The Computer Program FourPt (Version 95.01) — A Model for Simulating One-Dimensional, Unsteady, Open-Channel Flow

---

U.S. GEOLOGICAL SURVEY

Water-Resources Investigations Report 97-4016

The Computer Program FOURPT (Version 95.01) — A Model for Simulating One-Dimensional, Unsteady, Open-Channel Flow

by Lewis L. DeLong, David B. Thompson, and Jonathan K. Lee

U.S. GEOLOGICAL SURVEY

Water-Resources Investigations Report 97-4016

Bay St. Louis, Mississippi
1997

DEPARTMENT OF THE INTERIOR

BRUCE BABBITT, Secretary

U.S. GEOLOGICAL SURVEY

GORDON P. EATON, Director

For additional information write to:
Project Chief Surface-Water Hydraulics Project
U.S. Geological Survey
Building 2101
Stennis Space Center, MS 39529

Copies of this report can be purchased from:
U.S. Geological Survey
Branch of Information Services
Box 25286
Denver, CO 80225-0286

Contents

Abstract **1**

Introduction **2**

Governing equations **3**

Numerical solution **6**

Numerical integration in time 8

Numerical integration in space 8

Linearization over a time step 9

Interpolation in space 12

Direct matrix solution 15

Computer code **16**

Input **18**

Program control 18

Schematic description 26

Channel properties 30

Constraint properties 35

Three-parameter constraint 35

User-programmable constraint 37

File names and unit numbers 39

Initial values 39

Boundary values 42

Density values 43

Perturbation parameters 45

Output **48**

General results 48
Restart 48
Space series 48
Time series 50

Examples **51**

Example 1 – Steady flow in a channel with expanding width 51
Example 2 – Unsteady flow in a prismatic channel 52
Example 3 – Diffusion and kinematic wave approximations 54
Example 4 – Unsteady flow in a simple network of channels 59
Example 5 – Applying user-programmed constraints 59

References **66**

List of Figures

1 Computational grid. 6
2 Linear interpolation functions. 12
3 Example program-control data file. 19
4 Sinusoidal component used in optional computation of boundary values from equations. 25
5 Partial example schematic-data file, four channels of a six-channel network. 28
6 Example rectangular cross-section data file. 30
7 Example trapezoidal cross-section data file. 31
8 Example irregular cross-section data file. 33
9 Example three-parameter constraint file. 36
10 Example master file-name and unit-number file. 40
11 Example initial-values file. 41
12 Example boundary-values file. 43
13 Example density-values file for a two-channel network. 44

14 Example perturbation-parameter file. 45
15 Example general-results file. 49
16 Example tab-delimited space-series file. 50
17 Example tab-delimited time-series file. 50
18 Example-1 schematic-data file. 51
19 Spatial convergence in a channel with expanding width. 52
20 Example-2 program-control-data file. 53
21 Example-2 schematic-data file. 54
22 Spatial and temporal convergence in a prismatic channel, at a location 50,000 ft downstream. 55
23 Impact of θ on numerical convergence. 56
24 Comparison of dynamic- and diffusion-wave formulations on steady flow in the nonprismatic channel of example 1. 57
25 Failure of the kinematic approximation to reach steady flow when applied to the nonprismatic channel of example 1. 58
26 Different solution formulations and the prismatic channel section. 58
27 Schematic representation of a simple six-channel network. 59
28 Flow in a simple six-channel network. 60
29 Example-4 schematic-data file. 61
30 Example-4 schematic-data file continued. 62
31 Example-5 schematic-data file. 64
32 Example-5 constraint-properties file. 65
33 Inflow to reservoir and flow at dam for example 5. 66
34 Relation of water-surface elevation to discharge for example 5 (first 20 hours of simulation). 67

List of Tables

1 FORTRAN 77 modules used in FourPt. 17
2 Main-program and utility files used in FourPt. 17
3 Input files. 18
4 Output files. 48

FACTORS FOR CONVERTING INCH-POUND UNITS TO INTERNATIONAL SYSTEM (SI) UNITS

Multiply inch-pound units	By	To obtain SI units
foot (ft)	0.3048	meter
mile (mi)	1.609	kilometer
cubic foot per second (ft^3/s)	0.02832	cubic meter per second

The Computer Program FourPt(Version 95.01) — a Model for Simulating One-Dimensional, Unsteady, Open-Channel Flow

by Lewis L. DeLong, David B. Thompson, and Jonathan K. Lee

ABSTRACT

FourPt is a computer program capable of computing unsteady one-dimensional flow in networks of open-channels. Optionally, governing equations may represent dynamic-wave, diffusion-wave, or kinematic-wave equations and may include variable channel sinuosity and variable water density. If density is allowed to vary, density values must be supplied by the user as this version of FourPt does not currently compute transport or heat flow. Governing equations are converted to a set of linearized equations using a four-point-implicit finite-difference method with Newton-Raphson iteration. The resulting set of simultaneous equations are solved directly by Gaussian elimination.

Optional boundary conditions include known flow, water-surface elevation, water-surface slope, and three-parameter ratings. Additionally, a facility exists for users to program their own boundary constraints. Logarithmic and semi-logarithmic stage-discharge relations and dam-break functions also may be used as boundary constraints and are included in the program code as examples of user-programmed boundary constraints.

Channel cross sections may be represented by rectangular, trapezoidal, or irregular geometry, depending on which geometry routines are linked with the program code. Spacing of user-supplied channel cross sections is independent of the selection of spatial discretization for numerical solution.

FourPt is written in FORTRAN 77 using a data-encapsulation programming paradigm.

This report describes data input and output and demonstrates selected model capabilities with examples. The computer code and example data sets are available for electronic retrieval via World Wide Web from `http://water.usgs.gov/software/FourPt.html` or anonymous File Transfer Protocol from `/pub/software/surface_water/FourPt`.

INTRODUCTION

This report summarizes the formulation and use of FOURPT, version 95.01, a computer program for simulating one-dimensional, unsteady, open-channel flow. It is written in FORTRAN77 (American National Standards Institute 1978) using FORTRAN modules (DeLong, Thompson, and Fulford 1992; Thompson, DeLong, and Fulford 1992).

The primary purpose of version 95.01 is to provide a computer code capable of demonstrating nontrivial concepts important to the simulation of unsteady open-channel flow and yet is easy to read, modify, and run on a variety of computer systems. This report describes the governing equations, numerical formulation, computer code, input and output, and presents a set of examples selected to demonstrate concepts important to the numerical simulation of unsteady open-channel flow. The computer code and example data sets are available for electronic retrieval via World Wide Web from `http://water.usgs.gov/software/FourPt.html` or anonymous File Transfer Protocol from `/pub/software/surface_water/FourPt`.

Sections in this report cover distinct topics in a logical progression but may be referenced autonomously. For example, a reader interested only in testing the FOURPT code on a specific computer might first refer to sections *Computer Code* and *Input*.

GOVERNING EQUATIONS

The governing equations for FOURPT describe one-dimensional, unsteady, open-channel flow. They allow density to vary with time and location and effective channel length to vary with depth of flow. In differential form, the mass and momentum conservation equations are written, respectively,

$$\frac{\partial}{\partial t}\left(\rho M_a A\right) + \frac{\partial}{\partial x}\left(\rho Q\right) - \rho_\ell q = 0, \tag{1}$$

and

$$\frac{\partial}{\partial t}\left(\rho M_q Q\right) + \frac{\partial}{\partial x}\left(\beta\rho\frac{Q^2}{A} + \rho g I_1\right) + \rho g A\left(S_0 + S_f\right) - \rho g I_2 = 0, \tag{2}$$

where

$$I_1 = \int_0^h \left(h - \eta\right)\sigma d\eta, \tag{3}$$

and

$$I_2 = \int_0^h \left(h - \eta\right)\frac{\partial\sigma}{\partial x} d\eta, \tag{4}$$

in which

t = time,

ρ = density,

A = cross-sectional area,

M_a = area-weighted sinuosity coefficient,

x = downstream reference distance,

Q = volumetric discharge,

ρ_ℓ = density of lateral inflow,

q = lateral inflow,

M_q = flow-weighted sinuosity coefficient,

β = momentum coefficient,

g = acceleration due to gravity,

S_0 = channel-bottom slope,

S_f = friction slope,

h = depth of flow,

η = depth-integration variable, and

σ = width of channel.

Equations 1 through 4 are similar to those presented by Cunge and others (1980) with two exceptions: they have been extended to include the volumetric effects of sinuosity with the inclusion of metric coefficients, M_a and M_q (DeLong 1986), and density is assumed uniform in cross section but not necessarily constant with stream distance or time. The area-weighted sinuosity coefficient, M_a (DeLong 1989), may vary both with depth of flow and distance and is defined by

$$M_a = \frac{1}{A} \int_A m \, dA, \tag{5}$$

in which, for the increment of cross-sectional area, dA, m is the ratio of channel length, s, to the downstream reference distance, x, expressed as

$$m = \frac{ds}{dx}. \tag{6}$$

The flow-weighted sinuosity coefficient, M_q, may also vary with depth of flow and downstream distance and is similarly defined (Froehlich 1990)

$$M_q = \frac{1}{Q} \int_Q m \, dQ, \tag{7}$$

in which dQ is an increment of discharge corresponding to incremental area, dA. Note, however, that unlike mass, momentum is a vector quantity, the conservation of which can not be rigorously enforced by a single equation describing motion along a meandering streamline. Use of M_q for correction of momentum storage provides only partial correction of momentum errors potentially resulting from the one-dimensional approximation of a meandering stream.

The momentum coefficient, β, is defined by

$$\beta = \frac{1}{V^2 A} \int_A v^2 dA, \tag{8}$$

in which v equals velocity and V equals mean velocity in the cross section.

In equations 1 and 2, it is assumed that flow is one dimensional to the extent that the momentum coefficient can sufficiently account for nonuniform velocity distribution, streamline curvature and accelerations in directions other than the x direction are negligible, effects of turbulence and friction are adequately described by the resistance laws used for steady flow, the channel slope, S_0, is sufficiently mild so that the cosine of its angle with the horizontal is close to unity, and momentum associated with lateral inflow, q, is negligible.

Equation 2 is rearranged and simplified to eliminate the integral relations represented by I_1 and I_2. Substituting I_1 into equation 2 and using Leibnitz's rule (Flanders, Korfhage, and Price 1974) for differentiating an integral results in

$$g \frac{\partial}{\partial x} (\rho I_1) = g \frac{\partial}{\partial x} \left(\rho \int_0^h (h - \eta) \, \sigma d\eta \right) = g \int_0^h \frac{\partial}{\partial x} \left(\rho (h - \eta) \sigma \right) d\eta. \tag{9}$$

Differentiating within the integral and rearranging terms results in

$$g\frac{\partial}{\partial x}\left(\rho I_1\right) = g\rho\int_0^h \sigma\frac{\partial}{\partial x}\left(h-\eta\right)d\eta + g\frac{\partial\rho}{\partial x}\int_0^h \left(h-\eta\right)\sigma d\eta + g\rho\int_0^h \left(h-\eta\right)\frac{\partial\sigma}{\partial x}d\eta, \tag{10}$$

which reduces to

$$g\frac{\partial}{\partial x}\left(\rho I_1\right) = g\rho A\frac{\partial h}{\partial x} + gA\frac{\partial\rho}{\partial x}\bar{z} + g\rho I_2, \tag{11}$$

where $\bar{z}$ is the distance from the water surface to the centroid of the cross section and is defined by

$$\bar{z} = \frac{\int_0^h \left(h-\eta\right)\sigma d\eta}{\int_0^h \sigma d\eta}. \tag{12}$$

Substituting equation 11 into equation 2 results in

$$\frac{\partial}{\partial t}\left(\rho M_q Q\right) + \frac{\partial}{\partial x}\left(\beta\rho\frac{Q^2}{A}\right) + gA\left(\rho S_0 + \rho S_f + \rho\frac{\partial h}{\partial x} + \frac{\partial\rho}{\partial x}\bar{z}\right) = 0. \tag{13}$$

Substituting the relation

$$S_0 + \frac{\partial h}{\partial x} = \frac{\partial Z}{\partial x} \tag{14}$$

into equation 13 results in

$$\frac{\partial}{\partial t}\left(\rho M_q Q\right) + \frac{\partial}{\partial x}\left(\beta\rho\frac{Q^2}{A}\right) + gA\left(\rho\frac{\partial Z}{\partial x} + \rho S_f + \frac{\partial\rho}{\partial x}\bar{z}\right) = 0, \tag{15}$$

where Z is the distance of the water surface above a common datum.

The flow resistance term, S_f, is replaced through substitution of the empirical relation

$$Q = K\sqrt{S_f}, \tag{16}$$

where K is total channel conveyance, resulting in

$$\frac{\partial}{\partial t}\left(\rho M_q Q\right) + \frac{\partial}{\partial x}\left(\beta\rho\frac{Q^2}{A}\right) + gA\left(\rho\frac{\partial Z}{\partial x} + \rho\frac{Q\left|Q\right|}{K^2} + \frac{\partial\rho}{\partial x}\bar{z}\right) = 0. \tag{17}$$

The absolute value in equation 17 forces flow resistance to always oppose flow. Equation 17 is the equation governing momentum conservation in FourPt.

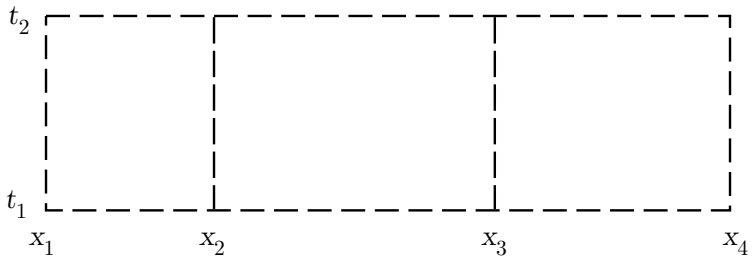

Figure 1: Computational grid.

NUMERICAL SOLUTION

Governing equations are solved numerically using a four-point-implicit method (Preissmann 1961). General analytical solutions to the unsteady, one-dimensional, open-channel flow equations do not exist. In practice, they may be solved by a variety of numerical techniques. One such technique, the four-point-implicit scheme, is the subject of this section. The scheme gets its name from the number of computational points and implicit nature of the algorithm. Because of the shape of the computational grid, it is sometimes referred to as a box scheme. In this numerical technique a continuous river or waterway of interest is divided into discrete reaches. Dependent variables, volumetric discharge, Q, and water-surface elevation, Z, are computed at reach extremities for discrete points in time. A simple example of a computational grid constructed for a single channel extending from x_1 to x_4 is shown in figure 1. The channel has been divided into three discrete reaches and one discrete time increment. In practice, time increments would be added as needed to span the time interval of interest.

The governing equations 1 and 17, respectively, may be integrated over a typical computational reach (figure 1) extending from x_1 to x_2 in space and from t_1 to t_2 in time, resulting in

$$\int_{x_1}^{x_2} \left((\rho M_a A)_{t_2} - (\rho M_a A)_{t_1} \right) dx + \int_{t_1}^{t_2} \left((\rho Q)_{x_2} - (\rho Q)_{x_1} \right) dt - \int_{x_1}^{x_2} \int_{t_1}^{t_2} \rho_\ell \, q \, dx \, dt = 0 \qquad (18)$$

and

$$\begin{aligned} \int_{x_1}^{x_2} \left((\rho M_q Q)_{t_2} - (\rho M_q Q)_{t_1} \right) dx \quad &+ \quad \int_{t_1}^{t_2} \left(\left(\frac{\rho \beta Q^2}{A} \right)_{x_2} - \left(\frac{\rho \beta Q^2}{A} \right)_{x_1} \right) dt \\ &+ \quad \int_{x_1}^{x_2} \int_{t_1}^{t_2} gA \left(\rho \frac{\partial Z}{\partial x} + \rho \frac{Q \left| Q \right|}{K^2} + \frac{\partial \rho}{\partial x} \bar{z} \right) dx \, dt = 0. \end{aligned} \qquad (19)$$

Integral equations 18 and 19 are prepared for iterative numerical solution in four steps:

1. numerical integration in time,

2. numerical integration in space,

3. linearization over a single iteration in terms of incremental change in dependent variables using approximations obtained from truncated Taylor series, and

4. spatial approximation in terms of dependent variables located specifically at computational-reach extremities.

More detailed presentation of these four steps is contained in the following four subsections. For the reach described by equations 18 and 19, the four steps result in determination of coefficients in a corresponding pair of linear equations

$$c_{2,1}\Delta Q_{x_1} + c_{2,2}\Delta Z_{x_1} + c_{2,3}\Delta Q_{x_2} + c_{2,4}\Delta Z_{x_2} = B_2 \tag{20}$$

and

$$c_{3,1}\Delta Q_{x_1} + c_{3,2}\Delta Z_{x_1} + c_{3,3}\Delta Q_{x_2} + c_{3,4}\Delta Z_{x_2} = B_3, \tag{21}$$

where, typical to all variables, ΔQ_{x_1} represents the incremental change, over an iteration, in discharge at x_1.

Assembly of equations resulting from similar operations on governing equations associated with all computational reaches along with appropriate boundary conditions results in a set of linear equations equal in number to the number of dependent variables,

$$\begin{bmatrix} c_{1,1} & & & & & & & \\ c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} & & & & \\ c_{3,1} & c_{3,2} & c_{3,3} & c_{3,4} & & & & \\ & & c_{4,3} & c_{4,4} & c_{4,5} & c_{4,6} & & \\ & & c_{5,3} & c_{5,4} & c_{5,5} & c_{5,6} & & \\ & & & & c_{6,5} & c_{6,6} & c_{6,7} & c_{6,8} \\ & & & & c_{7,5} & c_{7,6} & c_{7,7} & c_{7,8} \\ & & & & & & & c_{8,8} \end{bmatrix} \begin{Bmatrix} \Delta Q_{x_1} \\ \Delta Z_{x_1} \\ \Delta Q_{x_2} \\ \Delta Z_{x_2} \\ \Delta Q_{x_3} \\ \Delta Z_{x_3} \\ \Delta Q_{x_4} \\ \Delta Z_{x_4} \end{Bmatrix} = \begin{Bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \\ B_5 \\ B_6 \\ B_7 \\ B_8 \end{Bmatrix}. \tag{22}$$

The equation set 22 is a simple example corresponding to the single channel schematized in figure 1. More complex networks of interconnected channels would, perhaps, result in hundreds to thousands of equations.

First and last rows of the equation set 22 are obtained from boundary conditions or constraints enforced at the extremities of each channel or branch of a network of channels. The simultaneous set of equations is solved directly for incremental change in dependent variables, dependent variables are adjusted by the incremental change, new coefficients are computed, and the process is repeated until the incremental change in dependent variables falls within acceptable limits. The solution algorithm then advances in time, and the iterative process is repeated for successive time increments. Values of dependent variables are determined at extremities of all computational reaches at points in time bounding each time increment. Derivation of expressions used to compute coefficients and description of the method used to solve equation 22 are contained in the following five subsections.

