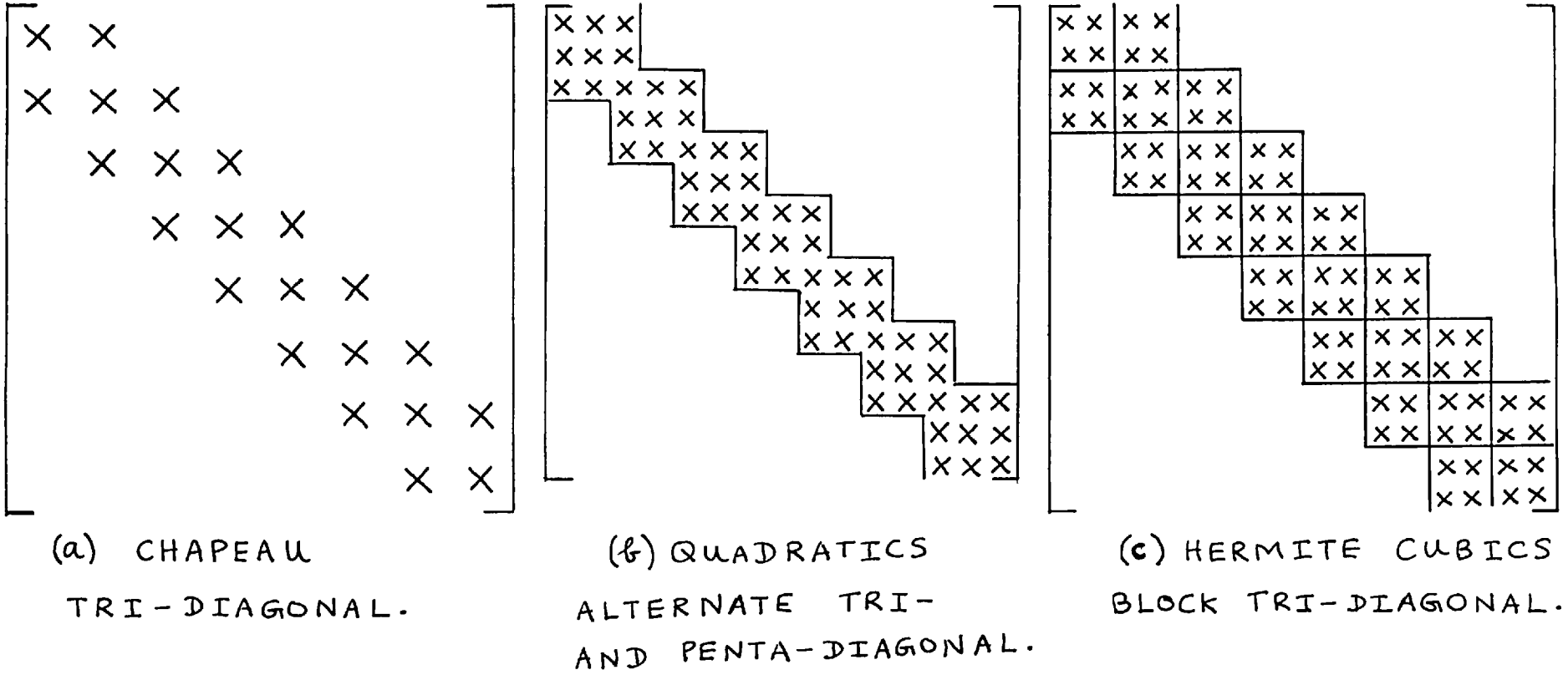


Figure 4.15 Matrix structures corresponding to the 1D systems given in Figure 4.14.

(5) This is followed by the assembly of the next element (i+1), and the resulting "element matrices" are added on to $\underline{A}$" and $\underline{b}$" to generate the new matrices $\underline{A}$ and $\underline{b}$. At this point the program returns to step 1, and repeats the complete cycle (1 through 5), until all the elements have been covered.

(6) Finally, the back-substitution (4.86) is performed.

(b) Algorithm for 1D Quadratics model

In the quadratics model, the presence of internal nodes (Figure 4.14b) introduces a slight variation into an otherwise simple matrix structure, and, as illustrated in Figure 4.15b, the "stepped" profile of the bandwidth gives rise to an alternating tri-and penta-diagonal pattern. Since each element has 3 nodes, the active matrix $\underline{A}$ now has (3x3) entries.

As before, let us consider the state of the matrices after the assembly of a typical element i, which involves the nodes (2i-1), 2i, (2i+1).

(1) $$\underline{A} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{bmatrix} \qquad \text{and } \underline{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} . \qquad (4.88)$$

Since the elimination of nodes is done two at a time, it is convenient to partition the above matrices so as to separate this pair of outgoing nodes from the third node (which is still active, and linked to the next element). Thus:

$$\underline{A} = \left[\begin{array}{cc|c} & & a_3 \\ \multicolumn{2}{c|}{\underline{a}_1} & a_6 \\ \hline a_7 & a_8 & a_9 \end{array}\right] \qquad \text{and } \underline{b} = \left[\begin{array}{c} \underline{b}_1 \\ \hline b_3 \end{array}\right] \qquad (4.89)$$

where $\underline{a}_1$ and $\underline{b}_1$ are submatrices of order (2x2) and (2x1) respectively.

(2) Conversion of $\underline{A}$ into upper triangular form yields:

$$\underline{A}' = \left[\begin{array}{cc|c} \multicolumn{2}{c|}{\underline{I}} & \underline{a}_1^{-1}\begin{bmatrix} a_3 & a_6 \end{bmatrix}^T \\ \hline 0 & 0 & a_9 - \begin{bmatrix} a_7 & a_8 \end{bmatrix}\underline{a}_1^{-1}\begin{bmatrix} a_3 & a_6 \end{bmatrix}^T \end{array}\right] \quad \text{and}$$

$$\underline{b}' = \left[\begin{array}{c} \underline{a}_1^{-1}\underline{b}_1 \\ \hline b_3 - \begin{bmatrix} a_7 & a_8 \end{bmatrix}\underline{a}_1^{-1}\underline{b}_1 \end{array}\right] \qquad (4.90)$$

(3) The coefficients for back-substitution are extracted from the top two rows of $\underline{A}'$ and $\underline{b}'$, and stored as :

$$\underline{c}_i = \underline{a}_1^{-1}\begin{bmatrix} a_3 & a_6 \end{bmatrix}^T \quad \text{and} \quad \underline{d}_i = \underline{a}_1^{-1}\underline{b}_1 . \qquad (4.91)$$

(4) Shifting the third row diagonally upwards, and setting all other terms to zero, we have:

$$\underline{A}'' = \begin{bmatrix} a_9 - \begin{bmatrix} a_7 & a_8 \end{bmatrix}\underline{a}_1^{-1}\begin{bmatrix} a_3 & a_6 \end{bmatrix}^T & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and}$$

$$\underline{b}'' = \begin{bmatrix} b_3 - \begin{bmatrix} a_7 & a_8 \end{bmatrix}\underline{a}_1^{-1}\underline{b}_1 \\ 0 \\ 0 \end{bmatrix} \qquad (4.92)$$

(5) The matrices for the next element (i+1) are now assembled, and superimposed on $\underline{A}''$ and $\underline{b}''$. Steps (1) through (5) are repeated for all elements.

(6) The final solution is obtained by back-substitution:

$$\underline{\psi}_i = \begin{bmatrix} \psi_{2i-1} \\ \psi_{2i} \end{bmatrix} = \underline{d}_i - \psi_{2i+1}\,\underline{c}_i \qquad (4.93)$$

(c) Algorithm for 1D Hermite cubics model

Due to the presence of two DOF's at each node, instead of one (Figure 4.14c), the global matrix for the 1D Hermite cubics model assumes a block tri-diagonal form, in which each block is a (2x2) matrix (Figure 4.15c). The active matrix $\underline{A}$, therefore, consists of four (2x2) matrices, while the active R.H.S. vector $\underline{b}$ contains two (2x1) vectors.

$$\underline{A} = \left[\begin{array}{c|c} \underline{a}_1 & \underline{a}_2 \\ \hline \underline{a}_3 & \underline{a}_4 \end{array}\right] \quad \text{and} \quad \underline{b} = \left[\begin{array}{c} \underline{b}_1 \\ \hline \underline{b}_2 \end{array}\right]. \qquad (4.94)$$

The steps of the solution algorithm are thus identical to those derived for the Chapeau model, with the sole exception that the scalars a_1, b_1 etc., are now replaced by the (2x2) matrices $\underline{a}_1$, $\underline{b}_1$ etc., and hence the algebraic operations are all matrix operations.

4.10.3 Algorithm for 2D models

On account of the large size and variability of the bandwidth of 2D matrices, specific algorithms, such as were developed for the 1D models, are of little use here. Consequently, in the present study, all three finite element types (Chapeau, Quadratics and H.cubics) are supported by a single solution-routine (known as GAUSS). Separate subroutines are available, however, to provide the necessary interface between this solution algorithm and the main program. In particular, they perform the following tasks:

(a) ordering of the DOF's ; and

(b) analysis of the bandwidth.

For convenience, we shall describe these aspects with reference to the global matrix (assuming that it is small enough to reside in core). Having done this, we shall then return to the concept of the(smaller) "active" matrix, and describe the application of the banded solution technique to the 2D problem.

(A) Ordering scheme for DOF's

A standard row-by-row ordering scheme is employed for all three elements, and, where multi-DOF nodes occur,the DOF's of each node are included sequentially before proceeding to the adjacent node. Figures 4.16a. b and c show the numbering sequences used for a (3x3) mesh consisting of the tensor product Chapeau,Quadratics and H.Cubics elements, respectively, while Figures 4.17a, b and c illustrate the structure of the corresponding global matrices ($\underline{A}$).

(B) Analysis of bandwidth

In order to economise on storage, the global matrix ($\underline{A}$) is dimensioned as a linear array in the program. It is necessary, therefore, to have a detailed knowledge of the structure of $\underline{A}$, before we can access any of its elements, lying either above,or below, or on,the leading diagonal.

Let us suppose that there are, in all, (n_t) degrees of freedom, and (n_a) coefficients above the leading diagonal. Since $\underline{A}$ is structurally symmetric, there will also be (n_a) coefficients below the diagonal. In the computer program, the matrix is represented in the

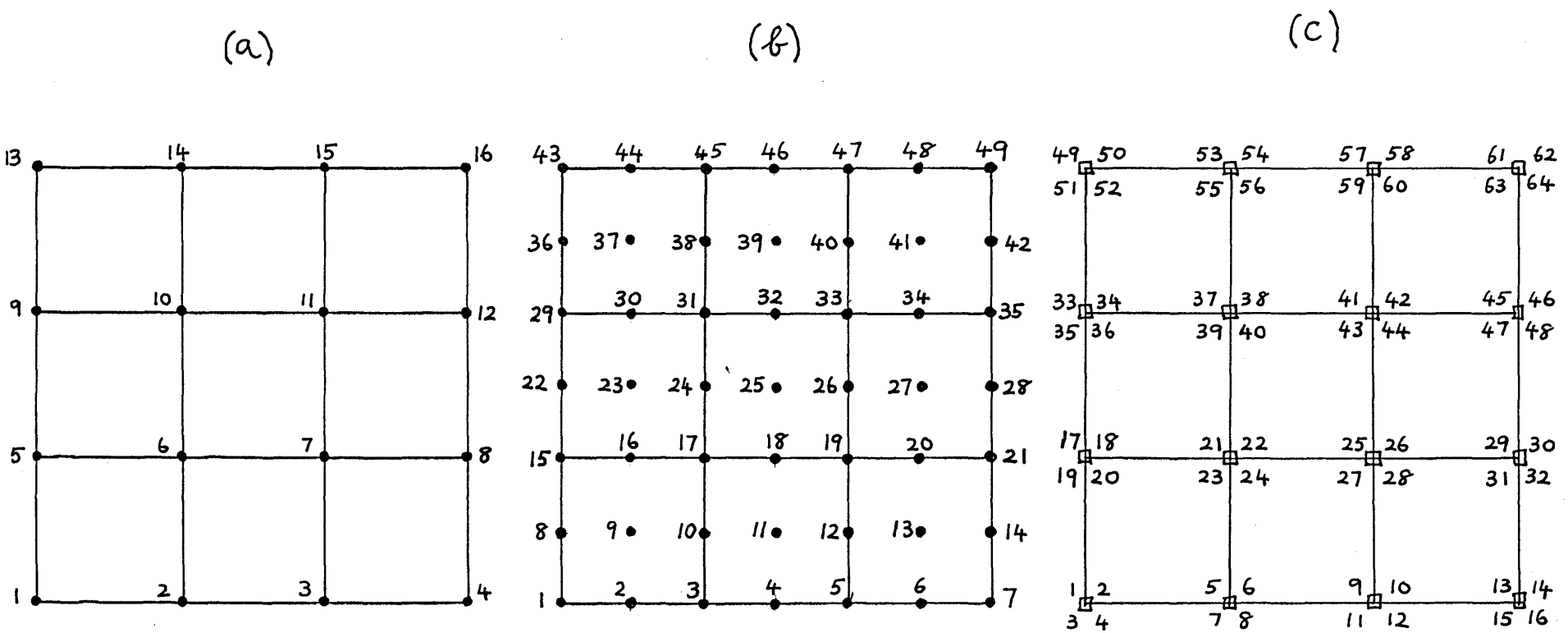
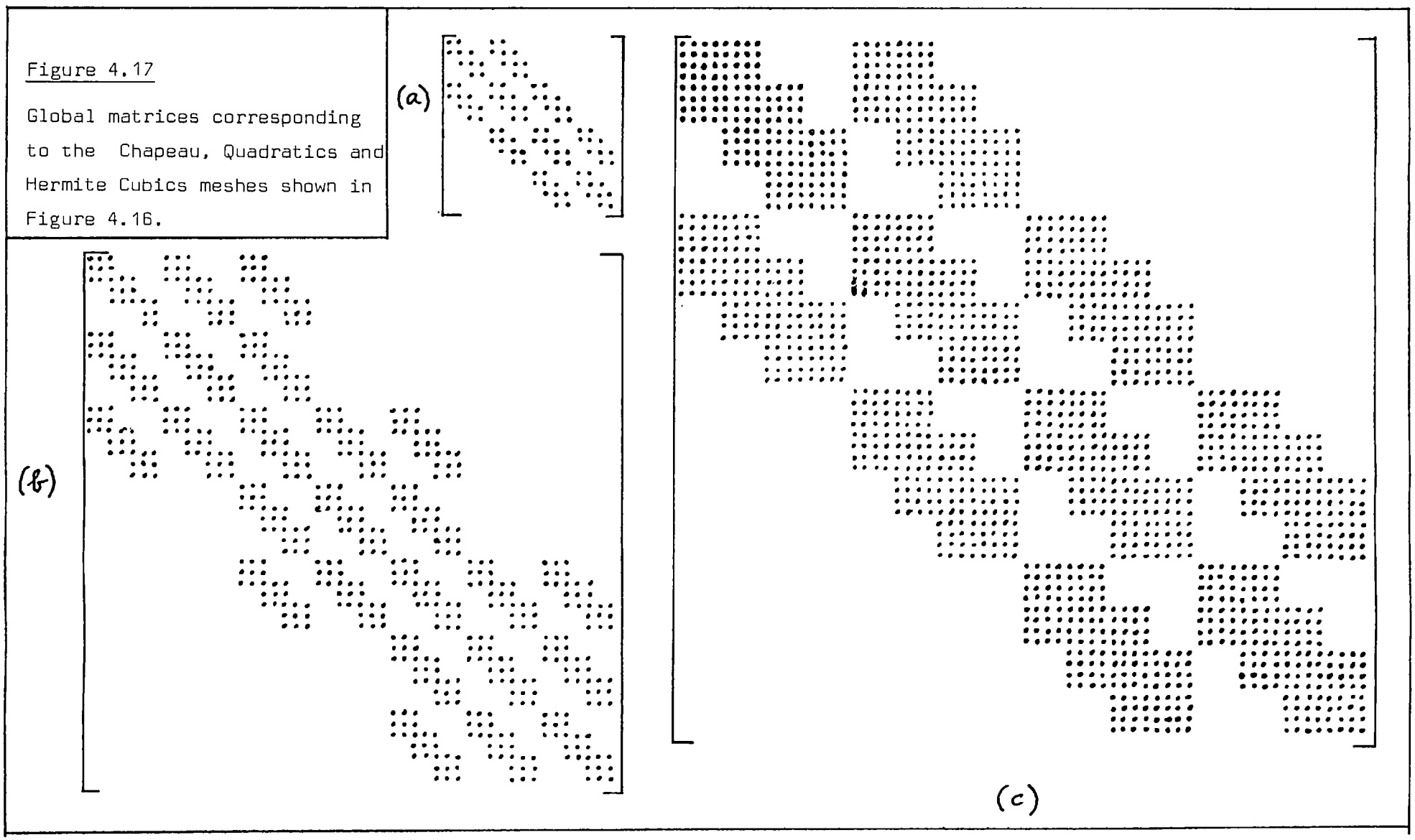


Figure 4.16 Standard row-by-row ordering of DOF's
(a) Chapeau, (b) Quadratics, (c) Hermite Cubics. (3x3 mesh).

Figure 4.17

Global matrices corresponding to the Chapeau, Quadratics and Hermite Cubics meshes shown in Figure 4.16.



following manner:

(1) Locations [1 to n_a] contain supra-diagonal terms, numbered row-by-row;

(2) Locations [(n_a+1) to $2n_a$] contain sub-diagonal terms, numbered column by column (i.e. mirror-image of (1));

(3) Locations [$(2n_a+1)$ to $(2n_a+n_t)$] contain the diagonal terms.

Figure 4.18 shows an enlarged sketch of the first few supra-diagonal rows of $\underline{A}$, for the Chapeau case shown in Figures 4.16a and 4.17a.

The principal task of the bandwidth analyser is to supply the program with a "key" index for accessing the contents of $\underline{A}$. This "key" consists of an integer array L1 [dimensioned to (n_t+1)] which gives the location within $\underline{A}$ of the first supra-diagonal term of each row. For example, the information given in Figure 4.18 will be recorded in L1 as follows:-

$$L1(1) = 1;\ L1(2) = 6\ ;\ L1(3) = 11\ ;\ L1(4) = 16\ ;$$

$$L1(5) = 20;\ \ldots\ldots\ L1(n_t-1) = n_a;\ L1(n_t) = n_a + 1.$$

Using this index, we can determine the location (l) which corresponds to any given pair of DOF's i and j [i.e. $\underline{A}$ (i,j)].

(a) If i<j, the location is supra-diagonal, and is given by:

$$l = L1(i) - 1 + j - i\ ;$$

(b) If i > j, then it is sub-diagonal, and

$$l = L1(j) - 1 + i - j + n_a\ ;$$

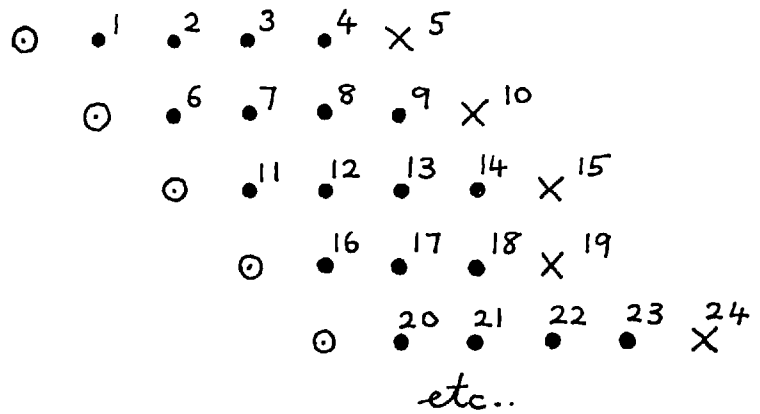


Figure 4.18 Illustration of numbering scheme used for supra diagonal terms of the finite element matrix (Chapeau case; Figure 4.16a).

⊙ = leading diagonal

× = outermost coefficient

● = intermediate coefficient

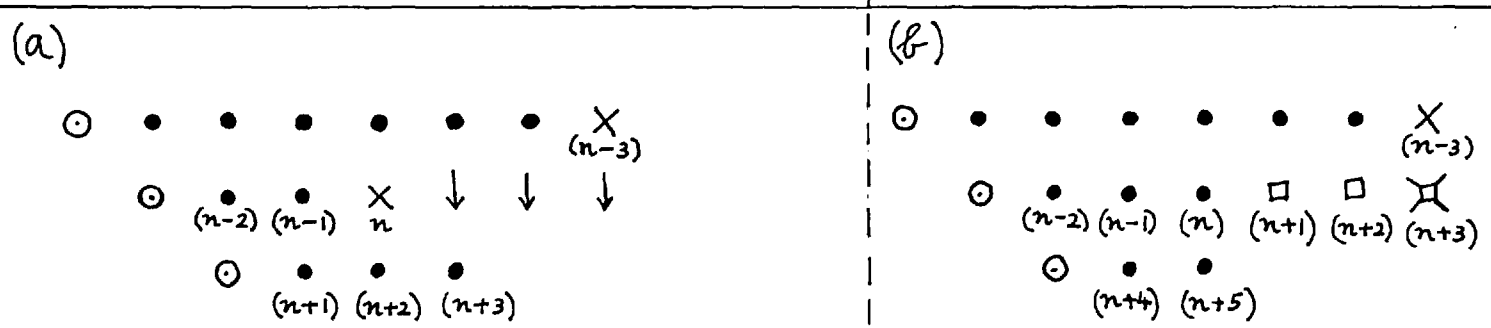


Figure 4.19 Illustration of (a) erroneous and (b) correct numbering of matrix coefficients when there is a shortening of the bandwidth. (□ = newly inserted locations).

(c) If i=j, it lies on the Leading diagonal :

$l = i + 2n_a$.

One possible pitfall, however, has to be guarded against when constructing the array L1. This arises whenever there is a decrease in the column number of the outermost location of a row. Figure 4.19 shows three consecutive rows of the matrix, in which the second row stops short of the first (at, say, location n). If special provision is not made, the numbering will proceed normally, as shown in Figure 4.19a, and the locations (n+1) onwards will belong to the third row. However, during the elimination phase, it is clear that the coefficients of the first row which lie outside the limits of the second row, will descend (as shown by the arrows) into unallocated locations, and will thus find their way erroneously into the "third row elements". It follows, therefore, that whenever there is a drop in the horizontal co-ordinate of the outermost coefficient, the shorter row must be extended as far as the end of the previous one, and the intermediate locations numbered accordingly (Figure 4.19b). An example of this is encountered in the tensor product quadratics model (Figure 4.16b, 4.17b), where the presence of rows of internal nodes gives rise to shrinkages in the bandwidth.

The bandwidth analyser also performs the task of relating the local numbers of the DOF's of each element to their global numbers. This information is contained in the array NGL (i,j) where the subscripts i and j refer to the numbers of the element and DOF respectively. Thus, in Figure 4.16, using element 3 as an example, we have:

(a) <u>For Chapeau</u> [NGL(3,j) ; j = 1 →4] :

3, 4, 7, 8.

(b) For Quadratics [NGL(3,j) ; j = 1 → 9] :

5, 6, 7, 12, 13, 14, 19, 20, 21.

(c) For Hermite cubics [NGL(3,j) ; j = 1 → 16] :

9, 10, 11, 12, 13, 14, 15, 16, 25, 26, 27, 28, 29,30, 31, 32.

(C) Solution method for 2D systems

The Banded solution method in 2 dimensions consists basically of a repetition of the threefold process of assembly, elimination and displacement (or shifting) as described for the 1D models. The only difference, however, is that these steps are now repeated after each row of elements, rather than after each element. Thus, the sequence is as follows:

{ Assembly of 1st row of elements
{ Elimination of 1st row of nodes (or DOF's)
{ Assembly of 2nd row of elements
{ Elimination of 2nd row of nodes (or DOF's)
etc...

Consequently, the active matrix A, and the indexing arrays L1 and NGL, will refer to a row of elements and their associated degrees of freedom. The solution technique can best be illustrated by a series of sketches showing the structure of the active matrix during various stages of the cycle.

Figure 4.20a shows a sketch of the global matrix for a grid consisting of 3 rows. The active matrix needed for the assembly and solution takes on the hexagonal shape shown in Figure 4.20b. The

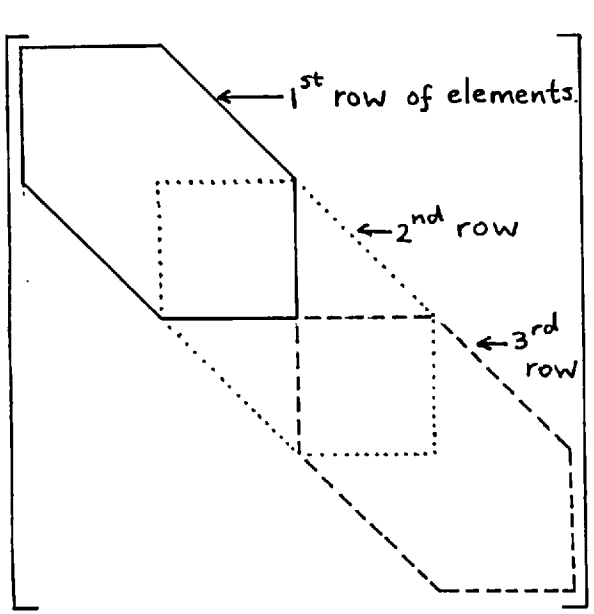


Figure 4.20 Schematic diagram showing (a) "Global" and (b) "active" matrices.

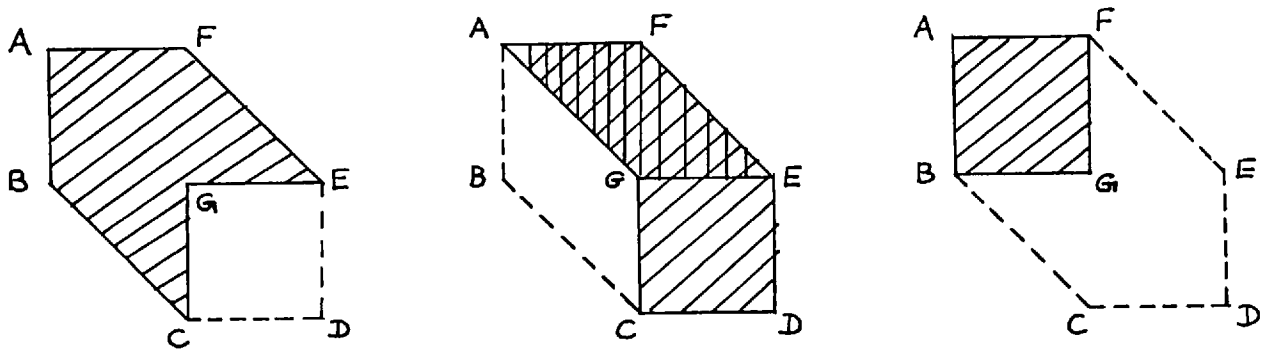


Figure 4.21 Structure of the "active" matrix at different stages of the solution procedure.

steps involved in the solution cycle can be summarised as follows:-

(1) After assembling a typical row (i) of elements, the matrix has the form ABCGEF shown in Figure 4.21a. [The solid lines enclose the non-zero elements].

(2) Elimination of the outgoing DOF's (A→G) converts it to the shape AGCDEF (Figure 4.21b)

(3) The supra-diagonal part AGEF of the eliminated section (shown cross-hatched in Figure 4.21b) is written to a disk file as a binary record, for use, later, during back-substitution.

(4) The active (lower) part GCDE is shifted diagonally upwards into the area ABGF, and all the remaining locations are cleared (Figure 4.21c).

(5) Assembly of the next row (i+1) of elements restores the matrix to its original shape ABCGEF, and brings us back to step 1.

These steps are repeated for each row, at the completion of which the solution vector is generated by back-substitution. The reading of records from a disk file in reverse order may be done in either of two ways :

(a) by using the Fortran statement : BACKSPACE, or

(b) If mass storage devices are available (as at Imperial College), it may be possible to create files, whose records can be accessed <u>randomly</u>. The latter approach has been used in the present study.

4.11 Miscellaneous features

The construction of the finite element models used in this study has been based on the theory described in the previous sections. Given below are some of the minor aspects of the programs, not covered before.

4.11.1 Time-step control

The selection of time-steps is done by means of an amplification factor, r, which is defined as the smaller of the ratios of the maximum allowable changes in pressure and saturation [$|\Delta p|_{lim}$ and $|\Delta S_w|_{lim}$, which are specified externally] to the maximum changes that occurred over the previous time-step [$|\Delta p|_{max}$, $|\Delta S_w|_{max}$]. Thus, the model will accelerate or decelerate, according as $r \gtrless 1$.

However, two further constraints are applied before the new time-step is determined. These are:

(a) r is not allowed to exceed 2; and

(b) the final time-step should be less than, or equal to, a certain pre-assigned maximum value (Δt_{max}).

These screening procedures can be expressed collectively, in mathematical notation, as follows:

$$\left.\begin{aligned} r &= \min\left[\frac{|\Delta p|_{lim}}{|\Delta p|_{max}}, \frac{|\Delta S_w|_{lim}}{|\Delta S_w|_{max}}, 2\right] \\ \Delta t_{new} &= \min\left[r\Delta t_{old}, \Delta t_{max}\right] \end{aligned}\right\} \qquad (4.95)$$

4.11.2 Calculation of well rates

Since the five-spot studies were confined to an incompressible, constant-rate displacement, the well rates are given by:

$$\left.\begin{aligned} q_o &= 0 \\ q_w &= q_{inj} \end{aligned}\right\} \text{ at the injection well; and} \qquad (4.96)$$

$$\left.\begin{aligned} q_o &= -q_{inj} f_o \\ q_w &= -q_{inj} f_w \end{aligned}\right\} \text{ at the production well.}$$

4.11.3 Graphical representation

The finite element models developed here are equipped with the necessary subroutines for producing graphical output at the specified "print-out" times.

For the 1D models, the graphs consist of saturation profiles, plotted with sufficiently many interpolation points to make them appear continuous. On each diagram, the analytical (Buckley-Leverett) solution is superimposed on the "numerical" profile for comparison.

For 2D results, a contouring program has been developed for printing accurate pressure - and/or saturation maps at the required times. A description of this contouring package forms the subject of the next chapter.

CHAPTER 5

A Contouring Program for Finite Element Results

5.1 Introduction and Theory

Results generated from a 2D Finite Element program can be analysed in their entirety only when they are presented in the form of a contour map. It is therefore necessary to possess a supporting program which is capable of transforming these raw results into a meaningful pictorial output, with as little distortion to the original results as possible.

Several contouring programs exist at Imperial College. Monro [40] and Raby [48], for example, have developed two independent packages for contouring arbitrary data. These general purpose programs, whose source listings are not normally available, pose two difficulties:

(1) they prefer a regular grid; and

(2) they employ interpolation polynomials and/or smoothing techniques which, in general, are different from the interpolation used in the finite element program . This inconsistency between the two interpolation schemes raises the obvious question as to the extent to which such programs are capable of reproducing the original results.

It was therefore necessary to develop a contouring program, specially designed for finite element models, and capable of overcoming these two difficulties. This chapter deals with the theory behind the development and application of such a program.

5.1.1 The need for "numerical" contouring

The process of locating contours "analytically" in two dimensional space is both an arduous and an expensive task. It reduces to a repeated application of a root-finding technique such as the method of "Newton-Raphson". Inside each finite element, a contour "c" satisfies some polynomial equation

$$f(x,y) = c \qquad (5.1)$$

which cannot normally be written explicitly in the form.

$$y = g(x) \quad ,$$

because the complexity of $f(x,y)$ will depend on the type of finite element that is being used. Furthermore, within each element, a contour has, in general, three possibilities - namely that it

(a) passes through (or exists) as a single curve;

(b) passes through (or exists) as several curves;

(c) is absent altogether.

An attempt was made by Murphy [42] to develop an "analytical" contouring package based on the root-finding principle. The particular finite element chosen by him to demonstrate these ideas was the cubic non-conforming triangle of Bazeley et al [5]. Although the contouring program has a sound theoretical basis, it nevertheless bears out some of the practical difficulties mentioned above.

The numerical method described below has been found to be a

simple and systematic way of tracing contours.

5.1.2 The principles of "numerical" contouring

We begin with two obvious assertions:

(1) Every continuous curve in space can be approximated by a finite number of straight-line segments. This "discrete" curve will converge to the "continuous" curve, as the number of segments is increased.

(2) A contour will always belong to one of the following two categories : either

a) a closed curve, lying entirely within the domain, or

b) an open curve, whose ends terminate at the outer boundary of the domain.

We shall use the terms "CLOSED" and "OPEN" to refer to the above two cases respectively.

The contouring program divides the original finite element mesh into numerous subtriangles, and, at each of the resulting vertices (or subnodes) the function value is calculated by means of the usual finite element interpolation. We thus have a "finer" secondary mesh consisting solely of triangles, at whose vertices the function value is known.

The next step is to specify a CHAPEAU (or linear) variation of the function over each of these subtriangles, and herein lies the essence of this numerical technique. The Chapeau interpolation means that a contour :

(1) can never be entirely within a subtriangle; and

(2) may pass through a subtriangle at most once, the path being given by a straight line joining two points on the sides of the subtriangle.

We have thus overcome the problem of multiple occurrences mentioned in section 5.1.1. The contour may now be pursued to its ends by proceeding from one subtriangle to another along these unique straight-line segments, until the contour either reaches the outer boundary (OPEN), or returns to whence it started (CLOSED).

This is the basic philosophy underlying the present approach. Subsequent sections will describe its detailed implementation within the contouring program, and illustrate the successive steps by means of an example problem.

5.1.3 Finite Element types

The types of finite elements for which the contouring program was originally designed are listed in Table 5.1, where NELNUM is the integer code used for each element .

NELNUM	Element type	Nodes	Degrees of freedom
1	Serendipity Chapeau Rectangle	4	4
2	Serendipity Quadratic Rectangle	8	8
3	Serendipity Cubic Rectangle	12	12
4	Chapeau triangle	3	3
5	Quadratic triangle	6	6
6	Cubic triangle	10	10
7	Hermite cubic Rectangle	4	16
8	Lagrange Quadratic Rectangle	4	9

Table 5.1 Finite element types and their code numbers.

The program, however, has been tested only on those elements which have been employed in this thesis - namely, elements 1, 7 and 8. Clearly, the nodes control the geometry of the elements, while the degrees of freedom control the variation of the function that is being approximated over the elements. Figure (5.1) shows the local ordering sequences used by the program for the nodes, and DOF's of the different elements. The element sides are ordered in an anticlockwise sense, commencing with the side which connects nodes 1 and 2 (Figure 5.2a).

All the above information concerning the element properties are held, in coded form, in an array called NPLOT, which is initialised by a BLOCK DATA statement. The arrays NCYC3 and NCYC4 represent cyclic rotations of the first three and four integers respectively (Figure 5.2b). The example problem chosen to demonstrate the contouring principles consists of a simple domain with four rectangular elements, within which it is required to trace three contours, as shown in Figure 5.3.

5.2 Generation of secondary mesh

The task of dissecting the original mesh into a finer one is assigned to the subroutine SUBDIV. One condition, however, must be met. The subdivision of each finite element into subtriangles must be such that all inter-element boundaries [e.g. OE, OF, OG, OH in Figure 5.3] are segmented into the same number of intervals. Let this number be n_D (=NELDIV in the program). It is a measure of the "fineness" of the contouring subdivisions, and can be varied to obtain any desired degree of accuracy.

Rectangular finite elements are first split into $(n_D \times n_D)$ smaller rectangles, each of which is then subdivided into 4 triangles.

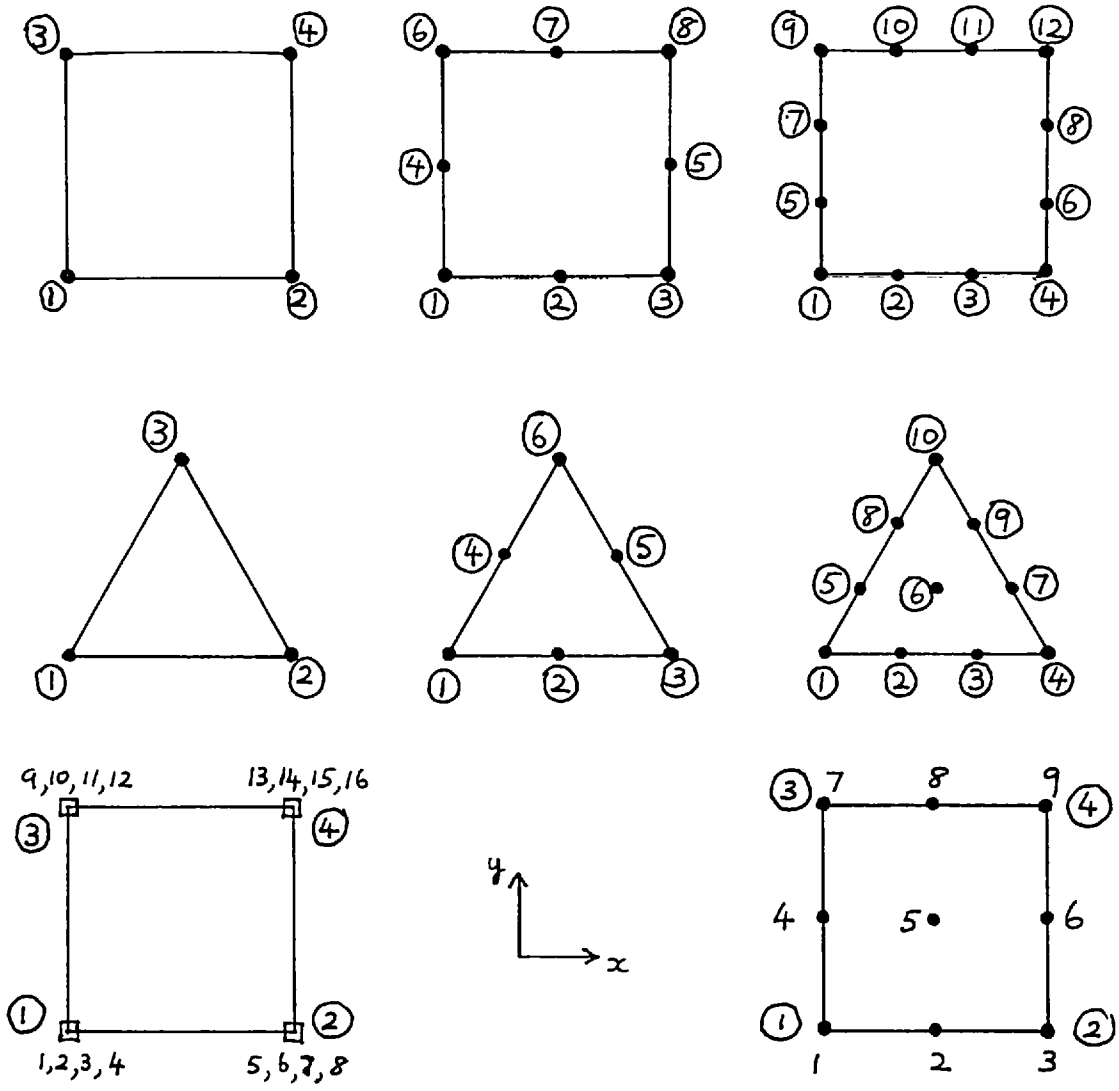
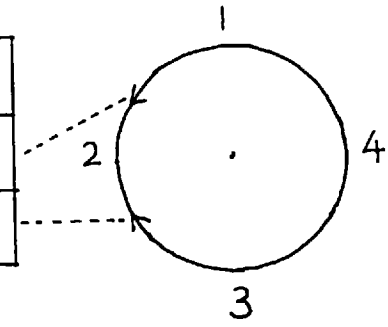


Figure 5.1: Local ordering sequences for the nodes and DOF's of various types of finite elements.

Figure 5.2(a): Local anticlockwise ordering used for element sides. (Rectangles and triangles).

i	1	2	3	4
NCYC4(1, i)	2	3	4	1
NCYC4(2, i)	4	1	2	3



i	1	2	3
NCYC3(1, i)	2	3	1
NCYC3(2, i)	3	1	2

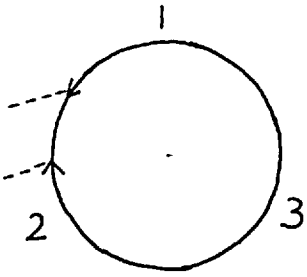


Figure 5.2(b): Cyclic permutation arrays NCYC4 and NCYC3.

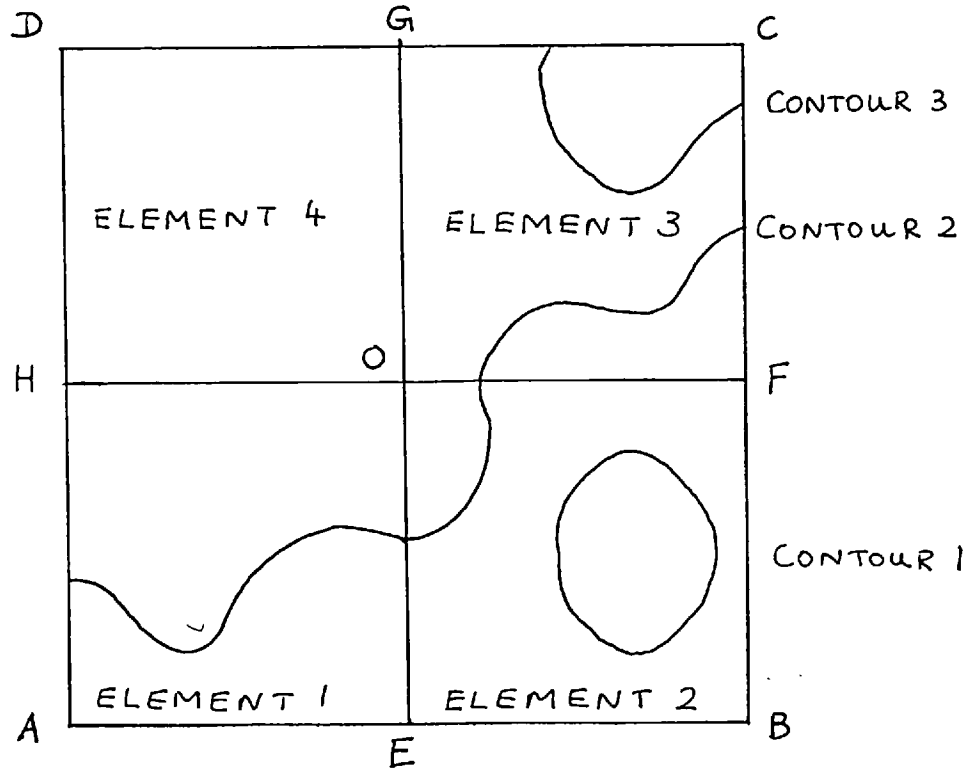


Figure 5.3: Example problem to demonstrate the principles of contouring.

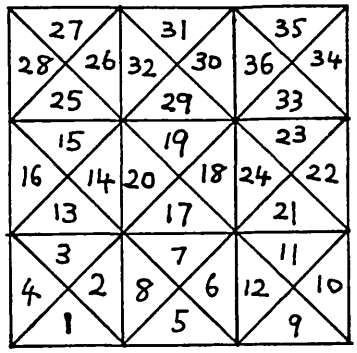
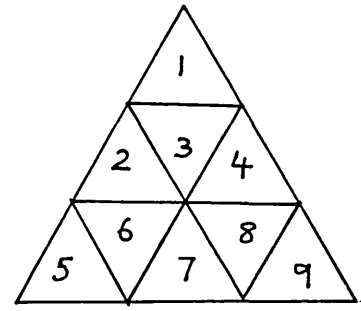
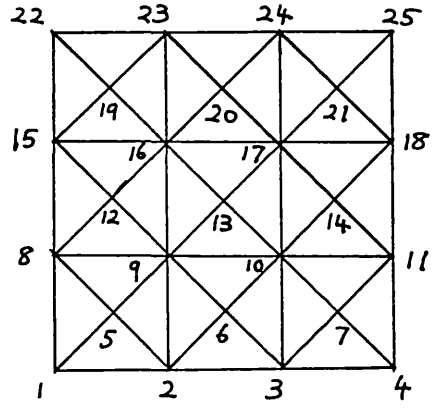
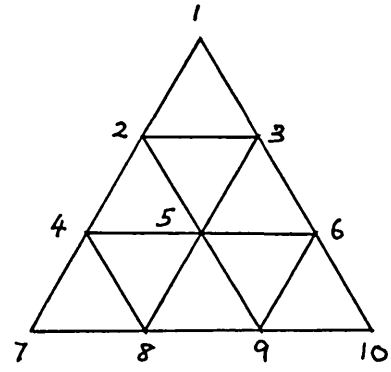
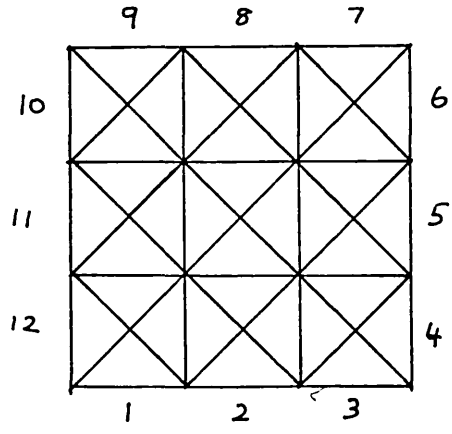
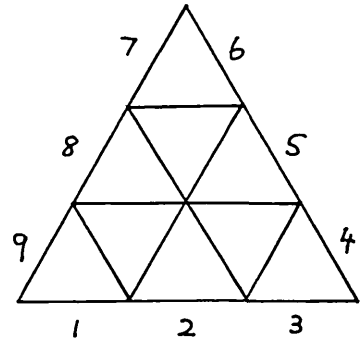
RECTANGULAR F.E.	TRIANGULAR F.E.
(a) Sub-triangles : $n_T = 4n_D^2$ (= 36). 	$n_T = n_D^2$ (= 9). 
(b) Sub-nodes : $n_N = 2n_D^2 + 2n_D + 1$ (= 25). 	$n_N = \frac{1}{2}(n_D+1)(n_D+2)$ (= 10). 
(c) Peripheral segments : $n_p = 4n_D$ (= 12). 	$n_p = 3n_D$ (= 9). 

Figure 5.4: Local numbering schemes used for the secondary mesh: (a) subtriangles, (b) subnodes and (c) peripheral segments, for the case $n_D = 3$.

If the parent element is a triangle, then the required subdivisions may be obtained by simply drawing lines parallel to its three sides. In both cases, it is clear that the stipulated condition (of equal peripheral divisions) will be satisfied.

Figure 5.4 illustrates the subdivisions for the case $n_D = 3$, and also shows the resulting subtriangles (n_T), subnodes (n_N) and the peripheral divisions (n_p). The local numbering scheme used for these parameters is quite straightforward. As can be seen from the Figure, the sub-nodes are numbered regularly row-by-row (Figure 5.4b), whilst the peripheral divisions are counted in an anticlockwise sense, beginning with side 1 (Figure 5.4c). The subtriangles are taken anticlockwise within each subrectangle, and this sequence is continued row-by-row (Figure 5.4a).

5.2.1 Subtriangle connectivity data

Having established the secondary mesh, the subroutine SUBDIV calculates, for each subtriangle i, six related parameters which are stored in the array NEL (i,6). The first three of these [namely, NEL(i,1) to NEL(i,3)] contain the subnode numbers of its three vertices (taken anticlockwise, starting with the vertex at the centre of the subrectangle). Thus, subtriangle 19, for example, (in Figure 5.4) has:

NEL(19, 1) = 13

NEL(19, 2) = 17

NEL(19, 3) = 16

Locations NEL(i,4) to NEL(i,6) each contain, in coded form, a pair of values associated with the three vertices respectively. Each pair is

made up of:

(a) the number of the opposite subtriangle; and

(b) the serial number (1, 2 or 3) of its facing vertex.

Thus, continuing with subtriangle 19, we have:

NEL(19,4) → (29,1) → 29 * 10 + 1 = 291
NEL(19,5) → (20,3) → 20 * 10 + 3 = 203
NEL(19,6) → (18,2) → 18 * 10 + 2 = 182

} Coded pairs.

Clearly, a difficulty arises when the subtriangle i lies on the periphery of the element (e.g. i = 22), since NEL (i, 4) would then refer to a subtriangle lying outside the element boundary (i.e. inside the neighbouring element). When this occurs, we simply record the number of the peripheral segment, and make it negative in order to indicate this situation. So, for example, in Fig 5.4, we have:

NEL(22,4) = - 5; NEL(4,4) = - 12 ; etc.

We may summarise the foregoing operations by stating that the array NEL contains information regarding each subtriangle and its immediate neighbours. Where the subtriangle straddles the element boundary, the array also records the number of the corresponding peripheral division. Since n_D is constant over the whole domain, it follows that all the elements have the same "internal structure" (which has been recorded in the array NEL), and that these elements communicate with one another via the same number (n_D) of peripheral segments.

5.2.2 Inter-element communication paths

It now remains to assign global numbers to the peripheral segments, so that the paths of inter-communication between neighbouring elements

can be uniquely determined. This operation is performed jointly by the subroutines ELSIDE and GRTAPE. The numbering is done sequentially by running anti-clockwise round the boundary of each element, in turn, and by-passing those segments which have already been numbered. Segments lying on the outer boundary of the domain are distinguished from internal segments by adding 1000000 to their normal values. Figure 5.5 shows this numbering scheme as applied to our example problem, while Table 5.2 gives the contents of the array (NPERI(i), $i = 1 \rightarrow n_p$) which holds the peripheral numbers of each element.

The importance of these peripheral segments lies in the fact that they constitute a finite number of "bridges" through which contours may pass from one element to another. Furthermore, as was remarked in section 5.1.2, each bridge can admit at most only one contour of a specified value.

5.2.3 Interpolation within secondary mesh

The program next calculates and stores the co-ordinates of the subnodes, as well as the function values at these points. The shape functions needed for interpolation are contained in the array BASIS.

Since only rectangular finite elements have been used in this thesis, a simple bilinear (Chapeau) interpolation from the four corner nodes suffices to determine the co-ordinates of any internal point. Thus, at each subnode j, the array locations [BASIS(i,j) : $i = 1 \rightarrow 4$] are reserved for these geometrical (bilinear) shape functions. The locations i = 5 onwards contain the shape functions of the object variable that is to be contoured (i.e. po or Sw). If the finite element

NPERI (1→12)	Element 1	Element 2	Element 3	Element 4
1	1000001	1000013	21	9
2	1000002	1000014	20	8
3	1000003	1000015	19	7
4	4	1000016	1000022	30
5	5	1000017	1000023	29
6	6	1000018	1000024	28
7	7	19	1000025	1000031
8	8	20	1000026	1000032
9	9	21	1000027	1000033
10	1000010	6	28	1000034
11	1000011	5	29	1000035
12	1000012	4	30	1000036

Table 5.2 Global numbers of the peripheral segments of each element.

---

if of the Chapeau type, then, clearly, the shape functions of the variables coincide with the geometrical shape functions. If, on the other hand, the finite element belongs to the Lagrange quadratic - or Hermite cubic varieties, then we need an additional 9 (i = 5 → 13), or 16 (i = 5 → 20) shape function locations, respectively.

For contouring purposes, the dimensions of the domain are normalised to the range [0, 1].

Thus if

[XVAL(i), YVAL(i) : i = 1 → 4]

denote the normalised co-ordinates of the four corner nodes of a typical element, and

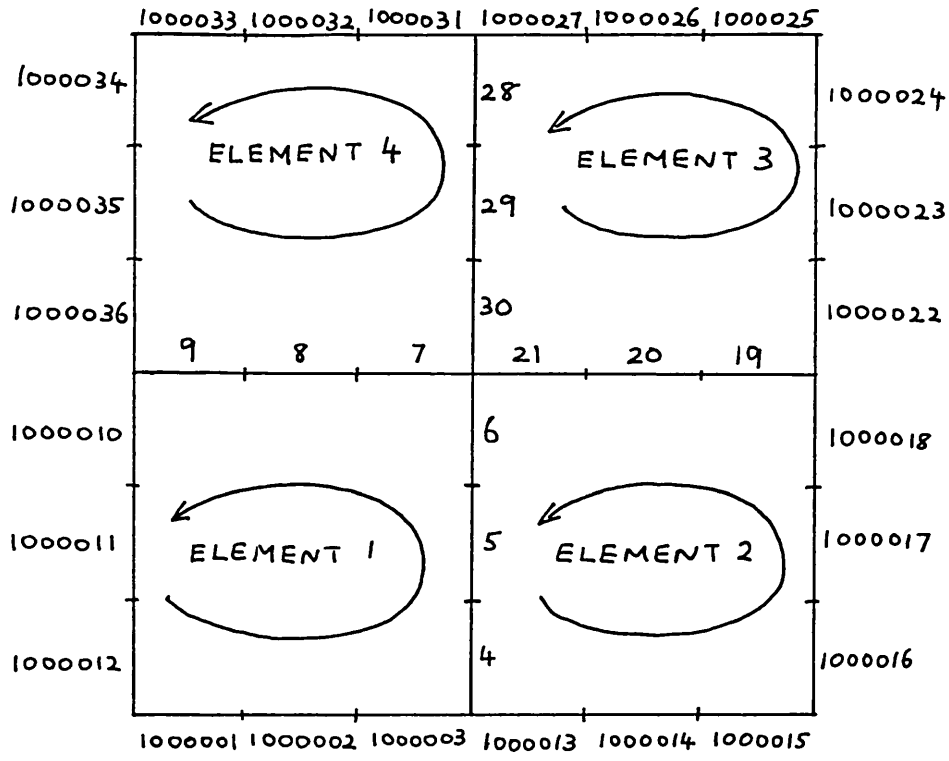


Figure 5.5: Global numbering scheme for peripheral segments.

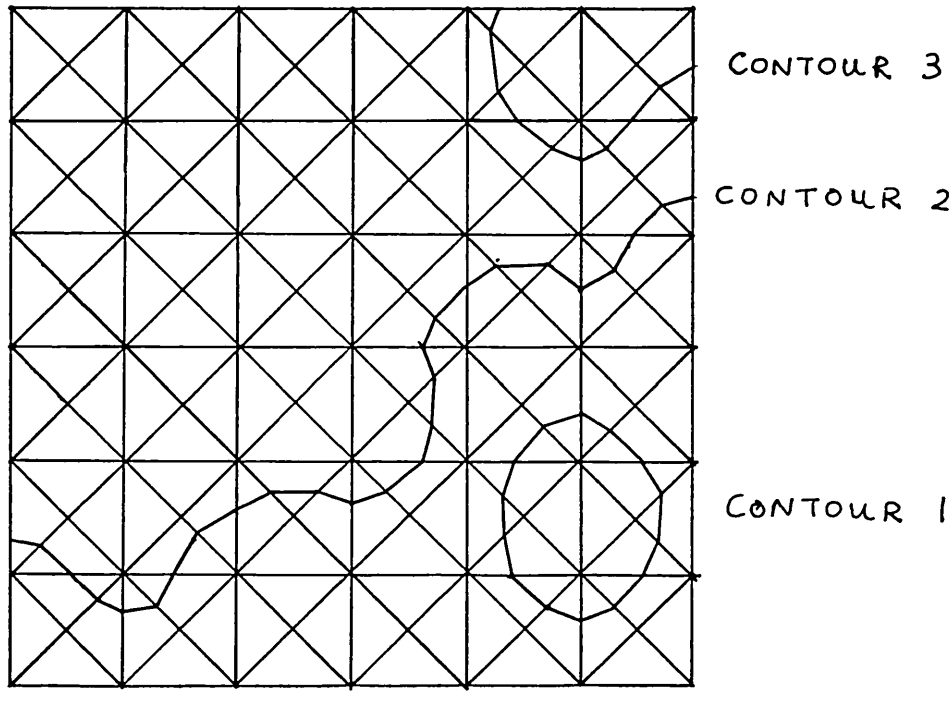


Figure 5.6: The passage of contours through subtriangles (see example problem shown in Figure 5.3)

[ZVAL(i); i = 1 → NDOF]

represent the elemental degrees of freedom, then, using the shape functions contained in the array BASIS, we can calculate the normalised co-ordinates (x_j, y_j) and the associated function value (z_j), at each subnode j of the element. Thus:

$$\left.\begin{aligned} x_j &= \sum_{i=1}^{4} BASIS(i,j) * XVAL(i) \\ y_j &= \sum_{i=1}^{4} BASIS(i,j) * YVAL(i) \\ z_j &= \sum_{i=5}^{4+NDOF} BASIS(i,j) * ZVAL(i) \end{aligned}\right\} \quad j = 1 \rightarrow n_N . \qquad (5.2)$$

The shape functions are evaluated by calls to the subroutine BASISF, while the interpolations described above are performed by subroutines GRTAPE (for x_j and y_j) and PICKUP (for z_j)

5.3 Contour tracking (inside elements)

As observed in section 5.1.2, the assumption of a Chapeau variation ensures that the path (if any) of a contour through a subtriangle consists of a single straight line, connecting two of its sides. This means that, once a contour is located within a subtriangle, it is possible, by using the connecting data described in section 5.2.1, to propagate this contour into the two adjacent subtriangles in the form of two additional straight-line segments. By continuing this procedure, the two ends of the contour are advanced, step by step, through successive subtriangles, until they either meet (to form a closed loop), or reach the sides of the

element. In the latter case,we record the global numbers of the peripheral divisions where the ends of the contour terminate. Figure 5.6 refers to our example problem, and shows, in detail, the paths of the three contours.

5.3.1 Locating a contour

Before pursuing a contour inside an element, it is first necessary to test for its existence, and then to locate it in any one of the subtriangles. To facilitate this search, the model computes, beforehand, the local maximum and minimum values assumed by the function over each subtriangle [ZMAX(i), ZMIN(i) : $i = 1 \rightarrow n_T$], as well as over the whole element [ZH and ZL]. Since the Chapeau relationship permits the occurrence of maxima and minima only at the nodes of triangles, the above quantities can easily be determined by running through the function values at the subnodes [ZVAL (i), $i = 1 \rightarrow n_N$]

Let us take, for example, a contour z = ZCON. The subroutine PICKUP tests for the presence of this contour by checking to see if

$$ZL < ZCON < ZH . \qquad (5.3)$$

If the test is positive, it then proceeds to do a systematic scan of all the subtriangles until it finds one which contains the required contour. [During this search, the array (NELCOV(i), $i = 1 \rightarrow n_T$) keeps a record of the subtriangles that have been examined. It is initialised to zero, and is set equal to unity as the search proceeds]. The contour will be picked up in the first subtriangle i to satisfy the inequality :

$$ZMIN(i) < ZCON < ZMAX(i) . \qquad (5.3a)$$

At this point, the search is interrupted, and control is transferred to subroutine FOLLOW which follows the contour through to its ends, as described in the next section. For example, in element 2 of our hypothetical problem, contour 3 is absent, while contours 1 and 2 will first be encountered in subtriangles 6 and 15, respectively (Figures 5.6 and 5.4a).

5.3.2 Following a contour

Let i, j, k represent the integers 1, 2, 3 (in any permutation), and let I, J, K be the corresponding subnode numbers of the subtriangle IST, in which the contour has been detected; i.e. :

NEL(IST,i) = I ; NEL(IST,j) = J ; etc.

Figure 5.7 shows schematically the layout of IST, together with its adjacent subtriangles and vertices.

So, given that the contour z = ZCON lies within IST, it is possible to find an anticlockwise permutation of ijk (i.e. 123, 231 or 312) such that

$$ZVAL(J) > ZCON > ZVAL(K) . \qquad (5.4)$$

From elementary geometry, it now follows that the co-ordinates of the point P1, where the contour meets the side jk, are given by:

$$\left.\begin{aligned} x_{P1} &= (1-f).XVAL(J) + f.XVAL(K), \text{ and} \\ y_{P1} &= (1-f).YVAL(J) + f.YVAL(K) \end{aligned}\right\} \qquad (5.5)$$

where

$$f = \frac{ZVAL(J) - ZCON}{ZVAL(J) - ZVAL(K)} .$$

With P1 known, we may now advance into the neighbouring subtriangle ISTI (facing node I), the relevant information concerning which is extracted from the databank array NEL (Section 5.2.1).

Thus we have:

$$\left.\begin{aligned} ISTI &= NEL(IST, i+3)/10 \\ i' &= NEL(IST, i+3) - 10 * ISTI \text{ ; and} \\ I' &= NEL(ISTI, i'). \end{aligned}\right\} \quad (5.6)$$

The new subtriangle ISTI is now treated in the same manner as IST, and the next point P2 is located on one of its two remaining sides, as follows:

(1) If ZCON > ZVAL (I') :

Then P2 lies on the side JI', while the vertices KJI' form an anticlockwise trio. The new values taken on by the parameters are:

$$\left.\begin{aligned} IST &= ISTI \quad ; \\ J &= \text{unchanged} \quad ; \\ K &= I' \quad ; \text{ and} \\ i &= NCYC3\ (1,i'). \end{aligned}\right\} \quad \begin{aligned} &(5.7a) \\ &(5.7b) \\ &(5.7c) \\ &(5.7d) \end{aligned}$$

(2) If ZCON < ZVAL(I') :

Then P2 lies on I'K, and JI'K constitute the required anticlockwise sequence. The parameters will be re-set to their new values, as follows:

$$\left.\begin{array}{lll} IST = ISTI & ; & \\ J = I' & ; & \\ K = \text{unchanged} & ; \text{ and} & \\ i = NCYC3\ (2,i') & . & \end{array}\right\} \qquad \begin{array}{l} (5.8a) \\ (5.8b) \\ (5.8c) \\ (5.8d) \end{array}$$

The program continues to repeat the steps from equation (5.5) onwards, thereby conducting the contour through successive subtriangles, and generating the sequence of points P_1, P_2, P_3, ---- etc. (Figure 5.7)

Should the contour arrive back at its starting point - namely, the original subtriangle IST - then, it clearly belongs to the "CLOSED" type, and its complete path is therefore given by the points:

$$P_1\ P_2\ P_3\ \text{-----}\ P_{n-1}\ P_n\ P_1 .$$

If, on the other hand, the contour reaches one of the sides of the element after tracing out the points P_1 to P_n (a situation which will be diagnosed by encountering a "negative" subtriangle - see section 5.2.1), then the program returns to the original subtriangle IST, and repeats the entire procedure described in this section, but this time in a clockwise sense. All occurrences of the word "anticlockwise" in the foregoing discussion will thus be replaced by "clockwise", and equations 5.7d, and 5.8d will be interchanged. The resulting points $Q_1\ Q_2$ ---- Q_m (Figure 5.7) will constitute the "second half" of the contour, terminating, as expected, at a peripheral segment. The passage of the contour through the element will thus be represented by the series of points :

$$Q_m\ Q_{m-1}\ \text{----}Q_2\ Q_1\ P_1\ P_2\ \text{----}\ P_{n-1}\ P_n ,$$

where Q_m and P_n are the two extremities.

5.3.3 Completing the search

The program returns to the end of section 5.3.1, and resumes its scan of the remaining subtriangles [from (IST+1) to n_T]. Any subtriangle that has already been covered (indicated by NELCOV(i) $\neq$ 0) will be ignored. This continued search enables the program to discover any further loops or segments of the contour ZCON within the element, thus ensuring that no section of a contour is missed.

The search for a single contour (z = ZCON) having been completed, sections 5.3.1 through 5.3.3 are now repeated for the rest of the contours, before passing on to the next element to repeat the entire procedure, all over again. Table 5.3 shows the results generated for our example problem, after traversing all the elements :

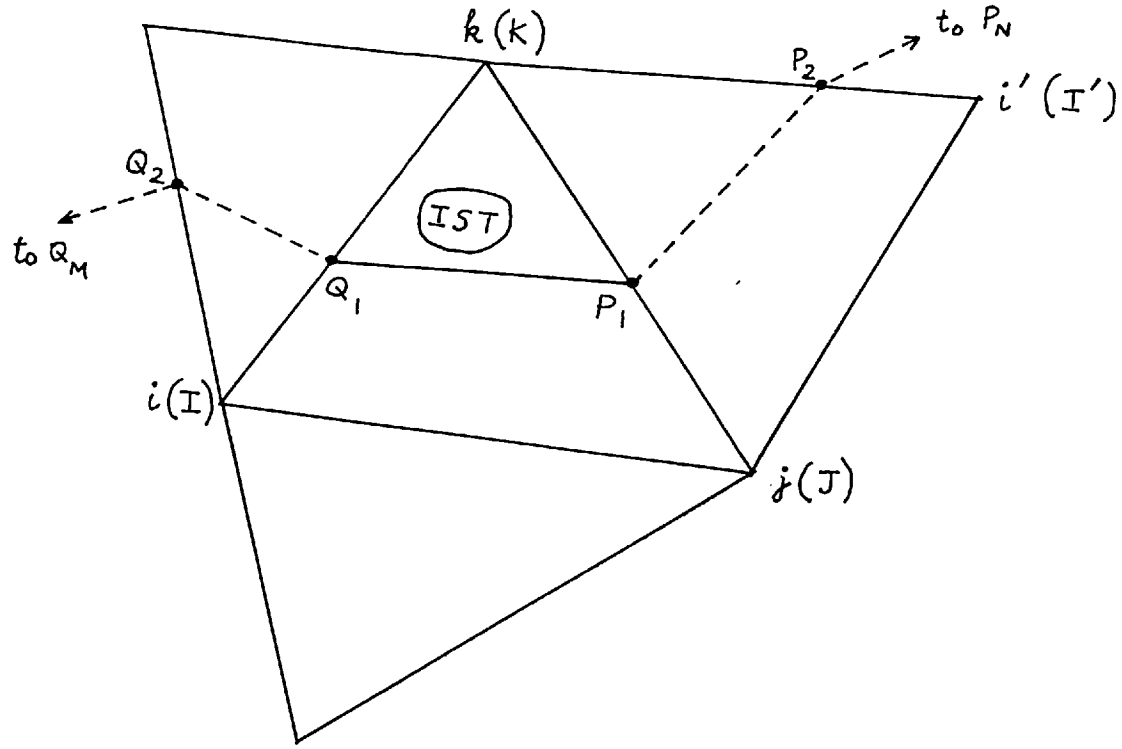


Figure 5.7: The pursuit of a contour from one subtriangle (IST) to another.

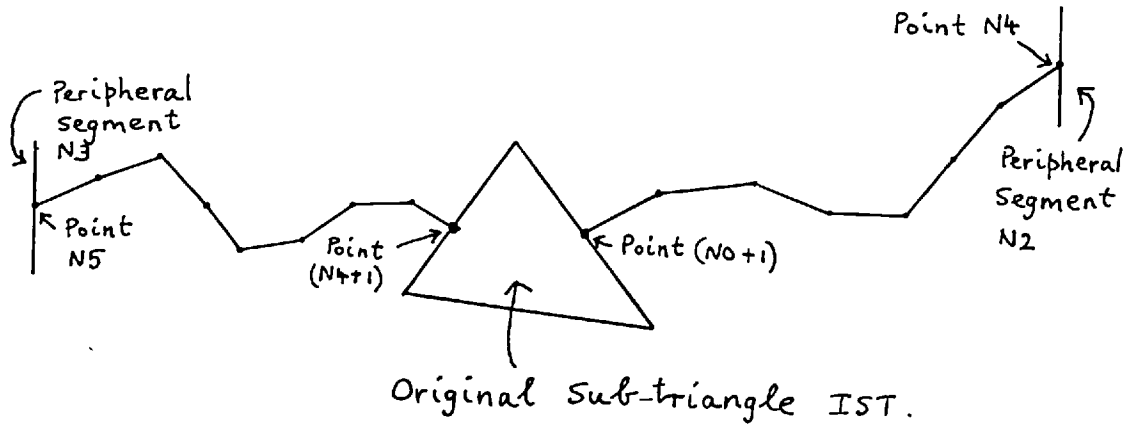


Figure 5.8: The two "halves" of a contour within an element.

Elm No.	Contour No	Starting Subtriangle	Subtriangles traversed	
			First Half **	Second Half**
EL 1	1	ABSENT	———	———
	2	2	2,8,7,17,18, 24,23,22, -4*	2,3,13, 16, - 11*
	3	ABSENT	———	———
EL 2	1	6	6,12,11,21,22, 23,33,36,30,29, 19,20,17,7,6	———
	2	15	15,25,26,27, -9*	15,16, -11*
	3	ABSENT	———	———
EL 3	1	ABSENT	———	———
	2	1	1,2,8,7,6, 12,11,21,22,-5*	1, - 1*
	3	18	18,24,23, 33,34,-6*	18,19,29, 32,31,-8*
EL 4	ALL CONTOURS ABSENT			

* Peripheral Segment
** One half corresponds to the "anticlockwise" sequence, and the other to the "clockwise" sequence.

TABLE 5.3 Trajectories of the contour-segments.

5.3.4 Storing the results

The co-ordinates of all the points calculated above are held in bulk storage arrays XP and YP. However, for book-keeping purposes, it is convenient to treat each loop (or segment) of contour as a single basic record, representing, for example, the n points:

$$[XP(N+1), YP(N+1)] \quad \text{to} \quad [XP(N+n), YP(N+n)].$$

To access any particular record in this bulk collection, and to extract all the relevant details regarding it,we make use of a "key" array called INDEX. Every five consecutive locations of this array contain information concerning a single record.

N1	N2	N3	N4	N5

N1 = Serial no. of the contour

N2 = Terminal boundary segment of the "first half"

N3 = Terminal boundary segment of "second half"

N4 = Location (within XP,YP) of last point of "first half"

N5 = Location (within XP,YP) of last point of "second half".

If the record represents a closed loop, then, obviously :

$$\left.\begin{array}{l} N4 = N5, \\ \text{and we set} \quad N2 = N3 = 0 \end{array}\right\}. \qquad (5.9)$$

since the latter are irrevelant.

From Figures 5.5 and 5.6 (or, equivalently, from Tables 5.2 and 5.3) it can be verified that, for our example problem, the contents of the array INDEX will be as shown in Table 5.4

Record No	El.No	N1	N2	N3	N4	N5
0	—	0	0	0	0	0
1	1	2	5	1000011	8	12
2	2	1	0	0	25	25
3	2	2	21	5	29	31
4	3	2	1000023	21	40	41
5	3	3	1000024	1000026	46	51
- - - - - -	4	- - -	- - None - - -	- - - - -	- - -	- - -

Table 5.4 Contents of the "key" array INDEX (N1 to N5) for example problem.

The presence of "Record 0" is merely to indicate the beginning of the Table. Contour 1 exists as a single closed loop, and connects points 13 to 25. Contour 2, however, is composed of three parts which involve the groups of points 1 to 12, 26 to 31, and 32 to 41, respectively. Contour 3 is confined to element 3, beginning at peripheral segment 1000024, ending at 1000026, and linking points 42 to 51.

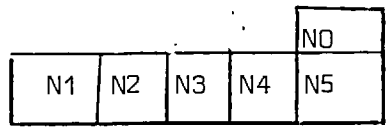
Thus the array INDEX not only provides a directory of addresses for the points constituting the individual records, but also supplies vital information regarding the character of the record - i.e. whether it represents a "closed" curve (in which case, the record is complete), or whether it is of the "open" type (in which case, it is incomplete, and its continuation has to be sought in another record).

5.4 Assembly of contours

The last major task confronting the model is to piece together the various disjointed records into a complete, properly - ordered sequence of points which the plotter can directly join. This process consists, in other words, of placing the records of each contour end-to-end (the ends being indicated by N2 and N3) until they fit into each other. It is somewhat analogous to the procedure described in sections 5.3.1 through 5.3.3, except that instead of pursuing the contours from subtriangle to subtriangle, we now move from element to element.

5.4.1 Arrangement of points within records

Let us consider again the structure of a basic record within INDEX



In this sketch, N0 represents the end of the previous record, and thus, we have: $N0 < N4 \leq N5$. From the description given in section 5.3.4, we know that:

(1) N2 and N3 represent peripheral segment numbers, which can be regarded as "links" joining this record to two others, and that

(2) the points belonging to this record are stored in locations [(N0+1) to N5] of XP and YP, and consist of two opposing sequences : (Figure 5.8)

(a) (N0+1) to N4, based on an anticlockwise movement, and

(b) (N4+1) to N5, based on a clockwise movement (c.f. section 5.3.2)

It is apparent, therefore, that before a fresh record can be added to a chain, one of the two "halves" of the new record must be re-sequenced in reverse order. So, given a new record which is to be inserted at the end (ILINK) of an existing chain, we have two possibilities:

(1) If ILINK = N2,

then the direction of the extension is from N2 to N3, as follows:

(a) N4 to (N0+1) in reverse order, and

(b) (N4+1) to N5 in normal order; and N3 becomes the open end. (i.e. the new ILINK)

(2) Conversely, if ILINK = N3, then the sequence of points to be added is from N3 to N2 :

(a) N5 to (N4+1) in reverse order, and

(b) (N0+1) to N4 in normal order; and N2 becomes the new ILINK.

In either case, the total number of new points that are added is (N5 - NO). The co-ordinates of these points are extracted from the bulk storage arrays XP and YP, and are transferred to the arrays XVAL and YVAL, which will eventually represent the complete and correctly-ordered sequence of points belonging to a single contour loop.

The rearrangements described above are performed jointly by the subroutines NULINK and CONECT.

5.4.2 Joining the records

The master subroutine CHAIN controls all the subroutines that are associated with the manipulation and assembly of records - namely JIGSAW, NULINK and CONECT. CHAIN itself, however, performs only the simple task of sifting through the array INDEX, and identifying those records which pertain to the contour z = ZCON. For ease of future access, the numbers of the starting locations (N1) of these records are stored in another array IIDX, which is later set to zero after the particular record has been analyzed.

The program deals first with OPEN contours. This means that the chain must begin, and end, on records which involve the outer boundary of the system. Accordingly, the subroutine JIGSAW scans these records until it finds one having either N2 or N3 greater than 10^6. Taking this as the first link (ILINK) of the chain to be constructed, control is then transferred to subroutine NULINK for the following operations :

(1) NULINK makes a note of the other (loose) end of the record, which will become the "next ILINK".

(2) It then calls subroutine CONECT which analyses the present

record, rearranges the points, and places them in the final arrays XVAL and YVAL. CONECT also cumulates the number of points making up the whole chain.

(3) Control is returned to NULINK which then searches among the remaining records of ZCON, to discover the continuation of the previous one (i.e. a record having N2 or N3 equal to the new ILINK).

By repeating the steps 1 through 3, fresh records are added onto the chain, one by one, until an ILINK is encountered with a value greater than 10^6, thereby indicating the emergence of this contour at another part of the outer boundary. Since this signifies the end of the contour in question, the program returns control to subroutine JIGSAW which proceeds to write out:-

(a) the serial number (ICONT) of the contour;
(b) the cumulative number (NPTS) of points involved; and
(c) the co-ordinates of these points (contained in XVAL,YVAL)

} onto a disk file named TAPE3:

WRITE(3) ICONT, NPTS, (XVAL(J), YVAL(J), J=1, NPTS).

Each binary record on TAPE 3 therefore represents a complete succession of points which are to be joined by the plotter.

The entire procedure is then repeated, as many times as required, to extract all the remaining OPEN curves (z=ZCON), and subsequently, all the CLOSED curves. In the latter case, the chain begins and ends on the same ILINK (rather than on an ILINK > 10^6, which was the criterion for OPEN contours).

Finally, control is returned to subroutine CHAIN, which re-constructs the array IIDX, and repeats all the foregoing operations for each of the remaining contour-values.

The tracking of contours is now complete, and a sentinel binary record is inserted to mark the end of TAPE 3. For our example problem, as expected, three contour loops will be generated, the final details of which are given in Table 5.5.

Loop No	Contour No	No of Points	Sequence of points, taken from XP,YP	Type
1	1	13	13 → 25	CLOSED
2	2	28	12 → 9,1 → 8, 31 → 28, 26 →27,41,32 → 40	OPEN
3	3	10	46 → 42, 47 → 51	OPEN

Table 5.5. The contour loops generated by this method for our hypothetical problem.

5.5. Labelling of contours

This aspect may seem unimportant at first sight, but it is not difficult to imagine the confusion that can arise from unlabelled contours. The procedure for labelling a contour can be expressed in two simple steps :

(1) The contour is split, in order to create a gap of sufficient length to accommodate the label, and

(2) the text of the label is printed in this gap, in its proper orientation.

In the case of OPEN contours, it is aesthetically desirable to locate the label approximately mid-way along the contour, while, for CLOSED contours, the position of the label is obviously immaterial. The label itself, being a number, consists merely of integer digits, with or without a decimal Point. Each character to be printed occupies a square region of height and width equal to "DH" inches. The scaling factor, "SCALE", is used for converting the normalised co-ordinates [XVAL(i), YVAL(i)] into inches $(\bar{x}_i, \bar{y}_i)$ relative to the plot origin. Thus, we have:

$$\left.\begin{aligned} \bar{x}_i &= XVAL(i)/SCALE, \text{ and} \\ \bar{y}_i &= YVAL(i)/SCALE \end{aligned}\right\}, \qquad (5.10)$$

for all points "i", where

SCALE = normalised co-ordinate-units per inch. [It is possible to modify the program so as to be able to use different scaling factors, "SCALEX" and "SCALEY", in the x and y directions, respectively].

5.5.1 Splitting the contour

Figure 5.9 shows, schematically, a contour made up of n points, such as would be obtained, for example, from a typical record on TAPE 3. Let the scaled co-ordinates of these points be:

$$(\bar{x}_i, \bar{y}_i) \quad : \quad i = 1 \rightarrow n .$$

The distance ΔLi,j (in inches) between any two points i and j on the contour is therefore:

$$\Delta L_{i,j} = \left[(\bar{x}_i - \bar{x}_j)^2 + (\bar{y}_i - \bar{y}_j)^2\right]^{1/2}, \qquad (5.11)$$

and the cumulative distance, L_i, along the contour, from point 1 to point i, is:

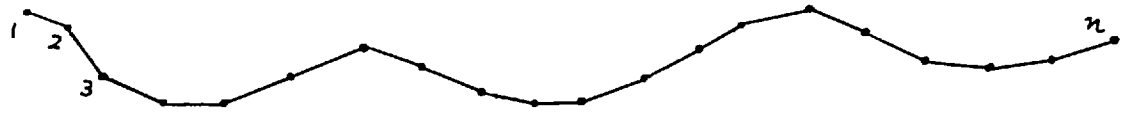


Figure 5.9: Sequence of points (P_1 to P_n) which make up a typical (unlabelled) contour.

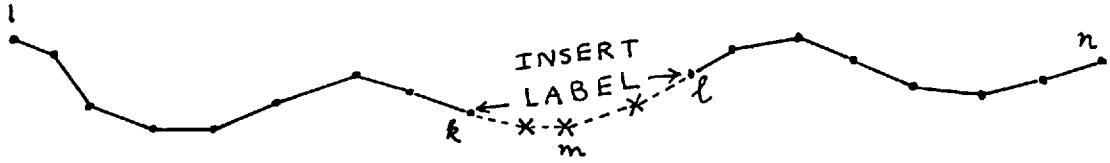


Figure 5.10: Creation of a gap in the contour to insert the label.

$$L_i = \sum_{j=2}^{i} \Delta L_{j-1,j} \qquad (i > 1) . \tag{5.12}$$

The total length of the contour is thus equal to L_n. If L_n does not exceed a certain prescribed minimum length "DMIN" (which is read in as data), the contour is considered to be too short, and hence left unlabelled. If, however, L_n > DMIN, then the program sets about creating a gap, whose length is approximately "DGAP". (DGAP is the recommended gap-length, which is also specified by the user). The creation of the gap is done as follows:

The mid-point Pm of the contour is taken to be that which falls just inside the half-way mark - i.e. the largest m such that:

$$L_m < \tfrac{1}{2} L_n .$$

Once this m has been determined, the points on either side of Pm are removed alternately, until an adequately large gap is produced. That is to say, starting at m, the model examines the following sequence of gap-lengths:

$\Delta L_{m-1,m}$; $\Delta L_{m-1,m+1}$; $\Delta L_{m-2,m+1}$; $\Delta L_{m-2,m+2}$ .... etc.

until it finds one such that

$$\Delta L_{k,l} > DGAP . \tag{5.13}$$

The contour has now been divided into two halves [points 1 to k, and l to n], and the intervening points, (k+1) to (l-1) inclusive, are removed from the string to make room for the label (Figure 5.10).

5.5.2 Insertion of the label

As shown in Figure 5.11, the program restricts the size of the label to a maximum of four characters. This is adequate for numbering pressure - and saturation contours in the normal range. Pressures less than 10000 psi require 4 integer digits (e.g. 9999), while saturations can be expressed by 3 digits and a decimal point (e.g. 0.52).

In order to print the label, the plotter must be supplied with two important parameters:

(a) the scaled co-ordinates $(\bar{x}_p, \bar{y}_p)$, in inches, of the lower left corner (P) of the first character of the string (Figure 5.11), and

(b) the angle θ_D (measured anticlockwise) at which the plotting direction is inclined to the x-axis.

Although these can be calculated by simple geometry, Figure 5.12 shows an embarrassing situation which can arise if sufficient care is not exercised. In Figure 5.12a, the label has been correctly printed, while that in Figure 5.12b is clearly upside down! The correct form of the erroneous label is shown in Figure 5.12c.

It is apparent from these figures, therefore, that, for a label to have the correct orientation, it must always be printed in the direction of increasing x co-ordinate. This means that its direction is from:

$$\left[\begin{array}{ccc} k & \text{to} & \ell \\ & \text{or} & \\ \ell & \text{to} & k \end{array}\right] \quad \text{according as} \quad \left[\bar{x}_k \lesseqgtr \bar{x}_\ell\right]. \tag{5.14}$$

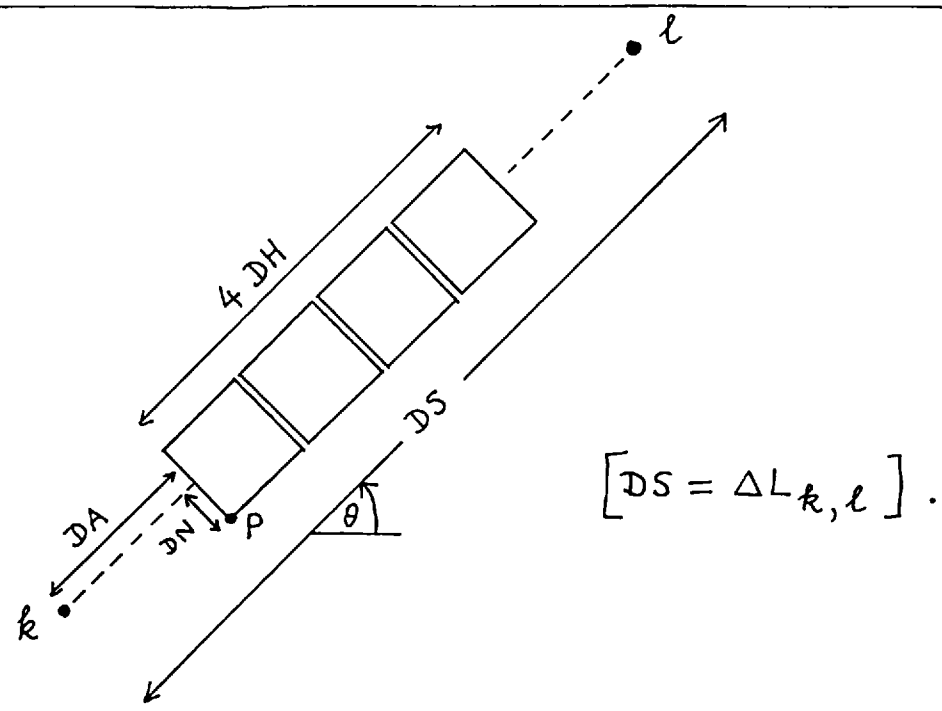


Figure 5.11: Enlarged diagram showing the insertion of a 4-character label between the points k and ℓ.

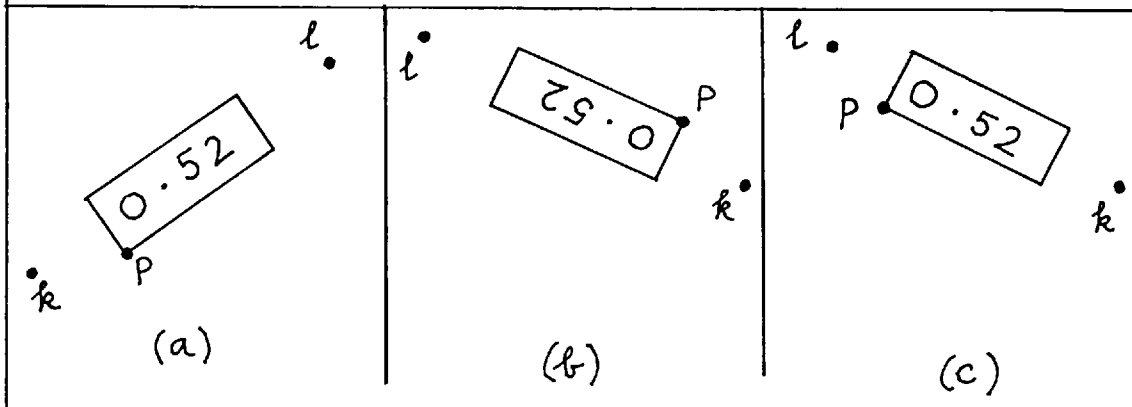


Figure 5.12: Possible orientations of the label: (a) correct, (b) upside down, (c) correct version of (b).

From these considerations, it follows that the angle of inclination θ_D is given by:

$$\theta_D = \begin{bmatrix} \theta \\ \text{or} \\ 360-\theta \end{bmatrix} \text{ according as } \left[\bar{x}_k \lessgtr \bar{x}_\ell\right], \tag{5.15}$$

where

$$\theta = \tan^{-1}\left[\left|\frac{\bar{y}_\ell - \bar{y}_k}{\bar{x}_\ell - \bar{x}_k}\right|\right] . \tag{5.16}$$

Let O denote whichever of the two points, k or l, that is closer to P. From Figure (5.11) we see that the point P may be reached from O, first by moving a distance "DA" in the direction θ_D, followed by a further distance "DN" normal to it $(\theta_D - 90)$, where :

$$\left.\begin{aligned} DA &= \tfrac{1}{2}(DS - 4DH), \text{ and} \\ DN &= \tfrac{1}{2}DH . \end{aligned}\right\} \tag{5.17}$$

The co-ordinates of P $(\bar{x}_p, \bar{y}_p)$ are therefore given (in matrix form) by:

$$\begin{bmatrix} \bar{x}_P \\ \bar{y}_P \end{bmatrix} = \begin{bmatrix} \bar{x}_o \\ \bar{y}_o \end{bmatrix} + \begin{bmatrix} \cos\theta_D & \sin\theta_D \\ \sin\theta_D & -\cos\theta_D \end{bmatrix} \begin{bmatrix} DA \\ DN \end{bmatrix} . \tag{5.18}$$

These newly computed values of $(\bar{x}_p, \bar{y}_p)$ and θ_D, together with the scaled co-ordinates of the points which constitute the two halves of the curve, enable the plotter to draw and label the contour, as desired. These operations are performed by subroutines CONTUR and CLABEL, for each contour loop recorded on TAPE 3.

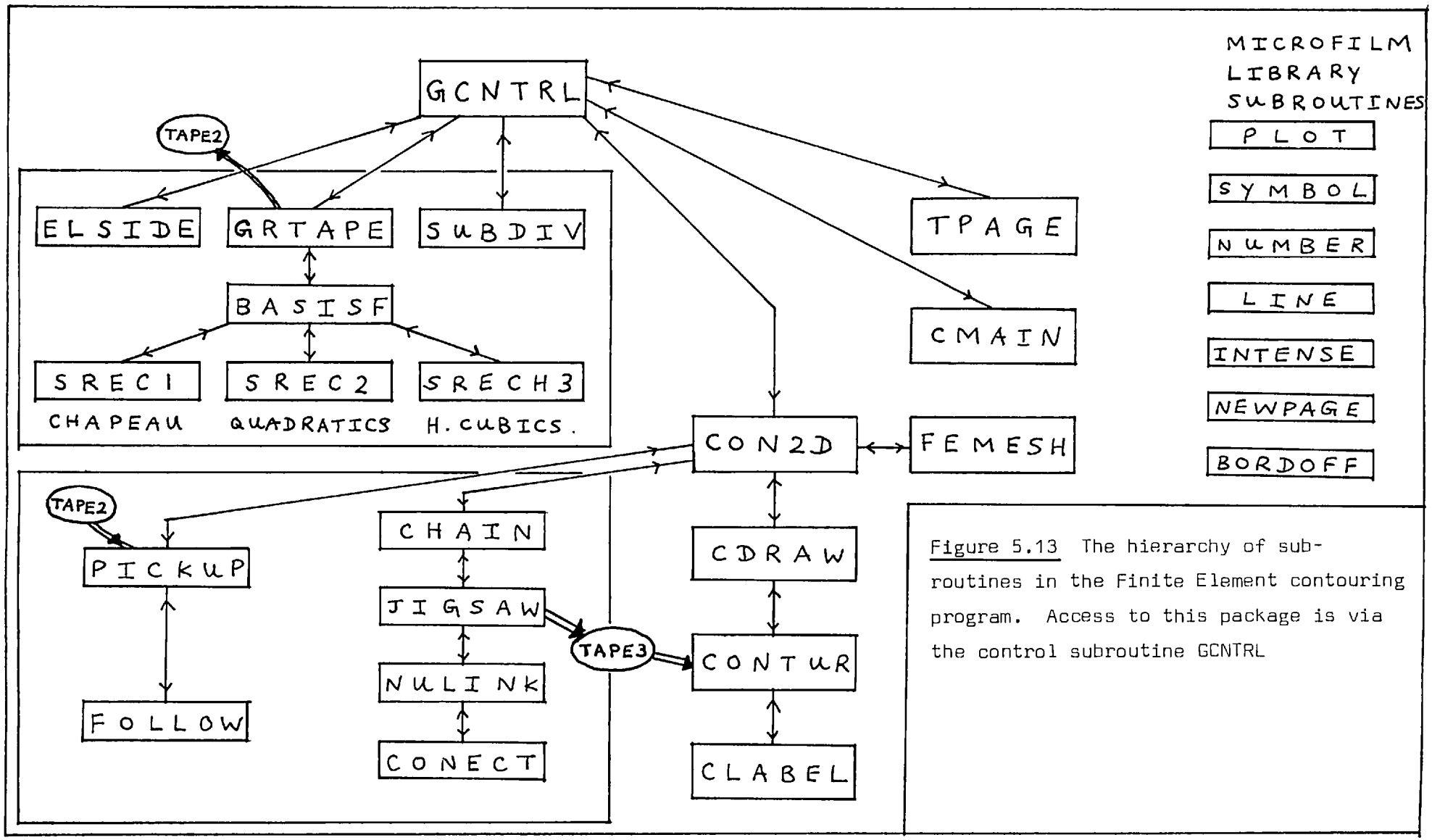


Figure 5.13 The hierarchy of sub-routines in the Finite Element contouring program. Access to this package is via the control subroutine GCNTRL

5.6. Concluding remarks

This contouring package was designed for use in conjunction with the Microfilm Graphics facility available at Imperial College. This combination enables contour maps to be produced on 35mm film, from which slides and/or hard copies can subsequently be made.

The analysis presented in this chapter, however, is practically independent of the plotting installation. The three most important "system subroutines" which perform the operations of "plotting" and "printing" are : PLOT, SYMBOL and NUMBER. Besides these "basic" ones, several auxiliary subroutines are available in the Imperial College graphics libraries, as described in references [26], [27]. The following are the chief advantages of the present model:

(1) it does not introduce any artificial smoothing ;

(2) by increasing n_0 (i.e. the fineness of the secondary mesh) it is possible to pick out (accurately) oscillations, irregularities etc., however small or localised they may be; and

(3) the model is applicable to finite difference results.

Figure 5.13 presents all the subroutines which make up this package, in the order of their importance.

CHAPTER 6

Discussion of Finite Element Results

6.1 Review of past work

The use of finite elements in reservoir engineering problems has attracted interest only in the past few years. Even so, a considerable amount of research remains to be done before the method can be brought into widespread use in the oil industry, or to the level of popularity that is enjoyed by finite difference simulators. Research, in the initial stages, however, has to be directed towards understanding the characteristic features of this new method, so that suitable techniques may be devised for controlling or eliminating any undesirable features as may arise.

6.1.1 Single phase problems

The earliest applications of the F.E.M. were concerned with single phase flow, and the ensuing results were found to be competitive with, if not superior to, those obtained from F.D. models. Javandal and Witherspoon [33] used linear (Chapeau) triangles to solve for the pressure distribution in a simple network. Cavendish et al [9] presented a generalised treatment of the Galerkin technique as applied to tensor product rectangles, and showed how even the simplest of these - namely, the bilinear (Chapeau) element - was capable of yielding an accurate solution for the pressure. The main problem which they considered was the solution of the Laplace equation within a rectangular domain containing point sources and sinks. Their theory was designed to introduce the class of tensor product hermitian elements, in particular the cubic and quintic types, and it also included a rigorous analysis of

the error-bounds associated with these approximations. Dogru et al [20] have successfully demonstrated the use of spline functions in transient single phase problems. More recently, Goldwater et al [28] have developed a full scale gas reservoir simulator based on the 6-node quadratic triangular element (see Figure 4.5c) which has been employed in the simulation of some of the major gas fields in the Southern North Sea. All the available evidence suggests (as has also been borne out by the present author's work), that F.E.M. is well suited to the simulation of single phase flow. The reason for this may lie partly in the fact that the character of the differential equation for pressure is either elliptic(for incompressible steady-state flow), or parabolic (for compressible flow). When more than one fluid is present, however, these observations no longer apply.

6.1.2 Two phase simulation

Finite element methods have also been applied to two phase immiscible displacement studies. The results and conclusions that have been reached appear to be author dependent.

Douglas et al [22] were among the first to simulate 1D water/oil displacement by finite elements. Using splines as their approximating polynomials , they concluded from their results that, in terms of computational effort, the Galerkin method was more efficient than finite differences. However, an examination of their results also shows that the saturation profile is quite smooth when it is spatially diffused, but that it is somewhat oscillatory when it is steeper and more piston-like. This phenomenon (as will be demonstrated further in this thesis) appears to be

a characteristic feature of two phase simulation by finite elements.

The results of Douglas et al were matched by those of Lewis et al [38] who used the 1D quadratic (3 node) element on an identical set of data. These (latter) authors also reported a radial (coning) model in which the bilinear (Chapeau) rectangle was employed. The graphs showing "cumulative oil recovery vs time" indicated a close match between the finite element results and the actual field performance. However, no saturation contour maps were shown by way of illustrating the distribution of water near the well. Yet another coning model, based on an alternating direction Galerkin procedure, was described by Kebaili and Thomas [35] who showed the dependance of cone formation on such well-known parameters as density difference and production rate. These authors represented the cone by a single curved interface separating the oil zone from the bottom water. However, as in the previous case, saturation contours were not shown.

2D Cartesian simulators have also been constructed by several authors using finite elements. McMichael and Thomas [39] reported the results obtained from their 2D, 3 phase model with bilinear elements. The potential- and saturation-maps shown here are smooth and qualitatively correct, despite the difficulty of assessing the accuracy of the numerical solutions due to the unavailability of analytical results. One of the most noteworthy papers in this field is perhaps that of Spivak et al [54] who applied the Hermite cubic element to 1D and 2D immiscible displacements. This work was supplemented by a parallel paper by Settari et al [50] concerning the simulation of miscible displacements, and the initiation of frontal instabilities (such as viscous fingering) due to the presence of reservoir heterogeneities. The saturation profiles and concentration maps presented in these papers were very encouraging,

and demonstrated the ability of the finite element method to track sharp fronts. However, there is reason to believe that some of the conclusions stated are based on inadequate evidence, and are therefore not universally valid. A case in point is the claim that the simulation of immiscible flow by finite elements does not require any artifice whatsoever (e.g. upstream weighting). In many of the papers, the eagerness of the authors to popularise the method has led to some of its limitations being either overlooked or bypassed.

A few other authors, however, have adopted a more cautious and pragmatic approach in their presentations. While commending the versatility of the finite element method in general, they have also identified some genuine difficulties which cannot be overlooked when making categoric statements about the advantages, or otherwise, of this technique. Significant among these are Dalen [16,17], Langsrud [36] and Faust and Mercer [24]. Dalen's work was concerned with the simulation of a 1D displacement using Chapeau elements, and of a 2D system consisting of Chapeau triangles (e.g. Figure 4.5b). Langsrud's paper was confined to the 1D problem, and showed results obtained with Chapeau and Hermite cubics elements, the test data being taken from an earlier paper by Nolen and Berry [44]. Faust and Mercer were engaged in a 1D simulation (using Chapeau elements) of a geothermal reservoir, where the two phases were taken to be steam and water. These papers collectively demonstrated that the classical Galerkin formulation, by itself, was inadequate to control the numerical behaviour of saturations. Dalen and Langsrud both concluded that an upstream weighting scheme was necessary to prevent saturation overshoots, and that, unless the capacity matrix was modified by some process such as lumping, the saturations were liable to exhibit violent oscillations as the flood front moved

through the reservoir.

It is apparent, therefore, that considerable differences of opinion exist among the various authors in this field, regarding the suitability of applying finite elements to waterflood calculations. The greater part of the present project has been devoted to exploring the sensitivity of the finite element solution to several factors, the most important of these being the mobility data (i.e. the viscosity ratio and the relative permeability curves of the two fluids). In subsequent sections we shall present the results of this thesis, discuss the problem areas in the light of these results, and, where possible, suggest means for overcoming these.

6.2 Data Sets

The 1D finite element models used in this thesis were applied to four different sets of mobility data. These vary from a largely unfavourable mobility ratio at one end of the scale, to a favourable, near piston-like displacement at the other, and were consequently thought to be representative of the wide spectrum of mobility data encountered in practice. These data sets, together with the sources from which they are taken, are given in Appendix 1. Figure 6.1 shows the relative permeability curves for these data sets, while Figure 6.2 illustrates the corresponding fractional mobility curves fw (which are equivalent to the fractional flow fw^* when capillarity and gravity are ignored). Figure 6.3 gives the analytical saturation profiles at t=300 days, based on the non-capillary Buckley-Leverett theory which was derived in Chapter 3.

Data Set 1 yields a (df/dSw) curve which is linear in Sw, and

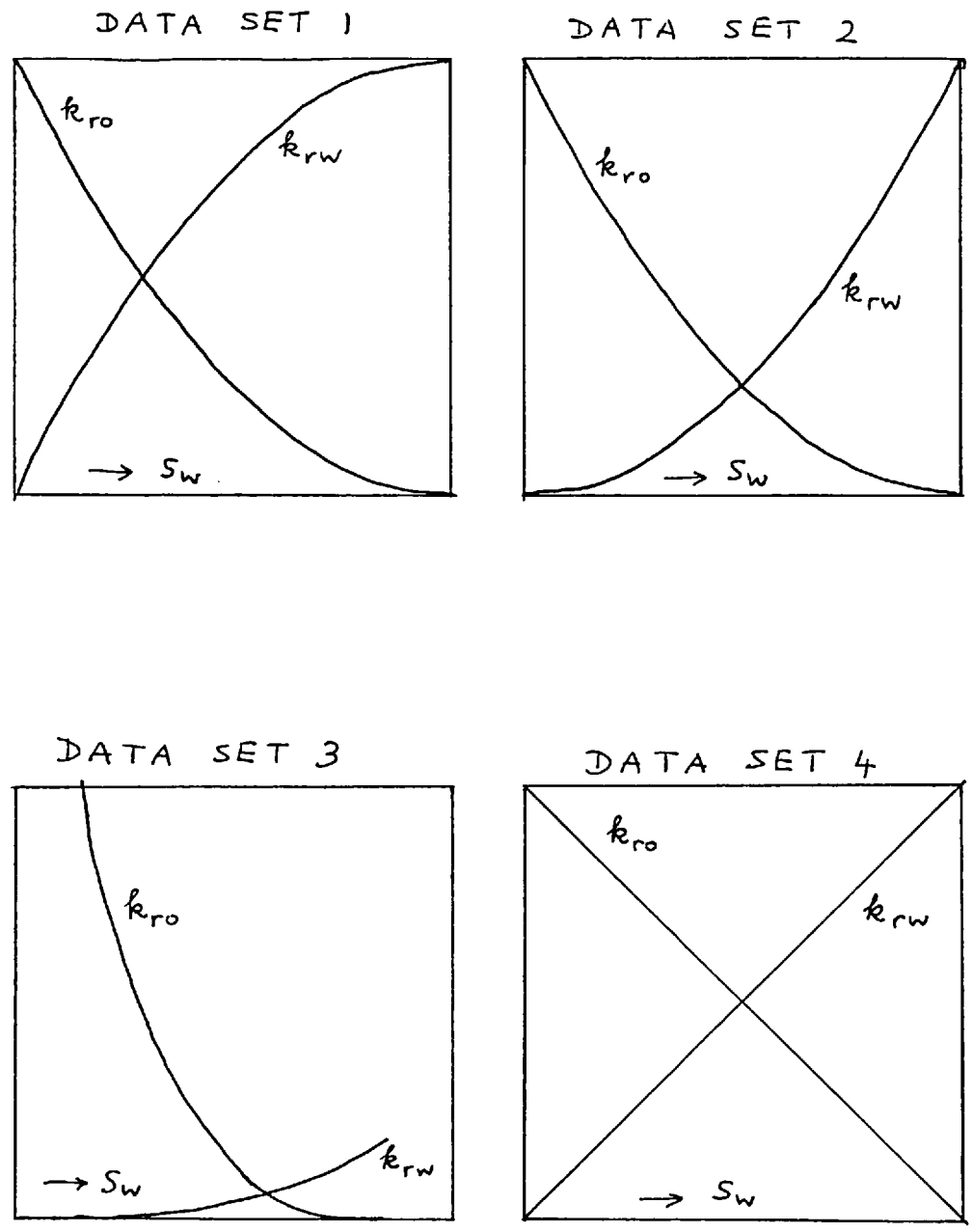


Figure 6.1: Relative permeability curves corresponding to the four data sets.

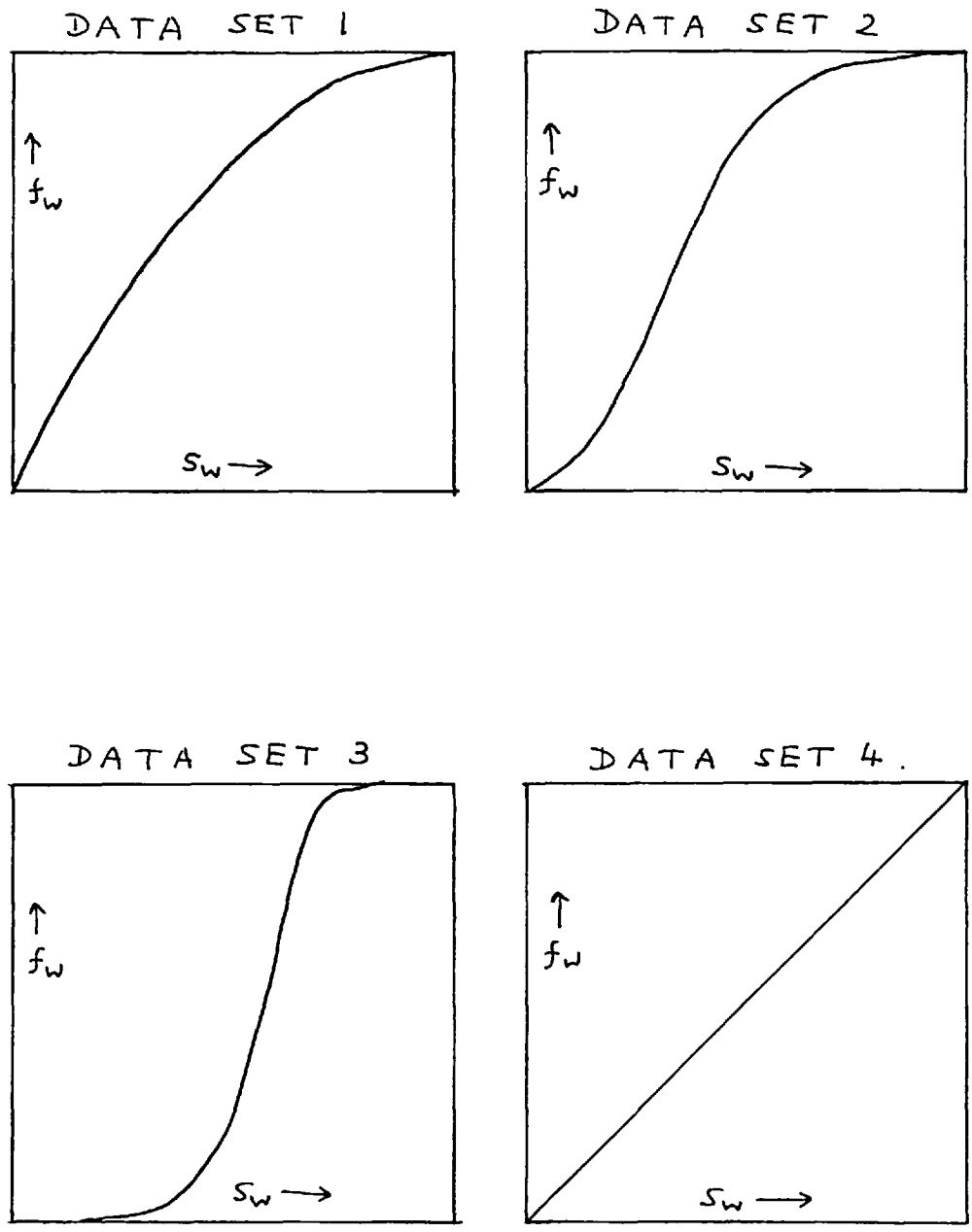


Figure 6.2: f_w curves corresponding to the four data sets.

