

# **OPEN CHANNEL HYDRAULICS COMPUTER DESIGN SOFTWARE**

**By**

**PERCY MAKAFUI GATI (BSc. Hons)**

**KNUST**

Thesis submitted to

The Department of Agricultural Engineering

Kwame Nkrumah University of Science and Technology

In Partial Fulfillment of the Requirement for Degree of

**MASTER OF SCIENCE**

**IN**

**SOIL AND WATER ENGINEERING**

**COLLEGE OF ENGINEERING**

**© MAY, 2013.**

## DECLARATION

I hereby declare that this submission is my own work towards the M.Sc. and that, to the best of my knowledge, it has no material previously published by another researcher nor material which has been accepted for the award of any other degree of the University, except where due acknowledgement has been properly accorded in the write up.

Percy Makafui Gati (PG4858910) .....  
(Student Name and ID) Signature Date

Certified by:

Prof. S.K. Agodzo .....  
(Principal Supervisor) Signature Date

Dr. E. Ofori .....  
(Co-Supervisor) Signature Date

Prof. E. Mensah .....  
(Head of Department) Signature Date

## **ACKNOWLEDGEMENT**

I am thankful to the Almighty God for his guidance and protection throughout the duration of my study. I am grateful to Prof. S. K. Agodzo who was highly affable and passionate and took the pain in supervising, and also making resources available to aid me conclude this work. I extend my appreciation to all lecturers and mates in the Department of Agricultural Engineering for their assistance in various ways. I am highly indebted to Reverend Lambert Ntibrey who supported both in prayers and designing of the software, not forgetting Mr. Nicholas Beckley and the Prayer Force of Global Evangelical Church, Akatsi who always assisted in prayers.



## **DEDICATION**

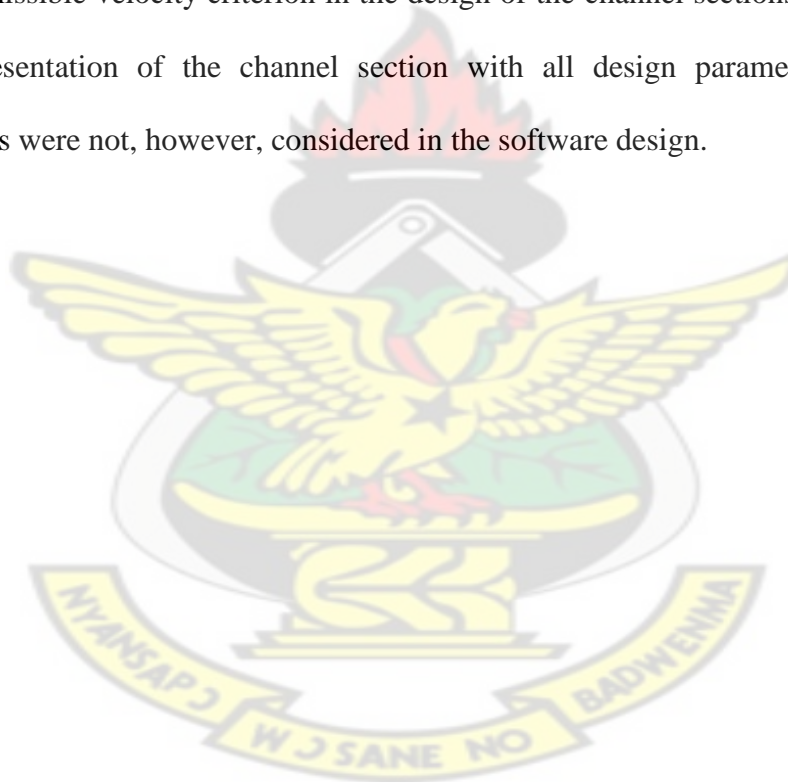
I dedicate this work to God Almighty, the creator of heaven and earth, my beloved mother, the Prayer Force of Global Evangelical Church, Akatsi and to all my loving friends.

# KNUST



## ABSTRACT

Open channel hydraulic computations can be tedious manually considering the channel geometries involved. This work addresses the development of a conceptual model using computer MATLAB software for the design of most commonly used channel sections, with the intention of saving time and cost for the design engineer. Using the Manning's formula, rectangular, triangular or V-shaped, parabolic, U-shaped and circular sections were considered for lined and unlined, hydraulically efficient channel sections. The software determines the state of the flow (sub-critical, critical and super-critical) and applies the maximum permissible velocity criterion in the design of the channel sections. It displays the graphical representation of the channel section with all design parameters. Compound channel sections were not, however, considered in the software design.



## TABLE OF CONTENTS

No.	Title	Page
	DECLARATION	i
	DEDICATION	ii
	ACKNOWLEDGEMENT	iii
	ABSTRACT	iv
	TABLE OF CONTENTS	v
	LIST OF FIGURES	vii
	LIST OF TABLES	viii
	CHAPTER ONE: INTRODUCTION	1
1.1	Background	1
1.2	Justification of the Research	3
1.3	Objectives of Study	4
	CHAPTER TWO: LITERATURE REVIEW	5
2.1	Introduction	5
2.2	Unlined channel	5
2.2.1	Design of Erodible channel	5
2.2.2	Design using Regime's Approach	6
2.3	Tractive Force Method	7
2.3.1	Tractive Force Ratio	8
2.4	Maximum Permissible Velocity	10
2.5	Design Lined channel /Non Erodible Channel	11
2.6	Free Board	12
2.7	Manning and Chezy's Equation	13
2.8	Best Hydraulic Section/Economic Section	14
2.9	Flow Classification	16
2.9.1	Steady and Unsteady Flow	16
2.9.2	Uniform Flow	16
2.9.3	Non-Uniform Flow	16
2.9.3a	Steady Gradually-Varied Flow	16
2.9.3b	Steady Rapid Varied Flow	17

2.10	Velocity Distribution in Open Channel	17
2.11	Fundamental Equations of Flow	18
2.11.1	Continuity Equation	18
2.11.2	Energy Equation	19
2.11.3	Momentum Equation	21
2.11.4	Specific Energy	23
2.11.5	Critical Depth	23
2.12	Froude's Number	24
CHAPTER THREE: MATERIALS AND METHODS		26
3.1	Research Methods	26
3.1.1	Conceptual Model	26
3.2	Characteristics of Open Channels	28
3.3	Tractive Force Theory	29
3.4	The Flow Chart	30
CHAPTER FOUR: RESULTS AND DISCUSSION		36
4.1	Results	36
4.2	Discussion	43
CHAPTER FIVE: CONCLUSION AND RECOMMENDATION		44
5.1	Conclusion	44
5.2	Recommendation	44
	References	45
	Appendix	49

## LIST OF FIGURES

2.1	Shear Stress Distribution in Trapezoidal Channel	8
2.2	State of Forces on the walls of a Trapezoidal Channel Sections	9
2.3	Fluid Flow in and out Channel	18
2.4	Flow of Fluid in the Channel	19
2.5	Flow through a Pipe	21
2.6	Diagram for Resistance of Fluid	22
3.2	The Flow Chart for Input Mode for the Programming	34
4.1	MATLAB Graphical Interface of Channel Flow System	37
4.2	Interface with a Channel Type Section	38
4.3	Interface Ready for Inputting of Data	38
4.4a	Interface with Directive on the User Guide Screen	39
4.4b	Interface with content of User Guide Screen Displayed	40
4.5	Interface for Lined and Economic Trapezoidal Channel	40
4.6	Interface for Lined and Uneconomic Triangular Channel	41
4.7	Interface with Diagnostic Message	41
4.8	Interface with Manning's Roughness Coefficient Value to be selected	42



## LIST OF TABLES

Tables		Page
Table 2.1	Maximum Permissible Velocity Material	11
Table 2.3	Open Channel Geometric Relationship for various Cross Sections	15

KNUST



## CHAPTER ONE

### INTRODUCTION

#### 1.1 BACKGROUND

An open channel is a physical system in which water flows with a free surface at atmospheric pressure. A channel can be classified as either natural or artificial according to its origin. Natural channels include all watercourses of varying sizes from tiny hillside rivulets, streams, small and large rivers to tidal estuaries that exist naturally on the earth. Subsurface streams carrying water with a free surface are also treated as natural open channels.

The cross sections of natural channels are irregular and hence hydraulic properties may vary from section to section, and reach to reach. A comprehensive study of the behavior of flow in natural channels (the mobile boundaries) requires knowledge of other fields, such as hydrology, geomorphology and sediment transportation. Artificial channels are those constructed or developed by human effort such as gutters, drains, ditches, floodways, tunnels, log chutes, navigation channels, power canals and troughs, spillways including model channels that are built in the laboratory for experimental investigation studies. Long distance canals have been constructed to achieve the inter-basin transfer of water at National and International levels. The artificial channel is known by different names, such as canal, chute, culvert, drop, flumes, open - flow tunnel and aqueduct. However, these names are used rather loosely and can be defined only in a general manner. The canal is usually a long and mild-sloped channel built in the ground, which may be lined or unlined with stone masonry, concrete, cement, wood or luminous materials etc.

The chute is a channel having steep slopes. The culvert, flowing partly full, is a covered channel of comparatively short length provided for draining water across roadways and

through railway embankments. The drop is similar to chute, but the change in elevation is effected within a short distance. The flume is a channel of wood, metal, fiber reinforced plastic, concrete, or masonry, usually supported on or above the surface of the ground to carry water across a depression. The open -flow tunnel, is a comparatively long covered channel used to carry water through a hill or any obstruction on the ground. Normally these artificial canals are with rigid boundaries. For example a spillway having variable width and a canal curved alignment. A channel built with constant cross section and constant bottom slope and fixed alignment is named as prismatic channel. Otherwise, the channel is non-prismatic.

Natural sections are in general very irregular, usually varying from an approximate parabola to approximate trapezoidal shapes and for streams subject to frequent floods, the channel may consist of a main channel section carrying normal discharges and one or more side channel sections for accommodating overflows. These are called compound channels. Artificial channels are usually designed with sections of regular geometrical shapes. Tables give the geometric properties for the cases of rectangular, trapezoidal, triangular, circular, parabolic channels. In addition, the details of round bottomed triangular and round bottom rectangular are also given. The primary factors in open channel flow analysis are the location of the free surface which is unknown beforehand. The free surface rises and falls in response to perturbation to the flow (Chanson, 2004).

The use of open channel improvement is not limited to agricultural lands. Many residential homes use subsurface drainage systems, irrigation, water supply scheme, similar to those used in agriculture to prevent water damage to foundation and basements. Golf courses make extensive use of surface and subsurface drains. Houses, streets and buildings in urban areas

depend heavily on surface and subsurface drainage system for protection. These generally are areas with combination of plastic/metals gutters and concrete pipes and channels (Brown and Ward, 1997).

## **1.2 JUSTIFICATION OF THE RESEARCH**

Manual procedures and calculators have been used for sometime in solving mathematical problems which are quiet laborious and time consuming .The use of computers has made it easy to solve many scientific problems with less time consumed. This is used for a lot of experimental analyses and in decision making of mechanization of farm operations. Computers are also used to develop models and to support open channel design of drainage, irrigation systems and water supply scheme.

MATLAB is one of the latest programs used in designing window based software. MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. Furthermore, it is a modern programming language environment which has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make it an excellent tool for teaching and research. It is an interactive system whose basic data element is an array that does not require dimensioning. Due to this it has powerful built-in routines that enable a very wide variety of computations and easy to use graphics commands that make the visualization of results immediately available (Houcque, 2005).

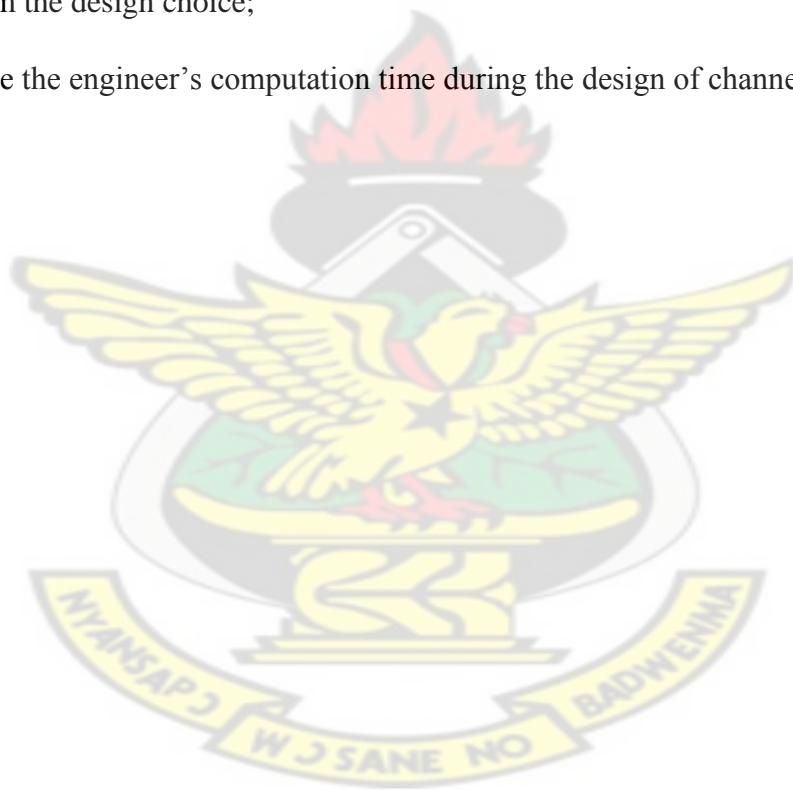
The entire world is shifting from traditional style of workmanship to technological work. More companies are now being restructured with advance technology. Software has now become the business and improved means of engineering ideology in the information technology world.

### 1.3 OBJECTIVES OF STUDY

The main objective of the study was to design a Windows based software for open channel hydraulic systems using MATLAB.

The specific objectives were:

- to compute flow parameters based on Manning's equation for lined economic, lined uneconomic, and unlined channel sections;
- to provide early indication of resulting flow status (sub-critical, critical, super-critical) to inform the design choice;
- to reduce the engineer's computation time during the design of channel sections.



## **CHAPTER TWO**

### **LITERATURE REVIEW**

#### **2.1 INTRODUCTION**

The design of open channel hydraulics involves the selection of a channel alignment, shape, size, bed slope and whether the channel should be lined in order to minimize seepage and /or to prevent erosion of channel side and bottom.

A lined channel offers less resistance to flow than unlined channel. The size of a lined channel required to convey a specified flow rate at a selected slope is small as compared to unlined channel. In most cases, lined channels are more economical than an unlined channel. There are no steps/rules readily available for optimum channel parameters outright. Every site has its own feature which demands a specific consideration (Chaudhry, 2008).

#### **2.2 UNLINED CHANNEL**

##### **2.2.1 DESIGN OF ERODIBLE CHANNEL**

The flow in erodible channel is influenced by many physical factors. The real design of this channel is quite difficult due to complexity and uncertainty of physical factors and fixed conditions. The stability of erodible channel, which governs the design, is dependent on channel type. There is a prevailing uniform flow in erodible channel if the channel section is stable (Das, 2008)



### 2.2.2 DESIGN USING REGIME APPROACH

A channel in which neither silting nor scouring takes place is called Regime Channel / Stable Channel. If a channel is in a stable state, the flow is such that silting and scouring are not considered. The fundamental of designing such an ideal channel is that whatever silt has entered the channel at its canal head is always kept in suspension and not allowed to settle anywhere along its course. More so, velocity of water does not produce local silt by erosion of channel beds or sides. According to Kennedy (1895) data collected on stable channel presented the following non-silting and non-scouring velocity

$$V_c = 0.55y^{0.64} \quad (2.1)$$

where  $V_c$  is the non-silting and non-scouring velocity in m/s and  $y$  = is the depth of flow in m.

Kennedy (1895), further stated that actual velocity  $V$  is given as;

$$M_r = \frac{V}{V_c} = \text{Critical Velocity Ratio.}$$

$$V = M_r V_c$$

$$V = M_r \times 0.55y^{0.64} \quad (2.2)$$

He found that  $M_r > 1.15$  for coarse sediment and  $M_r > 1$  for sediment finer than in his canal,  $V$  is the actual velocity in the channel which is given by Ganguillet – Kutter equation,

$$V = \left[ \frac{23 + \frac{0.0015}{S_b} + \frac{1}{n_k}}{1 + (23 + \frac{0.0015}{S_b}) \frac{n_k}{\sqrt{R}}} \right] \sqrt{R} S_b \quad (2.3)$$

Where  $n_k$  = kutter's coefficient

$S_b$  = bed slope

$R$  = hydraulic radius

Kennedy (1895) gave a judgment that, due to generation of eddies in the channel bed which rise to the channel surface, flowing water gets the silt supporting power. These eddies are generated due to friction of flowing water with Channel Surface Vertical component of such

eddies causing the sediment to be in suspension to avoid silting. The Kennedy theory is quite useful in open channel hydraulic but has been criticized for its limitations of simplifying assumptions used and the rigorous trial and procedure in computation. There are several complexities about the use of precise method of the design of channel.

However, few of these methods are available. These are:

- Maximum permissible velocity
- Regime approach
- Tractive force method

### 2.3 TRACTIVE FORCE METHOD

This is the force developed in flowing water of a channel in the direction of flow on the channel bed. This is also called shear or Drag force (duBoys, 1879).

Brahms (1754) stated that, the principle of balancing the tractive force with channel resistance force in uniform flow is equal to the component of gravity force acting on the body of water parallel to the channel bottom. If the wetted perimeter  $P$ , the length of reach  $L$ , the specific weight of water and the bed slope  $S$ , the mean value of tractive force is per unit area and is given as:

$$\text{Unit tractive force, } T_o = \frac{wALS}{PL} = wRS \quad (2.4)$$

$$T_o = wRS \quad (2.5)$$

For wide rectangular open unlined channel, the unit tractive force is not uniformly distributed along the wetted perimeter as compared to a trapezoidal channel where there is an even distribution of the unit tractive force acting on its bed slope and sides of channel as shown.



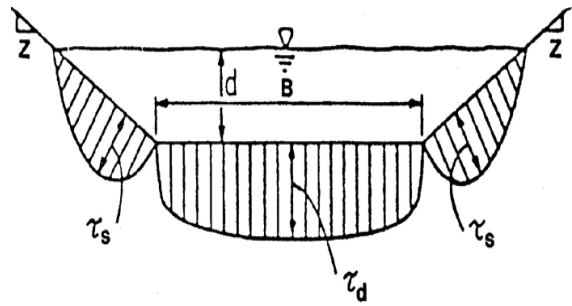


Figure 2.1 Shear Stress Distributions in a Trapezoidal Channel

Maximum of tractive force at the bottom  $\tau_d = 0.97wyds_b$  and on sides  $\tau_s = 0.76wds_b$ .

However,

$$T_s = \frac{WAS}{P} \quad (2.6)$$

Where

$P$ =wetted perimeter,  $S_b$ = bottom slope,  $W$ = specific weight of water

$T_s$  = average tractive force per unit area. ,  $R$ =hydraulic radius  $d$ =depth of flow

In a wide open channel,  $R=d$  ,  $T_s = wdS$

### 2.3.1 TRACTIVE FORCE RATIO

According to Das and Saika (2009), the slope side section of a trapezoidal channel where soil particle or silt is resting whilst water is flowing, two forces act; one is tractive force and the other is gravity.

Component ' $Ws\sin\Phi$ ' tends to cause the particle to roll down the slope as the water moves.

