```
H H H H H H H H H H H H H H H H H H H H H H H H H H
H                                                   H
H        //   //                                    H
H       //--//                                      H
H     //--// Y D R O L I N    Version 4.0            H
H    //   //                                        H
H                                                   H
H    Development of Conceptual Rainfall-Runoff      H
H      Models on the Personal Computer              H
H                                                   H
H          TU DELFT/Ct      881028 C.T. Chang       H
H                                                   H
H H H H H H H H H H H H H H H H H H H H H H H H H H
```

TU Delft

Technische Universiteit Delft

# Development of Conceptual Rainfall-Runoff Models on the Personal Computer

Reviewing Committee : Prof. dr. ir. J.C. van Dam
ir. H.A. Vermeulen
ir. A. van Mazijk

Supervisor : ir. H.A. Vermeulen

Adviser : Prof. dr. ir. J.C. van Dam

student : C.T. Chang

Date : September 1988

Contents

## 0. Abstract

Nowadays, the development in the personal computer hardware and software is very fast. It is more and more applicable with its recursive calculation character in hydrological modelling. By applying system analysis and system synthesis, it becomes a fashion in hydrological modelling to use the conceptual models instead of black-box models.

This thesis concerns the development of two conceptual models, Nash and O'kelly, which were based on an earlier made programme (HYDROLIN) and were constructed using Turbo Pascal 4.0 and Graphics Toolbox 4.0.

Using the method of moments, first estimates are made for the various parameters of the models. An extra option is the further improvement of all values of the parameters by an automatic stepwise procedure where the performance is evaluated by the so-called "coefficient of determination".

Both models have been applied to real data from a catchment in southern-France. It appears that these models can cope with data from complex-storms and their resulting hydrographs and that the "coefficient of determination" is an excellent criterion for the goodness of fit of the model parameters.

# 1. Introduction

## 1.1 General

The design and operation of water resources systems require information regarding hydrologic variables e.g. the flow rate at some point of interest along a stream, the maximum flood that can occur with a given return period etc. The development of methods that will lead to better estimates and forecasts of these variables and the practical application of such method is a task for the hydrologists. The science of hydrology has accordingly progressed to meet these goals.

The study of the relationship between precipitation and runoff and the quantitative modelling of the rainfall-runoff relationship is one of the major fields of research in hydrology. It is hoped that accurate prediction of the hydrologic behavior of watersheds could then become a reality. The operation of the watershed is a subsystem embedded in the larger system known as the hydrologic cycle. The principal components of the hydrologic cycle can be easily explained in a qualitative sense but can not be quantified with the same ease. This lack of quantitative concepts is likely to preclude the possibility that a mathematically precise science could be evolved, though this may be one of the ultimate goals of hydrology.

The purpose of this study is to develop simple, effective and reliable rainfall-runoff models in HYDROLIN (A hydrological interactive program that is developed by M.J. Vos). The model must be simple so that contain only a few watershed parameters, let the user easy to understand and easy to use. The model must be effective so that it can easily achieve the optimum solution. The model must be reliable so that the user can trust on it without any trouble during using it.

From the historical development of the rainfall-runoff relationship there are three main approaches: black-box approach, conceptual approach and hydraulic approach. the conceptual approach is the best choice for the purpose of this study.

The personal computer that is used in this research included 640 kb RAM (random access memory), two disket drivers (each with 360 kb storage capacity) and 640 * 200 pixels graphics card. The program language is Turbol Pascal which has a powerful compiler version 4.0 . With the modern developed personal computer one can construct interactive user friendly programmes.

## 1.2 The Need of Modelling, Model Formulation and Evaluation

A model is defined as a mathematical or physical system obeying certain specified conditions,whose behavior is used to understand a physical,biological or social system to which it is analogous in some way (McGraw-Hill 1974).

The systems we will deal with are real-world watersheds. Such systems receive natural inputs of water, energy and relative outputs in such forms as quntity and quality of streamflow, evaporation and other losses of material and energy through advection and radiation. Such input-output processes can be expresses rigorously for simple situations. But watershed are not simple. Even small watersheds are exceedingly complx. For reasons of practicality we must conceive of simpler, but adequate processes. The formalization of these simplified concepts of processes produces the hydrologic models that we substitute for the real-world watershed.

Some of the principal purposes for which modelling has or can be employed are given as follows:

- Research : Model offers an opportunity to extend the range of hydrologic research.
- Forecasting : The forecasting of streamflow.
- Engineering application :
  . record extension
  . operational simulation
  . data fill-in
  . data revision

Four stages are usually recognized in the development of a model. These stages are conceptualization, formulation, programming and testing.

Conceptualization means the transformation of all kind of natural processes into physically or mathematically usable models in order to be able to analyze practical problems and devising solutions. In the science base, models are developed for previous purposes, intellectual abstractions of logical analyses of processes, and mathematical and numerical methodologies, as well as a hard core of observed hydrologic reality. While in the art base, model is the assembly into an 'adequate and efficient' solution for problem. Tremendous freedom in practical conceptualization was gained wheen electronic computers became everyday working tools. The high-speed capacity of the computer opened up entirely new fields of mathematical and numerical analyses. These new techniques, in turn, allowed widerscope in conceptualization.

The formulation stage of model development means the conversion of concepts to forms for calculation. Formulation seems to imply a mathematical equation or formula. Certainly, equations may be model forms. However, any algorithm or sequence of calculationconverting input into outputs, is an acceptable formulatio. Such a sequence may contain equations, it may contain graphical relations, it may include 'Table look-up' or

highly sophisticated finite difference solutions of differential equations. Formulation is somewhat like preparation of a detailed flowchart of a problem.

Programming covers the mechanical but highly skilled efforts to translate the computational forms into a computer model.

Testing, the last stage in model development, covers all the steps taken to ensure that no errors are present and that computation forms and program structure are satisfactory. In a wider context, testing could also mean examination of the model to see how well it performs against recorded data. The goodness of performance is rather simple to quantify for rigorously stated testing criteria. On this quantified scale of goodness, however, it is more difficult to decide how good id good enough. The testing stage must involve numeric manipulations of the model. Such manipulations mean finding best values of the model parameters as well as of the model to predict outputs.

## 1.3 Systems Approach in Hydrology

As the hydrologic process (phenomenon) is very complx, it is easier to analyse with the help of system technique. The watershed is the real system, from which we need some information about this real system for special purpose. In the rainfall-runoff process, the three elements including in the system problem are: rainfall act as the input, watershed act as the system and the runoff act as the output. (fig. 1.1)

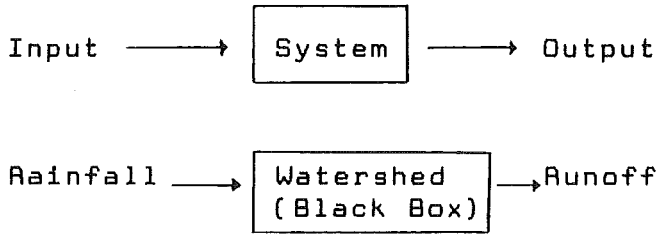



Fig. 1.1

After the rainfall fell on the watershed, some part will lose (evaporation, detention, infiltration and percolation), some part will flow through surface to breek, canal, river and finally to the outlet. Because the flow pattern of these two part are quite different, so that one can separate the watershed system into two subsystem: surface flow and ground water flow. Some water is in between: some time was groundwater, some time will flow out became surface water. (Fig. 1.2)

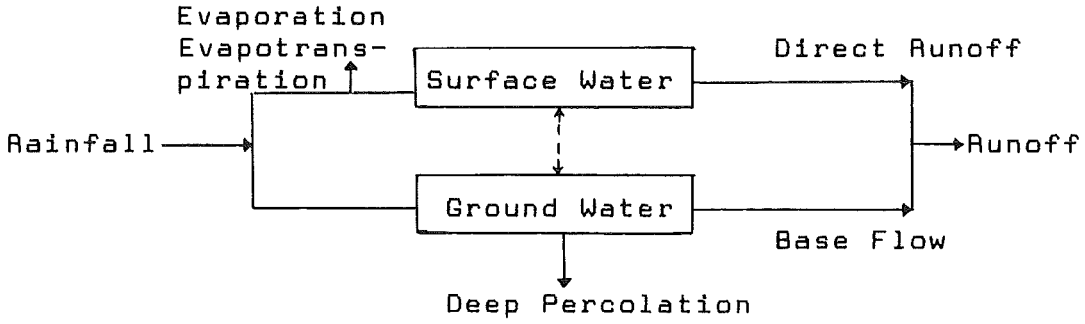



Fig. 1.2

System analysis and system synthesis give two ways in solving the system problem. Synthesis means to put together, while analysis is the process of breaking a given totality into essential elements for better understanding.

In systems analysis, the relationship between input and output is established through mathematical processes. The response of the system to any specified input may be studied. There is no explicit physical basis for this formulation and no attempt is made to describe the internal mechanisms of the system. Instead, the system is treated as a "black box". A transformation function converts input to produce the output. The function is usually unrelated to the physics of the prototype.

In systems synthesis, the operation of the system is explained through a combination of components which are actually presumed to exist in the system. In this formulation, the physical processes acting upon the input variables to produce the output variables are taken into account. The linkage between the different components takes into account the relationships in the hydrologic cycle so that appropriate interactions can occur.

Four types of system problem can be recognized in system analysis and synthesis. Output prediction can be done with the already known system and measured input. System identification and simulation can be done with the measured input and output. Input detection can be done with the already known system and measured output. (Table 1.1)

| System | System Problem | Input | System | Output |
|---|---|---|---|---|
| Analysis | Prediction | ✓ | ✓ | ? |
| | Identification | ✓ | ? | ✓ |
| | Detection | ? | ✓ | ✓ |
| Synthesis | Simulation | ✓ | ?? | ✓ |

Table 1.1

The unit hydrograph procedure, introduced by Sherman (1932), is a combination of system synthesis and system aalysis. The modification of the gross iput to yield effective rainfall and total runoff to yield direct runoff (making use of the infiltration properties of the catchment) constitute the synthesis part. The derivation of the unit hydrograph through deconvolution constitutes the analysis part. (Fig. 1.3)

Subsystem 1 performs the operation of subtracting the values of an infiltration function (by API, judgement or iteration) from the recorded input.

Subsystem 2 separates the so-called 'hydrograph component' from the recorded output.

Subsystem 3 perform linear convolution (by inversion).

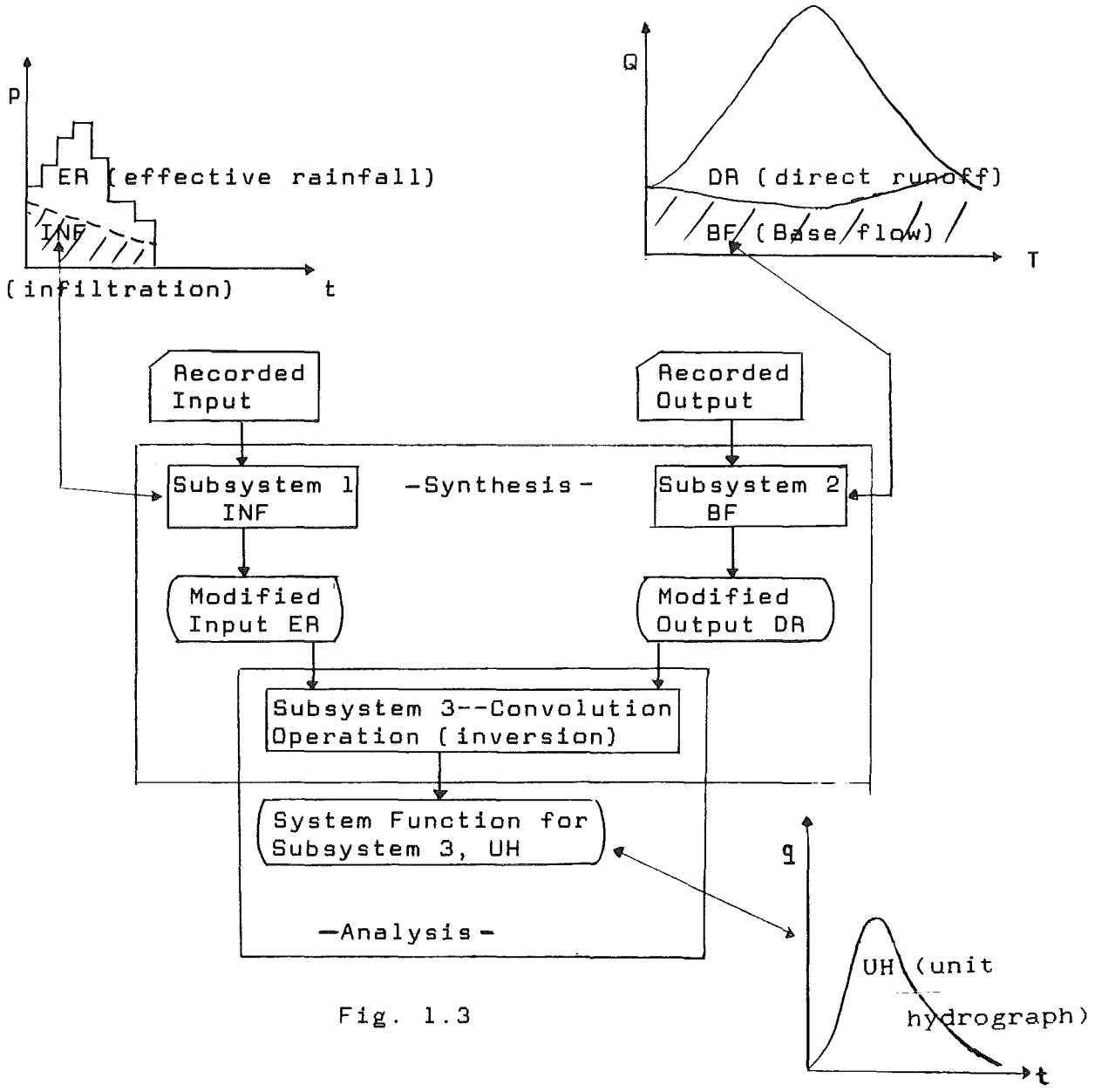

P
ER (effective rainfall)
INF
(infiltration)
t
Q
DR (direct runoff)
BF (Base flow)
T
Recorded Input
Recorded Output
Subsystem 1 INF
-Synthesis-
Subsystem 2 BF
Modified Input ER
Modified Output DR
Subsystem 3--Convolution Operation (inversion)
System Function for Subsystem 3, UH
-Analysis-
q
UH (unit hydrograph)
t

Fig. 1.3

## 1.4 Mathematical Models in Hydrology

There are some relative classification of models used in hydrology:

- Stochastic and deterministic models
- conceptual and empirical models
- linear and non-linear models
- lumped and distributed models

One model may have more than one of the above characters. It can be a linear deterministic lumped conceptual model. A model is a simplified representation of a complex system. The hydrologic model may be physically based, analogically based or mathematically based. In the mathematical based model, the system is represented by a set of equations, perhaps together with logical statements, expressing the relations between variables and parameters.

Denoting by $x(t)$, $y(t)$ the input and output variables of a system at time t, a mathematical model might be:

$$f(x(t),y(t);\ \partial x/\partial t,\ \partial y/\partial t,\ \partial^2 x/\partial t^2,\ \partial^2 y/\partial t^2,---;\ \theta_1,\ \theta_2)=0 \quad ---(1)$$

where $\theta_1$, $\theta_2$ are the system parameters.

In practice, the variables $x(t)$ and $y(t)$ will be measured at discrete intervals of time. It is then $x(t)$ and $y(t)$ will be replaced by $x_t$ and $y_t$, and $x/t$ and $^2x/t^2$ are replaced by $(x_{t+1}-x_{t-1})/2$ and $(x_{t+1}-2x_t+x_{t-1})/2$.

Then equation (1) may be rewritten as follows:

$$f(x_t,y_t;\ x_{t-1},y_{t-1};\ x_{t-2},y_{t-2};\ ---;\ \theta_1,\theta_2,--)=0 \quad ----------(2)$$

For the model's input and output, discrete data are usually used, even though it may be measured in continuous form. In this manner, it will be convenient to be used in computer modelling in digital way.

The major reason of confusion about mathematical models is the variety of types available and the different names used to define each type. From the configuration showing in Fig. 1.4, one may have a good understanding of their relationship.

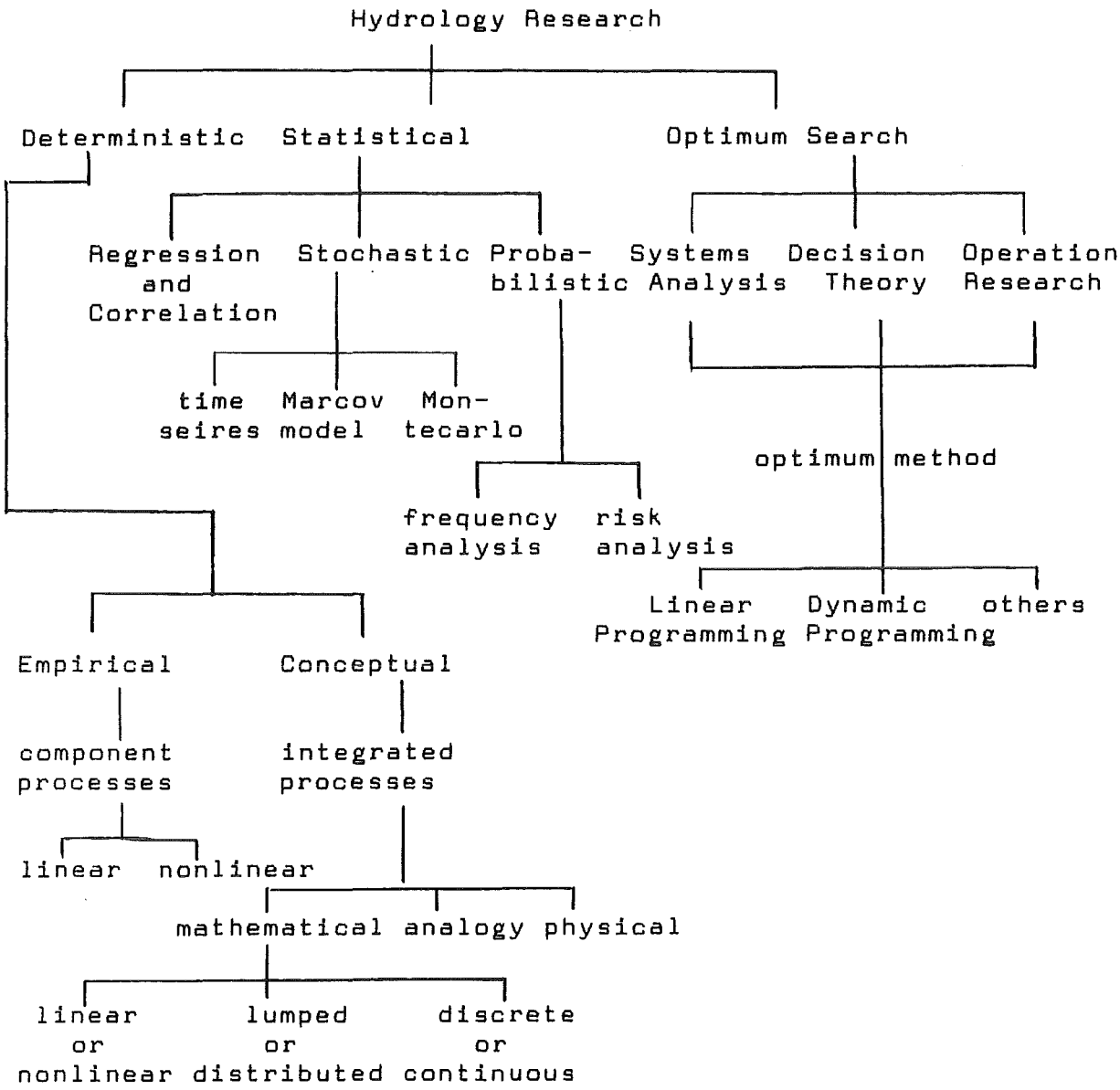

Hydrology Research
Deterministic
Statistical
Optimum Search
Regression and Correlation
Stochastic
Proba-bilistic
Systems Analysis
Decision Theory
Operation Research
time seires
Marcov model
Mon-tecarlo
optimum method
frequency analysis
risk analysis
Linear Programming
Dynamic Programming
others
Empirical
Conceptual
component processes
integrated processes
linear
nonlinear
mathematical
analogy
physical
linear or nonlinear
lumped or distributed
discrete or continuous

Fig. 1.4

## 1.5 Statement of the Problem

The hydrological cycle is close to the real system, but it is very complex not easy to analyse the relationships in between. For the purpose of hydrologic design and forecasting, it can be simplified into the following simplified catchment model:

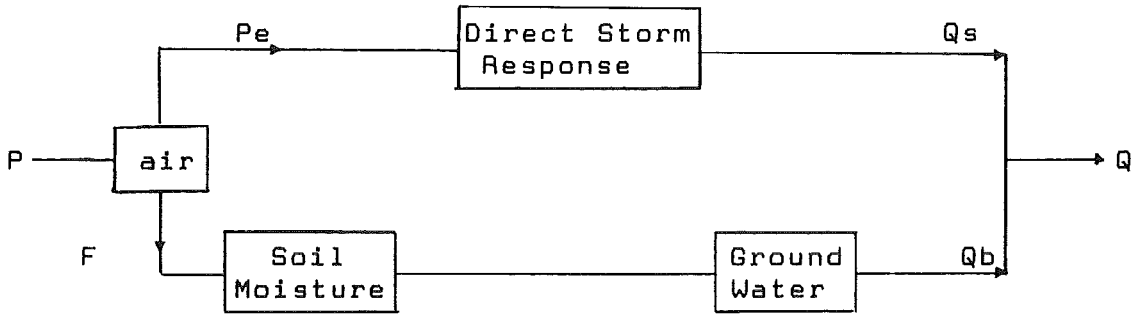



Fig. 1.5

Considering a watershed, with a uniform storm shower on it, the first part will infiltrate to the soil and fill the cavity of the surface, then the soil moisture rises to a certain content; by this moment, the surface begins to form a flow, and the infiltration also begins to percolate to the groundwater, the groundwater table begins to heave. The flow patterns of surface flow, interflow and groundwater are different, the first one is a fast response, while the last one is a slow response.

In short period, the human's activity is not so apparent, so that the time-invariant assumption can be made. In order to simplify the calculation in computer programming, linear behavior can be assumed so that the principle of superposition can be used.

The two parts of the relationship 'rainfall to effective rainfall' and 'effective rainfall to discharge' cannot be studied separately, but must be treated simultaneously.

The application Program HYDRLIN(1.0) offers the user the opportunity to have the treatment with the 'rainfall to effective rainfall' part, which is given in INPUT menu (seperation and losses), while the 'effective rainfall to discharge' part gives only one unit hydrograph method (matrix inversion). For the identification purpose, the approach method still needs to be strengthened for the alternative choice. In order to choose the best model to be developed in the HYDROLIN, it is necessary to look over the historical development of the rainfall-runoff models and make comparison among them (This is given in Chapter 2 ).

After comparison among these three approaches: black-box, conceptual and hydraulic, it appears that the conceptual approach is the properest for the use in HYDROLIN. Firstly, Nash Cascade and O'kelly routed triangle are chosen to develope in HYDROLIN.

The programming of HYDROLIN 1.0 was using TURBO PASCAL 3.0 (version) compiler, which has limitation in its CPU memory (for

every program, up to 64k), so it developed the utility of using including file.

In the beginning, the author tried to develop more models in HYDROLIN, but it caused new problems because of insufficient CPU memory for some programs. At the end of 1987, the new version compiler Turbo Pascal 4.0 was available for personal computer, it is a effective compiler (6 times faster than the old version) and has more CPU memory for every compiling program; but it can not use more than 5 including files and execusion procedure utility, that means there are many changes between the Turbo Pascal compiler version 3.0 and 4.0. It is a challenge to adapt HYDROLIN to new version 4.0 compiler.

## 1.6 Scope of the Research

This study deals with the development of Nash Cascade and O'kelly Routed Triangle modeling for the rainfall-runoff relationship of a watershed. The four stages are being developed, they are conceptualization, formulation, programming and testing, and are given separately in Chapter 3, 4, 5, 6 and 7. Chapter 6 is dealing with programming, Chapter 7 is dealing with testing.

## 1.7 The Requirement of Hardware and Software

The development of computer both in hardware and software are very fast. From Home Computer (8 bytes happy computer) to Personal Computer (16 bytes), and XT (16 bytes) to AT (32 bytes), Flopy disket may have 360 k bytes to 1 Mega bytes space. Harddisket may have 10 to 30 Mega bytes or more. Operation system from CPM to MS DOS, the compiler version from TURBOL PASCAL 1.0 to 4.0.

The author will suggest to use 4 diskets in applying the HYDROLIN 4.0 application program: Main disket, Source disket, Graphics disket and Data disket. The contained files and purposes are listed in Table 1.2.

| Disket | File Name | Purpose |
|---|---|---|
| Data Disket | fm.com | file management |
| | le.com | line editor |
| | ne.com | Newton editor |
| | turbo.com | turbo 3.0 editor |
| | ws.com | word star |
| | pni.dat | i= 0, 1, .....,7 |
| | qni.dat | i= 0, 1, .....,7 |
| Graphics Disket | turbo.exe | turbo compiler |
| | turbo.tpl | turbo library |
| | turbo.hlp | turbo h.l.p. |
| | turbo.tp | turbo.pascal unit |
| | types.tpu | t.p.u. of types pas |
| | hydrog.tpu | t.p.u. of hydrog.pas |
| | gdriver.tpu | t.p.u. of gdrver package |
| | gkernel.tpu | t.p.u. of gkernel package |
| | gshell.tpu | t.p.u. of gshell package |
| | error.msg | error message |
| Source Disket | turbo,exe | compiler |
| | hydrolin.pas | hydrolin main program |
| | types.pas | data types |
| | turbolib.pas | turbo library |
| | hydrol.pas | hydrolin including file |
| | hydrocal.pas | unit hydrograph model |
| | hydrog.pas | hydrographics |
| | nash.pas | Nash model |
| | Okelly.pas | O'kelly model |
| Main Disket | hydrolin.exe | HYDROLIN execute Program |
| | turbo.tpl | t.p.u.library |
| | types.tpu | t.p.u. of types.pas |
| | turbolib. tpu | t.p.u. of turbolib.pas |
| | hydrol.tpu | t.p.u. of hydrol.pas |
| | hydrocal.tpu | t.p.u. of hydrocal.pas |
| | nash.tpu | t.p.u. of Nash.model |
| | command.com | MS DOS |
| | Okelly.tpu | t.p.u. of O'kelly model |

Table 1.2

# 2. Literature Review

## 2.1. General

Many quantitative problems in hydrology are solved by determination and application of rainfall-runoff relationships. Such an approach has become necessary due to the scarcity of streamflow data and the relative abundance of precipitation data. However, if there were a sufficient number of stream gauging stations, their records will be useful insofar as they yield information about events that have already occurred at specific locations. Therefore, planning, design and operation of water resources development schemes will have high recourse to precipitation data.

The transformation that the catchment performs on the rainfall input to produce the runoff output must be found. The hydrologic literature contains descriptions of a vast number of methods to achieve this purpose. Many of these methods (or models) usually differ in the procedures adopted for evaluating parameters or in the use of different computational procedures for solving similar equations.

## 2.2. Black-box Approach Used in the Rainfall-Runoff Models

A. The earliest attempt at the modeling of the rainfall-runoff process involved the use of empirical formulae relating the peak flood discharge to catchment area and possibly one or two other variables such as slope, rainfall character.

Many authors attribute the origin of the rational method to the work of Mulvaney (1851). The relation between rainfall and runoff is expressed by the equation.

$$Q = CIA$$

Where Q = peak runoff rate
C = a dimensionless coefficient of runoff
I = mean intensity of rainfall during the time of concentration of the catchment
A = area of catchment

The rational method is still extensively used in situations, where quick, rough, preliminary or check estimates of peak runoff rates are required. It has limitations arising out of the assumptions of uniform rainfall intensity and that the whole catchment area produces maximum flood of the given frequency. It did not consider the effects of catchment storage and provide continuous streamflow data.

B. The unit hydrograph theory was proposed by Sherman (1932). It relates the effective rainfall over the catchment to the direct runoff at the catchment outlet. The unit hydrograph, as defined by Chou (1964), is the hydrograph of direct runoff that is produced by 1 inch of effective rainfall of uniform intensity falling in a specified length of time. The volume under the curve of the unit hydrograph is equal to 1 inch of rainfall occurring in the above specified length of time. Using the assumptions of constant time base of the unit hydrograph, linearity and time invariability etc. the unit hydrograph can be used to construct the discharge hydrograph for any duration of rainfall and any volume. Such information can then be used in the design of hydraulic structures.

In the analytical method, the error accumulated and the last value of U (unit hydrograph) show considerable oscillations. In the trial-and-error methods, a unit hydrograph is assumed. This unit hydrograph is then applied to all rainfalls by adjusting it untill a reasonable conformity with the recorded complex hydrograph is obtained.

C. The most rational way to obtain a rapid convergence is provided by the method of Collins(1939). The principle of this method is to account, in an approximate way, for the effect of the smaller rains and to derive the unit hydrograph from the largest rain.

D. The most general statement of a linear operation is du Hamel's integral (convolution integral), which has continuous and discrete forms.

If errors occured in the measurement of input or output data , the system identification or deconvolution will get unrealistic unit hydrograph ordinates.

Snyder (1955) suggested a least squares technique to determine the optimum solution for the unit hydrograph using all the available information. It assumes that the length of the unit hydrograph is equal to the difference between the known lengths of output and input data. The problem is to choose ordinates of unit hydrograph so as to minimize the squares of the residuals. This method involves optimization subject to the constraint that the length of the response function does not exceed the difference between the lengths of output and input.

E. O'Donnell (1960) presented an approach to the determination of the IUH by means of harmonic analysis. He postulated that the input and output hydrographs are both periodic with time T. The curve of rainfall excess, resultant hydrograph and IUH can each be represented by Fourier expansion. By applying the convolution integral and considering the n-th harmonics, he derived a relationship between the kernel coefficients and the input-output coefficients. The Fourier series (coefficients) were originally derived as infinite harmonic series and the Fourier coefficients defined by integrals. But in reality, hydrologic data is finite and therefore only harmonic coefficients can actually be used. The IUH coefficients will closely approach their true Fourier values as the acuracy and length of the rainfall-runoff record used in calculating the harmonic coefficients is increased. There is likelihood of negative ordinates of IUH being obtained due to oscilations; the result is thus unrealistic.

F. Dooge (1965) investigated the use of orthogonal Laguere functions as a basis for system identification. These functions are orthogonal over the semi-infinite interval 0 to $\infty$, with respect to the weighting function $e^{-t}$. This method was developed to improve the method of harmonic analysis, which was heavily damped due to its basic elements sine curves. By expanding the input, impulse response and output in terms of Laguerre functions; making use of the properties of orthogonality to derive the coefficients of input function and output function; then the identification problem can be solved (determining the coefficients of response function).

G. The extension of Laguerre function to discrete data uses Meixner polynomials and functions was performed by Amorocho and Brandstetter (1971). They were computed by means of simple recursion techniques. Expansion of any discrete function is expressed as a series of Meixner functions and the coefficients of the expansion evaluated. This series is usually truncated after a certain finite number of terms and this can result in a truncation error.

H. Eagleson et al. (1966) applied the Wiener-Hopf theory of optimum linear systems to determine the stable causal linear response of hydrologic systems from coincidental records of input and output data. The discrete form of the Wiener-Hopf equations were used since input and output records are usually in the form of discrete time series. The determination of the optimum

response in the least squares sense depends not on the original function but on the autocorrelation function of the input and the crosscorrelation function between input and output. If the data were completely error free, the output could be predicted closely. However negative ordinates, which are physically unrealizable, may be obtained. A hydrologically realizable unit hydrograph was obtained by solving the Wiener-Hopf equations subject to the constraint that all ordinates are non-negative.

I. Diskin (1967) used Laplace transforms to derive a general relationship for the moments about the origin of the three functions of the convolution integral. The resulting equations may be used for successive evaluation of the unknown moments of the IUH from the known moments of rainfall excess (input) hyetograph and of direct runoff (output) hydrograph. Simplified forms of the moment relations were derived for the moments of lowest orders, applicable to rainfall and runoff of equal totall volumes.

J. Kavvas and Schulz (1972) derived unit hydrographs from a number of recorded floods by an optimum matrix inversion method using a least squares procedures.

K. Papazafiriou (1975) employed Chebyshev polynomials and orthogonal polynomials to evaluate closed linear hydrologic systems. The kernels were expanded in sequences of polynomials. This method provides stable, physically realizable unit hydrograph and is not sensitive to noise. This polynomial approximation method yielded better results than the method of harmonic analysis.

L. Natale and Todini (1977) developed a linear black-box parameter estimation technique that gave stable solutions and could allow for multiple inputs.

M. Todini and Wallis (1977) developed a black-box rainfall-runoff correlation method known as constrained linear system (CLS) and used this for rainfall-runoff modelling. Though the CLS solves a linear equation, it approaches the problem of nonlinearity by separating the input stream into two or more separate input vectors and to calculate separate impulse response functions for each input. The separation is performed through the use of zero, one or two threshoulds that can be thought of as being similar to an antecedent moisture condition.

## 2.3 Conceptual Approach Used in the Rainfall-Runoff Models

A. The most widely used conceptual element used to model the direct storm runoff component are linear channels and linear reservoirs. These elements represent a separation and a concentration of the two distinct processes of translation and attenuation which are combined together in the case of unsteady flow over a surface or in an open channel. The conceptual model based on the diffusion analogy corresponds to a linearised version of the St. Venant equations for the case of a vanishingly small Froude number and may be said to represent a transition between a conceptual model of the direct storm response as a lumped system and a simplified mathematical model based on the equations for unsteady free surface flow.

B. The first synthetic unit hydrographs were derived by Hawken and Ross (1921). They modified the classical rational method (Mulvany in 1850) to include the effect of non-uniform rainfall distribution by the use of time-area-concentration curves. These curves were estimated on the basis of the time of travel from various parts of the catchment to the outlet as computed by hydraulic equations for steady flow. The time-area-concentration curve was in fact a synthetic unit hydrograph, which was unique to the catchment concerned.

C. Sherman (1932) published a second paper (besides the basic paper on the unit hydrograph approach) which was concerned with the relationship between the parameters of the unit hydrograph and the two important catchment characteristics of area and slope.

D. Owning to the time-area-concentration neglecting the attenuation effect due to surface storage, soil storage and channel storage, Clark (1945) suggested that the instantaneous unit hydrographs could be derived by routing the time-area-concentration curve through a single element of linear storage. In this case, each unit hydrograph would be unique but the variation between them would be reduced and the difference in catchment characteristics smoothed out to a greater or lesser extent depending on the degree of damping introduced by the storage routing.

E. O'kelly, Nash and Farrell (1955), working in the Irish Office of Public Works, found that there was no essential loss in accuracy in the synthesis of unit hydrographs if the routed time-area-concentration curve was replaced by a routed isoceles triangle. In case of a routed triangle, the unit hydrographs are no longer unique but belong to a family of two-parameter curves since the only parameters required to characterise such a unit hydrograph are the base of the isoceles triangle (T) and the storage delay time (K) of the linear storage element.

F. Lyshed (1955) in his study of Danish watercourses called attention to the various forms of storage through which rainfall excess has to pass on its way to the outlet. He described the rainfall-runoff relations with a sum of exponential functions which should represent the effect of a cascade of linear storage. He mentioned that the Gamma distribution, as suggested by Edson

(1951), could well describe the approximate form of unit hydrograph.

G. Sato and Mikkawa (1956) published a runoff routing method for the transformation of successive hourly rainfall rates into a discharge hydrograph for a small river in Japan. This routing equation is based on the second order Gamma distribution as a fundamental runoff function, from which a one hour unit hydrograph can be derived. In a final note the writers state that the n-orde Gamma distribution is a suitable element for the characterization of runoff in any river basin.

H. Nash (1957) suggested that a cascade of n storages is a sufficiently general model of the catchment mechanism and that the routing equation of n equal storages cascade could be taken as the general equation of the IUH. To allow non-integral n values, Nash substituted the factorial by a Gamma function.

I. Dooge (1959) introduced the linear channel. When a wave passes through a linear channel, there is pure translation only and no attenuation occurs. The shape of input and output waves keep the same and the lag in a linear channel equals the time of travel of a wave. He presented a general theory for the linear runoff model. It is based on the assumption that the composite effects of storage and translation in a linear drainage basin can be represented by the transformation performed by a cascade of linear channels connecting equal linear storage elements. The rainfall excess from the elementary areas between successive contour lines is fed into this cascade and subsequently routed through the appropriate length of linear channel and the corresponding number of equal linear storage elements.

J. Nash (1960) proved that the lag of the IUH also represents the distance in time between the centres of area of any inflow graph and the resulting outflow graph.

K. Laurenson (1962) discussed a number of runoff models and especially called attention to the fact that separation of translation from attenuation is unreal since any storage produces both.

L. Singh (1964) presented a model where the time area curve is routed through two linear storages respectively representing the effects of overland flow and channel flow.

## 2.4. Hydraulic Approach Used in the Rainfall-Runoff Models

When data is available on the topography of the channel and its hydraulic characteristics (e.g. roughness), hydraulic routing methods can be used to forecast discharges in downstream stations from upstream observation of discharge. These methods are based on the differential equations of motion (Saint Venant equations) and continuity in the channel and their solution under simplified conditions.

Analytical solution for the overland and channel flow is not available. Numerical solutions are available for simplified equations for particular cases:

(a) Linearization of the equation for non-steady flow as a dynamic wave (gravity forces prevail, friction and inertia forces negligible).

(b) Transformation to steady flow as a kinematic wave (discharge at each point unequivocally related to depth and cross-section of flow).

## 2.5. Conclusion

The tree approaches: hydraulics, conceptual models and black-box analysis, all have their place in applied hydrology. Each of them has its own particular area of effectiveness, depending on the degree of complexity of the problem, the objective of the study and the degree of accuracy required.

The use of conceptual models or black-box analysis in hydrology is a respectable way of approaching certain problems in hydrology. For a particular range of problems, they will provide useful results efficiently.

In general, the results of analytical approaches (black-box approaches) tend to be more affected by errors in the data than those of synthetical approaches (conceptual approaches). The accuracy of the results obtained from the synthesis technique depends on the fidelity of the simulation.

From the previous study showed that the response function is extremely sensitive to minor errors in the input-output data.

The best choice for the purpose of this study is appraently conceptural models. From which two models (Nash Cascade and O'kelly Routed Triangle) are chosen to develope in HYDROLIN.

# 3. Theoretical Background of the Nash Cascade and O'kelly Routed Triangle

## 3.1. General

The unit hydrograph (UH) is usually given in the form of a histogram or distributed unit hydrograph (DUH). If the unit hydrograph is represented by a conceptual model, the DUH may be described by a few parameter values only. The concentration of informatiion into a few parameters enhances a correlatiion of the parameter values with catchment characteristics. Another advantage of the use of conceptual models to represent the unit hydrograph is the suppression of the worst effects of data errors on the unit hydrograph derivation.

For the constructiion of conceptual models the following basic elements will be used: linear channel, linear reservoir and diffusion analogy. The linear channel with upstream inflow displaces the inflow without any change of shapes. This pure translation has only one parameter, the travel time of the linear channel. A linear channel with uniform lateral inflow results in an instantaneous unit hydrograph (IUH) with a rectangular shape. Other models based on translation may have the shape of an isosceles or scalene triangle with the base length as the only parameter.

The basic element based on storage assumes a linear relationship between the amount of water stored in the reservoir (S) and the outflow (Q).

$$S = KQ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Where K is the reservoir coefficient. Considering inflow into the reservoir P, it follows from continuity

$$P = Q + ds/dt = Q + K * dQ/dt \quad \ldots\ldots\ldots (2)$$

The boundary conditions for an instantaneous unit input are

$$P = S = 1 \text{ for } t = 0 \quad \ldots\ldots\ldots\ldots (3)$$

$$P = 0 \quad \text{for } t > 0 \quad \ldots\ldots\ldots\ldots (4)$$

Regarding the integrating equation (2) gives

$$\ln Q = -t/k + c$$

Since $Q = s/k = 1/k$ for $t = 0$, it follows for the integration constant that $c = \ln 1/k$, so that the expression for the IUH of a single linear reservoir (SLR) denoted by $u(0, t) = Q$ is given as

$$u(0, t) = (1/k) * e^{-t/k}$$

which is a one-parameter (k) model.

The basis elements may be combined into numerous conceptual models.

