

**A HYDROGRAPH-BASED PREDICTION OF
MEANDER MIGRATION**

A Dissertation

by

WEI WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2006

Major Subject: Civil Engineering

**A HYDROGRAPH-BASED PREDICTION OF
MEANDER MIGRATION**

A Dissertation

by

WEI WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,
Committee Members,

Head of Department,

Jean-Louis Briaud
Hamn-Ching Chen
Kuang-An Chang
Hongbin Zhan
David Rosowsky

May 2006

Major Subject: Civil Engineering

ABSTRACT

A Hydrograph-based Prediction of Meander Migration. (May 2006)

Wei Wang, B.S., Tongji University, Shanghai, China;

M.S., Tongji University, Shanghai, China

Chair of Advisory Committee: Dr. Jean-Louis Briaud

Meander migration is a process in which water flow erodes soil on one bank and deposits it on the opposite bank creating a gradual shift of the bank line over time. For bridges crossing such a river, the soil foundation of the abutments may be eroded away before the designed lifetime is reached. For highways parallel to and close to such a river, the whole road may be eaten away. This problem is costing millions of dollars to TxDOT in protection of affected bridges and highway embankments. This research is aimed at developing a methodology which will predict the possible migration of a meander considering the design life of bridges crossing it and highways parallel to it. The approaches we use are experimental tests, numerical simulation, modeling of migration, risk analysis, and development of a computer program.

Experimental tests can simulate river flow in a controlled environment. Influential parameters can be chosen, adjusted, and varied systematically to quantify their influence on the problem. The role of numerical simulation is to model the flow field and the stress field at the soil-water interface. Migration modeling is intended to integrate the results of experimental tests and numerical simulations and to develop a model which can make predictions. The Hyperbolic Model is used and its two major components M_{\max} equation and τ_{\max} equation are developed. Uncertainties in the parameters used for prediction make deterministic prediction less meaningful. Risk analysis is used to make the prediction based on a probabilistic approach. Hand calculation is too laborious to apply these procedures. Thus the development of a user friendly computer program is needed to automate the calculations.

Experiments performed show that the Hyperbolic Model matches the test data well and is suitable for the prediction of meander migration. Based on analysis of shear

stress data from numerical simulation, the τ_{\max} equation was derived for the Hyperbolic Model. Extensive work on the simplification of river geometry produced a working solution. The geometry of river channels can be automatically simplified into arcs and straight lines. Future hydrograph is critical to risk analysis. Tens of thousands of hydrographs bearing the same statistical characteristics as in history can be generated. The final product that can be directly used, the MEANDER program, consists of 11,600 lines of code in C++ and 2,500 lines of code in Matlab, not including the part of risk analysis. The computer program is ready for practice engineers to make predictions based on the findings of this research.

ACKNOWLEDGMENTS

I am beholden to my advisor, Dr. Jean-Louis Briaud, for the guidance and wonderful working environment he has provided. Dr. Briaud is a forerunner in many fields of Geotechnical Engineering and scour related subjects and has made significant contributions in adding to the greatness of the Department of Civil Engineering at Texas A&M University. I want to express my respect and admiration to Dr. Briaud for his achievements as a professional and for his success as a family man. Dr. Briaud belongs to the category of people who are smart and nice.

I want to thank Dr. Hamn-Ching Chen and Dr. Kuang-An Zhang, who are team members of the research project, for their guidance in my work and their insights about the project. I also want to thank Dr. Hongbin Zhan for his consistent support and critical review of my dissertation.

Many thanks go to Johnnie Reed of Hydraulic Engineering Lab, Matt Potter and Jeff Perry of the High-Bay Lab. They have graciously provided help in the setting up and running of flume test.

I applied to Texas A&M University because a certain person was here. He has been impressive for his exemplary work ethic and outstanding achievements. I have been enjoying discussions with him on many issues for years. My deep appreciation goes to this good friend of mine and my college classmate, Yiwen Cao.

I wish to express special appreciation to my friends and fellow students: Yuanyuan Ding, Diqing Lou, Dr. Ya Li, Dr. Xiong Zhang, Juanyu Liu, Dr. Yanfeng Li, Dr. Jun Wang, Zhigang Yao, Namgyu Park, Po-Hung Yeh, Xiaoyan Long, Junying Pan, Dr. Wentao Dai, Dr. Han Shi, Hongrui Hu, Xingnian Chen, Jinquan Zhong. It has been a wonderful experience to work and study with them. Ya Li has been a great help to my flume test. The discussions with Xiong Zhang helped me better understand unsaturated soil mechanics. I also wish to thank Jinming Xu from the Department of Mechanical Engineering, Guobin He and Guangtong Cao from the Department of Computer Science, and Shanfeng Chen and Xiang Lu from the Department of Electrical Engineering for their help in Visual C++ programming.

I want to acknowledge with deep appreciation the efforts in proofreading this dissertation provided by my friends: Stanley Mathew, Juanyu Liu, Jinquan Zhong, Li Liu and Ermilo Richer. Stan and Juanyu proofread several chapters and their work was very impressive. Jinquan's advice on both the probabilistic background about Chapter VII and my English language is appreciated.

TABLE OF CONTENTS

	Page
ABSTRACT	iii
ACKNOWLEDGMENTS.....	v
TABLE OF CONTENTS	vii
LIST OF FIGURES.....	x
LIST OF TABLES	xvi
CHAPTER	
I INTRODUCTION TO AND FUNDAMENTAL CONCEPTS OF MEANDER MIGRATION	1
1.1 Introduction	1
1.2 Fundamental Concept about Meander Migration.....	4
1.3 Factors Affecting Meander Migration.....	13
II EXISTING KNOWLEDGE OF MEANDER MIGRATION	15
2.1 General Approaches	15
2.2 Selected Empirical Methods.....	16
2.3 Selected Numerical Methods	30
2.4 SRICOS-EFA Method.....	39
III RESEARCH OBJECTIVES AND METHODOLOGY	49
3.1 Research Objectives	49
3.2 Methodology	49
IV FLUME TEST.....	54
4.1 Experimental Setup	54
4.2 Measurement of Velocity and Geometry Change	58
4.3 Calibration of the Flow Meter	63
4.4 Test Procedure.....	67
4.5 Sample Test Output.....	69
V THE APPLICATION OF SRICOS-EFA METHOD IN THE PREDICTION OF MEANDER MIGRATION	71

CHAPTER	Page
5.1	The Existence of Maximum Migration M_{\max} 71
5.2	Developing Equations for M_{\max} 74
5.3	Developing Equations for τ_{\max} 75
5.4	Application of the SRICOS-EFA Method 84
VI	GEOMETRY STUDY 87
6.1	Fit a Circle for a Given Group of Points 87
6.2	Calculate Radius of Curvature of Each Point on a Curve 92
6.3	Identify Bends for Which Circles Will be Fitted 100
6.4	At a Certain Bend Find the Best Fit Circle 104
6.5	Calculate Bend Angle..... 111
VII	FUTURE HYDROGRAPHS 112
7.1	Introduction 112
7.2	The Distribution of Daily Discharge 113
7.3	Definition of 100/500-year Flood 118
7.4	Compute Discharge Distribution Based on 100/500-year Flood 119
7.5	The Determination of 100/500-year Flood..... 121
7.6	Random Number Generation 124
7.7	Risk Analysis..... 126
VIII	THE MEANDER PROGRAM..... 127
8.1	Introduction 127
8.2	Graphic User Interface (GUI) 128
8.3	Center Line Method versus Bank Method 140
8.4	Mixed Language Programming—C++ and Matlab 144
8.5	An Overview of the Implementation of the Program..... 148
IX	VERIFICATION AND PARAMETRIC STUDY 155
9.1	Verification of Flume Tests..... 155
9.2	Preliminary Verification of the Case of the Brazos River 169
9.3	Parametric Study 175
9.4	Conclusion..... 181
X	CONCLUSIONS AND RECOMMENDATIONS 183
10.1	Conclusions 183
10.2	Recommendations for Future Research 186

	Page
REFERENCES	191
APPENDIX A FLUME TESTS CONDUCTED BY THE AUTHOR IN THE OLD HYDRO LAB	197
APPENDIX B A GALLERY OF PICTURES OF FLUME TESTS IN THE OLD HYDRO LAB	210
APPENDIX C SOURCE CODE OF THE COMPUTER PROGRAM	222
VITA	291

LIST OF FIGURES

	Page
Figure 1.1 A meandering river (courtesy of Dr. Jean-Louis Briaud)	1
Figure 1.2 Satellite picture of a section of Mississippi River (maps.google.com)	2
Figure 1.3 Trinity River at Highway 787 (Briaud et al. 2001)	3
Figure 1.4 Channel pattern classification devised by Brice (Brice, 1975)	5
Figure 1.5 Stream properties for classification and stability assessment (Brice and Blodgett, 1978)	6
Figure 1.6 Channel classification showing stability and types of hazards encountered with each pattern (Shen et al., 1981)	7
Figure 1.7 Interrelation between stream form and slope (Richardson et al. 1990).....	8
Figure 1.8 Interrelation between stream form, channel bed slope, and mean discharge (after Lane 1957).....	9
Figure 1.9 Bank erosion process (Nagata et al., 2000)	10
Figure 1.10 Modes of meander loop behavior (Brice, 1975)	11
Figure 1.11 Geometry parameters for meanders (after Briaud et al., 2001b).....	14
Figure 2.1 Graph for determining rate of meander migration (Keady and Priest 1977; Briaud, 2001b).....	18
Figure 2.2 Relation between catchment area and migration rate (Hooke, 1980; Briaud et al., 2001c)	19
Figure 2.3 The relationship between migration rate and channel width (Brice, 1982; Briaud et al., 2001b).....	21
Figure 2.4 The relationship between migration rate and geometry (Nanson and Hickin, 1983; Briaud et al., 2001b).....	22
Figure 2.5 Banklines and circles drawn along outer bankline positions for a hypothetical channel in 3 different years (Lagasse et al., 2004)	25
Figure 2.6 Diagram defining the outer bank radius of curvature in Years 1, 2, and 3 (R_{c1} , R_{c2} , and R_{c3}) and the amount (D_A and D_B) and direction (θ_A and θ_B) of migration of bend centroid during Periods A and B	26

	Page
Figure 2.7 Predicted position and radius of curvature of the circle that defines the outer bank of the hypothetical channel in Year 4 (Lagasse, et al., 2004) ...	26
Figure 2.8 Definition of the time sequence maps and extrapolation method (Briaud et al., 2001b).....	27
Figure 2.9 Cumulative percentage of extension migration (Lagasse et al., 2004)	28
Figure 2.10 Cumulative percentage of translation migration (Lagasse et al., 2004).....	29
Figure 2.11 Temporal changes in plan forms (Nagata et al., 2000)	32
Figure 2.12 Temporal changes in cross-sectional profiles for Run 2 (Nagata et al., 2000).....	33
Figure 2.13 Simulation of meandering channel widening due to bank erosion (Duan et al., 2001).....	35
Figure 2.14 Comparison of experimental and simulated results (Duan et al., 2001)	36
Figure 2.15 Simulated meandering process of laboratory case (Olsen, 2003)	37
Figure 2.16 EFA: (a) Conceptual diagram; (b) Photograph of test section (Briaud et al., 2001a).....	41
Figure 2.17 The Moody Chart (Munson et al., 1990).....	42
Figure 2.18 Erosion curve for coarse sand (Briaud et al. 2001a)	43
Figure 2.19 Variation of shear stress at bottom of scour hole as function of depth of scour hole (Briaud et al., 1999)	45
Figure 2.20 Scour depth versus time curve (Briaud et al., 1999)	46
Figure 2.21 Accumulation of scour depth: (1) Additional erosion occurs; (2) No additional erosion occurs.....	48
Figure 4.1 Experimental setup for flume test in the Old Hydro Lab	55
Figure 4.2 Upstream false bottom	56
Figure 4.3 Carving the channel.....	57
Figure 4.4 A finished channel.....	58
Figure 4.5 Diagram of a 2D ADV (Li, 2002)	59
Figure 4.6 Sample ADV data from flume test 9 in the old Hydro Lab (V_x)	59

	Page
Figure 4.7 Sample ADV data from flume test 9 in the old Hydro Lab (V_y)	60
Figure 4.8 Measurement of the slope of water surface and channel bend.....	61
Figure 4.9 A picture for applying photogrammetry technique	63
Figure 4.10 Flow meter: (a) Flow sensor; (b) Flow transmitter (source: www.dataindustrial.com)	64
Figure 4.11 Calibration of the flow meter--voltage versus time curve.....	65
Figure 4.12 Calibration of the flow meter--calibration curve.....	66
Figure 4.13 Result of flume test 7	70
Figure 5.1 Fitting hyperbolic curves for flume test data	73
Figure 5.2 Fitting hyperbolic curve for Des Moines River	74
Figure 5.3 $R/W=4$ $\phi=120^\circ$	76
Figure 5.4 Simulated shear stress along a channel	77
Figure 5.5 Influence of R/W	79
Figure 5.6 Influence of ϕ angle.....	79
Figure 5.7 Extreme value distribution	80
Figure 5.8 Simulated shear stress and fitted curves.....	81
Figure 5.9 Influence of R/W on location parameter	82
Figure 5.10 Influence of ϕ angle on location parameter.....	83
Figure 6.1 Using Matlab to fit a circle for a straight line	89
Figure 6.2 Comparison between original curve and smoothed curve.....	93
Figure 6.3 Comparison of parabolic fittings of different orders.....	95
Figure 6.4 Comparison of errors of polynomial fitting of different orders	95
Figure 6.5 Comparison of R_s of different fitting orders	96
Figure 6.6 R/W vs. Channel length, segment length=1.1W	98
Figure 6.7 R/W vs. Channel length, segment length=4.0W	99
Figure 6.8 R/W vs. Channel length, segment length=12.1W	99
Figure 6.9 A case showing sensitivity of criterion lines.....	101
Figure 6.10 Criterion line method	102

	Page
Figure 6.11 The second derivative of R vs. channel length.....	103
Figure 6.12 Visual comparison of fittings	105
Figure 6.13 Extension of bend boundaries	106
Figure 6.14 The influence of parameter b.....	110
Figure 6.15 Same b coefficient applied to all bends.....	110
Figure 7.1 Hydrograph of Guadalupe River Gauge Station 08176500	112
Figure 7.2 PDF of original data and fitted distribution – Guadalupe River	113
Figure 7.3 CDF of original data and fitted distribution – Guadalupe River.....	114
Figure 7.4 Hydrograph of Woodrow Wilson Bridge.....	115
Figure 7.5 PDF of original and fitted distribution	115
Figure 7.6 CDF of original data and fitted distribution	116
Figure 7.7 Probability of exceedance curves - Guadalupe River	122
Figure 7.8 Probability of exceedance curves – Woodrow Wilson Bridge.....	123
Figure 8.1 The main interface of the MEANDER program	128
Figure 8.2 Choose a unit.....	129
Figure 8.3 Digitize river banks with WinDIG program	129
Figure 8.4 Geometry input.....	132
Figure 8.5 Original channel and fitted circles for the center line of flume test 15.....	132
Figure 8.6 R/W versus channel lengthwise distance for the center line of flume test 15	133
Figure 8.7 Soil data input.....	134
Figure 8.8 Water data input	135
Figure 8.9 Input tables	136
Figure 8.10 Input plots.....	136
Figure 8.11 Output table	137
Figure 8.12 Output plots dialogue	138
Figure 8.13 Trace of a migrating center line	138
Figure 8.14 Predicted and measured banks by using Center Line Method	139

	Page
Figure 8.15 Migration versus time plot for a certain point.....	140
Figure 8.16 Trace of migrating banks.....	143
Figure 8.17 Predicted and measured banks by using Bank Method.....	143
Figure 8.18 Original channel and fitted circles for the left bank of flume test 15.....	144
Figure 8.19 Flow chart of the MEANDER program	149
Figure 8.20 Flow chart of Geometry Study	150
Figure 8.21 Essential procedures for Geometry Study	151
Figure 8.22 Flow chart of the implementation of the Hyperbolic Model.....	152
Figure 8.23 Flow chart of function OneHydrograph	153
Figure 8.24 Flow chart of function OneFlow	154
Figure 9.1 Verification of Flume Test 1 $R/W=4$, $\phi=120^\circ$, $v=0.25\text{m/s}$	157
Figure 9.2 Verification of Flume Test 1 with migrated banks at each time step.....	157
Figure 9.3 Verification of Flume Test 2 $R/W=4$, $\phi=65^\circ$, $v=0.25\text{ m/s}$	158
Figure 9.4 Verification of Flume Test 3 $R/W=2$, $\phi=65^\circ$, $v=0.25\text{ m/s}$	159
Figure 9.5 Verification of Flume Test 4 with small τ_{\max} $R/W=8$, $\phi=65^\circ$, $v=0.24\text{ m/s}$	160
Figure 9.6 Verification of Flume Test 4 with large τ_{\max} $R/W=8$, $\phi=65^\circ$, $v=0.24\text{ m/s}$	160
Figure 9.7 Verification of Flume Test 9 with small τ_{\max} , $v=0.25\text{ m/s}$	161
Figure 9.8 Verification of Flume Test 9 with large τ_{\max} , $v=0.25\text{ m/s}$	161
Figure 9.9 Flume Test 9, fitted circles of the left bank.....	162
Figure 9.10 Verification of Flume Test 5 $R/W=4$, $\phi=65^\circ$, $v=0.27\text{ m/s}$	162
Figure 9.11 Verification of Flume Test 6 $R/W=4$, $\phi=180^\circ$, $v=0.24\text{ m/s}$	163
Figure 9.12 Verification of Flume Test 7 $R/W=4$, $\phi=220^\circ$, $v=0.24\text{ m/s}$	163
Figure 9.13 Verification of Flume Test 8 $R/W=2$, $\phi=65^\circ$, $v=0.19\text{ m/s}$	164
Figure 9.14 Verification of Flume Test 11, $v=0.30\text{ m/s}$	165
Figure 9.15 Verification of Flume Test 13 $R/W=2$, $\phi=120^\circ$, $v=0.20\text{ m/s}$	165
Figure 9.16 Verification of Flume Test 14 $R/W=4$, $\phi=120^\circ$, $v=0.20\text{ m/s}$	166

	Page
Figure 9.17 Verification of Flume Test 15 R/W=4, $\phi=120^\circ$, $v=0.27$ m/s	167
Figure 9.18 Verification of Flume Test 16 R/W=3, $\phi=65^\circ$, $v=0.25$ m/s	167
Figure 9.19 Verification of Flume Test 17 R/W=4, $\phi=120^\circ$, $v=0.32$ m/s	168
Figure 9.20 Verification of Flume Test 18 R/W=6, $\phi=65^\circ$, $v=0.26$ m/s	169
Figure 9.21 Migration of the Brazos River (Park, 2001).....	170
Figure 9.22 Assume EFA Curve for the verification of the Brazos River	170
Figure 9.23 Hydrograph of the Brazos River SH105 1958-1993 (www.usgs.gov)	171
Figure 9.24 Discharge vs. velocity (Park, 2001)	172
Figure 9.25 Discharge vs. water depth (Park, 2001)	172
Figure 9.26 The Brazos River SH105 1958-1981 measured vs. predicted	173
Figure 9.27 The Brazos River SH105 1958-1981 predicted migration vs. time	173
Figure 9.28 The Brazos River SH105 1981-1993 measured vs. predicted	174
Figure 9.29 The Brazos River SH105 1981-1993 predicted migration vs. time	175
Figure 9.30 Parametric study reference case, migration of channel.....	177
Figure 9.31 Parametric study reference case, migration vs. time	177
Figure 9.32 Parametric study migration versus R/W.....	178
Figure 9.33 Parametric study migration versus bend angle.....	178
Figure 9.34 Parametric study migration versus slope of EFA curve	179
Figure 9.35 Parametric study migration versus velocity	180
Figure 9.36 Parametric study migration versus time	181
Figure 10.1 Start the prediction from different times	189

LIST OF TABLES

	Page
Table 2.1 Data used by Keady and Priest (1977)	17
Table 4.1 Calibration of the flow meter	64
Table 4.2 Verification of the calibration of flow meter	67
Table 5.1 Fitting hyperbolic curves for flume test data	72
Table 5.2 Fitting hyperbolic curve for Des Moines River	73
Table 7.1 Comparison of statistical parameters (Unit: m ³ /s)	117
Table 7.2 Comparison of random number generators	126
Table 9.1 Flume tests done in the Coastal Engineering Lab	156
Table 9.2 Matrix of parametric study for the Brazos River SH105	176

CHAPTER I

INTRODUCTION TO AND FUNDAMENTAL CONCEPTS OF MEANDER MIGRATION

1.1 INTRODUCTION

Meander migration is a process in which water flow erodes soil on one bank and deposits it on the opposite bank. Therefore, a gradual shift of bank line occurs over time. Bank erosion undermines bridge piers and abutments, scours the foundations of parallel highways, and causes loss of useful land. Figure 1.1 is a typical meandering river with two bridges running across it. The white area at the bends was once part of the main channel but is now part of the bank. At the site of the bridges, the river migrates towards downright side in the picture. Countermeasure is needed to retard the migration.