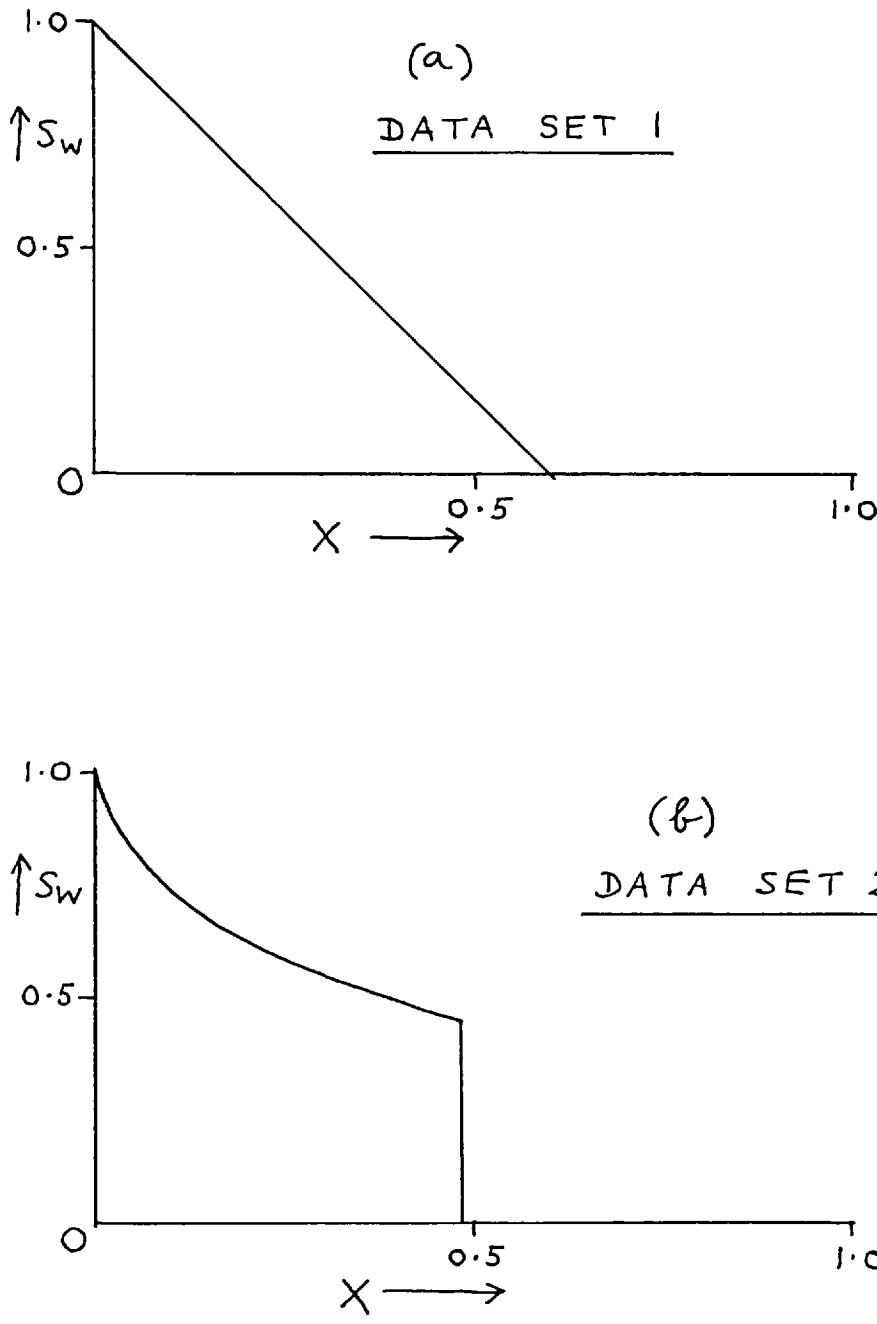


Figure 6.3a,b: Saturation profiles (at t = 300 days) corresponding to data sets 1 and 2.

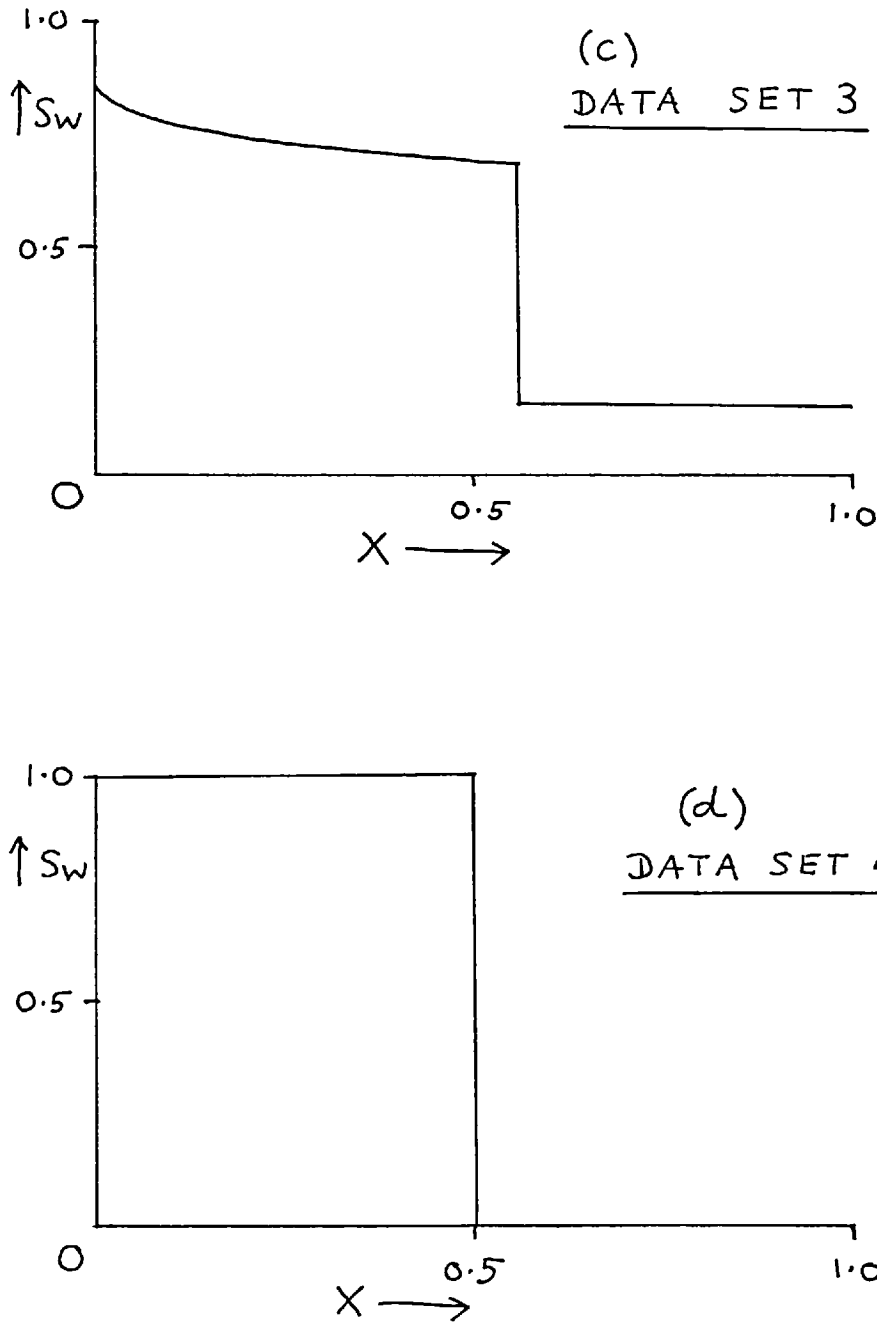


Figure 6.3c,d: Saturation profiles (at t = 300 days) corresponding to data sets 3 and 4.

consequently the analytical saturation profile shows a straight-line variation with distance. Due to the absence of a "front", this profile corresponds, in reservoir engineering terms, to an unfavourable displacement process. In Data Set 2, the quadratic relative permeability curves, together with an oil-to-water viscosity ratio of 4:1, produce an fw curve which is mildly S-shaped, thus representing a moderately favourable displacement, with a frontal saturation of 0.44. Data Set 3 is characteristic of a strongly water-wet system, where the relative permeability to water stays in a very low range. Furthermore, since the viscosities are equal, they do not counteract the effect of the relative permeability curves, and hence the resulting fw curve is found to be strongly S-shaped. The frontal saturation of 0.69 is very close to the residual oil end of the S_w scale, and the displacement is therefore almost piston-like, and very favourable. Data Set 4 consists of straight-line relative permeabilities and equal viscosities, and was so chosen in order to generate an fw curve which is linear in Sw. The constant value of (dfw/dSw) implies that all saturation (from S_w=0 to S_w=1) move forward with the same velocity, thus producing a perfect piston-type displacement. In theory, this is analogous to a miscible flood.

6.3 Factors influencing the finite element solution

As shown by Wilson and Casinader [59], it is possible, from 1 dimensional test runs, to study the effect of several factors on the numerical solution. In this thesis, the saturation profiles for each run are presented in triplets - a,b and c- which denote the Chapeau, Quadratics and Hermite Cubics respectively. All the 1D runs, unless otherwise stated, have been performed with 20 elements, and a maximum permissible time-step size of 0.5 day, and the graphs are plotted at

t = 300 days (i.e. when the saturation front has advanced approximately half way through the reservoir). Also, for the sake of comparison, the Buckley-Leverett solution is superimposed on the finite element solution.

6.3.1 Grid size

When analysing the influence (on the results) of any one parameter, it is customary to keep all other parameters constant. To study the effect of grid size, the data set chosen was that of Spivak et al [54] (i.e. data set 2), together with a constant capillary pressure gradient of -10psi. The reason for choosing such a capillary pressure is that (as will be shown later) the numerical solution converges to the analytical solution, as the number of elements is increased. Consequently, we may state that, if the number of elements is varied, while the data set is kept fixed, then, any changes in the quality of the numerical solution should reflect the effect of the grid size.

An argument which has often been advanced in favour of using high order finite elements (such as hermite cubics)is that it enables good accuracy to be achieved with relatively few elements. An examination of Figures G1a, b,c shows, however, that this is not necessarily the case. For although the finite element solution has a continuous variation within each element, the choice of too coarse a mesh (e.g. 3 elements) can result in a significant deviation from the true solution. This may be viewed as an attempt to stretch the capability of the finite element method beyond its limits. As the number of elements is increased, however, the quality of the results also improves, as expected, [Figures G.2a, b,c; G.3a,b,c] and beyond, say, 15 elements,

very little improvement is achieved by further grid refinement. These are obviously trivial observations for a 1D problem, but their implications cannot be ignored in a 2D situation. Clearly, if a 3-element grid yields unsatisfactory results on a 1D model, it is difficult to expect a (3x3) grid (representing, in the case of Hermite cubics, a total of 64 unknowns) to produce good quality results on a 2D model. Since the assembly of the finite element matrices and the solution of the relevant equations require substantial amounts of computer time, the apparent difficulties of using coarse meshes may become a severe limiting factor in 2D finite element analysis.

Figures G.1 through G.3 also reveal some additional features of the different element types. For example, with a given data set, the Hermite cubics appear to be the elements which are most prone to oscillations. The amplitude of these oscillations can be seen to decrease as the number of elements is increased. Furthermore, in the Herimte Cubics runs, amidst the peaks and the troughs which characterise the oscillatory profile, the nodal values (shown by circles) appear to fall close to the analytical solution. In the case of the quadratics, however, where each element has 3 nodes, the maxima and minima tend to occur at the middle node, and moreover, it is the values at these middle nodes that lie closest to the analytical solution.

A general feature in all these figures is that, when the displacement process involves the propagation of a front (such as in this data set), the oscillations that are set up in the numerical solution exhibit a definite spatial arrangement, as follows:

(a) the waves are approximately symmetric about the analytical solution; and

(b) the amplitude of these waves is small at the injection end, and increases progressively in the downstream direction, reaching a maximum just behind the flood front.

This pre-frontal hump is very characteristic of finite element solutions, and may be regarded as an effort made by the continuous (numerical) solution to match, as closely as possible, the discontinuous (Buckley-Leverett) solution. The properties of the wave-train in the unflooded region are simply a reverse of those in the flooded region. Thus, beginning with a fairly large amplitude just ahead of the front, the waves gradually die out in the direction of flow, oscillating, all the while, about the connate water saturation.

6.3.2 Capacity Lumping

Although solutions which exhibit spatial oscillations are mathematically valid, yet, from a physical standpoint they are unacceptable. This difficulty becomes even more serious when it is recalled that the relative permeabilities, which form the basis of all two-phase flow, are themselves strongly dependent on the saturations. Large undulations ahead of the front, such as seen, for example, in Figure G.1a, are very undesirable, since they cover a significant part of the flow domain with alternating positive and negative saturations. Some form of smoothing is therefore desirable, if not essential, in order to obtain physically meaningful solutions. We have already seen in the previous section that increasing the number of elements provides one means of dampening the oscillations. If, however, the required smoothing is to be achieved even on coarse meshes, then some other

practical device must be sought. Two alternatives are here available :

(1) to artificially smooth an oscillatory profile at the end of each time-step, in such a way as to preserve a material balance; or

(2) to incorporate within the finite element formulation, a standard artifice which would ensure a smooth saturation profile.

Since the first option has a certain degree of subjectiveness associated with it, the latter approach seems preferable. Thus, in the present study, the well-known concept of "Capacity Lumping" has been employed as a means of controlling the oscillations. This is a technique which transforms the "element capacity matrix" $\underline{C}_e$ (cf. equation 4.69) into a diagonal matrix. Physically, the process of lumping is analogous to replacing the "continuous" mass of an element by a set of "discrete" masses concentrated at its nodes. For Chapeau and Quadratics elements, the diagonalization of the matrix is achieved by adding all the off-diagonal terms into the diagonal ones, and then clearing the remainder of the array. For Hermite cubics, however, the degrees of freedom (as discussed in section 4.6.3) have different dimensionalities, and hence, following the method of Langsrud [36], a partial lumping scheme is used, which reduces the matrix to a block-diagonal form instead of making it strictly diagonal. Table 6.1 gives the element capacity matrices ($\underline{Ce}$) (in 1D), for the three types of elements, both before, and after, lumping. Clearly, the symmetry of the matrices is preserved during lumping.

Figures G.4a, b.c; G.5a,b.c and G.6a,b.c illustrate the saturation profiles obtained by repeating the earlier runs (shown in

Figures G.1 through G.3) with a lumped capacity matrix. This comparison shows the degree of smoothing which can result from using lumped matrices. The effect is most pronounced for the Hermite cubics, while the Quadratics seem to be the least affected, and the Chapeau fall between the two. However, despite the benefits of oscillation - removal, capacity lumping possesses a disadvantage in that the saturation profile becomes more smeared than in the absence of lumping. This, again, is particularly true of the Hermite cubics and the Chapeau models. We therefore have two options:

(a) either to leave the capacity matrix unlumped, and obtain an oscillatory solution which, nevertheless, shows a good frontal representation,

(b) or to lump the capacity matrix and obtain a smooth, though somewhat "diffused", saturation profile.

Numerical stability can be an important factor in deciding between these two alternatives. For example, in 2 Dimensions, it was found (in the present study) that the oscillations caused by the unlumped matrix on a 3x3 grid were violent enough to undermine the stability of the model. Consequently, it is the author's opinion that lumping could be even more indispensable in 2D models than in their 1D counterparts. All the 2D finite element runs reported in this thesis are based on the lumped matrices. The coefficients of these lumped 2D matrices can be obtained by a direct tensor-product extension of the corresponding 1D matrices given in Table 6.1.

(a) CHAPEAU	(b) QUADRATICS	(c) HERMITE CUBICS
BEFORE LUMPING		
$\frac{1}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$	$\frac{1}{30}\begin{bmatrix} 4 & 2 & -1 \\ 2 & 16 & 2 \\ -1 & 2 & 4 \end{bmatrix}$	$\frac{1}{420}\left[\begin{array}{cc\|cc} 156 & 22L & 54 & -13L \\ 22L & 4L^2 & 13L & -3L^2 \\ \hline 54 & 13L & 156 & -22L \\ -13L & -3L^2 & -22L & 4L^2 \end{array}\right]$
AFTER LUMPING		
$\frac{1}{6}\begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$	$\frac{1}{30}\begin{bmatrix} 5 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 5 \end{bmatrix}$	$\frac{1}{420}\left[\begin{array}{cc\|cc} 210 & 35L & 0 & 0 \\ 35L & 7L^2 & 0 & 0 \\ \hline 0 & 0 & 210 & -35L \\ 0 & 0 & -35L & 7L^2 \end{array}\right]$

TABLE 6.1 1D element capacity matrices before, and after, lumping.

$$\left[\underline{C}_e = \int_{\Omega_e} \underline{N}_e^T \underline{N}_e \, d\Omega_e .\right]$$

6.3.3 Mobility data

We shall now consider how the character of the mobility data influences the accuracy of the finite element solution. In the absence of capillary and gravity effects, the Buckley-Leverett theory postulates (as shown in Chapter 3) that the displacement is characterised by a saturation "front", the existence and magnitude of which are determined by the shape of the f_w curve.

In general, the greater the upward concavity (i.e. S-shape) of the curve, the higher the value of the frontal saturation, and hence, the more piston-like is the displacement. Such a curve generally corresponds to a case where the relative mobility of the water is very small at low water saturations, thereby ensuring that much of the oil is expelled from the pores of the rock before the water can flow in appreciable quantities. This , therefore, represents a favourable displacement.

In so far as the stability of the numerical solution is concerned, however, the opposite trend is observed, and the greater the upward concavity of the f_w curve, the less satisfactory do the results become. This is because the highly impeded flow of water at low saturations causes the saturations to build up to values far greater than those predicted by the Buckley-Leverett theory. At the early stages of the displacement, the regions near the injector are worst affected, but, as the simulation proceeds, this erroneous solution spreads throughout the domain. Since simulators maintain a material balance, these saturation overshoots cause the "numerical front" to lag behind the

"analytical front", and thus yield an erroneously higher oil recovery at breakthrough.

We shall now use our different data sets to substantiate the foregoing arguments. Attention will be focussed on the overall convergence of the finite element solution, rather than on the localised oscillations and near-frontal behaviour, which we have discussed in the previous sections.

We begin with the graphs G.7a,b,c, which are based on data set 1. This data set, as we have seen, is designed to give an f_w curve which is always convex upwards, and the resulting linear saturation profile demonstrates the unfavourable nature of the displacement. The numerical results, in this case, are seen to be in very good agreement with theory for all three element types. A conspicuous hump occurs near the foot of the saturation profile in the Hermite cubics and quadratics runs. This, however, can be reduced by lumping the capacity matrix.

We next examine the results of using data set 2, which is also the one used by Spivak et al [54]. For this "moderately favourable" displacement, we see that the finite element solutions (Figures G.8a,b,c) coincide with the analytical solution at high water saturations, but that, as the front is approached, they begin to depart from it by a noticable extent. The match between the numerical and analytical profiles is thus poor.

Our third data set is that of Langsrud [36], which can be presented as an example of extreme numerical degeneracy. The pronounced upward concavity of the f_w curve results in a front saturation of 0.69, which, allowing for irreducible saturations of 0.15 at either end (i.e.

Swc=Sor=0.15), implies that the band of frontal saturations (0.15 to 0.69) occupies 77% of the mobile saturation range (Figure 6.3c). The numerical solutions for this "favourable" displacement are found to be very bad (Figures G.9a,b,c), and show:

(1) a progressive rise in S_w, well above $(1-S_{or})$; and

(2) the consequent "retardation" of the "numerical front".

All the runs considered in this section (namely, Figures G.7 through G.9) have been performed without the aid of any artifice, and hence, the effects we have observed above are purely the reactions of the finite element method to the various mobility data. We are thus led to the conclusion that, so long as conventional relative permeability curves are used, the finite element method cannot be relied upon to give good results. This difficulty necessitates the incorporation of a suitable artifice, whose nature and extent of usage will have to be dictated by the character of the mobility data.

Pseudo capillary pressure and upstream relative permeabilities are two such "numerical remedies" currently in use. Indeed, any device which preferentially aids the flow of water relative to the oil will be a potential candidate for this purpose.

6.3.4 Pseudo Capillary Pressure

The role of true capillary pressure in porous media is to provide an additional ("imbibing") pressure gradient for the water phase. By contrast, the role of pseudo capillary pressure in simulators

is to correct an erroneous solution to the analytical solution. This obviously means that the effect of real capillary pressure can be simulated on a computer, only if the numerical solution converges to the Buckley-Leverett solution for the non-capillary displacement problem. Even so, there is the further complication introduced by numerical dispersion which makes it difficult to isolate the dispersion due to genuine capillary effects.

In this section, we shall consider the use of pseudo capillary pressures, merely as a correcting device for finite element simulators. Spivak et al, in their paper [54], inserted a pseudo capillary pressure gradient (p_c') of - 10psi in order to obtain convergence between their finite element and Buckley-Leverett solutions. Thus, although they claimed that the finite element method does not require an artifice such as upstream weighting, they were simply using another artifice in its place, namely, pseudo capillary pressure gradients. Moreover, the value of the gradient must have been based on a trial-and-error determination, since the required gradient would be expected to vary from data set to data set. The effect of pseudo capillary pressure gradients will now be demonstrated on our data sets.

Since data set 1 does not require any artifice for convergence, the use of a pseudo capillary gradient of - 10psi only serves to over-correct an already good answer, and thereby smear the profile even further (Figures G.10a,b,c). This shows that correcting devices should only be used where necessary. However, looked at from another angle, this "diffused" numerical profile may, in fact, be the correct solution under a real capillary gradient of -10psi. In the latter case, of course, the analytical profile shown in these Figures no longer applies.

The effect of introducing pseudo gradients is best demonstrated on data set 2. Figures G.11a,b,c; G.12a,b,c; and G.13a,b,c illustrate the saturation profiles that were obtained with $Pc' = -5$, -10 and -20 psi, respectively, and these may be compared with the corresponding non-capillary solutions ($Pc'=0$) shown in Figures G.8a,b,c. It is evident that, as the slope Pc' is gradually increased, the saturation-overshoot diminishes, and the finite element solution improves in quality. At $Pc'=-10$, both the numerical and analytical solutions virtually coincide. Higher values of Pc' (such as -20 and higher) tend to overcorrect the solution, and cause the "numerical" front to marginally overtake the "true" front.

For data set 3, where the original discrepancy between the numerical and Buckley-Leverett solutions is very large (Figures G.9a, b,c), pseudo gradients of the above order of magnitude are found to be inadequate for convergence (Figures G.14a,b,c), although the results do show an improvement over the non-capillary solutions.

From the foregoing observations, we may conclude that the required pseudo capillary gradient is a function of the mobility data. However, this gradient is also found to be dependent on the total flow rate in the system. For example, in the case of data-set 2, when the flow rate is raised to 100 times the original value, the pseudo gradient of -10 psi (which was used successfully before) now becomes ineffective. (Figures G.15a, b,c). In a later section, we shall consider a method for obtaining an approximate estimate of this gradient, based on factors such as the mobility data and the flow rate.

6.3.5 Upstream weighting

An alternative tool which is powerful in controlling the numerical behaviour of saturations is the method of upstream weighting, which has for long been an essential component of finite difference models. Conceptually, it means simply that the fluid flow between two adjacent blocks is determined either by the relative permeability at the upstream block (single-point weighting [15a]), or by an extrapolated value of the relative permeabilities at the two previous upstream blocks (two-point scheme [55]). In its general form, the single point scheme can be expressed through a mobility weighting factor α (ranging from 0 to 1), as follows:

$$k_r = \alpha k_{r_{up}} + (1-\alpha) k_{r_{down}} \; ; \qquad (6.1)$$

and, depending on the choice of α, it can vary from mid-stream weighting ($\alpha = \frac{1}{2}$) to full upstream weighting ($\alpha = 1$).

In finite element formulations, upstream weighting schemes are not as straightforward as the above, since the saturation-dependent quantities are evaluated at several Gauss points within each element (and not merely at the block-centre). To apply upstream weighting in a 1D finite element model, for example, we need to assign, to each Gauss point, the relative permeabilities calculated at another point lying some distance δ upstream from it. The choice of δ has largely been arbitrary, and Langsrud [36], after experimenting with different values of δ, has observed that the optimum value for his runs was intermediate between mid-stream and fully upstream, and that acceptable finite element solutions were obtained for a wider range of δ than with finite differences.

Two upstream weighting schemes have been employed in the 1D models of this thesis:

Scheme 1 : uses the relative permeabilities at the nearest upstream Gauss-point; and

Scheme 2 : uses these parameters at the upstream node of the element.

The first of these schemes has been used successfully on data set 3, whose results now show a close resemblance to the Buckley-Leverett profile (Figures G.16a,b,c).

Scheme 2 has been used successfully on the Chapeau model (Figure G.17a), but has proved unsatisfactory with the Quadratics and Hermite cubics models (Figures G.17 b,c), where a meaningless solution has been obtained. One possible explanation of this is that the choice of the upstream node is not suitable for cases where internal (i.e. middle) nodes are present. This suggestion, however, is purely speculative.

Another feature that can be observed in these runs is that the Chapeau saturation profile based on the second scheme exhibits a slightly more "diffused" front than that based on the first. This is consistent with the fact that scheme 2 involves a greater degree of upstream weighting than scheme 1. Both schemes, however, are open to some general criticisms:

(1) As the upstream shift distance δ is controlled by the number of Gauss points and/or the size of the elements (both of which can be varied at random), the schemes lack a quantitative

backing, and fail to provide a correlation between fluid flow data and the artifice of upstream weighting.

(2) Their applicability in 2 dimensions is uncertain. Despite these shortcomings, the upstream weighting schemes are useful for 1D models and, as pointed out earlier, scheme 1 seems to be preferable to Scheme 2.

6.3.6 Modified f_w curve (Dalen [16a])

We have seen so far that, with conventional fractional flow curves, the behaviour of the finite element solution is unpredictable, and that, depending on the magnitude of the upward concavity of the curve, a suitable artifice (such as pseudo capillary gradient or upstream weighting) must be used to correct the numerical results. This is, of course, based on the long held assumption that such "S-shaped" fw curves, usually derived from laboratory core flood experiments, are correct. We shall now briefly examine the validity of this assumption, and consider the use of a modified fw curve.

Figure 6.4 is a schematic diagram showing three fractional flow curves, all of which yield the same Buckley-Leverett solution. Numerically, however, three such widely differing curves cannot be expected to produce identical results. This obvious inconsistency can only be resolved by conceding that the (lower) part of the f_w curve which is concave upwards is irrelevant to the solution. In chapter 3, we have already seen, in connection with Welge's construction, that the analytical solution does indeed ignore the lower part of the curve.

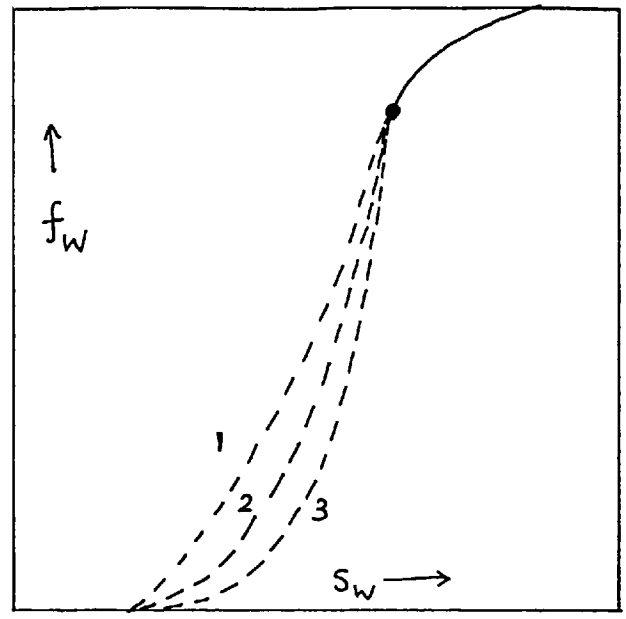


Figure 6.4: Three (hypothetical) f_w curves, which yield the same Buckley-Leverett solution.

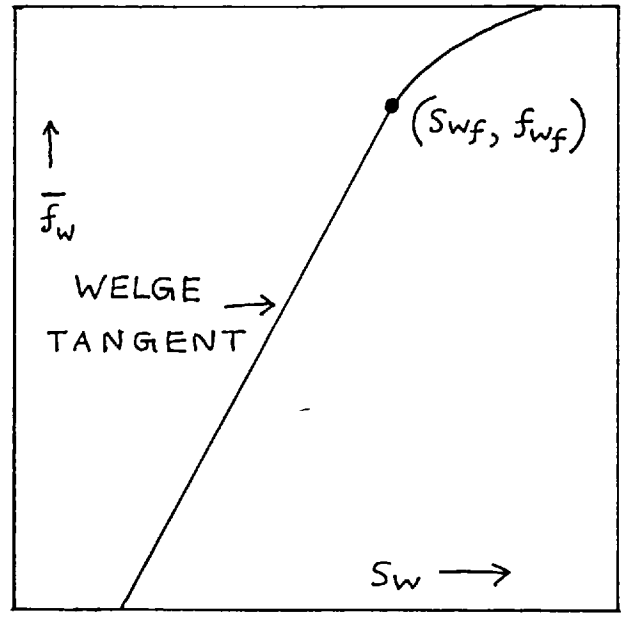


Figure 6.5: Modified ($\bar{f}_w$) curve, based on the Welge tangent.

It follows logically, therefore, that if the "S-shaped" portion cannot be used for analytical solutions, then its use is equally unacceptable in numerical models. This argument is further consolidated by Jones and Roszelle [34] who, from their experimental studies, have concluded that laboratory core flood data, if properly interpreted, will result in f_w curves which are never concave upward.

So, given a curve which has an upward concavity, the most plausible way to amend it is to replace it by:

(a) the Welge tangent from Swc to Swf; and

(b) the remainder of the original curve from Swf to (1- Sor).

(Figure 6.5). Thus :

$$\bar{f}_w = \frac{1}{1+(\bar{\lambda}_{ro}/\bar{\lambda}_{rw})} = \begin{cases} \dfrac{S_w - S_{wc}}{S_{wf} - S_{wc}} \cdot f_{wf} , & S_{wc} \leq S_w \leq S_{wf} \\ f_w , & S_{wf} \leq S_w \leq (1-S_{or}) \end{cases} \quad (6.5)$$

where the bars denote the modified curves. Clearly, if the original f_w curve does not show an upward concavity (as in data set 1), then the above modification would be unnecessary.

Since the relative permeabilities are input to the numerical model in the form of a table, we need to calculate the corresponding modified values ($\bar{k}_{ro}$ and $\bar{k}_{rw}$) below the flood front saturation. To do this, we require, at each saturation point on the table, a pair of simultaneous equations involving $\bar{k}_{ro}$ and $\bar{k}_{rw}$. One of these equations follows directly from equation (6.5):-

$$\left(\bar{\lambda}_{ro} / \bar{\lambda}_{rw}\right) = \frac{1}{\bar{f}_w} - 1 . \quad (6.6)$$

The second equation may be obtained by assuming a suitable relationship for the total relative mobility ($\overline{\lambda_{ro}} + \overline{\lambda_{rw}}$). For example, if these quantities are left unchanged from their original values, then the pressure distribution would be largely unaffected by the modification of the f_w curve. Thus, we may use:

$$\overline{\lambda_{ro}} + \overline{\lambda_{rw}} = \lambda_{ro} + \lambda_{rw} \text{ (original)} ; \qquad (6.7)$$

And so, from equations (6.6) and (6.7), we can obtain the modified relative permeabilities :

$$\overline{k_{rw}} = \mu_w \overline{\lambda_{rw}} = \mu_w (\lambda_{ro} + \lambda_{rw}) \overline{f_w} ; \qquad (6.8)$$

and $$\overline{k_{ro}} = \mu_o \overline{\lambda_{ro}} = \mu_o (\lambda_{ro} + \lambda_{rw})(1 - \overline{f_w}) . \qquad (6.9)$$

It is perhaps appropriate to comment on the shape of these modified relative permeability curves. We know that the modified $\overline{f}_w$ curve always increases monotonically with saturation. However, the total mobility ($\lambda_{ro} + \lambda_{rw}$) can decrease with S_w (and even undergo a minimum over part of the saturation range). Consequently, the product of these two curves can cause a lack of smoothness (i.e. irregularities) in the shapes of the modified $\overline{k}_{ro}$ and $\overline{k}_{rw}$ curves, as implied by equations (6.8) and (6.9). This phenomenon, however, does not affect the behaviour of the model, since the saturation distribution is ultimately governed by the f_w curve, and not merely by the individual relative permeabilities.

The above techniques for modifying the f_w curve, and for calculating an equivalent set of relative permeabilities, were originally proposed by Dalen [16a], and demonstrated on his finite element models.

These have also been applied to the different data sets used in the present study, as shown in the next section.

6.3.7 Velocity dependent upstream weighting

Parallel with the use of the modified $\bar{f}_w$ curve, an upstream weighting scheme, based directly on the Buckley-Leverett equation, was employed in the 1D runs of this thesis. Thus, if Q is the rate of injection in pore volumes per unit time, Δt is the current time-step, and L is the total length of the reservoir, then the upstream shift-distance δ is given by:

$$\delta = (\Delta t)\, Q\, L \left(\frac{d\bar{f}_w}{dS_w}\right). \qquad (6.10)$$

δ may be interpreted as the distance that will be traversed during the ensuing time-step by the saturation at the point in question. Hence, there is justification for using this formula, and it also removes, to some extent, the arbitrariness involved in the schemes adopted in section 6.3.5.

By combining:

(a) the modified $\bar{f}_w$ curve;

(b) the lumped capacity matrix; and

(c) the velocity dependent upstream weighting (equation 6.10), good agreement has been obtained between the finite element results and the theoretical saturation profiles for all four data sets considered in this work. (Figures G.18a,b,c; G.19a,b,c; G.20a, b,c; G.21a,b,c).

In order to assess the contribution of the modified $\bar{f}_w$ curve towards these successful results, we may examine the magnitude of the upstream shift implied by equation (6.10). Expressed as a fraction of the element length (Le), the maximum value which δ can take is:

$$(\delta_e)_{max} = \frac{\delta_{max}}{Le} = \frac{QL}{Le}(\Delta t)_{max}\left(\frac{d\bar{f}_w}{dS_w}\right)_{max} . \qquad (6.11)$$

Since the maximum time-step was fixed at 0.5 day, we may evaluate $(\delta_e)_{max}$ for the various data sets:

Data Set 1

$$(\delta_e)_{max} = \frac{0.001 \times 1000}{50} \times 0.5 \times 2 = \underline{0.020}$$

Data Set 2

$$(\delta_e)_{max} = \frac{0.001 \times 1000}{50} \times 0.5 \times 1.618 = \underline{0.016}$$

Data Set 3

$$(\delta_e)_{max} = \frac{0.000983 \times 2000}{100} \times 0.5 \times 1.721 = \underline{0.017}$$

Data Set 4

$$(\delta_e)_{max} = \frac{0.00167 \times 1000}{50} \times 0.5 \times 1.0 = \underline{0.017}$$

These calculations show that, due to the small time-steps involved, the maximum upstream shift does not exceed 2% of the element length, and is thus negligible. We may therefore say that the convergence of the finite element solutions to the corresponding Buckley-Leverett solutions is attributable, in large measure, to the use of the modified $\bar{f}_w$ curve.

6.3.8 Estimation of pseudo capillary gradients

In section 6.3.4 we saw how a pseudo capillary gradient of -10psi, determined by trial and error, proved useful in making the numerical solution match the Buckley-Leverett solution for data set 2, and how the

same gradient was inadequate to provide the necessary correction in the case of data set 3. Predicting the required value is obviously a complicated problem but, nevertheless, by using elementary two-phase theory, we can isolate the factors which control it, and thereby arrive at an estimate of its magnitude.

From Leverett's fractional flow equation (3.18), we know that the true water cut (fw*) of a two-phase stream (flowing horizontally) is given by:

$$f_w^* = f_w + \frac{k}{u_t} \frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}} \frac{\partial p_c}{\partial x} = f_w + \Delta f_w \; ; \qquad (6.12)$$

where f_w is the fractional mobility ($\lambda_{rw}/\lambda_{rt}$), and ($\partial p_c/\partial x$) is the spatial gradient of the capillary pressure.

If, on the basis of the observations made in the previous section, we suppose that the modified $\bar{f}_w$ curve is indeed the correct one to use in non-capillary simulations, then we may employ the pseudo-capillary term as a means of correcting the original (S-shaped) fractional mobility curve f_w, to the $\bar{f}_w$ curve. Equation (6.12) shows that, in order to obtain such a correction <u>exactly</u>, a separate value of ($\partial p_c/\partial x$) needs to be calculated at each saturation. We can, however, simplify the problem, and determine an "average" (though constant) value ($\partial p_c/\partial x$)$_{av}$ which would correct the f_w curve to $\bar{f}_w$ in an "average" sense. This means that, instead of requiring the condition:

$$f_w^* - \bar{f}_w = 0 \quad \text{at every } S_w \; , \qquad (6.13)$$

we use the integral equation:

$$\int_{S_{wc}}^{S_{wf}} (f_w^* - \bar{f}_w)\, dS_w = 0 , \qquad (6.14)$$

where the integration is taken over the frontal saturation range: S_{wc} to S_{wf}. Substituting equation (6.12) into (6.14), and bearing in mind that $(\partial p_c/\partial x)_{av}$ is now treated as a "constant" with respect to the integration, we obtain :

$$\int_{S_{wc}}^{S_{wf}} \left[f_w + \frac{k}{u_t} \frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}} \left(\frac{\partial p_c}{\partial x}\right)_{av} - \bar{f}_w \right] dS_w = 0 ; \qquad (6.15)$$

from which we deduce that :

$$\left(\frac{\partial p_c}{\partial x}\right)_{av} = \frac{u_t}{k} \frac{\int_{S_{wc}}^{S_{wf}} (\bar{f}_w - f_w)\, dS_w}{\int_{S_{wc}}^{S_{wf}} \left(\frac{\lambda_{ro}\lambda_{rw}}{\lambda_{rt}}\right) dS_w} , \qquad (6.16)$$

which, geometrically, is equivalent to the statement:

$$\left(\frac{\partial p_c}{\partial x}\right)_{av} = \frac{u_t}{k} \frac{\text{Area } A}{\text{Area } B} ; \qquad (6.17)$$

where A is the area enclosed between the original and modified f_w curves (Figure 6.6), and B is the area lying under the curve which represents half the harmonic mean of the relative mobilities (Figure 6.7), in the range S_{wc} to S_{wf}.

In simulators which are not affected excessively by numerical dispersion, it is possible for the saturation front to be confined (spatially) to a single element. So, if we express equation (6.17) in the units of "psi per element length", we have an estimate of the effective capillary pressure acting across the front. Thus:

$$\left(\frac{\partial p_c}{\partial x}\right)_{av} = 14.7 \left(\frac{u_t L_e}{k}\right) \frac{\text{Area } A}{\text{Area } B} \text{ psi/element}; \qquad (6.18)$$

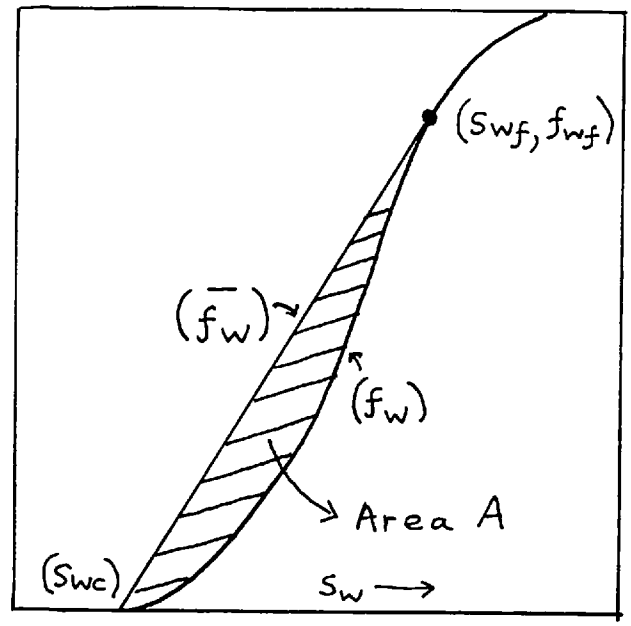


Figure 6.6: Calculation of "Area A", to determine the pseudo capillary gradient (equation 6.17).

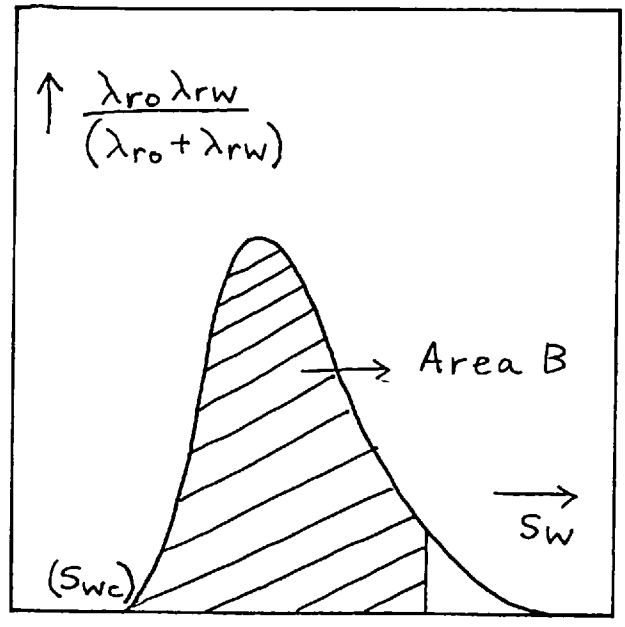


Figure 6.7: Calculation of "Area B" for the determination of the pseudo capillary gradient (equation 6.17).

where L_e is the element length, and U_t, L_e and K are in Darcy units. Due to the changing saturation gradients within the reservoir, it is difficult to relate the average spatial gradient $(\partial p_c/\partial x)_{av}$ given in equation (6.18) to the required pseudo capillary gradient $p_c' (= dp_c/dS_w)$. It can be said, at most, that $(\partial p_c/\partial x)_{av}$ is a measure of the required p_c'. We shall therefore stop at equation (6.18), and evaluate this quantity for our data sets, beginning with set 2.

Average pseudo capillary gradient for Data Set 2

U_t = 3.528x10^{-5} cm/S ; K = 0.158D

L_e = 50x30.48 cm; Area A = 0.0321 ; Area B = 0.0151

Hence,

$$\left(\frac{\partial p_c}{\partial x}\right)_{av} = 14.7\left[\frac{3.528\times10^{-5}\times 50\times 30.48}{0.158}\right]\left[\frac{0.0321}{0.0151}\right]$$

$$= 10.7 \text{ psi/element},$$

which turns out to be identical to the value of -10psi chosen for the pseudo capillary gradient P_c' in section 6.3.4.

Average pseudo capillary gradient for data set 3

U_t = 1.387x10^{-4} cm/s ; K = 0.500D;

L_e = 100x30.48 cm; Area A = 0.1160; Area B = 0.0177.

So, from equation (6.18),

$$\left(\frac{\partial p_c}{\partial x}\right)_{av} = 14.7\left[\frac{1.387\times10^{-4}\times 100\times 30.48}{0.5}\right]\left[\frac{0.1160}{0.0177}\right]$$

$$= 81.4 \text{ psi/element}.$$

It was observed earlier (from Figures G.14a,b,c) that a pseudo gradient of -10psi was insufficient to correct the finite element results for data set 3. These runs were repeated with $p_c' = -80$ psi, and Figures G.22a,b,c show the convergence brought about by this pseudo gradient.

Data Sets 1 and 4

For these data sets, the modified fractional flow curves ($\bar{f}_w$) are identical to the original f_w curves. Thus, Area A=0, and consequently, from equation (6.18), $(\partial p_c/\partial x)_{av}$ is also zero. Our theory implies, therefore, that pseudo capillary gradients are unnecessary for these two data sets. We have already demonstrated this for data set 1 (See Figures G.7a,b,c). The results for data set 4 (Figures G.23a,b,c) are also, on the whole, in accordance with this hypothesis, although the saturation profile does exhibit a slight departure from the analytical solution , and some localised oscillations.

It is also interesting to note that both these data sets require no pseudo gradients, despite the fact that they lie at opposite ends of the displacement-spectrum — the one representing an unfavourable displacement, while the other corresponds to a perfect, piston-like flood. It may be inferred, therefore, that the requirement of a pseudo capillary gradient is not dictated solely by the efficiency of the displacement process. It depends, instead, (as has been shown) on the internal structure of the mobility data- i.e. the shapes of the original f_w curve, and the harmonic mean curve of the relative mobilities.

To show that the equation proposed for pseudo capillary pressure was effective in a variety of conditions, the flow rate and

number of elements have been varied.

Using data set 2, for example, it has already been shown in section 6.3.4 that $P_c' = -10$ psi loses its effectiveness at higher flow rates. Indeed, at a flow rate of x 100, a P_c' of - 1000 psi should be required, and this has been found to be true (Figures G.24a,b,c). (It should be noted that rates of x 100 are being used for comparative purposes only, and would not exist in practice).

Equation (6.18) also says that P_c' should be proportional to the element length, so that, for a fixed reservoir length, the gradient is inversely proportional to the number of elements. With data set 3, Figures G22a,b,c illustrated that - 80 psi was required for 20 elements. Figures G25a,b,c show that - 80 psi is not adequate with 10 elements, whereas - 160 psi gives good results (Figures G26a,b,c).

These results prove the general applicability of equation (6.18) in the prediction of pseudo capillary pressure gradients for very different data sets. It is immediately apparent, however, that the p_c' value required for satisfactory convergence is, in many cases, so high that any real capillary effects would be masked.

In concluding the discussion on the 1D results, we may state that there are two possible ways of overcoming the problem caused by the sensitivity of the finite element method to the mobility data.

(1) To use the modified $\bar{f}_w$ curve, together with the lumped matrix and theoretical upstream weighting;

(2) to use the original f_w curve, and either :

a) upstream Gauss-point relative permeabilities; or

b) a pseudo capillary pressure gradient, estimated from equation (6.18).

6.4 2 Dimensional Runs

The two dimensional tensor product models were tested on a quarter five-spot domain, whose properties were as follows: (Spivak et al [54].)

Area	= 1000ft x 1000ft
Thickness	= 1ft
Flow rate	= 20ft^3/day (=0.0002 pore vol/day)
Initial pressure	= 6000psi.
Fluid compressibility	= 0.

The mobility data was taken from data set 2, and the runs were performed with the following additional features:

(a) modified f_w curve;

(b) lumped capacity matrix; and

(c) boundary conditions, as given in section (4.9).

Neither upstream weighting nor a pseudo capillary gradient was used, and the maximum time-step was restricted to 30 days. The domain was uniformly divided into a (5x5) diagonal element grid, so that the number of nodes and degrees of freedom involved in each of the three cases was as follows:

Chapeau	36 nodes	36 DOF's
Quadratics	121 nodes	121 DDF's
Hermite cubics	36 nodes	144 DDF's

6.5 Pressure distribution

In the early stages of the displacement, the five-spot domain is almost entirely filled with oil, and thus the pressure distribution will be very similar to that for single phase flow. We can, therefore, compare the numerical results with the steady-state analytical solution of Muskat (derived in chapter 3), in order to assess the accuracy of the finite element approximation.

To calculate the analytical solution, the two infinite series which occur in equation (3.68) were truncated after 6 terms. Increasing the number of summation terms to 10 was found to produce results which matched the 6-term solution to three decimal places (10^{-3}psi), thereby indicating the rapid rate of convergence of this logarithmic series.

By virtue of the symmetry, the analytical solution needs to be computed only at points lying within a quarter of the domain (the area IOB shown in Figure 6.8). This partial solution can then be extended to the remaining areas by taking mirror-images on the appropriate diagonals. For example, let us consider the points 1,2,3 and 4 (Figure6.8), where 2 is the reflection of 1 on the leading diagonal IOP, while 3 and 4 are the reflections of 1 and 2, respectively, on the median diagonal AOB. If, therefore, p_1 is calculated from the series, we can write down the remaining pressures directly:

$$\begin{cases} P_2 = P_1 \ ; \quad \text{and} \\ P_3 = P_4 = 2P_i - P_1 \ ; \end{cases}$$

where P_i is the initial pressure (which is preserved along AOB during steady-state, single phase flow).

The series expansion cannot be used at the injection and production points (I and P) since they represent singularities. We may, however, calculate the pressures at an arbitrary well-radius (r_w) surrounding these points. Taking r_w = 0.25 ft, we obtain, from equation (3.69),

$$p_{inj} - p_{prod} = \Delta p = \frac{4\left[\log_{10}(1414.2/0.25) - 0.2688\right] \times (80/5.615)}{0.001538 \times 158 \times 1}$$

$$= 816 \text{ psi.}$$

But we also know that

$$\tfrac{1}{2}(p_{inj} + p_{prod}) = p_i = 6000 \text{ psi.}$$

Combining these two equations, we have :

$$\begin{cases} p_{inj} = 6000 + \tfrac{1}{2}\Delta p = \underline{6408 \text{ psi}}; \text{ and} \\ p_{prod} = 6000 - \tfrac{1}{2}\Delta p = \underline{5592 \text{ psi}}. \end{cases}$$

Figure G.27 shows the contour map of the analytical solution, while Figures G.28a,b,c show the pressure distribution obtained from the Chapeau, Quadratics and Hermite cubics models, respectively, at 30 days. [To validate the comparison, all four maps have been contoured from the same set of interpolation points].

It can be seen that, at distances not too close to the wells, the finite element results are in good agreement with the Muskat solution, and thus also with one another. The Figures also display the expected symmetry of the contours about the median diagonal. The contours gradually change shape from being approximately linear at the centre of the quarter-five-spot, to being more circular near the wells. Ideally, of course, they would be perfect circles in the immediate neighbourhood of the wells.

By observing the spacing between adjacent contours, we can deduce, from these Figures, two obvious properties concerning the pressure gradients:

(1) The gradients are fairly small in the central and intermediate regions of the domain, but rise very rapidly in the vicinity of the wells. This is so because, as the flow converges towards the well, the "effective cross-sectional area" available for fluid transmission decreases, and hence, by Darcy's Law, larger pressure gradients are required in order to maintain the same flow rate.

(2) For a given pressure contour, the gradient normal to it is maximum at the point where it intersects the leading streamline (IOP), and decreases gradually towards the flanks.

These features are further illustrated by Figures G.29a,b,c and G.30a,b, c which represent the 1-dimensional pressure profiles taken along the diagonal OP, and the flank AP, respectively.

A significant portion of the pressure drop can be seen to occur very near the wells. However, it is precisely in these regions that the

CHAPEAU(5x5)	QUADRATICS(5x5)	H.CUBICS(5x5)	MUSKAT(r_w=0.25')	F.D (10x10)
5822	5768	5740	5592	5804

Table 6.2 Pressures (psi) at the production nodes (relative to 6000psi)

finite element solution departs substantially from the analytical solution, and exhibits pressure gradients which are much smaller than would be predicted from theory. Table 6.2, for example, gives the pressures at the producing node for the three types of elements, and compares them with Muskat's solution at r_w = 0.25 ft (which was calculated earlier). This inaccuracy of the finite element solution may be explained by recalling the fact that a polynomial approximation is being sought for a function which, in reality, is known to exhibit a logarithmic variation. Several authors [54, 28] have demonstrated that this problem can be overcome by incorporating additional basis functions near the wells. Such functions are commonly expressed in radial (instead of x,y) co-ordinates, and are made to possess a singularity at r = 0, a popular version of this being one which involves the term lnr. In the present thesis, however, such extra basis functions have not been successfully programmed, and hence no results are presented in this connection.

6.6 Saturation distribution

Figures G.31, G32 and G33 show saturation contour maps illustrating the flooding history of the quarter five-spot, using Chapeau, Quadratics and Hermite cubics respectively. Figures G.34 show the corresponding results from a nine-point finite difference simulator, constructed by Mr.T.C. Tan of Imperial College. The latter program is based on the theory developed by Yanosik and McCracken [61], and is considered to be superior to the conventional five-point scheme. It is included here merely to compare it with the finite element models, in terms of accuracy and efficiency. All simulations were run out to 2880 days (approximately 8 years), by which time, the water had broken through, and an appreciable water-cut had developed at the producing well. Contour

maps are presented at 180-day intervals from t = 0 to t = 720 days, and thereafter at 360-day intervals until the end of the simulation.

6.6.1 Shapes of Contours during displacement

The overall shape of the contacted region undergoes a continuous change during the flooding process, and is greatly influenced by the mobility data. In displacements characterized by medium-range mobility ratios, we can broadly expect to find three stages of contour development. At early times, the contours form circular arcs around the injection well (Figure 6.9a), but, as the flood front moves away from the injector, the outermost contours display a progressive change from the circular to a square-like pattern, with the corner becoming increasingly "pointed" in the direction of the producer (Figure 6.9b). Eventually, as the breakthrough time draws near, the saturation front shows a definite elongation (or"cusp") towards the producing well, bypassing a significant amount of oil which lies in the peripheral regions of the five-spot (Figure 6.9c).

These general features are observable in the results of all four numerical models (Figures G.31 through G.34). The tendancy of the contours to cusp towards the producing well is an indication of the areal sweep efficiency of the flood. A closer examination of Figures G.31.8, G.32.8, G.33.8 and G.34.8 reveals that the amount of cusping is more pronounced in the quadratics model than in the others. For example, the 50% and 60% saturation contours in Figure G.32.8 show a preferential elongation in the direction of the producing well, while the same contours maintain an approximately square-shaped configuration in

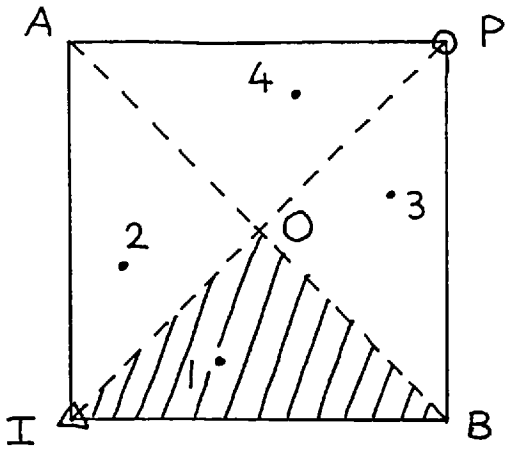


Figure 6.8: Use of symmetry in the calculation of analytical five-spot pressure distribution (solution is first computed inside the shaded region, using equation 3.68, and then extended to the other parts by symmetry.)

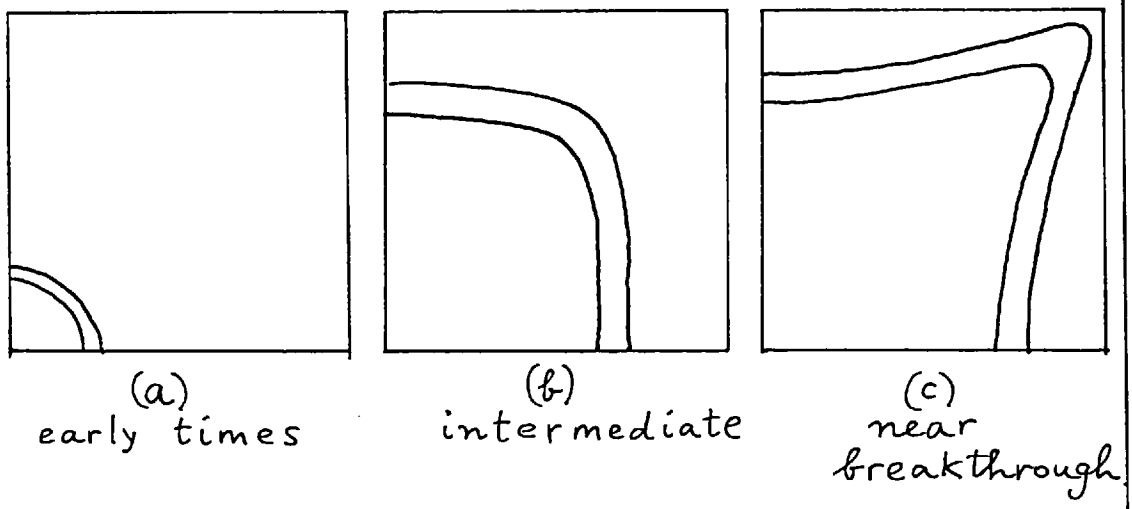


Figure 6.9: Schematic illustration of the shape of the outermost saturation contours, at various stages during a five-spot flood.

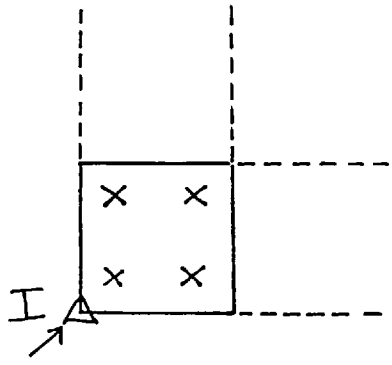


Figure 6.10: Sketch of the element, and Gauss points, connected to the injection node (I).

the Chapeau, Hermite cubics and Finite-Difference maps.

6.6.2 Flood front representation

Since the theoretical flood front saturation for data set 2 is 45%, we may use the separation between (say) the 10% and 50% contours to estimate the relative steepness of the front, for the different cases. Table 6.3 shows these inter-contour distances (in mm), measured along :

(a) the leading diagonal, and

(b) the side

of the maps shown in Figures G31.8, G32.8, G.33.8 and G34.8.

	Chapeau (5 x 5)	Quadratics (5 x 5)	H. Cubics (5 x 5)	9-pt F.D (10 x 10)
Diagonal	74	29	39	55
Side	44	20	25	31

Table 6.3 Distances (mm) between 10% and 50% contours, at t = 2160 days.

It can be seen that the front profile is steepest in the quadratics model - an observation which is borne out also by the 1D results presented earlier. A noteworthy feature of this quadratics model is its ability to create, and preserve, a narrow band of frontal saturations, all the way up to breakthrough. Even after breakthrough, the steepening of the front continues, as the various saturations converge on the producing well. Distances as small as 7 mm (which can be measured on Figure G32.9) indicate the extreme steepness of the advancing ridge of water. By contrast, the Chapeau

model exhibits the most "diffused" saturation front of the four models, while the Hermite cubics and finite difference models take up intermediate positions between the two extremes.

The foregoing remarks concerning the frontal steepness must be considered, however, against a constant background of numerical dispersion, which depends on such factors as:

(a) the grid size

(b) the type of element

(c) the use of lumped capacity matrix, etc.

In the case of the Chapeau model, it is likely that the poor frontal representation is due to the numerical dispersion resulting from the coarse (5 x 5) grid.

In terms of accuracy, it is the finite difference model which appears to match the Buckley-Leverett flood front closely. The steep saturation gradient which exists between the 10% and 40% contours (Figure G34.8) gives way suddenly to a distinct flattening of the slope between the 40% and 50% contours, and thus provides a good agreement with the analytical front saturation of 45%.

Table 6.3 also implies that the saturation fronts are marginally steeper along the sides, than along the leading diagonal.

6.6.3 Saturation overshoots

In all three finite element runs, the region immediately surrounding the injection well is characterized by saturations higher than $(1-S_{or})$,

which are therefore physically meaningless. On the saturation maps, the occurrence of such overshoots is indicated by the area within the contour $S_w = 1$ (e.g. Figures G31.5, G32.5 and G33.5). The finite difference model, on the other hand, is protected by the "upstream weighting scheme", and consequently, the saturations are numerically "well-behaved", and never exceed unity.

To understand this phenomenon, we need to consider the way in which saturation-dependent quantities are evaluated. The finite element formulation, as we have seen, requires the relative permeabilities to be determined at a series of Gauss points within each element. This requirement prevents the nodal saturations from exercising a direct control over the calculation of transmissibility terms. For example, even though the water saturation at a node may have reached the maximum permissible value, it is possible that the internal Gauss points are still at much lower saturations and hence, the element, as a whole, can continue to admit more water. This leads to a progressive build-up of saturation at the affected node (Figure 6.10).

The most vulnerable node is obviously the injection node itself, where the influx of water is concentrated in the form of a Dirac Delta function (see section 2.5). The development and propagation of saturations inside the five-spot can be visualized as an interplay between the "accumulative" and the "transmissive" properties of the system. Initially, when the reservoir is saturated with oil, the transmissibility to water is zero everywhere, and hence the node (I) can utilise only its "accumulative" capacity to absorb the incoming water. Thus, the saturation at this node rises very rapidly at early times. Due to the

polynomial approximation used for S_w, this build-up of saturation automatically implies that saturations throughout the element (including those at the Gauss points) will begin to rise, though at a much slower rate than node I. In general, the further a point is from I, the slower will be its rate of saturation increase.

As the system gradually develops an appreciable transmissibility to water, node I transmits the water in increasing quantities, thereby causing a decrease in its rate of accumulation. Eventually, when all the Gauss-points surrounding it have risen to high saturations, the water flows freely through the element, and hence the rate of increase of S_w at node I becomes progressively small, tending to an almost static saturation. The only difficulty, however, is that (as shown in the Figures), by the time node I attains this steady-state condition, its saturation has already risen well above $(1-S_{or})$. Thus, the region of overshoot (although small) remains throughout the simulation.

In our test runs, it can be seen that the overshoot stops after approximately 1080 days, and that it affects only a small part of the element surrounding the injection node. As the saturation front moves outwards, the task of distributing the water is shared by several nodes, and therefore the danger of the initial overshoot being propagated is generally small, unless of course, the relative permeability data has a pathological character. (An example of this is the use of the unmodified f_w curve of data set 3).

Even the localised overshoots which surround the injection well, may be controllable in either of two ways:

(1) By incorporating extra basis functions in the pressure

approximation (as was mentioned in section 6.5). A logarithmic function, for example, would provide large pressure gradients near the well, and thus help to dissipate the incoming water, and stabilize node I. Spivak et al [54] have reported that this technique is useful for controlling the behaviour of saturations.

(2) By replacing the "concentrated" source term (i.e. Dirac Delta function) by a "distributed" influx of water occurring over the whole, or part, of the element. This will result in the "injection-load" being shared by all the nodes of the element, rather than being borne by a single node.

6.6.4 Saturation oscillations

In the 1 dimensional results (section 6.3.2), we observed that the numerical solution always exhibits a certain degree of spatial oscillation. This tendency of the finite element method carries over also into the 2D simulations, in spite of the lumping of the capacity matrix. This is not to say that lumping is of no consequence in 2 dimensions - on the contrary, it has been found to be essential for the stability of the models. However, it does not succeed in eliminating the oscillations entirely.

Of the finite element models, the one which displays the greatest amount of oscillation is the quadratics model. This can be seen by comparing, for example, Figures G31.4, G32.4, G.33.4 and G34.4. Furthermore, it is also the only one which shows oscillations along the side of the five-spot (e.g. the 70% contour in Figure G32.4), in addition to the oscillations within the injection element.

In the Hermite cubics model (Figure G33.4), the 80% contour exhibits two conspicuous bulb-shaped protrusions. These distortions, however, are damped out with the passage of time, and, at 1440 days (Figure G33.6), the same contour is much smoother than it was before. The Chapeau model is perhaps the best among the finite element models, with regard to oscillations.

The tendency of the finite element models to oscillate can be related to the nature of the polynomial approximation that is being used. For example, the Chapeau model, being a bilinear representation in x and y, cannot, by definition, display oscillations inside an element. The quadratic and cubic polynomials, on the other hand, are obviously capable of developing local maxima or minima (or similar changes in slope) within the elements.

The nine-point finite difference model yields smooth and rounded contours throughout the simulation, and hardly any oscillations can be detected.

6.6.5 Inter-element continuity of contours

This property is determined by the degree of continuity of the polynomials. In the Chapeau and Quadratics models (which possess C^0 continuity), distinct "breaks" can be observed along the length of a contour. (e.g. Figures G31.6 and G32.6). These abrupt changes in the direction of contours are caused by the discontinuity of saturation gradients occurring at inter-element boundaries. Once again, it is the quadratics model which appears to be most sensitive to this phenomenon. The Hermite cubics model, by virtue of its C^1 continuity, yields perfectly smooth contours, regardless of the presence of inter-element boundaries

(e.g. Figures G33.6, G33.7 etc). It is perhaps worth mentioning that the small discontinuities which are recognisable in the Hermite cubics maps are due entirely to the (somewhat coarse) secondary mesh used by the contouring package, and are not caused by the approximating polynomial. We can thus see that the aspects of functional continuity, which were raised in the formulation of the finite element theory (chapter 4), are fully verified in the 1D and 2D model results.

6.7 Comparison of Breakthrough times and recoveries

In this section, we shall evaluate the breakthrough recovery for data set 2, using three different methods cited in the literature, and then compare these values with the recovery efficiency predicted by the four numerical models.

6.7.1 Guthrie and Greenberger Method [29]

This is strictly an empirical formula, developed in 1955, and relates the recovery efficiency of a natural water drive to such basic reservoir parameters as porosity, permeability, connate water saturation, thickness and oil viscosity. The constants involved were obtained from statistical correlation studies on several reservoirs:

$$E_R = 0.114 + 0.272 \log_{10} K + 0.256 S_{wc} - 0.136 \log_{10} \mu_o - 1.538 \phi - 0.00035 h .$$

Substituting into this equation, the relevant data for our test problem, namely :

$$K = 158 \text{ md}, \; S_{wc} = 0, \; \mu_o = 4 \text{ cp}, \; \phi = 0.1 \text{ and } h = 1 \text{ ft.},$$

We obtain the recovery efficiency as:

$$\underline{E_R = 0.476} .$$

6.7.2 Craig et al Method [14]

This method was described in section 3.7.2, where we also remarked that the calculation of the mobility ratio of the flood can be based on any of three possible saturations. For our test problem, these three values are as follows:

(1) Maximum water saturation

$$S_{w\ max} = 1 - S_{or} = \underline{1}$$

(2) Average water saturation in the flooded region:

$$\overline{S}_w = S_{wc} + [1/(dfw/dsw)_{swf}]$$

$$= 0 + (0.447/0.723) = \underline{0.618}$$

(3) Frontal saturation

$$S_{wf} = \underline{0.447}$$

Table 6.4 shows the three corresponding mobility ratios (M), and the breakthrough areal sweep efficiencies $(E_A)_{BT}$, as obtained from the published charts of Craig et al [14] (Figure 3.16b), and of Dyes et al [23] (Figure 3.16a).

The breakthrough displacement efficiency $(E_D)_{BT}$ for our problem is given by:

$$(E_D)_{BT} = \frac{\overline{S_w} - S_{wc}}{1 - S_{wc} - S_{or}} = \underline{0.618}\ ;$$

and so, finally, we can express the breakthrough oil recovery as:

$$(E_R)_{BT} = 0.618 \times (E_A)_{BT} .$$

This parameter $(E_R)_{BT}$, is also listed in Table 6.4, for the three cases of mobility evaluation. Two obvious features can be deduced from this Table:

(1) Both correlation charts - i.e. those of Craig et al and of Dyes et al - give approximately the same areal sweep efficiency at breakthrough.

(2) The predicted recovery efficiency is highest when the water mobility is evaluated at the flood front saturation; least when it is evaluated at the maximum saturation; and intermediate when the average saturation ($\overline{S}_w$) is used.

S_w	M	$(E_A)_{BT}$		$(E_R)_{BT}$
		Craig et al	Dyes et al	Craig et al
S_{wmax} = 1.0	4.00	0.535	0.54	0.33
$\overline{S}w$ = 0.618	1.53	0.63	0.62	0.39
S_{wf} = 0.447	0.80	0.72	0.73	0.44

Table 6.4 Calculation of flood mobility ratio, areal sweep, and recovery at breakthrough for test problem. Based on Craig's method (Section 3.7.2)

6.7.3 Higgins and Leighton Method

A major limitation of this model, as was pointed out in section 3.8.3, is the fact that it is based on a constant pressure difference between the injection and production wells, rather than on a constant rate of displacement. Consequently, the flow rate can show an appreciable variation during the history of the flood.

In the present test problem, due to the oil/water viscosity ratio of 4:1, the overall mobility of the system increases as the more mobile water invades the reservoir. In response to this enhanced mobility, the displacement rate rises rapidly at early times. It then settles down to a gentle increase until breakthrough, and during this period we may approximate the rate by a constant "average" value. Numerical experiments showed that, in this case, the "average" rate was approximately 50% higher than the initial rate. Thus, for our problem, the choice of an initial value of 0.00014 PV/day enabled the displacement rate to increase, and stabilize, around 0.00020 or 0.00021 PV/day, which was the specified rate of injection. Figure G35 shows the variation of flow rate with time, while Table 6.5 shows the cumulative oil produced from each channel (as a fraction of the total reservoir pore-volume), at the times of breakthrough of the four channels.

Clearly, the breakthrough of the five-spot, as a whole, is governed by the behaviour of Channel 1, which is the one adjacent to the leading diagonal, and thus also the fastest of all channels. Hence, the breakthrough oil recovery of the five-spot (from Table 6.5) is given by:

$$(E_R)_{BT} = \underline{0.462}\ .$$

The complete performance of the Higgins and Leighton 5-spot model is given in Table 6.6.

Breakthrough	Time (days)	Cumulative Oil (Pore Volumes)				
		CH.1	CH.2	CH.3	CH.4	Total
CH.1	2326	0.138	0.135	0.123	0.067	0.462
CH.2	2617	0.143	0.153	0.139	0.076	0.512
CH.3	3305	0.154	0.164	0.180	0.097	0.595
CH.4	4845	0.169	0.182	0.201	0.147	0.699

Table 6.5 Cumulative recovery from each channel (Higgins and Leighton method).

TIME(DAYS)	WATER-CUT	OIL RATE	WATER RATE	CUMULATIVE OIL	CUMULATIVE WATER
.00000010	0	.00014000	0	.00000000	0
180.00000010	0	.00016950	0	.02789523	0
360.00000010	0	.00018468	0	.05994287	0
540.00000010	0	.00019192	0	.09387546	0
720.00000010	0	.00019663	0	.12885391	0
900.00000010	0	.00020008	0	.16457493	0
1080.00000010	0	.00020285	0	.20084183	0
1260.00000010	0	.00020515	0	.23756501	0
1440.00000010	0	.00020715	0	.27467586	0
1620.00000010	0	.00020895	0	.31212599	0
1800.00000010	0	.00021063	0	.34988962	0
1980.00000010	0	.00021230	0	.38795109	0
2160.00000010	0	.00021413	0	.42632479	0
2340.00000010	.22958660	.00016995	.00005065	.46452788	.00172110
2520.00000010	.24434593	.00016977	.00005490	.49568748	.00223315
2700.00000010	.47140148	.00012333	.00010998	.52192962	.00469895
2880.00000010	.49120749	.00012135	.00011716	.54394341	.00645161

TABLE 6.6. Performance of the five-spot, as predicted by the Higgins and Leighton method.

6.7.4 Performance of the numerical models

Tables 6.7 and 6.8 list the cumulative oil recovery and the producing water-cut as a function of time for the four numerical models considered in this thesis. In studying these figures, it has to be remembered that numerical dispersion, which affects the models to varying degrees, usually results in a premature breakthrough of water, and causes the rise in water-cut at the producing well to be much more gradual, and spread out over a longer period of time, than would be theoretically predicted. Nevertheless, the results are fairly consistent with one another, and an examination of the development of the water-cut in these Tables shows that, in all the models, the oil recovery at the time when significant quantities of water begin to appear, is between 40% and 50%.

If, for the purpose of comparison, we choose $f_w = 0.1$ as the water -cut indicating breakthrough, then, the times at which this occurs, and the corresponding oil recoveries, can be obtained by linear interpolation from these Tables. These values, together with the semi-analytical results obtained in the previous three sections, are tabulated in Table 6.9.

The Quadratics and the Hermite cubics models, as well as the 9-pt finite difference model, show results which are in good agreement with those of the "Higgins and Leighton" and the "Craig et al" methods (provided that the water mobility in the latter method is based on $S_w = S_{wf}$). The comparatively early breakthrough and the low recovery observed in the Chapeau model are possibly due to the coarseness of its grid when compared to the other models.

Time (Years)	CHAPEAU		QUADRATICS	
	PV recov'd	fw	PV recov'd	fw
4.00	0.287	0.017	0.288	0.000
4.25	0.305	0.024	0.306	0.000
4.50	0.322	0.032	0.324	0.001
4.75	0.340	0.042	0.342	0.001
5.00	0.357	0.055	0.360	0.001
5.25	0.374	0.069	0.378	0.000
5.50	0.391	0.086	0.396	0.001
5.75	0.407	0.107	0.414	0.004
6.00	0.423	0.130	0.432	0.021
6.25	0.438	0.156	0.449	0.069
6.50	0.453	0.186	0.465	0.164
6.75	0.468	0.220	0.479	0.490
7.00	0.482	0.258	0.486	0.653
7.25	0.495	0.301	0.492	0.612
7.50	0.507	0.347	0.499	0.636
7.75	0.519	0.397	0.506	0.672
8.00	0.529	0.450	0.511	0.672

TABLE 6.7

Oil recovery and producing water-cut for five-spot flood, using Chapeau and Quadratics models.

Time (Years)	HERMITE CUBICS		9-POINT F.D.	
	PV recov'd	fw	PV recov'd	fw
4.00	0.288	0.000	0.288	0.000
4.25	0.306	0.000	-	-
4.50	0.324	0.000	0.324	0.000
4.75	0.342	0.000	-	-
5.00	0.360	0.001	0.360	0.000
5.25	0.378	0.002	-	-
5.50	0.396	0.006	0.396	0.000
5.75	0.414	0.015	-	-
6.00	0.431	0.029	0.432	0.000
6.25	0.449	0.053	-	-
6.50	0.466	0.090	0.467	0.177
6.75	0.482	0.147	-	-
7.00	0.497	0.230	0.493	0.364
7.25	0.510	0.339	-	-
7.50	0.521	0.460	0.513	0.469
7.75	0.530	0.567	-	-
8.00	0.537	0.633	0.531	0.536

TABLE. 6.8 Oil recovery and producing water-cut for five-spot flood, using Hermite cubics and 9-point finite difference models.

Model		Breakthrough Time(Days)**	Oil Recovery At Breakthrough
Chapeau	(5x5)	2041	0.385
Quadratics	(5x5)	2279	0.454
H. Cubics	(5x5)	2355	0.468
9 pt. Finite Diff	(10x10)	2261	0.452
Guthrie-Greenberger		-	0.476*
Craig et al	$Sw=\overline{Sw}$	not calculated	0.39
	Sw =Swf	not calculated	0.44
Higgins and Leighton		2326	0.462

* ultimate recovery under natural water drive

** For the numerical models, breakthrough is defined to be at $f_w = 10\%$.

TABLE 6.9 Comparison of breakthrough times and recoveries, using different models.

The correlation of Guthrie and Greenberger, being an empirical one, cannot be taken too seriously, but nevertheless, it is interesting to note that the predicted recovery is of the right order of magnitude. For Craig's method, the breakthrough times have not been calculated in the present study, but the recovery figures seem to indicate (in spite of Craig's recommendation) that the use of the average saturation in the flooded region (in evaluating the water mobility) may yield too pessimistic an estimate for the breakthrough recovery, and that a better prediction may be obtained by using the flood front saturation (S_{wf}).

6.8 Comparison of water-cut profiles

All three finite element models were run out only as far as 8 years, and hence their performance at late times was not determined. Within the period of simulation, however, the producing water-cut, in all cases, had risen to above 40% (Tables 6.7 and 6.8). Figures G36, G37, G38 and G39 show plots of the producing water-cut, from t=4 to t=8 years, for all the four numerical models. Figure G40 shows the water-cut trend, as predicted by the Higgins and Leighton method.

As might be expected, these graphs reflect the relative steepness of the saturation front, for the different cases. In the Chapeau model, where the front is considerably smeared, the water-cut shows a slow, but steady, increase over a long period of time. The Quadratics model, where the front is found to be steepest, shows a sudden and rapid increase in f_w after t = 6 years, indicating the arrival (at the producing well) of a closely spaced band of frontal saturations. Having risen to a maximum of 65%, the water-cut then flattens out markedly, and displays minor oscillations in the interval t = 7 to t = 8 years. These oscillations could be partly due to rate-instabilities occurring at the production node during the post-breakthrough period.

The Hermite cubics model yields a breakthrough time approximately equal to that of the Quadratics model, and shows an f_w profile which is smooth, and intermediate in steepness between the Chapeau and Quadratics curves, reaching a water-cut of 53% at the end of 8 years. Furthermore, the f_w curve of the Hermite cubics model shows a gentler rise than that of the 9-point Finite Difference model shortly after breakthrough,

but, thereafter, increases rapidly to overtake the latter, and reach a 63% water-cut at 8 years.

In the Higgins and Leighton model, the water-cut history is characterised by a series of discrete "jumps" (Figure G40), where each jump corresponds to the breakthrough of a new channel. Between these successive discontinuities, the rate of increase of f_w with time is much smaller than in the numerical models. It is interesting to realise that, if the number of channels used in this method is increased indefinitely, then, the resulting f_w profile will tend, in the limit, to a continuous curve.

6.9 Computer time

We shall conclude this chapter with a brief assessment of the Central Processor Unit (CPU) time taken by the various models. The final costing formula is, of course, very complicated, and takes into account all the computer resources used by the program. These include central memory, CPU time, disk operations, input/output routines etc. As was described in section 4.10.3(C), the solution algorithm for the 2 D model involves the use of a random access binary file, and this,in turn, increases substantially the real time required to execute the program. We can, however, use the CPU time as a measure of the efficiency of the program code, and use this parameter to compare the speeds of execution of the different models.

The principal variables associated with each model are:

(1) the number of elements; and

(2) the number of time-steps.

(The number of Gauss points per element is also an important factor, but since this is governed by the type of finite element that is being used, and moreover, since the assembly of the matrices is only a part of the entire scheme, this factor will be omitted in the normalisation of CPU time).

For comparative purposes, we shall therefore express the CPU time in

secs/time-step/element.

Table 6.10 shows typical values for the models used in this thesis.

Model	CPU time*	Elements	time-steps	Gauss-points	Normalised CPU time
1D Chapeau	21	20	604	2	1.74×10^{-3}
1D Quadratics	46	20	604	3	3.81×10^{-3}
1D H.cubics	53	20	606	3	4.37×10^{-3}
2D Chapeau	37	5 x 5	125	2 x 2	0.012
2D Quadratics	387	5 x 5	194	3 x 3	0.080
2D H.cubics	515	5 x 5	121	3 x 3	0.170
2D 9 pt F.D	98	10x10	240**	-	0.004

Table 6.10 Comparison of CPU times required by the various numerical models.
*times are in CDC 6600 seconds.
**estimated minimum number of steps.

These figures reveal the expected trend - namely, that the CPU time increases as the order of the approximating polynomial increases. Furthermore, the time required by the 2D models, is considerably higher than that required by the corresponding 1D models. For example, the

ratios of computing times (2D to 1D) can be seen to be 7, 21 and 39, respectively, for the Chapeau, Quadratics and the Hermite cubics models. By comparison, the 9-point finite difference model is the cheapest of all the 2D models.

CHAPTER 7

Conclusions, and Suggestions for further work

The numerical results shown in this thesis are consistent with the work of other authors in this field, and lead us to the following conclusions regarding the application of classical (Galerkin) finite elements to two-phase reservoir simulation.

7.1 Conclusions from 1D results

1. The finite element method is an accurate and elegant method for simulating waterfloods, provided that allowance is made for several important factors.

2. Despite the continuous nature of the polynomial approximation, the use of too few elements can lead to poor results. This statement applies, regardless of the degree of the polynomial. To achieve good results, the number of elements should be at least 10, preferably in the range 15 to 20.

3. The saturation profiles obtained from finite element models show a natural tendency to oscillate, and this tendency is most pronounced when the displacement is characterised by a sharp front. In such cases, the oscillations usually begin some distance upstream of the front; their amplitude increases gradually, reaching a maximum near the front; and they progressively diminish on the downstream side of the front.

4. Control of these localised oscillations may be achieved by

the use of a lumped capacity matrix. This has proved to be particularly effective with the Hermite Cubics model, and to a lesser extent with the Chapeau model. The response of the Quadratics model appears to be marginal. It is also observed that, although capacity lumping dampens the oscillations, it has the disadvantage of causing the front to become more smeared. As a general rule, however, the use of capacity lumping is desirable as a safeguard against severe oscillations.

5. The accuracy of the finite element solution is strongly dependent on the mobility data, and, in particular, on the shape of the f_w curve. With conventional f_w curves which show an upward concavity (i.e. S shape), the numerical solution cannot be guaranteed to converge to the Buckley-Leverett solution. If the curve has a uniformly decreasing slope, then the agreement between the finite element and the Buckley-Leverett solutions is good. However, as the f_w curve develops an upward concavity, and becomes more and more S shaped, the numerical solution suffers a progressive deterioration, and can even become absurd. Three possible methods are available for overcoming this problem. These involve, respectively, the use of:

   (a) pseudo capillary gradients;

   (b) upstream weighting;

   (c) a modified f_w curve.

6. The incorporation of a fictitious capillary pressure gradient (p_c') can provide the required correction to the finite element results. The magnitude of this gradient can be

estimated from four parameters:

(a) the area enclosed between the original f_w curve and the "Welge tangent";

(b) the area (from Sw_c to Sw_f) lying beneath the curve which represents half the harmonic mean of the relative fluid mobilities;

(c) the total superficial flow velocity; and

(d) the absolute permeability.

7. As with finite differences, the upstream weighting of relative permeabilities has proved to be a suitable artifice for finite elements. A scheme based on the nearest upstream Gauss point has produced good results with all three finite element models (Chapeau, Quadratics and Hermite Cubics). An alternative scheme, involving the upstream node (instead of Gauss points) also proved satisfactory with the Chapeau model, but had a very unfavourable effect on the Quadratics and Hermite Cubics models.

8. Dalen's method (16a) of modifying the f_w curve appears to be the most logical and least subjective way of ensuring numerical stability. This approach merely involves replacing the S-shaped portion of the original f_w curve (i.e. between Swc and Swf) by the Welge tangent. Together with a lumped capacity matrix, and a velocity-dependent upstream weighting scheme, this method has yielded satisfactory results on all the data sets studied in this thesis. Since the amount of upstream shift involved in these runs was negligible, the

success is attributable principally to the modified f_w curve, aided, of course, by the lumped capacity matrix. It is also interesting to note that the use of such a curve is perfectly consistent with the Buckley-Leverett/Welge solution method. It does, however, lead to numerical dispersion.

7.2 Conclusions from 2D results

9. The finite element method is very stable with respect to the pressure solution, and has yielded a smooth and symmetrical pressure distribution within the quarter five-spot domain. Except in the vicinity of point sources or sinks, these numerical results show a good match with Muskat's analytical solution.

10. Theoretically, the progress of a flood front through a five-spot would be expected to display a gradual change in the shape of the saturation contours (i.e. from being circular near the injection well, to being more elongated (diagonally) as the producing well is approached). This feature has been verified in all the finite element models.

11. In spite of using the modified f_w curve, the contour maps show a small region of saturation overshoot around the injection well. This undesirable phenomenon is shared by all three finite element models, and the cause of this is thought to be partly in the "point-source" formulation used for the wells. As some authors have indicated, it may be possible to prevent these overshoots by employing additional basis functions to obtain a better representation of the pressure distribution near the wells.

12. Table 7.1 summarizes the observed behaviour of the finite element models, with regard to the following three aspects: steepness of flood front; oscillatory behaviour in the flooded region; and contour smoothness.

	Chapeau	Quadratics	H.Cubics
Steepness of flood front	minimum	maximum	intermediate
Oscillations in the flooded region	best	worst	intermediate
Smoothness of Contours	intermediate	worst	best

TABLE 7.1 Summary of behaviour of finite element models

13. The computational cost increases with the degree of the polynomial, and, particularly, in going from 1D to 2D.

14. The breakthrough recoveries are in broad agreement with the results from the stream channel model of Higgins and Leighton.

15. The nine-point finite difference model is cheap, stable and accurate; and it yields smooth and rounded contours.

16. In terms of overall performance, the Hermite Cubics model, despite being expensive, appears to be the most attractive among the finite element models. The Chapeau model is a good substitute, provided a sufficiently fine mesh is used. The nine-point finite difference model, on the other hand, is fully competitive with all the finite element models.

7.3 Suggestions for further work

As has been mentioned earlier, only a handful of finite

element types have been studied in this thesis. A considerable amount of research remains to be done in order to be able to introduce a wider variety of elements, as well as to place the existing theory on a much more firm and stable footing.

The following are some areas which might merit further attention:

(1) The use of triangular, serendipity, and (eventually) isoparametric elements, in the simulation of two phase flow. (The 8-node rectangular element has already been used by White (57)).

(2) Introduction of greater refinement into the rectangular tensor-product family of elements. Recently, some papers have focussed attention on the control of numerical dispersion, and the optimisation of the simulation, by combining the finite element method with such techniques as:
   (a) the method of characteristics;
   (b) local mesh refinement (10);
   (c) the use of different elements in different parts of the domain (for example, Hermite Cubics in regions of rapid saturation changes, and Chapeau elements elsewhere).

(3) Extension of the work of Kebaili and Thomas (35), to create a more detailed coning model.

(4) Extension to 3-dimensional systems - a target which appears prohibitive in terms of computing time and storage.

Several obstacles will have to be overcome before any of these objectives can be fully realised. The numerical difficulties which

confront the finite element analyst appear to be far greater than in the field of finite differences, and, consequently, it has to be admitted that, at this stage, the use of finite elements in two-phase simulation cannot be approached with the same degree of confidence as finite differences. However, it is precisely this challenging situation that acts as a stimulant to future work, and reassures the researcher that, whatever the outcome of his efforts, the knowledge of having analysed a problem is a reward in itself.

APPENDIX 1

Data Set 1 (Wright et al. [60])

(a) Mobility data

$$k_{ro} = S_o^2 \; ; \; k_{rw} = 1 - k_{ro} \; ; \; \mu_o = \mu_w = 0.5 cp \; ;$$

$$S_{wc} = S_{or} = 0. \quad \text{(Figure 6.1a)}$$

(b) Rate data

$$q = 0.0178 \, b/d \; (= 10^{-3} \, PV/day).$$

(c) Reservoir and other data

$$\phi = 0.1 \; ; \; K = 158 \, md \; ; \; A = 1 \, ft^2 \; ;$$

System length L = 1000 ft ; Dip = 0 ;

Compressibilities : $c_o = c_w = \frac{1}{3} \times 10^{-6} \, psi^{-1}$.

Data Set 2 (Spivak et al.[54]).

(a) Mobility data

$$k_{ro} = S_o^2 \; ; \; k_{rw} = S_w^2 \; ; \; \mu_o = 4 \, cp \; ; \; \mu_w = 1 \, cp \; ;$$

$$S_{wc} = S_{or} = 0. \quad \text{(Figure 6.1b)}$$

(b) and (c) as for data set 1.

Data set 3 (Langsrud [36])

(a) Mobility data

$$S_{wc} = S_{or} = 0.15 \; ; \; \mu_o = \mu_w = 0.5 \, cp \; ;$$

Relative permeabilities are tabulated overleaf (See also Figure 6.1c).

(b) Rate data

q = 70 b/d (=0.0009826 PV/day)

(c) Reservoir and other data

$\phi = 0.2$; $k = 500$ md; A = 1000 ft^2

System length L = 2000 ft ; Dip = 0 ;

Compressibilities: $c_o = c_w = \frac{1}{3} \times 10^{-6}$ psi^{-1}.

s_w	k_{rw}	k_{ro}
0.15	0.0000	1.0000
0.20	0.0010	0.7500
0.25	0.0040	0.5880
0.30	0.0102	0.4460
0.35	0.0166	0.3340
0.40	0.0232	0.2450
0.45	0.0305	0.1770
0.50	0.0392	0.1200
0.55	0.0497	0.0724
0.60	0.0630	0.0375
0.65	0.0797	0.0163
0.70	0.1000	0.0056
0.75	0.1244	0.0020
0.80	0.1525	0.0004
0.85	0.1870	0.0000

(Figure 6.1c).

Data Set 4 (Linear relative permeabilities).

(a) Mobility data

$k_{ro} = S_o$; $k_{rw} = S_w$; $\mu_o = \mu_w = 0.5$ cp ;

$S_{wc} = S_{or} = 0.$ (Figure 6.1 d).

(b) Rate data

$q = 0.0297$ b/d (=0.0017 pv/day)

(c) as for data set 1

APPENDIX 2.1

1D MODEL - GENERAL SUBROUTINES

```
      PROGRAM MAIN(INPUT=100,OUTPUT=100,TAPE5=INPUT,TAPE6=OUTPUT,
     1TAPE7=100,TAPE82=100)
      COMMON/D-/DOMEAN,SWMEAN,QINJ,QWINJ,QOPROD,QWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LUMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      IN=5
      IP=6
      TDAYS=0.
      LPRINT=0
      RATIO=2.
      LTSTEP=0
      CALL INIT
      CALL PRINT(0)
    3 READ(IN,1) TPDELT,DTMIN,DTMAX
      IF(TPDELT.LT.1.E-06) GO TO 900
      IF(LTSTEP.EQ.0) DT=DTMIN
      READ(IN,2) TNEXT,ITMAX,IPRINT,IDUG,ITSUM
   10 TDAYS=TDAYS+TPDELT
      TDIFF=TNEXT-TDAYS
      IF(TDIFF.LT.1.E-06) GO TO 5
      TPRINT=TDAYS*86+00.
      LCE=0
      GO TO 6
    5 TPRINT=TNEXT*86+00.
      LCE=1
    6 TDIFF=TPRINT-CUMT
      IF(DT.LT.TDIFF) GO TO 7
      DUM=DT
      DT=TDIFF
      LPRINT=1
    7 CUMT=CUMT+DT
      DO 8 ITER=1,ITMAX
      CALL RATES
      DO 100 LVAR=1,2
      CALL ASSIGN
      CALL FACSUB
      IF(ITSUM.EQ.0) GO TO 100
      CALL PRINT(0)
  100 CONTINUE
    8 CONTINUE
      CALL PSMAX
      LTSTEP=LTSTEP+1
      IF(IPRINT.EQ.1.OR.LPRINT.EQ.1) CALL PRINT(LPRINT)
      IF(IDUG.EQ.0) GO TO 11
      WRITE(IP,9) CUMT,DT,DPMAX,DSWMAX,ITMAX,LPP,LSS
   11 IF(LPRINT.EQ.1) GO TO 12
      IF(ABS(DPMAX).LT.1.E-06) DPMAX=1.E-06
      IF(ABS(DSWMAX).LT.1.E-06) DSWMAX=1.E-06
      R1=ABS(EPSP/DPMAX)
      R2=ABS(EPSSW/DSWMAX)
      RATIO=AMIN1(R1,R2,2.)
      DT=DT*RATIO
      IF(DT.GT.DTMAX) DT=DTMAX
      GO TO 6
   12 DT=DUM
      LPRINT=0
      IF(LCE.EQ.1) GO TO 3
      GO TO 10
  900 TDAYS=-1.
      WRITE(7) TDAYS
      READ(IN,1000) LGRAPH
      IF(LGRAPH.EQ.0) STOP
      LGRAPH=NELNUM
      CALL PROF(NM1,LGRAPH)
      STOP
    1 FORMAT(8F10.4)
    2 FORMAT(F10.4,1+I5)
    9 FORMAT(1H0,4E15.4,5I5/)
 1000 FORMAT(16I5)
      END

      BLOCK DATA
      COMMON/DS/XI(5),WEIGHT(5),S(4,6),DS(4,6),SS(10,5),DD(10,5),
     1DDS(10,5),XX(15),WW(15),INUM
      DATA XX/.5,.211324865,.788675135,.112701665,.5,.887298335,
     1.069431844,.330009478,.669990522,.930568156,.046910077,.230765345,
     2.5,.769234655,.953089923/,WW/1.,.5,.5,.27777778,.44444444,
     3.27777778,.173927423,.326072577,.326072577,.173927423,.118463443,
     4.239314335,.284444444,.239314335,.118463443/
      END
```

```
      SUBROUTINE RATES
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(20),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(3),BODA(3),BWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1PCWDA(30),RRKODA(30),RRKWDA(30),DRKODA(30),DRKWDA(30),FWW(30),
     2DFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      COMMON/D2/AREA(20),RESK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D4/BOMEAN,BWMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CONT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      PRESBP=PO(IPROD)+PDATUM
      DO 2 LI=1,NUMPVT
      IF(PRESBP.LT.PRESDA(LI)) GO TO 3
    2 CONTINUE
    3 DELTA=(PRESBP-PRESDA(LI-1))/(PRESDA(LI)-PRESDA(LI-1))
      SDELTA=1.-DELTA
      ABO=SDELTA*BWDA(LI-1)+DELTA*BODA(LI)
      ABW=SDELTA*BWDA(LI-1)+DELTA*BWDA(LI)
      AVISO=SDELTA*VISODA(LI-1)+DELTA*VISODA(LI)
      AVISW=SDELTA*VISWDA(LI-1)+DELTA*VISWDA(LI)
      DO 10 I=1,NODES
      SAT=SW(I)
      DO 4 LI=1,NUMSW
      IF(SAT.LT.SWDA(LI)) GO TO 5
    4 CONTINUE
    5 DELTA=(SAT-SWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      SDELTA=1.-DELTA
      RRKO(I)=SDELTA*RRKODA(LI-1)+DELTA*RRKODA(LI)
      RRKW(I)=SDELTA*RRKWDA(LI-1)+DELTA*RRKWDA(LI)
      DKO(I)=SDELTA*DRKODA(LI-1)+DELTA*DRKODA(LI)
      DKW(I)=SDELTA*DRKWDA(LI-1)+DELTA*DRKWDA(LI)
   10 CONTINUE
      QW(INJ)=QINJ
      FO=RRKO(IPROD)/(AVISO*ABO)
      FW=RRKW(IPROD)/(AVISW*ABW)
      FS=QINJ*BWMEAN/(FO*BOMEAN+FW*BWMEAN)
      QO(IPROD)=-FS*FO
      QW(IPROD)=-FS*FW
      QO(IPROD)=QO(IPROD)*ABO
      QW(IPROD)=QW(IPROD)*ABW
      QW(INJ)=QW(INJ)*BWMEAN
      IF(ITER.LT.ITMAX) RETURN
      CONFAC=DT/(86400.*1.E+6)
      CWINJ=CWINJ+QW(INJ)*CONFAC
      COPROD=COPROD+QO(IPROD)*CONFAC
      CWPROD=CWPROD+QW(IPROD)*CONFAC
      RETURN
      END


      SUBROUTINE PSMAX
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(20),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D4/BOMEAN,BWMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CONT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      DPMAX=0.
      LPP=0
      DSWMAX=0.
      LSS=0
      DO 300 I=1,NODES
      DIFF=PO(I)-POOLD(I)
      IF(ABS(DPMAX).GT.ABS(DIFF)) GO TO 210
      DPMAX=DIFF
      LPP=I
  210 POOLD(I)=PO(I)
      DIFF=SW(I)-SWOLD(I)
      IF(ABS(DSWMAX).GT.ABS(DIFF)) GO TO 400
      DSWMAX=DIFF
      LSS=I
  400 SWOLD(I)=SW(I)
      IF(NELNUM.NE.3) GO TO 300
      DPOOLD(I)=DPO(I)
      DSWOLD(I)=DSW(I)
  300 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SELEV(NFW,NFRONT,Q)
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      DO 50 I=1,NFW
      N1=NFW+1-I
      DIFF=SWD(NFRONT)-SWD(N1)
      XC=DFW(N1)*Q
      IF(DIFF.GT.0.00001) GO TO 12
      Y(I)=SWD(N1)
      X(I)=XC
      IF(XC.GT.1.) GO TO 14
   40 CONTINUE
   14 DDY=(1.-X(I-1))*(Y(I-1)-Y(I))/(X(I)-X(I-1))
      Y(I)=Y(I-1)-DDY
      X(I)=1.
      I=I+1
   12 Y(I)=SWD(1)
      X(I)=X(I-1)
      Y(I+1)=SWD(1)
      X(I+1)=1.
      NPTEL=I+1
      X(NPTEL+1)=FVX
      X(NPTEL+2)=DVX
      Y(NPTEL+1)=FVSW
      Y(NPTEL+2)=DVSW
      CALL PEN(1)
      CALL LINE(X,Y,NPTEL,1,0,1)
      CALL PEN(1)
      RETURN
      END


      SUBROUTINE SFEM(NOPERL,INTFEM,NM1,DSP,DXTOT,LGRAPH)
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      READ(7) N1,(SW(I),I=1,N1)
      IF(LGRAPH.EQ.3) READ(7) (DSW(I),I=1,N1)
      N1=NOPERL
      IF(LGRAPH.EQ.2) N1=NOPERL/2
      CUMX=-DSP*DX(1)/DXTOT
      J1=1
      NPTFEM=0
      DO 14 I=1,NM1
      DCUMX=DSP*DX(I)/DXTOT
      DO 12 J=J1,NOPERL+1
      NPTFEM=NPTFEM+1
      GO TO (20,30,40),LGRAPH
   20 SAT=SP(1,J)*SW(I)+SP(2,J)*SW(I+1)
      GO TO 50
   30 I1=2*I-1
      SAT=SP(1,J)*SW(I1)+SP(2,J)*SW(I1+1)+SP(3,J)*SW(I1+2)
      GO TO 50
   40 X1=DX(I)
      SAT=SP(1,J)*SW(I)+SP(3,J)*SW(I+1)+X1*(SP(2,J)*DSW(I)+SP(4,J)*DSW(I
     1+1))
   50 CUMX=CUMX+DCUMX
      Y(NPTFEM)=SAT
   12 X(NPTFEM)=CUMX


   14 J1=2
      Y(NPTFEM+1)=FVSW
      Y(NPTFEM+2)=DVSW
      X(NPTFEM+1)=FVX
      X(NPTFEM+2)=DVX
      CALL PEN(4)
      CALL LINE(X,Y,NPTFEM,1,N1,INTFEM)
      CALL PEN(1)
      RETURN
      END
```

```
      SUBROUTINE PROF(NM1,LGRAPH)
      COMMON PO(41),POLD(21),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CD1(50),CD2(21),CD5(21),CD6(21),A(16),PHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      IF(NM1.NE.0) GO TO 800
      READ(5,1010) NM1
      READ(5,1020) (DX(I),I=1,NM1)
  800 CONTINUE
      DXTOT=0.
      DO 8 I=1,NM1
    8 DXTOT=DXTOT+DX(I)
      CALL START(2)
      CALL BORDOFF
      XPAGE=-0.45
      YPAGE=9.75
      XPAGE=8.25
      YPAGE=9.44
      CALL NEWPAGE
      CALL PLOT(2.,2.375,-3)
      DO 40 J=1,10
      READ(5,1000) (TITLE(I),I=1,8)
      IF(TITLE(1).EQ.3HEND) GO TO 12
      CALL SYMBOL(XPAGE,YPAGE,0.56,TITLE,0.,80)
      YPAGE=YPAGE-1.12
      GO TO 40
      CALL SYMBOL(XPAGE,YPAGE,0.28,TITLE,0.,80)


      YPAGE=YPAGE-0.84
   40 CONTINUE
   12 CONTINUE
      CALL BORDOFF
      FVX=0.
      DVX=1./15.
      FVSW=0.
      DVSW=0.1
      READ(5,1010) NFW,NFRONT,INTFEM,NDPERL
      READ(5,1020) QRATE
      N3=0
      IF(NFW.GT.0) GO TO 920
      N3=1
      NFW=-NFW
  920 CONTINUE
      READ(5,1020) (SWD(I),I=1,NFW)
      READ(5,1020) (DFW(I),I=1,NFW)
      IF(N3.EQ.0) GO TO 940
      DO 50 I=NFRONT+1,NFW-1
      S=(SWD(NFW)-SWD(I))*2.
   50 DFW(I)=(((0.0002837*S +0.0012444)*S+0.0012953)*S+0.0377444)
     1*S*20.
  940 CONTINUE
      IF(LGRAPH.EQ.1) NDPERL=1
      DGP=1./FLOAT(NDPERL)
      X1=-DGP
      DO 10 I=1,NDPERL+1
      X1=X1+DGP
      X2=1.-X1
      GO TO (110,120,130),LGRAPH
  110 SP(1,I)=X2
      SP(2,I)=X1
      GO TO 10
  120 X3=1.-2.*X1
      SP(1,I)=X2*X3
      SP(2,I)=4.*X1*X2
      SP(3,I)=-X1*X3
      GO TO 10
  130 X3=X1*X1
      X4=X2*X2
      SP(3,I)=X3*(3.-2.*X1)
      SP(1,I)=1.-SP(3,I)
      SP(2,I)=X4*X1
      SP(4,I)=-X3*X2
   10 CONTINUE
      REWIND 7
   20 READ(7) TDAYS
      IF(TDAYS.LT.0.) GO TO 30
      Q=QRATE*TDAYS
      GO TO 70
      CALL NEWPAGE
```

```
      CALL PLOT(2.,2.,3.75,-3)
      CALL SYMBOL(.5,1.,.14,11HTIME(DAYS)=,0.,11)
      CALL NUMBER(7.2,1.,.15,0.,56,TDAYS,0.,2)
   70 CONTINUE
      CALL FACTOR(3.75)
      CALL AXIS(0.,0.,22HFRACTIONAL DISTANCE(X),-22,4.,0.,0.,0.25)
      CALL FACTOR(2.5)
      CALL AXIS(0.,0.,24HSW,2,-1.,90.,0.,0.25)
      CALL FACTOR(1.)
      CALL SHELV(NFW,NFRONT,2)
      CALL SFEM(NOPERL,INTFEM,NM1,DOP,DXTOT,LGRAPH)
      GO TO 20
   30 CALL ENPLOT
      RETURN
 1000 FORMAT(8A10)
 1010 FORMAT(16I5)
 1020 FORMAT(8F10.4)
      END

      SUBROUTINE INIT
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(20),CO2(21),CO5(21),CO6(21),A(46),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(3),SODA(3),SWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1PCWDA(30),RRKODA(30),RRKWDA(30),DRKODA(30),DRKWDA(30),FWW(30),
     2DFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D3/XI(5),WEIGHT(5),S(4,6),DS(4,6),SS(10,5),DD(10,5),
     1DDS(10,5),XX(15),WW(15),INUM
      COMMON/D4/SOMEAN,SWMEAN,QINJ,QWINJ,QOPROD,QWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,COUNT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      GRAV=0.434/14.7
      READ(IN,1)    NELNUM
      READ(IN,2) NODES,INJN,LUMP,LUPST,INJ,IPROD,LIMPSW
      READ(IN,4) PDATUM,DATUM,DENOSC,DENWSC
      READ(IN,4)   EPSP,EPSSW
      READ(IN,4)   QINJ
      QINJ=QINJ*1.84
      USF=1.
      IF(LUPST.NE.0) LIMPSW=0
      WRITE(IP,1000)  NELNUM
      WRITE(IP,5) NODES,PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW
      WRITE(IP,100) LUMP,LIMPSW,LUPST,USF
      WRITE(IP,2)
      DO 3 I=1,50
      READ(IN,4)  PRESDA(I),SODA(I),SWDA(I),VISODA(I),VISWDA(I)
      IF(PRESDA(I).LT.1.) GO TO 5
      WRITE(IP,6) PRESDA(I),SODA(I),SWDA(I),VISODA(I),VISWDA(I)
      PRESDA(I)=PRESDA(I)/14.7
    3 CONTINUE
    5 NUMPVT=I-1
      WRITE(IP,7)
      DO 50 I=1,50
      READ(IN,4)  SWDA(I),PCWDA(I),RRKWDA(I),RRKODA(I)
      IF(SWDA(I).GT.5.) GO TO 9
      WRITE(IP,6) SWDA(I),PCWDA(I),RRKWDA(I),RRKODA(I)
      PCWDA(I)=-PCWDA(I)/14.7
   50 CONTINUE
    9 NUMSW=I-1
      NM1=NODES-1
      IF(NELNUM.EQ.2) NM1=(NODES-1)/2
      READ(IN,4)   (POR(I),I=1,NM1)
      READ(IN,4)   (ABSK(I),I=1,NM1)
      READ(IN,4)   (AREA(I),I=1,NM1)
      READ(IN,4)   (DX(I),I=1,NM1)
      READ(IN,4)    (DEPTH(I),I=1,NODES)
      READ(IN,4)   (SW(I),I=1,NODES)
      IF(NELNUM.NE.3) GO TO 200
      READ(IN,4)   (DPO(I),I=1,NODES)
      READ(IN,4)   (DSW(I),I=1,NODES)
  200 CONTINUE
      WRITE(IP,10)
      WRITE(IP,6) (POR(I),I=1,NM1)
      WRITE(IP,11)
      WRITE(IP,5) (ABSK(I),I=1,NM1)
      WRITE(IP,12)
      WRITE(IP,13)  (AREA(I),I=1,NM1)
      WRITE(IP,14)
      WRITE(IP,13)(DX(I),I=1,NM1)
      WRITE(IP,15)
      WRITE(IP,13)  (DEPTH(I),I=1,NODES)
      WRITE(IP,28)
```

```
      I1=(INUM*(INUM-1))/2
      DO 29 I=1,INUM
      I1=I1+1
      XI(I)=XX(I1)
      WEIGHT(I)=WW(I1)
      WRITE(12,30) I,XI(I),WEIGHT(I)
   29 CONTINUE
      IF(NELNUM.EQ.1) CALL LINESH(LUMP)
      IF(NELNUM.EQ.2) CALL QUADSH(LUMP)

      IF(NELNUM.EQ.3) CALL CUBESH(LUMP)
      FACT=30.48**3
      PDATUM=PDATUM/14.7
      DO 16 LI=1,NUMPVT
      IF(PDATUM.LT.PRESDA(LI)) GO TO 17
   16 CONTINUE
   17 DELTA=(PDATUM-PRESDA(LI-1))/(PRESDA(LI)-PRESDA(LI-1))
      SDELTA=1.-DELTA
      BOMEAN=SDELTA*BODA(LI-1)+DELTA*BODA(LI)
      BWMEAN=SDELTA*BWDA(LI-1)+DELTA*BWDA(LI)
      VOMEAN=SDELTA*VISODA(LI-1)+DELTA*VISODA(LI)
      VWMEAN=SDELTA*VISWDA(LI-1)+DELTA*VISWDA(LI)
      DENO=DENOSC/BOMEAN
      DO 18 I=1,NODES
      QO(I)=0.
      QW(I)=0.
      SWOLD(I)=SW(I)
      PO(I)=DENO*GRAV*(DEPTH(I)-DATUM)
      POOLD(I)=PO(I)
      IF(NELNUM.NE.3) GO TO 18
      DPO(I)=DPO(I)/(14.7*30.48)
      DSW(I)=DSW(I)/30.48
      DPOOLD(I)=DPO(I)
      DSWOLD(I)=DSW(I)
   18 CONTINUE
      DO 32 I=1,NM1
      I1=I
      I2=I1+1
      IF(NELNUM.NE.2) GO TO 210
      I1=2*I-1
      I2=I1+2
  210 CONTINUE
      TRANS(I)=ABSK(I)*AREA(I)*30.48*30.48
      DEPTH(I)=(DEPTH(I2)-DEPTH(I1))*GRAV/(DX(I)*30.48)
      POR(I)=POR(I)*AREA(I)*DX(I)*FACT
   32 DX(I)=DX(I)*30.48
      DO 70 I=1,NUMSW
      T1=RRKODA(I)/VOMEAN
      T2=RRKWDA(I)/VWMEAN
      FWW(I)=T2/(T1+T2)
   70 GWW(I)=T1*T2/(T1+T2)
      DO 40 I=1,(NUMSW-1)
      DELTA=SWDA(I+1)-SWDA(I)
      DFWW(I)=(FWW(I+1)-FWW(I))/DELTA
      DGWW(I)=(GWW(I+1)-GWW(I))/DELTA
      DRKWDA(I)=(RRKWDA(I+1)-RRKWDA(I))/DELTA
   40 DRKODA(I)=(RRKODA(I+1)-RRKODA(I))/DELTA
      DRKWDA(NUMSW)=DRKWDA(NUMSW-1)
      DRKODA(NUMSW)=DRKODA(NUMSW-1)
      DFWW(NUMSW)=DFWW(NUMSW-1)
      DGWW(NUMSW)=DGWW(NUMSW-1)
      CUMT=0.
      CWINJ=0.
      COPROD=0.
      CWPROD=0.
      RETURN
    1 FORMAT(16I5)
    2 FORMAT(1H0,15H PRESSURE TABLE//12X,4HPRES,13X,2HKO,13X,2HKW,9X,6HV
     1ISOIL,9X,6HVISWAT/)
    4 FORMAT(8F10.4)
    6 FORMAT(1H0,8F15.4)
    7 FORMAT(1H0,9H SW TABLE//14X,2HSW,12X,3HPCW,12X,3HKRW,12X,3HKRO)
    8 FORMAT(1H0,12HNO OF NODES=,I4,5X,16HDATUM PRESSURE =,F10.3,7H PSI
     1AT,F10.0,3H FT//17H DENSITIES  OIL =,F10.3,6H WATER =,F10.3//7H DP
     2MAX=,F10.4,4H ATM,5X,7HDSWMAX=,F10.4/)
   10 FORMAT(1H0,10HPOROSITIES/)
   11 FORMAT(1H0,21HPERMEABILITIES(DARCY)/)
   12 FORMAT(1H0,11HAREAS(SQFT)/)
   13 FORMAT(1H0,8F15.0)
   14 FORMAT(1H0,19HELEMENT LENGTHS(FT)/)
   15 FORMAT(1H0,16HNODAL DEPTHS(FT)/)
   28 FORMAT(1H0,10X,10HGAUSS PTS.,8X,7HWEIGHTS/)
   30 FORMAT(1X,15,2F15.8)
  100 FORMAT(1H0,6HLJMP =,I2,10X,8HLIMPSW =,I2,10X,7HLUPST =,I2,5H     ,
     15HUSF =,F10.4/)
 1000 FORMAT(1H0,8HNELNUM =,I5)
      END
```