## 3.2. The Derivation of the IUH for Nash cascade

Conceptual models use the physical analog or mathematical simulation to analogize the watershed as linear reservoir, linear channel or time area diagram. In 1957, Nash derived the two parameters mathematical model using the linear reservoir concept, storage s is in proportion to the outflow q:

$$S = K * Q.$$

An instantaneous inflow, with volume V put in the first reservoir, its outflow will instantaneous rise from zero to V/K, that is:

$$I_1 = 0,\ S_1 = V,\ O_1 = V/K.$$

Using the continuity equation:

$$I_1 - O_1 = K * dO_1/dt.$$

Because $I_1 = 0$, so that:

$$-O_1 = K * dO_1/dt.$$

The general solution of $O_1$:

$$O_1 = \exp(-t/K + c).$$

From the initial condition one can solve the constant c, that is:

$$\text{at } t = 0,\ S_1 = V \text{ and } O_1 = V/K,$$

Substitute in the general solution one get:

$$e^c = V/K, \text{so that } O_1 = V/K * e^{-t/K+c}.$$

The outflow of the first reservoir into the second reservoir act as its inflow . Using the linear reservoir assumption and continuity equation, one can derive the outflow of the second reservoir:

$$O_2 = V/K * e^{-t/K} * t/K.$$

In this manner, the outflow of the n-th reservoir can be derived as follows:

$$On = V/K * (t/K)^{n-1}/\Gamma(n) * e^{-t/K} \quad \ldots (1)$$

Where $\Gamma(\ )$ is the gamma function.

The peak flow and the peak time can be derived by taking the derivation of the equation (1) with t/k to be zero:

$$1/(k * \Gamma(n)) * e^{-t/K} * [(t/k)^{n-2} * \Gamma(n-1) - (t/k)^{n-1}] = 0$$

so that:

$$t_m = (n-1) * k.$$

Put $t=t_m$ into the equation (1), one get:

$$U_m = 1/(k * \Gamma(n)) * e^{-(n-1)} * (n-1)^{n-1}.$$

The instantaneous unit hydrograph of 1 cm effective rainfall is expressed as follows:

$$U(0,t) = 2.78/(k * \Gamma(n)) * e^{-t/k} * (t/k)^{n-1} \quad \ldots (2)$$

Peak time $t_m = (n-1) * k$

Peak flow $U_m = 2.78/(k * \Gamma(n)) * e^{-(n-1)} * (n-1)^{n-1}$

From the above equations (1) and (2) one can see that the instantaneous unit hydrograph is a function of Gamma function with argument n and reservoir constant k (two parameters), and varing with time. These two parameters determine the shape of the instantaneous unit hydrograph. (Fig. 3.1)

In order to determine these two parameters for a watershed, one can use the effective rainfall histogram (ERH) and direct runoff hydrograph (DRH), applying the moment method to derive them.

From the moment method study there are the following relations:

$$M_{IUH1} = M_{DRH1} - M_{ERH1} = N * K \quad \ldots(3)$$

$$M_{IUH2} = M_{DRH2} - M_{ERH2} - 2NK * M_{ERH1} = N(N+1) * K^2 \quad \ldots(4)$$

where: $M_{DRH1}$ is the 1st moment of direct runoff hydrograph
$M_{ERH1}$ is the 1st moment of effective rainfall histogram
$M_{DRH2}$ is the 2nd moment of direct runoff hydrograph
$M_{ERH2}$ is the 2nd moment of effective rainfall histogram

From the above two equations (3) and (4) one can determine the two unknown parameters N and K.

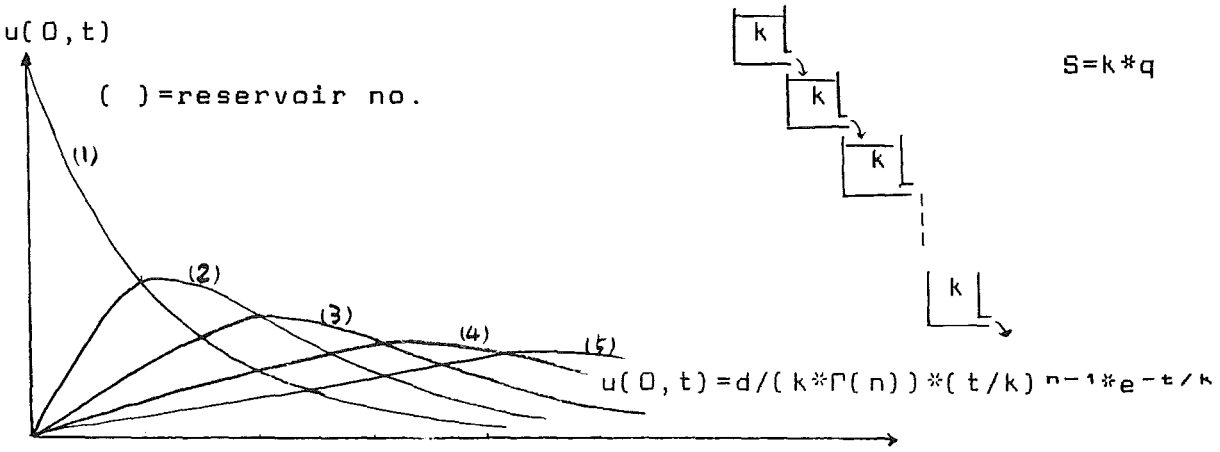



Fig. 3.1 Nash Cascade

The derivation of instantaneous unit hydrograph can be achieved by the followiny two method: direct solution and convolution intergral.

A. Direct solution

The continuity equation is:

$$i_n = q + ds/dt$$

Linear or nonlinear reservoir assumption:

$$s = k * q^p, \quad ds/dt = kpq^{p-1} * dq/dt$$

Combining the two equation:

$$kpq^{p-1} * dq/dt = i_n(t) - q \quad ...(5)$$

Assuming $i_n(t) = 0$ then:

$$kpq^{p-1} * dq/dt = -q$$

For linear reservoir p = 1, one gets:

$$k * dq/dt = -q$$

The general solution is:

$$q = e^{-t/k+c}$$

Using the initial condition:

$$\text{at } t = 0, \ s = d = kq_o$$

So that $q_o = d/k = e^c$, and one gets:

$$q = d/k * e^{-t/k} \text{ and } s = kq = d * e^{-t/k}.$$

That is: $u_1(0,t) = d/k * e^{-t/k}$ is the outflow of the 1st reservoir and the inflow of the second reservoir.

For the second reservoir :

$$k\, dq/dt = i_2(t) - q = d/k * e^{-t/k} - q$$

The general solution is:

$$q = d/k * t/k * e^{-t/k} + c$$

Using the initial condition:

$$\text{at } t = 0, \ u_2(0,t) = 0$$

One gets c = 0, so that:

$$u_2(0,t) = d/k * t/k * e^{-t/k}$$

is the outflow of the second reservoir and the inflow of the

third reservoir.

For the third reservoir :

$$k\ dq/dt = i_3(t) - q = d/k * t/k * e^{-t/k} - q$$

The general solution is :

$$q = d/k * (1/2) * (t/k)^2 * e^{-t/k} + c$$

Using the initial condition:

$$\text{at } t = 0,\ u_3(0,t) = 0$$

One gets c = 0, so that:

$$u_3(0,t) = d/k * 1/2 * (t/k)^2 * e^{-t/k}$$

is the outflow of the third reservoir and the inflow of the fourth reservoir.

Likewise for the n-th reservoir one gets:

$$u_n(0,t) = d/k * 1/(n-1)! * (t/k)^{n-1} * e^{-t/k}$$

B. Convolution integral

Outflow of the first reservoir:

$$u_1(0,t) = d/k * e^{-t/k}$$

Inflow of the second reservoir:

$$i_2(\tau) = u_1(0,\tau) = d/k * e^{-\tau/k}$$

Outflow of the second reservoir:

$$u_2(0,t) = \int_0^t d/k * e^{-(t-\tau)/k} * i_2(\tau) * d\tau/d$$

$$= d/k * e^{-t/k} \int_0^t e^{-\tau/k} * e^{\tau/k}\ d\tau/k$$

$$= d/k * t/k * e^{-t/k}$$

Inflow of the third reservoir:

$$i_3(\tau) = u_2(0,\tau) = d/k * \tau/k * e^{-\tau/k}$$

Outflow of the third reservoir:

$$u_3(0,t) = \int_0^t d/k * e^{-(t-\tau)/k} * i_3(\tau)\ d\tau/d$$

$$= d/k * e^{-t/k} \int_0^t e^{\tau/k} * e^{-\tau/k} * d\tau/k^2$$

$$= d/k * 1/2 * (t/k)^2 * e^{-t/k}$$

Likewise for the n-th reservoir one gets:

$$u_n(0,t) = d/k * 1/(n-1)! * (t/k)^{n-1} * e^{-t/k}$$

## 3.3. The Derivation of the IUH for the O'kelly Routed Triangle

O'kelly assumed that the instantaneous unit hydrograph could be obtained by routing an isosceles triangular inflow, of the correct volume and of base length T hours, through storage described by S = kQ (Fig. 3.2).

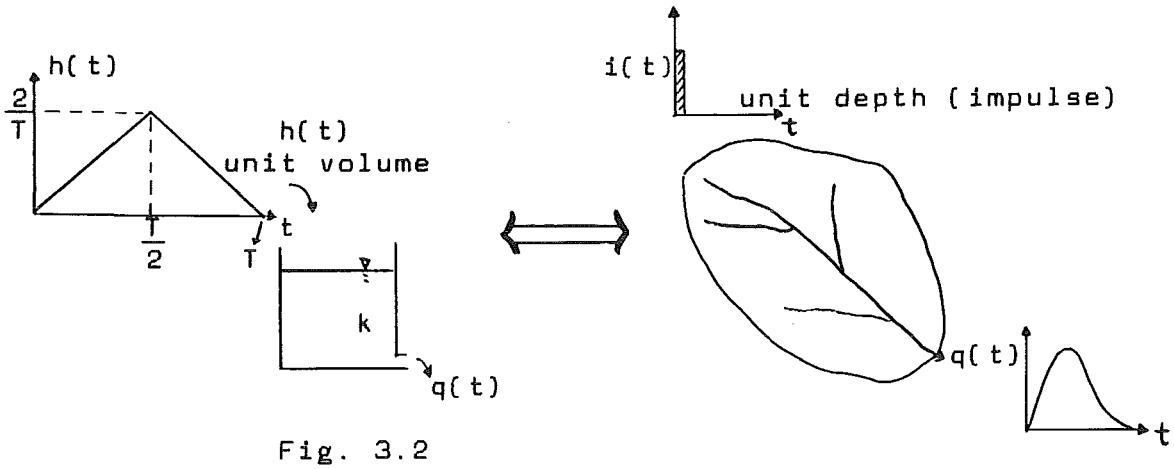



Fig. 3.2

The concept of O'kelly's is: the watershed react on the impulse (unit effective rainfall) as the mechanism of an isosceles triangular inflow (unit volume) routed through a linear reservoir (storage coefficient k). Its instantaneous unit hydrograph can be derived from the convolution integral.

The isosceles triangular inflow are separated into three parts, they are:

$$h(t)=4t/T^2 \qquad \text{for } 0 \le t < T/2$$

$$h(t)=-4t/T^2+4/T \qquad \text{for } T/2 \le t < T$$

$$h(t)=0 \qquad \text{for } t > T$$

The convolution integral for this mechanism is :

$$q(t)=\int_0^t u(0,t-\tau)\cdot h(\tau)\,d\tau$$

Where: the linear reservoir gives $u(0,t)=1/k * e^{-t/k}$
the isosceles triangular inflow include unit volume

$$\int_0^\infty h(t)\,dt=T/2 * h(T/2)=1$$

Substituting the given equation for the isosceles triangular inflow in the convolution integral. One can derive the IUH for the watershed:

$0 \le t < T$ :

$$q(t)=\int_0^t u(0,t-\tau)\,h(\tau)\,d\tau=\int_0^t (1/k)*e^{(-t+\tau)/k} * (4/T^2)*\,d\tau$$

$$=(4/kT^2)*e^{-t/k}\int_0^t e^{\tau/k}*\ d\ =(4/kT^2)*e^{-t/k}*(\ tke^{t/k}-k^2*e^{t/k}+k^2)$$

$$=(4/T^2)*(\ t-k+ke^{-t/k})$$

$T/2 < t \leq T$ :

$$q(\ t)=\int_0^{T/2}\ u(\ 0,t-\tau)\ h(\tau)\ d\tau+\int_{T/2}^t\ u(\ 0,t-\tau)\ h(\tau)\ d\tau$$

$$=\int_0^{T/2}\ (\ 1/k)*e^{(\ -t+\tau)\ /k}*(\ 4/T^2)*\ d\tau+\int_{T/2}^t\ (\ 1/k)*e^{(\ -t+\tau)\ /k}\ *$$

$$[(\ 4-4\ )\ /T^2]\ d\tau$$

$$=(\ 4/kT^2)*e^{-t/k}*[\ (\ T/2)*k*e^{T/2k}-k^2*e^{2T/k}+k^2-k*t*e^{t/k}-(\ kT/2)*$$

$$e^{T/2k}-k^2*e^{t/k}+k^2*e^{T/2k}]+(\ 4/Tk)*e^{-t/k}*(\ k*e^{t/k}-k*e^{T/2k})$$

$$=(\ 4/T^2)*[\ (\ T/2-k)*e^{-(\ t-T/2)\ /k}+k*e^{-t/k}]-t+k+(\ T/2-k)*e^{-(\ t-T/2)\ /k}$$

$$]+(\ 4/T)*(\ 1-e^{-(\ t-T/2)\ /k})$$

$t > T$ :

$$q(\ t)=\int_0^{T/2}\ u(\ 0,t-\tau)\ h(\tau)\ d\tau+\int_{T/2}^T\ u(\ 0,t-\tau)\ h(\tau)\ d\tau+\int_T^t\ u(\ 0,t-\tau)\ h(\tau)\ d\tau$$

$$=(\ 4/T^2)*[\ (\ T/2-k)*e^{-(\ t-T/2)\ /k}+k*e^{-t/k}]+\int_{T/2}^T\ (\ 1/k)*e^{-t+\tau/k}*$$

$$[(\ 4T-4\ )\ /T^2]\ d\tau$$

$$=(\ 4/T^2)*[\ (\ T/2-k)*e^{-(\ t-T/2)\ /k}+k*e^{-t/k}]+(\ -4/T^2)*[\ (\ T-k)*$$

$$e^{-(\ t-T)\ /k}-(\ T/2)*e^{-(\ t-T/2)\ /k}]+(\ 4/T)*[\ e^{-(\ t-T)\ /k}-e^{-(\ t-T/2)\ /k}]$$

## 3.4. From the Instantaneous Unit Hydrograph to Calculate Unit Hydrograph

From the difference of two successive summation curves one can derive the unit hydrograph.

The summation curve(S-curve) is defined as the integral of instantaneous unit hydrograph.

For the simplification of the calculation one can use the approximation method to calculate the unit hydrograph from instantaneous unit hydrograph.

$$U(\Delta t,t)=\{S(i_n,t)-S(i_n,t-\Delta t)\}*U/(i_n*\Delta t*A)$$

$$=(i_n*A)/U * \{\int_0^t U(0,t)\,dt-\int_0^{t-\Delta t} U(0,t)\,dt\}*U/(i_n*\Delta t*A)$$

$$=\int_{t-\Delta t}^{t} U(0,t)\,dt/\Delta t$$

$$=(U(0,t)+U(0,t-\Delta t))/2$$

The approximation can be realized from the following figure (fig. 3.3).

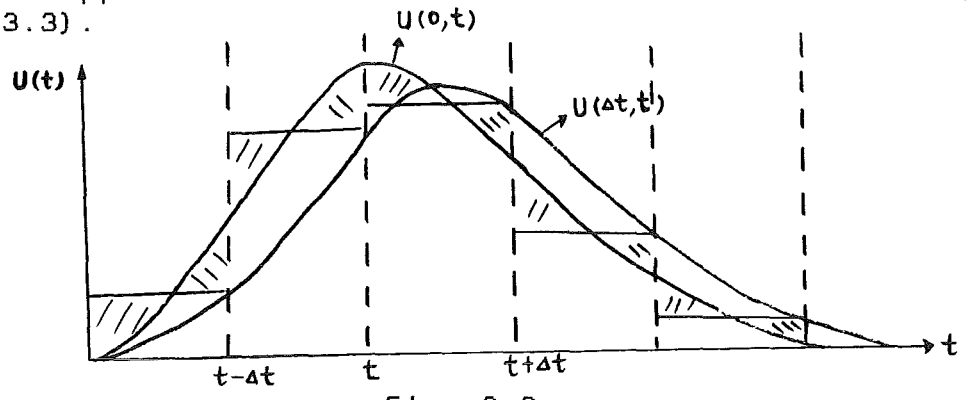



Fig. 3.3

## 3.5. From the Unit Hydrograph to Calculate Hydrograph

Assuming that the watershed system is linear and time invariant, one can apply the principle of superposition and matrix theory to derive the hydrograph from unit hydrograph.

The following is an example with simple storm, so that we can see how it works with matrix theory to calculate the hydrograph.

P1,P2,P3,P4 are effective rainfall depths in the successive unit storm periods. ($p=i*\Delta t$)

$U_1, U_2, U_3, U_4, U_5, U_6$ are the unit hydrograph series.

$Q_1, Q_2, Q_3, Q_4, Q_5, Q_6, Q_7, Q_8, Q_9$ are the hydrograph series.

| | $U_1$ | $U_2$ | $U_3$ | $U_4$ | $U_5$ | $U_6$ | | | |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | $U_1P_1$ | $U_2P_1$ | $U_3P_1$ | $U_4P_1$ | $U_5P_1$ | $U_6P_1$ | | | |
| $P_2$ | | $U_1P_2$ | $U_2P_2$ | $U_3P_2$ | $U_4P_2$ | $U_5P_2$ | U6P2 | | |
| $P_3$ | | | $U_1P_3$ | $U_2P_3$ | $U_3P_3$ | $U_4P_3$ | $U_5P_3$ | $U_6P_3$ | |
| $P_4$ | | | | $U_1P_4$ | $U_2P_4$ | $U_3P_4$ | $U_4P_4$ | $U_5P_4$ | $U_6P_4$ |
| | $Q_1$ | $Q_2$ | $Q_3$ | $Q_4$ | $Q_5$ | $Q_6$ | $Q_7$ | $Q_8$ | $Q_9$ |

That is:

$$
\begin{aligned}
Q_1 &= U_1P_1 \\
Q_2 &= U_2P_1 + U_1P_2 \\
Q_3 &= U_3P_1 + U_2P_2 + U_1P_3 \\
Q_4 &= U_4P_1 + U_3P_2 + U_2P_3 + U_1P_4 \\
Q_5 &= U_5P_1 + U_4P_2 + U_3P_3 + U_2P_4 \\
Q_6 &= U_6P_1 + U_5P_2 + U_4P_3 + U_3P_4 \\
Q_7 &= U_6P_2 + U_5P_3 + U_4P_4 \\
Q_8 &= U_6P_3 + U_5P_4 \\
Q_9 &= U_6P_4
\end{aligned}
$$

The general expression is :

$$Q_n = \sum_{i=1}^{n} U_i * P_{n-(i-1)} = \sum_{i=1}^{n} P_i * U_{n-(i-1)}$$

The above equation can also be expressed in the matrix form.

$$\bar{P} = \begin{pmatrix} P_1 & 0 & 0 & 0 & 0 & 0 \\ P_2 & P_1 & 0 & 0 & 0 & 0 \\ P_3 & P_2 & P_1 & 0 & 0 & 0 \\ P_4 & P_3 & P_2 & P_1 & 0 & 0 \\ 0 & P_4 & P_3 & P_2 & P_1 & 0 \\ 0 & 0 & P_4 & P_3 & P_2 & P_1 \\ 0 & 0 & 0 & P_4 & P_3 & P_2 \\ 0 & 0 & 0 & 0 & P_4 & P_3 \\ 0 & 0 & 0 & 0 & 0 & P_4 \end{pmatrix} \quad \bar{U} = \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \end{pmatrix} \quad \bar{Q} = \begin{pmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \\ Q_7 \\ Q_8 \\ Q_9 \end{pmatrix}$$

$$\bar{Q} = \bar{P} \cdot \bar{U}$$

# 4. Moment Method in the Derivation of Model Parameters

## 4.1. General

The moments of the individual unit hydrograph, which can be determined from the moments of effective rainfall and storm runoff, can be used systematically as the basis for a general synthetic scheme incorporating all three phases. Such a general synthetic scheme was first suggested by Nash (1959, 1960), who proposed that the derived moments of the unit hydrograph for the gauged catchments should be correlated with one another.

Nash (1959) suggested the use of the statistical moments of the instantaneous unit hydrograph as the determining parameters both for the identification of the unit hydrograph and the correlation with catchment characteristics.

The first moment of the origin of the instantaneous unit hydrograph is identical to the lag and recommended as an appropriate delay parameter. The second and third moments have the advantage over such parameters as the peak discharge that they are based on all of the ordinates of the unit hydrograph and not on single points and are therefore more stable in the presence of errors of measurement or derivation.

Since these parameters are chosen by moment matching or some other process, which takes the whole of the response curve into account, they have the stability characteristics spoken of above in regard to the statistical moments.

## 4.2. Determination of the K and N value Using the Moment Method

### A. From an Instantaneous Unit Hydrograph

Knowing the instantaneous unit hydrograph, take the zero, the first and the second moment relating to its point of origin, and calculate the second moment relate to its gravity center, then one can calculate the k and n value to get the first approach:

The instantaneous unit hydrograph :

$$u_n(o,t) = d/k * 1/(n-1)! * (t/k)^{n-1} * e^{-t/k}$$

Its zero moment related to the original point :

$$M_o = \int_o^t u_n(0,t)\, dt$$

$$= d \int_o^\infty 1/k * 1/(n-1)! * (t/k)^{n-1} * e^{-t/k}\, dt$$

$$= d$$

Its first moment related to the original point:

$$M_1 = \int_o^\infty u_n(0,t) * t\, dt$$

$$= \int_o^\infty d/k * 1/(n-1)! * (t/k)^{n-1} * e^{-t/k} * t\, dt$$

$$= d/k^n * 1/(n-1)! \int_o^\infty t^n * e^{-t/k}\, dt$$

$$= d/k^n * 1/(n-1)! * n * k^{n+1}$$

$$= d*n*k$$

Its second moment related to the original point :

$$M_2 = \int_o^\infty u_n(0,t) * t^2\, dt$$

$$= \int_o^\infty d/k * 1/(n-1)! * (t/k)^{n-1} * e^{-t/k} * t^2\, dt$$

$$= d/k^n * 1/(n-1)! \int_o^\infty t^{n+1} * e^{-t/k}\, dt$$

$$= d/k^n * 1/(n-1)! * (n+1)! * k^{n+2}$$

$$= d * n * (n+1) * k^2$$

Its second moment related to the gravity center:

$$M_2' = M_2 - M_1^2$$

$$= d * n * (n+1) * k^2 - (d * n * k)^2/d$$

$$= d*n*k^2$$

So that :

$$K = M_2'/M_1 \quad \text{and} \quad N = M_1^2/(d\, M_2').$$

B. From ERH and DRH

For practical application, one gets the direct runoff hydrograph and effective rainfall histogram from measurement and baseflow seperation,from the following proces one can also determine the K and N value from ERH and DRH.

Zero moment, 1st moment and 2nd moment with respect to the point of origin of the effective rainfall histogram are related as follows:

$$P_0 = \sum_{t=1}^{n} i(t) * \Delta t,$$

$$P_1 = \sum_{t=1}^{n} i(t) * (t-1/2) * \Delta t$$

$$P_2 = \sum_{t=1}^{n} i(t) * (t-1/2)^2 * \Delta t^2 = P_2' + P_1^2/P_0$$

First moment and 2nd moment with respect to the gravity center of the effective rainfall histogram are related as follows:

$$P_1' = 0, \quad P_2' = P_2 - P_1^2/P_0$$

Zero moment, 1st moment and 2nd moment with respect to the point of origin of the direct runoff hydrograph are related as follows:

$$Q_0 = \sum_{t=1}^{n} q(t) * \Delta t$$

$$Q_1 = \sum_{t=1}^{n} q(t) * (t-1/2) * \Delta t$$

$$Q_2 = \sum_{t=1}^{n} q(t) * (t-1/2)^2 * \Delta t^2$$

$$= Q_2' + Q_1^2/Q_0$$

First moment and 2nd moment with respect to the gravity center of the direct runoff hydrograph are related as follows:

$$Q_1' = 0, \quad Q_2' = Q_2 - Q_1^2/Q_0$$

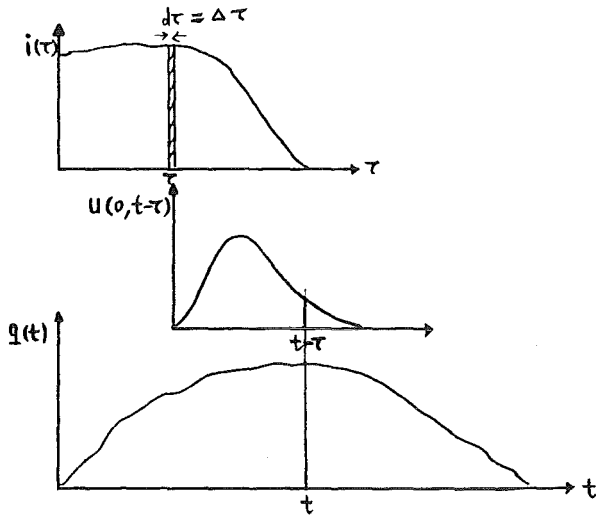



fig. 4.1 ERH, IUH and DRH relation

From above figure, one can derive the relation between ERH and DRH as follows:

$$Q_1=\Sigma(\tau+nk)*i(\tau)*_{\Delta\tau}=\Sigma\tau*i(\tau)*_{\Delta\tau}=P_1+nkp_o$$

$$\therefore \quad nk=(Q_1-P_1)/P_o \qquad ...(5)$$

Second moment of $u(0,t-\tau)$ for d, with respect to gravity center is:

$$M_2'=dnk^2.$$

Second moment of $u(0,t-\tau)$ for $i(\tau)d\tau$, with respect to gravity center is:

$$M_2'=i(\tau)d\tau*nk^2.$$

So that 2nd moment of the whole hydrograph with respect to the origin point is:

$$Q_2=\Sigma nk^2*i(\tau)d\tau+\Sigma(\tau+nk)^2*i(\tau)d\tau$$

$$=nk^2\Sigma i(\tau)d\tau+\Sigma\tau^2 i(\tau)d\tau+\Sigma 2\tau*nk*i(\tau)d\tau+n^2k^2*i(\tau)d\tau$$

$$=nk^2*P_o+P_2+2nk*P_1+n^2k^2*P_o$$

$$=k(Q_1-P_1)+P_2+2P_1*(Q_1-P_1)/P_o+(Q_1-P_1)^2/P_o$$

then: $Q_2P_o=kP_o(Q_1-P_1)+P_2P_o+2P_1Q_1-2P_1^2+Q_1^2-2P_1Q_1+P_1^2$

after deletion: $kP_o(Q_1-P_1)=(Q_2-P_2)P_o+P_1^2-Q_1^2$

one gets: $K=(Q_2-P_2)/(Q_1-P_1)-(P_1+Q_1)/P_o$ ...(6)

applying the equation (5) then:

$$Q_1-P_1=P_0nk=P_0n*(Q_2-P_2)/(Q_1-P_1)-n(P_1+Q_1)$$

also $(Q_1-P_1)^2=n\{P_0(Q_2-P_2)-(Q_1^2-P_1^2)\}$

so that: $n=(Q_1-P_1)^2/[PO(Q_2-P_2)-(Q_1^2-P_1^2)]$ ...(7)

From the moment relation:

$$Q_2-P_2=Q_2'-P_2'+(Q_1^2-P_1^2)/P_0=Q_2'-P_2'+(Q_1-P_1)(Q_1+P_1)/P_0$$

Substituting in equation (6) one gets

$$k=(Q_2'-P_2')/(Q_1-P_1)$$

$$=[(Q_2-Q_1^2/Q_0)-(P_2-P_1^2)/P_0]/(Q_1-P_1)...(8)$$

Substituting in equation (7) one gets

$$n=(Q_1-P_1)^2/[P_0(Q_2'-P_2')]$$

$$=(Q_1-P_1)2/\{P_0[(Q_2-Q_1^2/Q_0)-(P_2-P_1^2/P_0)]\}...(9)$$

In practical application, one uses the above two equations (8) and (9) to calculate the k and n values.

## 4.3. Moment Method for O'kelly Routed Triangle

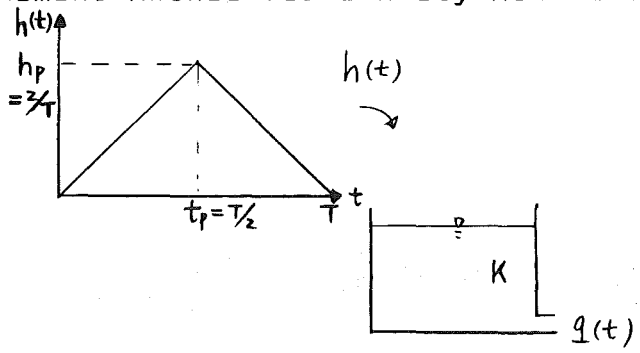



Fig. 4.2

The instantaneous unit hydrograph of O'kelly Routed Triangle (q(t)) is mentioned in paragraph 3.4. Its i-th moment related to the original point is defined as follows:

$$M_i=\int_0^\infty q(t)\cdot t^i dt$$

While the n-th moment related to the gravity center is defined as follows:

$$M_n'=\sum_{j=0}^{n}\binom{n}{j}(-1)^{n-j}*M_j*\left(\frac{M_1}{M_0}\right)^{n-j}$$

One can derive the following equation through the above definition:

$$M_0=1$$

$$M_1=t_p+K$$

$$M_2=(7/6)*t_p^2+2KM_1$$

$$M_3=(3/2)*t_p^3+3KM_2$$

$$M_1'=0$$

$$M_2'=(1/6)*t_p^2+K^2$$

$$M3'=(-1/270)*t_p^3+2k^3$$

where: $t_p$ is the peak time of the isosceles triangle
K is the reservoir storage constant (Fig. 4.2)

The i-th moment of ERH (effective rainfall histogram) and DRH (direct runoff hydrograph) are defined as follows:

$$Pi=\int_0^\infty \pi(t)*t^i dt$$

$$Qi=\int_0^\infty Q(t)*t^i dt$$

where: $\pi(t)$ is the effective rainfall
$Q(t)$ is the direct runoff

From the above definition, one can derive the following equations:

$$P_0=Q_0=V_e$$

$$Q_1=(t_p+k)*P_0+P_1 \quad \text{so that : } k=(Q_1-P_1)/P_0 - t_p$$

$$Q_2=(7/6)*t_p^2*P_0 - t_p*(Q_1-P_1) + 2Q_1*(Q_1-P_1)/P_0 +P_2$$

The above equation can derive the following second power equation of $t_p$ as follows:

$$(7/6)*t_p^2 - 2t_p*Z_1 + (2Z_q*Z_1 - Z_2) = 0$$

where: $V_e$ is the effective rainfall volume
$Z_1=(Q_1+P_1)/P_0$
$Z_2=(Q_2+P_2)/P_0$
$Z_q=Q_1/P_0$

The solution for $t_p$ is:

$$t_{p1,2}=T/2=(6/7)*Z_1 \pm \sqrt{(36/49)*Z_1^2 - (6/7)*(2Z_q*Z_1 - Z_2)}$$

These two parameters T and K can be solved with the help of moment method.

$$T=(12/7)*Z_1 \pm 2*\sqrt{(36/49)*Z_1^2 - (6/7)*(2Z_q*Z_1 - Z_2)}$$

$$K=(Q_1-P_1)/P_0 - t_p$$

# 5. Assessment Criteria for the Goodness of Fit Test

## 5.1. General

Finding a set of best fit parameter values for given physical systems, input and output data, is a frequently met problem in many fields of activity. Optimization or `hill climbing' techniques have been developed that determine values of system parameters which maximise or minimise some function dependent on those parameters. Such techniques are completely objective. They make many useless tests of situations, which would be dismissed out of hand by an experienced and skilled human investigator, but the tremendous speed of a computer can make such tests compensate for such inefficiency.

The assessment criteria are acting for the selection of model and model parameter optimization, which can be numerical or graphical.

The difference between measured and theoretical value are from:

- insufficient model structure
- system error of rainfall, evaporation and runoff measurement
- rainfall losses evaluation and direct runoff evaluation

In order to do the assesment of the goodness of fit for the model parameters one needs to set up the criteria for the model optimization simulation.

The criteria can be the following functions:

- objective function(least square method)
- non-dimensional objective function(coefficient of determination)
- modulus integration
- flow and volume double standard
- peak flow, volume and concentration time triple standard
- correlation coefficient and variance
- computer plotting technique and pattern recognition

The following three criteria are suggested to be used in the assement of the goodnees of fit test:

- the coefficient of determination
- modified coefficient of determination
- graphics recognition

## 5.2. The Coefficient of Determination

Dawdy and O'Donnell proposed to use "the minimum value of the sum of the squares of the difference between measured and simulated flow" act as the criterion.

$$F^2 = \sum_{i=1}^{n} (q_i - r_i)^2 \quad \text{minimised}$$

where $F^2$ = index of disagreement or objective function
$q_i$ = measured flow
$r_i$ = simulated flow
$n$ = number of record

Based on the above objective function, Nash and Sutcliffe proposed the coefficient of determination to act as the criterion. Ibbitt(1972) proposed it also for the testing of conceptual models.

$$R^2 = \frac{1/n \sum_{i=1}^{n} (q_i - q)^2 - 1/n \sum_{i=1}^{n} (q_i - r_i)^2}{1/n \sum_{i=1}^{n} (q_i - q)^2}$$

where $q = 1/n \sum_{i=1}^{n} q_i$

$R^2$ = analogous to the coefficient of determination

If every term of $r_i$ keeps the same as $q_i$, then $R^2$ become unit, is the best condition. Mostly, it lies between 1 and -1. In worst case, it can be smaller than -1.

## 5.3. Modified Coefficient of Determination

One can also modify the above non-dimensional objective function by multiplying every term by $q_i$ to give higher weight to higher flow. That is:

$$R^2 = \frac{1/n \sum_{i=1}^{n} q_i (q_i - q)^2 - 1/n \sum_{i=1}^{n} q_i (q_i - r_i)^2}{1/n \sum_{i=1}^{n} q_i (q_i - q)^2}$$

For different purpose projects, People have different part of accuracy demand in the modelling. For the flood control and forecasting, people require high accuracy in the peak flow part modelling. In this case, one can use the modified coefficient of determination, giving higher weight to higher flow to get better assessment criterion in the goodness of fit test. In case of the peak time of measured and calculated data are fit each other, the modified coefficient of determination will give the same result as the coefficient of determination. In case of lag occured between the peak time of measured and calculated data, this criterion will give a better assessment in the goodness of fit test, and a better improvement can be expected. It lies mostly between 1 and -1.

## 5.4. Graphics Recognition

Recently by using the computer plotting technique and pattern recognition one can also do the simulation test assessment.

Graphics recognition can be done by using the graphics programme HYDROG (its commando is hydrog). The graphics menu has the following options:

- P_Histo (rainfall histogram)
- Q_Poly (discharge polygon)
- PQ_Wind (rainfall-discharge window)
- Q_Compare (discharge comparison between measured and calculated data)

The Q_Compare option is used for the recognition purpose.

# 6. Construction of the Models

## 6.1. General

The programming language used in HYDROLIN is Turbo Pascal, which has the merit of structure, readibility, procedure block, easy to maintain, fast compiler and editor, automatic error tracing capacity etc.

From the main menu of HYDROLIN, one can use the Dir option to see the contents of directory, the Editor menu to do the editing work, the FileHandling menu to do the file handling work, the Help menu to see the helpful information, the Input menu to prepare the input data file for the modelling, the Model menu to run the models and the Quit option to drop out HYDROLIN. (Appendix B.1)

For the application of HYDROLIN, one can first go through the input menu to get the input data for the use of rainfall-runoff models. For the purpose of losses evaluation, one can use the losses menu. For the purpose of direct runoff separation, one can use the separate menu. (Appendix B.1 and B.2)

In the model menu, one can choose Nash, O'kelly or Unit Hydrograph. These models are built in a .pas (unit file) file. After compiling, a .tpu (Turbo Pascal Unit) file will be generated. The main program HYDROLIN uses these .tpu files acting as a subroutine.

In the HYDROLIN programme, the following unit files are used: (Appendix A)

- Hydrol.tpu (HYDROLIN library)
- Types.tpu (data types)
- Turbolib.tpu (Turbo library)
- Hydrocal.tpu (Unit Hydrograph model)
- Nash.tpu (Nash model)
- Okelly.tpu (O'kelly model)

## 6.2. Nash Cascade

In the Nash Cascade model, some procedures are written to do the recursive calculation. One main procedure (Nashmain) is constructed to call all these procedures. From the model menu, programme need only to call the main procedure. After the modelling is finished, the cursor will come back to the model menu.

The procedures that are used in the Nash Cascade model are explained as follows: (Appendix B.3)

- Help message (to see the helpful information about the modelling)
- Input menu (using the three options: CopyEdit (copy exist file and then edit it), ExistFile (to run with the existing file) and ModifyFile (modify the existing file and then run it) )
- NKvalue (to calculate the N and K values with the help of moment method)
- IUHtoUH (using the approximation method to calculate the unit hydrograph from the instantaneous unit hydrograph)
- UHtoH (using the matrix method to calculate the hydrograph from the unit hydrograph)
- Criteria (to calculate the coefficient of determination for the measured and calculated hydrograph)
- Simulation menu (using two simulation approaches: Auto- or Semi-simulation, to improve these two model parameters)
- Nashmain (to call all the above procedures and print output file)

## 6.3. O'kelly Routed Triangle

The O'kelly Routed Triangle model has the same structure as the Nash Cascade model. They have both only two parameters (N and K or K and T) by using the moment method to get the first approach and then doing the simulation improvement.

The only differences are that O'kelly Routed Triangle don't need to call the Gamma function and their equations for IUH and parameters are different.

The procedures that are used in the O'kelly Routed Triangle model are explained as follows: (Appendix B.4)

- Help message (to see the helpful information about the modelling)
- Input menu (using the three options: CopyEdit (copy exist file and then edit it), ExistFile (to run with the existing file) and ModifyFile (modify the existing file and then run it) )
- KTvalue (to calculate the N and K values with the help of moment method)
- IUHtoUH (using the approximation method to calculate the unit hydrograph from the instantaneous unit hydrograph)
- UHtoH (using the matrix method to calculate the hydrograph from the unit hydrograph)
- Criteria (to calculate the coefficient of determination for the measured and calculated hydrograph)
- Simulation menu (using two simulation approaches: Auto- or Semi-simulation, to improve these two model parameters)
- Okellymain (to call all the above procedures and print output file)

## 6.4. Auto-Simulation and Semi-Simulation

There exist several methods for different purpose calibrations. They are described as follows:

- semi-automatic simulation.
- automatic simulation.
- combination of above two method.

By semi-simulation one can step by step improve the parameter optimization by given parameter decrement (increment) to check the criteria of neighbor points. If the criteria is closer to one, one can replace it to get new parameter values. It takes longer time to do this process but it gives a chance to get a better approach by given bigger decrement to jump over the vally between local and real optimum parameter values. (Appendix B.6)

Automatic parameter optimization is using criteria acting as assesment media, given a convergent method (first decrement (increment) is given, untill it can not improve any more then decrease the decrement (increment) by a given value untill the decrement (increment) become zero) to do the simulation approach.

The combination of semi-simulation and auto-simulations can have the merit of the both methods.

There are several steps used in calibration:

- comparison of storm hydrograph shape and peak time
- comparison of storm hydrograph volume
- comparison of annual runoff volume
- comparison of seasonal distribution of runoff (daily flow)

The first and second steps are used in this study.

# 7. Calibration, Verification and Application of the Models

## 7.1. General Description of the Catchment and Data Used for Testing

The catchment parameter that is used in the conceptual model contained only the catchment area. The other two model parameters (K,T or n,K) are derived from the measured rainfall and runoff data by using the moment method.

The catchment area used for the model test are from 160 $km^2$ to 2266 $km^2$. The time intervals of the sampled data (rainfall and runoff) run from 1 hour to 12 hour.

The input data files used in the modelling are effective rainfall and direct runoff, which must first be derived by using the INPUT menu from HYDROLIN. They are stored in DATA floppy by giving the file name e.g. PNn.dat, QNn.dat (n=0...7).

## 7.2. Calibration of the Models

With the help of semi-simulation, one can improve again step by step by giving smaller decrement of n and k.