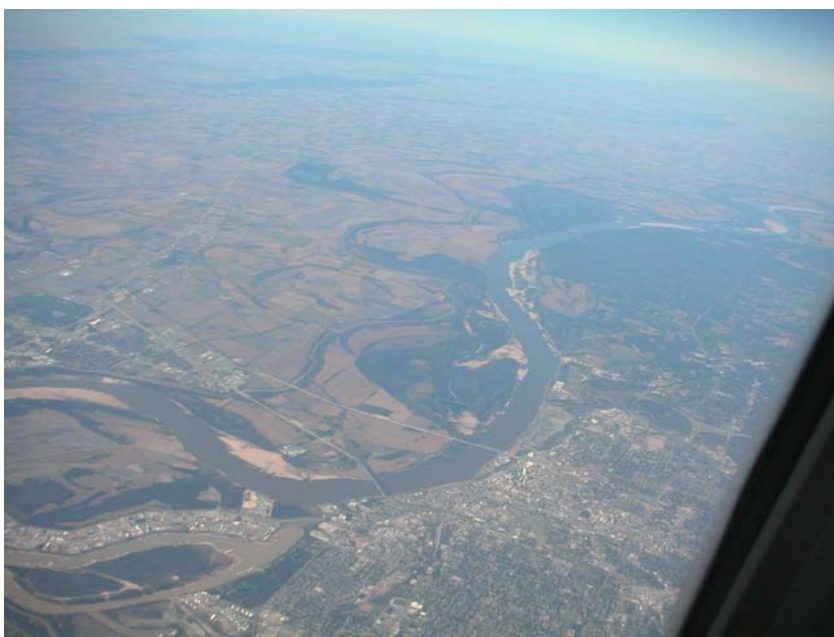


Figure 1.1 A meandering river (courtesy of Dr. Jean-Louis Briaud)

Figure 1.2 shows the satellite picture of a section of the Mississippi River from Helena, Arkansas to Newport, Mississippi. The convoluted channel and parallel traces demonstrate a wild history of this river. In some places, the river migrated more than a kilometer in less than 200 years. The average migration rate is 0.016 width/year (Larsen, 1995).



Figure 1.2 Satellite picture of a section of Mississippi River (maps.google.com)

The bridges crossing meandering rivers are often endangered due to the loss of pier or abutment foundations. This problem is costing millions of dollars to TxDOT (Texas Department of Transportation) in protection of affected bridges and highway

embankments. One recent meander migration threat (FM 787 at the Trinity River, Figure 1.3) has required a 0.3 M\$ emergency countermeasure and a 5.6 M\$ replacement bridge.

In order to avoid costly countermeasures for new bridges, TxDOT sponsored the project “Develop Guidance for Soils properties-based Prediction of Meander Migration.” The research is being conducted at Texas A&M University, College Station. The author is a member of the team. The purpose of this project is to develop a methodology which can be used to do site-specific predictions of meander migration. Then the piers and abutments of new bridges can be constructed away from risky locations.

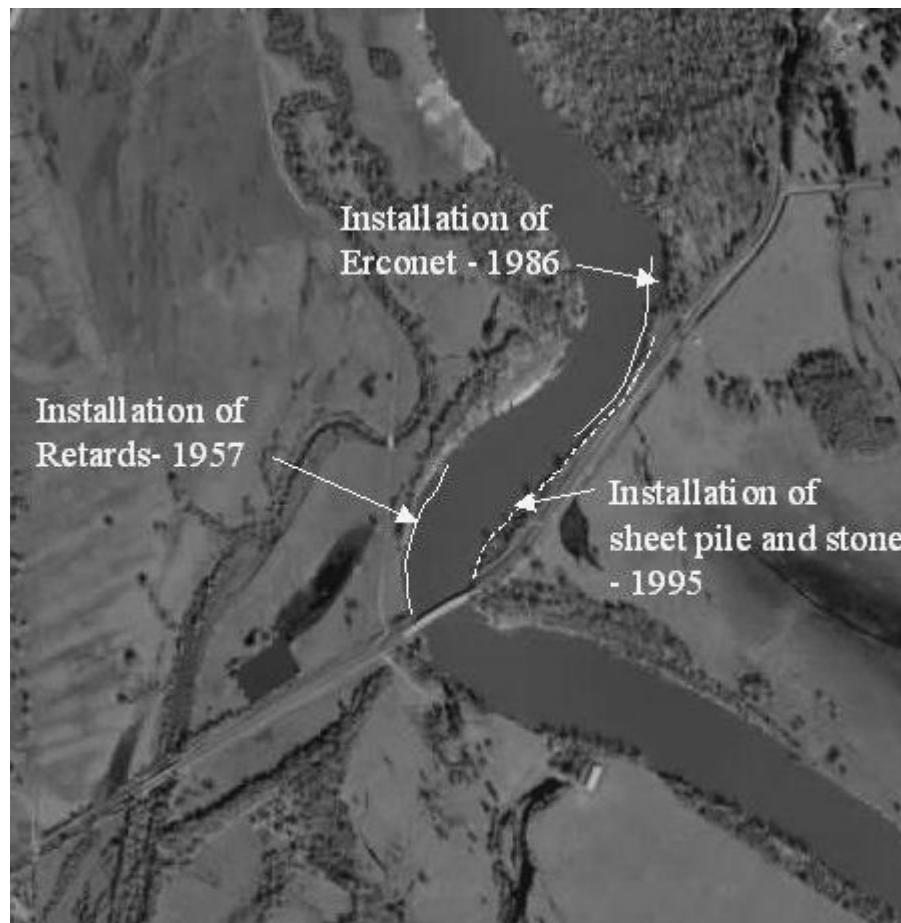


Figure 1.3 Trinity River at Highway 787 (Briaud et al. 2001)

1.2 FUNDAMENTAL CONCEPTS ABOUT MEANDER MIGRATION

1.2.1 Channel pattern and stability

Meander migration occurs as a response to natural or man-made disturbances of the fluvial system. Meander migration and related processes cause problems such as bridge scour, scour of highway foundations, and loss of land. In order to predict changes in channel morphology, location, and behavior, it is important to understand the relative stability of a channel which is revealed by its patterns.

1.2.1.1 Channel classification

Alluvial channels are dynamic systems subject to changes of different types and highly variable rates. Alluvial channel movements are the cumulative result of a combination of climatic, geological, topographic, hydrologic, and human disturbance factors. There are basically three types of channel patterns: straight, meandering, and braided. Rivers with different patterns behave differently, and their other morphologic characteristics are different. Therefore, pattern identification should be the first step toward evaluation of river stability and the identification of potential river hazards.

Brice (1975) developed a descriptive classification of channel patterns for alluvial rivers. He selected the following channel properties as being important for classification: the degree of sinuosity, braiding and anabranching, and the character of meandering, braided and anabranching streams (Figure 1.4).

Brice and Blodgett (1978) classified streams according to Figure 1.5, which is based on stream properties observable on aerial photographs and in the field. Figure 1.5 is intended to facilitate the assessment of streams for engineering purposes, with particular regard to lateral stability.

In Figure 1.6, sinuosity is the ratio of channel length to valley length or the ratio of thalweg length to valley length. Based on sinuosity, a channel can be classified as:

Straight: <1.05

Sinuosity: $1.05\sim 1.25$

Meandering: >1.25

It has been found that there is no definite relation between degree of sinuosity and lateral stability.





























Degree of Sinuosity	Degree of Braiding	Degree of Anabranching
 1 1-1.05	 0 <5%	 0 <5%
 2 1.06-1.25	 1 5-34%	 1 5-34%
 3 >1.26	 2 35-65%	 2 35-65%
 3 >1.26	 3 >65%	 3 >65%
Character of Sinuosity	Character of Braiding	Character of Anabranching
 A Single Phase, Equiwidth Channel, Deep	 A Mostly Bars	 A Sinuous Side Channels Mainly
 B Single Phase, Equiwidth Channel	 B Bars and Islands	 B Cutoff Loops Mainly
 C Single Phase, Wider at Bends, Chutes Rare	 C Mostly Islands, Diverse Shape	 C Split Channels, Sinuous Anabranches
 D Single Phase, Wider at Bends, Chutes Common	 D Mostly Islands, Long and Narrow	 D Split Channel, Sub-parallel Anabranches
 E Single Phase, Irregular Width Variation		 E Composite
 F Two Phase Underfit, Low-water Sinuosity		
 G Two Phase, Bimodal Bankfull Sinuosity		

Figure 1.4 Channel pattern classification devised by Brice (Brice, 1975)




















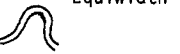

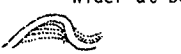
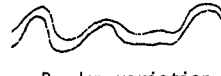

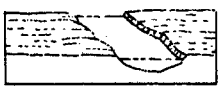

CHANNEL WIDTH	Small (<100 ft or 30 m wide)	Medium (100-500 ft or 30-150 m)	Wide (>500 ft or 150 m)		
FLOW HABIT	Ephemeral (Intermittent)	Perennial but flashy	Perennial		
CHANNEL BOUNDARIES	 Alluvial	 Semi-alluvial	 Non-alluvial		
BED MATERIAL	Silt-clay	Silt	Sand	Gravel	Cobble or boulder
VALLEY; OR OTHER SETTING	 Low relief valley (<100 ft or 30 m deep)	 Moderate relief (100-1000 ft or 30-300 m)	 High relief (>1000 ft or 300 m)	 No valley; alluvial fan	
FLOOD PLAIN	 Little or none (<2x channel width)	 Narrow (2-10x channel width)	 Wide (>10x channel width)		
DEGREE OF SINUOSITY	 Straight (Sinuosity 1-1.05)	 Sinuous (1.06-1.25)	 Meandering (1.26-2.0)	 Highly meandering (>2)	
DEGREE OF BRAIDING	Not braided (<5 percent)		 Locally braided (5-35 percent)	 Generally braided (>35 percent)	
DEGREE OF ANABRANCHING	Not anabranching (<5 percent)		 Locally anabranching (5-35 percent)	 Generally anabranching (>35 percent)	
VARIABILITY OF WIDTH AND DEVELOPMENT OF BARS	 Equiwidth  Narrow point bars	 Wider at bends  Wide point bars	 Random variation  Irregular point and lateral bars		
APPARENT INCISION	 Not incised		 Probably incised		
CUT BANKS	Rare	Local	General		
BANK MATERIAL	Coherent Resistant bedrock Non-resistant bedrock Alluvium		Non-coherent Silt; sand gravel; cobble; boulder		
TREE COVER ON BANKS	<50 percent of bankline	50-90 percent	>90 percent		

Figure 1.5 Stream properties for classification and stability assessment (Brice and Blodgett, 1978)

Shen, et al. (1981) presented five basic patterns (Figure 1.6) that will aid highway engineers in establishing the relative stability of the channel and in identifying some hazards that affect bridge stability. Figure 1.6 is more meaningful than a purely descriptive classification of channels because it is based on cause and effect relations, and it illustrates the differences to be expected when the type of sediment load, flow velocity, and stream power differ among rivers.

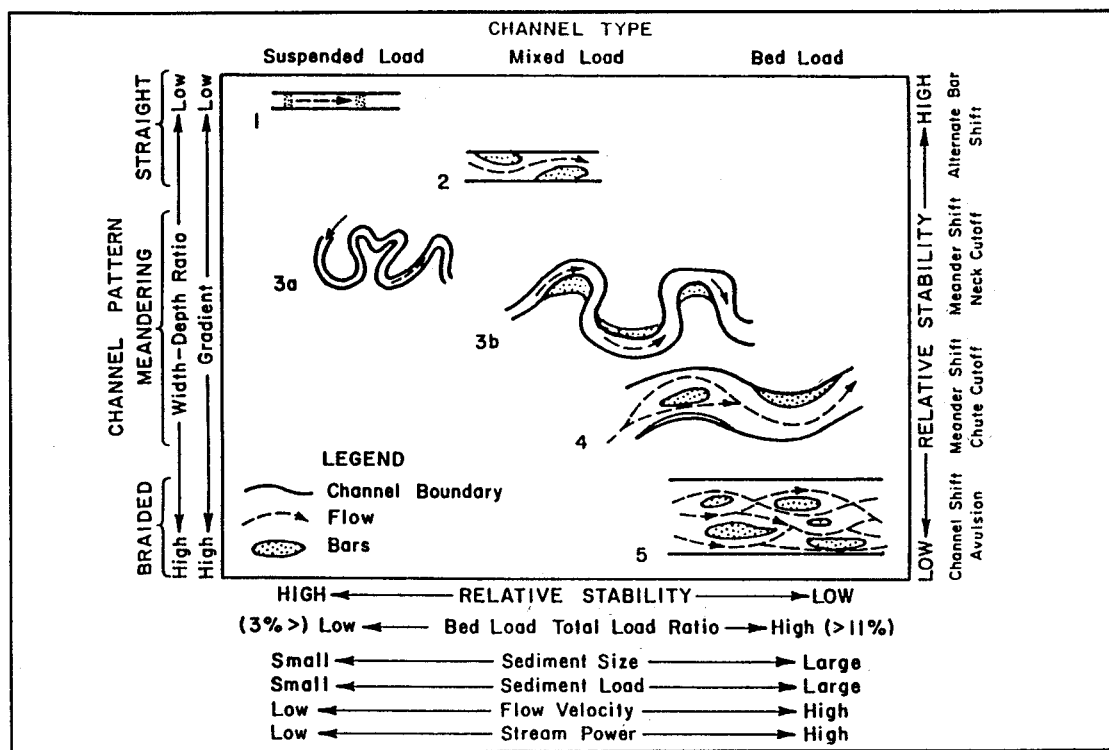


Figure 1.6 Channel classification showing stability and types of hazards encountered with each pattern (Shen et al., 1981)

In Figure 1.6 there is a difference between sediment load, bed load, and total load:

- *suspended load*: amount of sediment being moved by a stream;

- *bed load*: sediment that is transported by rolling, sliding, or skipping along the bed or very close to it; considered to be within the bed layer;
- *total load*: the sum of suspended load and bed load.

1.2.1.2 Lane relation

Richardson et al. (1990) proposed an interrelation among stream form, sinuosity, and slope as shown in Figure 1.7. The interrelation shows when the slope is smaller than a certain value, the channel can only be of meandering form; when the slope is larger than a certain value, the channel can only be of braided form. In between the two values are a combination of meandering and braided forms.

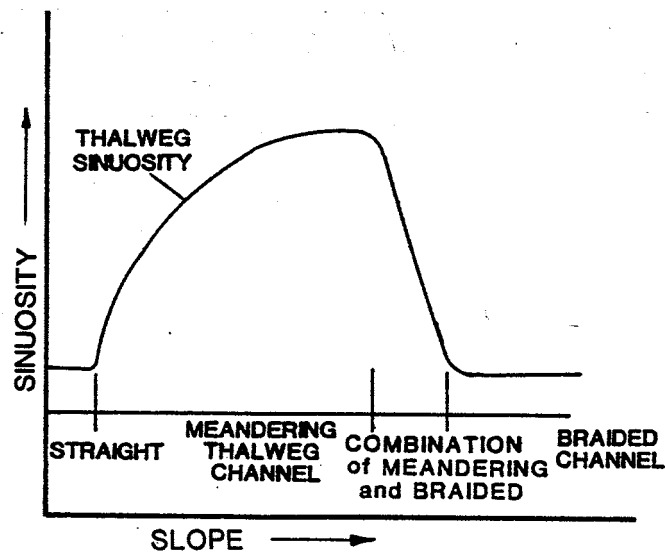


Figure 1.7 Interrelation between stream form and slope (Richardson et al. 1990)

Lane (1957) developed quantitative relations among stream form with sandbed, channel bed slope S_0 , and mean discharge Q shown as follows and in Figure 1.8:

$$\text{Meandering: } S_0 Q^{0.25} \leq 0.0007$$

$$\text{Transition: } S_0 Q^{0.25} = 0.0007 \sim 0.0041$$

Braided: $S_0Q^{0.25} \geq 0.0041$

While Leopold and Wolman (1960) proposed the following relations:

Meandering: $S_0Q^{0.44} \leq 0.0125$

Braided: $S_0Q^{0.44} \geq 0.0125$

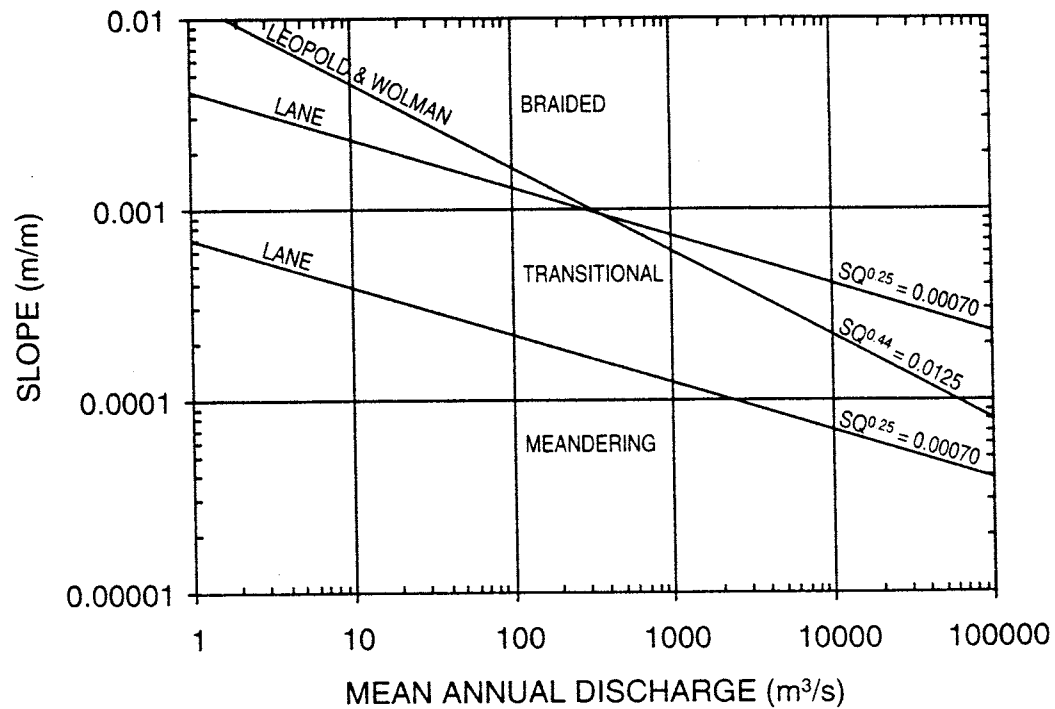


Figure 1.8 Interrelation between stream form, channel bed slope, and mean discharge (after Lane 1957)

1.2.2 Meander migration

The hazards contributing to meander migration and the types of meander migration are described here.

1.2.2.1 Bank erosion

It has been observed in flume tests and in real rivers that bank erosion occurs by either a grain-by-grain movement or by mass movement (slumping or toppling). The following factors can cause mass failure: undercutting of the toe of the bank, steepening of the slope, surcharging the bank by construction or dumping, or seepage forces and pore water pressures related to increased water movement through bank sediment. The bank erosion with non-cohesive materials usually involves the following processes as shown in Figure 1.9.

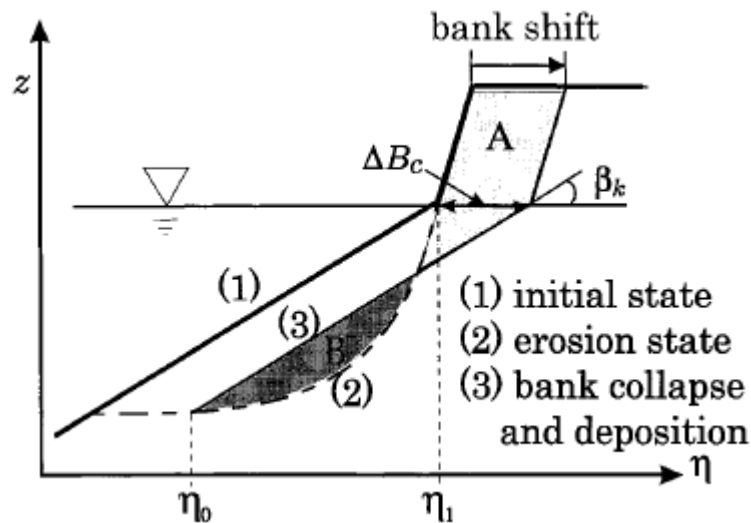


Figure 1.9 Bank erosion process (Nagata et al., 2000)

1. Bed scouring at the side bank;
2. Bank collapse due to instability of the scoured bank;
3. Deposition of the collapsed bank materials at the front of the bank;
4. Transportation of the deposited materials.