APPENDIX 2.2

1D MODEL - CHAPEAU SUBROUTINES

```
      SUBROUTINE LINEAR(LJMP)
      COMMON/D3/XI(5),WEIGHT(5),S(4,6),DS(4,6),SS(16,5),DD(16,5),
     1DDS(16,5),XX(15),WW(15),INUM
      DO 111 I=1,INUM
      S(1,I)=1.-XI(I)
      S(2,I)=XI(I)
      DS(1,I)=-1.
      DS(2,I)=1.
      LJ=0
      DO 112 J=1,2
      DO 114 K=1,2
      LJ=LJ+1
      SS(LJ,I)=S(J,I)*S(K,I)
      DD(LJ,I)=DS(J,I)*DS(K,I)
114   DDS(LJ,I)=DS(J,I)*S(K,I)
112   CONTINUE
      IF(LUMP.EQ.0) GO TO 111
      DO 115 K=1,4
115   SS(K,I)=0.
      SS(1,I)=S(1,I)
      SS(4,I)=S(2,I)
111   CONTINUE
      RETURN
      END


      SUBROUTINE UPSTWT(ST,LI,DELTA,SDELTA,QFRAC,GGP)
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(6),BODA(3),BWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1PCWDA(30),RRKODA(30),RRKWDA(30),DRKODA(30),DRKWDA(30),FWW(30),
     2DFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      DFT=SDELTA*DFWW(LI-1)+DELTA*DFWW(LI)
      USGP=GGP-DFT*QFRAC
      IE=IFIX(USGP)+1
      IF(IE.LT.1) GO TO 10
      X1=USGP-FLOAT(IE-1)
      X2=1.-X1


      ST=X2*SW(IE)+X1*SW(IE+1)
      GO TO 20
10    ST=SW(1)
20    RETURN
      END


      SUBROUTINE THOMAS(J)
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D4/BOMEAN,BWMEAN,QINJ,QWINJ,QOPROD,QWPROD,DT,NODES,NM1,
     1PDATUM,JATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LINPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      I=J
      CO1(I)=A(2)/A(1)
      V1=RHS(1)/A(1)
      A(1)=A(4)-A(3)*CO1(I)
      RHS(1)=RHS(2)-A(3)*V1
      GO TO (10,20),LVAR
10    PO(I)=V1
      IF((I+1).EQ.NODES) PO(NODES)=RHS(1)/A(1)
      RETURN
20    SW(I)=V1
      IF((I+1).EQ.NODES) SW(NODES)=RHS(1)/A(1)
      RETURN
      END
```

```
      SUBROUTINE BACSUB
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(60),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D4/BOMEAN,ENMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      DO 10 I=1,NM1
      J=NODES-1
      GO TO (20,30),LVAR
   20 PO(J)=PO(J)-CO1(J)*PO(J+1)
      GO TO 10
   30 SW(J)=SW(J)-CO1(J)*SW(J+1)
   10 CONTINUE
      RETURN
      END


      SUBROUTINE PRINT(LPRINT)
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(60),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D4/BOMEAN,ENMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      WRITE(IP,1)
      TDAYS=CUMT/86400.
      WRITE(IP,2) TDAYS
      WRITE(IP,3)
      DO 100 I=1,NODES
      PS1=(PO(I)+PDATUM)*14.7
      SW1=SW(I)
      QO1=QO(I)/1.84
      QW1=QW(I)/1.84
      WRITE(IP,5) I,PS1,SW1,QO1,QW1
  100 CONTINUE
      IF(LPRINT.EQ.0) RETURN
      WRITE(IP,7) CWINJ,COPROD,CWPROD,LTSTEP
      T1=TDAYS-300.
      IF(ABS(T1).GT.1.01) RETURN
      WRITE(7) TDAYS
      WRITE(7) NODES,(SW(I),I=1,NODES)
      RETURN
    1 FORMAT(1H0,135(1H-))
    2 FORMAT(1H ,6HTIME =,F15.3,5H DAYS)
    3 FORMAT(1H ,4HNODE,3X,7HPO(PSI),13X,2HSW,6X,9HQO(STB/D),6X,9HQW(STB
     1/D)/)
    5 FORMAT(1H ,I4,F15.2,3F15.6)
    7 FORMAT(1H ,4X,15HCUM WAT INJ(STB),3X,17HCUM OIL PROD(STB),3X,17HCU
     1M WAT PROD(STB),10X,10HTIME-STEPS/1X,3E20.4,I20)
      END


      SUBROUTINE ASSIGN
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(60),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(3),BODA(3),BWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1POWDA(30),RRKODA(30),RRKWDA(30),DRKODA(30),DRKWDA(30),FWW(30),
     2DFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D3/XI(5),WEIGHT(5),S(4,6),DS(4,6),SS(10,5),DD(10,5),
     1DDS(16,5),XX(15),WW(15),INUM
      COMMON/D4/BOMEAN,ENMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      A(1)=0.
      RHS(1)=0.
      DO 29 I=1,NM1
```

```
      QFRAC=USF*QINJ*DT/POR(I)
      I1=I+1
      DO 3 J=2,4
    3 A(J)=0.
      RHS(2)=0.
      DX1=DX(I)
      STP=SW(I)
      DO 4 II=1,INUM
      GGP=FLOAT(I-1)+XI(II)
      TRAN=TRANS(I)*WEIGHT(II)
      PVOL=POR(I)*WEIGHT(II)/DT
      PT=S(1,II)*PO(I)+S(2,II)*PO(I1)
      ST=S(1,II)*SW(I)+S(2,II)*SW(I1)
      DPT=(DS(1,II)*PO(I)+DS(2,II)*PO(I1))/DX1
      DST=(DS(1,II)*SW(I)+DS(2,II)*SW(I1))/DX1
      PRESER=PT+POATUM
      DO 5 LI=1,NUMPVT
      IF(PRESER.LT.PRESDA(LI)) GO TO 6
    5 CONTINUE
    6 DELTA=PRESDA(LI)-PRESDA(LI-1)
      ADBODP=((1./BODA(LI))-(1./BODA(LI-1)))/DELTA
      ADBWDP=((1./BWDA(LI))-(1./BWDA(LI-1)))/DELTA
      DELTA=(PRESER-PRESDA(LI-1))/DELTA
      SDELTA=1.-DELTA
      ABO=SDELTA*BODA(LI-1)+DELTA*BODA(LI)
      ABW=SDELTA*BWDA(LI-1)+DELTA*BWDA(LI)
      AVISO=SDELTA*VISODA(LI-1)+DELTA*VISODA(LI)
      AVISW=SDELTA*VISWDA(LI-1)+DELTA*VISWDA(LI)
      ADENO=DENOSC/ABO
      ADENW=DENWSC/ABW
      COMP=((1.-ST)*ABO*ADBODP+ST*ABW*ADBWDP)*PVOL
      ST1=ST
      IF(ST.LT.0.) ST=0.
      IF(LUPST.EQ.0) GO TO 920
      GO TO (810,820,830),LUPST
  810 CONTINUE
      DO 921 LI=1,NUMSW
      IF(ST.LT.SWDA(LI)) GO TO 922
  921 CONTINUE
  922 DELTA=(ST-SWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      SDELTA=1.-DELTA
      CALL UPSTWT(ST,LI,DELTA,SDELTA,QFRAC,GGP)
      GO TO 920
  820 CONTINUE
      STE=ST
      ST=STP
      STP=STE
      GO TO 920
  830 ST=STP
  920 CONTINUE
      DO 7 LI=1,NUMSW
      IF(ST.LT.SWDA(LI)) GO TO 8
    7 CONTINUE
    8 DELTA=(ST-SWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      SDELTA=1.-DELTA
      ARKO=SDELTA*ARKODA(LI-1)+DELTA*ARKODA(LI)
      ARKW=SDELTA*ARKWDA(LI-1)+DELTA*ARKWDA(LI)
      ADPCDS=(PCWDA(LI)-PCWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      T1=TRAN*ARKO/AVISO
      T2=TRAN*ARKW/AVISW
      T3=T2*ADPCDS*DST
      T4=DEPTH(I)*(T1*ADENO+T2*ADENW)
      T5=(T1+T2)/DX1
      GO TO (600,601),LVAR
  600 DO 700 J=1,4
  700 A(J)=A(J)+T5*CD(J,II)+COMP*SS(J,II)
      T7=T4-T3
      RHS(1)=RHS(1)+T7*DS(1,II)+COMP*(SS(1,II)*POOLD(I)+SS(2,II)*POCLD(I
     11))
      RHS(2)=RHS(2)+T7*DS(2,II)+COMP*(SS(3,II)*POOLD(I)+SS(4,II)*POCLD(I
     11))
      GO TO 4
  601 C1=AVISW/AVISO
      B1=ST-SWDA(LI-1)
      B2=ST-SWDA(LI)
      B3=ST-SWDA(LI+1)
      Q1=SWDA(LI)-SWDA(LI+1)
      Q2=SWDA(LI+1)-SWDA(LI-1)
      Q3=SWDA(LI-1)-SWDA(LI)
      R1=-B2*B3/(Q2*Q3)
      R2=-B3*B1/(Q3*Q1)
      R3=-B1*B2/(Q1*Q2)
      FT=R1*FWW(LI-1)+R2*FWW(LI)+R3*FWW(LI+1)
      GT=R1*GWW(LI-1)+R2*GWW(LI)+R3*GWW(LI+1)
      DFT=0.
      DGT=0.
```

```
      DUW=0.
      IF(LIMPSW.EQ.0) GO TO 605
      DTW=(SUBELTA*DFKWDA(LI-1)+DELTA*DFKWDA(LI))/AVISW
      DUW=-TRAN*DTW*(DPT+ADPDDS*DST-ADENW*DEPTH(I1))
  605 CONTINUE
      ST=ST1
      T6=TRAN*GT*ADPDDS/DX1
      PTO=S(1,II)*POOLD(I)+S(2,II)*POOLD(I1)
      T9=PVOL*(1.+AEW*ADEWDP*(PT-PTO))
      DO 720 J=1,4
  720 A(J)=A(J)+T9*SS(J,II)+T6*DD(J,II)+DUW*DDS(J,II)
      T5=T1+T2
      UU=-T5*DPT-T3+T4
      T8=(ADENW-ADENO)*DEPTH(I)*TRAN
      T11=UU*FT+T8+DUW*ST
      RHS(1)=RHS(1)+T11*DS(1,II)+PVOL*(SS(1,II)*SWOLD(I)+SS(2,II)*SWOLD(
     1I1))
      RHS(2)=RHS(2)+T11*DS(2,II)+PVOL*(SS(3,II)*SWOLD(I)+SS(4,II)*SWOLD(
     1I1))
    4 CONTINUE
      C1=FLOAT(2-LVAR)
      RHS(1)=RHS(1)+C1*QD(I)+QW(I)
      IF(I1.EQ.NODES) RHS(2)=RHS(2)+C1*QD(I1)+QW(I1)
      CALL THOMAS(I)
   29 CONTINUE
      RETURN
      END
```

APPENDIX 2.3

1D MODEL - QUADRATICS SUBROUTINES

```
      SUBROUTINE QUADSH(LJMP)
      COMMON/DS/XI(5),WEIGHT(5),S(4,5),DS(4,5),SS(16,5),DD(16,5),
     1DDS(16,5),XX(15),WW(15),INUM
      DO 111 I=1,INUM
      X1=XI(I)
      X2=1.-X1
      X3=1.-2.*X1
      S(1,I)=X2*X3
      S(2,I)=4.*X1*X2
      S(3,I)=-X1*X3
      DS(1,I)=4.*X1-3.
      DS(2,I)=4.*X2
      DS(3,I)=4.*X1-1.
      LJ=0
      DO 112 J=1,3
      DO 114 K=1,3
      LJ=LJ+1
      SS(LJ,I)=S(J,I)*S(K,I)
      DD(LJ,I)=DS(J,I)*DS(K,I)
114   DDS(LJ,I)=DS(J,I)*S(K,I)
112   CONTINUE
      IF(LUMP.EQ.1) GO TO 111
      DO 115 K=1,9
115   SS(K,I)=0.
      SS(1,I)=S(1,I)
      SS(5,I)=S(2,I)
      SS(9,I)=S(3,I)
111   CONTINUE
      RETURN
      END


      SUBROUTINE PRINT(LPRINT)
      COMMON PO(41),POLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SND(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RSKO(41),RSKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D4/BOMEAN,BWMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LISTEP,
     2OPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      WRITE(IP,1)
      TDAYS=CUMT/86400.
      WRITE(IP,2) TDAYS
      WRITE(IP,3)
      DO 100 I=1,NODES,2
      PS2=0.
      SW2=0.
      QO2=0.
      QW2=0.
      PS1=(PO(I)+PDATUM)*14.7
      SW1=SW(I)
      QO1=QO(I)/1.84
      QW1=QW(I)/1.84
      IF(I.EQ.NODES) GO TO 110
      PS2=(PO(I+1)+PDATUM)*14.7
      SW2=SW(I+1)
      QO2=QO(I+1)/1.84
      QW2=QW(I+1)/1.84
110   WRITE(IP,5) I,PS1,PS2,SW1,SW2,QO1,QO2,QW1,QW2
100   CONTINUE
      IF(LPRINT.EQ.0) RETURN
      WRITE(IP,7) CWINJ,COPROD,CWPROD,LTSTEP
      T1=TDAYS-300.
      IF(ABS(T1).GT.0.01) RETURN
      WRITE(7) TDAYS
      WRITE(7) NODES,(SW(I),I=1,NODES)
      RETURN

    1 FORMAT(1HO,135(1H-))
    2 FORMAT(1HO,6HTIME =,F15.3,5H DAYS)
    3 FORMAT(1HO,4HNODE,3X,7HPO(PSI),28X,2HSW,21X,9HQO(STB/D),21X,9HQW(S
     1TB/D)/)
    5 FORMAT(1H ,I4,2F15.2,6F15.6)
    7 FORMAT(1HO,4X,15HCUM WAT INJ(STB),3X,17HCUM OIL PROD(STB),3X,17HCU
     1M WAT PROD(STB),10X,10HTIME-STEPS/1X,3E20.4,I20)
      END
```

```
      SUBROUTINE THOMAS(J)
      COMMON PO(41),POO_D(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D4/BOMEAN,SWMEAN,QINJ,CWINJ,CCPROD,CWPROD,DT,NODES,NM1,
     1POATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,COUNT,ITER,ITMAX,LISTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      I=J
      I1=I+1
      DET=A(1)*A(5)-A(2)*A(4)
      B1=A(5)/DET
      B2=-A(2)/DET
      B3=-A(4)/DET
      B4=A(1)/DET
      CO1(I)=B1*A(3)+B2*A(6)
      CO1(I1)=B3*A(3)+B4*A(6)
      V1=B1*RHS(1)+B2*RHS(2)
      V2=B3*RHS(1)+B4*RHS(2)
      A(1)=A(9)-A(7)*CO1(I)-A(8)*CO1(I1)
      RHS(1)=RHS(3)-A(7)*V1-A(8)*V2
      GO TO (10,20),LVAR
10    PO(I)=V1
      PO(I1)=V2
      IF((I+2).EQ.NODES) PO(NODES)=RHS(1)/A(1)
      RETURN
20    SW(I)=V1
      SW(I1)=V2
      IF((I+2).EQ.NODES) SW(NODES)=RHS(1)/A(1)
      RETURN
      END


      SUBROUTINE BACSUB
      COMMON PO(41),POO_D(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D4/BOMEAN,SWMEAN,QINJ,CWINJ,CCPROD,CWPROD,DT,NODES,NM1,
     1POATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,COUNT,ITER,ITMAX,LISTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      N1=NODES
      DO 2 I=1,NM1
      N2=N1-1
      N3=N2-1
      GO TO (4,6),LVAR

 4    PO(N2)=PO(N2)-CO1(N2)*PO(N1)
      PO(N3)=PO(N3)-CO1(N3)*PO(N1)
      GO TO 2
 6    SW(N2)=SW(N2)-CO1(N2)*SW(N1)
      SW(N3)=SW(N3)-CO1(N3)*SW(N1)
 2    N1=N1-2
      RETURN
      END


      SUBROUTINE UPSTWT(ST,LI,DELTA,SDELTA,OFRAC,GGP)
      COMMON PO(41),POO_D(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWD(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(3),BODA(3),BWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1PCWDA(30),PRKODA(30),PRKWDA(30),DRKODA(30),DRKWDA(30),FWW(30),
     2OFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      DFT=SDELTA*OFWW(LI-1)+DELTA*OFWW(LI)
      USGP=GGP-DFT*OFRAC
      IE=IFIX(USGP)+1
      IF(IE.LT.1) GO TO 10
      X1=USGP-FLOAT(IE-1)
      X2=1.-X1
      X3=1.-2.*X1
      S1=X2*X3
      S2=4.*X1*X2
      S3=-X1*X3
      IE=2*IE-1
      ST=S1*SW(IE)+S2*SW(IE+1)+S3*SW(IE+2)
      GO TO 20
10    ST=SW(1)
20    RETURN
      END
```

```
      SUBROUTINE ASSIGN
      COMMON P0(41),P00LD(41),DP0(21),DP00LD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(50),CO2(41),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(3),BODA(3),BWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1PBWDA(30),RRKODA(30),RRKWDA(30),DRKODA(30),DFKWDA(30),FWW(30),
     2DFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      COMMON/D2/AREA(20),ABS(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RFKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D3/XI(5),WEIGHT(5),S(4,6),DS(4,6),SS(10,5),DD(10,5),
     1DDS(16,5),XX(15),NN(15),INUM
      COMMON/D4/BOMEAN,BWMEAN,QINJ,QWINJ,QOPROD,QWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUNT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUPF,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      A(1)=1.
      RHS(1)=0.
      DO 29 I=1,NM1
      QFRAC=USF*QINJ*DT/POR(I)
      I1=2*I-1
      I2=I1+1
      I3=I1+2
      DO 3 J=2,9
    3 A(J)=0.
      RHS(2)=0.
      RHS(3)=0.
      DX1=DX(I)
      STP=SW(I1)
      DO 4 II=1,INUM
      GGP=FLOAT(I-1)+XI(II)
      TRAN=TRANS(I)*WEIGHT(II)
      PVOL=POR(I)*WEIGHT(II)/DT
      PT=S(1,II)*PO(I1)+S(2,II)*PO(I2)+S(3,II)*PO(I3)
      DPT=(DS(1,II)*PO(I1)+DS(2,II)*PO(I2)+DS(3,II)*PO(I3))/DX1
      ST=S(1,II)*SW(I1)+S(2,II)*SW(I2)+S(3,II)*SW(I3)
      DST=(DS(1,II)*SW(I1)+DS(2,II)*SW(I2)+DS(3,II)*SW(I3))/DX1
      PRESER=PT+PDATUM
      DO 5 LI=1,NUMPVT
      IF(PRESER.LT.PRESDA(LI)) GO TO 6
    5 CONTINUE
    6 DELTA=PRESDA(LI)-PRESDA(LI-1)
      ADBODP=((1./BODA(LI))-(1./BODA(LI-1)))/DELTA
      ADBWDP=((1./BWDA(LI))-(1./BWDA(LI-1)))/DELTA
      DELTA=(PRESER-PRESDA(LI-1))/DELTA
      SDELTA=1.-DELTA
      ABO=SDELTA*BODA(LI-1)+DELTA*BODA(LI)
      ABW=SDELTA*BWDA(LI-1)+DELTA*BWDA(LI)
      AVISO=SDELTA*VISODA(LI-1)+DELTA*VISODA(LI)
      AVISW=SDELTA*VISWDA(LI-1)+DELTA*VISWDA(LI)
      ADENO=DENOSC/ABO
      ADENW=DENWSC/ABW
      COMP=((1.-ST)*ABO*ADBODP+ST*ABW*ADBWDP)*PVOL
      ST1=ST
      IF(ST.LT.0.) ST=0.
      IF(LUPST.EQ.0) GO TO 920
      GO TO (810,820,830),LUPST
  810 CONTINUE
      DO 921 LI=1,NUMSW
      IF(ST.LT.SWDA(LI)) GO TO 922
  921 CONTINUE
  922 DELTA=(ST-SWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      SDELTA=1.-DELTA
      CALL UPSTWT(ST,LI,DELTA,SDELTA,QFRAC,GGP)
      GO TO 920
  820 CONTINUE
      STE=ST
      ST=STP
      STP=STE
      GO TO 920
  830 ST=STP
  920 CONTINUE
      DO 7 LI=1,NUMSW
      IF(ST.LT.SWDA(LI)) GO TO 8
```

```
  7  CONTINUE
  8  DELTA=(ST-SWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
     SDELTA=1.-DELTA
     ARKO=SDELTA*ARKODA(LI-1)+DELTA*ARKODA(LI)
     ARKW=SDELTA*ARKWDA(LI-1)+DELTA*ARKWDA(LI)
     ADPODS=(POWDA(LI)-POWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
     T1=TRAN*ARKO/AVISO
     T2=TRAN*ARKW/AVISW
     T3=T2*ADPODS*DST
     T4=DEPTH(I)*(T1*ADENO+T2*ADENW)
     T5=(T1+T2)/DX1
     GO TO (500,501),LVAR
500  DO 700 J=1,9
700  A(J)=A(J)+T5*DD(J,II)+COMP*SS(J,II)
     T7=T4-T3
     DO 710 J=1,3
     I0=3*(J-1)
710  RHS(J)=RHS(J)+T7*DS(J,II)+COMP*(SS(I0+1,II)*POOLD(I1)+SS(I0+2,II)*
    1POOLD(I2)+SS(I0+3,II)*POOLD(I3))
     GO TO 4
501  C1=AVISW/AVISO
     B1=ST-SWDA(LI-1)
     B2=ST-SWDA(LI)
     B3=ST-SWDA(LI+1)
     Q1=SWDA(LI)-SWDA(LI+1)
     Q2=SWDA(LI+1)-SWDA(LI-1)
     Q3=SWDA(LI-1)-SWDA(LI)
     R1=-B2*B3/(Q2*Q3)
     R2=-B3*B1/(Q3*Q1)
     R3=-B1*B2/(Q1*Q2)
     FT=R1*FWW(LI-1)+R2*FWW(LI)+R3*FWW(LI+1)
     GT=R1*GWW(LI-1)+R2*GWW(LI)+R3*GWW(LI+1)
     DFT=0.
     DGT=0.
     DUW=0.
     IF(LIMPSW.EQ.0) GO TO 605
     DTW=(SDELTA*DRKWDA(LI-1)+DELTA*DRKWDA(LI))/AVISW
     DUW=-TRAN*DTW*(DPT+ADPODS*DST-ADENW*DEPTH(I2))
605  CONTINUE
     ST=ST1
     T6=TRAN*GT*ADPODS/DX1
     PTO=S(1,II)*POOLD(I1)+S(2,II)*POOLD(I2)+S(3,II)*POOLD(I3)
     T9=PVOL*(1.+AEW*ADSWDP*(PT-PTO))
     DO 720 J=1,9
720  A(J)=A(J)+T9*SS(J,II)+T6*DD(J,II)+DUW*DDS(J,II)
     T8=T1+T2
     UU=-T5*DPT-T3+T4
     T8=(ADENW-ADENO)*DEPTH(I)*TRAN
     T11=UU*FT+T8+DUW*ST
     DO 730 J=1,3
     I0=3*(J-1)
730  RHS(J)=RHS(J)-T11*DS(J,II)+PVOL*(SS(I0+1,II)*SWOLD(I1)+SS(I0+2,II)
    1*SWOLD(I2)+SS(I0+3,II)*SWOLD(I3))
  4  CONTINUE
     C1=FLOAT(2-LVAR)
     RHS(1)=RHS(1)+C1*QO(I1)+QW(I1)
     RHS(2)=RHS(2)+C1*QO(I2)+QW(I2)
     IF(I3.EQ.NODES) RHS(3)=RHS(3)+C1*QO(I3)+QW(I3)
     CALL THOMAS(I1)
 29  CONTINUE
     RETURN
     END
```

APPENDIX 2.4

1D MODEL - HERMITE CUBICS SUBROUTINES

```
      SUBROUTINE SPLINE(IJK)
      COMMON/D3/X1(5),WEIGHT(5),S(4,5),DS(4,5),SS(16,5),DD(16,5),
     1DDS(16,5),XX(15),AA(15),INUM
      DO 111 I=1,INUM
      X1=XI(I)
      X4=1.0-X1
      X2=X1*X1
      X3=X2*X1
      S(3,I)=(3.0*X2-2.0*X3)
      S(1,I)=1.0-S(3,I)
      S(4,I)=X3-X2
      S(2,I)=X1-X2-X4
      DS(3,I)=6.0*X1*X4
      DS(1,I)=-DS(3,I)
      DS(2,I)=X4*(1.0-3.0*X1)
      DS(4,I)=X1*(3.0*X1-2.0)
      JJ=0
      LJ=0
      DO 112 J=1,4
      DO 113 K=J,4
      JJ=JJ+1
      SS(JJ,I)=S(J,I)*S(K,I)
113   DD(JJ,I)=DS(J,I)*DS(K,I)
      DO 114 K=1,4
      LJ=LJ+1
114   DDS(LJ,I)=DS(J,I)*S(K,I)
112   CONTINUE
      IF(LUMP.EQ.0) GO TO 111
      SS(1,I)=S(1,I)
      SS(2,I)=S(2,I)
      SS(5,I)=S(2,I)*(S(2,I)-S(4,I))
      SS(9,I)=S(3,I)
      SS(13,I)=S(4,I)
      SS(10,I)=S(4,I)*(S(4,I)-S(2,I))
      SS(3,I)=0.0
      SS(4,I)=0.0
      SS(6,I)=0.0
      SS(7,I)=0.0
111   CONTINUE
      S(1,6)=1.0
      S(2,6)=0.0
      S(3,6)=0.0
      S(4,6)=0.0
      RETURN
      END


      SUBROUTINE THOMAS(J)
      COMMON PO(41),POLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(21),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D4/COMEAN,SWMEAN,QINJ,CWINJ,CCPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,COUNT,ITER,ITMAX,LISTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV


      I=J
      I1=I+1
      DET=A(1)*A(4)-A(2)*A(3)
      B1=A(4)/DET
      B2=-A(2)/DET
      B3=-A(3)/DET
      B4=A(1)/DET
      V1=B1*RHS(1)+B2*RHS(2)
      V2=B3*RHS(1)+B4*RHS(2)
      GO TO (150,151) LVAR
150   PO(I)=V1
      DPO(I)=V2
      GO TO 152
151   SW(I)=V1
      DSW(I)=V2
152   IF(I.EQ.NODES) RETURN
      CO1(I)=B1*A(5)+B2*A(7)
      CO2(I)=B1*A(6)+B2*A(8)
      CO5(I)=B3*A(5)+B4*A(7)
      CO6(I)=B3*A(6)+B4*A(8)
      A(1)=A(13)-A(9)*CO1(I)-A(10)*CO5(I)
      A(2)=A(14)-A(9)*CO2(I)-A(10)*CO6(I)
      A(3)=A(15)-A(11)*CO1(I)-A(12)*CO5(I)
      A(4)=A(16)-A(11)*CO2(I)-A(12)*CO6(I)
      RHS(1)=RHS(3)-A(9)*V1-A(10)*V2
      RHS(2)=RHS(4)-A(11)*V1-A(12)*V2
      RETURN
      END
```

```
      SUBROUTINE BACSUB
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(30),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D4/BOMEAN,BWMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      GO TO (170,171) LVAR
  170 DO 2 I=1,NM1
      J=NODES-I
      J1=J+1
      PO(J)=PO(J)-CO1(J)*PO(J1)-CO2(J)*DPO(J1)
    2 DPO(J)=DPO(J)-CO5(J)*PO(J1)-CO6(J)*DPO(J1)
      RETURN
  171 DO 3 I=1,NM1
      J=NODES-I
      J1=J+1
      SW(J)=SW(J)-CO1(J)*SW(J1)-CO2(J)*DSW(J1)
    3 DSW(J)=DSW(J)-CO5(J)*SW(J1)-CO6(J)*DSW(J1)
      RETURN
      END


      SUBROUTINE UPSTWT(ST,LI,DELTA,SDELTA,QFRAC,GGP)
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(30),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(3),PODA(3),SWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1PCWDA(30),KRKODA(30),KRKWDA(30),DRKODA(30),DRKWDA(30),FWW(30),
     2DFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      DFT=SDELTA*DFWW(LI-1)+DELTA*DFWW(LI)
      USGP=GGP-DFT*QFRAC
      IE=1FIX(USGP)+1
      IF(IE.LT.2) GO TO 10
      X1=USGP-FLOAT(IE-1)
      X2=X1*X1
      X3=X2*X1
      S3=X2+X2-X3-X3
      S1=1.-S3
      S4=(X3-X2)*DX(IE)
      S2=(X1-X2-X2+X3)*DX(IE)
      ST=S1*SW(IE)+S2*DSW(IE)+S3*SW(IE+1)+S4*DSW(IE-1)
      GO TO 20
   10 ST=SW(1)
   20 RETURN
      END


      SUBROUTINE ASSIGN
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(30),CO2(21),CO5(21),CO6(21),A(16),RHS(4),
     2TITLE(8),SWO(200),DFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D1/PRESDA(3),PODA(3),SWDA(3),VISODA(3),VISWDA(3),SWDA(30),
     1PCWDA(30),KRKODA(30),KRKWDA(30),DRKODA(30),DRKWDA(30),FWW(30),
     2DFWW(30),GWW(30),DGWW(30),NUMPVT,NUMSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D3/XI(5),WEIGHT(5),S(4,5),DS(4,5),SS(16,5),DD(16,5),
     1DDS(16,5),XX(15),WW(15),INUM
      COMMON/D4/BOMEAN,BWMEAN,QINJ,CWINJ,COPROD,CWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      DO 1 I=1,4
    1 A(I)=0.0
      RHS(1)=0.0
      RHS(2)=0.0
      DO 29 I=1,NM1
      QFRAC=USF*QINJ*DT/POR(I)
      I1=I+1
      DO 3 J=5,16
    3 A(J)=0.0
      RHS(3)=0.0
```

```
      RHS(4)=0.0
      DX1=DX(I)
      DX2=DX1*DX1
      STP=SW(I)
      DO 4 II=1,INUM
      GGP=FLOAT(I-1)+XI(II)
      TRAN=TRANS(I)*WEIGHT(II)
      PVOL=POR(I)*WEIGHT(II)/DT
      S1=S(1,II)
      S2=S(2,II)*DX1
      S3=S(3,II)
      S4=S(4,II)*DX1
      D1=DS(1,II)/DX1
      D2=DS(2,II)
      D3=DS(3,II)/DX1
      D4=DS(4,II)
      F1=SS(1,II)
      F2=SS(2,II)*DX1
      F3=SS(3,II)
      F4=SS(4,II)*DX1
      F5=SS(5,II)*DX2
      F6=SS(6,II)*DX1
      F7=SS(7,II)*DX2
      F8=SS(8,II)
      F9=SS(9,II)*DX1
      F10=SS(10,II)*DX2
      PT=S1*PO(I)+S2*DPO(I)+S3*PO(I1)+S4*DPO(I1)
      DPT=D1*PO(I)+D2*DPO(I)+D3*PO(I1)+D4*DPO(I1)
      ST=S1*SW(I)+S2*DSW(I)+S3*SW(I1)+S4*DSW(I1)
      DST=D1*SW(I)+D2*DSW(I)+D3*SW(I1)+D4*DSW(I1)
      PRESER=PT+PDATUM
      DO 5 LI=1,NUMPVT
      IF(PRESER.LT.PRESDA(LI)) GO TO 6
    5 CONTINUE
    6 DELTA=PRESDA(LI)-PRESDA(LI-1)
      ADBOOP=((1.0/BODA(LI))-(1.0/BODA(LI-1)))/DELTA
      ADBWOP=((1.0/BWDA(LI))-(1.0/BWDA(LI-1)))/DELTA
      DELTA=(PRESER-PRESDA(LI-1))/DELTA
      SDELTA=1.0-DELTA
      ABO=SDELTA*BODA(LI-1)+DELTA*BODA(LI)
      ABW=SDELTA*BWDA(LI-1)+DELTA*BWDA(LI)
      AVISO=SDELTA*VISODA(LI-1)+DELTA*VISODA(LI)
      AVISW=SDELTA*VISWDA(LI-1)+DELTA*VISWDA(LI)
      ADENO=DENOSC/ABO
      ADENW=DENWSC/ABW
      COMP=((1.0-ST)*ABO*ADBOOP+ST*ABW*ADBWOP)*PVOL
      ST1=ST
      IF(ST.LT.0.) ST=0.
      IF(LUPST.EQ.0) GO TO 920
      GO TO (810,820,830),LUPST
  810 CONTINUE
      DO 921 LI=1,NUMSW
      IF(ST.LT.SWDA(LI)) GO TO 922
  921 CONTINUE
  922 DELTA=(ST-SWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      SDELTA=1.-DELTA
      CALL UPSTWT(ST,LI,DELTA,SDELTA,QFRAC,GGP)
      GO TO 920
  820 CONTINUE
      STE=ST
      ST=STP
      STP=STE
      GO TO 920
  830 ST=STP
  920 CONTINUE
      DO 7 LI=1,NUMSW
      IF(ST.LT.SWDA(LI)) GO TO 8
    7 CONTINUE
    8 DELTA=(ST-SWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      SDELTA=1.0-DELTA
      ARKO=SDELTA*RRKODA(LI-1)+DELTA*RRKODA(LI)
      ARKW=SDELTA*RRKWDA(LI-1)+DELTA*RRKWDA(LI)
      ADPCOS=(PCWDA(LI)-PCWDA(LI-1))/(SWDA(LI)-SWDA(LI-1))
      T1=TRAN*ARKO/AVISO
      T2=TRAN*ARKW/AVISW
      T3=T2*ADPCOS*DST
      T4=DEPTH(I)*(T1*ADENO+T2*ADENW)
      T5=T1+T2
      GO TO (600,601) LVAR
  600 A(1)=A(1)-(DD(1,II)/DX1)*T5-SS(1,II)*COMP
      T13=DD(2,II)*T5+SS(2,II)*DX1*COMP
      A(2)=A(2)-T13
      A(3)=A(3)-T13
      A(5)=A(5)-(DD(3,II)/DX1)*T5-SS(3,II)*COMP
      A(6)=A(6)-DD(4,II)*T5-SS(4,II)*DX1*COMP
      A(4)=A(4)-DD(5,II)*DX1*T5-SS(5,II)*DX2*COMP
```

```
      A(7)=A(7)-DD(6,II)*T5-SS(6,II)*DX1*COMP
      A(8)=A(8)-DD(7,II)*DX1*T5-SS(7,II)*DX2*COMP
      A(13)=A(13)-(DD(8,II)/DX1)*T5-SS(8,II)*COMP
      A(14)=A(14)-DD(9,II)*T5-SS(9,II)*DX1*COMP
      A(16)=A(16)-DD(10,II)*DX1*T5-SS(10,II)*DX2*COMP
      T7=T4-T3
      T12=COMP*(F1*POOLD(I)+F2*DPOOLD(I)+F3*POOLD(I1)+F4*DPOOLD(I1))
      T13=COMP*(F2*POOLD(I)+F5*DPOOLD(I)+F6*POOLD(I1)+F7*DPOOLD(I1))
      T14=COMP*(F3*POOLD(I)+F6*DPOOLD(I)+F8*POOLD(I1)+F9*DPOOLD(I1))
      T15=COMP*(F4*POOLD(I)+F7*DPOOLD(I)+F9*POOLD(I1)+F10*DPOOLD(I1))
      RHS(1)=RHS(1)-DD(1,II)*T7-T12
      RHS(2)=RHS(2)-DD(2,II)*DX1*T7-T13
      RHS(3)=RHS(3)-DD(3,II)*T7-T14
      RHS(4)=RHS(4)-DD(4,II)*DX1*T7-T15
      GO TO 4
  601 C1=AVISW/AVISC
      S1=ST-SWDA(LI-1)
      S2=ST-SWDA(LI)
      S3=ST-SWDA(LI+1)
      Q1=SWDA(LI)-SWDA(LI+1)
      Q2=SWDA(LI+1)-SWDA(LI-1)
      Q3=SWDA(LI-1)-SWDA(LI)
      R1=-C2*S3/(Q2*Q3)
      R2=-S3*S1/(Q3*Q1)
      R3=-S1*S2/(Q1*Q2)
      FT=R1*FWW(LI-1)+R2*FWW(LI)+R3*FWW(LI+1)
      GT=R1*GWW(LI-1)+R2*GWW(LI)+R3*GWW(LI+1)
      DFT=0.
      DGT=0.
      IF(LIMPSW.EQ.0) GO TO 605
      DFT=SDELTA*DFWW(LI-1)+DELTA*DFWW(LI)
      DGT=SDELTA*DGWW(LI-1)+DELTA*DGWW(LI)
  605 CONTINUE
      ST=ST1
      UU=-T5*DPT-T3+T4
      T6=TRAN*GT*ADPDS
      T7=TRAN*DGT*DST*ADPDS
      T8=(ADENW-ADENO)*DEPTH(I)*TRAN
      T9=PVOL*ADSW*ADSWDP*(PO(I)-POOLD(I))+PVOL
      T10=(UU-DFT+T8*DGT-T7)*DX1
      DD1=DD(1,II)/DX1
      A(1)=A(1)-SS(1,II)*T9+D1*S1*T10-DD1*T6
      DD1=DD(2,II)
      A(2)=A(2)-SS(2,II)*DX1*T9+D1*S2*T10-DD1*T6
      A(3)=A(3)-SS(2,II)*DX1*T9+D2*S1*T10-DD1*T6
      DD1=DD(3,II)/DX1
      A(5)=A(5)-SS(3,II)*T9+D1*S3*T10-DD1*T6
      A(9)=A(9)-SS(3,II)*T9+D3*S1*T10-DD1*T6
      DD1=DD(4,II)
      A(6)=A(6)-SS(4,II)*DX1*T9+D1*S4*T10-DD1*T6
      A(11)=A(11)-SS(4,II)*DX1*T9+D4*S1*T10-DD1*T6
      A(4)=A(4)-SS(5,II)*DX2*T9+D2*S2*T10-DD(5,II)*DX1*T6
      DD1=DD(6,II)
      A(7)=A(7)-SS(6,II)*DX1*T9+D2*S3*T10-DD1*T6
      A(10)=A(10)-SS(6,II)*DX1*T9+D3*S2*T10-DD1*T6
      DD1=DD(7,II)*DX1
      A(8)=A(8)-SS(7,II)*DX2*T9+D2*S4*T10-DD1*T6
      A(12)=A(12)-SS(7,II)*DX2*T9+D4*S2*T10-DD1*T6
      A(13)=A(13)-SS(8,II)*T9+D3*S3*T10-DD(8,II)*T6/DX1
      DD1=DD(9,II)
      A(14)=A(14)-SS(9,II)*DX1*T9+D3*S4*T10-DD1*T6
      A(15)=A(15)-SS(9,II)*D(1*T9+D4*S3*T10-DD1*T6
      A(16)=A(16)-SS(10,II)*DX2*T9+D4*S4*T10-DD(10,II)*DX1*T6
      T11=(UU*(FT-DFT*ST)+T7*ST+T8*(GT-DGT*ST))*DX1
      T12=PVOL*(F1*SWOLD(I)+F2*DSWOLD(I)+F3*SWOLD(I1)+F4*DSWOLD(I1))
      T13=PVOL*(F2*SWOLD(I)+F5*DSWOLD(I)+F6*SWOLD(I1)+F7*DSWOLD(I1))
      T14=PVOL*(F3*SWOLD(I)+F6*DSWOLD(I)+F8*SWOLD(I1)+F9*DSWOLD(I1))
      T15=PVOL*(F4*SWOLD(I)+F7*DSWOLD(I)+F9*SWOLD(I1)+F10*DSWOLD(I1))
      RHS(1)=RHS(1)-D1*T11-T12
      RHS(2)=RHS(2)-D2*T11-T13
      RHS(3)=RHS(3)-D3*T11-T14
      RHS(4)=RHS(4)-D4*T11-T15
    4 CONTINUE
      GO TO (290,291) LVAR
  290 A(9)=A(5)
      A(11)=A(6)
      A(10)=A(7)
      A(12)=A(8)
      A(15)=A(14)
      RHS(1)=RHS(1)-QQ(I)-QW(I)
      GO TO 292
  291 RHS(1)=RHS(1)-QW(I)
  292 CALL THOMAS(I)
   29 CONTINUE
      RHS(1)=RHS(1)-QW(NODES)
      IF(LVAR.EQ.1) RHS(1)=RHS(1)-QQ(NODES)
```

```
      CALL THOMAS(NODES)
      RETURN
      END


      SUBROUTINE PRINT(LPRINT)
      COMMON PO(41),POOLD(41),DPO(21),DPOOLD(21),SW(41),SWOLD(41),
     1DSW(21),DSWOLD(21),CO1(21),CO2(21),CO5(21),CO6(21),A(15),RHS(5),
     2TITLE(8),QWO(200),QFW(200),SP(4,300),X(900),Y(900),FVX,
     3DVX,FVSW,DVSW
      COMMON/D2/AREA(20),ABSK(20),POR(20),TRANS(20),DX(20),DEPTH(41),
     1RRKO(41),RRKW(41),DKO(41),DKW(41),QO(41),QW(41)
      COMMON/D3/XI(5),WEIGHT(5),S(4,6),DS(4,6),SS(15,5),DD(15,5),
     1DDS(15,5),XX(15),WW(15),INUM
      COMMON/D4/BOMEAN,BWMEAN,QINJ,QWINJ,QOPROD,QWPROD,DT,NODES,NM1,
     1PDATUM,DATUM,DENOSC,DENWSC,EPSP,EPSSW,CUMT,ITER,ITMAX,LTSTEP,
     2DPMAX,DSWMAX,IN,IP,INJ,IPROD,LUMP,LUPST,LIMPSW,LVAR,USF,LPP,LSS,
     3NELNUM,GRAV
      WRITE(IP,1)
      TDAYS=CUMT/86400.
      WRITE(IP,2) TDAYS
      WRITE(IP,3)
      SW1=SW(1)
      DO 100 I=1,NM1
      SW2=SW(I+1)
      PPSI=(PO(I)+PDATUM)*14.7
      DPSI=DPO(I)*14.7*30.48
      DSWI=DSW(I)*30.48


      QO1=QO(I)/1.84
      QW1=QW(I)/1.84
      WRITE(IP,5) I,PPSI,DPSI,SW1,DSWI,QO1,QW1
      IF(ABS(SW2-SW1).LT.0.3) GO TO 180
      DX1=DX(I)
      DO 50 J=1,INUM
      S1=S(1,J)
      S2=S(2,J)*DX1
      S3=S(3,J)
      S4=S(4,J)*DX1
      D1=DS(1,J)/DX1
      D2=DS(2,J)
      D3=DS(3,J)/DX1
      D4=DS(4,J)
      PPSI=(S1*PO(I)+S2*DPO(I)+S3*PO(I+1)+S4*DPO(I+1)+PDATUM)*14.7
      SSW=S1*SW1+S2*DSW(I)+S3*SW2+S4*DSW(I+1)
      DPSI=(D1*PO(I)+D2*DPO(I)+D3*PO(I+1)+D4*DPO(I+1))*14.7*30.48
      DSWI=(D1*SW1+D2*DSW(I)+D3*SW2+D4*DSW(I+1))*30.48
      WRITE(IP,51) PPSI,DPSI,SSW,DSWI
   50 CONTINUE
  180 SW1=SW2
  100 CONTINUE
      PPSI=(PO(NODES)+PDATUM)*14.7
      DPSI=DPO(NODES)*14.7*30.48
      DSWI=DSW(NODES)*30.48
      QO1=QO(NODES)/1.84
      QW1=QW(NODES)/1.84
      WRITE(IP,6) NODES,PPSI,DPSI,SW1,DSWI,QO1,QW1
      IF(LPRINT.EQ.0) RETURN
      WRITE(IP,7) QWINJ,QOPROD,QWPROD,LTSTEP
      T1=TDAYS-360.
      IF(ABS(T1).GT.0.01) RETURN
      WRITE(7) TDAYS
      WRITE(7) NODES,(SW(I),I=1,NODES)
      WRITE(7) (DSW(I),I=1,NODES)
      RETURN
    1 FORMAT(1H0,120(1H-))
    2 FORMAT(1H0,6HTIME =,F15.3,5H DAYS)
    3 FORMAT(1H0,4HNODE,8X,7HPO(PSI),4X,11HDPO(PSI/FT),13X,2HSW,7X,6HDSW
     1(/FT),6X,9HQO(STB/D),6X,9HQW(STB/D)/)
    5 FORMAT(1X,I4,F15.2,5F15.6)
   51 FORMAT(4X,1H ,F15.2,3F15.6)
    7 FORMAT(1H0,4X,16HCUM WAT INJ(STB),3X,17HCUM OIL PROD(STB),3X,17HCU
     1M WAT PROD(STB),10X,10HTIME-STEPS/1X,3E20.4,I20)
      END
```

APPENDIX 2.5

2D PROGRAM

```
      PROGRAM CONTRL(INPUT=1000,OUTPUT=1000,TAPE5=INPUT,TAPE6=OUTPUT,
     1TAPE3=1000,TAPE7=1000,TAPE9=1000,TAPE10=1000,TAPE2=1000,TAPE62=100
     20)
      COMMON1/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NM,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON1/G1/NPLOT(150),NCYC3(2,3),NCYC4(2,4),NPERI(28),COORD(2,36),
     1NELTYP(25),NIX(130),NIXEND,NELEMZ,NDIMEN,NTTR,NTRECT,NELDIV,NDOF(2
     2,8)
      READ(5,1000) LCONT,IGFST,LTLIM,QTLIM,QTFCT
      WRITE(6,1010)LCONT,IGFST,LTLIM,QTLIM,QTFCT
      QTLIM=QTLIM*QTFCT
      CALL LIMEX(1)
      NDIMEN=2
      IF(LCONT.NE.1) CALL CONTRL(1)
      IF(LCONT.EQ.2) GO TO 10
      CALL MAIN(LTLIM,QTLIM)
      GO TO 20
   10 CALL CONTRL(2)
   20 IF(LCONT.NE.2) CALL CONTRL(3)
      CALL LIMEX(2)
      STOP
 1000 FORMAT(3I3,2F10.4)
 1010 FORMAT(1H0,3I5,2F10.4)
      END



      BLOCK DATA
      COMMON1/M2/QX(4,3,1),DQX(4,3,1),WQX(4,3,1),WDQX(4,3,1),WQQY(16,3,1)
     1,WDQY(16,3,1),UDWQX(16,3,1),SX(16,1),S(4,3),DS(4,3),S1(3,2),S2(4,2
     2),SS(4,1,2),SQ(16),LS(16,2),LP(16)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NM,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M7/NCDS,NCARD,TPD(5),DTMN(5),DTMX(5),TMX(5),ITM(5),IPT(5),
     1ITSG(5),ITS(5),IPRINT,LPRINT,KPRINT,DTMIN,DTMAX,TPOELT,TDAYS,TMEXT,
     2TPRINT,KTPLOT,IBUG,EPSP,EPSSW,ITMAX,LOE,ITSUM,LST,LLIM
      COMMON1/G1/NPLOT(150),NCYC3(2,3),NCYC4(2,4),NPERI(28),COORD(2,36),
     1NELTYP(25),NIX(130),NIXEND,NELEMZ,NDIMEN,NTTR,NTRECT,NELDIV,NDOF(2
     2,8)
      DATA (NPLOT(I),I=1,136)/2,13,13,0,4,0,0,1,4,4,4,3,3,3,4,4,0,0,
     129,41,57,77,96,98,113,125,137,0,
     22,1,2,2,2,4,2,4,3,2,3,1,3,1,2,3,3,3,5,8,3,8,7,6,3,5,4,1,4,1,2,3,4,
     34,4,6,3,13,4,12,11,10,9,4,9,7,5,1,2,1,2,2,2,3,2,3,1,3,1,2,3,3,3,5,
     46,3,6,4,1,4,1,2,3,4,4,4,7,9,10,4,10,8,5,1,2,1,2,2,2,4,2,4,3,2,3,1,
     53,1,2,2,2,4,2,4,3,2,3,1/,NDOF/2,4,3,8,4,12,2,3,3,6,4,10,4,16,3,9/,
     6NCYC4/2,4,3,1,4,2,1,3/,NCYC3/2,3,3,1,1,2/
      DATA S1/2.,1.,2.,3.,0.,3./,S2/4.,2.,-1.,16.,2.,4.,5.,0.,0.,20.,3.,
     15./,  QQ/156.,22.,84.,-13.,4.,13.,-3.,156.,-22.,4.,210.,35.,0.,0.,
     27.,1.,1.,210.,-35.,7./,CUMT/0./,TDAYS/0./,DPMAX/0./,DSWMAX/0./,
     3LPRINT/0/,LTSTEP/0/,LST/1/,LLIM/2/,NCARD/0/,KPRINT/1/,KTPLOT/0/
     4,DTOLD/0./,NDIMEN/2/
      END

      SUBROUTINE PSMAX
      COMMON P0(150),P01(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NM,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      DPMAX=0.
      DSWMAX=0.
      J=1-NDOFN
      DO 15 I=1,NODES
      J=J+NDOFN
      J1=J
      DIFF=ABS(P0(J1)-P01(J1))
      IF(DPMAX.LT.DIFF) DPMAX=DIFF
      DIFF=ABS(SW(J)-SW1(J))
      IF(DSWMAX.LT.DIFF) DSWMAX=DIFF
   15 CONTINUE
      DO 192 J=1,NDOFT
      P01(J)=P0(J)
      SW1(J)=SW(J)
  192 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MAIN(LTLIM,CTLIM)
      COMMON P0(150),P01(150),SW(150),SW1(150),SWT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF10,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NPOW,NRCW2,I3C,M,
     2MM,NN,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M7/NCDS,NCARD,TPD(5),DTMN(5),DTMX(5),TNX(5),ITM(5),IPT(5),
     1IBG(5),ITS(5),IPRINT,LPRINT,KPRINT,DTMIN,DTMAX,TPDELT,TDAYS,TNEXT,
     2TPRINT,KTPLOT,IBUG,EPSP,EPSSW,ITMAX,LCE,ITSUM,LST,LLIM
      CALL OPENMS(10,IDX10,61,0)
      CALL SECOND(T0)
      LTSNEW=0
      IF(IGFST.EQ.0) GO TO 200
      CALL TWOD2P
      CALL MATBAL(1)
    3 NCARD=NCARD+1
      TPDELT=TPD(NCARD)
      DTMIN=DTMN(NCARD)
      DTMAX=DTMX(NCARD)
      TNEXT=TNX(NCARD)
      ITMAX=ITM(NCARD)
      IPRINT=IPT(NCARD)
      IBUG=IBG(NCARD)
      ITSUM=ITS(NCARD)
      IF(TPDELT.LT.1.E-06)GO TO 300
      IF(LTSTEP.EQ.0) DT=DTMIN
   10 TDAYS=TDAYS+TPDELT
      TDIFF=TNEXT-TDAYS
      IF(TDIFF.LT.1.E-10) GO TO 5
      TPRINT=TDAYS*86400.
      LCE=0
      GO TO 6
    5 TPRINT=TNEXT*86400.
      LCE=1
    6 TDIFF=TPRINT-CUMT
      IF(DT.LT.TDIFF) GO TO 7
      DUM=DT
      DT=TDIFF
      LPRINT=1
    7 CONTINUE
      DO 8 ITER=1,ITMAX
      DO 939 LVAR=LST,LLIM
      IF(LVAR.EQ.1) CALL MATBAL(2)
      CALL RATES
      CALL ASEMBL
      IF(LVAR.NE.1) GO TO 940
      CALL MATBAL(3)
      IF(KPRINT.EQ.0) GO TO 960
      CALL PRINT(1)
      KPRINT=0
  960 IF(KTPLOT.EQ.0) GO TO 940
      CALL TPLOT(1)
      KTPLOT=0
      IF(LCONT.NE.1) GO TO 940
      REWIND 7
      CALL CONTRL(2)
      REWIND 7
  940 CONTINUE
      CALL BACSUB
  939 CONTINUE
      IF(ITSUM.EQ.1) CALL PRINT(0)
    8 CONTINUE
      DTOLD=DT
      CUMT=CUMT+DT
      LTSTEP=LTSTEP+1
      LTSNEW=LTSNEW+1
      CALL PSMAX
      CALL SECOND(T1)
      TD1=T1-T0
      IF(TD1.LT.CTLIM.AND.LTSNEW.LT.LTLIM) GO TO 200
  300 CALL TPLOT(2)
      CALL CLOSMS(10)
      RETURN
  200 CONTINUE
      IF(IPRINT.EQ.1.OR.LPRINT.EQ.1) KPRINT=1
      IF(IBUG.EQ.0) GO TO 11
      WRITE(6,9) CUMT,DT,DPMAX,DSWMAX,ITMAX
   11 IF(LPRINT.EQ.1) GO TO 12
      CALL NEWDT(DT,DTMAX,EPSP,EPSSW,DPMAX,DSWMAX)
      GO TO 6
   12 DT=DUM
```

```
      LPRINT=0
      KTPLOT=1
      IF(LCE.NE.1) GO TO 10
      GO TO 3
    9 FORMAT(1H0,4E15.4,I5/)
      END


      SUBROUTINE ELVEC(JJ,I7)
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M1/Z1(10,10),Z2(10,10),Z3(16,10),Z4(10,16),Z5(4,4),ZA(10),
     1ZB(10),ZD(10),ZF(16),ZX(4),ZY(4),CT(4),DCT(4),PA1(2,3,3),DPA1(2,3,
     23),SA1(2,3,3),DSA1(2,3,3),EP(4,2,3),ES(4,2,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NM,NM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      DO 110 I3=I7,NODE1D
      I10=(I3-1)*NDOFN+1-NDOF1D
      I5=0
      DO 111 I2=1,NODE1D
      I10=I10+NDOF1D
      I1=NGL(JJ,I10)-1
      DO 112 I6=1,NDOFN1
      I5=I5+1
      DO 113 I4=1,NDOFN1
      I1=I1+1
      EP(I5,I4,I3)=PO(I1)
  113 ES(I5,I4,I3)=SW(I1)
  112 CONTINUE
  111 CONTINUE
  110 CONTINUE
      RETURN
      END


      SUBROUTINE RATES
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NM,NM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),CWEL(10,2),QLIM(10),QO(10)
     1,QW(10),QOR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      COMMON/M8/PDATUM,DATUM,SWOC,HT,POP,AKX,AKY,VO,VW,V3,V4,V5,ADPCDS,
     1BOM,BWM,CO,CW,DENOSC,DENWSC,GRAV
      DO 40 I4=2,NWELS
      I1=LWEL(I4,1)
      I2=(LWEL(I4,2)-1)*NDOFN+1
      PT=PO(I2)
      CALL PVT(PT,PDATUM,CO,CW,BO,BW,BOM,BWM)
      GO TO (10,20),I1
   20 QTOT=QLIM(I4)*BW/BWM
      QOR(I4)=0.
      QWR(I4)=QTOT
      DFWDSW(I4)=0.
      GO TO 30
   10 ST=SW(I2)
      IF(ST.GT.1.) ST=1.
      IF(ST.LT.0.) ST=0.
      SO=1.-ST
      CALL SWFUN(ST,SO,VO,VW,V3,V4,V5,SD,TO,TW,FW,DFW,DTW,ADPCDS,1)
      FO=1.-FW
      X1=QLIM(I4)/((FO*BOM/BO)+(FW*BWM/BW))
      DFWDSW(I4)=DFW
      QOR(I4)=-FO*X1
      QWR(I4)=-FW*X1
      QTOT=QOR(I4)+QWR(I4)
   30 QO(I4)=QOR(I4)/(BO*1.84)
      QW(I4)=QWR(I4)/(BW*1.84)
      WCUT(I4)=QW(I4)/(QO(I4)+QW(I4))
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MATSAL(IPOS)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NSOW,NPOW2,I3C,M,
     2IN,NN,NM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CURT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUN,GS(3),WW(3),IB,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),QWEL(10,2),QLIM(10),QQ(10)
     1,QW(10),QOR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      COMMON/M8/PDATUM,DATUM,SWCC,HT,POR,AKX,AKY,VO,VW,V3,V4,V5,ADPCDS,
     1BOM,BWM,CO,CW,DENOSC,DENWSC,GRAV
      GO TO (50,60,70),IPOS
   50 DO 10 I=1,20
   10 WMB(I)=0.
      DO 25 II=1,NY
      IA=LXY(II,2)
      DO 30 JJ=1,NX
      J=LXY(JJ,1)
      IEL1=(IA-1)*LXX+J
      WMB(1)=WMB(1)+VOL(IEL1)*POR*(1.-SWCC)/BOM
      WMB(2)=WMB(2)+VOL(IEL1)*POR*SWCC/BWM
   30 WMB(3)=WMB(3)+VOL(IEL1)*POR
   25 CONTINUE
      RETURN
   60 DO 110 I=4,7
  110 WMB(I)=0.
      DO 80 I=1,NWELS
      I1=LWEL(I,1)+1
      GO TO (80,90,100),I1
   90 WMB(4)=WMB(4)-QQ(I)
      WMB(5)=WMB(5)-QW(I)
      GO TO 80
  100 WMB(6)=0.
      WMB(7)=WMB(7)+QW(I)
   80 CONTINUE
      RETURN
   70 DT1=DTOLD/86400.
      DO 40 I=8,11
   40 WMB(I)=WMB(I)+WMB(I-4)*DT1
      FACT=1.84*86400.
      WMB(14)=(WMB(1)+FACT*(WMB(10)-WMB(8))-WMB(12))/WMB(3)
      WMB(15)=(WMB(2)+FACT*(WMB(11)-WMB(9))-WMB(13))/WMB(3)
      SUM=WMB(4)+WMB(5)
      IF(SUM.GT.1.E-06) WMB(16)=WMB(5)/SUM
      SUM=WMB(8)+WMB(9)
      IF(SUM.GT.1.E-06) WMB(17)=WMB(9)/SUM
      WMB(18)=WMB(11)*FACT*BWM/WMB(3)
      WMB(19)=WMB(8)*FACT*BOM/WMB(3)
      RETURN
      END


      SUBROUTINE SWFUN(ST,SO,VO,VW,V3,V4,V5,SD,TO,TW,FW,DFW,DTW,V6,IFW)
      FD=.7234
      SD=.4472
      TO=SO*SO*V3
      X1=ST*V4
      TW=ST*X1
      DTW=X1+X1
      TTOT=TO+TW
      C1=V4/(TO+TW)
      FW=ST*ST*C1
      DFW=2.*V5*SO*ST*C1*C1
      DTW=0.
      IF(ST.GT.SD) RETURN
      DFW=FD/SD
      FW=DFW*ST
      DTW=DFW*(3.*ST*ST*(V3+V4)+(1.-4.*ST)*V3)
      TW=TTOT*FW
      TO=TTOT-TW
      DTW=0.
      RETURN
      END


      SUBROUTINE PVT(PN,PI,CO,CW,BO,BW,BOM,BWM)
      DELP=PI-PN
      BO=BOM*(1.+CO*DELP)
      BW=BWM*(1.+CW*DELP)
      RETURN
      END
```

```
      SUBROUTINE ELDATA
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I33,M,
     2MM,NM,NM1,LVAP,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IB,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/M8/PDATUM,DATUM,SWCC,HT,POR,AKX,AKY,VO,VW,V3,V4,V5,ADPCDS,
     1BOM,BWM,CO,CW,DENOSC,DENWSC,GRAV
      I2=NX
      DO 810 I1=1,2
      LL=1
      DO 811 I=1,I2
      DO 812 J=1,LL
      DIFF=DL(I,I1)-DL(J,I1)
      IF(ABS(DIFF).LT.1.E-06) GO TO 814
  812 CONTINUE
      LL=LL+1
      DL(LL,I1)=DL(I,I1)
      LXY(I,I1)=LL
      GO TO 811
  814 LXY(I,I1)=J
  811 CONTINUE
      IF(I1.EQ.1) LXX=LL
  810 I2=NY
      LYY=LL
      DO 815 I=1,LXX
  815 DL1(I)=DL(I,1)
      LT1=LXX
      DO 816 I=1,LYY
      DO 817 J=1,LT1
      DIFF=DL(I,2)-DL1(J)
      IF(ABS(DIFF).LT.1.E-06) GO TO 818
  817 CONTINUE
      LT1=LT1+1
      DL1(LT1)=DL(I,2)
      LY(I)=LT1
      GO TO 816
  818 LY(I)=J
  816 CONTINUE
      DO 819 I=1,LT1
  819 DL1(I)=DL1(I)*30.48
      DO 775 II=1,LYY


      I=LY(II)
      DO 776 J=1,LXX
      IEL=(II-1)*LXX+J
  776 VOL(IEL)=HT*DL1(I)*DL1(J)
  775 CONTINUE
      WRITE(6,1000) (LXY(I,1),I=1,NX)
      WRITE(6,1000) (LXY(I,2),I=1,NY)
      WRITE(6,1000) (LY(I),I=1,LYY)
      WRITE(6,1010) (DL1(I),I=1,LT1)
      RETURN
 1000 FORMAT(1H0,16I8)
 1010 FORMAT(1H0,10F13.4)
      END


      SUBROUTINE NEWDT(DT,DTMAX,EPSP,EPSSW,DPMAX,DSWMAX)
      IF(DPMAX.LT.1.E-05) DPMAX=1.E-06
      IF(DSWMAX.LT.1.E-06) DSWMAX=1.E-06
      R1=EPSP/DPMAX
      R2=EPSSW/DSWMAX
      RATIO=AMIN1(R1,R2,2.)
      DT=DT*RATIO
      IF(DT.GT.DTMAX) DT=DTMAX
      RETURN
      END
```

```
      SUBROUTINE PRINT(IPOS)
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWIT(150),V(50),NGL(3,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE10,N1DM1,NDOFN1,
     1NDOFN,NDOF10,I6,I8,I9,NDOFE,NX1,NY1,NODES,NLOFT,NROW,NPOW2,I30,M,
     2MM,NN,NN1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSNMAX
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),QWEL(10,2),QLIM(10),QO(10)
     1,QW(10),QOR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      COMMON/M8/PDATUM,DATUM,SWCC,HT,POR,AKX,AKY,VO,VW,V3,V4,V5,ADPCOS,
     1BOM,BWI,CO,CW,DENOSC,DENWSC,GRAV
      DIMENSION FP(4),FS(4),PPSI(21)
      DATA FP/14.7,44865.6,44865.6,136567468.8/,FS/1.,3048.,3048.,
     1929034./
      WRITE(6,10)
      C1=CUMT/86400.
      WRITE(6,20) C1,WMB(18),WMB(19),LTSTEP,RUNID
      DO 75 I=1,21
   75 PPSI(I)=0.
      DO 77 I2=1,2
      DO 100 I=1,NDOFN
      X1=0.
      IF(I2.EQ.1.AND.I.EQ.1) X1=PDATUM
      I1=(I2-1)*4+I
      GO TO (70,71,72,73,90,91,92,93),I1
   70 WRITE(6,1)
      GO TO 200
   71 WRITE(6,2)
      GO TO 200
   72 WRITE(6,3)
      GO TO 200
   73 WRITE(6,4)
      GO TO 200
   90 WRITE(6,5)
      GO TO 200
   91 WRITE(6,6)
      GO TO 200
   92 WRITE(6,7)
      GO TO 200
   93 WRITE(6,8)
  200 NX2=1
      NX3=15
  202 NX3=MIN0(NX1,NX3)
      IST1=(NX2-1)*NDOFN+I-NROW
      DO 301 J=1,NY1
      IST1=IST1+NROW
      IST=IST1-NDOFN
      NX4=NX3+1-NX2
      DO 302 K=1,NX4
      IST=IST+NDOFN
      IF(I2.EQ.1) PPSI(K)=(PO(IST)+X1)*FP(I)

      IF(I2.EQ.2) PPSI(K)=SW(IST)*FS(I)
  302 CONTINUE
      IF(I2.EQ.1) WRITE(6,81) (PPSI(K),K=1,NX4)
      IF(I2.EQ.2) WRITE(6,82) (PPSI(K),K=1,NX4)
  301 CONTINUE
      IF(NX3.EQ.NX1) GO TO 204
      WRITE(6,83)
      NX2=NX3+1
      NX3=NX3+15
      GO TO 202
  204 CONTINUE
  100 CONTINUE
   77 CONTINUE
      IF(IPOS.EQ.0.OR.NWELS.EQ.1) GO TO 140
      WRITE(6,120)
  120 FORMAT(1H0,6X,4HWELL,6X,4HTYPE,6X,4HNODE,5X,15HOIL RATE(STB/D),3X,
     117HWATER RATE(STB/D),11X,9HWATER CUT)
      DO 130 I=2,NWELS
      I1=I-1
      I2=LWEL(I,2)
      X1=1HP
      IF(LWEL(I,1).EQ.2) X1=1HI
      WRITE(6,160) I1,X1,I2,QO(I),QW(I),WCUT(I)
  130 CONTINUE
      WRITE(6,170) WMB(4),WMB(5),WMB(16)
      WRITE(6,180) WMB(6),WMB(7)
      WRITE(6,190) WMB(8),WMB(9),WMB(17)
      WRITE(6,220) WMB(10),WMB(11)
      WRITE(6,210) WMB(14),WMB(15)
  140 WRITE(6,10)
      RETURN
```

```
    1 FORMAT(1H0,7HPO(PSI)/)
    2 FORMAT(1H0,14HDPX(PSI/100FT)/)
    3 FORMAT(1H0,14HDPY(PSI/100FT)/)
    4 FORMAT(1H0,23HD2P/DXDY(PSI/10000SQFT)/)
    5 FORMAT(1H0,2HSW/)
    6 FORMAT(1H0,11HDSX(/100FT)/)
    7 FORMAT(1H0,11HDSY(/100FT)/)
    8 FORMAT(1H0,20HD2S/DXDY(/10000SQFT)/)
   10 FORMAT(1H0,135(1H-))
   20 FORMAT(1H0,12HTIME(DAYS) =,F10.4,3X,13HPV INJECTED =,F10.6,2X,13HP
     1V PRODUCED =,F10.6,2X,12HTIME-STEPS =,I10,3X,6HRUN ID.=,A10)
   81 FORMAT(1H ,15F9.2)
   82 FORMAT(1H ,15F9.5)
   83 FORMAT(1H0)
  160 FORMAT(1H0,I10,9X,A1,I10,2F20.3,F20.5)
  170 FORMAT(1H0,8X,22HTOTAL PROD RATE(STB/D),2F20.3,F20.5)
  180 FORMAT(1H0,9X,21HTOTAL INJ RATE(STB/D),2F20.3)
  190 FORMAT(1H0,10X,20HCUMULATIVE PROD(STB),2E20.6,F20.5)
  220 FORMAT(1H0,10X,20HCUMULATIVE  INJ(STB),2E20.6)
  210 FORMAT(1H0,14X,16HMATERIAL BALANCE,2E20.6)
      END

      SUBROUTINE CONSTR
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEN,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,IDOF1D,I6,I8,I9,NDOFS,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NN,M1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/ME/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IB,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VCL(1),

     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),QWEL(10,2),QLIM(10),QO(10)
     1,QW(10),QOR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      IF(LVAR.EQ.2) GO TO 105
      I5=M
      I2=L1(NTOT1)-1
C
C.....PRESSURE MATRIX IS SYMMETRIC.
C
      DO 106 I4=1,I2
      I5=I5+1
  106 A(I5)=A(I4)
C
C.....FIXED PRESSURES FOR INCOMPRESSIBLE FLOW.
C
      IF(IFX.EQ.1) GO TO 110
      DO 108 I4=2,IFX
      I5=LFX(I4)+MM
  108 A(I5)=A(I5)+.1E+99
      GO TO 110
C
C.....SATURATION GRADIENTS AT WELLS SET TO ZERO. APPLIES ONLY TO HERMITE
C
  105 IF(IBS.EQ.1) GO TO 110
      DO 112 I4=2,IBS
      I5=LBS(I4)+MM
  112 A(I5)=A(I5)+.1E+99
C
C.....WELL RATES (PROD./INJ.)
C
  110 IF(IWPI.EQ.1) GO TO 118
      DO 804 K=2,IWPI
      I1=LWPI(K,1)
      I3=LWPI(K,3)
      I4=LWPI(K,4)
      I5=I3+M1
      V(I3)=V(I3)+QWR(I4)+FCT*QOR(I4)
      IF(LVAR.EQ.2.AND.I1.EQ.1) A(I5)=A(I5)-DFWDSW(I4)*(QOR(I4)+QWR(I4))
  804 CONTINUE
C
C.....BOUNDARY DERIVATIVES SET TO ZERO. (FOR HERMITES ONLY.)
C
  118 IF(IB.EQ.1) GO TO 122
      DO 124 I4=2,IB
      I5=LB(I4)+MM
  124 A(I5)=A(I5)+.1E+99
  122 RETURN
      END
```

```
      SUBROUTINE BANWIT
      COMMON P0(150),P01(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NN,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CONT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IB,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      NM=0
      M=0
      DO 851 I=1,NY
      I5=0
      NREF=(I-1)*N1DM1*NROW
      DO 852 J=1,NX
      K1=NREF+(J-1)*N1DM1*NDOFN
      I3=0
      DO 806 I2=1,NODE1D
      DO 802 I1=1,I30
      I3=I3+1
  802 NGL(J,I3)=K1+I1
  806 K1=K1+NROW
  852 CONTINUE
C
      DO 860 I1=1,NODE1D
      JSET=1
      IF(I1.GT.1) JSET=0
      IF(I1.EQ.NODE1D.AND.I.LT.NY) JSET=1
C
      DO 862 I2=1,NX
      IF(JSET.EQ.1) NR=NROW2
      I32=N1DM1
      IF(I2.EQ.NX) I32=NODE1D


      DO 868 I3=1,I32
      DO 870 I4=1,NDOFN
      I5=I5+1
      NR=NR-1
  870 L1(I5)=NR
  868 CONTINUE
  862 CONTINUE
  860 CONTINUE
C
      I1=L1(1)
      L1(1)=1
      DO 97 J=2,I5
      I2=L1(J)
      L1(J)=L1(J-1)+I1
   97 I1=I2
C
      I5P1=I5+1
      L1(I5P1)=L1(I5)+I1
      M1=L1(I5P1)-1
      IF(M.LT.M1) M=M1
      I3=NROW*N1DM1
      IF(I.EQ.NY) I3=I3+NROW
      I3P1=I3+1
      IF(NM.LT.I3) NM=I3
      CALL BDRY(I)
      WRITE(8) I5,I5P1,I3,I3P1,NREF,((NGL(J,K),K=1,NDOFE),J=1,NX),
     1(L1(K),K=1,I5P1),IB,(LB(K),K=1,IB),IWPI,((LWPI(J,K),J=1,IWPI),K=1,
     24),IFX,(LFX(K),K=1,IFX),IBS,(LBS(K),K=1,IBS)
      IEL=(I-1)*NX+1
      WRITE(6,1000) IEL,I5,I5P1,I3,I3P1,NREF
      DO 880 J=1,NX
  880 WRITE(6,1000) (NGL(J,K),K=1,NDOFE)
      WRITE(6,1000) (L1(K),K=1,I5P1)
      WRITE(6,1000) IB,(LB(K),K=1,IB),IWPI,((LWPI(J,K),J=1,IWPI),K=1,4),
     1IFX,(LFX(K),K=1,IFX),IBS,(LBS(K),K=1,IBS)
      WRITE(6,1010)
      NDUMP(I,1)=I5*10000+I3P1
      NDUMP(I,2)=L1(I3P1)-1
      NDUMP(I,3)=NREF
  851 CONTINUE
      MM=M+M
      MM1=MM1+NN
      WRITE(6,1000) M,MM,NM,MM1
      RETURN
 1000 FORMAT(1H ,27I5)
 1010 FORMAT(1H ,135(1H-))
      END
```

```
      SUBROUTINE SFCAP
      COMMON/M2/CX(4,3,1),DCX(4,3,1),WCX(4,3,1),WDCX(4,3,1),WCCX(10,3,1)
     1,WDDX(10,3,1),WDWCX(16,3,1),SX(10,1),S(4,3),DS(4,3),S1(3,2),S2(6,2
     2),S3(10,2),SC(10),LS(16,2),LR(16)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I3C,M,
     2MM,NM,MM1,LVAR,FCT,NSW,FUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,COMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(5),WW(3),IB,LB(2.),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),L11,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      X1=6.
      X2=30.
      X3=420.
      I4=0
      DO 729 I0=1,NODE1D
      IY=(I0-1)*NDOFN1
      DO 730 I1=1,NODE1D
      IX=(I1-1)*NDOFN1
      IY1=IY
      DO 731 I2=1,NDOFN1
      IY1=IY1+1
      IX1=IX
      DO 732 I3=1,NDOFN1
      I4=I4+1
      IX1=IX1+1
      LS(I4,1)=IX1
  732 LS(I4,2)=IY1
  731 CONTINUE
  730 CONTINUE
  729 CONTINUE
C
      I1=0
      DO 733 I4=1,I6
      DO 734 I5=I4,I6
      I1=I1+1
      I2=(I4-1)*I6+I5
      I3=(I5-1)*I6+I4
      LR(I2)=I1
  734 LR(I3)=I1
  733 CONTINUE
C
      LP=LUMP+1
      GO TO (750,751,752),NELNO
  750 DO 758 I5=1,I8
  758 SC(I5)=S1(I5,LP)/X1
      GO TO 753
  751 DO 770 I5=1,I8
  770 SC(I5)=S2(I5,LP)/X2
      GO TO 753
  752 DO 759 I5=1,I8
  759 SC(I5)=S3(I5,LP)/X3
  753 CONTINUE
      DO 720 I=1,INUM
      X1=GS(I)
      X2=1.-X1
      GO TO (721,722,723),NELNO
C
C*****CHAPEAU SHAPE FUNCTIONS.
  721 S(1,I)=X2
      S(2,I)=X1
      DS(1,I)=-1.
      DS(2,I)=1.
      GO TO 720
C
C*****QUADRATICS SHAPE FUNCTIONS.
  722 X1=GS(I)
      X2=1.-X1
      X3=1.-2.*X1
      S(1,I)=X2*X3
      S(2,I)=4.*X1*X2
      S(3,I)=-X1*X3
      DS(1,I)=4.*X1-3.
      DS(2,I)=4.*X3
      DS(3,I)=4.*X1-1.
      GO TO 720
C
```

```
C*****HERMITE CUBICS SHAPE FUNCTIONS.
  723 X3=X1*X1
      X4=X2*X2
      S(3,I)=X3*(3.-2.*X1)
      S(1,I)=1.-S(3,I)
      S(2,I)=X1*X4
      S(4,I)=-X2*X3
      DS(3,I)=6.*X1*X2
      DS(1,I)=-DS(3,I)
      DS(2,I)=X2*(1.-3.*X1)
      DS(4,I)=X1*(3.*X1-2.)
  720 CONTINUE


      DO 711 I=1,LT1
      DX=DL1(I)
      X0=1./DX
      DO 712 J=1,INUM
      K3=0
      W1=WW(J)
      DO 705 I4=1,NODE1D
      X1=X0
      DO 706 I5=1,NDOFN1
      X2=X1
      X1=X1*DX
      K3=K3+1
      CX(K3,J,I)=X1*S(K3,J)
      WCX(K3,J,I)=W1*CX(K3,J,I)
      DCX(K3,J,I)=X2*DS(K3,J)
  706 WDCX(K3,J,I)=W1*DCX(K3,J,I)
  705 CONTINUE
      K3=0
      K4=0
      I4=0
      DO 760 I0=1,NODE1D
      X3=X0
      DO 761 I1=1,NDOFN1
      X3=X3*DX
      I4=I4+1
      X1=WCX(I4,J,I)
      X2=WDCX(I4,J,I)
      I5=0
      DO 762 I2=1,NODE1D
      X4=X0
      DO 763 I3=1,NDOFN1
      X4=X4*DX
      I5=I5+1
      K3=K3+1
      WDWCX(K3,J,I)=X2*CX(I5,J,I)
      IF(I5.LT.I4) GO TO 763
      K4=K4+1
C
C*****EVALUATION OF TRANSMISSIBILITY(GN-T-GN) AND CAPACITY(N-T-N)MATRIX.
      WCCX(K4,J,I)=X1*CX(I5,J,I)
      WDDX(K4,J,I)=X2*DCX(I5,J,I)
      IF(J.EQ.1) SX(K4,I)=X3*X4*SC(K4)
  763 CONTINUE
  762 CONTINUE
  761 CONTINUE
  760 CONTINUE
  712 CONTINUE
  711 CONTINUE
      WRITE(6,1000) (LS(I,1),I=1,NDOFE)
      WRITE(6,1000) (LS(I,2),I=1,NDOFE)
      WRITE(6,1000) (LR(I),I=1,I9)
      RETURN
 1000 FORMAT(1H0,16I8)
      END

      SUBROUTINE BODY(II)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NN,M1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CURT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IB,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),QWEL(10,2),QLIM(10),QQ(10)
```

```
     1,QW(10),QOR(10),QWR(10),WCUT(10),QFWDSW(10),WNB(25)
      IB=1
      LB(1)=0
      IWPI=1
      DO 900 K=1,4
  900 LWPI(1,K)=0
      IFX=1
      LFX(1)=0
      IBS=1
      LBS(1)=0
C
      IF(NDOFN1.EQ.1) GO TO 774
      I3=II
      I4=II
      IF(II.EQ.NY) I4=NY1
      DO 760 I=I3,I4
      ND=1
      IF(I.NE.1.AND.I.NE.NY1) ND=NX
      DO 761 J=1,NX1,ND
      I1=(I-1)*NROW+(J-1)*NDOFN+1
      I2=0
      I5=(I-1)*NX1+J
      DO 780 K=1,NWELS
      IF(I5.EQ.LWEL(K,2)) GO TO 761
  780 CONTINUE
      I1=(I-I3)*NROW+(J-1)*NDOFN+1
      IF(J.NE.1.AND.J.NE.NX1) GO TO 762
      IB=IB+1
      LB(IB)=I1+1
      I2=1
  762 IF(I.NE.1.AND.I.NE.NY1) GO TO 763
      IB=IB+1
      LB(IB)=I1+2
      I2=1
  763 IF(I2.EQ.3) GO TO 761
      IB=IB+1
      LB(IB)=I1+3
  761 CONTINUE
  760 CONTINUE
C
  774 I3=(II-1)*N1OM1+1
      I4=II*N1OM1
      IF(II.EQ.NY) I4=I4+1
      DO 772 I=I3,I4
      DO 770 J=1,NX1
      I2=(I-1)*NX1+J
      I1=((I2-(I3-1)*NX1)-1)*NDOFN+1
C
      DO 800 K=1,NWFIX
      IF(I2.NE.LWF(K)) GO TO 800
      IFX=IFX+1
      LFX(IFX)=I1
      GO TO 770
  800 CONTINUE
C
C.....PRODUCTION/INJECTION WELLS.
C
  790 DO 802 K=1,NWELS
      IF(I2.NE.LWEL(K,2)) GO TO 802
      IWPI=IWPI+1
      LWPI(IWPI,1)=LWEL(K,1)
      LWPI(IWPI,2)=(I2-1)*NDOFN+1
      LWPI(IWPI,3)=I1
      LWPI(IWPI,4)=K
      GO TO 794
  802 CONTINUE
      GO TO 770
C
  794 IF(NDOFN1.EQ.1) GO TO 770
      DO 765 K=1,(NDOFN-1)
      IBS=IBS+1
  765 LBS(IBS)=I1+K
  770 CONTINUE
  772 CONTINUE
      RETURN
      END
C
```

```
      SUBROUTINE TWOD2P
      COMMON P0(150),P01(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX1C(65),NDUMP(10,3)
      COMMON/B2/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE10,N1DM1,NDOFN1,
     1NDOFN,NDOF10,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I3C,M,
     2NM,NN,M1,LVAP,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/B5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IS,LE(20),
     1ISS,LSS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/B6/NWFIX,LWF(5),NWELS,LWEL(10,2),CWEL(10,2),QLIM(1 ),QQ(10)
     1,QN(1 )),QQR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      COMMON/B7/NCDS,NCARD,TPD(5),DTMN(5),DTMX(5),TNX(5),ITM(5),IPT(5),
     1IBG(5),ITS(5),IPRINT,LPRINT,KPRINT,DTMIN,DTMAX,IPDELT,TDAYS,TNEXT,
     2TPRINT,KTPLOT,IBUG,EPSP,EPSSW,ITMAX,LCE,ITSUM,LST,LLIM
      COMMON/B8/PDATUM,DATUM,SWCC,HT,POR,AKX,AKY,VO,VW,V3,V4,V5,ADPCDS,
     1BOM,BWM,CO,CW,DENOSC,DENWSC,GRAV
      COMMON/B9/SCALE,HEIGHT,NST(2),NCT(2),NDC(2),DRM(2),DMN(2),LBC(10,2
     1),INBM(10,2),CONTUS(10,2),TITLE(8)
      COMMON/G1/NPLOT(150),NCYC3(2,3),NCYC4(2,4),NPERI(28),COORD(2,36),
     1NELTYP(25),NIX(130),NIXEND,NELEMZ,NDIMEN,NTTR,NTRECT,NELDIV,NDOF(2
     2,8)
      GRAV=.434/(14.7*30.48)
      READ(5,1000) NX,NY,NWELS,INUM,NELNO,LUMP,LUPST,NWFIX
      WRITE(6,1020)NX,NY,NWELS,INUM,NELNO,LUMP,LUPST,NWFIX
      READ(5,1010)  (GS(I),WW(I),I=1,INUM)
      WRITE(6,1030)  (GS(I),WW(I),I=1,INUM)
      READ(5,1010)  EPSP,EPSSW
      WRITE(6,1030) EPSP,EPSSW
      READ(5,1010)  PDATUM,DATUM,SWCC
      WRITE(6,1030) PDATUM,DATUM,SWCC
      READ(5,1010)  HT,POR,AKX,AKY
      WRITE(6,1030) HT,POR,AKX,AKY
      READ(5,1010)  BOM,BWM,CO,CW,DENOSC,DENWSC,VO,VW
      WRITE(6,1030) BOM,BWM,CO,CW,DENOSC,DENWSC,VO,VW
      READ(5,1010) ADPCDS
      WRITE(6,1030)ADPCDS
      READ(5,1010)  (DL(I,1),I=1,NX)

      WRITE(6,1030)  (DL(I,1),I=1,NX)
      READ(5,1010)  (DL(I,2),I=1,NY)
      WRITE(6,1030)  (DL(I,2),I=1,NY)
      NWFIX=NWFIX+1
      LWF(1)=0
      IF(NWFIX.EQ.1) GO TO 200
      READ(5,1000)  (LWF(I),I=2,NWFIX)
      WRITE(6,1020)  (LWF(I),I=2,NWFIX)
  200 NWELS=NWELS+1
      LWEL(1,1)=0
      LWEL(1,2)=0
      QLIM(1)=0.
      IF(NWELS.EQ.1) GO TO 210
      READ(5,1000)  (LWEL(I,1),I=2,NWELS)
      WRITE(6,1020)  (LWEL(I,1),I=2,NWELS)
      READ(5,1000)  (LWEL(I,2),I=2,NWELS)
      WRITE(6,1020)  (LWEL(I,2),I=2,NWELS)
      READ(5,1010)  (QLIM(I),I=2,NWELS)
      WRITE(6,1030)  (QLIM(I),I=2,NWELS)
  210 CONTINUE
C
C.....CONTOURING SPECIFICATIONS.
C
      NTTR=-1
      READ(5,1000) NELDIV,NST(1),NST(2)
      WRITE(6,1020) NELDIV,NST(1),NST(2)
      READ(5,1010) SCALE,HEIGHT
      WRITE(6,1030) SCALE,HEIGHT
      DO 10 I=NST(1),NST(2)
      READ(5,1000) NCT(I),NDC(I)
      WRITE(6,1020)NCT(I),NDC(I)
      READ(5,1000)  (LBC(J,I),J=1,NCT(I))
      WRITE(6,1020)  (LBC(J,I),J=1,NCT(I))
      READ(5,1000)  (INBM(J,I),J=1,NCT(I))
      WRITE(6,1020)  (INBM(J,I),J=1,NCT(I))
      READ(5,1010)  (CONTUS(J,I),J=1,NCT(I))
      WRITE(6,1030)  (CONTUS(J,I),J=1,NCT(I))
      READ(5,1010)  DRM(I),DMN(I)
      WRITE(6,1030) DRM(I),DMN(I)
   10 CONTINUE
C
```

```
      DO 300 I=1,NDOFT,NDOFN
      SW(I)=SWCC
  300 SW1(I)=SWCC
      CALL CMAIN
      CALL ELDATA
      CALL JANWIT
      CALL SECAP
      RETURN
 1000 FORMAT(16I5)
 1010 FORMAT(8F10.4)
 1020 FORMAT(1H0,16I5)
 1030 FORMAT(1H0,8F10.4)
 1040 FORMAT(8A10)
 1050 FORMAT(1H0,8A10)
 1060 FORMAT(4F10.4,4I5)
 1070 FORMAT(1H0,4F10.4,4I5)
      END

      SUBROUTINE BACSUB
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/B3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NFOW,NFOW2,I30,M,
     2MM,NM,M11,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CURT,DUM,
     3DPMAX,DSWMAX
      COMMON/15/NTOT,NTOT1,NELIN,NELIM1,NREF,INUM,GS(3),WW(3),I5,L5(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      DO 100 II=1,NY
      IJ=NY+1-II
C.....READ UPPER TRIANGULAR BLOCKS FOR ROWS IN REVERSE ORDER.
C
      NREC=(IJ-1)*3+1
      I3=NDUMP(IJ,1)/10000
      I1=NDUMP(IJ,1)-I3*10000
      I2=NDUMP(IJ,2)
      NREF=NDUMP(IJ,3)
      NELIM=I1-1
      IF(I3.EQ.NELIM) GO TO 150
      I5=0
      DO 160 I=I1,I3
      I5=I5+1
  160 V(I)=V(I5)
  150 CONTINUE
      CALL READMS(10,L1,I1,NREC)
      NREC=NREC+1
      CALL READMS(10,A,I2,NREC)

      NREC=NREC+1
      CALL READMS(10,V,NELIM,NREC)
C

C.....BACK SUBSTITUTION FROM (NREF+NELIM) TO (NREF+1).
C
      DO 110 I=1,NELIM
      I1=I1-1
      X1=V(I1)
      I2=NREF+I1
      IF(I2.EQ.NDOFT) GO TO 120
      I3=L1(I1)
      I4=L1(I1+1)-1
      I5=I1
      DO 130 J=I3,I4
      I5=I5+1
  130 X1=X1-A(J)*V(I5)
      V(I1)=X1
  120 IF(LVAR.EQ.2) GO TO 140
      PO(I2)=PO(I2)+V(I1)
      GO TO 110
  140 SW(I2)=SW(I2)+V(I1)
  110 CONTINUE
  100 CONTINUE
      RETURN
      END
```

```
C
C.....READ TITLE PAGE CONTENTS,AND WRITE TO GRAPHICS FILE(7).
C
      DO 220 J=1,10
      READ(5,1040)  (TITLE(I),I=1,8)
      WRITE(6,1050)  (TITLE(I),I=1,8)
      WRITE(7)  (TITLE(I),I=1,8)
      IF(TITLE(1).EQ.3HEND) GO TO 320
  220 CONTINUE
C
C.....READ DATA WHICH CONTROLS PRINT-OUTS.
C
  320 DO 230 I=1,10
      READ(5,1060)  TPD(I),DTMN(I),DTMX(I),TMX(I),ITM(I),IPT(I),IBG(I),
     1ITS(I)
      WRITE(6,1070) TPD(I),DTMN(I),DTMX(I),TMX(I),ITM(I),IPT(I),IBG(I),
     1ITS(I)
      IF(TPD(I).LT.1.E-10) GO TO 310
  230 CONTINUE
  310 NCDS=I
      READ(5,1040)  RUNID
      WRITE(6,1050)  RUNID
      GO TO (240,250,260),NELNO
  240 NDOFN1=1
      NODE1D=2
      NELNUM=1
      GO TO 270
  250 NDOFN1=1
      NODE1D=3
      NELNUM=8
      GO TO 270
  260 NDOFN1=2
      NODE1D=2
      NELNUM=7
  270 CONTINUE
C
C.....INVARIANT PARAMETERS OF THE SYSTEM.
C
      NELEM=NX*NY
      NELEMZ=NELEM
      N1DM1=NODE1D-1
      I6=NODE1D*NDOFN1
      I9=I6*I6
      I8=(I9-I6)/2+I6
      NDOFN=NDOFN1*NDOFN1
      NDOF1D=NODE1D*NDOFN


      NDOFE=NDOF1D*NODE1D
      NX1=NX*N1DM1+1
      NY1=NY*N1DM1+1
      NODES=NX1*NY1
      NDOFT=NODES*NDOFN
      NROW=NX1*NDOFN
      I30=NDOF1D
      NROW2=NROW*N1DM1+I30
      DO 330 I=1,NELEM
  330 NELTYP(I)=NELNUM
C.....CONVERSION TO DARCY UNITS.
      DO 280 I=1,NWELS
      QLIM(I)=QLIM(I)*1.84*BWM
      QO(I)=0.
      QW(I)=0.
      QOR(I)=0.
      QWR(I)=0.
  280 WCUT(I)=0.
      PDATUM=PDATUM/14.7
      DATUM=DATUM*30.48
      ADPCDS=-ADPCDS/14.7
      HT=HT*30.48
      CO=CO*14.7
      CW=CW*14.7
      V3=1./VO
      V4=1./VW
      V5=VW*V3
      DO 290 I=1,NDOFT
      PO(I)=0.
      PO1(I)=0.
      SW(I)=0.
      SW1(I)=0.
      SWIT(I)=0.
  290 CONTINUE
```

```
      SUBROUTINE GLOBE(JJ)
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M1/Z1(10,10),Z2(10,10),Z3(16,10),Z4(10,16),Z5(4,4),ZA(10),
     1ZB(10),ZD(10),ZF(16),ZX(4),ZY(4),CT(4),DCT(4),PA1(2,3,3),DPA1(2,3,
     23),SA1(2,3,3),DSA1(2,3,3),EP(4,2,3),ES(4,2,3)
      COMMON/M2/CX(4,3,1),DCX(4,3,1),WCX(4,3,1),WDCX(4,3,1),WCCX(10,3,1)
     1,WDDX(10,3,1),WDWCX(16,3,1),SX(10,1),S(4,3),DS(4,3),S1(3,2),S2(6,2
     22),S3(10,2),SC(10),LS(16,2),LR(16)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I3C,M,
     2MM,NM,M1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IS,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      DO 100 I4=1,NDOFE
      J1=LS(I4,1)
      J2=LS(I4,2)
      J9=(J1-1)*I6
      J10=(J2-1)*I6
      NGLL1=IGL(JJ,I4)
      NGL1=NGLL1-NREF
      DO 101 I5=I4,NDOFE
      J3=LS(I5,1)
      J4=LS(I5,2)
      J5=J9+J3
      J6=J10+J4
      J7=LR(J5)
      J8=LR(J6)
      NGLL2=IGL(JJ,I5)
      NGL2=NGLL2-NREF
      NDIFF=IGL2-NGL1
      IF(NDIFF.EQ.0) GO TO 102
      NTRUE=L1(NGL1)-1+NDIFF

      GO TO 103
102   NTRUE=M1+NGL1
103   IF(LVAR.EQ.2) GO TO 104
      A(NTRUE)=A(NTRUE)+Z1(J7,J8)
      GO TO 101
104   A2=Z2(J7,J8)
      A1=Z1(J7,J8)+A2
      A(NTRUE)=A(NTRUE)+A1+Z3(J5,J8)+Z4(J7,J6)
      V(NGL1)=V(NGL1)-A2*SWIT(NGLL2)
      IF(NDIFF.EQ.0) GO TO 101
      NTRUE=NTRUE+M
      J5=(J3-1)*I6+J1
      J6=(J4-1)*I6+J2
      A(NTRUE)=A(NTRUE)+A1+Z3(J5,J8)+Z4(J7,J6)
      V(NGL2)=V(NGL2)-A2*SWIT(NGLL1)
101   CONTINUE
100   V(NGL1)=V(NGL1)+Z5(J1,J2)
      RETURN
      END


      SUBROUTINE MOVE(II)
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I3C,M,
     2MM,NM,M1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IS,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      NREC=(II-1)*3+1
      I1=NDUMP(II,1)-(NDUMP(II,1)/10000)*10000
      I2=NDUMP(II,2)
      NELIM=I1-1
      CALL WRITMS(10,L1,I1,NREC,-1)
      NREC=NREC+1
      CALL WRITMS(10,A,I2,NREC,-1)
```

```
      NREC=NREC+1
      CALL NREADMS(10,V,NELIM,NREC,-1)
      IF(II.EQ.NY) RETURN
      I5=0
      NELIM1=I1
      I1=L1(NELIM1)
      I2=L1(NTOT1)-1
C
      DO 840 I3=I1,I2
      I5=I5+1
      I5M=I5+MM
      I3M=I3+MM
      A(I5)=A(I3)
  840 A(I5M)=A(I3M)
      I5P1=I5+1
      DO 844 I3=I5P1,I2
      I3M=I3+MM
      A(I3)=0.
  844 A(I3M)=0.
C
      I5=0
      DO 842 I3=NELIM1,NTOT
      I5=I5+1
      I3MM=I3+MM
      I5MM=I5+MM
      A(I5MM)=A(I3MM)
  842 V(I5)=V(I3)
      I5P1=I5+1
      DO 846 I3=I5P1,NTOT
      I3MM=I3+MM
      A(I3MM)=0.
  846 V(I3)=0.
      RETURN
      END

      SUBROUTINE GAUSS
      COMMON PO(150),PO1(150),SW(150),SW1(150),SWT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I30,M,
     2MM,NM,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,ITSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IP,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      DO 1 I=1,NELIM
      N0=I+MM
      X1=1./A(N0)
      X2=X1*V(I)
      V(I)=X2
      IF(I.EQ.NTOT) GO TO 1
      L3=0
      J1=L1(I)
      J2=L1(I+1)-1
      J3=J1+M
      J4=J2+M
      DO 2 J=J1,J2
      L3=L3+1
      N5=N0+L3
      N6=I+L3
      N7=L1(N6)+M-1
      X3=X1*A(J)
      A(J)=X3
      J6=M+J
      V(N6)=V(N6)-A(J6)*X2
      L2=L3
      DO 3 K=J3,J4
      L2=L2-1
      IF(L2) 4,5,6
    4 N9=N7-L2
      GO TO 3
    5 N9=N5
      GO TO 3

    6 J5=N6-L2
      N9=L1(J5)-1+L2
    3 A(N9)=A(N9)-A(K)*X3
    2 CONTINUE
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE ASEMBL
      COMMON P0(150),P01(150),SW(150),SW1(150),SWT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M1/Z1(10,10),Z2(10,10),Z3(16,10),Z4(10,16),Z5(4,4),ZA(10),
     1ZB(10),ZD(10),ZF(16),ZX(4),ZY(4),CT(4),DCT(4),PA1(2,3,3),DPA1(2,3,
     23),SA1(2,3,3),DSA1(2,3,3),EP(4,2,3),ES(4,2,3)
      COMMON/M2/CX(4,3,1),DCX(4,3,1),WCX(4,3,1),WDCX(4,3,1),WCCX(10,3,1)
     1,WDDX(10,3,1),WDWCX(16,3,1),SX(10,1),S(4,3),DS(4,3),S1(3,2),S2(6,2
     2),S3(13,2),SC(10),LS(16,2),LR(16)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE10,N1DM1,NDOFN1,
     1NDOFN,NDOF10,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I3C,M,
     2MM,NM,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IB,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),CWEL(10,2),QLIM(10),QQ(10)
     1,QW(10),QQR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      COMMON/M8/PDATUM,DATUM,SWCC,HT,POR,AKX,AKY,VO,VW,V3,V4,V5,ADPCDS,
     1BOM,BWM,CO,CW,DENOSC,DENWSC,GRAV
      FCT=FLOAT(2-LVAR)
      WMB(12)=0.
      WMB(13)=0.
      DZX=0.
      DZY=0.
      DO 130 I=1,MM1
  130 A(I)=0.
      DO 131 I=1,NM
  131 V(I)=0.
      REWIND 9
      DO 71 II=1,NY
      READ(9) NTOT,NTOT1,NELIM,NELIM1,NREF,((NGL(J,K),K=1,NDOFE),J=1,NX)
     1,(L1(K),K=1,NTOT1),IB,(LB(K),K=1,IB),IWPI,((LWPI(J,K),J=1,IWPI),K=
     21,4),IFX,(LFX(K),K=1,IFX),IBS,(LBS(K),K=1,IBS)
      IA=LXY(II,2)
      I=LY(IA)
      I7=1
      DO 72 JJ=1,NX
      J=LXY(JJ,1)
      IEL1=(IA-1)*LXX+J
      IEL=(II-1)*NX+JJ
      CALL ELVEC(JJ,I7)
      X1=VOL(IEL1)
      W4=AKX*X1
      W5=AKY*X1
      PVOL=POR*X1
      W0=PVOL/DT
      DO 73 I4=1,I8
      X1=W0*SX(I4,I)
      DO 74 I5=1,I8
      Z2(I5,I4)=X1*SX(I5,J)
   74 Z1(I5,I4)=0.
      DO 75 I5=1,I9
      Z3(I5,I4)=0.
   75 Z4(I4,I5)=0.
   73 CONTINUE
      DO 76 I4=1,I6
      DO 76 I5=1,I6
   76 Z5(I5,I4)=0.
      XYMB0=0.
      XYMBW=0.
      DO 77 IY=1,INUM
      DO 78 I5=1,I9
   78 ZD(I5)=0.
      DO 79 I5=1,I8
      ZF(I5)=0.
      ZA(I5)=0.
   79 ZB(I5)=0.
      DO 80 I5=1,I6
      CT(I5)=CX(I5,IY,I)
      DCT(I5)=DCX(I5,IY,I)
      ZX(I5)=0.
   80 ZY(I5)=0.
C
C.....INTERPOLATE IN THE Y DIRECTION.
```

```
C
      DO 81 I3=I7,NODE1D
      DO 84 I4=1,NDOFN1
      YP=0.
      DYP=0.
      YS=0.
      DYS=0.
      DO 85 I5=1,I6
      X1=EP(I5,I4,I3)
      X2=ES(I5,I4,I3)
      YP=YP+CT(I5)*X1
      DYP=DYP+DCT(I5)*X1
      YS=YS+CT(I5)*X2
   85 DYS=DYS+DCT(I5)*X2
      PA1(I4,I3,IY)=YP
      DPA1(I4,I3,IY)=DYP
      SA1(I4,I3,IY)=YS
      DSA1(I4,I3,IY)=DYS
   84 CONTINUE
   81 CONTINUE
      XMBO=0.
      XMBW=0.
      DO 86 IX=1,INUM
      PT=0.
      DPX=0.
      DPY=0.
      ST=0.
      DSX=0.
      DSY=0.
      I4=0
C
CC.....INTERPOLATE IN THE X DIRECTION.
C
      DO 140 I3=1,NODE1D
      DO 87 I5=1,NDOFN1
      I4=I4+1
      X2=CX(I4,IX,J)
      X4=DCX(I4,IX,J)
      PT=PT+X2*PA1(I5,I3,IY)
      ST=ST+X2*SA1(I5,I3,IY)
      DPX=DPX+X4*PA1(I5,I3,IY)
      DSX=DSX+X4*SA1(I5,I3,IY)
      DPY=DPY+X2*DPA1(I5,I3,IY)
      DSY=DSY+X2*DSA1(I5,I3,IY)
   87 CONTINUE
  140 CONTINUE
      CALL SVT(PT,PDATUM,CO,CW,BO,BW,BOM,BWM)
      XMBO=XMBO+WW(IX)*(1.-ST)/BO
      XMBW=XMBW+WW(IX)*ST/BW
      IF(ST.GT.1.) ST=1.
      IF(ST.LT.0.) ST=0.
      SO=1.-ST
      CALL SWFUN(ST,SO,VO,VW,V3,V4,V5,SD,TO,TW,FW,DFW,DTW,ADPCDS,0)
      T1=ADPCDS*TW
      T2=TO*DENOSC+TW*DENWSC
      TSUM=TO+TW
      IF(LVAR.EQ.2) GO TO 18
      W6=W4*TSUM
      W7=W5*TSUM
      UX=-W4*(TSUM*DPX+T1*DSX-T2*DZX)
      UY=-W5*(TSUM*DPY+T1*DSY-T2*DZY)
      W12=UX
      W13=UY
      GO TO 17
   18 CONTINUE
      UX=-W4*(DPX+ADPCDS*DSX-DENWSC*DZX)
      UY=-W5*(DPY+ADPCDS*DSY-DENWSC*DZY)
      W12=TW*UX
      W13=TW*UY
      W10=DTW*UX
      W11=DTW*UY
      W6=W4*T1
      W7=W5*T1
   17 CONTINUE
      DO 88 I5=1,I8
      ZA(I5)=ZA(I5)+W6*WDDX(I5,IX,J)
   88 ZB(I5)=ZB(I5)+W7*WCCX(I5,IX,J)
      DO 89 I5=1,I6
      ZX(I5)=ZX(I5)+W12*WDCX(I5,IX,J)
   89 ZY(I5)=ZY(I5)+W13*WCX(I5,IX,J)
      IF(LVAR.EQ.1) GO TO 86
      DO 90 I5=1,I9
   90 ZD(I5)=ZD(I5)+W10*WDWCX(I5,IX,J)
      DO 91 I5=1,I8
   91 ZF(I5)=ZF(I5)+W11*WCCX(I5,IX,J)
   86 CONTINUE
```

(PLEASE INSERT :)