Five set of storm events are used for the test of Nash Cascade and O'kelly Routed Triangle models. (Table 7.1)

| Data Set (area) | Approach Method | NASH | | | O'KELLY | | |
|---|---|---|---|---|---|---|---|
| | | n | k (hour) | $R^2$ | k (hour) | T (hour) | $R^2$ |
| set 1 (560 $km^2$) | Moment | 3.74 | 2.76 | .992 | 4.83 | 30.08 | .216 |
| | Simulation | 3.94 | 2.68 | .996 | 0.83 | 23.08 | .916 |
| set 2 (432 $km^2$) | Moment | 2.59 | 1.28 | .994 | 1.02 | 8.67 | .170 |
| | Simulation | 2.31 | 1.46 | .996 | .196 | 6.17 | .814 |
| set 3 (160 $km^2$) | Moment | 0.99 | 15.52 | .890 | 3.67 | 30.85 | .954 |
| | Simulation | 2.36 | 8.78 | .980 | 3.82 | 33.19 | .963 |
| set 4 (160 $km^2$) | Moment | 7.26 | 3.48 | .931 | 8.31 | 67.05 | .341 |
| | Simulation | 7.08 | 3.51 | .945 | 9.71 | 31.45 | .959 |
| set 5 (2266 $km^2$) | Moment | 3.32 | 4.9 | .987 | 4.8 | 41.2 | .207 |
| | Simulation | 3.44 | 4.46 | .991 | 2.9 | 23.1 | .918 |

Table 7.1

## 7.3. Verification

Verification can be done by using the determined parameters to another storm event to see whether these determined parameters are feasible for the studied catchment.

Because of lack of measured data, this could not be done in this study.

From different storm event, one always derive different set of parameters for the same catchment. A master unit hydrograph is suggested to be used for a certain catchment.

## 7.4. Comparison Between the Models

From the test examples (Appendix B5.), one can easily get the conclusion that Nash Cascade may yield a better approach than theO'kelly Routed Triangle. That is because the isosceles triangle is not feasible for different watershed situation. One needs to add another parameter to determine the peak time of the scaline triangle.

Because in all the test cases the calculated series of O'kelly model are shifted to the right hand side, the SCS triangle ($T_b$=2.67 $T_p$) is suggested to be used in stead of an isosceles triangle, then it will remain a two parameters model.

# 8. Conclusion and Recommendation

## 8.1. Conclusion

This study proves that :

- The moment method is a rough method, but may be considerably improved by the simulation method.
- Nash Cascade and O'kelly Routed Triangle can accomodate complex storm
- Using the approximation method to calculate UH from IUH is acceptable for different time interval storms.
- Using the matrix method to calculate the hydrograph from UH is acceptable with the same assumption of unit hydrograph.
- Using a computer to do the automatic simulation approach is fast and always one can improve the n and k values by given smaller n and k.
- Automatic simulation and Semi-simulation are reciprocally supplementary
- The dimensionless objective function (coefficient of determination ) is an excellent criterion for the goodness of fit test of the conceptual model's parameters.
- With the help of new version TURBO PASCAL 4.0 one can restructure the interactive program HYDROLIN (which was originally programmed with old version TURBO PASCAL 3.0 and use many chain files and execute many compiled files (.com).
- Including more models in the interactive program HYDROLIN with two floppy driver ( each for 360 kb space) is possible.

## 8.2. Recommendation

For further study, some recommendation are given as follows:

- One can module the base flow model for the base flow and direct runoff separation, and then apply the Nash or O'kelly models to construct the prediction model
- Special attention must be paid to the quality of the input data
- One can improve by trial and error with the rainfall loss evaluation and direct runoff separation
- There are still many more models that deserve to be developed in HYDROLIN or in separate (one HYDROLIN for conceptual models, one HYDROLIN for black-box models)

## 9. Acknowledgement

First, I am grateful to professor J.C. van Dam for his showing me the direction of this study. Especially, I appreciate supervisor H.R. Vermeulen for his elaborate guidance. Last but not least, I must say: J.A. Smith, W. Getachew, ir. W. Schuurmans , ir. M.J. Vos and dr. Koga Kenichi gave me much help in my programming and report writing, I express also my appreciation to them.

## 10. References


- Borland International
  Graphics Toolbox 4.0 User Menu, 1987

- Borland International
  TURBO PASCAL 4.0 User Menu, 1987

- Committee for Hydrological Reserch TNO
  Recent Trends in Hydrograph Synthesis, Proceedings of Technical Meeting 21, Proceedings and Informations No. 13 TNO, The Hague, 1966.

- Dam, J.C. van
  Lecture-notes :
  - Hydrologie (f 15 N), Augustus 1985
  - Hydrological Modelling (f 15 D), 87/88

- Dooge, J.C.I.
  Lecture note : Deterministic Methods in Systems Hydrology , 1986.

- Fleming, G.
  Computer Simulation Techniques in Hydrology
  Environmental Science Series, 1975

- Fleming, G.
  Deterministic Models in Hydrology
  FAO Irrigation and Drainage Paper 32, FAO, Rome 1979

- International Institute for Land Reclamation and Improvement (ILRI) Drainage Principles and applications, Publication 16, Vol. II, Theories of Field Drainage and Watershed Runoff, ILRI, Wageningen 1973

- Nash, J.E.
  Determining Runoff from Rainfall , Proc. Inst. Civ. Eng. Vol 10, pp 163-184, 1988.

- Slot, A.F.M.
  Thesis: Etudes Sur Les Crues Des Rivières Casconnes,1981.

- Vos, M.J.
  Interactief Hydrologisch Computer Programma, 1988

11. symbols

A. report

| | | |
|---|---|---|
| . q(t) | : | hydrograph |
| . u(0,t) | : | instantaneous unit hydrograph |
| . S($i_n$,t) | : | summation curve |
| . S, s | : | storage |
| . U | : | unit volume |
| . A | : | catchment area |
| . d=U/A | : | unit depth |
| . $i_n$(t) | : | netrainfall intensity |
| . i(t) | : | specifiek inflow in a reservoir |
| . k | : | lag time, basin lag |
| . n | : | number of reservoirs in a NASH cascade |
| . dt, $\Delta$t | : | time interval of discrete rainfall and runoff data |

B. Program

| | | |
|---|---|---|
| . p[i] | : | measured effective rainfall intensity |
| . q[i] | : | measured direct runoff |
| . qh[i] | : | calculated hydrograph |
| . qt[i] | : | NASH cascade's IUH |
| . qtt[i] | : | UH with effective rainfall duration dt ($\Delta$t) |
| . a, aa | : | catchment area |
| . d, d1 | : | unit depth (mm, m) |
| . r2, r22[i] | : | coefficient of determination |

APPENDIX A: Program

A.1 HYDROLIN

```
(*****************************************************************)
                        program hydrolin;
(*****************************************************************)
{$R-}    {Range checking off}
{$B+}    {Boolean complete evaluation on}
{$S+}    {Stack checking on}
{$I+}    {I/O checking on}
{$N-}    {No numeric coprocessor}
{$M $4000,0,0} {Turbo 3 default stack and heap}

Uses
  Crt,Printer,turbo3,
  dos,turbolib,types,Okelly,
  hydrol,Nash,hydrocal{,hydrogr
  ,msdoslib,msdosdir,comm300};

 LABEL
    Start,Exit;

(*=============================================================*)
PROCEDURE Help_Menu;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='HELP MENU';
  CASE Menu('Nash_cascade  Hydrolin Okelly Unit_hydrograph Quit')
 OF  'N' : BEGIN
             Help_NASH;
            END;
      'H' : BEGIN
             Help_Hydrolin;
            END;
      'O' : BEGIN
             Help_Okelly;
            END;
      'U' : BEGIN
             Help_UH;
            END;
      'Q' : BEGIN
             IF Yes('... are you sure ?') THEN
             Quit:=True;
            END;
  END; (* Case *)
 HV;
 UNTIL Quit=True;
 ClrScr;
END;  (* Help_Menu *)

(*END==================  HYDROLOO.INC   ==================*)

(*=============================================================*)
PROCEDURE File_Name_Num;
(*=============================================================*)
```

```
BEGIN
 REPEAT
  WriteLn;
  Write('              *** DATA-FILE : NAME / NUMBER ***');
  WriteLn;Delay(500);
  LV;Write('Default drive is');HV;Write(' "space"=main  ');
  WriteLn;Delay(500);
 (LV;WriteLn('Give your file-identification : ');WriteLn;
  Write('of exactly');HV;Write(' 5 ');LV;Write('characters ');
  WriteLn('without: drivenr, and filetype:',Chr(22));
  WriteLn;Write('XXXXX',Chr(13));HV;Readln(FNM);WriteLn;
  Delay(500);) Write('Give the drive name: (a: ,b: or c: ) ');
  Readln(fnm);
  LV;WriteLn('Give your file-specification : ');WriteLn;
  HV;Write('PB');LV;Write(' = bruto              precipitation;');
  WriteLn;
  HV;Write('PW');LV;Write(' = zeros wiped      precipitation :');
  WriteLn;
  HV;Write('PI');LV;Write(' = initial losses    precipitation;');
  WriteLn;
  HV;Write('PF');LV;Write(' = phi-index loss    precipitation;');
  WriteLn;
  HV;Write('PU');LV;Write(' = user-def loss     precipitation;');
  WriteLn;
  HV;Write('PN');LV;Write(' = nett              precipitation;');
  WriteLn;
  HV;Write('PT');LV;Write(' = accumulated nett  precipitation;');
  WriteLn;WriteLn;Delay(500);
  HV;Write('QB');LV;Write(' = bruto             discharge;');
  WriteLn;
  HV;Write('QC');LV;Write(' = constant separ.  discharge ;');
  WriteLn;
  HV;Write('QL');LV;Write(' = linear separation discharge ;');
  WriteLn;
  HV;Write('QU');LV;Write(' = user-def separ.   discharge ;');
  WriteLn;
  HV;Write('QN');LV;Write(' = nett              discharge;');
  WriteLn;
  HV;Write('QW');LV;Write(' = zeros wiped       discharge');
  WriteLn;
  HV;Write('QT');LV;Write(' = accumulated nett  discharge;');
  WriteLn;WriteLn;
  HV;Write('UH');LV;Write(' = unit hydrograph             .');
  WriteLn;WriteLn;
  Write('XX',Chr(13));HV;Readln(FileSpec);WriteLn;
  LV;Write('Give your file-number : (');HV;Write('0,1,2');
  LV;Write(' or');HV;Write(' 3) :');WriteLn;WriteLn;
  Write('X',Chr(13));HV;Readln(FileNume);WriteLn;
  WriteLn(Chr(22));
  FileName:=Concat(FNM,FileSpec,FileNume,'.DAT');
  IF Exist(FileName) = True THEN
  BEGIN
   Bell;
   WriteLn(FileName,' already exists !');Delay(1000);
   WriteLn('You should assemble another name ');
   Write('or do you want to overwrite an existing file ?');LV;
   Write(' (Y/N)');HV;WriteLn;Read(Kbd,Ch);
```

```
  END;
 UNTIL (Exist(FileName) = False) OR (Ch IN ['y','Y']);
END; (* File_Name_Num *)

(*=========================================================*)
PROCEDURE File_Renum;
(*=========================================================*)

Label  DoIt;

BEGIN
 Write('                    *** DATA-FILE : RENUMBER ***');
 WriteLn;Delay(500);
 LV;Write('Default drive is ');HV;Write('"space"=main drive');
 LV;WriteLn; WriteLn; Delay(500); WriteLn
 ('If you desire substitutions in the source-program,consult');
 HV;WriteLn('                  TU DELFT / HOLLAND.');WriteLn;
 WriteLn; Write('Do you want to use File_Handling ?  ');
 LV;Write('(Y/N)');HV; Read(Kbd,Ch);WriteLn;IF Ja THEN
 File_Handling; FNM:=Copy(FileName,1,2);
 FileSpec:=Copy(FileName,3,4); FileNumE:=Copy(FileName,5,5);
 IF FileNumE='0' THEN
 BEGIN FileNumR:='1';GOTO DoIt; END ELSE
 IF FileNumE='1' THEN
 BEGIN FileNumR:='2';GOTO DoIt; END ELSE
 IF FileNumE='2' THEN
 BEGIN FileNumR:='3';GOTO DoIt; END ELSE
 IF FileNumE='3' THEN
 BEGIN
  LV;Write('Overwrite an existing datafile ? ');HV;
  Write('(Y/N)');Read(Kbd,Ch);WriteLn;
  IF Ja THEN
  REPEAT
   LV;Write('Give datafile-');HV;Write('number : ');
   GotoXY(20,2);Readln(FileNumR);WriteLn;
  UNTIL Yes('... are you sure ?') ELSE
  BEGIN
   WriteLn('This edited version will be saved as number 4.');
   WriteLn('but will be operable after: Copy/Delete/Rename');
   WriteLn('with sequence-number 1,2 or 3 in FileHandling.');
   WriteLn; FileNumR:='4';
  END;
 END;
 DoIt;
 FileName:=Concat(FNM,FileSpec,FileNumR,'.DAT');
END; (* File_Renum *)

(*=========================================================*)
PROCEDURE File_Text; {Implemented with six textlines}
(*=========================================================*)

BEGIN
  Scroll;ClrScr;
  GotoXY(20,1);Write('*** DATA-FILE : TEXT ***');WriteLn;
  LV;Write('Project/version/Nr.                    : ');HV;
  Readln(Titel);
  Write('Date      (jjmmdd)                  : ');
  Readln(jjmmdd);
```

```
  Write('Dimension of ',StrPQ:20,'      : ');
  Readln(DimStrPQ);
  Write('Dimension of ',Strt:20,'       : ');
  Readln(DimStrt);
  Seq:='SEQ.NR';Spc:='       ';
  WriteLn(FV,'Project/Version/Nr.       : ',Titel);
  WriteLn(FV,'Date       (jjmmdd)       : ',jjmmdd);
  WriteLn(FV);
  WriteLn(FV,Seq:6,StrPQ:20,Strt:20);
  WriteLn(FV,Spc:6,DimStrPQ:20,DimStrt:20);
  WriteLn(FV);
  Scroll;ClrScr;
END; (* File_Text *)

(*=============================================================*)
PROCEDURE Input_File;
(*=============================================================*)
LABEL
    Start;
(VAR  bTEL,fTEL,TEL,MaxTEL,Bf : Integer;
                          Af   : Real   ;)
BEGIN
Writeln;
Start:
 Write('Give input-filename '); Write('(d:inputfile.dat) : ');
 WriteLn; Readln(FileName);WriteLn;
 IF Exist(FileName)=False THEN
 BEGIN
  Bell;
  WriteLn('File does not exist !');
  Delay(3000);
 END;
 IF Exist(FileName)=False THEN GOTO Start;
Assign(FV,FileName);
Reset(FV);
WriteLn;
FOR TEL:=0 TO Arr_Afm DO
BEGIN
 PQ[TEL]:=0.0E00;
 t[TEL]:=00;
END; (* Initialisation *)

BEGIN
FOR tTEL:=1 TO 3 DO
 BEGIN
  Readln(FV,Line);
   WriteLn(Line);
 END;
 ReadLn(FV,Seq,StrPQ,Strt);
 WriteLn(Seq:6,StrPQ:20,Strt:20);
 ReadLn(FV,Spc,DimStrPQ,DimStrt);
 WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
 Readln(FV,Line);
 WriteLn(Line);
 Delay(200);
END; (* Text *)
fTEL:=0;bTEL:=1;
 REPEAT
```

```
   fTEL:=fTEL+1;
   IF fTEL=bTEL*15 THEN
   BEGIN
    bTEL:=bTEL+1;
    WriteLn;PressKey;
    Scroll;ClrScr;
    WriteLn;
    WriteLn(Seq:6,StrPQ:20,Strt:20);
    WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
    WriteLn('-----------------------------------------------');
    WriteLn;
   END;
    Readln(FV,fTEL,Af,Bf);
   TEL:=fTEL;
   PQ[TEL]:=Af;
   t[TEL]:=Bf;
   WriteLn(TEL:6,PQ[TEL]:20:3,t[TEL]:20);
   Delay(200);
 UNTIL Eof(FV);
MaxTEL:=TEL;
Close(FV);
REPEAT
 WriteLn;
 Write('Repeated display of the file-contents ?  ');LV;
 Write('(Y/N)');HV;Read(Kbd,Ch);WriteLn;Scroll;ClrScr;
 IF JA THEN
 BEGIN
  WriteLn;
  WriteLn(Seq:6,StrPQ:20,Strt:20);
  WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
  WriteLn('-----------------------------------------------');
  WriteLn;
  BEGIN
   bTEL:=1;
   FOR TEL:=1 TO MaxTEL DO
   BEGIN
    IF TEL=bTEL*15 THEN
    BEGIN
     bTEL:=bTEL+1;
     WriteLn;PressKey;
     Scroll;ClrScr;
     WriteLn;
     WriteLn(Seq:6,StrPQ:20,Strt:20);
     WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
     WriteLn('-----------------------------------------------');
     WriteLn;
    END;
    WriteLn(TEL:6,PQ[TEL]:20:3,t[TEL]:20);
    Delay(200);
   END;
  END;
 END;
UNTIL Nee;
END; (* Input_File *)
(*****************************************************************
* Depletion; Separ_disch ;Nett_Disch; Disch_Conversion;          *
* Accumul_Nett_Disch; Disch_Menu; Precip_Disc_Menu.              *
*****************************************************************)
```

```
(*============================================================*)
PROCEDURE Depletion;
(*============================================================*)

BEGIN
 Scroll;ClrScr;
 WriteLn('BRUTO_DISCHARGE DATAFILE .......................');
 WriteLn('assumed to have one maximum discharge ..........');
 WriteLn('to calculate end resp. start of depletion-curves ..');
 {PressKey;}
 Input_File;
 Qbr:=PQ;tQbr:=t;
 TEL:=1;
 BEGIN
  REPEAT
   TEL:=TEL+1;
   QT1:=PQ[TEL]/PQ[TEL-1];
   QT2:=PQ[TEL+1]/PQ[TEL];
   QT3:=QT1-QT2;
  UNTIL ((QT2>=0.0E00) OR (ABS(QT3)<=2E-2));
  Qs:=PQ[TEL+1];tQs:=t[TEL+1];
 END; (* Separation starting point *)

 TEL:=1;
 REPEAT
  TEL:=TEL+1;
 UNTIL (((PQ[TEL]-PQ[TEL-1])/(t[TEL]-t[TEL-1]))>=0.0E00) AND
       (((PQ[TEL+1]-PQ[TEL])/(t[TEL+1]-t[TEL]))< 0.0E00);
 LV;WriteLn('                    *** DISCHARGE SEPARATION  ***');HV;
 WriteLn;WriteLn;
 WriteLn('Maximum discharge                    : ',PQ[TEL]:20:3);
 WriteLn('              time                   : ',t[TEL]:20);
(* Maximum discharge *)
 WriteLn;
TEL:=TEL-1;
 BEGIN
  REPEAT
   TEL:=TEL+1;
   QT1:=PQ[TEL]/PQ[TEL-1];
   QT2:=PQ[TEL+1]/PQ[TEL];
   QT3:=QT1-QT2;
  UNTIL ABS(QT3)<=2E-2;
  Qd:=PQ[TEL-1];tQd:=t[TEL-1];
 END; (* Depletion starting point  *)

 WriteLn('Start separation curve at time       : ',tQs:20);
 WriteLn('                    and discharge    : ',Qs:20:3);
 WriteLn;Delay(3000);
 WriteLn('Start depletion  curve at time       : ',tQd:20);
 WriteLn('                    and discharge    : ',Qd:20:3);
 WriteLn; PressKey;
END; (* Depletion *)

(*============================================================*)
PROCEDURE Separ_Disch;
(*============================================================*)
```

```
BEGIN
 Depletion;
 Quit:=False;
 REPEAT
  Menu_Heading:='DISCH. SEPARATION';
  CASE Menu('Constant Linear User_def Quit') OF
'C' :
   BEGIN
    WriteLn('Separation assumed to be constant with time .....');
    WriteLn;
    WriteLn('Use now the QC-filenamespecification_option .....');
    StrPQ:='CONSTANT SEPAR. Qc';Strt:='TIME tQc';
    PressKey;
    File_Name_Num;
    Assign(FV,FileName);
    ReWrite(FV);
    File_Text;
    TEL:=0;
    REPEAT
     TEL:=TEL+1;
     Qsc[TEL]:=PQ[TEL];
    UNTIL PQ[TEL]=Qs;   (* First depletion curve   *)
    REPEAT
     IF PQ[TEL]>=Qs THEN
     Qsc[TEL]:=Qs;
     TEL:=TEL+1;
    UNTIL PQ[TEL]<=Qs; (* Constant separationline *)
    TEL:=TEL-1;
    REPEAT
     TEL:=TEL+1;
     Qsc[TEL]:=PQ[TEL];
    UNTIL TEL=MaxTEL;
    FOR TEL:=1 TO MaxTEL DO
    WriteLn(FV,TEL:6,Qsc[TEL]:20:3,t[TEL]:20);
    Close(FV);
    PressKey;
   END; (* Linear constant *)

'L' :
   BEGIN
    WriteLn('Separation assumed to be linear with time .......');
    WriteLn;
    WriteLn('Use now the QL-filenamespecification_option .....');
    StrPQ:='LINEAR SEPAR. Ql';Strt:='TIME tQl';
    PressKey;
    File_Name_Num;
    Assign(FV,FileName);
    ReWrite(FV);
    File_Text;
    TEL:=0;
    REPEAT
     TEL:=TEL+1;
     Qsl[TEL]:=PQ[TEL];
    UNTIL PQ[TEL]=Qs;   (* First depletion curve   *)
    TEL:=TEL-1;
    REPEAT
     TEL:=TEL+1;
     Qsl[TEL]:=Qs+((t[TEL]-tQs)*(Qd-Qs)/(tQd-tQs));
```

```
    UNTIL PQ[TEL]=Qd;   (* Linear separation line  *)
    IF TEL<MaxTEL THEN
    REPEAT
     TEL:=TEL+1;
     Qsl[TEL]:=PQ[TEL];
    UNTIL t[TEL]=MaxTEL; (* Last depletion curve    *)
    FOR TEL:=1 TO MaxTEL DO
    WriteLn(FV,TEL:6,Qsl[TEL]:20:3,t[TEL]:20);
    Close(FV);
    PressKey;
   END; (* Linear up or down *)

'U' :
   BEGIN
    WriteLn('Separation assumed to be user defined ..........');
    WriteLn;
    WriteLn('Use now the QU-filenamespecification_option .....');
    StrPQ:='USER DEF SEPAR. Qu';Strt:='TIME tQu';
    PressKey;
    File_Name_Num;
    Assign(FV,FileName);
    ReWrite(FV);
    File_Text;
    FOR TEL:=1 TO MaxTEL DO
    BEGIN
     IF t[TEL]<=tQs THEN Qsu[TEL]:=PQ[TEL];
     IF ((t[TEL]>tQs) AND (t[TEL]<tQd)) THEN Qsu[TEL]:=0.0E00;
     IF t[TEL]>=tQd THEN Qsu[TEL]:=PQ[TEL];
    END;
    FOR TEL:=1 TO MaxTEL DO
    WriteLn(FV,TEL:6,Qsu[TEL]:20:3,t[TEL]:20);
    Close(FV);
    Delay(3000);
    WriteLn;  WriteLn
    ('.Continue to use NORTEN EDITOR to edit the QU file .');
    Delay(1000);
    Exec('b:ne.com',FileName);
   END; (* User-def *)

'Q' :
   BEGIN
    IF Yes('... are you sure ?') THEN Quit:=True;
    ClrScr;
   END;
  END; (* Case *)
  HV;
 UNTIL Quit=True;
END; (* Separ_disch *)

(*=============================================================*)
PROCEDURE Disch_Conversion;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='DISCHARGE CONVERSION';
```

```
  CASE Menu('Ocub.m/sec 1cub.m/min 2m/sec 3quit') OF
(* Discharge-conversion to cubic m/hr; time in hour          *)
(* Not implemented l/s , 10E6 cub.m/day or month or year, cf/s*)
     '0' :  BEGIN
              MulQ:=1;
              Quit:=True;
            END;
     '1' :  BEGIN
              MulQ:=1/60;
              Quit:=True;
            END;
     '2' :  BEGIN
              Write('Give the cross-section in sqm of the
                    river:');
              Readln(R); WriteLn;
              MulQ:=3600*R;
              Quit:=True;
            END;
     '3' :  BEGIN
              IF Yes('... are you sure ?') THEN Quit:=True;
              ClrScr;
            END;
  END; (* Case *) HV;
 UNTIL Quit=True;
END; (* Discharge_Conversion *)

(*=============================================================*)
PROCEDURE Nett_Disch;
(*=============================================================*)

BEGIN
 WriteLn('BRUTO_DISCHARGE-DATAFILE .........................');
 Delay(3000);
 Input_File;
 Qbr:=PQ;
 tQbr:=t;
 WriteLn('DISCHARGE-SEPARATION_DATAFILE .....................');
 WriteLn
 ('This must be in connection with the bruto_discharge-file');
 Delay(10000);
 Input_File;
 Qbs:=PQ;
 tQbs:=t;
 Disch_Conversion;
 FOR TEL:=1 TO MaxTEL DO
 BEGIN
  StrPQ:='NETT DISCHARGE Qnett';Strt:='TIME tQnett';
  Qnett[TEL]:=Qbr[TEL]-Qbs[TEL];
  Qnett[TEL]:=MulQ*Qnett[TEL];
  tQnett[TEL]:=t[TEL];
 END;
 WriteLn('Give filename to the NETT DISCHARGE file ..........');
 WriteLn('Use now the QN-filenamespecification_option .......');
 Delay(5000);
 File_Name_Num;
 Assign(FV,FileName);
 ReWrite(FV);
 File_Text;
```

```
 FOR TEL:=1 TO MaxTEL DO
 BEGIN
  Af:=Qnett[TEL];
  Bf:=tQnett[TEL];
  WriteLn(FV,TEL:6,Af:20:3,Bf:20);
 END;
 Close(FV);
END; (* Nett_Disch *)

(*=============================================================*)
PROCEDURE Accumul_Nett_Disch;
(*=============================================================*)
(*        Accumulated Nett Discharge                           *)
BEGIN
 WriteLn('WIPED! NETT_DISCHARGE-DATAFILE ..................... ');
 Delay(3000);
 Line:=Line2;
 Input_File;
 Qnett:=PQ;
 tQnett:=t;
 Qntot:=0.0E00;
 FOR TEL:=1 TO MaxTEL DO
 Qntot:=
 Qntot+(Qnett[TEL]+Qnett[TEL-1])*0.5*(tQnett[TEL]-tQnett[TEL-1])
       *3600;
 Write('Do you want to save this in a datafile ? (Y/N) ');HV;
 Read(Kbd,Ch);WriteLn;
 IF Ch IN ['Y','y'] THEN
 BEGIN
  FNM:=Copy(FileName,3,7);
  FileSpec:='QT';
  FileNumE:=Copy(FileName,10,10);
  FileName:=Concat('b:',FNM,'QT',FileNumE,'.DAT');
  Assign(FV,FileName);
  ReWrite(FV);
  FOR TEL:=1 TO 6 DO WriteLn(FV,Line2);
  WriteLn(FV,'Accumulated nett discharge ',Qntot,'in cubic m.');
  Close(FV);
 END;
END; (* Accumul_Nett_Disch *)

(*=============================================================*)
PROCEDURE Disch_Menu;
(*=============================================================*)

 BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='DISCHARGE MENU';
  CASE Menu('Bruto Separ Nett Zero_wiping Accum_nett Quit') OF
     'A' :  BEGIN
              Accumul_Nett_Disch;
            END;
     'B' :  BEGIN
              StrPQ:='DISCHARGE Qbr';Strt:='TIME tQbr';
              PQ:=Qbr;t:=tQbr;
              Keyboard_Input;
```

```
            END;
      'S' : BEGIN
             StrPQ:='DISCHARGE Qbs';Strt:='TIME tQbs';
             Separ_disch;
            END;
      'N' : BEGIN
             StrPQ:='DISCHARGE Qnett';Strt:='TIME tQnett';
             Nett_Disch;
            END;
      'Z' : BEGIN
             Zero_Wiping;
            END;
      'Q' : BEGIN
             IF Yes('... are you sure ?') THEN Quit:=True;
             ClrScr;
            END;
  END; (* Case *) HV;
 UNTIL Quit=True;
END; (* Disch_Menu *)

(*=============================================================*)
PROCEDURE Precip_Disch;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='PRECIP./DISCH. MENU';
  CASE Menu('Precipition Discharge Compare Timestep Quit') OF
      'P' : BEGIN
             Precip_Menu;
            END;
      'D' : BEGIN
             Disch_Menu;
            END;
      'C' : BEGIN
             WriteLn('Do you want to compare ');
             Write('accumulated precipitation and discharge?');
             LV;Write('(Y/N)');HV;
             Read(Kbd,Ch);WriteLn;
             IF Ch IN ['Y','y'] THEN
             BEGIN
              Scroll;ClrScr;
              Accumul_Nett_Precip;
              WriteLn('Accumulated nett precipitation  : ',Pntot,
              ' cubic m.'); WriteLn; Accumul_Nett_Disch;
              WriteLn('Accumulated nett discharge       :
               ',Qntot,
              ' cubic m.');  WriteLn; WriteLn
              ('Pntot-Qntot          = ',Pntot-Qntot,' cubic m.');
              WriteLn
          ('If you are satisfied you can QUIT and use GRAPHICS');
              WriteLn('or derive a Unit Hydrograph ..........');
              WriteLn
          ('Otherwise retry Precipitation/Discharge operations');
              PressKey;
             END; (* Compare *)
```

```
              Delay(2000);ClrScr;
             END;
      'T' :  BEGIN
              Time_Step;
             END;
      'Q' :  BEGIN
              IF Yes('... are you sure ?') THEN Quit:=True;
              ClrScr;
             END;
   END; (* Case *)
   HV;
 UNTIL Quit=True;
END; (* Precip_Disch *)

(*====================== DISCHARGE ============================*)

(*************************************************************
* Editer_Menu; Unit_Hydrograph;Nash_Model; Okelly_Model       *
*                                  , Model_Menu; Main_Menu.   *
*************************************************************)

(*=========================================================*)
PROCEDURE Editer_Menu;
(*=========================================================*)

BEGIN
WriteLn('Required : all editers/filehandlers in drive B !');
Delay(5000);ClrScr;Quit:=False;
 REPEAT
  Menu_Heading:='EDITER MENU';
  CASE Menu('Edlin File_mngr Hydr_edit Nort Turbo Words Quit')
   OF
      'E' : BEGIN
             FileName:='b:EDLIN.COM';
             IF NOT Exist(FileName) THEN
             BEGIN
              Bell;
              WriteLn(FileName,' does not exist !');
              Delay(1000);
             END ELSE
             BEGIN
              Write('Give Edit File Name : ');
              ReadLn(FileName);Delay(1000);
              Exec('b:Edlin.com',FileName);
             END;
             ClrScr;
            END;
      'H' : BEGIN
             Hydr_Edit;
            END;
      'N' : BEGIN
             FileName:='b:NE.COM';
             IF NOT Exist(FileName) THEN
             BEGIN
              Bell;
              WriteLn(FileName,' does not exist !');Delay(1000);
             END ELSE
             BEGIN
```

```
                Write('Give Edit File Name : ');
                ReadLn(FileName);Delay(1000);
                Exec('b:NE.COM',FileName);
               END;
               ClrScr;
              END;
      'T' : BEGIN
               FileName:='b:TURBO.COM';
               IF NOT Exist(FileName) THEN
               BEGIN
                Bell;
                WriteLn(FileName,' does not exist !');Delay(1000);
               END ELSE
               BEGIN
                Exec('b:TURBO.COM','b:');
               END;
               ClrScr;
              END;
      'W' : BEGIN
               FileName:='b:WS.COM';
               IF NOT Exist(FileName) THEN
               BEGIN
                Bell;
                WriteLn(FileName,' does not exist !');Delay(1000);
               END ELSE
               BEGIN
                Exec('b:WS.COM','b:');
               END;
               ClrScr;;
              END;
      'F' : BEGIN
               FileName:='b:FM.COM';
               IF NOT Exist(FileName) THEN
               BEGIN
                 Bell;
                 WriteLn(FileName,' not present on B: !');
                 Delay(1000);
               END ELSE
               BEGIN
                 Write('Give Target Drivers Name :');
                 Readln(FileName);
                 Exec('b:FM.COM',FileName);
               END;
               ClrScr;
              END;
      'Q' : BEGIN
               IF Yes('... are you sure ?') THEN Quit:=True;
               ClrScr;
              END;
  END;  (* Case *)
  HV;
 UNTIL Quit=True;
END; (* Editer_Menu *)

(*=========================================================*)
PROCEDURE Model_Menu;
(*=========================================================*)
```

```
BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='MODEL MENU';
  CASE Menu('Nash_casc Okelly_model Unit_hyg Quit') OF
     'N' :  BEGIN
             NASHMAIN;
             Delay(1000);ClrScr;
            END;
     'O' :  BEGIN
             OKELLYMAIN;
             Delay(1000);ClrScr;
            END;
     'U' :  BEGIN
             WriteLn
             ('Derivation of the Unit Hydrograph must exist:');
             WriteLn;WriteLn('1. Zero-wiped datafiles;');
             WriteLn('2. Files with nett data;');
             WriteLn;
             IF Yes
             ('are you sure these conditions are fulfilled?')
             THEN  BEGIN
              Chain(FV);} hydrocm;
             END ELSE
             WriteLn('Choose again the precip_disch-menu ...');
             PressKey;
            END;
     'Q' :  BEGIN
             IF Yes('... are you sure ?') THEN Quit:=True;
             ClrScr;
            END;
  END; (* Case *)
  HV;
 UNTIL Quit=True;
END; (* Model_Menu *)

(*=============================================================*)
PROCEDURE Main_Menu;
(*=============================================================*)

BEGIN
ClrScr;
Quitt:=False;
 REPEAT
  Menu_Heading:='MAIN MENU';
  CASE Menu(' Dir Editer Filehandling Help Input Model Quit') OF
      'D' : BEGIN
             Write('Give Directive Name ( a:,b: or c: ) : ');
             Readln(FN);
             Exec('a:command.com','/c dir '+FN+'/p'); Presskey;
            END;
      'E' : BEGIN
             Editer_Menu;
            END;
      'F' : BEGIN
             File_Handling;
            END;
```

```
      'H' : BEGIN
             Help_Menu;
            END;
      'I' : BEGIN
             Precip_Disch;
            END;
      'M' : BEGIN
             Model_Menu;
            END;
      'Q' : BEGIN
             IF Yes('... are you sure ?') THEN Quitt:=True;
             ClrScr;
            END;
  END; (* Case *)
  HV;
 UNTIL Quitt=True;
END; (* Main-Menu *)
BEGIN
 HYDROLIN_Heading;
 Start:
 Main_Menu; (* to subsequent menues *)
 ClrLines(1,5);GotoXY(20,12);
 Write(' Quit  /// HYDROLIN  ///   ???');LV;Write('    (Y/N)');
 HV;Read(Kbd,Ch);WriteLn;
 IF Ja or Yes('...are you sure to quit ?') THEN
 BEGIN
  Scroll;
  GOTO Exit;
 END else
 GOTO Start;
 Exit:
 ClrScr;
END.
(*======================END HYDROLIN======================== *)
```