Let

$a$  = effective area of the slope,

$T_s$  = unit tractive force on the side of the channel,

$W_s$  = submerged weight of the particle

$\Phi$  = is the angle of side slope

The resultant of these forces is perpendicular to each other.

$$\text{Resultant} = \sqrt{W_s^2 \sin^2 \Phi + T_s^2 a^2} \quad (2.5)$$

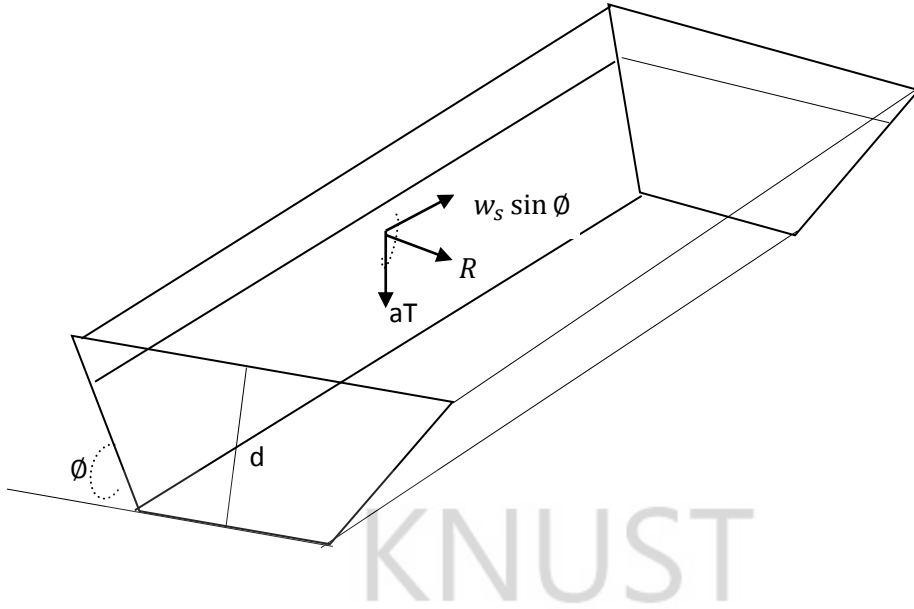


Figure 2.2. State of Forces on the walls a Trapezoidal Channel Section.

It implies that, if  $R$  is large enough, particles will move, by the laws of frictional motion; if motion is intercepted then, the resistance of motion is equal to the force tending to cause the motion. The resistance to motion of the particle is equal to the product of  $W_s \cos \Phi$  and coefficient of friction.

$$W_s \cos \Phi \cdot \tan \theta = \sqrt{W_s^2 \sin^2 \Phi + T_s^2 a^2} \quad (2.7)$$

$$T_s = \frac{W_s}{a} \cos \Phi \cdot \tan \theta \sqrt{1 - \frac{\tan^2 \Phi}{\tan^2 \theta}} \quad (2.8)$$

Similarly, when motion of a particle at the bed slope is impending owing to tractive force, then  $\Phi = 0$ .

$$W_s \tan \theta = a T_l \quad (2.9)$$

$$T_l = \frac{W_s}{a} \tan \theta$$

$$\text{For a unit tractive force, } T_l = \frac{\tan \theta}{a} \quad (2.10)$$

$$K = \frac{T_s}{T_l} = \cos \Phi \sqrt{1 - \frac{\sin^2 \theta}{\sin^2 \Phi}} \quad (2.11)$$

Hence the ratio of  $T_s$  to  $T_l$ , is called the tractive ratio and it is for design and is symbolized as  $K$ .

$$K = \sqrt{1 - \frac{\sin^2 \theta}{\sin^2 \Phi}} \quad (2.12)$$

## 2.4 MAXIMUM PERMISSIBLE VELOCITIES

Fortier and Scobey (1926) stated that, the permissible or non silting velocity is the lowest velocity of flow that does not allow sedimentation and growth of aquatic plant and moss.

According to Chaudhry (2008), the mean velocity at/or below which the channel bottom and sides are not eroded is the maximum permissible velocity. This velocity depends primarily upon the type of soil and the size of particles even though it has been recognized that it should depend upon the flow depth and whether the channel is straight or not. This is because, for the same value of mean velocity, the flow velocity at the channel bottom is higher for low depths than those at large depths. Similarly, a curved alignment induces secondary currents. These produce higher flow velocities near the channel sides, which may cause erosion.

If average cross sectional velocity in the channel does not exceed the maximum permissible velocity, then it becomes uncertain and the value cannot be determined exactly. According to Fortier and Scobey (1926) permissible velocity for clean water varies from 0.45 m/s for fine sand to 1.52m/s for cobbles and shingles and transporting colloidal silt varies from 0.752 m/s to 1.68 m/s.

**Table 2.1 Maximum Permissible Velocity of Materials**

Material	V (m/s)
Fine sand	0.6
Coarse sand	1.0
Fine gravel	2.0
<b>Earth</b>	
Sandy silt	0.6
Silt clay	1.0
Clay	2.0
<b>Grass-lined earth</b>	
Bermuda grass	1.2
Sandy silt	2.0
Silt clay	2.0
<b>Kentucky Blue grass</b>	
Sandy silt	2.0
Silt clay	2.0
Poor rock (usually sedimentary)	3.0
Soft sand stone	2.0
Soft shale	1.1
Good rock (usually igneous or hard metamorphic)	6.0

Source: U.S. Army Corps of Engineers (1991)

## 2.5 DESIGN OF LINED CHANNEL/NON-ERODIBLE CHANNEL

The design of lined channel deals with the relationship between amount of silt or sediment transport by the flowing water without deposition, discharge material forming the bed shape and the side of the channel. The lining is done to prevent the sides and the bottom of the channel from erosion due to the drag/hear stress caused by the flow or moving water. Channel lining could be rigid or flexible. The rigid channels can be considered to have only one degree of freedom; for a given geometry, the only change that may take place is the depth

of flow which may vary with space and depending upon the nature of the flow, Subramanya, (1986). According to Akan (2006), the rigid lining materials include cast-in-place, concrete, plastic concrete, stone, masonry, soil cement, grouted riprap, among others. These resist the drag force and the permissible velocity of the moving water and in turn provide a much higher conveyance capacity. Rigid channel can be susceptible to failure of structural instability caused by freeze-thaw, swelling and excessive soil pressures. Flexible lining can be put into permanent and temporal linings. Permanent linings are the rock riprap, rubble, wire enclosed riprap and gravel. Temporal linings are used for protection against erosion before vegetation is established. This includes straw with net, curled wood mat, jute net, synthetic mat and fiber glass roving. They are less susceptible to structural failure because they conform to changes in the shape of the channel and this allows infiltration which provide opportunity for local flora and fauna. Flexible linings can only sustain limited magnitudes of erosive forces.

## **2.6 FREE BOARD**

In the design of a canal/channel, it is essential to have a free board .Free board of a canal is the vertical distance between full supply levels to the top of the lining. It is essential to prevent overtopping of bank when wave is produced due to wind. Similarly, in designing irrigation canals, sometimes width-depth ratio and side slope values are required depending on the type of soil and lining material. Chow (1959), recommends that free board varies from 5% to 30% of the depth which is mostly used in design.

## 2.7 MANNING AND CHEZY'S EQUATIONS

The average velocity of a uniform flow can be estimated by a number of semi-empirical equations that have the general form;

$$V = CR^x S^y \quad (2.13)$$

Where  $C$  = a resistance coefficient,  $R$  = hydraulic radius,  $S$  = channel longitudinal slope, and  $x$  and  $y$  are exponents. Chezy (1769), designed an improved water system by deriving an equation and relating the uniform velocity of flow to the hydraulic radius and the longitudinal slope of the channel,

$$V = C\sqrt{RS} \quad (2.14)$$

$$C = \sqrt{\frac{2g}{f}} \quad (2.15)$$

Where,

$C$  = Chezy resistance coefficient and it is Similar to the Darcy pipe flow equation. Manning (1891), proposed what has become known as Manning's equation as

$$V = \frac{\phi}{n} R^{2/3} \sqrt{S} \quad (2.16)$$

But,  $\phi = 1$

Hence Manning's equation is given as;

$$V = \frac{1}{n} R^{2/3} \sqrt{S} \quad (2.17)$$

Where  $n$  is Manning's resistance coefficient. The relationship among  $C$ ,  $n$ , and the Darcy-Weisbach friction factor ( $f$ ) is

$$V = \frac{\phi}{n} R^{1/6} = \sqrt{\frac{8g}{f}} \quad (2.18)$$

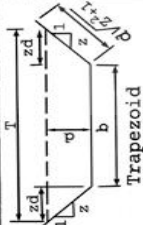

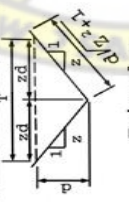

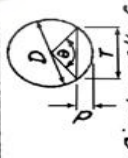
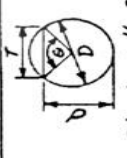
At this point, it is pertinent to observe that  $n$  is a function of boundary roughness, Reynolds number and the hydraulic radius (French, 1985).



## 2.8 BEST HYDRAULIC SECTION / ECONOMIC SECTION

For a given discharge there are many channel shapes. There is the need to find the best proportions of bed width 'B' and 'P' which will make discharge a maximum for a particular area. A section that gives maximum section factor,  $AR^{2/3}$ , for a specified flow area, A, is called the most efficient hydraulic section or best hydraulic section. Since  $Q$  is proportional to  $AR^{2/3}$  for a given channel (i.e.,  $n$  and  $So$  are specified) and  $R = A/P$ , we can say that the most efficient hydraulic section is the one that yields the minimum wetted perimeter, P for a given A. Theoretically speaking, the most efficient hydraulic section yields the most economical channel. However, it must be kept in mind that the above formulation is oversimplified. For example, we did not take into consideration the possibility of scour and erosion which may impose restrictions on the maximum flow velocity. And, for channel excavation we have to take into account the amount of overburden, viability of changing the bottom slope to suit the existing topographical conditions for minimizing the amount of excavation, ease of access, transportation of the excavated material to the disposal site, and the viability of the matching of the cut and fill volumes, etc. In addition, for a lined channel, the cost of lining as compared to the unit cost of excavation has to be taken into consideration for an overall economical design (Chaudhry, 2008).

Table 2.2 Open Channel Geometric Relationships for Various Cross Sections

Section	Area $A$	Wetted Perimeter $P$	Hydraulic Radius $R$	Top Width $T$	Critical Depth Factor, $Z$
 Trapezoid	$bd + zd^2$	$b + 2d\sqrt{z^2 + 1}$	$\frac{bd + zd^2}{b + 2d\sqrt{z^2 + 1}}$	$b + 2zd$	$\frac{[(b + zd)d]^{1.5}}{\sqrt{b + 2zd}}$
 Rectangle	$bd$	$b + 2d$	$\frac{bd}{b + 2d}$	$b$	$bd^{1.5}$
 Triangle	$zd^2$	$2d\sqrt{z^2 + 1}$	$\frac{zd}{2\sqrt{z^2 + 1}}$	$2zd$	$\frac{\sqrt{2}}{2} zd^{2.5}$
 Parabola	$\frac{2}{3} dT$	$T + \frac{8d^2}{3T}$	$\frac{2dT^2}{3T^2 + 8d^2}$	$\frac{3a}{2d}$	$\frac{2}{9}\sqrt{6} Td^{1.5}$
 Circle - $< 1/2$ full <sup>12</sup>	$\frac{D^2}{8} (\frac{\pi\theta}{180} - \sin\theta)$	$\frac{\pi D\theta}{360}$	$\frac{45D(\frac{\pi\theta}{180} - \sin\theta)}{\pi\theta}$	$D \sin \frac{\theta}{2}$ or $2\sqrt{d(D-d)}$	$a\sqrt{\frac{a}{D \sin \frac{\theta}{2}}}$
 Circle - $> 1/2$ full <sup>13</sup>	$\frac{D^2}{8} (2\pi - \frac{\pi\theta}{180} + \sin\theta)$	$\frac{\pi D(360 - \theta)}{360}$	$\frac{45D(2\pi - \frac{\pi\theta}{180} + \sin\theta)}{\pi(360 - \theta)}$	$D \sin \frac{\theta}{2}$ or $2\sqrt{d(D-d)}$	$a\sqrt{\frac{a}{D \sin \frac{\theta}{2}}}$

Note: Small  $z$  = Side Slope Horizontal Distance  
Large  $Z$  = Critical Depth Section Factor

<sup>11</sup> Satisfactory approximation for the interval  $0 < \frac{d}{T} \leq 0.25$   
When  $d/T > 0.25$ , use  $p = \frac{1}{2}\sqrt{6d^2 + T^2} + \frac{T^2}{8d} \sinh^{-1} \frac{4d}{T}$

<sup>12</sup>  $\theta = 4 \sin^{-1} \sqrt{d/D}$  Insert  $\theta$  in degrees in above equations

<sup>13</sup>  $\theta = 4 \cos^{-1} \sqrt{d/D}$

Source: USDA, SCS.NEH-5 (1956)



## **2.9 FLOW CLASSIFICATION**

### **2.9.1 STEADY AND UNSTEADY FLOW**

The flow in an open channel can be classified as steady or unsteady. The flow is said to be steady if the depth of flow at a section, for a given discharge, is constant with respect to time. The flow is considered unsteady if the depth of flow varies with respect to time (French, 1985)

### **2.9.2 UNIFORM FLOW**

The flow is said to be uniform if the depth of flow and quantity of water are constant at every section of the channel under consideration. Uniform flow can be maintained only when the shape, size, roughness and slope of the channel are constant. Under uniform flow conditions, the depth and mean velocity of flow is said to be normal. Under these conditions the water surface and flow lines will be parallel to the stream bed and a hydrostatic pressure condition will exist, the pressure at a given section will vary linearly with depth. Uniform flow conditions are rarely attained in the field, but the error in assuming uniform flow in a channel of fairly constant slope, roughness and cross section is relatively small when compared to the uncertainties of estimating the design discharge. (Sleigh and Goodwill, 2008).

### **2.9.3 NON-UNIFORM FLOW**

#### **2.9.3a STEADY GRADUALLY-VARIED FLOW**

Das (2008), stated that, steady gradually-varied flow is a condition under which flow varies slowly along the channel but does not change with time. The most common situation where this arises is in the vicinity of a control in a channel, where there may be a structure such as a

weir, which has a particular discharge relationship between the water surface level and the discharge. Far away from the control, the flow may be uniform, and there the relationship between surface elevation and discharge is in general a different one, typically being given by Manning's equations. The transition between conditions at the control and where there is uniform flow is described by the gradually-varied Flow equation, which is a differential equation for the water surface height. The solution will approach uniform flow if the channel is prismatic, but in general we can treat it as non-prismatic waterways.

### **2.9.3b RAPID VARIED FLOW**

According to Otey (2008), when there is a sudden change in the cross section of a channel, an obstruction or a very steep bed slope; rapid changes in stage and velocity occur and the resulting flow is termed rapidly varied flow. In this case, the depth has a large change over a short distance. Such flow is experienced in waterfalls and the other obstruction like the sluice gates. In this type of flow, the surface is highly curved and the streamlines are not parallel. Hence, the predictions of a hydrostatic pressure do not apply. But energy and momentum equation can be applied to obtain an approximate result to the flow. In the analysis of open channel it is endeavor to know whether the critical flow may take place. This is because critical condition imposes a limitation on the discharge. The effect of the critical condition is felt when there is a change from tranquil flow to rapid flow.

### **2.10 VELOCITY DISTRIBUTION IN OPEN CHANNELS**

According to Subramanya (1986), the presence of corners and boundaries in open channels causes the velocity vectors of flow to have components in longitudinal, lateral and normal direction to the flow. The distribution of velocity in a channel is dependent on the geometry of the channel. The distribution of velocity is zero at the solid boundaries and gradually increases

with distance from the boundary. The maximum velocity of the cross section occurs at a certain distance below the free surface. These tip of the maximum velocity point, giving surface velocities are less than the maximum velocity is due to secondary currents and is a function of depth to width of the channel. Thus for a deep narrow channel, the location of the maximum velocity point will be much lower from the water surface than for wider channel of the same depth. This characteristic location of the maximum velocity point below the surface to the direction of flow has nothing to do with wind shear of the free surface.

## 2.11 FUNDAMENTAL EQUATIONS OF FLOW

### 2.11.1 CONTINUITY EQUATION

Douglas et al (2005) stated that, when a fluid is in motion, it must move in such a way that mass is conserved. When this principle is applied under steady flow conditions to one dimensional fluid flow, it is generally expressed as the mass flow rate or flux past a second section equals the mass flow rate past the first section plus the mass flow rate entering between the sections.

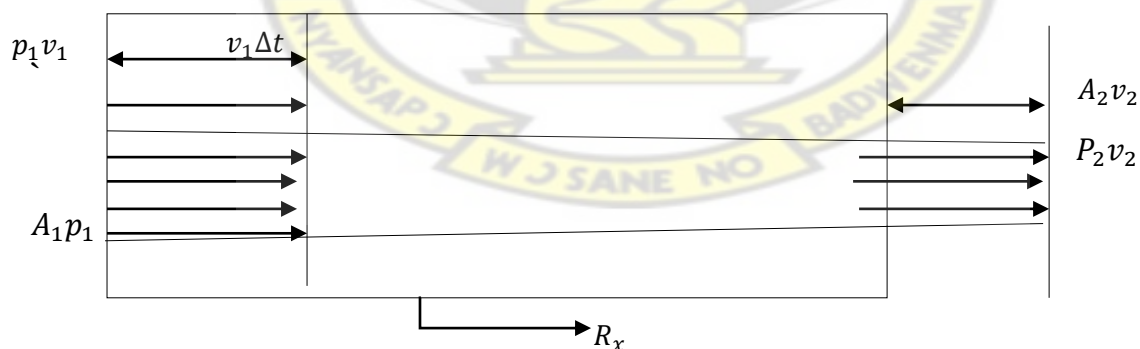


Figure 2.3 Fluid flows in and out of a channel.

One dimensional duct showing control volume .Applying the principle of mass conservation since there is no flow through the side walls of the ducts, the flow is steady so that there is no mass accumulation.

Volume of flow into  $A_1 = A_1 V_1 \Delta t$       Volume flow out of  $A_2 = A_2 V_2 \Delta t$

Mass of flow into  $A_1 = \rho A_1 V_1 \Delta t$  ,      Mass flow out of  $A_2 = \rho A_2 V_2 \Delta t$

So,  $\rho A_1 V_1 = \rho A_2 V_2$  (2.18)

### 2.11.2 ENERGY EQUATION

According to Sleight and Goodwill (2008)

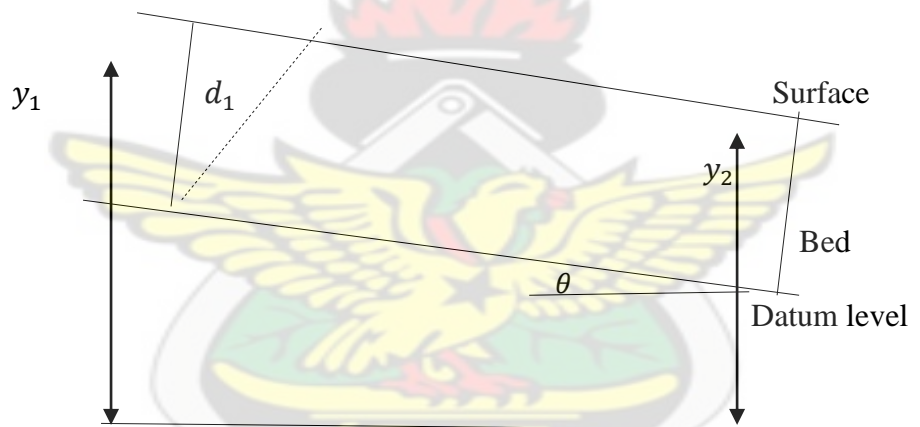


Figure 2.4 Flow of fluid in the channel.