PT = PT + PDATUM

```
      XYMBO=XYMBO+XMBO*WW(IY)
      XYMBW=XYMBW+XMBW*WW(IY)
      DO 775 I4=1,NDOFN1
      PA1(I4,1,IY)=PA1(I4,NODE1D,IY)
      DPA1(I4,1,IY)=DPA1(I4,NODE1D,IY)
      SA1(I4,1,IY)=SA1(I4,NODE1D,IY)
      DSA1(I4,1,IY)=DSA1(I4,NODE1D,IY)
  775 CONTINUE
      DO 92 I4=1,I8
      X1=WCCX(I4,IY,I)
      X2=WDDX(I4,IY,I)
      DO 93 I5=1,I8
   93 Z1(I5,I4)=Z1(I5,I4)+X1*ZA(I5)+X2*ZB(I5)
   92 CONTINUE
      DO 94 I4=1,I6
      X1=WCX(I4,IY,I)
      X2=WDCX(I4,IY,I)
      DO 95 I5=1,I6
   95 Z5(I5,I4)=Z5(I5,I4)+X2*ZY(I5)+X1*ZX(I5)
   94 CONTINUE
      IF(LVAR.EQ.1) GO TO 77
      DO 96 I4=1,I8
      X1=WCCX(I4,IY,I)
      DO 97 I5=1,I9
   97 Z3(I5,I4)=Z3(I5,I4)+X1*ZD(I5)
   96 CONTINUE
      DO 98 I4=1,I9
      X1=WDWCX(I4,IY,I)
      DO 99 I5=1,I8
   99 Z4(I5,I4)=Z4(I5,I4)+X1*ZF(I5)
   98 CONTINUE
   77 CONTINUE
      WMB(12)=WMB(12)+PVOL*XYMBO
      WMB(13)=WMB(13)+PVOL*XYMBW
      CALL GLOBE(JJ)
      I7=2
   72 CONTINUE
      CALL CONSTR
      CALL GAUSS
      CALL MOVE(II)
   71 CONTINUE
      RETURN
      END


      SUBROUTINE CMAIN
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,I3C,M,
     2MN,MM,MM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,COUNT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IB,LB(20),


     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),CWEL(10,2),QLIM(10),QO(10)
     1,QW(10),QOF(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      COMMON/M9/SCALE,HEIGHT,NST(2),NCT(2),NDC(2),DBM(2),DMN(2),LEC(10,2
     1),INBM(10,2),CONTUS(10,2),TITLE(8)
      COMMON/G1/NPLOT(150),NCYC3(2,3),NCYC4(2,4),NPERI(28),COORD(2,36),
     1NELTYP(25),NIX(130),NIXEND,NELEMZ,NDIMEN,NTTR,NTRECT,NELDIV,NDOF(2
     2,8)
      NIXEND=5*NELEMZ
      NIXE1=NIXEND+1
C
      DXTOT=0.
      DYTOT=0.
      DO 140 I=1,NX
  140 DXTOT=DXTOT+DL(I,1)
      DO 150 I=1,NY
  150 DYTOT=DYTOT+DL(I,2)
      LINTYP=-1
      INTEQ=1
      IBM=16
      NCOR=0
      I1=NODE1D
      Y1=-DL(1,2)/(DYTOT*FLOAT(N1DM1))
      CWEL(1,1)=0.
      CWEL(1,2)=0.
```

```
      DO 120 I=1,NY
      DY1=DL(I,2)/(DYTOT*FLOAT(N1DM1))
      ICOR=0
      DO 122 II=1,I1
      Y1=Y1+DY1
      I2=NODE1D
      X1=-DL(1,1)/(DXTOT*FLOAT(N1DM1))
      DO 124 J=1,NX
      DX1=DL(J,1)/(DXTOT*FLOAT(N1DM1))
      DO 126 JJ=1,I2
      X1=X1+DX1
      ICOR=ICOR+1
      NCOR=NCOR+1
      COORD(1,ICOR)=X1
      COORD(2,ICOR)=Y1
      IF(NWELS.EQ.1) GO TO 126
      DO 162 K=2,NWELS
      IF(NCOR.NE.LWEL(K,2)) GO TO 162
      CWEL(K,1)=X1/SCALE
      CWEL(K,2)=Y1/SCALE
      GO TO 126
  162 CONTINUE
  126 CONTINUE
  124 I2=N1DM1
  122 CONTINUE
      DO 128 K=1,2
      COORD(K,ICOR+1)=0.
  128 COORD(K,ICOR+2)=SCALE
      WRITE(7) ICOR,IBM,LINTYP,INTEQ,(COORD(1,K),COORD(2,K),K=1,(ICOR+2)
     1)
  120 I1=N1DM1
      LINTYP=0
      P0=0.
      P1=1.
      NPTS=2
      WRITE(7) NPTS,IBM,LINTYP,INTEQ,P0,P0,P1,P0,P0,P0,SCALE,SCALE
      WRITE(7) NPTS,IBM,LINTYP,INTEQ,P0,P0,P0,P1,P0,P0,SCALE,SCALE
      Y1=0.
      COORD(1,1)=0.
      COORD(2,1)=0.
      DO 154 J=1,NX
      COORD(2,J+1)=0.
  154 COORD(1,J+1)=COORD(1,J)+(DL(J,1)/DXTOT)
      ICOR=NX+1
      DO 156 I=1,NY
      Y1=Y1+(DL(I,2)/DYTOT)
      ICOR=ICOR+1
      COORD(2,ICOR)=Y1
      COORD(1,ICOR)=0.
      WRITE(7) NPTS,IBM,LINTYP,INTEQ,P0,Y1,P1,Y1,P0,P0,SCALE,SCALE
      X1=0.
      DO 158 J=1,NX
      ICOR=ICOR+1
      COORD(2,ICOR)=Y1
      X1=X1+(DL(J,1)/DXTOT)
      IF(I.EQ.1) WRITE(7) NPTS,IBM,LINTYP,INTEQ,X1,P0,X1,P1,P0,P0,SCALE,
     1SCALE
  158 COORD(1,ICOR)=X1
  156 CONTINUE
      NPTS=-1


      WRITE(7) NPTS,IBM,LINTYP,INTEQ,X1,P0
      NIZZ=0
      N1=NIXE1
      DO 100 I=1,NY
      DO 110 J=1,NX
      N1=N1-1
      NIX(N1)=NIZZ+4
      K1=(I-1)*(NX+1)+J
      NIX(NIZZ+1)=K1
      IF(I.EQ.1.OR.J.EQ.1) NIX(NIZZ+1)=-K1
      NIX(NIZZ+2)=K1+1
      IF(I.EQ.1.OR.J.EQ.NX) NIX(NIZZ+2)=-(K1+1)
      NIX(NIZZ+3)=K1+NX+1
      IF(I.EQ.NY.OR.J.EQ.1) NIX(NIZZ+3)=-(K1+NX+1)
      NIX(NIZZ+4)=K1+NX+2
      IF(I.EQ.NY.OR.J.EQ.NX) NIX(NIZZ+4)=-(K1+NX+2)
  110 NIZZ=NIZZ+4
  100 CONTINUE
      NELEMZ=NX*NY
      RETURN
      END
```

```
      SUBROUTINE INDEX(IPOS)
      COMMON/ 1/PO(250),PO1(250),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,LT1(50),A(3200),IDX16(50),NCOMP(16,3)
      COMMON/ 2/CX(5,5,1),DCX(5,3,1),WCX(5,3,1),NDCX(5,3,1),WDCX(16,3,1)
     1,WDDX(16,3,1),NDWDCX(16,3,1),SX(16,1),S(5,3),DS(5,3),S1(3,5),S2(5,2
     2),S3(25,2),SC(16),CL(16,2),P(16)
      COMMON/ 3/ IX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NEQFN,NDOFE,16,16,19,NDOFE,NX1,NY1,NODES,NDOFT,NROW,NROW2,130,1,


     2 10,10, IM1,LVAR,ECT,NSM,NUMED,LCONT,IGEST,LTST_P,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/ 5/NTOT,NTOT1,NELEM,NELEM1,NAEF,INUM,GS(3),WW(3),IP,LS(25),
     1 10,LS(5),IFX,LFX(5),IMP1,LWP1(5,5),DL(16,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(25,2),LY(25)
      COMMON/ 6/NWFIX,LWF(5),NWELS,LWEL(10,2),CWEL(10,2),QLIM(10),QQ(10)
     1,QW(10),QOR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(20)
      COMMON/ 7/ NODS,NCARD,TPD(5),DTMN(5),DTMX(5),TMX(5),ITM(5),IPT(5),
     1ITSG(5),ITS(5),IPRINT,LPRINT,KPRINT,DTMIN,DTMAX,TPDELT,TDAYS,TNEXT,
     2TPRINT,KTPLOT,IBUG,EPSP,EPSSW,ITMAX,LCOE,ITSUM,LST,LLIM
      COMMON/ 8/ PDATUM,DATUM,SWCC,HT,POR,AKX,AKY,VO,VW,V3,V4,V5,ADPCDS,
     1DEN,DENW,DS,DW,DENDSC,DENWSC,GRAV
      COMMON/ 9/ SCALE,HEIGHT,NST(2),NDC(2),DMN(2),DMN(2),LBC(10,2
     1),INBM(25,2),CONTUS(10,2),TITLE(8)
      COMMON/ 2/ NPLOT(150),NCYCS(2,3),NCYC4(2,4),NPERI(28),COORD(2,36),
     1NELTYP(25),NIX(25),NIXEND,NELEMZ,NDIMEN,NTTR,NIRECT,NELDIV,NDOF(2
     2,8)
      DIMENSION (2,2),IPOS
10    IF(IGEST.EQ.1.AND.LCONT.LT.2) RETURN
      READ(5)  NODES,((COORD(J,I),J=1,2),I=1,NODES),NWELS,((LWEL(J,I),
     1LWEL(J,I),J=1,NWELS),I=1,2),NELNUM,NELEMZ,(NELTYP(I),I=1,NELEMZ),
     2NIXEND,(NIX(I),I=1,NIXEND),NDIMEN,NTTR,SCALE,HEIGHT,NELDIV,NST(1),
     3NST(2),(NDC(I),NDS(I),DMN(I),DMN(I),(LBC(J,I),INBM(J,I),CONTUS(J,I
     4),J=1,NDC(I)),I=NST(1),NST(2))
      IF(LCONT.EQ.2) RETURN
      READ(5)  NX,NY,NELNU,NELEM,LT1,(VOL(I),I=1,LT1),I6,I8,I9,INUM,
     1(((CX(K,J,I),WCX(K,J,I),DCX(K,J,I),WDCX(K,J,I),K=1,16),(WDWCX(K,J,
     2I),K=1,19),(WDDX(K,J,I),WDDX(K,J,I),K=1,18),J=1,INUM),(SX(K,I),K=1
     3,18),DL1(I),I=1,LT1),(GS(I),WW(I),I=1,INUM),NCDS,(TPD(I),DTMN(I),
     4DTMX(I),TMX(I),ITM(I),IPT(I),ITSG(I),ITS(I),I=1,NCDS),NWFIX,(LWF(I)
     5,I=1,NWFIX),(QLIM(I),QQ(I),QW(I),WCUT(I),QOR(I),QWR(I),DFWDSW(I),
     6I=1,NWELS),NDOFT,(PO(I),SW(I),PO1(I),SW1(I),SWIT(I),I=1,NDOFT),
     7NDOFE,((LS(I,J),I=1,NDOFE),J=1,2),(LR(I),I=1,19),(WMB(I),I=1,20)
      READ(5)  IPRINT,LPRINT,KPRINT,KTPLOT,IBUG,COMT,OT,DUM,DTOLD,DPMAX,
     1DSWMAX,EPSP,EPSSW,ITMAX,LCOE,NCARD,ITSUM,LTSTEP,DTMIN,DTMAX,LST,
     2LLIM,LUMP,LUPST,TPDELT,TDAYS,TNEXT,TPRINT,GRAV,PDATUM,DATUM,SWCC,
     3HT,POR,AKX,AKY,BOM,BWR,CO,CW,VO,VW,V3,V4,V5,ADPCDS,DENDSC,DENWSC,
     4NDOFN2,NODE1D,N1DM1,NDOFN,NDOF1D,NX1,NY1,NROW,IBS,NROW2,LXX,LYY,M,
     5NM,NM1,NM,NUMID,(LXY(I,1),I=1,NX),(LXY(I,2),I=1,NY),(LY(I),I=1,LYY
     6),((NDUMP(I,J),I=1,NY),J=1,3)
      RETURN
20    IF(LCONT.EQ.2) RETURN
      WRITE(3)  NODES,((COORD(J,I),J=1,2),I=1,NODES),NWELS,((LWEL(J,I),
     1LWEL(J,I),J=1,NWELS),I=1,2),NELNUM,NELEMZ,(NELTYP(I),I=1,NELEMZ),
     2NIXEND,(NIX(I),I=1,NIXEND),NDIMEN,NTTR,SCALE,HEIGHT,NELDIV,NST(1),
     3NST(2),(NDC(I),NDS(I),DMN(I),DMN(I),(LBC(J,I),INBM(J,I),CONTUS(J,I
     4),J=1,NDC(I)),I=NST(1),NST(2))
      WRITE(3)  NX,NY,NELNU,NELEM,LT1,(VOL(I),I=1,LT1),I6,I8,I9,INUM,
     1(((CX(K,J,I),WCX(K,J,I),DCX(K,J,I),WDCX(K,J,I),K=1,16),(WDWCX(K,J,
     2I),K=1,19),(WDDX(K,J,I),WDDX(K,J,I),K=1,18),J=1,INUM),(SX(K,I),K=1
     3,18),DL1(I),I=1,LT1),(GS(I),WW(I),I=1,INUM),NCDS,(TPD(I),DTMN(I),
     4DTMX(I),TMX(I),ITM(I),IPT(I),ITSG(I),ITS(I),I=1,NCDS),NWFIX,(LWF(I)
     5,I=1,NWFIX),(QLIM(I),QQ(I),QW(I),WCUT(I),QOR(I),QWR(I),DFWDSW(I),
     6I=1,NWELS),NDOFT,(PO(I),SW(I),PO1(I),SW1(I),SWIT(I),I=1,NDOFT),
     7NDOFE,((LS(I,J),I=1,NDOFE),J=1,2),(LR(I),I=1,19),(WMB(I),I=1,20)
      WRITE(3)  IPRINT,LPRINT,KPRINT,KTPLOT,IBUG,COMT,OT,DUM,DTOLD,DPMAX,
     1DSWMAX,EPSP,EPSSW,ITMAX,LCOE,NCARD,ITSUM,LTSTEP,DTMIN,DTMAX,LST,
     2LLIM,LUMP,LUPST,TPDELT,TDAYS,TNEXT,TPRINT,GRAV,PDATUM,DATUM,SWCC,
     3HT,POR,AKX,AKY,BOM,BWR,CO,CW,VO,VW,V3,V4,V5,ADPCDS,DENDSC,DENWSC,
     4NDOFN2,NODE1D,N1DM1,NDOFN,NDOF1D,NX1,NY1,NROW,IBS,NROW2,LXX,LYY,M,
     5NM,NM1,NM,NUMID,(LXY(I,1),I=1,NX),(LXY(I,2),I=1,NY),(LY(I),I=1,LYY
     6),((NDUMP(I,J),I=1,NY),J=1,3)
      RETURN
      END
```

```
      SUBROUTINE TPLOT(IPOS)
      COMMON P0(150),P01(150),SW(150),SW1(150),SWIT(150),V(50),NGL(5,16)
     1,L1(50),A(3200),IDX10(65),NDUMP(10,3)
      COMMON/M3/NX,NY,NELEM,NELNO,NELNUM,LUMP,LUPST,NODE1D,N1DM1,NDOFN1,
     1NDOFN,NDOF1D,I6,I8,I9,NDOFE,NX1,NY1,NODES,NDOFT,NEQW,NRGW2,I3C,M,
     2MM,NM,NM1,LVAR,FCT,NSW,RUNID,LCONT,IGFST,LTSTEP,DT,DTOLD,CUMT,DUM,
     3DPMAX,DSWMAX
      COMMON/M5/NTOT,NTOT1,NELIM,NELIM1,NREF,INUM,GS(3),WW(3),IB,LB(20),
     1IBS,LBS(5),IFX,LFX(5),IWPI,LWPI(5,4),DL(10,2),LT1,DL1(1),VOL(1),
     2LXX,LYY,LXY(10,2),LY(10)
      COMMON/M6/NWFIX,LWF(5),NWELS,LWEL(10,2),QWEL(10,2),QLIM(10),QO(10)
     1,QW(10),QQR(10),QWR(10),WCUT(10),DFWDSW(10),WMB(25)
      COMMON/M8/PDATUM,DATUM,SWCC,HT,POR,AKX,AKY,VO,VW,V3,V4,V5,ADPCDS,
     1BOM,BW1,CO,CW,DENOSC,DENWSC,GRAV
      COMMON/M9/SCALE,HEIGHT,NST(2),NCT(2),NDC(2),DMX(2),DMN(2),LBC(10,2
     1),INBM(10,2),CONTUS(10,2),TITLE(8)
      GO TO (50,60),IPOS
50    CONTINUE
      C1=CUMT/86400.
      WRITE(7) C1,WMB(18),WMB(19)


      DO 40 LCVAR=NST(1),NST(2)
      REWIND 9
      DO 10 II=1,NY
      I1=LXY(II,2)
      I=LY(I1)
      DY=DL1(I)
      READ(9) NTOT,NTOT1,NELIM,NELIM1,NREF,((NGL(J,K),K=1,NDOFE),J=1,NX)
     1,(L1(K),K=1,NTOT1)
      DO 20 JJ=1,NX
      J=LXY(JJ,1)
      DX=DL1(J)
      K=0
      DO 30 KK=1,NDOFE,NDOFN
      DO 32 LL=1,NDOFN
      GO TO(34,35,36,37),LL
34    F1=1.
      GO TO 38
35    F1=DX
      GO TO 38
36    F1=DY
      GO TO 38
37    F1=DX*DY
38    K=K+1
      K1=NGL(JJ,K)
      GO TO (42,44),LCVAR
42    X1=0.
      IF(LL.EQ.1) X1=PDATUM
      V(K)=(X1+PO(K1))*14.7*F1
      GO TO 32
44    V(K)=SW(K1)*F1
32    CONTINUE
30    CONTINUE
      WRITE(7) K,(V(KK),KK=1,K)
20    CONTINUE
10    CONTINUE
40    CONTINUE
      IF(LCONT.NE.1) RETURN
60    C1=-1.
      WRITE(7) C1,WMB(18),WMB(19)
      RETURN
      END
```

APPENDIX 2.6

GRAPHS AND CONTOUR MAPS

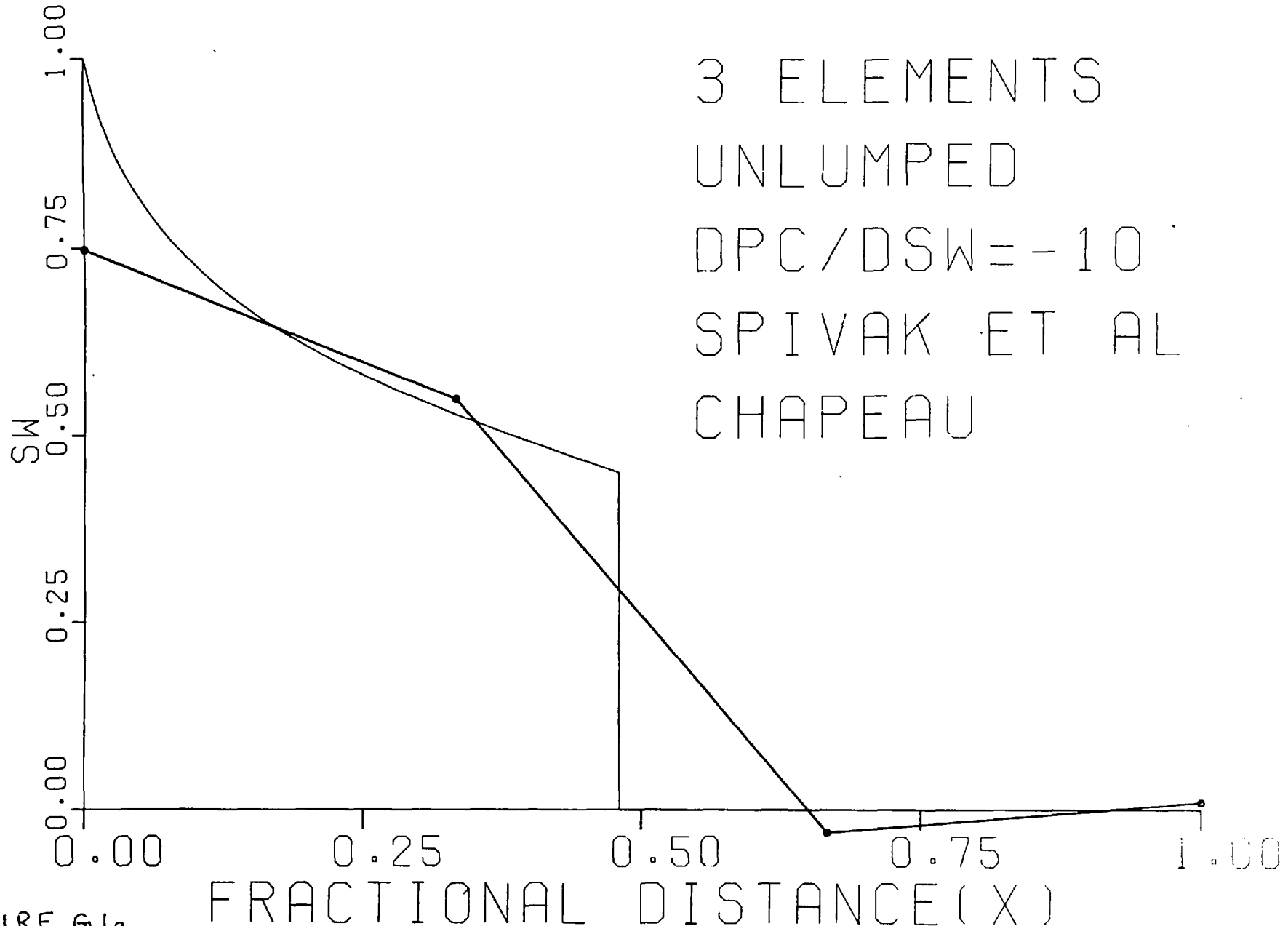

3 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G1a

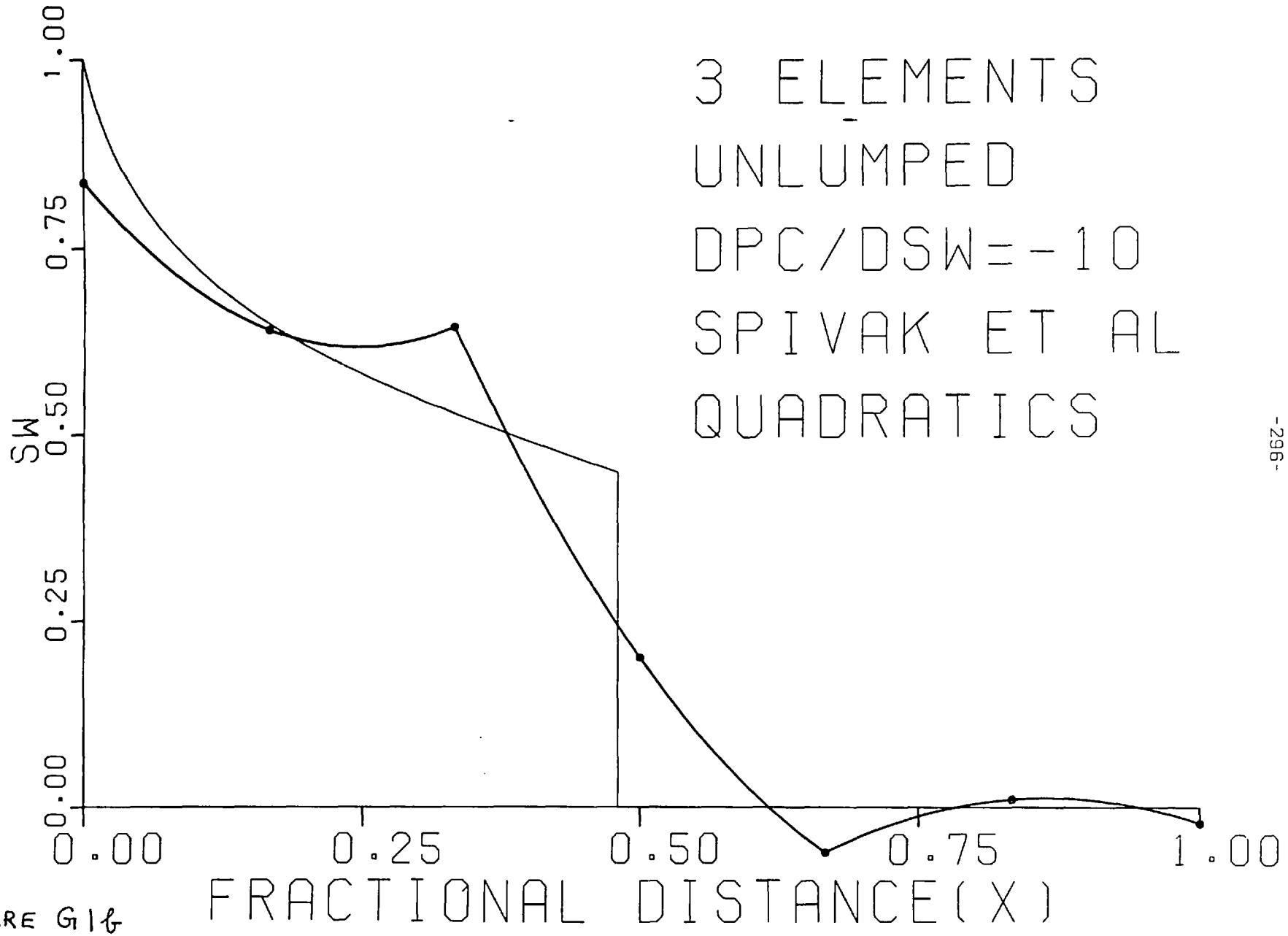

3 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G16

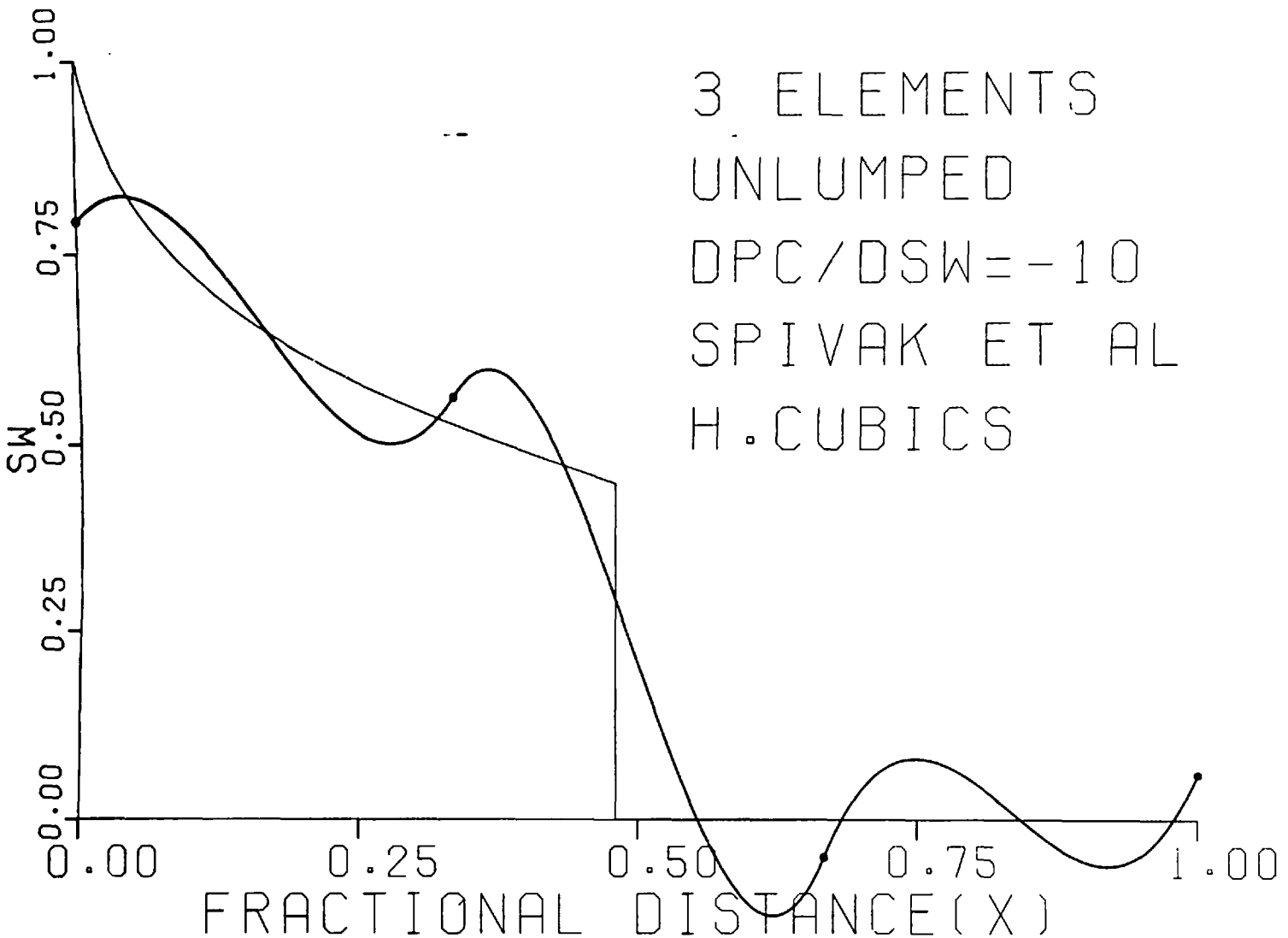

3 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G1c

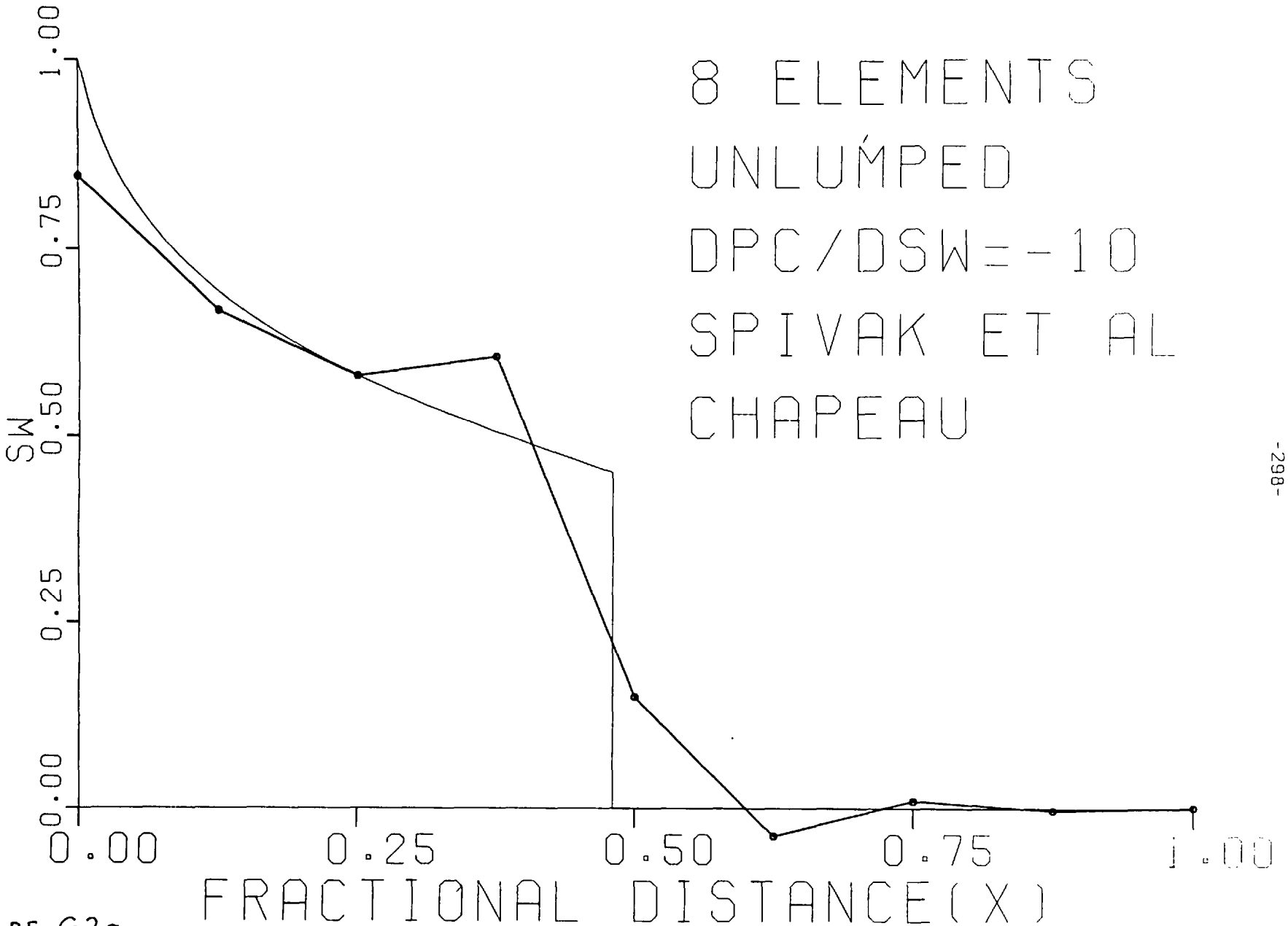

8 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G2a

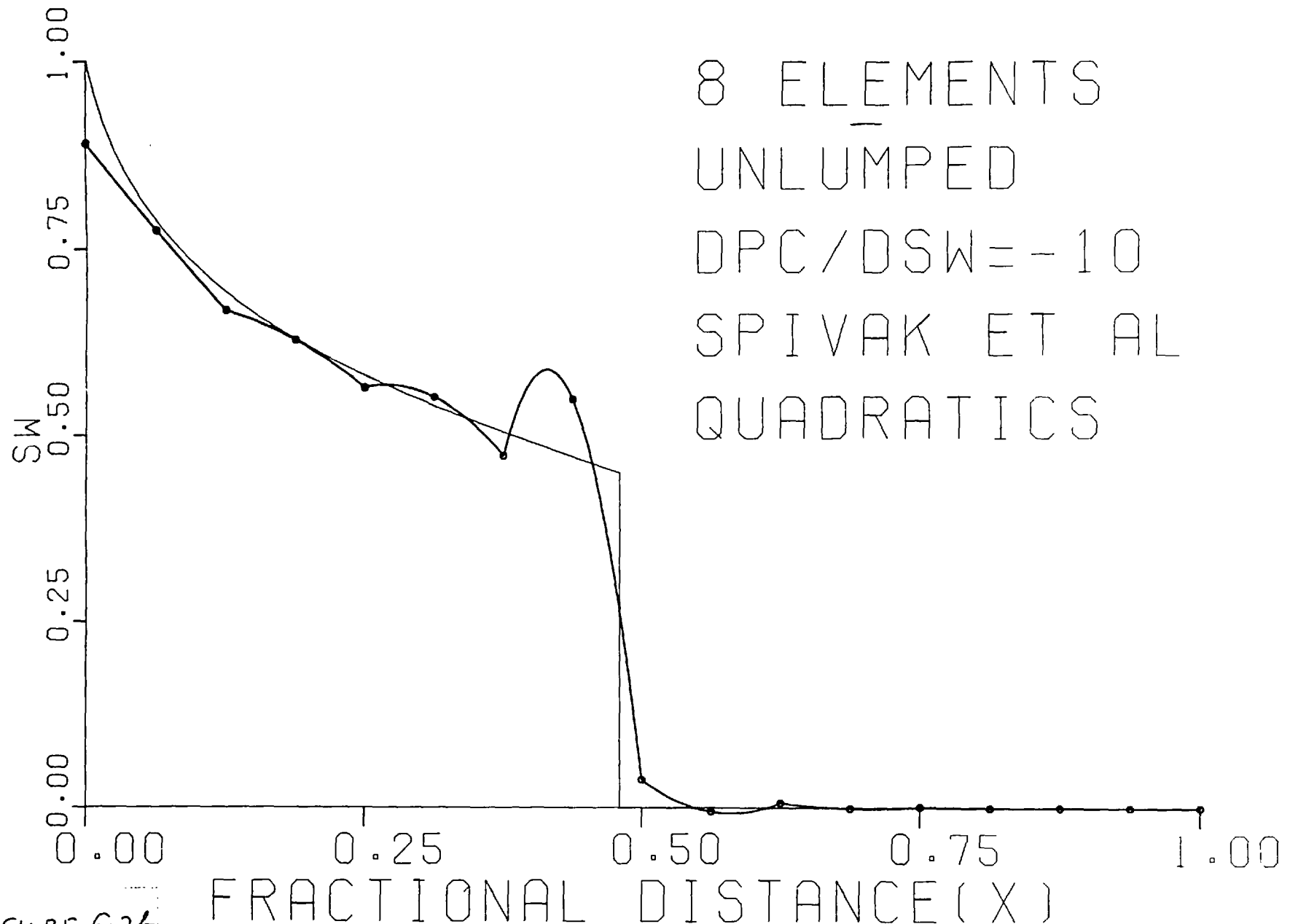

8 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G26

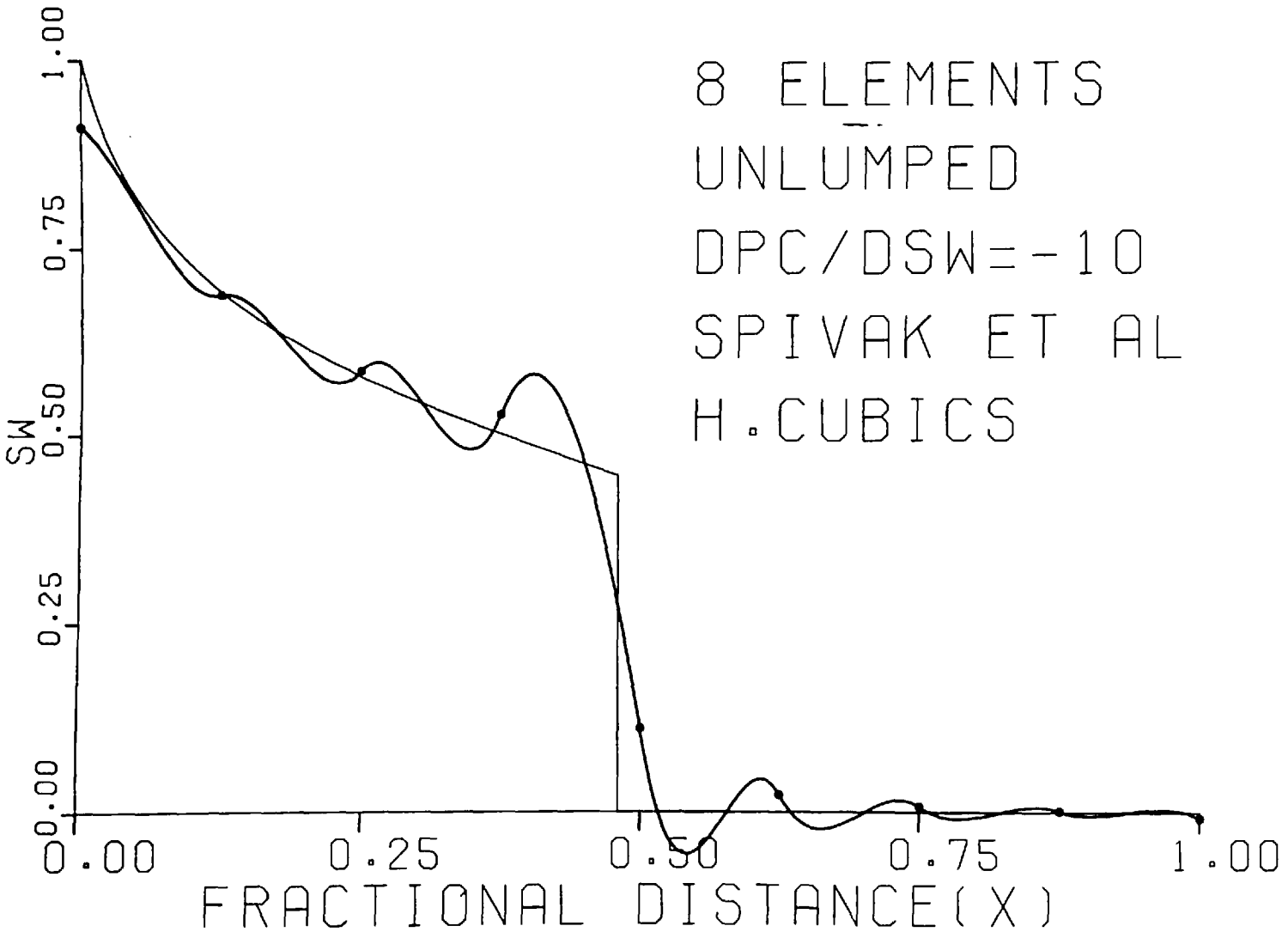

8 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G2c

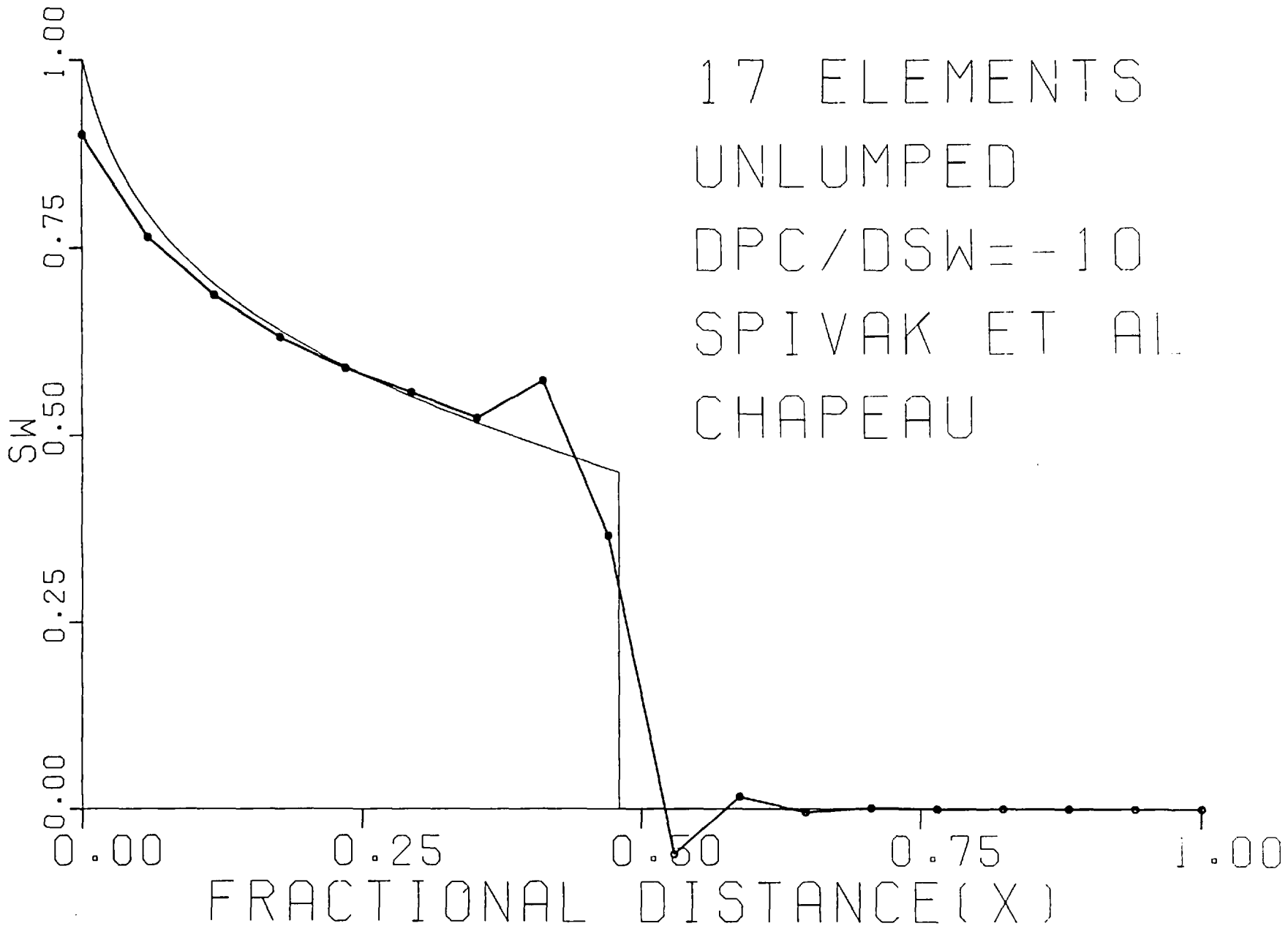

17 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G3a

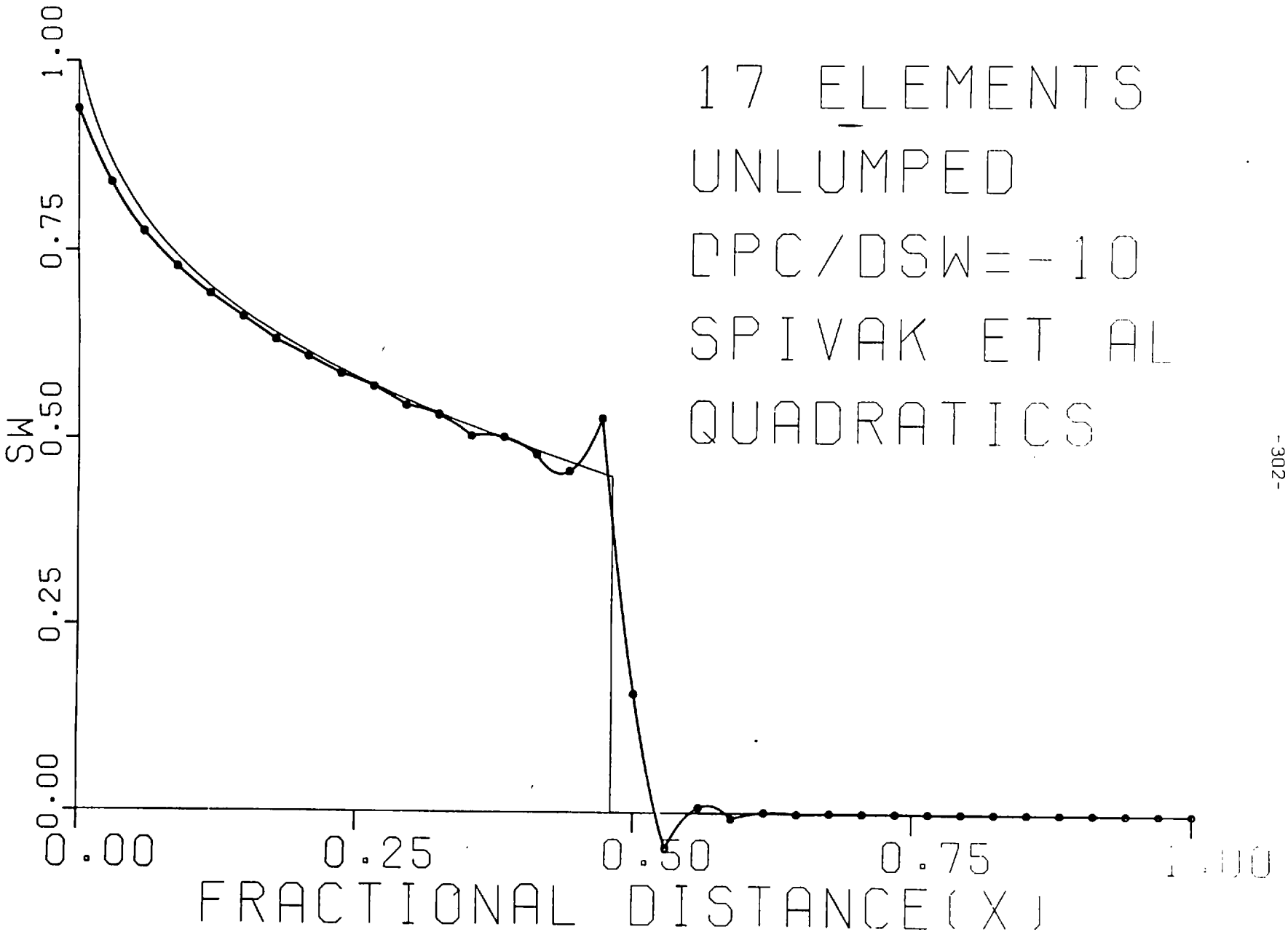

17 ELEMENTS
UNLUMPED
CPC/DSW=-10
SPIVAK ET AL
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G3b

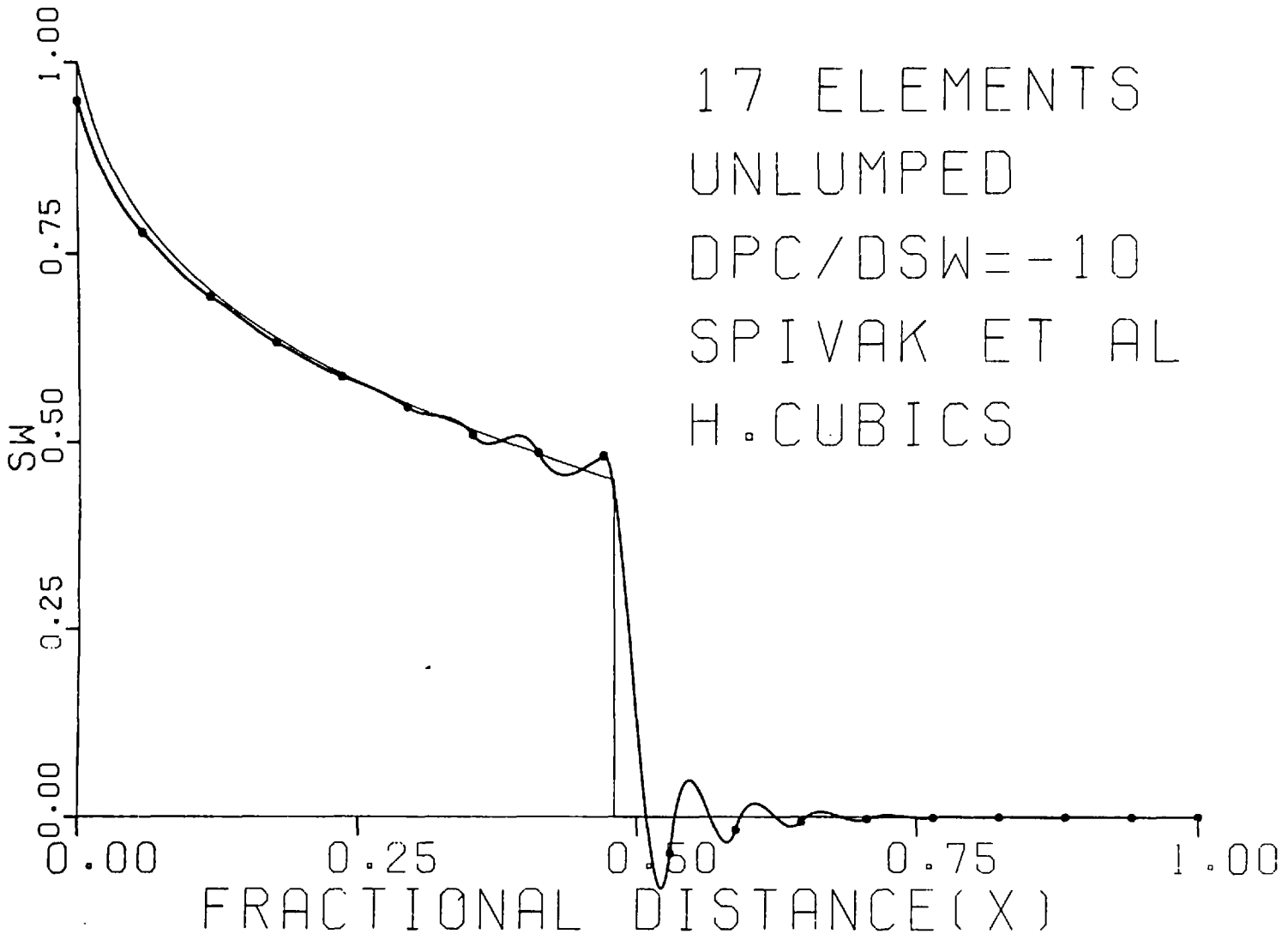

17 ELEMENTS
UNLUMPED
DPC/DSW=-10
SPIVAK ET AL
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G3c

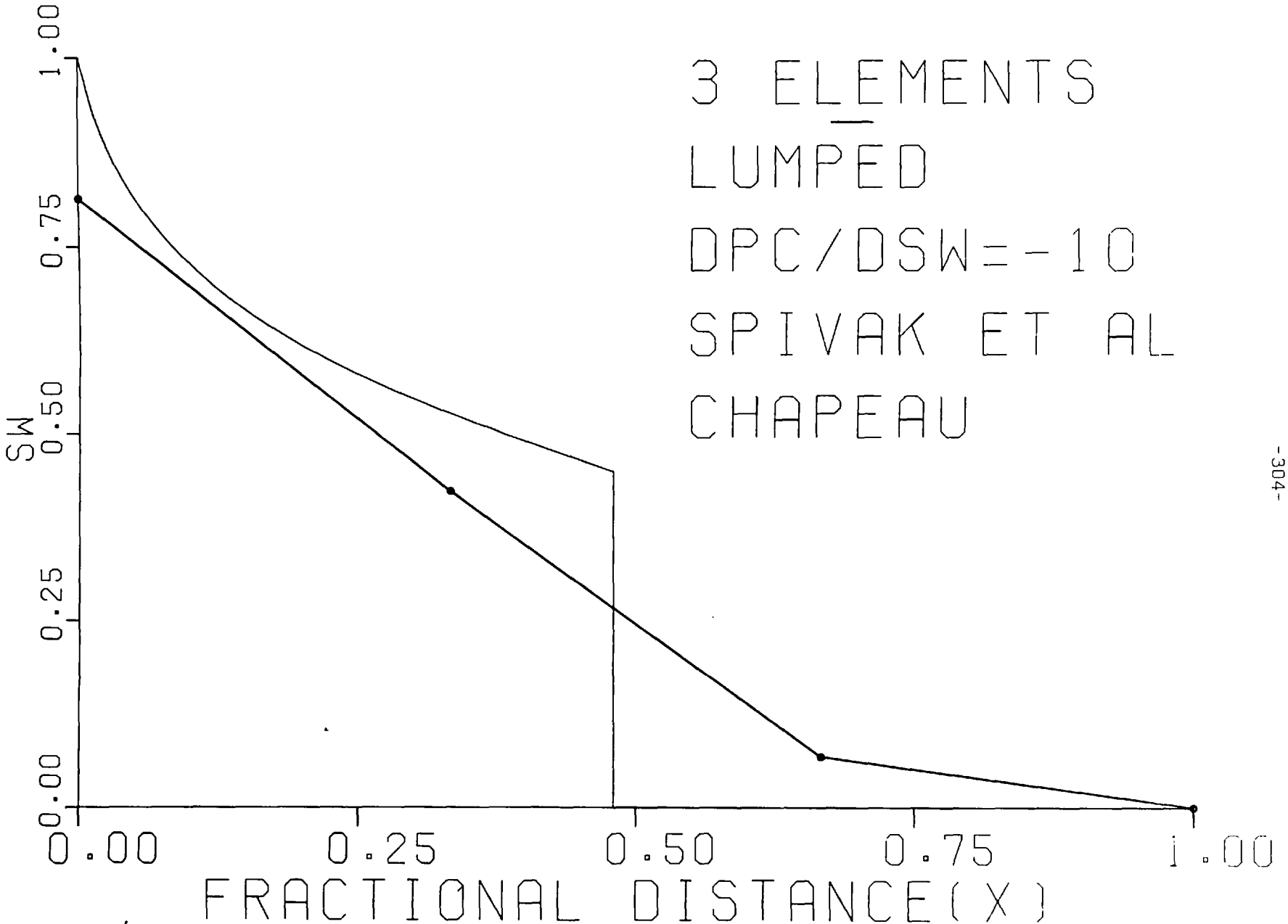

3 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G4a

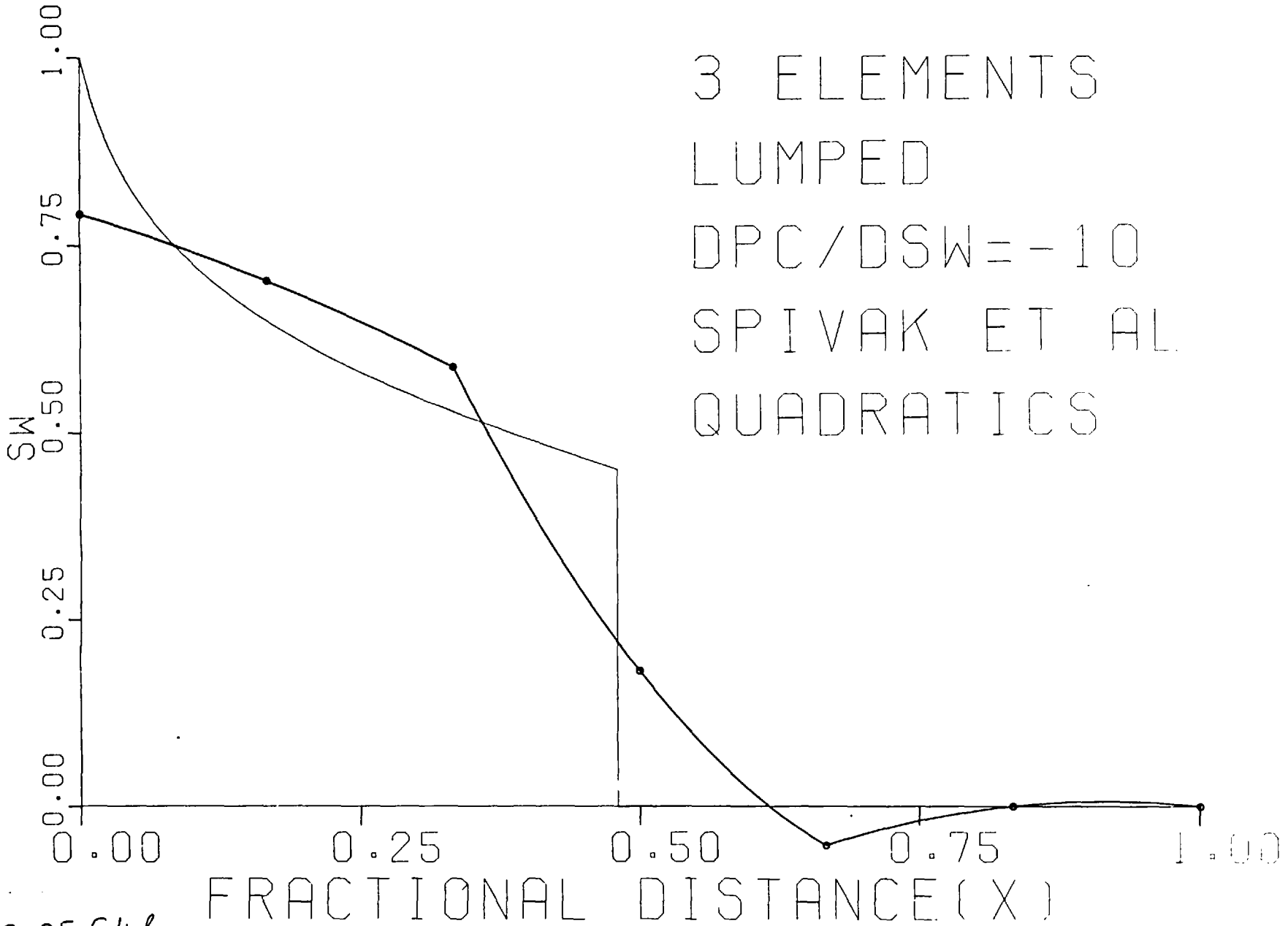

3 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G4b

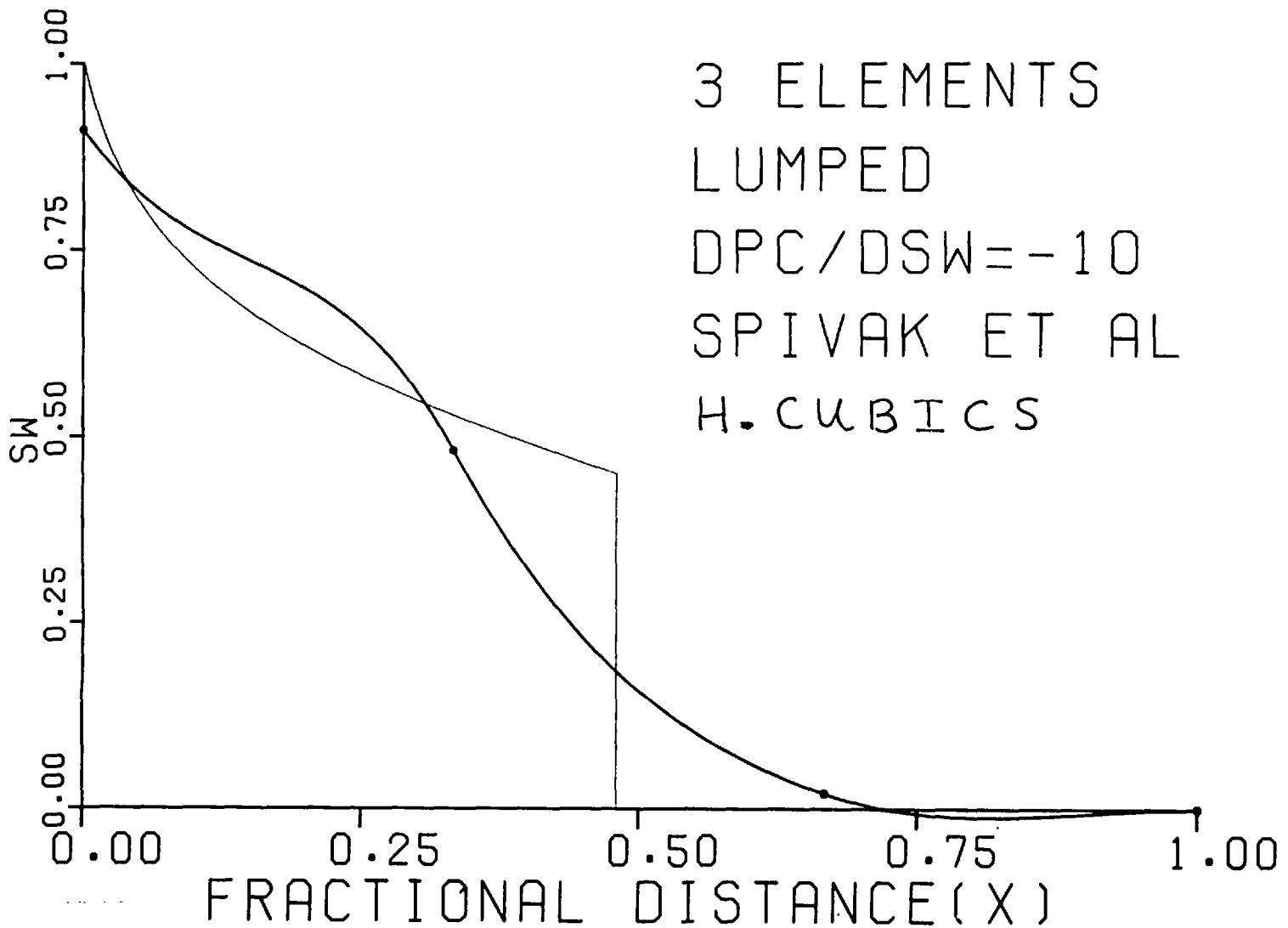

3 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G4c

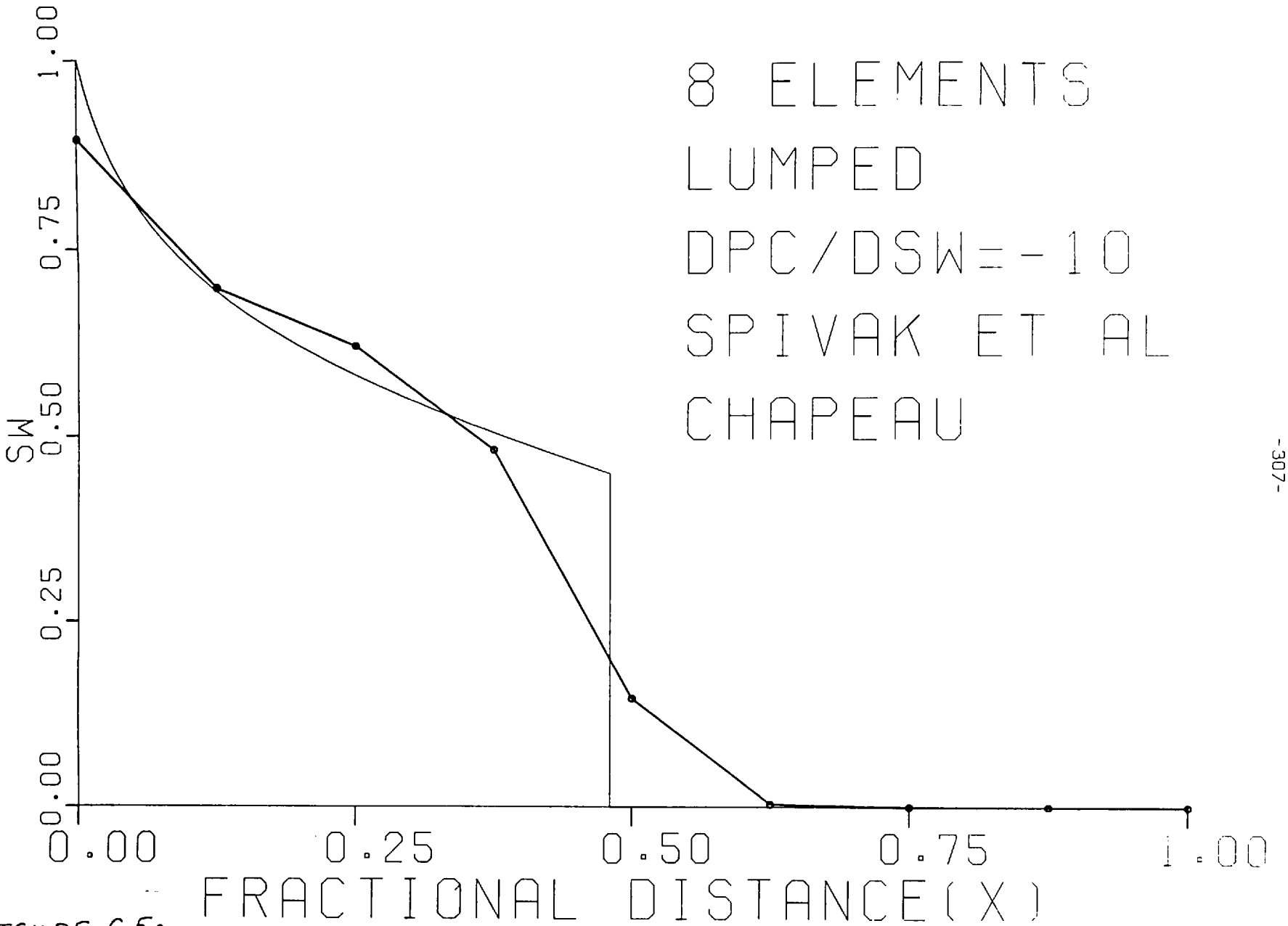

8 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G5a

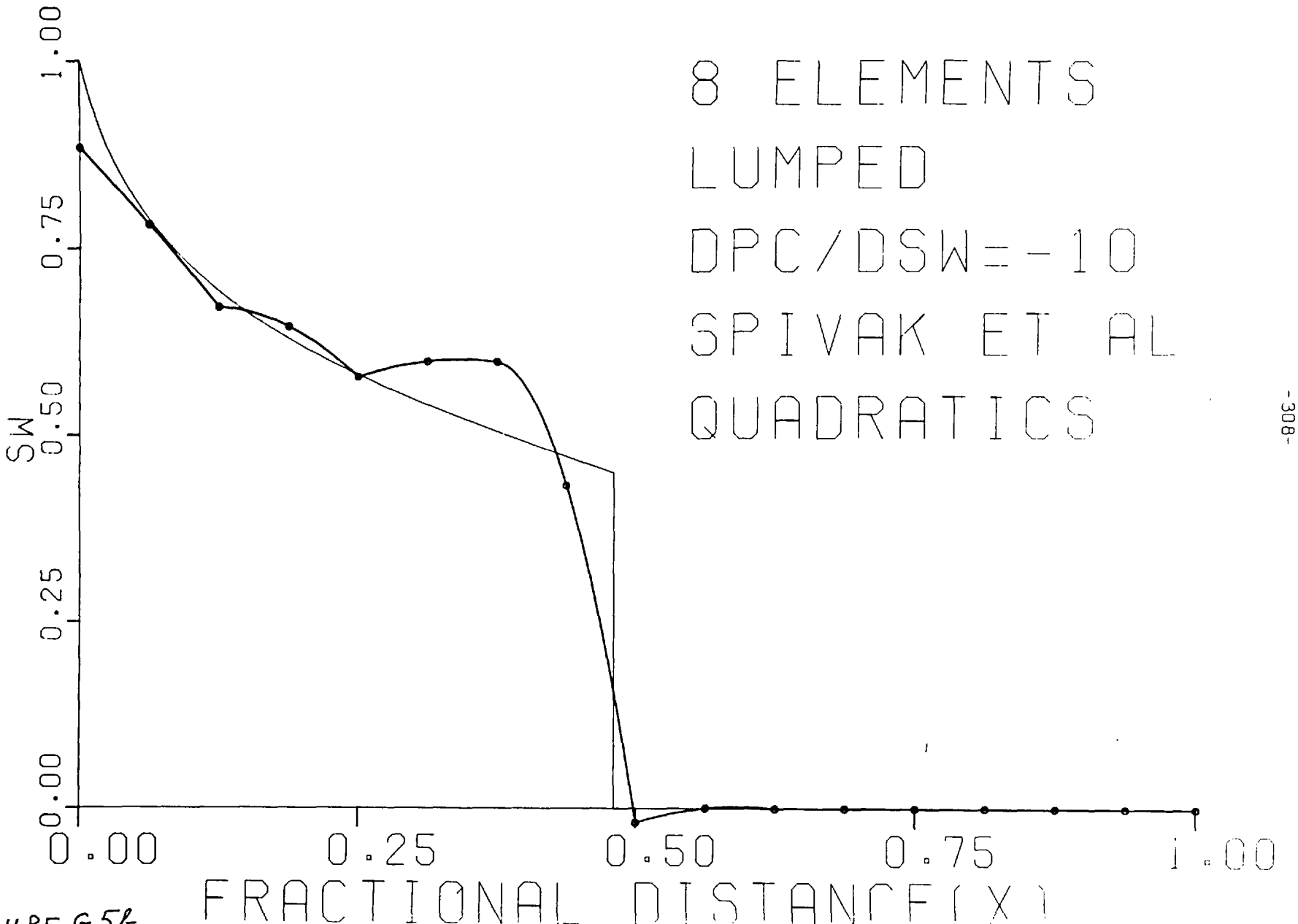

8 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G5b

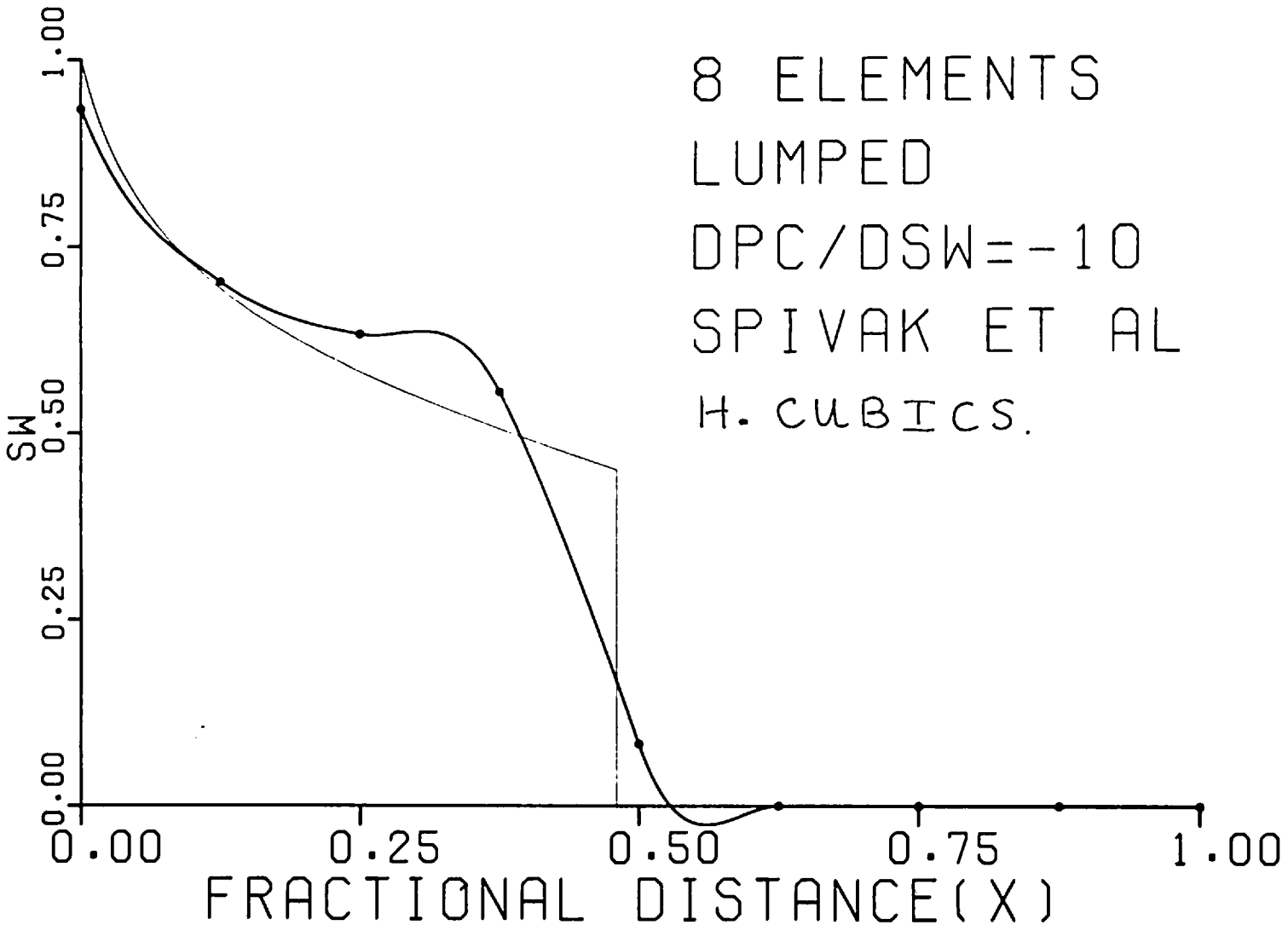

8 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
H. CUBICS.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G5c

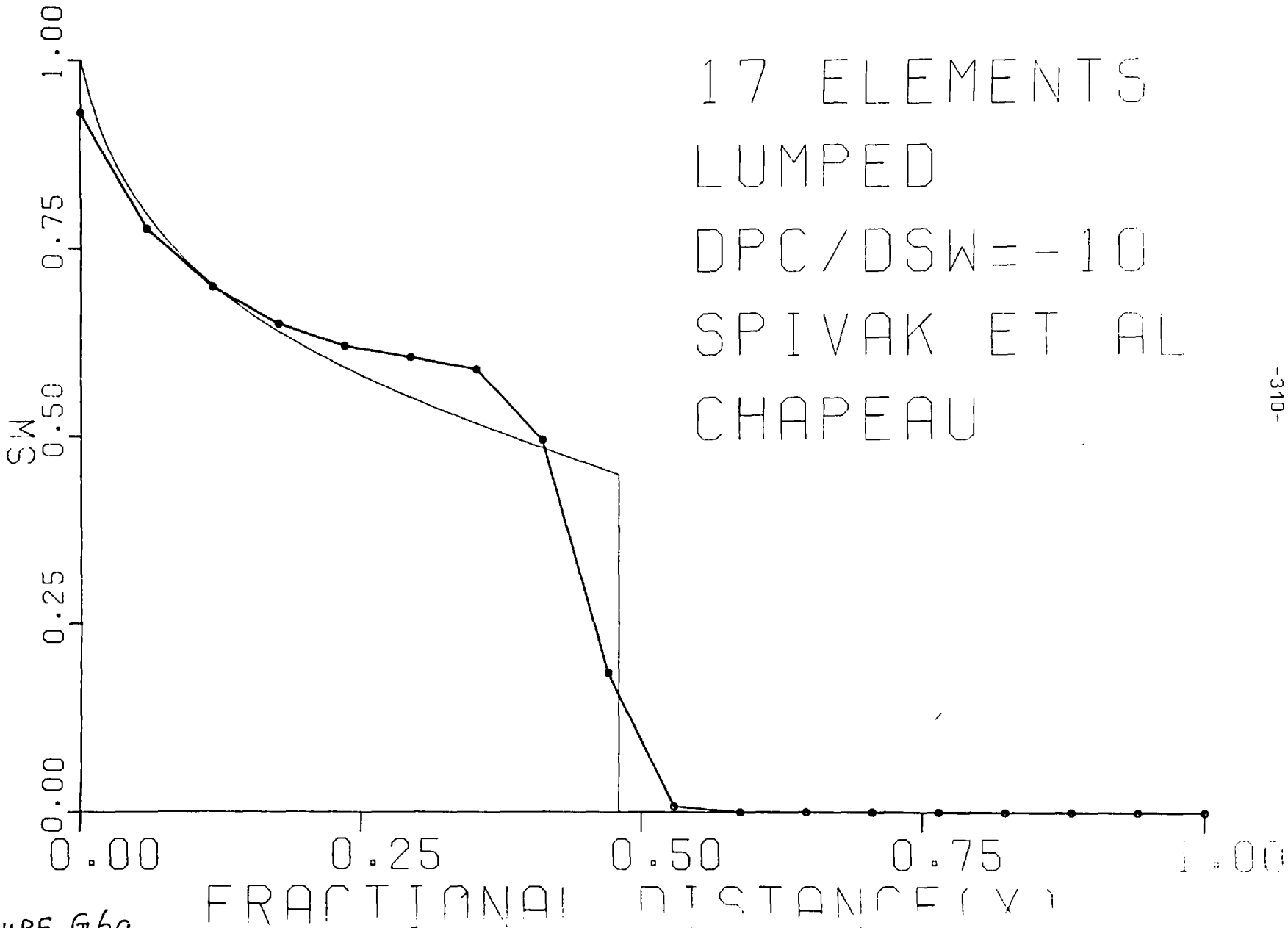

17 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G6a

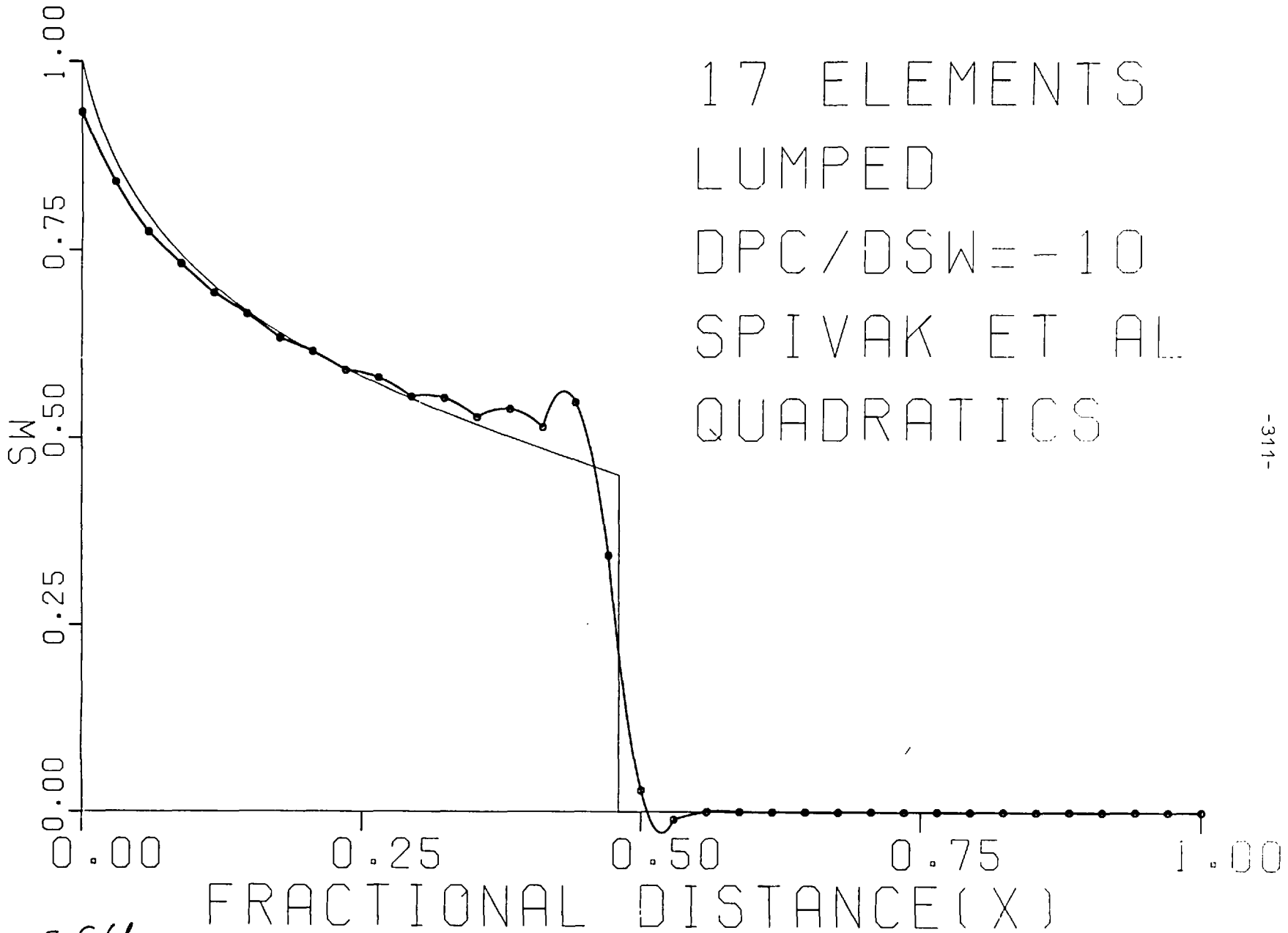

17 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G6b

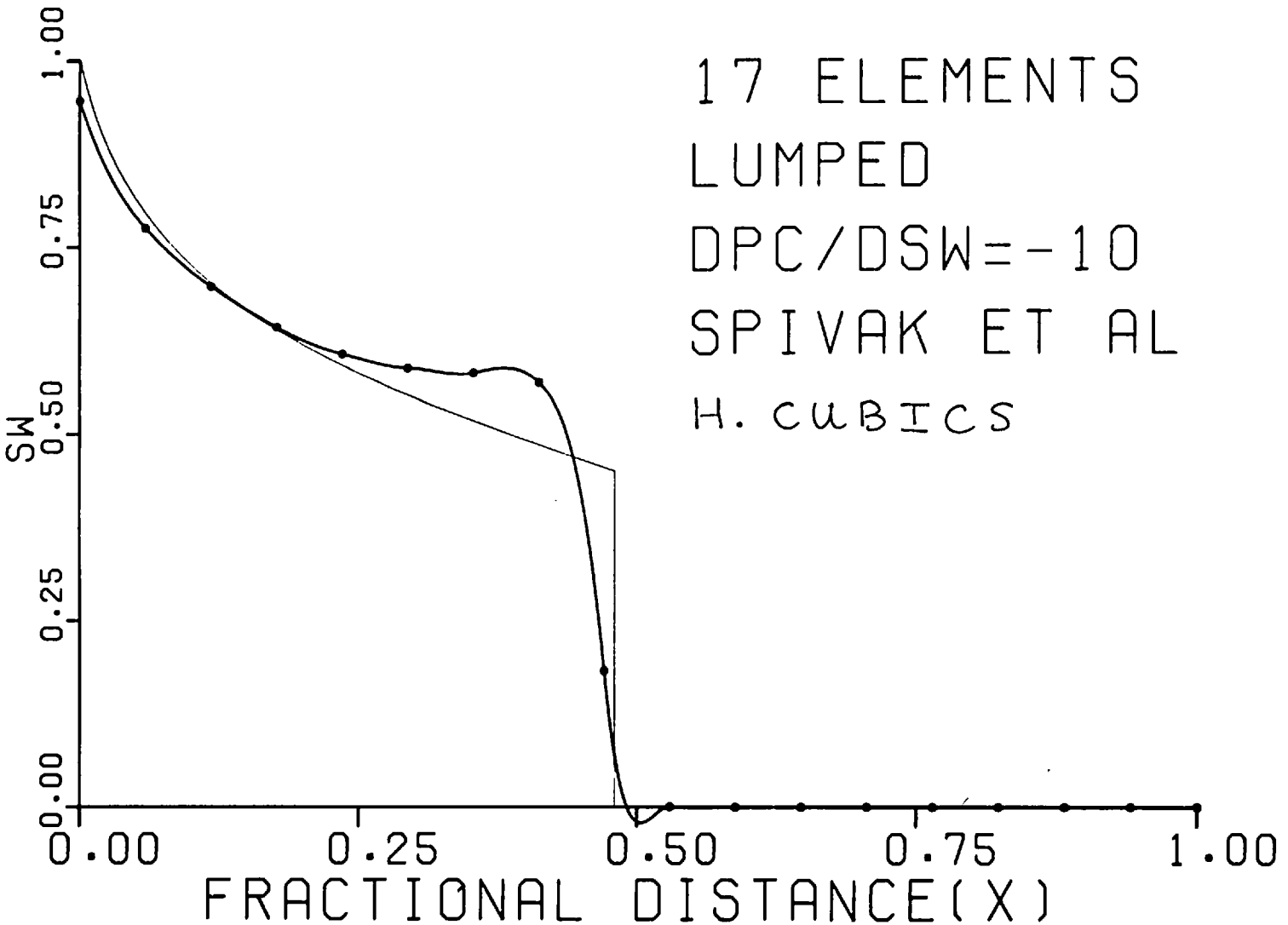

17 ELEMENTS
LUMPED
DPC/DSW=-10
SPIVAK ET AL
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G6c

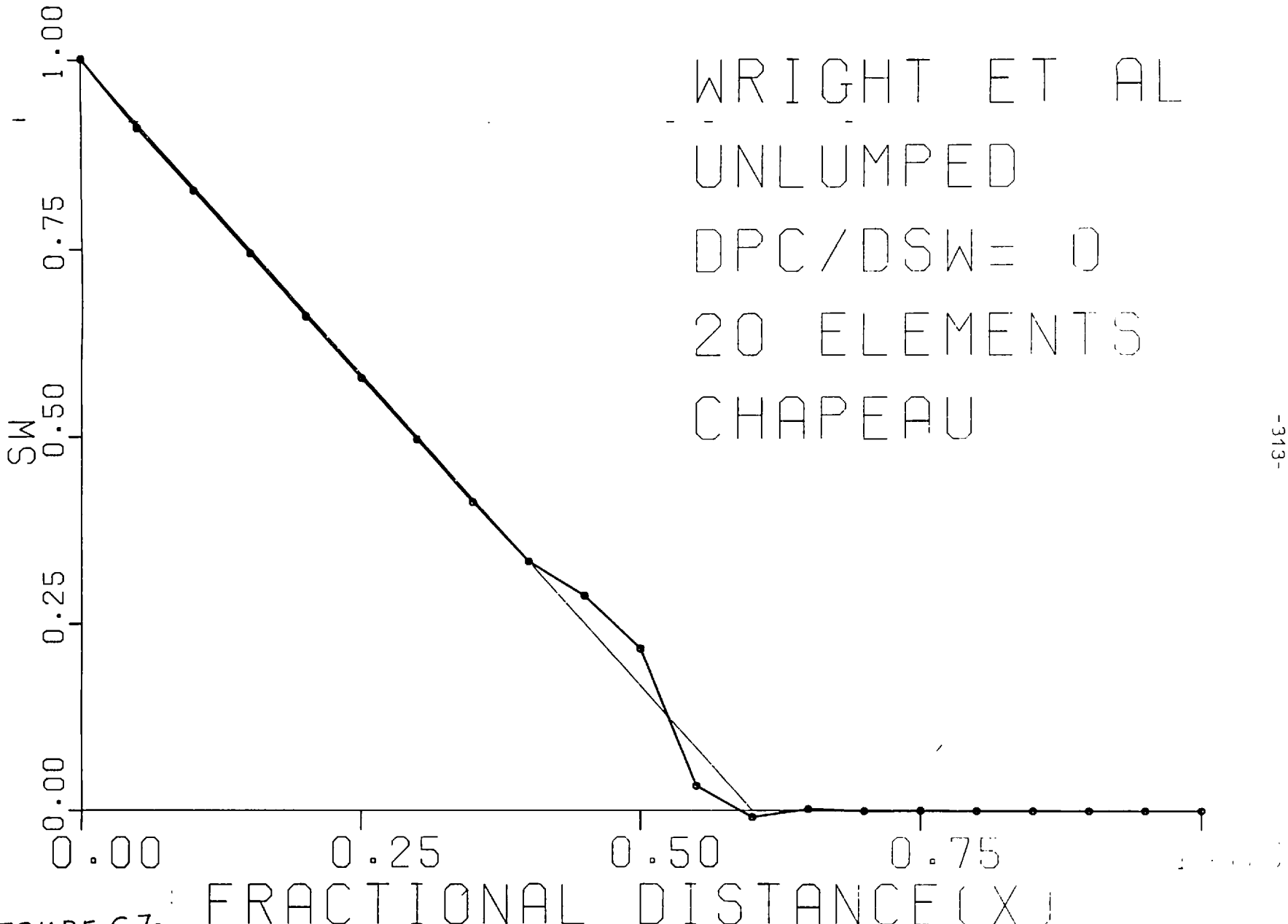

WRIGHT ET AL
UNLUMPED
DPC/DSW= 0
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
FRACTIONAL DISTANCE(X)

FIGURE G7a

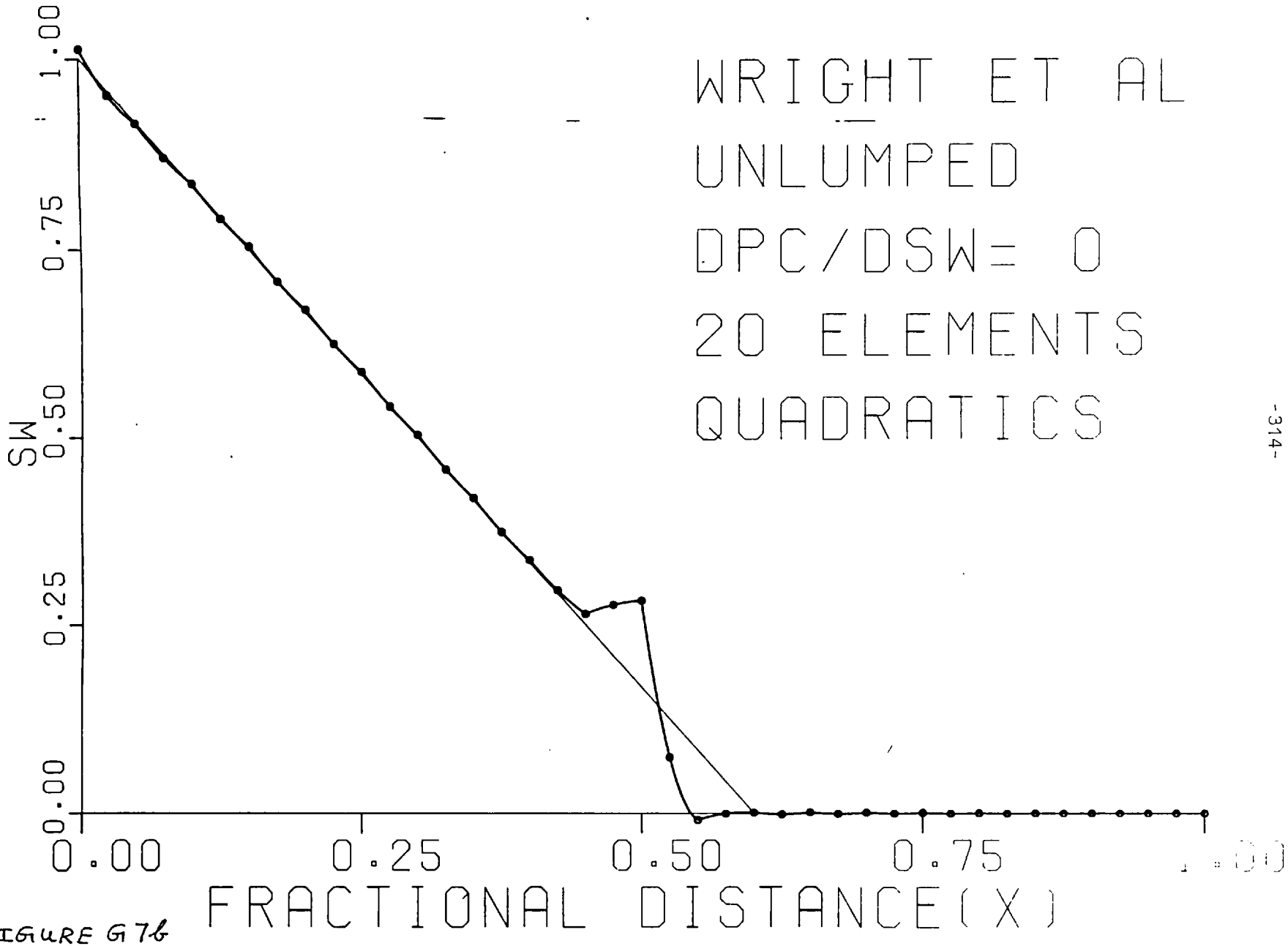

WRIGHT ET AL
UNLUMPED
DPC/DSW= 0
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G 7b

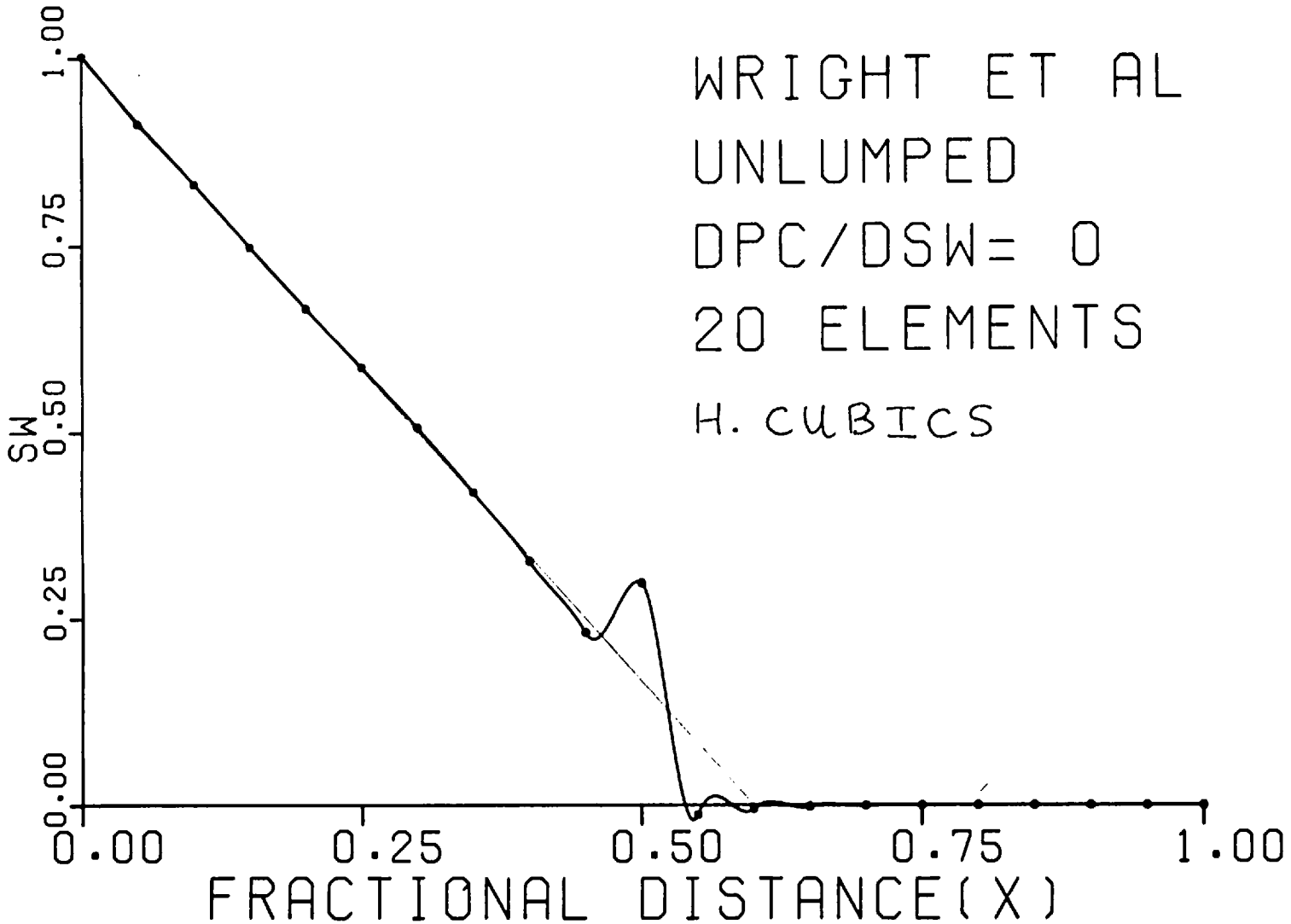

WRIGHT ET AL
UNLUMPED
DPC/DSW= 0
20 ELEMENTS
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G7c

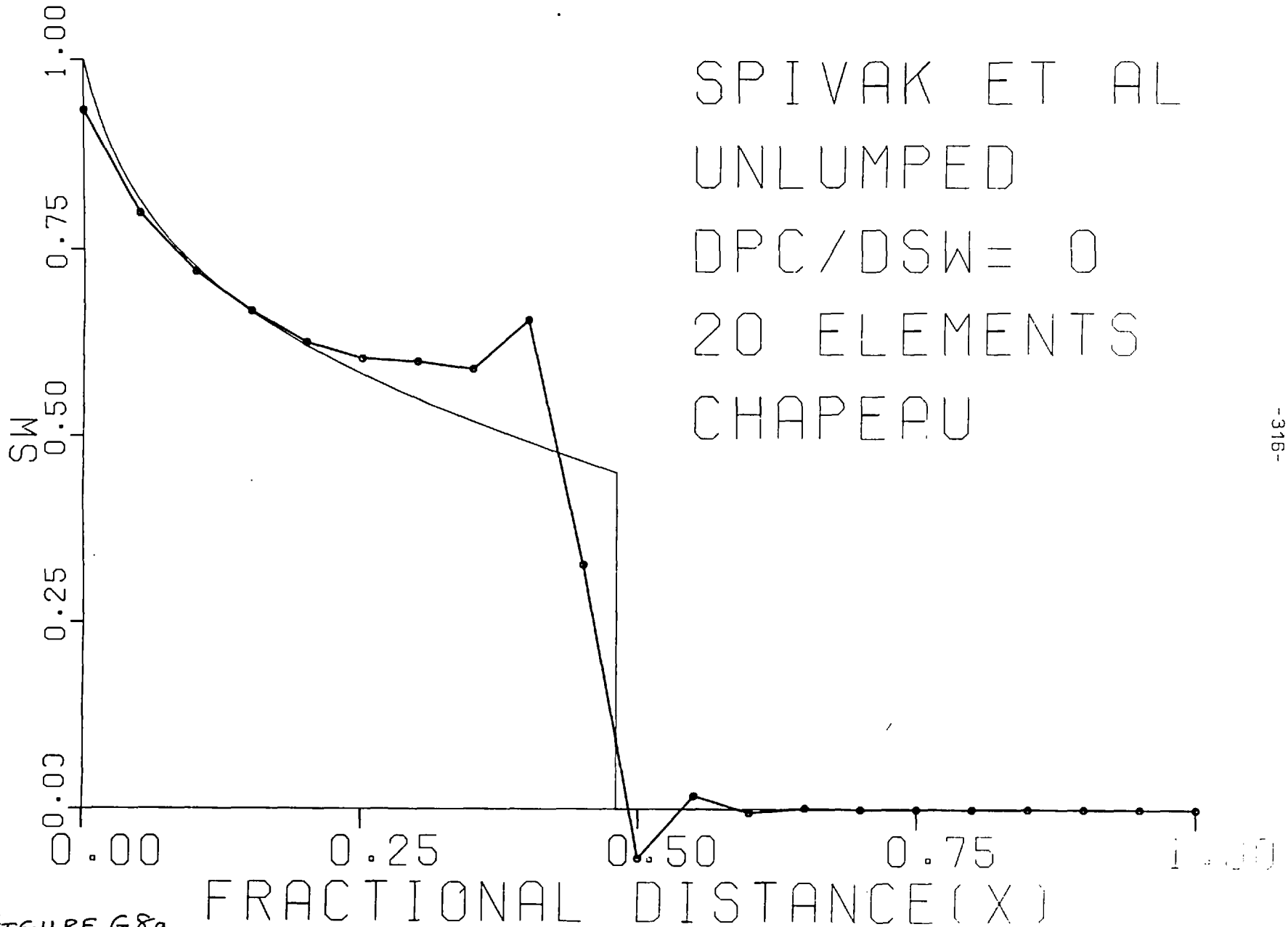

SPIVAK ET AL
UNLUMPED
DPC/DSW= 0
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G8a

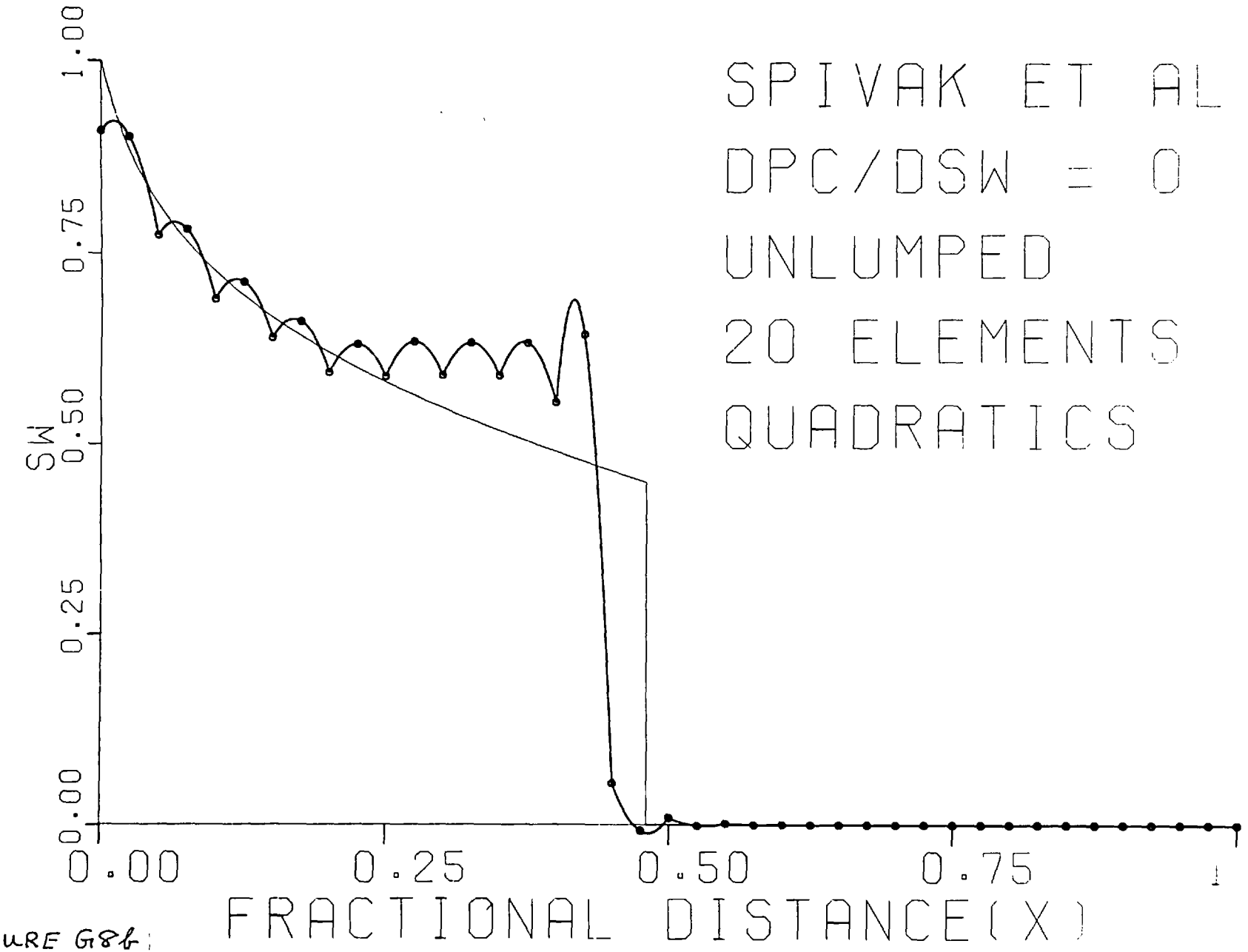

SPIVAK ET AL
DPC/DSW = 0
UNLUMPED
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1
FRACTIONAL DISTANCE(X)

FIGURE G8b

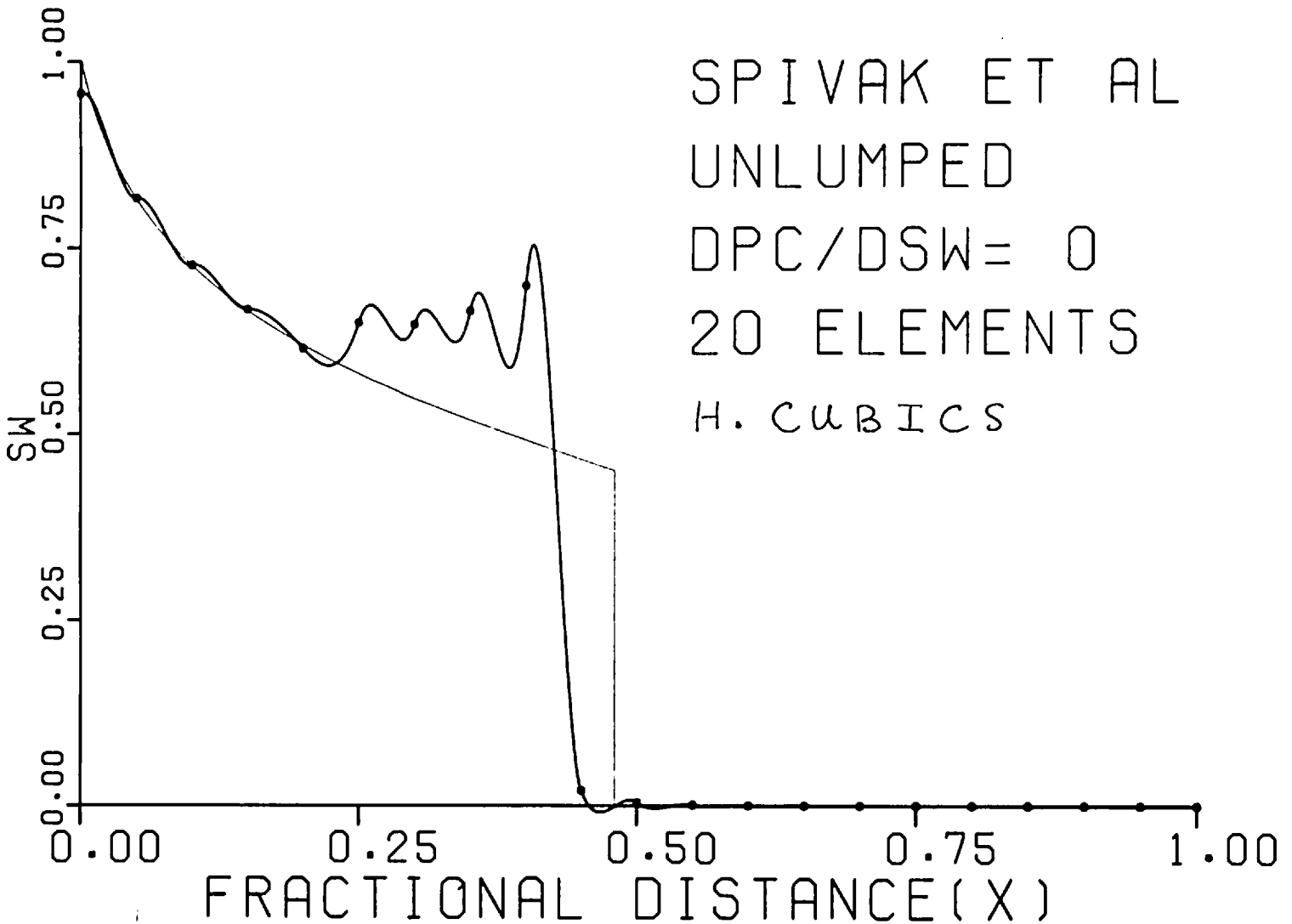

SPIVAK ET AL
UNLUMPED
DPC/DSW= 0
20 ELEMENTS
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G8c