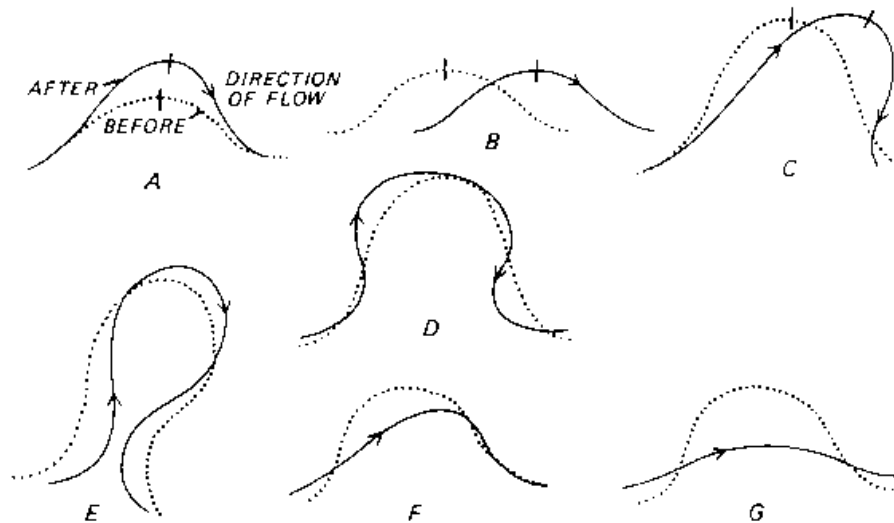
The removal of bank materials on one side is often accompanied by the deposition of the materials on the opposite bank. The shift of the river channel occurs as a result. If

both banks erode at the same cross-section, the channel widens. This may lead to aggradation. According to a survey of various state highway engineers (Brice and Blodgett 1978), bank erosion is rated as a major stream-related hazard.

Aerial photograph is useful for identifying bank erosion and soil deposition. The white area near the toe of a bend often indicates a fair amount quantity of deposition. The soil mainly comes from the nearest upstream bend. The technique to quantitatively evaluate the amount of bank shift by using aerial photographs is called photogrammetry. This technique is widely used to build case histories.

1.2.2.2 Meander growth and shift

Meander growth involves a change in the dimensions of a meander. Meander amplitude and width increase as a meander enlarges. At the same time the radius of curvature of the bend will increase. Meander shift involves the displacement of the meander in a downstream direction. It happens but it is rare that some parts of the bend can actually shift upstream. Figure 1.10 presents the various modes of meander loop behavior.



A. Extension, B. Translation, C. Rotation, D. Conversion to a Compound loop, E. Neck Cutoff by Closure,
F. Diagonal Cutoff by Chute, G. Neck Cutoff by Chute

Figure 1.10 Modes of meander loop behavior (Brice, 1975)

1.2.2.3 Cutoffs

A cutoff is a new and relatively short channel formed across the neck of a meander bend. This drastically reduces the length of the stream in that reach and significantly steepens its gradient. The neck cutoff has the greatest effects on the channel. Another type of cutoff is the chute cutoff, which forms by cutting across a portion of the point bar. The chute cutoff generally forms in recently deposited alluvium, whereas the neck cutoff forms both in recent alluvium and in older consolidated alluvium or even in weak bedrock.

The consequence of both types is that the river is steepened abruptly at the point of the cutoff. This can lead to scour at that location and a propagation of the scour in an upstream direction. In the downstream direction, the gradient of the channel is not changed below the site of the cutoff, and therefore the increased sediment load caused by upstream scour will usually be deposited at the site of the cutoff or below it, forming a large bar.

It is easy to identify the location of a cutoff by examining a sequence of aerial photographs. When the neck of a meander is getting closer and closer at a certain rate, a cutoff is likely to occur.

1.2.2.4 Avulsion

Avulsion is the abrupt change of the course of a river. A channel is abandoned and a new one formed as the water and sediment take a new course across the flood plain, alluvial fan, or alluvial plain. A meander cutoff is a type of avulsion because of relatively rapid change in the course of a river during a short period of time. However, avulsion, as defined here, involves a major change of channel position below the point of avulsion.

A new channel forms below the point of avulsion. If the channel avulses into an existing, smaller channel, a large increase in discharge and sediment load will result, and the bridges downstream of this channel will be inadequate and presumably destroyed. A bridge on the abandoned channel below the site of avulsion will appear to be significantly oversized. If, through avulsion, the river takes a shorter course to the

sea, the gradient will become steeper, and scour above the point of avulsion is certain unless a bedrock control prevents upstream degradation. A bridge located above the point of avulsion will still span the channel, but it may be subjected to degradation and nickpoint migration.

1.3 FACTORS AFFECTING MEANDER MIGRATION

Meander migration is an interactive process between flow and soil. The condition of flow and properties of soil are major factors affecting meander migration. Any factors that affect flow condition and soil properties affect meander migration as a result. A lot of investigation and research have been done to identify influential factors. A relatively complete list was made by previous TxDOT project (Briaud et al., 2001b):

- Soil properties (soil erosion function)
- Flow condition (discharge, velocity, water depth)
- Meander geometry (width, depth, radius of curvature, sinuosity)
- Stream pattern (straight, meandering, braided)
- Free surface slope
- Channel roughness (Manning's "n", friction factor "f")
- Sediment load
- Vegetation
- Debris Problem
- Channel relocation
- Human activities on the floodplain of river

The geometry of meander channel is believed to influence flow condition to a large extent. Flow, soil, and geometry are considered as the most important factors affecting meander migration and are studied in detail in this research.

Geometry of real rivers is often very complicated. In order to reduce the complexity to an acceptable level but with sufficient precision, channel bends are treated as arcs. Figure 1.11 defines the parameters needed to describe the geometry.

C = center of best fit meander circle
 A = apex of meander
 a = meander amplitude
 $W = b$ = channel width
 $R = r_c$ = radius of curvature of meander
 ϕ = bend angle

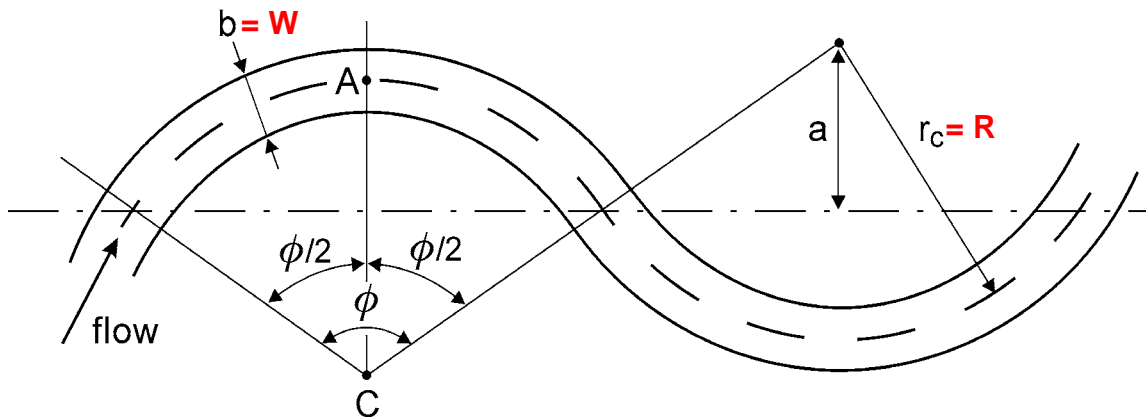


Figure 1.11 Geometry parameters for meanders (after Briaud et al., 2001b)

CHAPTER II

EXISTING KNOWLEDGE OF MEANDER MIGRATION

2.1 GENERAL APPROACHES

The existing approaches to predict meander migration make use of geometry, water, and soil parameters in various ways. These approaches can be divided into three categories: those using time-sequence maps and extrapolation (Briaud et al. 2001b; Lagasse et al. 2004b), those using empirical equations (Keady and Priest 1977; Nanson and Hickin 1983), and those using fundamental modeling (Nagata et al. 2000; Duan et al. 2001).

With the time-sequence maps and extrapolation approach, meander migration is predicted by accumulating topographic maps and aerial photographs of the riverbanks at various dates in the past, measuring the migration rate from those maps, and extrapolating into the future. These maps and aerial photographs can be obtained from local libraries, or from web sites such as <http://mac.usgs.gov/mac/>, <http://terraserver.microsoft.com>, and <http://earth.google.com/>. The advantages of this approach are that it is relatively simple and is based on full-scale observations at the site. The drawbacks are the limited availability of maps and photographs, and the assumption that future flow and soil conditions will be the same as in the past. Departments of Transportation commonly use this method.

With the empirical approach, a database of observed meander migrations and associated parameters is assembled, most influential parameters are selected, a regression is performed, and an equation is proposed. The advantages of this approach are that it is simple and is based on full scale observed data. The drawbacks are that the equation may not include all the essential parameters influencing the process, and that the applicability of the equation is limited by the extent of the database both in terms of quantity of data and geographical area. This approach is also quite common.

A fundamental modeling approach consists of modeling the erosion process at the water-soil interface and projecting it into time by using future hydrographs (daily

discharge versus time). This approach has the advantage of simulating the real phenomenon on a site-specific basis. It has the drawback of being more complicated because it requires the site-specific measurement of soil properties and the selection of future hydrographs. A fundamental modeling approach can also be based on establishing and solving constitutive equations such as conservation of mass (flow and sediment). This approach has the advantage of modeling erosion at the particle level. So far there is still a big gap between the computer model and the reality. This approach is not discussed here.

2.2 SELECTED EMPIRICAL METHODS

2.2.1 Keady, D. M., and Priest, M. S. (1977)

The rate of downstream migration is considered as a function of the free surface slope of the river, the meander amplitude, and the specific weight of water, expressed by the following formula:

$$\frac{V}{\sqrt{gA}} = \phi(s)$$

Where,

V (ft/yr): Migration rate,

g (ft/sec²): Acceleration of gravity,

A (ft): Meander amplitude,

s : Free surface slope,

ϕ : A function of s .

The graph presented in Figure 2.1 is based on data from the Red River in Arkansas, and Louisiana, from the Red Deer River in Alberta, Canada and from other rivers, as shown in Table 2.1. The points corresponding to the case histories in the previous study (Briaud et al. 2001b) are also plotted on the graph.

Free surface slope is very close to channel bed slope which is determined by the slope of the terrain. Water flows to a lower place. Slope of the terrain provides potential energy to keep water flowing and determines how fast water flows. The authors

correctly identified the importance of channel slope. But channel slope is not the only factor affecting flow condition. Rainfall is another important factor influencing flow condition. Usually a large portion of total migration distance occurs during the few big floods. Soil properties are another important issue ignored by the authors. Different rivers neither have the same precipitation nor have the same soil properties. There is not enough evidence to prove that the exceptional data point from Red River, Arkansas was caused by a free surface slope of 1.5×10^{-4} . The data points from previous study don't fit the proposed trend either.

Table 2.1 Data used by Keady and Priest (1977)

Identification	Velocity of Migration (ft/yr)	Meander Amplitude (ft)	Slope
Mississippi R (LA)	60	13,000	.0000436
Mississippi R (MS)	111	11,000	.0000588
Mississippi R (TN)	225	13,200	.0000777
Red R (ARK)	350	2,900	.000132
Pearl R (LA)	20	1,050	.000200
Red Deer R (Canada)	20	1,200	.000275
Tombigdee R (MS)	13	800	.000421
Buffalo R (MS)	17	1,560	.000689

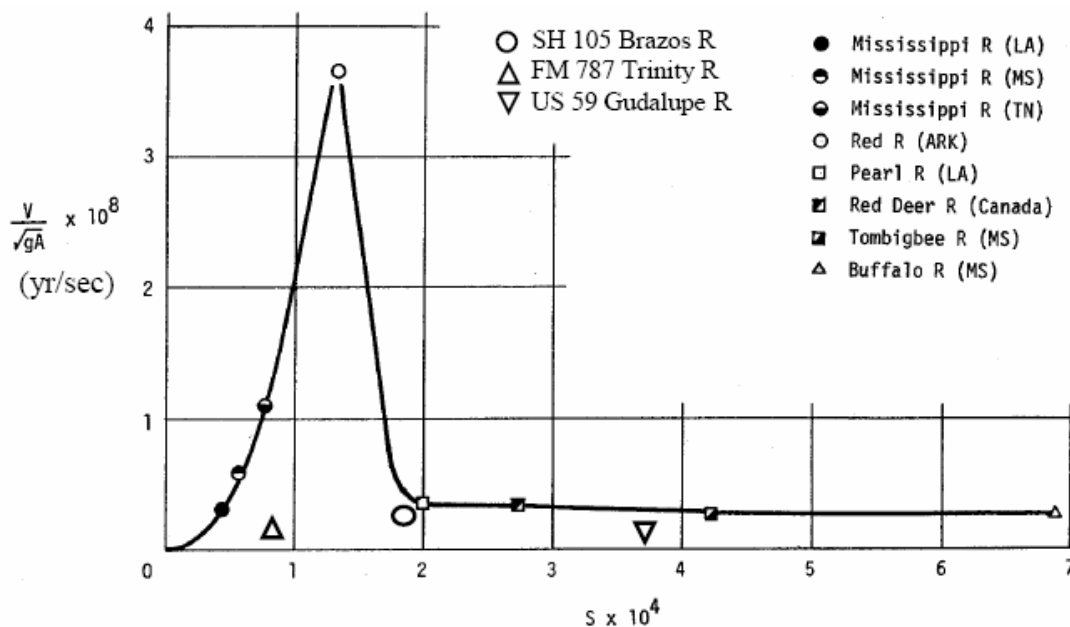


Figure 2.1 Graph for determining rate of meander migration (Keady and Priest, 1977; Briaud, 2001b)

2.2.2 Hooke, J. M. (1980)

The author proposed that the erosion rate is most closely related to catchment area (as a surrogate of discharge and width). Rates of bank erosion were determined from field measurements and historical maps for 11 streams in Devon, England. Then the rates of bank erosion were compared with worldwide published rates in 43 streams. The equation was derived through multiple regression analysis using the 54 data points and resulted in very high rates of bank erosion. The equation was then modified with the same data from 11 streams in Devon, England and 43 streams from literature, given as follows:

$$Y = 0.05 A^{0.5}$$

Where,

Y (m/year): bank erosion rate

A (km²): catchment area

Figure 2.2 shows the data and regression for Hooke's equation. The data points resulting from previous study were added to this figure.

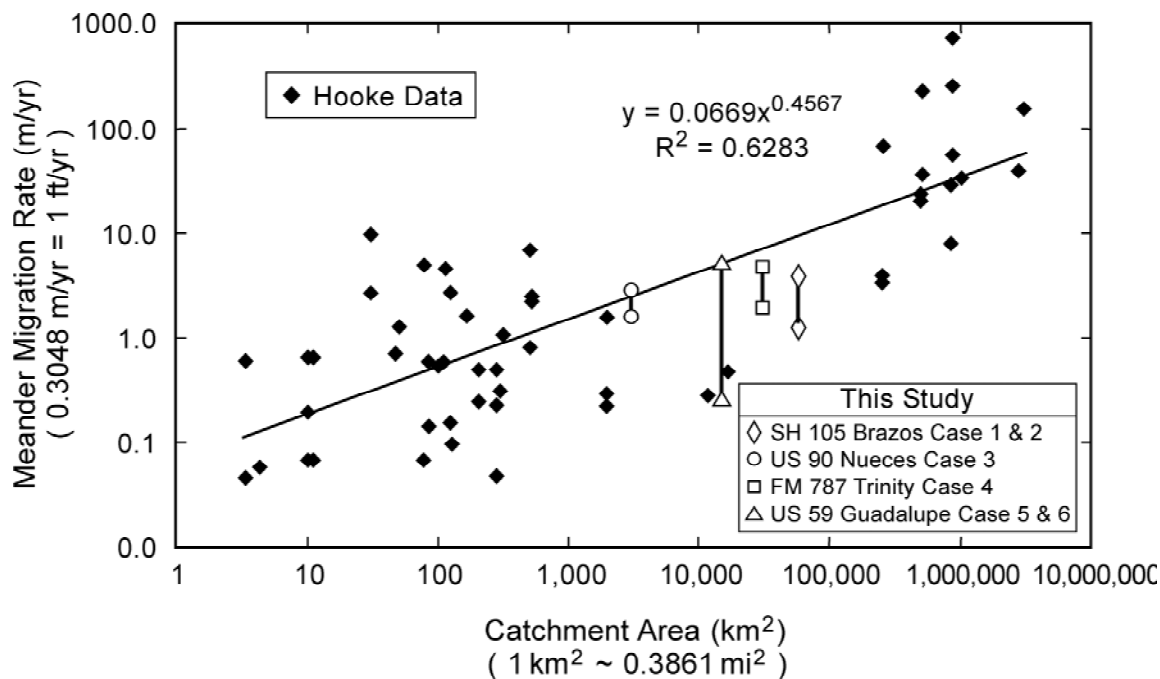


Figure 2.2 Relation between catchment area and migration rate (Hooke, 1980; Briaud et al., 2001c)

The catchment area for a reach of a river is an area whose runoff all goes to that reach. Rainfall is an importance source of river flow. For the same amount of rainfall, the larger the catchment area is, the larger the flow rate is. For the same river, it is reasonable to assume that the migration rate of different reaches is related to their catchment area. For different rivers it might not be a good idea to relate migration rate to catchment area only and disregard different precipitation levels. Rivers having the same catchment area might have quite different flow rate due to different precipitations. Soil properties are also ignored here. Although both axes in Figure 2.2 are in logarithmic scale, there is still a big scatter. The data from previous study are far from the fitted line.

2.2.3 Brice, J. C. (1982)

Brice (1982) proposed that the rate of bank retreat increases with increasing channel width.

$$Y = 0.01 \times B$$

Where,

Y (m/yr): Mean erosion rate

B (m): Channel width

Brice data consisted of 43 data points from four different stream types (equiwidth, wide bend, braided point bar, braided) as shown in Figure 2.3. The data points for previous study were added to Figure 2.3. There are several points for each site because there are several periods of observations for each site.

This formula is beautiful regarding its simplicity. But it is too simple that it is below the bar. The author only related migration rate to channel width. Almost all the important factors are missing in this formula. Figure 2.3 shows the value 0.01 is meaningful for only a few points in the data set. The migration rate is normally not constant along a river or over time. The ratio of migration rate to channel width is not a constant either. The average ratio of migration rate to channel width can roughly predict the amount of migration within a short period of time for the same river with no guarantee of precision. It is not likely that a universal formula like the author proposed can be applied to all rivers.