Numerical Integration in Time

Numerical integration in time is accomplished through the use of a time-weighting parameter, θ. For a typical variable, ϕ,

$$\phi = f(x,t) \tag{23}$$

time integration is accomplished by

$$\int_{t_1}^{t_2} \phi_{x_i} dt \cong \left(\theta \left(\phi_{x_i}\right)_{t_2} + (1-\theta)\left(\phi_{x_i}\right)_{t_1}\right) \Delta t. \tag{24}$$

The weighting parameter θ may vary from 0.5 to 1.0 in the four-point-implicit scheme 4. (Effect of θ is demonstrated in example 2.) Applying similar numerical integration to equations 18 and 19 and moving values known at time t_1 to the right-hand side results in

$$\begin{aligned} \int_{x_1}^{x_2} (\rho M_a A)_{t_2} dx + \theta \Delta t \left((\rho Q)_{x_2} - (\rho Q)_{x_1}\right)_{t_2} &= \int_{x_1}^{x_2} (\rho M_a A)_{t_1} dx \\ -(1-\theta)\Delta t \left((\rho Q)_{x_2} - (\rho Q)_{x_1}\right)_{t_1} + \theta \Delta t \int_{x_1}^{x_2} (\rho_\ell q)_{t_2} dx &+ (1-\theta)\Delta t \int_{x_1}^{x_2} (\rho_\ell q)_{t_1} dx \end{aligned} \tag{25}$$

and

$$\begin{aligned} &\int_{x_1}^{x_2} (\rho M_q Q)_{t_2} dx + \theta \Delta t \left(\left(\frac{\rho \beta Q^2}{A}\right)_{x_2} - \left(\frac{\rho \beta Q^2}{A}\right)_{x_1}\right)_{t_2} \\ &\quad + \theta \Delta t \int_{x_1}^{x_2} \left(gA\left(\rho \frac{\partial Z}{\partial x} + \rho \frac{Q|Q|}{K^2} + \frac{\partial \rho}{\partial x}\bar{z}\right)\right)_{t_2} dx = \\ &\int_{x_1}^{x_2} (\rho M_q Q)_{t_1} dx - (1-\theta)\Delta t \left(\left(\frac{\rho \beta Q^2}{A}\right)_{x_2} - \left(\frac{\rho \beta Q^2}{A}\right)_{x_1}\right)_{t_1} \\ &\quad - (1-\theta)\Delta t \int_{x_1}^{x_2} \left(gA\left(\rho \frac{\partial Z}{\partial x} + \rho \frac{Q|Q|}{K^2} + \frac{\partial \rho}{\partial x}\bar{z}\right)\right)_{t_1} dx. \end{aligned} \tag{26}$$

Numerical Integration in Space

Numerical integration in space is accomplished through the use of a more general quadrature than previously used for time integration and, for a typical variable, ϕ, is expressed by

$$\int_{x_1}^{x_2} \phi_{t_j} dx \cong \Delta x \sum_{k=1}^{n} \omega_k \left(\phi_{t_j}\right)_{\xi_k}, \tag{27}$$

where ω is a weighting function similar in concept to θ used in numerical integration over time. The number, n, and location, ξ_k, of integration points and magnitude of corresponding weights in

general determine accuracy of the approximation. Weights, ω_k, must sum to 1,

$$\sum_{k=1}^{n} \omega_k = 1, \tag{28}$$

and integration locations, ξ_k, must fall within the integration interval, in this instance,

$$x_1 \leq \xi \leq x_2. \tag{29}$$

Applying this form of spatial integration to equations 25 and 26 results in

$$\begin{aligned}
&\Delta x \sum_{k=1}^{n} \left(\omega \left(\rho M_a A \right)_{t_2} \right)_{\xi_k} + \theta \Delta t \left(\left(\rho Q \right)_{x_2} - \left(\rho Q \right)_{x_1} \right)_{t_2} = \\
&\Delta x \sum_{k=1}^{n} \left(\omega \left(\rho M_a A \right)_{t_1} \right)_{\xi_k} - \left(1 - \theta \right) \Delta t \left(\left(\rho Q \right)_{x_2} - \left(\rho Q \right)_{x_1} \right)_{t_1} \\
&+ \theta \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho_\ell q \right)_{t_2} \right)_{\xi_k} + \left(1 - \theta \right) \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho_\ell q \right)_{t_1} \right)_{\xi_k}
\end{aligned} \tag{30}$$

and

$$\begin{aligned}
&\Delta x \sum_{k=1}^{n} \left(\omega \left(\rho M_q Q \right)_{t_2} \right)_{\xi_k} + \theta \Delta t \left(\left(\rho \beta \frac{Q^2}{A} \right)_{x_2} - \left(\rho \beta \frac{Q^2}{A} \right)_{x_1} \right)_{t_2} \\
&+ \theta g \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(A \left(\rho \frac{\partial Z}{\partial x} + \rho \frac{Q \left| Q \right|}{K^2} + \frac{\partial \rho}{\partial x} \bar{z} \right) \right)_{t_2} \right)_{\xi_k} = \\
&\Delta x \sum_{k=1}^{n} \left(\omega \left(\rho M_q Q \right)_{t_1} \right)_{\xi_k} - \left(1 - \theta \right) \Delta t \left(\left(\rho \beta \frac{Q^2}{A} \right)_{x_2} - \left(\rho \beta \frac{Q^2}{A} \right)_{x_1} \right)_{t_1} \\
&- \left(1 - \theta \right) g \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(A \left(\rho \frac{\partial Z}{\partial x} + \rho \frac{Q \left| Q \right|}{K^2} + \frac{\partial \rho}{\partial x} \bar{z} \right) \right)_{t_1} \right)_{\xi_k}.
\end{aligned} \tag{31}$$

Linearization Over a Time Step

Nonlinear terms in equations 30 and 31 are approximated with truncated Taylor series written in terms of incremental changes in dependent variables Q and Z. For a typical variable or function, the approximation may be expressed by

$$\phi_{t_2} \cong \phi^*_{t_2} + \frac{\partial \phi^*_{t_2}}{\partial Z} \Delta Z + \frac{\partial \phi^*_{t_2}}{\partial Q} \Delta Q \tag{32}$$

where

$$\Delta Z = Z_{t_2} - Z^*_{t_2} \tag{33}$$

$$\Delta Q = Q_{t_2} - Q^*_{t_2} \tag{34}$$

and the superscript "$*$" indicates evaluation using current values of unknowns obtained from the preceding iteration.

Applying similar approximations to equation 30 results in

$$\Delta x \sum_{k=1}^{n} \left(\omega \left(\rho \left(M_a A \right)^* + \rho \left(M_a \frac{\partial A}{\partial Z} + A \frac{\partial \left(M_a \right)}{\partial Z} \right)^* \Delta Z \right)_{t_2} \right)_{\xi_k}$$

$$+ \theta \Delta t \left(\left(\rho \left(Q^* + \Delta Q \right) \right)_{x_2} - \left(\rho \left(Q^* + \Delta Q \right) \right)_{x_1} \right)_{t_2} = D_{t_1} \tag{35}$$

where

$$D_{t_1} = \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho M_a A \right)_{t_1} \right)_{\xi_k} - \left(1 - \theta \right) \Delta t \left(\left(\rho Q \right)_{x_2} - \left(\rho Q \right)_{x_1} \right)_{t_1}$$

$$+ \theta \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho_\ell q \right)_{t_2} \right)_{\xi_k} + \left(1 - \theta \right) \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho_\ell q \right)_{t_1} \right)_{\xi_k}. \tag{36}$$

Moving terms (equation 35) known from the preceding iteration to the right-hand side results in

$$\Delta x \sum_{k=1}^{n} \left(\omega \left(\rho \left(M_a \frac{\partial A}{\partial Z} + A \frac{\partial \left(M_a \right)}{\partial Z} \right)^* \Delta Z \right)_{t_2} \right)_{\xi_k}$$

$$+ \theta \Delta t \left(\left(\rho \Delta Q \right)_{x_2} - \left(\rho \Delta Q \right)_{x_1} \right)_{t_2} = D_{t_1} - E^*_{t_2} \tag{37}$$

where

$$E^*_{t_2} = \sum_{k=1}^{n} \left(\omega \left(\rho \left(M_a A \right)^* \right)_{t_2} \right)_{\xi_k} + \theta \Delta t \left(\left(\rho Q^* \right)_{x_2} - \left(\rho Q^* \right)_{x_1} \right)_{t_2}. \tag{38}$$

Similarly, nonlinear terms appearing in equation 31 may be approximated by

$$\sum_{k=1}^{n} \left(\omega \left(\rho M_q Q \right)_{t_2} \right)_{\xi_k} \cong \sum_{k=1}^{n} \left(\omega \left(\rho M_q Q \right)^*_{t_2} \right)_{\xi_k}$$

$$+ \sum_{k=1}^{n} \left(\omega \left(\rho M^*_q \right)_{t_2} \Delta Q \right)_{\xi_k} + \sum_{k=1}^{n} \left(\omega \left(\rho Q \frac{\partial M_q}{\partial Z} \right)^*_{t_2} \Delta Z \right)_{\xi_k}, \tag{39}$$

$$\left(\rho \beta \frac{Q^2}{A} \right)_{t_2} \cong \left(\rho \beta \frac{Q^2}{A} \right)^*_{t_2} + \left(2 \rho \beta \frac{Q}{A} \right)^*_{t_2} \Delta Q - \left(\rho \beta \frac{Q^2}{A^2} \frac{\partial A}{\partial Z} \right)^*_{t_2} \Delta Z, \tag{40}$$

$$\left(\rho A\frac{\partial Z}{\partial x}\right)_{t_2} \cong \left(\rho A\frac{\partial Z}{\partial x}\right)^*_{t_2} + \left(\rho\frac{\partial A}{\partial Z}\frac{\partial Z}{\partial x}\right)^*_{t_2}\Delta Z + \left(\rho A^*\right)_{t_2}\Delta\left(\frac{\partial Z}{\partial x}\right), \tag{41}$$

$$\left(\rho A\frac{Q\left|Q\right|}{K^2}\right)_{t_2} \cong \left(\rho A\frac{Q\left|Q\right|}{K^2}\right)^*_{t_2} + \left(2\rho A\frac{\left|Q\right|}{K^2}\right)^*_{t_2}\Delta Q + \left(\rho\frac{Q\left|Q\right|}{K^2}\right)^*_{t_2}\left(\frac{\partial A}{\partial Z} - 2\frac{A}{K}\frac{\partial K}{\partial Z}\right)^*_{t_2}\Delta Z, \tag{42}$$

and

$$\left(A\frac{\partial\rho}{\partial x}\bar{z}\right)_{t_2} \cong \left(A\frac{\partial\rho}{\partial x}\bar{z}\right)^*_{t_2} + \left(\frac{\partial\rho}{\partial x}\left(\frac{\partial A}{\partial Z}\bar{z} + A\frac{\partial\bar{z}}{\partial Z}\right)\right)^*_{t_2}\Delta Z. \tag{43}$$

Substituting approximations 39 through 43 into equation 31 and moving terms known from the preceding iteration to the right-hand side results in

$$\begin{aligned}
&\Delta x\sum_{k=1}^{n}\left(\omega\left(\rho M_q^*\right)_{t_2}\Delta Q\right)_{\xi_k} + \Delta x\sum_{k=1}^{n}\left(\omega\left(\rho Q\frac{\partial M_q}{\partial Z}\right)^*_{t_2}\Delta Z\right)_{\xi_k} \\
&\quad + \theta\Delta t\left(\left(2\rho\beta\frac{Q}{A}\right)^*_{x_2}\Delta Q_{x_2} - \left(2\rho\beta\frac{Q}{A}\right)^*_{x_1}\Delta Q_{x_1}\right)_{t_2} \\
&\quad - \theta\Delta t\left(\left(\rho\beta\frac{\partial A}{\partial Z}\frac{Q^2}{A^2}\right)^*_{x_2}\Delta Z_{x_2} - \left(\rho\beta\frac{\partial A}{\partial Z}\frac{Q^2}{A^2}\right)^*_{x_1}\Delta Z_{x_1}\right)_{t_2} \\
&\quad + \theta g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(\left(\rho\frac{\partial A}{\partial Z}\frac{\partial Z}{\partial x}\right)^*_{t_2}\Delta Z + \left(\rho A^*\right)_{t_2}\Delta\left(\frac{\partial Z}{\partial x}\right)\right)\right)_{\xi_k} \\
&\quad + \theta g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(2\rho A\frac{\left|Q\right|}{K^2}\right)^*_{t_2}\Delta Q\right)_{\xi_k} \\
&+ \theta g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(\left(\rho\frac{Q\left|Q\right|}{K^2}\right)\left(\frac{\partial A}{\partial Z} - 2\frac{A}{K}\frac{\partial K}{\partial Z}\right) + \left(\frac{\partial\rho}{\partial x}\left(\frac{\partial A}{\partial Z}\bar{z} + A\frac{\partial\bar{z}}{\partial Z}\right)\right)\right)^*_{t_2}\Delta Z\right)_{\xi_k} \\
&\quad = F_{t_1} - G^*_{t_2}
\end{aligned} \tag{44}$$

where

$$\begin{aligned}
F_{t_1} = \Delta x\sum_{k=1}^{n}\left(\omega\left(\rho M_q Q\right)_{t_1}\right)_{\xi_k} - \left(1-\theta\right)\Delta t\left(\left(\rho\beta\frac{\beta Q^2}{A}\right)_{x_2} - \left(\rho\beta\frac{\beta Q^2}{A}\right)_{x_1}\right)_{t_1} \\
- \left(1-\theta\right)g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(A\left(\rho\frac{\partial Z}{\partial x} + \rho\frac{Q\left|Q\right|}{K^2} + \frac{\partial\rho}{\partial x}\bar{z}\right)\right)_{t_1}\right)_{\xi_k}
\end{aligned} \tag{45}$$

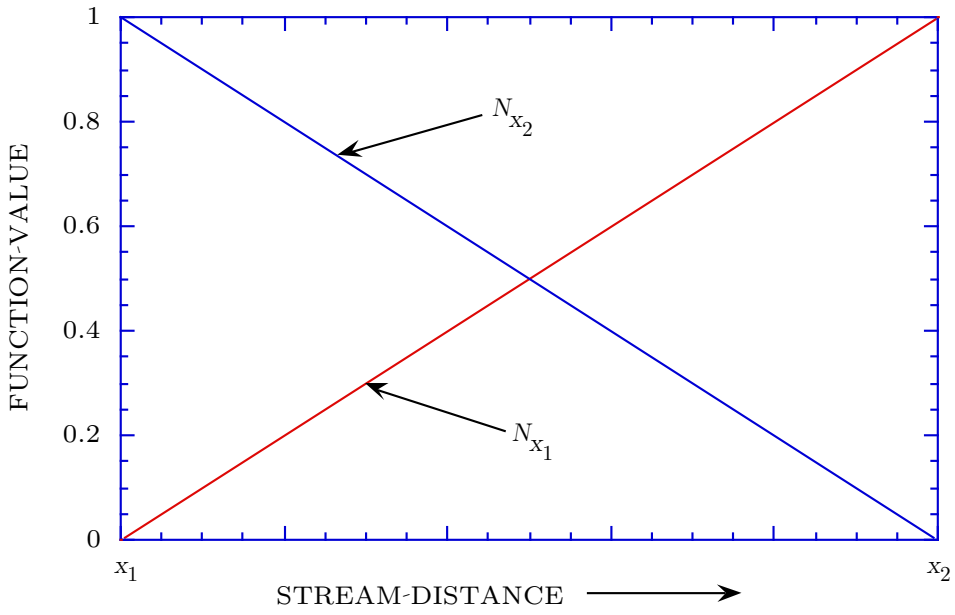

Figure 2: Linear interpolation functions.

and

$$G^*_{t_2} = \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho M_q Q \right)^*_{t_2} \right)_{\xi_k} + \theta \Delta t \left(\left(\rho \beta \frac{Q^2}{A} \right)_{x_2} - \left(\rho \beta \frac{Q^2}{A} \right)_{x_1} \right)^*_{t_2}$$
$$+ \theta g \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(A \left(\rho \frac{\partial Z}{\partial x} + \rho \frac{Q \left| Q \right|}{K^2} + \frac{\partial \rho}{\partial x} \bar{z} \right) \right)^*_{t_2} \right)_{\xi_k} . \tag{46}$$

Interpolation in Space

Equations 37 and 44 are in terms of incremental changes in unknowns located at discrete reach boundaries and intermediate integration points. Unknowns at intermediate points may be approximated in terms of unknowns at reach boundaries through the use of spatial-interpolation functions (figure 2). For a typical variable or function, ϕ, a linear approximation may be written

$$\phi_k \cong N_{x_1,\xi_k} \phi_{x_1} + N_{x_2,\xi_k} \phi_{x_2}, \tag{47}$$

where

$$N_{x_1,\xi_k} = \frac{x_2 - \xi_k}{x_2 - x_1}, \tag{48}$$

and

$$N_{x_2,\xi_k} = \frac{\xi_k - x_1}{x_2 - x_1}. \tag{49}$$

In matrix notation the approximation may be written

$$\phi_k \cong N_{x_1,\xi_k}\phi_{x_1} + N_{x_2,\xi_k}\phi_{x_2} = \left\{ N_{x_1,\xi_k} \quad N_{x_2,\xi_k} \right\} \left\{ \begin{array}{c} \phi_{x_1} \\ \phi_{x_2} \end{array} \right\} = \left\{ N_{\xi_k} \right\}^T \left\{ \phi \right\}. \tag{50}$$

The spatial derivative of a typical variable represented by the linear interpolation functions may be approximated by

$$\frac{\partial \phi_{\xi_k}}{\partial x} \cong \frac{\partial N_{x_1,\xi_k}}{\partial x}\phi_{x_1} + \frac{\partial N_{x_2,\xi_k}}{\partial x}\phi_{x_2}, \tag{51}$$

where

$$\frac{\partial N_{x_1,\xi_k}}{\partial x} = \frac{-1}{x_2 - x_1}, \tag{52}$$

and

$$\frac{\partial N_{x_2,\xi_k}}{\partial x} = \frac{1}{x_2 - x_1}, \tag{53}$$

and in matrix notation written as

$$\frac{\partial \phi_{\xi_k}}{\partial x} \cong \frac{\partial N_{x_1,\xi_k}}{\partial x}\phi_{x_1} + \frac{\partial N_{x_2,\xi_k}}{\partial x}\phi_{x_2} = \left\{ \begin{array}{cc} \frac{\partial N_{x_1,\xi_k}}{\partial x} & \frac{\partial N_{x_2,\xi_k}}{\partial x} \end{array} \right\} \left\{ \begin{array}{c} \phi_{x_1} \\ \phi_{x_2} \end{array} \right\} = \left\{ \frac{\partial N_{\xi_k}}{\partial x} \right\}^T \left\{ \phi \right\}. \tag{54}$$