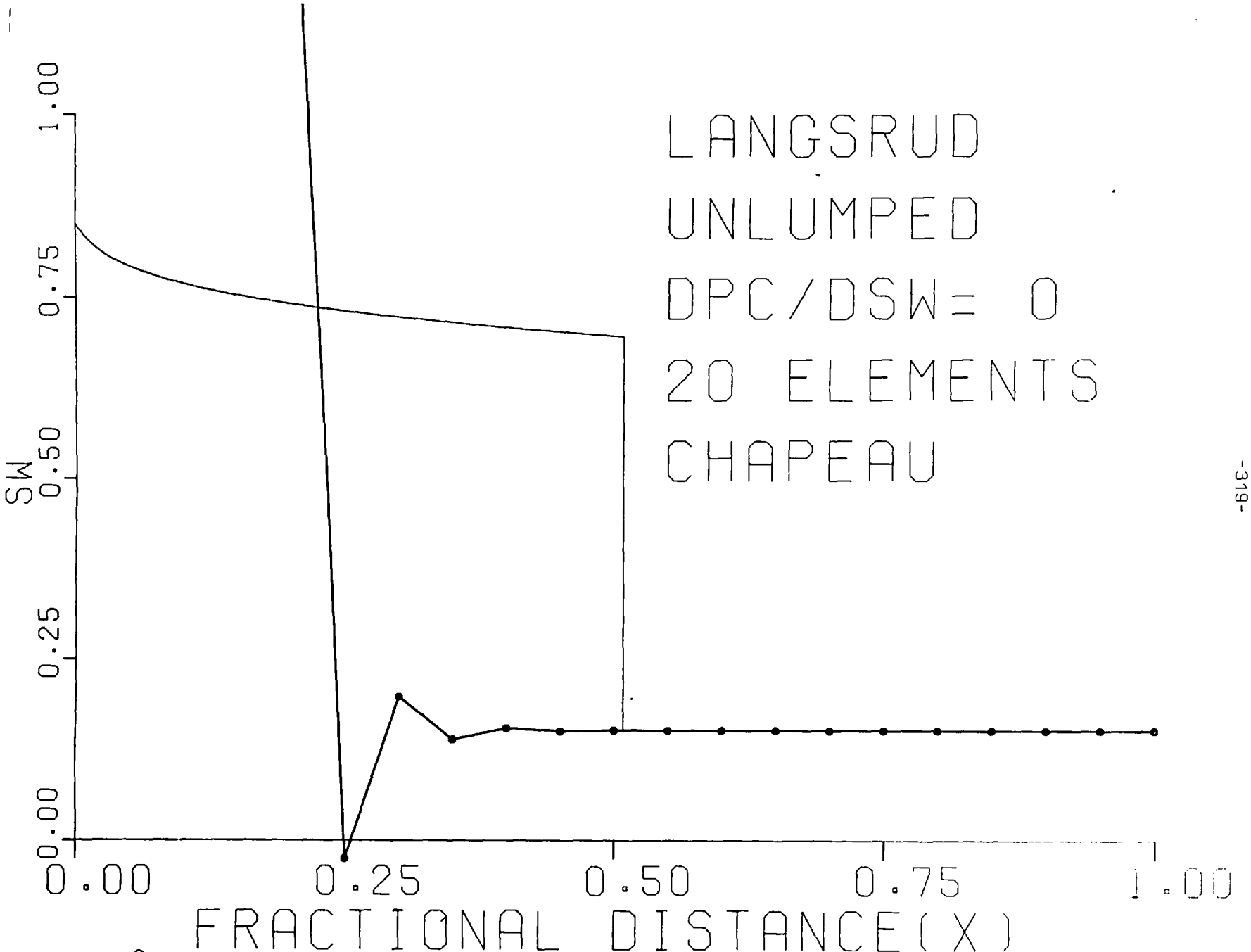

LANGSRUD
UNLUMPED
DPC/DSW= 0
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G9a

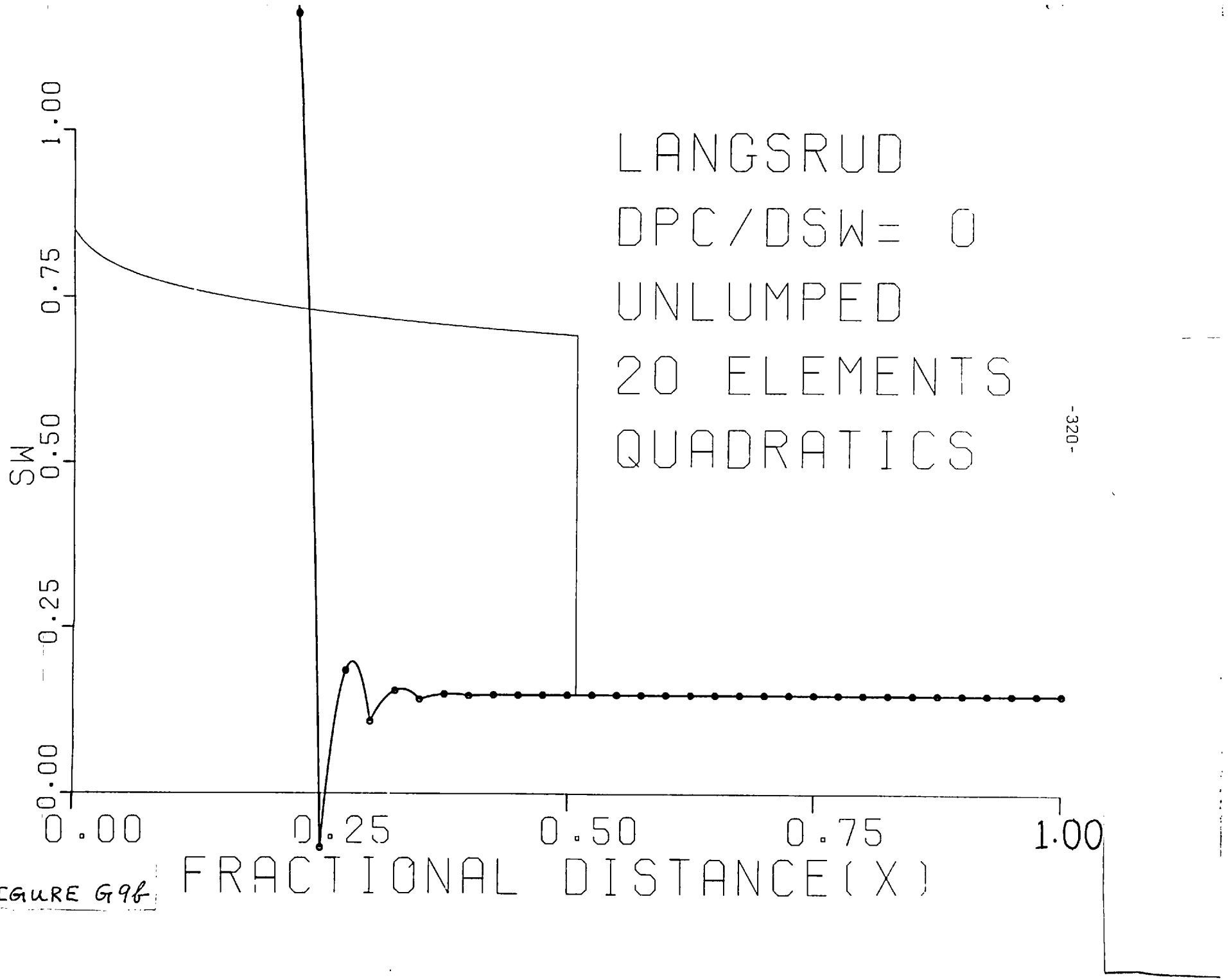

LANGSRUD
DPC/DSW= 0
UNLUMPED
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G9b

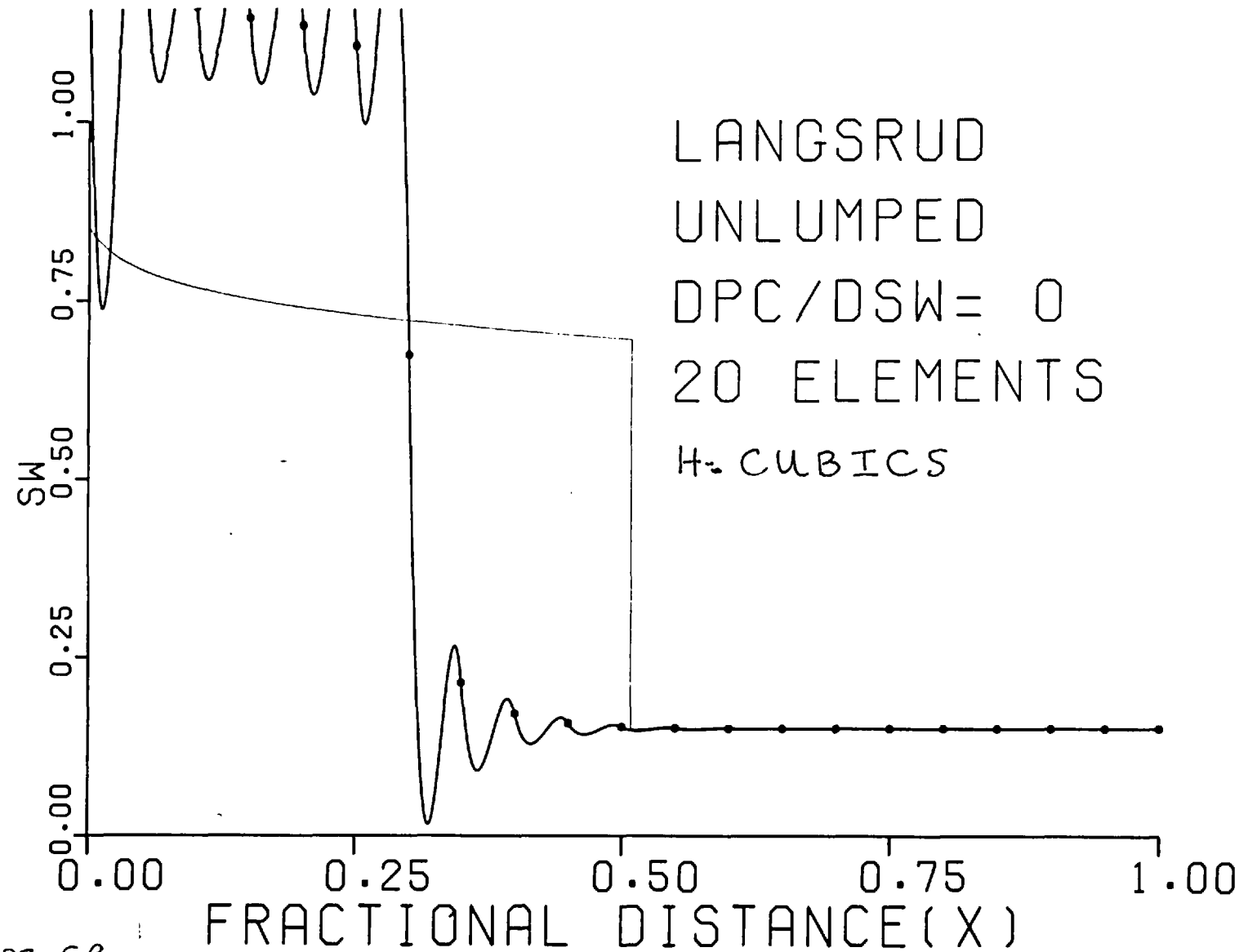

LANGSRUD
UNLUMPED
DPC/DSW= 0
20 ELEMENTS
H- CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G9c

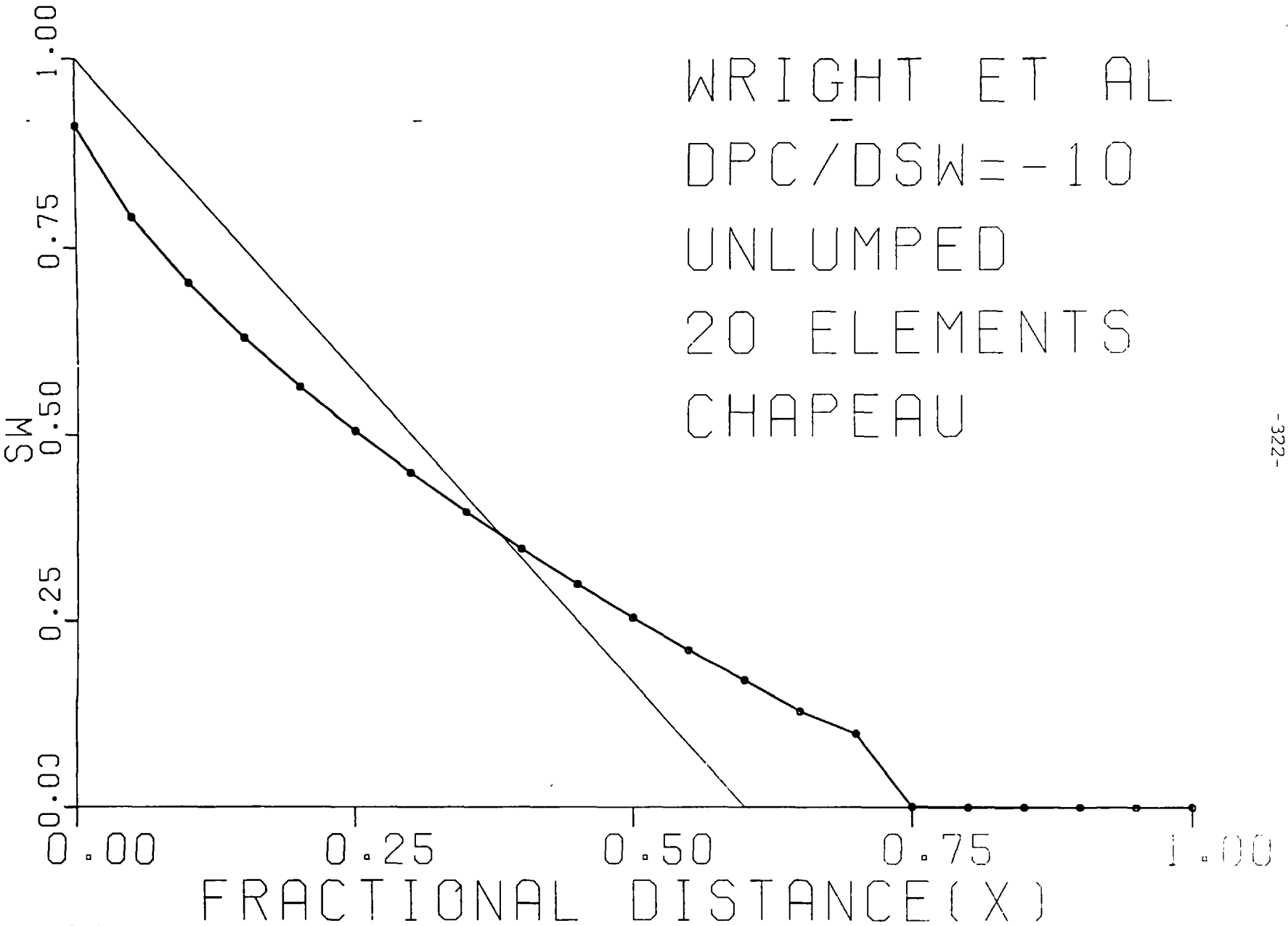

WRIGHT ET AL
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G10a

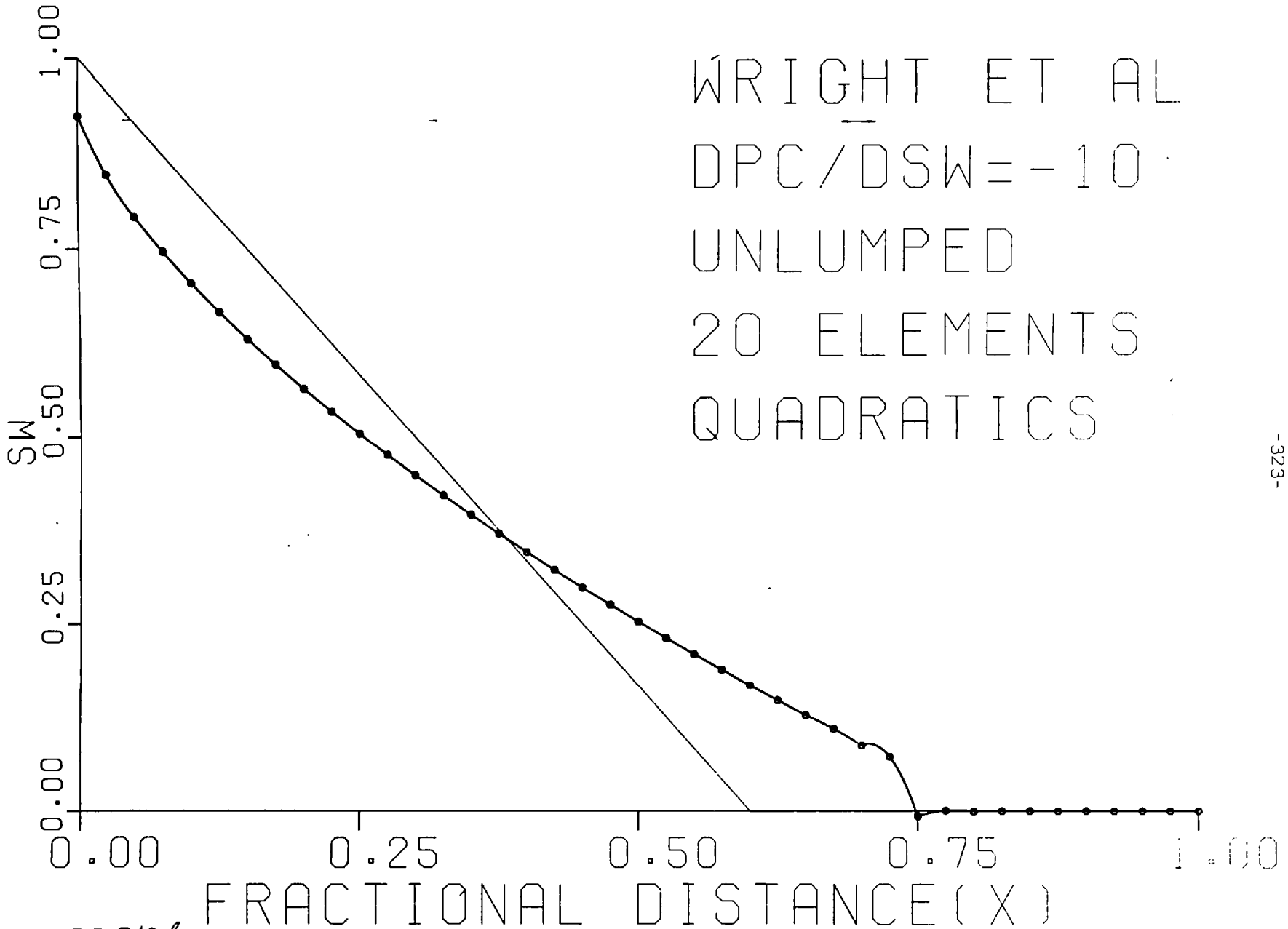

WRIGHT ET AL
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G10b

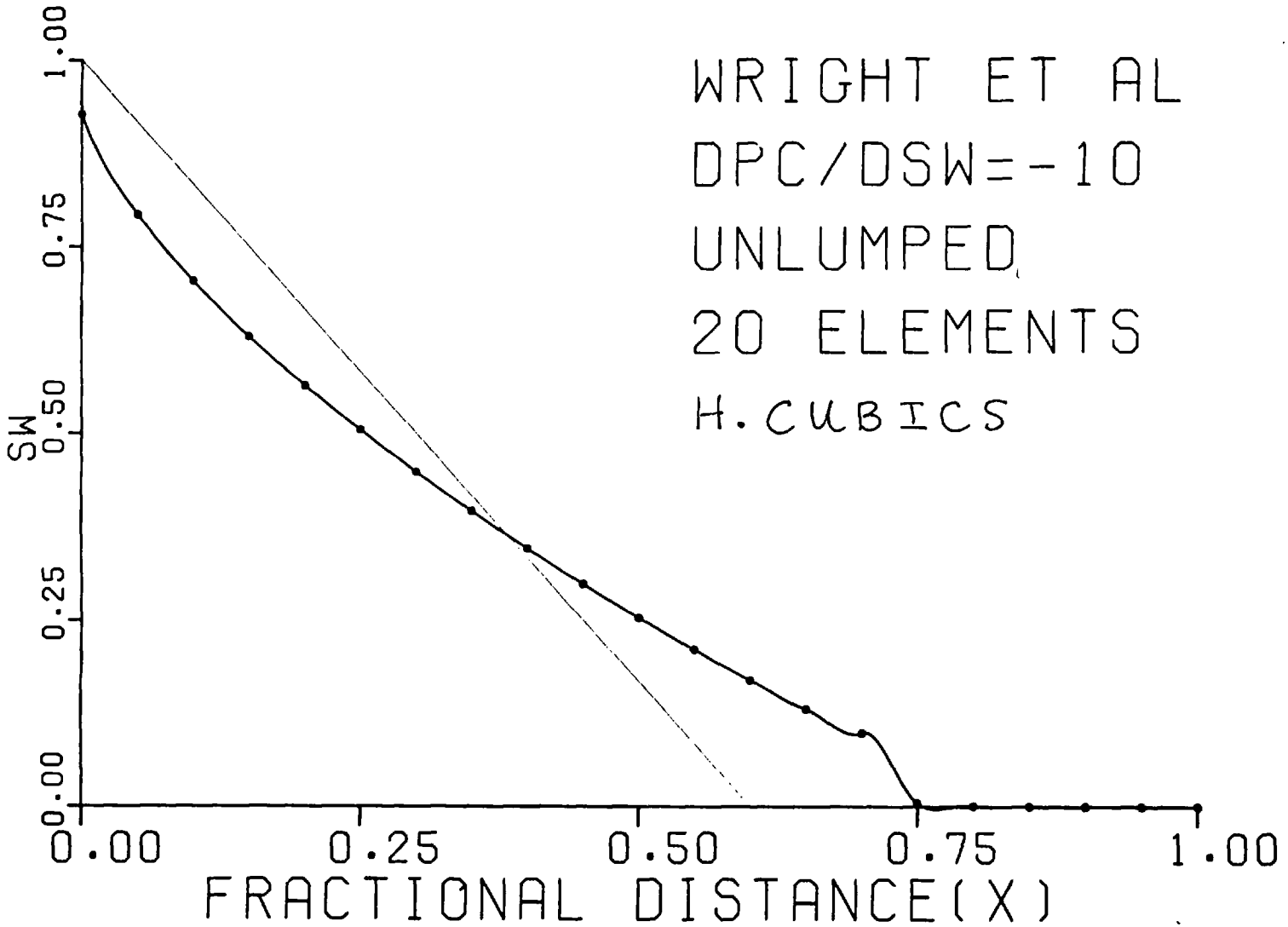

WRIGHT ET AL
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G10c

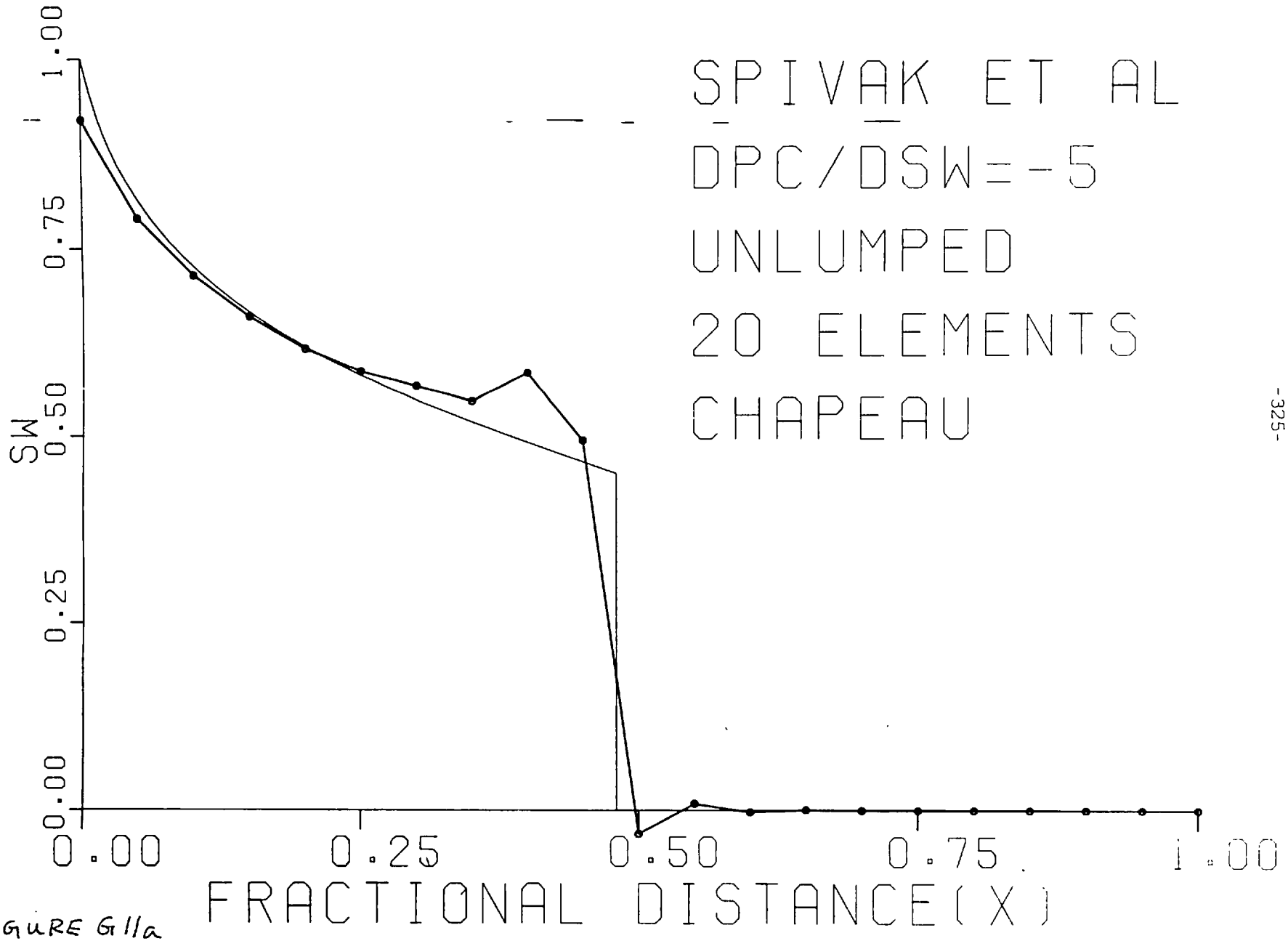

SPIVAK ET AL
DPC/DSW=-5
UNLUMPED
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G11a

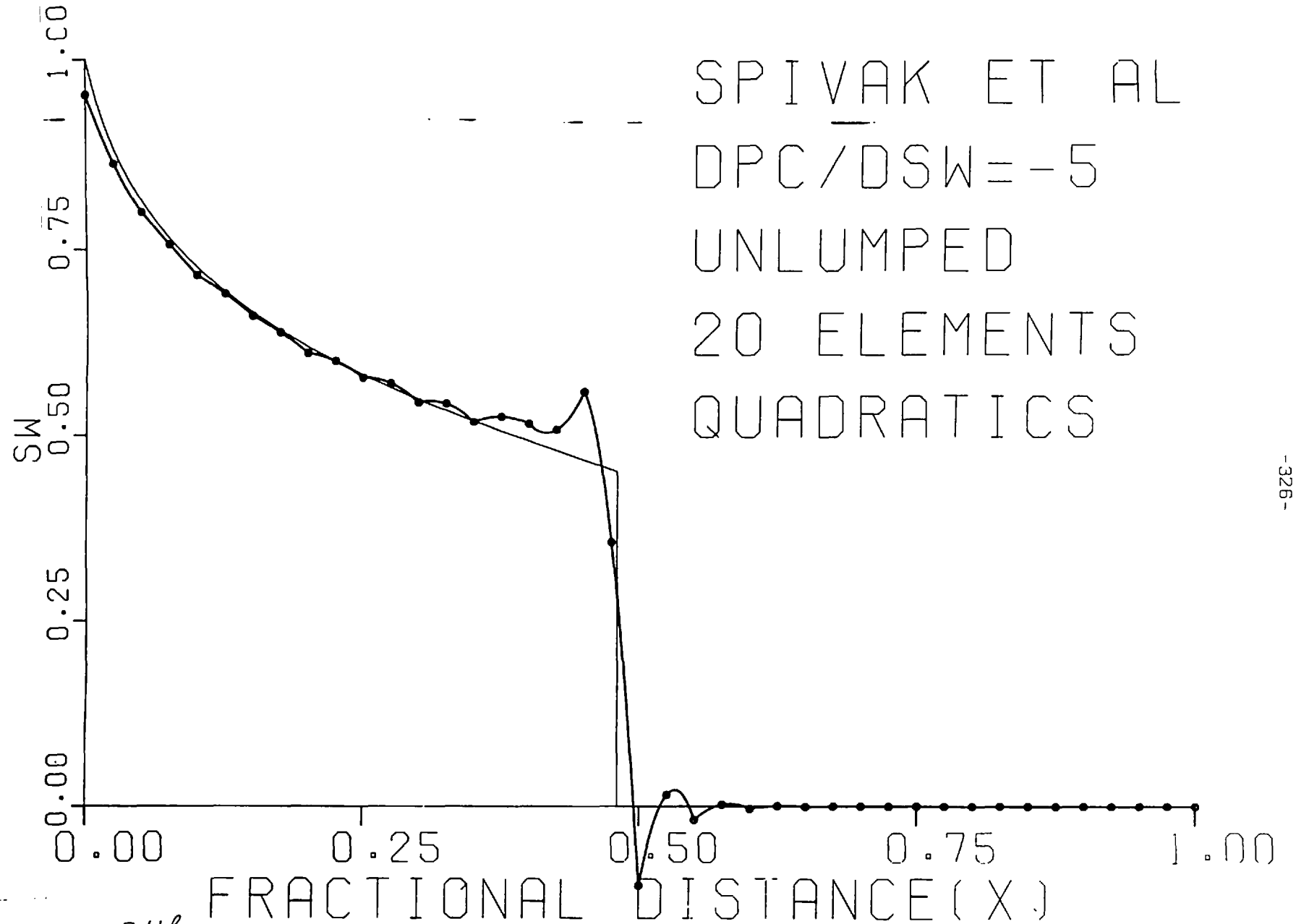

SPIVAK ET AL
DPC/DSW=-5
UNLUMPED
20 ELEMENTS
QUADRATICS
SW
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G11b

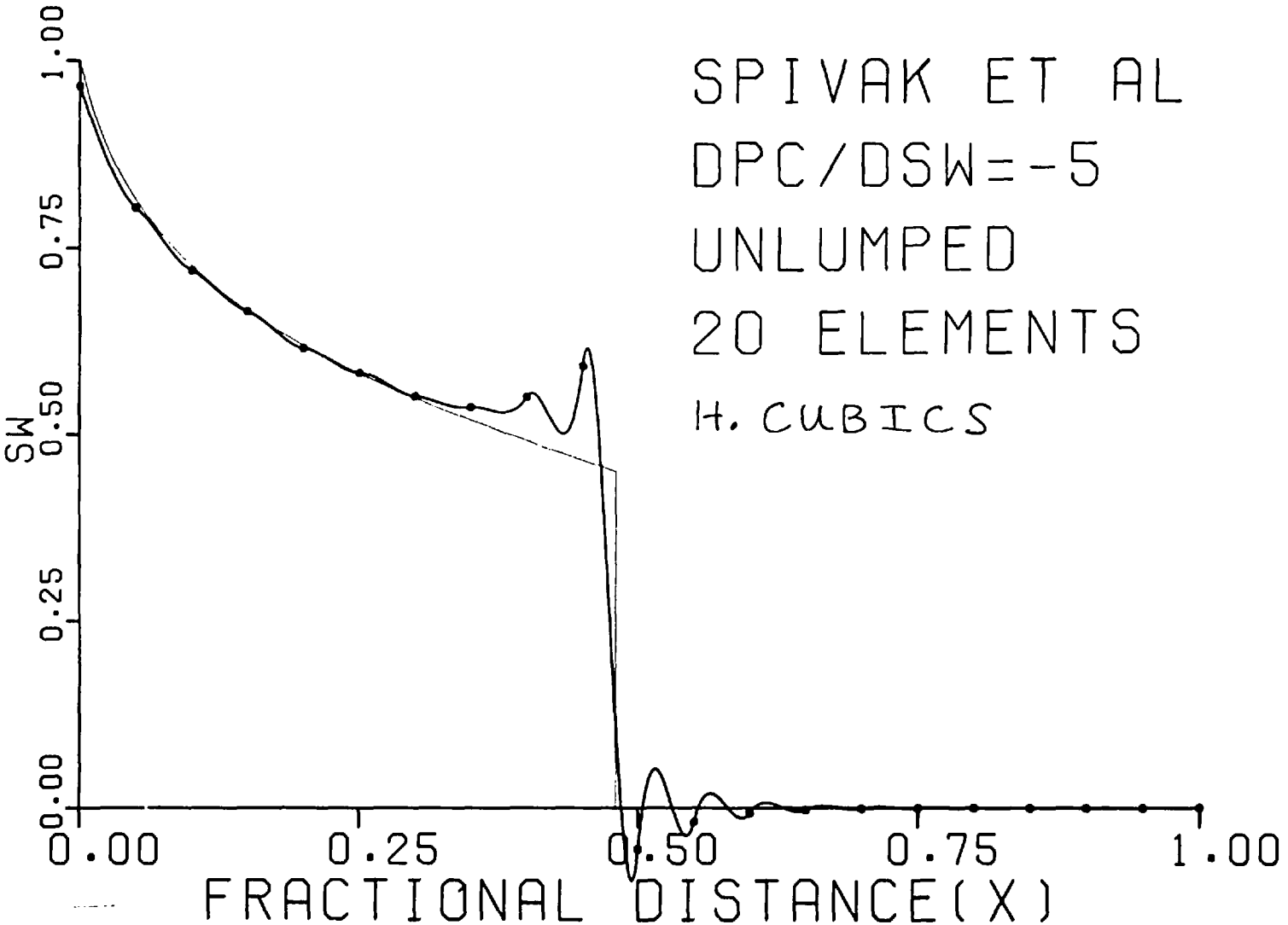

SPIVAK ET AL
DPC/DSW=-5
UNLUMPED
20 ELEMENTS
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G11c

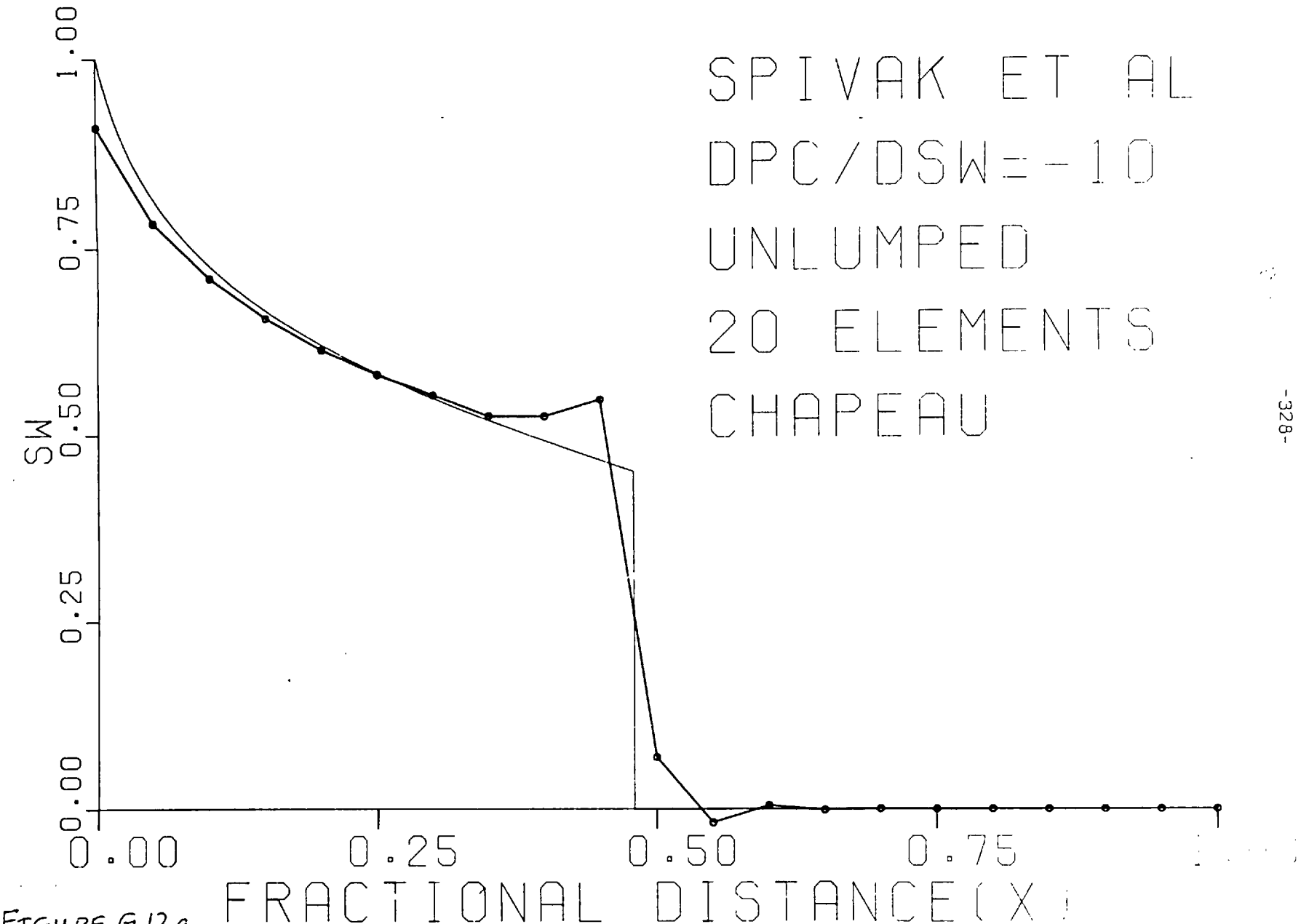

SPIVAK ET AL
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
FRACTIONAL DISTANCE(X)

FIGURE G 12a

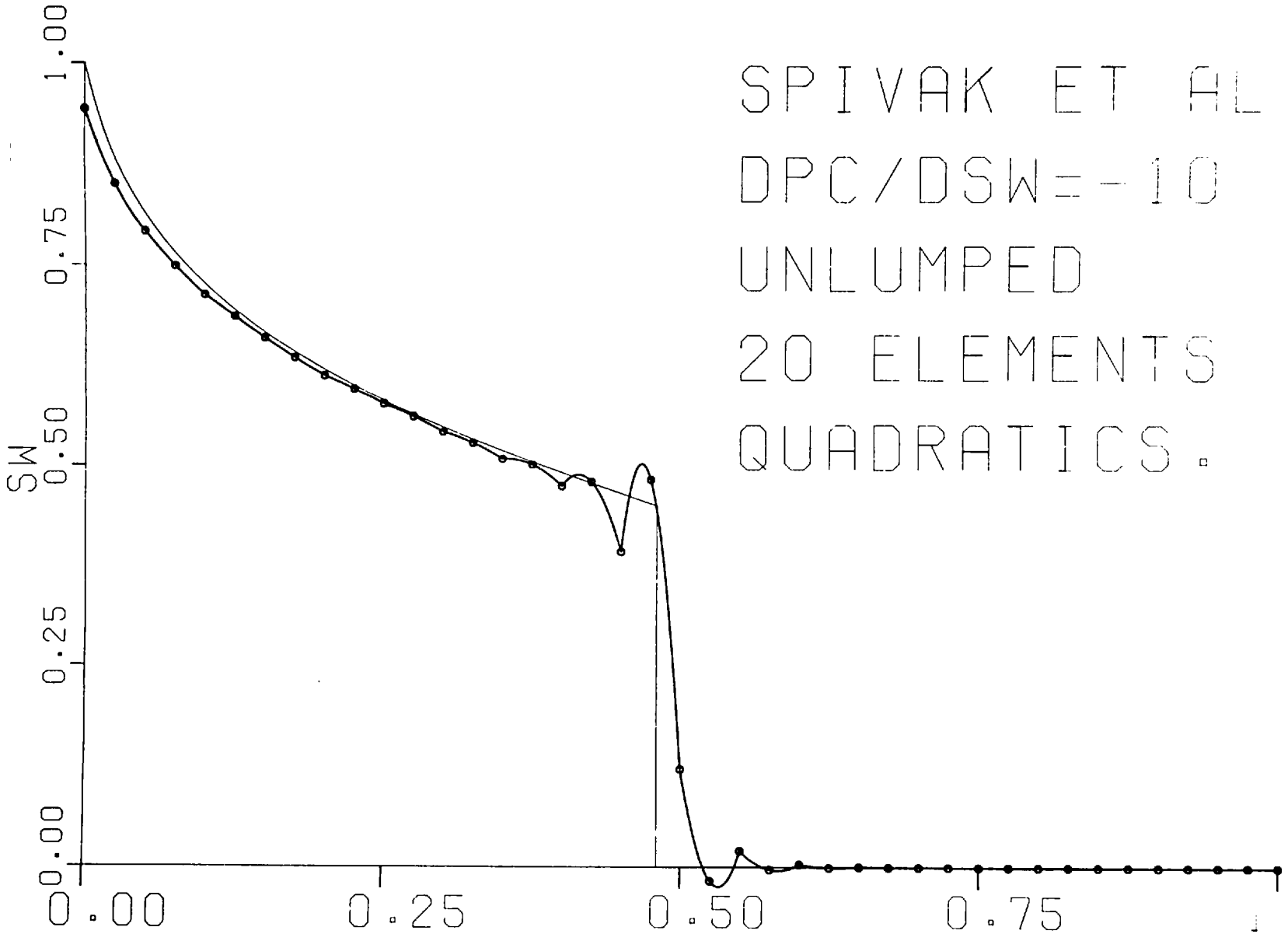

SPIVAK ET AL
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
QUADRATICS.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
FRACTIONAL DISTANCE(X)

FIGURE G126

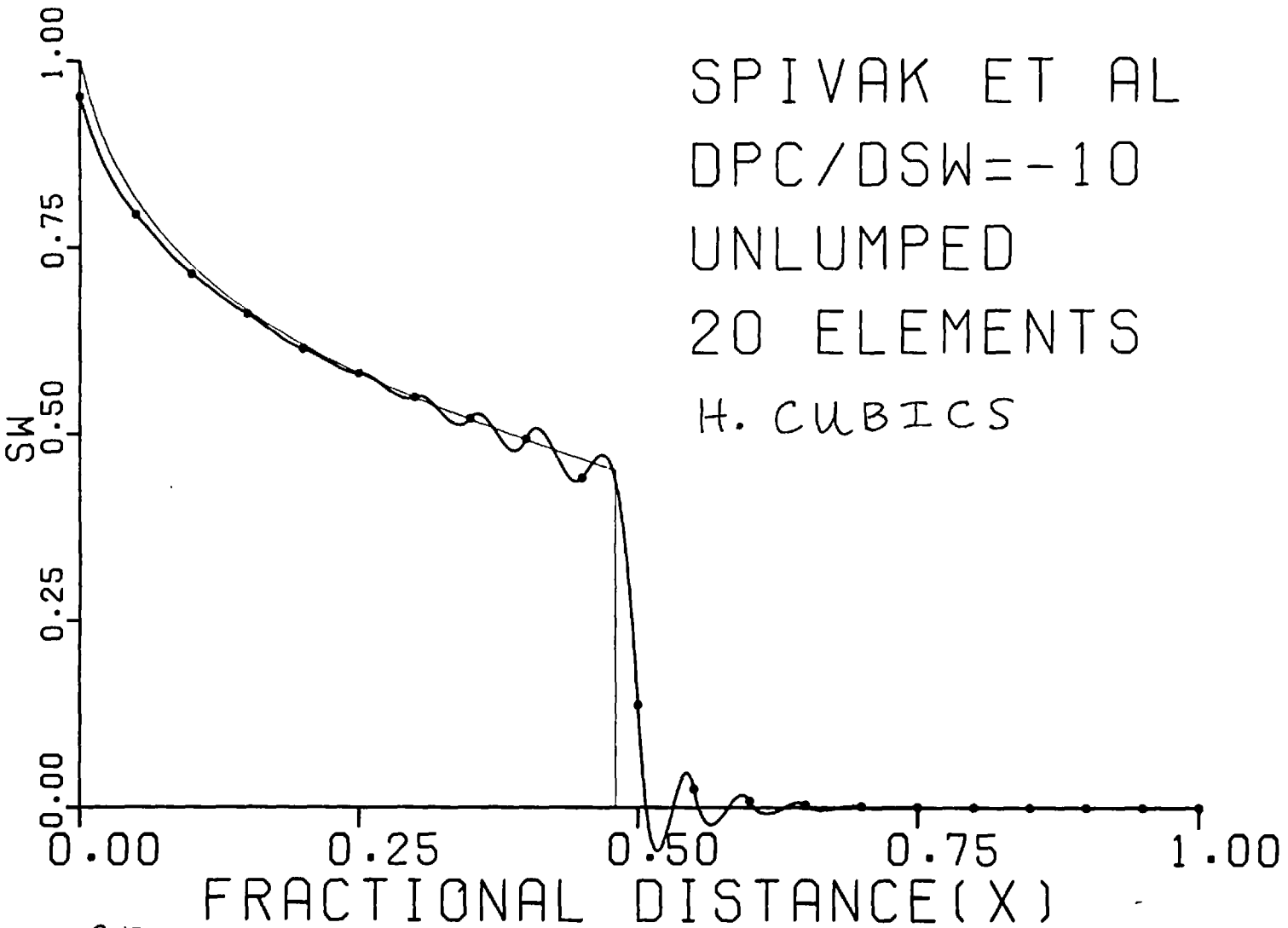

SPIVAK ET AL
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G12c

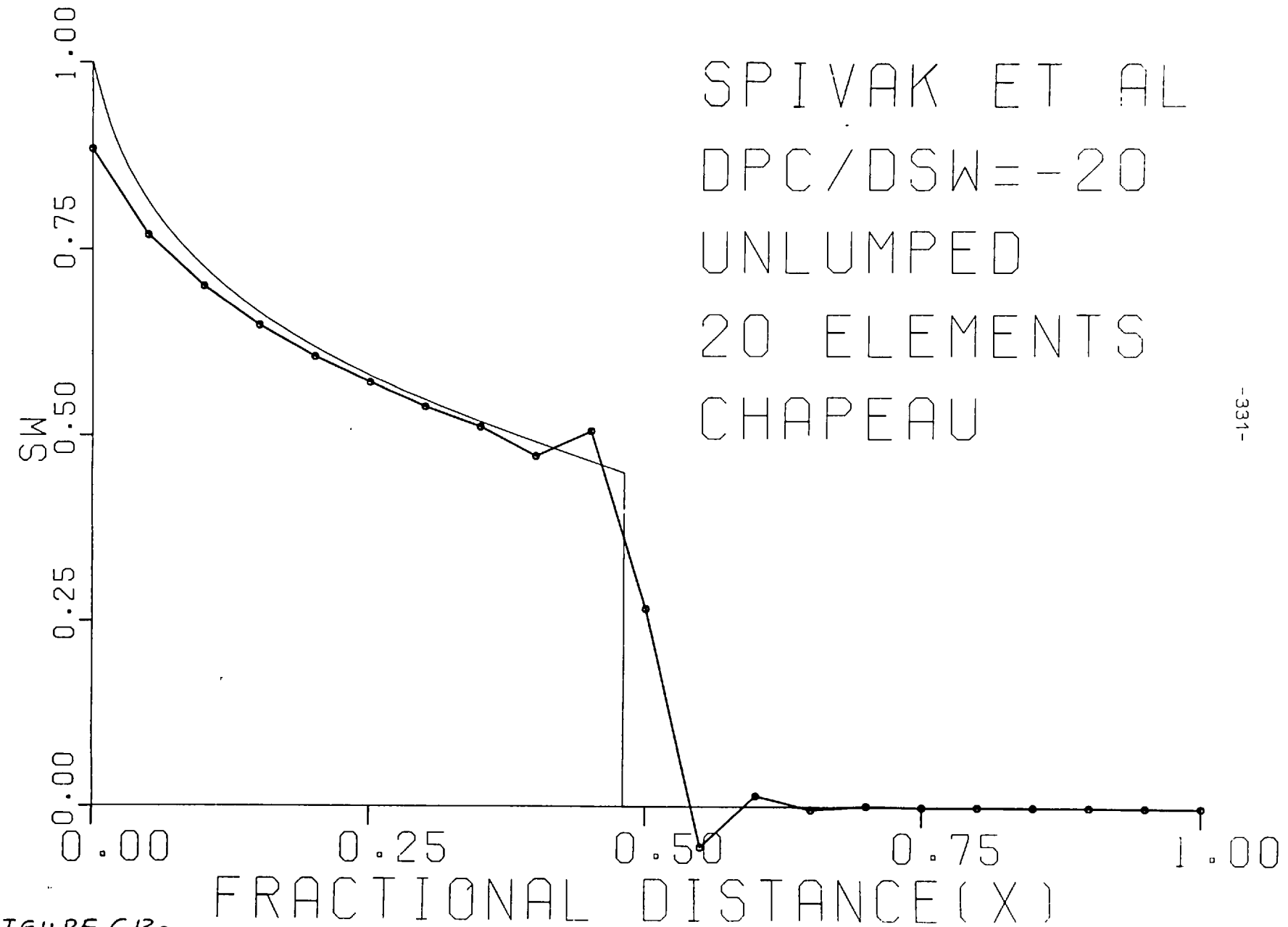

SPIVAK ET AL
DPC/DSW=-20
UNLUMPED
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G13a

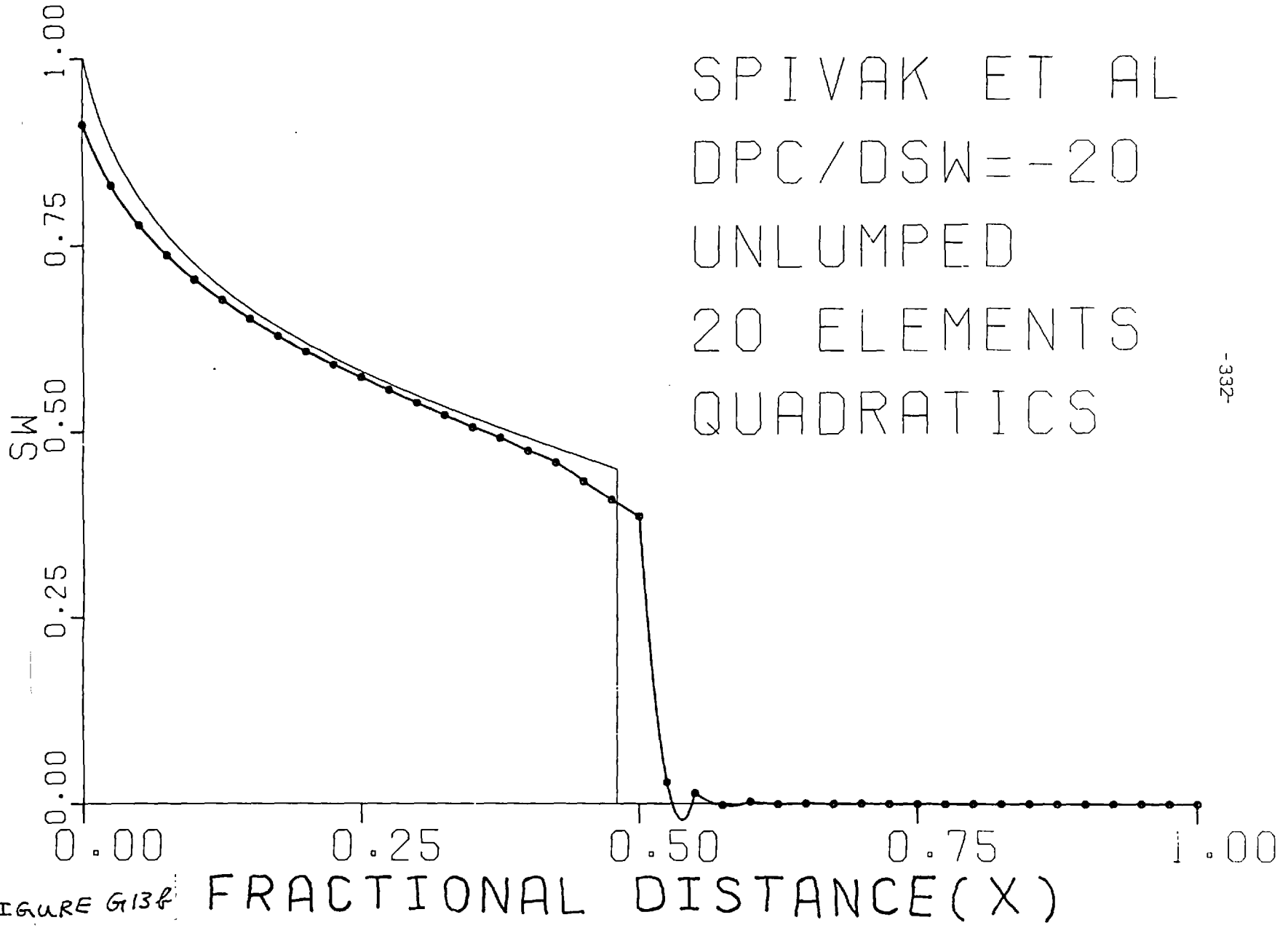

SPIVAK ET AL
DPC/DSW=-20
UNLUMPED
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G13f

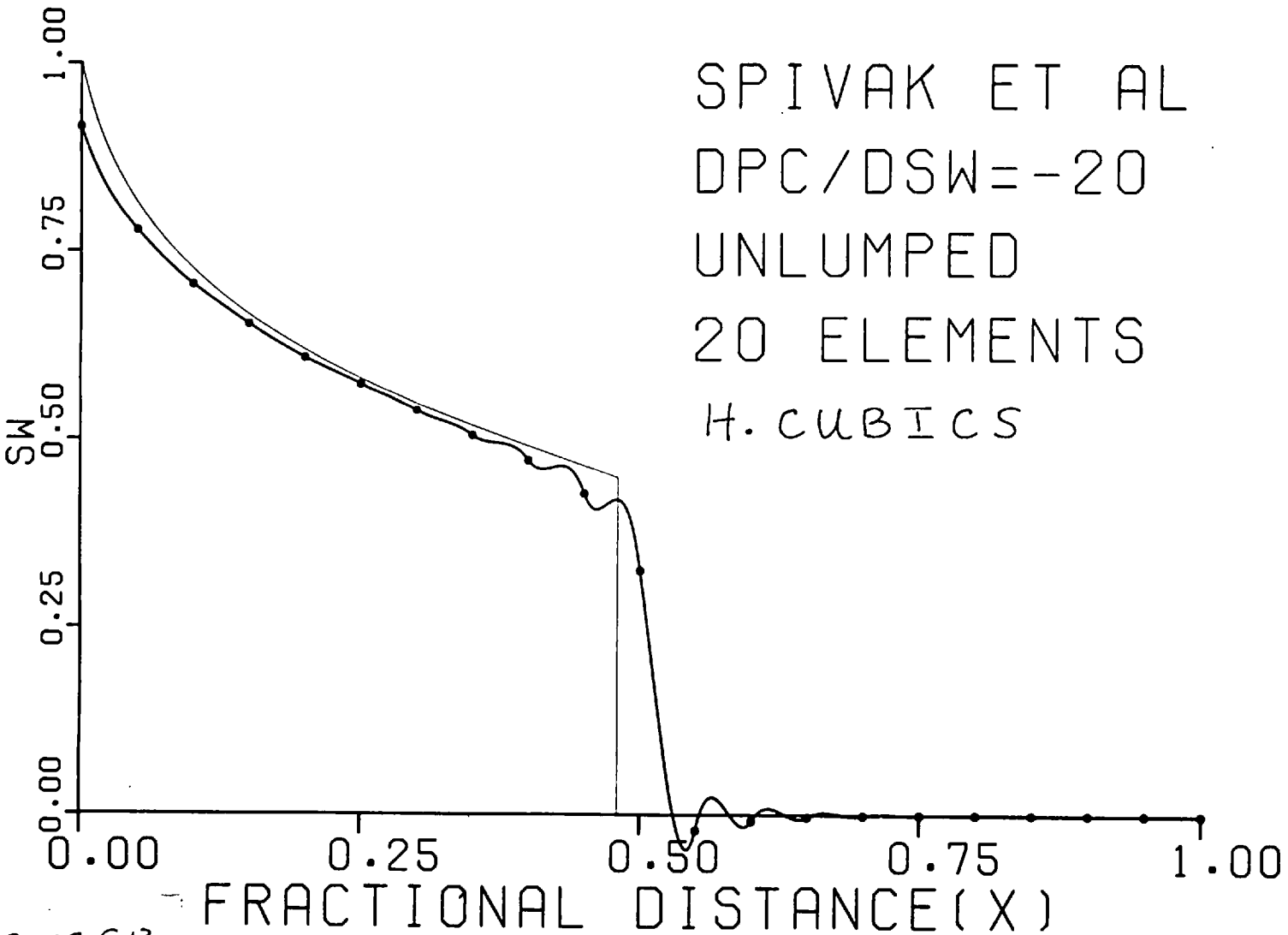

SPIVAK ET AL
DPC/DSW=-20
UNLUMPED
20 ELEMENTS
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G13c

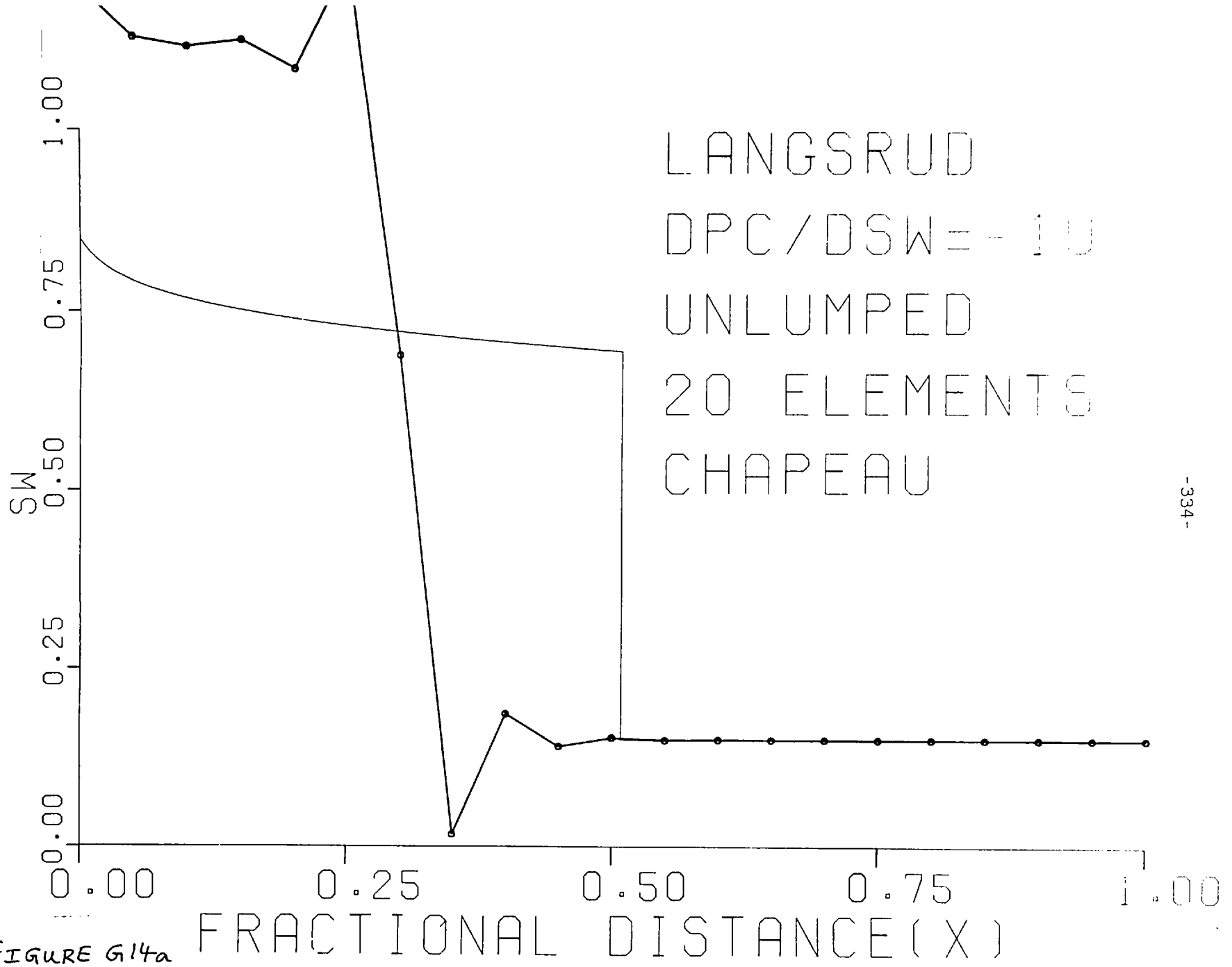

LANGSRUD
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
CHAPEAU
SW
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G14a

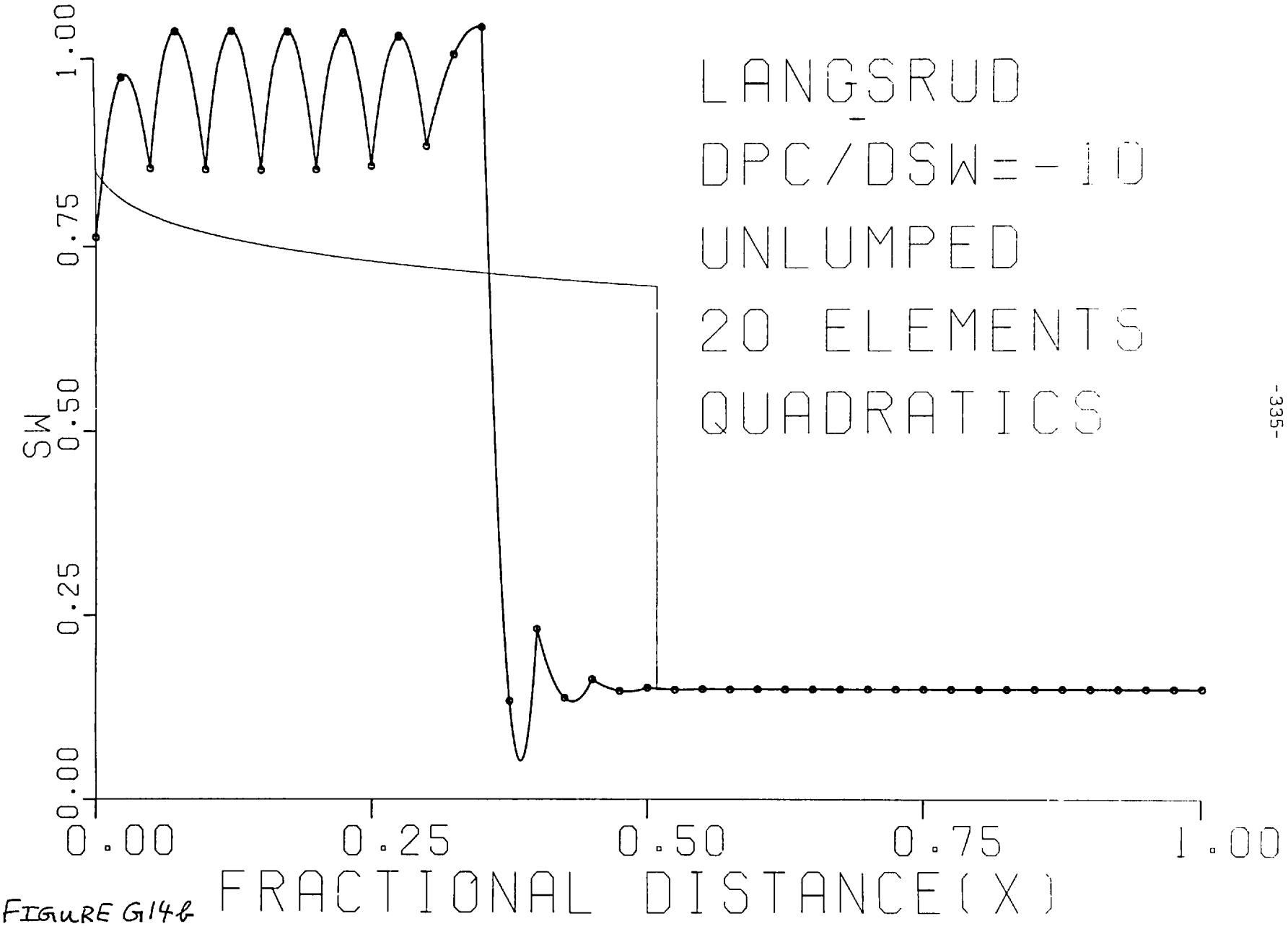

LANGSRUD
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G14b

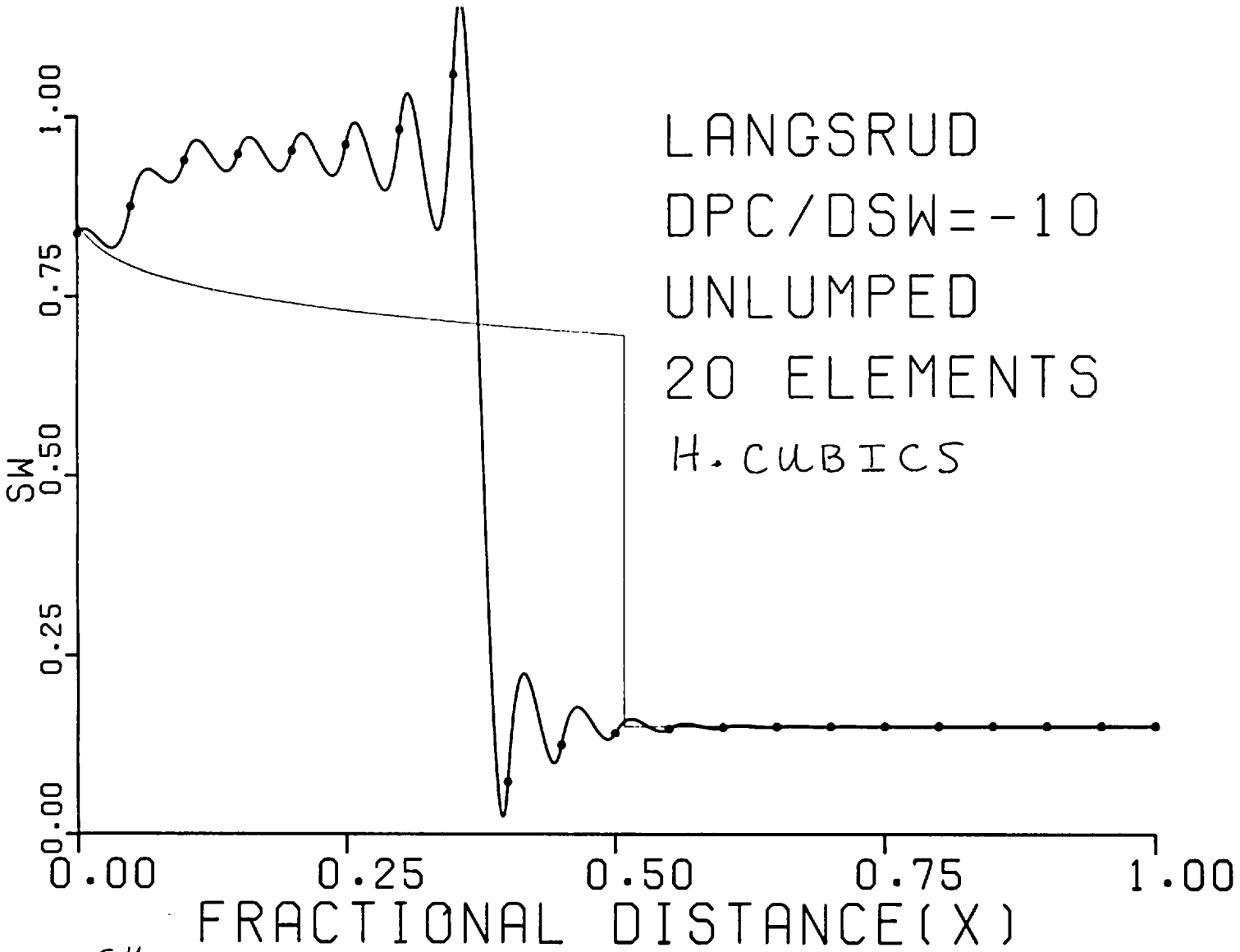

LANGSRUD
DPC/DSW=-10
UNLUMPED
20 ELEMENTS
H. CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G14C

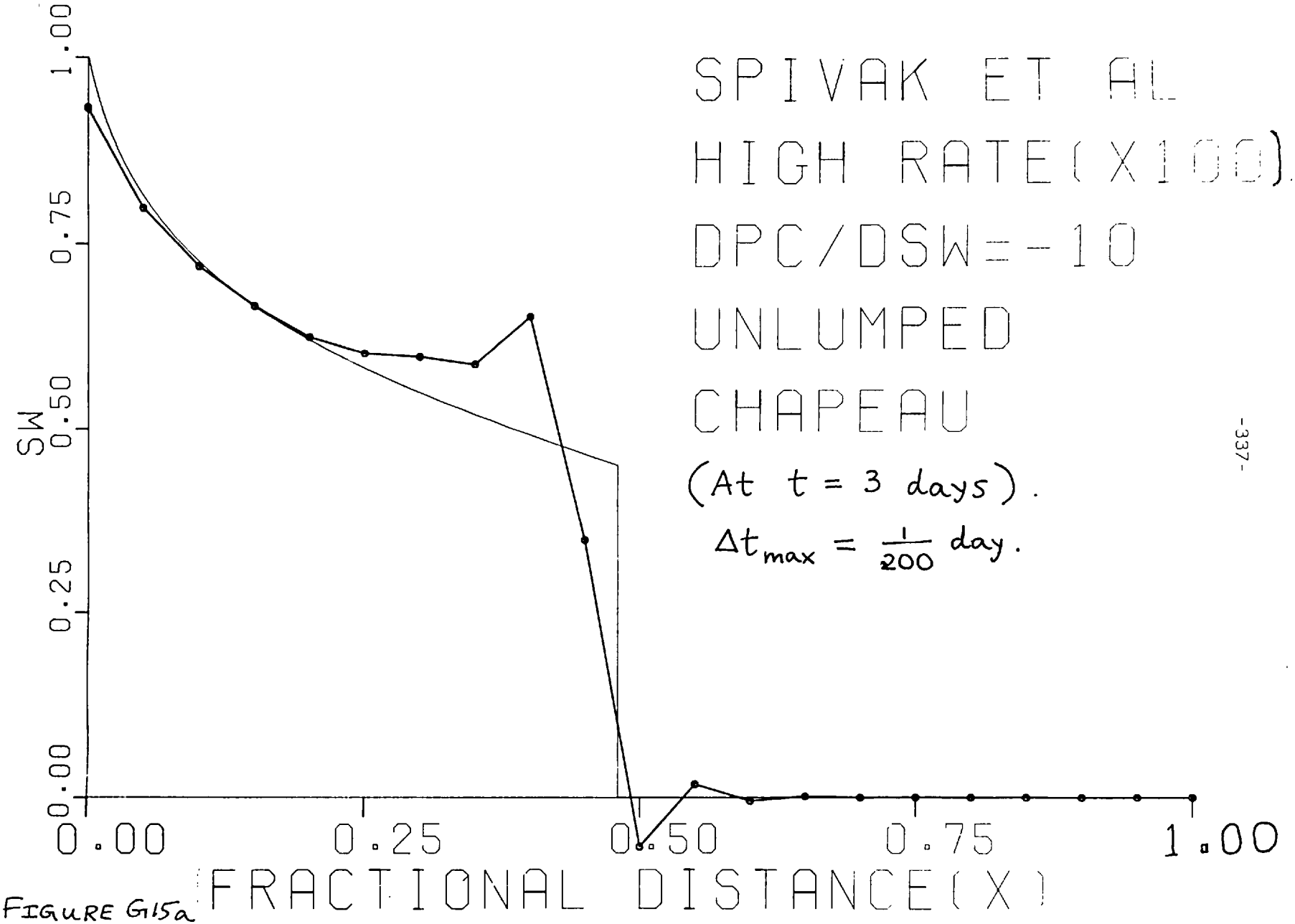

SPIVAK ET AL
HIGH RATE (X100).
DPC/DSW = -10
UNLUMPED
CHAPEAU
(At t = 3 days).
$\Delta t_{max} = \frac{1}{200}$ day.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE (X)

FIGURE G15a

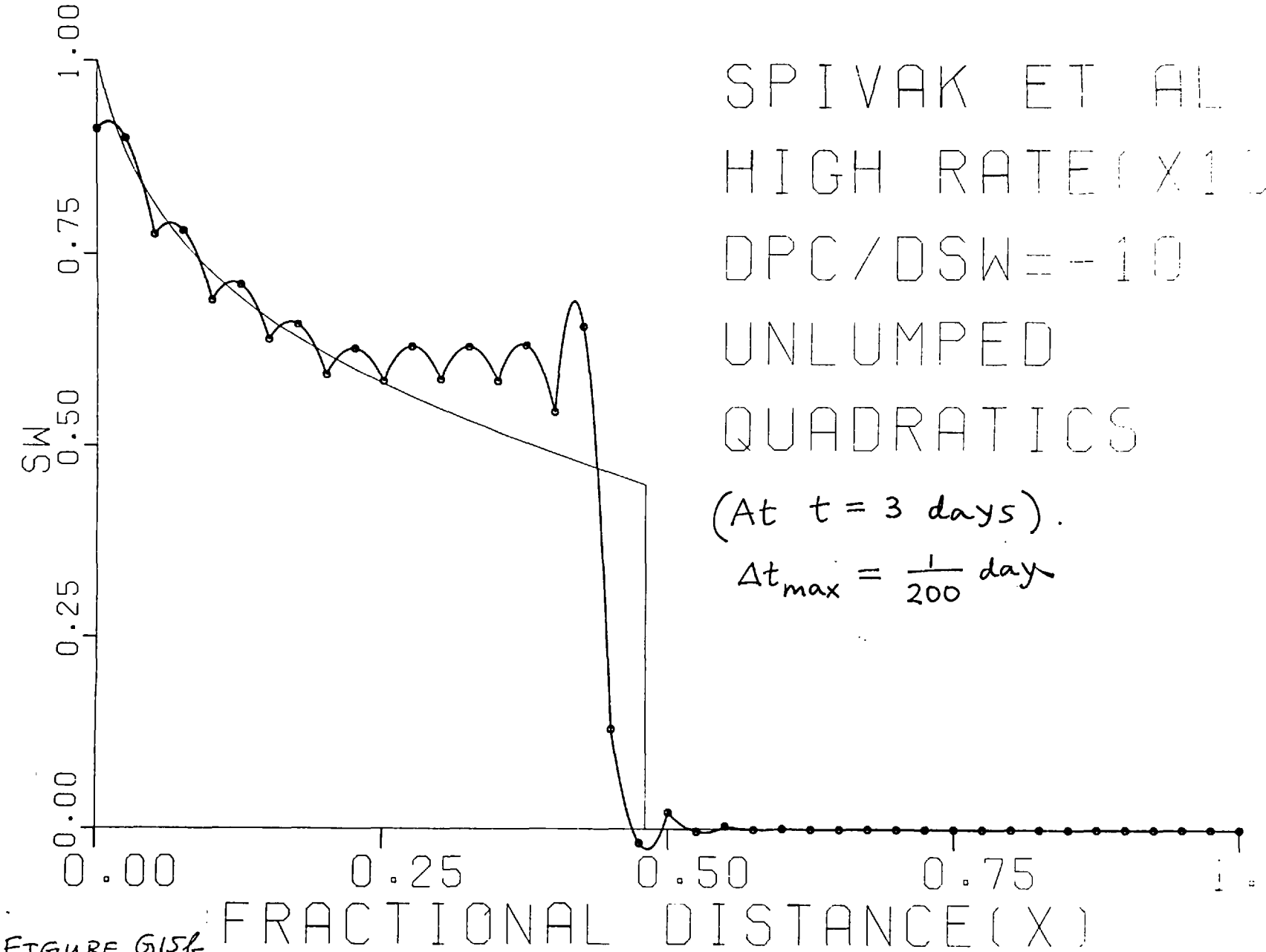

SPIVAK ET AL
HIGH RATE(X100)
DPC/DSW=-10
UNLUMPED
QUADRATICS
(At t = 3 days).
$\Delta t_{max} = \frac{1}{200}$ day
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.
FRACTIONAL DISTANCE(X)

FIGURE G15b

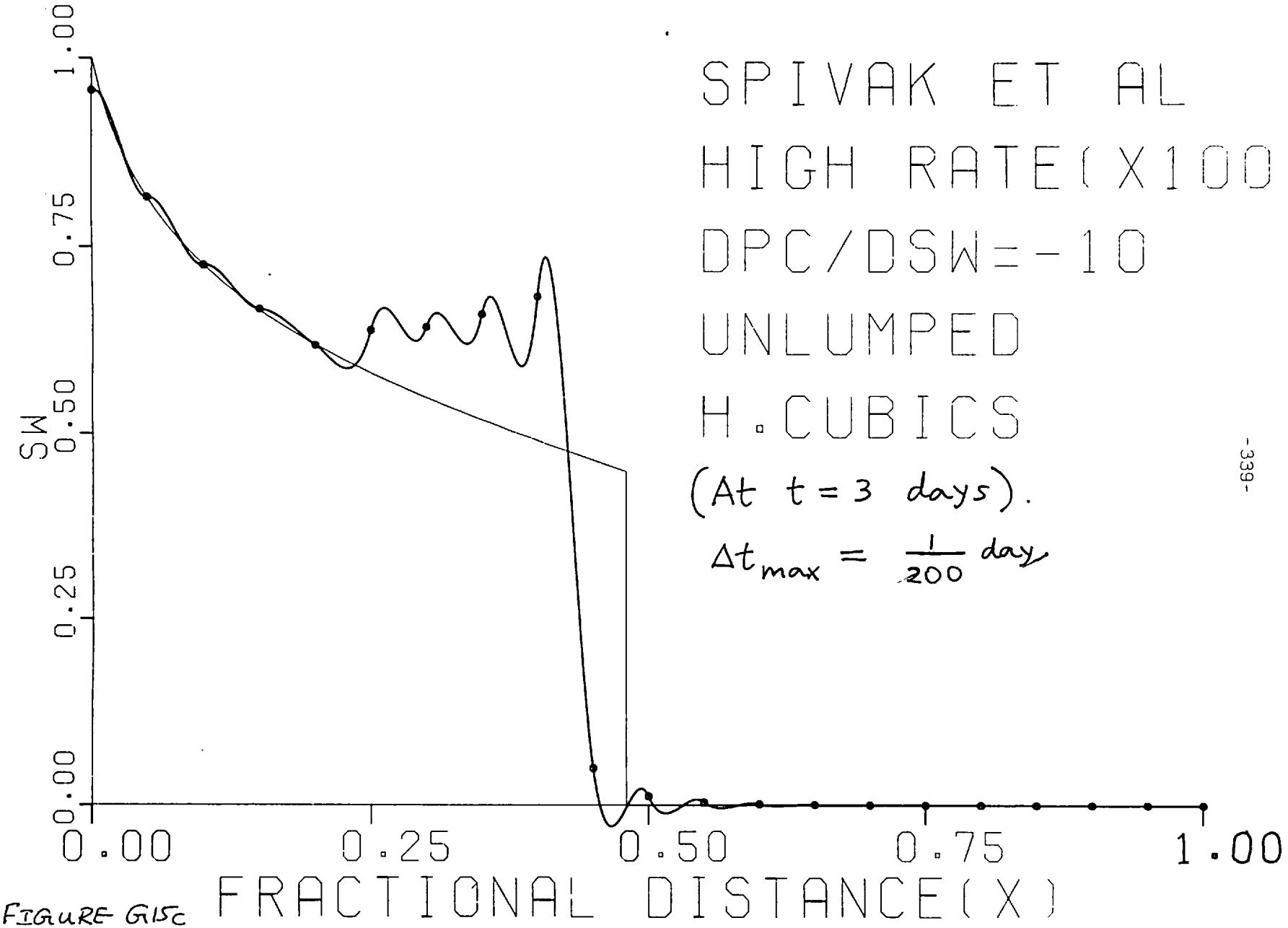

SPIVAK ET AL
HIGH RATE(X100)
DPC/DSW=-10
UNLUMPED
H.CUBICS
(At t = 3 days).
$\Delta t_{max} = \frac{1}{200}$ day
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G15c

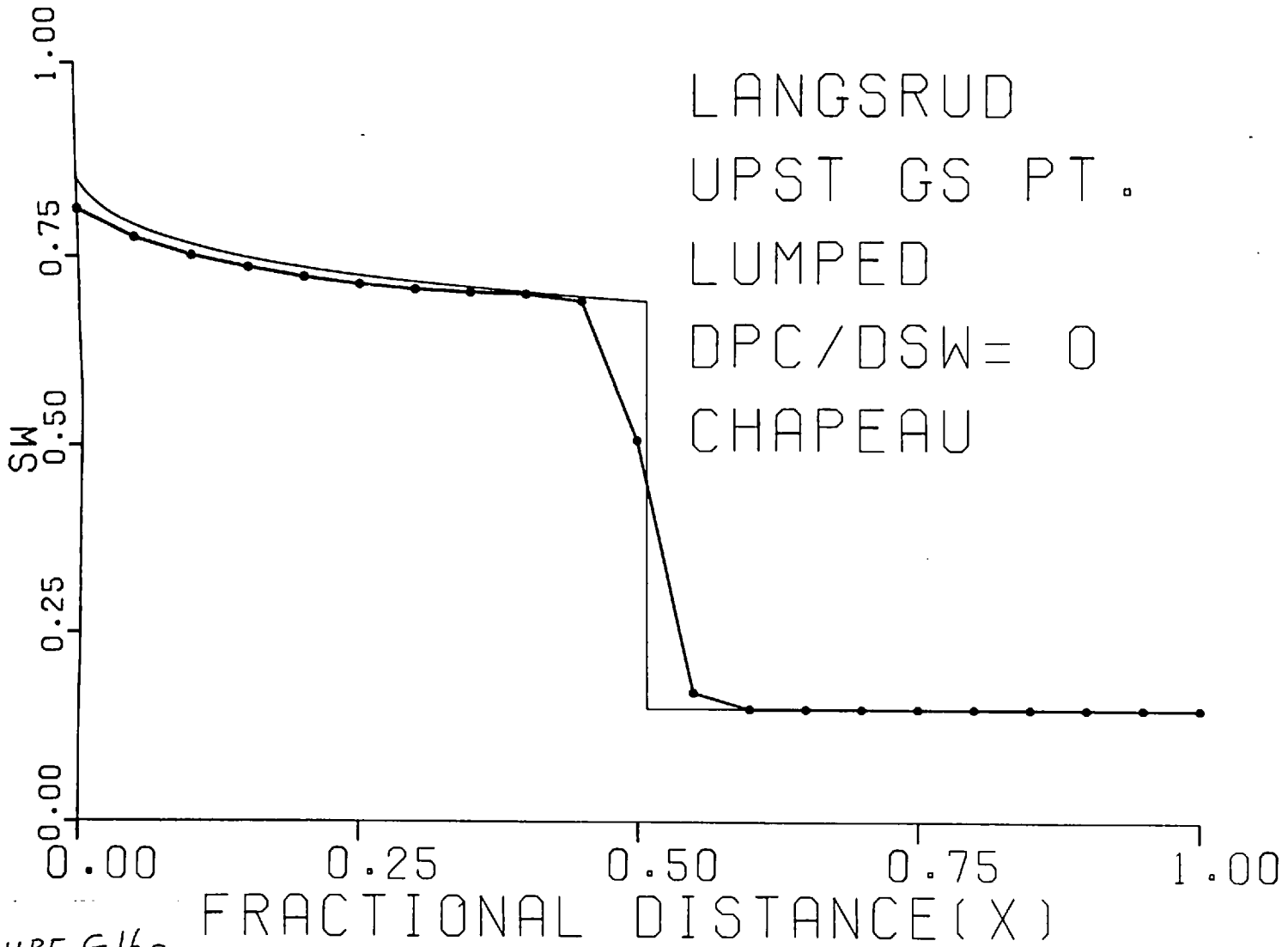

LANGSRUD
UPST GS PT.
LUMPED
DPC/DSW= 0
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G16a

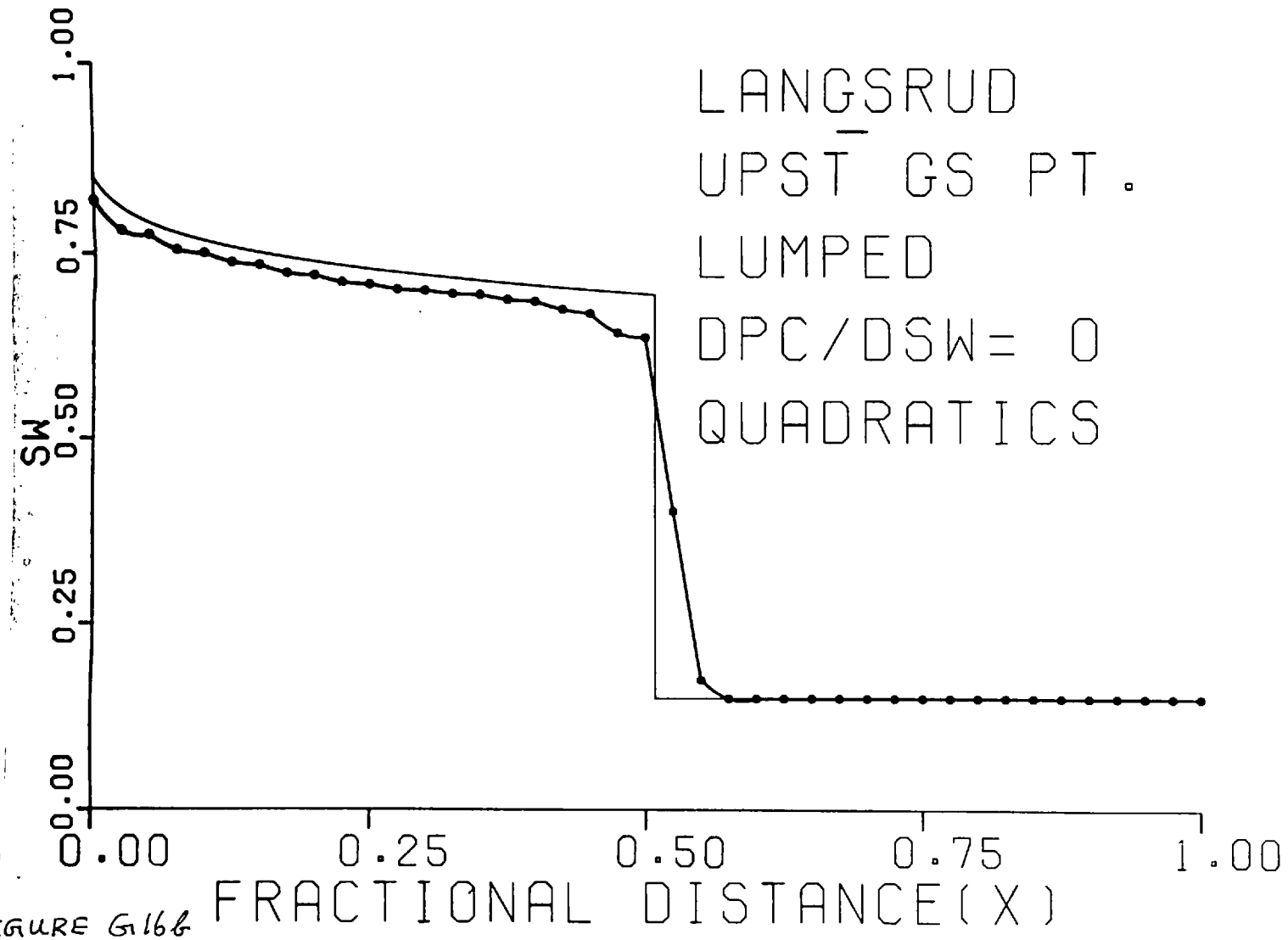

LANGSRUD
UPST GS PT.
LUMPED
DPC/DSW= 0
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G166

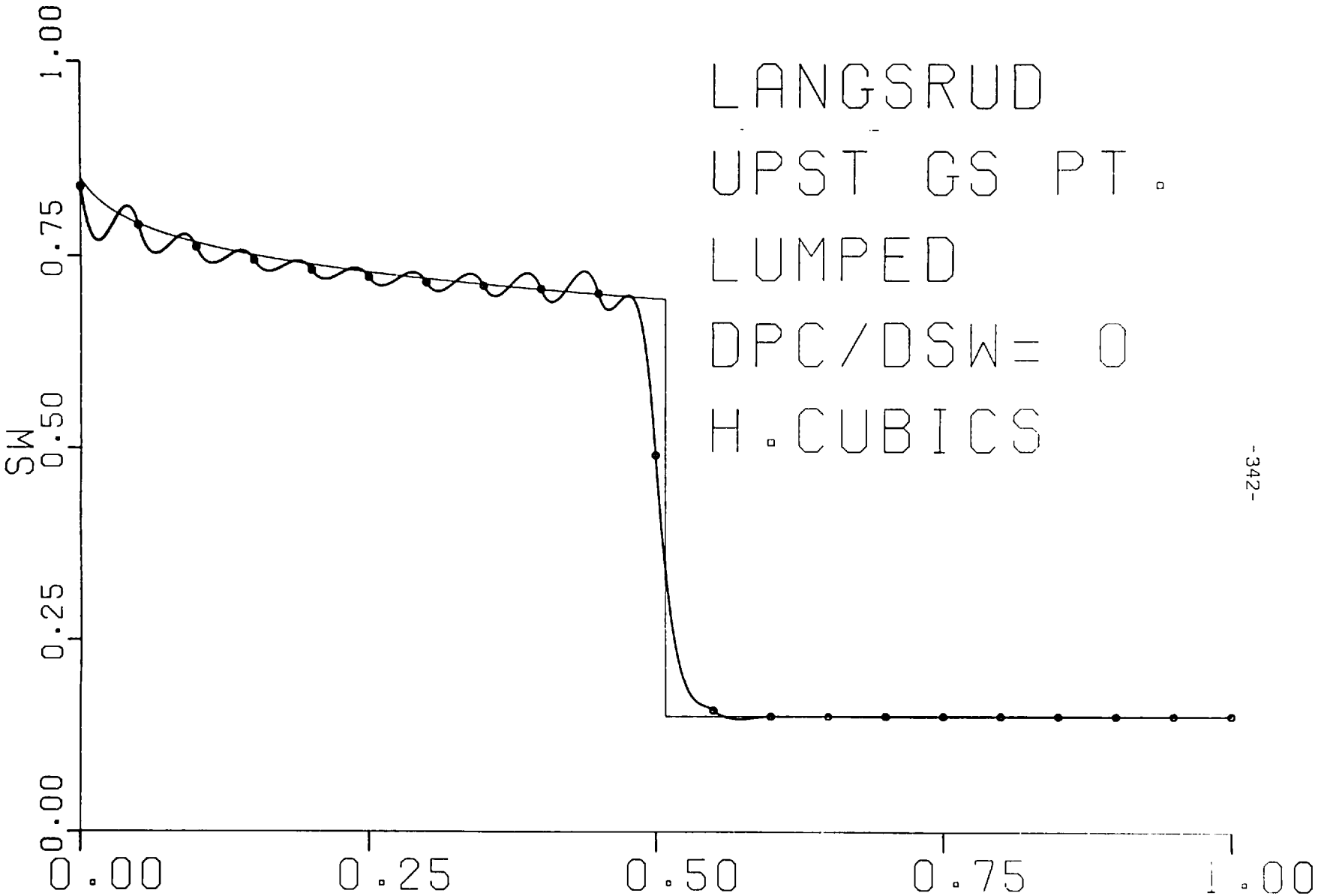

LANGSRUD
UPST GS PT.
LUMPED
DPC/DSW= 0
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G16c

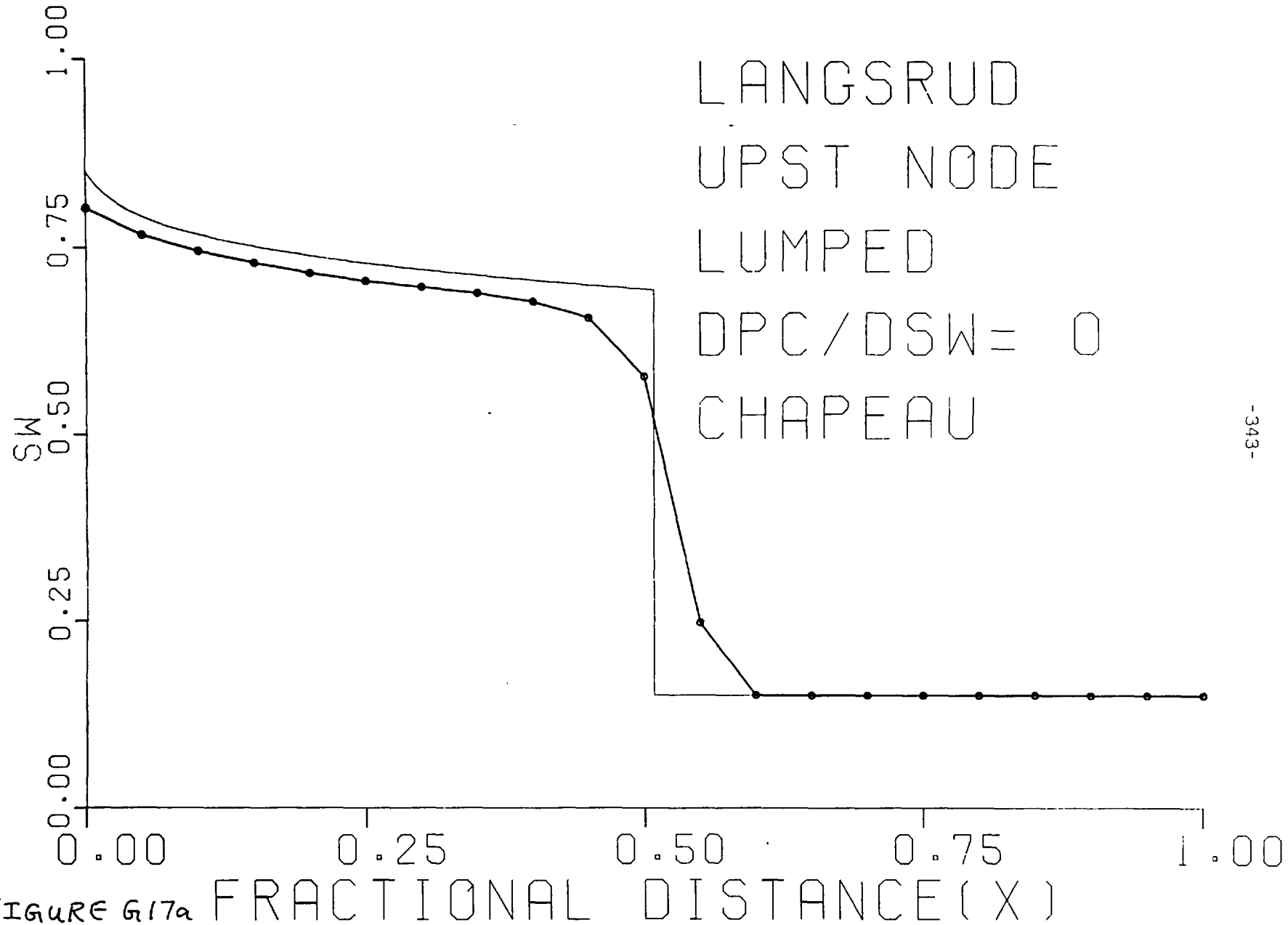

LANGSRUD
UPST NODE
LUMPED
DPC/DSW= 0
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G17a

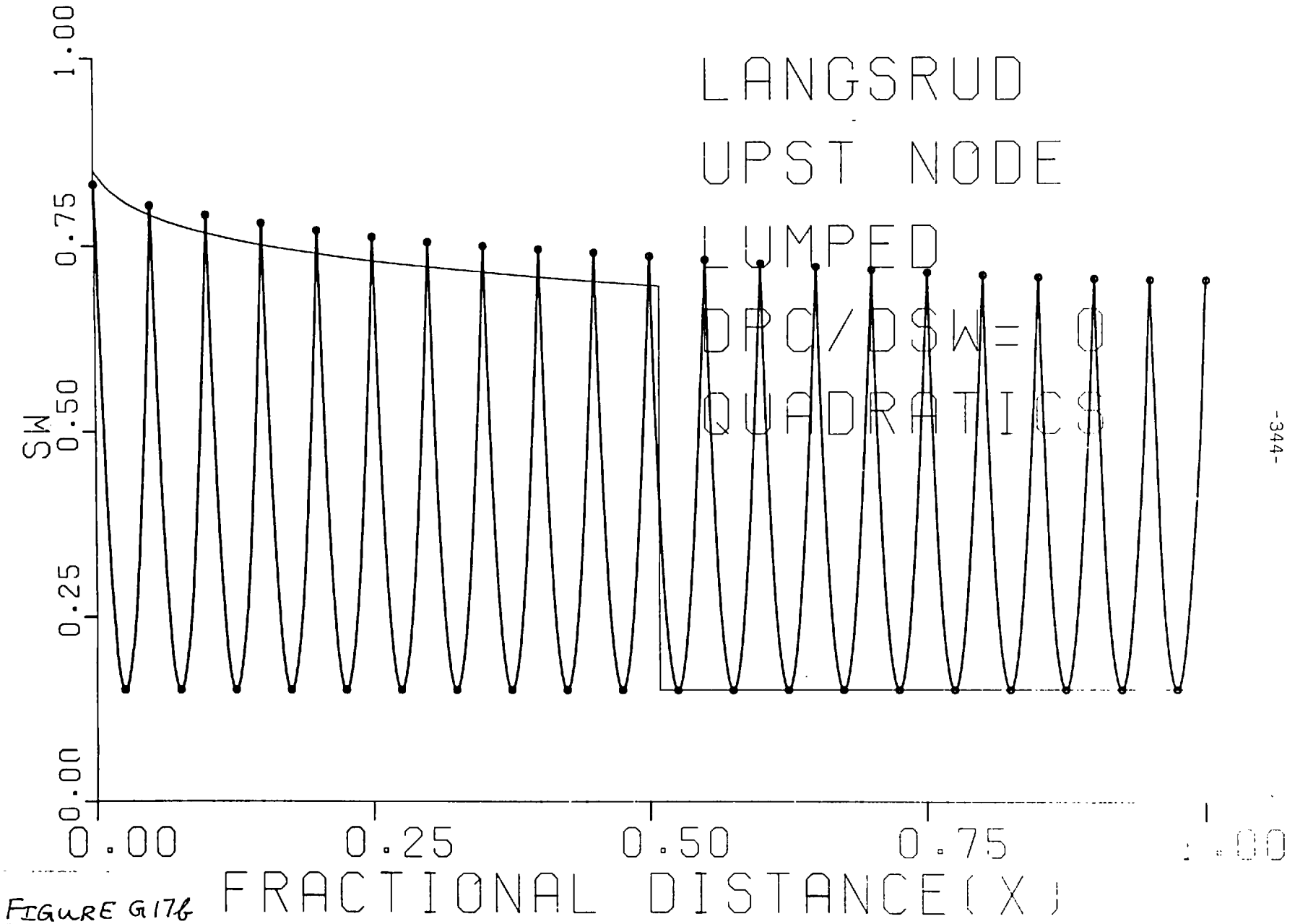

LANGSRUD
UPST NODE
LUMPED
DPC/DSW=0
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G17b

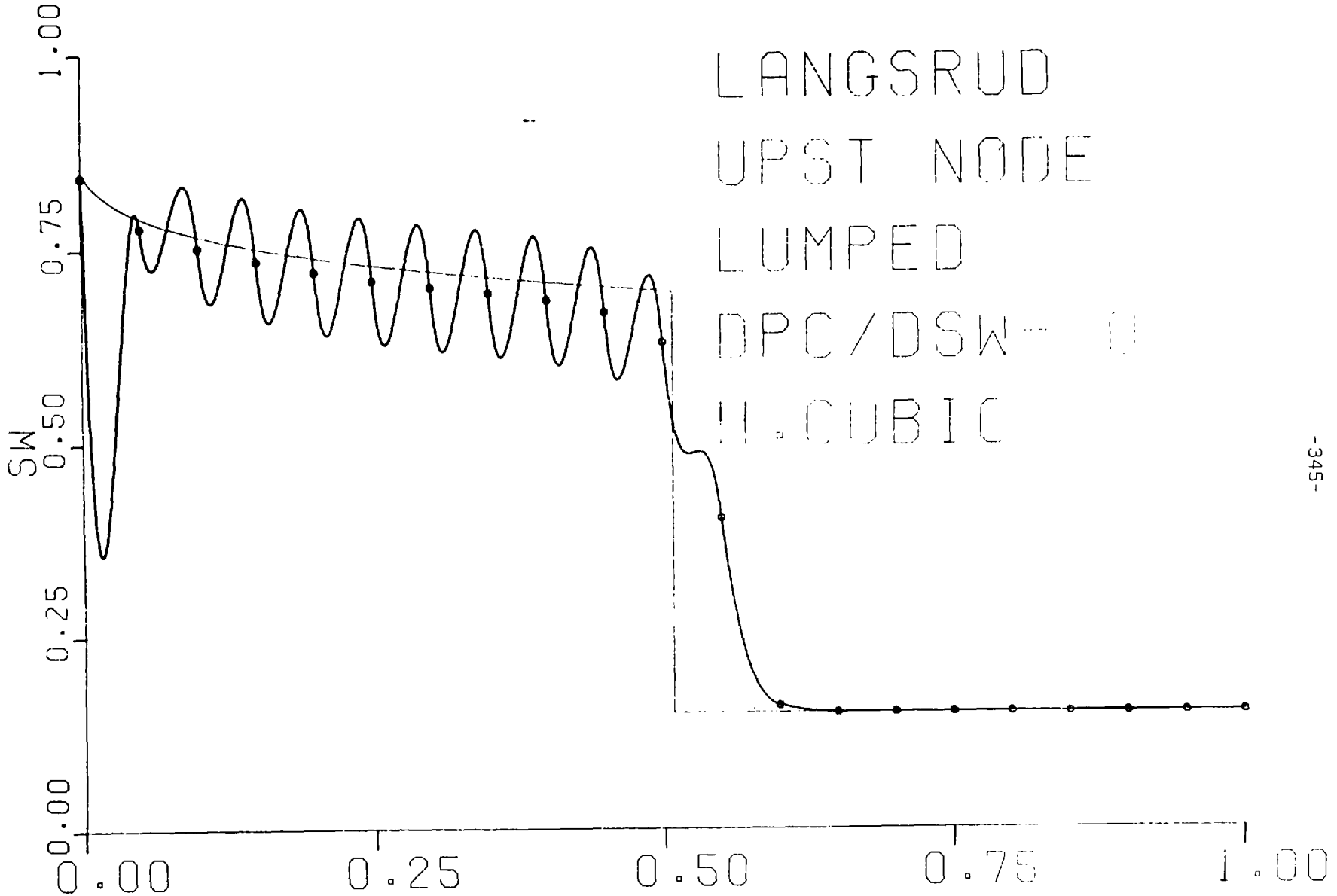

LANGSRUD
UPST NODE
LUMPED
DPC/DSW= 0
H.CUBIC
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G17c