Consider the form of energy available for the control volume. If the fluid moves from the upstream face1, to downstream face2 in  $\delta t$  over the length  $L$ , the work done in moving the fluid through face 1, during this time is

$$\text{Work done} = P_1 A_1 L \quad (2.19)$$

Where,

$P_1$  = pressure at face 1

$A_1$  = area at face 1

The mass entering through face 1 is given as

$$\text{Mass entering} = \rho_1 A_1 L \quad (2.20)$$

Therefore,

$$\text{Kinetic energy} = \frac{1}{2} m V^2 = \frac{1}{2} \rho_1 A_1 L U_1^2 \quad (2.21)$$

If  $z_1$  the height of the centroid face  $L$ , then the potential energy of the fluid entering the control volume is given as,

$$\text{Potential energy} = mgz = \rho_1 A_1 L g z_1 \quad (2.22)$$

The total energy entering the control volume is the sum of work done, potential and kinetic energy

$$\text{Total energy} = \rho_1 A_1 L + \frac{1}{2} \rho_1 A_1 L U_1^2 + \rho_1 A_1 L g z_1 \quad (2.23)$$

This can be put in the form as energy per unit weight at face 1

$$\text{Total energy} = \frac{p_1}{\rho_1 g} + \frac{U_1^2}{2g} + z_1 \quad (2.24)$$

At the exit to control volume face 2, a similar consideration can be deduced;

$$\text{Total energy} = \frac{p_2}{\rho_2 g} + \frac{U_2^2}{2g} + z_2 \quad (2.25)$$

If no energy is supplied to the control volume between the inlet and the outlet then,

Energy = energy leaving and if the fluid is compressible,  $\rho_1 = \rho_2 = \rho$

Hence Bernoulli equation,

$$\frac{p_1}{\rho_1 g} + \frac{U_1^2}{2g} + z_1 = \frac{p_2}{\rho_2 g} + \frac{U_2^2}{2g} + z_2 = H = \text{constant} \quad (2.26)$$

### 2.11.3 MOMENTUM EQUATION

Douglas et al (2001) stated that, the momentum of a particle is the product of its mass and velocity

$$\text{Momentum} = mv \quad (2.27)$$

Where,

$M = \text{Mass}$

$V = \text{Velocity}$

The objects of a fluid stream will possess momentum and anytime the velocity of the stream is change in magnitude and direction, there will be corresponding change in the momentum of fluid. According to Newton's 2<sup>nd</sup> law, a force is required to cause a change which may be directly proportional to the rate at which the change of momentum occurs. The force is applied at the point of contact between the fluid and solid boundary by one part of the fluid stream acting on another. By so doing the fluid exerts an equal and opposite force on the solid boundary/body of the fluid producing the change of velocity called dynamic force, since they emerged from the motion of fluid. These are added to static forces due to pressure in the fluid, they occur even when the fluid is at rest.

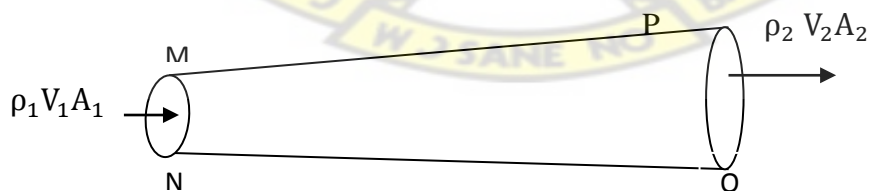


Figure 2.5 Flow of fluid through a pipe

$$\rho_2 V_2 A_2 = \rho_1 V_1 A_1 = m \quad (2.28)$$



Since there is no storage MNMN and OP then,  $m$  is the fluid mass flow change of momentum across the control volume may be seen to be,  $m = \rho_2 V_2 A_2 - \rho_1 V_1 A_1$

$$\text{Mass} \times \text{change in velocity per unit time} = V_1 \rho_1 A_1 (V_2 - V_1) = m(V_2 - V_1) \quad (2.29)$$

The resultant force is required is given as:

$$F = m(V_2 - V_1) \quad (2.30)$$

Where  $V_2$  and  $V_1$  were in the same direction. However for a two dimensional flow  $V_1$  makes an angle with  $x$  axis, while  $V_2$  makes a corresponding angle  $\theta$ . since both momentum and force are vector resolution comes into play in both  $x$  and  $y$  direction.

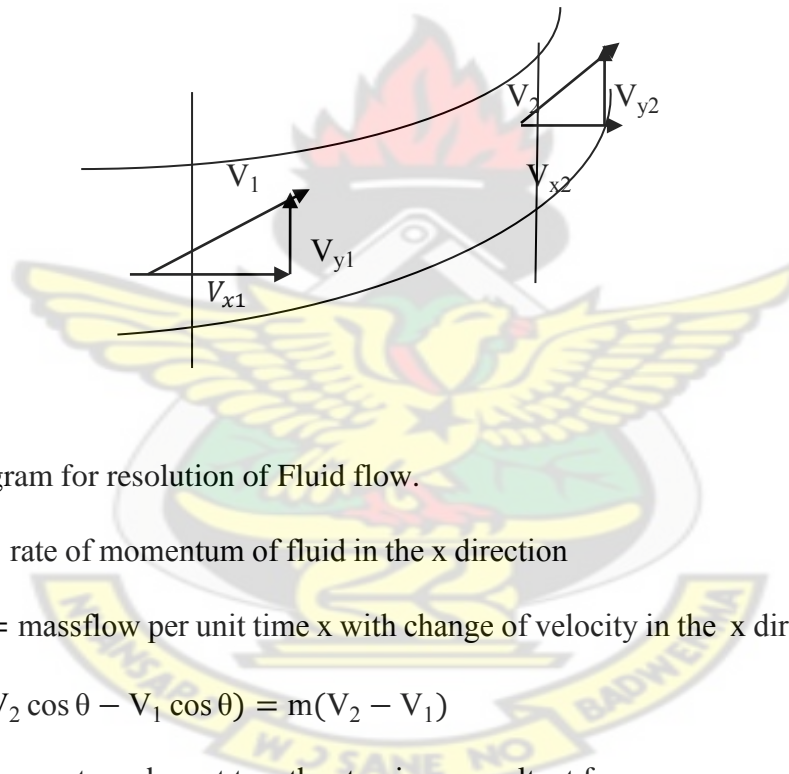


Figure 2.6 Diagram for resolution of Fluid flow.

$F_x$  = rate of momentum of fluid in the  $x$  direction

$F_y$  = massflow per unit time  $x$  with change of velocity in the  $x$  direction

$$m(V_2 \cos \theta - V_1 \cos \theta) = m(V_2 - V_1) \quad (2.31)$$

The above component can be put together to give a resultant force

$$F = \sqrt{F_x^2 + F_y^2} \quad (2.32)$$

In addition the force applied on the surrounding are equal and opposite. For 3 – dimensional flow the same fluid has component velocity  $V_2, V_1$  in the  $Z$  direction which corresponds to the rate of change of momentum in the direction of the required a force,

$$F_x = m(V_2 - V_1) \quad (2.33)$$

#### 2.11.4 SPECIFIC ENERGY

This is defined as the energy per unit weight of the liquid at a cross section measured above bed level at that point.

If the depth is given as  $d$  and  $v_m$  is the mean velocity, then:

$$E = d + \frac{v_m^2}{2g}, \quad (2.34)$$

This shows that for a given specific energy  $E$  the depth  $d$  of flow is limited. Taking into consideration a wide rectangular channel, width  $B$ , cross-sectional area,  $A$ , through which there is volume of a flow  $Q$ , per unit time.

$$V_m = \frac{Q}{A} = \frac{Q}{Bd}, \text{ Substituting for } V_m$$

$$\text{But, } q = \frac{Q}{b} = \text{volume flow rate of per unit width, } E = d + \frac{q^2}{2gd^2} \quad (2.35)$$

$$d^3 - Ed^2 + q^2/2g = 0 \quad (2.36)$$

For a constant value of specific energy  $E$ , there are two alternate depths for a given discharge  $q$ . Similarly, for a constant value of the discharge per unit width  $q$ , there will be two alternate depths for a given specific energy. The greater of these two values correspond to the condition of deep flow, it is known as subcritical, tranquil or streaming flow. Similarly, when a flow is characterized as rapid and has shallow fast flow it is known as supercritical or shooting flow (Henderson, 1966).

#### 2.11.5 CRITICAL DEPTH

According to Chanson (1999), critical depth occurs at the point where two roots coincide, when the discharge for a given specific energy is a maximum, the energy required for a given discharge is a minimum. This is given as

$$E = d_c + \frac{gd_c^3}{2gd_c^2} \quad (2.37)$$



Hence critical depth,

$$d_c = \frac{2}{3}E \quad (2.38)$$

Where,

$E$  = specific energy

$g$  = acceleration due to gravity.

$d_c$  = critical depth

## 2.12 FROUDE NUMBER AND FLOW STATES

The Froude number,  $Fr$ , is a dimensionless value that describes different flow regimes of open channel flow. The Froude number is a ratio of inertial and gravitational forces.

$$Fr = \frac{V}{\sqrt{gD}} \quad (2.39)$$

Where,

$V$  = Water velocity

$D$  = Hydraulic depth (cross sectional area of flow / top width)

$g$  = acceleration due to gravity.

When,

$Fr = 1$ , critical flow,

$Fr > 1$ , supercritical flow (fast rapid flow),

$Fr < 1$ , subcritical flow (slow / tranquil flow)

The Froude number is a measurement of bulk flow characteristics such as waves, sand bed forms, and flow/depth interactions at a cross section or between boulders.

The denominator represents the speed of a small wave on the water surface relative to the speed of the water, called wave celerity. At critical flow, celerity equals flow velocity. Any

disturbance to the surface will remain stationary. In subcritical flow the flow is controlled from a downstream point and information is transmitted upstream. This condition leads to backwater effects. Supercritical flow is controlled upstream and disturbances are transmitted downstream (Calvert, 2003).

# KNUST



## CHAPTER THREE

### MATERIALS AND METHODS

#### 3.1 RESEARCH METHODS

The research work was a desktop study and materials used were obtained from the internet and with the support of secondary data. The design of software for an open channel hydraulics is a mathematical interpretation of the design work in computer software, using the fundamental principles of mathematics and science.

Fundamental scientific principles such as the continuity equation and Manning's equation were used. A flow chart was used to illustrate the relationship between the parameters of the various types of open channel (lined and unlined) as well as the best hydraulic /economic sections, non-economic section, tractive force and permissible velocity. Froude number was used to determine the state of flow. Flow charts were designed with inputs and output data well illustrated. The conceptual model depicts the idea of how the design work would perform. The MATLAB program is the computer language used for the design work. The assessment and validations were made in order to prove that several assumptions can be put in place such that the designer would have confidence in predictions for future design works.

##### 3.1.1 CONCEPTUAL MODEL

The Manning's equation which was used in the channel design is given as;

Manning's formula, 
$$V = \frac{R^{2/3} S^{1/2}}{n} \quad (3.1)$$

Law of Continuity is also given as:

$$Q = VA \quad (3.2)$$

Hence 
$$Q = A \frac{R^{2/3} S^{1/2}}{n} \quad (3.3)$$

Where,

Q = discharge (cubic. metre per second)

S = bed slope of the channel

n= Manning's roughness coefficient

R = Hydraulic Radius (m)

The hydraulic radius is the area of the water flowing in the channel ( $m^2$ ) divided by the wetted perimeter (m). The wetted perimeter of the water flowing in the channel is the part of the channel that touches the water and therefore slows down the water by friction. It is the part of the channel that is wetted. Hydraulic radius is even more confusing in a circular pipe. The equation is a good (although not perfect) estimator for the probable flow in a channel at a given depth and slope. It can be used for estimating the flow in stormwater channels for erosion and sedimentation control, for sizing drains and spacing inlets in a stormwater collection system. The Manning equation is best used for uniform steady state flows. Uniform means that the cross-section geometry of the channel remains constant along the length of the channel, and steady state means that the velocity, discharge, and depth do not change with time. Though these assumptions are rarely ever strictly achieved in reality, the Manning equation is still used to model most open channel flows where conditions are relatively steady and for reaches (portions of rivers) that have a reasonably constant cross-section for a long enough distance that the depth remains fairly constant.

The Manning equation is a semi-empirical equation. Thus, its units are inconsistent. However, to ensure that, the Manning's equation works to perfection, the software works on conditions such as:

1. The maximum design velocity shall be  $1.5 \text{ ms}^{-1}$  , but for rock surface the velocity can be  $2 \text{ ms}^{-1}$ . This is to minimize channel erosion.
2. The Froude number shall not be greater than one (1) so that the flow would not be super critical.
3. The freeboard is assumed to be 20% of the normal depth in order to prevent air waves or water surface fluctuations from overtopping the bank.
4. The Manning's roughness 'n' depends on the type of channel lining.
5. The wetted perimeter of the channel is reduced in order to ensure that discharge is maximum and the cross-sectional area remains constant to ensure that there is best hydraulic section or economic section of the channel.

### 3.2 CHARACTERISTICS OF AN OPEN CHANNEL

An open channel can be characterized by the following features:

- Flow depth (y): This the vertical distance from the bottom of the channel to the free surface of the water.(m)
- Top width (T): This is the width of the channel section at free surface. (m)
- Hydraulic depth/radius (D): This is given as the flow area divided by top width.(m)
- Bed slope (S): This is the longitudinal slope of the channel bottom.
- Wetted Perimeter (P): This is the length of the interface between the water and the channel boundary. (m)
- Side Slope (N: V): This is the horizontal to the vertical side of the side slope.
- Cross-Sectional Area (A): This is the flow area perpendicular to the direction of flow.(m<sup>2</sup>)
- Free board (f): The perpendicular distance between the top of the channel and the free surface (m)

- Radius(r): The imaginary distance from the center of the circular section to the inner circumference of the channel. (m)
- Channel bed width (b): This is the breadth/width of the channel section (m)

### 3.3 TRACTIVE FORCE THEORY

This is the fraction of the shear stress on the side slope to the shear stress on the bottom side. However, the shear stress at the boundary should not exceed critical value so as to cause erosion of the channel.

$$F = \sqrt{1 - \frac{\sin^2 \theta}{\sin^2 \phi}} \quad (3.4)$$

Where,

F= Tractive Force (N)

$\phi$  =Angle of internal soil friction

$\theta$  = Angle of inclination of the channel side slope to the horizontal

g = Acceleration due to gravity ( $\text{m/s}^2$ )

$\rho$  = Density of water ( $\text{kg/m}^3$ )

$$\tau_{cs} = 0.76\rho g y S \quad (3.5)$$

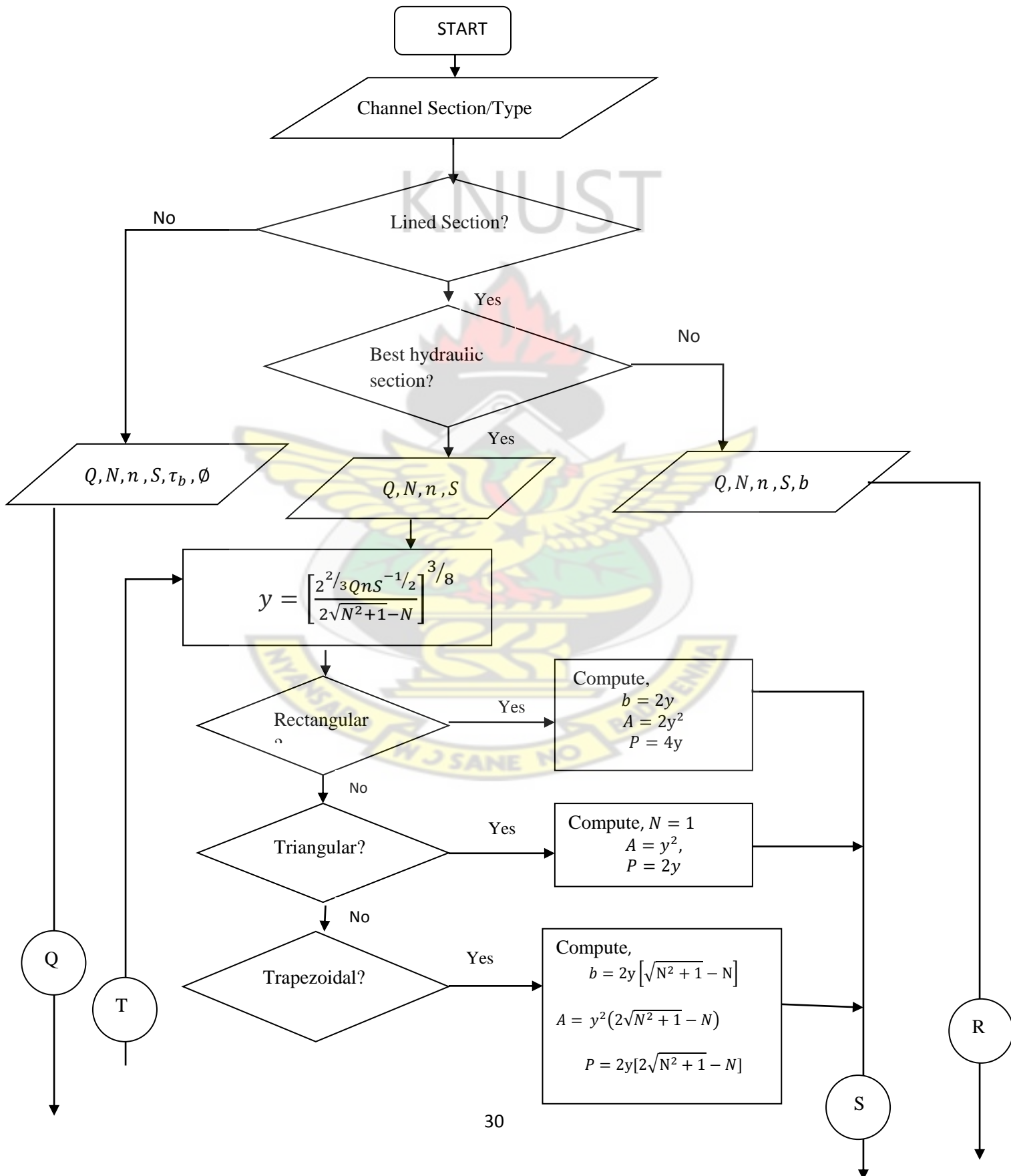
$$\tau_{cs} = F\tau_b \quad (3.6)$$

$\tau_{cs}$ = Maximum shear force at the channel side ( $\text{N/m}^2$ ).