Substituting the interpolation functions and the relation,

$$\Delta \left(\frac{\partial Z}{\partial x} \right)_{\xi_k} = \left\{ \frac{\partial N_{\xi_k}}{\partial x} \right\}^T \left\{ \Delta Z \right\}, \tag{55}$$

into equations 37 and 44 results in

$$\begin{aligned} \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho \left(M_a \frac{\partial A}{\partial Z} + A \frac{\partial M_a}{\partial Z} \right)^* \right)_{t_2} \left\{ N \right\}^T \right)_{\xi_k} \left\{ \Delta Z \right\} \\ + \theta \Delta t \left(\left(\rho \Delta Q \right)_{x_2} - \left(\rho \Delta Q \right)_{x_1} \right)_{t_2} = D_{t_1} - E^*_{t_2} \end{aligned} \tag{56}$$

and

$$\begin{aligned} \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho M_q^* \right)_{t_2} \left\{ N \right\}^T \right)_{\xi_k} \left\{ \Delta Q \right\} + \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho Q \frac{\partial M_q}{\partial Z} \right)^*_{t_2} \left\{ N \right\}^T \right)_{\xi_k} \left\{ \Delta Z \right\} \\ + \theta \Delta t \left(\left(2\rho\beta \frac{Q}{A} \right)^*_{x_2} \Delta Q_{x_2} - \left(2\rho\beta \frac{Q}{A} \right)^*_{x_1} \Delta Q_{x_1} \right)_{t_2} \\ - \theta \Delta t \left(\left(\rho\beta \frac{\partial A}{\partial Z} \frac{Q^2}{A^2} \right)^*_{x_2} \Delta Z_{x_2} - \left(\rho\beta \frac{\partial A}{\partial Z} \frac{Q^2}{A^2} \right)^*_{x_1} \Delta Z_{x_1} \right)_{t_2} \\ + \theta g \Delta t \, \Delta x \left(\sum_{k=1}^{n} \left(\omega \left(\rho \frac{\partial A}{\partial Z} \frac{\partial Z}{\partial x} \right)^*_{t_2} \left\{ N \right\}^T \right)_{\xi_k} + \sum_{k=1}^{n} \left(\omega \left(\rho A \right)^*_{t_2} \left\{ \frac{\partial N}{\partial x} \right\}^T \right)_{\xi_k} \right) \left\{ \Delta Z \right\} \end{aligned}$$

$$+\theta g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(2\rho A\frac{|Q|}{K^2}\right)_{t_2}^{*}\{N\}^T\right)_{\xi_k}\{\Delta Q\}$$

$$+\theta g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(\left(\rho\frac{Q\,|Q|}{K^2}\right)\left(\frac{\partial A}{\partial Z}-2\frac{A}{K}\frac{\partial K}{\partial Z}\right)+\left(\frac{\partial\rho}{\partial x}\left(\frac{\partial A}{\partial Z}\bar{z}+A\frac{\partial\bar{z}}{\partial Z}\right)\right)\right)_{t_2}^{*}\{N\}^T\right)_{\xi_k}\{\Delta Z\}$$

$$=F_{t_1}-G_{t_2}^{*}. \quad (57)$$

Accumulating terms in equations 56 and 57, coefficients in linear equations 20 and 21 may now be written as

$$c_{2,1}=-\theta\rho_{x_1,t_2}\Delta t, \quad (58)$$

$$c_{2,2}=\Delta x\sum_{k=1}^{n}\left(\omega\left(\rho\left(M_a\frac{\partial A}{\partial Z}+A\frac{\partial M_a}{\partial Z}\right)^{*}\right)_{t_2}N_{x_1}\right)_{\xi_k}, \quad (59)$$

$$c_{2,3}=\theta\rho_{x_2,t_2}\Delta t, \quad (60)$$

$$c_{2,4}=\Delta x\sum_{k=1}^{n}\left(\omega\left(\rho\left(M_a\frac{\partial A}{\partial Z}+A\frac{\partial\left(M_a\right)}{\partial Z}\right)^{*}\right)_{t_2}N_{x_2}\right)_{\xi_k}, \quad (61)$$

and

$$B_2=D_{t_1}-E_{t_2}^{*} \quad (62)$$

for the mass-conservation equation, and

$$c_{3,1}=\Delta x\sum_{k=1}^{n}\left(\omega\left(\rho M_q^{*}\right)_{t_2}N_{x_1}\right)_{\xi_k}-\theta\Delta t\left(\left(2\rho\beta\frac{Q}{A}\right)_{x_1}^{*}\right)_{t_2}$$

$$+\theta g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(2\rho A\frac{|Q|}{K^2}\right)_{t_2}^{*}N_{x_1}\right)_{\xi_k}, \quad (63)$$

$$c_{3,2}=\Delta x\sum_{k=1}^{n}\left(\omega\left(\rho Q\frac{\partial M_q}{\partial Z}\right)_{t_2}^{*}N_{x_1}\right)_{\xi_k}+\theta\Delta t\left(\left(\rho\beta\frac{\partial A}{\partial Z}\frac{Q^2}{A^2}\right)_{x_1}^{*}\right)_{t_2}$$

$$+\theta g\Delta t\,\Delta x\left(\sum_{k=1}^{n}\left(\omega\left(\rho\frac{\partial A}{\partial Z}\frac{\partial Z}{\partial x}\right)_{t_2}^{*}N_{x_1}\right)_{\xi_k}+\sum_{k=1}^{n}\left(\omega\left(\rho A\right)_{t_2}^{*}\frac{\partial N_{x_1}}{\partial x}\right)_{\xi_k}\right)$$

$$+\theta g\Delta t\,\Delta x\sum_{k=1}^{n}\left(\omega\left(\left(\rho\frac{Q\,|Q|}{K^2}\right)\left(\frac{\partial A}{\partial Z}-2\frac{A}{K}\frac{\partial K}{\partial Z}\right)+\left(\frac{\partial\rho}{\partial x}\left(\frac{\partial A}{\partial Z}\bar{z}+A\frac{\partial\bar{z}}{\partial Z}\right)\right)\right)_{t_2}^{*}N_{x_1}\right)_{\xi_k}, \quad (64)$$

$$c_{3,3} = \Delta x \sum_{k=1}^{n} \left(\omega \left(\left(\rho M_q^* \right)_{x_2} \right)_{t_2} N_{x_2} \right)_{\xi_k} + \theta \Delta t \left(\left(2\rho\beta \frac{Q}{A} \right)_{x_2}^* \right)_{t_2} + \theta g \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(2\rho A \frac{|Q|}{K^2} \right)_{t_2}^* N_{x_2} \right)_{\xi_k}, \tag{65}$$

$$c_{3,4} = \Delta x \sum_{k=1}^{n} \left(\omega \left(\rho Q \frac{\partial M_q}{\partial Z} \right)_{t_2}^* N_{x_2} \right)_{\xi_k} + \theta \Delta t \left(\left(\rho\beta \frac{\partial A}{\partial Z} \frac{Q^2}{A^2} \right)_{x_1}^* \right)_{t_2} + \theta g \Delta t \, \Delta x \left(\sum_{k=1}^{n} \left(\omega \left(\rho \frac{\partial A}{\partial Z} \frac{\partial Z}{\partial x} \right)_{t_2}^* N_{x_2} \right)_{\xi_k} + \sum_{k=1}^{n} \left(\omega \left(\rho A \right)_{t_2}^* \frac{\partial N_{x_2}}{\partial x} \right)_{\xi_k} \right) + \theta g \Delta t \, \Delta x \sum_{k=1}^{n} \left(\omega \left(\left(\rho \frac{Q\,|Q|}{K^2} \right) \left(\frac{\partial A}{\partial Z} - 2\frac{A}{K}\frac{\partial K}{\partial Z} \right) + \left(\frac{\partial \rho}{\partial x} \left(\frac{\partial A}{\partial Z} \bar{z} + A \frac{\partial \bar{z}}{\partial Z} \right) \right) \right)_{t_2}^* N_{x_2} \right)_{\xi_k}, \tag{66}$$

and

$$B_3 = F_{t_1} - G_{t_2}^* \tag{67}$$

for the momentum-conservation equation.

Direct Matrix Solution

The equation set 22 is solved directly by Gaussian elimination (Carnahan, Luther, and Wilkes 1969) using computer routines previously employed in another one-dimensional flow model, HYDRAUX (DeLong and Schoellhamer 1989).

Because the coefficient matrix is very sparse (very few locations in the coefficient matrix contain nonzero terms), a technique is used to avoid unnecessary computer storage and computation. The virtual two-dimensional coefficient matrix is transformed into a one-dimensional array. The one-dimensional array stores only coefficients actually computed from equations and coefficients potentially computed during Gaussian elimination. Gaussian elimination is performed only on coefficients stored in the one-dimensional array . . . avoiding unnecessary computation on void locations in the sparse two-dimensional coefficient matrix.

COMPUTER CODE

FourPt is written in FORTRAN 77 (American National Standards Institute 1978) and has been used on a variety of minicomputers, workstations, and microcomputers without modification. Nonstandard uses in the code include the use of symbolic names with lengths greater than six (up to 30) characters and with mixed upper and lower case characters. The purpose is to provide better in-code documentation. The greatest departure from traditional FORTRAN programming in FourPt is in the extensive use of data encapsulation.

FourPt is constructed of FORTRAN 77 modules (DeLong et al. 1992; Thompson et al. 1992). Herein, module is defined (Stroustrup 1988) as a programming construct composed of a set of procedures combined with the data that they manipulate. Data encapsulation refers to the technique of encapsulating data with functions that are used by client routines to access and manipulate data indirectly. Direct use of data is restricted to the few functions comprising the module. This technique helps to reduce overall program dependence on the form of the data. For example, FourPt uses a channel-properties module to supply geometric or hydraulic properties needed throughout the program. Cross-sectional area at x distance downstream and h depth of flow is obtained from a function invocation such as

$$area = CxArea(\, x,\, h\,).$$

The arguments x and h are all that are required by $CxArea$ (or any of the properties functions) and would logically be known in routines requiring cross-sectional area. No knowledge is required concerning the shape of the channel, number of data points representing the channel, or any detail specific to the computation of cross-sectional area. Consequently, routines invoking $CxArea$ are not dependent on the form of the data describing the channel, and a variety of channel descriptions may be substituted without modification to the client code. At present (FourPt, version 95.01), three such channel-properties modules exist. Properties modules describing channels of rectangular cross section (file *chcxrect.f* as shown in table 1), trapezoidal cross section (file *chcxtrap.f*), and irregular cross section (file *chcxtbl.f*) may alternatively be linked with FourPt to provide different channel descriptions without modifying any existing code. Other data fundamental to the simulation of unsteady open-channel flow are grouped according to type and encapsulated with functions in a manner similar to channel properties. The source code for FourPt, version 95.01, is contained in 12 modules (table 1), one main program, and four utility (table 2) files. Because modules originally written for FourPt encapsulate data that are fundamental to the general field of streamflow computation, some of the modules could be of use to other models or pre- and post-processing programs.

Use of FORTRAN 77 modules, like the use of top-down structured programming, is a programming technique allowed but not supported or enforced by the FORTRAN 77 language. Consequently, successful implementation of modules with FORTRAN 77 and realization of potential benefits depend on voluntary adherence to the programming technique. To most efficiently share future revisions and additions to the FourPt computer code and to enhance sharing of related computer codes in general, persons modifying or adding to FourPt are encouraged to do so through the use of modules.

Module	*Source Files*	*Module-Data Include Files*
All	all module source	`network.inc`
Boundary values	`netbnd.f`	`netbnd.inc`
Buffered output	`bufout.f`	`bufout.inc`
Channel properties[a]		
rectangular	`chcxrect.f`	`chcxrec1.inc, chcxrec2.inc`
trapezoidal	`chcxtrap.f`	`chcxtrap.inc`
irregular	`chcxtbl.f`	`chcxtbl.inc`
Channel-flow status	`chstatus.f`	`chstatus.inc`
Channel schematic	`chschmt.f`	`chconnec.inc, chnlcomp.inc, chnluser.inc`
Matrix solver	`solvefpt.f`	`skyline1.inc, skyline2.inc, skyline3.inc`
Program control	`netcntr1.f, netcntr2.f`	`netcntrl.inc`
Channel constraint	`chcnstrt.f`	`chcnstrt.inc, strmcnst.inc, dambrk.inc, logcnst.inc`
Master file	`master.f`	`master.inc`
3-parameter rating	`table3.f`	`table3.inc`
Volume/mass balance	`netblnce.f`	`netblnce.inc`
Fluid density	`netdense.f`	`netdense.inc`

[a]Only one channel-properties module, rectangular, trapezoidal, or irregular, may be linked at any time with the main program.

Table 1: FORTRAN 77 modules used in FourPt.

Source file	*Purpose*
`fourpt.f`	main program
`floweq1d.f`	governing flow equations
`linear.f`	linear interpolation
`fileutil.f`	file utilities
`strings.f`	string utilities

Table 2: Main-program and utility files used in FourPt.

Default filename	*Data type*
`control.dat`	program control
`schemat.dat`	schematic description
`cxgeom.dat`	channel properties
`strmcnst.dat`	constraint properties
`master.fil`	file names and unit numbers
`initcond.dat`	initial values
`bndval.dat`	boundary values
`density.dat`	water density values
`perturb.dat`	perturbation parameters

Table 3: Input files.

INPUT

Data read by FourPt are grouped by type as shown in table 3. Each data group is stored in a separate file. Use of the first three (program control, schematic description, and channel properties) is mandatory. Use of the remaining data groups is optional.

By **convention**, the **upstream** end of any channel is the end of the channel through which **positive flow enters** the channel. Conversely, the **downstream** end of a channel is the end of any channel through which **positive flow leaves** the channel. Spatial units of measure must be consistent throughout all input data. A description of each data group follows, in the order shown in table 3.

Program Control

Program-control data required by FourPt include indices (such as those determining which terms will be included in the numerical solution and what level of printed output is requested) and parameter values (such as the values of the number of time steps to be simulated, time increment, and time-weighting factor). An example file is shown in figure 3. The default name of the file containing program-control data is `control.dat`. The first two lines of input are 80 character title lines that are simply copied to the beginning of the print output file for identification of the model run. The remainder of the data are entered in free-field format. Note that each data group is and should be terminated by a "/". Characters following the "/" are not read by the program but are included for explanation to simplify data preparation. The authors recommend that users of the model continue this practice by copying and modifying existing data input files when preparing input for new applications.

An item by item description of program-control data follows. Values appearing next to data names are examples or allowable ranges. **There are no defaults**.

```
   Hypothetical example, Swamprat Creek, Near Pete's Bayou
                Lew DeLong, May, 1991
1          / Terms, 1=dynamic, 2=diffusion, 3=kinematic
0          / 0 = contant density, 1 = variable density.
0          / 0 = constant sinuosity, 1 = variable sinuosity.
0          / 0 = do not read boundary values, 1 = read values.
0          / 0 = perturbation inactive, 1 = perturbation active.
32.2       / acceleration due to gravity.

40         / MaxTimeSteps, maximum number of time steps.
0          / NetStartTime, starting elapse time, in seconds.
900        / DT, time step, in seconds.
0.6        / Theta, time-weighting factor.
0.5, 1.0   / local ( 0 to 1 ) quadrature-point coordinate and weight, paired.

5          / MaxIterations, maximum number of iterations per time step.
1          / LuInc, interval for complete forward eliminations.
0.005      / ToleranceQ, tolerance for closure on discharge.
0.005      / ToleranceY, tolerance for closure on water-surface elevation.

1          / PrintLevel, amount of printing, 0 to 9, increasing with number.
0          / PrintCount, initial value of print counter.
4          / PrintInc, print increment.
0          / TimeSeriesCount, initial value of time-series counter.
1          / TimeSeriesInc, time-series increment.
1,0.0
1,50000.0  / Requested time-series output, paired channel no. & downstream distance.

0          / SpaceCount, initial value of spatial-series output counter.
99999      / SpaceInc, spatial-series increment.

0          / RestartFile, if 1, write restart file.

1          / ChannelNumber, for equation boundary (+ upstream, - downstream).
1          / Number of sinusoidal components.
488.733    / Base, base value for equation boundary condition.

238.733    / Amplitude.
2.5        / Period, in hours.
1.25       / Phase angle, in hours.

0.0        / EqStart, time at which equation takes effect.
2.5        / EqStop, time at which equation is no longer effective.


-1         / Channel number for equation boundary (+ upstream, - downstream).
0          / Number of sinusoidal components.
1.71       / Base, base value for equation boundary condition.

0.0        / EqStart, time at which equation takes effect.
99999.9    / EqStop, time at which equation is no longer effective.
```

Figure 3: Example program-control data file.

Title lines:

Two 80-character lines of text that will be copied to the print output file. Use these lines to identify a particular model run.

Terms: `1, 2, or 3`

This integer index determines which terms of the governing equations are included in the numerical solution, and consequently, whether the equations solved represent a

`[1]` dynamic-wave equation,

`[2]` diffusion-wave equation, or

`[3]` kinematic-wave equation.

For discussion of the above equations, see *Example 3* under *Examples*. As a result of underlying physics, all equation types will not work correctly with all boundary conditions, initial conditions, or applications in general.

Density Index: `0 or 1`

`[0]` density is assumed constant, or

`[1]` density is allowed to vary with space and time.

In the current version (95.01), if density is allowed to vary, for each time step a value of density must be supplied by the user for both ends of each channel. For a description of density input values, see *Density Values* under *Input*. In the absence of transport computations, density at intervening points is linearly estimated from values at channel ends.

Sinuosity Index: `0 or 1`

`[0]` sinuosity is assumed constant with depth of flow, or

`[1]` sinuosity is allowed to vary with depth of flow.

Sinuosity, as used, is a ratio of channel length to main-valley length. As depth of flow increases in a natural meandering river, effective flow paths tend to shorten. Changes in effective channel length may significantly affect shape and speed of flood peaks. For more information regarding change in sinuosity with depth of flow, see DeLong (1986). Input of sinuosity as a function of flow depth is described for irregular cross sections under *Channel Properties*.

Boundary-Values Index: `0 or 1`

`[0]` time varying boundary values will not be read, or

`[1]` boundary values will be read for each time step.

If the index is 1, boundary values must be supplied by the user in a separate boundary-value file. See *Boundary Values* under *Input*.

Perturbation Index: `0 or 1`

`[0]` input data will not be randomly perturbed, or

`[1]` selected input data will be randomly perturbed.

In the current version (95.01), this option allows user to perturb coefficients used in boundary-value equations. The purpose of such perturbation might be to determine the relation of uncertainty in model results to uncertainty in boundary values. If this option is selected, the model will attempt to read normal-distribution parameters from a separate file with the default name `perturb.dat`. See *Perturbation Parameters* under *Input*.

Acceleration Due to Gravity: `32.2`

This floating-point value represents the acceleration due to gravity. **Spatial units should be consistent with other input such as initial conditions, boundary conditions, and schematic data.** For example, 32.2 feet per second per second would be consistent with boundary and initial conditions entered in terms of cubic feet per second and feet. A value of 9.8 meters per second per second would be consistent with the use of cubic meters per second and meters. Program output will be in the same units as input.

MaxTimeSteps: `0, 1, 2, ...`∞

This integer value is the maximum number of time steps that will be simulated. It is limited only by available computer time or user perseverance. Total simulation time may be calculated

$$\texttt{total simulation time} = (\texttt{MaxTimeSteps})(\texttt{DT})$$

NetStartTime: `-`∞`, 0, 1, 2, ...`∞

This integer value is the starting elapsed time in seconds. Elapsed time at the end of a simulation may be calculated

$$\texttt{ending elapse time} = (\texttt{MaxTimeSteps})(\texttt{DT}) + \texttt{NetStartTime}$$

DT: `1, 2, 3, ...`∞

DT is the time increment, the length of a time step, in seconds. It is a positive, non-zero, integer value. It should be small enough to adequately represent time dependent boundary conditions and simulated features of the flow (such as flood peaks). Adequacy of a selection and general sensitivity of results to DT should be determined by comparing results using a range of values for DT. Effects of DT and theta are interrelated. Additionally, the effects of DT and spacing of computational cross sections (see Schematic Data) may be interrelated. Changes in DT will cause corresponding changes in the total simulated time. See definition of MaxTimeSteps given above.

Theta: `0.5 to 1.0, normally about 0.6`

Theta, a floating-point value, is the time-weighting factor (see subsection Numerical Integration in Time under Numerical Formulation). Larger values of Theta numerically damp the

solution. A value of 0.5 represents equal time weighting of values and causes least numerical dampening of results. The effect of theta is related to length of the time step, DT, and is application dependent. Sensitivity of model results to theta and DT should be determined to avoid selection of values causing excessive numerical dampening. Use of Theta to dampen numerical oscillations occurring in results is not recommended. Numerical oscillations can generally be attributed to lack of numerical convergence with respect to space and or time and should be eliminated by appropriate adjustment of space and time increments.

Quadrature Points: `0.5, 1.0`

A minimum of 1 to a maximum of 5 pairs of floating-point numbers represent, for each pair respectively, the location in local coordinates (0.0 to 1.0) and relative weight (0.0 to 1.0) to be given each point used in spatial integration. The example given above represents integration by evaluation at a point centered between end points of an interval and weighted by 1.0. If multiple integration points are used, their respective weights should sum to 1.0. Use of more than one integration point has not been shown to be advantageous but results in greater computational effort. Unless otherwise inclined for purpose of experimentation, users are advised to use the example values given above. This feature has been retained only for purpose of completeness and demonstration. More detailed discussion of numerical integration may be found in texts concerned with applied numerical techniques (Carnahan, Luther, and Wilkes 1969).