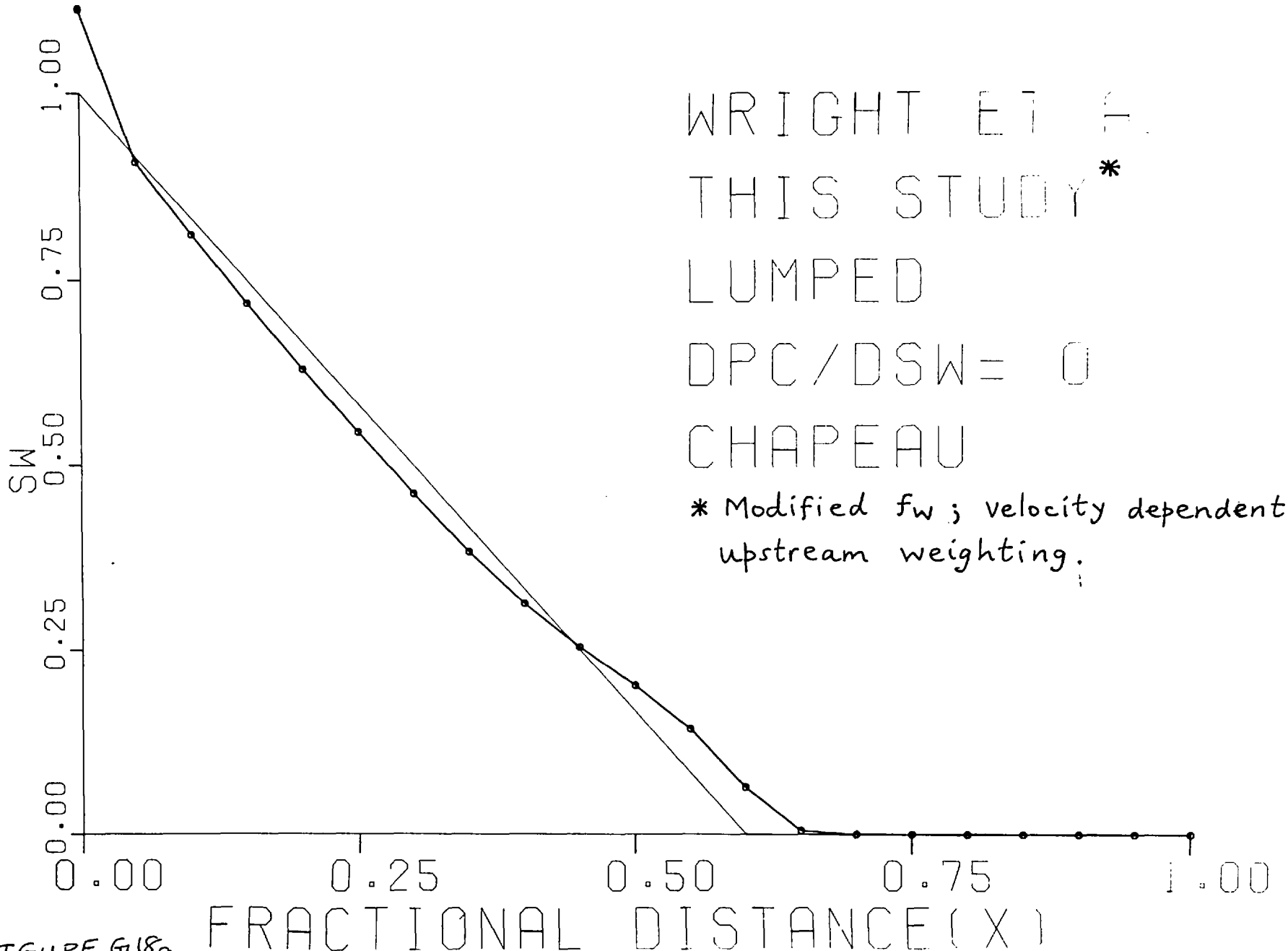

WRIGHT ET AL
THIS STUDY*
LUMPED
DPC/DSW= 0
CHAPEAU
* Modified fw ; velocity dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G18a

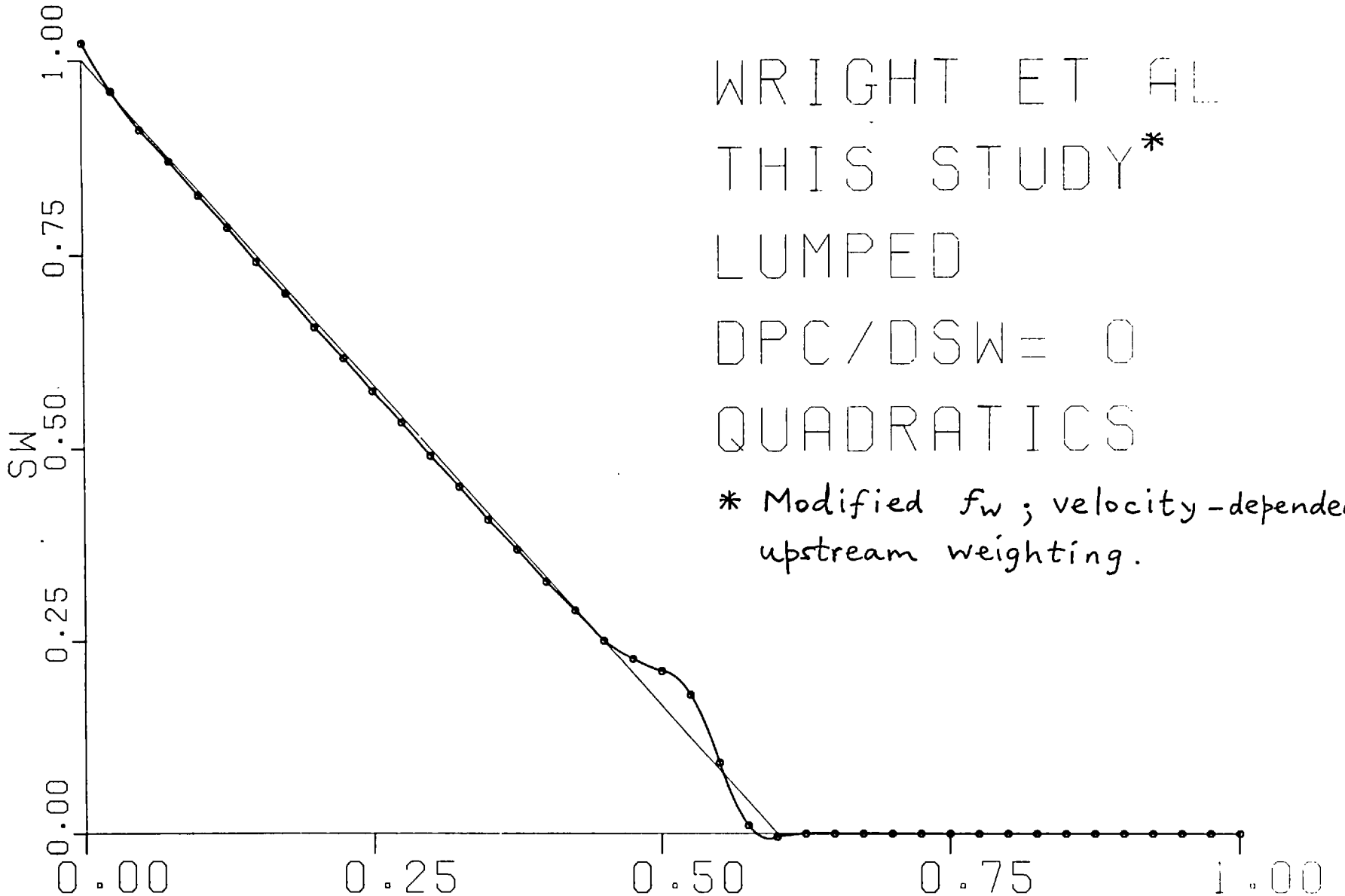

WRIGHT ET AL
THIS STUDY*
LUMPED
DPC/DSW= 0
QUADRATICS
* Modified f_w ; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G18b

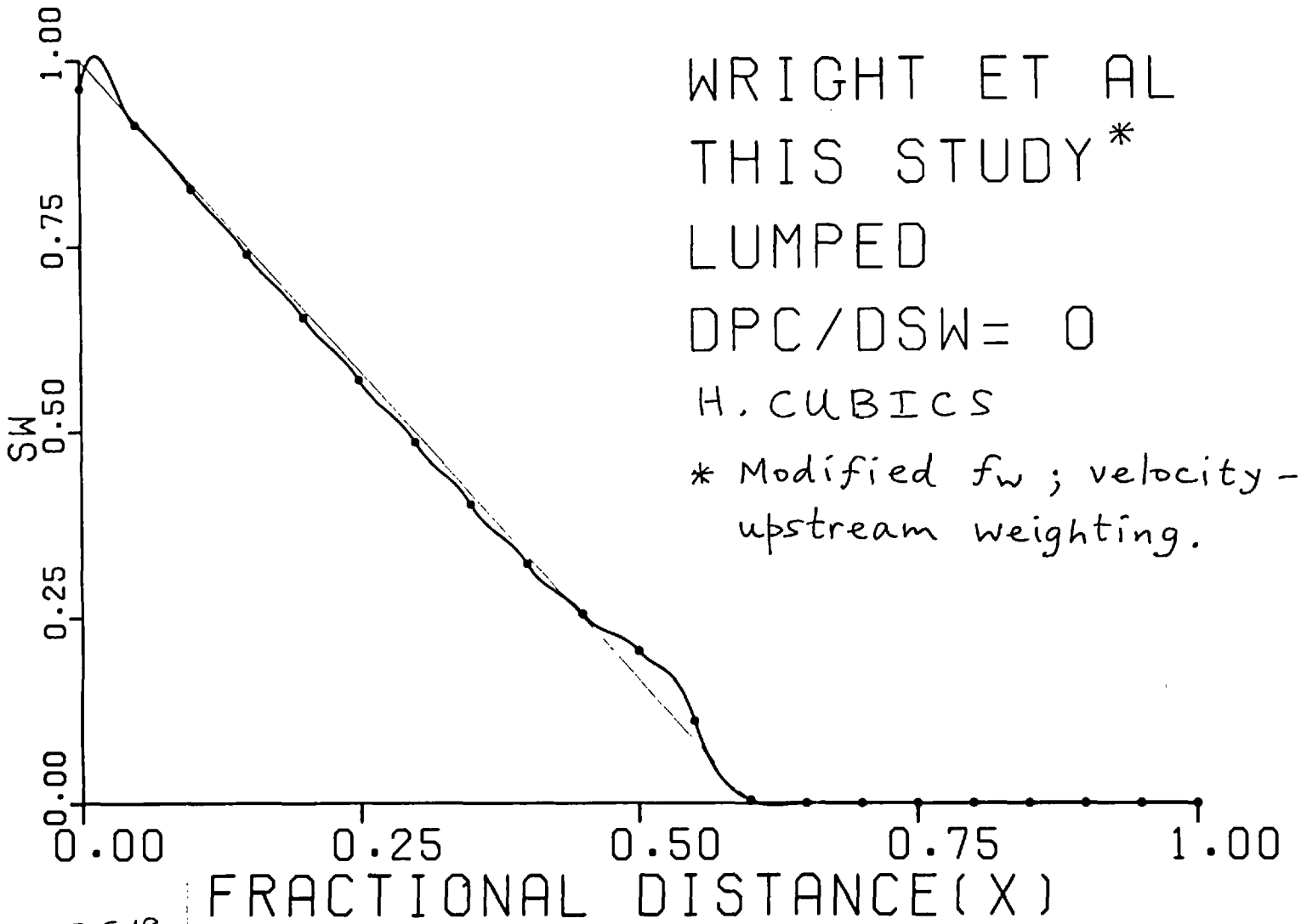

WRIGHT ET AL
THIS STUDY*
LUMPED
DPC/DSW= 0
H. CUBICS
* Modified f_w ; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G18c

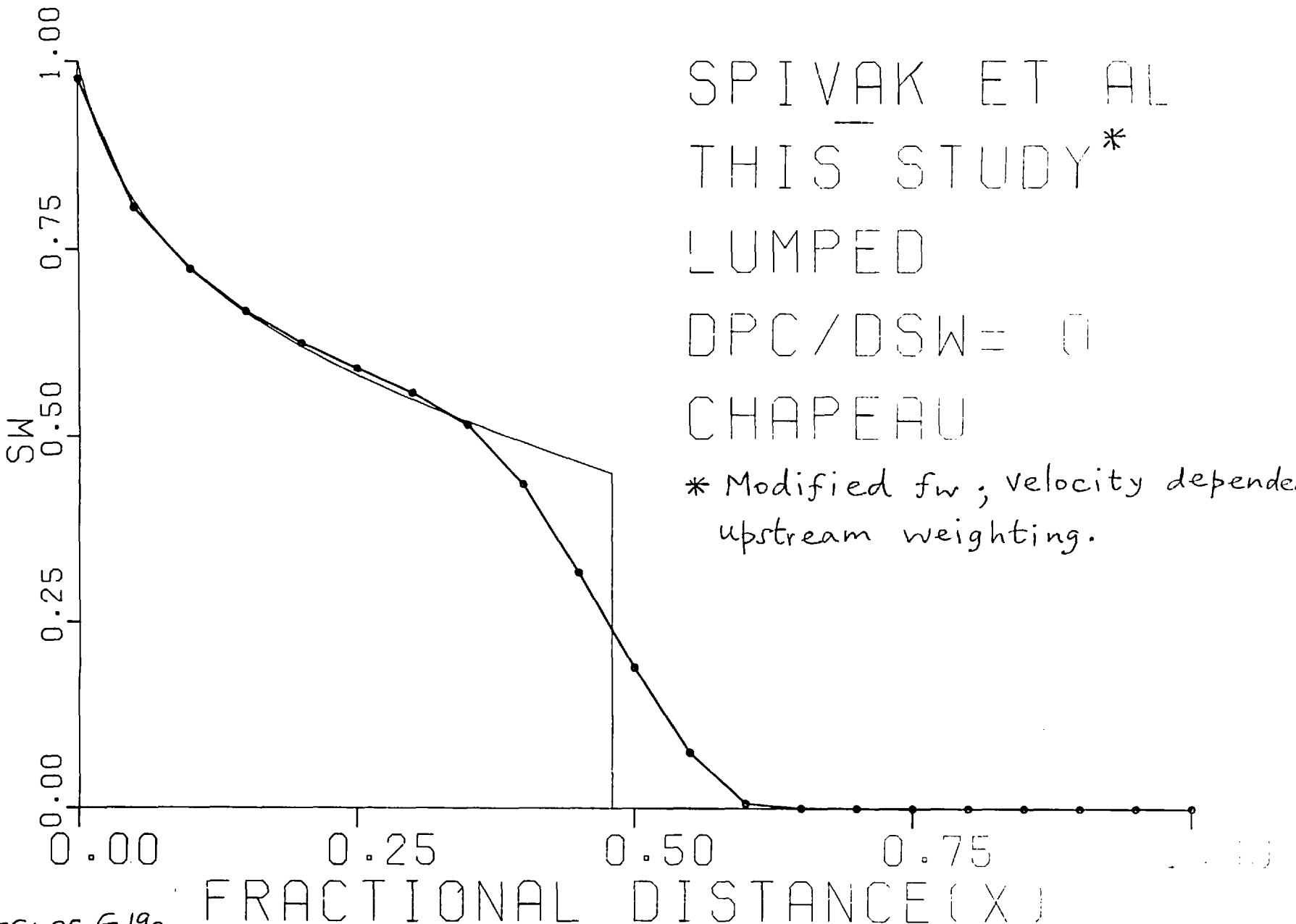

SPIVAK ET AL
THIS STUDY*
LUMPED
DPC/DSW= 0
CHAPEAU
* Modified fw ; velocity dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
FRACTIONAL DISTANCE(X)

FIGURE G19a

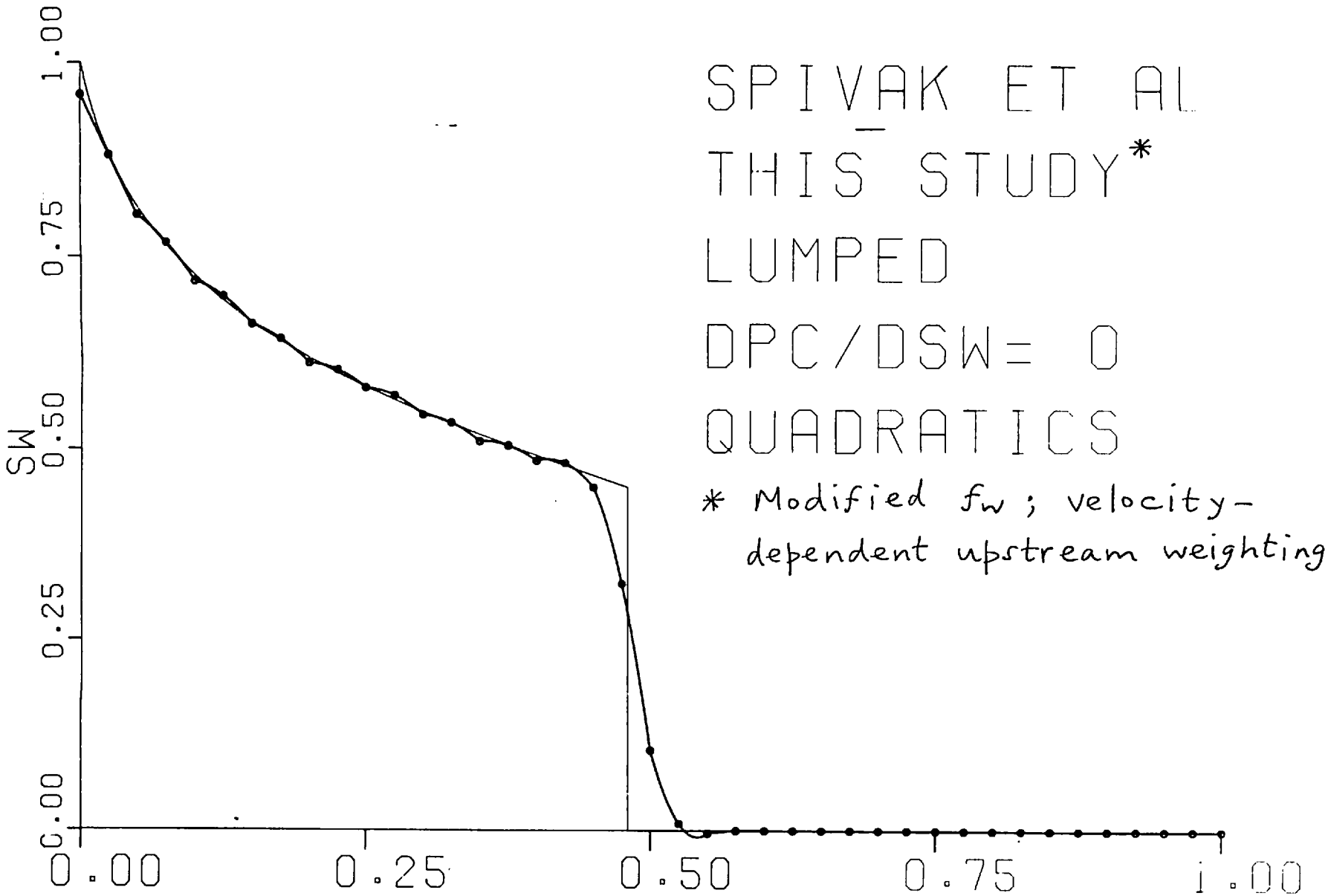

SPIVAK ET AL
THIS STUDY*
LUMPED
DPC/DSW= 0
QUADRATICS
* Modified f_w; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G19b

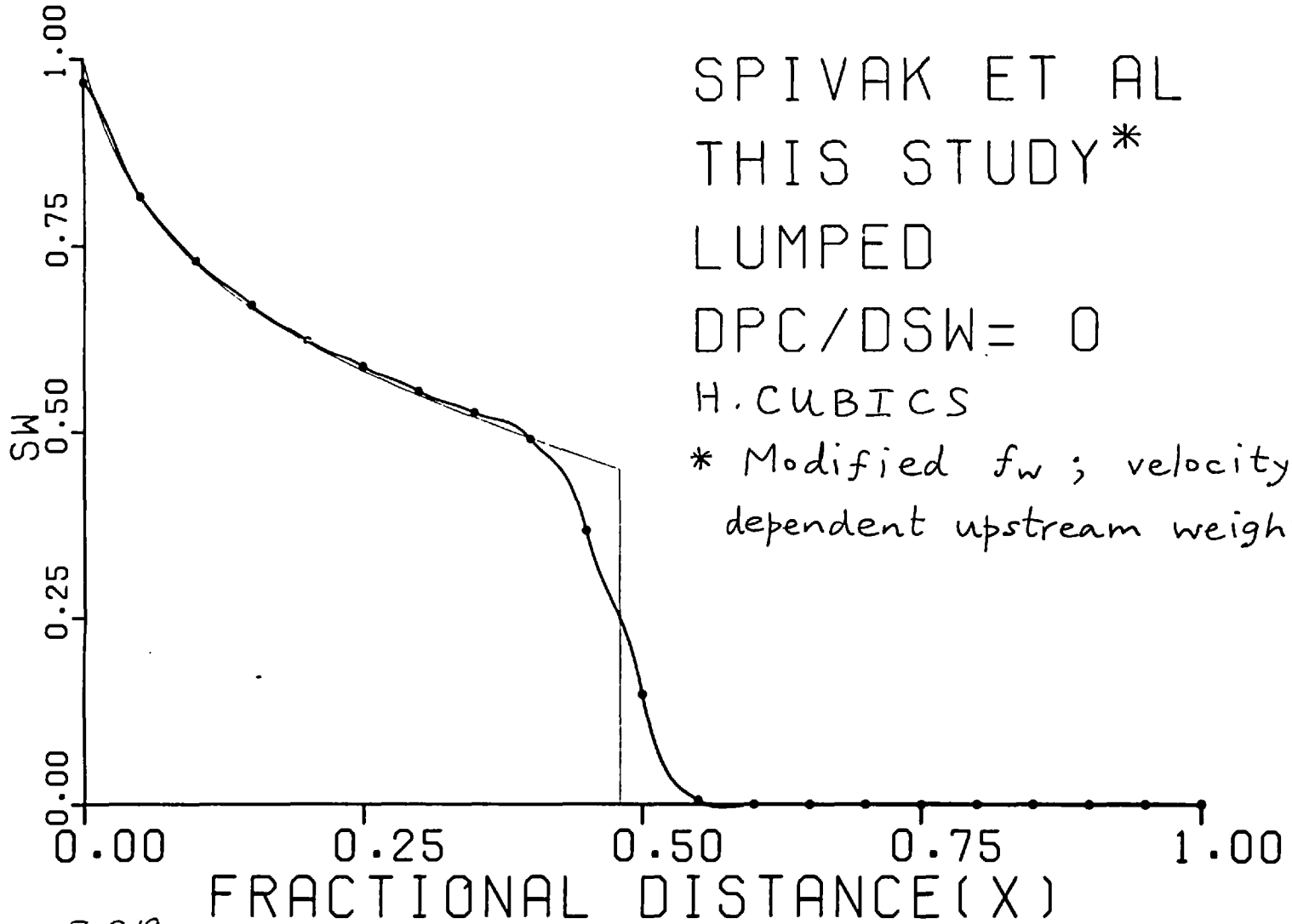

SPIVAK ET AL
THIS STUDY*
LUMPED
DPC/DSW= 0
H.CUBICS
* Modified f_w ; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G19C

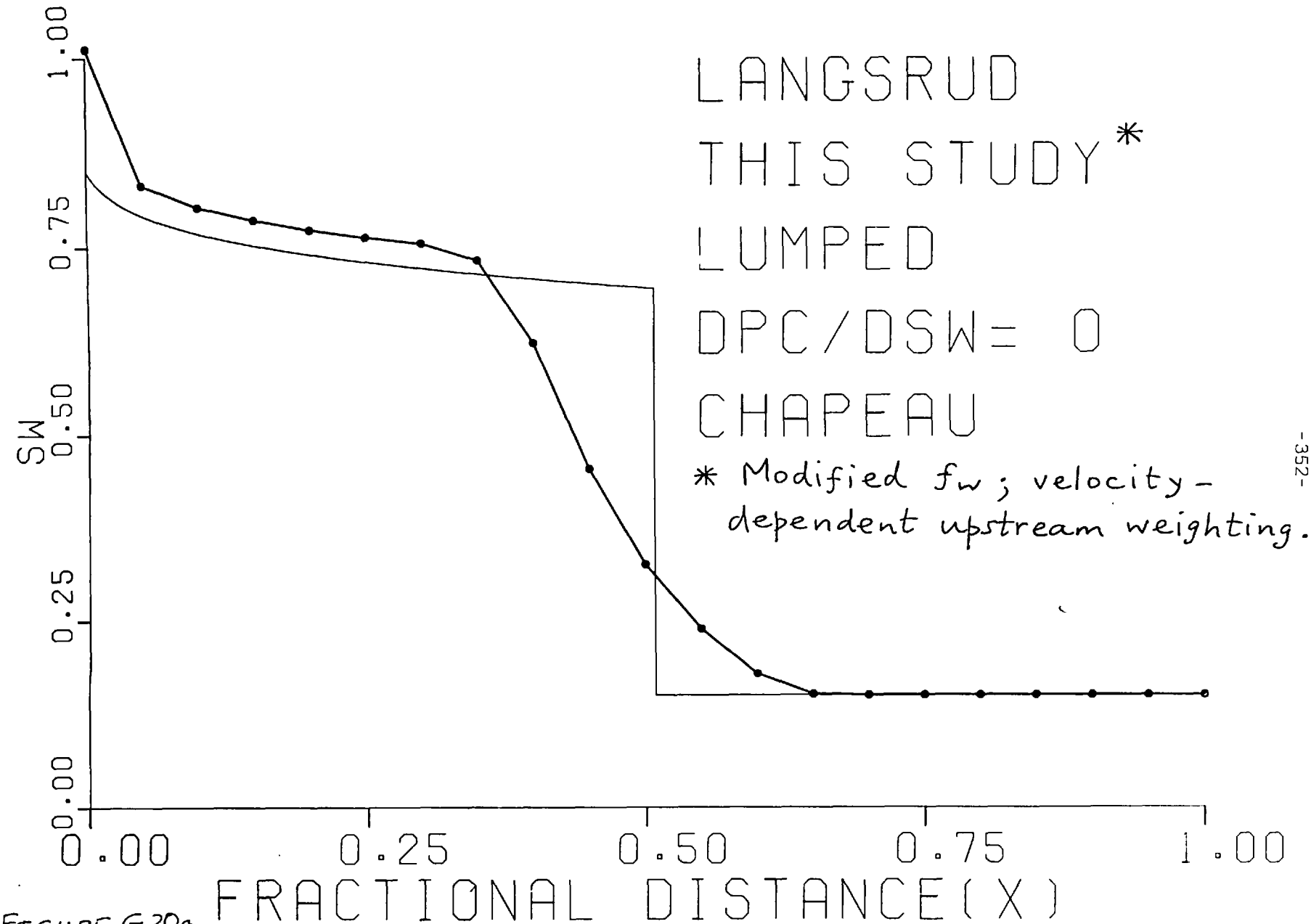

LANGSRUD
THIS STUDY*
LUMPED
DPC/DSW= 0
CHAPEAU
* Modified f_w; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G20a

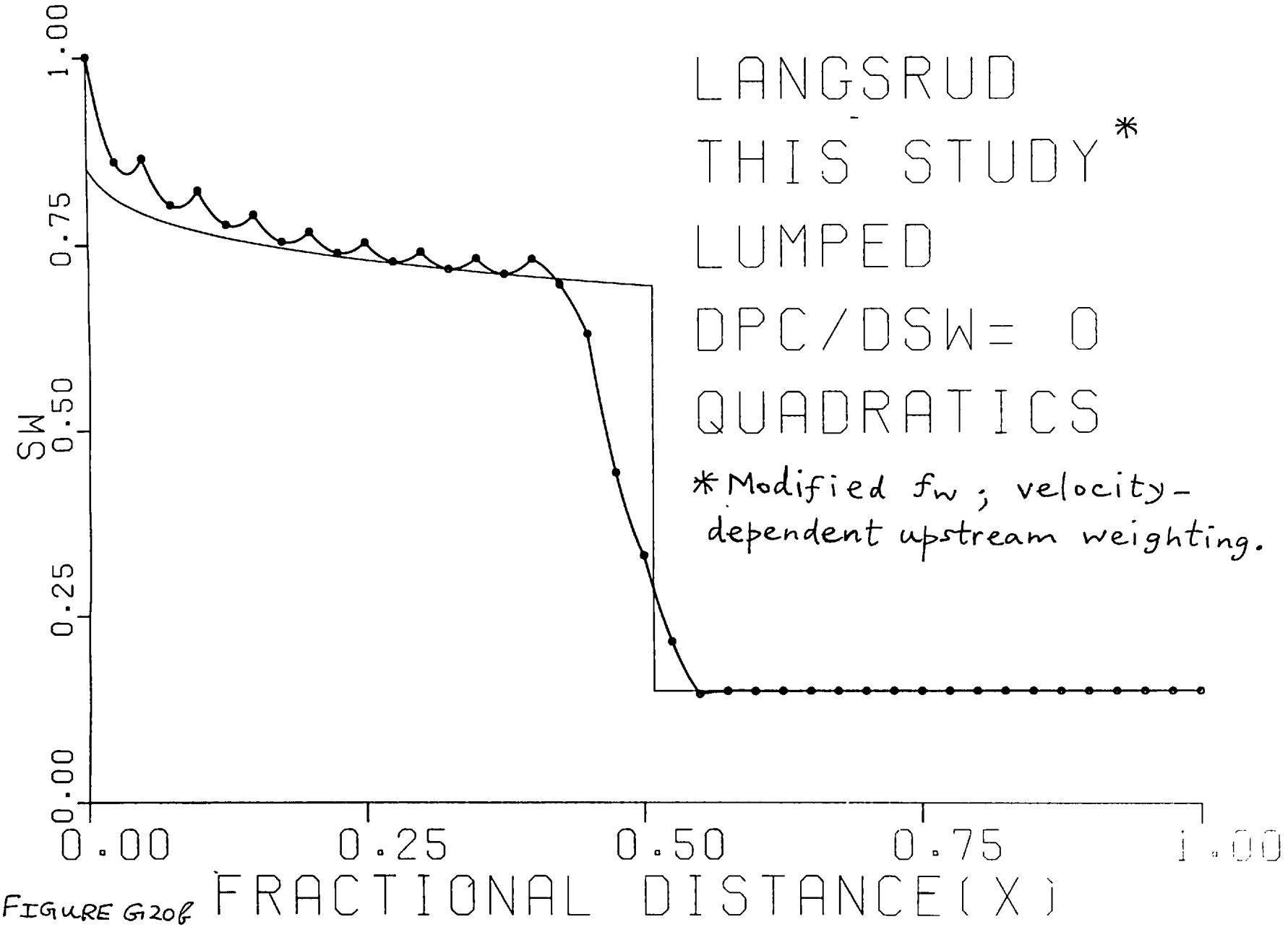

LANGSRUD
THIS STUDY*
LUMPED
DPC/DSW= 0
QUADRATICS
*Modified f_w; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G20f

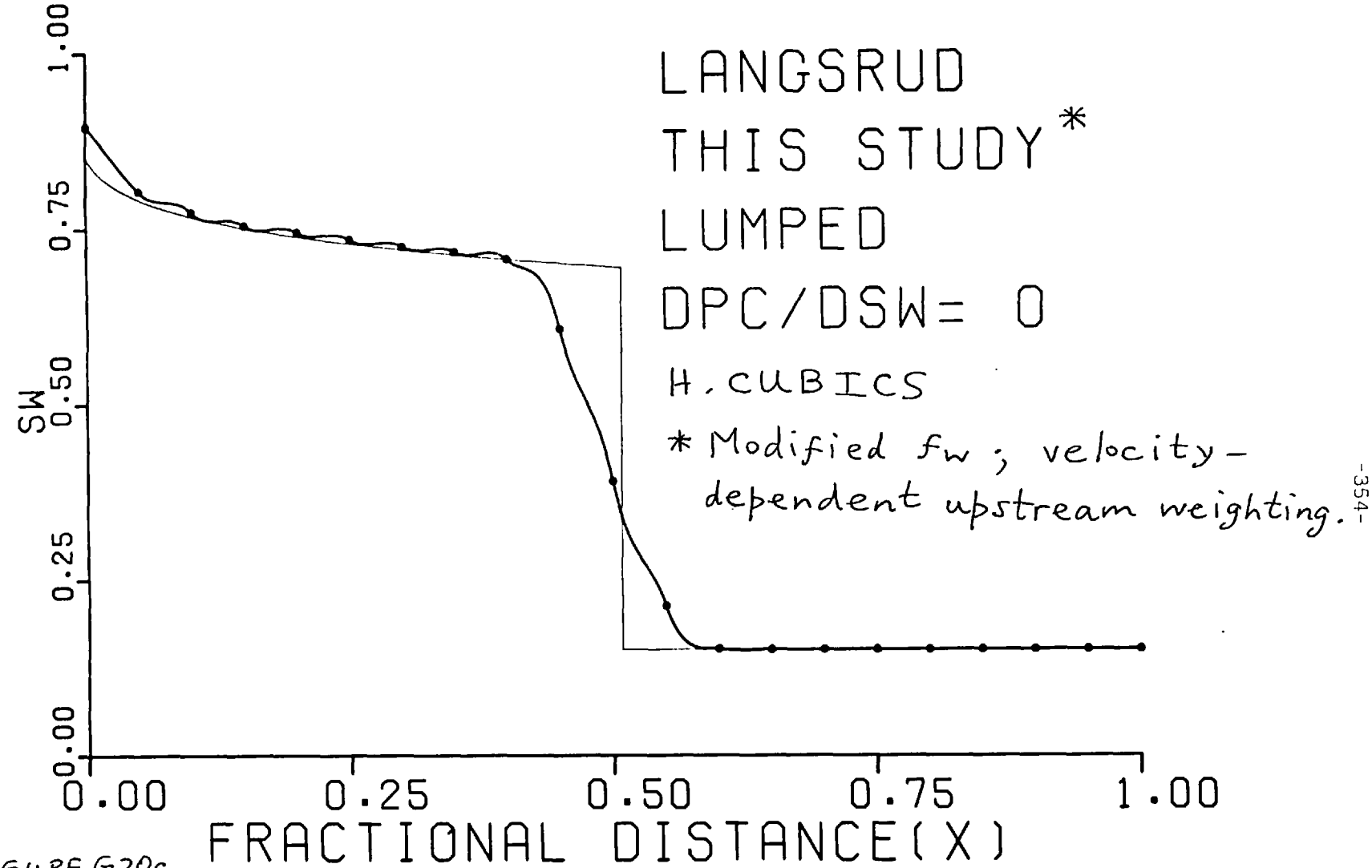

LANGSRUD
THIS STUDY*
LUMPED
DPC/DSW= 0
H.CUBICS
*Modified fw ; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G20c

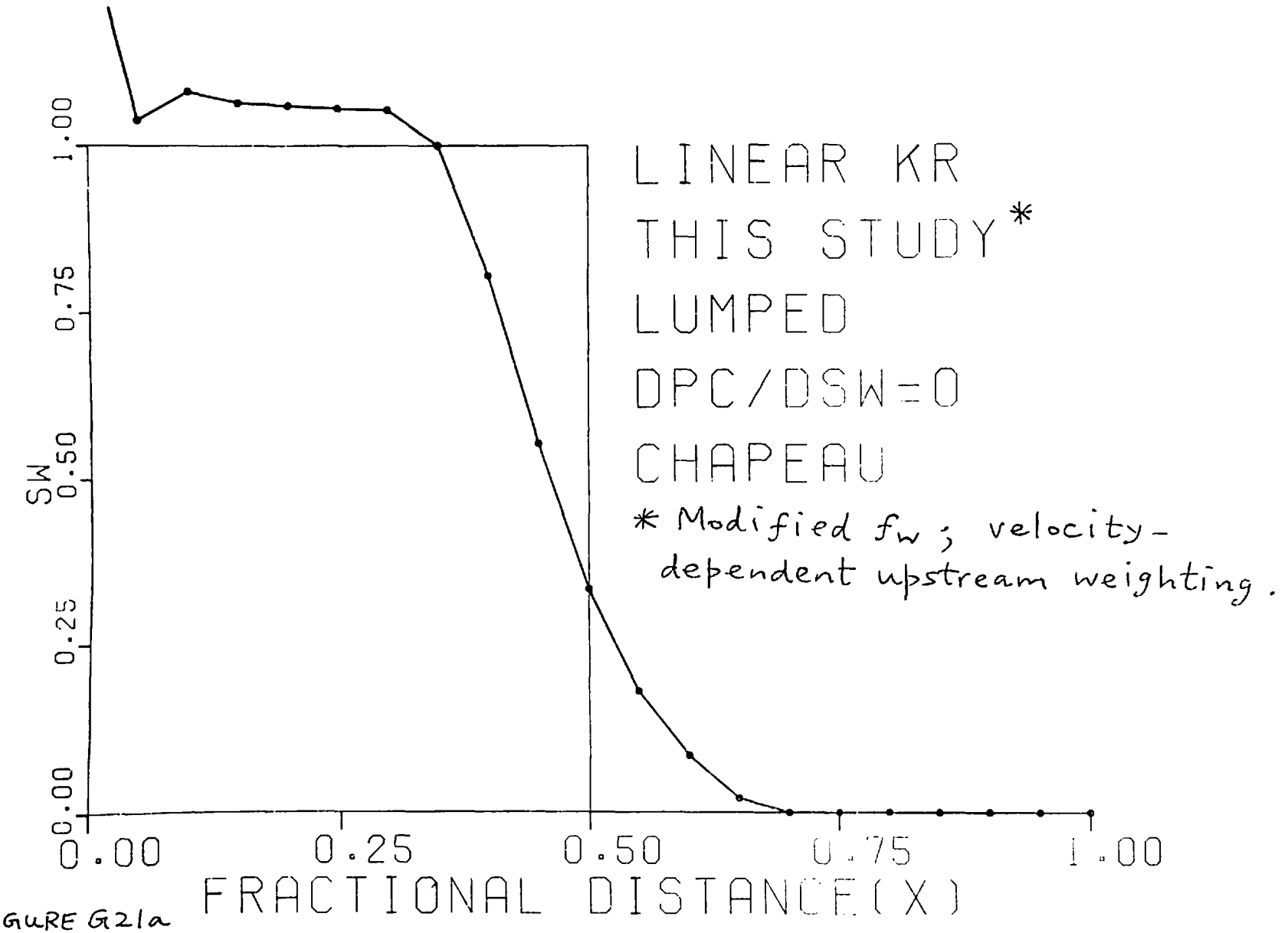

LINEAR KR
THIS STUDY*
LUMPED
DPC/DSW=0
CHAPEAU
* Modified f_w; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G21a

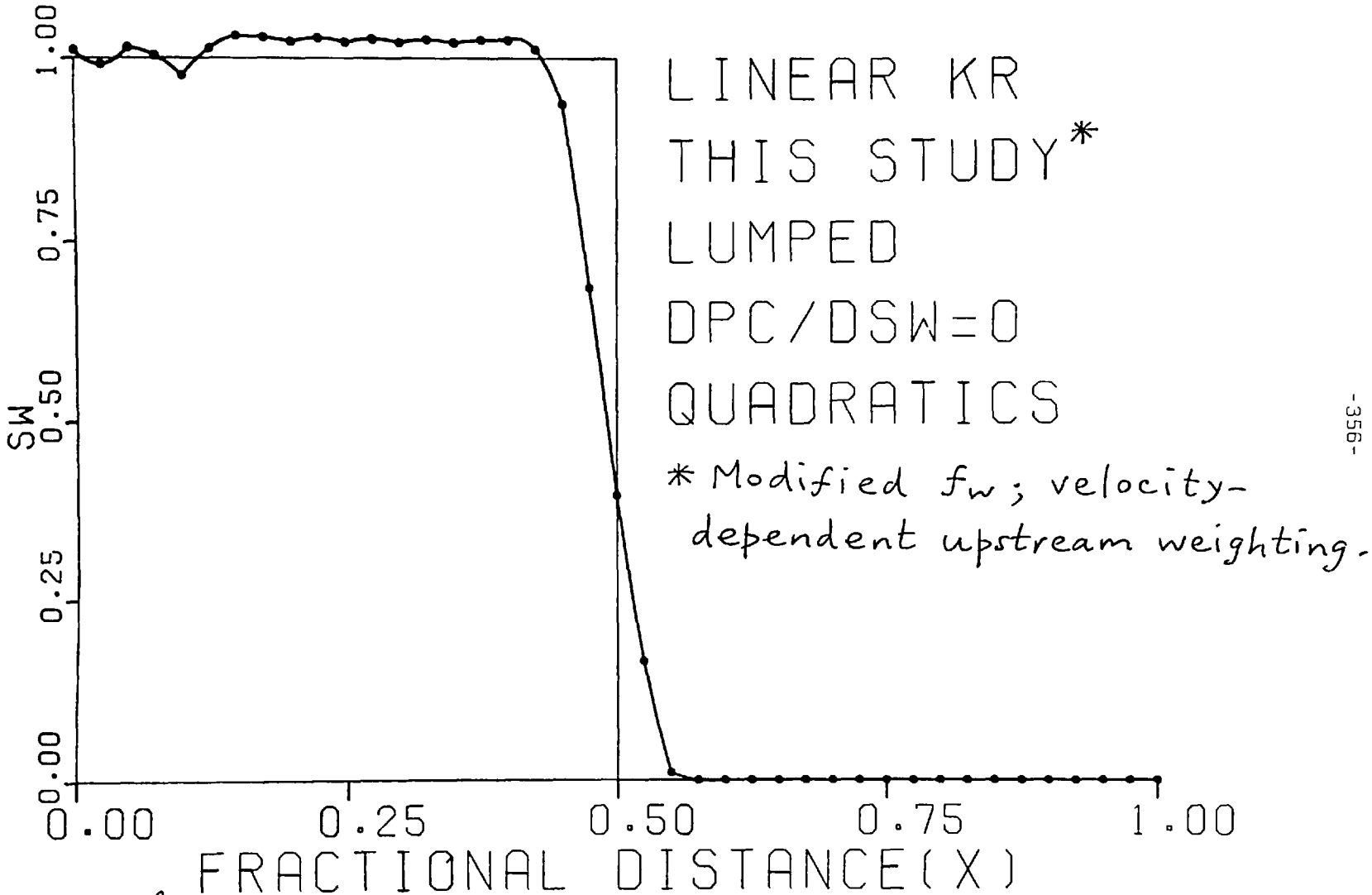

1.00
0.75
0.50
0.25
0.00
SW
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)
LINEAR KR
THIS STUDY*
LUMPED
DPC/DSW=0
QUADRATICS
* Modified f_w; velocity-dependent upstream weighting.

FIGURE G21b

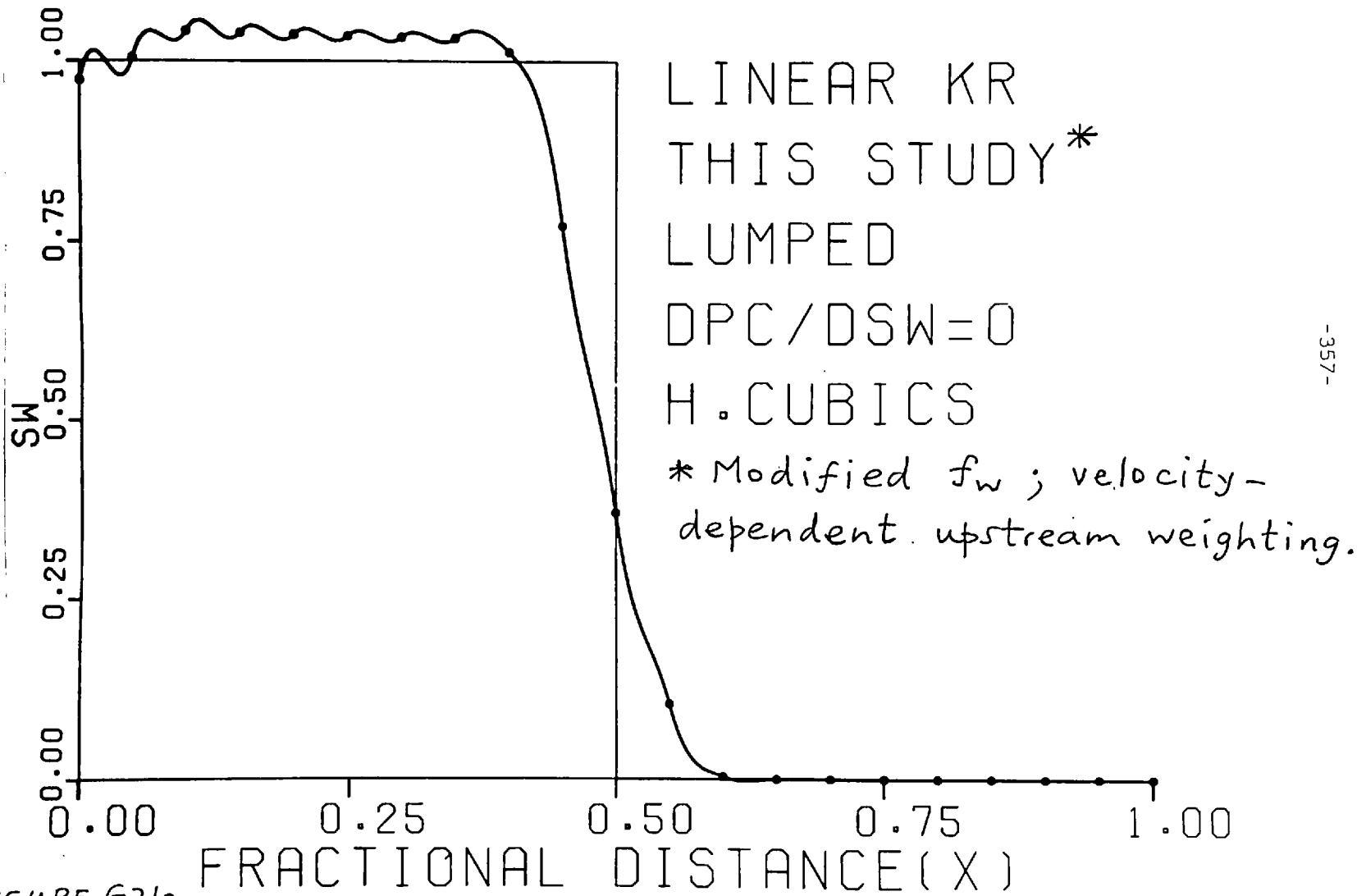

LINEAR KR
THIS STUDY*
LUMPED
DPC/DSW=0
H.CUBICS
* Modified f_w ; velocity-dependent upstream weighting.
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G21c

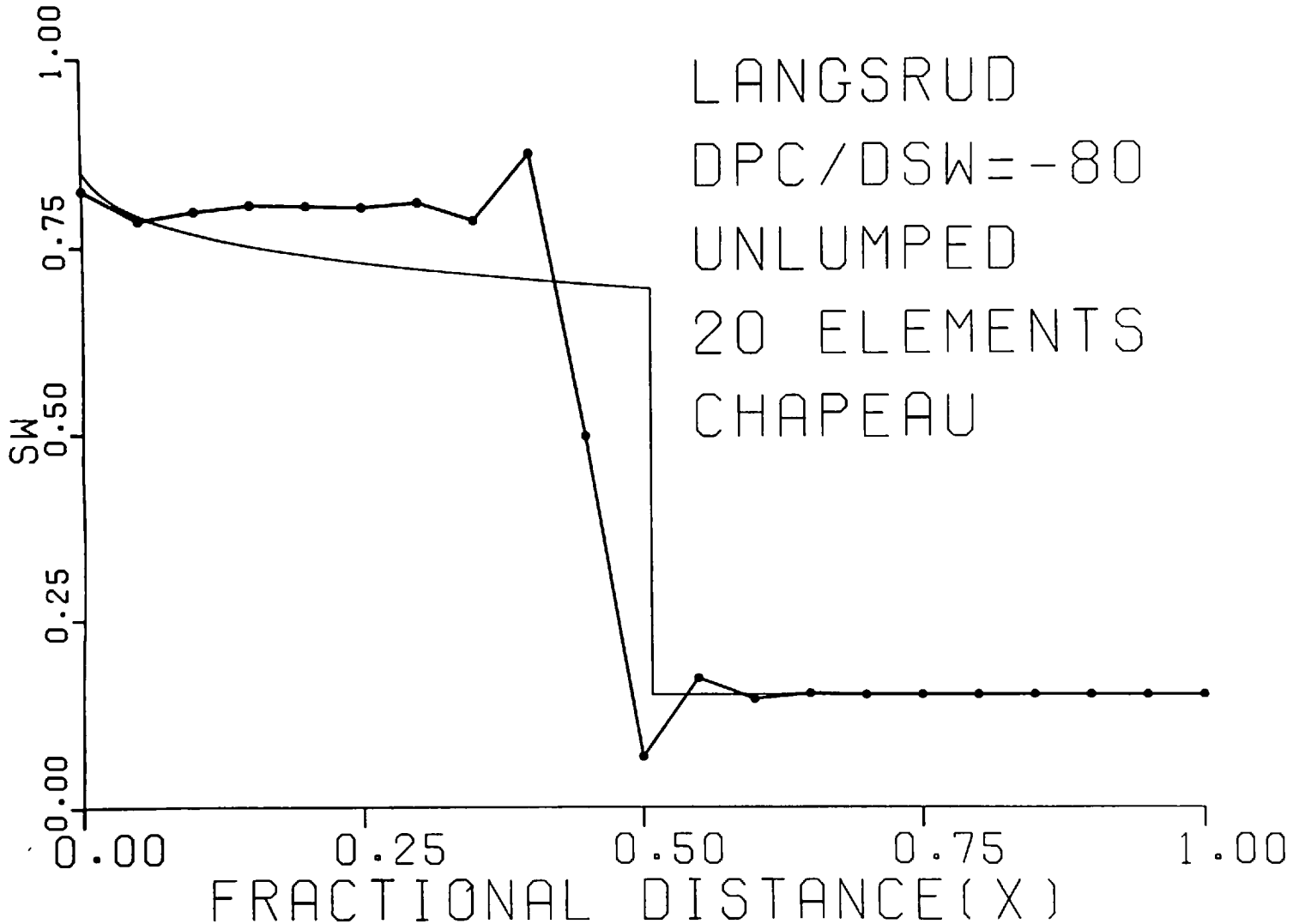

LANGSRUD
DPC/DSW=-80
UNLUMPED
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G22a

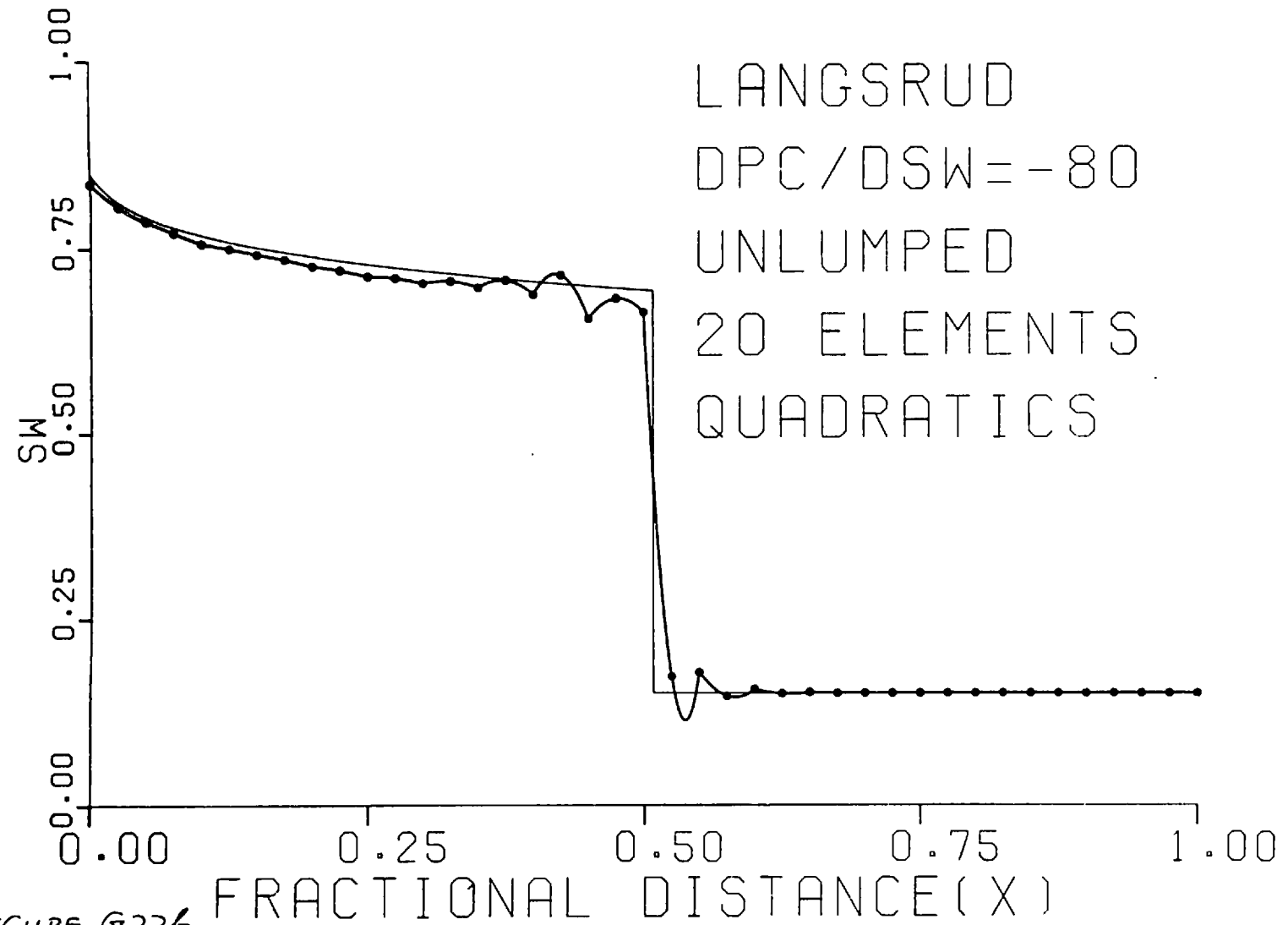

LANGSRUD
DPC/DSW=-80
UNLUMPED
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G226

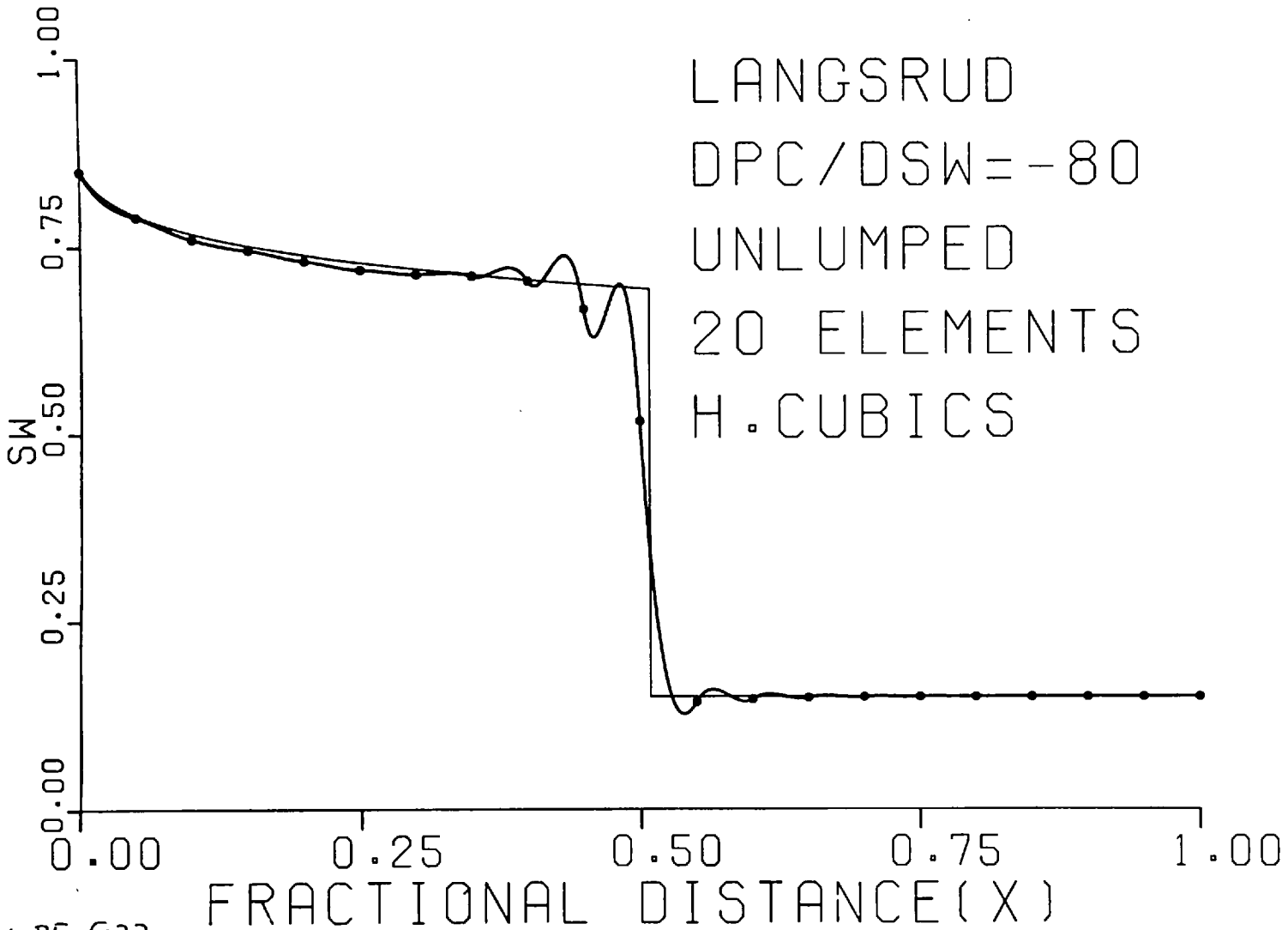

LANGSRUD
DPC/DSW=-80
UNLUMPED
20 ELEMENTS
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G22c

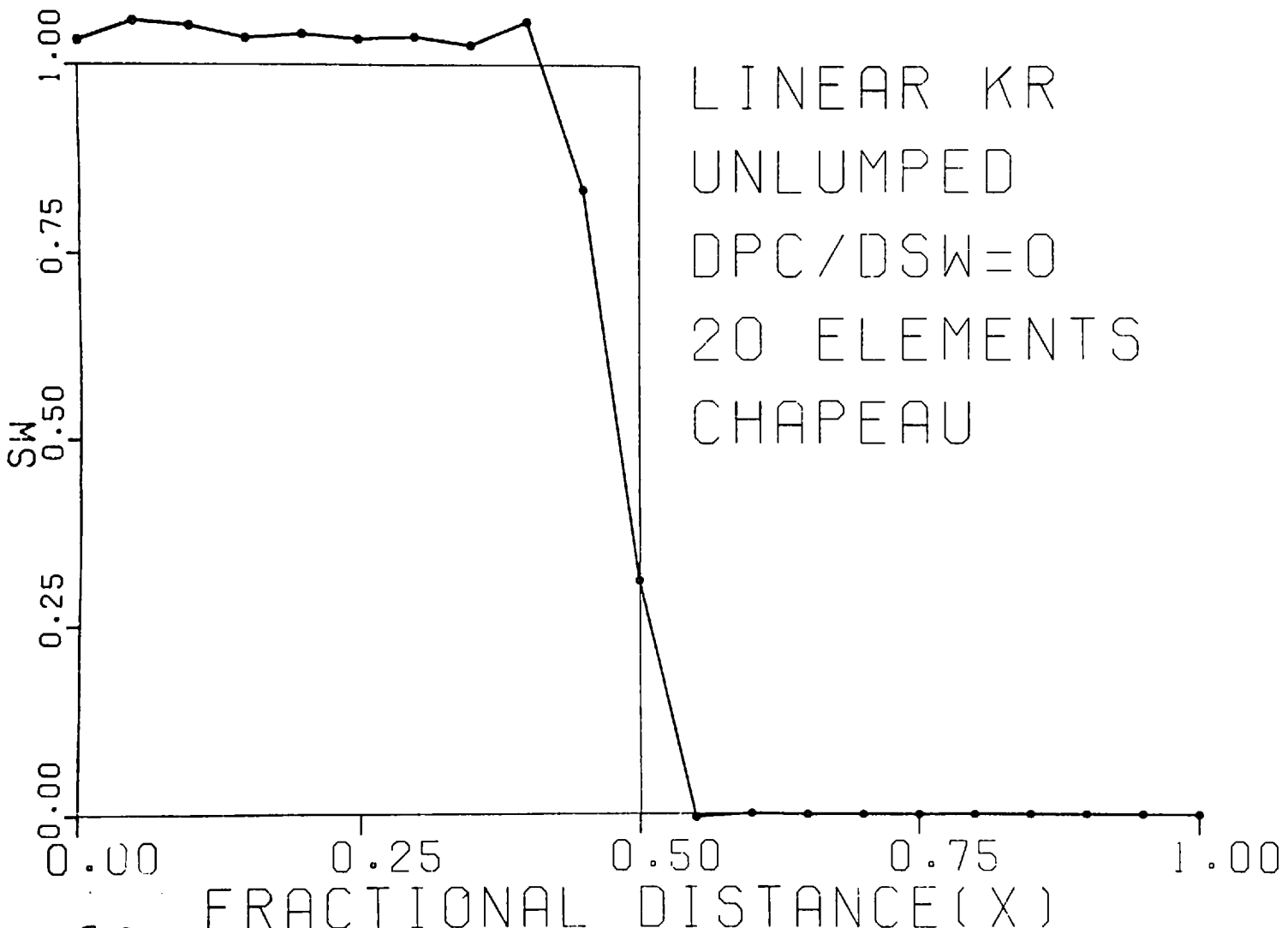

LINEAR KR
UNLUMPED
DPC/DSW=0
20 ELEMENTS
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G23a

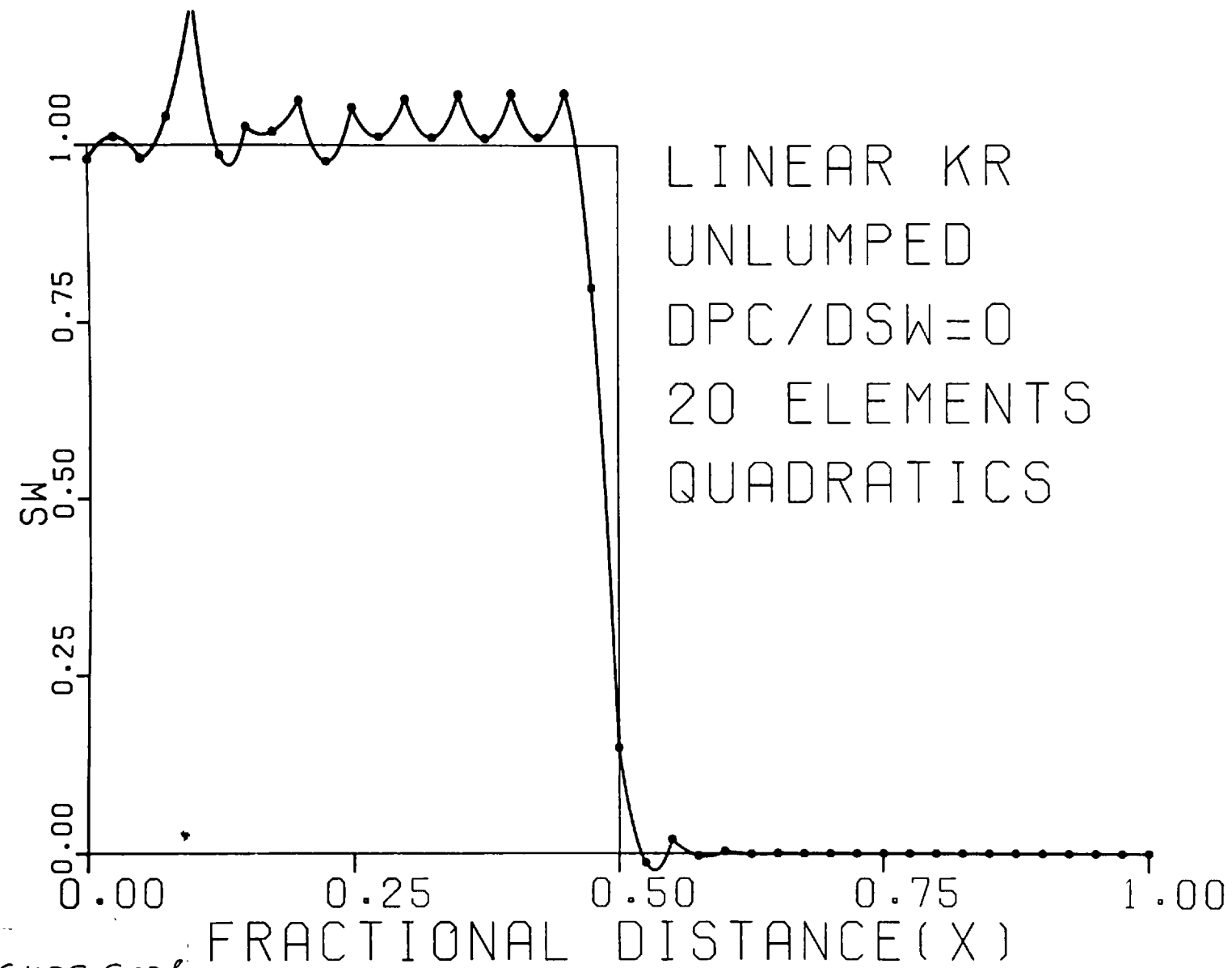

LINEAR KR
UNLUMPED
DPC/DSW=0
20 ELEMENTS
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G23b

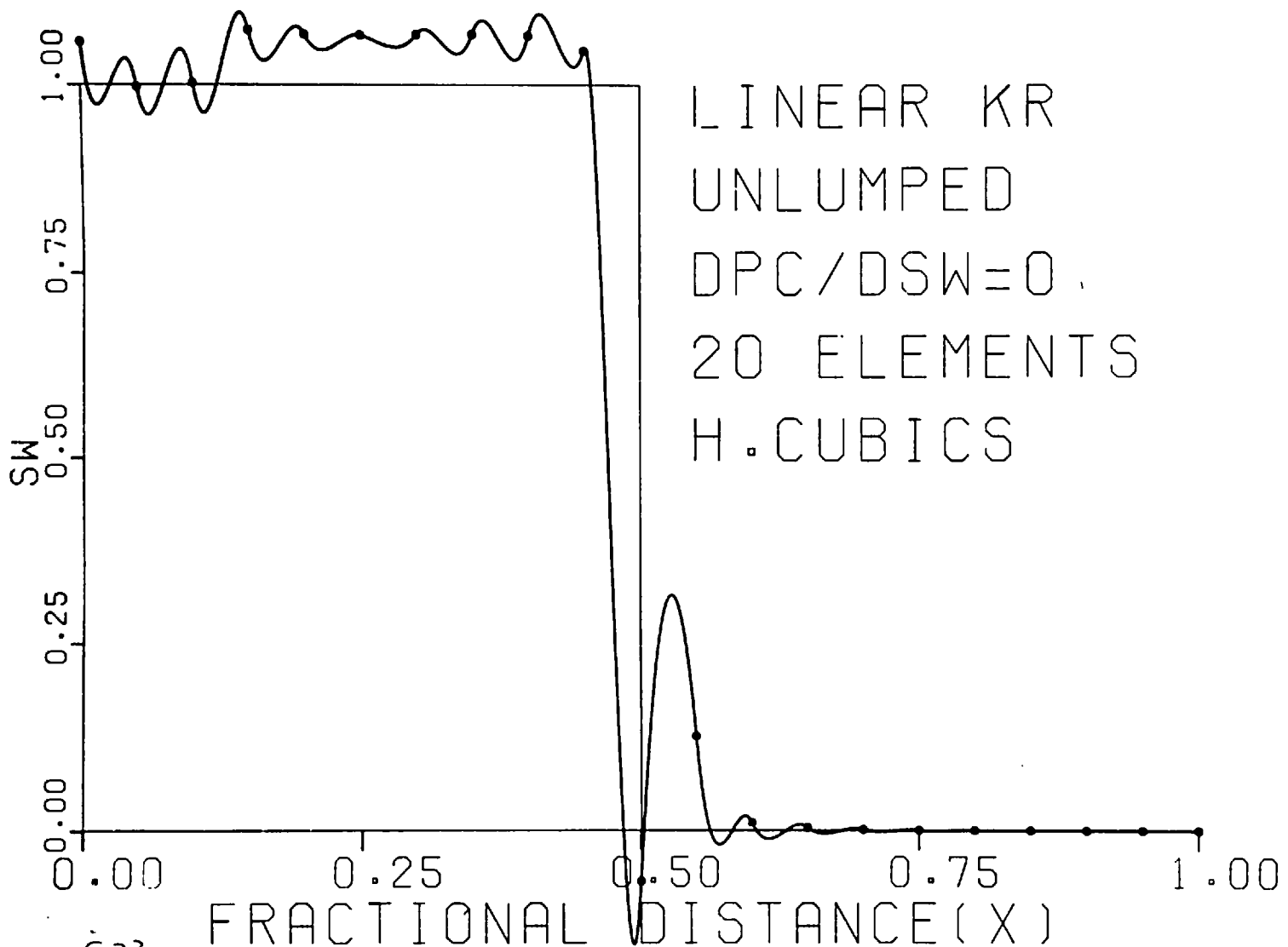

LINEAR KR
UNLUMPED
DPC/DSW=0.
20 ELEMENTS
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G23c

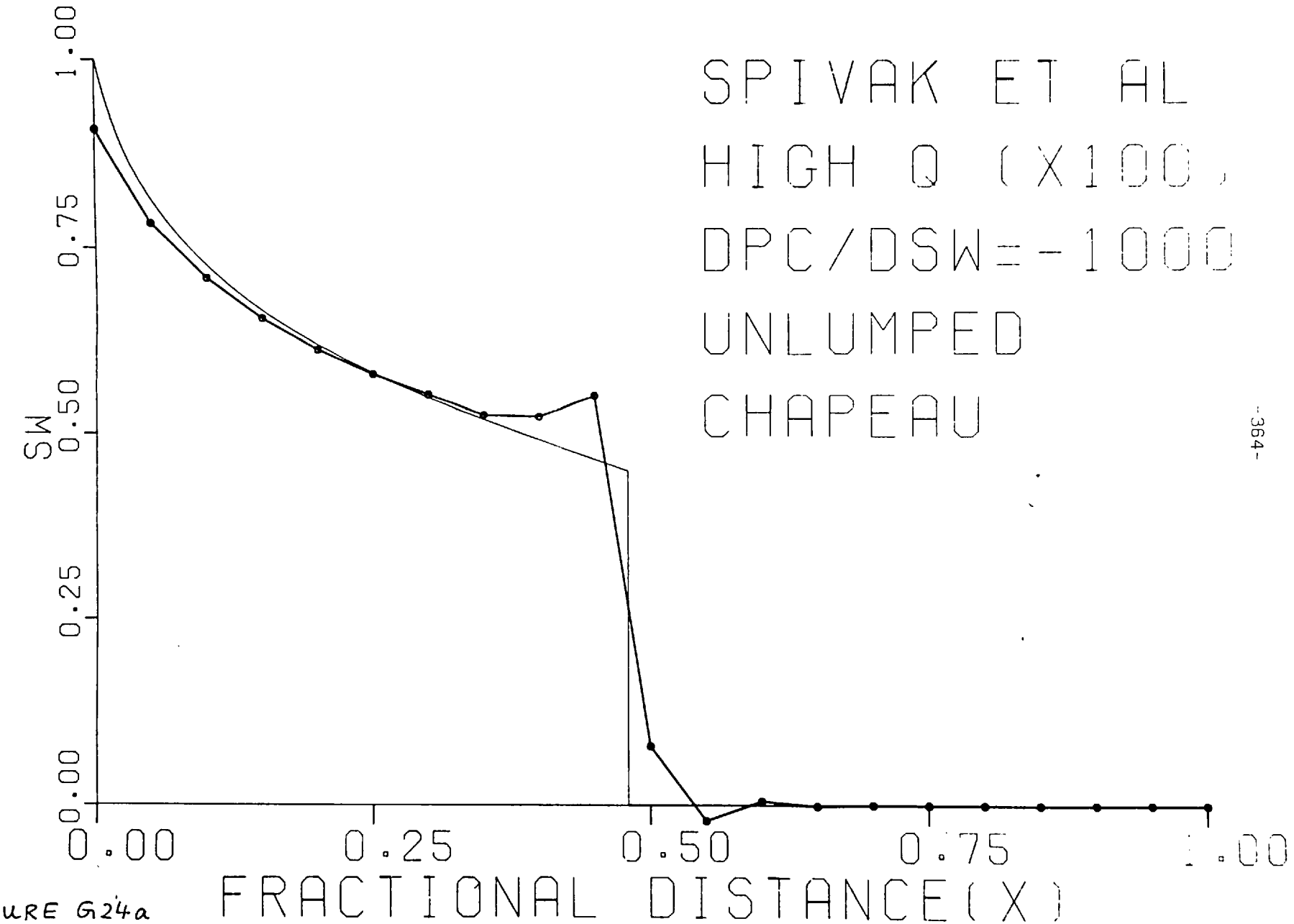

SPIVAK ET AL
HIGH Q (X100)
DPC/DSW=-1000
UNLUMPED
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G24a

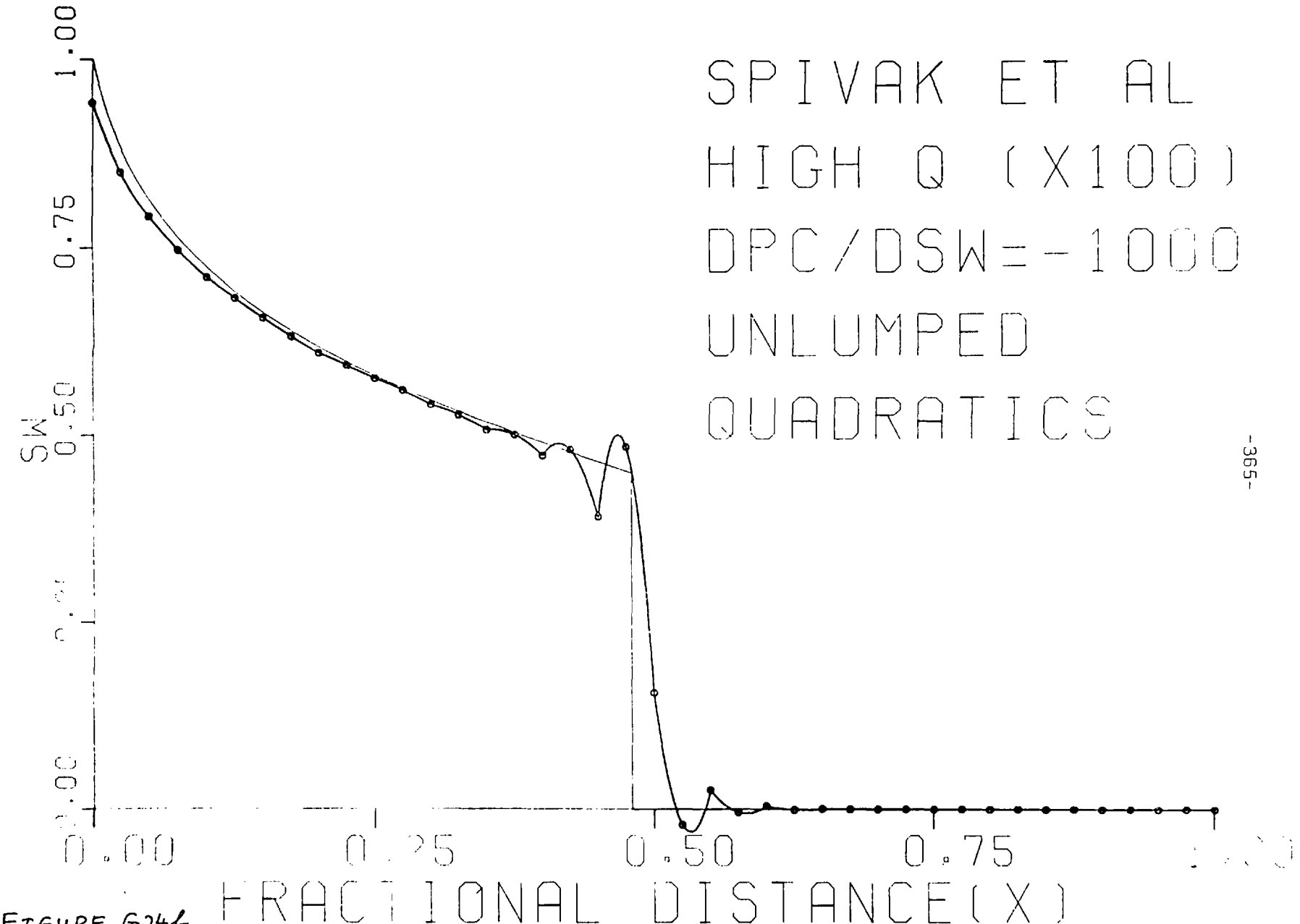

SPIVAK ET AL
HIGH Q (X100)
DPC/DSW=-1000
UNLUMPED
QUADRATICS
SW
1.00
0.75
0.50
0.00
0.00
0.50
0.75
FRACTIONAL DISTANCE(X)

FIGURE G24b

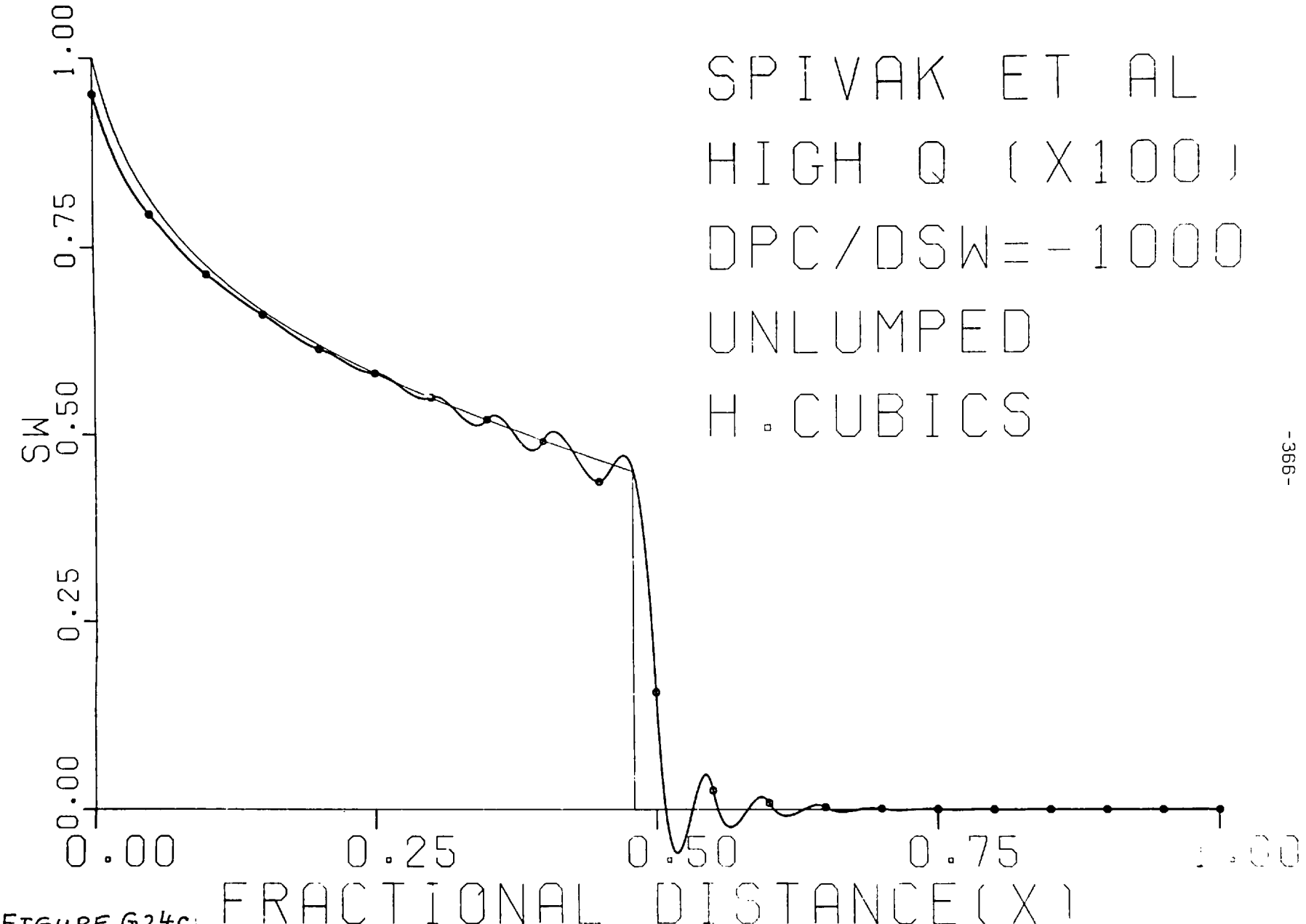

SPIVAK ET AL
HIGH Q (X100)
DPC/DSW=-1000
UNLUMPED
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G24c

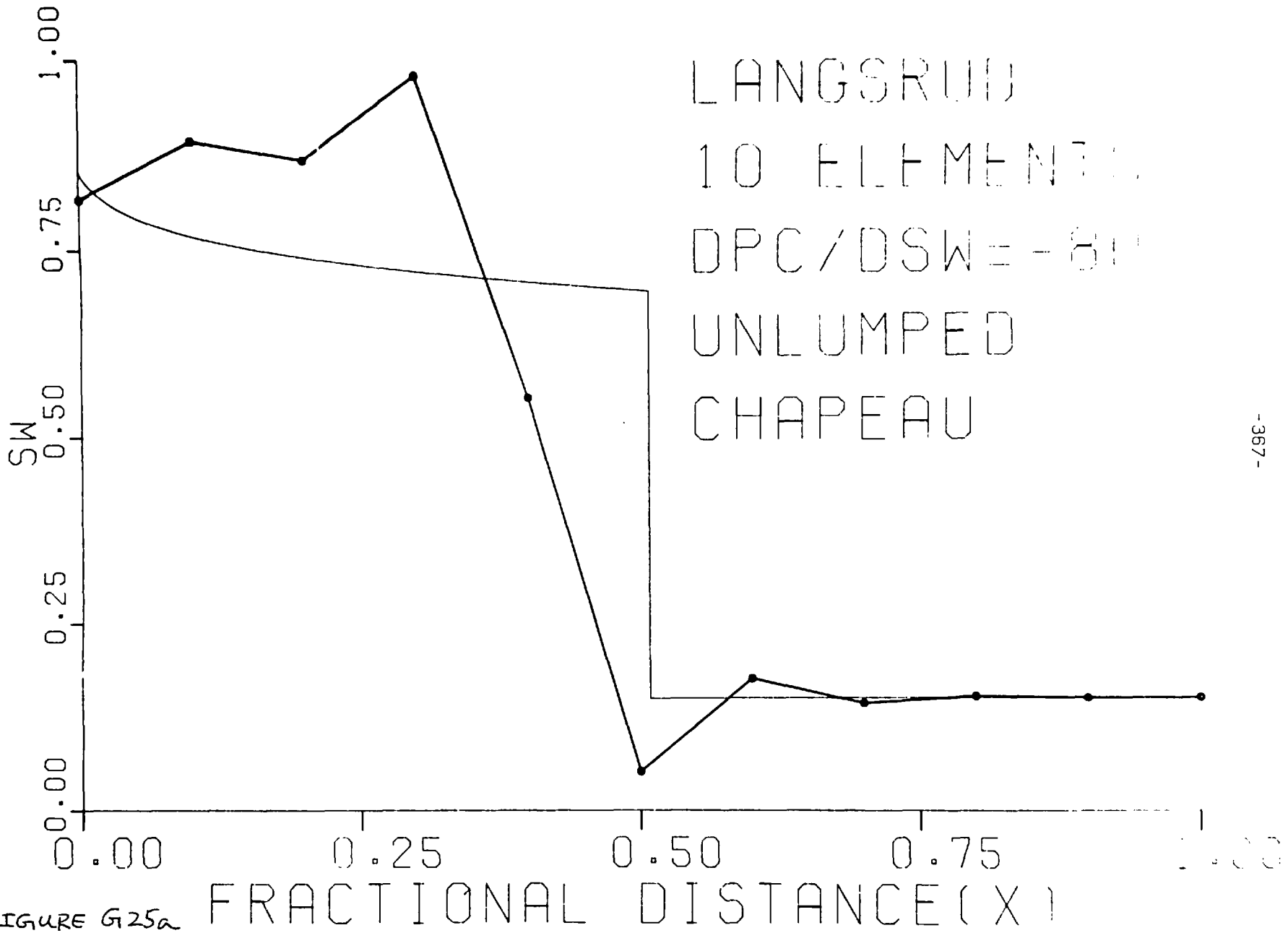

LANGSRUD
10 ELEMENT
DPC/DSW=-6
UNLUMPED
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
FRACTIONAL DISTANCE(X)

FIGURE G25a

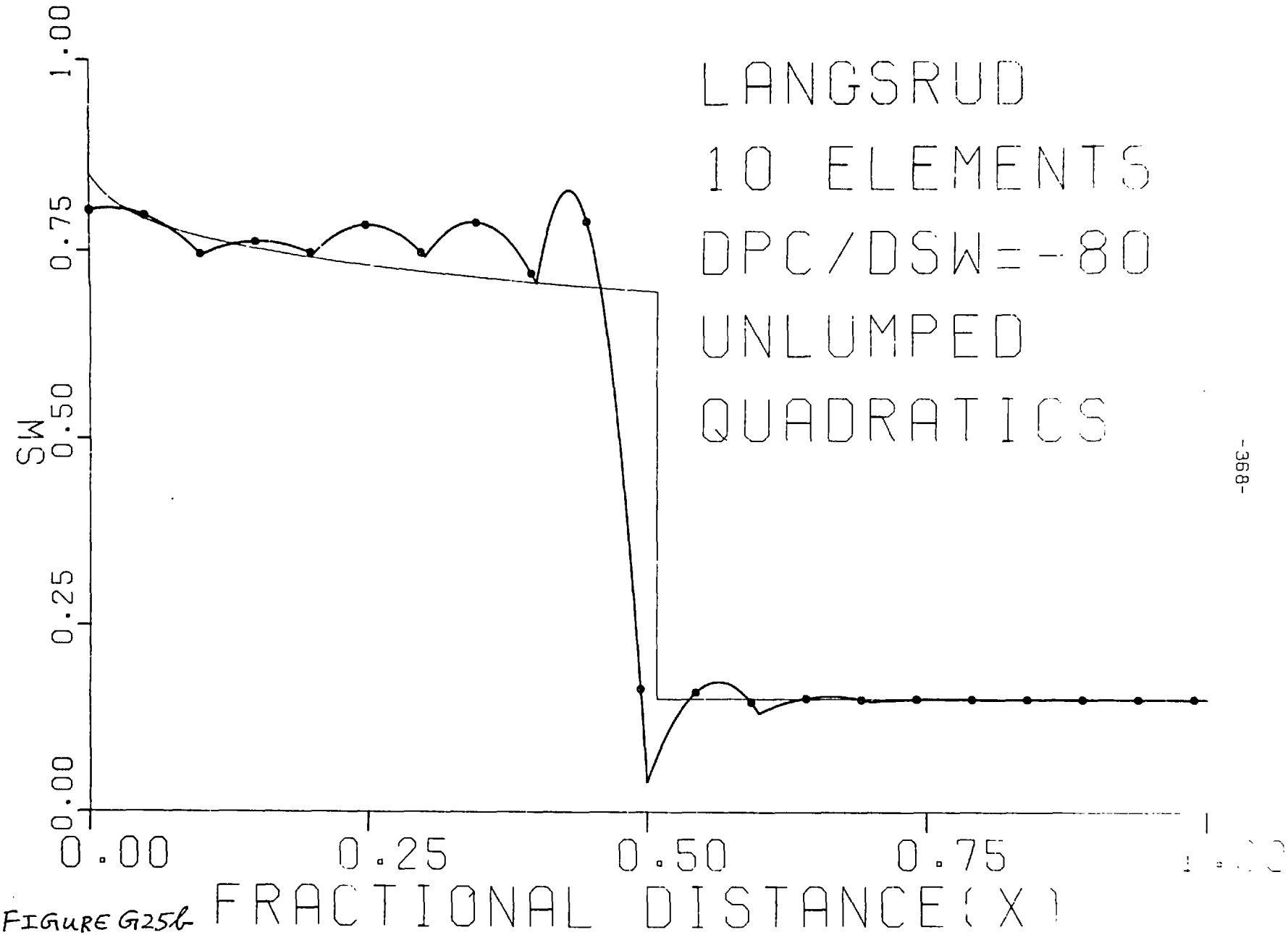

LANGSRUD
10 ELEMENTS
DPC/DSW=-80
UNLUMPED
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G25b

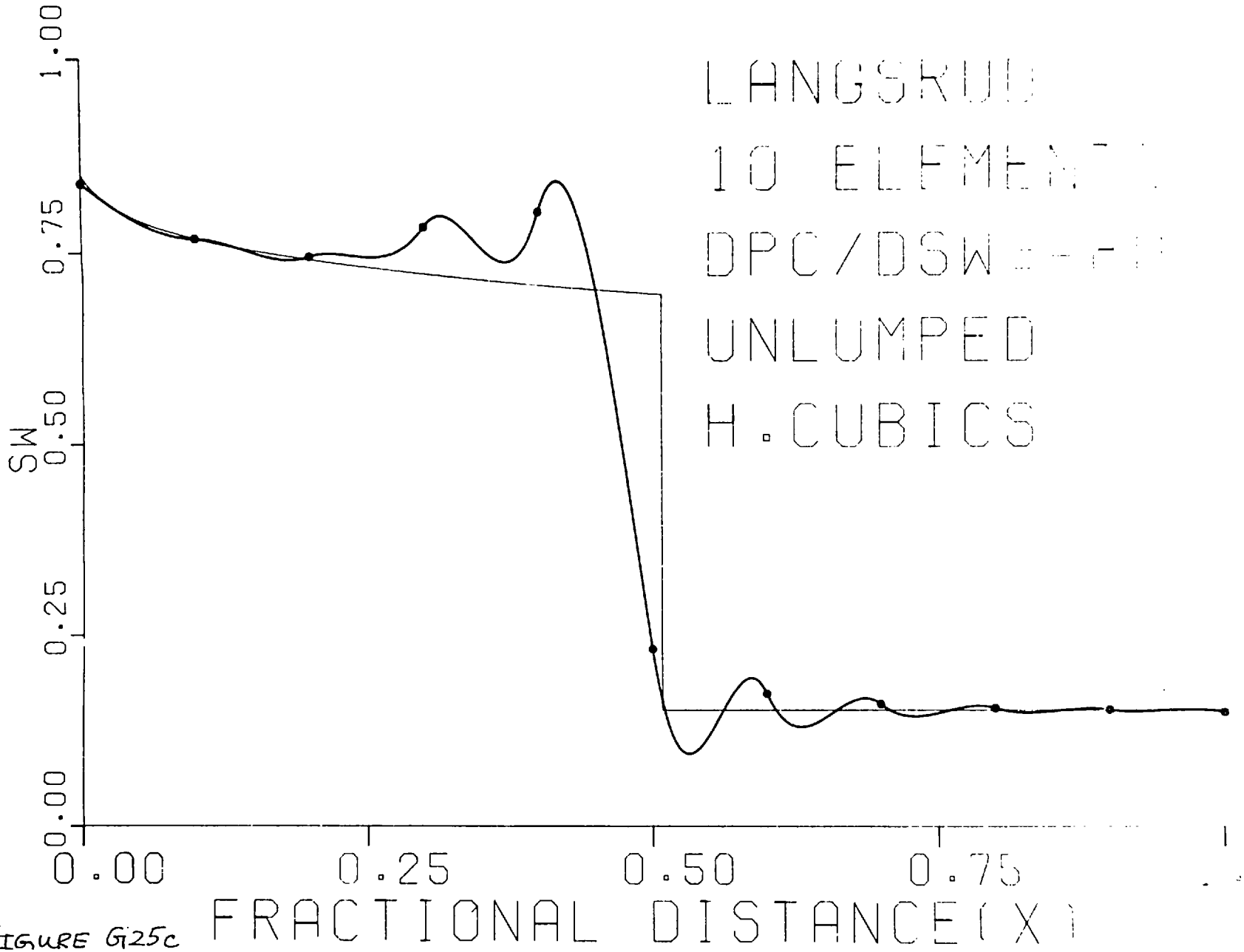

LANGSRUD
10 ELEMENT
DPC/DSW=-F
UNLUMPED
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1
FRACTIONAL DISTANCE(X)

FIGURE G25c

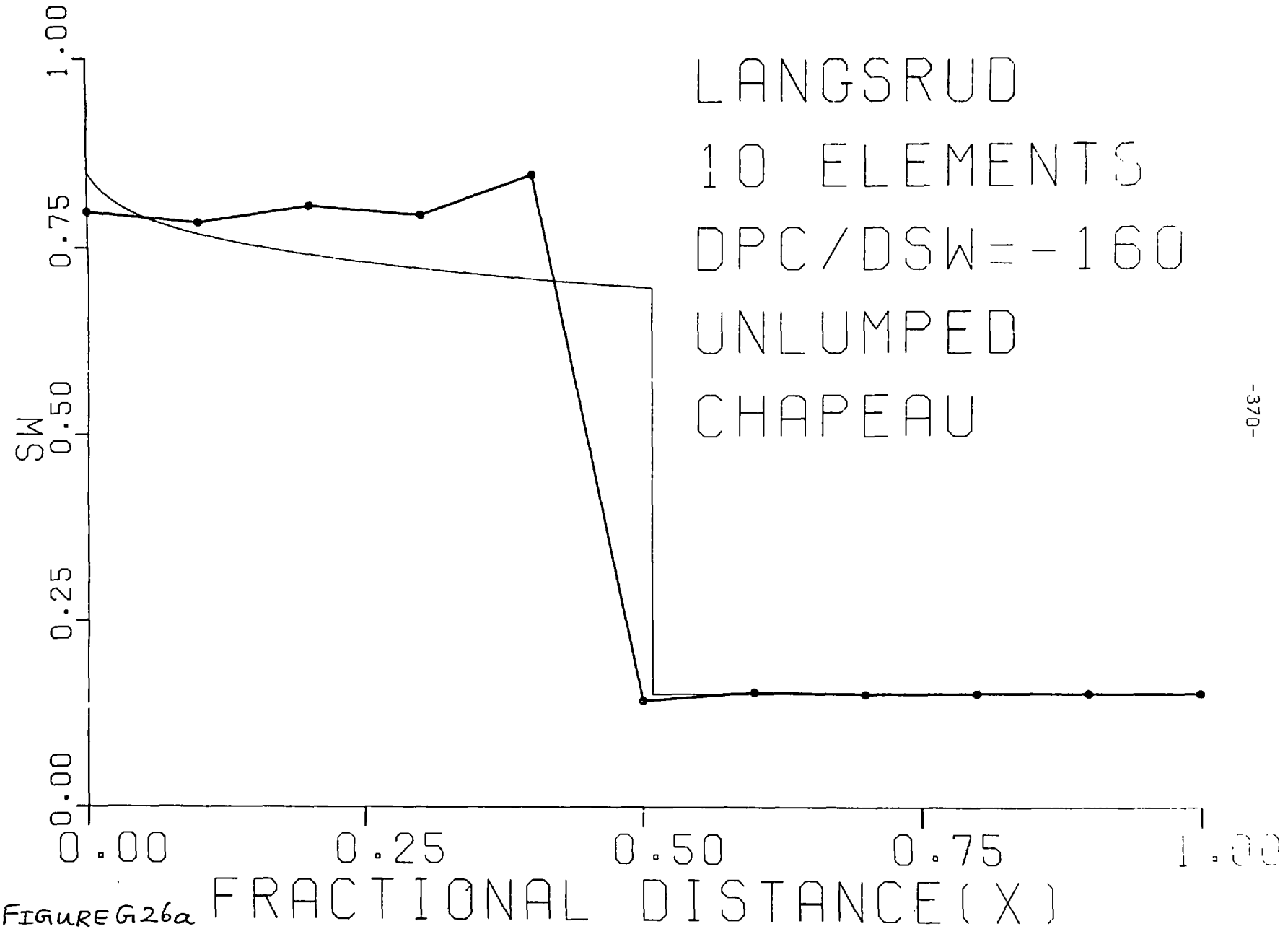

LANGSRUD
10 ELEMENTS
DPC/DSW=-160
UNLUMPED
CHAPEAU
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G26a

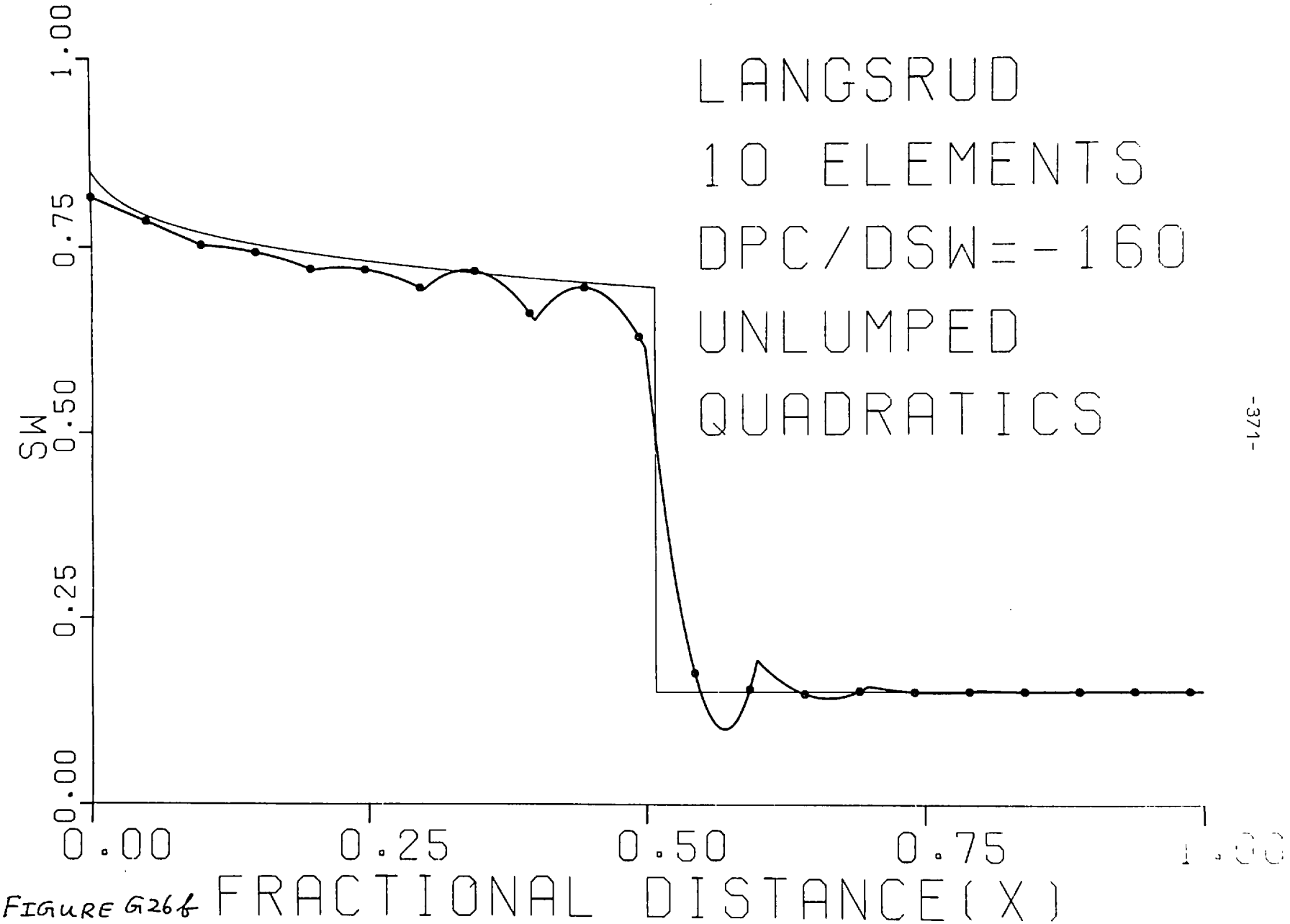

LANGSRUD
10 ELEMENTS
DPC/DSW=-160
UNLUMPED
QUADRATICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G26b

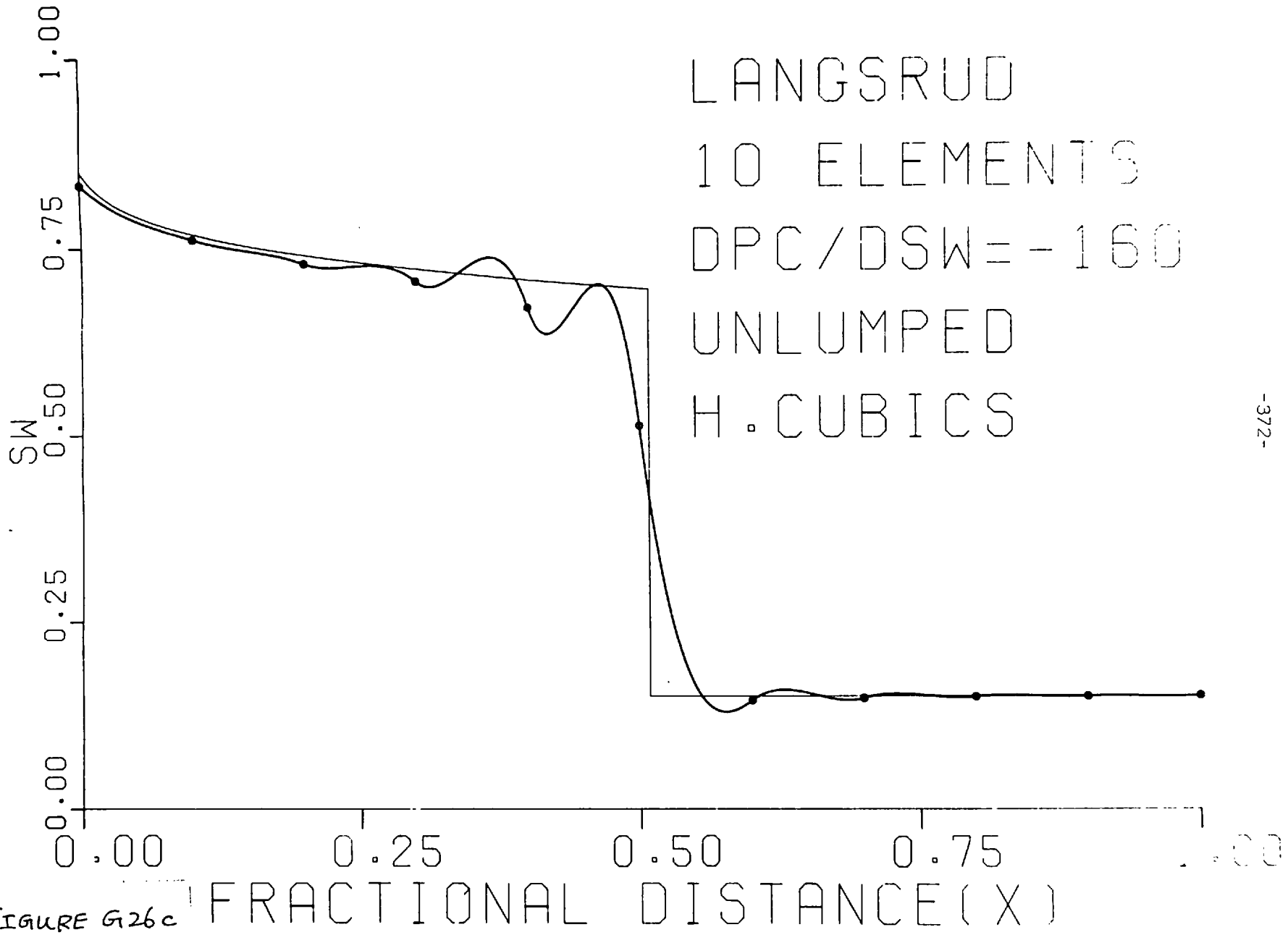

LANGSRUD
10 ELEMENTS
DPC/DSW=-160
UNLUMPED
H.CUBICS
SW
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
FRACTIONAL DISTANCE(X)

FIGURE G26c

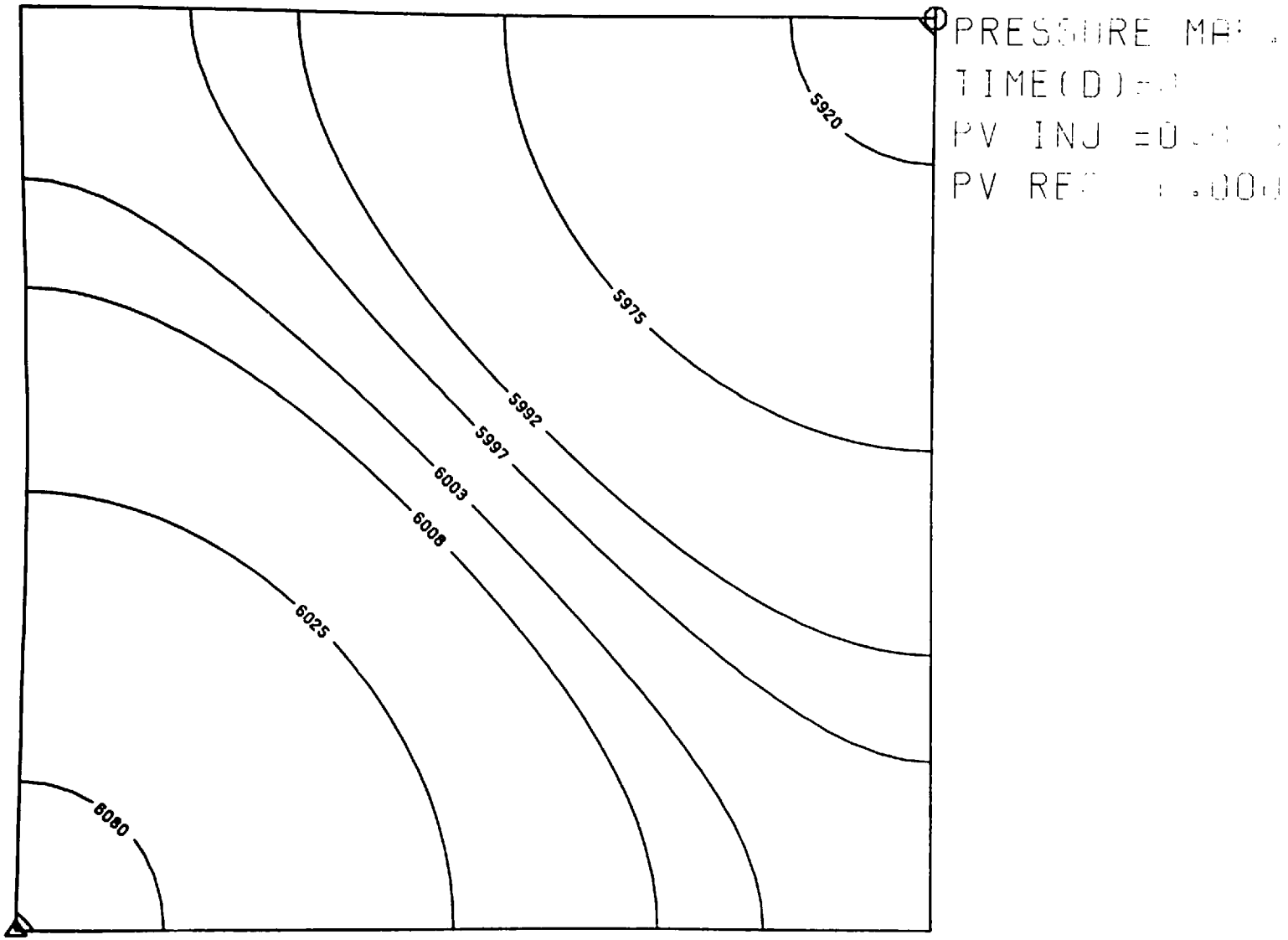


FIGURE G27. MUSKAT SOLUTION (STEADY STATE).