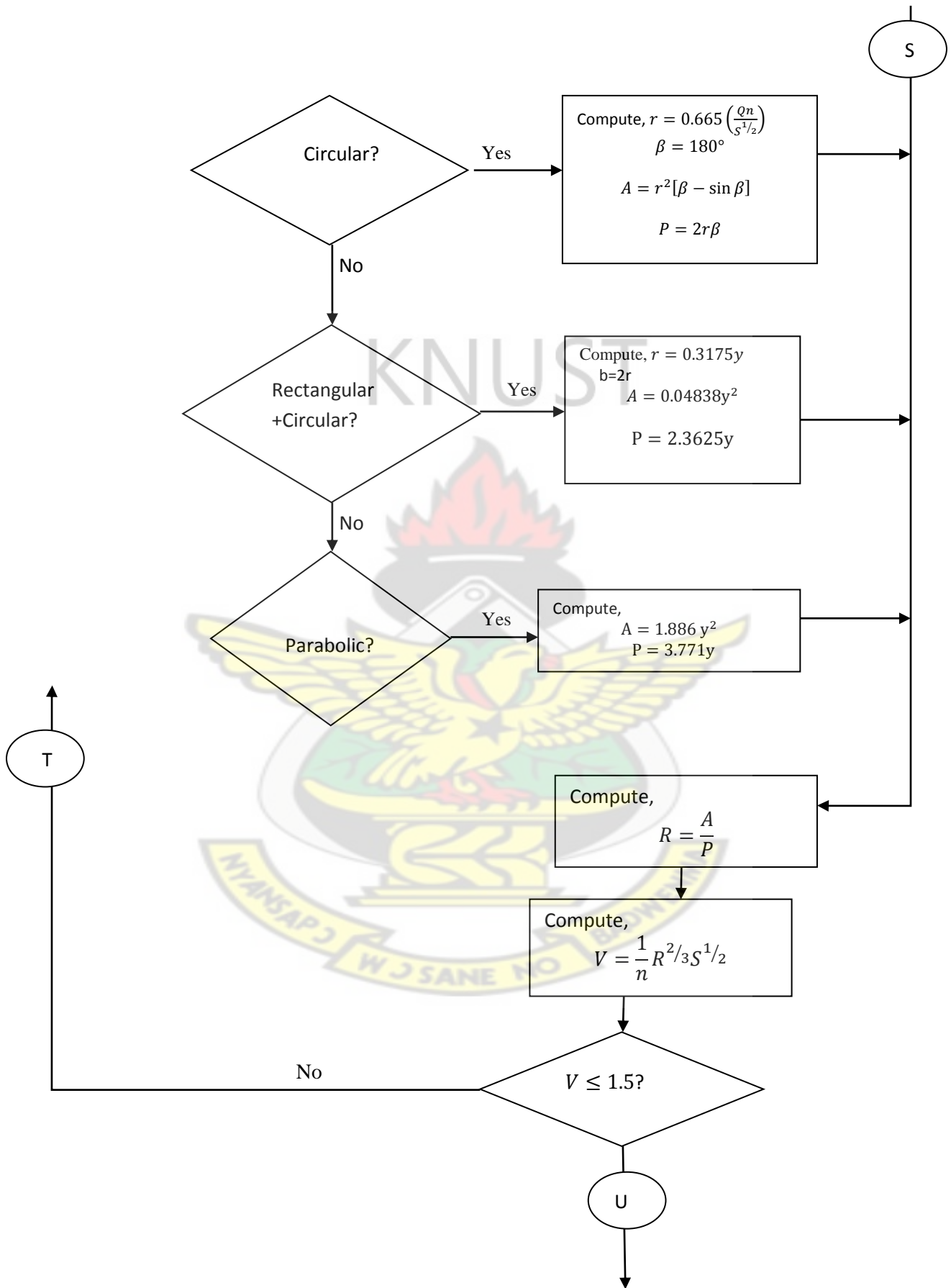
$\tau_b$ = Maximum shear force at the bottom side of the channel ( $\text{N/m}^2$ )

### 3.4 THE FLOW CHART

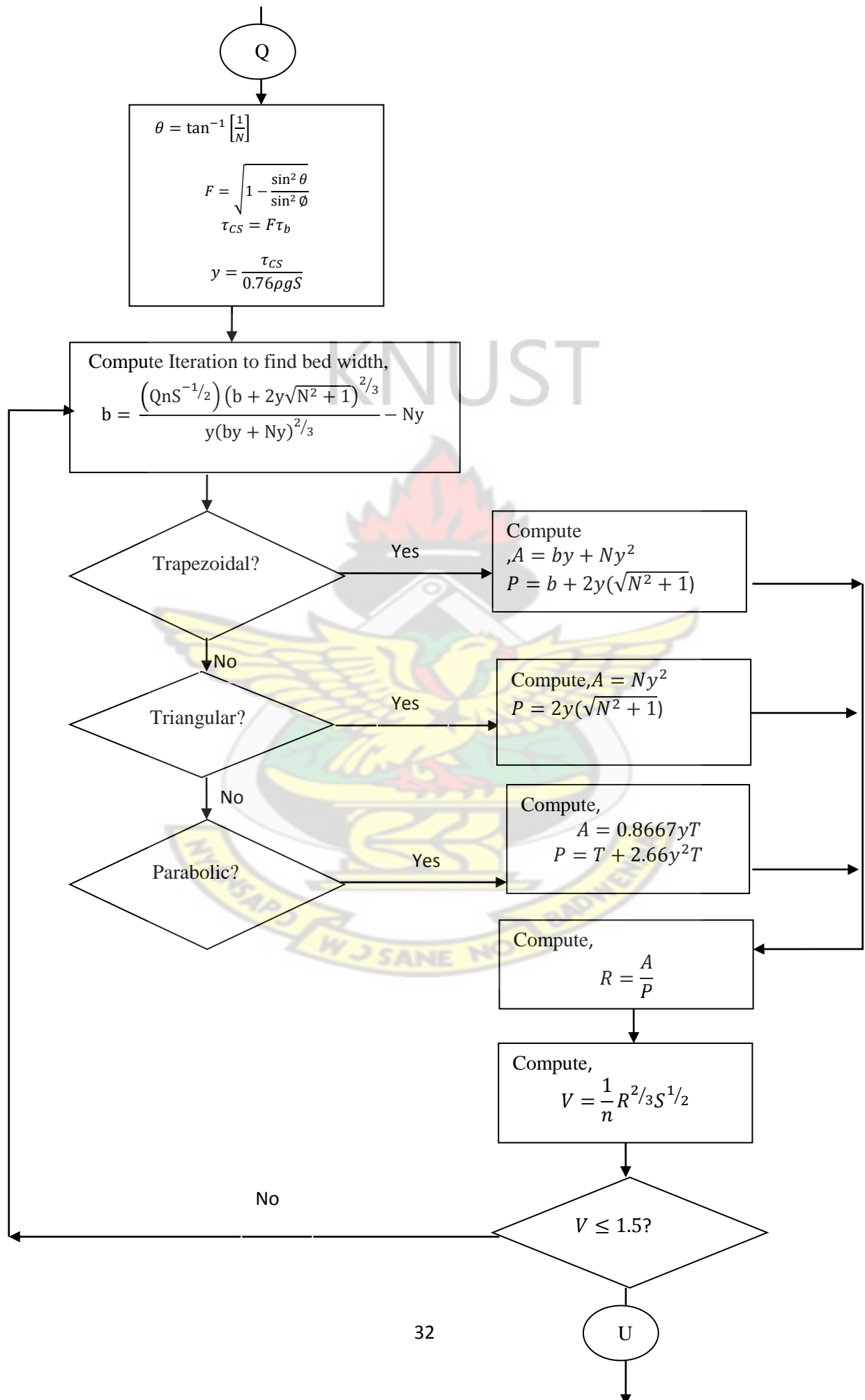
The flow chart displayed the design procedure in Figure 3.1. The chart depicts the various stages for computation of the parameters of the section of open channel hydraulics.

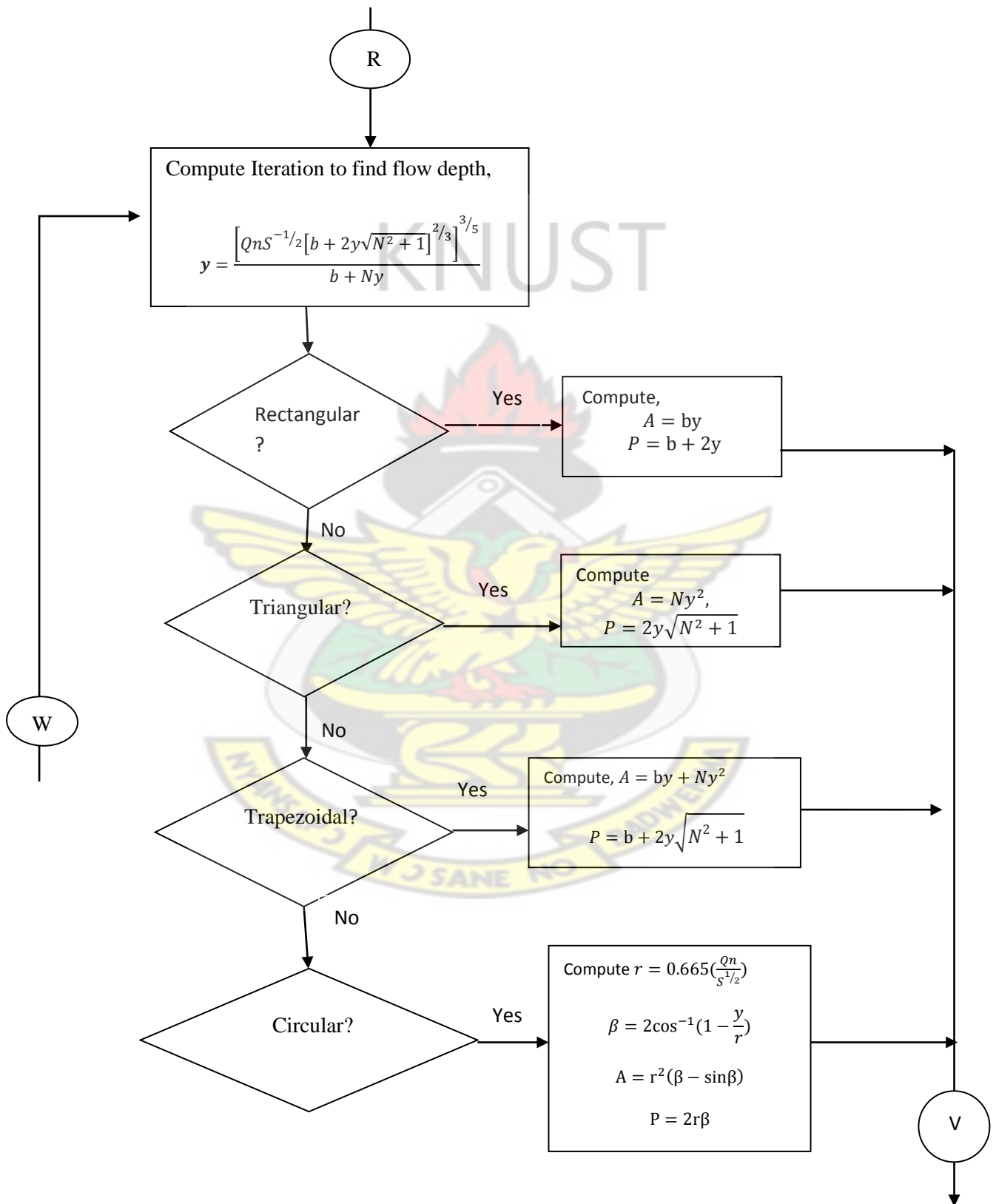


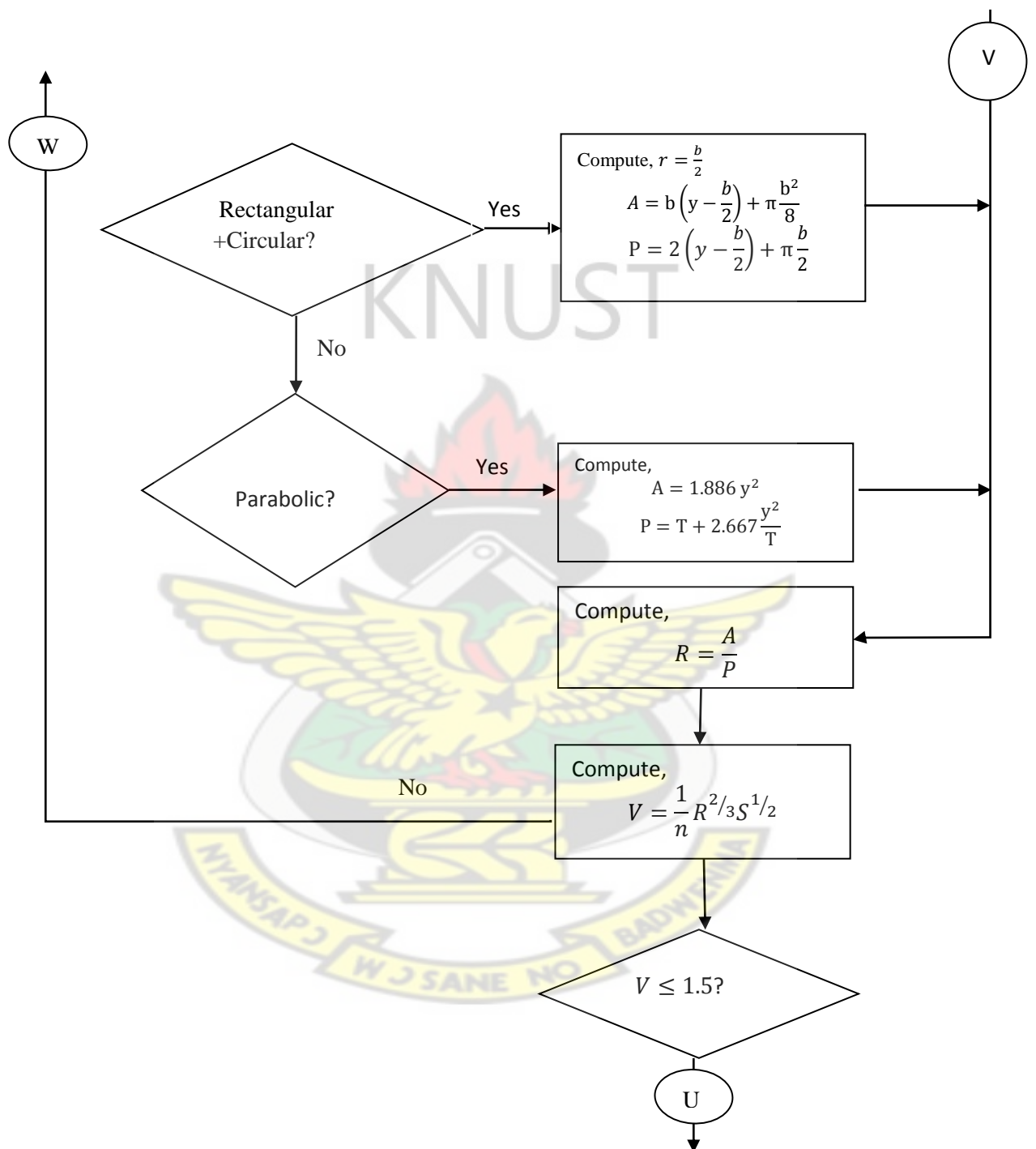












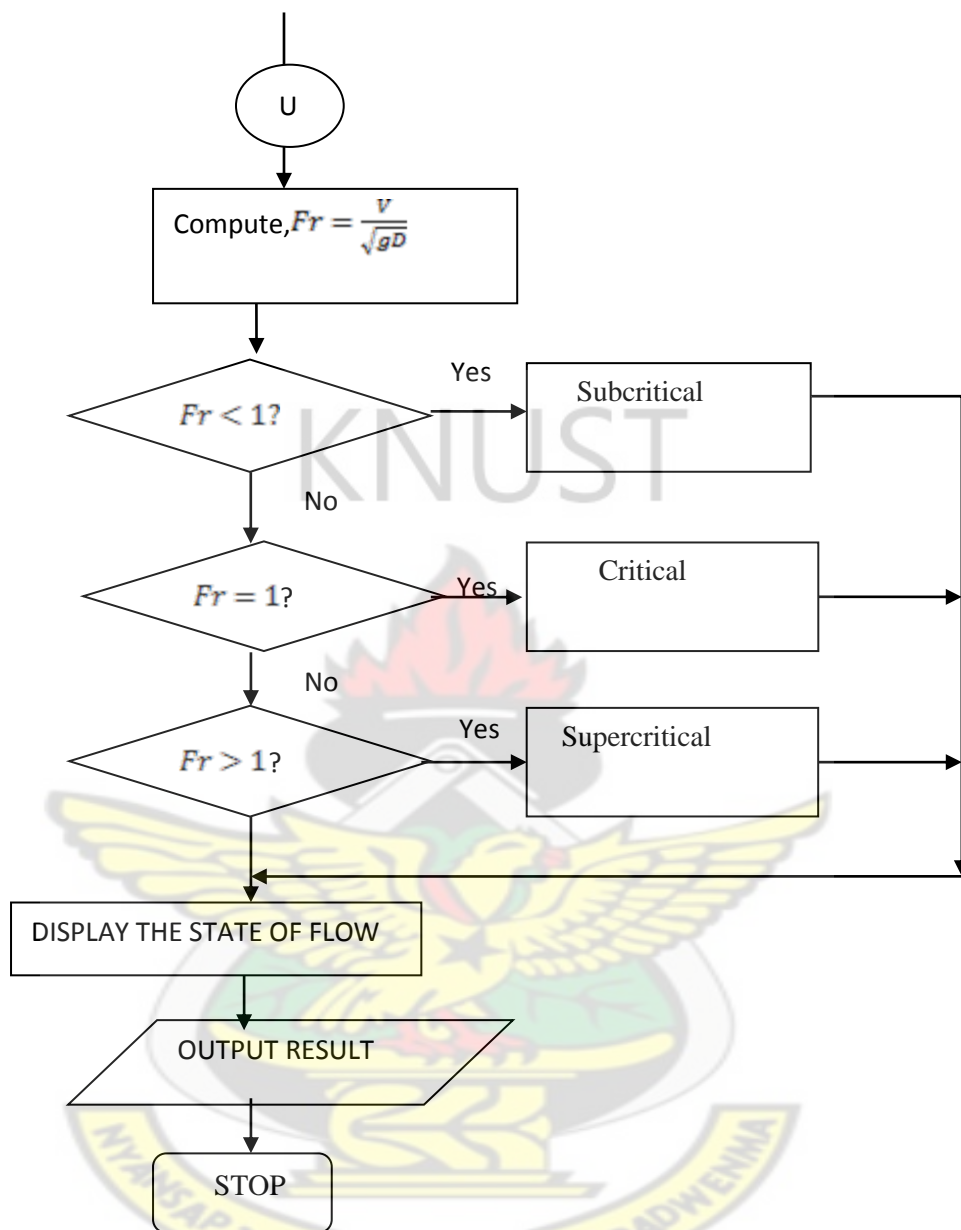


Figure 3.1. Flow Chart for input mode for the programming.

## CHAPTER FOUR

### RESULTS AND DISCUSSION

#### 4.1 RESULTS

Figure 4.1 shows the Graphic User Interface (GUI) of open channel hydraulics software. The program for the software development work is MATLAB. The software, called Open Channel Flow System, is made up of seven (7) sections.

The first section at the top left is the **Sketch of Channel**; this is the portion where the shape of the type of channel chosen is displayed for visualization. The immediate right of the Sketch of Channel is the **User Guide**; this aids the user when the User Guide button is prompted. It aids the user if there is any difficulty with the usage in terms of inputting data when a channel is chosen to be designed. Just below the **Sketch of Channel** is the **Channel Lining**, the condition under which a channel is designed; lined economic, lined uneconomic and unlined. Below this are two not always visible sections whose components for input request are based on a combination of input conditions.

- The first of these two is the **Channel Type** section; this is where the six (6) different channels under study are displayed. For lined economic and lined uneconomic all the six channel types are displayed for selection. However, for unlined condition **rectangular**, **circular** and **U-shape** channel types are excluded. A channel is selected at a time for a design based on the engineer's choice
- The second is not always visible section **Input Parameters** section; this is where input parameters are collected. Its components are based on a combination of inputs made in the **Channel Lining** and **Channel Type** sections of the screen; this is where data are inputted into the system. The immediate right of the input parameter section

is the **Output Parameters** section which produces the engineering picture of the model to be developed using numeracy.

Below the **Output Parameter** is the **Flow Status** section; this portion draws the attention of the user to be aware of how the state of flow in the channel will be after the computation (critical, subcritical, and super critical). After computation there is a **Reset** button which aids in restoring the system to its default state.

The **Calculate** button serves as driver for the computation after having finished with the computation. Computation does not proceed until all mandatory input parameters have been made. The system is shut down by a press of the **Close** button.