MaxIterations: `1, 2, 3, ...∞, normally about 5`

This positive integer value is the maximum number of iterations allowed per time step. Fewer than the maximum may occur if the numerical scheme satisfies closure criteria. Generally, if a numerical scheme (FourPt) does not satisfy closure criteria within 5 iterations, more iterations alone do not significantly improve results.

LuInc: `1`

This positive integer value relates to the algorithm for direct simultaneous solution of equations during model execution. It is the interval, in terms of iterations within a time step, at which complete forward elimination of the coefficient matrix will be performed. Intervening solutions will be computed with the coefficient matrix formed from a preceding iteration. Forward elimination automatically occurs on the first iteration of each time step. In general, more iterations will be required if a complete forward elimination is not performed every iteration. However, for large networks of channels, solutions without complete forward elimination for every iteration may require less overall execution time. The effect is problem dependent and specifically dependent on the relation of computational effort required for equation assembly to that required for subsequent forward elimination of the equations.

ToleranceQ: `0.005`

This positive floating-point value is part of the closure criterion for flow. One of two separate criteria are exercised depending on whether the magnitude of the flow is larger or smaller than SmallQ, a value preset in the FourPt computer code. If larger than SmallQ, computed flow at a location is considered to be sufficiently accurate when its magnitude of change over

the last iteration divided by its current value is equal to or less than ToleranceQ. If less than or equal to SmallQ, computed flow at a location is considered to be sufficiently accurate when the magnitude of its change over the last iteration divided by SmallQ is equal to or less than ToleranceQ. SmallQ is preset equal to 100.0 in the FOURPT computer code. The overall solution for a time step is considered to have closed with respect to flow when the appropriate discharge criteria is satisfied at all computational locations.

ToleranceZ: `0.005`

This positive floating-point value is part of the closure criterion for water-surface elevation. The computed water-surface elevation at a location is considered to be sufficiently accurate when the magnitude of the change in water-surface elevation over an iteration is smaller than ToleranceZ. The overall solution is considered to have closed with respect to water-surface elevation when the criteria is satisfied at all computational locations.

PrintLevel: `1`

Positive integer sets the level of output written to the print file. Increasingly larger numbers produce increasing voluminous output:

`[ 1]` standard output (recommended).

`[>1]` adds schematic-data input and numbers of computational locations.

`[>8]` adds low-level debugging information regarding solution matrix.

PrintCount: `0`

This integer value is the initial value of the print counter. For more information on how it may be used, see PrintInc below.

PrintInc: `0 to` ∞

This integer value is the interval, in time steps, at which printed output will be written during model execution. PrintCount is incremented by one at the beginning of each time step. At the conclusion of a time step, if PrintCount is equal to PrintInc, user-requested output (see PrintLevel above) is written to the print output file and PrintCount is set to zero.

TimeSeriesCount: `0`

This integer value is the the initial value of the time-series output counter. For more information on how it may be used, see TimeSeriesInc below.

TimeSeriesInc: `1 to` ∞

This integer value is the interval, in time steps, at which time-series output will be saved during model execution. TimeSeriesCount is incremented by one at the beginning of each time step. At the conclusion of a time step, if TimeSeriesCount is equal to TimeSeriesInc, timeseries output is stored in the time-series output file and TimeSeriesCount is set to zero. See Time-series locations below.

Time-series locations: `1, 0.0, 1, 50000.0`

These values are paired channel number (integer value) and downstream distance (floating-point value) specifying locations at which time-series of flow and water-surface elevations are to be saved. The maximum number of time series allowed is equal to MaxTS, set in the include file `netcntrl.inc`. In the example above, time series are requested at 0.0 and 50000.0 feet in channel 1. Channel-location pairs may be entered on more than one line. If no time series are requested, a line should still be entered and terminated with a "/".

SpaceCount: `0`

This integer value is the initial value of the spatial-series output counter. For more information on how it may be used, see SpaceInc below.

SpaceInc: `1 to` ∞

This integer value is the interval, in time steps, at which spatial-series of flow and flow depths will be saved during model execution. SpaceCount is incremented by one at the beginning of each time step. At the conclusion of a time step, if SpaceCount is equal to SpaceInc, spatial-series output is stored, and SpaceCount is set to zero.

RestartFile: `0`

`[0]` Restart file **is not** requested.

`[1]` Restart file **is** requested.

A restart file contains values of flow and water-surface elevation that may subsequently be used as FourPt initial conditions. To use the restart file for initial conditions, the correct initial condition flags must be set (see *Schematic Data*), and the file must be named appropriately (see File Names and Unit Numbers). If, when used, computational locations in the initial-condition file match those set in the current execution of FourPt, values from the initial-condition file are used directly. If locations do not match, values at required locations are linearly interpolated from those contained in the initial-condition file.

The remainder of lines in the program-control file are optional and provide one of two methods for setting boundary values at channel extremities. The method described here is to estimate boundary values from equations with user-supplied parameters. A second method that involves reading a time series of boundary values from a user-supplied file is described in the section *Boundary Values* under *Input*. When values are available from both sources for a specific boundary, values read from the time series are used.

Data required for the computation of boundary values from equations are parameters and coefficients related to the general harmonic equation

$$Value = EqBase + \sum_{i=1}^{EqComponets} Amplitude_i \cos\left(2\pi \frac{(ElapseTime + PhaseAngle_i)}{Period_i}\right) \qquad (68)$$

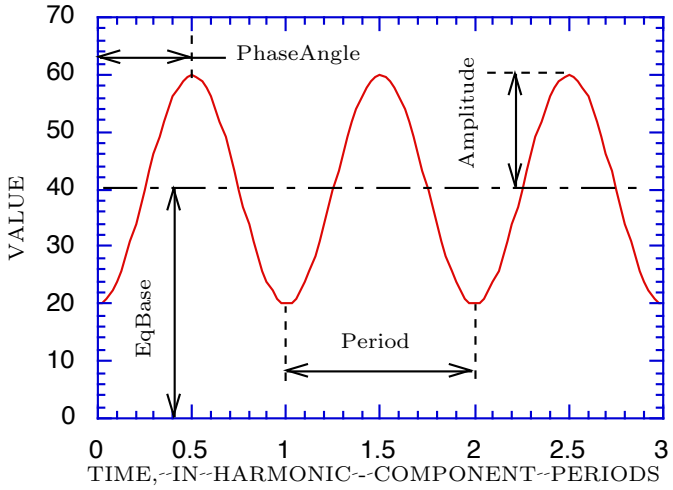

Figure 4: Sinusoidal component used in optional computation of boundary values from equations.

where $Value$ is the computed value that may be used as a boundary value, $ElapseTime$ is the elapse time in hours, and the remaining parameters are described below. One component of the equation is shown in figure 4. If no equations are to be used, no lines are required.

Channel number: `1`

This integer value specifies to which channel an equation refers. If positive, it refers to the upstream end of the channel, and if negative, it refers to the downstream end of the channel. By convention, positive flow enters the upstream end of a channel and exits the downstream end of a channel. Lines beginning with this line and extending through EqStop (see below) are repeated for each equation.

EqComponents: `1`

This integer value is the number of sinusoidal components used in the current equation. The minimum number of sinusoidal components is 0, and the maximum number of sinusoidal components is limited by the FORTRAN parameter `MaxComponents` set in the include file `netcntrl.inc`.

EqBase: `488.733`

This floating-point number is the base value about which sinusoidal components oscillate.

The following data lines for Amplitude through PhaseAngle are repeated EqComponents times in the sequence shown. If no sinusoidal components are required, EqComponents is 0, these lines should not be entered.

Amplitude: `238.733`

This floating-point number is the amplitude of a sinusoidal component.

Period: `9000.0`

This floating-point number is the period, in hours, of a sinusoidal component.

PhaseAngle: `6750.0`

This floating-point number is the phase angle, in hours, of a sinusoidal component.

EqStart: `0.0`

This floating-point value is the beginning time, in hours, of the time interval during which this equation will be used to compute a boundary value.

EqStop: `9000.0`

This floating-point number is the ending time, in hours, of the time interval during which the equation will be used to compute boundary values. If the model continues to run after this time, the particular boundary value will remain constant at the last computed value.

Schematic Description

Schematic data (figure 5) required by FourPt include information describing individual channels (such as which user-supplied cross sections and how many computational cross sections define a particular channel) and the nature of connections among channels. The default name for the schematic-data file is `schemat.dat`. Data are entered in free-field format except for cross-section identifiers which are read from the first 16 spaces of their lines. Note that each data group is and should be terminated by a "/". Characters following the "/" are not read by the program but are included for explanation to simplify data preparation. The authors recommend that users of the model continue this practice by copying and modifying existing data input files when preparing input for new applications.

Description of schematic data required for a channel follows. Values appearing next to data names are examples or allowable ranges. **There are no defaults.**

Channel Number: `1`

This integer value identifies by number the channel, referred to as the current channel, to which the following information applies. Channels should be numbered sequentially, beginning with 1, and entered in a monotonically increasing sequence.

User Cross Sections: `2`

This integer value is the number of user-supplied cross sections in the current channel. The minimum number is 2, one at each end of a channel. The maximum number of user-supplied cross sections allowed per channel is limited by the particular hydraulic-properties module linked with the FourPt model (see *Channel Properties*).

dx: `2000.0`

This floating-point value is the requested spacing, in feet, among computational cross sections. Actual spacing may be larger than dx in order to obtain an integer number of computational cross sections.

NKEEP: `0 or 1, normally 1`

This integer index specifies that computational cross sections

`[0] will not, or`

`[1] will`

be placed at all locations occupied by user-supplied cross sections. During model runs, unknowns (such as water-surface elevation and flow) are located only at computational cross sections. This means that if NKEEP = 0, the only two user cross sections at which it can be ensured that unknowns will be determined will be those cross sections located at the extremities of the channel. All user-supplied cross sections would still be used for determining geometric and hydraulic properties at computational cross sections and in intervening reaches. When using NKEEP = 0, particular attention should be addressed to checking spatial convergence of solutions.

ICNDAP: `0 to 7, only options 0, 1, 5, and 7 are currently implemented.`

Integer indicates the type of approximation to be used for determining initial values of unknowns in the absence of observed values or values obtained from previous model runs. Initial conditions will be approximated from

`[0]` user input in schematic-data file,

`[1]` normal depth computations,

`[2]` (this option not assigned),

`[3]` same as 1, but channel filled to remove adverse slopes,

`[4]` maximum water-surface elevation for each channel,

`[5]` initial values file,

`[6]` not approximated, or

`[7]` steady flow computation within a channel.

For normal depth computations, option 1, water-surface elevation is approximated using initial flow values entered in the schematic-data file. For steady flow approximation, option 7, computations begin with the flow and water-surface elevation specified (in the schematic data file) at the downstream end of the channel and proceed upstream.

NVAL: `0`

This index is reserved for future use.

```
1                               / Channel number
        2, 2000.0, 1, 0, 0      / user cross sections, dx, NKEEP, ICNDAP, NVAL
Upstream_1                      / user cross-section ID
       38.617,      1059.3      / water-surface elevation, discharge
Downstream_1                    / user cross-section ID
       38.617,       928.653    / water-surface elevation, discharge
 2                              /upstream boundary condition code
  0                              /number of connecting channels
11                              /downstream boundary condition code
  1                              /number of connecting channels
   +3                             / connecting Channel number


2                               / Channel number
        2, 2000.0, 1, 0, 0      / user cross sections, dx, NKEEP, ICNDAP, NVAL
Upstream_2                      / user cross-section ID
         38.617, 0.0            / water-surface elevation, discharge
Downstream_2                    / user cross-section ID
         38.617, -116.523       / water-surface elevation, discharge
 2                              /upstream boundary condition code
  0                              /number of connecting channels
11                              /downstream boundary condition code
  1                              /number of connecting channels
   +3                             / connecting Channel number


3                               / Channel number
        2, 2000.0, 1, 0, 0      / user cross sections, dx, NKEEP, ICNDAP, NVAL
Upstream_3                      / user cross-section ID
         38.617, 617.925        / water-surface elevation, discharge
Downstream_3                    / user cross-section ID
         38.55175, 67.089       / water-surface elevation, discharge
12                              /upstream boundary condition code
  3                              /number of connecting channels
   -1                             / connecting Channel number
   -2                             / connecting Channel number
   +4                             / connecting Channel number
11                              /downstream boundary condition code
  1                              /number of connecting channels
   +5                             / connecting Channel number


4                               / Channel number
        2, 2000.0, 1, 0, 0      / user cross sections, dx, NKEEP, ICNDAP, NVAL
Upstream_4                      / user cross-section ID
         38.617, 197.736        / water-surface elevation, discharge
Downstream_4                    / user cross-section ID
         38.55175, 105.93       / water-surface elevation, discharge
11                              /upstream boundary condition code
  1                              /number of connecting channels
   +3                             / connecting Channel number
11                              /downstream boundary condition code
  1                              /number of connecting channels
   +5                             / connecting Channel number
```

Figure 5: Partial example schematic-data file, four channels of a six-channel network.

User Cross-section ID: `Upstream_1`

This string, up to 16 characters beginning in column 1, identifies a user-supplied cross section. It may be used by other modules to properly construct tables of geometric and hydraulic properties defining the channel.

Water-surface Elevation: `38.62`

This floating-point value is an estimate of initial water-surface elevation. Under some circumstances, it may be used to estimate initial conditions. See ICNDAP.

Flow: `1059.3`

This floating-point value is an estimate of initial stream flow. Under some circumstances it may be used to estimate initial conditions. See ICNDAP.

Upstream Boundary Condition Code: `2`

This integer value indicates type of boundary condition to be applied to the upstream end of the current channel. The upstream end by convention is the end through which positive flow enters the channel. (See boundary-condition codes below).

Number of Connecting Channels: `0`

This integer value indicates the number of channels to be connected to the upstream end of the current channel.

Connecting Channel Number: This integer value indicates the channel number of a channel that is connected to the upstream end of the current channel. It is positive if connected to the upstream end of the connecting channel, and negative if connected to the downstream end of the connecting channel. This line must be repeated for each connecting channel. The number of lines must be consistent with the preceding number of connecting channels. In this case, zero lines would be required.

Downstream Boundary Condition Code: `11`

This integer value indicates the type of boundary condition to be applied to downstream end of the channel. The downstream end by convention is the end through which positive flow exits the channel. (See boundary-condition codes below).

Number of Connecting Channels: `1`

This integer value indicates the number of channels to be connected to the downstream end of the current channel.

Connecting Channel Number: `+3`

This integer value indicates the channel number of a channel that is connected to the downstream end of the current channel. It is positive if connected to the upstream end of the connecting channel, and negative if connected to the downstream end of the connecting channel. This line must be repeated for each connecting channel. The number of lines must be consistent with the preceding number of connecting channels.

```
1                   / output index ( 0 - no echo, 1 - echo ).

1                   / channel number.
0.045               / flow resistance coefficient.
Upstream_1          / upstream cross-section identifier.
0.0 100.0 100.0     / downstream distance, width, bottom elevation.
Downstream_1        / downstream cross-section identifier.
50000.0 100.0 50.0  / downstream distance, width, bottom elevation.

2                   / channel number.
0.045               / flow resistance coefficient.
Upstream_2          / upstream cross-section identifier.
0.0 100.0 50.0      / downstream distance, width, bottom elevation.
Downstream_2        / downstream cross-section identifier.
50000.0 100.0 0.0   / downstream distance, width, bottom elevation.
```

Figure 6: Example rectangular cross-section data file.

Boundary-condition codes

`[ 1]` known water-surface elevation.

`[ 2]` known flow.

`[ 4]` known water-surface slope (downstream only).

`[11]` water-surface elevation equal to that of a connecting channel.

`[12]` sum of flows equal zero.

`[...31]` 3-parameter rating applied to water-surface elevation (see *Constraint Properties*).

`[...32]` 3-parameter rating applied to flow (see *Constraint Properties*).

`[...51]` user-defined constraint applied to water-surface elevation (see *User-programmable constraints*).

`[...52]` user-defined constraint applied to discharge (see *User-programmable constraints*).

Channel Properties

Channel properties in FourPt, version 95.01, are represented by one of three modules characterized by either rectangular, trapezoidal, or irregular cross sections. Only one hydraulic properties module may be linked to the FourPt program at a time, and input-data requirements differ for each.

Rectangular and trapezoidal cross-section modules require and accept only two cross sections per channel, located at channel extremities. Data required for rectangular and trapezoidal cross sections (figures 6 and 7) are entered in free-field format and are identical with one exception. The

```
1                   / output index ( 0 - no echo, 1 - echo ).

1                   / channel number.
0.045               / flow resistance coefficient.
Upstream_1          / upstream cross-section identifier.
0.0 100.0 10.0   1.0 / downstream distance, bottom width, bottom elevation,d(width)/dh.
Downstream_1        / downstream cross-section identifier.
5000.0 100.0 5.0 1.2 / downstream distance, bottom width, bottom elevation, d(width)/dh.

2                   / channel number.
0.045               / flow resistance coefficient.
Upstream_2          / upstream cross-section identifier.
5000.0 100.0 5.0 0.5 / downstream distance, bottom width, bottom elevation, d(width)/dh.

Downstream_2        / downstream cross-section identifier.
9000.0 100.0 0.0 0.8 / downstream distance, bottom width, bottom elevation, d(width)/dh.
```

Figure 7: Example trapezoidal cross-section data file.

trapezoidal cross sections require an additional parameter — the ratio of change in channel width to depth of flow. Data required for the trapezoidal-cross-section module are itemized below. Note (figures 6 and 7) that each data group is and should be terminated by a "/". Characters following the "/" are not read by the program but are included for explanation to simplify data preparation. The authors recommend that users of the model continue this practice by copying and modifying existing data input files when preparing input for new applications.

Output index: `(I1)`

Index indicating data input

[**0**] `will not, or`

[**1**] `will`

be written to the print file (default name `netprint.dat`).

Channel indicator record `1`

This integer value is the channel number for which properties are being entered.

Flow resistance coefficient `0.045`

This floating-point value is the flow resistance for the channel, constant with depth of flow and reach.

Upstream cross-section identifier `Upstream_1`

This record, up to 16 characters, identifies the cross section at the upstream end of the channel.

Upstream properties record `0.0 100.0 10.0 1.0`

field 1 `0.0`

This floating-point value is the downstream reference distance.

field 2 `100.0`

This floating-point value is the bottom width of the upstream cross section.

field 3 `10.0`

This floating-point value is the elevation of the bottom of the upstream cross section.

field 4 `1.0`

This floating-point value is the ratio of change in channel width with respect to depth in the upstream cross section.

Downstream cross-section identifier `Downstream_1`

This record, up to 16 characters, identifies the cross section at the downstream end of the channel.

Downstream properties record `5000.0 100.0 0.5 1.2`

field 1 `5000.0`

This floating-point value is the downstream reference distance.

field 2 `100.0`

This floating-point value is the bottom width of the downstream cross section.

field 3 `5.0`

This floating-point value is the elevation of the bottom of the downstream cross section.

field 4 `1.2`

This floating-point value is the ratio of change in channel width with respect to depth in the downstream cross section.