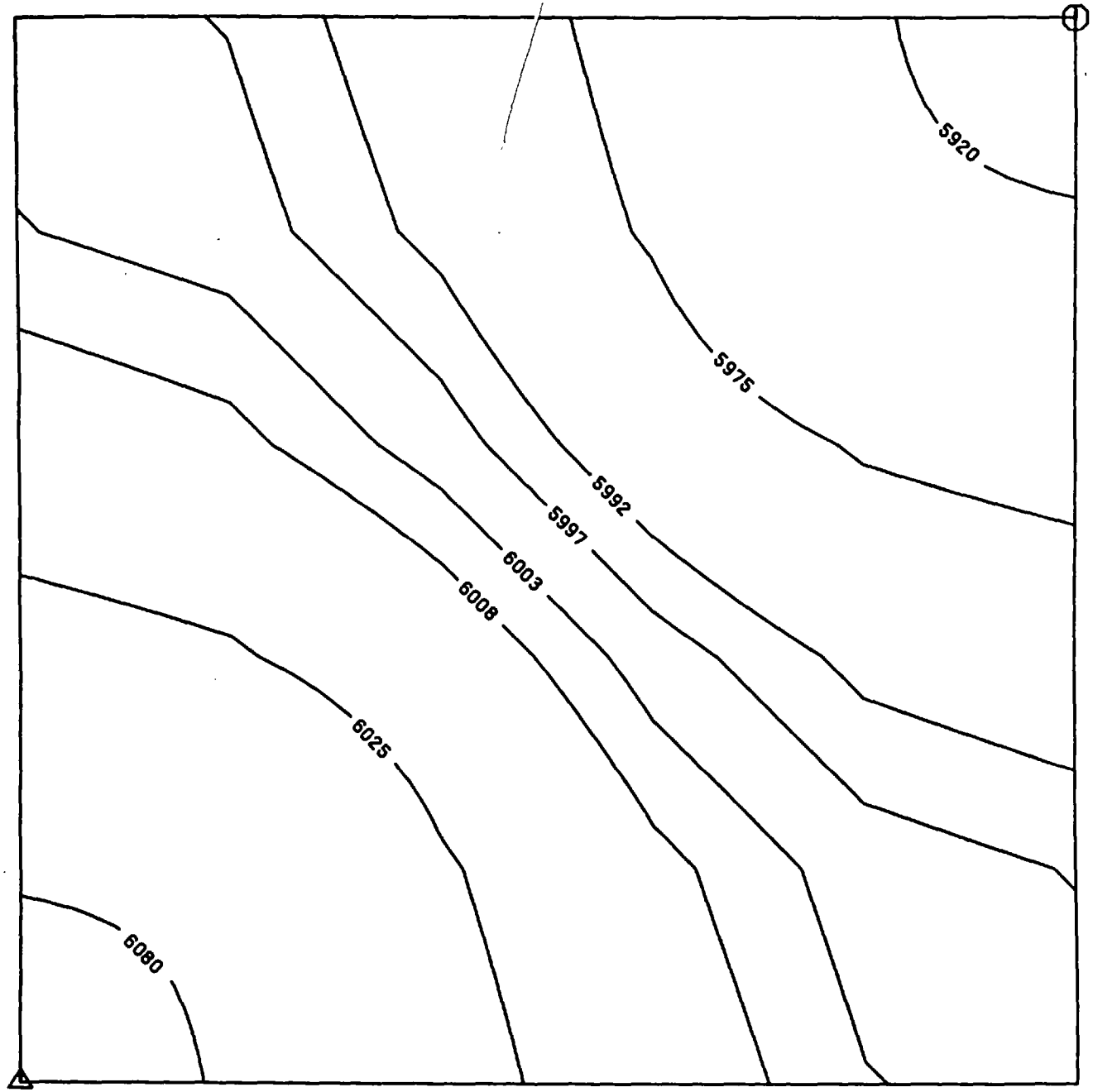


FIGURE G28a CHAPEAU (5X5 ELEMENTS).

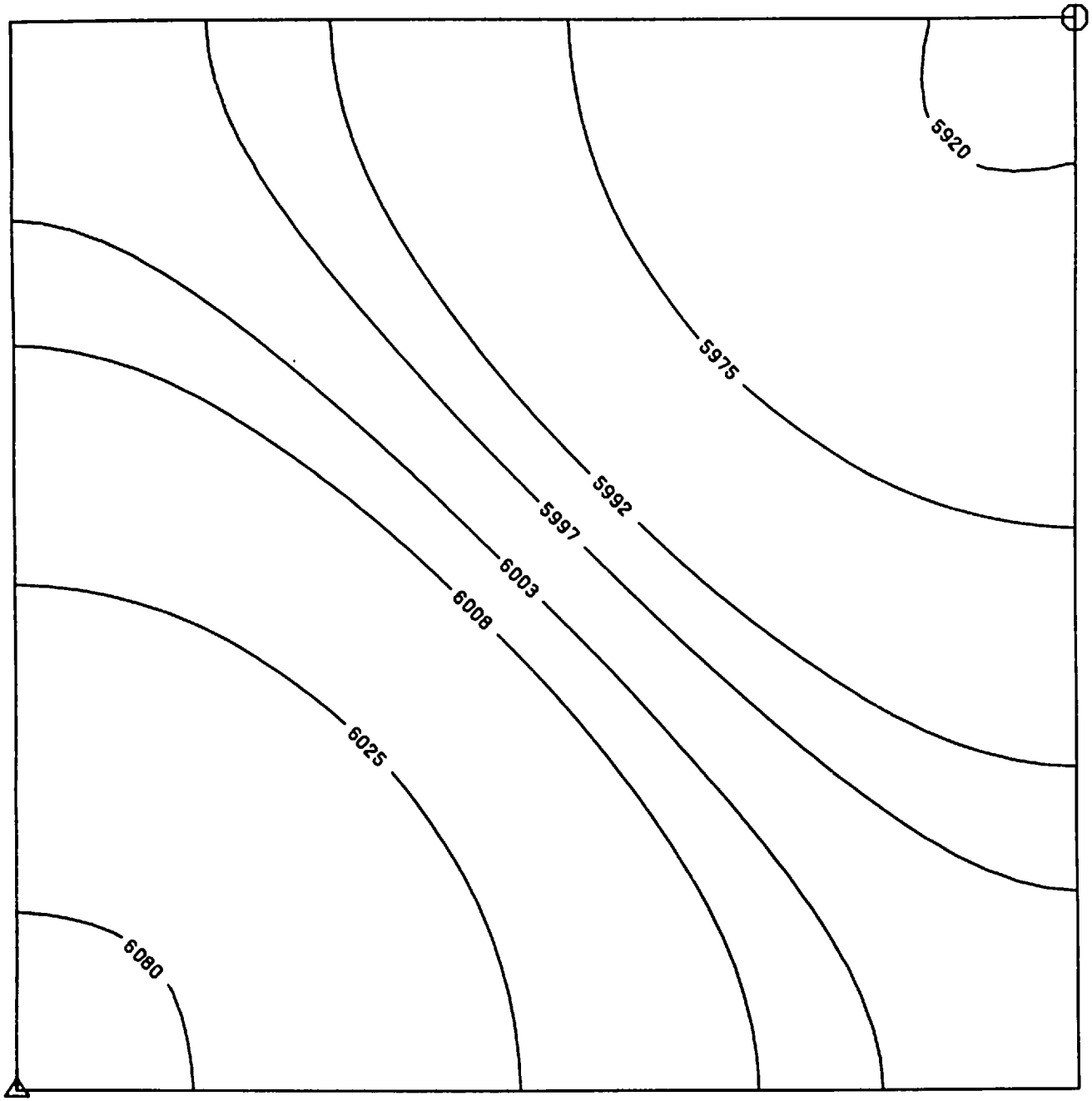
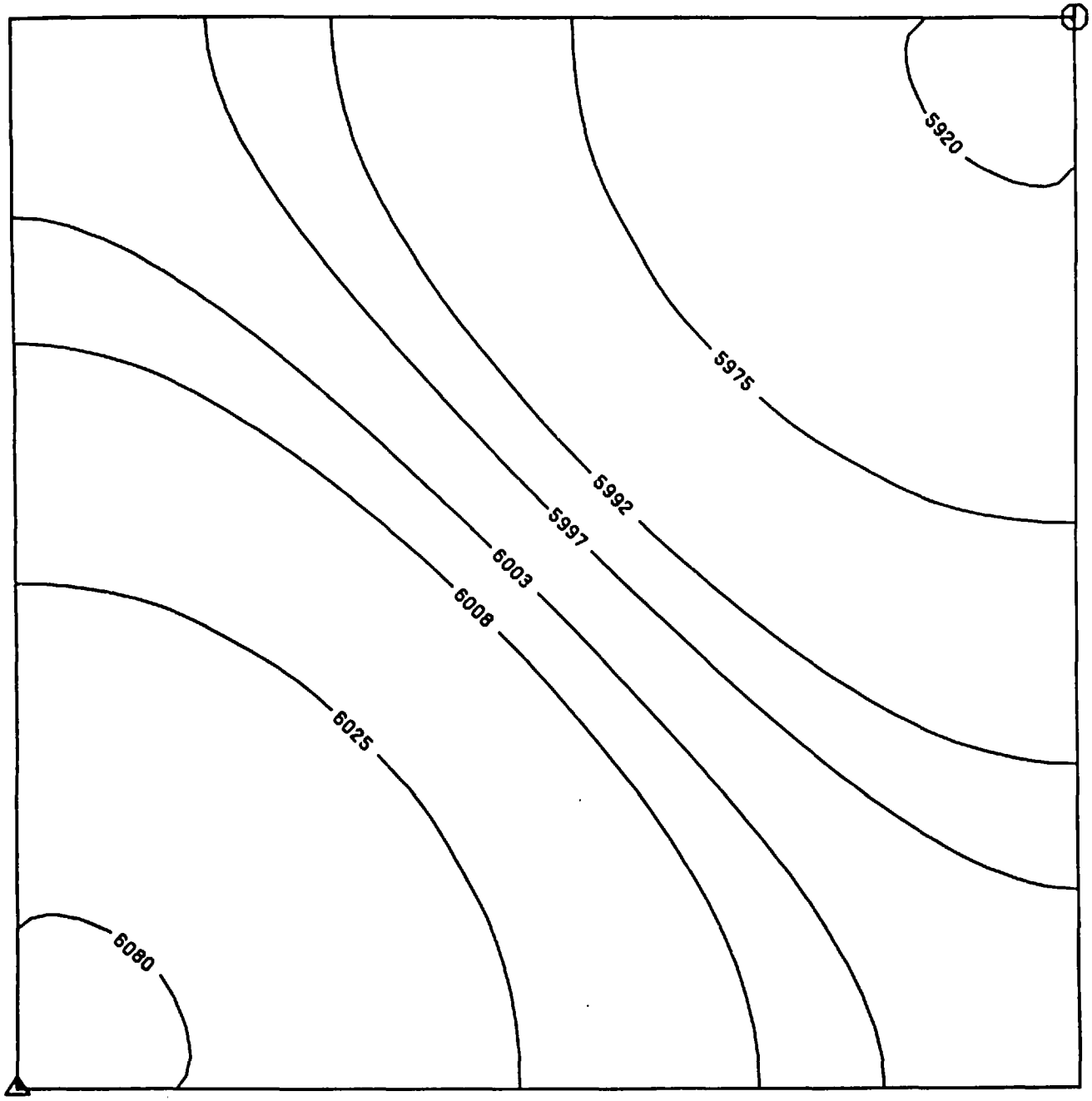


FIGURE G28b QUADRATICS (5X5 ELEMENTS)



H. CUBICS (5×5 ELEMENTS).

FIGURE G28c

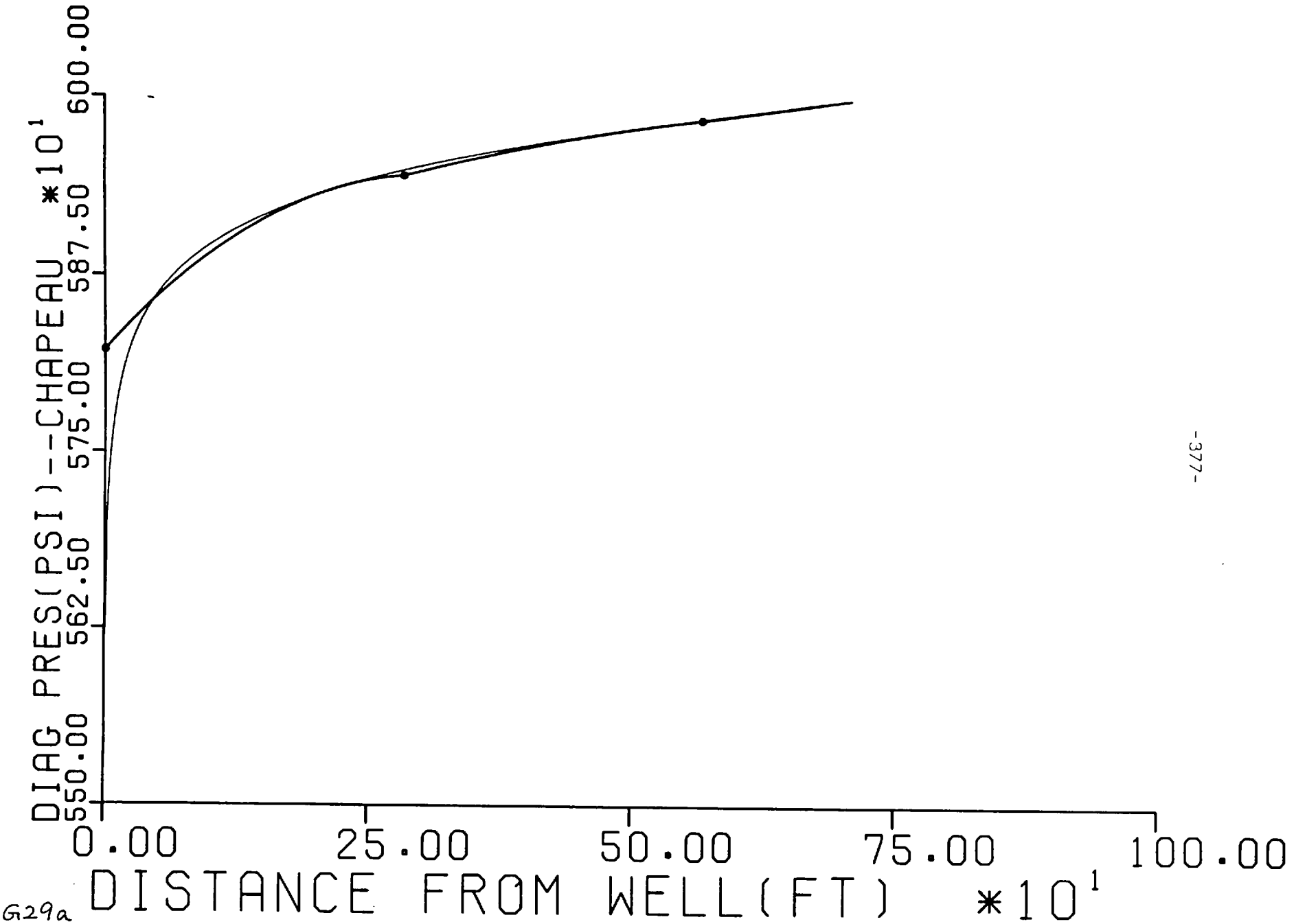

DIAG PRES(PSI)--CHAPEAU *10¹
600.00
587.50
575.00
562.50
550.00
0.00
25.00
50.00
75.00
100.00
DISTANCE FROM WELL(FT) *10¹

FIGURE G29a

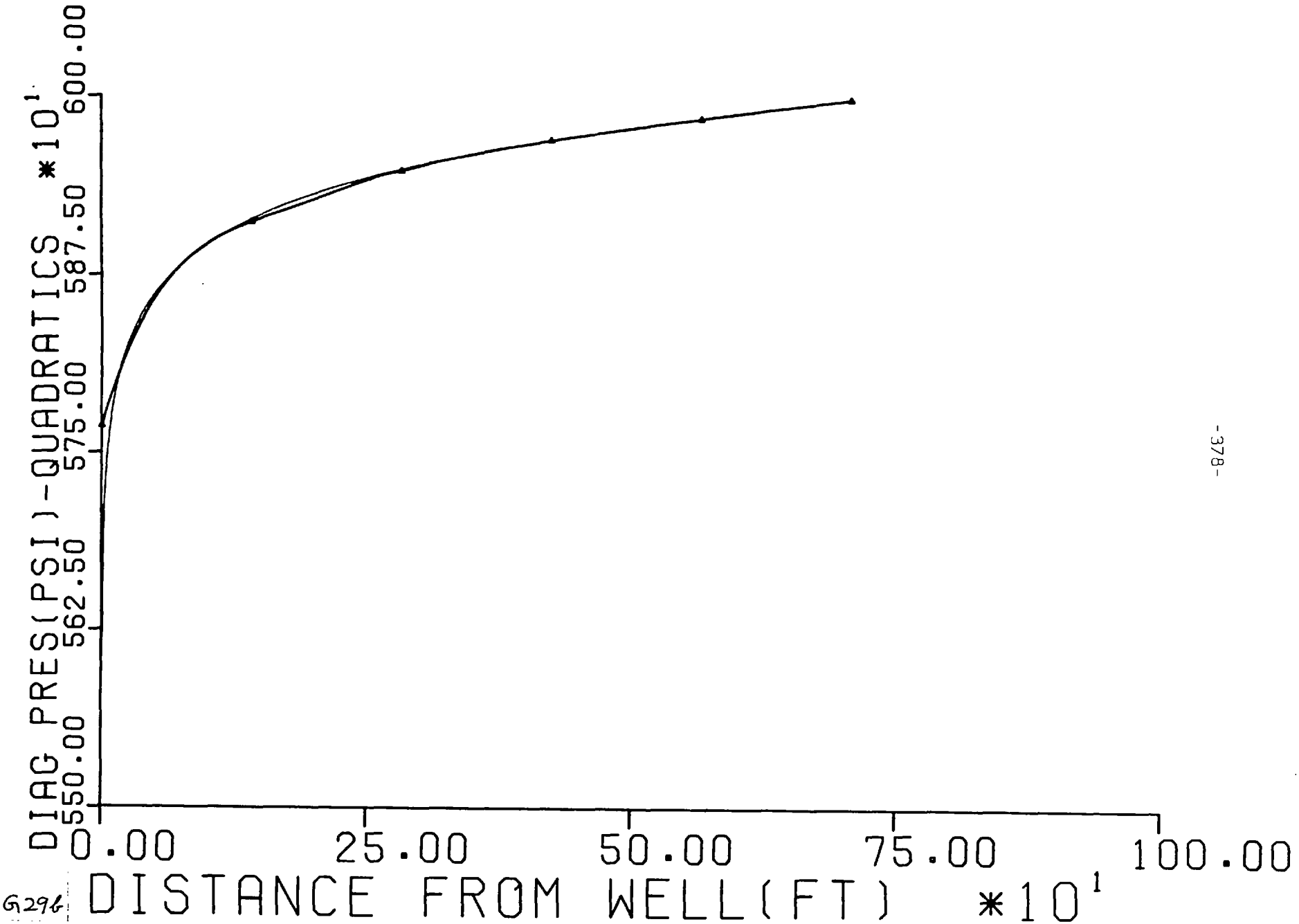

DIAG PRES(PSI)-QUADRATICS *10¹
600.00
587.50
575.00
562.50
550.00
0.00
25.00
50.00
75.00
100.00
DISTANCE FROM WELL(FT) *10¹

FIGURE G296

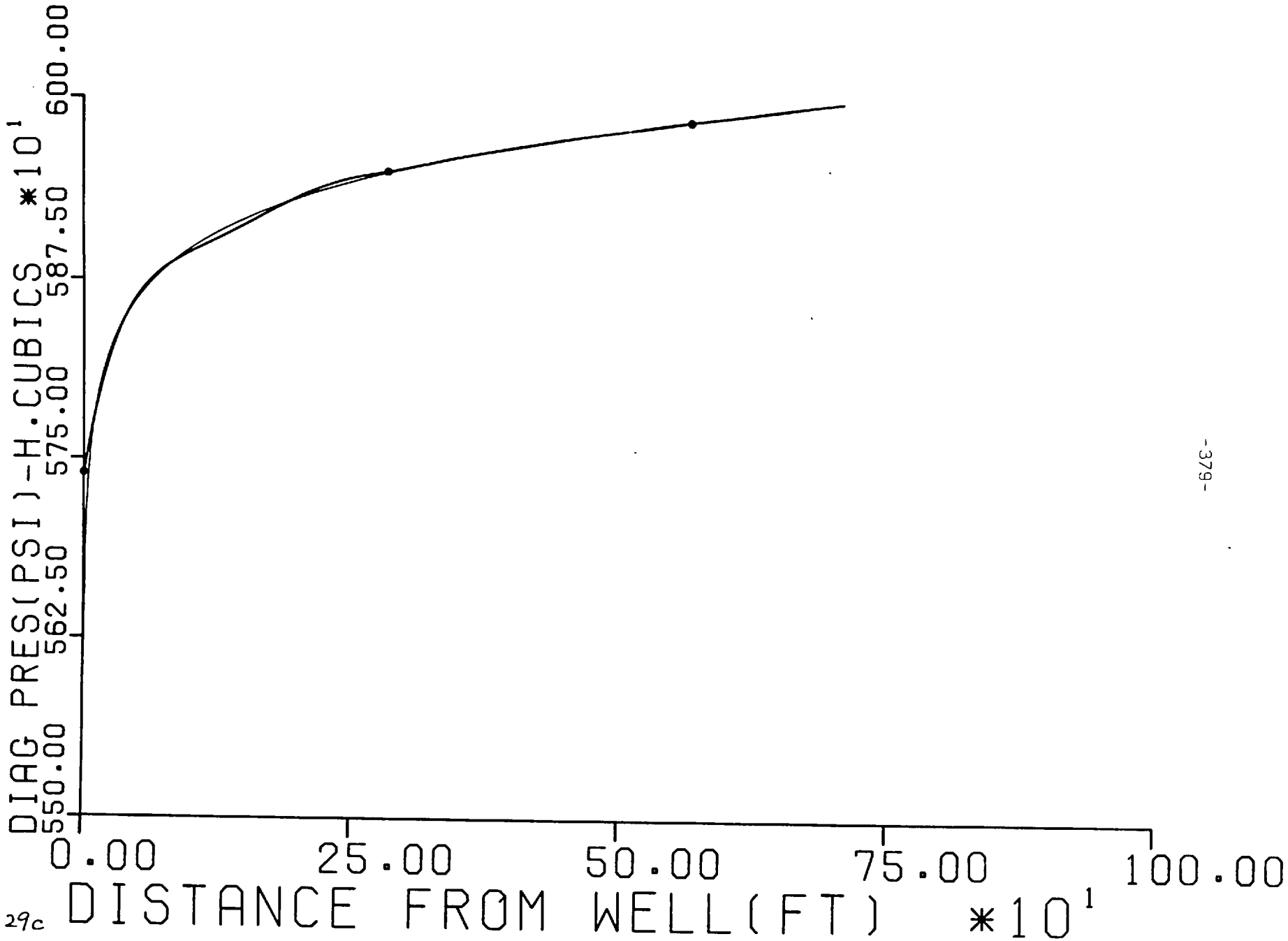

DIAG PRES(PSI)-H.CUBICS *10¹
600.00
587.50
575.00
562.50
550.00
0.00
25.00
50.00
75.00
100.00
DISTANCE FROM WELL(FT) *10¹

FIGURE G29c

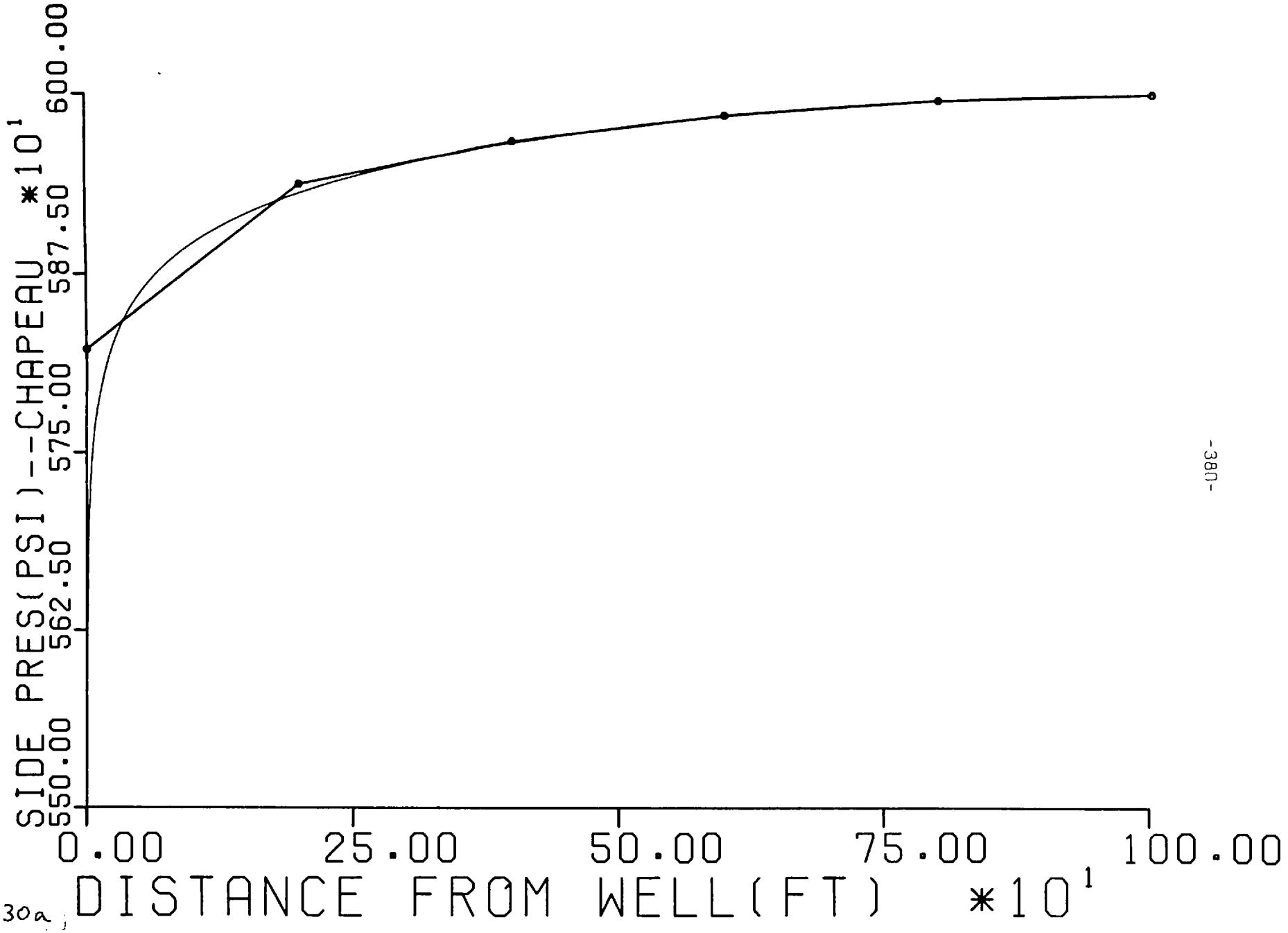

SIDE PRES(PSI)--CHAPEAU *10¹
600.00
587.50
575.00
562.50
550.00
0.00
25.00
50.00
75.00
100.00
DISTANCE FROM WELL(FT) *10¹

FIGURE G30a

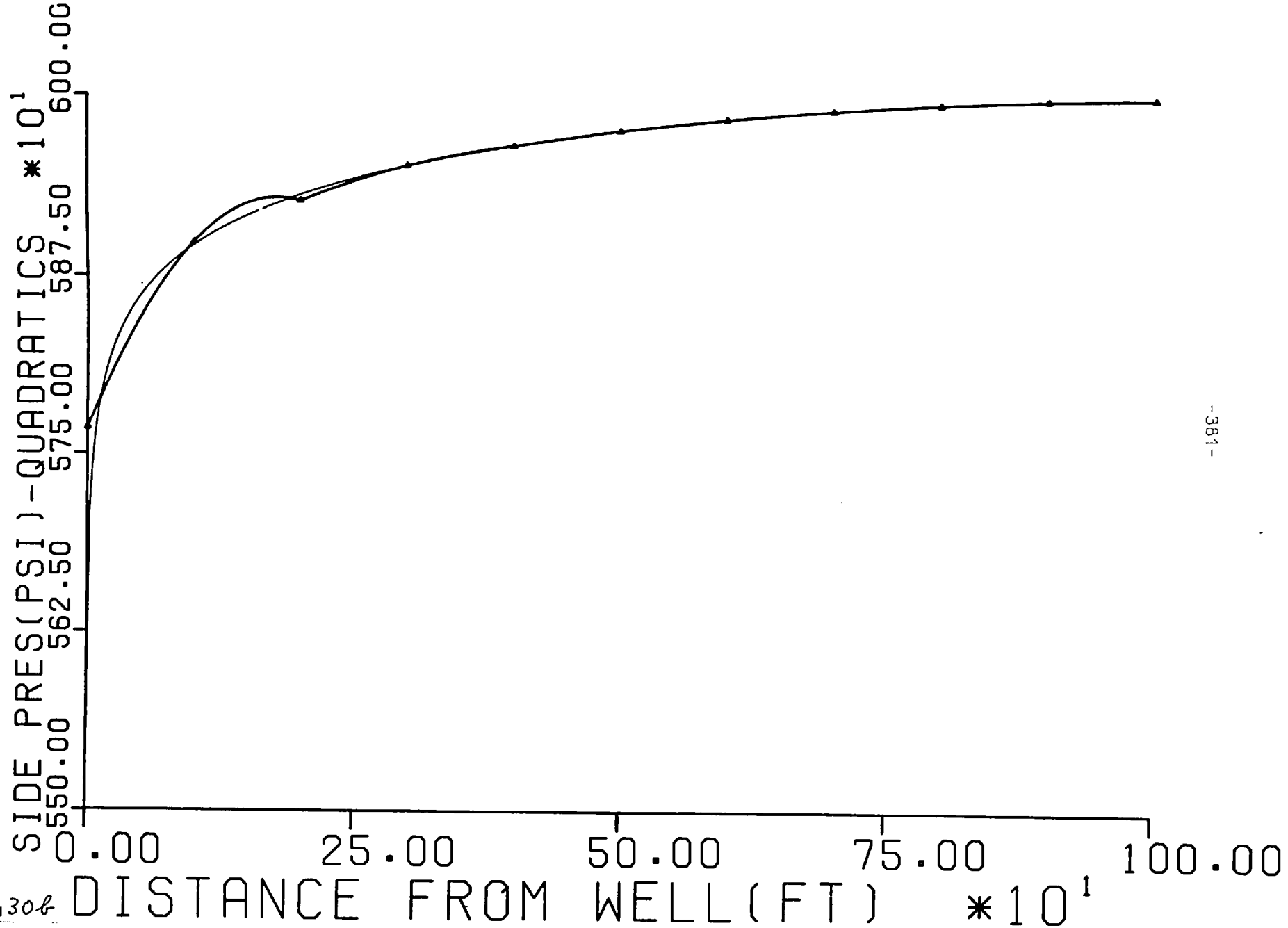

SIDE PRES(PSI)-QUADRATICS *10¹
600.00
587.50
575.00
562.50
550.00
0.00
25.00
50.00
75.00
100.00
DISTANCE FROM WELL(FT) *10¹

FIGURE G30b

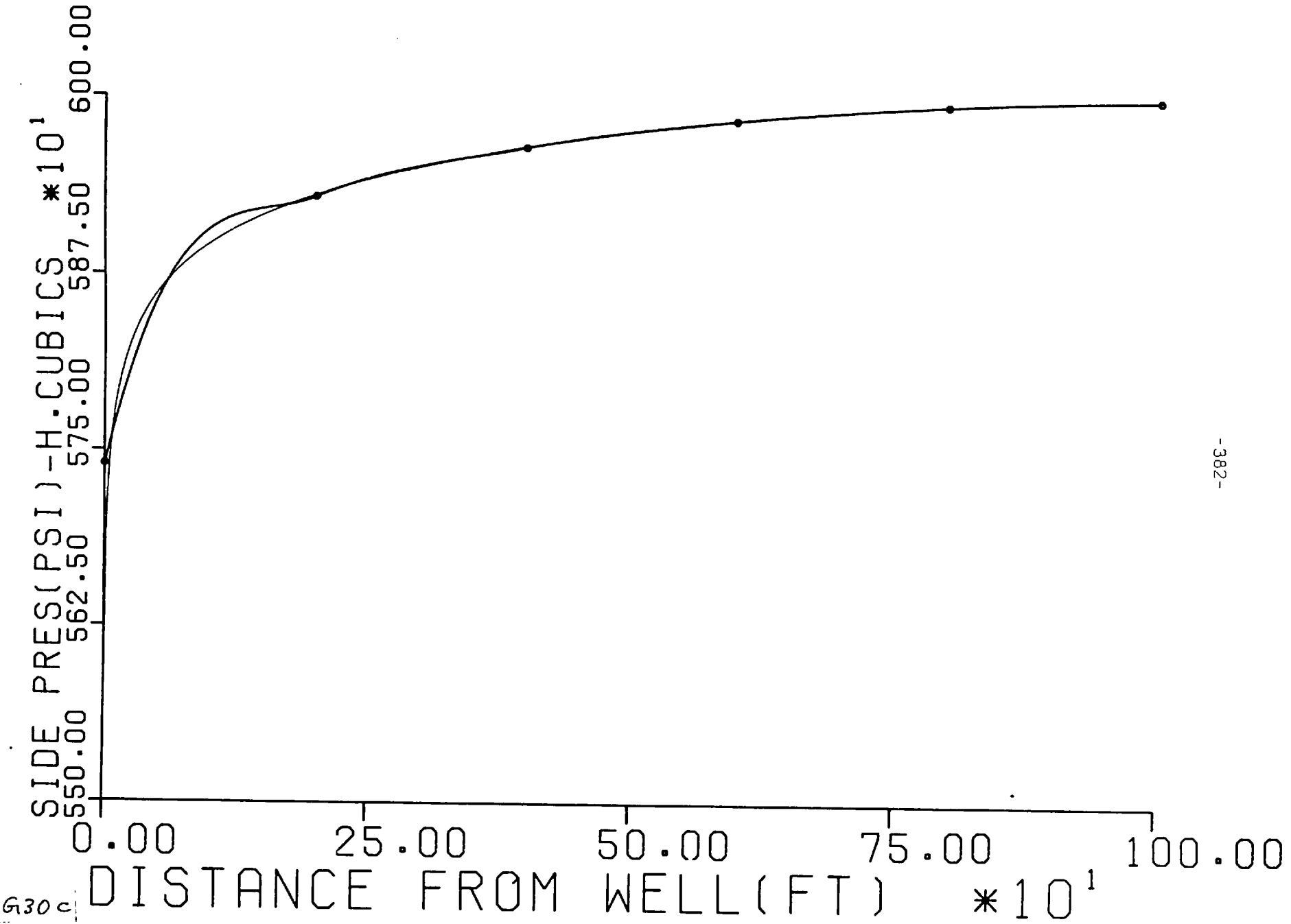

SIDE PRES(PSI)-H.CUBICS *10¹
600.00
587.50
575.00
562.50
550.00
0.00
25.00
50.00
75.00
100.00
DISTANCE FROM WELL(FT) *10¹

FIGURE G30c

2 D CHAPEAU FUNCTIONS.

5X5 ELEMENTS.

SPIVAK ET AL DATA.

DPC/DSW=0.

MODIFIED FW CURVE.

LUMPED CAPACITY MATRIX.

INJ. RATE =0.0002 PV/DAY.

NO UPSTREAM WEIGHTING.

DTMAX = 30 DAYS.

FIGURE G31-1

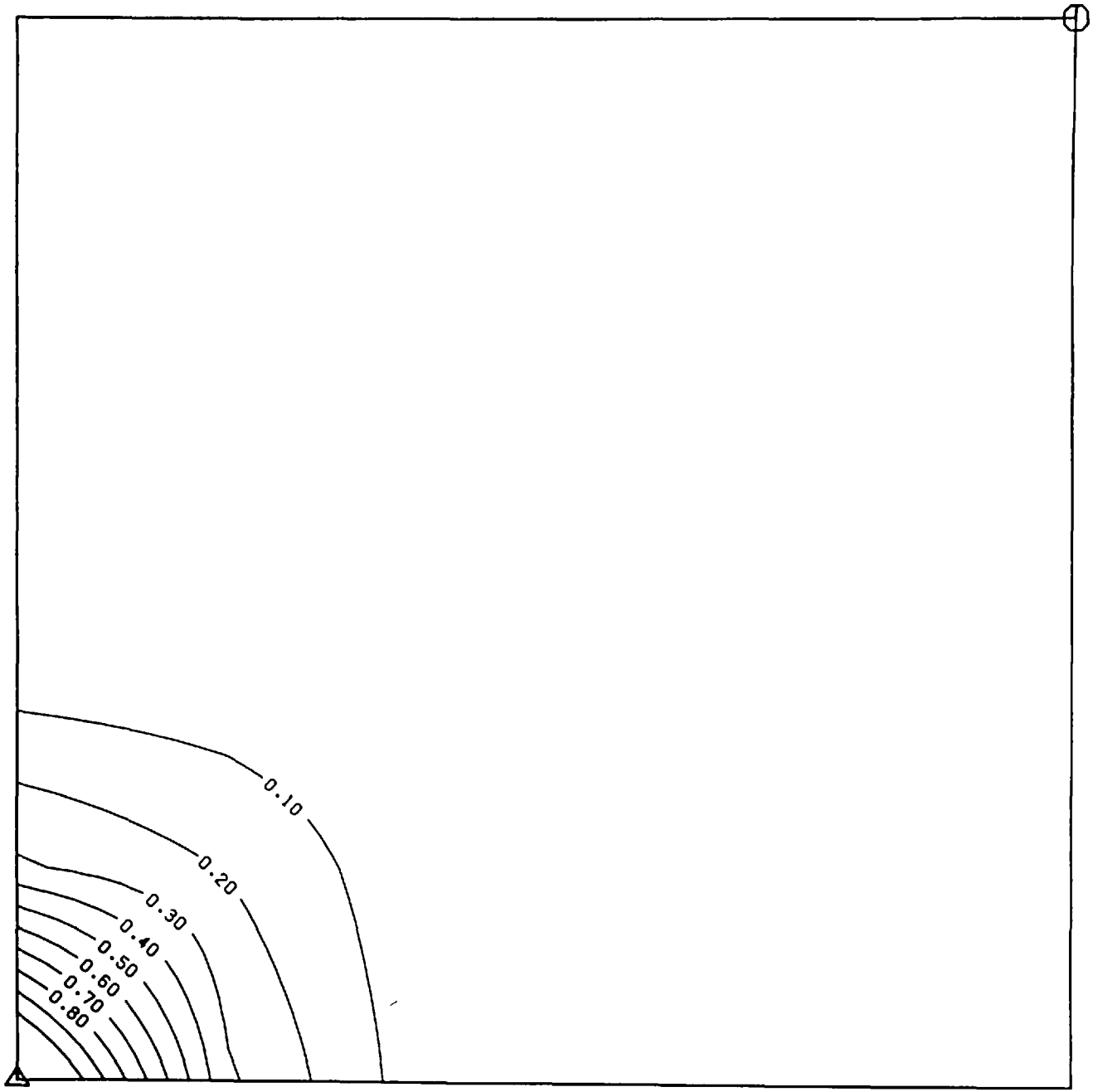

SW MAP.
TIME(D)=180
PV INJ =0.036
PV REC =0.036
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80

FIGURE G31-2

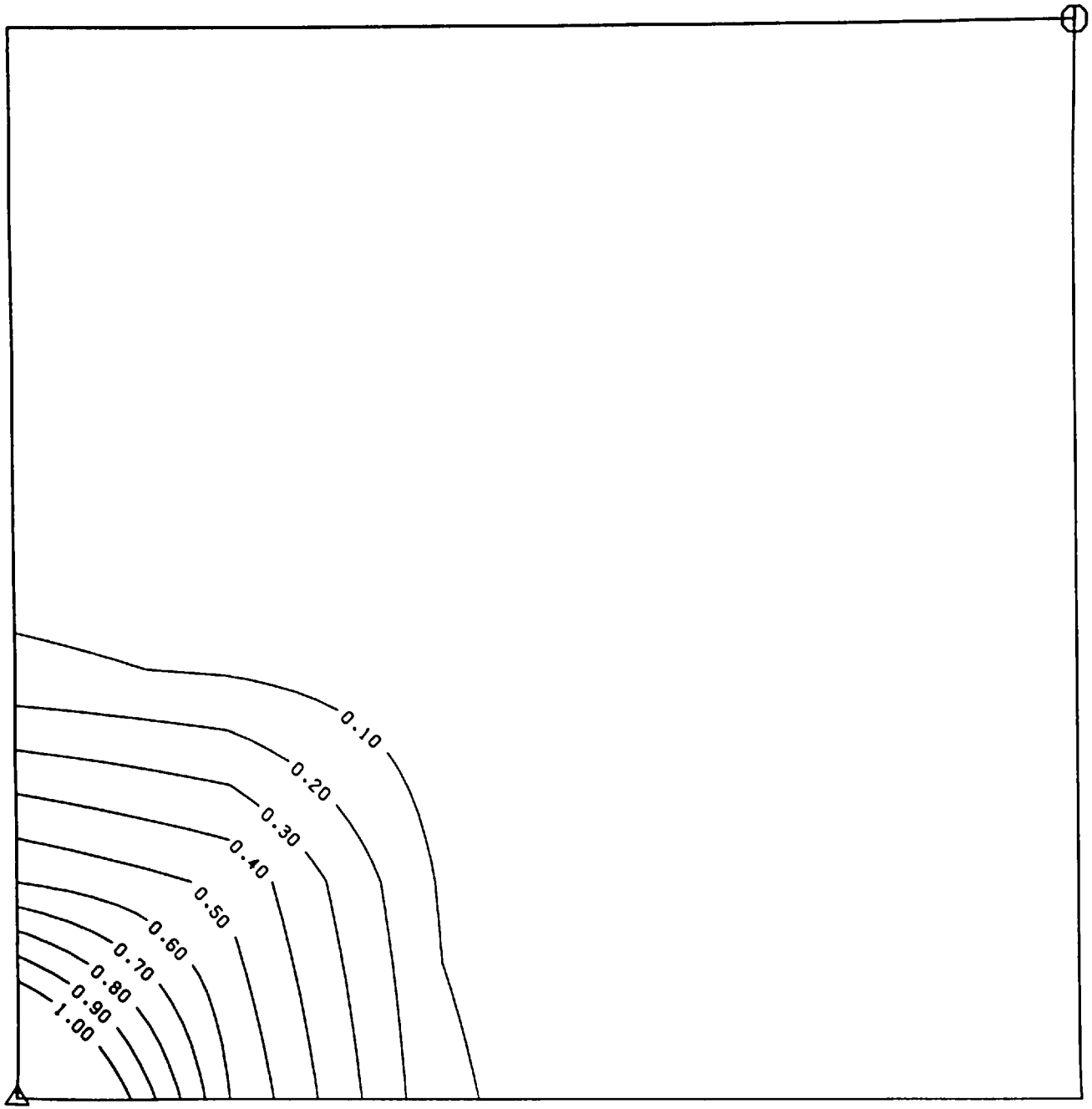

SW MAP.
TIME(D)=360
PV INJ =0.072
PV REC =0.072
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G31-3

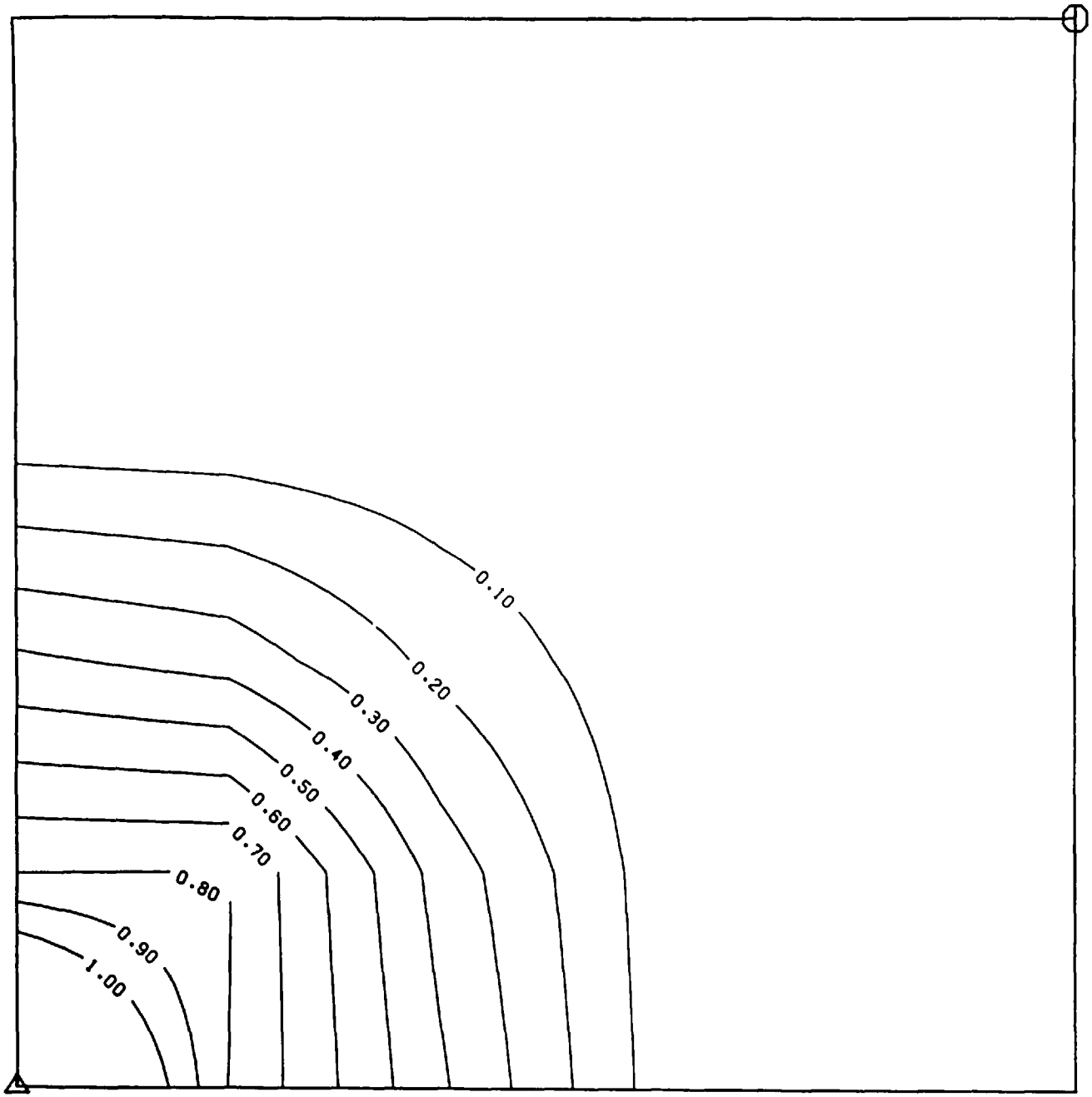

SW MAP.
TIME(D)=720
PV INJ =0.144
PV REC =0.144
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G31-4

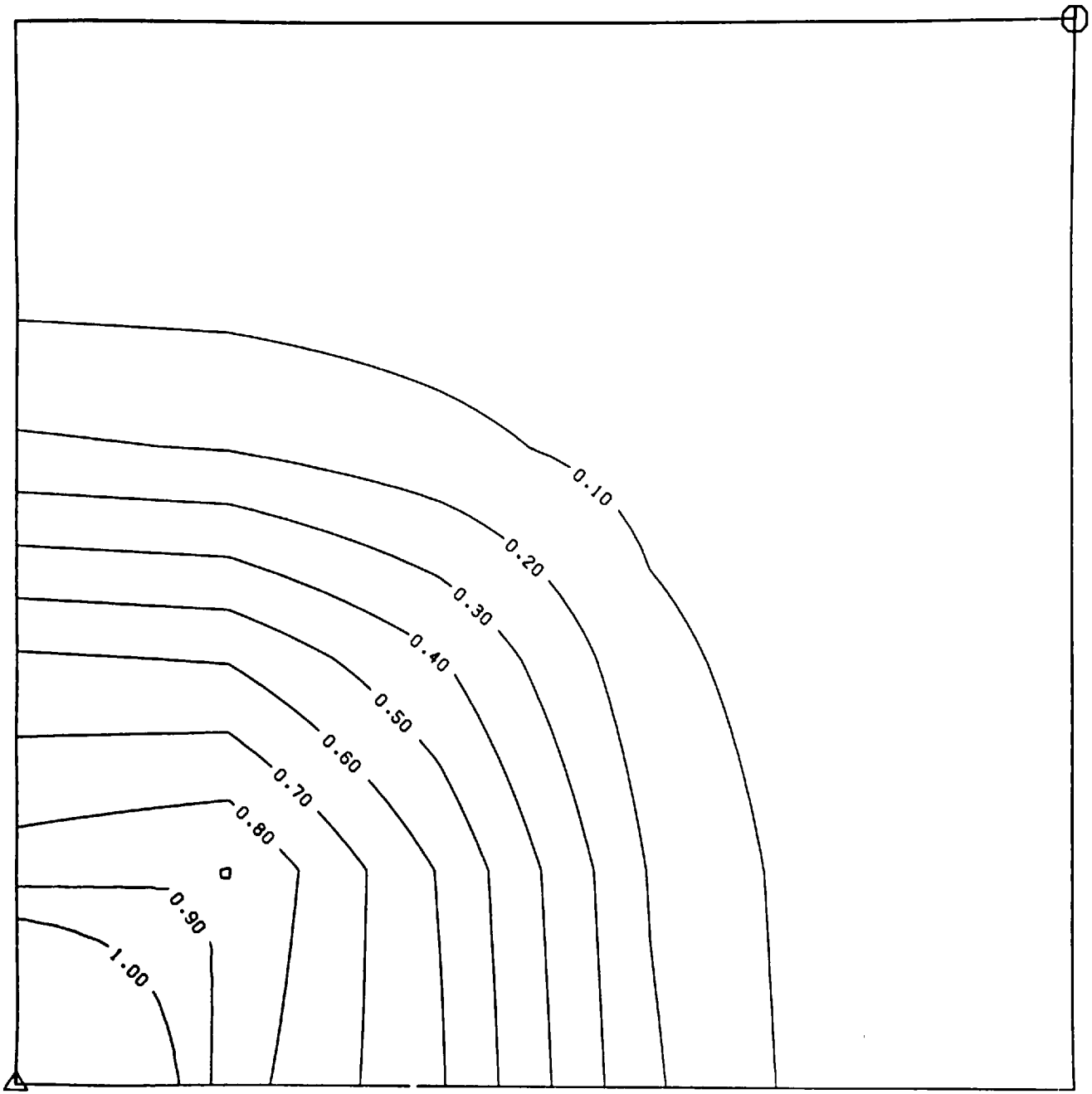

SW MAP.
TIME(D)=1080
PV INJ =0.216
PV REC =0.216
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G31-5

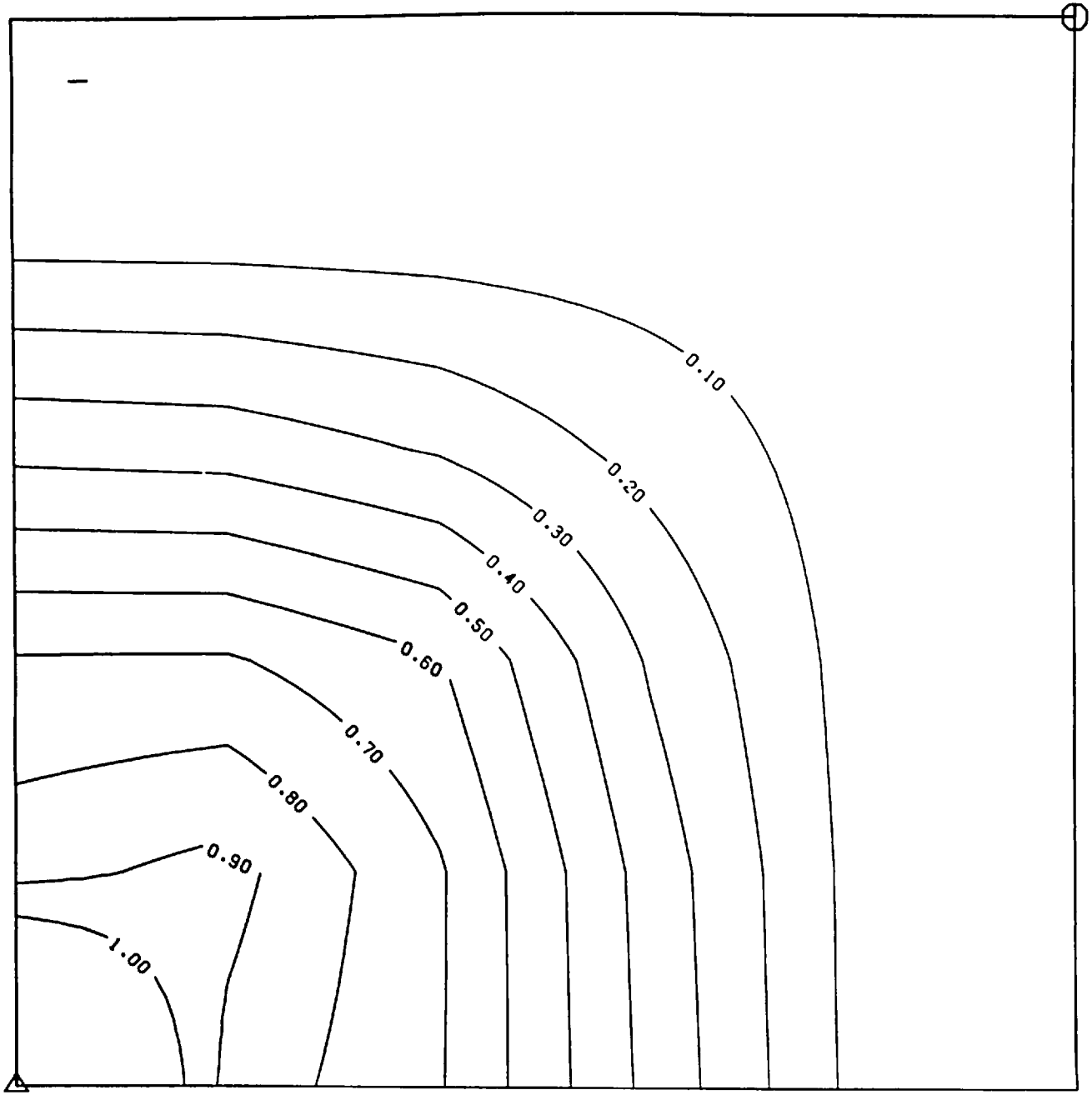

SW MAP.
TIME(D)=1440
PV INJ =0.288
PV REC =0.287
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G31.6

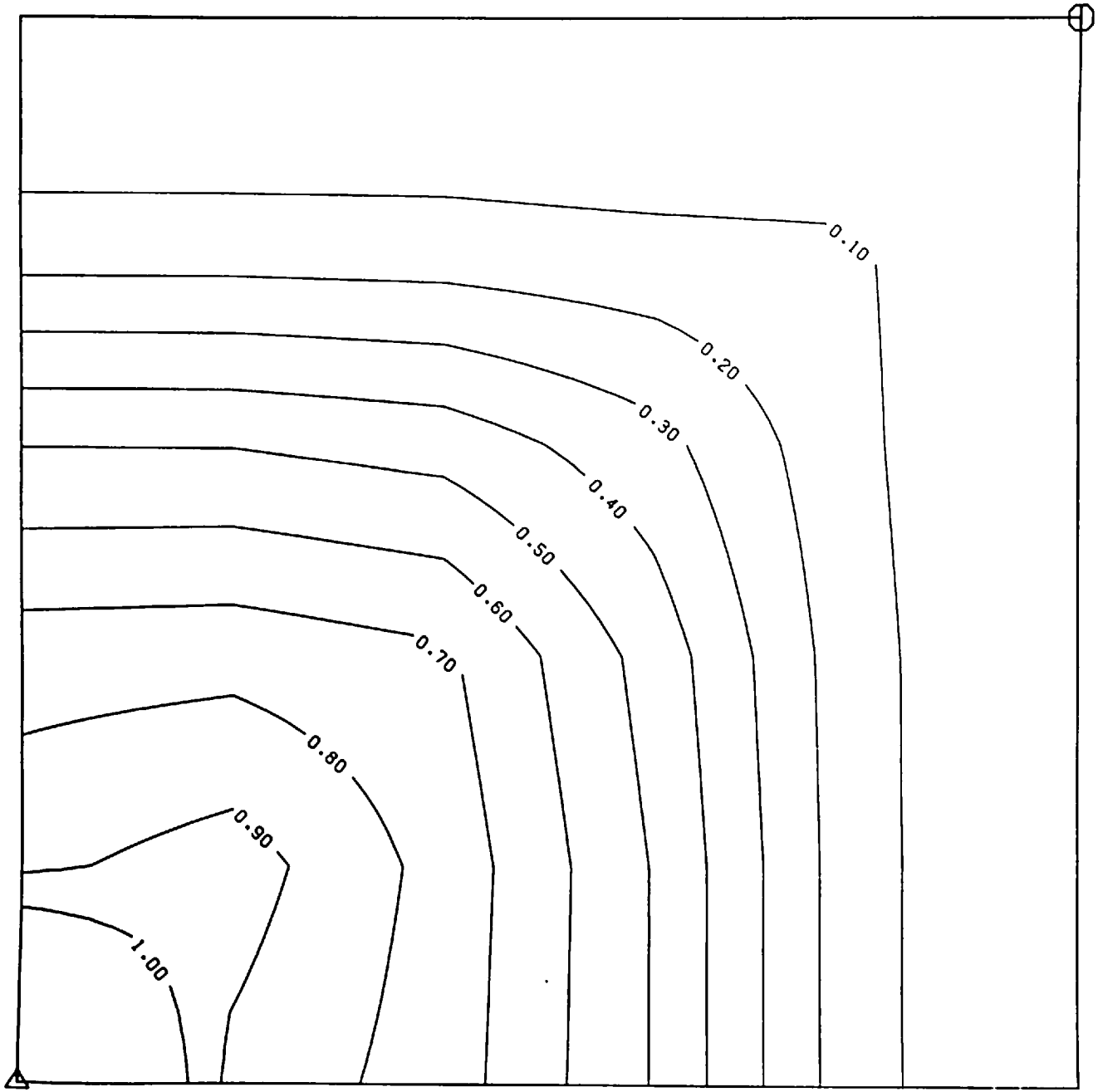

SW MAP.
TIME(D)=1800
PV INJ =0.360
PV REC =0.357
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G31.7

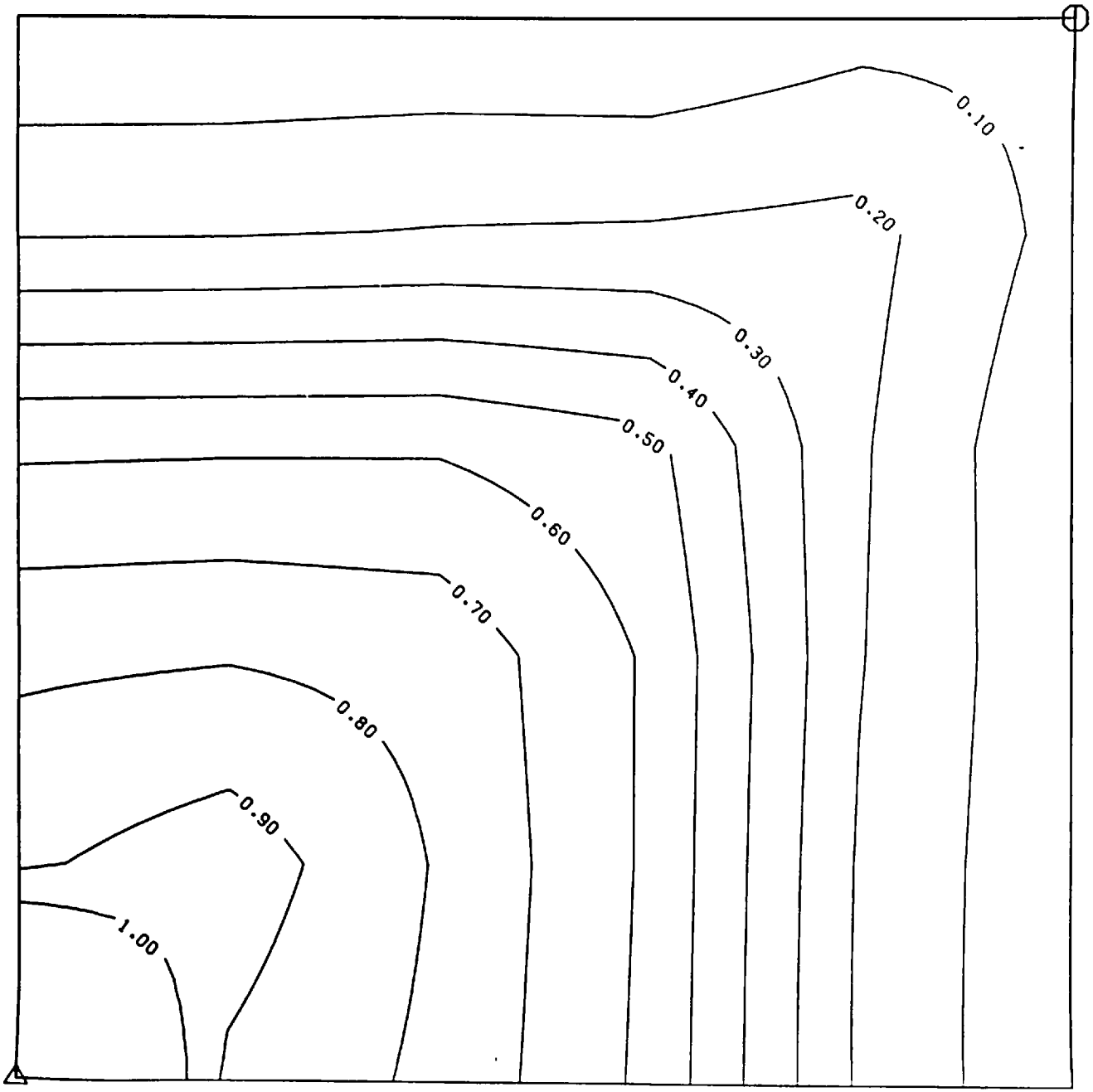

SW MAP.
TIME(D)=2160
PV INJ =0.432
PV REC =0.423
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G31·8

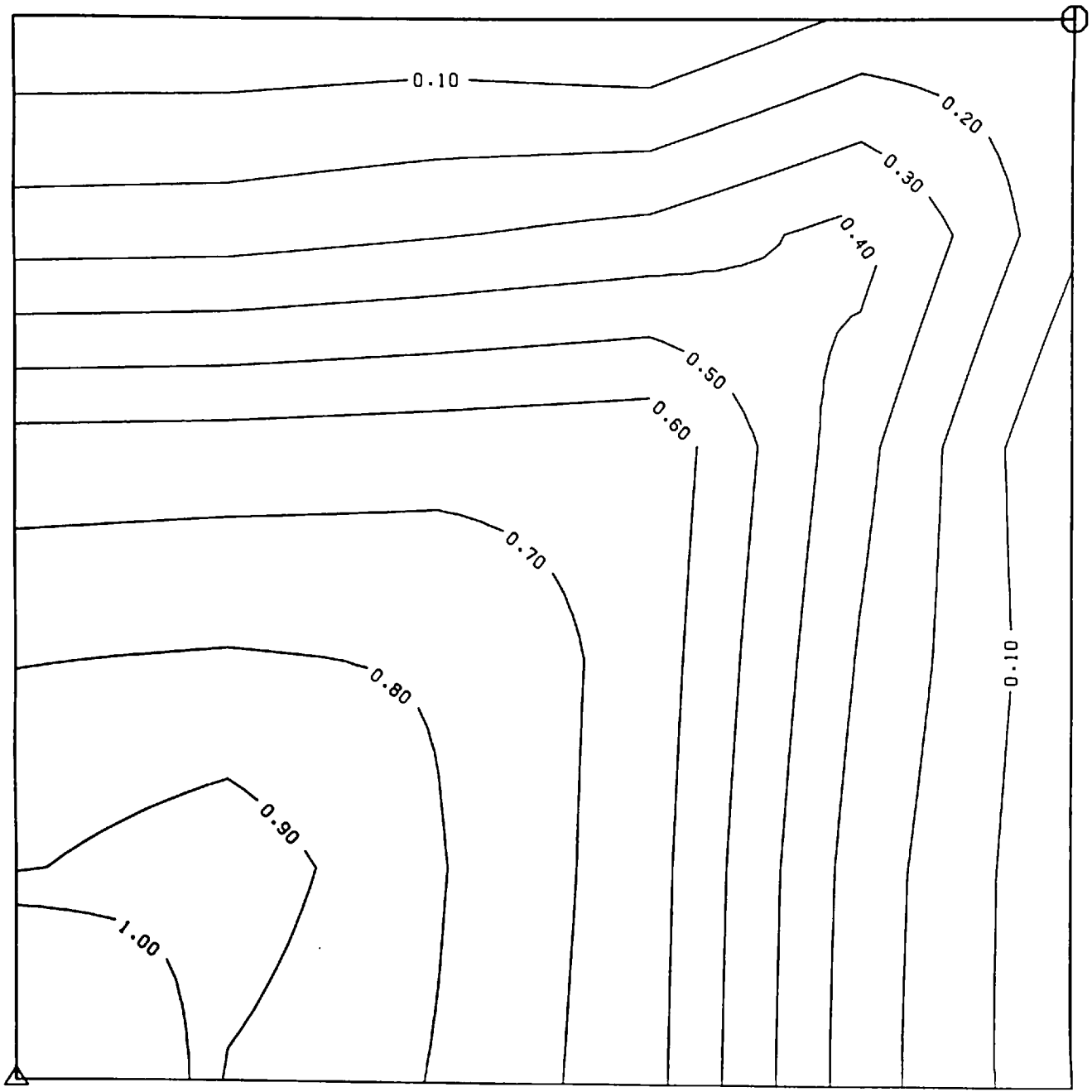
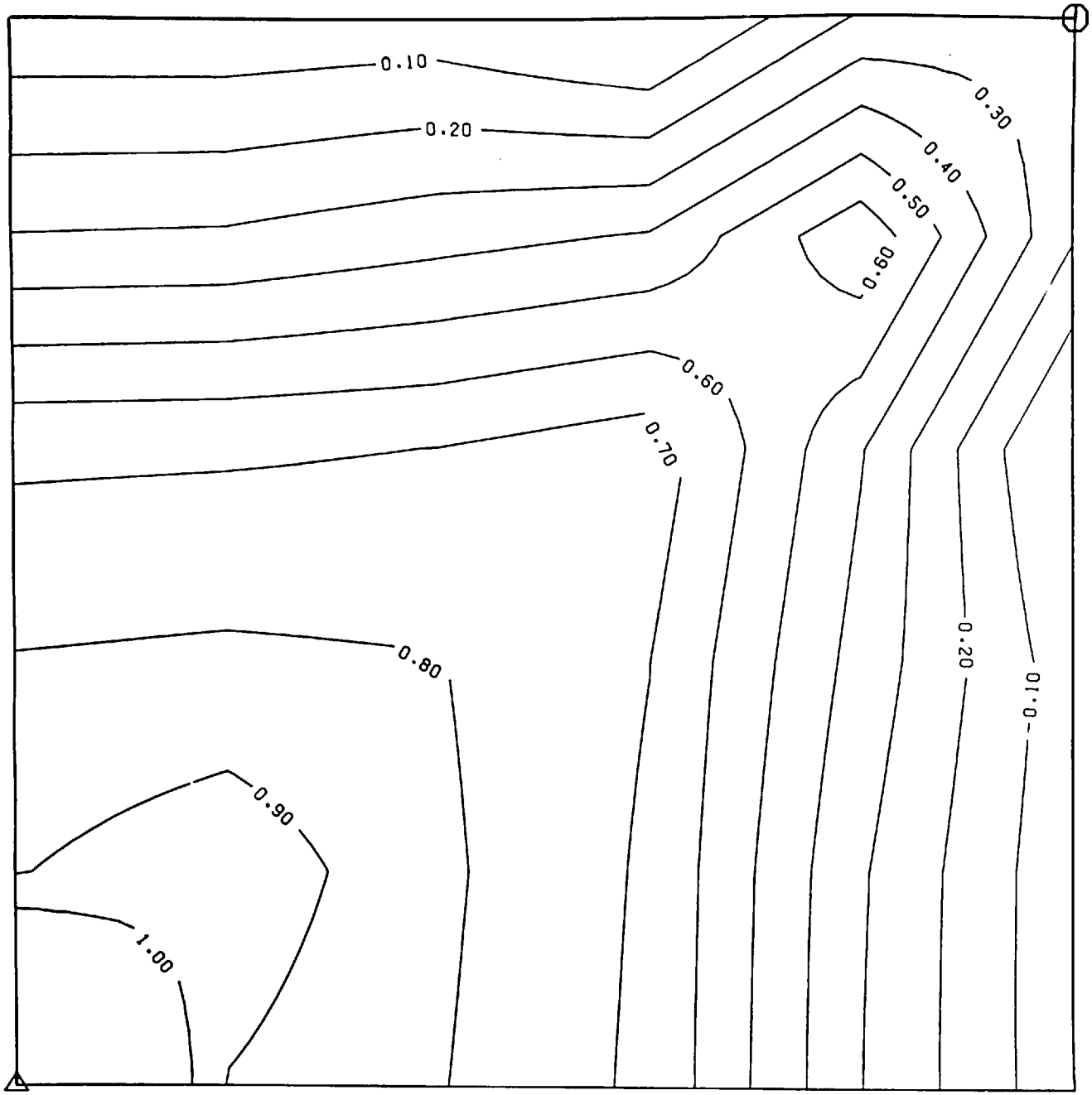

SW MAP.
TIME(D)=2520
PV INJ =0.504
PV REC =0.482
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00
0.10

FIGURE G31.9



SW MAP.
TIME(D)=2880
PV INJ =0.576
PV REC =0.529

FIGURE G31.10

2 D LAGRANGE QUADRATICS.

5X5 ELEMENTS.

SPIVAK ET AL DATA.

DPC/DSW=0.

MODIFIED FW CURVE.

LUMPED CAPACITY MATRIX.

INJ. RATE =0.0002 PV/DAY.

NO UPSTREAM WEIGHTING.

DTMAX = 30 DAYS.

FIGURE G32.1

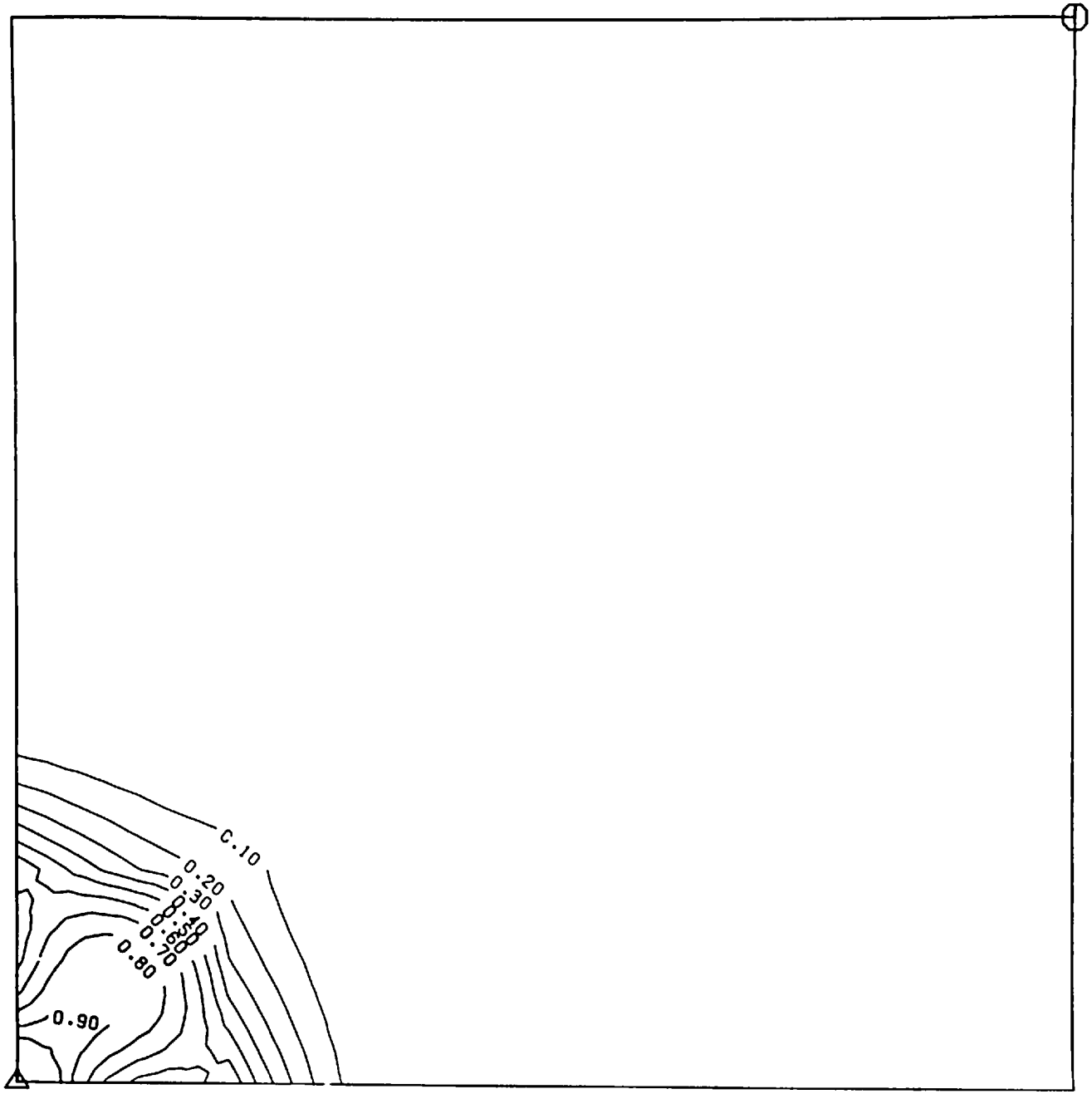

SW MAP.
TIME(D)=180
PV INJ =0.036
PV REC =0.036
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90

FIGURE G32.2

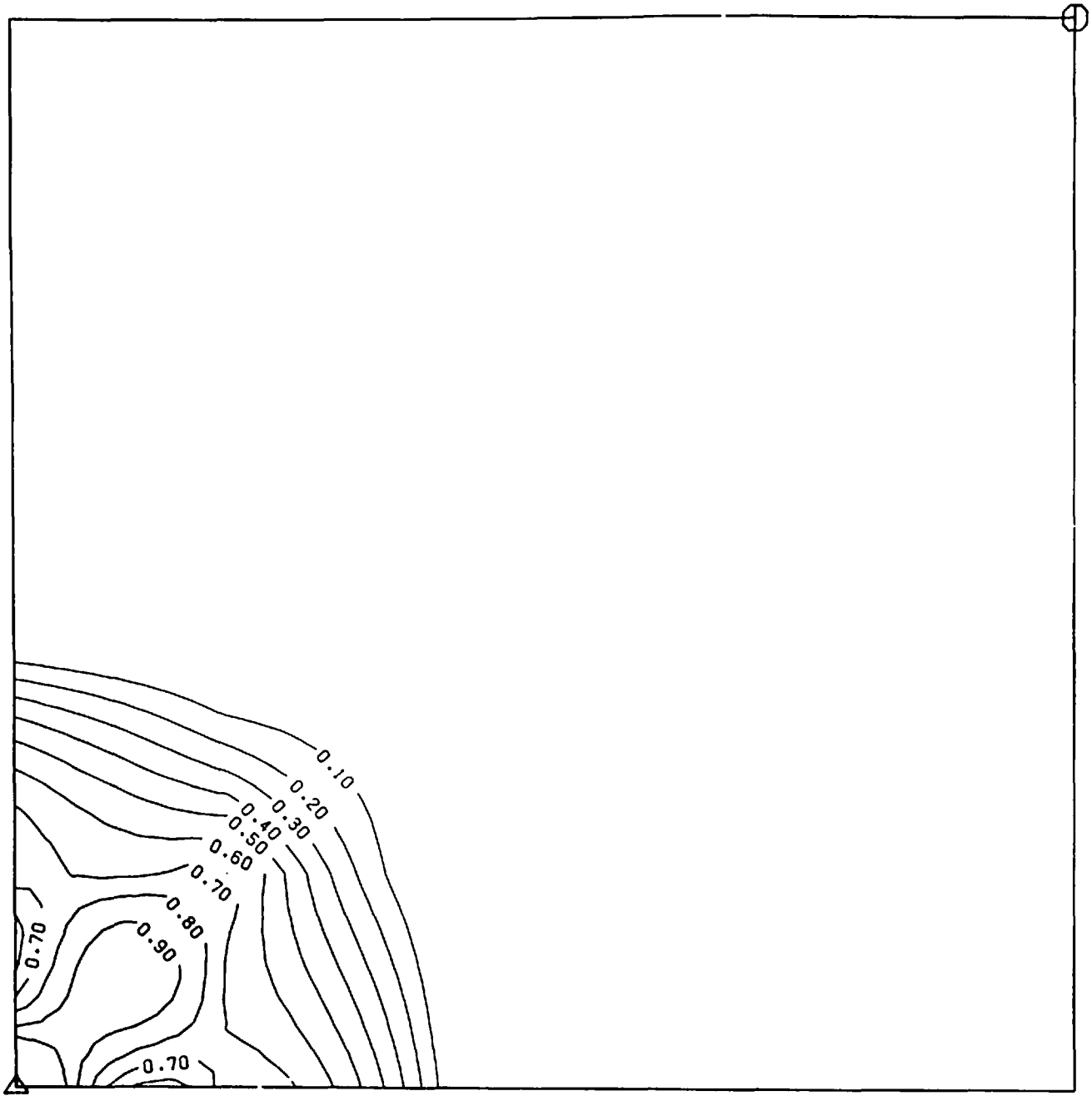

SW MAP.
TIME(D)=360
PV INJ =0.072
PV REC =0.072
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
0.70
0.70

FIGURE G32.3

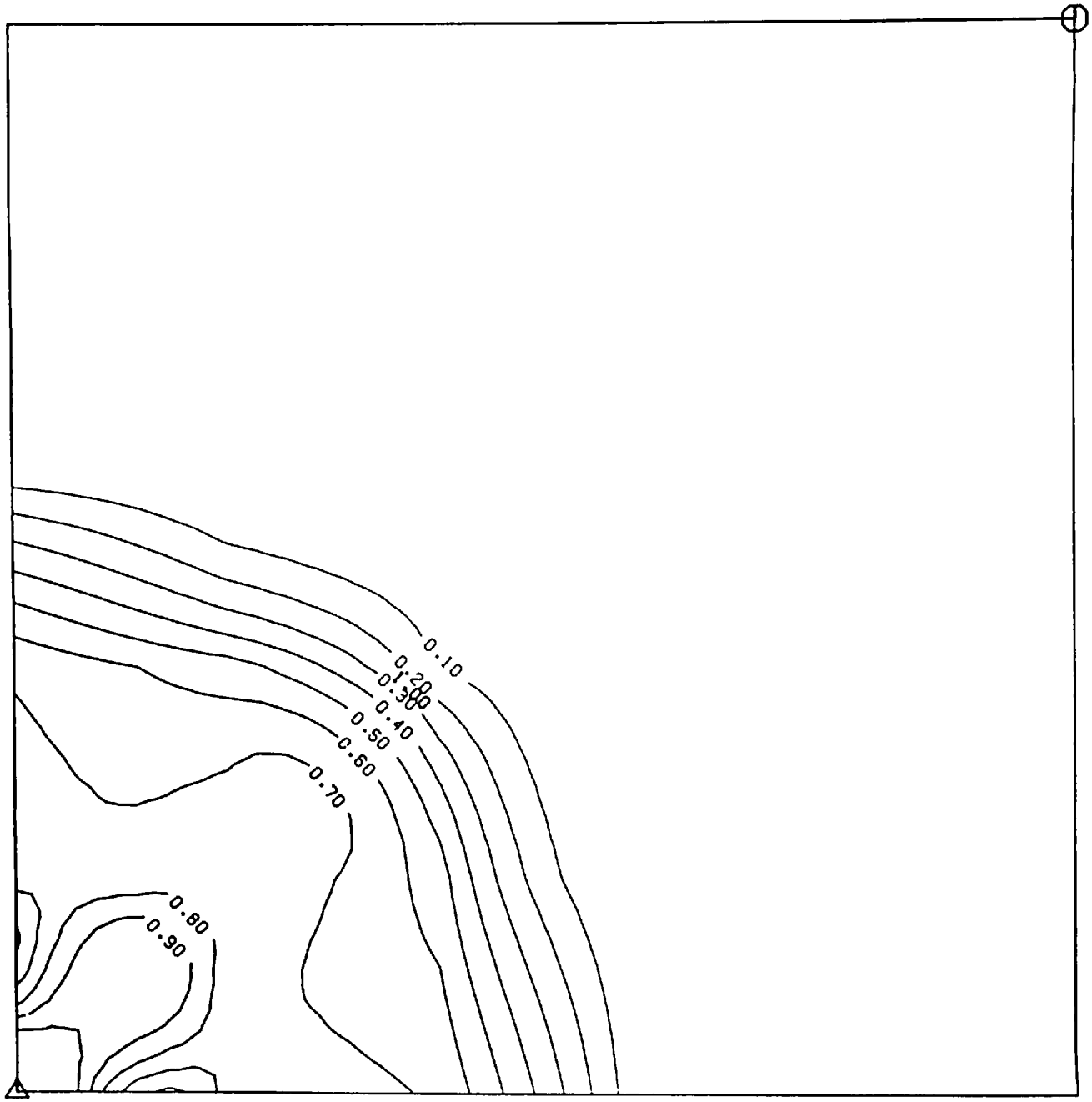

SW MAP.
TIME(D)=720
PV INJ =0.144
PV REC =0.144
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90

FIGURE G32.4

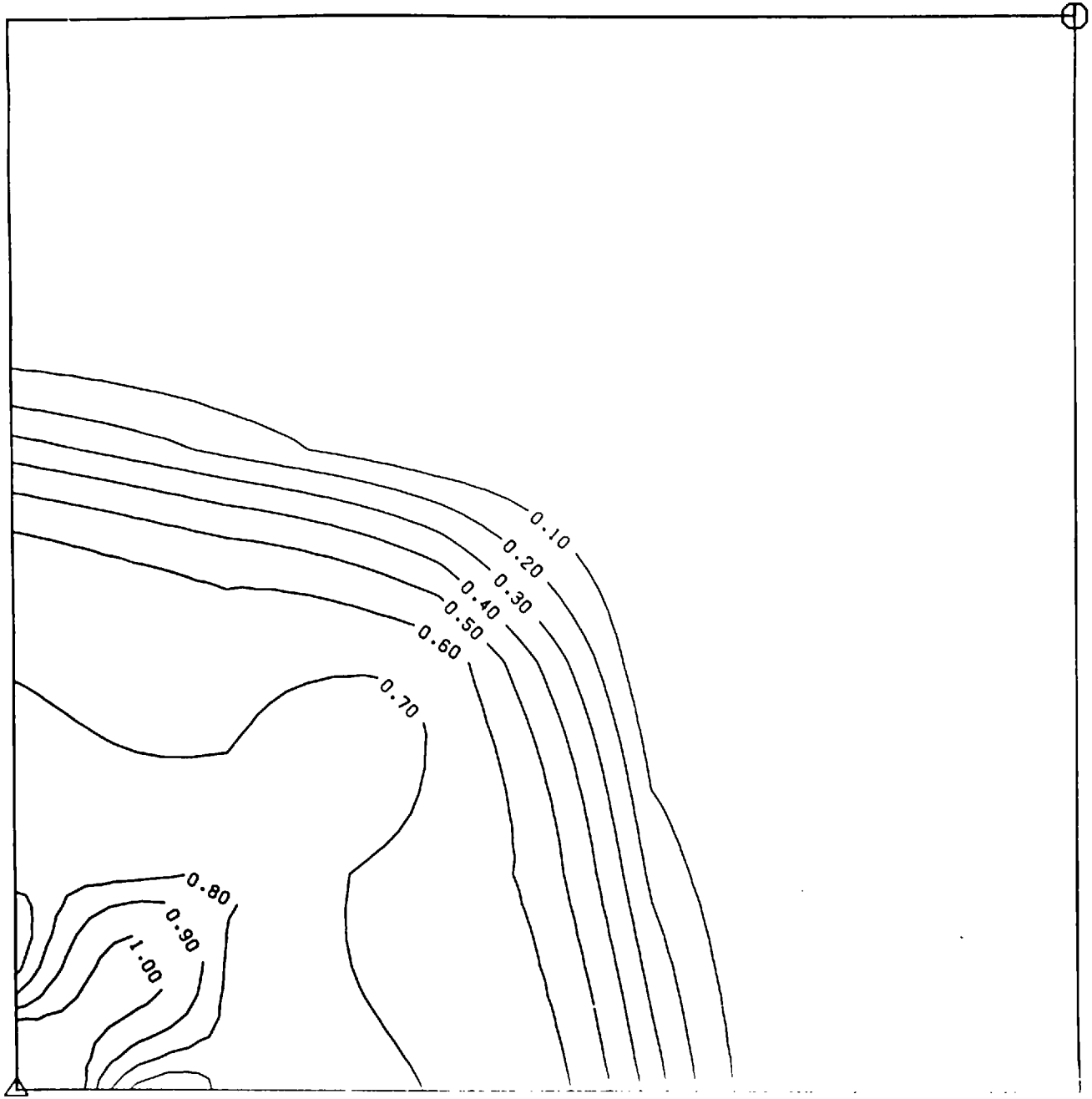

SW MAP.
TIME(D)=1080
PV INJ =0.216
PV REC =0.216
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G32.5

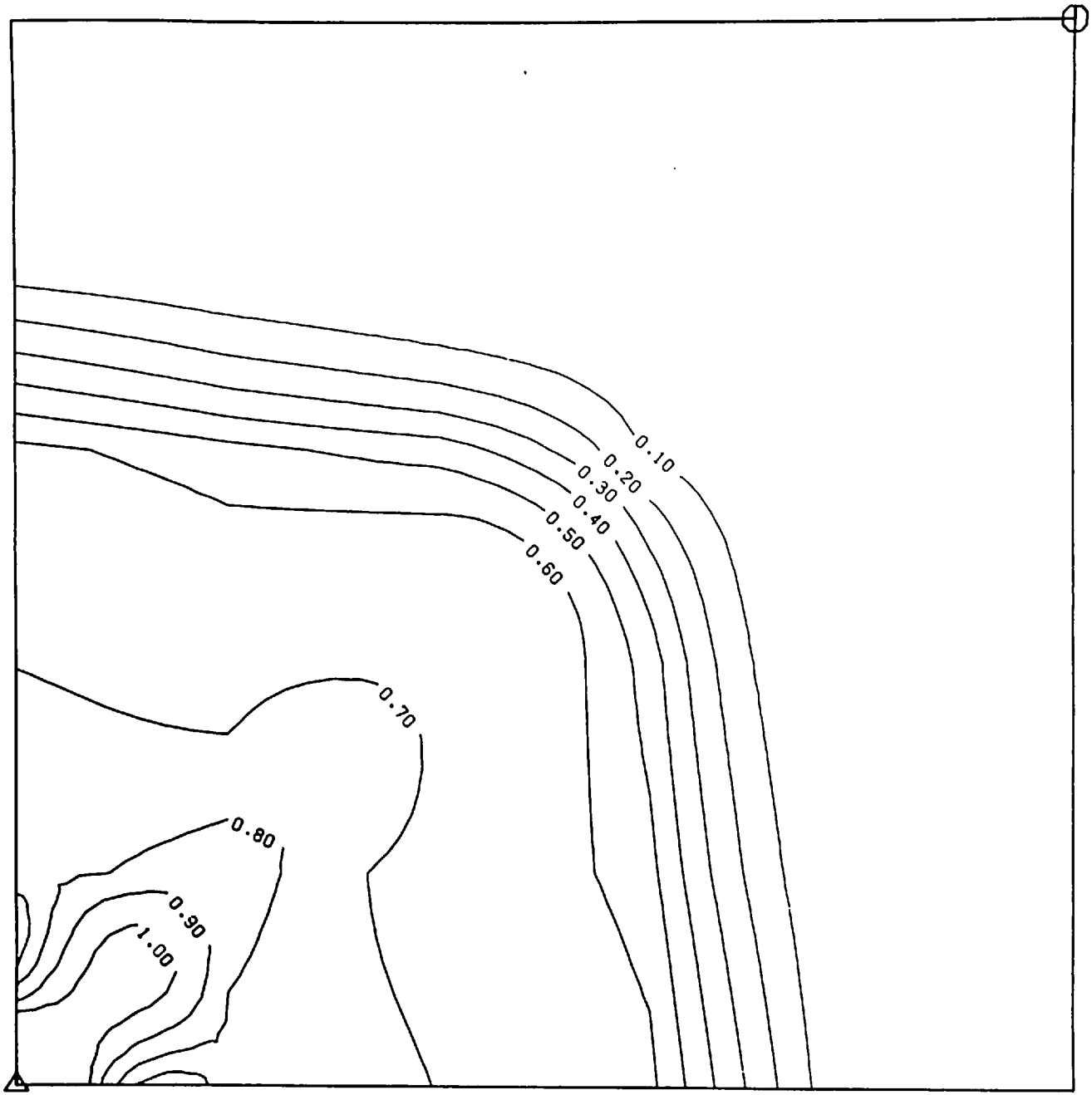

SW MAP.
TIME(D)=1440
PV INJ =0.288
PV REC =0.288
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G32.6

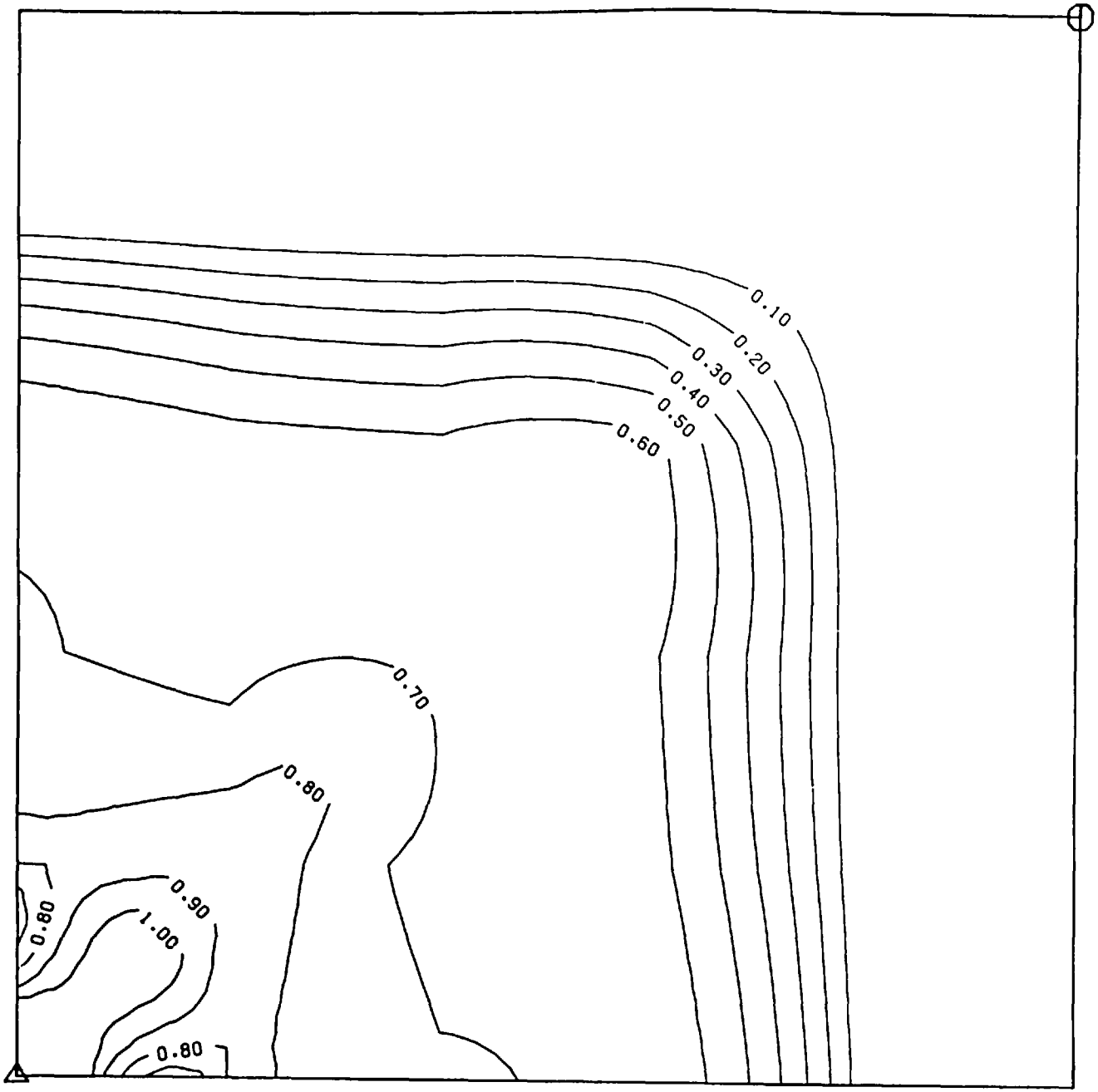

SW MAP.
TIME(D)=1800
PV INJ =0.360
PV REC =0.360
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00
0.80
0.80

FIGURE G32-7

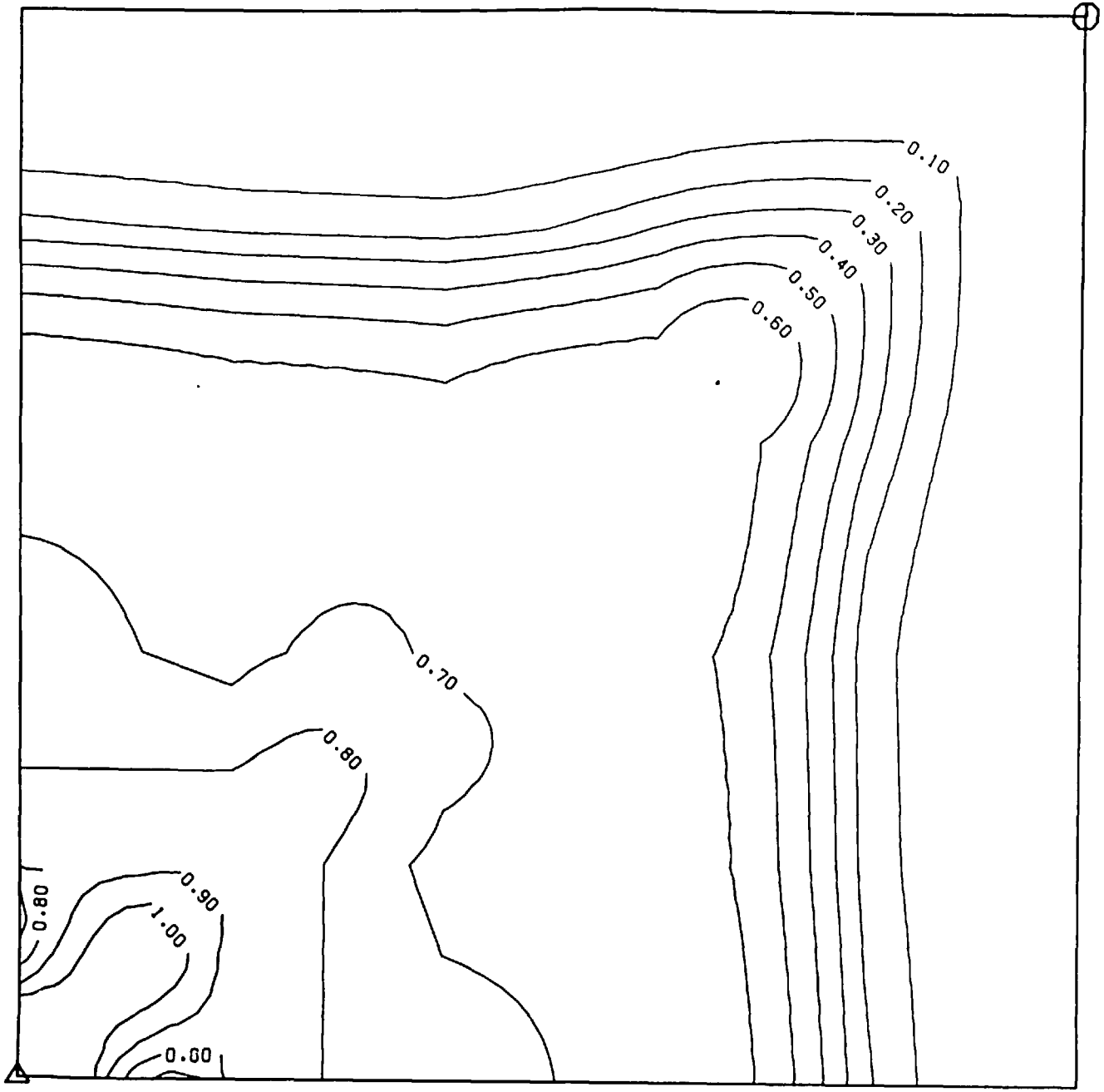

SW MAP.
TIME(D)=2160
PV INJ =0.432
PV REC =0.432
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00
0.00
0.80

FIGURE G32-8

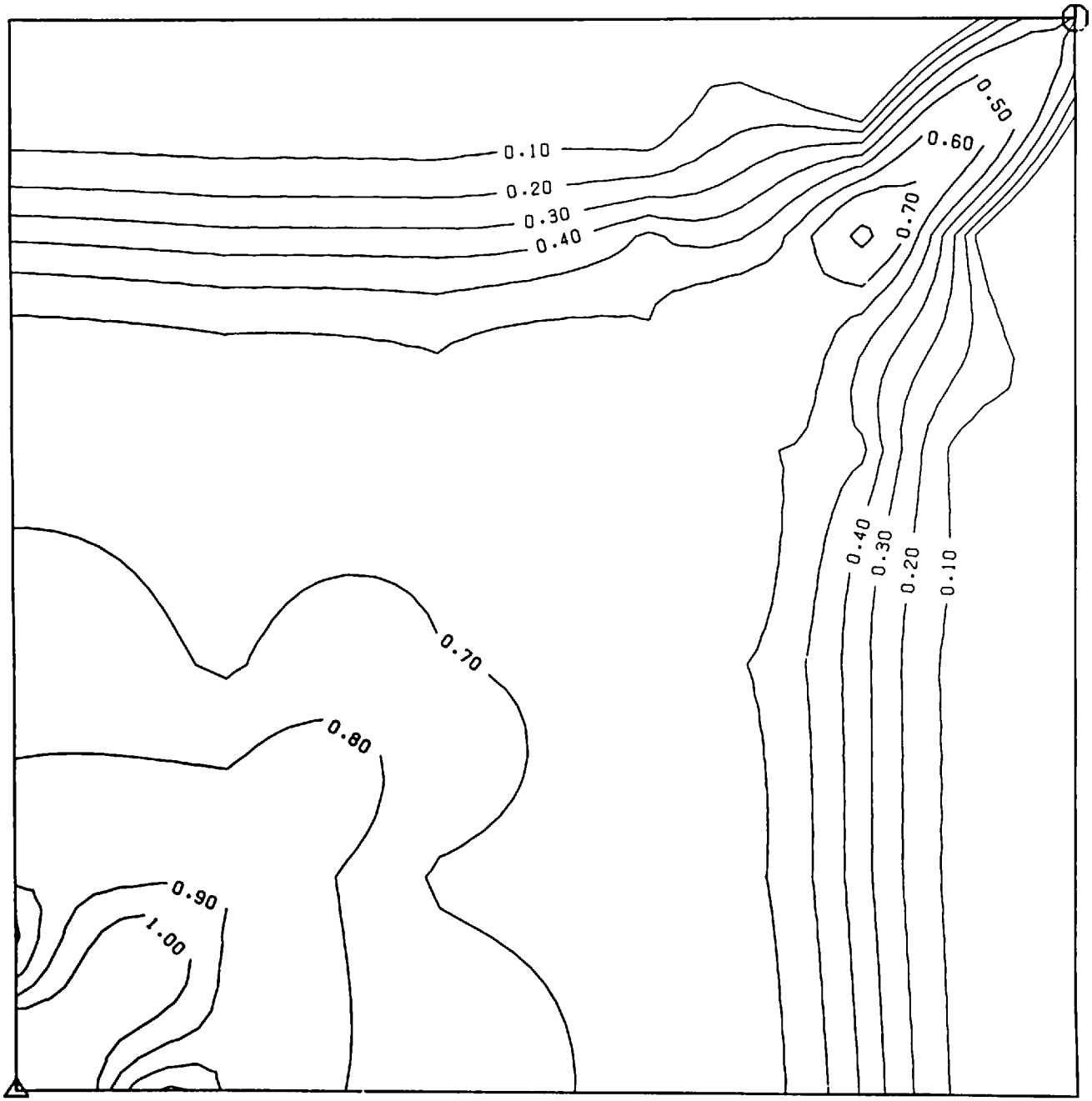

SW MAP.
TIME(D)=2520
PV INJ =0.504
PV REC =0.486
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.40
0.30
0.20
0.10
0.70
0.80
0.90
1.00

FIGURE G32·9

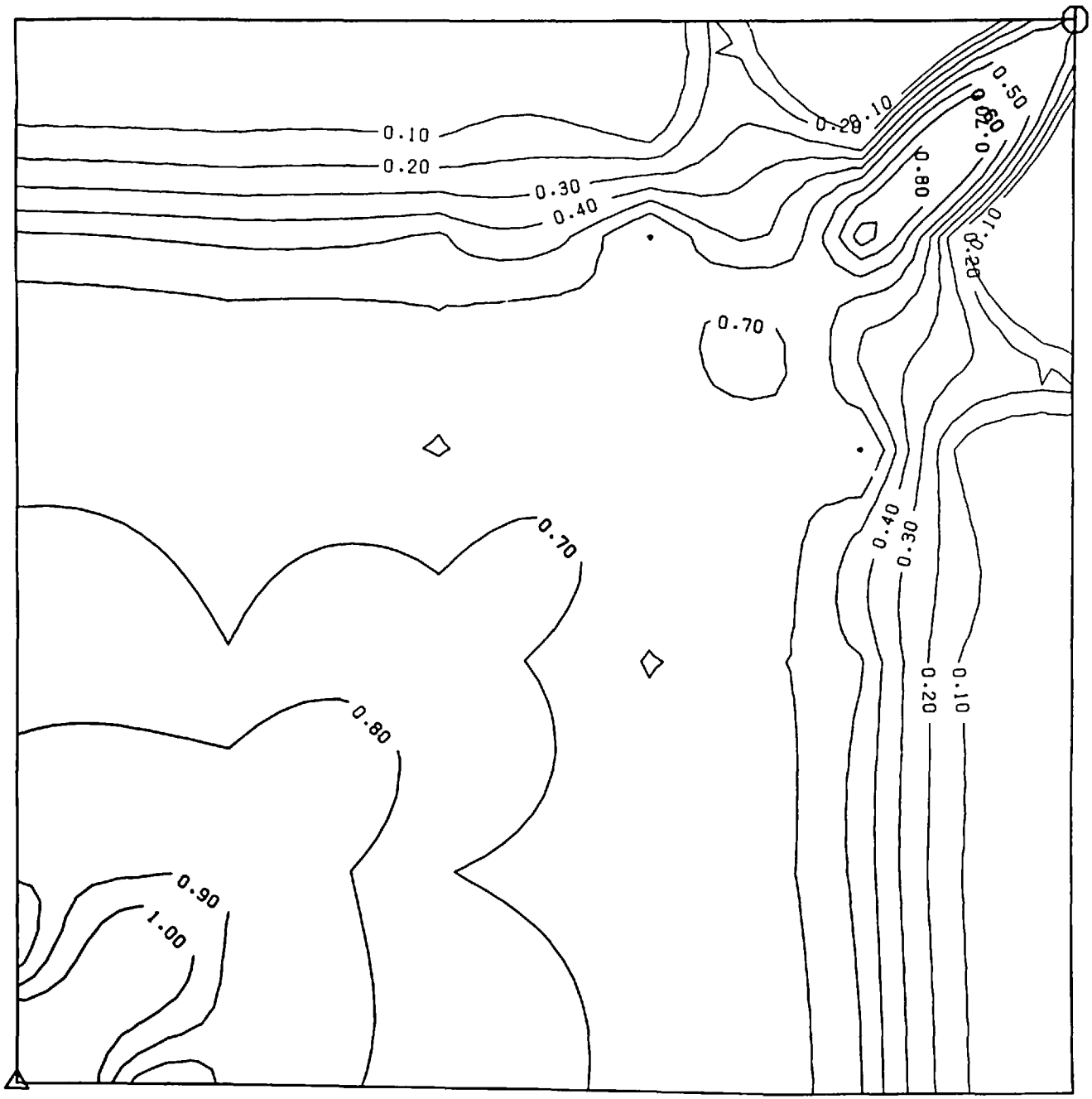


FIGURE G32.10

2 D HERMITE CUBICS.

5X5 ELEMENTS.