## A.2 HYDROL

```
(*****************************************************************)
                          unit hydrol;
(*****************************************************************)

{$R-}    {Range checking off}
{$B+}    {Boolean complete evaluation on}
{$S+}    {Stack checking on}
{$I+}    {I/O checking on}
{$N-}    {No numeric coprocessor}

INTERFACE

  Uses
    Crt,Printer,Turbo3,dos,types,
    turbolib(,comm300,msdosdir);

  procedure hydrolin_heading;procedure help_hydrolin;
  procedure Help_UH; procedure Help_NASH;
  procedure File_Name_Num; PROCEDURE File_Renum;
  procedure File_Text; PROCEDURE Input_File;
  procedure Keyboard_Input;Procedure Input_Menu;
  procedure Hydr_Edit; Procedure Time_Step;
  Procedure Area_Define; Procedure Zero_Wiping;
  Procedure Initial_Loss; Procedure Phi_Index;
  Procedure Nett_Precip; Procedure Precip_Conversion;
  Procedure Accumul_Nett_Precip;Procedure Precip_loss;
  Procedure Precip_Menu; procedure Help_TANK;

implementation

(*****************************************************************
* Hydrolin_Heading; Help_Hydrolin; Help_UH;Help_NASH;Help_TANK*
* Help_Menu.                                                    *
*****************************************************************)

(*===============================================================*)
PROCEDURE HYDROLIN_Heading;
(*===============================================================*)

BEGIN
FOR TEL:=1 TO 24 DO WriteLn;Delay(1000);
WriteLn('   H H H H H H H H H H H H H H H H H H H H H H H H H H');
WriteLn('   H                                                 H');
WriteLn('   H                                                 H');
WriteLn('   H                //  //                         H');
WriteLn('                   //  //                           ');
WriteLn('                  //--//                            ');
WriteLn('                 //--//  Y D R O L I N  version 4.00 ');
WriteLn('   H            //  //                               H');
WriteLn('   H                                                 H');
WriteLn('   H                                                 H');
WriteLn('   H   INTERACTIVE HYDROLOGICAL COMPUTERPROGRAM       H');
WriteLn('   H                                                 H');
WriteLn('   H         TU DELFT/Ct (C) 880226 M.J. Vos         H');
WriteLn('   H         TU DELFT/Ct (C) 880701 C.T. Chang       H');
WriteLn('   H                                                 H');
```

```
WriteLn('   H                                                    H');
WriteLn('   H              UNIVERSITY OF TECHNOLOGY  DELFT       H');
WriteLn('   H              DEPARTMENT OF CIVIL ENGINEERING       H');
WriteLn('   H                                                    H');
WriteLn('   H                                                    H');
WriteLn('   H                                                    H');
WriteLn('   H H H H H H H H H H H H H H H H H H H H H H H H H H H');
LV;WriteLn;WriteLn('   WAIT !');HV;
Delay(5000);
Scroll;
END;

(*=============================================================*)
PROCEDURE Help_Hydrolin;   (* INTRODUCTION AND INFO HYDROLIN *)
(*=============================================================*)

BEGIN
ClrScr;
LV;WriteLn('HYDROLIN Help page');HV;
WriteLn;
WriteLn('HYDROLIN is een interactief hydrologisch analyse-,');
WriteLn('programma dat menugericht U ten dienste wil staan.');
WriteLn('Hierbij wordt U gevraagd om gegevens en een keuze ');
WriteLn('uit opties. Stapt u uit een menu dan komt u terug');
WriteLn('in het hoofd menu enkunt u opnieuw een traject kie-');
WriteLn('zen.Een manual is dientengevolge niet strikt nodig.');
WriteLn;
WriteLn('Voorlopig is geimplementeerd :                      ');
WriteLn;
WriteLn('DE METHODE VAN DE EENHEIDSAFVOERGOLF-UNIT HYDROGRAPH');
Delay(500);PressKey;
END; (*Help_Hydrolin *)

(*=============================================================*)
PROCEDURE Help_UH;                      (* INFO UNIT HYDROGRAPH *)
(*=============================================================*)

BEGIN
ClrScr;
LV;WriteLn('UNIT HYDROGRAPH Help page');HV;
WriteLn;
WriteLn('De METHODE VAN DE EENHEIDSAFVOERGOLF berust op empi-');
WriteLn('risch gevonden relaties -L.K. Sherman 1932- en dient');
WriteLn('ter bepaling van oppervlakteafvoercomponent( afvoer)');
WriteLn('die ontstaat ten gevolge van de gemiddelde netto ne-');
WriteLn('erslag; deze neerslag wordt gelijkmatig verdeeld ge-');
WriteLn('dacht over het stroomgebied. De grootte(A)van het');
WriteLn('stroomgebied heeft een bovengrens van ca. 10.000 km2');
WriteLn('Bij de METHODE VAN DE EENHEIDSAFVOERGOLF wordt uitg-');
WriteLn('egaan van een EENHEIDSDIEPTE-UNIT DEPTH -b.v. 1 inch');
WriteLn('of een EENHEIDSVOLUME A ,gelijkmatig verdeeld over ');
WriteLn('het stroomgebied.                                   ');
WriteLn('De tijdsduur van een enkelvoudige bui wordt DT geno-');
WriteLn('emd Vanaf het einde van de bui tot het einde van de ');
WriteLn('eenheidsafvoerkromme is de tijdsduur TR - de AFLOOP-');
WriteLn('TIJD.Het afvoerdebiet wordt weergegeven als Q(q=Q/A)');
WriteLn('Een over een bepaalde tijd gemeten neerslagkromme kan');
WriteLn('worden opgedeeld in een aantal enkelvoudige buien van');
```

```
WriteLn('zo mogelijk zelfde tijdsduur DT.                    ');
WriteLn;
PressKey;
WriteLn('Allereerst zult u data moeten invoeren. Dit geschiedt');
WriteLn('via INPUT/KEYBOARD naar een bestand op schijf. Veran-');
WriteLn('deringen hieraan moeten worden gedaan met een EDITER.');
WriteLn('Allerlei manipulaties met bestanden kunnen worden ');
WriteLn('gedaan met COMMAND(=MSDOS commandos)of FILEHANDLING.');
WriteLn('Vervolgens kunt u Regenverliezen PRECIP en Afvoersch-');
WriteLn('eiding(en)DISCH invoeren, welke eveneens op schijf  ');
WriteLn('worden gezet onder een voor u kenmerkende naam.      ');
WriteLn('De bij elkaar behorende bruto en verlies-datafiles ');
WriteLn('leveren netto-datafiles op.Vergelijking van geaccumu-');
WriteLn('leerde netto regen met afvoer is dan mogelijk.Repeti-');
WriteLn('tie van elk procedee is mogelijk tot een aantal van  ');
WriteLn('gemodificeerde gelijksoortige bestanden op schijf.Het');
WriteLn('beheer geschiedt door twee geimplemenprocedures welke');
WriteLn('u vragen om characters t.b.v. identificatie specific-');
WriteLn('atie en volgnummer.Door filemanipulaties is het moge-');
WriteLn('lijk meer bestanden te bewaren b.v. onder een andere ');
WriteLn('naam,op een andere schijf, op een andere drive etc.De');
WriteLn('GRAPHICS-optie biedt de mogelijkheid tot graphische ');
WriteLn('weergave met HYDROGR en eventuele terugkeer in HYDRO-');
WriteLn('LIN.                  ');
PressKey;
END;  (* HELP_UH *)

(*=============================================================*)
PROCEDURE Help_NASH;              (* INFO NASH CASCADE *)
(*=============================================================*)

BEGIN
ClrScr;
LV;WriteLn('NASH CASCADE Help page');HV;
WriteLn;
WriteLn('   In 1957, J.E. Nash derived the two parameters con-');
WriteLn('ceptual mathematical model using the linear reservoir');
WriteLn('concept:S=K*Q.Storage(S) is in proportion to the out-');
WriteLn('flow(Q).An instantaneous inflow,with volume V, put in');
WriteLn('the first reservoir,its flow will instantaneous rise');
WriteLn('from 0 to V/K, that is:I=0,S=V,O=V/K.Using the cont- ');
WriteLn('inuity equation,We get :Q=(V/K)*EXP(-T/K).The outflow');
WriteLn('of the first reservoir flow to the second reservoir,');
WriteLn('taking the same proces we get the outflow for the se-');
WriteLn('cond reservoir is: Q=(V/K)*EXP(-T/K)*(T/K). Likewise ');
WriteLn('the n-th reservoir :Q=(V/K)*(T/K)**(n-1)*EXP(-T/K)/  ');
WriteLn('Gamma(n). The peak flow and peak time can be derived ');
WriteLn('by taking the deviation of the above equation witht/k');
WriteLn('to be zero:Tm=(N-1)*K ,Um=(1/K*Gamma(N))*EXP(-(N-1))*');
WriteLn('(N-1)**(N-1).The instantaneous unit hydrograph of 1cm');
WriteLn('ER is:U(0,t)=(2.78/K*Gamma(n))*EXP(-T/K)*(T/K)**(N-1)');
WriteLn('. In order to determin K and N for a watershed, we ');
WriteLn;
Delay(500);PressKey;
Scroll;ClrScr;
WriteLn('can use the effective rainfall histogram(ERH)and dir-');
WriteLn('ect runoff hydrograph(DRH),apply the moment method to');
WriteLn('derive it. From the moment method study we have the ');
```

```
WriteLn('following relations:  M1(IUH)=M1(DRH)-M1(ERH)=N*K     ');
WriteLn('   M2(IUH)=M2(DRH)-M2(ERH)-2*N*K*M1(ERH)=N*(N+1)*K*K ');
WriteLn('From the above two equations we can determine the two');
WriteLn('parameters N and K. For the practical application, we');
WriteLn('need to derive the unithydrographU(T,t)with effective');
WriteLn('rainfall duration T,from the instantaneous unithydro-');
WriteLn('graph through the  summation hydrograph. That is:');
WriteLn('U(T,t)=1/T*[S(t)-S(t-T)]=1/T*[Ig((t/k)/N**(1/2)),N-1)');
WriteLn('-Ig((t/k)/N**(1/2))-T,N-1)].          ');
WriteLn('U(deltat,t)=(U(0,t)+U(0,t-deltat))/2  ');
WriteLn('Using matrix inversion we can get hydrograph: (P)*(U)');
WriteLn('=(Q)This model can do the semisimulation approach');
WriteLn('from the ERH and DRH using moment method to determind');
WriteLn('the initial value K and N,then give deltak and del-');
WriteLn('tan using deterministic coefficient to do the bestfit');
WriteLn('proces approach. After best fit K and N is decided,we');
WriteLn('can use it in forcasting any storm.It is alsopossible');
WriteLn('to HYDROGR in HYDROLIN for grafics picture.     ');
PressKey;
WriteLn(' NASH model  execute the following procedure: ');
WriteLn('1. see the helpful message                              ');
WriteLn('2. use the input menu to select the input data          ');
WriteLn('   - copy the exist file and edit the copied file       ');
WriteLn('   - using the existing file                            ');
WriteLn('   - modified the existing file and then to use it      ');
WriteLn('3. give catchment area in km square                     ');
WriteLn('   give unit depth of UH in mm (10mm or 20mm)           ');
WriteLn('4. get the first aproach k, n and R2 value              ');
WriteLn('   compare measured DRH and calculated DRH              ');
WriteLn('5. using autosimulation or semisimulation to improve');
WriteLn('   the k and n value                                    ');
WriteLn('6. deltak and deltan must smaller than 0.01             ');
WriteLn('7. store the best fit unit hydrograph for forcasting ');
WriteLn('8. using the graphics floppy to compare the measured ');
WriteLn(' and calculated DRH ( use HYDROGR command line )       ');
PressKey;
END;  (* HELP_NASH *)

(*=============================================================*)
PROCEDURE Help_OKELLY;              (* INFO O'KELLY MODEL *)
(*=============================================================*)

BEGIN
ClrScr;
LV;WriteLn('O`KELLY MODEL Help page');HV;
WriteLn;
WriteLn(' O'KELLY Model was suggested by O'kelly in 1949.  Its');
WriteLn('concept is considered the rainfall-runoff structureof');
WriteLn(' watershed to be composed of an isosceles triangle    ');
WriteLn('lateral inflow routed through a linear reservoir.     ');
WriteLn(' O'kelly model is a two parameters model (T an K).    ');
WriteLn('The input data, that are used in the model are only   ');
WriteLn('effective rainfal & direct runoff, which are derived  ');
WriteLn('HYDROLIN input menu. Its file name can be PNn.dat and');
WriteLn('QNn.dat (n=0..n).  With the help of moment method     ');
WriteLn('first approach of these two parameters can be derived');
WriteLn('. Using the coefficient of determination acting as    ');
WriteLn('assessment criteria, one can improve these two        ');
```

```
WriteLn('parameters by using the simulation technique          ');
PressKey;
WriteLn(' O'kelly model  execute the following procedure: ');
WriteLn('1. see the helpful message                            ');
WriteLn('2. use the input menu to select the input data        ');
WriteLn('   - copy the exist file and edit the copied file      ');
WriteLn('   - using the existing file                          ');
WriteLn('   - modified the existing file and then to use it     ');
WriteLn('3. give catchment area in km square                   ');
WriteLn('   give unit depth of UH in mm (10mm or 20mm)         ');
WriteLn('4. get the first aproach k, T and R2 value            ');
WriteLn('   compare measured DRH and calculated DRH            ');
WriteLn('5. using autosimulation or semisimulation to improve');
WriteLn('   the k and T value                                  ');
WriteLn('6. deltak and deltaT may be any value                 ');
WriteLn('7. store the best fit unit hydrograph for forcasting ');
WriteLn('8. using the graphics floppy to compare the measured ');
WriteLn(' and  calculated DRH ( use HYDROG command line )     ');
PressKey;
END;  (* HELP_O'KELLY *)

(*****************************************************************
* File_Name_Num; File_Renum; File_Text; Input_File; Keyboard_  *
* Input; Input_Menu; Hydr_Edit.                                 *
*****************************************************************)

(*=============================================================*)
PROCEDURE File_Name_Num;
(*=============================================================*)

BEGIN
 REPEAT
  Writeln;
  Write('              *** DATA-FILE : NAME / NUMBER ***');
  WriteLn;Delay(500);
  LV;Write('Default drive is');HV;Write(' "space"=main  ');
  WriteLn;Delay(500);
 {LV;WriteLn('Give your file-identification : ');WriteLn;
  Write('of exactly');HV;Write(' 5 ');LV;Write('characters ');
  WriteLn('without: drivenr, and filetype:',Chr(22));
  WriteLn;Write('XXXXX',Chr(13));HV;Readln(FNM);WriteLn;
  Delay(500);} Write('Give the drive name: (a: ,b: or c: ) ');
  Readln(fnm);
  LV;WriteLn('Give your file-specification : ');WriteLn;
  HV;Write('PB');LV;Write(' = bruto              precipitation;');
  WriteLn;
  HV;Write('PW');LV;Write(' = zeros wiped      precipitation ;');
  WriteLn;
  HV;Write('PI');LV;Write(' = initial losses    precipitation;');
  WriteLn;
  HV;Write('PF');LV;Write(' = phi-index loss    precipitation;');
  WriteLn;
  HV;Write('PU');LV;Write(' = user-def loss     precipitation;');
  WriteLn;
  HV;Write('PN');LV;Write(' = nett              precipitation;');
  WriteLn;
  HV;Write('PT');LV;Write(' = accumulated nett  precipitation;');
  WriteLn;WriteLn;Delay(500);
```

```
  HV;Write('QB');LV;Write(' = bruto              discharge;');
  WriteLn;
  HV;Write('QC');LV;Write(' = constant separ.  discharge ;');
  WriteLn;
  HV;Write('QL');LV;Write(' = linear separation discharge ;');
  WriteLn;
  HV;Write('QU');LV;Write(' = user-def separ.   discharge ;');
  WriteLn;
  HV;Write('QN');LV;Write(' = nett              discharge;');
  WriteLn;
  HV;Write('QW');LV;Write(' = zeros wiped       discharge');
  WriteLn;
  HV;Write('QT');LV;Write(' = accumulated nett  discharge;');
  WriteLn;WriteLn;
  HV;Write('UH');LV;Write(' = unit hydrograph             .');
  WriteLn;WriteLn;
  Write('XX',Chr(13));HV;Readln(FileSpec);WriteLn;
  LV;Write('Give your file-number : (');HV;Write('0,1,2');
  LV;Write(' or');HV;Write(' 3) :');WriteLn;WriteLn;
  Write('X',Chr(13));HV;Readln(FileNume);WriteLn;
  WriteLn(Chr(22));
  FileName:=Concat(FNM,FileSpec,FileNume,'.DAT');
  IF Exist(FileName) = True THEN
  BEGIN
   Bell;
   WriteLn(FileName,' already exists !');Delay(1000);
   WriteLn('You should assemble another name ');
   Write('or do you want to overwrite an existing file ?');LV;
   Write(' (Y/N)');HV;WriteLn;Read(Kbd,Ch);
  END;
 UNTIL (Exist(FileName) = False) OR (Ch IN ['y','Y']);
END; (* File_Name_Num *)

(*=============================================================*)
PROCEDURE File_Renum;
(*=============================================================*)

Label  DoIt;

BEGIN
 Write('                    *** DATA-FILE : RENUMBER ***');
 WriteLn;Delay(500);
 LV;Write('Default drive is ');HV;Write('"space"=main drive');
 LV;WriteLn; WriteLn; Delay(500); WriteLn
 ('If you desire substitutions in the source-program,consult');
 HV;WriteLn('                 TU DELFT / HOLLAND.');WriteLn;
 WriteLn; Write('Do you want to use File_Handling ?  ');
 LV;Write('(Y/N)');HV; Read(Kbd,Ch);WriteLn;IF Ja THEN
 File_Handling; FNM:=Copy(FileName,1,2);
 FileSpec:=Copy(FileName,3,4); FileNumE:=Copy(FileName,5,5);
 IF FileNumE='0' THEN
 BEGIN FileNumR:='1';GOTO DoIt; END ELSE
 IF FileNumE='1' THEN
 BEGIN FileNumR:='2';GOTO DoIt; END ELSE
 IF FileNumE='2' THEN
 BEGIN FileNumR:='3';GOTO DoIt; END ELSE
 IF FileNumE='3' THEN
 BEGIN
```

```
  LV;Write('Overwrite an existing datafile ? ');HV;
  Write('(Y/N)');Read(Kbd,Ch);WriteLn;
  IF Ja THEN
  REPEAT
   LV;Write('Give datafile-');HV;Write('number : ');
   GotoXY(20,2);Readln(FileNumR);WriteLn;
  UNTIL Yes('... are you sure ?') ELSE
  BEGIN
   WriteLn('This edited version will be saved as number 4,');
   WriteLn('but will be operable after: Copy/Delete/Rename');
   WriteLn('with sequence-number 1,2 or 3 in FileHandling.');
   WriteLn; FileNumR:='4';
  END;
 END;
 DoIt;
 FileName:=Concat(FNM,FileSpec,FileNumR,'.DAT');
END; (* File_Renum *)

(*=============================================================*)
PROCEDURE File_Text; {Implemented with six textlines}
(*=============================================================*)

BEGIN
  Scroll;ClrScr;
  GotoXY(20,1);Write('*** DATA-FILE : TEXT ***');WriteLn;
  LV;Write('Project/version/Nr.                    : ');HV;
  Readln(Titel);
  Write('Date      (jjmmdd)                    : ');
  Readln(jjmmdd);
  Write('Dimension of ',StrPQ:20,'     : ');
  Readln(DimStrPQ);
  Write('Dimension of ',Strt:20,'     : ');
  Readln(DimStrt);
  Seq:='SEQ.NR';Spc:='       ';
  WriteLn(FV,'Project/Version/Nr.       : ',Titel);
  WriteLn(FV,'Date      (jjmmdd)        : ',jjmmdd);
  WriteLn(FV);
  WriteLn(FV,Seq:6,StrPQ:20,Strt:20);
  WriteLn(FV,Spc:6,DimStrPQ:20,DimStrt:20);
  WriteLn(FV);
  Scroll;ClrScr;
END; (* File_Text *)

(*=============================================================*)
PROCEDURE Input_File;
(*=============================================================*)
LABEL
    Start;
{VAR  bTEL,fTEL,TEL,MaxTEL,Bf : Integer;
                          Af   : Real    ;}
BEGIN
Writeln;
Start:
 Write('Give input-filename '); Write('(d:inputfile.dat) : ');
 WriteLn; Readln(FileName);WriteLn;
 IF Exist(FileName)=False THEN
 BEGIN
  Bell;
```

```
  WriteLn('File does not exist !');
  Delay(3000);
 END;
 IF Exist(FileName)=False THEN GOTO Start;
 Assign(FV,FileName);
Reset(FV);
WriteLn;
FOR TEL:=0 TO Arr_Afm DO
BEGIN
 PQ[TEL]:=0.0E00;
 t[TEL]:=00;
END; (* Initialisation *)
BEGIN
 FOR tTEL:=1 TO 3 DO
 BEGIN
  Readln(FV,Line);
   WriteLn(Line);
 END;
 ReadLn(FV,Seq,StrPQ,Strt);
 WriteLn(Seq:6,StrPQ:20,Strt:20);
 ReadLn(FV,Spc,DimStrPQ,DimStrt);
 WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
 Readln(FV,Line);
 WriteLn(Line);
 Delay(200);
END; (* Text *)
fTEL:=0;bTEL:=1;
 REPEAT
  fTEL:=fTEL+1;
  IF fTEL=bTEL*15 THEN
  BEGIN
   bTEL:=bTEL+1;
   WriteLn;PressKey;
   Scroll;ClrScr;
   WriteLn;
   WriteLn(Seq:6,StrPQ:20,Strt:20);
   WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
   WriteLn('-----------------------------------------------');
   WriteLn;
  END;
   Readln(FV,fTEL,Af,Bf);
  TEL:=fTEL;
  PQ[TEL]:=Af;
  t[TEL]:=Bf;
  WriteLn(TEL:6,PQ[TEL]:20:3,t[TEL]:20);
  Delay(200);
 UNTIL Eof(FV);
MaxTEL:=TEL;
Close(FV);
REPEAT
 WriteLn;
 Write('Repeated display of the file-contents ?  ');LV;
 Write('(Y/N)');HV;Read(Kbd,Ch);WriteLn;Scroll;ClrScr;
 IF JA THEN
 BEGIN
  WriteLn;
  WriteLn(Seq:6,StrPQ:20,Strt:20);
  WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
```

```
  WriteLn('-----------------------------------------------');
  WriteLn;
  BEGIN
   bTEL:=1;
   FOR TEL:=1 TO MaxTEL DO
   BEGIN
    IF TEL=bTEL*15 THEN
    BEGIN
     bTEL:=bTEL+1;
     WriteLn;PressKey;
     Scroll;ClrScr;
     WriteLn;
     WriteLn(Seq:6,StrPQ:20,Strt:20);
     WriteLn(Spc:6,DimStrPQ:20,DimStrt:20);
     WriteLn('-----------------------------------------------');
     WriteLn;
    END;
    WriteLn(TEL:6,PQ[TEL]:20:3,t[TEL]:20);
    Delay(200);
   END;
  END;
 END;
UNTIL Nee;
END; (* Input_File *)

(*=============================================================*)
PROCEDURE Keyboard_Input;
(*=============================================================*)

LABEL Exit;

BEGIN
 ClrScr;

 FOR TEL:=0 TO Arr_Afm DO
 BEGIN
  PQ[TEL]:=0.0E00;
  t[TEL]:=00;
 END; (* Initialisering *)
 GotoXY(1,1);Write(Seq);
 GotoXY(15,1);Write(StrPQ);GotoXY(40,1);Write(Strt);LV;
 GotoXY(1,2);Write('-----------------------------------------');
 HV;TEL:=0;
 REPEAT
  TEL:=TEL+1;
  GotoXY(1,TEL+3);Write(TEL:6);
  GotoXY(15,TEL+3);Readln(Apr);
  GotoXY(40,TEL+3);Readln(Bpr);
  PQ[TEL]:=Apr;
  t[TEL]:=Bpr;
  GotoXY(1,TEL+4);LV;
  Write('Continue with <RETURN> <<==>> Stop with  S ');
  Read(Kbd,Ch);  ClrLine(1,TEL+4);HV;
 UNTIL (Ch IN ['S','s']);
 MaxTEL:=TEL;
 Scroll;ClrScr;
 Write(Seq:6);Write(StrPQ:20);Write(Strt:20);WriteLn;
 LV;WriteLn('-----------------------------------------');HV;
```

```
 WriteLn;
 REPEAT
  BEGIN
   bTEL:=1;
   FOR TEL:=1 TO MaxTEL DO
   BEGIN
    IF TEL=bTEL*15 THEN
    BEGIN
     bTEL:=bTEL+1;
     WriteLn;PressKey;
     Scroll;ClrScr;
     Write(Seq:6);Write(StrPQ:20);Write(Strt:20);WriteLn;
     LV;Write('----------------------------------------');HV;
     WriteLn;
    END;
    WriteLn(TEL:6,PQ[TEL]:20:3,t[TEL]:20);
    Delay(200);
   END;
  END;
  WriteLn;
  Write('Repeated display of the file-contents ?  ');LV;
  Write('(Y/N)');HV;Read(Kbd,Ch);WriteLn;Scroll;ClrScr;
 UNTIL Nee;
 Write('SAVE keyboard-input ?');LV;Write(' (Y/N)');
 HV;Read(Kbd,Ch);WriteLn;ClrScr;
 IF Nee THEN GOTO EXIT ;
 IF Ja THEN
 REPEAT
  File_Name_Num;
  IF Exist(FileName)=True THEN
  BEGIN
   Bell;
   WriteLn(FileName,' exists !');
   Delay(1000);
   Write('Overwrite ',FileName,' ?');LV;Write(' (Y/N)');HV;
   Read(Kbd,Ch);WriteLn;
  END;
 UNTIL ((Exist(FileName)=False) OR Ja);
 Assign(FV,FileName);
 ReWrite(FV);
 File_Text;
 FOR TEL:=1 TO MaxTEL DO
 BEGIN
  fTEL:=TEL;
  Af:=PQ[TEL];
  Bf:=t[TEL];
  WriteLn(FV,fTEL:6,Af:20:3,Bf:20);
 END;
 Close(FV);
 Exit;
 PressKey;
END; (* Keyboard_Input *)
(*=============================================================*)
PROCEDURE Input_Menu;
(*=============================================================*)

BEGIN
ClrScr;
```

```
Quit:=False;
 REPEAT
  Menu_Heading:='INPUT MENU';
  CASE Menu('Keyboard Filehandling Inputfile Test Quit') OF
       'F' :  BEGIN
                File_Handling;
              END;
       'K' :  BEGIN
                WriteLn('Used for first input of data.');
                WriteLn('For modifications use the I option.');
                Write('Do you want to continue ? (Y/N)');
                HV;Read(Kbd,Ch);WriteLn;PressKey;
                IF Ja THEN Keyboard_Input ELSE
              END;
       'I' :  BEGIN
                WriteLn('Used for modifications. ');Delay(1000);
                WriteLn('Use for first input data the K option.');
                Write('Do you want to continue ? (Y/N)');
                HV;Read(Kbd,Ch);WriteLn;{PressKey;}
                IF Ja THEN Input_File ELSE
              END;
       'T' :  BEGIN
                LV;WriteLn('TestFiles :');
                Write('Precipitation : ');HV;Delay(1000);
                Write('b:PBO.DAT');WriteLn;LV;
                Write('Discharge     : ');HV;Delay(1000);
                Write('b:QBO.DAT');WriteLn;
                Write('Do you want to continue ? (Y/N) ');HV;
                WriteLn;Read(Kbd,Ch);
                IF Ja THEN Input_File ELSE
              END;
       'Q' :  BEGIN
                IF Yes('... are you sure ?') THEN Quit:=True;
              END;
  END; (* Case *)
  HV;
 UNTIL Quit=True;
 Scroll;ClrScr;
END; (* Input-Menu *)

(*=============================================================*)
PROCEDURE Hydr_Edit;
(*=============================================================*)

LABEL Exit;

BEGIN
 Scroll;GotoXY(20,12);
 Write('      /// HYDR');LV;Write('olin-');HV;Write('EDIT');
 LV;Write('er ');HV;Write('1.00 ///  ');
 Delay(3000);Scroll;ClrScr;
 WriteLn('You will see the contents of a chosen input-file.');
 WriteLn('Afterwards remember the line-numbers to be edit !');
 {PressKey;}Writeln;
 Input_File;
 Write('Do you want to continue with HYDREDIT ?  ');LV;
 Write('(Y/N)');HV;Read(Kbd,Ch);WriteLn;IF Nee THEN GOTO Exit;
 REPEAT
```

```
 PressKey;
 GotoXY(1,1);LV;WriteLn('HYDREDIT 1.00');HV;
 GotoXY(1,5);Write(Seq:6);
 GotoXY(15,5);Write(StrPQ:20);GotoXY(40,5);Write(Strt:20);LV;
 GotoXY(1,6);Write('----------------------------------------');
 HV;WriteLn;
 sTEL:=0;
 REPEAT
  GotoXY(1,7+sTEL);Write(Seq:6,' : ');GotoXY(20,1);Readln(TEL);
  GotoXY(1,7+sTEL);ClrEol;
  GotoXY(1,7+sTEL);Write(TEL:6);
  GotoXY(15,7+sTEL);Write(PQ[TEL]:20);LV;Write(' wordt ');
  HV;Af:=PQ[TEL];GotoXY(50,7+sTEL);Readln(Af);PQ[TEL]:=Af;
  GotoXY(15,7+sTEL);ClrEol;Write(PQ[TEL]:20);
  GotoXY(40,7+sTEL);Write(t[TEL]:20);LV;Write(' wordt ');
  HV;Bf:=t[TEL];GotoXY(70,7+sTEL);Readln(Bf);t[TEL]:=Bf;
  GotoXY(40,7+sTEL);ClrEol;
  Write(t[TEL]:20);WriteLn;WriteLn;
  GotoXY(1,23);LV;Write('More ?   (Y/N)');Read(Kbd,Ch);
  GotoXY(1,23);ClrEol;
  WriteLn;HV;
  sTEL:=sTEL+1;
 UNTIL Nee;
 REPEAT
  Scroll;ClrScr;
  GotoXY(1,1);LV;WriteLn('HYDREDIT 1.00');HV;
  GotoXY(1,5);Write(Seq:6);
  Write(StrPQ:20);Write(Strt:20);LV;
  GotoXY(1,6);Write('----------------------------------------');
  HV;WriteLn;
  TEL:=1;bTEL:=1;
  WHILE t[TEL]<>00 DO
  BEGIN
   WriteLn(TEL:6,PQ[TEL]:20:3,t[TEL]:20);
   Delay(200);
   TEL:=TEL+1;
   IF TEL=bTEL*15 THEN
   BEGIN
    bTEL:=bTEL+1;
    WriteLn;PressKey;
   END; (* Block *)
  END;
  WriteLn;WriteLn;WriteLn('End of data-file.');
  WriteLn;Write('Repeated display of file-contents ?  ');LV;
  Write('(Y/N)');HV;Read(Kbd,Ch);WriteLn;
 UNTIL Nee; ClrScr;
 WriteLn;WriteLn;GotoXY(1,12);
 Write('      !!! End of Edit-session !!! ?  ');LV;
 Write('(Y/N)');HV;Read(Kbd,Ch);WriteLn;
UNTIL (Ja AND Yes('... o.k. ?'));
Write('Save data ?  ');LV;Write(' (Y/N)');Read(Kbd,Ch);WriteLn;
IF Ch in ['Y','y'] THEN {Save_Data}
BEGIN
 LV;Write('Previous FileName                    : ');HV;
 Write(FileName);WriteLn;
 REPEAT
 File_Renum;
 LV;Write('Next FileName in sequence    : ');HV;
```

```
  Write(FileName);WriteLn;
  IF Exist(FileName)=True THEN
   BEGIN
    Bell;WriteLn(FileName,' exists !');
    Delay(1000);
    WriteLn('Overwrite ',FileName,' ?');LV;Write(' (Y/N)');HV;
    Read(Kbd,Ch);WriteLn;
   END;
  UNTIL ((Exist(FileName)=False) OR Ja);
  Assign(FV,FileName);
  Rewrite(FV);
  File_Text;
  TEL:=1;
  WHILE t[TEL]<>0 DO
  BEGIN
   fTEL:=TEL;
   Af:=PQ[TEL];
   Bf:=t[TEL];
   WriteLn(FV,fTEL:6,Af:20,Bf:20);
   TEL:=TEL+1;
  END;
  Close(FV);
  PressKey;
  Scroll;ClrScr;
 END;
 ClrScr;Exit:
END; (* Hydr_Edit *)

(*****************************************************************
* Time_Step; Area_Define; Zero_Wiping; Initial_Loss; Phi_Index;*
* Nett_Precip;Precip_Conversion; Accumul_Nett_Precip; Precip_Mn*
*****************************************************************)

(*=============================================================*)
PROCEDURE Time_Step;
(*=============================================================*)

{VAR  dt,ts,nTEL,nMaxTEL,
      timestep          : Integer;
      nPQ               : Arr;
      nt                : ArrInt;
      iPQ               : Real;}

BEGIN
 ClrScr;
 Quit:=False;
 WriteLn('WARNING !');Delay(1000);WriteLn;LV; WriteLn
 ('The program will give runtime error if the declared bounds ');
 WriteLn('of      a      new      timestep-array-initiated     in
constant:Arr_Afm');
 WriteLn('in the declaration at the beginning of the HYDROLIN');
 WriteLn('will be to small.');HV;PressKey;
 REPEAT
  Menu_Heading:='TIME-STEP MENU';
  CASE Menu('Precipitation Discharge Quit') OF
'P' :
   BEGIN
    WriteLn;Write('Give filename ...');LV;
```

```
      Write('  (d:XXXXXPXXX.DAT)');HV; WriteLn;Delay(3000);
      Input_File;
      Write('Give a new timestep              : ');
      GotoXY(30,10);Readln(timestep);WriteLn;
      ts:=timestep;
      dt:=t[2]-t[1];
      TEL:=0;nTEL:=0;
      PQ[0]:=0.0E00;nPQ[0]:=0.0E00;
      t[0]:=0;nt[0]:=0;
      IF ts<=dt THEN      (* Smaller timestep *)
      REPEAT
       TEL:=TEL+1;
       REPEAT
        nTEL:=nTEL+1;
        nt[nTEL]:=nt[nTEL-1]+ts;
        nPQ[nTEL]:=PQ[TEL];
       UNTIL nt[nTEL]>=t[TEL];
       nPQ[nTEL]:=
PQ[TEL]*(t[TEL]-nt[nTEL-1])/dt+PQ[TEL+1]*(nt[nTEL]-t[TEL])/dt;
      UNTIL TEL=MaxTEL;
      IF ts>dt THEN      (* Bigger timestep *)
      REPEAT
       nTEL:=nTEL+1;
       nt[nTEL]:=nt[nTEL-1]+ts;
       REPEAT
        TEL:=TEL+1;
        nPQ[nTEL]:=nPQ[nTEL-1]+PQ[TEL];
       UNTIL t[TEL]>=nt[nTEL];
       iPQ:=((nt[nTEL]-t[TEL-1])/dt)*PQ[TEL];
       iPQ:=iPQ+((t[TEL]-nt[nTEL])/dt)*PQ[TEL+1];
       nPQ[nTEL]:=(nPQ[nTEL]-PQ[TEL]+iPQ)*dt/(nt[nTEL]-nt[nTEL-1]);
      UNTIL TEL=MaxTEL;
      nMaxTEL:=nTEL;
      FNM:=Copy(FileName,1,2);
      FileName:='PTS.DAT';
      WriteLn('Your timestep-filename will be :PTS.DAT');
      Write('Do you agree ? ');LV;Write(' (Y/N)');HV;Read(Kbd,Ch);
      IF Nee THEN
       BEGIN
        LV;Write('Give your  name of  the file  :
');HV;Readln(FileName);
       END;
      Assign(FV,FileName);
      ReWrite(FV);
      File_Text;
      FOR nTEL:=1 TO nMaxTEL DO
      BEGIN
       Af:=nPQ[nTEL];
       Bf:=nt[nTEL];
       WriteLn(FV,nTEL:6,Af:20:3,Bf:20);
      END;
      Close(FV);
      WriteLn('Your filename has received a new number only !');
      WriteLn('If you want to rename:use the Filehandling');
      PressKey;
    END; (* Precipitation *)
'D' :
    BEGIN
```

```
WriteLn;Write('Give filename ...');LV;
Write('  (d:XXXXXQXX.DAT)');HV; WriteLn;Delay(3000);
Input_File;
Write('Give a new timestep   : ');
GotoXY(30,10);Readln(timestep);WriteLn;
ts:=timestep;
dt:=t[2]-t[1];
TEL:=0;nTEL:=0;
PQ[0]:=0.0E00;nPQ[0]:=0.0E00;
t[0]:=0;nt[0]:=0;
IF ts<=dt THEN      (* Smaller timestep *)
REPEAT
 TEL:=TEL+1;
 REPEAT
  nTEL:=nTEL+1;
  nt[nTEL]:=nt[nTEL-1]+ts;
  nPQ[nTEL]:=nPQ[nTEL-1]+(PQ[TEL]-PQ[TEL-1])*ts/dt;
 UNTIL nt[TEL]>=t[TEL];
 nPQ[nTEL]:=
 nPQ[nTEL-1]+(PQ[TEL]-PQ[TEL-1])*(t[TEL]-nt[nTEL-1])/dt;
 nPQ[nTEL]:=
 nPQ[nTEL]+(PQ[TEL+1]-PQ[TEL])*(nt[nTEL]-t[TEL])/dt;
UNTIL TEL=MaxTEL;
IF ts>dt THEN      (* Bigger timestep *)
REPEAT
 nTEL:=nTEL+1;
 nt[nTEL]:=nt[nTEL-1]+ts;
 REPEAT
  TEL:=TEL+1;
  nPQ[nTEL]:=nPQ[nTEL-1]+PQ[TEL]-PQ[TEL-1];
 UNTIL t[TEL]>=nt[nTEL];
 iPQ:=((nt[nTEL]-t[TEL-1])/dt)*(PQ[TEL]-PQ[TEL-1]);
 nPQ[nTEL]:=nPQ[nTEL-1]+PQ[TEL-1]-PQ[TEL-2]+iPQ;
UNTIL TEL=MaxTEL;
nMaxTEL:=nTEL;
FNM:=Copy(FileName,3,7);
FileName:='QTS.DAT';
WriteLn('Your timestep-filename will be QTS.DAT');
Write('Do you agree ? ');LV;Write(' (Y/N)');HV;Read(Kbd,Ch);
WriteLn; IF Nee THEN
 BEGIN
  LV;Write('Give your name of the file : ');HV;
  GotoXY(30,15);Readln(FileName);WriteLn;
 END;
Assign(FV,FileName);
ReWrite(FV);
File_Text;
FOR nTEL:=1 TO nMaxTEL DO
BEGIN
 Af:=nPQ[nTEL];
 Bf:=nt[nTEL];
 WriteLn(FV,nTEL:6,Af:20:3,Bf:20);
END;
Close(FV);
WriteLn('Your filename has received a new number only !');
WriteLn('If you want to rename : use the Filehandling .');
PressKey;
END; (* Discharge *)
```

```
'Q' :
   BEGIN
    IF Yes('... are you sure ?') THEN Quit:=True;
    ClrScr;
   END;
  END;  (* Case *)
  HV;
 UNTIL Quit=True;
END; (* Time_Step *)

(*=============================================================*)
PROCEDURE Area_Define;
(*=============================================================*)
(*                    Conversion to square m                   *)

{VAR  are,sqm,ha,sqkm  :  Real;}
BEGIN
 ClrScr;
 WriteLn('HYDROLIN/test/1.00 - Area       :  52510 ha ........');
 PressKey;
 Quit:=False;
 REPEAT
  Menu_Heading:='AREA DEFINE MENU';
  CASE Menu('Are Hectare M(sq) Km(sq)  Quit') OF
      'A' : BEGIN
             Write('Area                (are) : ');
             GotoXY(30,10);Readln(are);WriteLn;
             Area:=are*1E+2;{sq.m}
             WriteLn('Area               (sq.m) : ',AREA);
             Delay(5000);
             Quit:=True;
            END;
      'H' : BEGIN
             Write('Area                 (ha) : ');
             GotoXY(30,10);Readln(ha);WriteLn;
             Area:=ha*1E+4;{sq.m}
             WriteLn('Area               (sq.m) : ',AREA);
             Delay(5000);
             Quit:=True;
            END;
      'M' : BEGIN
             Write('Area               (sq.m) : ');
             GotoXY(30,10);Readln(sqm);WriteLn;
             Area:=sqm;{sq.m}
             WriteLn('Area               (sq.m) : ',AREA);
             Delay(5000);
             Quit:=True;
            END;
      'K' : BEGIN
             Write('Area                (sq.km) : ');
             GotoXY(30,10);Readln(sqkm);WriteLn;
             Area:=sqkm*1E+06;{sq.m}
             WriteLn('Area               (sq.m) : ',AREA);
             Delay(5000);
             Quit:=True;
            END;
      'Q' : BEGIN
             IF Yes('... are you sure ?') THEN  Quit:=True;
```

```
            ClrScr;
           END;
  END;  (* Case *)
  HV;
  UNTIL Quit=True;
  WriteLn;
  WriteLn('Further characteristics  : not yet implemented.');
  Delay(3000);
 END; (* Area_define *)

(*=============================================================*)
PROCEDURE Zero_Wiping;
(*=============================================================*)

{VAR    WipeTEL : Integer;}
BEGIN
 WriteLn('DATAFILE to apply zero-wiping to .................');
 WriteLn;{PressKey;;}
 Input_File;
 WriteLn('ZERO-WIPED DATAFILENAME ......................... ');
 WriteLn;PressKey;
 File_Name_Num;
BEGIN
  TEL:=0;
  REPEAT
   TEL:=TEL+1;
  UNTIL PQ[TEL]<>0.0E00;
  WipeTEL:=TEL;
  REPEAT
   TEL:=TEL+1;
  UNTIL PQ[TEL]=0.0E00;
  MaxTEL:=TEL;
 END;
 Assign(FV,FileName);
 Rewrite(FV);
 File_Text;
 PressKey;
 FOR TEL:=WipeTEL TO MaxTEL DO
 BEGIN
  Af:=PQ[TEL];Bf:=t[TEL]-t[WipeTEL-1];
  WriteLn(FV,TEL-WipeTEL+1:6,Af:20:3,Bf:20);
 END;
 Close(FV);
END; (* Zero_Wiping *)

(*========================= PRECIPITATION ====================*)

(*=============================================================*)
PROCEDURE Initial_Loss;
(*=============================================================*)

BEGIN
 Write('ZERO-WIPED  PRECIPITATION-DATAFILE  to  apply  initial
lose');
 LV;Write(' (d:XXXXXPWX.DAT)');HV;WriteLn;{PressKey;}
 Input_File;
 WriteLn('INITIAL-LOSS DATAFILE  :');Delay(3000);WriteLn;WriteLn;
 StrPQ:='INITIAL LOSS Iinit';Strt:='TIME tIinit';
```

```
 FOR TEL:=1 TO MaxTEL DO PQ[TEL]:=0.0E00;
 WriteLn('First an initialisation-datafile is written .......');
 WriteLn('give INITIAL_LOSS_DATAFILE name ..................');
 WriteLn('Use the PI-option of .............................');
 {PressKey;}
 File_Name_Num;
 Assign(FV,FileName);
 ReWrite(FV);
 File_Text;
 FOR TEL:=1 TO MaxTEL DO
 WriteLn(FV,TEL:6,PQ[TEL]:20:3,t[TEL]:20);
 Close(FV);Delay(3000); (* Initialisation *)
 LV; WriteLn;WriteLn;
 WriteLn('.Continue to use Norten Editor to edit the PI file.');
 Exec('b:ne.com',FileName);
END; (* Initial_Loss *)

(*============================================================*)
PROCEDURE Phi_Index;
(*============================================================*)

{VAR       Pbrtot : Real;}

BEGIN
 Write('ZERO-WIPED PRECIPITATION-DATAFILE to apply phi-index');
 LV; Write(' (d:XXXXXPWX.DAT)');WriteLn;HV;{PressKey;}
 Input_File;
 WriteLn('HYDROLIN/test/1.00 - Phi-index  :  2.48 mm/hour ...');
 WriteLn;
 Write('Give estimated phi-index        : ');
 GotoXY(30,10);Readln(Apr);WriteLn;Delay(1000);
 WriteLn;
 WriteLn('PHI_INDEX-DATAFILE name ........................... ');
 WriteLn('Use the PF option of .............................. ');
 PressKey;
 File_Name_Num;
 StrPQ:='PHI-INDEX LOSS Iphi';Strt:='TIME tIphi';
 {
 Write('Phi-index                       : ');Readln(Phi);WriteLn;
 TEL:=0;Pbrtot:=0.0E00;t[0]:=0;
 BEGIN
  REPEAT
   TEL:=TEL+1;
   Pbrtot:=Pbrtot+PQ[TEL]*(t[TEL]-t[TEL-1]);
   MaxTEL:=TEL-1;
  UNTIL PQ[TEL]=0.0E00;
 END;
 Apr:=Phi*Pbrtot/t[MaxTEL];
 Write('Phi-index precipitation loss    : ',Apr); }
 Assign(FV,FileName);
 Rewrite(FV);
 File_Text;
 FOR TEL:=1 TO MaxTEL DO
 BEGIN
  Iphi[TEL]:=Apr;
  Af:=Iphi[TEL];Bf:=t[TEL];
  WriteLn(FV,TEL:6,Af:20:3,Bf:20);
 END;
```

```
 Close(FV);
 PressKey;
END; (* Phi_index *)

(*=============================================================*)
PROCEDURE User_Def_Loss;
(*=============================================================*)

BEGIN
 Write('ZERO-WIPED      PRECIPITATION-DATAFILE      to      apply
userdef.lose');
 LV;Write(' (d:XXXXXPWX.DAT)');HV;WriteLn;{PressKey;}
 Input_File;
 WriteLn('USERDEF-LOSS DATAFILE  :');Delay(3000);WriteLn;WriteLn;
 StrPQ:='USERDEF LOSS Iusr';Strt:='TIME tIusr';
 FOR TEL:=1 TO MaxTEL DO PQ[TEL]:=0.0E00;
 WriteLn('First an initialisation-datafile is written ........');
 WriteLn('USERDEF_LOSS_DATAFILE name ........................');
 WriteLn('Use the PU-option of ..............................');
 PressKey;
 File_Name_Num;
 Assign(FV,FileName);
 ReWrite(FV);
 File_Text;
 FOR TEL:=1 TO MaxTEL DO
 WriteLn(FV,TEL:6,PQ[TEL]:20:3,t[TEL]:20);
 Close(FV);Delay(3000); (* Initialisation *)
 LV;
 WriteLn;WriteLn;
 WriteLn('.Continue to use NORTEN EDITOR to edit the PU file.');
 Exec('b:ne.com',FileName);
END; (* User_Def_Loss *)

(*=============================================================*)
PROCEDURE Precip_Loss;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='PRECIP. LOSS MENU';
  CASE Menu('Initial Phi_index User_def Quit') OF
     'I' :  BEGIN
             StrPQ:='PRECIP_LOSS Iinit';Strt:='TIME tIinit';
             PQ:=Iinit;t:=tIinit;
             Initial_Loss;
            END;
     'P' :  BEGIN
             StrPQ:='PRECIP_LOSS Iphi';Strt:='TIME tIphi';
             PQ:=Iphi;t:=tIphi;
             Phi_Index;
            END;
     'U' :  BEGIN
             StrPQ:='PRECIP_LOSS Iusr';Strt:='TIME tIusr';
             PQ:=Iusr;t:=tIusr;
             User_Def_Loss;
            END;
```

```
      'Q' :  BEGIN
               IF Yes('... are you sure ?') THEN Quit:=True;
               ClrScr;
             END;
  END; (* Case *)
  HV;
  UNTIL Quit=True;
END; (* Precip_Loss *)

(*=============================================================*)
PROCEDURE Precip_Conversion;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='PRECIP. CONVERSION';
  CASE Menu('Omm/hour 1mm/half_an_hour 2mm/day 3quit ') OF
(* Conversion Intensity to mm/hour counted ; time in hour.    *)
(* Not implemented l/s.ha= 8.64 mm/day , mm/year              *)
      '0' :  BEGIN
               MulP:=1.0;
               Quit:=True;
             END;
      '1' :  BEGIN
               MulP:=0.5;
               Quit:=True;
             END;
      '2' :  BEGIN
               Mulp:=1/24;
               Quit:=True;
             END;
      '3' :  BEGIN
               IF Yes('... are you sure ?') THEN Quit:=True;
               ClrScr;
             END;
  END; (* Case *)
  HV;
UNTIL Quit=True; ClrScr;
END; (* Precip_Conversion *)

(*=============================================================*)
PROCEDURE Nett_Write;
(*=============================================================*)

BEGIN
 WriteLn('Give filename to the NETT PRECIPITATION file .......');
 WriteLn('Use now the PN-specification_option of .............');
 Delay(5000);
 File_Name_Num;
 Assign(FV,FileName);
 ReWrite(FV);
 File_Text;
 FOR TEL:=1 TO MaxTEL DO
 BEGIN
  Af:=Inett[TEL];
  Bf:=tInett[TEL];
```

```
  WriteLn(FV,TEL:6,Af:20:3,Bf:20);
 END;
 Close(FV);
END; (* Nett_Write *)

(*=============================================================*)
PROCEDURE Nett_Precip;
(*=============================================================*)

BEGIN
 Write('ZERO-WIPED PRECIPITATION-DATAFILE   ');LV;
 Write(' (d:XXXXXPWX.DAT)');WriteLn; HV;{PressKey;}
 Input_File;
 Ibr:=PQ;
 tIbr:=t;
 WriteLn('PRECIPITATION_LOSS_DATAFILE(S)  ');LV;
 WriteLn('d:XXPIX.DAT and/or d:XXPFX.DAT and/or d:XXPUX.DAT');
 WriteLn;HV;PressKey;
 Quit:=False;
 REPEAT
  Menu_Heading:='PRECIP_LOSS_DATAFILE';
  CASE Menu('Initial Phi_index User_def Quit') OF
'I' :
  BEGIN
   WriteLn('You should give a filename like d:XXXXXPIX.DAT ...');
   {PressKey;}
   Input_File;
   Iinit:=PQ;tIinit:=t;
   StrPQ:='NETT PRECIP Inett';Strt:='TIME tInett';
   Precip_Conversion;
   FOR TEL:=1 TO MaxTEL DO
   BEGIN
    IF Ibr[TEL]-Iinit[TEL]<=0.0E00 THEN Inett[TEL]:=0.0E00 ELSE
    BEGIN
     Inett[TEL]:=Ibr[TEL]-Iinit[TEL];
     Inett[TEL]:=Inett[TEL]*MulP;
     tInett[TEL]:=t[TEL];
    END;
   END;
   Nett_Write;
  END;
'P' :
  BEGIN
   WriteLn('You should give a filename like d:XXXXXPFX.DAT ...');
   {PressKey;}
   Input_File;
   Iphi:=PQ;tIphi:=t;
   StrPQ:='NETT PRECIP Inett';Strt:='TIME tInett';
   Precip_Conversion;
   FOR TEL:=1 TO MaxTEL DO
   BEGIN
    IF Ibr[TEL]-Iphi[TEL]<=0.0E00 THEN Inett[TEL]:=0.0E00 ELSE
    BEGIN
     Inett[TEL]:=Ibr[TEL]-Iphi[TEL];
     Inett[TEL]:=MulP*Inett[TEL];
     tInett[TEL]:=t[TEL];
    END;
   END;
```

```
    Nett_Write;
   END;
 'U' :
   BEGIN
    WriteLn('You should give a filename like d:XXXXXPUX.DAT ..');
    {PressKey;}
    Input_File;
    Iusr:=PQ;tIusr:=t;
    StrPQ:='NETT PRECIP Inett';Strt:='TIME tInett';
    Precip_Conversion;
    FOR TEL:=1 TO MaxTEL DO
    BEGIN
     IF Ibr[TEL]-Iinit[TEL]<=0.0E00 THEN Inett[TEL]:=0.0E00 ELSE
     BEGIN
      Inett[TEL]:=Ibr[TEL]-Iusr[TEL];
      Inett[TEL]:=Inett[TEL]*MulP;
      tInett[TEL]:=t[TEL];
     END;
    END;
    Nett_Write;
   END;
 'Q' :
   BEGIN
    IF Yes('... are you sure ?') THEN Quit:=True;
     ClrScr;
   END;
 END; (* Case *)
  HV;
  UNTIL Quit=True;(* Precip_Loss_Menu *)
 END; (* Nett_Precip *)

 (*=============================================================*)
 PROCEDURE Accumul_Nett_Precip;
 (*=============================================================*)
 (*        Accumulated nett precipitation in cubic m            *)

 BEGIN
  Area_Define;
  WriteLn('WIPED NETT-PRECIPITATION-DATAFILE ..... ');
  {PressKey;}
  Line:=Line1;
  Input_File;
  Inett:=PQ;
  tInett:=t;
  Pntot:=0.0E00;
  FOR TEL:=1 TO MaxTEL DO
  Pntot:=
  Pntot+(Inett[TEL]+Inett[TEL-1])*0.5*(tInett[TEL]-tInett[TEL-1]);
  Pntot:=Pntot*1E-3*Area;
  WriteLn('Accumulated nett precipitation : ',Pntot,' cubic m.');
  Write('Do you want to save this in a datafile ? (Y/N) ');HV;
  Read(Kbd,Ch);WriteLn;
  IF Ja THEN
  BEGIN
   FNM:=Copy(FileName,1,2);
   FileSpec:='PT';
   FileNumE:=Copy(FileName,5,5);
   FileName:=Concat(FNM,'PT',FileNumE,'.DAT');
```

```
  Assign(FV,FileName);
  ReWrite(FV);
  FOR TEL:=1 TO 6 DO WriteLn(FV,Line1);
  WriteLn(FV,'Accumulated Nett Precipitation ',Pntot,'cubic m.');
  Close(FV);
 END;ClrScr;
END; (* Accumul_Nett_Precip *)

(*=============================================================*)
PROCEDURE Precip_Menu;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='PRECIPITATION MENU';
CASE Menu('Bruto Zero_wiping Losses Nett Accum_nett Quit') OF
     'B' :  BEGIN
             StrPQ:='PRECIPITATION Ibr';Strt:='TIME tIbr';
             Seq:=' SEQ.NR'; PQ:=Ibr;t:=tIbr;
             Keyboard_Input;
            END;
     'Z' :  BEGIN
             Zero_Wiping;
            END;
     'L' :  BEGIN
             Precip_Loss;
            END;
     'N' :  BEGIN
             StrPQ:='PRECIPITATION Inett';Strt:='TIME tInett';
             PQ:=Inett;t:=tInett;
             Nett_Precip;
            END;
     'A' :  BEGIN
             Accumul_Nett_Precip;
            END;
     'Q' :  BEGIN
             IF Yes('... are you sure ?') THEN Quit:=True;
             ClrScr;
            END;
  END; (* Case *)
  HV;
 UNTIL Quit=True;
END; (* Precip_Menu *)

END;
(*END ======================== HYDROL.PAS ================ *)
```