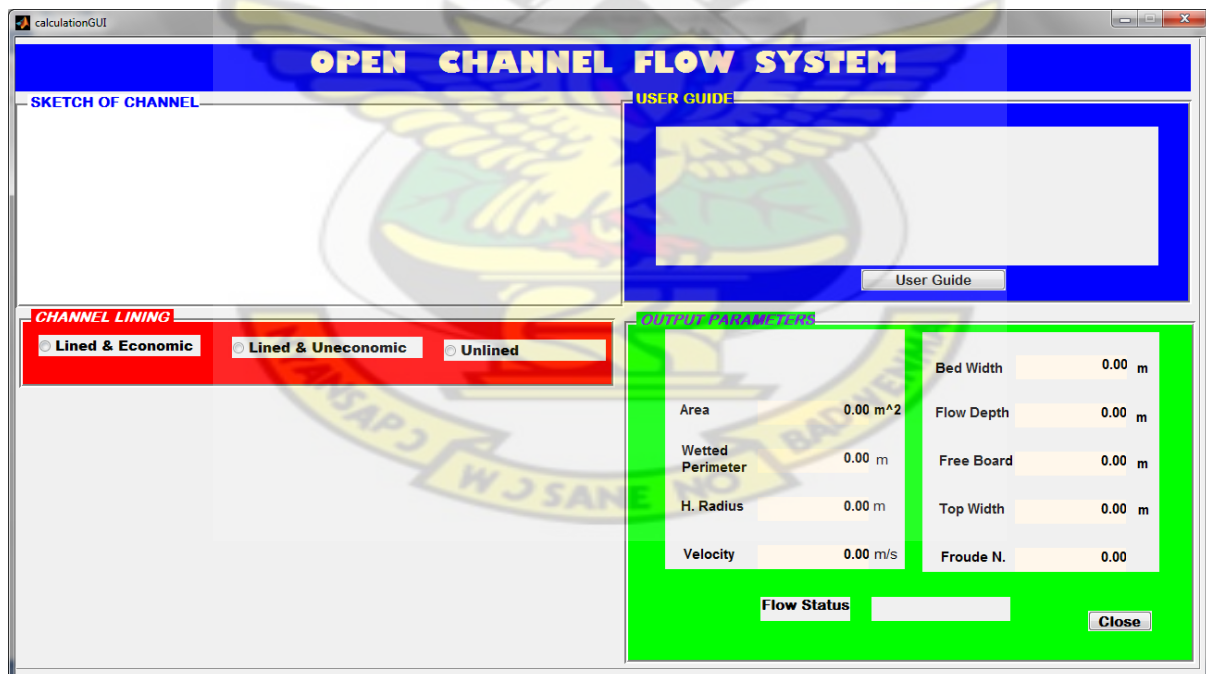


Figure 4.1 MATLAB graphical interface of open channel flow system

Upon selecting the desired **Channel Lining**, the **Channel Type** section is made visible as shown in Figure 4.2



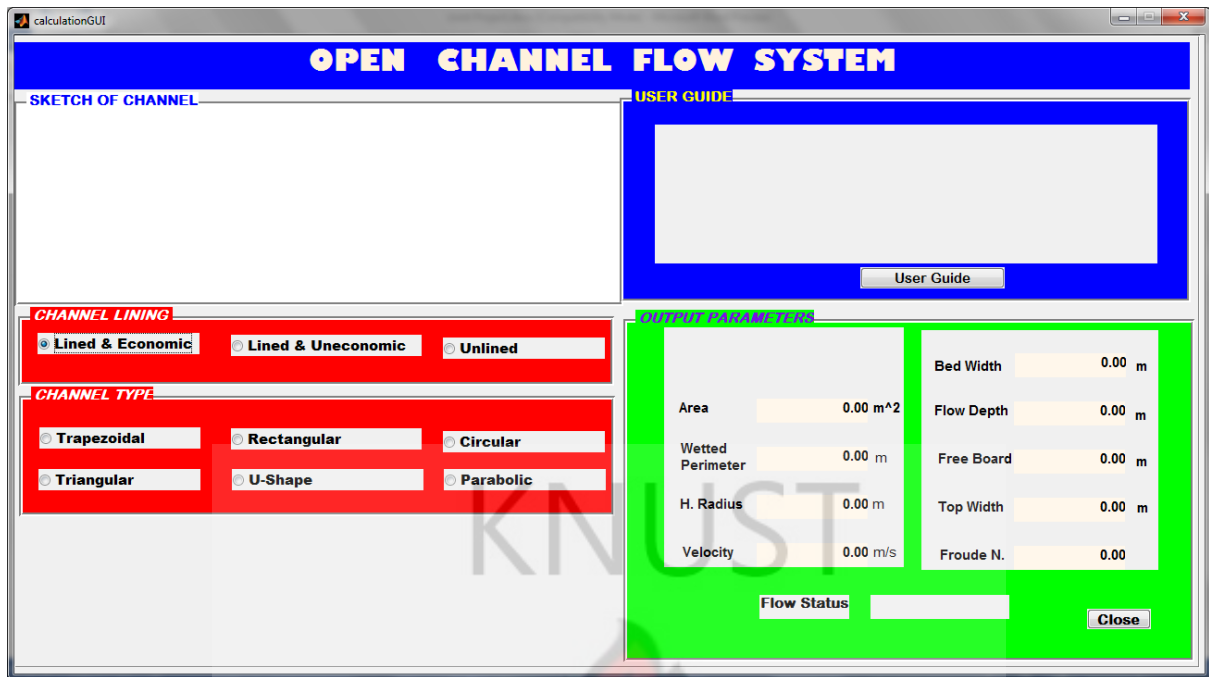


Figure 4.2 Interface with Channel Type section.

After the selection of channel type the **Input Parameter** section appears on the interface as shown in Figure 4.3.

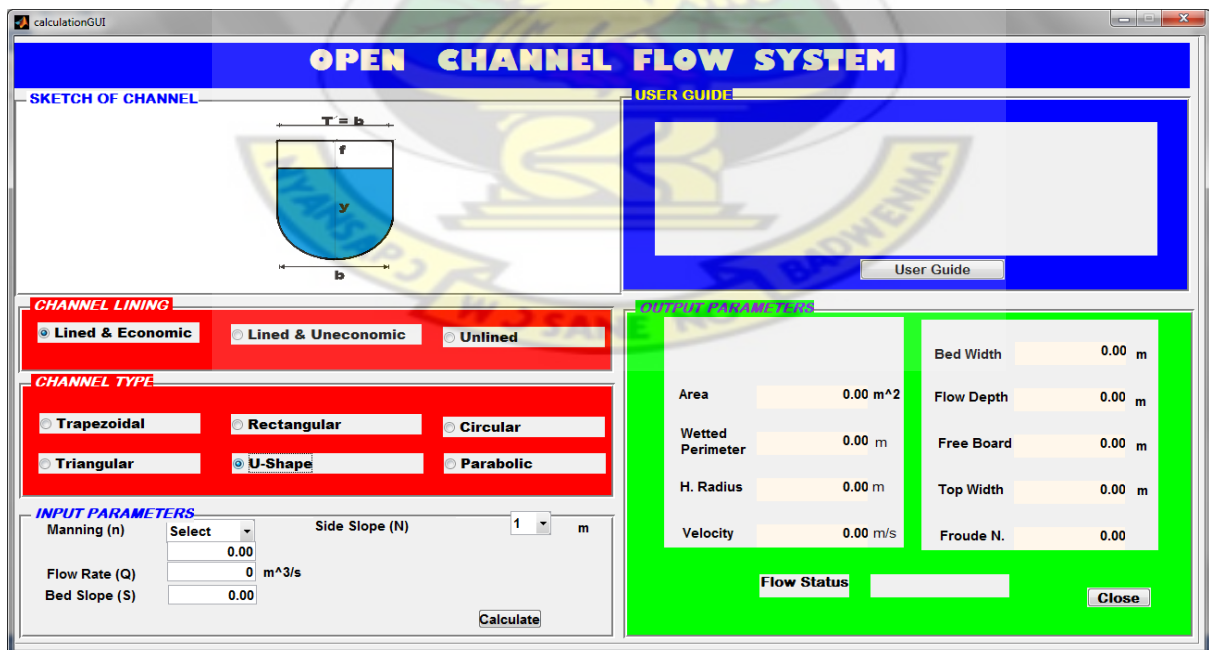


Figure 4.3 Interface ready for inputting of data.

When inputting data becomes quite irritating as shown in Figure 4.3, then the engineer needs to fall on the user guide button and prompt it. The directives appear on the **User Guide** screen and help the engineer to choose the right input of data for computation as shown in Figure 4.4.

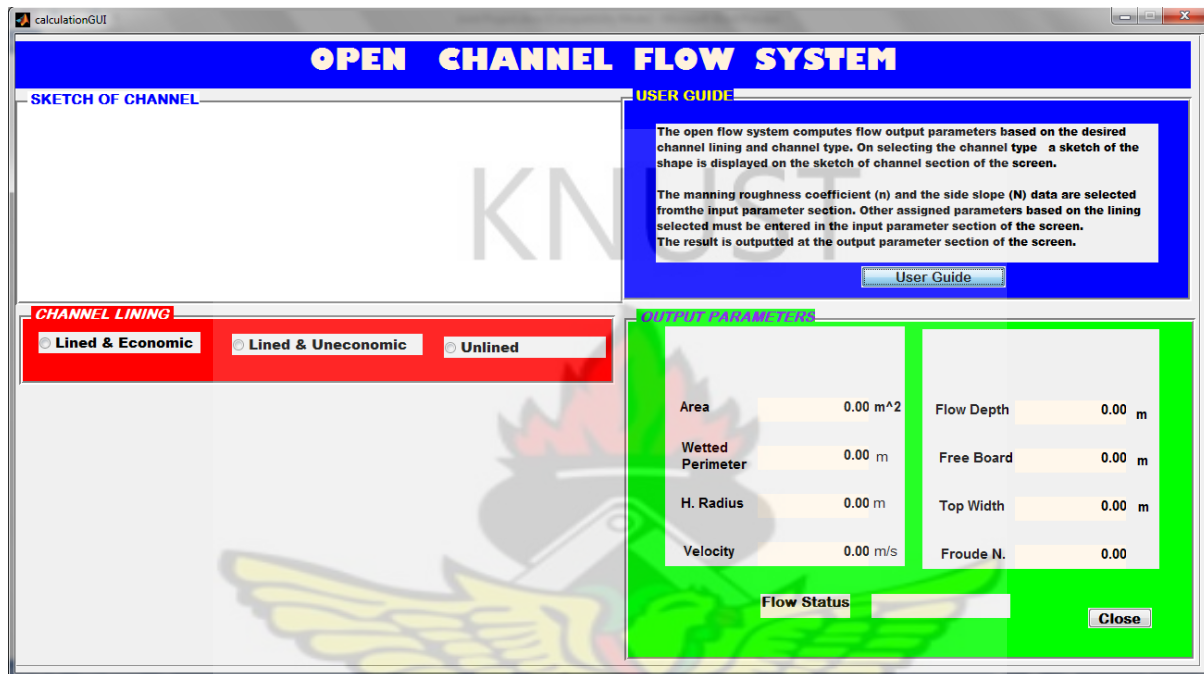


Figure 4.4a Interface with directive on the User Guide screen.

Figure 4.4b shows interface with a displayed directives on the **User Guide** screen showing remaining required inputs to be made.

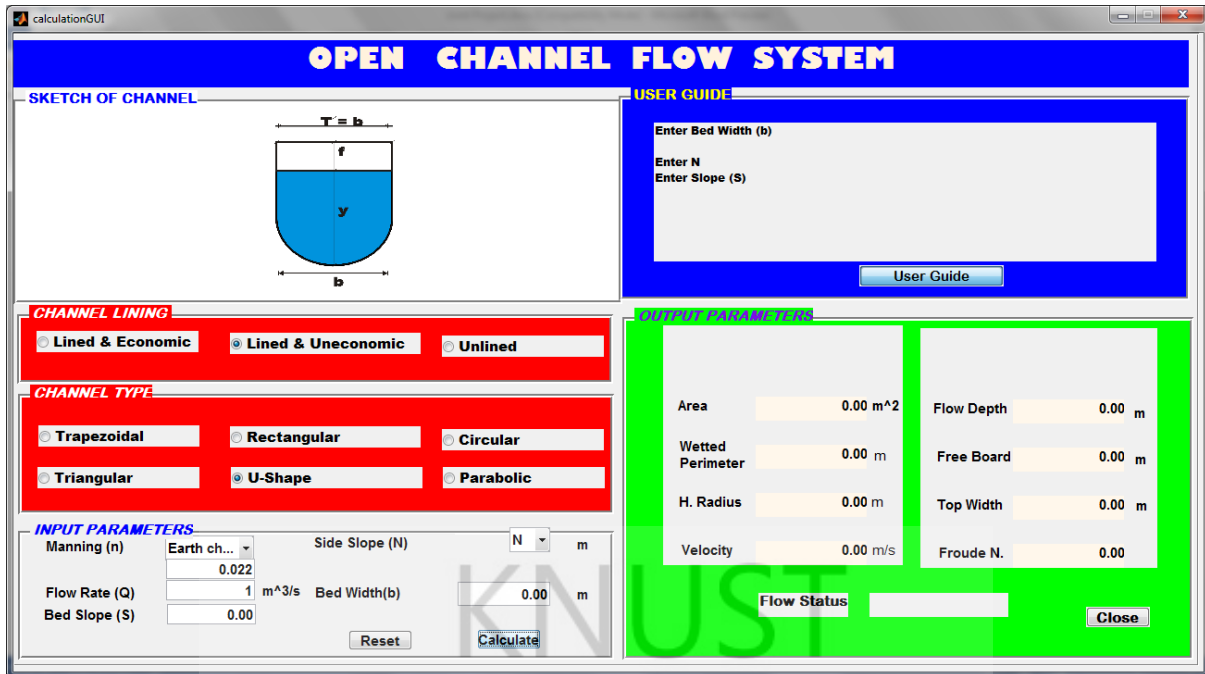


Figure 4.4b Interface with content of User Guide displayed.

With the directive offered in Figure 4.4a and Figure 4.4b, the engineer can perform computation in order to design a channel.

A computation was conducted for a design of lined and economic trapezoidal channel. The outcome is displayed on an interface shown in Figure 4.5.

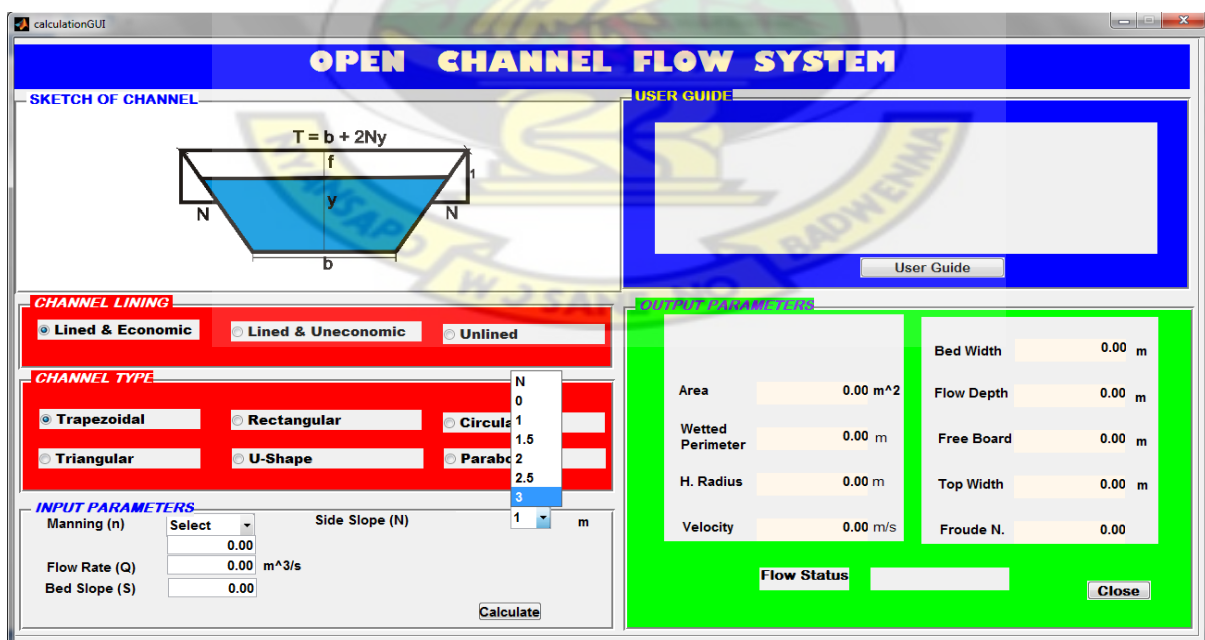


Figure 4.5 Interface for Lined and Economic trapezoidal channel.

A design of lined and uneconomic triangular channel is shown on the interface in Figure 4.6

The screenshot shows the 'OPEN CHANNEL FLOW SYSTEM' interface. The 'SKETCH OF CHANNEL' section displays a triangular channel cross-section with a top width  $T = 2Ny$ , a flow depth  $y$ , and side slopes  $N$ . The 'CHANNEL LINING' section has three radio buttons: 'Lined & Economic' (selected), 'Lined & Uneconomic', and 'Unlined'. The 'CHANNEL TYPE' section has six radio buttons: 'Trapezoidal', 'Rectangular', 'Circular', 'Triangular' (selected), 'U-Shape', and 'Parabolic'. The 'INPUT PARAMETERS' section includes a 'Manning (n)' dropdown set to '0.00', a 'Side Slope (N)' dropdown set to 'N', a 'Flow Rate (Q)' input of '0' m<sup>3</sup>/s, and a 'Bed Slope (S)' input of '0.00'. A 'Calculate' button is at the bottom right. The 'USER GUIDE' section is empty. The 'OUTPUT PARAMETERS' section shows various flow parameters: Area (0.00 m<sup>2</sup>), Flow Depth (0.00 m), Wetted Perimeter (0.00 m), Free Board (0.00 m), H. Radius (0.00 m), Top Width (0.00 m), Velocity (0.00 m/s), and Froude N. (0.00). A 'Flow Status' button and a 'Close' button are at the bottom right.

Figure 4.6 Interface for Lined and Uneconomic triangular section.

Instances where an engineer may not carefully choose desired valued for a design, the software prompts him automatically by pop up of diagnostic message as to what he should do.

The screenshot shows the 'OPEN CHANNEL FLOW SYSTEM' interface with a diagnostic message. The 'SKETCH OF CHANNEL' section displays a rectangular channel cross-section with a top width  $T = b$ , a flow depth  $y$ , and a bed width  $b$ . The 'CHANNEL LINING' section has three radio buttons: 'Lined & Economic' (selected), 'Lined & Uneconomic', and 'Unlined'. The 'CHANNEL TYPE' section has six radio buttons: 'Trapezoidal', 'Rectangular' (selected), 'Circular', 'Triangular', 'U-Shape', and 'Parabolic'. The 'INPUT PARAMETERS' section includes a 'Manning (n)' dropdown set to 'Earth ch...' with a value of '0.025', a 'Side Slope (N)' dropdown set to '0', a 'Flow Rate (Q)' input of '400' m<sup>3</sup>/s, and a 'Bed Slope (S)' input of '0.002'. 'Reset' and 'Calculate' buttons are at the bottom right. The 'USER GUIDE' section displays a message: 'Computation Failed. Please check your input parameters' and 'Diagnostics message: Function converged to a solution , but V > 1.5m/s'. A 'User Guide' button is at the bottom right. The 'OUTPUT PARAMETERS' section shows various flow parameters: Area (0.00 m<sup>2</sup>), Flow Depth (0.00 m), Wetted Perimeter (0.00 m), Free Board (0.00 m), H. Radius (0.00 m), Top Width (0.00 m), Velocity (0.00 m/s), and Froude N. (0.00). A 'Flow Status' button and a 'Close' button are at the bottom right.

Figure 4.7 Interface with Diagnostic Message.

For this interface, the diagnostic message reads; “Function converged to a solution but  $V > 1.5$  m/s”. This means that the maximum velocity criterion is not met and hence the design should re-select appropriate input parameters that will design a channel section that satisfies the maximum velocity criterion.

**Select** is prompted in front of the Manning’s (n) in Figure 4.8, and this displays Manning’s data for designer to choose a suitable value for the type of material to be used for his design work.