The irregular cross-section module also requires two cross sections located at channel extremities but can accept as many cross sections as program dimensions will allow. The total of cross sections in all channels is limited by the FORTRAN parameter `MaxTables` found in the include file `chcxtbl.inc`. Unlike data input for rectangular and trapezoidal cross sections, irregular cross-section data are explicitly formatted as shown in figure 8 and described below with corresponding FORTRAN formats. Normally such records would be produced by a hydraulic properties preprocessing progam such as the "Hydraulic Information Exchange Program" (Fulford 1993) that operates on station-elevation data obtained from field surveys or estimated from maps.

```
0
CH    1
HY XS 8                    .0     -63.30
DP        .00  0.000000E+00  0.000000E+00  1.00  1.00  1.00   575.7   575.7
DP       8.37  0.125007E+05  0.206631E+07  1.00  1.00  1.00  2411.3  2411.4
DP      13.83  0.275014E+05  0.686749E+07  1.00  1.00  1.00  3083.5  2857.1
DP      20.02  0.455024E+05  0.155143E+08  1.00  1.00  1.00  2732.7  2962.9
DP      27.14  0.671035E+05  0.287387E+08  1.00  1.00  1.00  3335.0  3103.8
DP      35.29  0.930249E+05  0.477787E+08  1.00  1.00  1.00  3026.0  3276.3
DP      44.48  0.124131E+06  0.729458E+08  1.00  1.00  1.00  3743.5  3572.2
DP      54.04  0.161457E+06  0.997988E+08  1.00  1.00  1.00  4065.3  4307.2
DP      63.06  0.206249E+06  0.127022E+09  1.00  1.00  1.00  5866.4  5532.2
DP      72.82  0.260000E+06  0.186420E+09  1.00  1.00  1.00  5148.1  5551.7
HY XS 7                 3839.0    -120.40
DP        .00  0.000000E+00  0.000000E+00  1.00  1.00  1.00  -215.5  -215.5
DP      34.98  0.125007E+05  0.386947E+07  1.00  1.00  1.00   930.2   941.0
DP      46.17  0.275014E+05  0.983300E+07  1.00  1.00  1.00  1750.9  1667.6
DP      56.47  0.455024E+05  0.211926E+08  1.00  1.00  1.00  1744.4  1855.8
DP      67.66  0.671035E+05  0.380435E+08  1.00  1.00  1.00  2116.3  2037.9
DP      79.65  0.930249E+05  0.598927E+08  1.00  1.00  1.00  2207.5  2334.4
DP      92.40  0.124131E+06  0.914351E+08  1.00  1.00  1.00  2671.9  2545.4
DP     107.00  0.161457E+06  0.138261E+09  1.00  1.00  1.00  2441.3  2641.4
DP     122.50  0.206249E+06  0.184210E+09  1.00  1.00  1.00  3338.4  3167.7
DP     139.75  0.260000E+06  0.269038E+09  1.00  1.00  1.00  2893.6  3202.2
HY XS 6                 8037.0     -63.30
DP        .00  0.000000E+00  0.000000E+00  1.00  1.00  1.00  1617.0  1617.0
DP       6.35  0.125007E+05  0.211972E+07  1.00  1.00  1.00  2320.2  2320.9
DP      12.59  0.275014E+05  0.752290E+07  1.00  1.00  1.00  2487.7  2492.0
DP      19.55  0.455024E+05  0.165714E+08  1.00  1.00  1.00  2685.0  2684.0
DP      27.28  0.671035E+05  0.300023E+08  1.00  1.00  1.00  2903.9  2909.8
DP      35.93  0.930249E+05  0.500373E+08  1.00  1.00  1.00  3089.5  3057.0
DP      45.88  0.124131E+06  0.774957E+08  1.00  1.00  1.00  3163.0  3262.2
DP      56.95  0.161457E+06  0.115283E+09  1.00  1.00  1.00  3580.6  3469.2
DP      69.73  0.206249E+06  0.170976E+09  1.00  1.00  1.00  3429.1  3542.5
DP      85.05  0.260000E+06  0.250080E+09  1.00  1.00  1.00  3588.0  3573.1
CH    2
HY XS 3                22538.0     -79.40
DP        .00  0.000000E+00  0.000000E+00  1.00  1.00  1.00   557.0   557.0
DP      12.36  0.125007E+05  0.287426E+07  1.00  1.00  1.00  1465.8  1469.9
DP      20.76  0.275014E+05  0.810829E+07  1.00  1.00  1.00  2105.8  2227.0
DP      28.63  0.455024E+05  0.180114E+08  1.00  1.00  1.00  2468.8  2368.6
DP      37.52  0.671035E+05  0.329484E+08  1.00  1.00  1.00  2390.8  2528.4
DP      47.51  0.930249E+05  0.544861E+08  1.00  1.00  1.00  2798.6  2690.3
DP      59.13  0.124131E+06  0.872188E+08  1.00  1.00  1.00  2555.3  2732.2
DP      72.66  0.161457E+06  0.129103E+09  1.00  1.00  1.00  2962.3  2927.4
DP      87.72  0.206249E+06  0.188581E+09  1.00  1.00  1.00  2986.2  3058.2
DP     105.72  0.260000E+06  0.275260E+09  1.00  1.00  1.00  2986.1  3094.2
HY XS 2                27197.0     -93.20
DP        .00  0.000000E+00  0.000000E+00  1.00  1.00  1.00      .1      .1
DP      34.17  0.125007E+05  0.456299E+07  1.00  1.00  1.00   731.6   734.8
DP      49.56  0.275014E+05  0.117304E+08  1.00  1.00  1.00  1217.8  1279.8
DP      60.86  0.455024E+05  0.197191E+08  1.00  1.00  1.00  1968.2  2067.7
DP      68.10  0.671035E+05  0.246931E+08  1.00  1.00  1.00  3998.9  3897.0
DP      73.96  0.930249E+05  0.369072E+08  1.00  1.00  1.00  4847.9  4825.8
DP      80.06  0.124131E+06  0.555847E+08  1.00  1.00  1.00  5350.8  5370.3
DP      86.67  0.161457E+06  0.806372E+08  1.00  1.00  1.00  5943.0  5930.3
DP      93.78  0.206249E+06  0.114352E+09  1.00  1.00  1.00  6656.7  6476.7
DP     102.10  0.260000E+06  0.167933E+09  1.00  1.00  1.00  6264.2  6493.3
```

Figure 8: Example irregular cross-section data file.

Output index: `(I1)`

Index indicating data input

[**0**] `will not, or`

[**1**] `will`

be written to the print file (default name `netprint.dat`).

Channel indicator record `(A2, I10)`

field 1 `CH`

field 2 *Channel number.*

Cross-section header record `(A2,1X, A16,1X, F10.0,1X, F10.0)`

field 1 `HY`

field 2 *Cross-section identifier.*

field 3 *Downstream reference distance.*

field 4 *Elevation of lowest point in cross section.*

Properties record `(A2, 1X, F10.0, 2(1X,E13.6), 3(1X,F5.0), 2(1X,F7.0))`

field 1 `DP`

field 2 *Depth corresponding to hydraulic properties.*

Depth is the distance from the lowest point in the cross section to the elevation corresponding to hydraulic properties.

field 3 *Cross-sectional area.*

field 4 *Conveyance.*

field 5 *Momentum coefficient.*

field 6 *Area-weighted sinuosity.*

field 7 *Flow-weighted sinuosity.*

field 8 *Width.*

field 9 *Wetted perimeter.*

The properties record is repeated for each chosen depth. The minimum number of properties records per cross section is 2, the first of which must correspond to the lowest point in the cross section. The maximum number of records per cross section is limited by FORTRAN parameter `MaxLinesPerTable` contained in the include file `chcxtble.inc`. The number of records per cross section must be identical for all cross sections within an individual channel.

Constraint Properties

In FOURPT constraining equations, referred to as constraints, provide functional descriptions of the connections among channels. Simple constraints such as those enforcing mass conservation at the junction of two or more channels or forcing equal water-surface elevations in adjacent cross sections of two connecting channels require only that a user set appropriate boundary conditions in the schematic-data file (See *Schematic Description* under *Input*). Other constraints such as those enforcing unique three-parameter relations among headwater, tailwater, and flows through hydraulic structures require additional data. Preparation and use of constraint-properties files are optional. Such files will only be read and used if appropriate boundary conditions are set in the schematic-data file.

FOURPT, version 95.01, has the facility to use two different types of constraints that require input of constraint properties. The first type, the unique three-parameter constraint, requires a table of headwater elevations corresponding to individual tailwater elevations and flows through a hydraulic structure. The second type is a user-programmable constraint. Data requirements for it depend on its implementation. Both types are described below.

Three-parameter constraint.– – The three-parameter constraint may be used to simulate virtually any hydraulic structure that can be characterized by a unique three-parameter relation (DeLong and Fulford 1993) such as culverts, weirs, and highway crossings. Two different three-parameter tables may be used at one location if the two table identifiers differ only by sign. The table with positive identifier is used for positive flow, and the table with negative identifier is used for negative flows. If a table with like number but negative sign does not exist, the table with positive identifier will be used for all flow. An example of a three-parameter constraint data file is shown in figure 9 and described below:

Constraint-type record (*One per file.*)

field 1 `tables30.dat` *Columns 1-12.*

Table delimiter record (*One at the start of each table.*)

field 1 `TAB` *Columns 1-3*

Table header record `(I10,1X,I2,1X,I2)`

field 1 *Table identifier.*

The table identifier is a unique integer of length 10 or less. The last digit (either a 1 or a 2) indicates whether the constraint will be applied to (1) water-surface elevation or to (2) discharge. The second to last digit is a 3 indicating the table represents a three-parameter relation. Numbers appearing to the left of the last two numbers identify the specific three-parameter relation. For example, the table identifier of the third table in figure 9 is the number 231. The last digit, 1, indicates that it applies to water-surface elevation. The second to last

```
tables30.dat
TAB
       131 10 10
0.0
     .0   1.60   2.00   2.50   3.00   3.50   4.00   4.50   5.00   6.46
     20.     40.     60.     80.    100.    125.    150.    180.    210.    250.
   2.80   2.80   2.80   2.81   3.16   3.60   4.07   4.55   5.04   6.50
   3.61   3.61   3.61   3.61   3.61   3.86   4.25   4.68   5.15   6.62
   4.29   4.29   4.29   4.29   4.29   4.31   4.53   4.89   5.31   6.82
   4.90   4.90   4.90   4.90   4.90   4.90   4.95   5.17   5.52   7.10
   5.47   5.47   5.47   5.47   5.47   5.47   5.46   5.55   5.80   7.46
   6.14   6.14   6.14   6.14   6.14   6.14   6.14   6.13   6.25   8.02
   7.49   7.49   7.49   7.49   7.49   7.49   7.49   7.49   7.49   8.71
   8.84   8.84   8.84   8.84   8.84   8.84   8.84   8.84   8.84   9.70
  10.45  10.45  10.45  10.45  10.45  10.45  10.45  10.45  10.45  10.87
  12.90  12.90  12.90  12.90  12.90  12.90  12.90  12.90  12.90  12.90
TAB
       132 10 10
0.0
     .0   1.60   2.00   2.50   3.00   3.50   4.00   4.50   5.00   6.46
     20.     40.     60.     80.    100.    125.    150.    180.    210.    250.
   2.80   2.80   2.80   2.81   3.16   3.60   4.07   4.55   5.04   6.50
   3.61   3.61   3.61   3.61   3.61   3.86   4.25   4.68   5.15   6.62
   4.29   4.29   4.29   4.29   4.29   4.31   4.53   4.89   5.31   6.82
   4.90   4.90   4.90   4.90   4.90   4.90   4.95   5.17   5.52   7.10
   5.47   5.47   5.47   5.47   5.47   5.47   5.46   5.55   5.80   7.46
   6.14   6.14   6.14   6.14   6.14   6.14   6.14   6.13   6.25   8.02
   7.49   7.49   7.49   7.49   7.49   7.49   7.49   7.49   7.49   8.71
   8.84   8.84   8.84   8.84   8.84   8.84   8.84   8.84   8.84   9.70
  10.45  10.45  10.45  10.45  10.45  10.45  10.45  10.45  10.45  10.87
  12.90  12.90  12.90  12.90  12.90  12.90  12.90  12.90  12.90  12.90
TAB
       231  9 10
0.0
     .0   3.20   4.00   5.00   5.50   6.00   7.00   7.90   9.00
     20.     40.     60.    100.    150.    200.    250.    300.    400.    500.
   4.58   4.58   4.58   5.08   5.55   6.03   7.02   7.92   9.02
   5.33   5.33   5.33   5.33   5.70   6.13   7.08   7.97   9.08
   5.94   5.94   5.94   5.94   5.94   6.30   7.18   8.05   9.17
   6.98   6.98   6.98   6.98   6.98   6.86   7.51   8.33   9.48
   8.15   8.15   8.15   8.15   8.15   8.15   8.16   8.88  10.07
   9.34   9.34   9.34   9.34   9.34   9.20   9.09   9.69  10.91
  10.72  10.72  10.72  10.72  10.72  10.72  10.33  10.90  11.98
  11.92  11.92  11.92  11.92  11.92  11.92  11.92  11.92  13.29
  15.60  15.60  15.60  15.60  15.60  15.60  15.60  15.60  16.64
  20.41  20.41  20.41  20.41  20.41  20.41  20.41  20.41  20.93
```

Figure 9: Example three-parameter constraint file.

digit, 3, indicates that it is a three-parameter relation. The number appearing to the left of the last two digits, 2, is an identifier (1 to 8 digits) assigned by the user.

In order to apply the three-parameter relation, the table identifier is entered in the schematic-data file as the boundary-condition code for the specific channel constrained. (See *Schematic description*).

field 2 *Number of tailwater depths.*

This integer value is the number of tailwater depths (columns) in the three-parameter table.

field 3 *Number of discharges.*

This integer value is the number of discharges (rows) in the three-parameter table.

Datum record `(F9.3)`

This floating-point value is the elevation of the structure above the common datum. All depths in the three-parameter table are measured from this datum.

Tailwater depths record (*Free field format.*)

These floating-point values are a list, monotonically increasing, of tailwater depths (distance above datum) corresponding to columns of the three-parameter table. Number of depths must agree with the number of tailwater depths specified in the `Table header record` above.

Discharges record (*Free field format.*)

These floating-point values are a list, monotonically increasing, of discharges corresponding to rows of the three-parameter table. Number of discharges must agree with the number of discharges specified in the `Table header record` above.

Headwater-table record (*Free field format.*)

These floating-point values are a list of headwater depths (distance above datum) corresponding to paired tailwater depth and discharge values entered above. Values are listed in order of increasing tailwater depth beginning with the smallest discharge. When all headwater depths have been entered for the lowest discharge, the sequence is repeated for the second and following discharges in succession. There must be a headwater depth for each possible combination of tailwater depth and discharge.

User-programmable constraint.– – The user-programmable type constraint is a facility added to FourPt to simplify programming of constraining equations. Modification of the FourPt computer code is isolated to two existing FourPt routines, and the bulk of any required programming is accomplished in added routines independent of existing code. Three examples of constraints programmed using this facility are contained in the channel constraint computer code file (*chcnstrt.f*). They implement constraints representing a logarithmic stage-discharge relation, a semi-logarithmic stage-discharge relation, and a more complex example, dam-break functions (enlarging

trapezoidal breach, flow over dam, flow over spillway, and flow through gates) patterned after the National Weather Service DAMBREAK model (Fread 1984). An example application of two of these constraints is found in the section *Applying User-Programmed Constraints* under *EXAMPLES*. The basic process of programming a constraint is briefly given here. Persons attempting to add constraints should be familiar with the FORTRAN 77 computer language (American National Standards Institute 1978), the use of FORTRAN 77 modules (DeLong et al. 1992; Thompson et al. 1992), hydraulic and numerical properties of the structure to be simulated, and implicit numerical representation of equations using truncated Taylor series as demonstrated in section *Linearization Over a Time Step* under *Numerical Formulation*. Persons not concerned with programming constraints need not read the balance of this section.

Boundary-condition codes that invoke user-programmable constraints are assigned in the following manner. Any boundary-condition code with the digit 5 in the second to last position is assumed to refer to a user-programmable constraint. The third and fourth from last digits identify the specific type of user-programmable constraint. Preceding digits indicate specific instances of a specific type of programmable constraint. For example, suppose a logarithmic stage-discharge relation is implemented as a user-programmable constraint. A boundary-condition code that would invoke a specific instance of the relation might be 40352. The 5 indicates that it refers to a user-programmable constraint, the 03 refers to the logarithmic relation, and the 4 refers to the specific instance. The 03 was originally selected by the programmer of the constraint to identify the logarithmic relation, and it must remain unique to that type of constraint. A total of 99 types of user-programmable constraints may be identified. There may be many instances of the logarithmic constraint presumably involving different parameters. The particular instance in the example is 4. The last digit passes additional information to FourPt regarding the nature of coefficients in the constraint equation. The digit may be 1 or 2. If the digit is 1, it indicates that the coefficient of water-surface elevation at the constraint location will remain non-zero during model execution, and if 2, indicates that the coefficient of discharge at the constraint location will remain non-zero during model execution. This is of importance to the programmer of a constraint in that the unexpected occurrence of a zero coefficient during execution will cause FourPt to fail.

User-programmable constraints work in the following manner. During the initial processing phase of FourPt execution, boundary-condition codes and connecting channel numbers are read from the schematic-data file. If any boundary-condition codes associated with user-programmable constraints are encountered, the function *InitUserStreamConstraint* is invoked. This function is invoked only once, at most, during program execution. It reads the user-programmable constraint data file and invokes any initialization functions required by the user-programmed constraints encountered. *InitUserStreamConstraint* should be modified to appropriately invoke initialization functions required by any newly-programmed constraints. A second function *ForceUserStreamConstraint* is invoked periodically throughout FourPt execution to compute user-programmable constraint-equation coefficients. This function should be modified to appropriately invoke computational routines specific to added constraints. Computed coefficients are inserted into a coefficient matrix occupied by other equation coefficients comprising the overall solution. To summarize, a person wishing to add a constraint would program an initialization function, modify *InitUserStreamConstraint* to invoke it, program a function to compute the coefficients of the new constraint equation,

and modify *ForceUserStreamConstraint* to invoke it.

The coefficient-computing routine, programmed by a person adding a new constraint, should use (but not modify) the FORTRAN COMMON block *UserConstraintVariables* contained in the file `strmcnst.inc` to store constraint-equation coefficients. Coefficients in this COMMON block are later inserted into the coefficient matrix by existing routines that do not require modification. Also, current values of discharge and water-surface elevations at the constraint location and extremities of any connecting channels are available in this COMMON block which is documented in the computer code. Persons adding constraints should review examples of the implementation of user-programmed constraints contained in the file `chcnstrt.f`. For an example of the application of user-programmed constraints, see *Applying User-Programmed Constraints* under *EXAMPLES*.

File Names and Unit Numbers

Default names for file input to and output from FourPt, version 95.01, are assigned in the computer code within the master-file module. Default names for specific files are presented in sections of this document where the corresponding input or output is described. The maximum length of any file name is 12 characters.

File names used by the program may be changed by editing the file *master.fil*, an example of which is shown in figure 10. If this file is located by FourPt during execution, it will automatically be read. Information contained in the file is used to modify default names and FORTRAN unit numbers. FourPt writes a new master file-name and unit-number file during each execution. The file written (*master.fil*) contains default names and unit numbers of files actually used and/or previously supplied through edits of file *master.fil*. File format is:

File-name unit-number record (3 *fields separated by blanks*, 80 *characters or less.)*

field 1 *Default file name.*

field 2 *File name used or to be used by program.*

filed 3 *Unit number used or to be used by program.*

Rather than initially creating the master file-name and unit-number file, a user may simply execute FourPt and edit the file produced by the program if names or unit numbers are to be changed. Only fields two and three should be modified. Field one is the key used to match filenames internal to the program with filenames supplied or modified by the user in field two.

Initial Values

Initial values of dependent variables may be read from a file by FourPt or computed during program execution. Different options may be selected (in the schematic-data file) for each channel. The default name for the optional file containing initial values to be read is `initcond.dat`. The file will only be required if the option to read initial conditions is selected for one or more channels.

```
control.dat  SwampRat.dat   57
schemat.dat  schemat.dat    58
cxgeom.dat   cx_rect.dat    30
netspcz.dat  netspcz.dat    31
netbnd.dat   netbnd.dat     32
netts.dat    netts.dat      33
neterror.dat neterror.dat   39
netspcq.dat  netspcq.dat    40
netprint.dat netprint.dat   41
nettsq.dat   nettsq.dat     42
nettsz.dat   nettsz.dat     43
density.dat  netdense.dat   44
```

Figure 10: Example master file-name and unit-number file.