## A.3 Types

```
(*****************************************************************)
                          unit types;
(*****************************************************************)

{$R-}    {Range checking off}
{$B+}    {Boolean complete evaluation on}
{$S+}    {Stack checking on}
{$I+}    {I/O checking on}
{$N-}    {No numeric coprocessor}

INTERFACE

  CONST
    Arr_Afm             = 30; (* Max. number in dataset *)

  TYPE
    Str1                = STRING[1];
    Str2                = STRING[2];
    Str5                = STRING[5];
    Str6                = STRING[6];
    Str10               = STRING[10];
    Str12               = STRING[12];
    Str14               = STRING[14];
    Str20               = STRING[20];
    Str80               = STRING[80];
    Arr                 = ARRAY[0..Arr_Afm] OF Real;
    ArrInt              = ARRAY[0..Arr_Afm] OF Integer;
    ArrArr              = ARRAY[0..Arr_Afm,0..Arr_Afm] OF Real;

  VAR
    Ch,dumy             : Char;
    Quit,Found,Quitt    : Boolean;
    FileName,FN,FN0,
    FN1,FN2,St          : Str14;
    Filvar,FV,FV0,FV1,
    FV2,F               : Text;
    Line,Line1,Line2,
    MenuStr             : Str80;
    Menu_Heading,Name,
    Titel,Method        : Str20;
    jjmmdd,Seq,Spc      : Str6;
    tt_tt               : Str5;
    Version,FileSpec,
    FNM                 : Str2;
    FileNumE,FileNumR   : Str1;
    PQ,rcQ,Qbr,Qsc,Qsl,
    Qsu,Qnett,Ibr,Qbs,
    Iinit,Iphi,Inett,
    Iusr,nPQ,U,Y,AY,B,C : Arr;
    A,AA,TA             : ArrArr;
    t,tQbr,tQbs,tQnett,
    tIbr,tIinit,tIphi,
    tInett,tIusr,nt     : ArrInt;
    tIinit1,tIinit2,
    tQs,tQd,Bf,Bpr,
    TEL,tTEL,fTEL,i,j,
```

```
    bTEL,sTEL,MaxTEL,
    ChPos,Lengte,k,p,
    N,S,dt,ts,nTEL,
    LineTEL,LineTelMax,
    nMaxTEL,timestep,Nr,
    bNr,Dr,WipeTEL,Q,
    M,II,tQu          : Integer;
    Area,Af,Apr,Qs,
    Pbrtot,Qbrtot,Qd,
    Pntot,R,Qntot,Phi,
    QT1,QT2,QT3,H,G,iPQ,
    MulP,MulQ,are,sqm,
    sqkm,ha,AAH,Qu    : Real;
    StrPQ,Strt,
    DimStrPQ,DimStrt  : Str20;
    LineArr : Array[1..Arr_Afm] of Str80;
IMPLEMENTATION
END.
```

## A.4 Turbolib

```
(*************************************************************)
                        unit turbolib;
(*************************************************************)

*************************************************************
* LV; HV; Bell; ClrLine; ClrLines; Scroll; CursorOff;CursorOn;*
* CopyFile; WriteXY, PressKey; DisplayFile; DrawBloc;Exist;Ja;*
* File_Text; PrintFile; Yes; Menu; MSDOS_Volume;File_Handling;*
* Init_BitImage; Sort; Quicksort; Nee                        *
*************************************************************)

interface

  uses crt,turbo3,printer,dos,types(,MSDOSDIR,COMM300);

  procedure lv; procedure hv; procedure clrline(x,y:integer);
  procedure displayfile(filename:string); procedure bell;
  procedure writexy(x,y:integer;str:string);procedure presskey;
  procedure copyfile;procedure file_handling;
  procedure printfile(Filename:str14);
  procedure sort(var x:arr;var n: integer);
  procedure quicksort(var x:arr; i1,i2 : integer);
  procedure clrlines(first,last:integer);
  function exist(filename:string):boolean;function ja: boolean;
  function nee:boolean; function yes(question: str80):boolean;
  procedure scroll;function menu(menustr:str80):char;

implementation

(*=========================================================*)
PROCEDURE LV;
(*=========================================================*)

BEGIN
  LowVideo;
END; (* LV *)

(*=========================================================*)
PROCEDURE HV;
(*=========================================================*)

BEGIN
  HighVideo;
END; (* HV *)

(*=========================================================*)
PROCEDURE Bell;
(*=========================================================*)
BEGIN
  Write(Chr(7));
  Delay(500);
END; (* Bell *)
{
(*=========================================================*)
PROCEDURE Init_BitImage;CRT-Hdcopy 640X400 with M24-PROCEDURE
(*=========================================================*)
```

```
BEGIN
 Write(LST,#$27,'*',#$4,#$128,#$2);
END; (* Init_BitImage *)
}
(*=============================================================*)
PROCEDURE ClrLine(X,Y:Integer);
(*=============================================================*)

BEGIN
 GotoXY(X,Y);
 ClrEol;
END; (* ClrLine *)

(*=============================================================*)
PROCEDURE ClrLines(First,Last:Integer);
(*=============================================================*)

BEGIN
 FOR i:=First TO Last DO ClrLine(1,i);
END;  (* ClrLines *)

(*=============================================================*)
PROCEDURE Scroll;
(*=============================================================*)

BEGIN
 FOR i:=1 TO 32 DO
 BEGIN
  WriteLn;
  Delay(10);
 END;
END;  (* Scroll *)
{
(*=============================================================*)
PROCEDURE CursorOn;
(*=============================================================*)

BEGIN
  Write(Chr(27),'B',4);
END; (* CursorOn *)

(*=============================================================*)
PROCEDURE CursorOff;
(*=============================================================*)

BEGIN
  Write(Chr(27),'C',4);
END; (* CursorOff *)
}
(*=============================================================*)
PROCEDURE CopyFile;
(*=============================================================*)
{
CONST
  Arr_Afm               = 20;(* max number in dataset *)
VAR LineTel,LineTelMax  : Integer;
    LineArr             : ARRAY[1..Arr_Afm] OF String[80];
```

```
    Filvar,FV           : Text;
    Filename,FN         : string[14];
 )
BEGIN
 Assign(Filvar,FileName);
 Reset(FilVar);
 Assign(FV,FN);
 ReWrite(FV);
 LineTel:=0;
 REPEAT
  LineTel:=LineTel+1;
  ReadLn(Filvar,LineArr[LineTel]);
  IF Eof(Filvar) THEN LineTelMax:=LineTel;
 UNTIL Eof(Filvar);
 FOR LineTel:=1 TO LineTelMax DO
 WriteLn(FV,LineArr[LineTel]);
 Close(Filvar);
 Close(FV);
END; (* CopyFile *)

(*=============================================================*)
PROCEDURE WriteXY(X,Y(,video):Integer;Str:String);
(*=============================================================*)

BEGIN
 GotoXY(X,Y);
 {VideoMode(video,True;)
 Write(Str);
 {VideoMode(video,False);}
END; (* WriteXY *)

(*=============================================================*)
PROCEDURE Presskey;
(*=============================================================*)

{VAR  Ch : Char;}
BEGIN
  ClrLines(23,24);
  LV;WriteXY(1,23,{1,}'Continue with a key ...');
  Read(Kbd,Ch);
  Scroll;ClrScr;HV;
END; (* PressKey *)

(*=============================================================*)
PROCEDURE DisplayFile(FileName:String);
(*=============================================================*)

{VAR Line    : String[80];
    F       : Text;
    Nr,bNr  : Integer;}
BEGIN
 Assign(F,FileName);
 Reset(F);
 Nr:=0;bNr:=1;
 REPEAT
   Nr:=Nr+1;
   ReadLn(F,Line);
   WriteLn(Line);
```

```
    Delay(200);
    IF Nr=BNr*20 THEN
    BEGIN
     bNr:=bNr+1;
     PressKey;
    END;
 UNTIL Eof(F);
 Close(F);
END; (* DisplayFile *)
{
(*=============================================================*)
PROCEDURE DrawBloc(X1,Y1,X2,Y2:Integer);          (* MS DOS *)
(*=============================================================*)

BEGIN
 ClrScr;
 Write(Chr(27),'L',Chr(Y1+31),Chr(X1+31),Chr(Y1+31),Chr(X2+31));
 Write(Chr(27),'L',Chr(Y1+31),Chr(X2+31),Chr(Y2+31),Chr(X2+31));
 Write(Chr(27),'L',Chr(Y2+31),Chr(X2+31),Chr(Y2+31),Chr(X1+31));
 Write(Chr(27),'L',Chr(Y2+31),Chr(X1+31),Chr(Y1+31),Chr(X1+31));
END;  (* DrawBloc *)
}
(*=============================================================*)
FUNCTION Exist(FileName:String) : Boolean;
(*=============================================================*)

{VAR  F : File;}
BEGIN
  Exist:=True;
  IF (FileName<>'TPU:') AND (FileName<>'TPU:') THEN
  BEGIN
    (*$I-*)
    Assign(FV,FileName);
    Reset(FV);
    Close(FV);
    (*$I+*)
    Exist:=(IOresult=0);
  END;
END; (* Exist *)

(*=============================================================*)
FUNCTION Ja:Boolean;
(*=============================================================*)

BEGIN
 Ja:=False;
 IF (Ch IN ['Y','y']) THEN Ja:=True;
END; (* Ja *)

(*=============================================================*)
FUNCTION Nee:Boolean;
(*=============================================================*)

BEGIN
 Nee:=False;
 IF (Ch IN ['N','n']) THEN Nee:=True;
END; (* Nee *)
```

```
(*=========================================================*)
PROCEDURE PrintFile(FileName : Str14);
(*=========================================================*)

BEGIN
  Assign(FV,FileName);
  Reset(FV);
  WriteLn(LST,#$64);
  REPEAT
    ReadLn(FV,Line);
    WriteLn(LST,Line);
  UNTIL Eof(FV);
  WriteLn(LST,#$12);
  Close(FV);
END; (* PrintFile *)

(*=========================================================*)
FUNCTION Yes(question:Str80):Boolean;
(*=========================================================*)

BEGIN
  Write(question);LV;Write(' (Y/N) : ');HV;
  REPEAT Read(Kbd,Ch); UNTIL Ch IN ['y','Y','n','N'];
  WriteLn;
  Yes:=(Ch IN ['y','Y']);
END; (* Yes *)
{
(*====================== MSDOS ============================*)
FUNCTION MSDOS_Volume(drive_nr : Byte) : Str255;
(*=========================================================*)

VAR
    nam : Str11;
    k   : Byte;
    FCB : RECORD
            flag              : Byte;
            dos_reserved      : ARRAY[1..5] OF Byte;
            attribute,
            drive             : Byte;
            name              : ARRAY[1..11] OF Char;
            not_used          : ARRAY[12..36] OF Byte;
          END;
    reg : RECORD
            ax,bx,cx,dx,bp,si,di,ds,es,flags
                              : Integer;
          END;
BEGIN
WITH FCB,reg DO
 BEGIN
  ds:= Seg(FCB);
  dx:= Ofs(FCB);
  ax:= $1A00;
  MSDOS(reg);
  flag:=$FF;
  attribute:=$08;
  drive:=drive_nr;
  FillChar(name,11,'?');
  ax:=$1100;
```

```
  MSDOS(reg);
  nam:=' ';
  k:=1;
  IF Lo(ax)=0 THEN
  WHILE (k<=11) AND (name[k]<>' ') DO
  BEGIN
   nam:=nam+name[k];
   k:=Succ(k);
  END;
  MSDOSVolume:=nam;
 END;
END; (* MSDOS_Volume *)
)
(*=============================================================*)
FUNCTION Menu(MenuStr:Str80):Char;
(*=============================================================*)

{VAR  Ch             : Char;
     ChPos,Lengte,i: Integer;
     Found          : Boolean;
     Line                                :   Str80;}      PROCEDURE
PrintMenu(Menu_Heading:Str20;MenuStr:Str80);

BEGIN
  GotoXY(1,1);ClrEol;LV;
  Write('              *****       ',Menu_Heading,'       *****');
  HV;
  GotoXY(1,3); ClrEol;
  Lengte:=LENGTH(MenuStr);i:=1;
  for  i:= 1 to Lengte DO
  BEGIN
    Ch:=MenuStr[i];
    IF Ch IN ['A'..'Z','0'..'9'] THEN HV ELSE LV;
    Write(Ch);
  END;
  HV; Write(' ? ');
END;
BEGIN
  PrintMenu(Menu_Heading,MenuStr); Read(Kbd,Ch);
  IF Ch IN ['a'..'z'] THEN Ch:=UpCase(Ch);
  ChPos:=1; Found:=False; Line:=MenuStr;
  REPEAT
    Lengte:=LENGTH(Line);
    Found:=(Ch = Line[1]);
    ChPos:=POS(' ',Line)+1;
    Line:=Copy(Line,ChPos,Lengte);
  UNTIL ((ChPos=1) OR Found);
  IF Found THEN Menu:=Ch ELSE
  BEGIN
    Menu:=' ';
    Bell;WriteLn(' ... is no choice !');Delay(500);
  END;
  ClrScr;PrintMenu(Menu_Heading,MenuStr);WriteLn(Ch);
END; (* Menu *)

(*=============================================================*)
PROCEDURE File_Handling;
(*=============================================================*)
```

```
BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='FILE HANDLING MENU';
  CASE Menu('Copy Directory Erase File_mngr Nt_edit Print'+
            ' Rename Show Quit ')   OF
      'C' : BEGIN
             REPEAT
               Write('Give filename');LV;
               Write('  (d:filename.typ) : ');
               HV;Readln(FileName);WriteLn;
               IF Exist(FileName)=True THEN
              BEGIN
               REPEAT
                LV;Write('New filename     : ');
                HV;Readln(FN);WriteLn;
                IF Exist(FN)=True THEN
                BEGIN
                 Bell;WriteLn('File exists !');
                 Delay(1000);
                 Write('Replace ',FileName,' ?');
                END;
               UNTIL Yes('... o.k. ?');
               LV;WriteLn('WAIT !');HV;
               CopyFile;
               LV;Write(FileName,'  copied to  ');
               HV;Write(FN);WriteLn;
              END;
              IF Exist(FileName)=False THEN
              BEGIN
               Bell;
               WriteLn('File does not exist  !');
               Delay(1000);
              END;
              Write('More COPY ?');LV;
              Write('  (Y/N) ');HV;
              Read(Kbd,Ch);WriteLn;WriteLn;
             UNTIL Nee;
             Delay(1000);ClrScr;
            END; (* Copy *)
      'D' : BEGIN
             Write('Give Directive Name (a:, b: or c:): ');
             Readln(FN);
             Exec('a:command.com','/c dir '+FN+'/p');PressKey;
            END; (* Directory *)
      'E' : BEGIN
             REPEAT
              Write('Give filename');LV;
              Write('  (d:filename.typ) : ');
              HV;Readln(FileName);WriteLn;
              Assign(Filvar,FileName);
              IF NOT Exist(FileName) THEN
              BEGIN
               Bell;
               WriteLn('File does not exist !');
               Delay(1000);
              END;
```

```
          IF Exist(FileName)=True THEN
          BEGIN
           Erase(Filvar);
           WriteLn('File is erased !');
           Delay(1000);
          END;
          Write('More ERASE ?');LV;
          Write('  (Y/N) ');HV;
          Read(Kbd,Ch);WriteLn;WriteLn;
         UNTIL Nee;
         Delay(1000);ClrScr;
        END; (* Erase *)
'N' :  BEGIN
         Repeat
          Write('Give File Name b:filename.typ ');
          Readln(FN);
         Until Exist(FN);
         Assign(FV,FN); Reset(FV);
         Exec('b:NE.COM',FN); Close(FV);
        END;  (* Edit *)
'F' : BEGIN
         FileName:='b:FM.COM';
         IF NOT Exist(FileName) THEN
         BEGIN
           Bell;
           WriteLn(FileName,' not present on B: !');
           Delay(1000);
         END ELSE
         BEGIN
           Write('Give Target Drivers Name :');
           Readln(FileName);
           Exec('b:FM.COM',FileName);
         END;
         ClrScr;
        END;  (* FileManager *)

'P' : BEGIN
         Bell;
         LV;WriteLn('You will use the ... ');HV;
         Delay(1000);GotoXY(20,12);
         WriteLn('/// HYDROLIN /// print-program : ');
         Delay(2000);Scroll;ClrScr;
         REPEAT
          Write('Give filename');LV;
          Write('  (d:filename.typ) : ');
          HV;Readln(FileName);WriteLn;
          IF Exist(FileName)=True THEN
          PrintFile(FileName) ELSE
          BEGIN
           Bell;WriteLn('File does not exist');
           Delay(1000);
          END;
          Write('More PRINT ?');LV;
          Write('  (Y/N) ');HV;
          Read(Kbd,Ch);
         UNTIL Nee;
         Delay(1000);ClrScr;
        END; (* Print *)
```

```
        'Q' : BEGIN
               IF Yes('... are you sure ?') THEN Quit:=True;
               ClrScr;
              END; (* Quit *)
        'R' : BEGIN
               REPEAT
                Write('Give old filename');LV;
                Write('  (d:filename.typ) : ');
                HV;Readln(FileName);WriteLn;
                Assign(Filvar,FileName);
                IF Exist(FileName)=True THEN
                BEGIN
                  REPEAT
                   LV;Write('New name                    : ');
                   HV;Readln(FN);WriteLn;
                   IF Exist(FN)=True THEN
                   BEGIN
                    Bell;
                    WriteLn('File exists !');
                   END;
                  UNTIL Exist(FN)=False;
                  Rename(Filvar,FN);
                  WriteLn(FileName,' renamed.',FN);
                END ELSE
                BEGIN
                 Bell;WriteLn('File does not exist !');
                 Delay(1000);
                END;
                Write('More RENAME ?');LV;
                Write('  (Y/N) ');HV;
                Read(Kbd,Ch);WriteLn;WriteLn;
               UNTIL Nee;
               Delay(1000);ClrScr;
              END; (* Rename *)
        'S' : BEGIN
               LV;WriteLn('You will use the ... ');HV;
               Delay(1000);GotoXY(20,12);
               WriteLn('/// HYDROLIN /// display-program : ');
               Delay(2000);Scroll;ClrScr;
               REPEAT
                Write('Give filename');LV;
                Write('  (d:filename.typ):  ');
                HV;Readln(FileName);WriteLn;
                IF Exist(FileName)=True THEN
                DisplayFile(FileName) ELSE
                BEGIN
                 Bell;WriteLn(FileName,' does not exist !');
                 Delay(1000);
                END;
                WriteLn;Write('SHOW more ?');
                LV;Write('  (Y/N) ');HV;
                Read(Kbd,Ch);WriteLn;ClrScr;
               UNTIL Nee;
              END; (* Show *)
{       'U' : BEGIN
               Write('Give filename');LV;
               Write('  (b:filename.typ) :  ');
               HV;Readln(FileName);WriteLn;
```

```
              IF NOT Exist(FileName) THEN
              BEGIN
               Bell;
               WriteLn(FileName,' not yet implemented !');
               Delay(1000);
              END ELSE
              BEGIN
               LV;WriteLn('Input command : ',FileName);HV;
               Delay(3000);
              END;
              ClrScr;
             END; (* User_defined *)        )
  END; (* Case *)
 HV;
 UNTIL Quit=True;
END; (* File_Handling *)

(*=========================================================*)
PROCEDURE SORT(Var X : Arr ; Var N :Integer);
(*=========================================================*)

PROCEDURE WISSEL(Var X,Y : Real);

BEGIN
 H:=X;X:=Y;Y:=H;
END; (*WISSEL *)

BEGIN
 j:=0;
 WHILE j<>N DO
 BEGIN
  j:=j+1;
  k:=j;p:=j;
  WHILE p<>N DO
  BEGIN
   p:=p+1;
   IF X[p]>X[k] THEN k:=p
  END;
  WISSEL(X[j],X[k]);
 END;
END; (* SORT *)

(*=========================================================*)
PROCEDURE QUICKSORT (Var X : Arr; I1,I2 : Integer);
(*=========================================================*)
Var      R       : Integer;
{Var     R,S     : Integer;
        G       : Real;}

PROCEDURE WISSEL(Var X,Y : Real);

BEGIN
H:=X;X:=Y;Y:=H;
END; (*WISSEL *)

BEGIN
 IF I1<I2 THEN
 BEGIN
```

```
  G:=X[(I1+I2) DIV 2];
  R:=I1;S:=I2;
  WHILE R<=S DO
  BEGIN
   WHILE X[R]>G DO R:=R+1;
   WHILE X[S]<G DO S:=S-1;
   IF R<=S THEN
   BEGIN
    WISSEL(X[R],X[S]);
    R:=R+1;S:=S-1;
   END;
  END;
  QUICKSORT(X,I1,S);
  QUICKSORT(X,R,I2);
 END;
END; (* QUICKSORT *)

(*END ================== TURBOLIB.PAS ================== *)
END.
```

## A.5 Hydrocal

```
(*****************************************************************)
                          unit hydrocal;
(*****************************************************************)

{$R-}    {Range checking off}
{$B+}    {Boolean complete evaluation on}
{$S+}    {Stack checking on}
{$I+}    {I/O checking on}
{$N-}    {No numeric coprocessor}

(*****************************************************************
* HydroMat; MatPrint; VecPrint; Gauss;                           *
*****************************************************************)

interface

uses crt,turbo3,types,turbolib,hydrol{,HYDROGR};

procedure gauss(var a:arrarr;var b,x:arr;var n:integer);
procedure matprint(head:str80;n,m:integer;a:arrarr);
procedure vecprint(head:str80;m:integer;y:arr);
procedure hydromat;procedure hydrocm;

implementation

(*===============================================================*)
PROCEDURE Gauss(Var A:ArrArr; Var B,X:Arr; Var N:Integer);
(*===============================================================*)

Var J,K,ROW,COL,MAX,JP1,NM1,KP1 :Integer;
    MUL,TEMP                    :REAL;

BEGIN
    NM1 := N-1;
    FOR J:= 1 TO NM1 DO
    BEGIN
        JP1 := J+1;
        MAX := J;
        FOR ROW := JP1 TO N DO
        BEGIN
            IF A[ROW,J] > A[MAX,J] THEN MAX:=ROW;
        END;
        FOR COL:=J TO N DO
          BEGIN
              TEMP := A[J,COL];
              A[J,COL] := A[MAX,COL];
              A[MAX,COL] := TEMP
          END;
        TEMP := B[J];
        B[J] := B[MAX];
        B[MAX] := TEMP;
        FOR K:=JP1 TO N DO
          BEGIN
            MUL := -A[K,J]/A[J,J];
            FOR COL:=J TO N DO A[K,COL]:=A[K,COL]+MUL*A[J,COL];
            B[K] := B[K] + MUL*B[J]
```

```
          END;
    END;
    X[N] := B[N]/A[N,N];
    FOR K:= NM1 DOWNTO 1 DO
      BEGIN
          TEMP := 0.0;
          KP1 := K+1;
          FOR J:= KP1 TO N DO TEMP:=TEMP+A[K,J]*X[J];
          X[K] := (B[K] - TEMP)/A[K,K]
      END;
END; (* Gauss *)

(*=============================================================*)
PROCEDURE MatPrint(Head : Str80; N,M : Integer; A : ArrArr);
(*=============================================================*)
BEGIN
 WriteLn(Head);WriteLn;
 FOR I:=1 TO M DO
 BEGIN
  FOR J:=1 TO N DO
  WriteLn(A[I,J]:12:3);
 END;
END; (* MatPrint *)

(*=============================================================*)
PROCEDURE VecPrint(Head : Str80; M : Integer; Y: Arr);
(*=============================================================*)

BEGIN
 WriteLn(Head);WriteLn;
 FOR I:=1 TO M DO
 WriteLn(Y[I]:12:3);
END; (* VecPrint *)

(*=============================================================*)
PROCEDURE Hydromat;
(*=============================================================*)

BEGIN
 Write('Give wiped nett-precipitation datafilename  ');LV;
 Write('(d:XXXXXPNX.DAT)');HV;WriteLn;
 For TEL:=1 TO MaxTEL DO PQ[TEL]:=0.0;
 Input_File;
 FOR TEL:=1 TO MaxTEL DO PQ[TEL]:=PQ[TEL]*1E-3;
 B:=PQ;P:=MaxTEL;
 WriteLn('Number of precipitation data P : ',P);
 Write('Give the extent of the area ');Delay(3000);
 Area_Define;PressKey;
 Write('Give wiped nett-discharge     datafilename  ');LV;
 Write('(d:XXXXXQWX.DAT)');HV;WriteLn;
 FOR TEL:=1 TO MaxTEL DO PQ[TEL]:=0.0;
 Input_File;
 FOR TEL:=1 TO MaxTEL DO PQ[TEL]:=PQ[TEL]/Area;
 C:=PQ;Q:=MaxTEL;
 WriteLn('Number of discharge data     Q : ',Q);WriteLn;WriteLn;
 WriteLn('Matrix MxN from : ');WriteLn;
 WriteLn('q1 = u1.i1                                ');
 WriteLn('q2 = u2.i1 + u1.i2                        ');
```

```
 WriteLn('q3 = u3.i1 + u2.i2 + u1.i3                ');
 WriteLn('etc.                                      ');
 WriteLn;WriteLn;WriteLn;
 WriteLn('i1                                u1    q1');
 WriteLn('i2  i1                            u2    q2');
 WriteLn('i3  i2  i1                        u3    q3');
 WriteLn('i4  i3  i2  i1                    u4    q4');
 WriteLn('.   .   .   .   .                 .     . ');
 WriteLn('in  .                       *     .  =  . ');
 WriteLn('                                        qm');
 WriteLn('etc.');PressKey;
 M:=Q;
 N:=M-P; (* Matrix-dimensions *)
 FOR I:=1 TO M DO
 BEGIN
  FOR J:=1 TO N DO
  BEGIN
   A[I,J]:=0.0E00;
  END;
 END; (* Initialisation *)
 MatPrint('INITIALISATION',N,M,A);
 FOR J:=1 TO N DO        (* Column *)
 BEGIN
  FOR I:=J TO (J+P) DO   (* Row *)
  BEGIN
   A[I,J]:=B[J];
   TA[J,I]:=A[I,J];
  END;
 END; (* MatInp *)
 MatPrint('INPUT MATRIX',N,M,A);PressKey;
 MatPrint('TRANSFORM. MATRIX',N,M,TA);PressKey;
 FOR I:=1 TO M DO
 Y[I]:=C[I]; (* Vector *)
 VecPrint('INPUT VECTOR',M,Y);PressKey;
 FOR II:=1 TO N DO
 BEGIN
  FOR I:=1 TO N DO
  BEGIN
   AAH:=0.0;
   FOR J:=1 TO M DO
   AAH:=AAH + TA[I,J] * A[J,II];
   AA[I,II]:=AAH;
  END;
 END;  (* AT*A Matrix *)
 MatPrint('AT*A MATRIX',N,M,AA);PressKey;
 FOR I:=1 TO N DO
 BEGIN
  AAH:=0.0;
  FOR J:=1 TO M DO
  AAH:=AAH + TA[I,J] * Y[J];
  AY[I]:=AAH;
 END;   (* AT*Y *)
 VecPrint('TRANSFORM.VECTOR',M,AY);PressKey;
 Gauss(AA,Y,U,N);
 WriteLn('Calculated Unit Hydrograph ordinates : ');WriteLn;
 FOR I:=1 TO M DO WriteLn('I = ',I:6,'U[',I,'] = ',U[I]:20:3);
 PressKey;
END; (* HydroMat *)
```

```
PROCEDURE HYDROCM;
BEGIN  (* HydroCal *)
 HydroMat;
 Write('Do you want to Save these data ?');LV;
 Write('  (Y/N)');HV; Read(Kbd,Ch);WriteLn;
 IF Ja THEN File_Name_Num;
 Assign(FV,FileName);
 ReWrite(FV);
 StrPQ:='UNIT DISCHARGE QUH';Strt:='TIME tUH';
 DimStrPQ:='cubic m/s';DimStrt:='hour';
 File_Text;
 FOR I:=1 TO M DO WriteLn(I:6,U[I]:20:3,t[I]:20);
 Close(FV);Scroll;ClrScr;
 WriteLn;
END; (* HydroCal *)

(*END ==================HYDROCAL.PAS ======================*)
end.
```