**OPEN CHANNEL FLOW SYSTEM**

**SKETCH OF CHANNEL**

**USER GUIDE**

**CHANNEL LINING**

Select

- Asbestos cement (0.011)
- Asphalt (0.016)
- Brass (0.011)
- Brickwork (0.015)
- Cast-iron, new (0.012)
- Clay tile (0.014)
- Concrete - centrifugally spun (0.013)
- Concrete - finished (0.012)
- Concrete - steel forms (0.011)
- Concrete - wooden forms (0.015)
- Copper (0.011)
- Corrugated metal (0.022)
- Earth (0.025)
- Earth channel - clean (0.022)
- Earth channel - gravelly (0.025)
- Earth channel - stony, cobbles (0.03)
- Earth channel - weedy (0.035)
- Floodplains - heavy brush (0.075)
- Floodplains - light brush (0.05)

**CHANNEL TYPE**

☒ Lined & Economic

☒ Trapezoidal

☐ Triangular

**INPUT PARAMETERS**

Manning (n) **Select** Side Slope (N)  m

Flow Rate (Q)  m<sup>3</sup>/s

Bed Slope (S)

**Calculate**

**Results:**

Area	0.00 m <sup>2</sup>	Bed Width	0.00 m
Wetted Perimeter	0.00 m	Flow Depth	0.00 m
H. Radius	0.00 m	Free Board	0.00 m
Velocity	0.00 m/s	Top Width	0.00 m
		Froude N.	0.00

**Flow Status**  **Close**

Fig 4.8 Interface with Manning’s Roughness Coefficient value to select.

## 4.2 DISCUSSION

This is a useful computer software program designed to calculate common open channel flow hydraulic characteristics using only the methods of the Manning's Equation for regular channel geometries for steady uniform flow. It provides an easy to use interface. Extensive efforts were undertaken to provide a single screen interface to provide the most functionality for the design engineer. Effectively this allows the design engineer to monitor and correct input data while at the same time being able to view the computational results immediately. Along with the data input controls for manual data entry, the computer program supports copying computational results to the clipboard.

In addition to entering regular channel sections, the computer program includes a comprehensive list of Manning's roughness coefficient ( $n$ ) which allows the design engineer to select any of the  $n$  values depending on the channel lining required. Output data displayed on the clipboard includes the dimensions of the channel, and the flow characteristics. This computational software did not, however, consider compound channels but includes graphical representation of the channel section.



## **CHAPTER FIVE**

### **CONCLUSION AND RECOMMENDATION**

#### **5.1 CONCLUSION**

The developed computer design software system for an open channel hydraulic system has been made simple to ease and lessen time consumed as compared to laborious manual computation of the parameters of an open hydraulic channel section. The software is a modern program which integrates computation, visualization and programming language environment that has sophisticated data structures, in-built editing and debugging tools and supports elemental –oriented programming which makes it an excellent tool for design work.

#### **5.2 RECOMMENDATION**

Further improvement to the software could consider computation and display of energy grade line of flow in the channel. This will display the energy component of the flow at any section of the channel and inform the design engineer on any changes necessary to achieve optimum flow. Other compound channels also need be considered. Future improvements to this software could include more sophisticated graphical display modes.

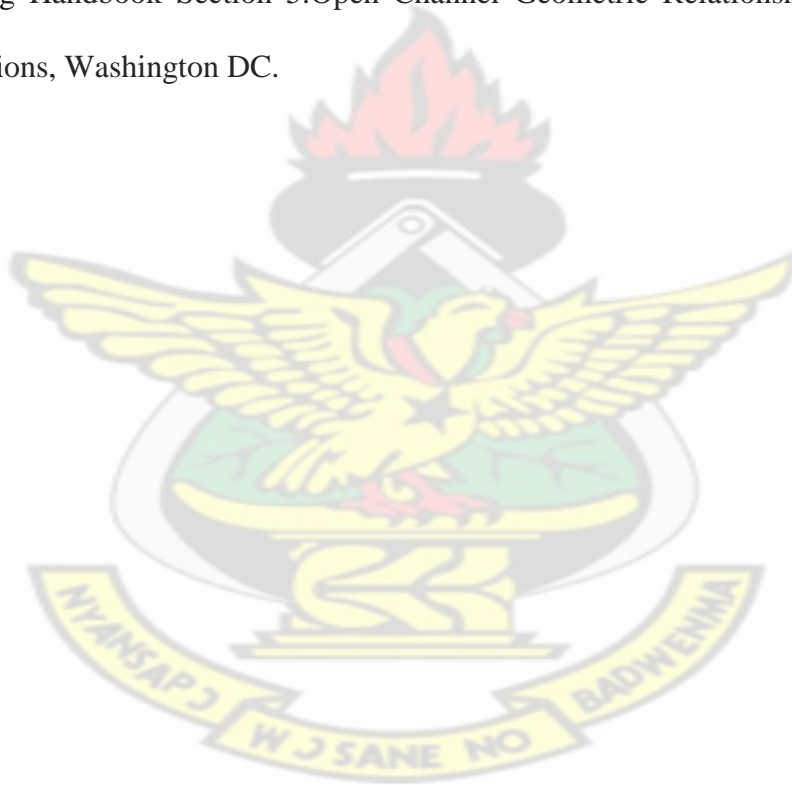
## REFERENCE

1. Akan, O.A. (2006). Open Channel Hydraulics. 1<sup>st</sup> Edn., Jordan Hill, Oxford, UK.
2. Anon. (March 1963). ASCE Report of Task Force on Friction Factors In Open Channels Hydraul. Div. ASCE, 89 (HY2), 97.
3. Arcement, G. J and Schneider, V. R. (1989). Guide for Selecting Manning's Roughness Coefficients for Natural Channels and Floodplains, US Geological Survey, Water Supply Paper 2339.
4. Brahms, A. (1754). Element of Dam and Hydraulics Engineers. Zurich, Germany. Vol. 1. Pp. 105 .
5. Brown, L.C., and Ward A.D. (1997). Understanding Agricultural Drainage. OSU Extension Factsheet AEX-320-97. Ohio State University Extension.
6. Calvert, J (2003). Open Channel Flow.  
Retrieved from: <http://mysite.du.edu/~jcalvert/tech/techom.htm>  
Date Accessed: 24/4/12.
7. Chanson, H. (1999). The Hydraulics of Open Channel Flow, Arnold, London.
8. Chanson, H. (2004). Hydraulics of open channel flow, 2<sup>nd</sup> Edn, Elsevier,
9. Chaudhry, M. H. (1993). Open-Channel Flow. Prentice Hall, Englewood Cliffs. New Jersey.
10. Chaudhry, M.H. (2008). Open Channel Hydraulics. 2<sup>nd</sup> Edn., Springer, New York.
11. Chezy, A., (1769). Antoine Chezy, histoire d' une formulae d' hydraulique, by G. Mouret, Annales des Ponts et Chaussees, vol.11. 1921.
12. Chow, V. T. (1959). Open Channel Hydraulics, McGraw-Hill Book Co., New York.
13. Das, M. M. (2008). Open Channel Flow Prentice-Hall, New Delhi.

14. Das M.M and Saika M. D. (2009). Irrigation and Water Power Engineering. PHI Learning Private Limited, New Delhi
15. Douglas, J. F., Gasoirek, J. M and Swaffield, J. A. (2001). Fluid Mechanics, 4th Edn. Prentice-Hall, London.
16. Douglas, J. F., Gasoirek, J. M and Swaffield, J. A. and Jack, L. B. (2005). Fluid Mechanics, 5th Edn. Prentice-Hall, Harlow.
17. DuBoys, M. P. (1879). The Rhone and Streams with Moveable Beds. Annales des pontes et chaussées section 5, vol.18, pp.141-195.
18. Fortier, S., and Scobey, F.C., (1926). Permissible Canal Velocities, Transaction, American Society of civil Engineer, Vol.89 pp. No.940-984.
19. Fox, R. W., and MacDonald, (1985). Introduction to Fluid Mechanics, 3<sup>rd</sup> Edn, Wiley, New York.
20. Freeze, R. A., Cherry, J. A. (1979). Ground Water, Prince-Hall, Englewood Cliffs, New Jersey.
21. French, R. H. (1985), Open-Channel Hydraulics, McGraw-Hill, New York.
22. Ganguillet, E. and Kutter, W. R. (1879). An Investigation to Establish a New General for Uniform Flow of Water in Canals and Rivers. Ziets chrift des oesterrlichishen Ingenier-and Architekten Vereines, vol.21, No. 1, pp.6-25, No. 2-3, pp.46-59.
23. Henderson, F. M. (1966). Open Channel Flow, MacMillan Co., New York.
24. Houcque, D. (2005). Introduction to MATLAB, Evanston, Illinois.
25. Jaeger, C. (1956). Engineering Fluid Mechanics, Blackie, London.
26. Karim, F., Bed Configuration and Hydraulic Resistance in Alluvial-Chanel Flows, Journal Of The Hydraulic Engineering, ASCE, Vol. 121, No. 1, Paper No. 5739, Jan. 1995, pp. 15-25.

27. Kennedy, R.G. (1895). The Provision of Silting in Irrigation Canals, Proc. No. 2826, Proc. ICE, Vol. 1, London.39.
28. Manning, R., (1895). The flow of Water in Open Channels and Pipes. Transactions Institute of Civil Engineers of Ireland. vol. 20, pp. 161-209, Dublin 1891, Supplement, Vol. 24, pp. 179-207.
29. Meyer-Peter, E and Muller, R., (1948). Formula for Bed Transport. Second meeting of the IAHR, Annex, pp.39-64.Stocklin, Sweden.
30. Montes, S. (1998). Hydraulics of Open Channel Flow, ASCE, New York
31. Munson, B. R., Young, D. F., & Okiishi, T. H. (2002). Fundamentals of Fluid Mechanics, 4th Ed., John Wiley and Sons, Inc. New York.
32. Otey U., (2008). Mechanics of Fluid. Bloomington, Indiana.
33. Rhodes, D. D., (1978). World wide variations in hydraulic geometry exponents of stream channels: an analysis and some observations-Comments. Journal of Hydrology, Vol. 33, pp. 133-146.
34. Roberson, J. A., and Crowe C. T. (1985). Engineering Fluid Mechanics, 3rd Ed., Houghton-Mifflin, Boston.
35. Rouse, H. (June 1965). Critical analysis of Open Channel Roughness, J. hydraul. Div. ASCE, 92, 1-15.
36. Sleigh, P. A and Goodwill, I. M (2008). Cive2400 Fluid Mechanics Section 2: Open channel hydraulics.
- Retrieved from: [http://www.efm.leeds.ac.uk/CIVE/CIVE2400/open\\_channel\\_hydraulic2.pdf](http://www.efm.leeds.ac.uk/CIVE/CIVE2400/open_channel_hydraulic2.pdf)
37. Date Accessed: 4/3/2012
38. Stephenson, D. (1981). Storm hydrology and Drainage .Elsevier, Amsterdam.

39. Subramanya, K. (1986). Flow in Open Channels, 2<sup>nd</sup> Edn, Tata McGraw-Hill, and New Delhi.
40. Swaffield, J. A. and Galowin, L. S. (1992). The Engineering Design of Building Drainage Systems. Ash gate, Alder shot.
41. Tritton, D.J., Physical Fluid Dynamics, Second Edition: Oxford University Press, 519 p.
42. US Army Corps of Engineers (ASACE), (1991). Hydraulic Design of Flood Control Channels. EM 1110-2-1601. U.S. Army Corps of Engineers, Washington, D.C
43. U.S. Department of Agriculture, Soil Conservation Service. (1956). National Engineering Handbook Section 5. Open Channel Geometric Relationships for Various Cross Sections, Washington DC.

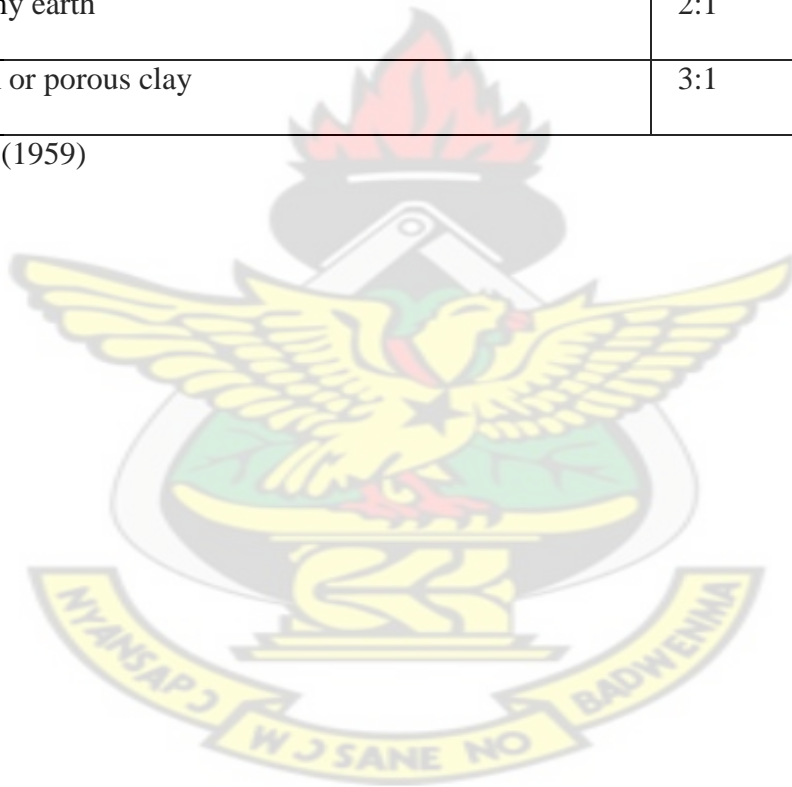


## APPENDIX A

### Recommended side slopes in various types of Materials

Materials	side slope (H: V)
Rocks	Nearly vertical
Muck and peat soil	0.5:1
Stiff clay or earth concrete lining	1:1
Earth with stone lining or earth for large channel	1:1
Firm clay or earth for small ditches	1.5:1
Loose, loamy earth	2:1
Sandy loam or porous clay	3:1

Source: Chow (1959)





## APPENDIX B

### Average Values for Manning's Roughness Coefficient

Surface Material	Manning's Roughness Coefficient  - <i>n</i> -
Asbestos cement	0.011
Asphalt	0.016
Brass	0.011
Brickwork	0.015
Cast-iron, new	0.012
Clay tile	0.014
Concrete - steel forms	0.011
Concrete – finished	0.012
Concrete - wooden forms	0.015
Concrete - centrifugally spun	0.013
Copper	0.011
Corrugated metal	0.022
Earth	0.025
Earth channel - clean	0.022
Earth channel - gravelly	0.025
Earth channel - weedy	0.030
Earth channel - stony, cobbles	0.035
Floodplains - pasture, farmland	0.035
Floodplains - light brush	0.050
Floodplains - heavy brush	0.075
Floodplains - trees	0.15
Galvanized iron	0.016
Glass	0.010
Gravel	0.029

Surface Material	Manning's Roughness Coefficient - <i>n</i> -
Lead	0.011
Masonry	0.025
Metal - corrugated	0.022
Natural streams - clean and straight	0.030
Natural streams - major rivers	0.035
Natural streams - sluggish with deep pools	0.040
Plastic	0.009
Polyethylene PE - Corrugated with smooth inner walls	0.009 - 0.015
Polyethylene PE - Corrugated with corrugated inner walls	0.018 - 0.025
Polyvinyl Chloride PVC - with smooth inner walls	0.009 - 0.011
Steel - Coal-tar enamel	0.010
Steel - smooth	0.012
Steel - New unlined	0.011
Steel - Riveted	0.019
Wood - planed	0.012
Wood - unplaned	0.013
Wood stave	0.012

Source: Chow (1959)

## APPENDIX C

### PROGARM: Makafui Open Channel Flow System

#### The MATLAB Programming Language

#### Graphic User Interface File

#### FILENAME: calculationGUI.m

```
function varargout = calculationGUI(varargin)
% CALCULATIONGUI MATLAB code for calculationGUI.fig
%   CALCULATIONGUI, by itself, creates a new CALCULATIONGUI or raises the
existing
%   singleton*.
%
%   H = CALCULATIONGUI returns the handle to a new CALCULATIONGUI or the
handle to
%   the existing singleton*.
%
%   CALCULATIONGUI('CALLBACK',hObject,eventData,handles,...) calls the local
function named CALLBACK in CALCULATIONGUI.M with the given input
arguments.
%
%   CALCULATIONGUI('Property','Value',...) creates a new CALCULATIONGUI or
raises the
%   existing singleton*. Starting from the left, property value pairs are
%   applied to the GUI before calculationGUI_OpeningFcn gets called. An
%   unrecognized property name or invalid value makes property application
%   stop. All inputs are passed to calculationGUI_OpeningFcn via varargin.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help calculationGUI

% Last Modified by GUIDE v2.5 20-Apr-2013 09:23:32

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',    mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @calculationGUI_OpeningFcn, ...
    'gui_OutputFcn', @calculationGUI_OutputFcn, ...
    'gui_LayoutFcn', [] , ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
```

```

end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before calculationGUI is made visible.
function calculationGUI_OpeningFcn(hObject, ~, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
% varargin    command line arguments to calculationGUI (see VARARGIN)

% Choose default command line output for calculationGUI
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes calculationGUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% {
Q=0; %discharge
N=0; %Raynold Number for pipe flow
y=0; %hydraulic depth
n=0; %Manning coefficient
S=0; %slope of channel
Tb=0;
phi=0;
theta =0;
%R=0; %Hydraulic radius in metres
%A=0; %surface area
W=0; %wetted perimeter
%Fr=0; %Froude Number (gravity versus inertial forces)
V =0; % Velocity of flow (v) in metres/sec
stateOfFlow='Critical';
b=0; %bed width
% }
set(handles.channelType,'SelectedObject',[]);
set(handles.channelSection,'SelectedObject',[]);

% --- Outputs from this function are returned to the command line.