2.2.4 Nanson G. C., and Hickin, E. J. (1983)

Nanson and Hickin described that the ratio of radius of curvature of a bend (R_c) to channel width (W) influences the lateral migration rate of a meandering river. The relationship between channel migration rate (MR) and the ratio of radius of curvature to channel width for the Beatton River, Canada and other rivers is shown in Figure 2.4. The Normalized migration rate (MR/W) is highest when the ratio of radius of curvature to channel width (R_c/W) is about 3. The data points from previous study were added to

the figure. Their data (Nanson and Hickin 1983) conform, approximately, to the following relation:

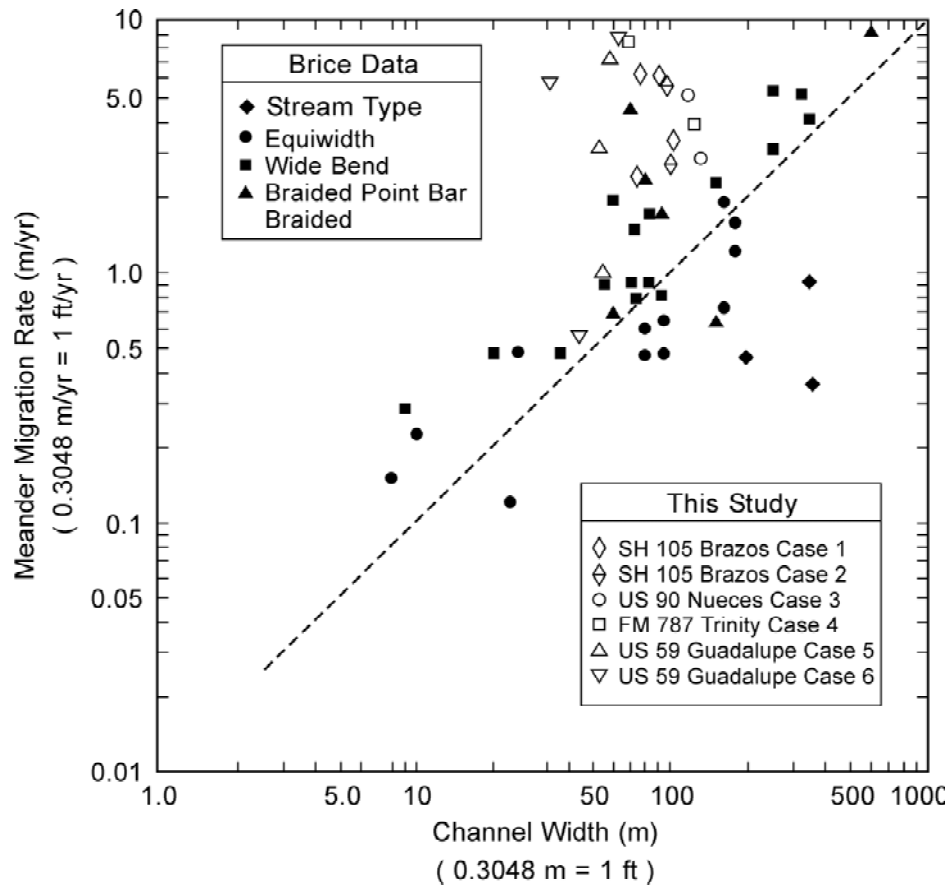


Figure 2.3 The relationship between migration rate and channel width (Brice, 1982; Briaud et al., 2001b)

$$MR = 0.2 (R_c / W) \quad \text{for } (R_c / W) < 3$$

$$MR = 2.0 (R_c / W)^{-1} \quad \text{for } (R_c / W) \geq 3$$

Where:

MR (m/year): Mean erosion rate

W (m): Channel width

R_c (m): Radius of curvature

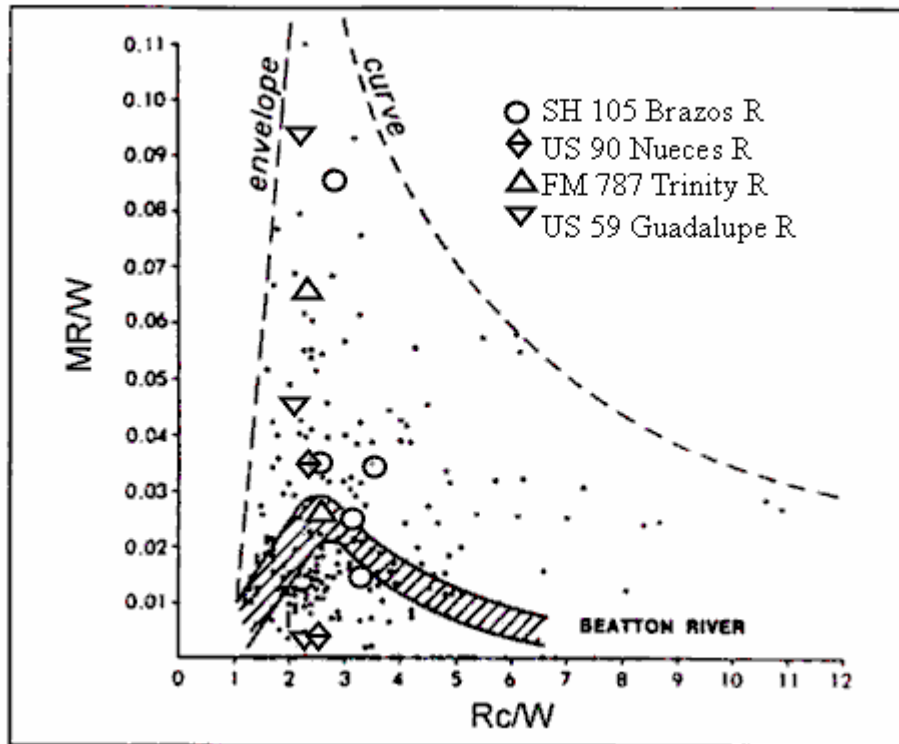


Figure 2.4 The relationship between migration rate and geometry (Nanson and Hickin, 1983; Briaud et al., 2001b)

This relationship between migration rate and geometry was widely cited in literature. But the scatter in data doesn't necessarily lead to the relationship the authors proposed. Together with the data provided by previous project, the data points spread almost all over the graph. With the same data, different persons may come up with completely different relationships. Geometry is only one factor affecting meander migration. Other important factors such as velocity and soil properties were ignored in the authors' conclusion. This study described the relation between migration rate and geometry qualitatively rather than quantitatively.

2.2.5 Odgaard, A. J. (1987)

Odgaard solved constitutive equations by assuming that the rate of bank erosion is proportional to the difference between the near-bank depth-averaged mean velocity

and the reach-averaged mean velocity at bank full discharge. The resulted equation indicates that the erosion rate is correlated with channel characteristics such as width, depth, curvature, bend angle of channel centerline, channel slope, friction factor, and degree of vegetation on the banks. The predictions using this equation agreed well with data measured by using historical records (air photos, maps, and stream flow records), field measurements, and soil analysis in East Nishnabotna River and Des Moines River in Iowa.

$$\bar{v} / u = 2E \frac{b}{r_c} \left(1 + \frac{b}{2r_c}\right)^{-1} F$$

Where,

\bar{v} (m/yr): the average rate of erosion

u (m/s): reach-average mean velocity

$$E = \frac{e}{8} \left(\frac{3\alpha}{2} \frac{\sqrt{\theta}}{\kappa} \frac{m+1}{m+2} F_{D_c} - 1 \right)$$

e : erosion constant

α : 1.27

θ : Shields' parameter, 0.06

m : friction parameter

κ : Karman's constant, 0.40

F_{D_c} : particle Froude number

b (m): bank-full width of channel

r_c (m): radius of curvature

$$F = 1 - \exp \left[-B \frac{r_c \phi}{b} \left(1 - \frac{\beta}{\phi}\right) \right]$$

$$B = \frac{2\kappa^2}{(m+1)^2} \frac{b}{d_c}$$

ϕ : bend angle

β : angle from cross over to first outer bank erosion occurrence

d_c : centerline flow depth

In the calculations, two variables were assumed for this study: the friction factor (m) was taken as 3 and particle Froude number (F_{D_c}) as 10 (Odgaard, 1987). In addition, the erosion constant was taken as $e = 6.4 \times 10^{-7}$ and $e = 4.4 \times 10^{-7}$ for the case of light or no vegetation and dense vegetation on outer bank, respectively.

This method considers flow (u), soil (e), and geometry (b/r_c). If the right erosion constant is picked, the predicted migration rate is close to the measured one. Although soil erodibility is a fundamental property of soil, it is treated as an empirical coefficient here. An e value can be chosen based on existing records. But there is no guarantee that the value will work for a true prediction case. Soil erodibility can be obtained by using EFA (Erosion Function Apparatus) to test the erosion function of soil (Briaud et al. 2001a). It is worthwhile to study how to replace erosion constant e here with EFA function.

The above method was a simplified version of a more complex solution developed by the same author (Odgaard 1986). A modified version came out soon (Odgaard 1989a). But unfortunately the answer was very sensitive to some parameters. In the example given in another article (Odgaard 1989b), the calculated migration rate of a hypothetical river is 11 m/yr. If the flux factor B is changed from 6 to 4.8, the migration rate will be -17 m/yr. If $B=4.9$, the migration rate will be 16 m/yr. If the factor f is changed from 0.08 to 0.07, the migration rate will be -12.1 m/yr. So this method is not good for practical use.

2.2.6 Lagasse, P. F., Spitz, W. J., Zevenbergen, L. W., and Zachmann, D. W. (2004a, 2004b)

This group conducted the NCHRP research Project 24-16. The principal product of this research was a stand-alone handbook for predicting stream meander migration using aerial photographs and maps. The handbook deals with the problem of incremental channel shift and provides a methodology for predicting the rate and extent of lateral channel shifting and down valley migration of meanders. The methodology is basically

the approach mentioned above of using time-sequence maps and extrapolation. First circles for a bend at two different times are fitted. Then the location of the center and the magnitude of the radius are linearly extrapolated. The direction of new migration increment can also be extrapolated based on the directions of two previous increments. The process is described by Figure 2.5 through Figure 2.7 and the equations that follow. A program named “The Data Logger and Channel Migration Predictor” was developed to assist this process. It was an ArcView extension and was written in Visual Basic for Application (VBA) for ArcView.

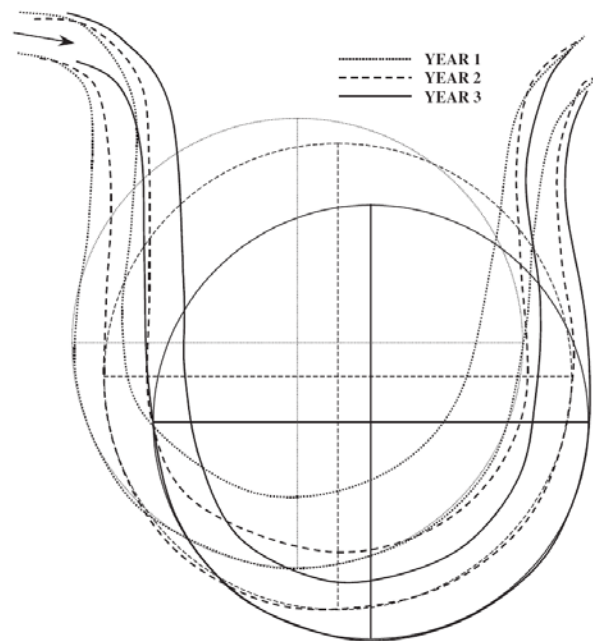


Figure 2.5 Banklines and circles drawn along outer bankline positions for a hypothetical channel in 3 different years (Lagasse et al., 2004b)

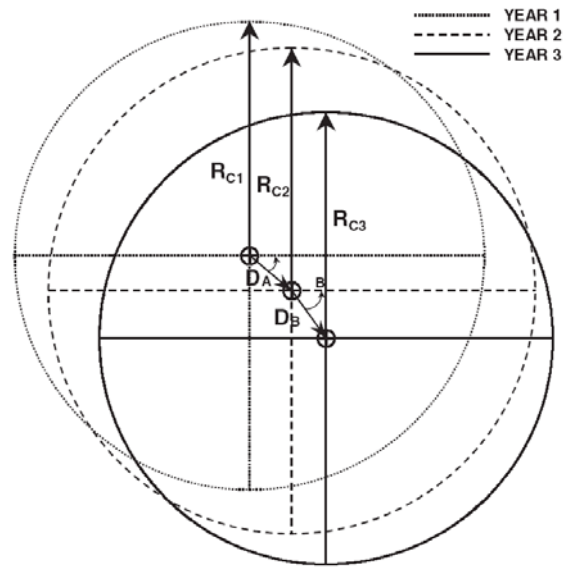


Figure 2.6 Diagram defining the outer bank radius of curvature in Years 1, 2, and 3 (R_{c1} , R_{c2} , and R_{c3}) and the amount (D_A and D_B) and direction (θ_A and θ_B) of migration of bend centroid during Periods A and B (Lagasse et al., 2004b)

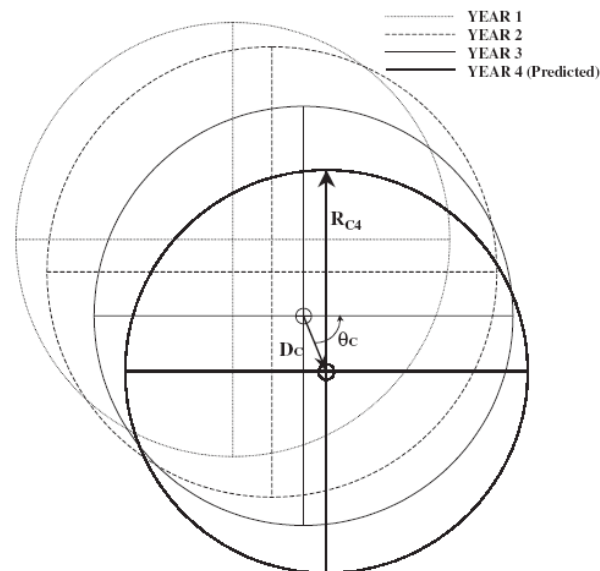


Figure 2.7 Predicted position and radius of curvature of the circle that defines the outer bank of the hypothetical channel in Year 4 (Lagasse, et al., 2004b)

$$R_{C4} = R_{C3} + \left[\left(\frac{R_{C3} - R_{C2}}{Y_B} \right) (Y_C) \right]$$

$$\theta_C = \left[\left(\frac{\theta_B - \theta_A}{Y_B} \right) (Y_C) \right] + \theta_B$$

Briaud et al. (2001b, Figure 2.8) have applied this time sequence and extrapolation method and have showed some of its limitations. A constant migration rate is assumed in the extrapolation, which can be valid only when the flow condition of the predicted period is the same as historical record, the soil doesn't change and the predicted period is short. The reality showed that the ideal situations like these rarely happen. A big flood can completely change the migration trend this method is supposed to predict. Even under the same flow condition, a change in soil can also turn the trend to a different direction. This method is good in the sense that it gives a quick first hand estimation. But a more reliable method is needed to give better predictions.

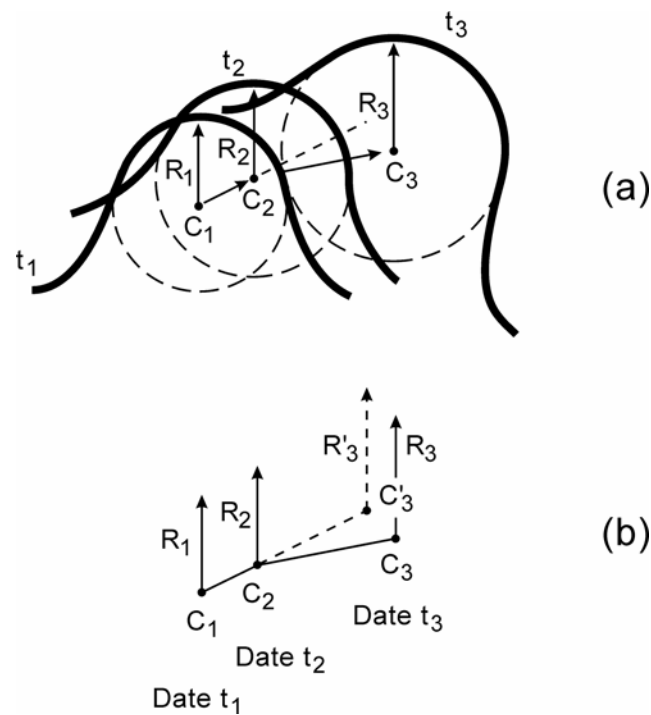


Figure 2.8 Definition of the time sequence maps and extrapolation method (Briaud et al., 2001b)

A way to estimate extension and translation migration based on statistical data was also provided. Figure 2.9 shows the cumulative probability of normalized extension migration (migration/width) of a certain type of river reaching a certain quantity. For example, for meanders of the type of Brice C Sites, the probability of the normalized migration reaching 0.02 or less is 79%. Figure 2.10 is for translation migration. In this way, migration can be estimated with a certain level of confidence when historic records are not available.

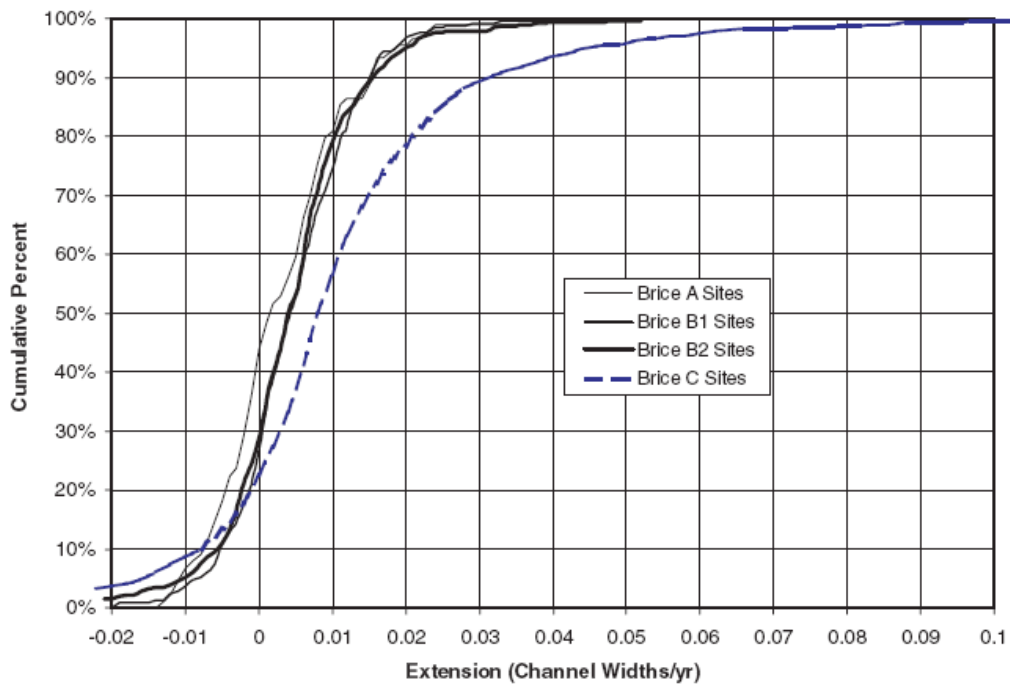


Figure 2.9 Cumulative percentage of extension migration (Lagasse et al., 2004b)

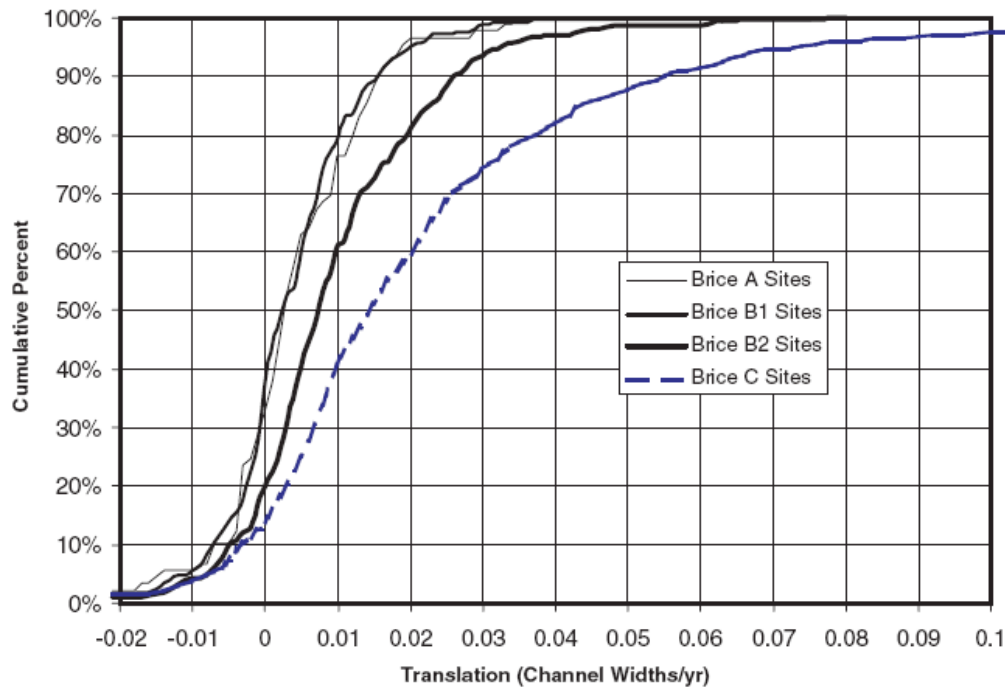


Figure 2.10 Cumulative percentage of translation migration (Lagasse et al., 2004b)

2.2.7 Hudson, P. F., and Richard, H. K. (2000)

The authors examined the channel migration and meander-bend morphology for the lower Mississippi River between 1877 and 1924, prior to channel cutoffs, revetments, and change in sediment regime. Average migration rate was 45.2 m/yr in the upper alluvial valley but increased to 59.1 m/yr in the lower alluvial valley. The highest migration rate occurred when the ratio of radius of curvature to channel width was between 1.0 and 2.0. No prediction method was proposed.

2.3 SELECTED NUMERICAL METHODS

Numerical simulation belongs to the category of fundamental modeling. In some cases empirical equations are also used. A numerical simulation method normally contains these three components: modeling of flow field, modeling of sediment transport, and modeling of bank erosion. The calculation of flow field and sediment

transport is an important field in Hydraulic Engineering. Bank erosion is an interaction of water and soil, while the study of soil properties is a subject of Geotechnical Engineering. Attacking this problem requires multi-disciplinary efforts including expertise in both Hydraulic Engineering and Geotechnical Engineering. The following review is an attempt to cover the development in numerical simulation from a geotechnical engineer's point of view. It is encouraged to refer to the original articles for more details in Hydraulic Engineering.

2.3.1 Ikeda, S. et al. (1981), and Parker, G. et al (1982)

Although the authors didn't propose a method for predicting meander migration, they developed a theory explaining the formation of meanders. The alternate-bar instability has been identified as the cause of meander migration. Bend instability in sinuous channels is different from the instability of alternate bars in straight channels. Two mechanisms work at the same characteristic wavelengths. An expansion technique involving the Stokes expansion for water waves is developed to perform a nonlinear stability analysis, which explains the skewing and fattening and shows the lateral and downstream migration rates is proportional to the bend amplitude.