## A.6 Hydrog

```
(*****************************************************************)
                        program hydrog;
(*****************************************************************)

{$R-}    {Range checking off}
{$B+}    {Boolean complete evaluation on}
{$S+}    {Stack checking on}
{$I+}    {I/O checking on}
{$N-}    {No numeric coprocessor}
{$M $4000,16384,65536} {Turbo 3 default stack and heap}

(*****************************************************************
* P_Histo; Q_Poly; PQ_Wind(P_Hist_Wind/Q_Poly_Wind); U_hgr;      *
* Q_Compare; Graphics_Menu; Main program.                        *
*****************************************************************)

interface

Uses
  Dos,Crt,GDriver,Turbo3,types,turbolib,
  printer,GKernel,GWindow,GShell;

CONST
  Arr_Afm             = 30; (* Dimension measurement-datasets *)
      MaxWorldsGlb=4;
      MaxWindowsGlb=16;
      MaxPiesGlb=10;
      MaxPlotGlb=100;
      StringSizeGlb=80;
      HeaderSizeGlb=10;
      RamScreenGlb:boolean=true;
      CharFile:string[StringSizeGlb]='4x6.fon';
      MaxProcsGlb=27;
      MaxErrsGlb=7;

procedure p_histo;procedure q_poly;
procedure pq_wind;procedure graphics_menu;
procedure hydrolin_graphics;

implementation

(*===============================================================*)
PROCEDURE P_Histo;
(*===============================================================*)

VAR     I, DisplyLen, HatchDen : Integer;
        A                      : PlotArray;
        T                      : ARRAY[1..6] OF Str80;
        R                      : Real;
        Ch                     : Char;
        Hatch                  : Boolean;
        FilVar                 : Text;
        FN                     : Str14;
        TEL,fTEL,MaxTEL,Bf     : Integer;
        Af                     : Real;
        Line                   : Str80;
```

```
PROCEDURE P_Histogram;

BEGIN

 DisplyLen:=fTEL;
 SetColorWhite;
 SetBackground(0);
 SetHeaderOn;
 DefineWindow(1,0,0,XMaxGlb,YMaxGlb);
 DefineHeader(1,'PRECIPITATION BAR CHART WITH HATCHING');
 DefineWorld(1,0.0,0.0,100.0,100.0);
 SelectWorld(1);
 SelectWindow(1);
 DrawBorder;
 DrawAxis(8,-8,0,0,0,0,0,0,True);
 Hatch := True;
 HatchDen := 7;
 DrawHistogram(A,-DisplyLen,Hatch,HatchDen);
 I:=1;
 While I <= 6 DO
 BEGIN
  GotoXY(32,1+I);Write(T[I]);
  I := I+1;
 END;
END; (P_Histogram)
BEGIN
 REPEAT
  Write('Give name of precipitation-datafile  ');
  LV;Write('(d:XXXXXPXX.DAT)  ');
  HV;Readln(FN);WriteLN;
  IF Exist(FN)=False THEN
  BEGIN
   Bell;WriteLn(FN,' does not exist !');
   WriteLn('Try again .');Delay(3000);
  END;
 UNTIL (Exist(FN)=True);
 Assign(Filvar,FN);
 Reset(FilVar);
 i:=1;
 While I <= 6 DO
 BEGIN
  Readln(Filvar,Line);
  T[I]:=Line;i:=i+1;
 END;
 I:=0;
 REPEAT
  I:=I+1;
  ReadLn(Filvar,fTEL,Af,Bf);
  A[I,2]:=Af;
 UNTIL Eof(Filvar);
 Close(FilVar);
 InitGraphic;
 P_Histogram;
 REPEAT UNTIL KeyPressed;
 LeaveGraphic;
END; { P_Histo }
```

```
(*=============================================================*)
PROCEDURE Q_Poly;
(*=============================================================*)

VAR
  N,I                          : Integer;
  B, A                         : PlotArray;
  T                            : ARRAY[1..6] OF Str80;
  Ch                           : Char;
  X1, X2                       : Integer;
  FV                           : Text;
  FN                           : Str14;
  TEL,fTEL,MaxTEL,Bf           : Integer;
  Af                           : Real;
  Line                         : Str80;

PROCEDURE Q_Polygon;

BEGIN

 ClearScreen;
 DefineWindow(1, 0, 0, XMaxGlb, YMaxGlb);
 DefineHeader(1, 'DISCHARGE CURVE AS A POLYGON');
 DefineWorld(1,0.0,0.0,100.0,5000.0);
 SelectWorld(1);
 SelectWindow(1);
 SetBackground(0);
 SetHeaderOn;
 DrawBorder;
 DrawAxis(8,-8,0,0,0,0,0,0,True);
 DrawPolygon(A,1,N,0,0,0);
 I:=1;
 While I <= 6 DO
 BEGIN
  GotoXY(32,1+I);Write(T[I]);
  I := I+1;
 END;
END; {Q_Polygon}
BEGIN
 REPEAT
  Write('Give name of discharge-datafile  ');
  LV;Write('(d:XXXXXQXX.DAT)  ');
  HV;Readln(FN);WriteLN;
  IF Exist(FN)=False THEN
  BEGIN
   Bell;WriteLn(FN,' does not exist !');
   WriteLn('Try again .');Delay(3000);
  END;
 UNTIL (Exist(FN)=True);
 Assign(FV,FN);
 Reset(FV);
 i:=1;
 While I <= 6 DO
 BEGIN
  ReadLn(FV,Line);
  T[I]:=Line;i:=i+1;
 END;
 I:=0;
```

```
 REPEAT
  I:=I+1;
  ReadLn(FV,fTEL,Af,Bf);
  A[I,1]:=Bf;
  A[I,2]:=Af;
 UNTIL Eof(FV);
 MaxTEL:=fTEL;
 N:=MaxTEL;
 Close(FV);
 InitGraphic;
 Q_Polygon;
 REPEAT UNTIL KeyPressed;
 LeaveGraphic;
END; { Q_Poly }

(*=============================================================*)
PROCEDURE PQ_Wind;
(*=============================================================*)

VAR
  I,N,DisplyLen,HatchDen,
  X1, X2                          : Integer;
  A,B,C                           : PlotArray;
  T                               : ARRAY[1..6] OF Str80;
  R                               : Real;
  Ch                              : Char;
  FilVar,FV                       : Text;
  FN,FileName                     : Str14;
  TEL,fTEL,MaxTEL,Bf              : Integer;
  Af                              : Real;
  Line                            : Str80;
  CharHeight,CharWidth,y          : Real;

PROCEDURE P_Histo_Wind;

VAR
  Hatch : Boolean;

BEGIN
 Assign(Filvar,FileName);
 Reset(Filvar);
 i:=1;
 While I <= 6 DO
 BEGIN
  Readln(Filvar,Line);
  T[I]:=Line;i:=i+1;
 END;
 I:=0;
 REPEAT
 I:=I+1;
  ReadLn(Filvar,fTEL,Af,Bf);
  A[I,2]:=Af;
 UNTIL Eof(Filvar);
 Close(FilVar);
 DisplyLen:=fTEL;
 DefineWindow(2,Trunc(XMaxGlb/10),Trunc(YMaxGlb/10),
                Trunc(XMaxGlb/2),Trunc(YMaxGlb/2));
 DefineWorld(2,0.0,0.0,100.0,100.0);
```

```
 SelectWorld(2);
 SelectWindow(2);
 SetHeaderOn;
 DefineHeader(2,'PRECIPITATION BAR CHART WITH HATCHING');
 SetBackground(0);
 DrawBorder;
 DrawAxis(8,-8,0,0,0,0,0,0,True);
 Hatch := True;
 HatchDen := 7;
 DrawHistogram(A,-DisplyLen,Hatch,HatchDen);
 i:=1;
 While I <= 6 DO
 BEGIN
  y:=i*25.0;
  DrawTextW(450.0,y,1,T[I]);
  i:=i+1;
 END;
END; {P_Histo_Wind}
PROCEDURE Q_Poly_Wind;

BEGIN
 Assign(FV,FN);
 Reset(FV);
 i:=1;
 While I <= 6 DO
 BEGIN
  ReadLn(FV,Line);
  T[I]:=Line;   i:=i+1;
 END;
 I:=0;
 REPEAT
  I:=I+1;
  ReadLn(FV,fTEL,Af,Bf);
  B[I,1]:=Bf;
  B[I,2]:=Af;
 UNTIL Eof(FV);
 MaxTEL:=fTEL;
 N:=MaxTEL;
 Close(FV);
 DefineWindow(3,Trunc(XMaxG1b/10),Trunc((YMaxG1b*11)/20),
                Trunc(XMaxG1b*5/10),Trunc((YMaxG1b*9)/10));
 DefineWorld(3,0.0,0.0,100.0,5000.0);
 SelectWorld(3);
 SelectWindow(3);
 SetBackground(0);
 SetHeaderOn;
 DefineHeader(3,'DISCHARGE CURVE AS A POLYGON');
 DrawBorder;
 DrawAxis(8,-8,0,0,0,0,0,0,True);
 DrawPolygon(B,1,N,0,0,0);
 i:=1;
 While I <= 6 DO
 BEGIN
  y:=250.0+i*25.0;
  DrawTextW(450.0,y,1,T[I]);
  i:=i+1;
 END;
END;  {Q_Poly_Wind}
```

```
BEGIN (Main PQ_Wind )
 REPEAT
  Write('Give name of precipitation-datafile  ');
  LV;Write('(d:XXXXXPXX.DAT)  ');
  HV;Readln(FileName);WriteLN;
  IF Exist(FileName)=False THEN
  BEGIN
   Bell;WriteLn(FileName,' does not exist !');
   WriteLn('Try again .');Delay(3000);
  END;
 UNTIL (Exist(FileName)=True);
 REPEAT
  Write('Give name of discharge    -datafile  ');
  LV;Write('(d:XXXXXQXX.DAT)  ');
  HV;Readln(FN);WriteLN;
  IF Exist(FN)=False THEN
  BEGIN
   Bell;WriteLn(FN,' does not exist !');
   WriteLn('Try again .');Delay(3000);
  END;
 UNTIL (Exist(FN)=True);
 InitGraphic;
 DrawBorder;
 DefineWindow(1,2,10,XmaxGlb-2,YMaxGlb-10);
 DefineHeader(1,'CONCATENATED PRECIPITATION/DISCHARGE GRAPHICS');
 SetHeaderOn;
 SelectWindow(1);
 SelectWorld(1);
 DefineWorld(1,0.0,0.0,XmaxGlb,YMaxGlb);
 DrawBorder;
 SetBackground(51);
 P_Histo_Wind;
 Q_Poly_Wind;
 REPEAT UNTIL KeyPressed;
 LeaveGraphic;
END; ( PQ_Wind )

(*=============================================================*)
       PROCEDURE Q_Compare;
(*=============================================================*)

VAR
  N,I                      : Integer;
  B, A                     : PlotArray;
  T                        : ARRAY[1..6] OF Str80;
  Ch                       : Char;
  X1, X2                   : Integer;
  FV,FilVar                : Text;
  FN,FileName              : Str14;
  TEL,fTEL,MaxTEL,Bf       : Integer;
  Af                       : Real;
  Line                     : Str80;


PROCEDURE Q_Polygon;

BEGIN
```

```
 ClearScreen;
 DefineWindow(1, 0, 0, XMaxGlb, YMaxGlb);
 DefineHeader(1, 'DISCHARGE CURVE AS A POLYGON');
 {DefineWorld(1,0.0,0.0,100.0,5000.0);}
 FindWorld(1,A,18,1,1);
 FindWorld(1,B,18,1,1);
 SelectWorld(1);
 SelectWindow(1);
 SetBackground(0);
 SetHeaderOn;
 DrawBorder;
 DrawAxis(8,-8,0,0,0,0,0,0,True);
 DrawPolygon(A,1,N,0,0,0);
 DrawAxis(8,-8,0,0,0,0,0,0,True);
 DrawPolygon(B,1,N,7,0,0);
 I:=1;
 While I <= 6 DO
 BEGIN
  GotoXY(32,1+I);Write(T[I]);
  I := I+1;
 END;
END; {Q_Polygon}

BEGIN
 REPEAT
  Write('Give          name          of          discharge-datafile
');LV;Write('(d:QNn.DAT)  ');
  HV;Readln(FN);WriteLN;
  IF Exist(FN)=False THEN
  BEGIN
   Bell;WriteLn(FN,' does not exist !');
   WriteLn('Try again .');Delay(3000);
  END;
 UNTIL (Exist(FN)=True);
 Assign(FV,FN);
 Reset(FV);
 i:=1;
 While I <= 6 DO
 BEGIN
  ReadLn(FV,Line);
  T[I]:=Line;i:=i+1;
 END;
 I:=0;
 REPEAT
  I:=I+1;
  ReadLn(FV,fTEL,Af,Bf);
  A[I,1]:=Bf;
  A[I,2]:=Af;
 UNTIL Eof(FV);
 MaxTEL:=fTEL;
 {N:=MaxTEL;}
 Close(FV);

 REPEAT
  Write('Give name of  clculated-discharge  file
');LV;Write('(b:H.DAT)  ');
  HV;Readln(FileName);WriteLN;
  IF Exist(FileName)=False THEN
```

```
   BEGIN
    Bell;WriteLn(FileName,' does not exist !');
    WriteLn('Try again .');Delay(3000);
   END;
  UNTIL (Exist(FileName)=True);
  Assign(FilVar,FileName);
  Reset(FilVar);
  i:=1;
  While I <= 6 DO
  BEGIN
   ReadLn(FilVar,Line);
   T[I]:=Line;i:=i+1;
  END;
  I:=0;
  REPEAT
   I:=I+1;
   ReadLn(FilVar,fTEL,Af,Bf);
   B[I,1]:=Bf;
   B[I,2]:=Af;
  UNTIL Eof(FilVar);
  MaxTEL:=fTEL;
  N:=MaxTEL;
  Close(FilVar);

  InitGraphic;
  Q_Polygon;
  REPEAT UNTIL KeyPressed;
  LeaveGraphic;
END; { Q_Compare }

(*=============================================================*)
FUNCTION Menu(MenuStr:Str80):Char;
(*=============================================================*)

VAR  Ch              : Char;
     ChPos,Lengte,i: Integer;
     Found           : Boolean;
     Line            : Str80;

 PROCEDURE PrintMenu(Menu_Heading:Str20;MenuStr:Str80);

VAR
  i       :   integer;

BEGIN
  GotoXY(1,1);ClrEol;LV;
  Write('              *****       ',Menu_Heading,'       *****');
  HV;
  GotoXY(1,3); ClrEol;
  Lengte:=LENGTH(MenuStr);i:=1;
  for  i:= 1 to Lengte DO
  BEGIN
    Ch:=MenuStr[i];
    IF Ch IN ['A'..'Z','0'..'9'] THEN HV ELSE LV;
    Write(Ch);
  END;
  HV; Write(' ? ');
END;
```

```
END;
BEGIN
  PrintMenu(Menu_Heading,MenuStr); Read(Kbd,Ch); WriteLn(Ch);
  IF Ch IN ['a'..'z'] THEN Ch:=UpCase(Ch);
  ChPos:=1; Found:=False; Line:=MenuStr;
  REPEAT
    Lengte:=LENGTH(Line);
    Found:=(Ch = Line[1]);
    ChPos:=POS(' ',Line)+1;
    Line:=Copy(Line,ChPos,Lengte);
  UNTIL ((ChPos=1) OR Found);
  IF Found THEN Menu:=Ch ELSE
  BEGIN
    Menu:=' ';
    Bell;WriteLn(' ... is no choice !');Delay(500);
  END;
  ClrScr;PrintMenu(Menu_Heading,MenuStr);WriteLn(Ch);
END; (* Menu *)

(*=============================================================*)
PROCEDURE Graphics_Menu;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='GRAPHICS MENU';
  CASE Menu('Histo_p  Poly_q  Wind_pq  Compare_q  Quit') OF
     'H' :  BEGIN
             P_Histo;
            END;
     'P' :  BEGIN
             Q_Poly;
            END;
     'C' :  BEGIN
             Q_Compare;
            END;
     'W' :  BEGIN
             PQ_Wind;
            END;
     'Q' :  BEGIN
             IF Yes('... are you sure ?') THEN
             Quit:=True;
            END; (* Quit *)
  END; (* Case *)
  HV;
 UNTIL Quit=True;
END;   (* Graphics_Menu *)
(*=============================================================*)
(*                        MAINPROGRAM                          *)
(*=============================================================*)
procedure hydrolin_graphics;

BEGIN
 Scroll;ClrScr;
 Scroll;ClrScr;LV;
 GotoXY(12,5);
```

```
 Write('************************************************************');
 GotoXY(12,6);
 Write('*                                                          *');
 GotoXY(12,7);
 Write('*                                                          *');
 GotoXY(12,8);
 Write('*         HYDROLIN 1.00  TURBO  GRAPHICS                   *');
 GotoXY(12,9);
 Write('*           -------------  -----  --------                 *');
 GotoXY(12,10);
 Write('*                                                          *');
 GotoXY(12,11);
 Write('*                                                          *');
 GotoXY(12,12);
 Write('*         ');LV;Write(' (C) 1988  M.J. Vos TU/DELFT/Ct');
 Write('              *');
 GotoXY(12,13);
 Write('*                                                          *');
 GotoXY(12,14);
 Write('*                                                          *');
 GotoXY(12,15);
 Write('************************************************************');
 Delay(3000);Scroll;ClrScr;HV;
 Graphics_Menu;
END; (* HYDROGR.PAS *)
END.
(*END ==================== HYDROGR.PAS ====================== *)
```

## A.7 Nash

```
(*****************************************************************)
                          unit nash;
(*****************************************************************)

{Unit NASH;}
{$R-}    {Range checking off}
{$B+}    {Boolean complete evaluation on}
{$S+}    {Stack checking on}
{$I+}    {I/O checking on}
{$N-}    {No numeric coprocessor}
{$M $4000,0,0} {Turbo 3 default stack and heap}

interface

  Uses Crt,Turbo3,Graph3,Dos
       ,Types,Turbolib,hydrol;

  TYPE
    String1                  = String[1];

  VAR
    dummy                    : String1;
    f1,f2,f3,f4,f5 ,FV       : Text;
    i, j, kk,nk,ii,dt,Teller: Integer;
    n,k,k0,k1,aa,a,d,d1,dum,p0,p1,p2,q0,q1,
    q2,q21,p21,sq,meanq,sqm,sqq,r2,deltak,
    deltan,km,nm,im,tt,r2m,n0,max: Real;
    r22                      : ARRAY [1..9]  OF Real;
    nq                       : ARRAY [1..9]  OF Real;
    kq                       : ARRAY [1..9]  OF Real;
    p                        : ARRAY [1..100] OF Real;
    q                        : ARRAY [-30..120] OF Real;
    qh                       : ARRAY [-30..199] OF Real;
    qt                       : ARRAY [-30..199] OF Real;
    qtt                      : ARRAY [-30..199] OF Real;
    choice ,dumy             : char;
    answer                   : boolean;
    FN1,FN2,FN,FileName      : String[14];
  function Gamma(nn: real): real;
  procedure presspace;
  procedure input;
  procedure nkvalue;
  procedure iuhtouh;
  procedure uhtoh;
  procedure criteria;
  procedure semisimulation;
  procedure nashmain;

implementation

(*=============================================================*)
Function GAMMA(nn: real): real;
(*=============================================================*)

var
  pp : real;
```

```
  n1 : real;
  n2 : integer;
  pi : array[0..10] of real;
begin
  pp := 1.0;
  n1 := nn-1.0;
      repeat
        pp := pp * n1;
        n1 := n1 - 1.0;
      until n1 < 1.0;
  n1 :=n1*10;
  n2 :=trunc(n1);
  pi[0] := 1.000;
  pi[1] := 0.951;
  pi[2] := 0.918;
  pi[3] := 0.897;
  pi[4] := 0.887;
  pi[5] := 0.886;
  pi[6] := 0.894;
  pi[7] := 0.909;
  pi[8] := 0.931;
  Pi[9] := 0.962;
  Pi[10]:= 1.000;
  nn := pp * pi[n2];
  gamma :=nn;
end;

(*=============================================================*)
FUNCTION Menu(MenuStr:Str80):Char;
(*=============================================================*)

VAR  Ch              : Char;
     ChPos,Lengte,i: Integer;
     Found           : Boolean;
     Line            : Str80;

PROCEDURE PrintMenu(Menu_Heading:Str20;MenuStr:Str80);

VAR
  i      :   integer;

BEGIN
  GotoXY(1,1);ClrEol;LV;
  Write('              *****       ',Menu_Heading,'       *****');
  HV;
  GotoXY(1,3); ClrEol;
  Lengte:=LENGTH(MenuStr);i:=1;
  for  i:= 1 to Lengte DO
  BEGIN
    Ch:=MenuStr[i];
    IF Ch IN ['A'..'Z','0'..'9'] THEN HV ELSE LV;
    Write(Ch);
  END;
  HV; Write(' ? ');
END;
BEGIN
  PrintMenu(Menu_Heading,MenuStr); Read(Kbd,Ch); WriteLn(Ch);
  IF Ch IN ['a'..'z'] THEN Ch:=UpCase(Ch);
```

```
  ChPos:=1; Found:=False; Line:=MenuStr;
  REPEAT
    Lengte:=LENGTH(Line);
    Found:=(Ch = Line[1]);
    ChPos:=POS(' ',Line)+1;
    Line:=Copy(Line,ChPos,Lengte);
  UNTIL ((ChPos=1) OR Found);
  IF Found THEN Menu:=Ch ELSE
  BEGIN
    Menu:=' ';
    Bell;WriteLn(' ... is no choice !');Delay(500);
  END;
  ClrScr;PrintMenu(Menu_Heading,MenuStr);WriteLn(Ch);
END; (* Menu *)

(*=============================================================*)
PROCEDURE Presspace;
(*=============================================================*)
  BEGIN
    GoTOXY(22,22);
    WriteLn('Strike space key when ready...');
    Read(Kbd,dummy);
    ClrScr;
  END;
(*=============================================================*)
PROCEDURE Input_Menu;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='INPUT MENU';
  CASE Menu(' Copyedit Existfile Modifyfile ') OF
      'C' :  BEGIN
               Writeln('Copy the net rainfall file and edit');
               Repeat
                 Write('Give input-filename ');
                 Write('(b:12345pn0.dat) ');
                 Readln(FileName);
               Until Exist(FileName);
               Write('Give output-filename ');
               Write('(b:12345pnN.dat) ');Readln(FN1);
               Exec('\command.com','/c copy '+FileName+' '+FN1);
               Assign(f1,FN1); Reset(f1);
               Exec('b:\NE.COM',FN1); Close(f1);
               Scroll;Clrscr;
               Writeln('Copy the direct runoff file and edit');
               Repeat
                 Write('Give input-filename ');
                 Write('(b:12345qn0.dat): ');Readln(FileName);
               Until Exist(FileName);
               Write('Give output-filename ');
               Write('(b:12345qnN.dat):');Readln(FN2);
               Exec('\command.com','/c copy '+filename+' '+fn2);
               Assign(f2,FN2); Reset(f2);
               Exec('b:\NE.COM',FN2); Close(f2);
               IF Yes('.are you ready to run NASH ?')
```

```
                THEN Quit:=True;
              END;
        'M' : BEGIN
                WriteLn('modify the existing net rainfall file ');
                Delay(100);Write('Net Rainfall : ');HV;Delay(100);
                Repeat
                  Write('Give File Name b:12345pnN.dat ');
                  Readln(FN1);
                Until Exist(FN1);
                Assign(f1,FN1); Reset(f1);
                Exec('b:\NE.COM',FN1); Close(f1);
                WriteLn('modify the existing direct runoff file ');
                Delay(100);Write('Direct Runoff : ');HV;Delay(100);
                Repeat
                  Write('Give File Name b:12345qnN.dat ');
                  Readln(FN2);
                Until Exist(FN2);
                Assign(f2,FN2); Reset(f2);
                Exec('b:\NE.COM',FN2); Close(f2);
                IF Yes('... are you ready to run NASH ?')
                THEN Quit:=True;
              END;
        'E' : BEGIN
                LV;WriteLn('TestFiles :');
                Write('Net Rainfall : ');HV;Delay(100);
                Repeat
                  Write('Give File Name b:12345pn0.dat ');
                  Readln(FN1);
                Until Exist(FN1);
                Write('Direct runoff: ');HV;Delay(100);
                Repeat
                  Write('Give File Name b:12345qn0.dat ');
                  Readln(FN2);WriteLn;
                Until Exist(FN2);
                IF Yes('... are you already to run NASH model ?')
                THEN Quit:=True;
              END;
      ('Q' : BEGIN
                IF Yes('... are you sure ?') THEN Quit:=True;
              END;          )
  END; (* Case *)
  HV;
 UNTIL Quit=True;
 Scroll;ClrScr;
END; (* Input-Menu *)
(*=============================================================*)
PROCEDURE input;
(*=============================================================*)
  BEGIN
    Input_Menu;
    Assign(f1,FN1); Reset(f1);
    Assign(f2,FN2); Reset(f2);
    Assign(f3,'b:nkiuh.dat');  ReWrite(f3);
    Assign(f4,'b:nkuh.dat');   Rewrite(f4);
    Assign(f5,'b:h.dat');      Rewrite(f5);
    ClrScr;
    for i := 1 to 120 do q[i]:=0.0;
    for i := -20 to 10 do qt[i]:=0.0;
```

```
    for i := -20 to 10 do qtt[i]:=0.0;
    GotoXY(10,10);
    Begin
      FOR i :=1 to 6 do
      Readln(f1);
    End;
    Begin
      FOR i :=1 to 6 do
      Readln(f2);
    End;ClrScr;
    Write('Give Area in Km(sq):');Readln(aa);
    Write('Give Unit Depth in mm:');Readln(d);
    WriteLn('Input net rainfall(mm/hr) and its time(hr)');
    ReadLn(f1,i,p[1],dt);p[dt]:=p[1];GotoXY(10,4);
    WriteLn('p[',dt:2,']=',p[dt]:4:1);
    Repeat
      BEGIN
        ReadLn(f1,i,p[i*dt],j);GotoXY(10,3+i);
        WriteLn('p[',j:2,']=',p[j]:6:2);
        kk:=i;
      END;
    Until Eof(f1);Close(f1);
    PRESSPACE;
    WriteLn('Input direct run-off(cms) and its time(hr)');
    Repeat
      BEGIN
        ReadLn(f2,i,q[i*dt],j);GotoXY(10,3+i);
        WriteLn('q[',j:2,'] = ',q[j]:7:2);
        j:=i;
      END;
    Until Eof(f2);Close(f2);
  PRESSPACE;
  END;
(*============================================================*)
PROCEDURE nkvalue;
(*============================================================*)
  BEGIN
    p0 := 0;
    p1 := 0;
    p2 := 0;
    q0 := 0;
    q1 := 0;
    q2 := 0;
    q[0]:=0;
    FOR i := 1 TO kk DO
      BEGIN
        p0 := p0+p[i*dt]*aa*1000*dt;
        p1 := p1+p[i*dt]*(i-0.5)*aa*1000*3600*dt*dt;
       p2:=p2+p[i*dt]*(i-0.5)*(i-0.5)*aa*1000*3600*3600*dt*dt*dt
      END;
    FOR i := 1 TO j DO
      BEGIN
        q0 := q0+((q[(i-1)*dt]+q[i*dt])/2)*3600*dt;
        q1 := q1+(q[i*dt]/2)*((i-1/3)+(i+1/3))*3600*3600*dt*dt;
        q2 := q2+(q[i*dt]/2)*((i-1/3)*(i-1/3)+(i+1/3)*(i+1/3))*
              3600*3600*3600*dt*dt*dt;
      END;
    Writeln('p0=',p0:12:0);
```

```
    Writeln('p1=',p1:16:0);
    Writeln('p2=',p2:20:0);
    Writeln('q0=',q0:12:0);
    Writeln('q1=',q1:16:0);
    Writeln('q2=',q2:20:0);
    p21 := p2-p1*p1/p0;
    q21 := q2-q1*q1/q0;
    k   := (q21-p21)/((q1-p1)*3600);
    n   := (q1-p1)*(q1-p1)/(p0*(q21-p21));
    k0  :=k;
    n0  :=n;
   {Writeln(f3,' k = ');}
    Write(' k =  ',k:5:2,'hr,   n =   ',n:5:2);
   {Writeln(f3,k);}
   {Writeln(f3,' n = ');}
   {Writeln(f3,n);}
    PRESSPACE;
  END;
(*============================================================*)
PROCEDURE iuhTOuh;
(*============================================================*)
  BEGIN
    k1  :=k*3600;
    d1  :=d/1000;
    qt[0] :=0.0;
   {Writeln(f3,'j=',j:4);}
   {WriteLn(f3,'qt[0]=',qt[0]:4:2);}
    FOR i := 1 TO j+3 DO
    BEGIN
      qt[i*dt] := aa*1000000.0*(d1/k1)*
            (1.0/gamma(n))*(exp((n-1)*ln(i*dt/k)))*exp(-i*dt/k);
{     WriteLn(f3,'qt[',i*dt,']=',qt[i*dt]:8:2);  }
{     WriteLn('qt[',i:2,'] = ',qt[i]:5:1);       }
    END;
{   WriteLn(f4,'uh[  0]=',qtt[0]:8:2);           }
    FOR i :=1 TO j+3 DO
    BEGIN
      qtt[i*dt] := (qt[i*dt]+qt[(i-1)*dt])/2;
{     WriteLn(f4,'uh[',i*dt:3,']=',qtt[i*dt]:8:2);     }
{     WriteLn('qtt[',i*dt:2,'] = ',qtt[i*dt]:5:1);     }
    END;
{    PRESSPACE;                                   }
  END;
(*============================================================*)
PROCEDURE uhtoh;
(*============================================================*)
  BEGIN
{    RESET(f4);                                   }
    FOR i :=1 to j+3 DO
    BEGIN
      dum  := 0.0;
      nk   := 0;
      Repeat
        nk  := nk+dt;
        dum := dum+qtt[(i+1)*dt-nk]*p[nk]*dt/d;
      Until nk = kk*dt;
      qh[i*dt] := dum;
      WriteLn('qh[',i*dt:2,']=',qh[i*dt]:5:1,'  q[',i*dt:2,']=',
```

```
                q[i*dt]:5:1);
    END;
{   PRESSPACE;                                  }
  END;
(*=============================================================*)
PROCEDURE criteria;
(*=============================================================*)
  BEGIN
    sq := 0.0;
    FOR i :=1 TO j+3 DO
    BEGIN
      sq := sq+q[i*dt];
    END;
    meanq :=sq/(j+3);
    sqm := 0.0;
    sqq := 0.0;
    FOR i :=1 TO j+3 DO
    BEGIN
       sqm := sqm+(q[i*dt]-meanq)*(q[i*dt]-meanq)*q[i*dt];
      sqq := sqq+(q[i*dt]-qh[i*dt])*(q[i*dt]-qh[i*dt])*q[i*dt];
    END;
    r2 := (sqm/(j+3)-sqq/(j+3))/(sqm/(j+3));
  END;
(*=============================================================*)
PROCEDURE semisimulation;
(*=============================================================*)
  BEGIN
    ClrScr;{Scrool;}
    GotoXY(2,2);WriteLn('which deltak AND deltan will be used'
                        ,'in the semisimulation') ;
    GotoXY(4,4);Write('delta-k = ');Readln(deltak);
    GotoXY(6,6);Write('delta-n = ');Readln(deltan);
    ClrScr;
    km :=0.0;
    im :=0.0;
    nm :=0.0;
    FOR ii := 1 TO 9 DO
      BEGIN
        IF ii = 1 THEN
        BEGIN
          k:=k0;
          n:=n0;
        END;
        IF ii = 2 THEN
        BEGIN
          k:=k0+deltak;
          n:=n0;
        END;
        IF ii = 3 THEN
        BEGIN
          k:=k0-deltak;
          n:=n0;
        END;
        IF ii = 4 THEN
        BEGIN
          k:=k0;
          n:=n0+deltan;
        END;
```

```
    IF ii = 5 THEN
    BEGIN
      k:=k0;
      n:=n0-deltan;
    END;
    IF ii = 6 THEN
    BEGIN
      k:=k0+deltak;
      n:=n0+deltan;
    END;
    IF ii = 7 THEN
    BEGIN
      k:=k0+deltak;
      n:=n0-deltan;
    END;
    IF ii = 8 THEN
    BEGIN
      k:=k0-deltak;
      n:=n0+deltan;
    END;
    IF ii = 9 THEN
    BEGIN
      k:=k0-deltak;
      n:=n0-deltan;
    END;
    iuhTOuh;
    uhtoh;
    criteria;
    r22[ii] :=r2;
    GotoXY(20,20);
    WriteLn('ii=',ii:2,'  k=',k:4:2,'hr,  n=',n:4:2,'  r2=',
            r22[ii]:6:4); Presspace;
    nq[ii]  :=n;
    kq[ii]  :=k;
    IF r22[ii] > r2m  THEN
    BEGIN
      km:=k;
      nm:=n;
      im:=ii;
      r2m:=r22[ii];
    END;
  END;
FOR ii := 1 to 9 DO
  BEGIN
   WriteLn('ii=',ii:2,'   k=',kq[ii]:4:2,'hr,
          n=',nq[ii]:4:2,' r2=',r22[ii]:6:4);
  END;
GoToXY(12,12);
Writeln('old k value = ',k0,'hr');GoToXY(12,14);
Writeln('old n value = ',n0);GoToXY(12,16);
Writeln('new k value = ',km,'hr');GoToXY(12,18);
Writeln('new n value = ',nm); GotoXY(10,21);
Write('Do you want to change k n value ?(y/n)');
Readln(dummy);
IF dummy ='y' THEN
BEGIN
  k0:=km;
  n0:=nm;
```

```
    END;
    GotoXY(10,22);
    Write('Do you intend to improve again  ?(y/n)');
    Readln(dummy);ClrScr;
    IF dummy ='y' THEN
    BEGIN
      semisimulation ;
    END;
  END;
(*============================================================*)
PROCEDURE autosimulation;
(*============================================================*)
  BEGIN
    ClrScr;GotoXY(10,10);
    Writeln('AUTOSIMULATION (Waiting!)');
    deltak:=0.1;
    deltan:=0.1;
    km :=0.0;
    im :=0.0;
    nm :=0.0;
   REPEAT
    Teller:=1;
    FOR ii := 1 TO 9 DO
      BEGIN
        IF ii = 1 THEN
        BEGIN
          k:=k0;
          n:=n0;
        END;
        IF ii = 2 THEN
        BEGIN
          k:=k0+deltak;
          n:=n0;
        END;
        IF ii = 3 THEN
        BEGIN
          k:=k0-deltak;
          n:=n0;
        END;
        IF ii = 4 THEN
        BEGIN
          k:=k0;
          n:=n0+deltan;
        END;
        IF ii = 5 THEN
        BEGIN
          k:=k0;
          n:=n0-deltan;
        END;
        IF ii = 6 THEN
        BEGIN
          k:=k0+deltak;
          n:=n0+deltan;
        END;
        IF ii = 7 THEN
        BEGIN
          k:=k0+deltak;
          n:=n0-deltan;
```

```
    END;
    IF ii = 8 THEN
    BEGIN
      k:=k0-deltak;
      n:=n0+deltan;
    END;
    IF ii = 9 THEN
    BEGIN
      k:=k0-deltak;
      n:=n0-deltan;
    END;
    {iuh to uh}
    BEGIN
     k1  :=k*3600;
     d1  :=d/1000;
     qt[0] :=0.0;
     FOR i := 1 TO j+3 DO
      BEGIN
       qt[i*dt] := aa*1000000.0*(d1/k1)*
       (1.0/gamma(n))*(exp((n-1)*ln(i*dt/k)))*exp(-i*dt/k);
      END;
     FOR i :=1 TO j+3 DO
      BEGIN
       qtt[i*dt] := (qt[i*dt]+qt[(i-1)*dt])/2;
      END;
    END;
  { uh to h }
    BEGIN
     FOR i :=1 to j+3 DO
      BEGIN
       dum  := 0.0;
       nk   := 0;
       Repeat
        nk  := nk+dt;
        dum := dum+qtt[(i+1)*dt-nk]*p[nk]*dt/d;
       Until nk = kk*dt;
       qh[i*dt] := dum;
      END;
    END;
  { criteria }
    BEGIN
     sq := 0.0;
     FOR i :=1 TO j+3 DO
      BEGIN
       sq := sq+q[i*dt];
      END;
     meanq :=sq/(j+3);
     sqm := 0.0;
     sqq := 0.0;
     FOR i :=1 TO j+3 DO
      BEGIN
       sqm := sqm+(q[i*dt]-meanq)*(q[i*dt]-meanq);
       sqq := sqq+(q[i*dt]-qh[i*dt])*(q[i*dt]-qh[i*dt]);
      END;
     r2 := (sqm/(j+3)-sqq/(j+3))/(sqm/(j+3));
    END;
    r22[ii] :=r2;
    nq[ii]  :=n;
```

```
          kq[ii]  :=k;
          IF r22[ii] > r2m  THEN
          BEGIN
            km:=k;
            nm:=n;
            im:=ii;
            r2m:=r22[ii];
            Teller:=2;
          END;
        END;
        IF Teller = 1 THEN
        BEGIN
          deltak:=deltak-0.01;
          deltan:=deltan-0.01;
        END;
        IF Teller = 2 THEN
        BEGIN
          kO:=km;
          nO:=nm;
        END;
    UNTIL deltak <= 0.0;
    GotoXY(10,12);Writeln('BEST K VALUE = ',km:5:2,'hr');
    GotoXY(10,13);Writeln('BEST N VALUE = ',nm:5:2);
    Writeln('Its Deterministic Coefficient =',r2m:6:4);
    for i :=1 to 3 do Writeln; iuhtouh; uhtoh;
    for i :=1 to 3 do Writeln;
    Presspace;
  END;

(*=============================================================*)
PROCEDURE Simulation_Menu;
(*=============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='SIMULATION MENU';
  CASE Menu('Autosimulation Semisimulation Quit') OF
       'A' :  BEGIN
                Autosimulation;
              END;
       'S' :  BEGIN
                Semisimulation;
              END;
       'Q' :  BEGIN
                IF Yes('... are you sure ?') THEN Quit:=True;
              END;
  END; (* Case *)
  HV;
 UNTIL Quit=True;
 Scroll;ClrScr;
END; (* Simulation-Menu *)
    {
(*=============================================================*)
PROCEDURE NASH_GRAPHICS;
(*=============================================================*)
```

```
VAR
  X1, X2 ,GrDriver,GrMode: Integer;
  dx,dy                  : Real;
BEGIN
 ClrScr;
 GrDriver := CGA;
 InitGraph(GrDriver,GrMode,'');
 SetBkColor(0);
 Rectangle(0,GetMaxY,GetMaxX,0);
 MoveTo(0,0);dx:=(j+5)/300;dy:=max/200;
 FOR I:=1 TO J+3 DO
 BEGIN
    x1:=round(i/dx);x2:=round(qt[i*dt]/dy);x2:=200-x2;
    lineto(x1,x2);
 END;
 Repeat until keypressed;
 CloseGraph;
 WriteLn('MaxX : ',GetMaxX:4);
 WriteLn('MaxY : ',GetMaxY:4);
 Delay(3000);
 Repeat until keypressed;
END;) { NASH_GRAPHICS }

procedure nashmain;

LABEL
     Start;

BEGIN
START:
  IF YES('Do you want to see helpful message about NASH model ?')
  THEN  help_nash;
  input;
  nkvalue;
  iuhTOuh;
  uhtoh;
  criteria;
  GoToXY(20,20);Write('r2(deterministic coefficient)=',r2:6:4);
  PRESSPACE;
  r2m := r2;
  simulation_menu;
  IF Yes
  ('Do you want to store n&k value and UH in file b:nkuh.dat ? ')
  THEN
   BEGIN
    Writeln(f3,'BEST N VALUE = ',n:5:2);
    Writeln(f3,'BEST K VALUE = ',k:5:2);
    Writeln(f4,'BEST N VALUE = ',n:5:2);
    Writeln(f4,'BEST K VALUE = ',k:5:2);
    k1  :=k*3600;
    d1  :=d/1000;
    qt[0] :=0.0;
    WriteLn(f3,'IUH[  0]=',qt[0]:8:2);
    FOR i := 1 TO j+3 DO
    BEGIN
      qt[i*dt] := aa*1000000.0*(d1/k1)*
             (1.0/gamma(n))*(exp((n-1)*ln(i*dt/k)))*exp(-i*dt/k);
      WriteLn(f3,'IUH[',i*dt:3,']=',qt[i*dt]:8:2);
```

```
    END;
    WriteLn(f4,'UH[  0]=',qtt[0]:8:2);
    FOR i :=1 TO j+3 DO
    BEGIN
      qtt[i*dt] := (qt[i*dt]+qt[(i-1)*dt])/2;
      WriteLn(f4,'UH[',i*dt:3,']=',qtt[i*dt]:8:2);
    END;
   END;
   for i :=1 to 6 do Writeln(f5);
   Writeln(f5,'     0                0.000                          0 ');
   Max:=0;
   FOR i :=1 to j+3 DO
    BEGIN
      dum  := 0.0;
      nk   := 0;
      Repeat
        nk  := nk+dt;
        dum := dum+qtt[(i+1)*dt-nk]*p[nk]*dt/d;
      Until nk = kk*dt;
      qh[i*dt] := dum; If dum > max then max:=dum;
      Writeln(f5,i:6,qh[i*dt]:20:3,'                    ',i*dt:3);
    END;
  GotoXY(1,10);
  Writeln
  ('Change floppy A2 to run HYDROGR use Q_Compare selection to');
  Writeln
  ('compare the measured(b:qw.day) and calculated(b:h.dat)DRH!');
  Writeln; Close(f3);Close(f4);Close(f5);
  IF YES('Do you want to run again the NASH model ?') THEN
  GOTO START;
  ClrScr;Scroll;
END;
END.
```