If computational locations in the current execution of FourPt do not match locations at which dependent variables are specified in the initial-value file, the necessary values are linearly interpolated . Values are not extrapolated . . . values must be present for channel extremities. An example initial-value file is shown in figure 11.

To simplify restarting FourPt from a condition obtained from a preceding execution, format of this file is similar to the restart file optionally written at the conclusion of an execution. Free-field format is used throughout. Labels are included to provide better identification but are not read or used by FourPt. Description of the initial-values file follows:

Channels record (*One per file.*)

field 1 *Number of channels.*

Total number of channels for which initial values are to be read, limited by FORTRAN parameter `MaxChannels` in the include file `network.inc`.

Channel identifier record (*One per channel.*)

field 1 *Channel number.*

Channel number of channel. Channel numbers need not be sorted. Channel numbers are assumed to refer to channel numbers in the current execution of the model.

Locations per channel (*One per channel.*)

field 1 *Locations.*

Number of locations in channel for which initial values will be supplied.

```
2/Channels
1/Channel number
8/Locations
     90000.000          62.456          414.962 /Distance, WSElev, Q
     90625.000          61.926          446.125 /Distance, WSElev, Q
     91250.000          61.385          474.897 /Distance, WSElev, Q
     91875.000          60.835          501.283 /Distance, WSElev, Q
     97500.000          56.070          587.684 /Distance, WSElev, Q
     98125.000          55.836          561.716 /Distance, WSElev, Q
     98750.000          55.691          528.455 /Distance, WSElev, Q
    100000.000          55.555          451.934 /Distance, WSElev, Q
2/Channel number
17/Locations
    100000.000          52.427          451.934 /Distance, WSElev, Q
    100625.000          51.776          445.107 /Distance, WSElev, Q
    101250.000          51.122          437.378 /Distance, WSElev, Q
    101875.000          50.466          428.772 /Distance, WSElev, Q
    102500.000          49.807          419.338 /Distance, WSElev, Q
    103125.000          49.145          409.154 /Distance, WSElev, Q
    103750.000          48.481          398.325 /Distance, WSElev, Q
    104375.000          47.816          386.987 /Distance, WSElev, Q
    105000.000          47.150          375.302 /Distance, WSElev, Q
    105625.000          46.483          363.451 /Distance, WSElev, Q
    106250.000          45.817          351.631 /Distance, WSElev, Q
    106875.000          45.151          340.040 /Distance, WSElev, Q
    107500.000          44.487          328.868 /Distance, WSElev, Q
    108125.000          43.825          318.284 /Distance, WSElev, Q
    108750.000          43.165          308.428 /Distance, WSElev, Q
    109375.000          42.508          299.404 /Distance, WSElev, Q
    110000.000          41.855          291.277 /Distance, WSElev, Q
```

Figure 11: Example initial-values file.

Initial values record (*One per location.*)

field 1 *Downstream distance.*

field 2 *Water-surface elevation.*

field 3 *Flow.*

Boundary Values

Instantaneous boundary values may be read from a user-supplied boundary-value file. The default name for this file is *bndval.dat*. Values will be read from this file only if the appropriate flag is set in the program control file (see *Program Control* under *Input*). Free-field format is used throughout the boundary-values file. Labels are included to provide better identification but are not read or used by FourPt. A partial example file is shown in figure 12. Description of the boundary-values file follows:

Start-time record (*One per file.*)

field 1 *Starting elapse time.*

Starting elapse time (integer) is the value of the starting time, in seconds, corresponding to the first set of values.

Time increment record. (*One per file.*)

field 1 *Time increment.*

Time increment (integer), in seconds, is the time interval between sets of boundary values. The boundary-values time increment does not have to match the time increment used by FourPt during execution. Boundary values will be linearly interpolated as required.

Values-per-time record (*One per file.*)

field 1 *Items per record.*

This integer value is the number of boundary values per set of values corresponding to each point in time.

Channel identifier record (*One for each value at a point in time.*)

field 1 Channel identifier.

This integer value is the channel number to which boundary values, listed later in the file, will apply. Positive or no sign indicates the upstream end of the channel, and negative sign indicates the downstream end of the channel. Channel numbers are listed in a sequence corresponding to boundary values supplied for each point in time.

```
0           / starting elapse time, in seconds (integer).
900         / time-series time increment, in seconds (integer).
2           / number of values for each point in time.
1           / channel number; + upstream or - downstream.
-1          / channel number; + upstream or - downstream.
     2.360  / value for upstream end of channel 1 at starting elapse time.
     2.530  / value for downstream end of channel 1 at starting elapse time.
     2.510  / value for upstream end of channel 1, end of first time step.
     2.670  / value for downstream end of channel 1, end of first time step.
     2.620  / repeat last two lines for all time increments....
     2.710
     2.690
     2.750
     2.740
     2.740
     2.770
     2.700
     2.760
     2.590
     2.710
     2.460
     2.640
     2.330
     2.520
     2.130
     2.350
     1.920
     2.160
```

Figure 12: Example boundary-values file.

Boundary-value record (*One for each item per record for each point in time.*)

field 1 *Boundary value.*

A boundary value is a floating-point number.

Just how or if the boundary values read by FourPt are used depends on the boundary-condition code set in the schematic-data file (See *Schematic Description* under *Input*). For example if a boundary-condition code for a specific end of a specific channel is set equal to 2 (known discharge) and a boundary value is read for that location, the value would be applied as a known discharge. Otherwise, if the boundary-condition code were set equal to 1 (known water-surface elevation), the value would be applied as a known water surface. **Equation-type and read-boundary values can be mixed in a network, but read values override when equation and read values both exist for a location**. Equation-type boundary values are described under *Input* in the section *Program Control*.

Density Values

FourPt optionally allows density to vary with space and time. FourPt version 95.01, does not include transport computations. Consequently, if variable-density simulation is desired, time series

```
0            / starting elapse time, in seconds (integer).
900          / time-series time increment, in seconds (integer).
     1.00980  / value for upstream end of channel 1 at starting elapse time.
     1.01420  / value for downstream end of channel 1 at starting elapse time.
     1.00980  / value for upstream end of channel 2 at starting elapse time.
     1.01420  / value for downstream end of channel 2 at starting elapse time.
     1.00990  / value for upstream end of channel 1, end of first time step.
     1.01480  / value for downstream end of channel 1, end of first time step.
     1.00990  / value for upstream end of channel 2, end of first time step.
     1.01480  / value for downstream end of channel 2, end of first time step.
     1.01010  / repeat last four lines (2 per channel) for all time increments...
     1.01510  /
     1.01020  /
     1.01540  /
     1.01030  /
     1.01550  /
     1.01050  /
     1.01550  /
     1.01070  /
     1.01560  /
     .
     .
     .
     .
```

Figure 13: Example density-values file for a two-channel network.

of instantaneous density values for the duration of a model run are input for both ends of each channel. Intervening density values are approximated by linear interpolation.

Instantaneous density values are read from a user-supplied density-value file. The default name for this file is `density.dat`. An example of such a file for a two-channel network is shown in figure 13.

Values will be read from this file only if the appropriate flag is set in the program control file (see *Program Control* under *Input*). Free-field format is used throughout the density-values file. Labels are included to provide better identification but are not read or used by FourPt. Description of the density-values file follows:

Start-time record (*One per file.*)

field 1 *Starting elapse time.*

Starting elapse time (integer) is the value of the starting time, in seconds, corresponding to the first set of values.

Time increment record. (*One per file.*)

field 1 *Time increment.*

Time increment (integer), in seconds, is the time interval between sets of density values. The density-values time increment does not have to match the time increment used by FourPt during execution. Density values will be linearly interpolated as required.

```
12345       / pseudo-random-number-generator seed (between 1 and 2147483646).
0           / cumulative number of model runs.

-1          / Channel number (+ upstream end, - downstream end).
1           / Number of sinusoidal equation components.

0.02        / Std. dev. of EqBase (among model runs).
0.00        / Std. dev. of EqBase (among time step).

0.00        / Std. dev. of Amplitude (among model runs).
0.00        / Std. dev. of Amplitude (among time step).
0.00        / Std. dev. of Period (among model runs).
0.00        / Std. dev. of Period (among time step).
0.00        / Std. dev. of PhaseAngle (among model runs).
0.00        / Std. dev. of PhaseAngle (among time step).
```

Figure 14: Example perturbation-parameter file.

Density-value record (*One value for each end of each channel for each point in time.*)

field 1 *Density value.*

A density value is a floating-point number.

Perturbation Parameters

It is sometimes desirable to determine the sensitivity of model results to small changes or perturbations in boundary values. Such knowledge may be obtained simply by systematically changing or perturbing boundary values and recording results of successive model runs. Where boundary values are computed from equations with user-supplied parameters, FourPt, version 95.01, can optionally perturb boundary values, if the appropriate flag and equation parameters are set in the program control file (see *Program Control* under *Input* for description of flags and boundary-equation parameters). Two perturbation parameters exist for each boundary-value equation parameter. The first perturbation parameter is the standard deviation of perturbations among model runs. It is applied once at the beginning of a model run and held constant for the duration of a model run. The second perturbation parameter is the standard deviation of perturbations applied once each time step of a model run. The perturbations are quasi-random numbers (Kirby 1983) drawn from standard normal (Gaussian) distributions. Perturbations are applied as multipliers of unperturbed values. Standard deviations of the perturbations for each boundary-equation parameter are set in a user-supplied file with default name *perturb.dat*. Free-field format is used throughout the perturbation-parameter file. An example of this file, set up to perturb values at a downstream boundary, is shown in figure 14.

Labels are included to provide better identification but are not read or used by FourPt. Description of the perturbation-parameter file follows:

Seed record (*One per file.*)

field 1 *Quasi-random number generator seed.*

The seed is an integer between 1 and 2147483646.

Counter record. (*One per file.*)

field 1 *Cumulative number of model runs.*

This integer value is increased by one each execution of the model when the perturbation feature is active. Thus, if initially set to zero, it will indicate the number of model runs executed in the current sequence of model runs.

Channel-identifier record (*One per boundary perturbed.*)

field 1 *Channel number.*

Integer specifies to which channel a perturbation refers. If positive, it relates to the upstream end of the channel, and if negative, it refers to the downstream end of the channel.

Equation components record. (*One per boundary perturbed.*)

field 1 *Number of sinusoidal components.*

Integer is the number of sinusoidal components used in the current equation. This value must match the number of components entered in the program control file (see *Program Control* under *Input*).

First equation-base perturbation record (*One per boundary perturbed.*)

field 1 *First standard deviation of equation base.*

Floating-point number is the standard deviation of the base value (about which sinusoidal components oscillate). This is the standard deviation of the distribution of values among model runs. The perturbation applied remains constant throughout a model run.

Second equation-base perturbation record (*One per boundary perturbed.*)

field 1 *Second standard deviation of equation base.*

Floating-point number is the standard deviation of the base value (about which sinusoidal components oscillate). This is the standard deviation of the distribution of values within a model run. A different perturbation is applied each time step throughout a model run.

First amplitude perturbation record (*One per sinusoidal component.*)

field 1 *First standard deviation of amplitude.*

Floating-point number is the standard deviation of the amplitude of a sinusoidal component. This is the standard deviation of the distribution of values among model runs. The perturbation applied remains constant throughout a model run.

Second amplitude perturbation record (*One per sinusoidal component.*)

field 1 *Second standard deviation of amplitude.*

Floating-point number is the standard deviation of the amplitude of a sinusoidal component. This is the standard deviation of the distribution of values within a model run. A different perturbation is applied each time step throughout a model run.

First period perturbation record (*One per sinusoidal component.*)

field 1 *First standard deviation of period.*

Floating-point number is the standard deviation of the period of a sinusoidal component. This is the standard deviation of the distribution of values among model runs. The perturbation applied remains constant throughout a model run.

Second period perturbation record (*One per sinusoidal component.*)

field 1 *Second standard deviation of period.*

Floating-point number is the standard deviation of the period of a sinusoidal component. This is the standard deviation of the distribution of values within a model run. A different perturbation is applied each time step throughout a model run.

First phase-angle perturbation record (*One per sinusoidal component.*)

field 1 *First standard deviation of phase-angle.*

Floating-point number is the standard deviation of the phase-angle of a sinusoidal component. This is the standard deviation of the distribution of values among model runs. The perturbation applied remains constant throughout a model run.

Second phase-angle perturbation record (*One per sinusoidal component.*)

field 1 *Second standard deviation of phase-angle.*

Floating-point number is the standard deviation of the phase-angle of a sinusoidal component. This is the standard deviation of the distribution of values within a model run. A different perturbation is applied each time step throughout a model run.

Default filename	*Contents*
`netprint.dat`	general results
`netrstrt.dat`	initial conditions for restarting
`netspcq.dat`	discharge space series
`netspcz.dat`	depth-of-flow space series
`nettsq.dat`	discharge time series
`nettsz.dat`	water-surface-elevation time series

Table 4: Output files.

OUTPUT

FourPt, version 95.01, writes up to six files of interest to the model user. Information in the files (table 4) includes general model results and messages, data necessary to restart the model with initial conditions identical to when it concluded, space series of depths of flow and discharge, and time series of water-surface elevation and discharge. Flags set in the program-control data file determine when and how any of these output files are written. (See *Program Control* under *Input*).

General Results

The general-results file (default filename `netprint.dat`) contains program-control data, selected results, and warning and error messages that occur during model execution. At the completion of each model run volume balance and mass balance (if variable density is in effect) are summarized. An abbreviated example of this file is shown in figure 15.

Restart

Information contained in the restart file, is similar to the initial values file. (See *Initial Values* under *Input*). This file is optionally written at the conclusion of a model run. It can be substituted for an initial-values file to resume simulation under conditions reached at the conclusion of an earlier model run.

Space Series

The space-series file contains values of depth of flow or discharge at all computational locations at specific points in time. An example of a space-series file containing depth-of-flow values is shown in figure 16. Fields are tab delimited . . . tabs exist between columns 1 and 2, 2 and 3, and between any additional columns. The first line is a header record containing, left to right, a string identifier of the first column *Distance*, time in hours at which the first space series was collected, and time in hours at which the second space series was collected.

The second and following lines contain, from left to right, downstream distance, corresponding depth of flow in the first space series, and corresponding depth of flow in the second space series.

```
             .
             .
 Full dynamic-wave solution...
 Water density is assumed constant.
 Acceleration due to gravity =   32.2000
 Maximum time steps =   24
 Starting elapse time, in seconds, =   0
 Time increment, in seconds, =   300
 Time-weighting factor =   1.00000
 Number of quadrature points =   1
  local coordinates..... weights...
        .500000       1.000000
 Maximum number of iterations per time step =   9
 Interval for complete forward eliminations =   4
 Tolerance for closure on discharge =   5.000000E-03
 Tolerance for closure on water-surface elevation =   5.000000E-03

 Level of printing activity =   1
 Initial value of print counter =   0
 Print increment =   24
 Initial value of time-series output counter =  0
 Time-series increment =   1

 Time-series output selected for  2  Location(s):
 Channel    Location
    1     99500.00
    2    100500.00

 Initial value of spatial-series output counter =  0
 Spatial-series output increment =   24
             .
             .
  Time          ....Q....            ....Z....
  Step Iter. Channel  Section     Channel  Section
   24    3       1      17            2       1

        Channel...  1

CompCxNo  Location   Discharge   WS_Elev     Depth
   1     90000.00     414.96      62.46       2.46
    .         .          .          .          .
    .         .          .          .          .
  17    100000.00     506.97      56.99       6.99
             .
             .
                              ___Volume Balance___

 Channel   Initial vol.    Final vol.     Junction       Lateral    Difference

       1    3374468.20    3819365.00     444893.40           .00          3.13
       2    8556511.00   10179270.00    1622762.00           .00         -6.00
 ------------------------------------------------------------------------------
           11930980.00   13998632.00    2067655.30           .00         -2.88
```

Figure 15: Example general-results file.

```
Distance             .00            6.67
            .00            7.20            7.83
         400.00            7.20            7.75
         800.00            7.20            7.68
        1200.00            7.20            7.63
        1600.00            7.20            7.58
        2000.00            7.20            7.54
        2400.00            7.20            7.50
        2800.00            7.20            7.47
        3200.00            7.20            7.44
        3600.00            7.20            7.42
        4000.00            7.20            7.40
        4400.00            7.20            7.38
        4800.00            7.20            7.36
        5200.00            7.20            7.34
```

Figure 16: Example tab-delimited space-series file.

```
Time        41012.50        29529.00
            .00            67.09          105.93
            .25            45.40           88.52
            .50            42.17           74.56
            .75           -15.23           57.87
           1.00          -102.89           45.59
           1.25          -173.91           34.35
           1.50          -223.70           11.46
           1.75          -273.85          -19.87
           2.00          -339.60          -43.82
```

Figure 17: Example tab-delimited time-series file.

The maximum number of space series allowed is set by parameter in the include file `bufout.inc`.

Time Series

The time-series files contain water-surface elevations or discharges at specific locations for each time step computed by FourPt. An example of a time-series file containing discharges for two locations is shown in figure 17. Fields are tab delimited . . . tabs exist between columns 1 and 2, 2 and 3, and between any additional columns. The first line is a header record containing, left to right, a string identifier of the first column *Time*, the downstream distance at which the first time series was collected, and the downstream distance at which the second time series was collected.

The second and following lines contain, from left to right, time in hours, corresponding discharge in the first time series, and corresponding discharge in the second time series. The maximum number of time series allowed is set by parameter in the include file `netcntrl.inc`.

```
     1                   / channel number
      2, 12800., 1, 0, 0 / no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_1               / cx ID
          2.2,   1.0     / WS, Q
Downstream_1             / cx ID
          2.2,    1.0    / WS, Q

 2                       /upstream boundary-condition code
  0                      /connections

 1                       /downstream boundary-condition code
  0                      /connections
```

Figure 18: Example-1 schematic-data file.

EXAMPLES

Example applications of FourPt are presented to demonstrate program capability, provide tests for program modification and implementation on new computer systems, and demonstrate basic principles common to the general numerical simulation of surface-water flow.

Example 1 – Steady Flow in a Channel with Expanding Width

The object of this example is to demonstrate an effect of poor spatial convergence and to emphasize the need for testing numerical convergence. FourPt is applied to a nonprismatic channel. In cross section, the channel is rectangular. However, the width changes with longitudinal position. The length of the channel is 12,800 ft. The initial section is 100 ft wide. The terminal section is 800 ft wide. Bottom elevation is -5.0 ft throughout the reach. The upstream boundary condition begins with a discharge of 1.0 ft^3/s and increases gradually to a final discharge that remains constant at 4,000 ft^3/s, simulating steady flow for the balance of the simulation. The downstream boundary condition is a constant water-surface elevation of 2.2 ft. The flow resistance coefficient is 0.026.