```

```

function varargout = calculationGUI_OutputFcn(hObject, ~, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over pushbutton1.
function pushbutton1_ButtonDownFcn(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in reset.
function reset_Callback(hObject, eventdata, handles)
% hObject handle to reset (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
cla(handles.channelImage);
set(handles.channelType,'SelectedObject',[]);
set(handles.channelSection,'SelectedObject',[]);
set(handles.channelType,'Visible','off');
set(handles.inputPanel,'Visible','off');
set(handles.areaValue,'String','0.00');
set(handles.wettedPerimeterValue,'String','0.00');
set(handles.radiusValue,'String','0.00');
set(handles.velocityValue,'String','0.00');
set(handles.bedWidthOutputValue,'String','0.00');
set(handles.flowDepthValue,'String','0.00');
set(handles.freeBoardValue,'String','0.00');
set(handles.topWidthValue,'String','0.00');

```



```

set(handles.FroudeNumberValue,'String','0.00');
set(handles.stateOfFlow,'String','');
set(handles.reset,'Visible','off');
set(handles.slopeValue,'String','0.00');
set(handles.phiValue,'String','0.00');
set(handles.TbValue,'String','0.00');
set(handles.inputMn,'Value',1);
set(handles.inputMnValue,'String','0.00');
set(handles.bedWidthInputValue,'String','0.00');
set(handles.flowRateValue,'String','0.00');
set(handles.NValue,'Value',1);
userGuideStr(1) = {'';
    userGuideStr(2) = {'';
    userGuideStr(3) = {'';
    userGuideStr(4) = {'';
    userGuideStr(5) = {'';
    userGuideStr(6) = {'';
    userGuideStr(7) = {'';
    userGuideStr(8) = {'';
set(handles.guideTag,'Style','text','String',userGuideStr,'HorizontalAlignment','left')

```

```

% --- Executes on button press in submitInputParaRequest.
function submitInputParaRequest_Callback(hObject, eventdata, handles)
% hObject    handle to submitInputParaRequest (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
disp('Starting the execution of the open flow computation');
set(handles.inputPanel,'Visible','on');
%makafuiOpenChanelFlowParam;

```

```

% --- Executes on button press in submitInputValues.
function submitInputValues_Callback(hObject, eventdata, handles)
% hObject    handle to submitInputValues (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
shape=[];
set(handles.userGuideTag,'Value',0);

```

```

set(handles.userGuideTag,'Value',1);
    userGuideStr(1) = {'';
    userGuideStr(2) = {'';
    userGuideStr(3) = {'';

```



```

userGuideStr(4) = {''};
userGuideStr(5) = {''};
userGuideStr(6) = {''};
userGuideStr(7) = {''};
userGuideStr(8) = {''};
set(handles.guideTag,'Style','text','String',userGuideStr,'HorizontalAlignment','left');
[Q,N,n,S,Tb,shape,channelSection,phi,b,allInputsDone] = readInputParams(hObject,
eventdata, handles);
if allInputsDone==1
switch shape
case {'Trapezoidal'}
    rgb = imread('trapezoidalFlow.png');
    image(rgb); %title('Trapezoidal Channel');
    axis image;
    axis off;

case {'Rectangular'}
    rgb = imread('rectangularFlow.png');
    image(rgb); %title('Circular Channel')
    axis image;
    axis off;

case {'Circular'}
    rgb = imread('circularFlow.png');
    image(rgb); %title('Circular Channel')
    axis image;
    axis off;

case {'Triangular'}
    rgb = imread('triangularFlow.png');
    image(rgb); %title('Triangular Channel')
    axis image;
    axis off;

case {'U-Shape'}
    rgb = imread('ushapeFlow.png');
    image(rgb); %title('UshapeFlow Channel')
    axis image;
    axis off;

case {'Parabolic'}

    rgb = imread('parabolicFlow.png');
    image(rgb); %title('Parabolic Channel')
    axis image;
    axis off;

end

```

```

[A,P,R,V,b,y,f,T,Fr,N,stateOfFlow,computationStatus,itrexiflag] =
makafuiOpenChanelFlow(Q,N,n,S,Tb,phi,channelSection,shape,b);

if strcmp(computationStatus,'Failed')==1;
    [itrMsg] = iteratationStatus(itrexiflag);
    computation = 'Computation Failed. Please check your input parameters';
    set(handles.areaValue,'String','0.00');
    set(handles.wettedPerimeterValue,'String','0.00');
    set(handles.radiusValue,'String','0.00');
    set(handles.velocityValue,'String','0.00');
    set(handles.bedWidthOutputValue,'String','0.00');
    set(handles.flowDepthValue,'String','0.00');
    set(handles.freeBoardValue,'String','0.00');
    set(handles.topWidthValue,'String','0.00');
    set(handles.FroudeNumberValue,'String','0.00');
    set(handles.stateOfFlow,'String',(stateOfFlow));
    set(handles.userGuideTag,'Value',1);
    userGuideStr(1) = {'';
    userGuideStr(2) = {'';
    userGuideStr(3) = {computation};
    userGuideStr(4) = {'';
    userGuideStr(5) = {'';
    userGuideStr(6) = {'';
    userGuideStr(7) = {'Diagnostics message:'};
    userGuideStr(8) = {itrMsg};
    set(handles.guideTag,'Style','text','String',userGuideStr,'HorizontalAlignment','left')
else

    set(handles.areaValue,'String',sprintf('%0.6g',(A)));
    set(handles.wettedPerimeterValue,'String',sprintf('%0.6g',(P)));
    set(handles.radiusValue,'String',sprintf('%0.6g',(R)));
    set(handles.velocityValue,'String',sprintf('%0.6g',(V)));
    set(handles.bedWidthOutputValue,'String',sprintf('%0.6g',(b)));
    if b==0 ||
strcmp(get((get(handles.channelType,'SelectedObject')),'String'),'Triangular')==1 ||
strcmp(get((get(handles.channelType,'SelectedObject')),'String'),'Circular')==1 ||
strcmp(get((get(handles.channelType,'SelectedObject')),'String'),'Parabolic')==1
        set(handles.bedWidthOutputValue,'Visible','off');
        set(handles.bedWidthOutputUnit,'Visible','off');
        set(handles.bedWidthOutputTag,'Visible','off');
    end
    set(handles.flowDepthValue,'String',sprintf('%0.6g',(y)));
    set(handles.freeBoardValue,'String',sprintf('%0.6g',(f)));
    set(handles.topWidthValue,'String',sprintf('%0.6g',(T)));
    set(handles.FroudeNumberValue,'String',sprintf('%0.6g',(Fr)));
    set(handles.stateOfFlow,'String',(stateOfFlow));
end
else
    userGuideTag_Callback(hObject, eventdata, handles);
end
end

```

```

function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

% --- Executes during object creation, after setting all properties.
function channelType_CreateFcn(hObject, eventdata, handles)
% hObject    handle to channelType (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

```

```

% --- Executes on key press with focus on submitInputParaRequest and none of its controls.
function submitInputParaRequest_KeyPressFcn(hObject, eventdata, handles)
% hObject    handle to submitInputParaRequest (see GCBO)
% eventdata  structure with the following fields (see UICONTROL)
%   Key: name of the key that was pressed, in lower case
%   Character: character interpretation of the key(s) that was pressed
%   Modifier: name(s) of the modifier key(s) (i.e., control, shift) pressed
% handles    structure with handles and user data (see GUIDATA)
if strcmp(get((get(handles.channelSection,'SelectedObject')),'String'),'Lined &
Uneconomic')==1
    set(handles.bedWidthInputValue,'Visible','on');
    set(handles.bedWidthOutputValue,'String','0.00');
    set(handles.bedWidthOutputValue,'Visible','off');
end
makafuiOpenChanelFlow;

```

```

% --- Executes on selection change in listBox2.

```

```
function listBox2_Callback(hObject, eventdata, handles)
% hObject    handle to listBox2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listBox2 contents as cell array
%         contents{get(hObject,'Value')} returns selected item from listBox2
```

```
% --- Executes during object creation, after setting all properties.
function listBox2_CreateFcn(hObject, ~, handles)
% hObject    handle to listBox2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: listBox controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function flowRateValue_Callback(hObject, eventdata, handles)
% hObject    handle to flowRateValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of flowRateValue as text
Q = str2double(get(hObject,'String'));
set(handles.reset,'Visible','on'); userGuideTag_Callback(hObject, eventdata, handles);
```

```
% --- Executes during object creation, after setting all properties.
function flowRateValue_CreateFcn(hObject, eventdata, handles)
% hObject    handle to flowRateValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function slopeValue_Callback(hObject, eventdata, handles)
% hObject    handle to slopeValue (see GCBO)
```



```
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of slopeValue as text
S = str2double(get(hObject,'String'));
set(handles.reset,'Visible','on'); userGuideTag_Callback(hObject, eventdata, handles);
```

```
% --- Executes during object creation, after setting all properties.
function slopeValue_CreateFcn(hObject, ~, handles)
% hObject handle to slopeValue (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
```

```
function phiValue_Callback(hObject, eventdata, handles)
% hObject handle to phiValue (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of phiValue as text
phi = str2double(get(hObject,'String'));
set(handles.reset,'Visible','on'); userGuideTag_Callback(hObject, eventdata, handles);
```

```
% --- Executes during object creation, after setting all properties.
function phiValue_CreateFcn(hObject, ~, handles)
% hObject handle to phiValue (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
```

```
function TbValue_Callback(hObject, eventdata, handles)
% hObject handle to TbValue (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of TbValue as text
Tb = str2double(get(hObject,'String'));
set(handles.reset,'Visible','on'); userGuideTag_Callback(hObject, eventdata, handles);
```

```
% --- Executes during object creation, after setting all properties.
function TbValue_CreateFcn(hObject, ~, handles)
% hObject    handle to TbValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function NValue_Callback(hObject, eventdata, handles)
% hObject    handle to NValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of NValue as text
N = str2double(get(hObject,'Value'));
set(handles.reset,'Visible','on');
userGuideTag_Callback(hObject, eventdata, handles);
```

```
% --- Executes during object creation, after setting all properties.
function NValue_CreateFcn(hObject, eventdata, handles)
% hObject    handle to NValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
```



```
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a double
```

```
% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on selection change in inputMn.
function inputMn_Callback(hObject, eventdata, handles)
% hObject    handle to inputMn (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.reset,'Visible','on');
inputMn = str2double(get(hObject,'Value'));
val = get(handles.inputMn,'Value');
```

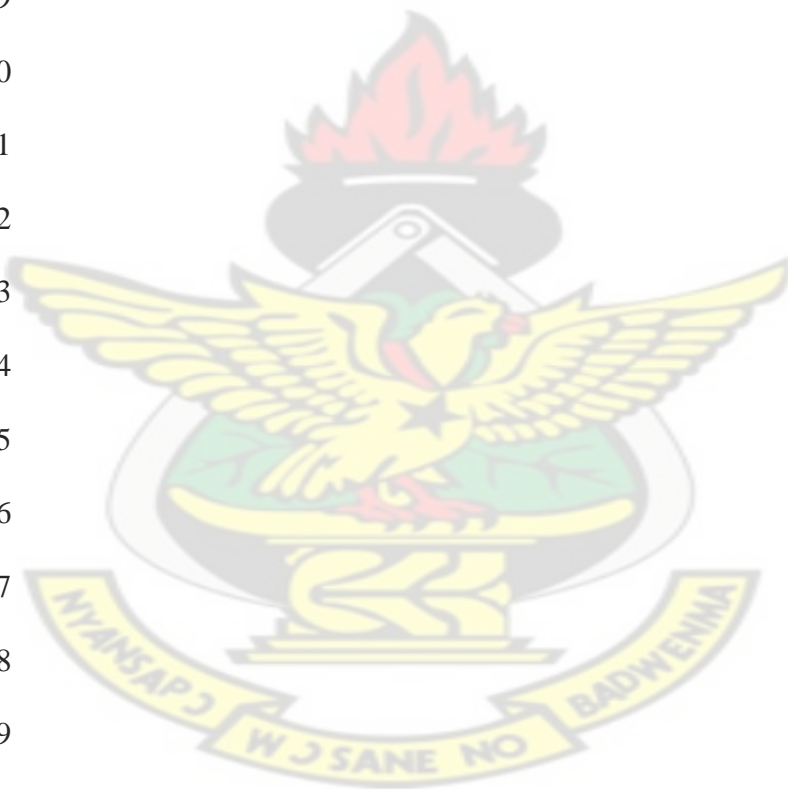
```
if val == 1
n = 999999 ;
elseif val == 2
n = 0.011 ;
elseif val == 3
n = 0.016 ;
elseif val == 4
n = 0.011 ;
elseif val == 5
n = 0.015 ;
elseif val == 6
n = 0.012 ;
elseif val == 7
n = 0.014 ;
elseif val == 8
n = 0.013 ;
elseif val == 9
n = 0.012 ;
elseif val == 10
n = 0.011 ;
```

```

elseif val == 11
n = 0.015 ;
elseif val == 12
n = 0.011 ;
elseif val == 13
n = 0.022 ;
elseif val == 14
n = 0.025 ;
elseif val == 15
n = 0.022 ;
elseif val == 16
n = 0.025 ;
elseif val == 17
n = 0.03 ;
elseif val == 18
n = 0.035 ;
elseif val == 19
n = 0.075 ;
elseif val == 20
n = 0.05 ;
elseif val == 21
n = 0.035 ;
elseif val == 22
n = 0.15 ;
elseif val == 23
n = 0.016 ;
elseif val == 24
n = 0.01 ;
elseif val == 25
n = 0.029 ;
elseif val == 26
n = 0.011 ;
elseif val == 27
n = 0.025 ;
elseif val == 28
n = 0.022 ;
elseif val == 29
n = 0.03 ;
elseif val == 30
n = 0.035 ;
elseif val == 31
n = 0.04 ;
elseif val == 32
n = 0.009 ;
elseif val == 33
n = (0.018 + 0.025)/2;
elseif val == 34
n = (0.009 + 0.015)/2;
elseif val == 35
n = (0.009 + 0.011)/2;

```

KNUST



```

elseif val == 36
n = 0.01 ;
elseif val == 37
n = 0.011 ;
elseif val == 38
n = 0.019 ;
elseif val == 39
n = 0.012 ;
elseif val == 40
n = 0.012 ;
elseif val == 41
n = 0.013 ;
elseif val == 42
n = 0.012 ;

end
set(handles.inputMnValue,'String',sprintf('%0.6g',(n)));

userGuideTag_Callback(hObject, eventdata, handles);
% Hints: contents = cellstr(get(hObject,'String')) returns inputMn contents as cell array

% --- Executes during object creation, after setting all properties.
function inputMn_CreateFcn(hObject, eventdata, handles)
% hObject handle to inputMn (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: listbox controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in close.
function close_Callback(hObject, eventdata, handles)
% hObject handle to close (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
close();

% --- Executes during object creation, after setting all properties.
function inputPanel_CreateFcn(hObject, eventdata, handles)
% hObject handle to inputPanel (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

```

```
function bedWidthInputValue_Callback(hObject, eventdata, handles)
% hObject    handle to bedWidthInputValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of bedWidthInputValue as text
set(handles.reset,'Visible','on'); userGuideTag_Callback(hObject, eventdata, handles);
```

```
% --- Executes during object creation, after setting all properties.
function bedWidthInputValue_CreateFcn(hObject, eventdata, handles)
% hObject    handle to bedWidthInputValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function bedWidthValueOutput_Callback(hObject, eventdata, handles)
% hObject    handle to bedWidthOutputValue (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of bedWidthOutputValue as text
%       str2double(get(hObject,'String')) returns contents of bedWidthOutputValue as a
double
```

```
% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on button press in pushbutton9.
function channelSelection_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```

function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as text
%        str2double(get(hObject,'String')) returns contents of edit13 as a double

% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of edit12 as a double

% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```



```

function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%        str2double(get(hObject,'String')) returns contents of edit11 as a double

% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%        str2double(get(hObject,'String')) returns contents of edit10 as a double

% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in pushbutton10.
function pushbutton10_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton10 (see GCBO)

```



```

% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on selection change in listbox4.
function listbox4_Callback(hObject, ~, handles)
% hObject    handle to listbox4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listbox4 contents as cell array
%         contents{get(hObject,'Value')} returns selected item from listbox4

% --- Executes during object creation, after setting all properties.
function listbox4_CreateFcn(hObject, ~, handles)
% hObject    handle to listbox4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%         str2double(get(hObject,'String')) returns contents of edit9 as a double

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```
% --- Executes during object creation, after setting all properties.
function uipanel21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to uipanel21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
function edit14_Callback(hObject, ~, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as text
%        str2double(get(hObject,'String')) returns contents of edit14 as a double
```

```
% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, ~, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on selection change in listbox5.
function listbox5_Callback(hObject, eventdata, handles)
% hObject    handle to listbox5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns listbox5 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from listbox5
```

```
% --- Executes during object creation, after setting all properties.
function listbox5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: listbox controls usually have a white background on Windows.
```

```

% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(~, eventdata, ~)
% hObject handle to pushbutton11 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

function edit15_Callback(hObject, ~, handles)
% hObject handle to edit15 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit15 as text
% str2double(get(hObject,'String')) returns contents of edit15 as a double

% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, ~, handles)
% hObject handle to edit15 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit16_Callback(hObject, eventdata, handles)
% hObject handle to edit16 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit16 as text
% str2double(get(hObject,'String')) returns contents of edit16 as a double

% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, ~, handles)

```

```
% hObject   handle to edit16 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit17_Callback(hObject, eventdata, handles)
% hObject   handle to edit17 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit17 as text
%       str2double(get(hObject,'String')) returns contents of edit17 as a double
```

```
% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject   handle to edit17 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit18_Callback(hObject, eventdata, handles)
% hObject   handle to edit18 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit18 as text
%       str2double(get(hObject,'String')) returns contents of edit18 as a double
```

```
% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject   handle to edit18 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
```

```

% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%    See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes when selected object is changed in channelType.
function channelType_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in channelType
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
shape =get((get(handles.channelType,'SelectedObject')),'String');

switch shape
    case {'Trapezoidal'}

% ax(1) = subplot(1,2,1);
rgb = imread('trapezoidalFlow.png');
image(rgb); %title('Trapezoidal Channel')
axis image;
axis off;

    case {'Circular'}
        rgb = imread('circularFlow.png');
        image(rgb); %title('Circular Channel')
        axis image;
        axis off;
    case {'Triangular'}
        rgb = imread('triangularFlow.png');
        image(rgb); %title('Triangular Channel')
        axis image;
        axis off;

    case {'U-Shape'}
        rgb = imread('ushapeFlow.png');
        image(rgb); %title('UshapeFlow Channel')
        axis image;
        axis off;

    case {'Parabolic'}

```



```

    rgb = imread('parabolicFlow.png');
    image(rgb); %title('Parabolic Channel')
    axis image;
    axis off;

    case {'Rectangular'}
        rgb = imread('rectangularFlow.png');
        image(rgb); %title('Rectangular Channel');
        axis image;
        axis off;
    end

    set(handles.inputPanel,'Visible','on');
    if strcmp(get(handles.inputPanel,'Visible'),'on')==1
        channelSection_SelectionChangeFcn(hObject, eventdata, handles);
    end

% --- Executes when selected object is changed in channelSection.
function channelSection_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in channelSection
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
shape =get((get(handles.channelType,'SelectedObject')),'String');
set(handles.NValue,'Value',1);
if strcmp(shape,'Rectangular')==1
    set(handles.NValue,'Value',2);
elseif strcmp(get(get(handles.channelSection,'SelectedObject'),'String'),'Lined &
Economic')==1
    set(handles.NValue,'Value',3);
end
switch get(get(handles.channelSection,'SelectedObject'),'String');
    case 'Lined & Economic'
        set(handles.rectangularTypeTag,'Visible','on');
        set(handles.circularTypeTag,'Visible','on');
        set(handles.ushapeTypeTag,'Visible','on');
        if strcmp(get((get(handles.channelType,'SelectedObject')),'String'),'Triangular')==1
            set(handles.bedWidthInputValue,'String','0.00');
            set(handles.bedWidthInputValue,'Visible','off');
            set(handles.bedWidthInputTag,'Visible','off');
            set(handles.bedWidthinputUnitTag,'Visible','off');
            set(handles.bedWidthOutputValue,'String','0.00');
            set(handles.bedWidthOutputValue,'Visible','off');
            set(handles.bedWidthOutputUnit,'Visible','off');
            set(handles.bedWidthOutputTag,'Visible','off');
        else
            set(handles.bedWidthInputValue,'String','0.00');

```



```

        set(handles.bedWidthInputValue,'Visible','off');
        set(handles.bedWidthInputTag,'Visible','off');
        set(handles.bedWidthinputUnitTag,'Visible','off');
        set(handles.bedWidthOutputValue,'Visible','on');
        set(handles.bedWidthOutputUnit,'Visible','on');
        set(handles.bedWidthOutputTag,'Visible','on');
    end
    set(handles.flowRateValue,'Visible','on');
    set(handles.NValue,'Visible','on');
    set(handles.NTag,'Visible','on');
    set(handles.NTagUnit,'Visible','on');
    set(handles.slopeValue,'Visible','on');
    set(handles.phiValue,'String','0.00');
    set(handles.phiValue,'Visible','off');
    set(handles.phiUnitTag,'Visible','off');
    set(handles.angleTag,'Visible','off');
    set(handles.phiUnitTag,'Visible','off');
    set(handles.shearStressTag,'Visible','off');
    set(handles.TbValue,'String','0.00');
    set(handles.TbValue,'Visible','off');
    set(handles.TbUnitTag,'Visible','off');
    set(handles.inputMn,'Visible','on');
    if strcmp(get((get(handles.channelType,'SelectedObject')),'String'),'Circular')==1
        set(handles.NValue,'Visible','off');
        set(handles.NTag,'Visible','off');
        set(handles.NTagUnit,'Visible','off');
    end

case 'Lined & Uneconomic'
    set(handles.rectangularTypeTag,'Visible','on');
    set(handles.circularTypeTag,'Visible','on');
    set(handles.ushapeTypeTag,'Visible','on');
    set(handles.flowRateValue,'Visible','on');
    set(handles.NValue,'Visible','on');
    set(handles.slopeValue,'Visible','on');
    set(handles.phiValue,'String','0.00');
    set(handles.phiValue,'Visible','off');
    set(handles.phiUnitTag,'Visible','off');
    set(handles.angleTag,'Visible','off');
    set(handles.TbValue,'String','0.00');
    set(handles.TbValue,'Visible','off');
    set(handles.TbUnitTag,'Visible','off');
    set(handles.shearStressTag,'Visible','off');
    set(handles.inputMn,'Visible','on');
    set(handles.NTag,'Visible','on');
    set(handles.NTagUnit,'Visible','on');

```