A.8 O'kelly

```
(*************************************************************)
            unit   Okelly;
(*************************************************************)
{Unit O'kelly;}
{$R-}    {Range checking off}
{$B+}    {Boolean complete evaluation on}
{$S+}    {Stack checking on}
{$I+}    {I/O checking on}
{$N-}    {No numeric coprocessor}
{$M $4000,0,0} {Turbo 3 default stack and heap}

interface

  Uses Crt,Turbo3,Graph3,Dos
       ,Types,Turbolib,hydrol;

  TYPE
    String1                  = String[1];
  VAR
    dummy                    : String1;
    f1,f2,f3,f4,f5,FV,f6     : Text;
    flg,i, j, kk,nk,ii,dt,Teller: Integer;
    x,n,k,k0,k1,aa,a,d,d1,dum,p0,p1,p2,q0,q1,
    q2,q21,p21,sq,meanq,sqm,sqq,r2,deltak,
    deltan,km,nm,im,tt,r2m,n0: Real;
    r22                      : ARRAY [1..9]  OF Real;
    nq                       : ARRAY [1..9]  OF Real;
    kq                       : ARRAY [1..9]  OF Real;
    p                        : ARRAY [1..200] OF Real;
    q                        : ARRAY [-99..399] OF Real;
    qh                       : ARRAY [-99..399] OF Real;
    qt                       : ARRAY [-99..399] OF Real;
    qtt                      : ARRAY [-99..399] OF Real;
    choice ,dumy             : char;
    answer                   : boolean;
    FN1,FN2,FN,FileName      : String[14];
  procedure presspace;
  procedure input;
  procedure nkvalue;
  procedure iuhtouh;
  procedure uhtoh;
  procedure criteria;
  procedure semisimulation;
  procedure okellymain;

implementation

(*=========================================================*)
    FUNCTION Menu(MenuStr:Str80):Char;
(*=========================================================*)

VAR  Ch              : Char;
     ChPos,Lengte,i: Integer;
     Found           : Boolean;
     Line            : Str80;
```

```
 PROCEDURE PrintMenu(Menu_Heading:Str20;MenuStr:Str80);

VAR
  i       :   integer;

BEGIN
  GotoXY(1,1);ClrEol;LV;
  Write('              *****       ',Menu_Heading,'       *****');
  HV;
  GotoXY(1,3); ClrEol;
  Lengte:=LENGTH(MenuStr);i:=1;
  for  i:= 1 to Lengte DO
  BEGIN
    Ch:=MenuStr[i];
    IF Ch IN ['A'..'Z','0'..'9'] THEN HV ELSE LV;
    Write(Ch);
  END;
  HV; Write(' ? ');
END;
BEGIN
  PrintMenu(Menu_Heading,MenuStr); Read(Kbd,Ch); WriteLn(Ch);
  IF Ch IN ['a'..'z'] THEN Ch:=UpCase(Ch);
  ChPos:=1; Found:=False; Line:=MenuStr;
  REPEAT
    Lengte:=LENGTH(Line);
    Found:=(Ch = Line[1]);
    ChPos:=POS(' ',Line)+1;
    Line:=Copy(Line,ChPos,Lengte);
  UNTIL ((ChPos=1) OR Found);
  IF Found THEN Menu:=Ch ELSE
  BEGIN
    Menu:=' ';
    Bell;WriteLn(' ... is no choice !');Delay(500);
  END;
  ClrScr;PrintMenu(Menu_Heading,MenuStr);WriteLn(Ch);
END; (* Menu *)


PROCEDURE Presspace;
  BEGIN
    GoTOXY(22,22);
    WriteLn('Strike space key when ready...');
    Read(Kbd,dummy);
    ClrScr;
  END;

PROCEDURE Infilename;
{LABEL
    Start;}
BEGIN
{Start:}
 Write('Give   input-filename    ');   Write('(b:12345pn0.dat)  or
(b:12345qn0.dat): ');
 Readln(FileName);
{IF Exist(FileName)=False THEN
 BEGIN
  WriteLn('File does not exist !');
```

```
 END;
IF Exist(FileName)=False THEN GOTO Start;
Clrscr;)
END;

PROCEDURE Outfilename;
(LABEL
    START; )
BEGIN
(Start:)
  Write('Give   output-filename   ');Write('(b:12345pnN.dat)   or
(b:12345qnN.dat):');
  Readln(FN);
( IF Exist(FN)=True THEN
  BEGIN
   WriteLn(FN,' exists !');
   Write('Overwrite ',FN,' ?');LV;Write(' (Y/N)');HV;
   Read(Kbd,Ch);WriteLn;
   IF Ja THEN GOTO Start ;
  END;    ClrScr; )
END;

(*=============================================================*)
    PROCEDURE CopyFile;
(*=============================================================*)
(
CONST
  Arr_Afm               = 20;(* max number in dataset *)
 )
VAR LineTel,LineTelMax : Integer;
 (  LineArr             : ARRAY[1..Arr_Afm] OF String[80];
    Filvar,FV           : Text;
    Filename,FN         : string[14];
 )
BEGIN
 Assign(Filvar,FileName);
 Reset(FilVar);
 Assign(FV,FN);
 ReWrite(FV);

 LineTel:=0;
 REPEAT
  LineTel:=LineTel+1;
  ReadLn(Filvar,LineArr[LineTel]);
  IF Eof(Filvar) THEN LineTelMax:=LineTel;
 UNTIL Eof(Filvar);
 FOR LineTel:=1 TO LineTelMax DO
 WriteLn(FV,LineArr[LineTel]);

 Close(Filvar);
 Close(FV);
END; (* CopyFile *)

(*=============================================================*)
PROCEDURE Input_Menu;
(*=============================================================*)
```

```
BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='INPUT MENU';
  CASE Menu(' Copyedit Existfile Modifyfile ') OF

      'C' :  BEGIN
              Writeln('Copy the net rainfall file and edit');
              Write('Give input-filename ');
              Write('(b:12345pn0.dat) ');
              Readln(FileName);
              Write('Giveoutput-filename');
              Write('(b:12345pnN.dat) ');
              Readln(FN1);
              Exec('\command.com','/c copy '+FileName+' '+FN1);
              Assign(f1,FN1); Reset(f1);
              Exec('b:\NE.COM',FN1); Close(f1);
              Scroll;Clrscr;
              Writeln('Copy the direct runoff file and edit');
              Write('Give input-filename ');
              Write('(b:12345qn0.dat): ');
              Readln(FileName);
              Write('Give output-filename');
              Write('(b:12345qnN.dat):');
              Readln(FN2);
              Exec('\command.com','/c copy '+filename+' '+fn2);
              Assign(f2,FN2); Reset(f2);
              Exec('b:\NE.COM',FN2); Close(f2);
              IF Yes('... are you ready to run OKELLY ?') THEN
              Quit:=True;
             END;

      'M' :  BEGIN
              WriteLn('modify the existing net rainfall file');
              Delay(100);
              Write('Net Rainfall : ');HV;Delay(100);
              Write('Give File Name b:12345pnN.dat ');
              Readln(FN1);
              Exec('b:\NE.COM',FN1);
              WriteLn('modify the existing direct runoff file ');
              Delay(100);
              Write('Direct Runoff : ');HV;Delay(100);
              Write('Give File Name b:12345qnN.dat ');
              Readln(FN2);
              Exec('b:\NE.COM',FN2);
              IF Yes('... are you ready to run OKELLY ?') THEN
              Quit:=True;
             END;
      'E' :  BEGIN
              LV;WriteLn('TestFiles :');
              Write('Net Rainfall : ');HV;Delay(100);
              Write('Give File Name b:12345pn0.dat ');
              Readln(FN1);WriteLn;LV;
              Write('Direct runoff: ');HV;Delay(100);
              Write('Give File Name b:12345qn0.dat ');
              Readln(FN2);WriteLn;
              IF Yes('... are you ready to run OKELLY ?') THEN
```

```
                Quit:=True;
               END;
     ('Q' :   BEGIN
                IF Yes('... are you sure ?') THEN Quit:=True;
               END;          )
  END; (* Case *)
  HV;
 UNTIL Quit=True;
 Scroll;ClrScr;
END; (* Input-Menu *)

PROCEDURE input;
  BEGIN
    Input_Menu;
    Assign(f1,FN1); Reset(f1);
    Assign(f2,FN2); Reset(f2);
    Assign(f3,'b:nkiuh.dat');  ReWrite(f3);
    Assign(f4,'b:nkuh.dat');   Rewrite(f4);
    Assign(f5,'b:hg.dat');     Rewrite(f5);
    Assign(f6,'b:hgb.dat');    Rewrite(f6);
    ClrScr;flg := 1;
    FOR i :=1 to 6 DO WriteLn(f6,'  ');
    WriteLn(f6,'     0                  0.0                          0');
    for i := 1 to 399 do q[i]:=0.0;
    for i := -99 to 399 do qt[i]:=0.0;
    for i := -99 to 399 do qtt[i]:=0.0;
    GotoXY(10,10);
    Begin
      FOR i :=1 to 6 do
      Readln(f1);
    End;
    Begin
      FOR i :=1 to 6 do
      Readln(f2);
    End;ClrScr;
    Write('Give Area in Km(sq):');Readln(aa);
    Write('Give Unit Depth in mm:');Readln(d);
    WriteLn('Input net rainfall(mm/hr) and its time(hr)');
    ReadLn(f1,i,p[1],dt);p[dt]:=p[1];GotoXY(10,4);
    WriteLn('p[',dt:2,']=',p[dt]:4:1);
    Repeat
      BEGIN
        ReadLn(f1,i,p[i*dt],j);GotoXY(10,3+i);
        WriteLn('p[',j:2,']=',p[j]:6:2);
        kk:=i;
      END;
    Until Eof(f1);Close(f1);
    PRESSPACE;
    WriteLn('Input direct run-off(cms) and its time(hr)');
    Repeat
      BEGIN
        ReadLn(f2,i,q[i*dt],j);GotoXY(10,3+i);
        WriteLn('q[',j:2,'] = ',q[j]:7:2);
        j:=i;
      END;
    Until Eof(f2);Close(f2);
  PRESSPACE;
  END;
```

```
PROCEDURE nkvalue;
  BEGIN
    p0 := 0.0;
    p1 := 0.0;
    p2 := 0.0;
    q0 := 0.0;
    q1 := 0.0;
    q2 := 0.0;
    q[0]:=0.0;
    FOR i := 1 TO kk DO
      BEGIN
        p0 := p0+p[i*dt]*aa*1000*dt;
        p1 := p1+p[i*dt]*(i-0.5)*aa*1000*dt*dt;
        p2 := p2+p[i*dt]*(i-0.5)*(i-0.5)*aa*1000*dt*dt*dt
      END;
    FOR i := 1 TO j DO
      BEGIN
        q0 := q0+((q[(i-1)*dt]+q[i*dt])/2)*3600*dt;
        q1 := q1+(q[i*dt]/2)*((i-1/3)+(i+1/3))*3600*dt*dt;
        q2 :=q2+(q[i*dt]/2)*((i-1/3)*(i-1/3)+(i+1/3)*
             (i+1/3))*3600*dt*dt*dt;
      END;
    Writeln('p0=',p0:12:0);
    Writeln('p1=',p1:16:0);
    Writeln('p2=',p2:20:0);
    Writeln('q0=',q0:12:0);
    Writeln('q1=',q1:16:0);
    Writeln('q2=',q2:20:0);
    p21 := p2-p1*p1/p0;
    q21 := q2-q1*q1/q0;
    {k   := (q21-p21)/((q1-p1)*3600);
    n   := (q1-p1)*(q1-p1)/(p0*(q21-p21));}
    x   := (36/49)*(q1+p1)*(q1+p1)/(p0*p0)-(6/7)*
           ((2*q1*(q1+p1)/(p0*p0))-(q2+p2)/p0);
    n   := (12/7)*((q1+p1)/p0)-2*sqrt(x);
    if n <= 0.0  then
    n   := (12/7)*((q1+p1)/p0)+2*sqrt(x);
    k   :=abs(((q1-p1)/p0)-n/2);
    k0  :=k;
    n0  :=n;
   {Writeln(f3,' k = ');}
    Write(' k =  ',k:5:2,'  T =    ',n:5:2);
   {Writeln(f3,k);}
   {Writeln(f3,' n = ');}
   {Writeln(f3,n);}
    PRESSPACE;
  END;
PROCEDURE iuhTOuh;
  BEGIN
    k1  :=k*3600;
    d1  :=d/1000;
    qt[0] :=0.0;
   {Writeln(f3,'j=',j:4);}
   {WriteLn(f3,'qt[0]=',qt[0]:4:2);}
    FOR i := 1 TO j+3 DO
    BEGIN
      x  := i*dt;
      if i*dt < n/2
```

```
      then qt[i*dt] :=(4/(n*n))*(i*dt-k+k*
                       exp(-i*dt/k))*aa*1000*d/3600;
      if x <= n then if x >= n/2
      then qt[i*dt] :=((4/(n*n))*((n/2-k)*
                       exp((-i*dt+n/2)/k)+k*exp((-i*dt)/k))+4/n
                       *(1-exp((-i*dt+n/2)/k))-4/(n*n)*(i*dt-k-
                     (n/2-k)*exp((-i*dt+n/2)/k)))*aa*1000*d/3600;
      if i*dt > n
      then qt[i*dt] := ((4/(n*n)*((n/2-k)*exp((-i*dt+n/2)/k)+
                       k*exp((-i*dt)/k))+4/n *(exp((-i*dt+n)/k)-
                       exp((-i*dt+n/2)/k))-4/(n*n)*((n-k)*exp
                       ((-i*dt+n)/k)-(n/2-k)*
                       exp((-i*dt+n/2)/k))))*aa*1000*d/3600;
    END;
    FOR i :=1 TO j+3 DO
    BEGIN
      qtt[i*dt] := (qt[i*dt]+qt[(i-1)*dt])/2;
    END;
  END;

PROCEDURE uhtoh;
  BEGIN
    FOR i :=1 to j+3 DO
    BEGIN
      dum  := 0.0;
      nk   := 0;
      Repeat
        nk  := nk+dt;
        dum := dum+qtt[(i+1)*dt-nk]*p[nk]*dt/d;
      Until nk = kk*dt;
      qh[i*dt] := dum;
      WriteLn('qh[',i*dt:2,']=',qh[i*dt]:5:1,'
              q[',i*dt:2,']=',q[i*dt]:5:1);
      if flg = 1 then WriteLn(f6,i:6,qh[i*dt]:20:1,i*dt:20);
    END;
  END;

PROCEDURE criteria;
  BEGIN
    sq := 0.0;
    FOR i :=1 TO j+3 DO
    BEGIN
      sq := sq+q[i*dt];
    END;
    meanq :=sq/(j+3);
    sqm := 0.0;
    sqq := 0.0;
    FOR i :=1 TO j+3 DO
    BEGIN
       sqm := sqm+(q[i*dt]-meanq)*(q[i*dt]-meanq)*q[i*dt];
      sqq := sqq+(q[i*dt]-qh[i*dt])*(q[i*dt]-qh[i*dt])*q[i*dt];
    END;
    r2 := (sqm/(j+3)-sqq/(j+3))/(sqm/(j+3));
  END;
PROCEDURE semisimulation;
  BEGIN
    ClrScr;{Scrool;}
    GotoXY(2,2);WriteLn('which deltak AND deltan will be used'
```

```
                    ,'in the semisimulation') ;
GotoXY(4,4);Write('delta-k = ');Readln(deltak);
GotoXY(6,6);Write('delta-n = ');Readln(deltan);
ClrScr;
km :=0.0;
im :=0.0;
nm :=0.0;
FOR ii := 1 TO 9 DO
  BEGIN
    IF ii = 1 THEN
    BEGIN
      k:=k0;
      n:=n0;
    END;
    IF ii = 2 THEN
    BEGIN
      k:=k0+deltak;
      n:=n0;
    END;
    IF ii = 3 THEN
    BEGIN
      k:=k0-deltak;
      n:=n0;
    END;
    IF ii = 4 THEN
    BEGIN
      k:=k0;
      n:=n0+deltan;
    END;
    IF ii = 5 THEN
    BEGIN
      k:=k0;
      n:=n0-deltan;
    END;
    IF ii = 6 THEN
    BEGIN
      k:=k0+deltak;
      n:=n0+deltan;
    END;
    IF ii = 7 THEN
    BEGIN
      k:=k0+deltak;
      n:=n0-deltan;
    END;
    IF ii = 8 THEN
    BEGIN
      k:=k0-deltak;
      n:=n0+deltan;
    END;
    IF ii = 9 THEN
    BEGIN
      k:=k0-deltak;
      n:=n0-deltan;
    END; flg :=2;
    iuhTOuh;
    uhtoh;
    criteria;
    r22[ii] :=r2;
```

```
        GotoXY(20,20);
        WriteLn('ii=',ii:2,'  k=',k:4:2,'  n=',n:4:2,'
                r2=',r22[ii]:6:4);
        Presspace;
        nq[ii]  :=n;
        kq[ii]  :=k;
        IF r22[ii] > r2m  THEN
        BEGIN
          km:=k;
          nm:=n;
          im:=ii;
          r2m:=r22[ii];
        END;
      END;
    FOR ii := 1 to 9 DO
      BEGIN
        WriteLn('ii=',ii:2,'  k=',kq[ii]:4:2,'  n=',nq[ii]:4:2,'
                r2=',r22[ii]:6:4);
      END;
    GoToXY(12,12);
    Writeln('old k value = ',k0);
    GoToXY(12,14);
    Writeln('old T value = ',n0);
    GoToXY(12,16);
    Writeln('new k value = ',km);
    GoToXY(12,18);
    Writeln('new T value = ',nm);
    GotoXY(10,21);Write('Do you want to change k T value?(y/n)');
    Readln(dummy);
    IF dummy ='y' THEN
    BEGIN
      k0:=km;
      n0:=nm;
    END;
    GotoXY(10,22);Write('Do you intend to improve again?(y/n)');
    Readln(dummy);ClrScr;
    IF dummy ='y' THEN
    BEGIN
      semisimulation ;
    END;
  END;
PROCEDURE autosimulation;
  BEGIN
    ClrScr;GotoXY(10,10);
    Writeln('AUTOSIMULATION (Waiting!)');
    deltak:=0.6;
    deltan:=0.6;
    km :=0.0;
    im :=0.0;
    nm :=0.0;
   REPEAT
    Teller:=1;
    FOR ii := 1 TO 9 DO
      BEGIN
        IF ii = 1 THEN
        BEGIN
          k:=k0;
          n:=n0;
```

```
END;
IF ii = 2 THEN
BEGIN
  k:=k0+deltak;
  n:=n0;
END;
IF ii = 3 THEN
BEGIN
  k:=k0-deltak;
  n:=n0;
END;
IF ii = 4 THEN
BEGIN
  k:=k0;
  n:=n0+deltan;
END;
IF ii = 5 THEN
BEGIN
  k:=k0;
  n:=n0-deltan;
END;
IF ii = 6 THEN
BEGIN
  k:=k0+deltak;
  n:=n0+deltan;
END;
IF ii = 7 THEN
BEGIN
  k:=k0+deltak;
  n:=n0-deltan;
END;
IF ii = 8 THEN
BEGIN
  k:=k0-deltak;
  n:=n0+deltan;
END;
IF ii = 9 THEN
BEGIN
  k:=k0-deltak;
  n:=n0-deltan;
END;
{iuh to uh}
BEGIN
 k1  :=k*3600;
 d1  :=d/1000;
 qt[0] :=0.0;
 FOR i := 1 TO j+3 DO
  BEGIN
   if i*dt <= n/2
   then qt[i*dt] := (4/(n*n))*(i*dt-k+k*exp
                    (-i*dt/k))*aa*1000*d/3600;
   if i*dt > n/2 then if i*dt <= n  then
   qt[i*dt]:=((4/(n*n))*((n/2-k)*exp((-i*dt+n/2)/k)+
             k*exp((-i*dt)/k))+4/n *(1-exp((-i*dt+
             n/2)/k))-4/(n*n)*(i*dt-k-n/2-k)*exp((-i
             *dt+n/2)/k)))*aa*1000*d/3600;
   if i*dt > n  then
   qt[i*dt] := ((4/(n*n)*((n/2-k)*exp((-i*dt+n/2)/k)+
```

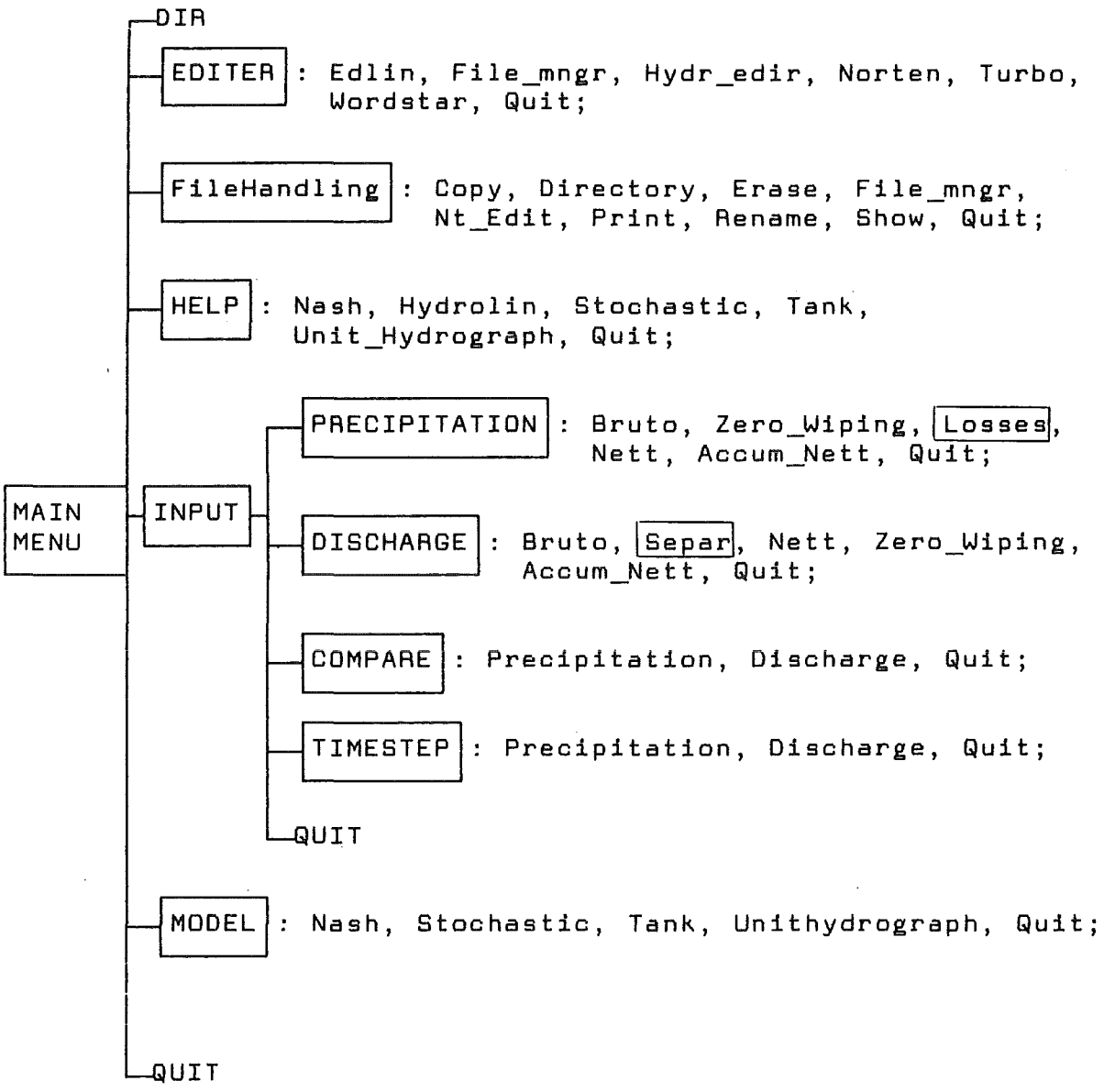
```
                    k*exp((-i*dt)/k))+4/n*(exp((-i*dt+n)/k)-exp
                    ((-i*dt+n/2)/k))-4/(n*n)*((n-k)*exp
                    ((-i*dt+n)/k)-(n/2-k)*exp((-i*dt
                    +n/2)/k))))*aa*1000*d/3600;
       END;
      FOR i :=1 TO j+3 DO
      BEGIN
       qtt[i*dt] := (qt[i*dt]+qt[(i-1)*dt])/2;
      END;
    END;

  { uh to h }
    BEGIN
     FOR i :=1 to j+3 DO
      BEGIN
       dum  := 0.0;
       nk   := 0;
       Repeat
        nk  := nk+dt;
        dum := dum+qtt[(i+1)*dt-nk]*p[nk]*dt/d;
       Until nk = kk*dt;
       qh[i*dt] := dum;
      END;
    END;

 { criteria }
    BEGIN
     sq := 0.0;
     FOR i :=1 TO j+3 DO
      BEGIN
       sq := sq+q[i*dt];
      END;
     meanq :=sq/(j+3);
     sqm := 0.0;
     sqq := 0.0;
     FOR i :=1 TO j+3 DO
      BEGIN
       sqm := sqm+(q[i*dt]-meanq)*(q[i*dt]-meanq);
       sqq := sqq+(q[i*dt]-qh[i*dt])*(q[i*dt]-qh[i*dt]);
      END;
     r2 := (sqm/(j+3)-sqq/(j+3))/(sqm/(j+3));
    END;
    r22[ii] :=r2;
    nq[ii]  :=n;
    kq[ii]  :=k;
    IF r22[ii] > r2m  THEN
    BEGIN
      km:=k;
      nm:=n;
      im:=ii;
      r2m:=r22[ii];
      Teller:=2;
      write('r2= ',r2m:6:4);
      write('k = ',km :8:2);
      write('T = ',nm :8:2);writeln;
    END;
  END;
  IF Teller = 1 THEN
```

```
      BEGIN
        deltak:=deltak-0.2;
        deltan:=deltan-0.2;
        Gotoxy(20,20); write('r2m=',r2m:5:3);
      END;
      IF Teller = 2 THEN
      BEGIN
        k0:=km;
        n0:=nm;
      END;
    UNTIL deltak <= 0.0;
    GotoXY(10,12);Writeln('BEST K VALUE = ',km:5:2);
    GotoXY(10,13);Writeln('BEST T VALUE = ',nm:5:2);
    Writeln('Its Deterministic Coefficient =',r2m:6:4);
    Presspace; flg :=2;
    iuhtouh; uhtoh;
    Presspace;
  END;

(*============================================================*)
   PROCEDURE Simulation_Menu;
(*============================================================*)

BEGIN
ClrScr;
Quit:=False;
 REPEAT
  Menu_Heading:='SIMULATION MENU';
  CASE Menu('Autosimulation Semisimulation Quit') OF

      'A' :  BEGIN
              Autosimulation;
             END;

      'S' :  BEGIN
              Semisimulation;
             END;
      'Q' :  BEGIN
              IF Yes('... are you sure ?') THEN Quit:=True;
             END;
  END; (* Case *)
  HV;
 UNTIL Quit=True;
 Scroll;ClrScr;
END; (* Simulation-Menu *)

procedure okellymain;
BEGIN
  input;
  nkvalue;
  iuhTOuh;
  uhtoh;
  criteria;
  GoToXY(20,20);Write('r2(deterministic coefficient)=',r2:6:4);
  PRESSPACE;
  r2m := r2;
  simulation_menu;GotoXY(1,10);
  IF  Yes('Do  you  want  to  store  n&k  value  and  UH  in file
```

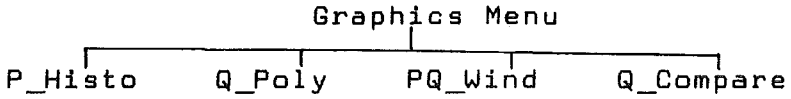
```
b:nkuh.dat ? ')
  THEN
   BEGIN
    Writeln(f3,'BEST T VALUE = ',nm:5:2);
    Writeln(f3,'BEST K VALUE = ',km:5:2);
    Writeln(f4,'BEST T VALUE = ',nm:5:2);
    Writeln(f4,'BEST K VALUE = ',km:5:2);
    n :=nm;
    k :=km;
    qt[0] :=0.0;
    WriteLn(f3,'IUH[  0]=',qt[0]:8:2);
    FOR i := 1 TO j+3 DO
    BEGIN
       if i*dt <= n/2  then
       qt[i*dt] :=(4/(n*n))*(i*dt-k+k*exp(-i*dt/k))*
                  aa*1000*d/3600;
       if i*dt > n/2 then if i*dt <= n  then
       qt[i*dt] :=((4/(n*n))*((n/2-k)*exp((-i*dt+n/2)/k)+
                  k*exp((-i*dt)/k))+4/n *(1-exp
                  ((-i*dt+n/2)/k))-4/(n*n)*(i*dt-k-(n/2-k)*exp
                  ((-i*dt+n/2)/k)))*aa*1000*d/3600;
       if i*dt > n  then
       qt[i*dt] := ((4/(n*n)*((n/2-k)*exp((-i*dt+n/2)/k)+
                   k*exp((-i*dt)/k))+4/n*(exp((-i*dt+n)/k)-
                   exp((-i*dt+n/2)/k))-4/(n*n)*((n-k)*
                   exp((-i*dt+n)/k)-(n/2-k)*exp((-i*dt
                   +n/2)/k))))*aa*1000*d/3600;
       WriteLn(f3,'IUH[',i*dt:3,']=',qt[i*dt]:8:2);
    END;
    WriteLn(f4,'UH[  0]=',qtt[0]:8:2);
    FOR i :=1 TO j+3 DO
    BEGIN
      qtt[i*dt] := (qt[i*dt]+qt[(i-1)*dt])/2;
      WriteLn(f4,'UH[',i*dt:3,']=',qtt[i*dt]:8:2);
    END;

    FOR i :=1 to 6 DO WriteLn(f5,'    ');
    WriteLn(f5,'     0                  0.0                        0');

    FOR i :=1 to j+3 DO
    BEGIN
      dum  := 0.0;
      nk   := 0;
      Repeat
        nk  := nk+dt;
        dum := dum+qtt[(i+1)*dt-nk]*p[nk]*dt/d;
      Until nk = kk*dt;
      qh[i*dt] := dum;
      WriteLn(f5,i:10,qh[i*dt]:20:3,i*dt:20);
    END;
    Close(f3);Close(f4);Close(f5);Close(f6);
   END;
END;
END.
```

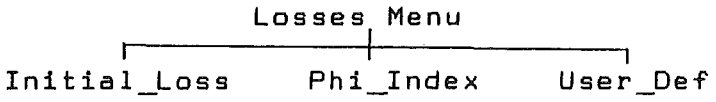
APPENDIX B : Table, Menu, Flow Chart, Input Data and Graphics

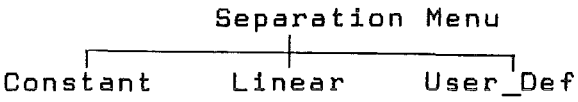
## B.1 HYDROLIN Main Menu Table



## B.2 Graphics Menu, Losses Menu and Separation Menu

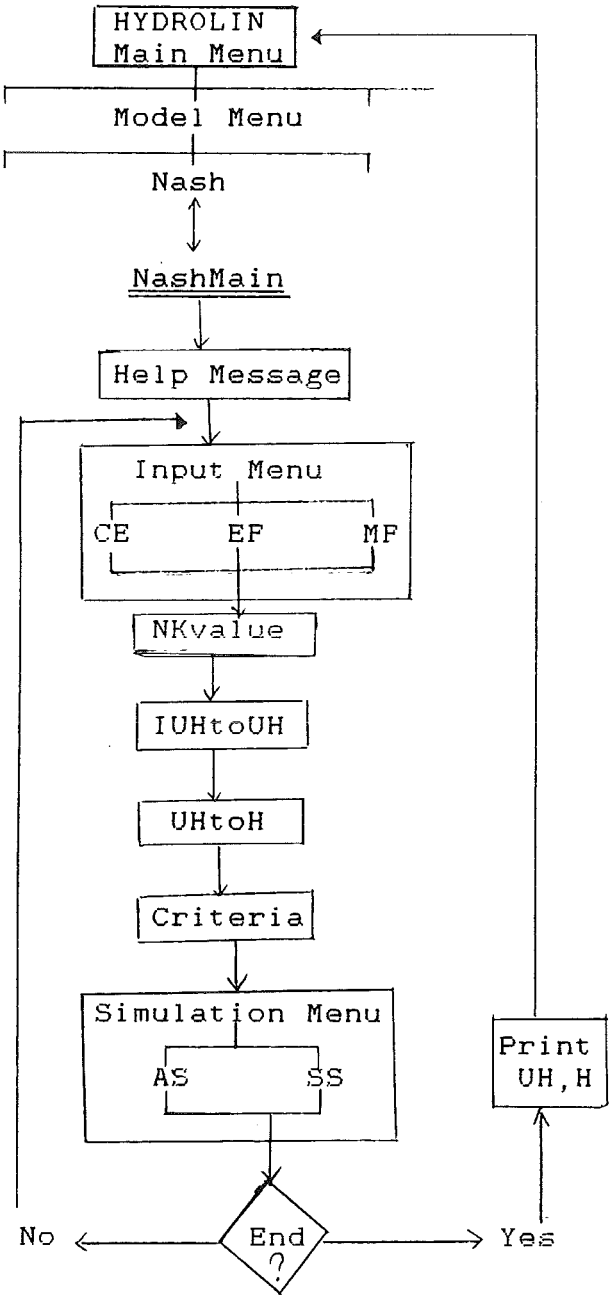


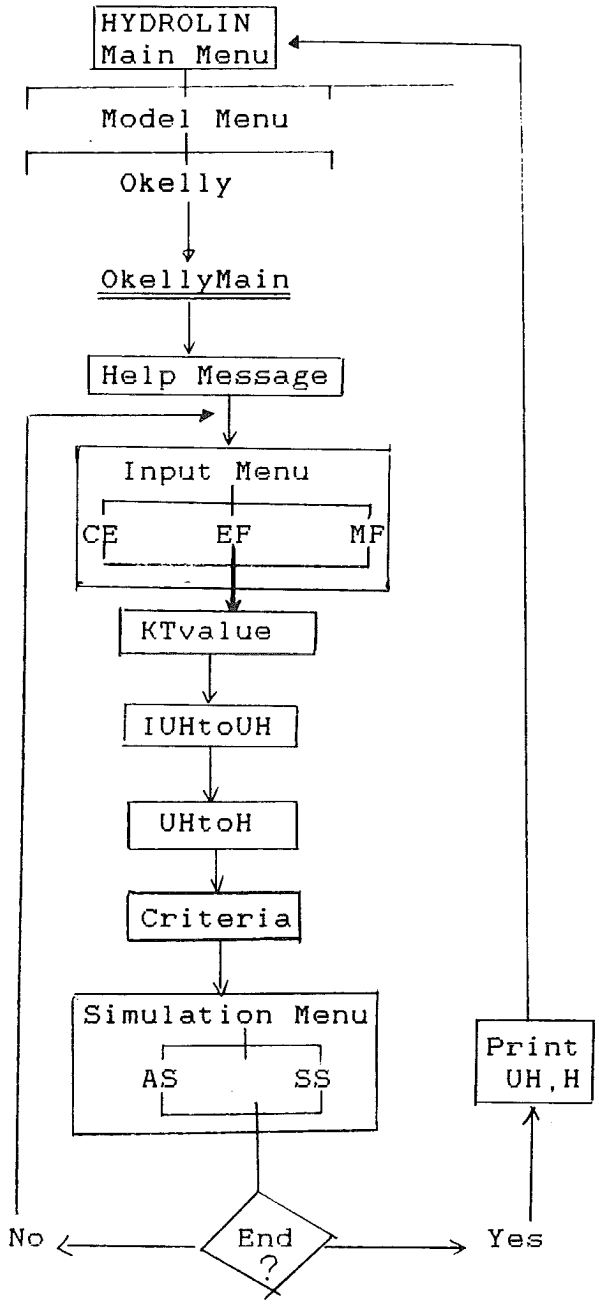
## B.3 Nash Model Flow Chart

- Help Message
- Input Menu
  - . CopyEdit
  - . ExistFile
  - . ModifyFile
- NKvalue
- IUHtoUH
- UHtoH
- Criteria
- Simulation Menu
  - . Auto-Simulation
  - . Semi-Simulation
- NashMain



B.4 O'kelly Model Flow Chart

- Help Message
- Input Menu
  . CopyEdit
  . ExistFile
  . ModifyFile
- KTvalue
- IUHtoUH
- UHtoH
- Criteria
- Simulation Menu
  . Auto-Simulation
  . Semi-Simulation
- OkellyMain



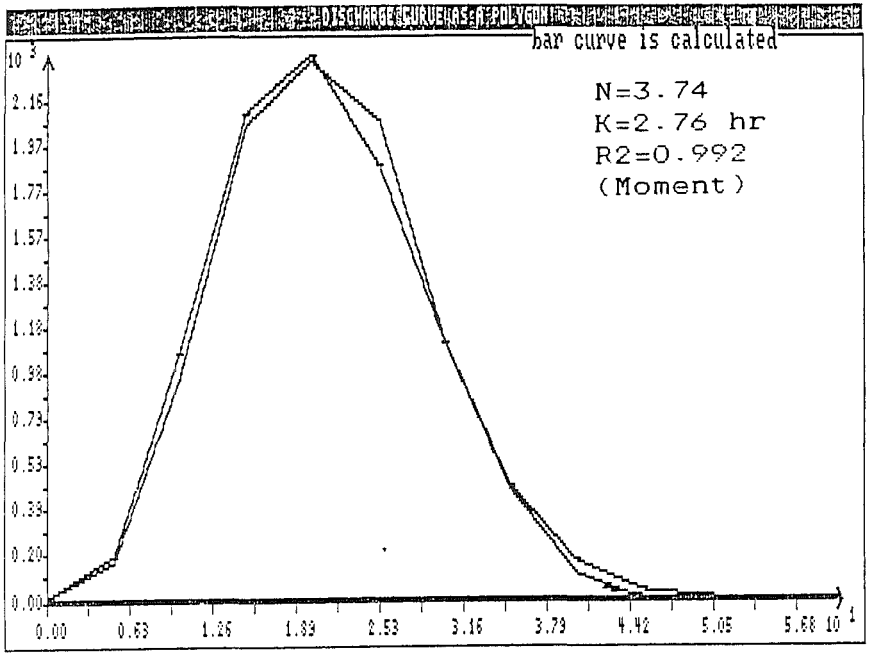
## B.5 Test Input Data, Result and Graphics

### (1) Data Set 1

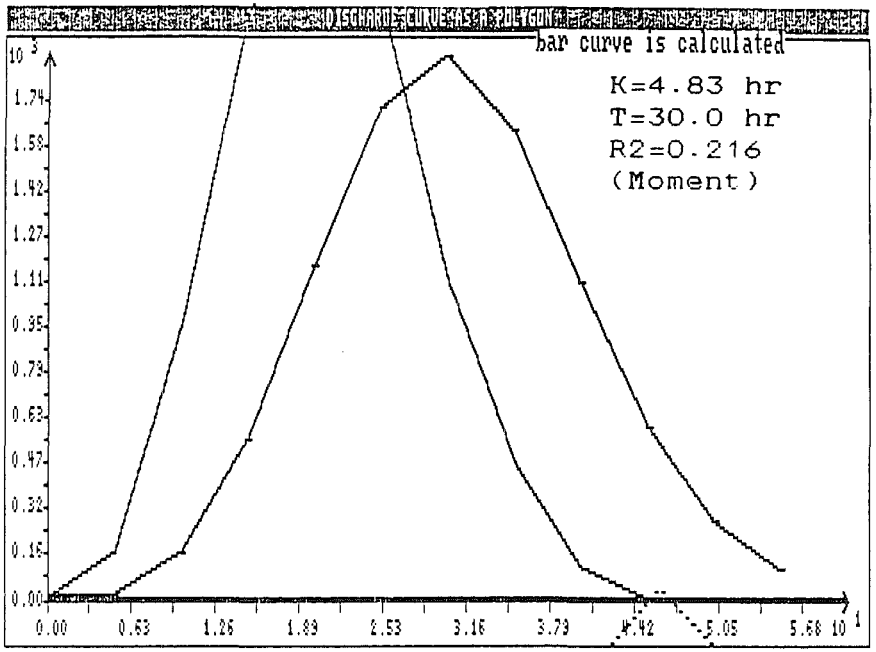
| TIME hour | Measured Effective Rainfall mm/hour | Measured Direct Runoff cms | NASH | | OKELLY | |
|---|---|---|---|---|---|---|
| | | | Moment (Calculated) cms | Simulation (Calculated) cms | Moment (Calculated) cms | Simulation (Calculated) cms |
| 0 | | 0.00 | 0.0 | 0.0 | 0.0 | 0.0 |
| 5 | 6.850 | 165.371 | 191.1 | 175.3 | 22.2 | 83.513 |
| 10 | 24.030 | 953.216 | 1064.3 | 998.5 | 168.2 | 559.946 |
| 15 | 15.120 | 2046.755 | 2106.1 | 2047.2 | 556.2 | 1482.042 |
| 20 | 10.670 | 2320.267 | 2361.7 | 2368.1 | 1160.0 | 2242.072 |
| 25 | 3.030 | 2076.843 | 1882.5 | 1926.8 | 1714.5 | 2227.178 |
| 30 | | 1114.164 | 1115.6 | 1168.3 | 1897.6 | 1526.841 |
| 35 | | 482.826 | 494.4 | 529.0 | 1628.8 | 740.066 |
| 40 | | 122.652 | 173.4 | 187.7 | 1104.6 | 241.051 |
| 45 | | 0.000 | 52.0 | 56.3 | 603.3 | 37.835 |
| 50 | | | 14.0 | 15.1 | 271.0 | 0.725 |
| 55 | | | 3.5 | 3.7 | 103.4 | 0.002 |
| 60 | | | 0.8 | 0.9 | 36.7 | 0.000 |