Suppose the objective of this application is to determine the upstream water-surface elevation when flow is steady at 4,000 cubic feet per second. What Δx (distance between computational cross sections) should be used for this application of FourPt? To test numerical convergence with respect to space (that is, to determine an adequately small Δx), Δx is systematically varied in a sequence of model runs. The size of Δx is adjusted in the schematic data file (figure 18). Starting with 12,800 ft, Δx is reduced by half in each subsequent model run. Results of this sequence of model runs are shown in figure 19. For Δx from 12,800 ft to 1,600 ft, model results change significantly. Note that errors are particularly apparent at the upstream terminus of the reach. However, the simulated upstream depth of flow increases only 0.04 and 0.01 ft when Δx is reduced from 1,600 to 800, and from 800 to 400 ft respectively. As Δx is reduced, results become less dependent on the size of Δx, and the model is said to have converged numerically with respect to space. **The size of Δx required to obtain numerical convergence is application dependent and therefore, must be determined independently for each model application.** In this example applica-

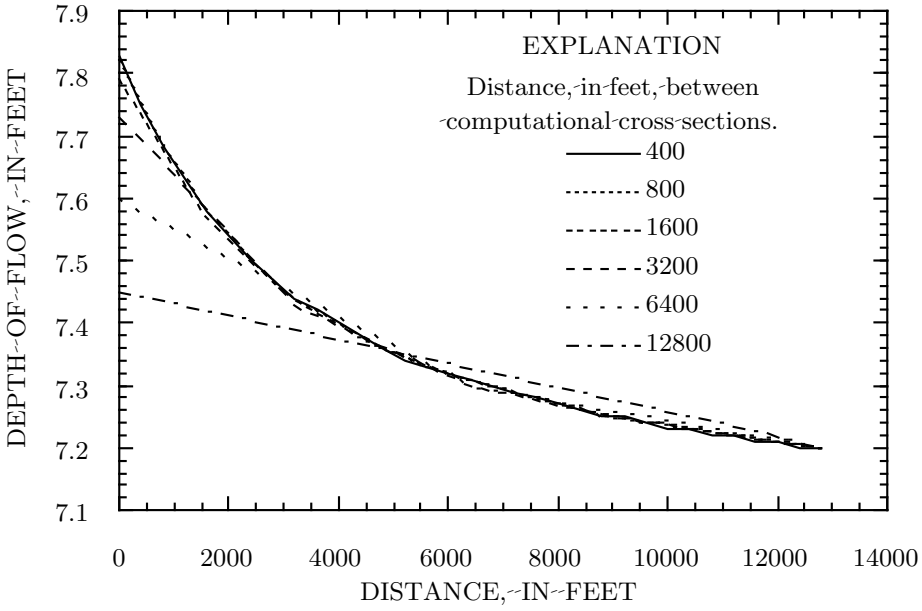

Figure 19: Spatial convergence in a channel with expanding width.

tion, a Δx of 400 ft or less would probably be considered sufficiently small, but for some uses of the results, 800 or perhaps even 1,600 ft might also be considered sufficiently small. Note that at downstream distances in excess of 1,600 ft, model results are essentially identical using Δx of 400 to 1,600 ft.

Example 2 – Unsteady Flow in a Prismatic Channel

The object of this example is to examine convergence properties of FourPt in both time and space, and to examine the impact of the time-weighting factor (θ) on solution of the converged model. The channel is rectangular in cross section and prismatic. Channel width is 100 ft, channel length is 70,000 ft, and bottom slope is 0.001. The flow-resistance coefficient is 0.045.

The upstream boundary condition is a specific discharge hydrograph, referred to as *Pete's Hydrograph* (Smith 1981), a specific cosine wave used to test flow models. The wave begins from a steady-state base of 250 ft^3/s at 0 seconds, rises to a maximum value of 727.465 ft^3/s at 4,500 seconds, and returns to the steady-state value of 250 ft^3/s at 9,000 seconds; discharge remains constant at 250 ft^3/s for the remainder of the simulation. When specifying an equation-type boundary condition in FourPt (figure 20), these values correspond to a base value of 488.733 ft^3/s, an amplitude of 238.733 ft^3/s, a period of 2.5 hours, and a phase angle of 1.25 hours.

FourPt was used to route Pete's hydrograph through Swamprat Creek Near Pete's Bayou. The initial Δx (figure 21) was 10,000 ft and Δt (figure 20) was 15 minutes (900 seconds). For each

```
   Hypothetical example, Swamprat Creek, Near Pete's Bayou
        Patterned after work by P. E. Smith, 1981.
1          / Terms, 1=dynamic, 2=diffusion, 3=kinematic
0          / 0 = contant density, 1 = variable density.
0          / 0 = constant sinuosity, 1 = variable sinuosity.
0          / 0 = do not read boundary values, 1 = read values.
0          / 0 = perturbation inactive, 1 = perturbation active.
32.2       / acceleration due to gravity.

150        / MaxTimeSteps, maximum number of time steps.
900.0      / DT, time step, in seconds.
0.6        / Theta, time-weighting factor.
0.5, 1.0   / local ( 0 to 1 ) quadrature-point coordinate and weight, paired.

5          / MaxIterations, maximum number of iterations per time step.
1          / LuInc, interval for complete forward eliminations.
0.005      / ToleranceQ, tolerance for closure on discharge.
0.005      / ToleranceY, tolerance for closure on water-surface elevation.

0          / PrintLevel, amount of printing, 0 to 9, increasing with number.
0          / PrintCount, initial value of print counter.
48         / PrintInc, print increment.
0          / TimeSeriesCount, initial value of time-series counter.
1          / TimeSeriesInc, time-series increment.
1,50000.0  / Requested time-series output, paired channel no. & downstream distance.

0          / SpaceCount, initial value of spatial-series output counter.
99999      / SpaceInc, spatial-series increment.


1          / ChannelNumber, for equation boundary (+ upstream, - downstream).
1          / Number of sinusoidal components.
488.733    / Base, base value for equation boundary condition.

238.733    / Amplitude.
2.5        / Period, in hours.
1.25       / Phase angle, in hours.

0.0        / EqStart, time at which equation takes effect.
2.5        / EqStop, time at which equation is no longer effective.


-1         / Channel number for equation boundary (+ upstream, - downstream).
0          / Number of sinusoidal components.
1.71       / Base, base value for equation boundary condition.

0.0        / EqStart, time at which equation takes effect.
2.5        / EqStop, time at which equation is no longer effective.
```

Figure 20: Example-2 program-control-data file.

```
     1                      / channel number
2,  10000.0, 1, 1, 0       / no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_1                 / cx ID
         71.7,       250.0 / WS, Q
Downstream_1               / cx ID
         1.7,        250.0 / cx no., WS, Q

 2                         /condition code upstream
  0                        /connections

 1                         /condition code downstream
  0                        /connections
```

Figure 21: Example-2 schematic-data file.

successive model run, both Δx and Δt were approximately halved. The sequence of simulated hydrographs (taken from a location 50,000 ft downstream from the upstream terminus of the channel) is shown in figure 22.

Clearly, as Δx and Δt are reduced, the shape of the modeled hydrograph changes. Leading phase error is apparent for $\Delta x \geq 2,500$ ft. However, for $\Delta x \leq 1,250$ ft, little change is apparent in the shape of the hydrograph. Only minor differences near the peak are evident. Selection of a Δx of 1,250 ft and a Δt of 112 seconds would be acceptable for most uses of model results.

Next, the impact of changes to θ is examined. A series of model runs were made using $\theta = 1$ and changing Δx and Δt to again test for numerical convergence. The results of this sequence of model runs are shown in figure 23. Note that when θ is increased from 0.6 to 1.0, it is necessary to decrease Δx and Δt to maintain comparable results. Results obtained with $\Delta x = 1,250$ ft, $\Delta t = 112$ seconds, and $\theta = 0.6$ are similar to results obtained with $\Delta x = 312.5$ ft, $\Delta t = 28$ seconds, and $\theta = 1.0$.

The effect of increasing θ is to dampen or attenuate results. This can be useful in cases where the model might otherwise fail to successfully conclude a run. However, the use of θ larger than 0.6 should be looked upon merely as a temporary measure to enable the model user to determine and correct causes of model failure or physically meaningless oscillations such as the leading-phase error shown in figure 22.

Example 3 – Diffusion and Kinematic Wave Approximations

The object of this example is to examine effects of simplifying assumptions that reduce the governing dynamic-wave equations to diffusion-wave or kinematic-wave equations. Selection of FORTRAN functions used by FourPt to represent the governing equations, *DynamicWave*, *DiffusionWave*, or *KinematicWave*, depends on the value specified for *TERMS* (1, 2, or 3, respectively) in the program control file. The function *DynamicWave* applies the dynamic flow equations to the problem. The functions *DiffusionWave* and *KinematicWave* apply simplified flow equations.

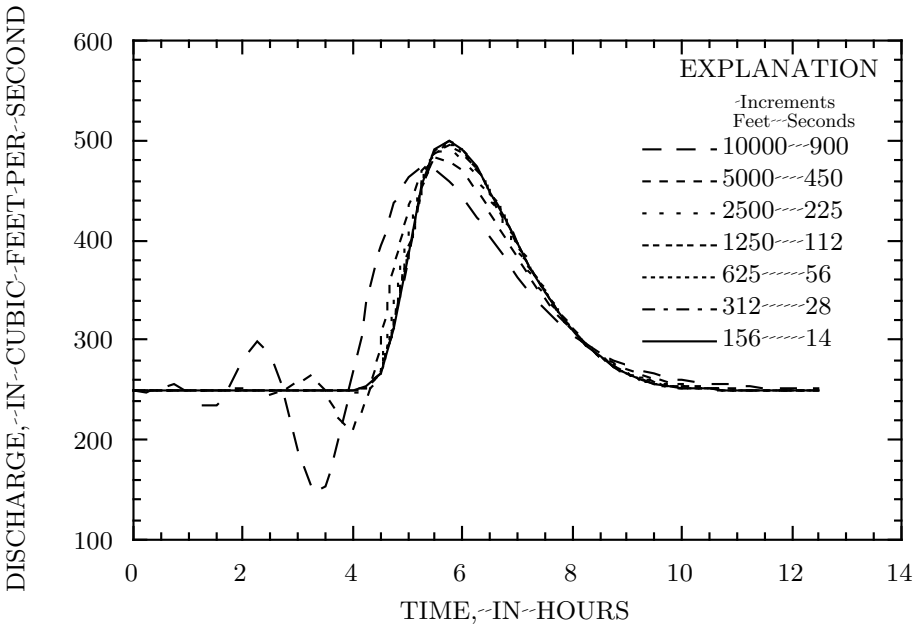

Figure 22: Spatial and temporal convergence in a prismatic channel, at a location 50,000 ft downstream.

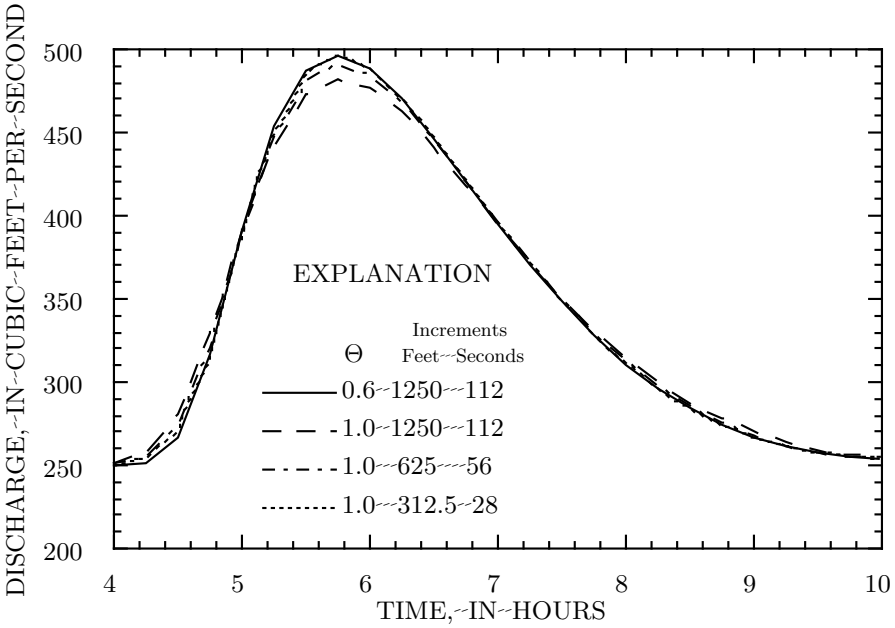

Figure 23: Impact of θ on numerical convergence.

For purpose of demonstration, the governing mass- and momentum-conservation equations (1 and 17) may be written

$$\frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} - q = 0, \tag{69}$$

and

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(\beta \frac{Q^2}{A}\right) + gA\left(\frac{\partial Z}{\partial x} + \frac{Q\,|Q|}{K^2}\right) = 0, \tag{70}$$

assuming constant density and sinuosity. Simplifications are obtained by modifying the momentum equation (equation 70). The first term of this equation represents the change with time of momentum stored in the channel. The second term represents the change in momentum flux with channel distance. The diffusion-wave approximation neglects both terms, resulting in

$$gA\left(\frac{\partial Z}{\partial x} + \frac{Q\,|Q|}{K^2}\right) = 0. \tag{71}$$

To obtain the kinematic approximation, equation 71 is first expanded through substitution of equation 14 in equation 71

$$gA\left(\frac{\partial Z_0}{\partial x} + \frac{\partial h}{\partial x} + \frac{Q\,|Q|}{K^2}\right) = 0, \tag{72}$$

and the second term is neglected, resulting in

$$gA\left(\frac{\partial Z_0}{\partial x} + \frac{Q\,|Q|}{K^2}\right) = 0. \tag{73}$$

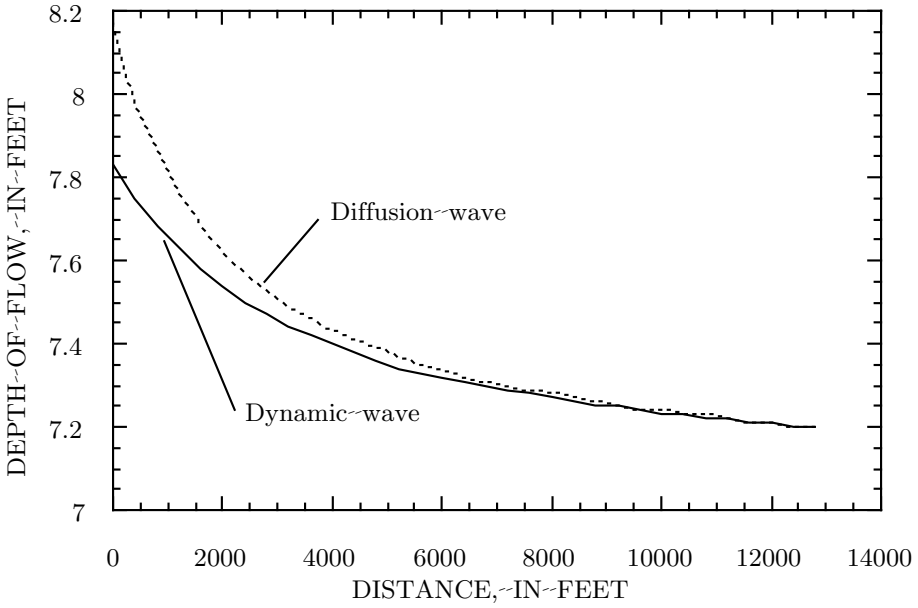

Figure 24: Comparison of dynamic- and diffusion-wave formulations on steady flow in the nonprismatic channel of example 1.

There are no known advantages to the use of diffusion- or kinematic-wave equations in the FourPt code. The simplification does not result in significantly faster executing code as it may in other codes specifically written to solve such equations. The option is included in FourPt only to allow users to conveniently experiment with and determine effects of such approximations in nontrivial applications.

The channels previously described in exercises 1 and 2 are used to examine the effects of the simplifying assumptions. A comparison of the dynamic- and diffusion-wave equations (equations 70 and 71, respectively) applied to the steady-flow problem (nonprismatic channel) is shown in figure 24.

Clearly, there are substantial differences between results of the dynamic- and diffusion-wave equations. The water-surface profile resulting from use of the kinematic-wave approximation doesn't reach steady state as shown in figure 25. The upstream water-surface elevation continues to rise with each additional time step.

The kinematic form of the one-dimensional flow equations simply forces friction slope and bottom slope to be equal. In this example, the bed slope is zero, hence the problem is ill posed when the kinematic-wave approximation is selected. A zero friction slope implies zero flow resistance, which can not occur at steady state with nonzero depth of flow, flow-resistance coefficient, and flow.

If a similar exercise is carried out using the long prismatic channel of example 2, then the results

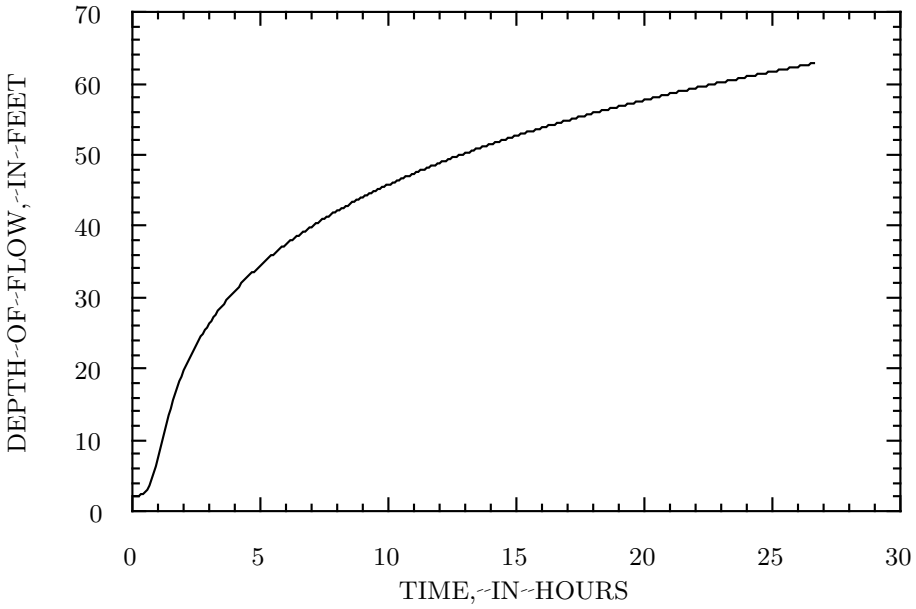

Figure 25: Failure of the kinematic approximation to reach steady flow when applied to the non-prismatic channel of example 1.

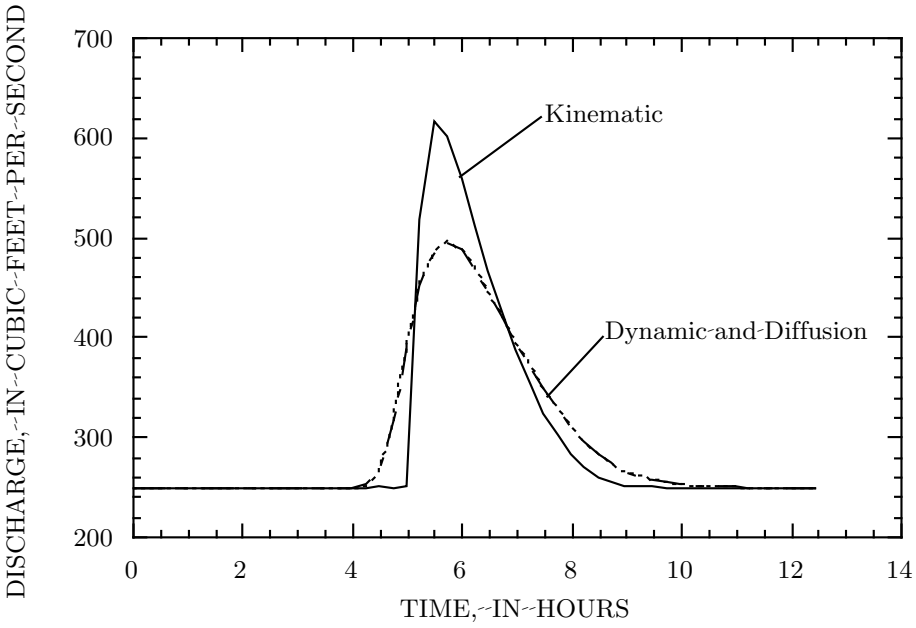

Figure 26: Different solution formulations and the prismatic channel section.