2.3.2 Blondeaux, P., and Seminara, G. (1985)

The authors developed and applied a 2D model of flow and bed topography in sinuous channels with erodible boundaries in order to study the mechanism of meander initiation. A "resonance" phenomenon undiscovered by Ikeda (1981) was detected when the values of relevant parameters fall within a certain range. It was proposed that the resonance controls the bend growth and is connected with bar instability. Comparison with experimental observations showed that resonance is associated with meander formation.

2.3.3 Pizzuto, J. E. (1990)

The author developed a 2D numerical model to predict the distribution of boundary shear stress, cross-channel sediment transport rates and the evolution of the bed topography. Equilibrium values of dimensionless depth increases with the decrease

of the slope. With the progress of the computation, flat bed and a curved bank will be developed.

2.3.4 Darby, S. E. et al. (1994, 1996a, 1996b, 2002)

Darby and Thorne (1994, 1996a) solved the governing equations of flow continuity, flow resistance, conservation of flow momentum, sediment transport, bank stability and conservation of sediment mass. A physically-based 2D numerical model was developed to simulate channel widening. Comparison was made with a 13.5 Km reach of the South Fork of the Forked Deer River, in west Tennessee (Darby and Thorne 1996b). Qualitative agreements were observed and quantitative prediction was not reliable.

A numerical model was developed and tested for river morphology for cohesive erodible banks (Darby et al. 2002). The model couples a 2D depth-averaged model of flow and bed topography with a mechanistic model of bank erosion. Deposition and removal of failed bank material were simulated. The authors also solved the governing conservation equations in a moving boundary fitted coordinate system. Model performance was encouraging in regard to agreements between simulations results and data from two flume tests and a real river.

2.3.5 Mosselman, E. (1998)

The proposed prediction method consists of a 2D depth-averaged flow model and a bank erosion model. The prediction method is applied to a reach of the meandering gravel-bed River Ohre in the former state of Czechoslovakia. A poor agreement is observed. It is claimed that the inclusion of a 3D flow model, a sediment transport model will improve the prediction results.

2.3.6 Nagata, N., Hosoda, T., Muramoto, Y. (2000)

The authors presented a numerical model that can be used for investigating both bed-deformation and bankline shifting in 2D plan form. The governing equations are composed of 2D continuity and momentum balance equations in a moving boundary fitted-coordinate system. The authors developed a sediment transport model that

included the theory of non-equilibrium sediment transport proposed by Nakagawa and Tsujimoto (1980). An intermittent bank erosion model proposed by Hasegawa (1981) was used, which means only the continuity of the volume of sediment during the process of bank slide.

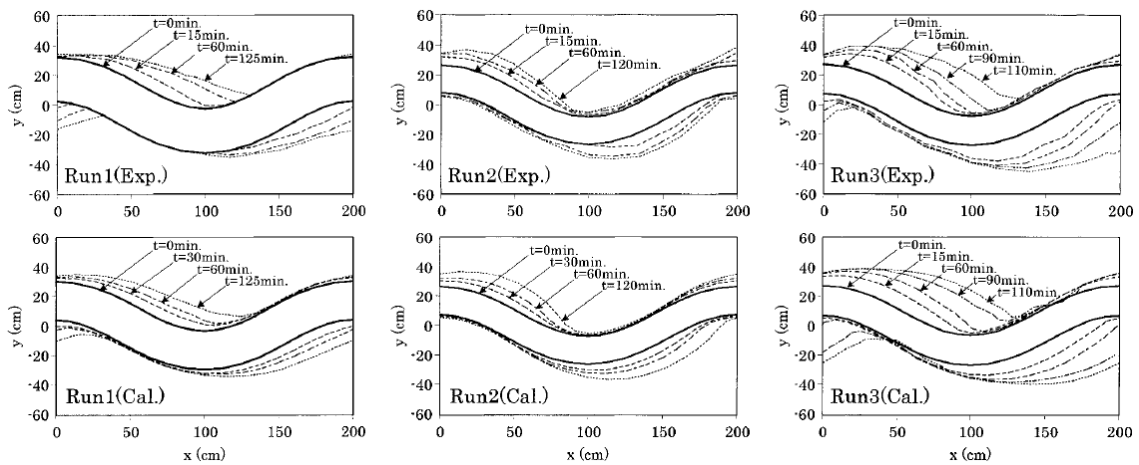


Figure 2.11 Temporal changes in plan forms (Nagata et al., 2000)

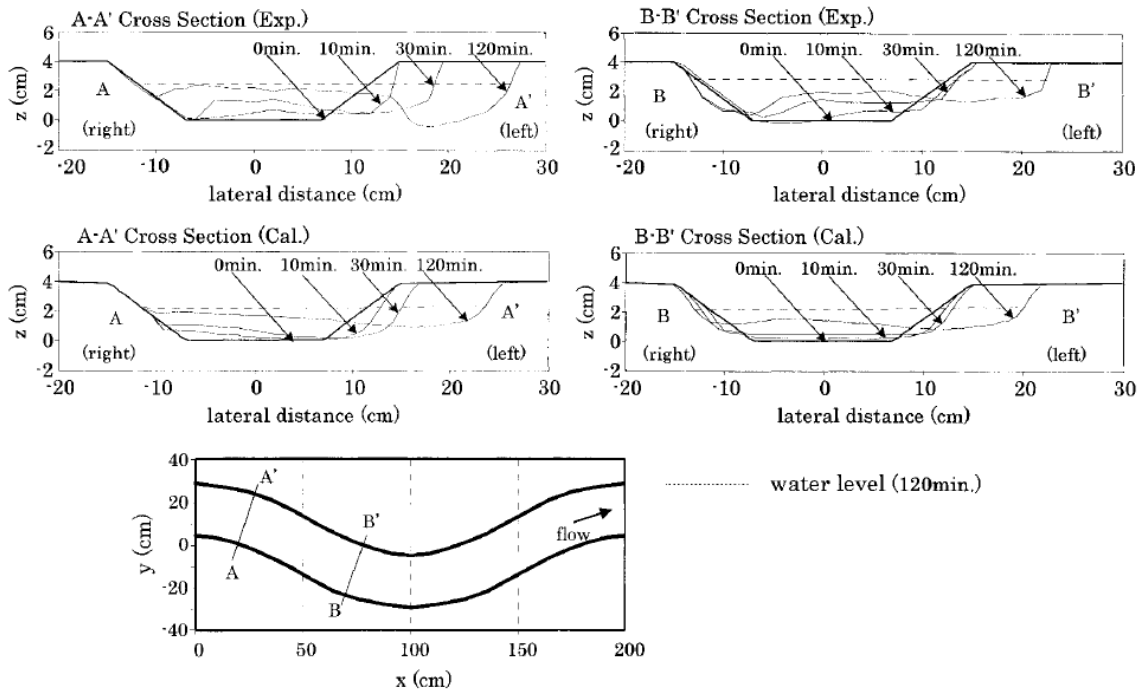


Figure 2.12 Temporal changes in cross-sectional profiles for Run 2 (Nagata et al., 2000)

Verification was done on a laboratory experiment which was conducted in a 10 m long, 1 m wide, 0.2 m deep steel flume. All of the initial plan forms of the meandering channel were set as a sine-generated curve with 2 m linear wavelength. A constant discharge of 1.98 l/s was maintained at the inlet, which had a mean flow depth of 3.0 cm and a valley slope of 1/300. The sediment was uniform with a diameter of 1.42 mm. Sediment was fed manually during the experiment. Figure 2.11 shows the observed and calculated plan forms for runs 1 to 3. Good agreements between calculation and test were achieved. A comparison between observed and calculated cross-sectional profiles is shown in Figure 2.12. The same trend can be seen in calculation.

2.3.7 Sun, T. et al. (1996, 2001a, 2001b)

The linear theory of Johannesson and Parker (1989) was used to develop a 2D computer model for meandering rivers that couples water flow, bed topography, the sorting of sediments with different grain sizes. The model for bed load sediment

transport and sorting came from the theory of Parker and Andrews (1985). Simulation results showed that curvature-related instabilities is a primary factor affecting the initial growth of meandering rivers and the alternate bars have little impact on the initial development of meander loops.

2.3.8 Duan, J. G. et al. (2001, 2005a, 2005b)

Duan et al. (2001) developed a numerical-empirical model called Enhanced CCHE2D (EnCCHE2D) to simulate alluvial channel migration phenomena. EnCCHE2D model is capable of predicting quasi-3D flow field and shear stress distribution on the bed. The process of sediment transport and meander migration were predicted based on these quasi-3D flow solutions. The advance or retreat of bank is calculated by considering not only the hydraulic erosion of bank surface and toe, but also the mass balance of sediment flux in the near-bank zone.

The flow field was initially obtained by the 2D depth-averaged hydrodynamic model, CCHE2D. A set of empirical functions were then used to transform the flow and the bed shear stress field into quasi-3D ones. Details are explained in Duan's publication (1998).

Sediment consists of bed load and suspended load. Meyer-Peter and Muller (1948)'s formula for bed load sediment transport was used. The primitive definition indicates the volumetric sediment transport rate per unit width and length is equal to the integration of the product of velocity and concentration along flow depth. The original formula of Rouse (1938) was adopted for calculating sediment concentration profile. Van Rijn's formula (1989) was found to give the best results in computing reference concentration.

Basal erosion and bank failure occur in the process of bank erosion. Basal erosion that caused by flow induced shear stress on the submerged part of bank surface was calculated. Bank failure was studied at a later time (Duan, 2005b). The bank erosion rate is determined by the flow and sediment transport fluxes near the bank not just the excessive velocity. After some improvements were made, this prediction model was used to simulate the inception of channel meandering (Duan and Julien 2005a).

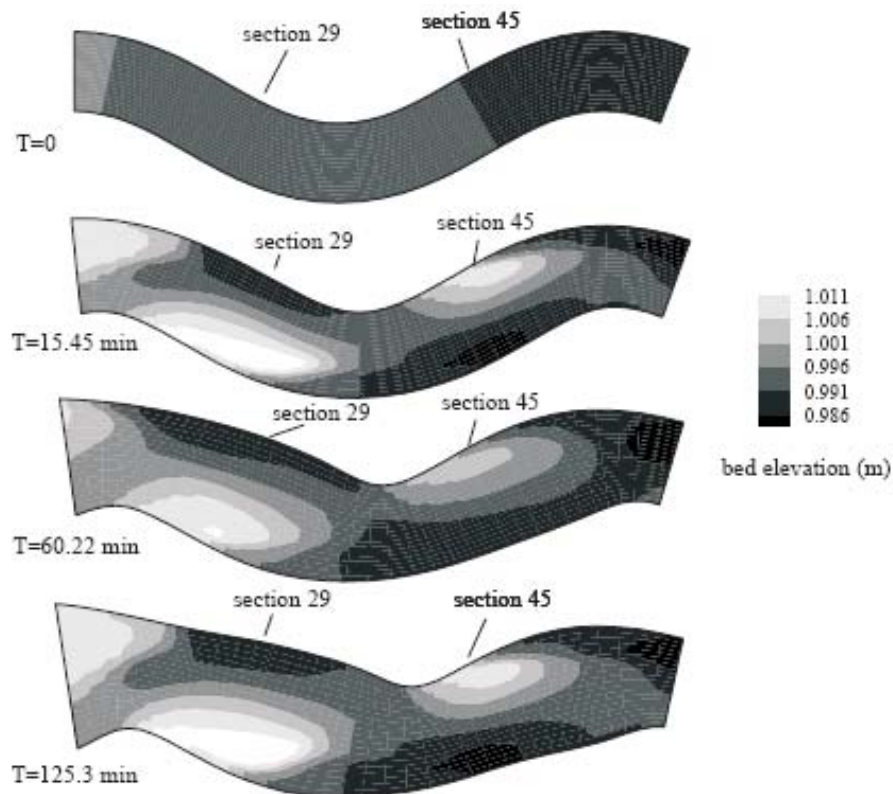


Figure 2.13 Simulation of meandering channel widening due to bank erosion (Duan et al., 2001)

Verification was done on the same experiment as the previous section (Nagata et al. 1997). Figure 2.13 is a simulation result which shows the channel widens and bars and pools are formed. Since flow field and bed profile measurements are not available, it is not possible to compare simulated flow field and bed configurations with experimental data. Good agreements were observed in the comparison of simulated bank lines with those measured, as shown in Figure 2.14.

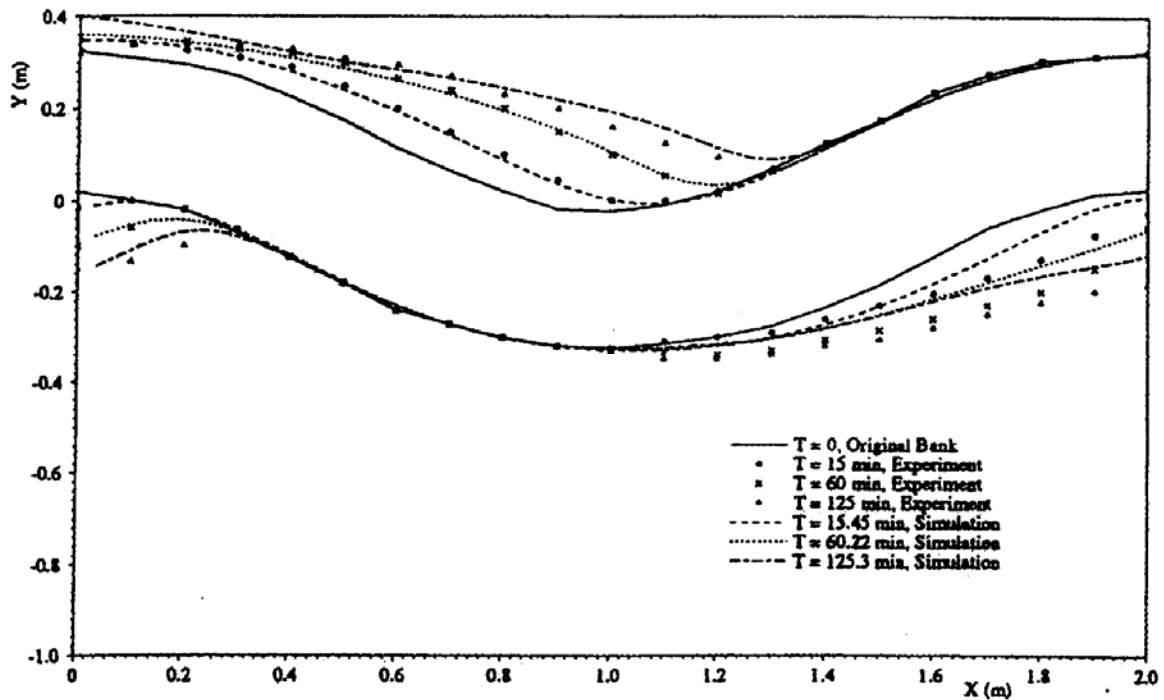


Figure 2.14 Comparison of experimental and simulated results (Duan et al., 2001)

The authors demonstrated that the model is capable of predicting the alternate bars in a straight reach of channel, and simulating the initiation process of a meandering channel. No quantitative comparison of the formation of bed form was made due to lack of data. The application of the model is still at the stage of experimental test.

2.3.9 Chen, H.-C. (2002)

The author used a Reynolds-Averaged Navier-Stokes (RANS) method together with a scour rate equation to calculate scour around simplex piers. Unsteady RANS equations were solved in a general curvilinear coordinate system. The same method is used to calculate maximum shear stress for the prediction of meander migration. The shear stress is used by the author of this dissertation.

2.3.10 Olsen, N. R. B. (2003)

Olsen developed a 3D CFD model which was based on the finite volume method using an unstructured grid with dominantly hexahedral cells. The $k-\varepsilon$ model was used to predict turbulence. The sediment transport was computed as bed load in addition to solving the convection-diffusion equation for suspended sediment transport. The CFD model computes the erosion as a function of the sediment transport formulas and the computed 3D flow field.

Verification was done on a laboratory experiment conducted by another researcher. Figure 2.15 displays how a straight channel developed into a meandering channel in the simulation. No comparison with the test results was made.

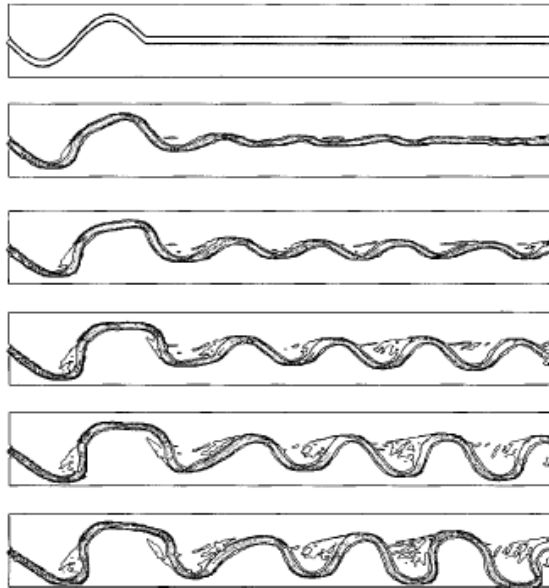


Figure 2.15 Simulated meandering process of laboratory case (Olsen, 2003)

2.3.11 Abad, J. D., Garcia, M. H. (2004)

A simplified river model was introduced for predicting river morphodynamics processes, including river migration. The methodology consists of components such as

designing stream restoration, linear and nonlinear analyses, and a planform migration model. The mean annual flood discharge was used for the verification of a real river. Partial agreement was observed. This method can be classified as a quasi-2D model.

2.3.12 Rodriguez, J. F., Bombardelli, F. A., Garcia, M. H., Frothingham, K. M., Rhoads, B. L., and Abad, J. D. (2004)

High resolution models were developed to simulate flow field. Two models are included: a depth-averaged model with secondary flow correction, and a fully 3D Computational Fluid Dynamics (CFD) model. Comparison with field data showed a successful simulation of the main flow features.

2.3.13 Jang, C.-L., Shimizu, Y. (2005)

In this article, the flow field was modeled by using the cubic interpolated pseudoparticle method. Secondary flow was considered in the sediment transport equation for the streamline and transverse transport to assess bed and bank evolution over time. In the 2D simulation, bank erosion occurs when the cross-sectional slope of the banks is larger than the submerged angle of repose. The model can reproduce the phenomena such as bar growth, and channel widening etc. It can also reproduce the features of braided rivers. Good agreements with laboratory experiments were observed.

2.3.14 Comments

Numerical simulation can simulate the flow field, sediment transport, and erosion process that happen in nature. This is a fundamental approach to reproduce the phenomena and make predictions. Due to the complexity of the natural governing processes, significant simplifications were introduced in the early stage of attacking this problem (Ikeda et al. 1981; Parker et al. 1982; Kitanidis and Kennedy 1984; Blondeaux and Seminara 1985; Odgaard 1989a). Including what are introduced above, several recent studies consider both bed deformation and bank-line shifting (Kovacs and Parker 1994; Nagata et al. 1996; Howard 1996; Duan et al. 1997, 2005a, 2005b; Mosselman, E. 1998; Sun, T. 2001a, 2001b; Darby, S. E. 2002; Abad and Garcia, 2004; Jang and Shimizu 2005). Most models have a limitation that the channel widening effect is not

taken into account. Darby and Thorne (1994, 1996a, b) and Pizzuto (1990) developed models to address this problem. The effect of downstream and upstream influence in river meandering was tackled by Zolezzi and Seminara (2001a, b).

The effectiveness of the numerical simulation results highly relies on the soundness of the models. The reliable modeling of each of the phenomena presents a challenge to researchers of Hydraulic Engineering. A review of the selected three cases indicates there is still a long way to go before numerical simulation can be practically used to predict the migration of real rivers.

Among the above three cases, Nagata and Duan's methods can verify the bank movements of the flume tests quantitatively. Nagata's simulation of bed-deformation has the same trend as that of the tests. No quantitative verification is seen in Olsen's article. For all the three cases, no verification was done on a real river. Darby et al. (2002) did a verification for his model on a reach of Goodwin Creek, Mississippi. Good agreements were reported. Even for the presented good quantitative agreements between simulation and test or field data, there should exist a significant amount of manual adjustment of involved parameters. Not being able to consider a hydrograph is another weakness of the numerical methods because rainfall induced flood is the major cause of bank erosion.

2.4 SRICOS-EFA METHOD (Briaud et al. 2003)

From the literature review we know that the major factors that affect meander migration are soil, water, and geometry. The factors of water and geometry have been well addressed in the literature. But the factor of soil is often treated as an empirical soil erodibility coefficient. The topic of this research is "Develop guidance for soils properties-based prediction of meander migration". Soils are given extreme importance and soil erodibility is treated as a fundamental property. Previous scour research projects (Briaud et al. 2003) conducted at Texas A&M University provided a good knowledge about soil erodibility. The model used to predict scour depth of bridge piers will be adapted for the prediction of meander migration.

2.4.1 Erosion Function Apparatus (EFA)

Soil erosion is an interactive process between soil and water occurring at the interface of soil and water. As velocity increases, the shear stress τ imposed by the water on the soil particles becomes large enough to overcome the bonding force. For coarse grained soils erosion occurs by rolling and sliding of particles. Fine grained soils may erode particle by particle but electromagnetic and electrostatic forces play an important role (Briaud et al. 2001a). The threshold shear stress at which the erosion is initiated is called critical shear stress τ_c .