SPIVAK ET AL DATA.

DPC/DSW=0.

MODIFIED FW CURVE.

LUMPED CAPACITY MATRIX.

INJ. RATE =0.0002 PV/DAY.

NO UPSTREAM WEIGHTING.

DTMAX = 30 DAYS.

FIGURE G33.1

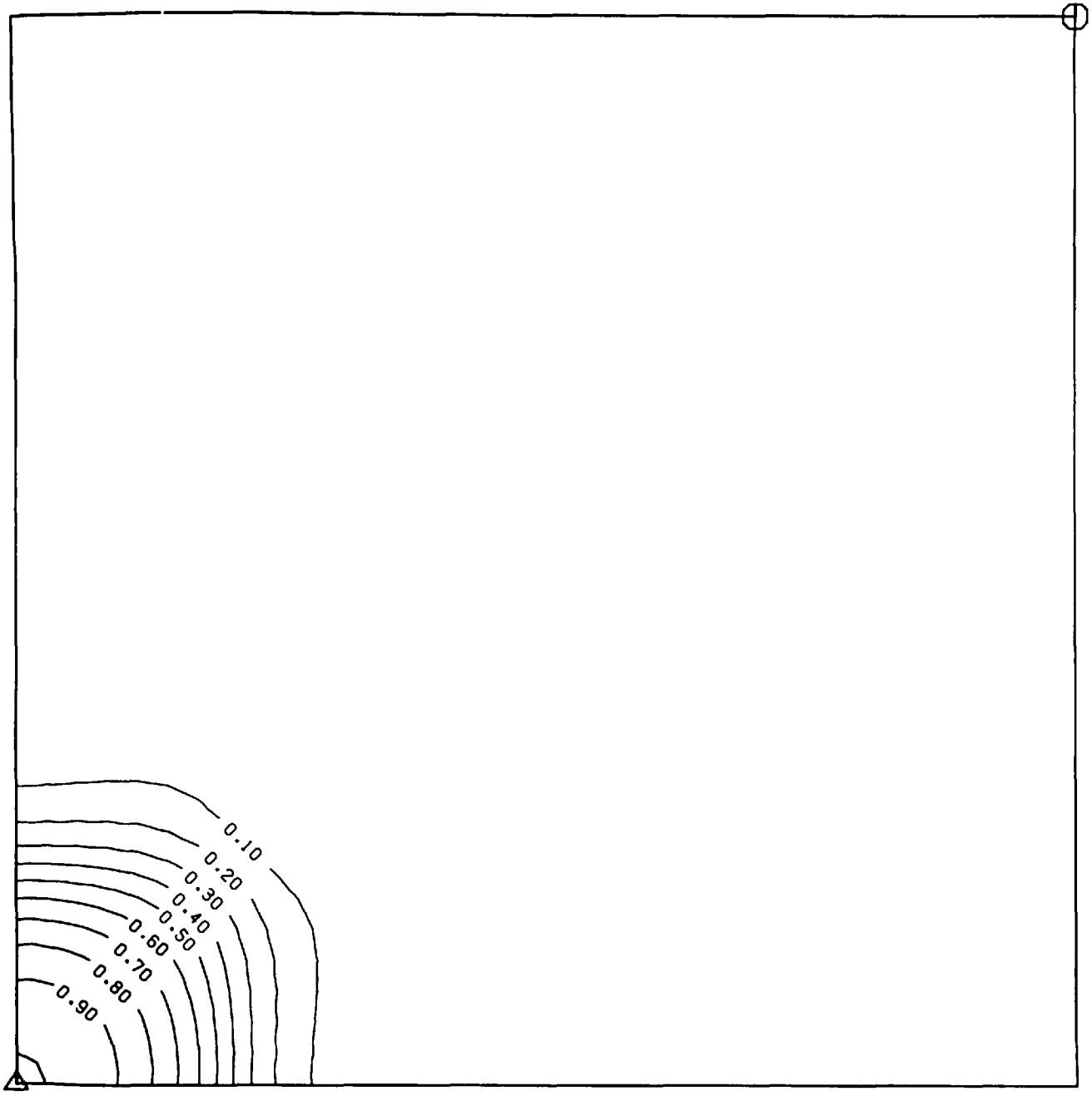

SW MAP.
TIME(D)=180
PV INJ =0.036
PV REC =0.036
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90

FIGURE G33.2

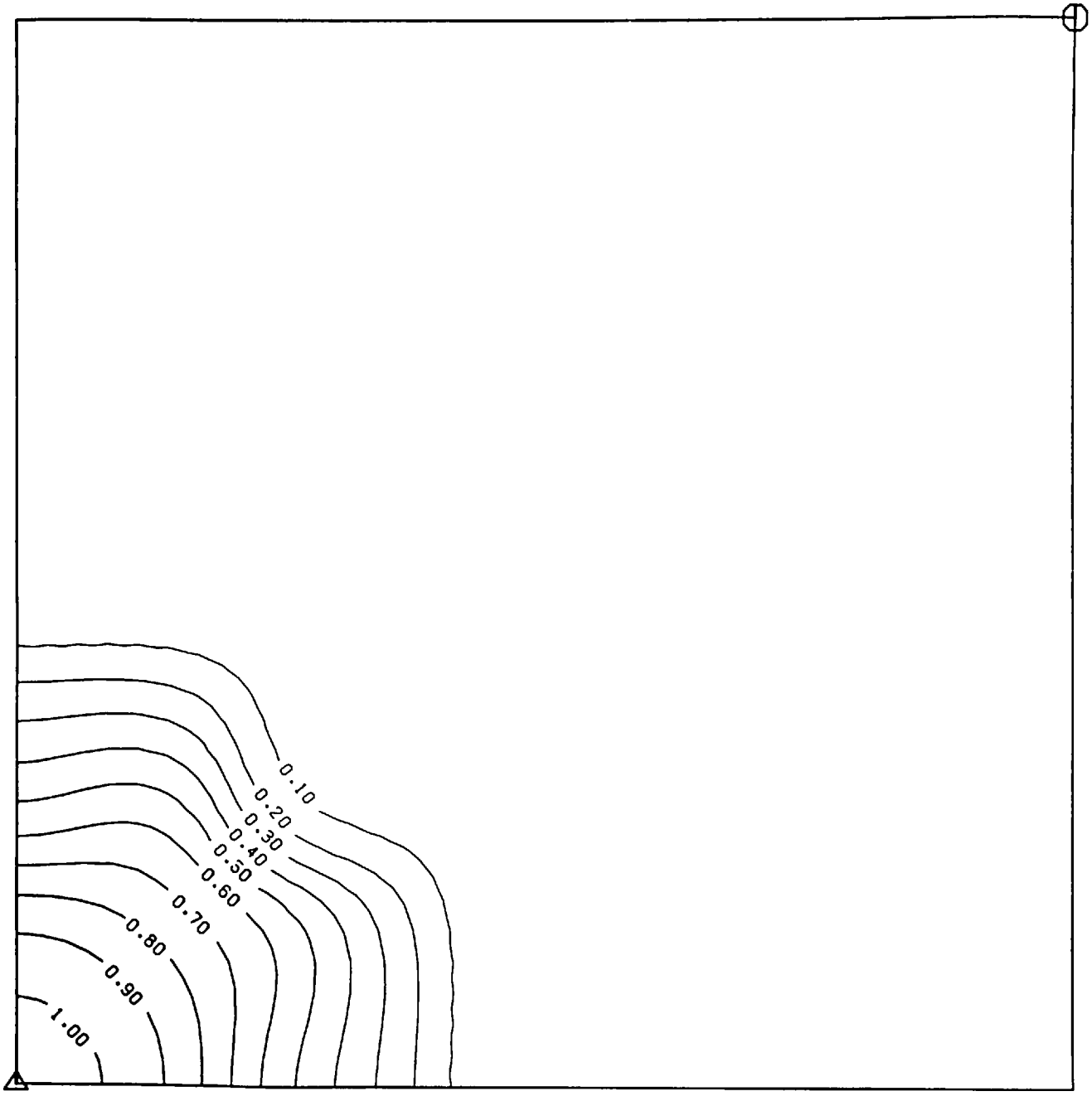

SW MAP.
TIME(D)=360
PV INJ =0.072
PV REC =0.072
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G33.3

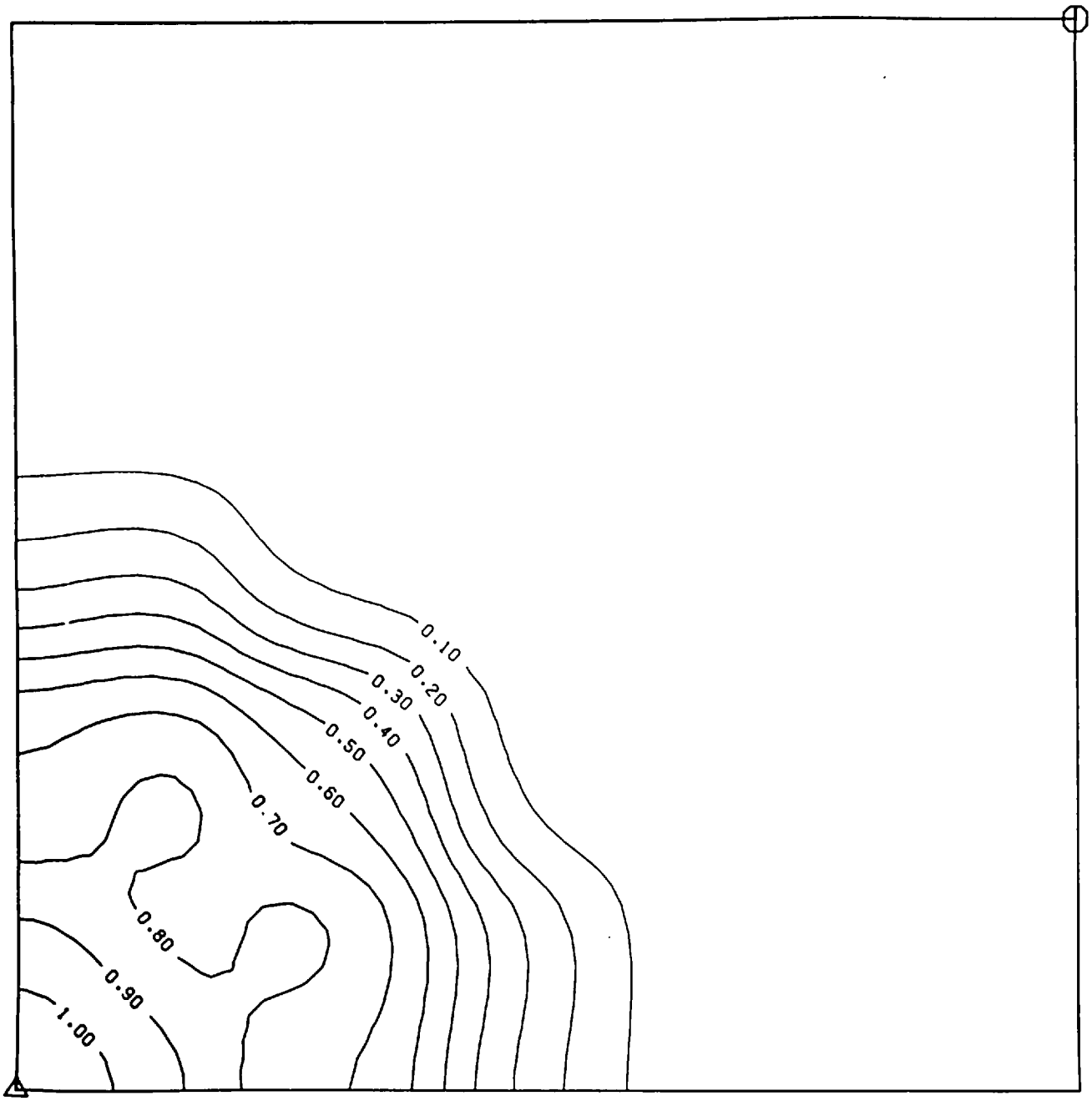

SW MAP.
TIME(D)=720
PV INJ =0.144
PV REC =0.144
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G33.4

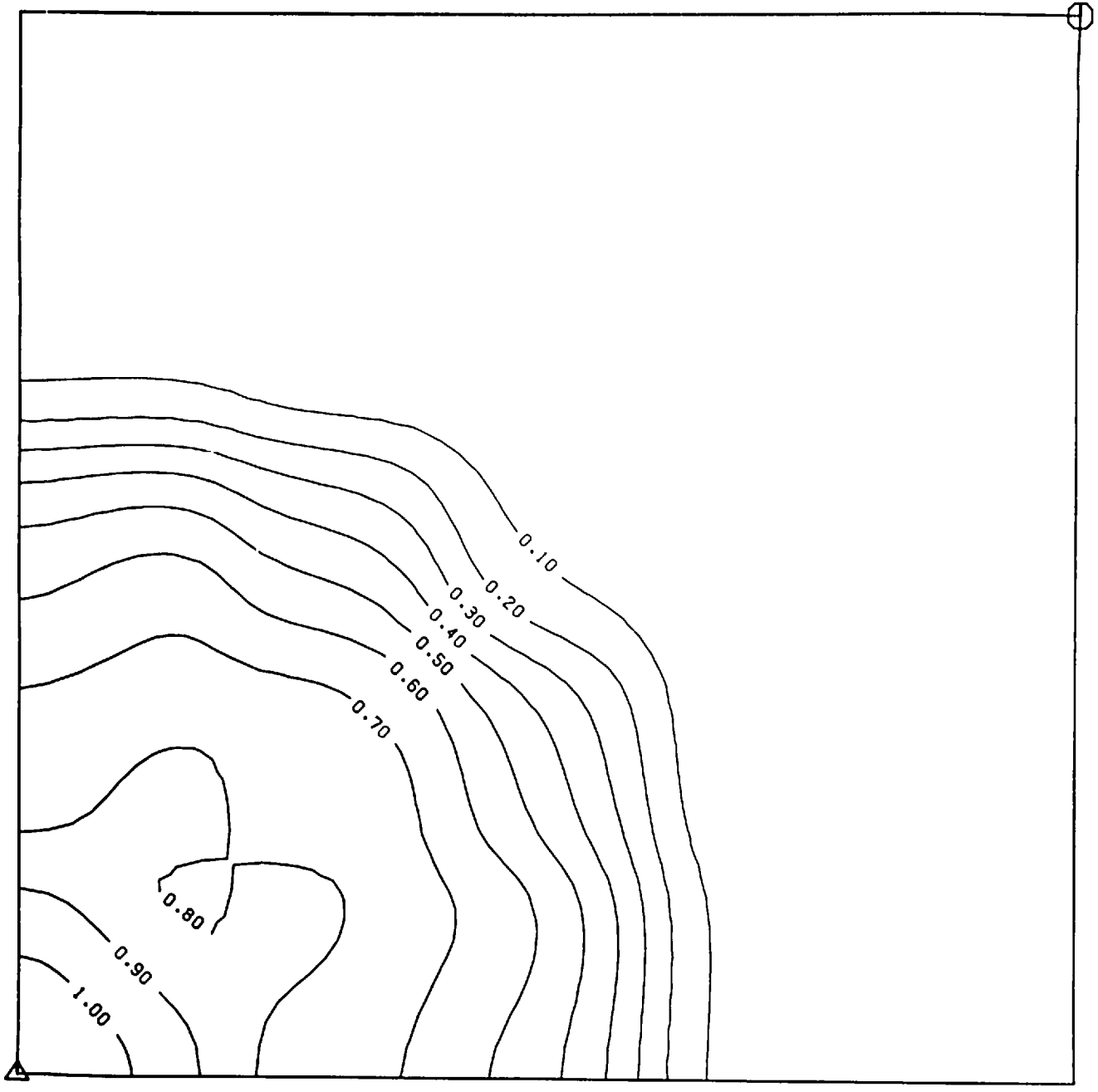

SW MAP.
TIME(D)=1080
PV INJ =0.216
PV REC =0.216
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G33-5

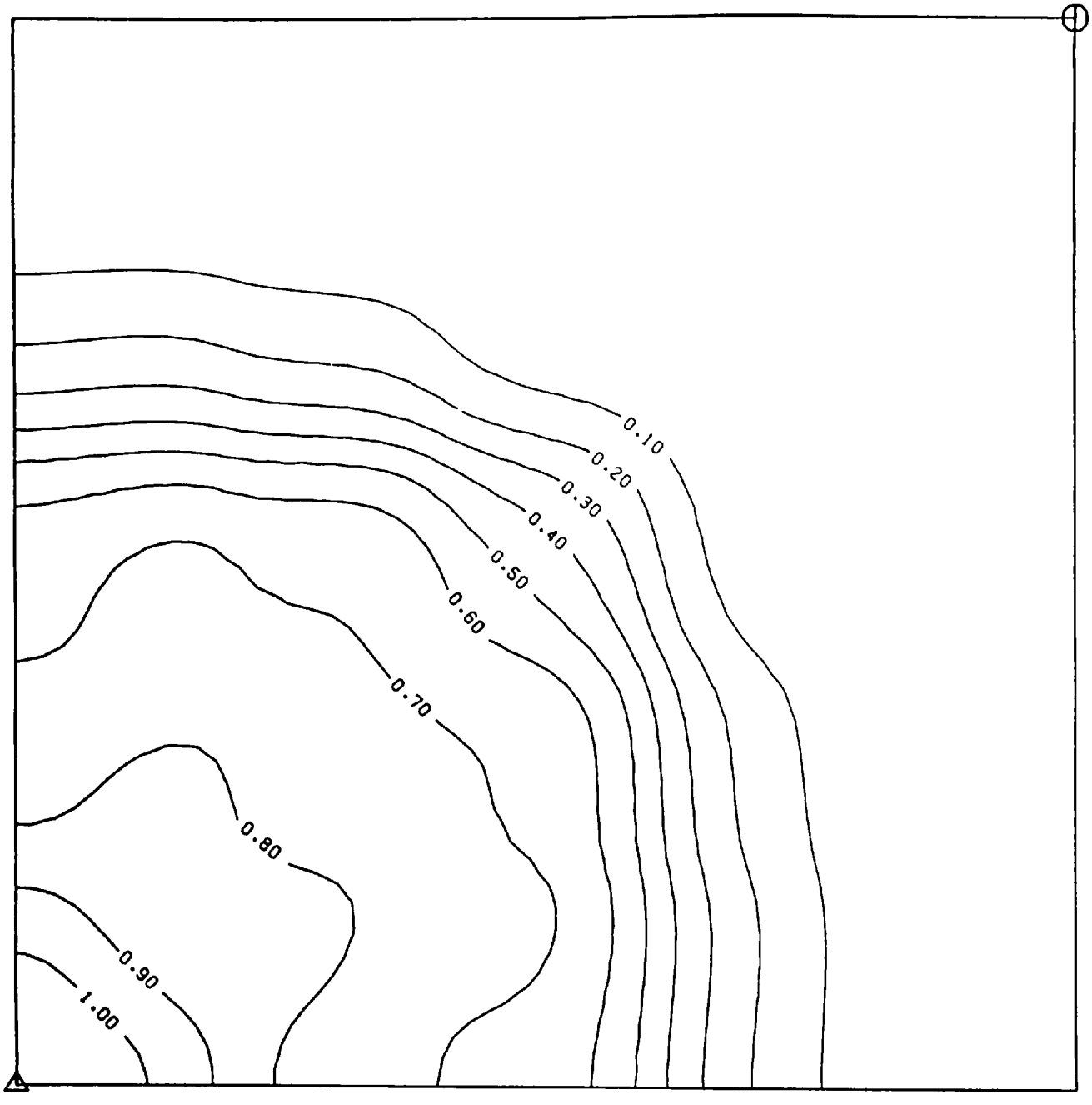

SW MAP.
TIME(D)=1440
PV INJ =0.288
PV REC =0.288
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G33.6

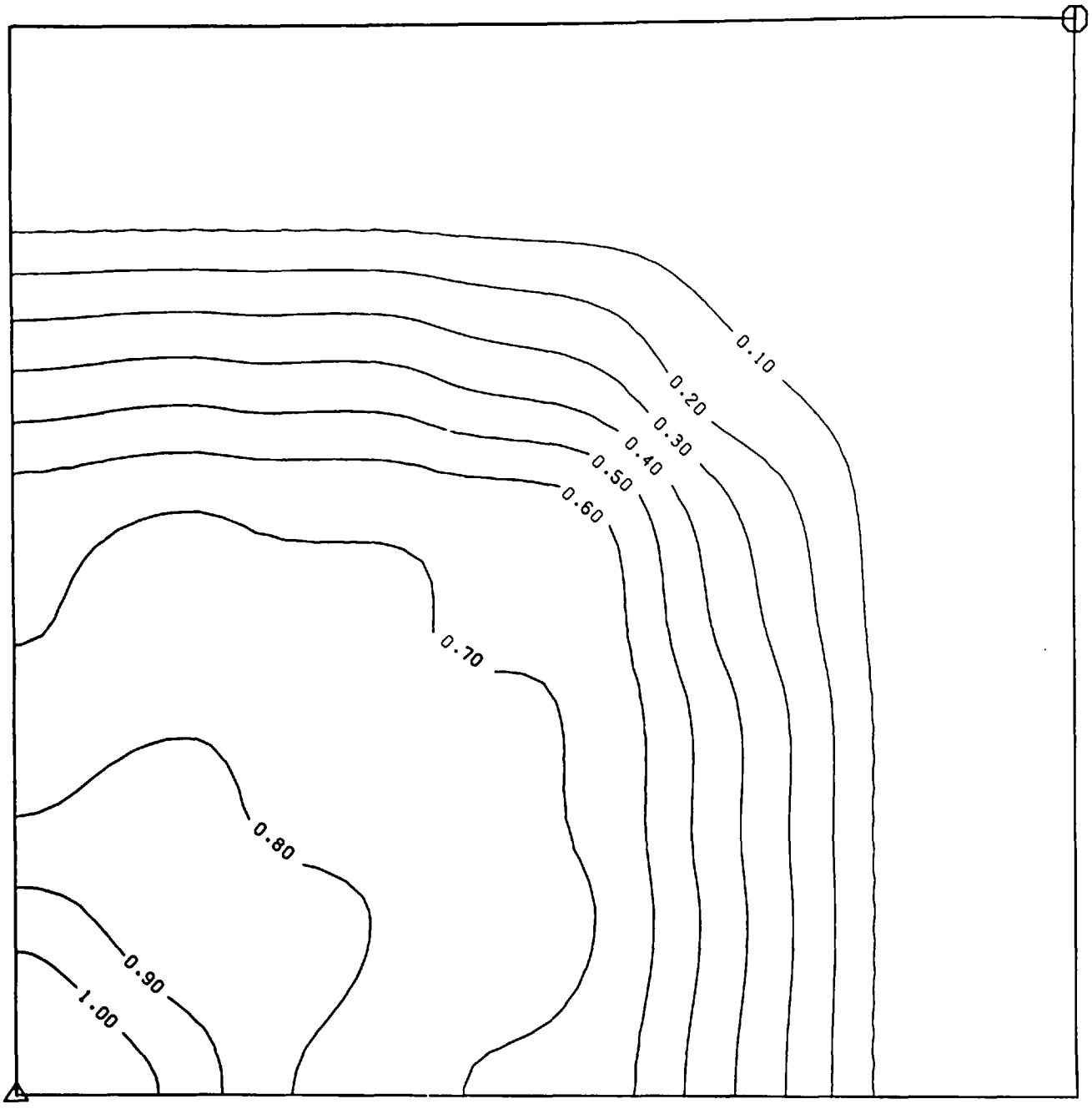

SW MAP.
TIME(D)=1800
PV INJ =0.360
PV REC =0.360
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G33.7

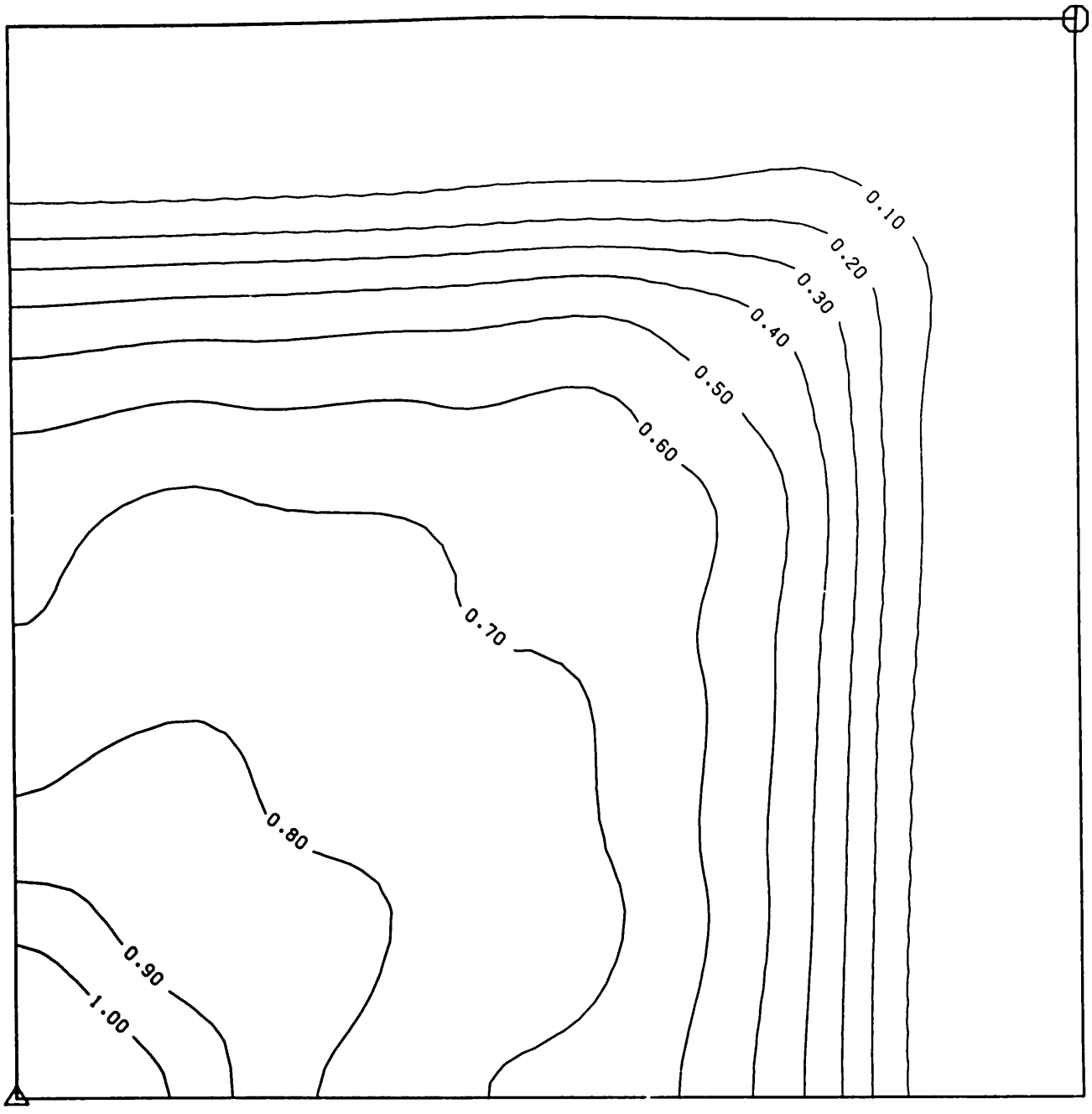

SW MAP.
TIME(D)=2160
PV INJ =0.432
PV REC =0.431
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

FIGURE G33·8

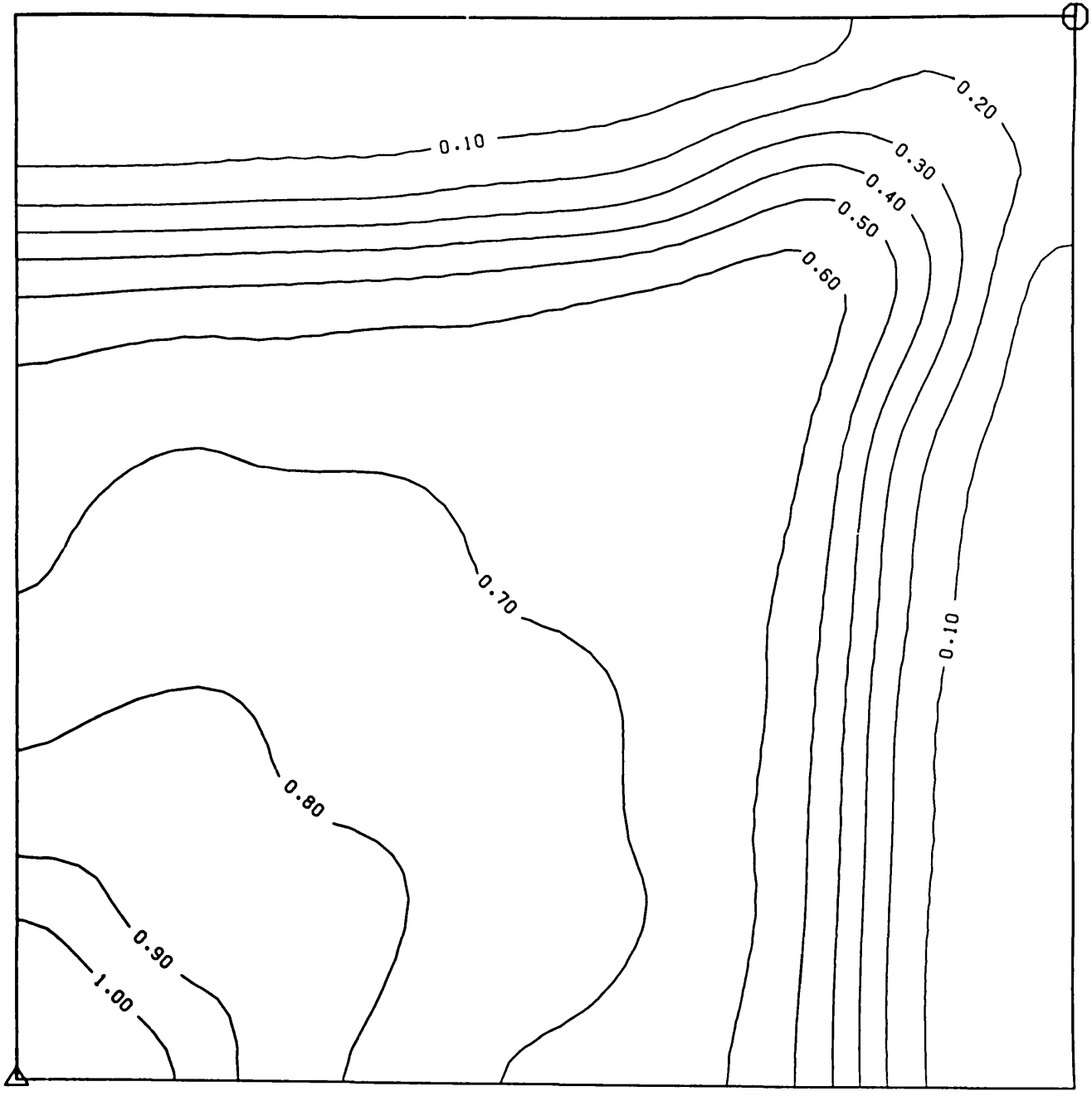

SW MAP.
TIME(D)=2520
PV INJ =0.504
PV REC =0.496
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00
0.10

FIGURE G33.9

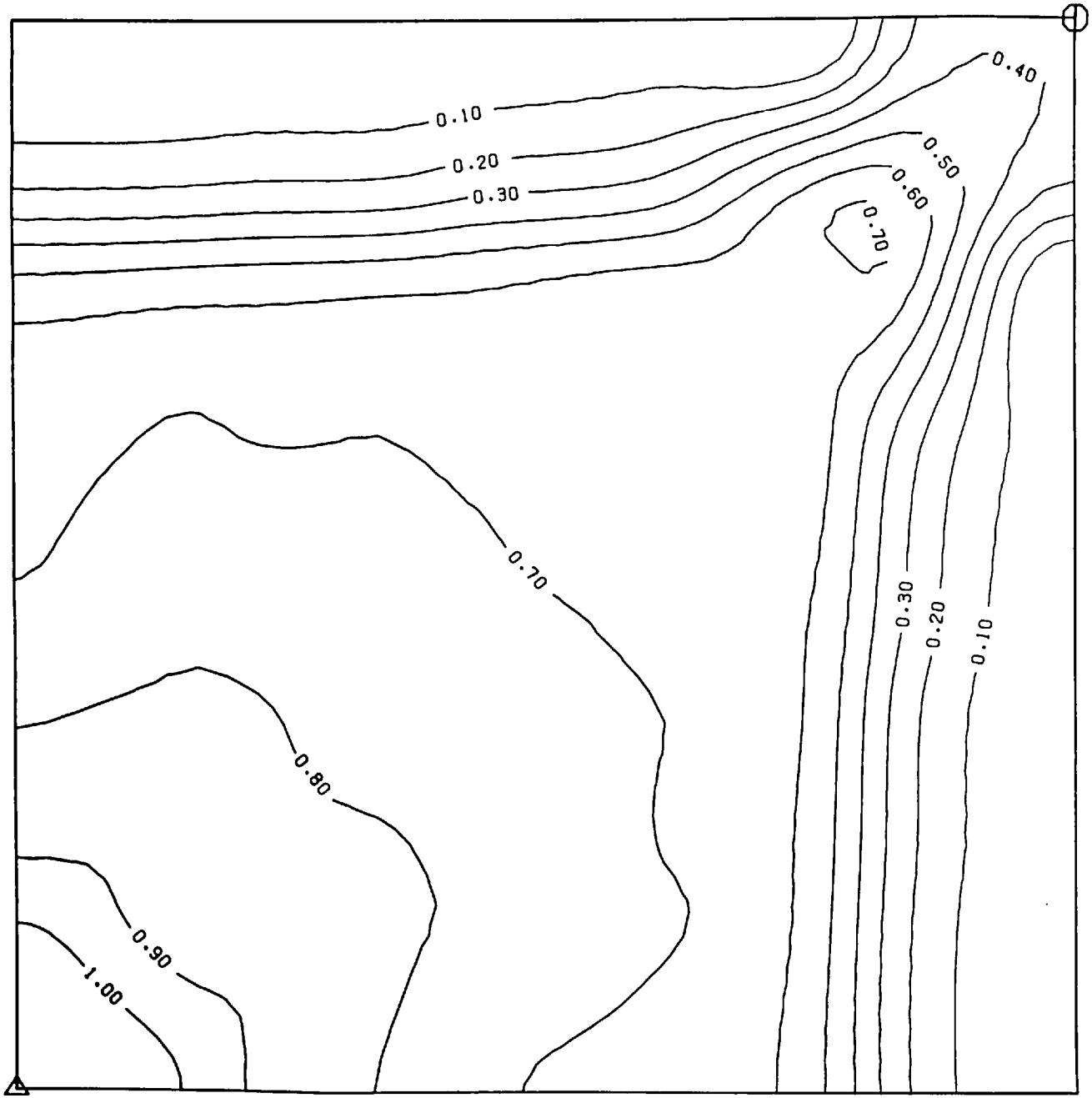


FIGURE 6.33.10

NINE POINT FD.

10X10 ELEMENTS.

SPIVAK ET AL DATA.

DPC/DSW = 0.

UNMODIFIED FW CURVE.

INJ. RATE = 0.0002 PV/DAY.

TWO POINT UPSTREAM.

DTMAX = 30 DAYS.

FIGURE G34.1

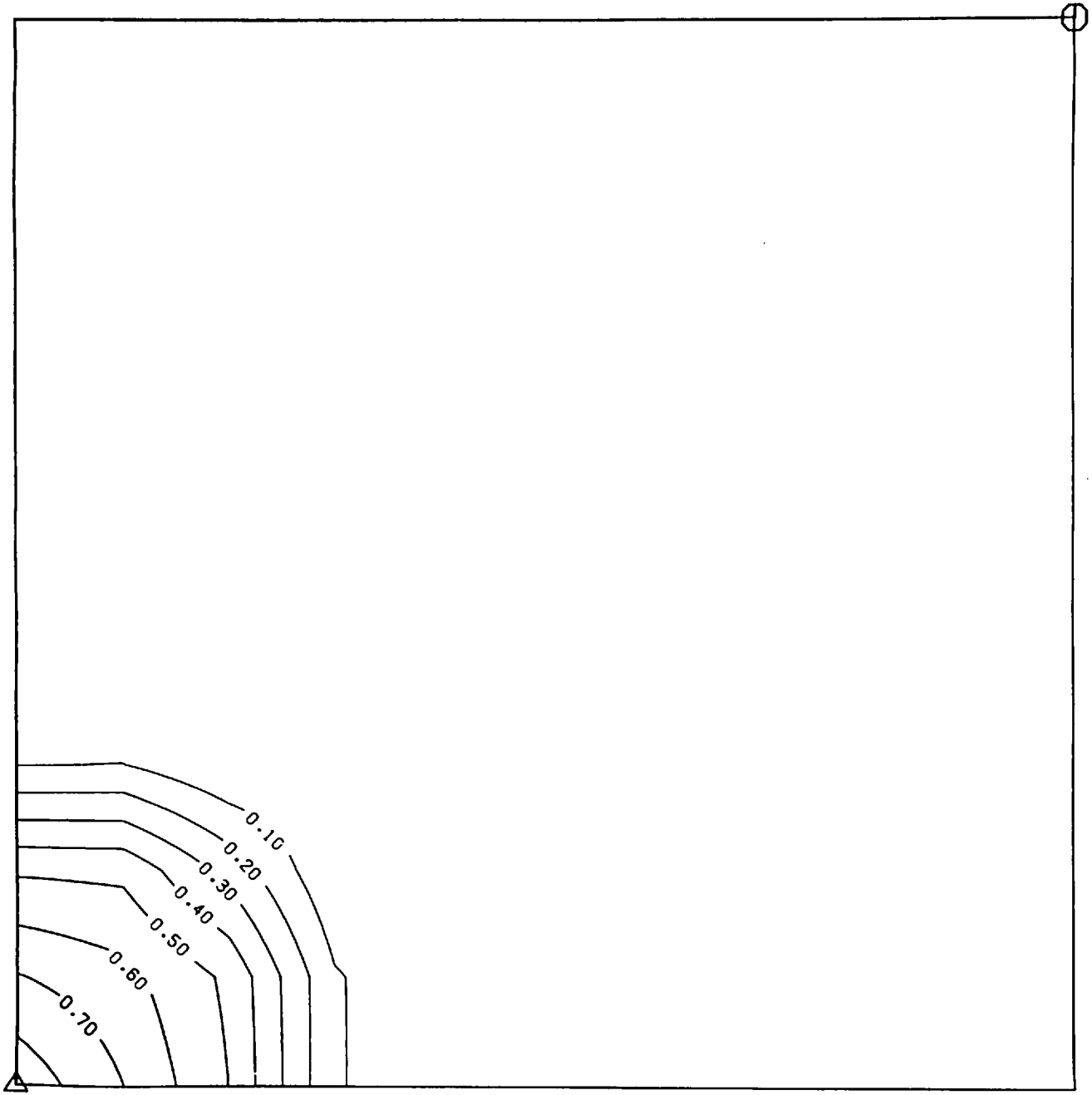

SW MAP.
TIME(D)=180
PV INJ =0.036
PV REC =0.036
0.10
0.20
0.30
0.40
0.50
0.60
0.70

FIGURE G34.2

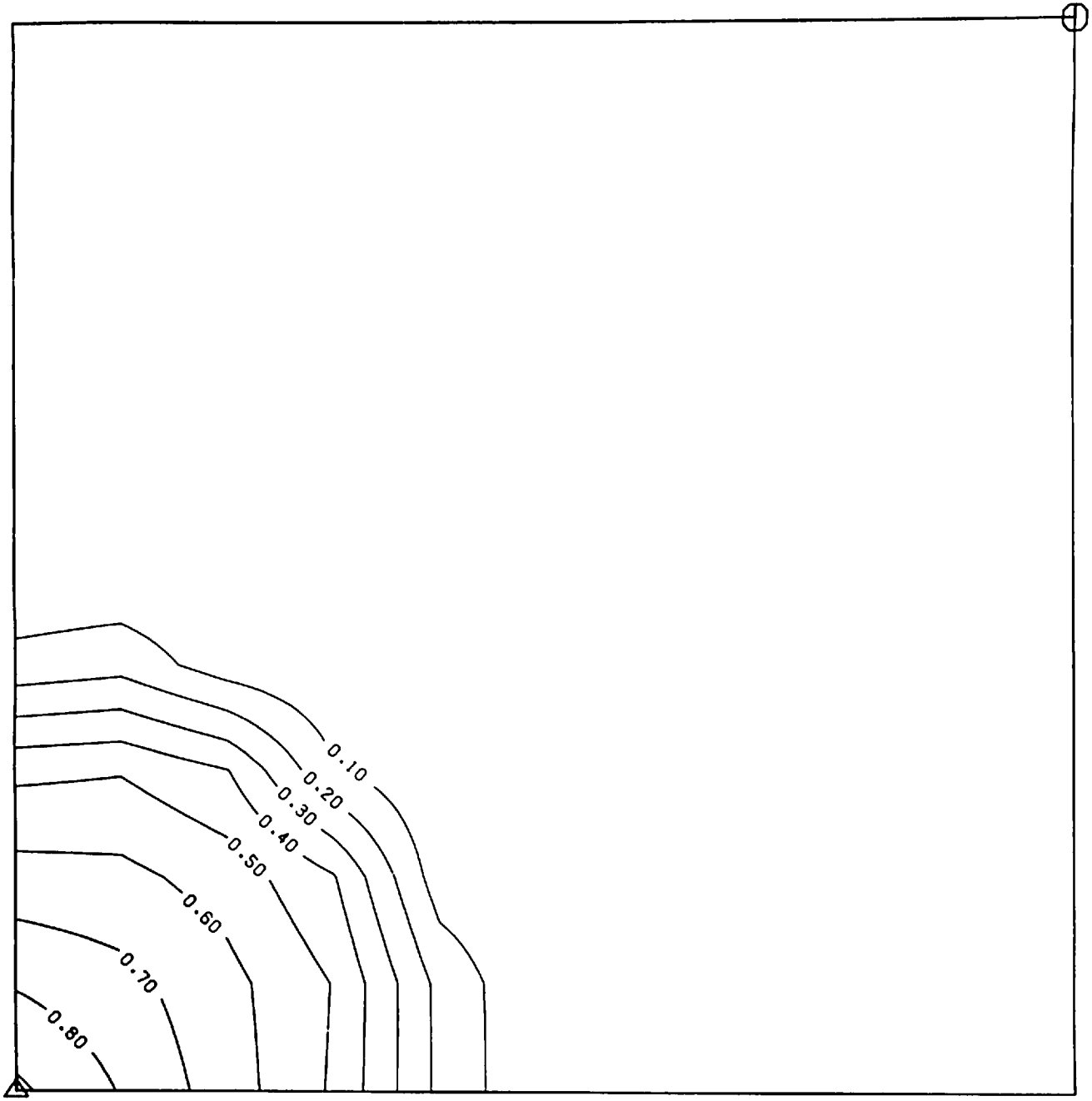

SW MAP.
TIME(D)=360
PV INJ =0.072
PV REC =0.072
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80

FIGURE G34.3

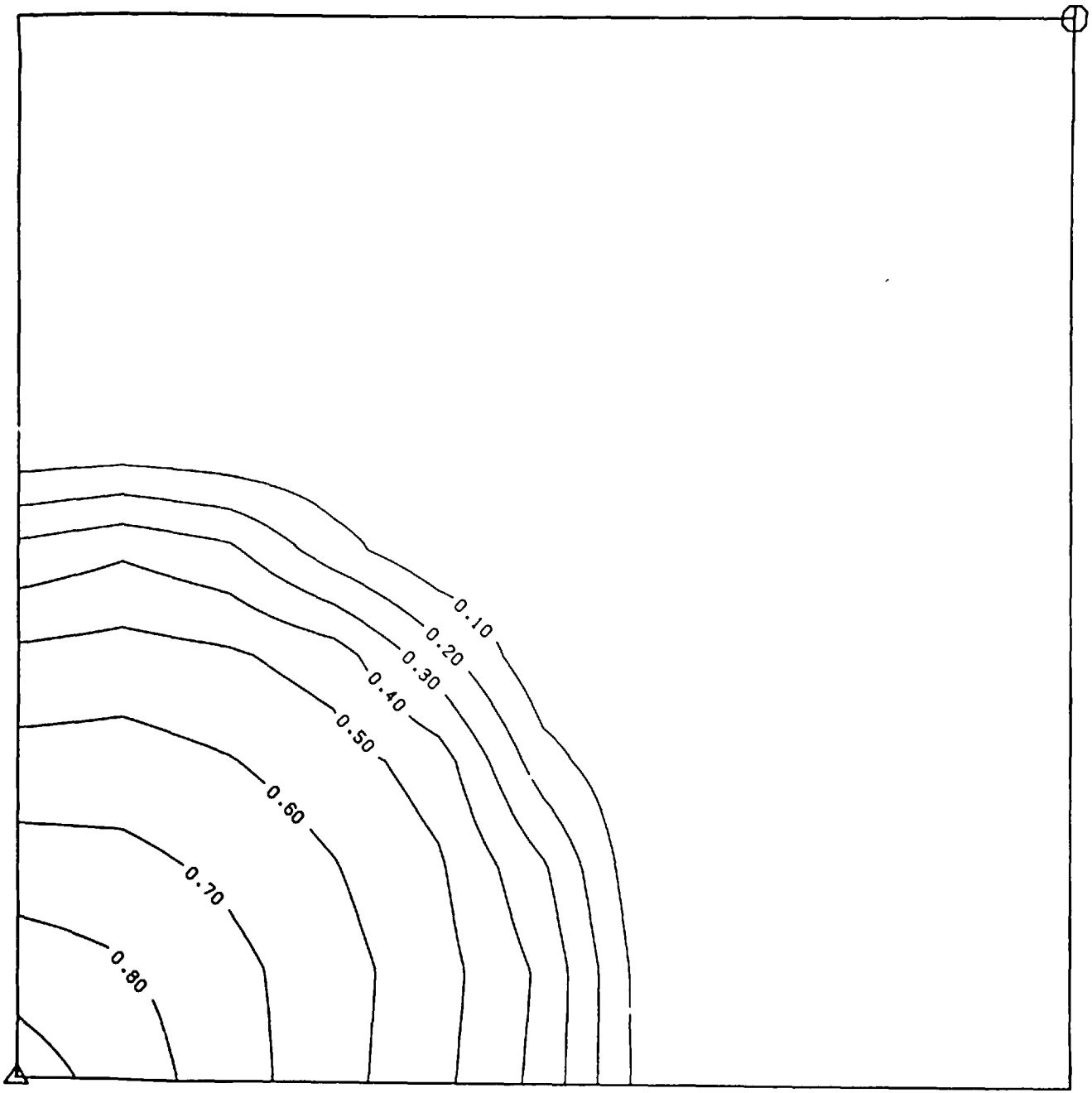

SW MAP.
TIME(D)=720
PV INJ =0.144
PV REC =0.144
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80

FIGURE 634.4

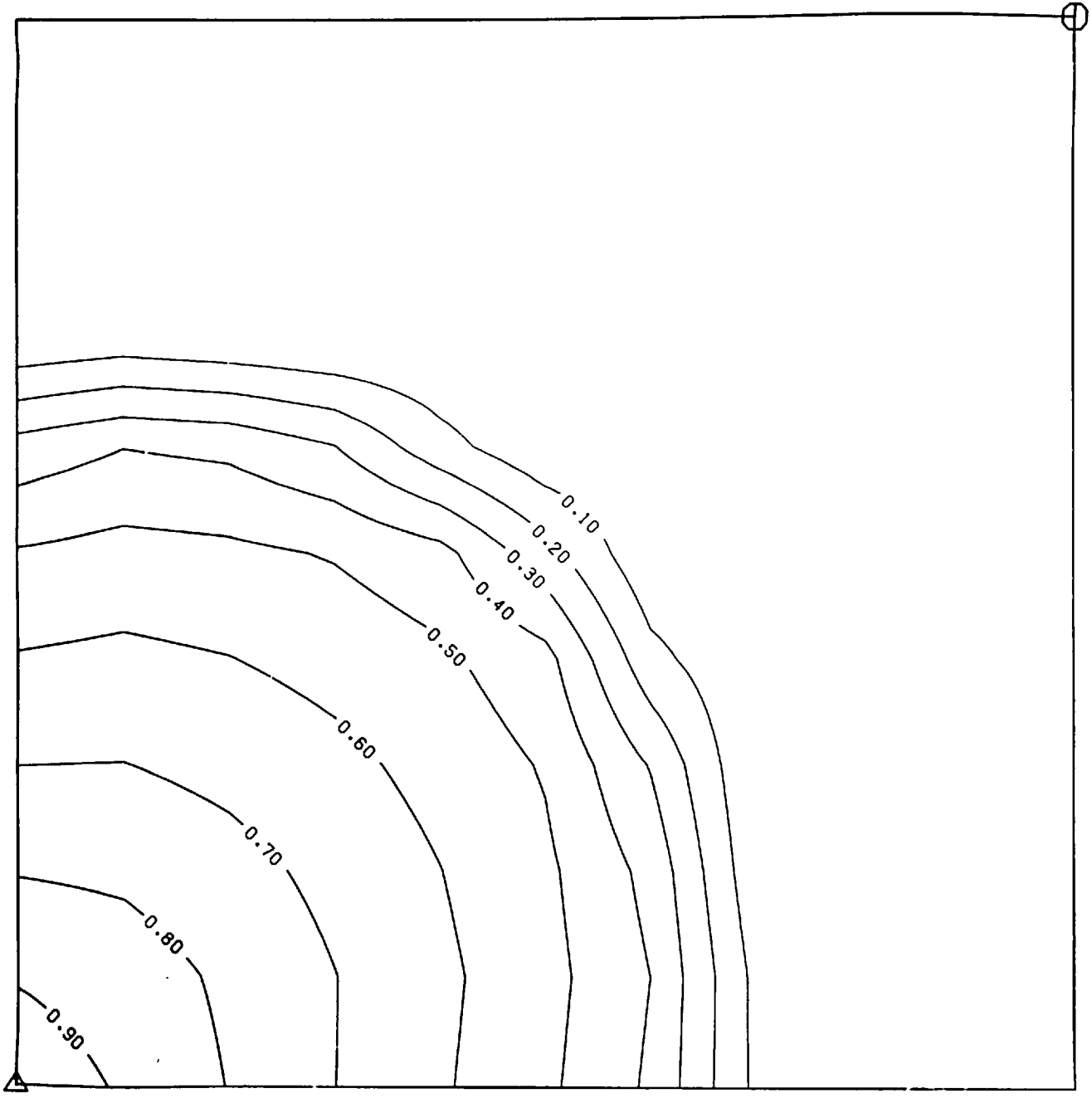

SW MAP.
TIME(D)=1080
PV INJ =0.216
PV REC =0.216
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90

FIGURE G34-5

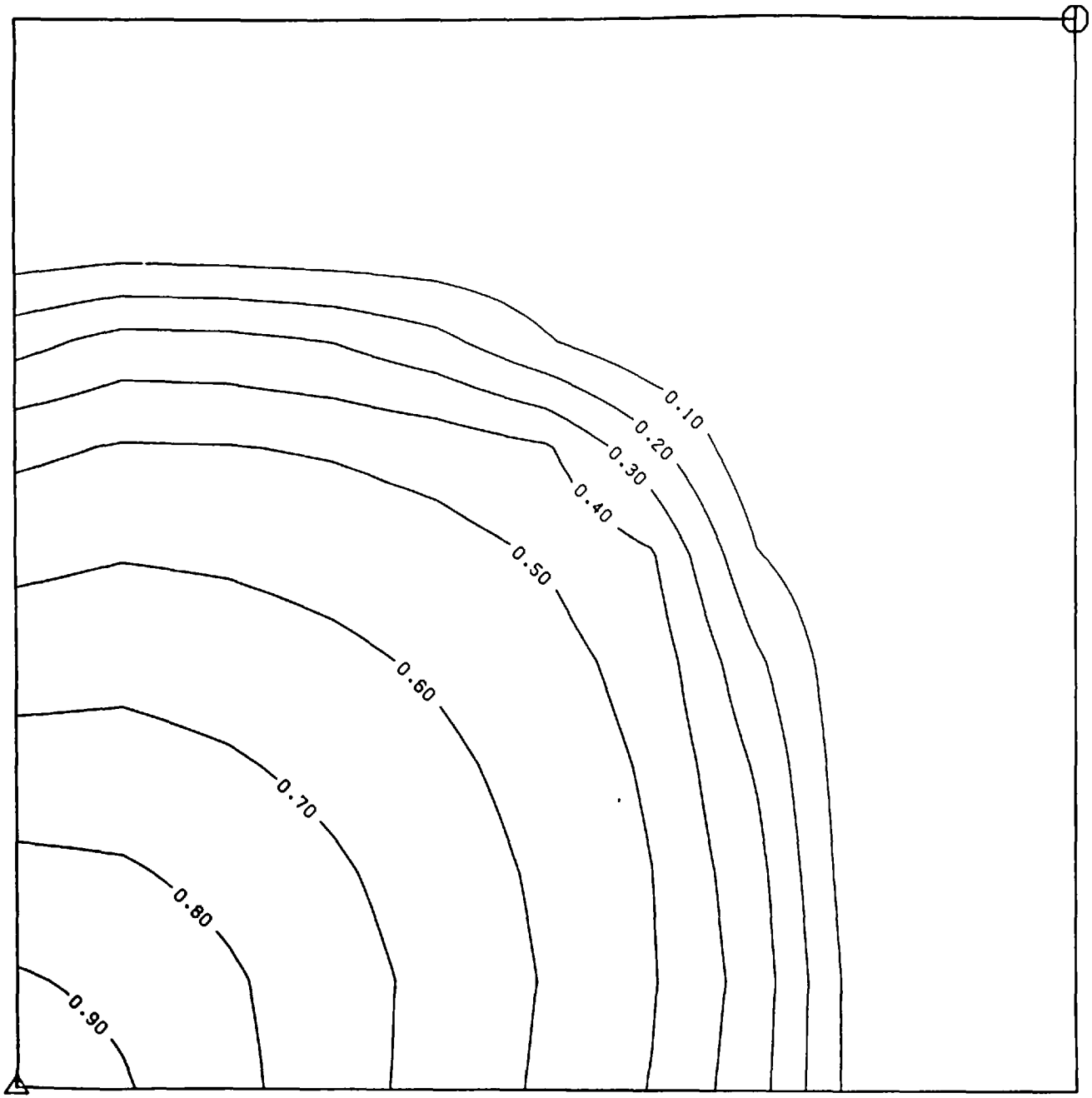

SW MAP.
TIME(D)=1440
PV INJ =0.288
PV REC =0.288
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90

FIGURE G34.6

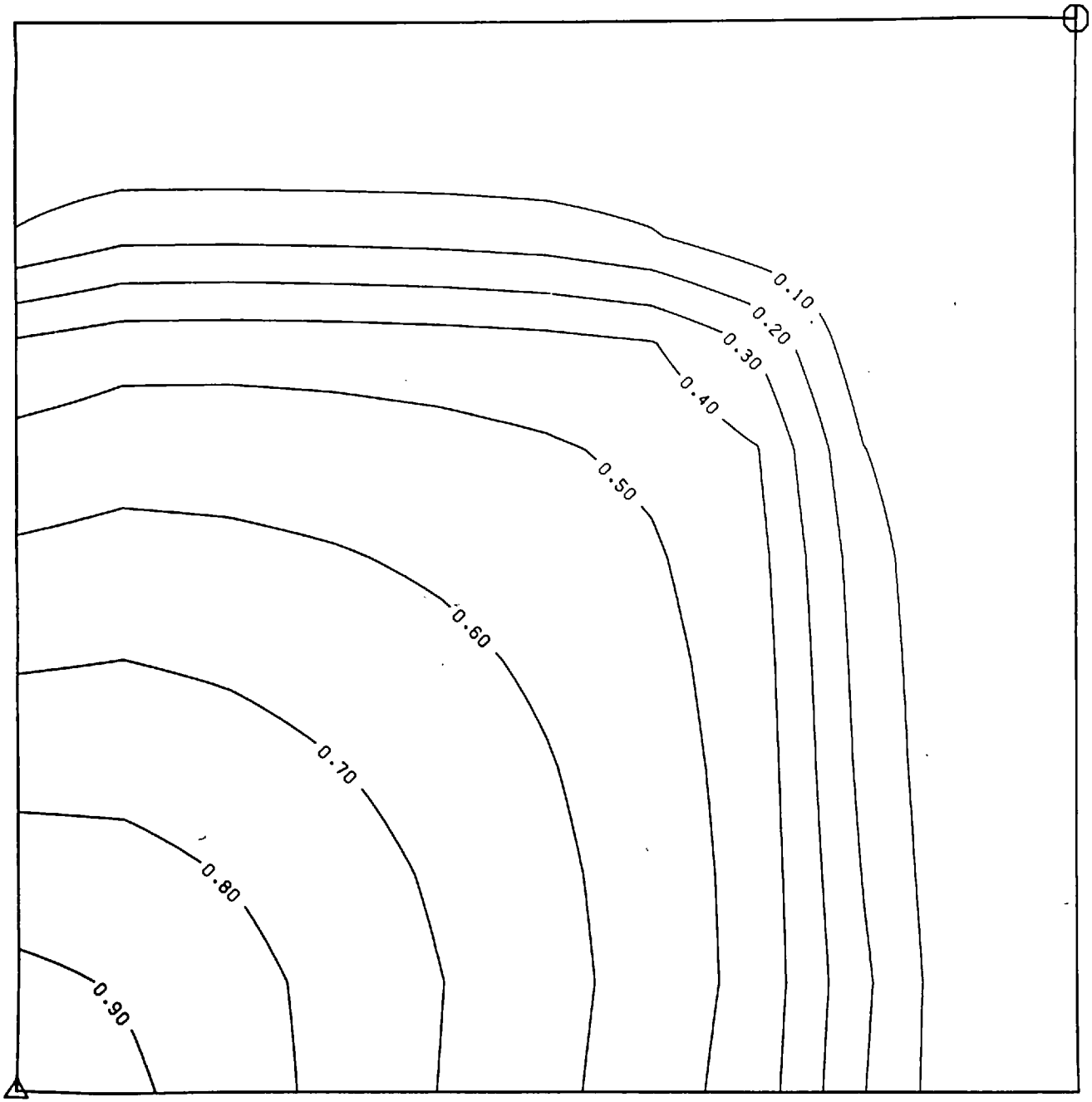

SW MAP.
TIME(D)=1800
PV INJ =0.360
PV REC =0.360
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90

FIGURE G34-7

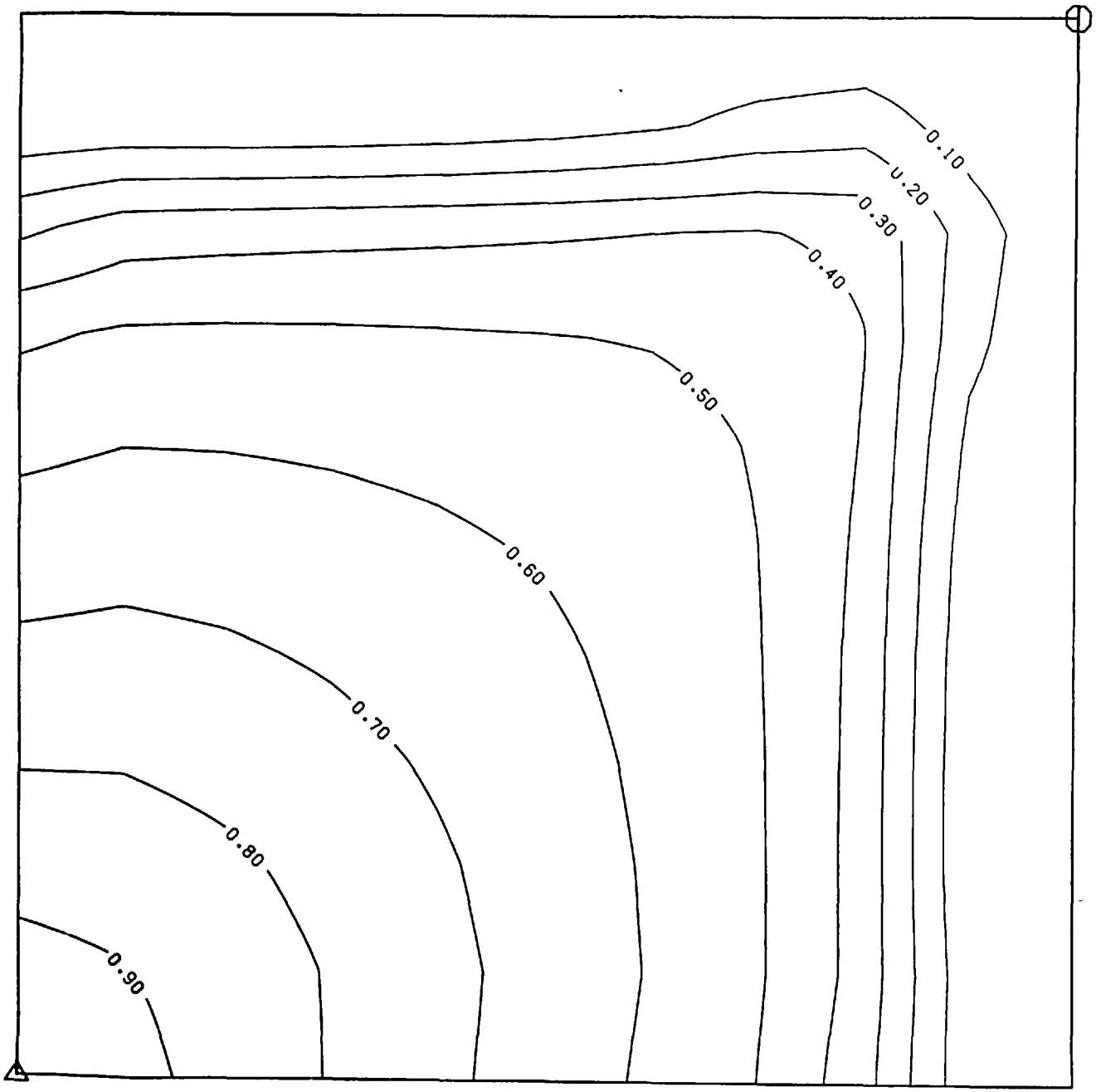
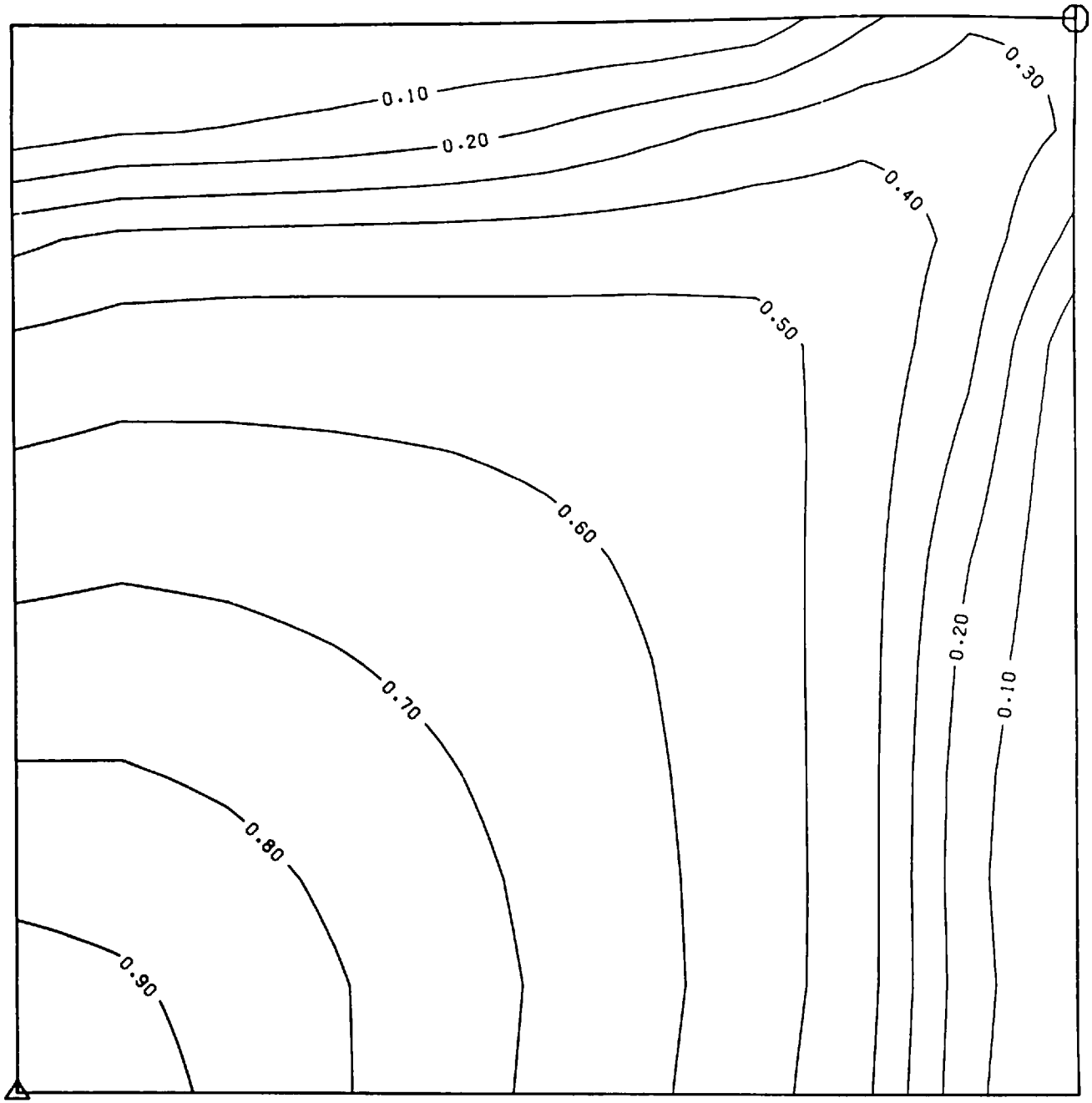

SW MAP.
TIME(D)=2160
PV INJ =0.432
PV REC =0.432
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90

FIGURE G34.8



SW MAP.
TIME(D)=2520
PV INJ =0.504
PV REC =0.493

FIGURE G34.9

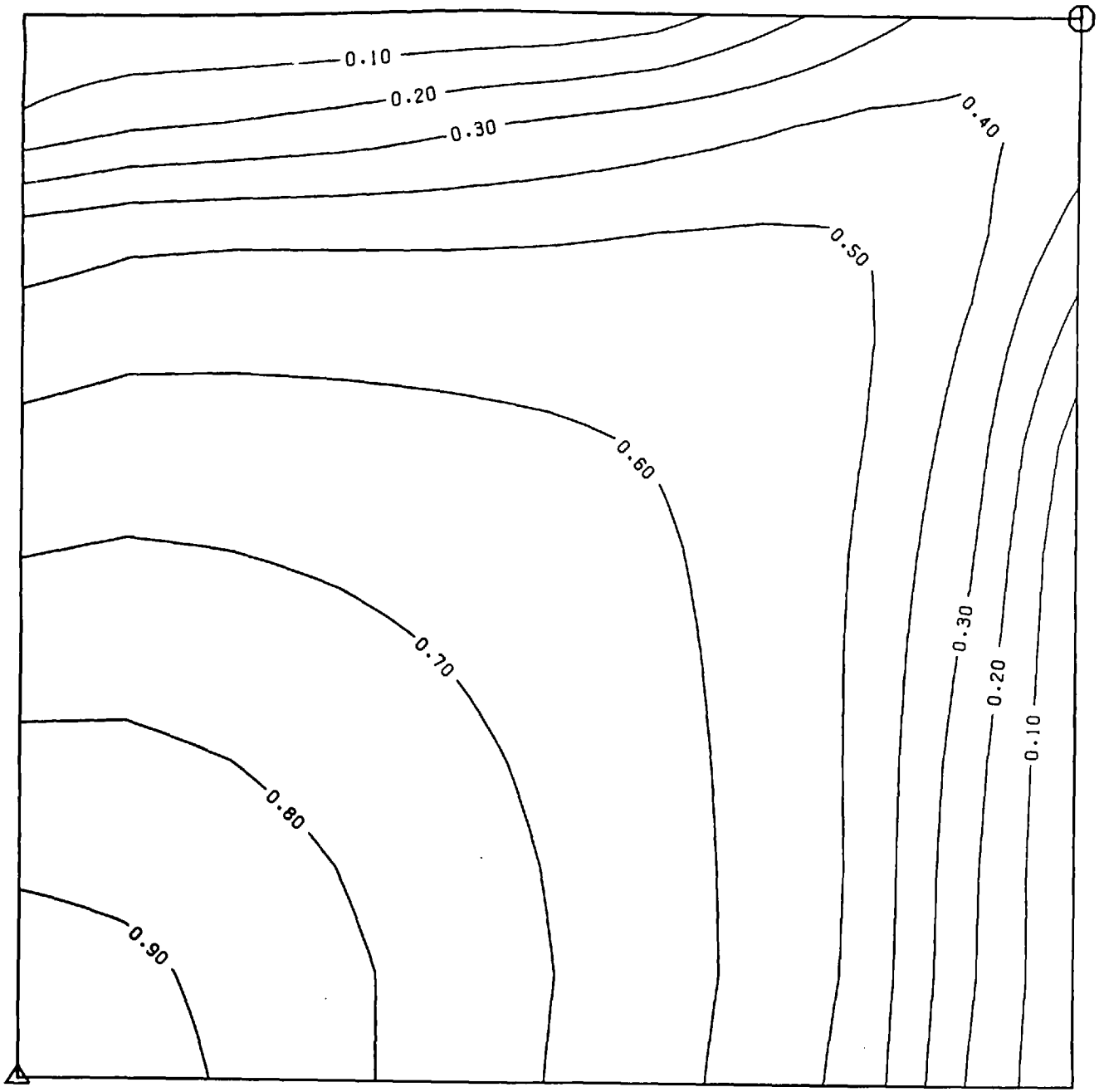


FIGURE G34.10

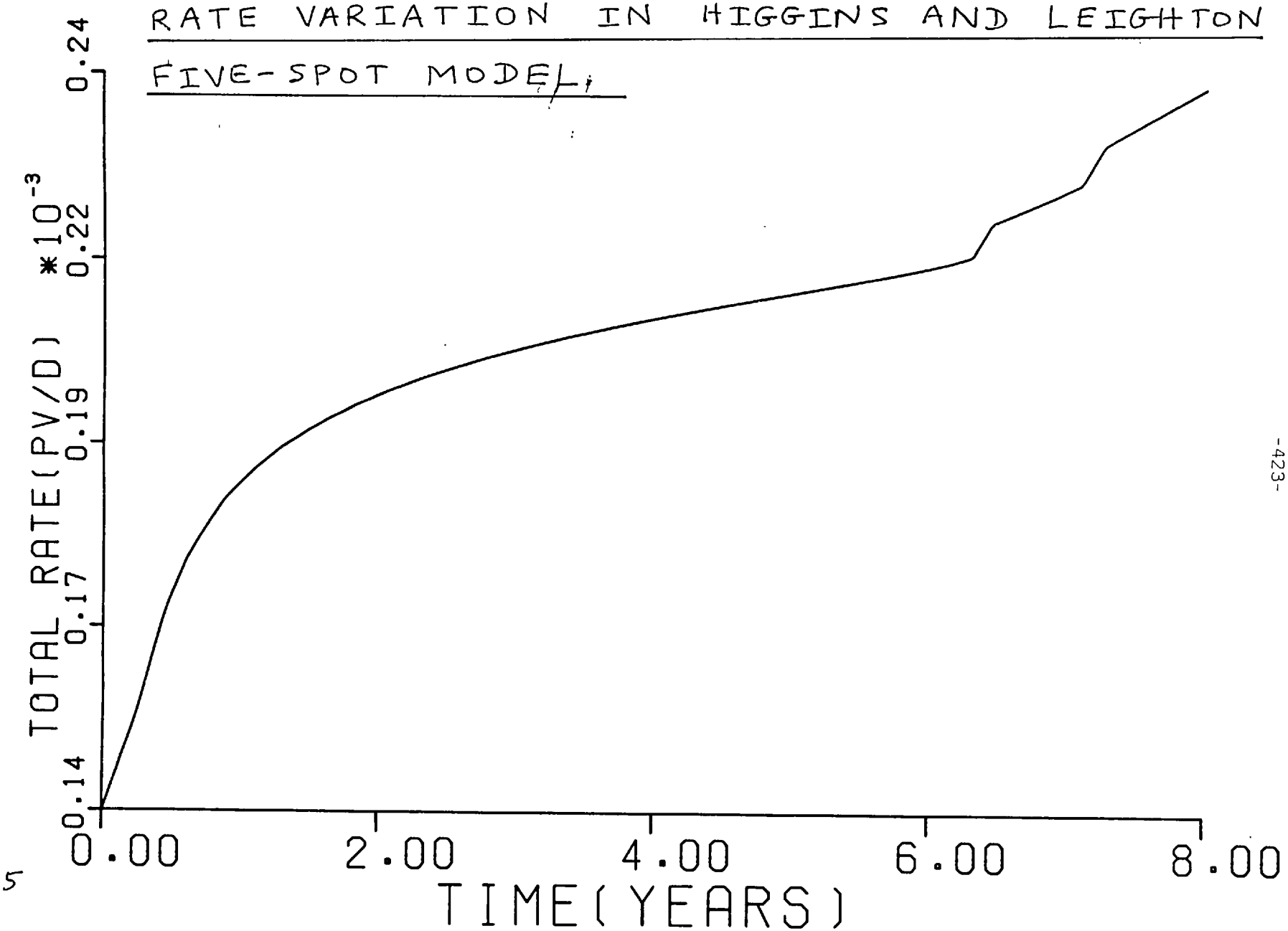

RATE VARIATION IN HIGGINS AND LEIGHTON
FIVE-SPOT MODEL
TOTAL RATE(PV/D) *10^-3
0.14
0.17
0.19
0.22
0.24
0.00
2.00
4.00
6.00
8.00
TIME(YEARS)

FIGURE G35

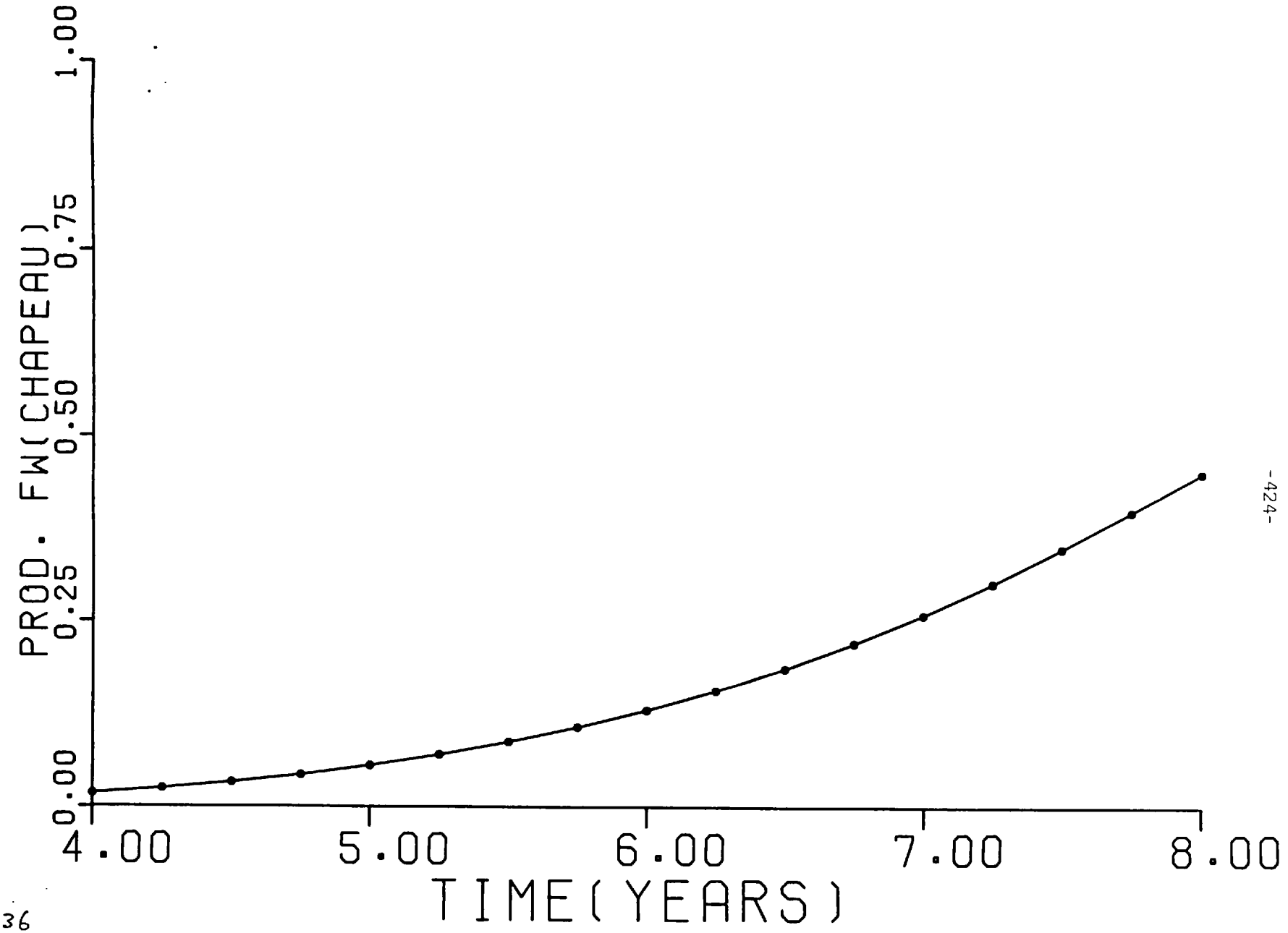

PROD. FW(CHAPEAU)
0.00
0.25
0.50
0.75
1.00
4.00
5.00
6.00
7.00
8.00
TIME(YEARS)

FIGURE G36

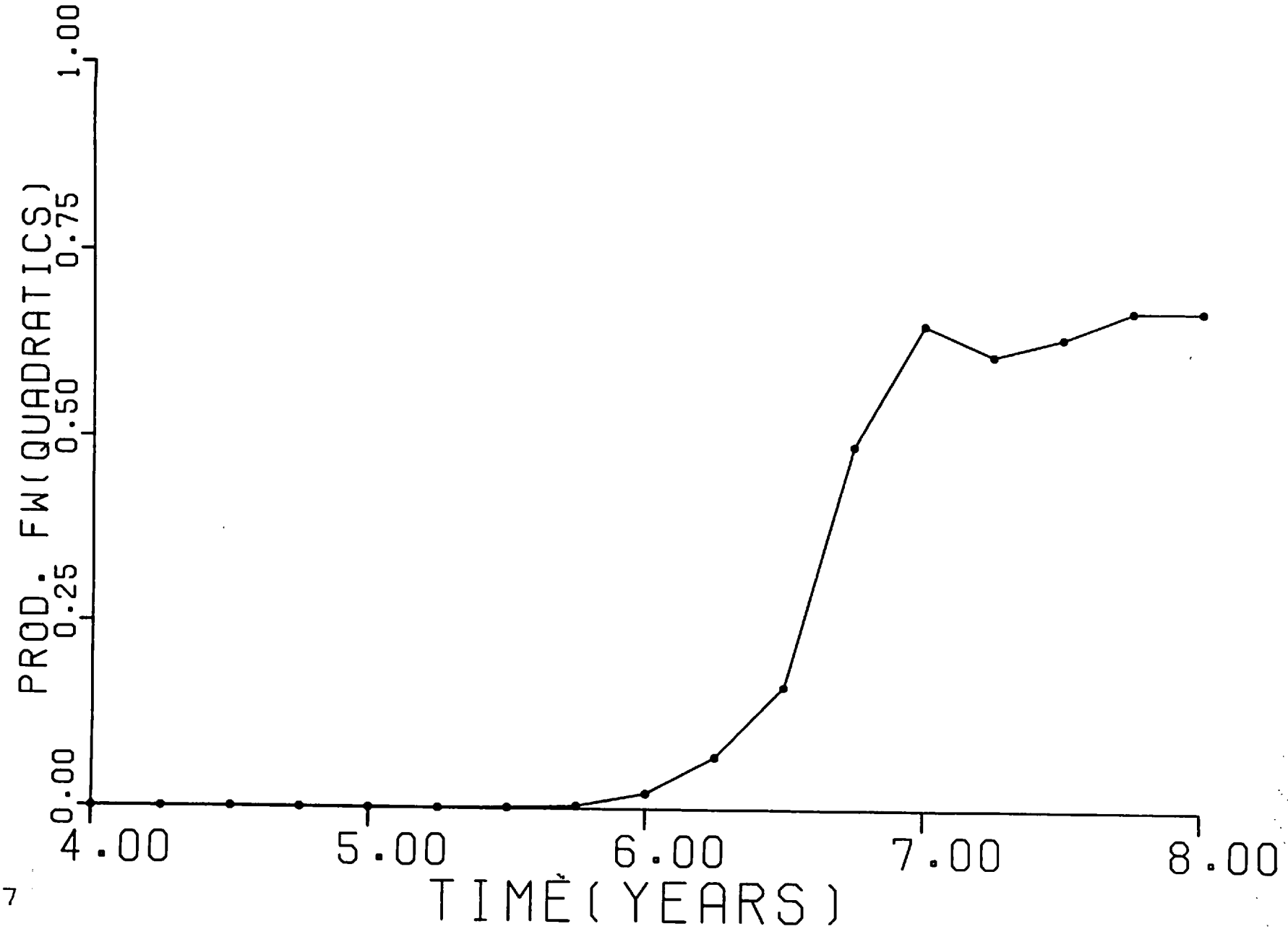

PROD. FW(QUADRATICS)
0.00
0.25
0.50
0.75
1.00
4.00
5.00
6.00
7.00
8.00
TIME(YEARS)

FIGURE G37

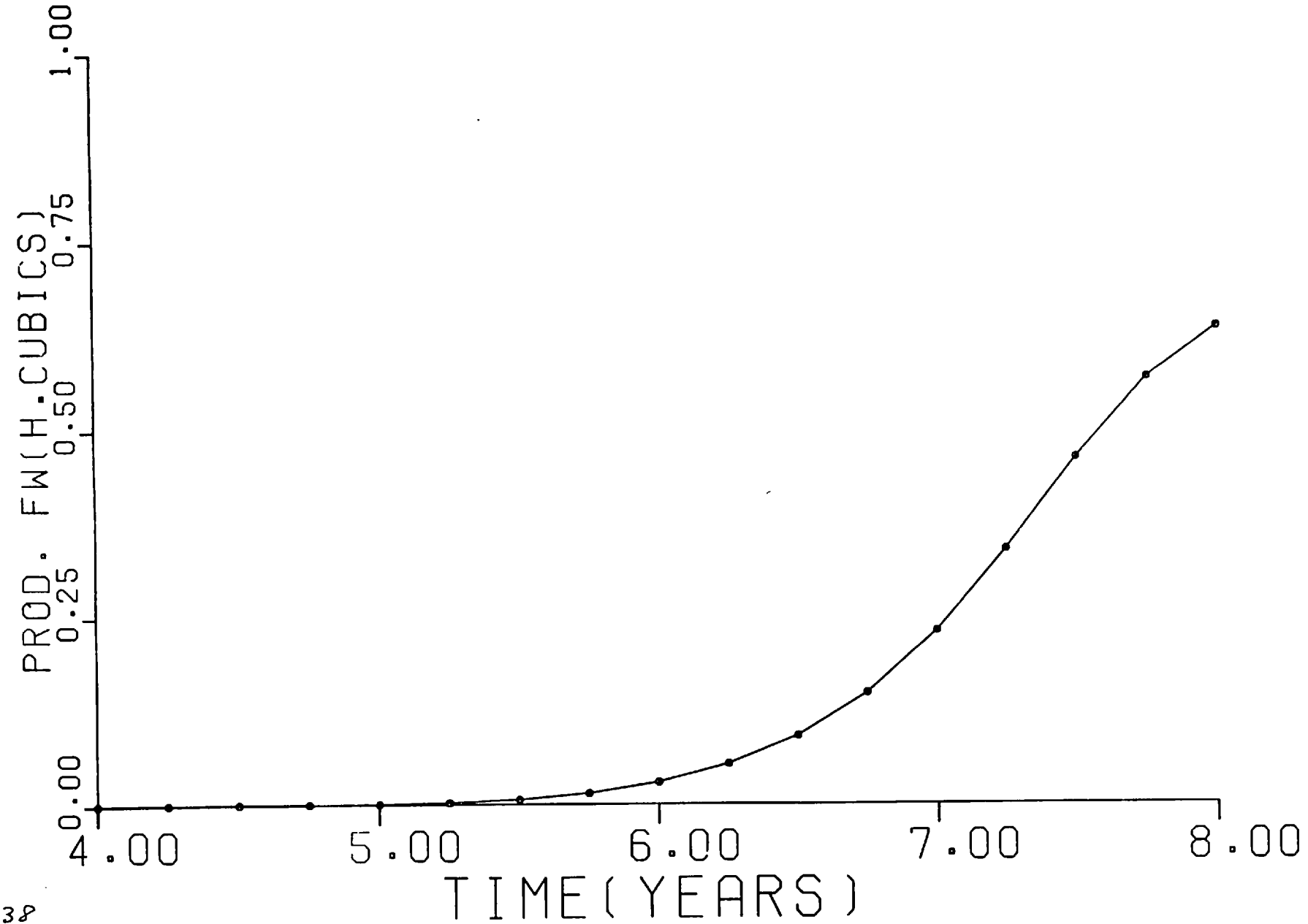

PROD. FW(H.CUBICS)
0.00
0.25
0.50
0.75
1.00
4.00
5.00
6.00
7.00
8.00
TIME(YEARS)

FIGURE G38

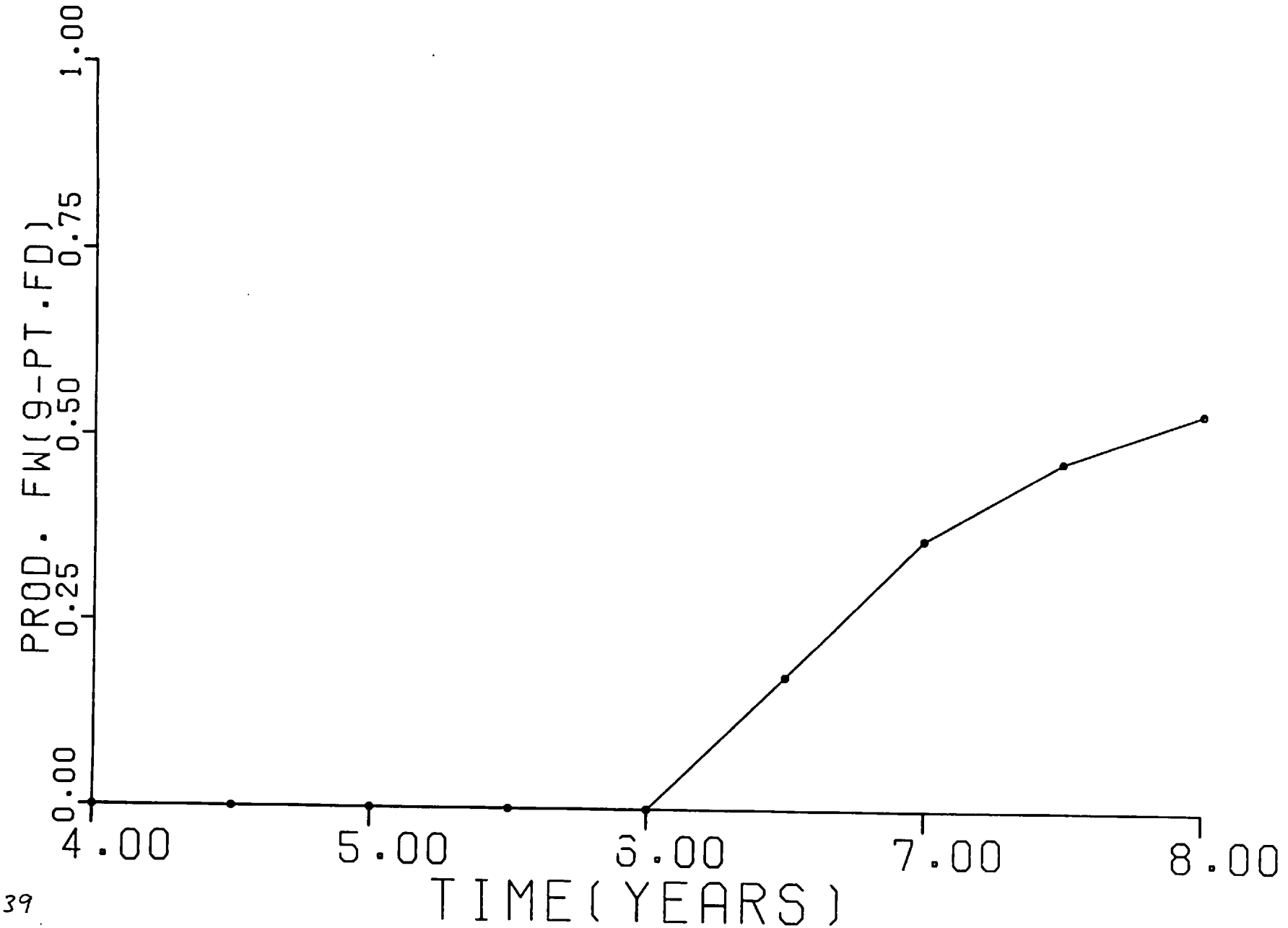

PROD. FW(9-PT.FD)
0.00
0.25
0.50
0.75
1.00
4.00
5.00
6.00
7.00
8.00
TIME(YEARS)

FIGURE G39

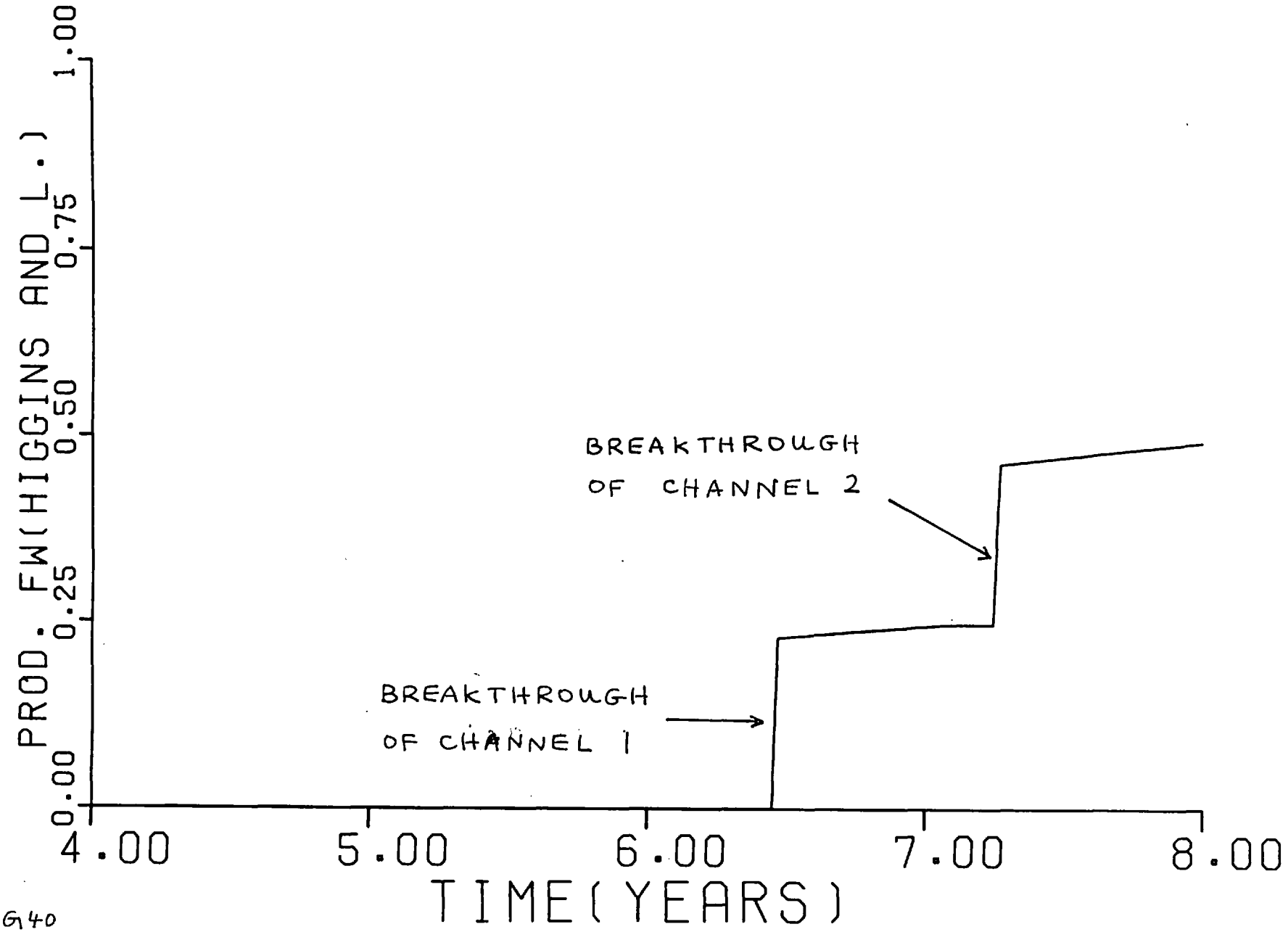

PROD. FW(HIGGINS AND L.)
0.00
0.25
0.50
0.75
1.00
4.00
5.00
6.00
7.00
8.00
TIME(YEARS)
BREAKTHROUGH OF CHANNEL 2
BREAKTHROUGH OF CHANNEL 1

FIGURE G40

REFERENCES

1. ALBRIGHT, N., CONCUS, P. and PROSKUROWSKI, W. (1979) - "Numerical solution of the multi-dimensional Buckley-Leverett equation by a sampling method". Paper SPE 7681 presented at the 5th SPE Symposium on Reservoir Simulation, Denver.

2. ARGYRIS, J.H. (1965). "Continua and Discontinua". Proceedings of the Conference on Matrix Methods in Structural mechanics, Wright Patterson Air Force Base, Ohio.

3. ARONOFSKY, J. (1952) "Mobility ratio, its influence on flood patterns during water encroachment". Trans. AIME. 195, pp 15 - 24.

4. AZIZ, K. and SETTARI, A. (1979). "Petroleum reservoir simulation". Applied Science Publishers.

5. BAZELEY, G.P., CHEUNG, Y.K., IRONS, B.M. and ZIENKIEWICZ, O.C. (1965). "Triangular elements in bending - conforming and non-conforming solutions". Proceedings of the Conference on matrix methods in structural Mechanics, Wright Patterson Air Force Base, Ohio.

6. BOGNER, F.K., FOX, R.L. and SCHMIT, L.A. (1965). "The generation of interelement - compatible stiffness and mass matrices by the use of interpolation formulae". Proceedings of the Conference on matrix methods in structural mechanics, Wright Patterson Air Force Base, Ohio.

7. BUCKLEY, S.E. and LEVERETT, M.C. (1942). "Mechanism of fluid displacements in sands". Trans. AIME, 146, pp 107 - 116.

8. CARDWELL, W.T. Jr. (1959). "The meaning of the triple value in non-capillary Buckley-Leverett theory". Trans AIME., 216, pp 271 - 276.

9. CAVENDISH, J.C., PRICE, H.S., and VARGA, R.S. (1969) "Galerkin methods for the numerical solution of boundary value problems". Soc. Pet. Eng. J.,9 (2) pp. 204 - 220.

10. CHASE, C.A. (1979). "Variational simulation with numerical decoupling and local mesh refinement". Paper SPE 7680 presented at the 5th SPE Symposium on Reservoir Simulation, Denver.

11. CHESHIRE, I.M. and CURTIS, A.R. (1979). "The effect of capillary pressure on the saturation distribution for two phase radial flow near an injection well". Computer Science and systems division, AERE Harwell, Didcot, Oxfordshire.

12. CONNOR, J.J. and BREBBIA, C.A. (1976). "Finite element techniques for fluid flow" London, Newnes-Butterworths.

12a. IBID., p. 127.

13. CRAIG, F.F. Jr. (1971). "The reservoir engineering aspects of waterflooding". Monograph Volume 3, Society of Petroleum Engineers of AIME.

13a. IBID., p.49

13b. IBID., pp. 45-46 and Appendix E.

13c. IBID., p. 122, Figure E.6.

14. CRAIG, F.F. Jr., GEFFEN, T.M. and MORSE, R.A.(1955). "Oil recovery performance of pattern gas or water injection operations from model tests". Trans AIME., 204, pp 7-15.

15. CRICHLOW., H.B. (1977). "Modern reservoir engineering - a simulation approach". Prentice Hall.

15a. IBID., p. 109 et seq.

16. DALEN, V. (1975). "Numerical solutions of the Buckley-Leverett problem". SINTEF Report STF71 A75024, Trondheim.

16a. IBID., pp. 35 - 37

17. DALEN, V. (1975). "A finite element model for the analysis of waterflood performance". SINTEF Report STF 71 A75036, Trondheim.

18. DAWE., R.A. (1979). "The obtaining of oil from an oil reservoir". The School Science Review v. 60 (No.213) pp 642-658.

19. DESAI, C.S. and ABEL. J.F. (1972). "Introduction to the finite element method; a numerical method for engineering analysis". New York, Van Nostrand Reinhold.

20. DOGRU, A.H., ALEXANDER, W.C. and PANTON, R.L. (1976) "Numerical solution of slightly compressible single phase transient flow problems by spline functions". Paper SPE 5733 presented at the 4th SPE Symposium on Numerical Simulation, Los Angeles.

21. DOUGHERTY, E.L. and SHELDON, J.W. (1964). "The use of fluid-fluid interfaces to predict the behaviour of oil recovery processes". Soc. Pet. Eng. J. 4 (2) pp 171-182.

22. DOUGLAS, J. Jr., DUPONT, T. and RACHFORD, H.H. (1969) "The Application of variational methods to waterflooding problems". J. Cdn. Pet. Tech. 8 (3), pp. 79-85.

23. DYES, A.B., CAUDLE, B.H. and ERICKSON, R.A. (1954) "Oil production after breakthrough as influenced by mobility ratio". Trans. AIME., 201, pp. 81-86.

24. FAUST, C.R. and MERCER, J.W. (1976). "An analysis of finite difference and finite element techniques for geothermal reservoir simulation". Paper SPE5742 presented at the 4th SPE Symposium on Numerical Simulation, Los Angeles

25. FAYERS, F.J. and SHELDON, J.W. (1959). "The effect of capillary pressure and gravity on two-phase fluid flow in a porous medium". Trans. AIME, 216, pp. 147-155.

26. GLUCK, H. (1977). "The Calcomp graph plotting facility". ICCC Bulletin 4.6/1, 5th version.

27. GLUCK, H. (1978). "Microfilm software at Imperial College". ICCC Bulletin 4.8/1, 5th version.

28. GOLDWATER, M.H., COLLINS, P.A. and TAYLOR, B.A.(1978) "The use of GRASP, a finite element program, to model faulted gas reservoirs of the southern North Sea Basin". Paper EUR 87 presented at the European Offshore Petroleum Conference and Exhibition, London.

29. GUTHRIE, R.K. and GREENBERGER, M.H. (1955) "The use of multiple correlation analysis for interpreting pertroleum engineering data". Drilling and Production Practice, API, pp. 130-137.

30. HIGGINS, R.V. and LEIGHTON, A.J. (1962). "A computer method to calculate two-phase flow in any irregularly bounded porous medium". Jour. Pet. Tech. 14 (6), pp. 679-683.

31. HIGGINS, R.V., BOLEY, D.W. and LEIGHTON, A.J. (1964) "Aids to forecasting the performance of waterfloods". Jour. Pet. Tech. 16 (9) pp. 1076-1082.

32. IRONS, B.M. (1970). "A frontal solution program for finite element analysis". Int. J. Num. Meth. Eng. 2 (1), pp. 5-32.

33. JAVANDEL, I. and WITHERSPOON, P.A. (1968). "Application of the finite element method to transient flow in porous media". Soc. Pet. Eng. J., 8 (3), pp.241-252.

34. JONES, S.C. and ROSZELLE, W.O. (1978). "Graphical techniques for determining relative permeability from displacement experiments". Jour. Pet. Tech, 30 (5), pp. 807-817.

35. KEBAILI, A.L. and THOMAS, G.W. (1976). "A two-phase coning model using alternating direction Galerkin procedure". Paper SPE 5724 presented at the 4th SPE Symposium on Numerical Simulation, Los Angeles.

36. LANGSRUD, O. (1976). "Simulation of two-phase flow by finite element methods". Paper SPE 5725 presented at the 4th SPE Symposium on Numerical Simulation, Los Angeles

37. LEVERETT, M.C. (1941). "Capillary behaviour in porous solids". Trans AIME, 142, pp 152-169.

38. LEWIS,.R.W., VERNER, R.A. and ZIENKIEWICZ, O.C.(1975) "A finite element approach to two-phase flow in porous media"; in "Finite Elements in Fluids", Vol 1, Edited by Gallagher, R.H. et al; John Wiley and Sons, pp. 183-199.

39. McMICHAEL, C.L. and THOMAS, G.W. (1973) "Reservoir simulation by Galerkin's method". Soc. Pet. Eng. J., 13 (3), pp. 125-138.

40. MONRO. D.M. (1977). "A contouring package for regular or irregular data". ICCC Bulletin 4.8/3, 3rd version.

41. MOREL-SEYTOUX, H.J. (1965). "Analytical - numerical method in waterflooding predictions". Soc. Pet. Eng. J. 5 (3) pp. 247-258.

42. MURPHY, P.J. (1977). "Flood front tracking". M.Sc. Thesis, Imperial College, University of London.

43. MUSKAT, M. (1946). "The flow of homogeneous fluids through porous media". Michigan, J.W. Edwards, Inc.

43a. IBID., pp. 459-461.

43b. IBID., p. 190.

43c. IBID., p. 525.

43d. IBID., p. 526, Figure 201.

43e. IBID., p. 586, equation (1)

43f. IBID., p. 577, Figure 236.

43g. IBID., p. 587, equation (4)

43h. IBID., p. 596.

43i. IBID., p. 597 equation (3)

44. NOLEN, J.S. and BERRY, D.W. (1972). "Tests of the stability and time-step sensitivity of semi-implicit reservoir simulation techniques". Trans AIME, 253, pp. 253-266.

45. ODEN, J.T. (1969). "A general theory of finite elements - I. Topological considerations". Int. J. Num. Meth. Eng., 1 (2), pp. 205-221.

46. ODEN, J.T. (1969), "A general theory of finite elements - II Applications" Int. J. Num. Meth. Eng., 1 (3), pp.247-259.

47. PEACEMAN,D.W. (1977). "Fundamentals of numerical reservoir simulation". Amsterdam, Elsevier.

48. RABY, R.E. (1978). "MATMAP - Matrix mapping package". ICCC Bulletin 6.3/8, 3rd version.

49. RACHFORD, H.H. Jr. (1976). "A sampling of variational methods". Paper SPE 5720 presented at the 4th SPE Symposium on Numerical Simulation, Los Angeles.

50. SETTARI, A. PRICE, H.S. and DUPONT, T. (1977)
"Development and application of variational methods for the simulation of miscible flow in porous media". Trans. AIME., 263 , pp. 228-246.

51. SHELDON, J.W. and DOUGHERTY, E. L. (1964). "A numerical method for computing the dynamical behaviour of fluid - fluid interfaces in permeable media". Soc. Pet. Eng. J., 4 (2), pp. 158 - 170.

52. SHELDON, J.W., ZONDEK, B. and CARDWELL, W.T. Jr. (1959)
"One dimensional, incompressible, non-capillary, two-phase fluid flow in a porous medium". Trans. AIME., 216 , pp. 290-296.

53. SLOBOD, R.L. and CAUDLE, B.H. (1952). "X-ray shadowgraph studies of areal sweepout efficiencies"; Trans. AIME, 195, pp. 265-270.

54. SPIVAK, A., PRICE, H.S. and SETTARI, A. (1977)
"Solution of the equations for multi-dimensional two-phase, immiscible flow by variational methods". Soc. Pet. Eng. J., 17 (1), pp. 27-41.

55. TODD, M.R., O'DELL, P.M. and HIRASAKI, G.J.(1972)
"Methods for increased accuracy in numerical reservoir simulators" ; Trans. AIME., 253, pp. 515-530.

56. WELGE, H.J. (1952). "A simplified method for computing oil recovery by gas or water drive"., Trans. AIME., 192, pp. 285 - 296.

57. WHITE, I.R. (1978) "Finite element analysis of multi-phase flow in porous media, and its application to reservoir engineering". PhD thesis, University College, Swansea.

58. WHITTAKER, E.J. and WATSON, G.N. (1927). "A course of modern analysis" - 4th edition., Cambridge University Press., p. 137.

59. WILSON, D.C. and CASINADER, P.C. (1978) "The use of finite element models in simulating North Sea waterfloods". Paper EUR 86, presented at the European offshore pertroleum exhibition and conference, London.

60. WRIGHT, R.J., EGBOGAH, E.O. and DAWE, R.A. (1978) Personal communication.

61. YANOSIK, J.L. and McCRACKEN, T.A. (1979). "A nine-point finite difference reservoir simulator for realistic prediction of adverse mobility ratio displacements"., Soc. Pet. Eng. J. 19 (4), pp. 253-262.

62. ZIENKIEWICZ, O.C. (1971). "The finite element method in engineering science"., London, McGraw-Hill.

62a. IBID., pp. 107-110

62b. IBID., pp. 39-40

62c. IBID., p. 9