```

    if (strcmp(get((get(handles.channelType,'SelectedObject')), 'String'), 'Parabolic')==1 ||
        strcmp(get((get(handles.channelType,'SelectedObject')), 'String'), 'Triangular')==1 ||
        strcmp(get((get(handles.channelType,'SelectedObject')), 'String'), 'Circular')==1)
        set(handles.bedWidthInputValue, 'String', '0.00');
        set(handles.bedWidthInputValue, 'Visible', 'off');
        set(handles.bedWidthInputTag, 'Visible', 'off');
        set(handles.bedWidthinputUnitTag, 'Visible', 'off');
        set(handles.bedWidthOutputValue, 'String', '0.00');
        set(handles.bedWidthOutputValue, 'Visible', 'off');
        set(handles.bedWidthOutputUnit, 'Visible', 'off');
        set(handles.bedWidthOutputTag, 'Visible', 'off');
    else
        set(handles.bedWidthInputValue, 'Visible', 'on');
        set(handles.bedWidthInputTag, 'Visible', 'on');
        set(handles.bedWidthinputUnitTag, 'Visible', 'on');
        set(handles.bedWidthOutputValue, 'String', '0.00');
        set(handles.bedWidthOutputValue, 'Visible', 'off');
        set(handles.bedWidthOutputUnit, 'Visible', 'off');
        set(handles.bedWidthOutputTag, 'Visible', 'off');
    end
    if strcmp(get((get(handles.channelType,'SelectedObject')), 'String'), 'Circular')==1
        set(handles.NValue, 'Visible', 'off');
        set(handles.NTag, 'Visible', 'off');
        set(handles.NTagUnit, 'Visible', 'off');
    else
        set(handles.NValue, 'Visible', 'on');
        set(handles.NTag, 'Visible', 'on');
        set(handles.NTagUnit, 'Visible', 'on');
    end
end

case 'Unlined'
    set(handles.rectangularTypeTag, 'Visible', 'off');
    set(handles.circularTypeTag, 'Visible', 'off');
    set(handles.ushapeTypeTag, 'Visible', 'off');
    if strcmp(get((get(handles.channelType,'SelectedObject')), 'String'), 'Triangular')==1
        set(handles.bedWidthInputValue, 'String', '0.00');
        set(handles.bedWidthInputValue, 'Visible', 'off');
        set(handles.bedWidthInputTag, 'Visible', 'off');
        set(handles.bedWidthinputUnitTag, 'Visible', 'off');
        set(handles.bedWidthOutputValue, 'String', '0.00');
        set(handles.bedWidthOutputValue, 'Visible', 'off');
        set(handles.bedWidthOutputUnit, 'Visible', 'off');
        set(handles.bedWidthOutputTag, 'Visible', 'off');
    else
        set(handles.bedWidthInputValue, 'String', '0.00');
        set(handles.bedWidthInputValue, 'Visible', 'off');
        set(handles.bedWidthInputTag, 'Visible', 'off');
        set(handles.bedWidthinputUnitTag, 'Visible', 'off');
        set(handles.bedWidthOutputValue, 'Visible', 'on');
        set(handles.bedWidthOutputUnit, 'Visible', 'on');
    end
end

```

```

        set(handles.bedWidthOutputTag,'Visible','on');
    end

    set(handles.flowRateValue,'Visible','on');
    set(handles.NValue,'Visible','on');
    set(handles.slopeValue,'Visible','on');
    set(handles.phiValue,'Visible','on');
    set(handles.phiUnitTag,'Visible','on');
    set(handles.angleTag,'Visible','on');
    set(handles.TbValue,'Visible','on');
    set(handles.TbUnitTag,'Visible','on');
    set(handles.shearStressTag,'Visible','on');
    set(handles.inputMn,'Visible','on');
    otherwise
        set(handles.bedWidthInputValue,'String','0.00');
        set(handles.bedWidthInputValue,'Visible','off');
        set(handles.bedWidthOutputValue,'Visible','on');
    end

    set(handles.channelType,'Visible','on');

    % --- Executes during object creation, after setting all properties.
    function bedWidthOutputValue_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to bedWidthOutputValue (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns called

    % --- Executes on button press in userGuideTag.
    function userGuideTag_Callback(hObject, eventdata, handles)
    % hObject    handle to userGuideTag (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    guideRequest = get(handles.userGuideTag,'Value');
    userGuideStr(1) = {'';
    userGuideStr(2) = {'';
    userGuideStr(3) = {'';
    userGuideStr(4) = {'';
    userGuideStr(5) = {'';
    userGuideStr(6) = {'';
    userGuideStr(7) = {'';
    userGuideStr(8) = {'';
    if guideRequest == get(hObject,'Min')

    elseif guideRequest == get(hObject,'Max')
        if isequal(get(handles.reset,'Visible'),'off')

```

```

userGuideStr(1) = {'The open flow system computes flow output parameters based on
the desired channel lining and channel type. On selecting the channel type a sketch of the
shape is displayed on the sketch of channel section of the screen.'};
userGuideStr(2) = {''};
userGuideStr(3) = {'The manning roughness coefficient (n) and the side slope (N) data
are selected from the input parameter section. Other assigned parameters based on the lining
selected must be entered in the input parameter section of the screen.'};
userGuideStr(4) = {'The result is outputted at the output parameter section of the
screen.'};
userGuideStr(5) = {''};
userGuideStr(6) = {'With one input parameter entered, a RESET button pops up for
resetting all parameters to their default values.'};
userGuideStr(7) = {''};
userGuideStr(8) = {''};
set(handles.guideTag,'Style','text','String',userGuideStr,'HorizontalAlignment','left')
else
[Q,N,n,S,Tb,shape,~,phi,b,allInputsDone] = readInputParameters(hObject, eventdata,
handles);
if ~allInputsDone==1
if (strcmp(get(handles.bedWidthInputValue,'Visible'),'on')==1 && isequal(b,0)==1);
userGuideStr(1) = {'Enter Bed Width (b)'};
end

if (strcmp(get(handles.flowRateValue,'Visible'),'on')==1 && isequal(Q,0.00));
userGuideStr(2) = {'Enter Flow Rate (Q)'};
end
if (strcmp(get(handles.NValue,'Visible'),'on')==1 && isequal(N,9999999999)==1);
userGuideStr(3) = {'Enter N'};
end
if (strcmp(get(handles.slopeValue,'Visible'),'on')==1 && isequal(S,0)==1);
userGuideStr(4) = {'Enter Slope (S)'};
end
if (strcmp(get(handles.phiValue,'Visible'),'on')==1 && isequal(phi,0)==1);
userGuideStr(5) = {'Enter angle (phi)'};
end
if (strcmp(get(handles.TbValue,'Visible'),'on')==1 && isequal(Tb,0)==1);
userGuideStr(6) = {'Enter Shear Stress (Tb)'};
end
if (strcmp(get(handles.inputMn,'Visible'),'on')==1 && isequal(n,0)==1);
userGuideStr(7) = {'Enter Mannings Number (n)'};
end
if ~(strcmp(shape,'Trapezoidal')==1 || ...
strcmp(shape,'Rectangular')==1 || ...
strcmp(shape,'Parabolic')==1 || ...
strcmp(shape,'Triangular')==1 || ...
strcmp(shape,'U-Shape')==1 || ...
strcmp(shape,'Circular')==1 && ...
strcmp(get(handles.channelType,'Visible'),'on')==1)
userGuideStr(8) = {'Select Channel Type'};
end

```



```

else
    userGuideStr(1) = {'';
    userGuideStr(2) = {'';
    userGuideStr(3) = {'Click CALCULATE in the INPUT PARAMETERS section to
compute the output parameters, OR'};
    userGuideStr(4) = {'';
    userGuideStr(5) = {'Click RESET in the INPUT PARAMETERS section to reset
parameters to their default values'};
    userGuideStr(6) = {'';
    userGuideStr(7) = {'';
    userGuideStr(8) = {'';
end
end

end
set(handles.guideTag,'Style','text','String',userGuideStr,'HorizontalAlignment','left');

```

```

function [Q,N,n,S,Tb,shape,channelSection,phi,b,allInputsDone] =
readInputParametrs(hObject, eventdata, handles)
% hObject    handle to submitInputValues (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
Q = str2double(get(handles.flowRateValue,'String'));
N = str2double(get(handles.NValue,'String'));
    val = get(handles.NValue,'Value');
    if val == 1
        N=9999999999;
    elseif val == 2
        N= 0;
    elseif val == 3
        N= 1;
    elseif val == 4
        N= 1.5;
    elseif val == 5
        N= 2;
    elseif val == 6
        N= 2.5;
    elseif val == 7
        N= 3;
    end
end

```

```

S = str2double(get(handles.slopeValue,'String'));
phi = str2double(get(handles.phiValue,'String'))/360*2*pi;
Tb = str2double(get(handles.TbValue,'String'));
n = str2double(get(handles.inputMnValue,'String'));
b= str2double(get(handles.bedWidthInputValue,'String'));
shape =get((get(handles.channelType,'SelectedObject'),'String');
channelSection =get((get(handles.channelSection,'SelectedObject'),'String');

```

```

allInputsDone=~((isequal(Q,0)==1 &&
strcmp(get(handles.flowRateValue,'Visible'),'on')==1 || ...
    (isequal(n,0)==1 && strcmp(get(handles.inputMn,'Visible'),'on')==1 || ...
    (isequal(S,0)==1 && strcmp(get(handles.slopeValue,'Visible'),'on')==1 ||...
    (isequal(Tb,0)==1 && strcmp(get(handles.TbValue,'Visible'),'on')==1 || ...
    (isequal(N,999999999)==1 && strcmp(get(handles.NValue,'Visible'),'on')==1 || ...
    ~((strcmp(shape,'Trapezoidal')==1 || ...
    strcmp(shape,'Rectangular')==1 || ...
    strcmp(shape,'Parabolic')==1 ||...
    strcmp(shape,'Triangular')==1 || ...
    strcmp(shape,'U-Shape')==1 ||...
    strcmp(shape,'Circular')==1) ...
    && strcmp(get(handles.channelType,'Visible'),'on')==1 )|| ...
    (strcmp(channelSection,[])&&
(strcmp(get(handles.channelSectionValue,'Visible'),'on')==1 || ...
    (isequal(phi,0)&& strcmp(get(handles.phiValue,'Visible'),'on')==1 || ...
    (isequal(b,0)&& strcmp(get(handles.bedWidthInputValue,'Visible'),'on')==1)));

```

```

function clearUserGuideScreen(hObject,eventdata,handles)
    set(handles.userGuideTag,'Value',1);
    userGuideStr(1) = {'';
    userGuideStr(2) = {'';
    userGuideStr(3) = {'';
    userGuideStr(4) = {'';
    userGuideStr(5) = {'';
    userGuideStr(6) = {'';
    userGuideStr(7) = {'';
    userGuideStr(8) = {'';
    set(handles.guideTag,'Style','text','String',userGuideStr,'HorizontalAlignment','left')

```

```

function [itrMsg] = iteratIterationStatus(itrexitflag)

```

```

switch itrexitflag

```

```

    case 1

```

```

        itrMsg='Function converged to a solution , but  $V > 1.5\text{m/s}$ ';

```

```

    case -1

```

```

        itrMsg='Algorithm was terminated by the output function.';

```

```

    case -3

```

```

        itrMsg = 'NaN or Inf function value was encountered during search for an interval
containing a sign change.';

```



case -4

itrMsg='Complex function value was encountered during search for an interval containing a sign change.';

case -5

itrMsg = 'Might have converged to a singular point.';

case -6

itrMsg='Cannot detect a change in sign of the function.';

case 'NOB0'

itrMsg = 'N cannot be zero in this instance';

case 'Bad Start up'

itrMsg='Complex Start up point encountered in optimization'

case 'Negb'

itrMsg = 'b computes to a negative value';

case 'Negy'

itrMsg = 'y computes to a negative value';

end

function inputMnValue\_Callback(hObject, eventdata, handles)

% hObject handle to inputMnValue (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of inputMnValue as text

% str2double(get(hObject,'String')) returns contents of inputMnValue as a double

% --- Executes during object creation, after setting all properties.

function inputMnValue\_CreateFcn(hObject, eventdata, handles)

% hObject handle to inputMnValue (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.

% See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),

get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

## Open Channel Flow File

**FILENAME: MakafuiOpenChannelFlow.m**

```
function [A,P,R,V,b,y,f,T,Fr,N,stateOfFlow,computationStatus,itexitflag] =  
makafuiOpenChanelFlow(Q,N,n ,S,Tb,phi,chanelSection,shape,b)
```

%NEEDED STARTING PARAMETERS

g=9.80;

V=2.00; % set initial V above the threshold as starting point for the iteration.

switch (chanelSection)

    %if chanelSection =='Hydraulic Efficient'

    case 'Lined & Economic'

$y = (2^{2/3} * Q * n * S^{(-1/2)} / (2 * \sqrt{N^2 + 1} - N))^{3/8}$ ;

        exitflag=1;

        localitexitflag=exitflag;

    switch (shape);

        case {'Rectangular'}

            N=0;

            b=2\*y;

            A=2\*y^2;

            P=4\*y;

            T=b;

        case {'Triangular'}

            b=0;

            N=1;

            A=y^2;

            P= 2\*y;

            T=2\*N\*y;

        case {'Trapezoidal'}

            b = 2\*y\*(sqrt(N^2+1)-N);

            A= y^2\*(2\*sqrt(N^2+1)-N);

            P=2\*y\*(2\*sqrt(N^2+1)-N);

            T=b + 2\*N\*y;

        case {'Circular'}

            r= 0.665\*(Q\*n/S^(1/2));

            beta= 2\*acos(0);

            A=(r^2)\*(beta - sin(beta));

            P= 2\*r\*beta;

            T = 2\*r;

        case {'U-Shape'}

            r=0.3175\*y;

            b=2\*r;

            A=0.04838\*(y^2);

```

P= 2.3625*y;
T = b;

case {'Parabolic'}
    A=1.886*(y^2);
    P= 3.771*y;
    T=4*y*N;

end

R=A/P;

V= 1/n*(R^(2/3))*S^(1/2);
V=min(V);

case 'Lined & Uneconomic'

V=2.00; % set initial V above the threshold as starting point for the iteration.
y=.001; % y cannot be zero
if( N==0 && b==0)
    localitrexiflag='N0B0';
    [A,P,R,V,b,y,f,T,Fr,stateOfFlow,computationStatus,itrexiflag] =
cleanExit(localitrexiflag);
    return;

end

y1=y-(((Q*n/S.^(1/2))*(b+2*y*sqrt(N.^2+1)).^(2/3)).^(3/5))/(b+N*y);
if isreal(y1)==0
    localitrexiflag='Bad Start up';
    [A,P,R,V,b,y,f,T,Fr,stateOfFlow,computationStatus,itrexiflag] =
cleanExit(localitrexiflag);
    return;
end

while ~(y1>-1 & y1<1) ;
    y=y+0.1;
    y1=y-(((Q*n/S.^(1/2))*(b+2*y*sqrt(N.^2+1)).^(2/3)).^(3/5))/(b+N*y);
    if isreal(y1)==0
        localitrexiflag='Bad Start up';
        [A,P,R,V,b,y,f,T,Fr,stateOfFlow,computationStatus,itrexiflag] =
cleanExit(localitrexiflag);
        return;
    end
end

while ~(y1>-0.02 && y1<0.02);
    y=y+0.0001;
    y1=y-(((Q*n/S.^(1/2))*(b+2*y*sqrt(N.^2+1)).^(2/3)).^(3/5))/(b+N*y);

```

```

end
switch (shape)
    case {'Rectangular'}
        N=0;
        A=b*y;
        P= b+2*y;
        T=b;

    case {'Triangular'}
        b=0;
        A=N*y^2;
        P= 2*y*sqrt(N^2+1);
        T=2*N*y;

    case {'Trapezoidal'}

        A=b.*y+N.*y^2;
        P= b+2*y*sqrt(N^2+1);
        T=2*N*y;

    case {'Circular'}
        r=0.665*(Q*n/S^(1/2));
        beta=2*acos(1-y/r);
        A=(r^2)*(beta - sin(beta));
        P= 2*r*beta;
        T = 2*r;

    case {'U-Shape'}
        r=b/2;
        A=b*(y-b/2)+pi*b^2/8;
        P= 2*(y-b/2)+pi*b/2;
        T = b;

    case {'Parabolic'}
        A=1.886.*(y^2);
        P= 3.771*y;
        T=4*y*N;

end
R=A/P;
V= 1/n*(R^(2/3))*S^(1/2);

case 'Unlined'
    theta= atan(1/N);
    F= sqrt(1-sin(theta)^2/sin(phi)^2);
    Tcs = F*Tb;
    Ro=1000;

```

```

y = Tcs/(0.76*Ro*g*S);
b=0.01;
b1=b- (Q*n*S^(-1/2)*(b+2*y*sqrt(N^2+1))^(2/3)/(y*(b*y+N*y^2)^(2/3))-N*y);
if isreal(b1)==0
    localitrexiflag='Bad Start up';
    [A,P,R,V,b,y,f,T,Fr,stateOfFlow,computationStatus,itrexiflag] =
cleanExit(localitrexiflag);
    return;
end
while ~(b1>-1 && b1<1);
    b=b+0.005;
    b1=b- (Q*n*S^(-1/2)*(b+2*y*sqrt(N^2+1))^(2/3)/(y*(b*y+N*y^2)^(2/3))-N*y);
    if isreal(b1)==0

        localitrexiflag='Bad Start up';
        [A,P,R,V,b,y,f,T,Fr,stateOfFlow,computationStatus,itrexiflag] =
cleanExit(localitrexiflag);
        return;
    end
end

while ~(b1>-0.02 && b1<0.02);
    b=b+0.01;
    b1=b- (Q*n*S^(-1/2)*(b+2*y*sqrt(N^2+1))^(2/3)/(y*(b*y+N*y^2)^(2/3))-N*y);
end
switch shape
    case {'Trapezoidal'}
        A=b*y+N*y^2;
        P= b+2*y*sqrt(N^2+1);
        T=b+2*N*y;
    case {'Triangular'}
        b=0;
        A=N*y^2;
        P= 2*y*sqrt(N^2+1);
        T=2*N*y;

    case {'Parabolic'}
        T=4*y*N;
        A=0.866*(y*T);
        P= T+2.66*y^2/T;
        b=0;

end
R=A/P;

V= 1/n*R^(2/3)*S^(1/2);
end
localitrexiflag=1;

```

```

    itrexitflag=1;
    if V>1.5

        [A,P,R,V,b,y,f,T,Fr,stateOfFlow,computationStatus,itrexitflag] =
        cleanExit(localitrexitflag);

    else
        computationStatus = 'OK';
        f = 0.2*y;
        D = A/T;
        Fr=V/sqrt(g*D);
        if Fr<1
            stateOfFlow='Subcritical';
        else if Fr==1
            stateOfFlow='Critical';

        else if Fr>1
            stateOfFlow='Critical';
        end
        end
    end
end

function [A,P,R,V,b,y,f,T,Fr,stateOfFlow,computationStatus,itrexitflag] =
cleanExit(localitrexitflag)
computationStatus = 'Failed';
stateOfFlow='';
f=0;
D=0;
Fr=0;
A=0;
P=0;
R=0;
V=0;
b=0;
y=0;
T=0;
itrexitflag=localitrexitflag;

f=0;
Fr=0;

```