EFA was invented by Jean-Louis Briaud at Texas A&M University in 1991 (US Patent No. US6260409B1 July 17, 2001). Figure 2.16 shows the conceptual diagram and test section of the machine. Soil sample sit in an ASTM standard Shelby tube with a 76.2 mm outside diameter (ASTM 1999a). A motor pushes the sample until it protrudes by 1mm. The protruded soil erodes under flows of velocity varying from 0.1m/s to 6m/s. The procedure for EFA test is as follows:

1. Place the sample in the EFA, fill the pipe with water, and wait one hour.
2. Set the velocity to 0.3 m/s.
3. Push the soil 1 mm into the flow.
4. Record how much time it takes for the 1 mm soil to be eroded.
5. When the 1 mm of soils is eroded or after 1 hour of flow, whichever comes first, increase the velocity to 0.6 m/s and bring the soil back to a 1 mm protrusion.
6. Repeat step 4.
7. Next, repeat steps 5 and 6 for velocities equal to 1 m/s, 1.5 m/s, 2 m/s, 3 m/s, 4.5 m/s, and 6 m/s.

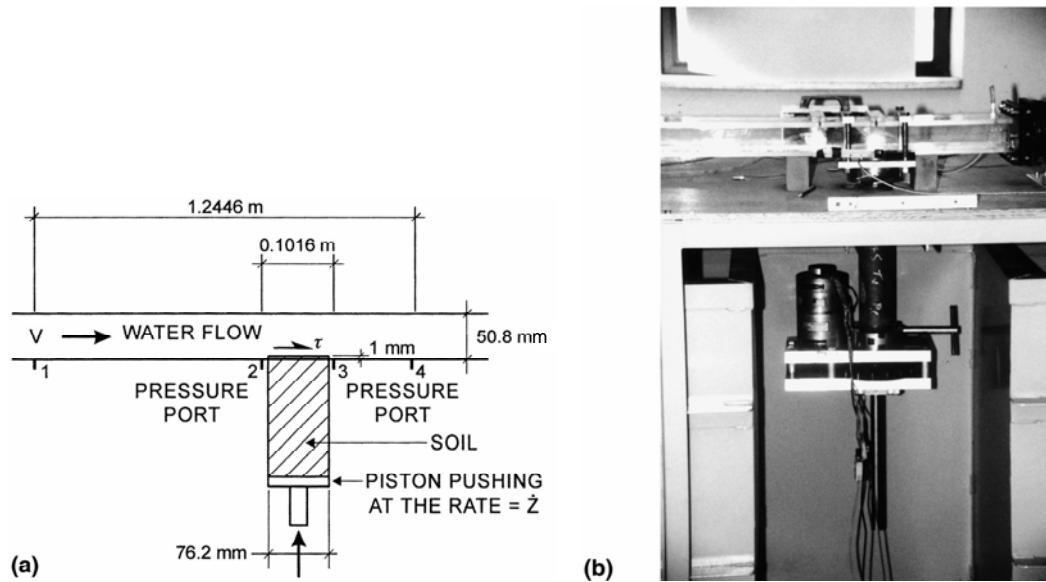


Figure 2.16 EFA: (a) Conceptual diagram; (b) Photograph of test section (Briaud et al., 2001a)

Direct test result is an erosion rate \dot{z} versus velocity curve. Erosion rate is the height of the soil eroded divided by the time taken. Shear stress cannot be measured directly and needs to be solved in an indirect manner.

Shear stress at the soil-water interface is caused by the flow and the magnitude is related to the roughness of soil surface. The pressure difference between point 2 and 3 is caused by the friction across the sample. By drawing a free body diagram, the shear stress can be calculated with reference to the pressure difference. But the pressure head difference is small and fluctuates. It was found that the best way to calculate shear stress is to use the Moody Chart (Moody 1944, Figure 2.17). The shear stress τ is:

$$\tau = \frac{1}{8} f \rho v^2$$

Where,

- τ : Shear stress on the wall
- f : Friction factor obtained from the Moody Chart
- ρ : Mass density of water (1000kg/m^3)
- v : Mean flow velocity in the pipe

On the Moody Chart, Re is Reynolds number; μ is dynamic viscosity of water; $\nu = \mu/\rho$ is kinematic viscosity of water ($10^{-6} \text{ m}^2/\text{s}$ at 20°C); and D is pipe diameter. For non-circular shape, $D=4A/P$ where A =cross-sectional flow area; P =wetted perimeter. For a rectangular cross section pipe, $D=2ab/(a+b)$, where a and b are dimensions of the sides of the rectangle. ϵ is the average height of roughness elements. It is taken as $(1/2)D_{50}$ where D_{50} is the mean particle diameter for the soil. ϵ/D is a measure of relative roughness.

Figure 2.18 is the result of a test on a clean coarse sand ($D_{50}=3.375 \text{ mm}$). The critical shear stress for this soil is $\tau_c=3 \text{ N/m}^2$.

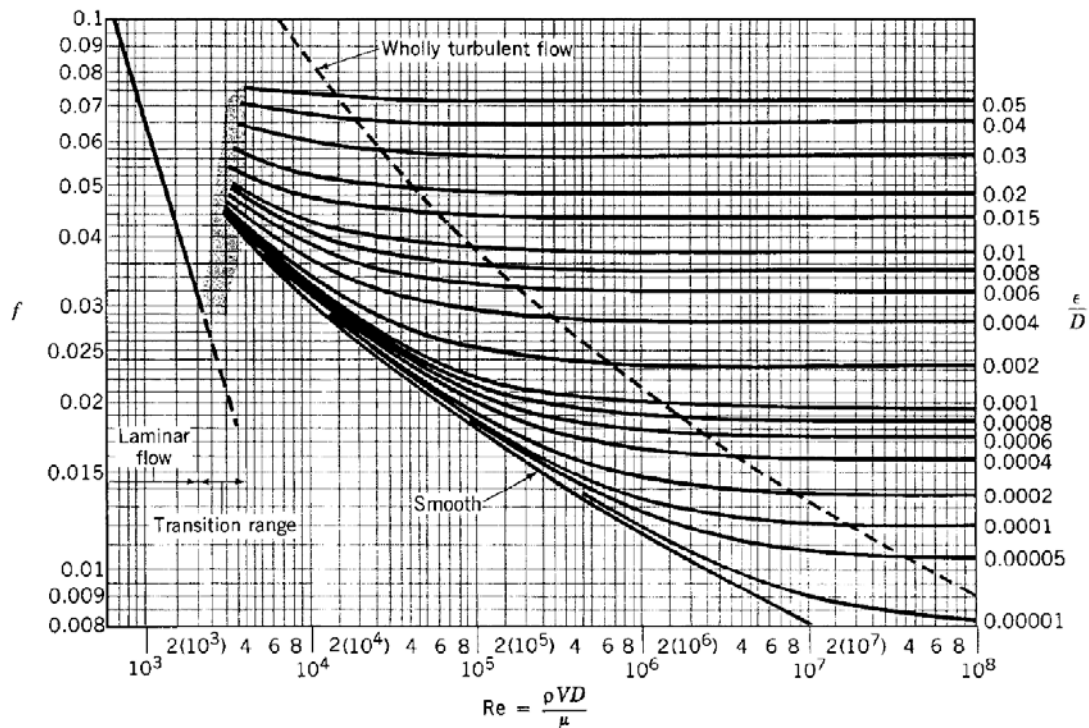


Figure 2.17 The Moody Chart (Munson et al., 1990)

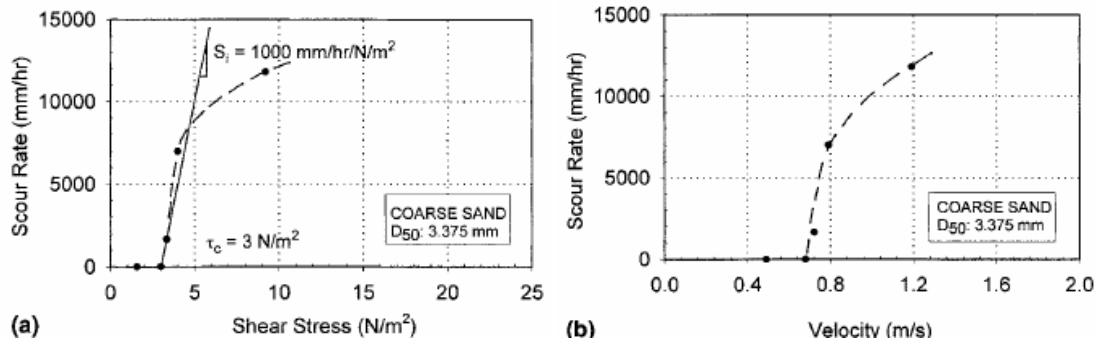


Figure 2.18 Erosion curve for coarse sand (Briaud et al. 2001a)

A lot of experiments have been done to establish relationships between erosion function and common soil properties that can be easily obtained in a regular soils lab. The erosion function is a non-linear relationship between erosion rate \dot{z} and shear stress τ which requires a lot of parameters to do curve fitting. Critical shear stress τ_c and initial slope of the curve S_i are two of the parameters involved. In order to find the correlation between erodibility and soil parameters, one would try to find the correlation between τ_c , S_i and soil parameters. For coarse grained soils, τ_c is proportional to D_{50} . For Fine grained soil, there is no such correlation existing. Experimental data also show weak correlation between τ_c , S_i and soil properties such as undrained shear strength, plasticity index, and percentage passing #200 sieve etc. So it is not realistic to obtain erodibility from common soil properties. A direct measurement with the EFA for a specific site is favored.

2.4.2 SRICOS-EFA method

FHWA hydraulic circular HEC-18 recommended an equation for calculating maximum scour depth in sand. The erosion mechanism for clay is quite different from sand. Therefore Briaud et al. (1999) developed a method called SRICOS which stands for Scour Rate in Cohesive Soils. It deals with pier scour in cohesive soils under constant velocity. With varying velocity and multiple soil layers taken into account, the

method is extended to SRICOS-EFA method. This method has been successfully applied in bridge scour projects. The computer program is free for download at <http://ceprofs.tamu.edu/briaud/sricos-efa.htm>.

2.3.2.1. Maximum shear stress τ_{\max}

At the soil-water interface, the flow is tangential to the soil surface. The shear stress imposed by the flowing water affects the erosion process. The water velocity in a river is in the range of 0.1 to 3 m/s, while river bed shear stress ranges from 1 to 50 N/m² and increases with the square of water velocity. Equations for the shear stress of flat river bed have been developed. As for the case of a circular pier, numerical simulation was performed to analyze the shear stress around it. The shear stress goes down with the increase of scour depth until it reaches the critical shear stress τ_c and the erosion stops. Figure 2.19 shows how shear stress decreases with scour depth at a scour hole. The maximum shear stress τ_{\max} occurs at soil surface when the scour of the pier initiates. Extensive numerical simulation led to the following equation for τ_{\max} :

$$\tau_{\max} = 0.094\rho V^2 \left(\frac{1}{\log R_e} - \frac{1}{10} \right)$$

Where,

ρ : Mass density of water (1000 kg/m³)

V : Mean flow velocity (m/s)

R_e : $=VD/\nu$, Reynolds number

D : Pier diameter (m)

ν : Kinematic viscosity of water (m²/s)

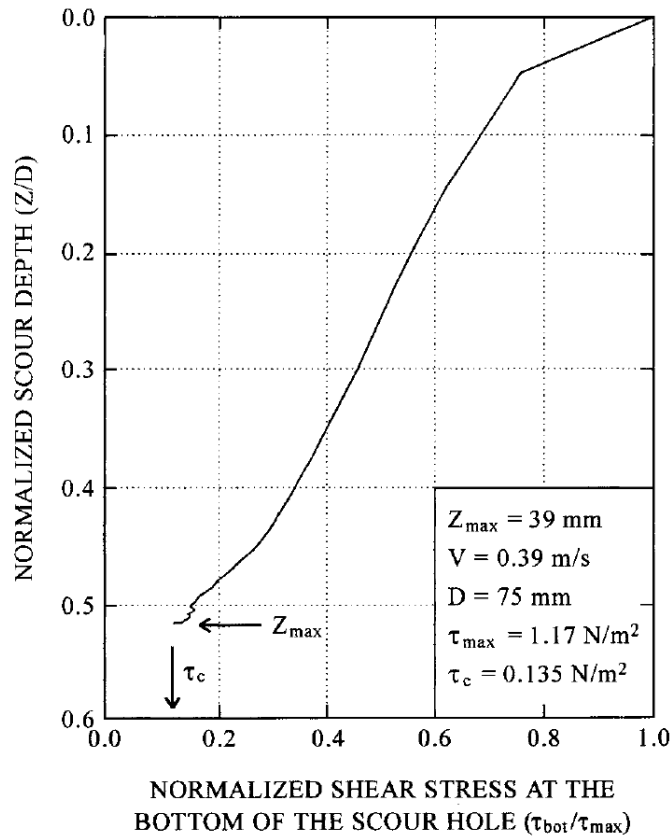


Figure 2.19 Variation of shear stress at bottom of scour hole as function of depth of scour hole
(Briaud et al., 1999)

2.3.2.2. Maximum scour depth z_{\max}

Numerous flume tests have been conducted to study the scour around piers. It has been observed that the erosion stops under a constant velocity when a maximum scour depth is reached. Figure 2.20 shows how scour depth changes with time in a flume test. Several types of curves were used to fit the data. Hyperbola curve was found to be the best. The equation is as the following:

$$z = \frac{t}{\frac{1}{\dot{z}_i} + \frac{t}{z_{\max}}}$$

Where, \dot{z}_i is the initial slope of the z versus t curve and z_{\max} is the asymptotic value of the hyperbola curve which stands for the final scour depth at $t = \infty$. Based on flume test data, the correlation between z_{\max} and various parameters was studied. The most well behaved relationship was found to be between z_{\max} and Reynolds number R_e :

$$z_{\max} (\text{mm}) = 0.18R_e^{0.635}$$

The definition of R_e is the same as the one in the τ_{\max} equation. This relationship is good for both sand and clay which is also confirmed by making a comparison with HEC-18 equation. Water depth was varied in the flume tests but it had very little influence. The most important factors are the mean flow velocity V and the pier diameter D .

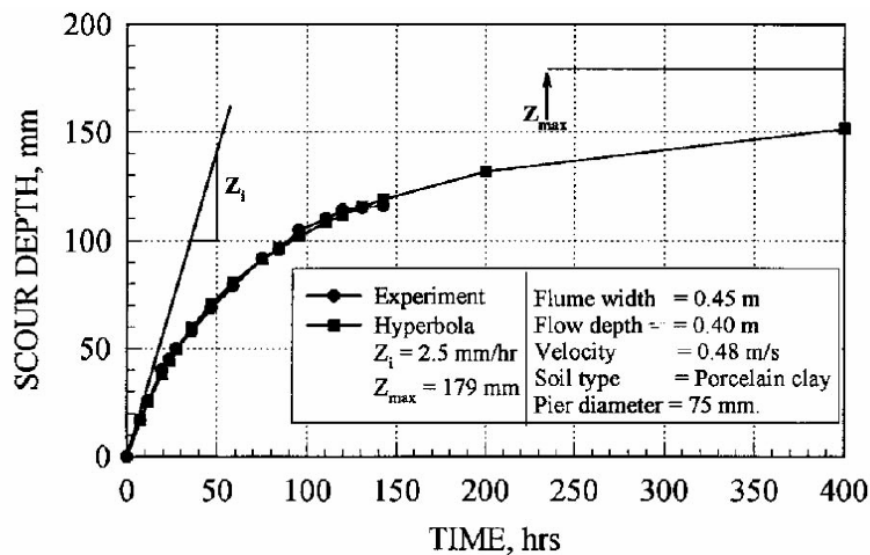


Figure 2.20 Scour depth versus time curve (Briaud et al., 1999)

2.3.2.3. Procedure of SRICOS-EFA method

Given constant flow and uniform soil, \dot{z}_i and z_{\max} can be calculated. With these two parameters ready, a scour depth z versus time t curve can be obtained. Detailed SRICOS method consists of:

1. Collecting Shelby tube samples near the bridge pier,
2. Testing them in the EFA (Figure 2.16) to obtain the erosion rate \dot{z} (mm/hr) versus hydraulic shear stress τ (N/m²) curve,
3. Calculating the maximum hydraulic shear stress τ_{\max} around the pier before scour starts,
4. Reading the initial erosion rate \dot{z}_i (mm/hr) corresponding to τ_{\max} on the \dot{z} vs. τ curve,
5. Calculating the maximum depth of scour z_{\max} ,
6. Constructing the scour depth z versus time t curve using a hyperbolic model,
7. Reading the scour depth corresponding to the duration of the flood on the z vs. t curve.

In real world, the flow velocity in a river changes from hour to hour and from day to day. Within a short period of time, the flow can be treated as constant and the SRICOS method can be applied. The hydrographs downloaded from USGS website are a history of daily average flows. For this case the short period time with constant flow is one day and each day may have a different z versus t curve. Looking at the scour hole on a certain day, one can see that it has two properties: one is existing scour depth z_0 and the other is the z versus t curve (\dot{z}_i, z_{\max}) of that day. These two properties determine how much erosion will be caused by that day's flow. The accumulation of the influence of each day's flow can be done in this way. To calculate the scour depth Δz caused by a constant flow of time Δt , two cases must be considered as shown in Figure 2.21. In case 1, existing scour depth z_0 is less than z_{\max} , where the erosion starts from the point corresponding to z_0 at time t_e and continues for time Δt . At the end of time Δt , the total scour depth is $z_0 + \Delta z$. t_e is called equivalent time which is the time would be needed for the current flow to cause the same amount of scour depth as the accumulated scour depth caused by all previous flows. In case 2, existing scour depth z_0 is larger than z_{\max} , which means the flow is not able to cause any additional erosion. There is no meaning for t_e here. If $z_0 < z_{\max}$, the expression for equivalent time t_e is:

$$t_e = \frac{1}{\dot{z}_i \left(\frac{1}{z_0} - \frac{1}{z_{\max}} \right)}$$

The scour depth increment caused by the constant flow of time period Δt is:

$$\Delta z = \frac{t_e + \Delta t}{\frac{1}{\dot{z}_i} + \frac{t_e + \Delta t}{z_{\max}}} - z_0$$

If a hydrograph has 10,000 flow data, the calculation will be repeated for 10,000 times. The computer program SRICOS-EFA can do all the computation and output the scour depth z versus time t curve.

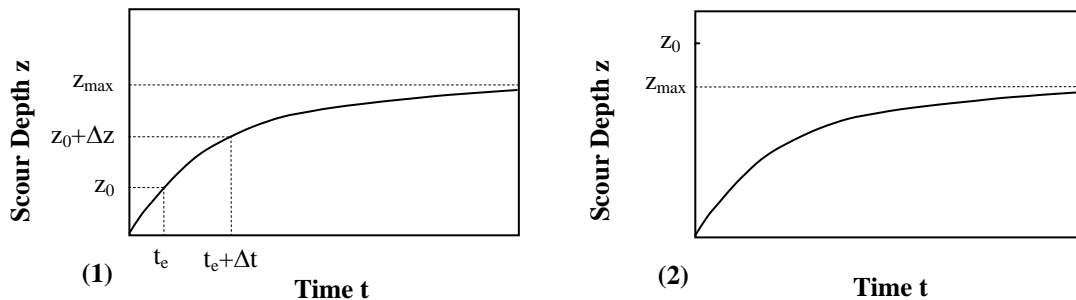


Figure 2.21 Accumulation of scour depth: (1) Additional erosion occurs; (2) No additional erosion occurs

CHAPTER III

RESEARCH OBJECTIVES AND METHODOLOGY

3.1 RESEARCH OBJECTIVES

This research is aimed at working on these issues:

1. Apply a soil erosion model to the prediction of meander migration.
2. Based on the model, study the elements affecting meander migration: geometry, soil and water.
3. Conduct risk analysis so that the prediction of meander migration can be made on a probabilistic approach.
4. Develop a program to assemble all the work of the team members and deliver the program as a final product to the sponsor.

3.2 METHODOLOGY

The literature shows a variety of solutions to this tough problem. However none of them gives satisfactory result. This research tries to apply a good methodology and provide a better solution. The methodology consists of:

1. Flume test;
2. EFA test;
3. Numerical simulation;
4. Fundamental modeling;
5. Risk analysis;
6. Development of the MEANDER program.

All the work is done by a team of six. For the work that is done by other team members, the contributors will be mentioned explicitly.

3.2.1 Flume test

It is well known that physical model tests are important in understanding and solving hydraulic problems in rivers. In contrast to numerous uncontrollable variables and parameters, and sometime the unpredictable nature of field data, physical model

tests are performed under a well-controlled environment so that essential parameters can be chosen, adjusted, and varied systematically to quantify their influence on the problem. Not only do flume tests directly give physical insight on meander migration, they also counter the worry about the correctness of usually simplified theories as well as the inevitable errors due to spatial and temporal discretizations in numerical simulations.