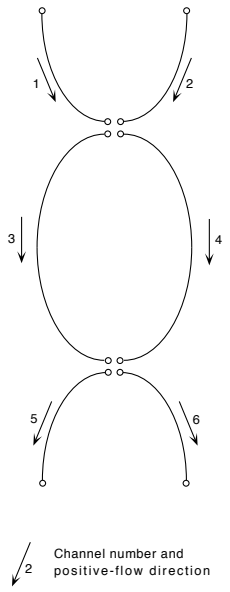

Figure 27: Schematic representation of a simple six-channel network.

shown in figure 26 are generated. For this channel, the dynamic and diffusion forms of the one-dimensional flow equations yield nearly the same results. However, the kinematic approximation yields a hydrograph with a significantly greater and sharper peak than the other forms of the equations. Clearly, the kinematic approximation is inappropriate for this particular problem. In fact, a θ of one was required to reach a solution for this problem.

Example 4 – Unsteady Flow in a Simple Network of Channels

The object of this example is to demonstrate flow simulation in a small (six-channel) network (figure 27). Input files for this example can be used as a guide for construction of more complex networks. This example is derived from work by Jobson and Schoellhamer (Jobson and Schoellhamer 1992).

FourPt was operated and time series output were taken from channel 3, distance 41,012 ft and channel 4, distance 29,529 ft. These results are shown in figure 28. The example schematic-data file (figure 29) can be used as a template for creation of more complex channel networks. Note that each channel has both an upstream and a downstream boundary-condition code.

Example 5 – Applying User-Programmed Constraints

This example demonstrates the application of two of the three user-programmed constraints that have been provided with the source code as examples of user-programmed constraints (see section *Constraint Properties* under *INPUT*). Two channels, rectangular in cross section, are simulated. The upstream channel is 1,000-feet wide and 10,000-feet long with a bed slope of 0.01 and flow-

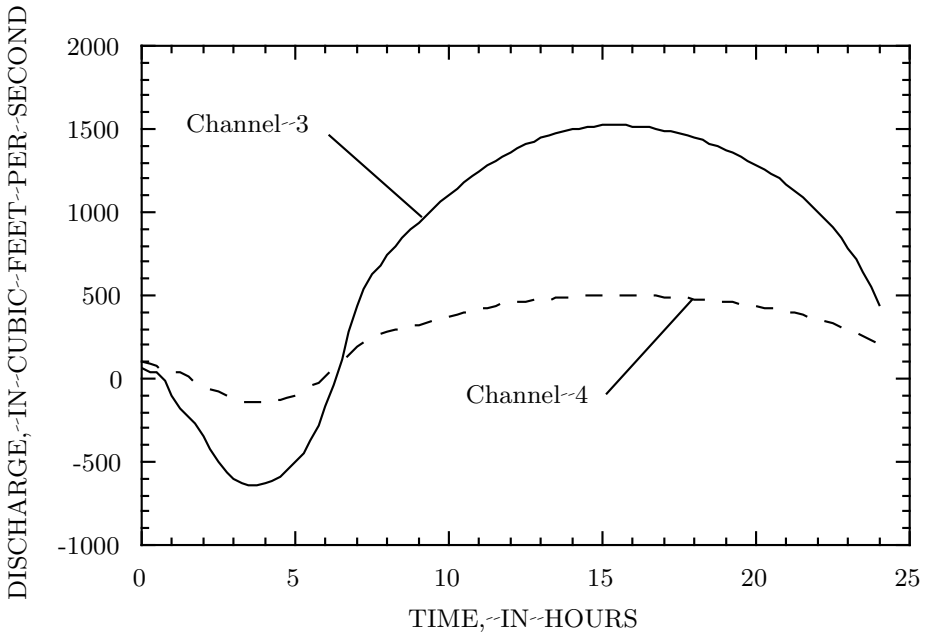

Figure 28: Flow in a simple six-channel network.

resistance coefficient of 0.045. The downstream channel is 100-feet wide and 50,000-feet long with a slope of 0.001 and a flow-resistance coefficient of 0.045. The downstream end of the upstream channel is connected (figure 31) to the upstream end of the downstream channel by a dam-break constraint fashioned after functions used in the National Weather Service DAMBRK model (Fread 1984). As a result, the upstream channel effectively becomes an in-channel reservoir.

The dam-break functions approximate variable flow through a trapezoidal breach expanding in time, over an uncontrolled spillway, through a gated spillway, over the crest of the dam, and constant flow through a turbine. The empirical dam-break functions are only briefly described in the following, as the primary purpose of this example is to demonstrate the use of constraints previously programmed by the user. For more detail regarding the empirical relations themselves see documentation for the National Weather Service DAMBRK model (Fread 1984). Values of the necessary empirical parameters are entered in the constraint-properties files as shown in figure 32.

Flow through an expanding trapezoidal breach is approximated by

$$Q_B = C_1 \left(Z - Z_B\right)^{\frac{3}{2}} + C_2 \left(Z - Z_B\right)^{\frac{5}{2}} \tag{74}$$

where Q_B is the discharge through the breach, Z is the current water-surface elevation at the dam, Z_B is the current elevation of the bottom of the breach, and C_1 and C_2 are coefficients defined by

$$C_1 = 3.1 B_i C_V K_S \tag{75}$$

```
     1    / branch number
        2, 2000.0, 1, 0, 0   / no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_1                   / cx ID
       38.617,      1059.3   / WS, Q
Downstream_1                 / cx ID
       38.617,       928.653 / cx no., WS, Q

 2                     /condition code     upstream
  0                     /connections

 11                    /condition code     downstream
  1                     /connections
   +3                    / connecting branch


     2    / branch number
        2, 2000.0, 1, 0, 0/ no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_2                    / cx ID
         38.617, 0.0          / WS, Q
Downstream_2                  / cx ID
         38.617, -116.523     / cx no., WS, Q

 2                     /condition code     upstream
  0                     /connections

 11                    /condition code     downstream
  1                     /connections
   +3                    / connecting branch


     3    / branch number
        2, 2000.0, 1, 0, 0/ no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_3                 / cx ID
         38.617, 617.925  / WS, Q
Downstream_3               / cx ID
         38.55175, 67.089 / cx no., WS, Q

 12                    /condition code     upstream
  3                     /connections
   -1                    / connecting branch
   -2                    / connecting branch
   +4                    / connecting branch

 11                    /condition code     downstream
  1                     /connections
   +5                    / connecting branch
```

Figure 29: Example-4 schematic-data file.

```
     4    / branch number
        2, 2000.0, 1, 0, 0/ no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_4                / cx ID
         38.617, 197.736  / WS, Q
Downstream_4              / cx ID
         38.55175, 105.93 / cx no., WS, Q

 11                    /condition code     upstream
  1                     /connections
   +3                    / connecting branch

 11                    /condition code     downstream
  1                     /connections
   +5                    / connecting branch


      5   / branch number
        2, 2000.0, 1, 0, 0/ no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_5                / cx ID
         38.55175, 423.72 / WS, Q
Downstream_5              / cx ID
         38.3877, 250.701 / cx no., WS, Q

 12                    /condition code     upstream
  3                     /connections
   -3                    / connecting branch
   -4                    / connecting branch
   +6                    / connecting branch

 1                     /condition code     downstream
  0                     /connections


     6    / branch number
        2, 2000.0, 1, 0, 0/ no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_6                / cx ID
         38.55175, -250.701 / WS, Q
Downstream_6              / cx ID
         38.650, -448.437   / cx no., WS, Q

 11                    /condition code     upstream
  1                     /connections
   +5                    / connecting branch

 1                     /condition code     downstream
  0                     /connections
```

Figure 30: Example-4 schematic-data file continued.

and

$$C_2 = 2.45 I_{SS} C_V K_S. \tag{76}$$

Here B_i is the current breach bottom width, I_{ss} is inverse side slope of the breach, C_V is the velocity-correction coefficient defined by

$$C_V = 1.0 + \frac{0.023 Q^2}{B_D^2 \left(Z - Z_{BM}\right)^2 \left(Z - Z_B\right)}, \tag{77}$$

where Q is the total outflow from the reservoir, Z_{BM} is the final breach bottom elevation, and B_D is the width of the reservoir at the dam. K_S in equations 75 and 76 is a submergence-correction coefficient defined by

$$K_S = 1.0, \qquad \text{for} \quad \frac{Z_T - Z_B}{Z - Z_B} < 0.67 \tag{78}$$

or

$$K_S = 1.0 - 27.8 \left(\frac{Z_T - Z_B}{Z - Z_B} - 0.67 \right)^3, \qquad \text{for} \quad \frac{Z_T - Z_B}{Z - Z_B} \geq 0.67, \tag{79}$$

where Z_T is the tailwater elevation.

Flow over an uncontrolled spillway is approximated by

$$Q_S = C_S L_S \left(Z - Z_S\right)^{\frac{3}{2}}, \tag{80}$$

where Q_S is flow over the spillway, C_S is the spillway coefficient, L_S is the spillway width, and Z_S is the spillway crest elevation.

Flow over the dam crest is approximated by

$$Q_D = C_D L_D \left(Z - Z_D\right)^{\frac{3}{2}}, \tag{81}$$

where Q_D is flow over the crest of the dam, C_D is an empirical coefficient, L_D is the width of the dam less the width of the uncontrolled spillway, and Z_D is the dam-crest elevation.

Flow through a gated spillway is approximated by

$$Q_G = C_G L_G \left(Z - Z_G\right)^{\frac{1}{2}}, \tag{82}$$

where Q_G is flow through the gate, C_G is an empirical coefficient, L_G is the gate flow area, and Z_G is the center-line elevation of the gated spillway. The gated spillway function is not active in this example; as shown in figure 32, the gated-spillway coefficient is zero.

Total discharge at the dam is the sum of the component discharges

$$Q = Q_b + Q_S + Q_D + Q_G. \tag{83}$$

A second user-programmed constraint, a logarithmic relation between water-surface elevation and discharge, is applied as the boundary condition at the downstream end of the downstream channel. This relation takes the form

$$Z = Z_0 + a Q^b, \tag{84}$$

```
     1                    / channel number
2,  625.0, 1, 7, 0        / no. of cx, dx, NKEEP, ICNDAP, NVAL
HeadOfReservoir           / cx ID
        61.7,       250.0  / WS, Q
AdjacentToDam             / cx ID
        78.0,       250.0 /  WS, Q

2                         /condition code upstream
  0                       /connections

10152                     /condition code downstream
  1                       /connections
    2                     /connecting channel



     2                    / channel number
2,  625.0, 1, 1, 0       / no. of cx, dx, NKEEP, ICNDAP, NVAL
Upstream_2                / cx ID
         51.7,       250.0 / WS, Q
Downstream_2              / cx ID
         1.7,        250.0 / WS, Q

12                        /condition code upstream
  1                       /connections
   -1                     /connecting channel

10351                     /condition code downstream
  0                       /connections0
```

Figure 31: Example-5 schematic-data file.

```
begin     / beginning of data for log-relation constraint

10351     / boundary-condition code, Z = Zo + a * Q ** b
0.05413   / a
0.62556   / b
0.0       / Zo

end       / end of data for log-relation constraint


begin     / beginning of data for a dam-break constraint

10152     / boundary-condition code
80.0      / elevation of top of dam
1000.0    / width of the reservoir at the dam

50.0      / final breach width
6.0       / total time of breach formation, in hours
0.5       / inverse of side slope of breach (horizontal/vertical)
50.0      / final elevation of breach bottom
81.0      / water surface elevation at which breach initiates

0.5       / uncontrolled spillway coefficient
100.0     / spillway width
78.5      / spillway crest elevation

0.0       / gated-spillway discharge coefficient
25.0      / gate cross section area
55.0      / center-line elevation of gated spillway

0.3       / coefficient of flow over crest of dam
900.0     / length of dam less uncontrolled spillway length
80.0      / dam crest elevation

250.0     / constant flow through turbine

end       / end of data for a dam-break constraint
```

Figure 32: Example-5 constraint-properties file.

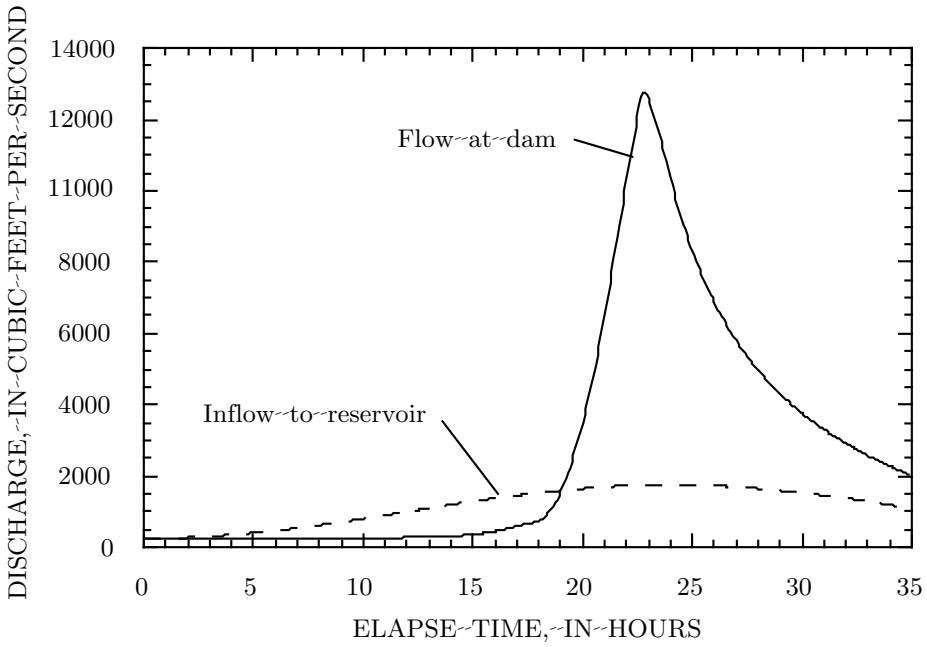

Figure 33: Inflow to reservoir and flow at dam for example 5.

where Z is water-surface elevation, Z_0 is a reference datum, Q is discharge, and a and b are empirical coefficients.

In this example, reservoir inflow is simulated with an equation-type boundary (see *Program control* under *Input*). Inflow begins at a base 250 ft^3/s, rises to 1,750 ft^3/s, and returns to 250 ft^3/s, in 48 hours. Results are shown in figure 33. The reservoir is nearly full, and flow at the dam abruptly rises to a peak of nearly 13,000 ft^3/s after 22 to 23 hours have elapsed. This peak flow is primarily a result of breaching of the dam. Effects of flow over the uncontrolled spillway and over the dam crest cause a gradual increase in flow prior to the breach initiation at 18 hours. Effects of the different types of flow may be seen in the relation of water-surface elevation to flow (figure 34) drawn from the first 20 hours of the simulation. Significant increases in the flow to water-surface-elevation ratio, apparent in slope of the curve, occur as water reaches the spillway elevation and the dam crest and when the breach begins to form.

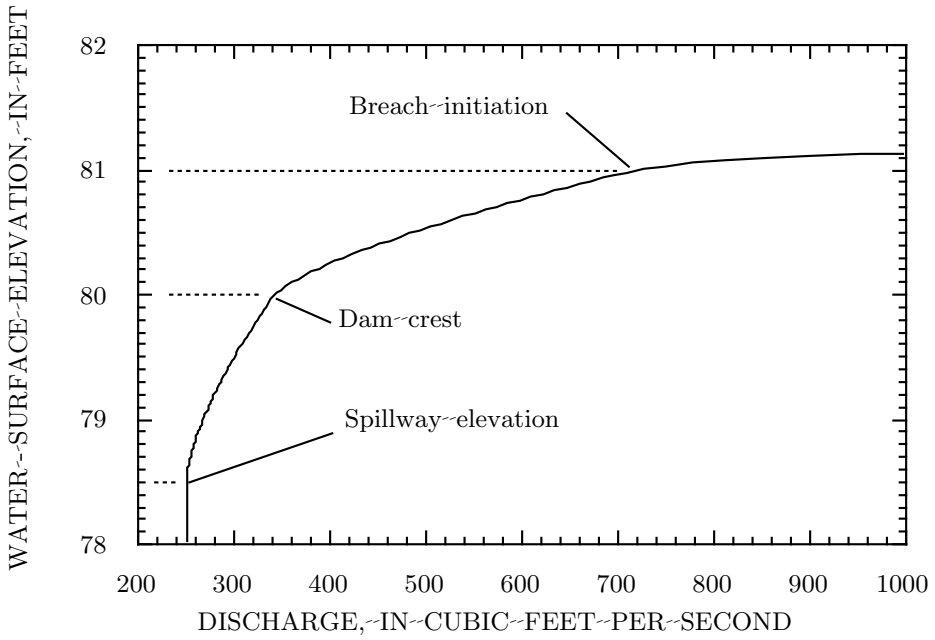

Figure 34: Relation of water-surface elevation to discharge for example 5 (first 20 hours of simulation).

References

American National Standards Institute (1978). *American National Standard Programming Language FORTRAN*. New York, N. Y.: American National Standards Institute.

Carnahan, B., H. A. Luther, and J. O. Wilkes (1969). *Applied numerical methods*. New York, New York: John Wiley and Sons, Inc.

Cunge, J. A., F. M. Holly. Jr., and A. Verwey (1980). *Practical aspects of computational river hydraulics*. Marshfield, MA: Pitman Publishing Inc.

DeLong, L. L. (1986). Extension of the unsteady one-dimensional open-channel flow equations for flow simulation in meandering channels with flood plains. In S. Subitzky (Ed.), *Selected papers in the hydrol. sci.*, pp. 101–105. U.S. Geolog. Water-Supply Paper 2290.

DeLong, L. L. (1989). Mass conservation: 1-d open channel flow equations. *J. Hydr. Div., ASCE 115*(2), 263–269.

DeLong, L. L. and J. M. Fulford (1993, Jul.). General implicit representation of hydraulic structures in numerical flow models. In *Proc. 1993 Nat. Conf. on Hydr. Engrg.*, pp. 1143–1148. ASCE.

DeLong, L. L. and D. H. Schoellhamer (1989). *Computer program HYDRAUX – a model for simulating one-dimensional, unsteady, open-channel flow*. U.S. Geolog. Survey Water Resources Investigations Report 88-4226.

DeLong, L. L., D. B. Thompson, and J. M. Fulford (1992). Data encapsulation using fortran 77 modules. *Fortran Forum 11*(3), 11–19.

Flanders, H., R. R. Korfhage, and J. J. Price (1974). *A Second Course in Calculus*. New York, NY: Academic Press.

Fread, D. L. (1984). *DAMBRK: The NWS dam break forecasting model*. Silver Spring, Md.: Ofc. of Hydrol., National Weather Service.

Froehlich, D. C. (1990, Jul.). Hermite-Galerkin model for dam-break floods. In *Proc. 1990 Nat. Conf. on Hydr. Engrg.*, pp. 557–562. ASCE.

Fulford, J. M. (1993). User's guide to hydraulic information exchange program: A computer program for hydraulic properties computations. On file with U.S. Geolog. Survey, at Stennis Space Center, MS 39529.

Jobson, H. E. and D. H. Schoellhamer (1992). *User's manual for a branched Lagrangian transport model*. U.S. Geolog. Survey Water Resources Investigations Report 87-4163.

Kirby, W. H. (1983). *Computer routines for probability distributions, random numbers, and related functions*. U.S. Geolog. Survey Water Resources Investigations Report 83-4257.

Preissmann, A. (1961, Sep.). Propagation des intumescences dans les canaux et rivieres. In *1st Congress de l' Assoc Francaise de Calcul*, Grenoble, pp. 433–42.

Smith, P. E. (1981). Numerical experiments on three unsteady flow models. On file with U.S. Geolog. Survey, at Stennis Space Center, MS 39529.

Stroustrup, B. (1988, May). What is object-oriented programming? *Software Magazine, IEEE*, 10–20.

Thompson, D. B., L. L. DeLong, and J. M. Fulford (1992). *Data Encapsulation Using Fortran-77 Modules — a First Step Toward Object-Oriented Programming*. U.S. Geolog. Survey Water Resources Investigations Report 92-4123.