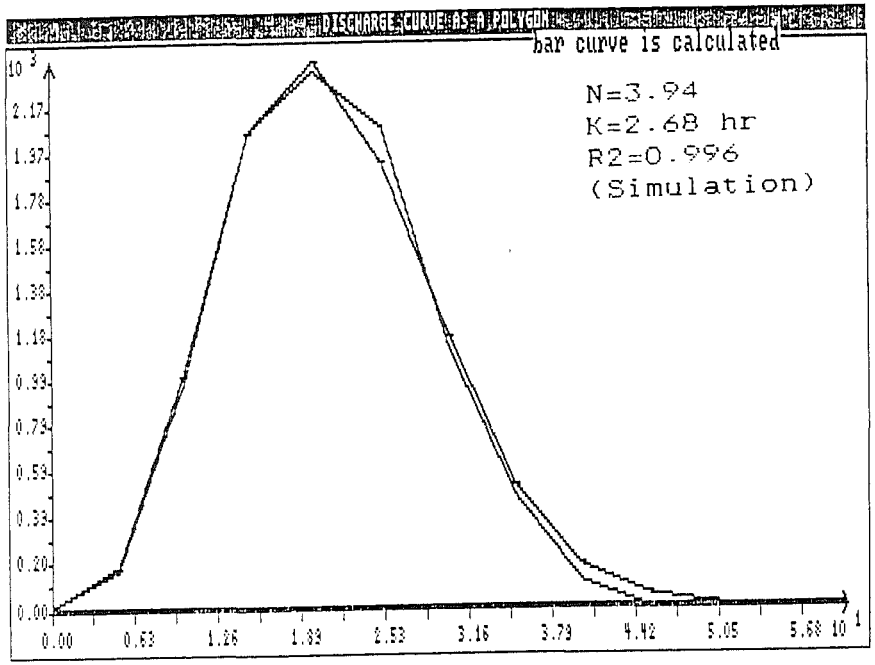
(1) Data set 1 (A=560 km²)

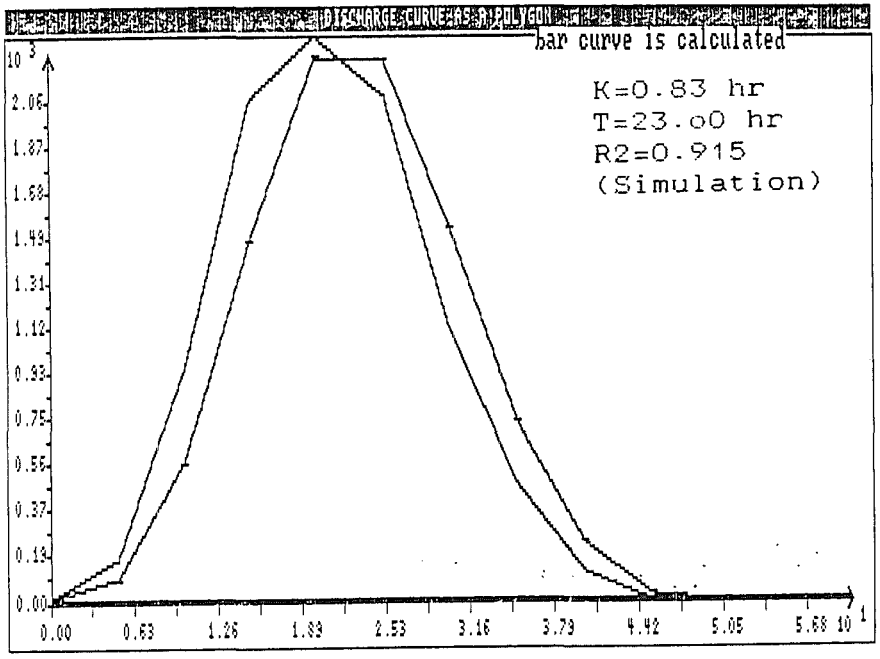
NASH




O'KELLY



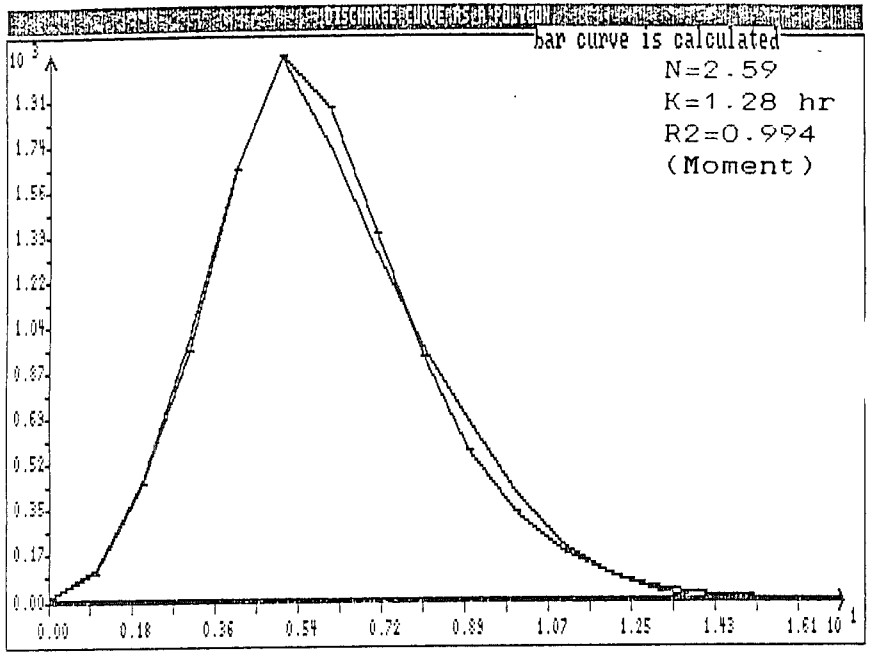



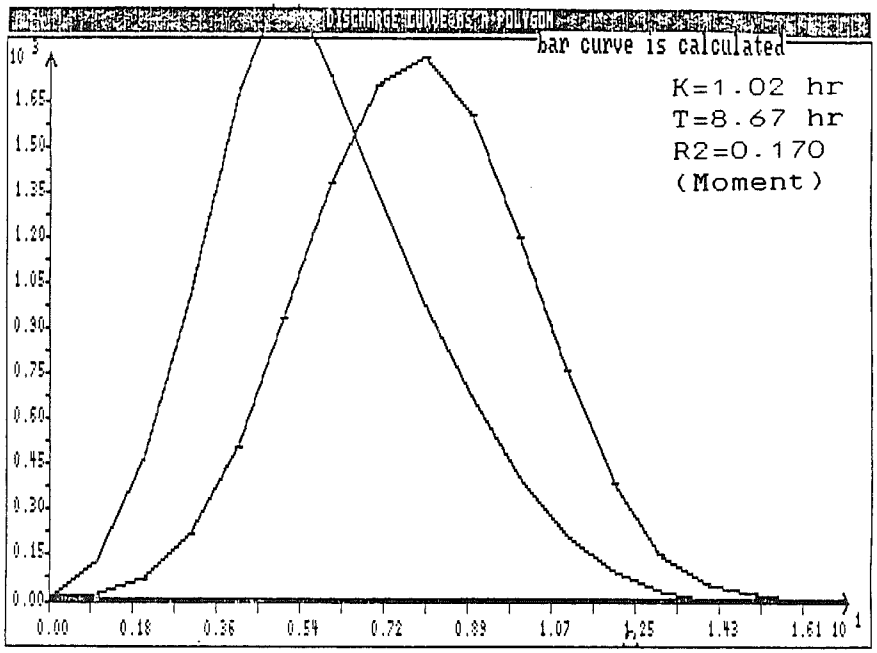
| TIME (hour) | Measured Effective Rainfall (mm/hour) | Measured Direct Runoff (cms) | NASH | | OKELLY | |
|---|---|---|---|---|---|---|
| | | | Moment (Calculated) (cms) | Simulation (Calculated) (cms) | Moment (Calculated) (cms) | Simulation (Calculated) (cms) |
| 0 | | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 1 | 10.0 | 120.0 | 102.7 | 107.1 | 11.6 | 50.757 |
| 2 | 20.0 | 460.0 | 449.7 | 455.4 | 70.8 | 266.008 |
| 3 | 20.0 | 1000.0 | 959.6 | 945.8 | 219.3 | 721.027 |
| 4 | 40.0 | 1660.0 | 1643.7 | 1608.9 | 504.5 | 1438.792 |
| 5 | | 2070.0 | 2083.3 | 2004.4 | 926.2 | 2125.520 |
| 6 | | 1730.0 | 1879.2 | 1773.6 | 1371.3 | 2392.816 |
| 7 | | 1330.0 | 1395.1 | 1329.3 | 1696.8 | 2014.570 |
| 8 | | 970.0 | 927.6 | 909.5 | 1796.5 | 1205.185 |
| 9 | | 680.0 | 575.1 | 587.6 | 1598.9 | 483.525 |
| 10 | | 410.0 | 339.8 | 365.0 | 1192.4 | 93.319 |
| 11 | | 220.0 | 193.9 | 220.4 | 756.7 | 0.725 |
| 12 | | 100.0 | 107.7 | 130.2 | 389.8 | 0.004 |
| 13 | | 40.0 | 58.6 | 75.6 | 159.0 | 0.000 |
| 14 | | 0.0 | 31.4 | 43.3 | 59.4 | 0.000 |
| 15 | | | 16.6 | 24.5 | 22.2 | 0.000 |
| 16 | | | 8.6 | 13.8 | 8.3 | 0.000 |
| 17 | | | 4.5 | 7.7 | 3.1 | 0.000 |

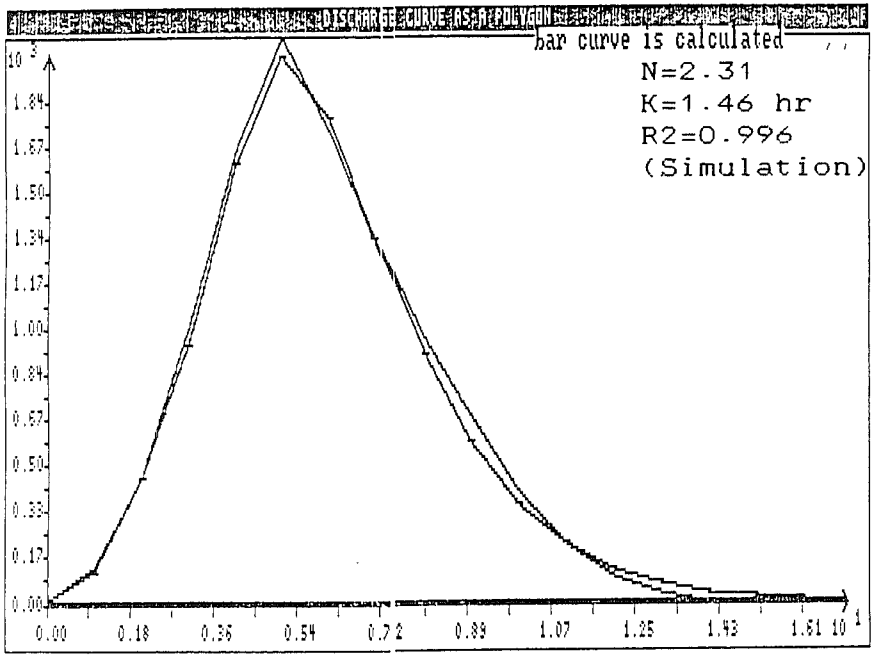
(2) Data set 2 (A=400 km²)

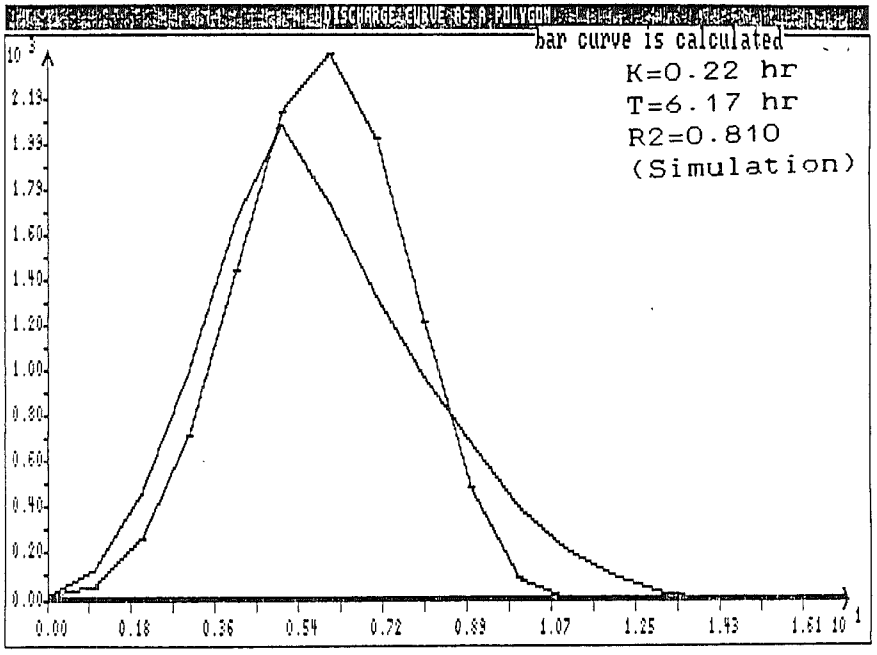
NASH




O'KELLY



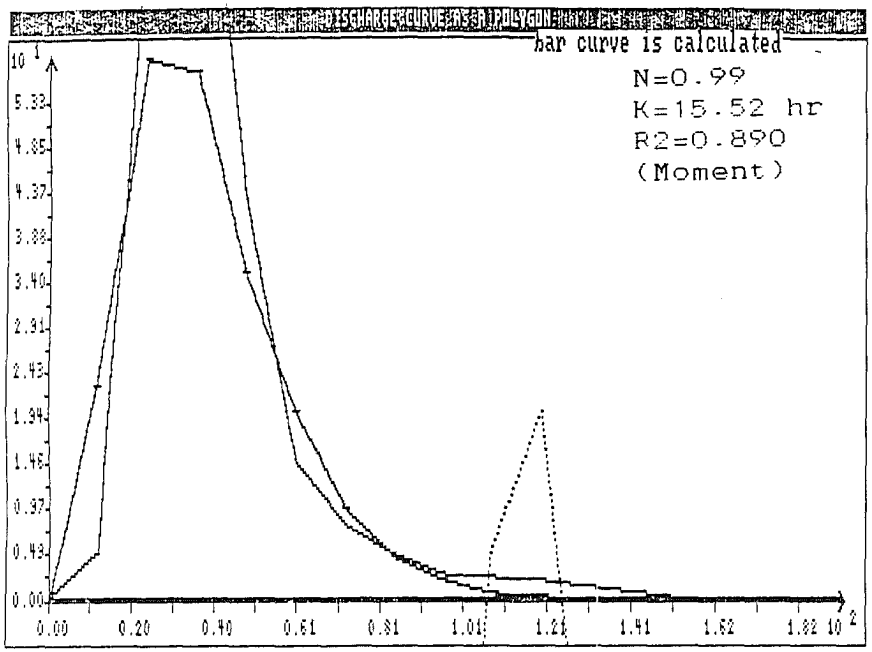



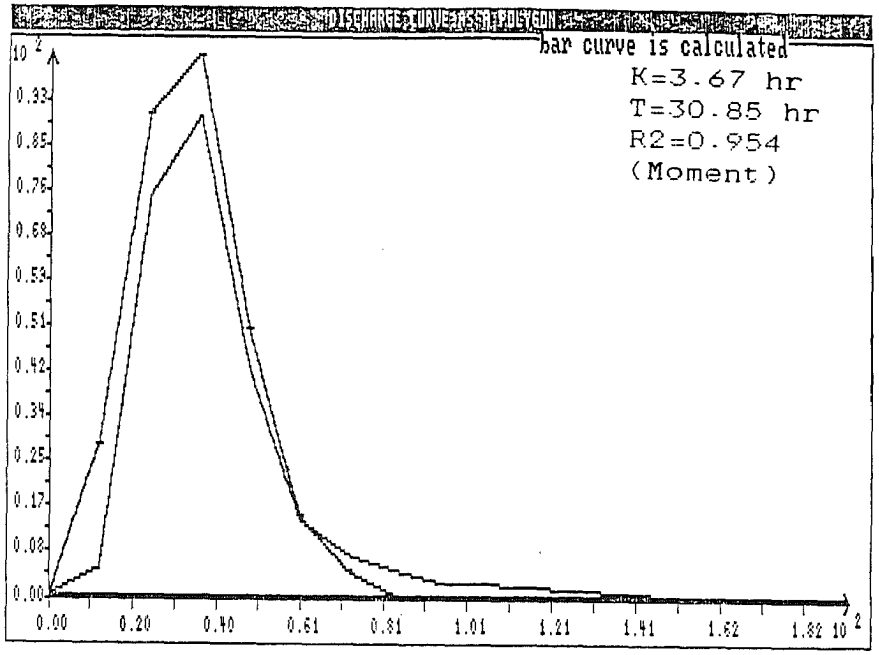
| TIME (hour) | Measured Effective Rainfall (mm/hour) | Measured Direct Runoff (cms) | NASH | | OKELLY | |
|---|---|---|---|---|---|---|
| | | | Moment (Calculated) (cms) | Simulation (Calculated) (cms) | Moment (Calculated) (cms) | Simulation (Calculated) (cms) |
| 0 | | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 12 | 3.027 | 5.1 | 22.9 | 29.3 | 28.7 | 24.560 |
| 24 | 3.026 | 75.0 | 58.2 | 77.9 | 90.8 | 84.022 |
| 36 | 0.377 | 90.0 | 56.8 | 80.0 | 101.9 | 102.555 |
| 48 | 0.377 | 43.0 | 35.1 | 49.2 | 50.6 | 55.847 |
| 60 | | 15.0 | 20.3 | 25.6 | 15.4 | 17.965 |
| 72 | | 8.0 | 9.9 | 10.7 | 5.0 | 5.886 |
| 84 | | 5.0 | 4.7 | 4.0 | 0.4 | 0.704 |
| 96 | | 3.0 | 2.2 | 1.4 | 0.0 | 0.029 |
| 108 | | 2.5 | 1.0 | 0.4 | 0.0 | 0.001 |
| 120 | | 2.1 | 0.5 | 0.1 | 0.0 | 0.000 |
| 132 | | 1.7 | 0.2 | 0.0 | 0.0 | 0.000 |
| 144 | | 0.8 | 0.1 | 0.0 | 0.0 | 0.000 |
| 156 | | 0.0 | 0.0 | 0.0 | 0.0 | 0.000 |
| 168 | | | 0.0 | 0.0 | 0.0 | 0.000 |
| 180 | | | 0.0 | 0.0 | 0.0 | 0.000 |
| 192 | | | 0.0 | 0.0 | 0.0 | 0.000 |

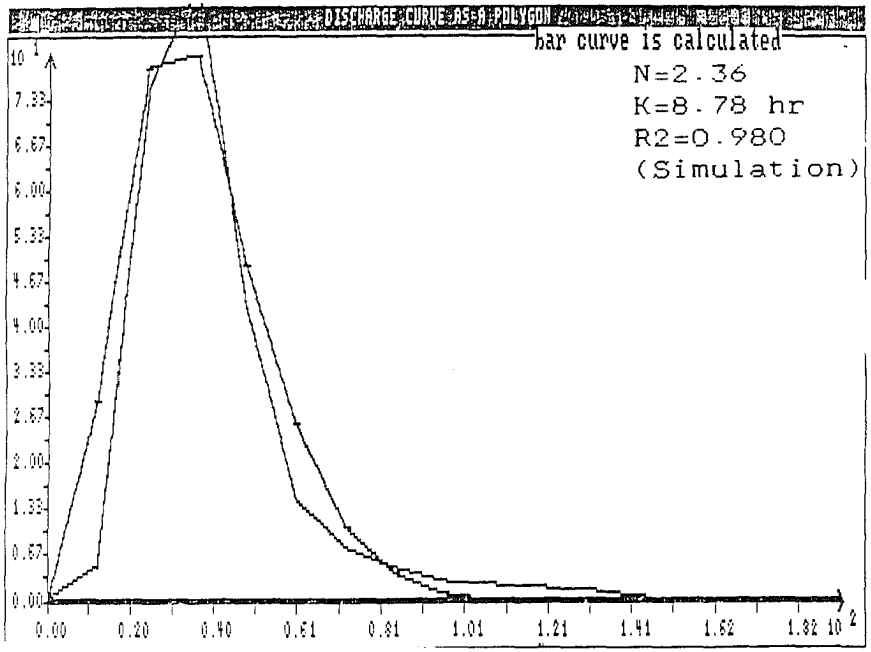
(3) Data set 3 (A=160 km²)

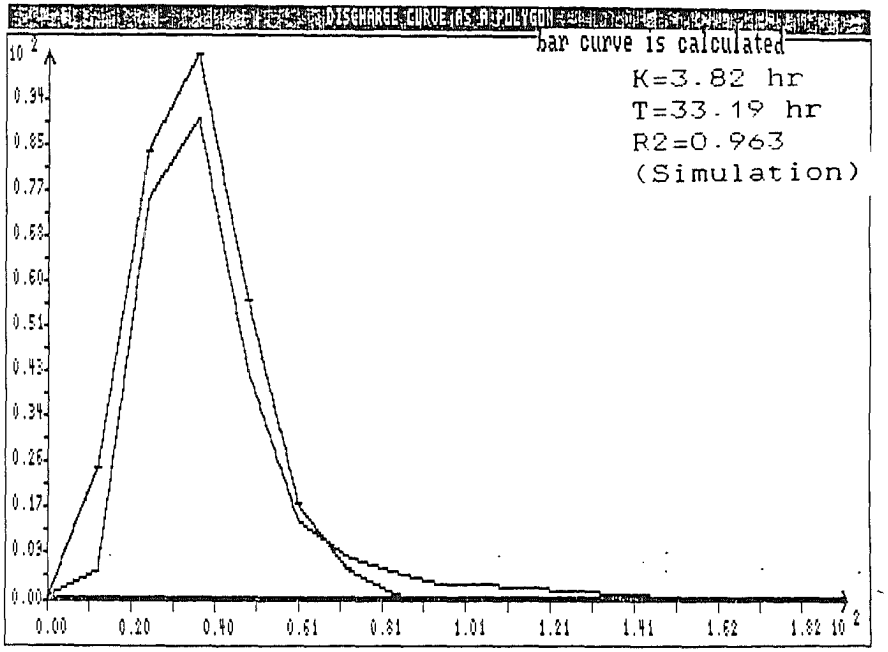
NASH




O'KELLY



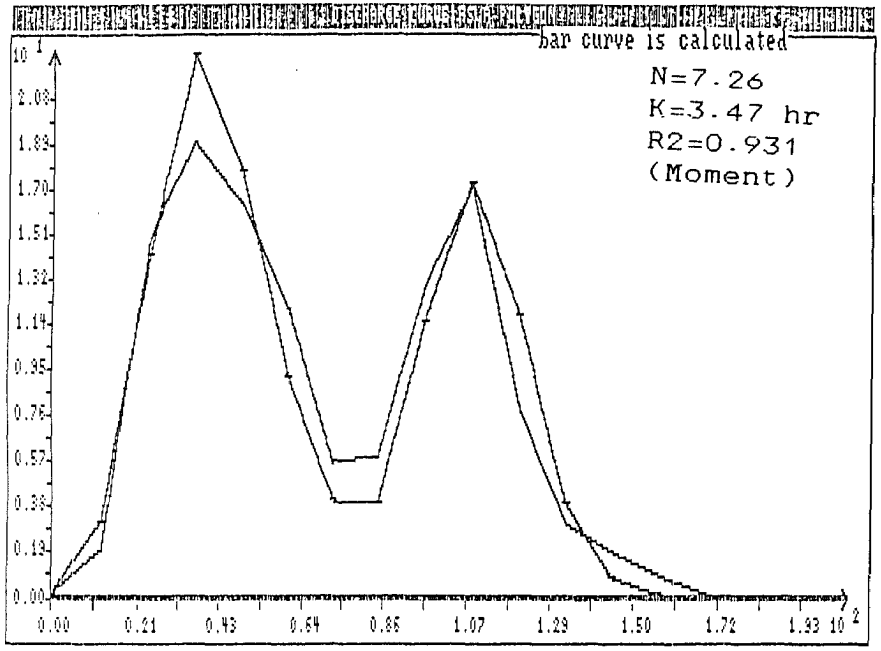



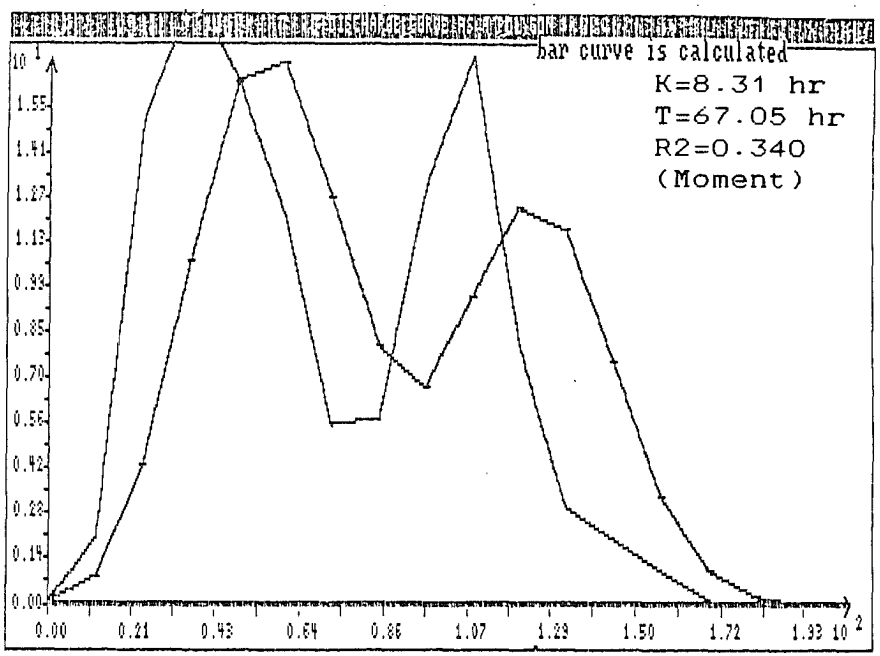
| TIME (hour) | Measured Effective Rainfall (mm/hour) | Measured Direct Runoff (cms) | NASH | | OKELLY | |
|---|---|---|---|---|---|---|
| | | | Moment (Calculated) (cms) | Simulation (Calculated) (cms) | Moment (Calculated) (cms) | Simulation (Calculated) (cms) |
| 0 | | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 12 | 0.680 | 2.0 | 3.2 | 3.4 | 0.9 | 3.749 |
| 24 | 0.680 | 15.0 | 14.3 | 14.4 | 4.4 | 14.611 |
| 36 | 0.145 | 19.0 | 22.7 | 22.2 | 10.7 | 21.639 |
| 48 | 0.145 | 16.4 | 17.8 | 17.0 | 16.3 | 16.792 |
| 60 | 0.011 | 12.0 | 9.2 | 8.7 | 16.9 | 9.277 |
| 72 | 0.011 | 5.6 | 4.1 | 3.9 | 12.7 | 4.539 |
| 84 | 0.525 | 5.8 | 3.9 | 3.9 | 8.0 | 4.652 |
| 96 | 0.525 | 13.0 | 11.5 | 11.5 | 6.8 | 11.914 |
| 108 | 0.015 | 17.0 | 17.2 | 16.7 | 9.5 | 16.369 |
| 120 | 0.015 | 8.0 | 11.7 | 11.1 | 12.4 | 10.939 |
| 132 | | 3.1 | 3.9 | 3.6 | 11.7 | 4.161 |
| 144 | | 2.0 | 0.9 | 0.8 | 7.6 | 1.377 |
| 156 | | 1.0 | 0.2 | 0.2 | 3.4 | 0.418 |
| 168 | | 0.2 | 0.0 | 0.0 | 1.0 | 0.121 |
| 180 | | 0.0 | 0.0 | 0.0 | 0.3 | 0.035 |
| 192 | | | 0.0 | 0.0 | 0.1 | 0.010 |
| 204 | | | 0.0 | 0.0 | 0.0 | 0.003 |

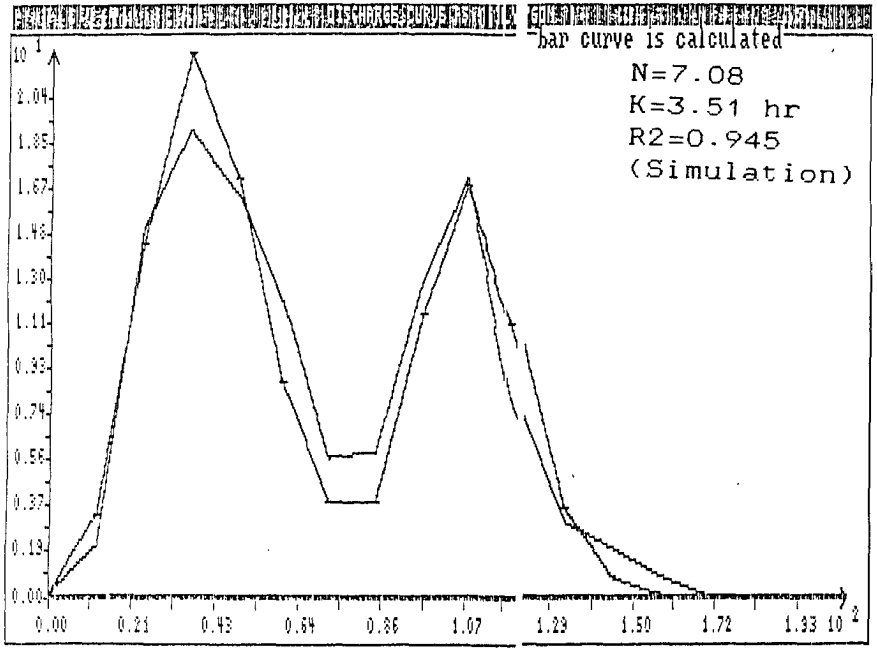
(4) Data set 4 (A=160 km²)

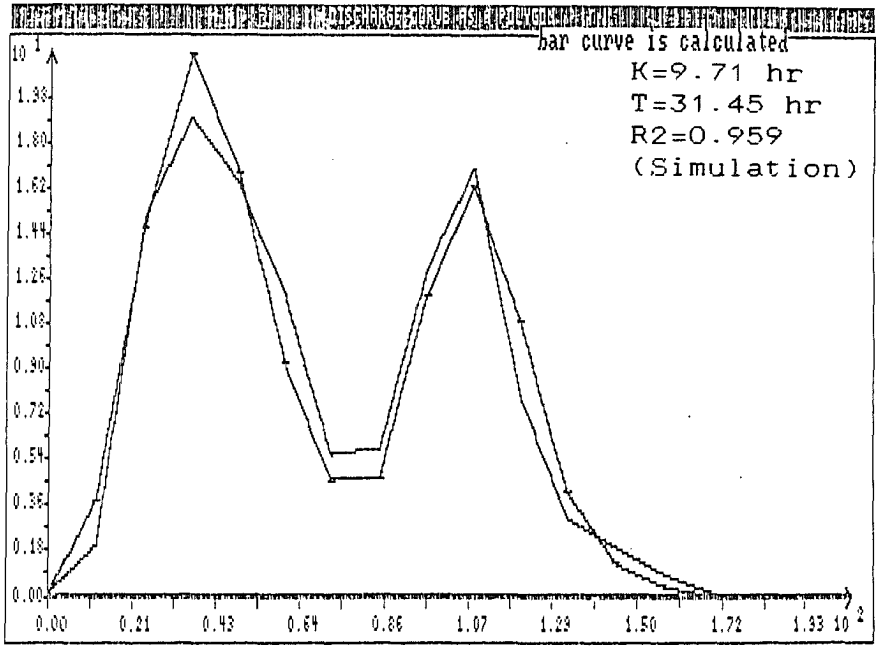
NASH




O'KELLY



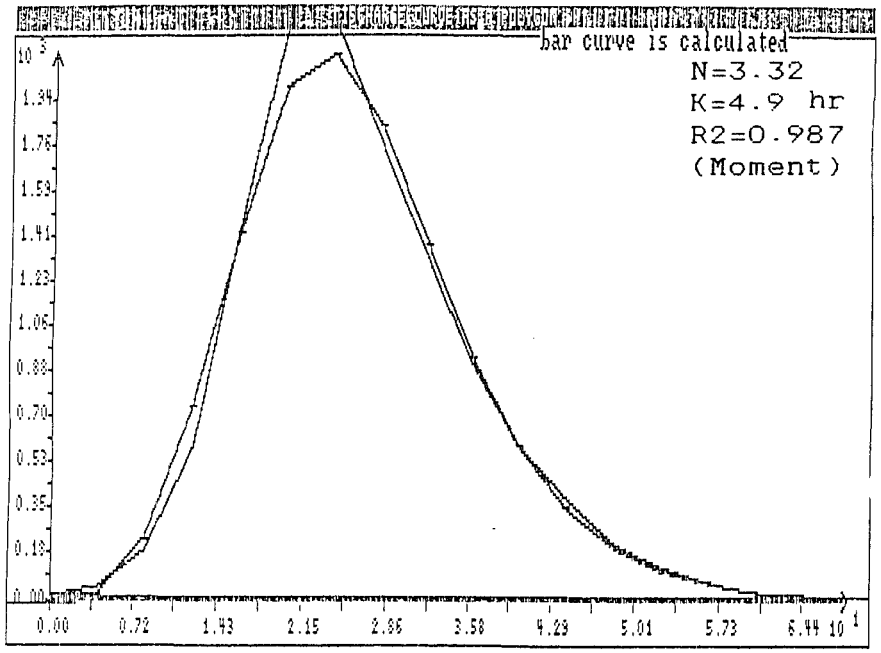



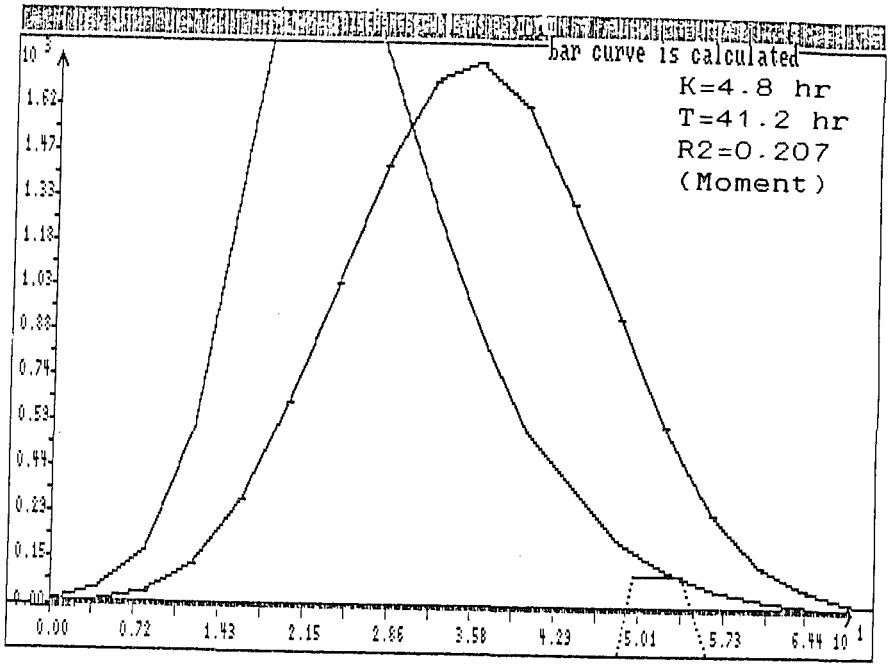
| TIME (hour) | Measured Effective Rainfall (mm/hour) | Measured Direct Runoff (cms) | NASH | | OKELLY | |
|---|---|---|---|---|---|---|
| | | | Moment (Calculated) (cms) | Simulation (Calculated) (cms) | Moment (Calculated) (cms) | Simulation (Calculated) (cms) |
| 0 | | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 | 0.762 | 48.0 | 22.1 | 21.5 | 2.9 | 12.867 |
| 8 | 5.588 | 175.9 | 229.7 | 226.9 | 33.5 | 144.664 |
| 12 | 5.080 | 575.6 | 738.8 | 750.7 | 135.2 | 556.738 |
| 16 | 6.350 | 1438.9 | 1424.7 | 1469.4 | 342.2 | 1318.639 |
| 20 | 1.524 | 2238.3 | 1983.5 | 2060.1 | 658.1 | 2165.149 |
| 24 | | 2238.3 | 2113.4 | 2197.0 | 1045.7 | 2622.871 |
| 28 | | 1758.6 | 1829.7 | 1880.7 | 1426.7 | 2397.107 |
| 32 | | 1311.0 | 1373.5 | 1378.5 | 1695.6 | 1648.851 |
| 36 | | 911.3 | 936.2 | 908.3 | 1768.7 | 848.637 |
| 40 | | 591.6 | 596.1 | 555.1 | 1623.7 | 319.125 |
| 44 | | 399.7 | 361.0 | 320.8 | 1315.8 | 93.105 |
| 48 | | 223.8 | 210.5 | 177.7 | 940.9 | 24.588 |
| 52 | | 127.9 | 119.0 | 95.2 | 586.8 | 6.493 |
| 56 | | 64.0 | 65.7 | 49.7 | 313.9 | 1.715 |
| 60 | | 32.0 | 35.6 | 25.3 | 145.6 | 0.453 |
| 64 | | 16.0 | 18.9 | 12.7 | 63.7 | 0.120 |
| 68 | | 0.0 | 9.9 | 6.3 | 27.7 | 0.032 |

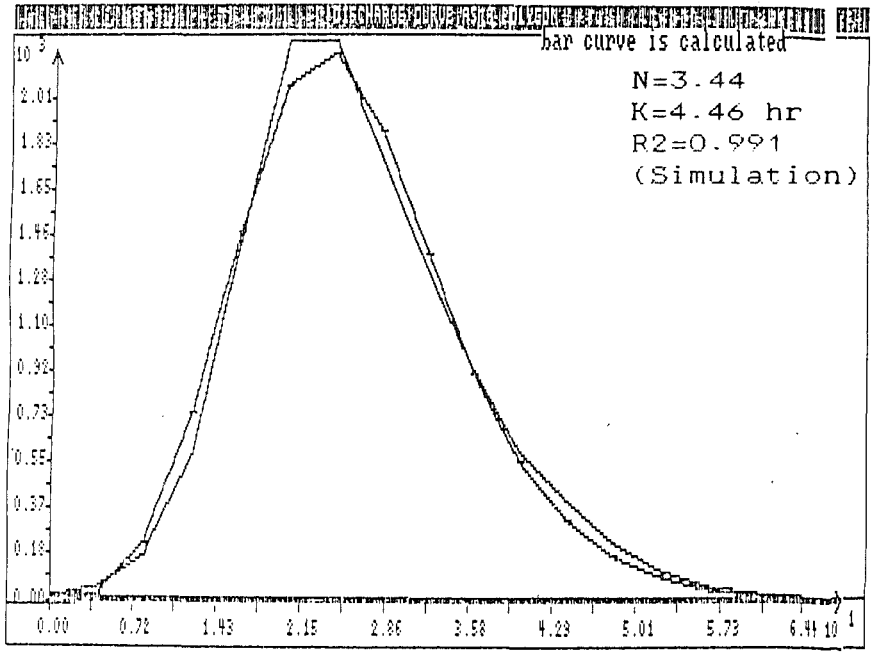
(5) Data set 5 (A=2266 km²)

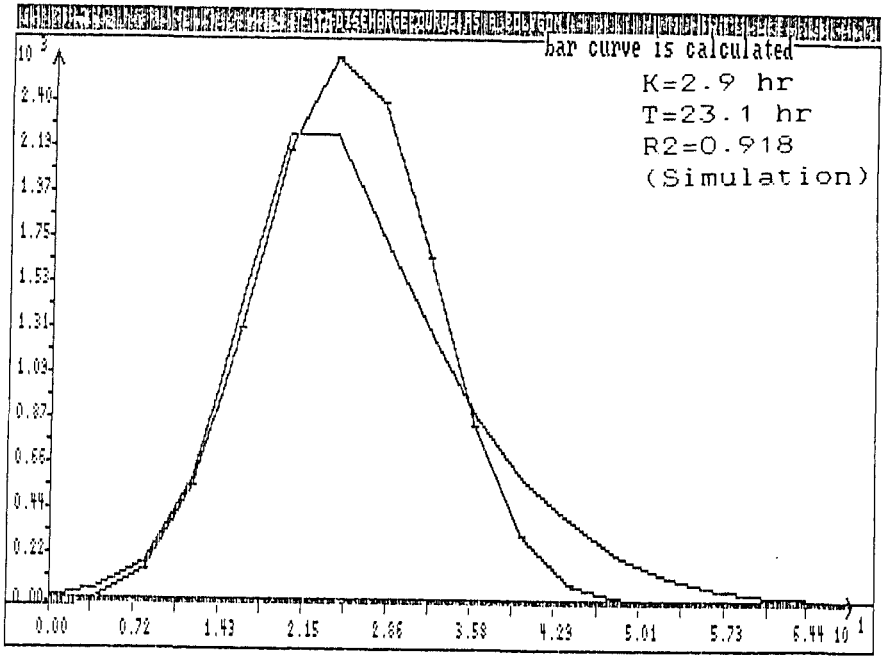
NASH




O'KELLY



B.6 Figure of Auto- and Semi-Simulation