In the flume test, these parameters will be studied systematically: soil properties, channel geometry, and velocity. Fine sand and clay will be used as bank material. The ratio of radius of curvature of a bend to channel width is treated as the geometry parameter. Cross sectional average velocity is measured and analyzed. A reference case is set. Each test varies only one parameter from the reference case.

The first set of flume tests were done in the old Hydro Lab. Later tests were moved to the new Coastal Engineering Lab. The author is responsible for the preparation of the setup in the old Hydro Lab and running the first 13 flume tests.

3.2.2 EFA test

A big advantage of this study is that EFA (Briaud et al. 1999, 2001a) is used to test the erodibility of soils at specific sites, which is a fundamental approach to model soil erosion behavior. The obtained erosion curves are fundamental properties of the soil. With the application of this test, it is more likely to do accurate modeling of soils. The EFA tests were done by Jun Wang from Pier Scour team.

3.2.3 Numerical simulation

The flow field of a meandering channel is a transient 3-D free surface flow with strong secondary flow and a moving streambed boundary. Computation of such complex flow will require a more advanced numerical model than that of a steady flow with fixed boundaries. Analyses for this purpose can generate maximum shear stress as a function of the major factors affecting meander migration. In this study, it is proposed to employ an existing state-of-the-art Reynolds-Averaged Navier-Stokes (RANS) code in conjunction with a flexible chimera domain decomposition technique for accurate and

efficient prediction of stream migration problems. This part is done by Prof. Hamn-Ching Chen.

3.2.4 Modeling of prediction

Many available modeling approaches consist of a velocity model and a bank erosion model. In verifying these models, the calculated velocities, instead of the recorded hydrograph are used. These methods don't consider that floods contribute to most part of meander migration and rainfall is the most important reason for floods. This research proposes the application of Hyperbolic Model which assumes the migration rate gradually reduces to zero when the major factors are kept constant. A hydrograph can be applied so that the true migration process can be simulated. This model was developed years ago and has been successfully applied in Pier Scour project. The expression is:

$$M = \frac{t}{\frac{1}{\dot{M}_i} + \frac{t}{M_{\max}}}$$

Where, \dot{M}_i is the initial migration rate, obtained from EFA curve of the soil based on τ_{\max} . M_{\max} is the maximum migration distance under a certain condition. Both τ_{\max} and M_{\max} can be expressed as functions of the following parameters:

$$\tau_{\max}, M_{\max} = f(\phi, \theta, R, W, y, v(t), \{Z - \tau\})$$

Where,

- ϕ : Bend angle defined by the two inflection points of a bend;
- θ : Location angle, from the first inflection point to the point of interest;
- R : Radius of curvature of the bend. Use the radius of the fitted circle;
- W : Channel width. Use width from aerial photos;
- y : Water depth. Calculate from hydrograph Q vs. t, knowing cross section and using HECRAS;
- $v(t)$: Average velocity of a certain period of time.;

$\{\dot{Z} - \tau\}$: EFA curve, erosion rate versus shear stress.

The numerical simulation was done by Prof. Hamn-Ching Chen. The τ_{\max} equation was developed by the author of this dissertation based on the shear stress data from numerical simulation. The equation for M_{\max} will be developed from flume tests by Po-Hung Yeh and Namgyu Park (Park, 2006; Yeh, 2006).

Like developing the equations for τ_{\max} and M_{\max} , calculation of geometry parameters R , ϕ , θ is another critical issue. The idea of geometry study is to write a program to simplify a curvy channel into circles and straight lines. Circles are chosen to represent the bend shape due to its simplicity and good match in most situations. The fitting process needs to be done automatically because it will be repeated thousands of times when a hydrograph is applied. Geometry study is a major task of this research.

3.2.5 Risk analysis

Due to the many uncertainties in the factors affecting meander migration, deterministic prediction is not very meaningful. Prediction in a probabilistic manner is a more appropriate approach. Like what we know in weather broadcasting, the occurrence of a certain event is associated with a probability. The engineers are allowed to choose a predicted migration distance based on a chosen confidence level.

The most uncertain factor is what level of discharge a river will experience on a day in the future. It is reasonable to assume that the statistical properties of future flow are the same as the past. New future hydrographs can be generated based on these statistical properties. Each future hydrograph produces a different location for the future river. If tens of thousands of future hydrographs are applied, a 2-D distribution of the locations of the future river can be obtained. Thus an engineer can tell the probability of the river reaching this line and further is approximately 10%.

The author is responsible for generating future hydrographs and calculating the new location of the channel corresponding to each hydrograph. Namgyu Park (2006) will make probabilistic predictions based on the new locations of the channel.

3.2.6 Develop the MEANDER program

To apply the methods developed in this research, it's impossible for practice engineers to calculate migration distance by hand. A user friendly computer program is necessary to assemble the products of all team members. The program includes the following components:

1. Geometry study. This is a major effort for this program. The user can read in the coordinates of a river. The program automatically fits circles to each bend of the river. Radii of curvature and bend angles are returned accordingly. If a hydrograph is applied, the automatic fitting process is repeated for each time step.
2. Input of soil properties. The EFA curves are input here.
3. Input of water data. The following information goes here: discharge vs. velocity curve, discharge vs. water depth curve, existing hydrograph. The user can choose to do risk analysis. Then the risk analysis period, a hydrograph or Q_{100} , Q_{500} (100/500 year flood) are needed.
4. Implementation of Hyperbolic Model. With the above information and the equations for τ_{\max} and M_{\max} ready, Hyperbolic Model is implemented. The code for risk analysis is also here. This process goes on behind the user interface.
5. Input Plots. This part graphically checks whether the input data look correct.
6. Output Plots. The migrated channels for a single hydrograph or risk analysis are presented here.

The program was written in Matlab and Visual C++. Matlab compiler was used so that the two parts can be integrated together.

CHAPTER IV

FLUME TEST

In the Pier Scour project (Briaud et al. 2003) around 100 flume tests were conducted. The data shed light on the process and mechanism of the scour of bridge piers. It wouldn't have been possible to develop the SRICOS-EFA method without flume tests showing the trend, providing basis for the hypothesis of hyperbolic model, and furnishing data for the verification of the model. Just as flume test was important to the pier scour problem, it also plays a critical role in the prediction of meander migration. In the flume tests for Meander Project, the factors of soil, flow, and geometry were controlled and varied to study their influence on migration rate.

Dozens of flume tests have been conducted at two labs of Texas A&M University. One was the old Hydro Lab where the test section was 9.0 meters by 1.5 meters. The other one was the new Coastal Engineering Lab on west campus where the test section was 27 meters by 13 meters, 25 times larger than the old one. The author prepared the setup and conducted the first 13 flume tests in the old Hydro Lab. So only this part will be described in detail.

4.1 EXPERIMENTAL SETUP

The setup in the old Hydro Lab consisted of these parts: flume, false bottom, soil tank, water circulation system, and data measurement equipments, as shown in Figure 4.1. The flume was 1.5-meter wide and 3.0-meter deep. At the end of the flume was a reservoir of 3.9 meters in depth. In order to keep the setup of flume test for Pier Scour Project for possible future use, only 16.6 meters of the flume was allocated for this test.

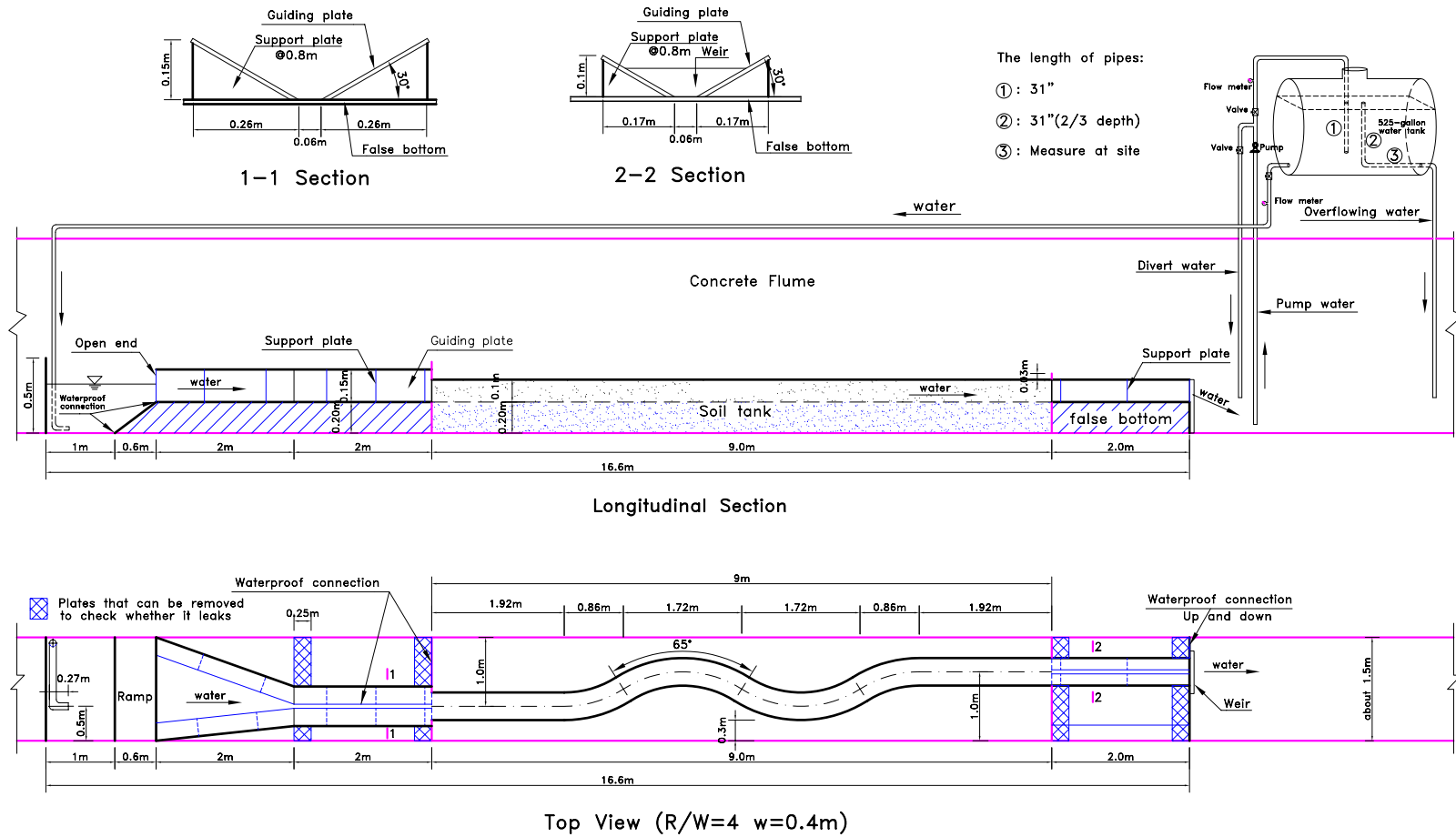


Figure 4.1 Experimental setup for flume test in the Old Hydro Lab

The function of false bottom was to reduce turbulence before water flowed into model channel and to avoid a sudden change of channel cross section on the entry and exit. The upstream false bottom had a ramp of 0.6 meters in length followed by a 2.0-meter long transitional channel with varying cross section. The wooden channel connecting the transitional channel with the soil tank was also 2.0 meters long and had the same cross section as the sand channel (Figure 4.2). The tiny pool preceding it held water flowing from water tank 3.0 meters above. The strong turbulence in the pool was reduced on the transitional channel and finally almost disappeared before it reached the soil tank. The downstream false bottom had a 2.0-meter long channel of the same cross section as the model channel. A weir was attached to the end to control water level.

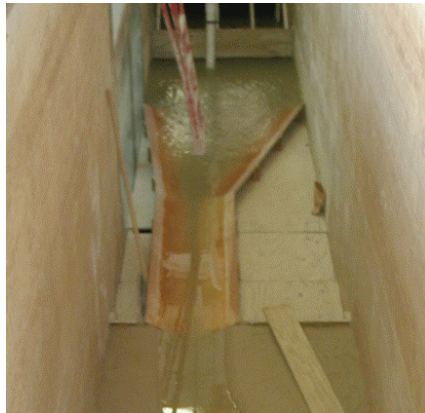


Figure 4.2 Upstream false bottom

The soil tank lay between the upstream false bottom and downstream false bottom. It was 9.0 meters long, 1.5 meters wide and 0.3 meters in depth. About 4.0 cubic meters of mortar sand was used to fill the tank. Channels of predetermined geometry were carved in the sand. Figure 4.3 shows a channel in preparation. Due to the limitation of space, the margin between the apex of a bend and the wall was only 0.25 meters. The test had to stop when the wall was reached. This situation was greatly improved when

the test was moved to the new Coastal Engineering Lab where the margin was about 1.4 meters.



Figure 4.3 Carving the channel

Water circulation system consisted of a pump, a water tank, piping system, a channel, and a reservoir. The pump and the water tank were on the ground floor, while the soil tank was 3.0 meters below. Water was pumped from the reservoir into the tank and then flowed down to the channel. The reservoir was both the start and end point of the circulation. Diverting pipe was installed to get rid of extra pumping capacity. The water tank had a volume of 2.0 m^3 (525 gallons). Three pipes were connected to the tank, including the overflowing pipe which was about $2/3$ of the tank's height and kept water level at its top end. Several valves worked with the diverting pipe and overflowing pipe to adjust the flow. The 3-meter free drop provided enough momentum for the water to flow through the soil tank at required velocity.



Figure 4.4 A finished channel

Figure 4.4 shows a prepared channel ready for water flow. Data measurements during the test included: the flow rate and flow velocity in the channel, the locations of the original and migrated channel banks, water depth, and cross sectional profile of water flow. The flow meter from the old EFA was used to measure the flow rate in the pipe. Before using the flow meter, a calibration was conducted.

A carriage across the two walls of the flume ran back and forth over the test section. Acoustic Doppler Velocimeter (ADV), a computer, and a digital camera were installed on the carriage.

4.2 MEASUREMENT OF VELOCITY AND GEOMETRY CHANGE

Three methods have been used to measure flow velocity in the channel: 1. ADV (Figure 4.5); 2. Paper tracer (also called paper boat); 3. With flow rate given, measuring the area of the cross sectional profile of the flow. ADV can accurately measure the velocity at a certain point. But when water depth is less than 5 cm, noise becomes dominant, as can be seen in Figure 4.6 and Figure 4.7. This method was abandoned due to frequent occurrence of shallow water depth. Method 2 was to drop a piece of tiny paper scrap on water surface and measure its traveling speed. The result well matched visual observation of the flow. What was measured was mostly the maximum velocity of

the surface flow. This data showed a gradual change of velocity along the channel. The largest difference could be 10 cm/s. Cross sectional average velocity didn't fluctuate that much along the channel. In real rivers, what can be obtained is also cross sectional average velocity. For these reasons, method 3 was preferred over method 2.

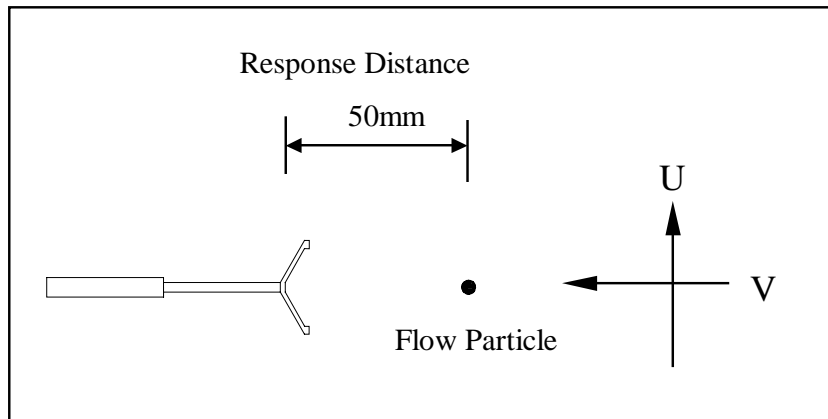


Figure 4.5 Diagram of a 2D ADV (Li, 2002)

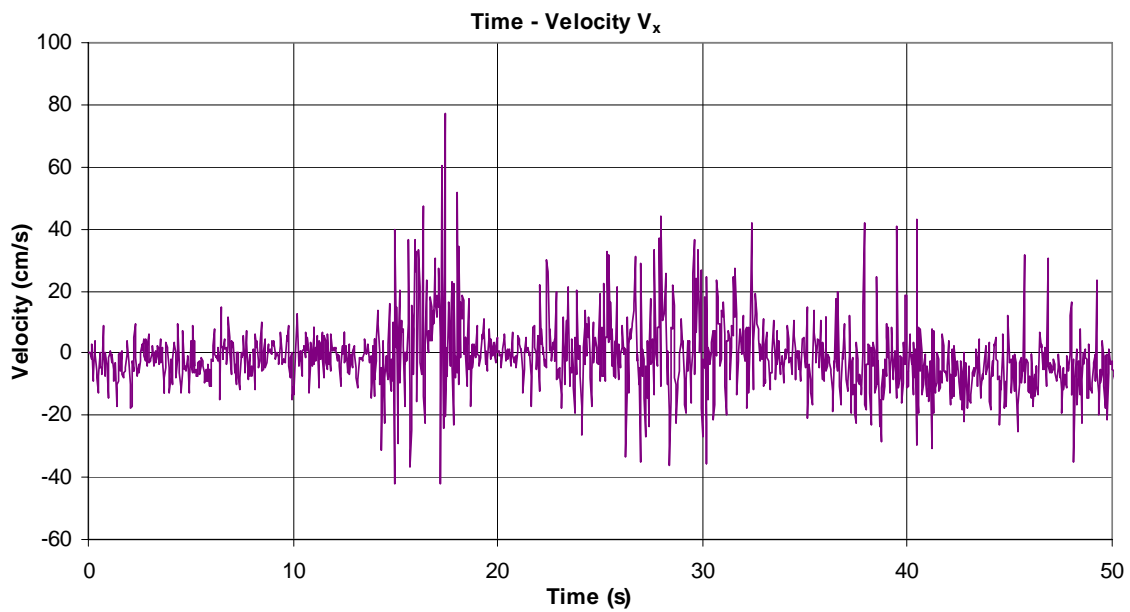


Figure 4.6 Sample ADV data from flume test 9 in the old Hydro Lab (V_x)

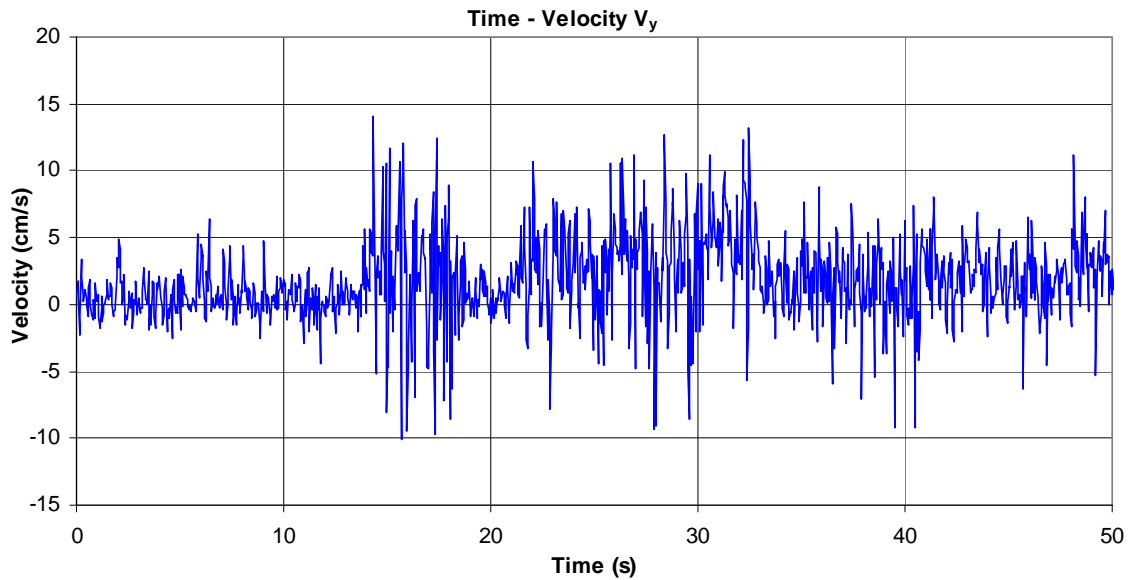


Figure 4.7 Sample ADV data from flume test 9 in the old Hydro Lab (V_y)

Point Gauge (Li 2002) is a useful tool for measuring water depth with high precision. It is basically a circuit consisting of a stiff rod with wire attached and a needle probe fixed to one end, conductive wire, voltmeter, water and soil. The wire is tied to a tiny steel block which is placed on the ground. When the probe touches water surface, a circuit is formed. When the probe touches soil surface, water is bypassed in the circuit since soil has much larger conductivity. These changes are clearly shown on the voltmeter. Location of the needle probe can be recorded at the moment when the voltmeter reading changes. This equipment was frequently used in the Pier Scour Project and was also used in this flume test. A high precision measurement can be achieved for clay. It also produces a good result for sand for which the change in voltmeter reading is not as obvious as that for clay. When the flow is clear, a ruler can also measure water depth with a lower precision. Compared to a ruler, the point gauge needs a lot of manual adjustment which is time consuming. In measuring the cross section profile of the flow, one measurement was done every 5 cm across the channel. Due to the large amount of time a point gauge would need, a ruler was used instead.

A point gauge was very useful in measuring the slope of water surface and channel bed. The carriage holding the point gauge slid along two horizontal tracks on the ground (Appendix B). The tracks were not level by themselves. A level surface needed to be established as a reference plane. Before a test started, the channel was filled with static water the surface of which was treated as the reference plane. When the probe touched the water surface, what was measured could be considered as nominal vertical distance between a point on the track and the reference plane. To get the exact distance, the height of the carriage should be added. The nominal vertical distances of all the stations indicated how much the tracks were above or below a certain level plane at different points. If each station had the same nominal vertical distance, the tracks were on a level plane. Any point gauge reading relative to the tracks could be made relative to a level plane by subtracting corresponding nominal distance. Figure 4.8 shows the slope of water surface and channel bed at different times of the 11th flume test. Elevation 0.0 was the reference level plane. The lines above reference plane were free water surfaces. The lines below were channel beds.

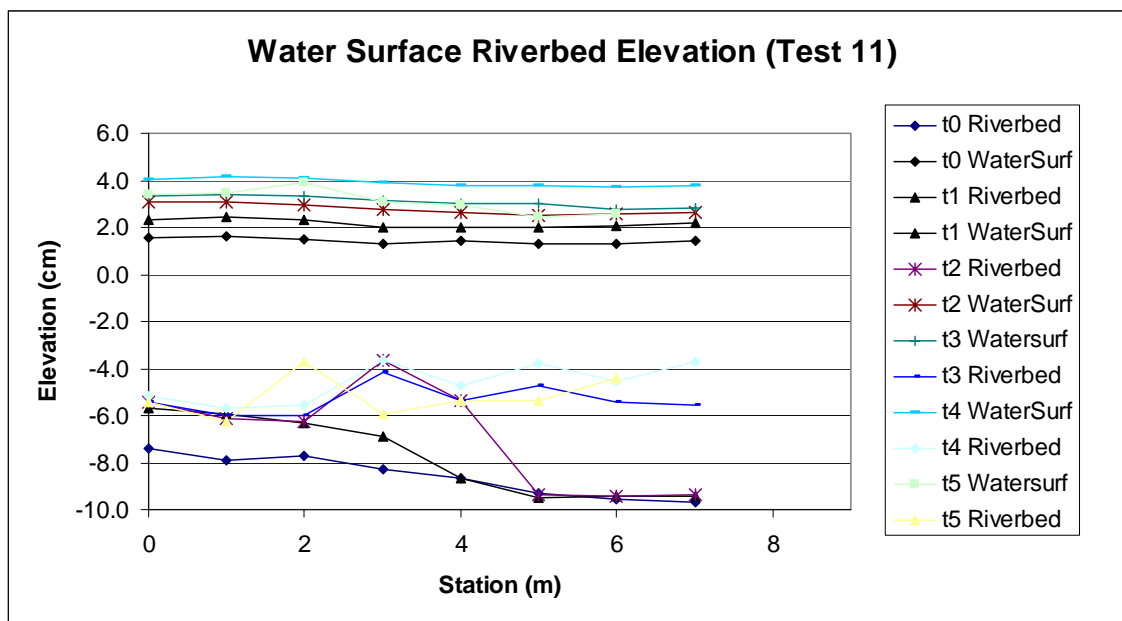


Figure 4.8 Measurement of the slope of water surface and channel bend

Photogrammetry is the technique of obtaining 2-D/3-D geometric information about physical objects by processing and taking measurements on photographic images. This technique was proposed for measuring the location of migrated banks. The idea was to obtain coordinates of the banks by analyzing digital pictures which were original records of the migrating channel at certain stages. A lot of efforts had been input into developing and applying this technique. Two major issues about this technique were: correction of distortion of pictures and automatic identification of boundaries. Distortion occurs when the camera doesn't shoot perpendicular to the object surface, or the lighting is bad, or the atmosphere is not uniform, or the lens is defect. Lane et al. (2001) provided a solution to this problem. This error could be ignored in this test due to the high resolution and the way of shooting pictures. But automatic identification was a challenge. A free program from the internet called WinDIG can identify clear boundaries on raster images. In these flume tests, water was mostly clear. Sometimes it is even hard to tell the water soil interface with bare eyes, not mention relying on WinDIG to do the job. Figure 4.9 is a typical case for which WinDIG cannot automatically identify the boundaries. Taking and analyzing pictures took more time than manual measurement with a tape measure. But the future of photogrammetry technique is promising when the boundaries can be made more distinguishable on site and more automation can be incorporated into the analyzing process. With a tape measure, normally one point was measured every 20 cm along the reference wall. More points were inserted where there was a sudden change in geometry.



Figure 4.9 A picture for applying photogrammetry technique

4.3 CALIBRATION OF THE FLOW METER

The flow meter used in this test consisted of a flow sensor and a flow transmitter (Figure 4.10). The flow sensor was inserted into a 3-inch PVC pipe and the transmitter converted mechanical signal into electric signal. A digital voltmeter was connected with the transmitter to read the output voltage. A calibration curve relating voltage to flow rate was needed. The available commercial service on campus was for 4-inch pipe and the price was not very reasonable. It was decided that the calibration should be done in the lab with the original piping system.

A calibration slot was installed between the pump and the water tank to hold the flow sensor (Figure 4.1). Flow rate was adjusted by tuning the diverting gate valve. A certain amount of water was pumped and flowed through the flow meter into the water tank. The voltmeter reading was taken every 5 seconds until the overflowing pipe started to receive water. Accurate measurement of the amount of water in the tank was critical to the calibration. Two methods were applied and four measurements were done to ensure accuracy. First method was to let tap water from the university water supply flow into the water tank where constant flow rate was assumed. The second method was to pump water out of the water tank to fill two buckets alternatively and count how many

buckets of water there were. One measurement was discarded due to the big difference from other three. The results are shown in Table 4.1. The volume divided by the time needed to pump it was the average flow rate the flow meter experienced.

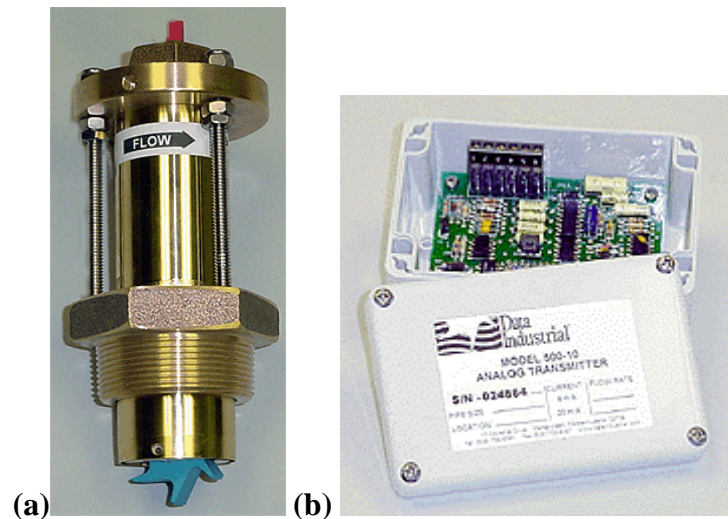


Figure 4.10 Flow meter: (a) Flow sensor; (b) Flow transmitter (source: www.dataindustrial.com)

Table 4.1 Calibration of the flow meter

	Tap water → Water Tank	Water Tank → Buckets 20226 ml/bucket, D=28.5 cm
1 st measurement	t=58'38", Q=485.5 ml/s (7.7GPM) vol=1708.7 liters (451.45 gallons)	73 buckets+3.5 cm deep vol=1172.2 liters (Discarded)
2 nd measurement	t=47'30", Q=612.53 ml/s (9.7GPM) vol=1745.7 liters (461.2 gallons)	84 buckets+9.0 cm deep vol=1704.7 liters (450.4 gallons)

Figure 4.11 shows the voltage versus time curves for several flow rates. V0 indicated the diverting valve was fully closed which meant the flow going through the

flow meter was the largest. V5.5 meant the diverting valve was turned open by 5.5 rounds. V12 indicated the diverting valve was fully open which meant the flow going through the flow meter was the smallest. The pumping capacity of the pump was considered as stable. The flow going through the flow was adjusted by the diverting valve. During the first minute, the voltmeter reading fluctuated a lot. This was because the pipe was not full yet and there existed some turbulence. The pipe coming into the water tank ended at $2/3$ of the depth from the top. When the pipe opening was submerged in water, the rising water level started to reduce pumping capacity, which explained why the voltage steadily went down. For most flow rates, two measurements were taken and the results were very close. Although the voltage and the flow rate changed during the calibration, an average voltage can be reasonably used due to the linearity of the change. Twelve flow rates were calibrated and the calibration curve is shown in Figure 4.12.

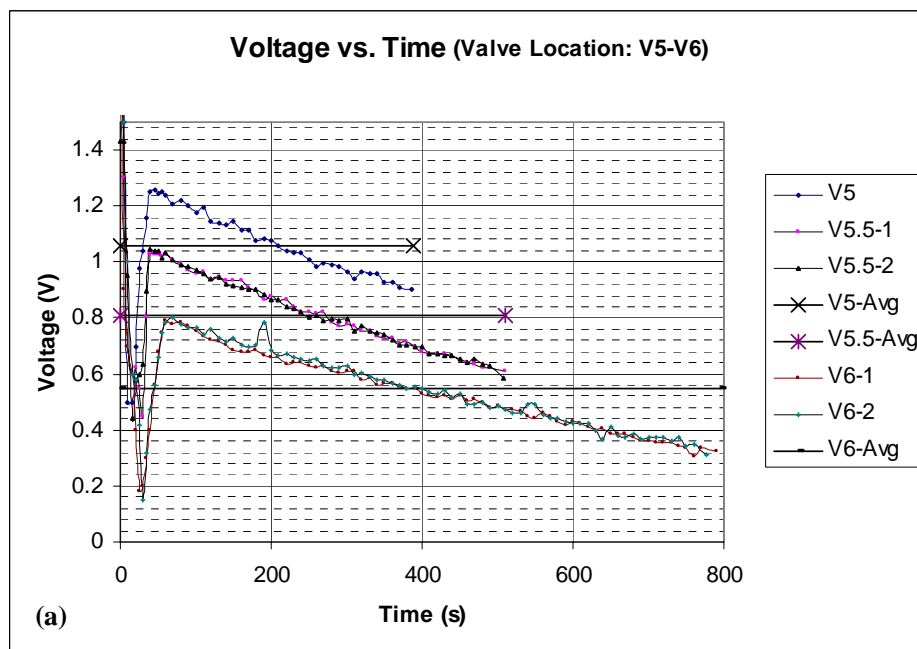


Figure 4.11 Calibration of the flow meter--voltage versus time curve

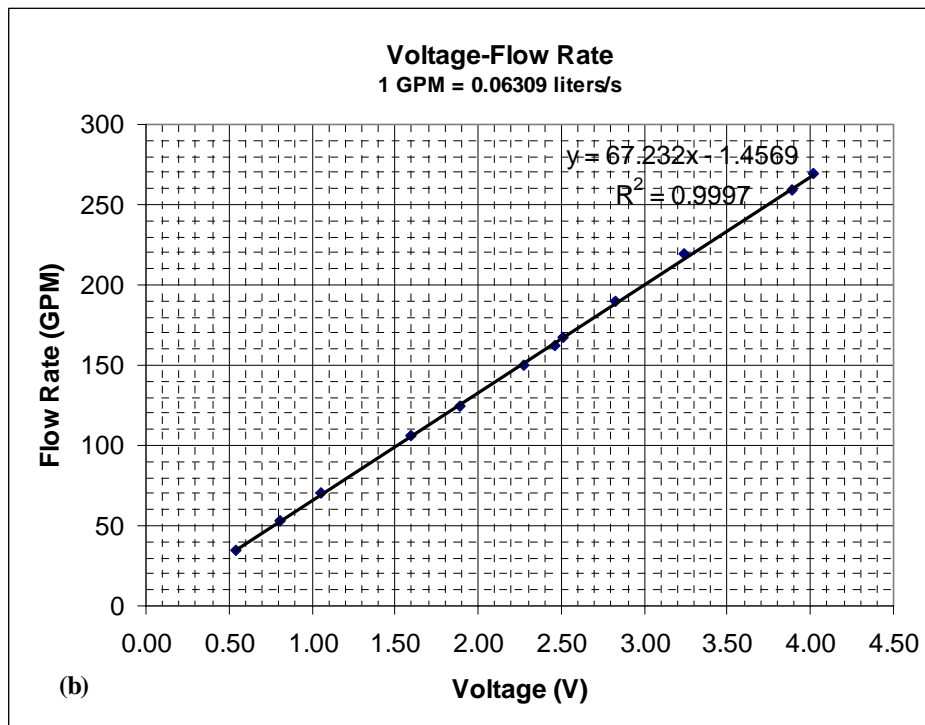


Figure 4.12 Calibration of the flow meter--calibration curve

The wooden channel on the false bottom had a fixed cross section. The velocity can be measured by using an ADV or a paper boat. With the area of cross section and flow velocity available, a flow rate was calculated which was used to verify the calibration curve. Table 4.2 shows a maximum difference of only 5.0% for this comparison.

Table 4.2 Verification of the calibration of flow meter

Test 10 1GPM= 0.06309 liter/sec 1 liter/s= 15.850 GPM
 Flow Rate by Flow Meter = **51.6** GPM = 3.255 Liter/Sec

Station	-2	-1	0	Average	Q=V*A (liter/Sec)	Q=V*A (GPM)	Difference (%)
t2 Water Depth on False Bottom(cm)	9.0	8.8	8.1	8.6			
t2 Section Area (m ²)	0.0189	0.0181	0.0157	0.0175			
t2 Velocity by Calculation (cm/s)	17.3	18.0	20.7	18.6			
t2 Velocity by Paperboat (cm/s)	19.5				3.4	54.2	5.0
t3 Water Depth on False Bottom(cm)	8.9	8.5	8.2	8.5			
t3 Section Area (m ²)	0.0185	0.0171	0.0160	0.0172			
t3 Velocity by Calculation (cm/s)	17.6	19.1	20.3	18.9			
t3 Velocity by Paperboat (cm/s)	19.1				3.3	52.0	0.8
t4 Water Depth on False Bottom(cm)	8.8	8.5	8.3	8.5			
t4 Section Area (m ²)	0.01813	0.01707	0.016381	0.0172			
t4 Velocity by Calculation (cm/s)	18.0	19.1	19.9	18.9			
t4 Velocity by Paperboat (cm/s)	18.6				3.2	50.7	-1.8

Note:

1. Station -2, -1, 0 are located on upstream wooden channel; -2 means 2 meters in front of the soil tank;
2. Section Area: Cross sectional area of the flow;
3. Velocity by Calculation: The flow rate measured by the flow meter divided by Section Area;
4. Q=V*A: V=Velocity by Paperboat; A=Average of the Section Area of station -2, -1, 0.

4.4 TEST PROCEDURE

After several flume tests have been run, a test procedure was developed for easy follow-up.

The procedure for the preparations is as follows:

1. Refill sand, saturate the sand and level it;
2. Submerge sand in water and measure with point gauge the elevation of the two tracks relative to the static water surface;
3. Dewater the sand;
4. Build the channel according to predetermined geometry;
5. Use point gauge to measure the elevation of riverbed; If the riverbed is not level, reshape it;
6. Prime the pump with water.

What follows is the procedure for initial measurement before the test:

1. Use a tape measure to measure initial bank edges;
2. Fill the channel with water and use point gauge to measure the elevation of the static water surface;

3. Take pictures of the channel;
4. Open the valve and start the test;
5. Adjust weir height.

The procedure for regular measurements when the test is running is as follows:

1. Take pictures;
2. Use a tape measure to measure the location of soil-water interface on the bank;
3. Use point gauge to measure the slope of water surface;
4. Use an ADV and/or a paper boat to measure velocity in soil tank;
5. Use a paper boat to measure velocity in upstream and downstream wooden channels;
6. Use a tape measure or a better tool to measure water depth;
7. Use a tape measure or a better tool to measure the profile of cross sections every one meter. For each measurement, velocity at that station can be obtained.

The procedure for data reduction is as follows:

1. Process images when it is desired. Input data of shifting bank into AutoCAD; Data should be input as soon as a measurement is done. If irregular migration occurs, caution should be taken about possible cause. Makeup measurement needs to be done if the situation requires.
2. Input data of velocity, water surface elevation, and water depth etc. into Excel;
3. Draw profile of water surface and channel bed elevation. Draw charts of velocity distribution along the channel.
4. Mark migration direction and quantity in AutoCAD; calculate migration rate \dot{M} for each time interval;
5. Calculate radius of curvature R for different bank edges using a MATLAB program and a AutoCAD VBA program;
6. Measure water surface width W in AutoCAD and calculate R/W ;
7. Calculate Froude numbers F_r ;
8. Draw relation curves among \dot{M} , v , R/W , y , F_r ;

4.5 SAMPLE TEST OUTPUT

The outcome of first a few tests was welcomed with eagerness and surprise. It hadn't been expected that it would be so hard to maintain constant velocity throughout the channel. In fact, the velocity varied from upstream to downstream and from beginning to end. What the tester did was to control the amount of change so that the average velocity was representative of the whole flow condition.

The flume tests run by the author were only part of the complete plan. The findings showed some characteristics of the test and provided information for future improvement. Better results have been obtained from tests conducted in the new Coastal Engineering Lab and will be described in detail in Namgyu Park and Po-Hung Yeh's theses.

Figure 4.13 shows the geometry change occurred in the 7th flume test. The test was stopped when the wall was reached. It can be seen that migration distance increased when it got closer to the exit. One explanation was that secondary flow became larger and larger along the channel. At the end of soil tank, the channel sharply widened. At the interface of soil channel and wooden channel, two materials with quite different strength were put together. A phenomenon similar to "stress concentration" occurred here. Erosion started first at the interface and worsened with time going. This part was strengthened later to reduce erosion. Graphs of all of the author's flume tests in the old Hydro Lab are shown in Appendix A. Pictures of these tests are shown in Appeneix B.

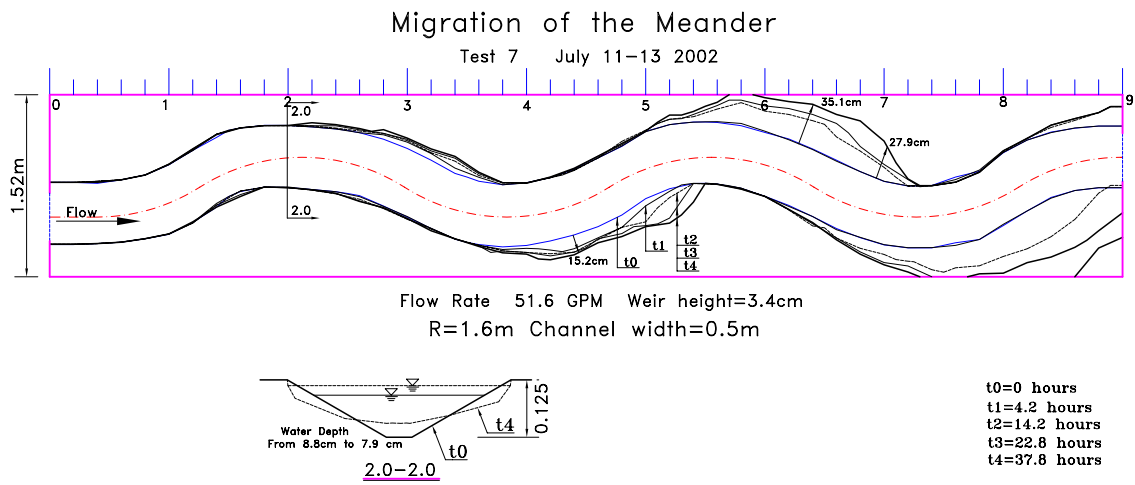


Figure 4.13 Result of flume test 7

CHAPTER V
THE APPLICATION OF SRICOS-EFA METHOD IN THE PREDICTION OF
MEANDER MIGRATION

The process of meander migration was far more complicated than pier scour so that the researchers didn't consider the application of SRICOS-EFA method until consistent decreasing migration rate was observed in flume tests. Some data from the field also showed that there existed a maximum migration. It is worthwhile to explore the possibilities of applying the SRICOS-EFA method in the prediction of meander migration. The verification and parametric study done later on proved the application to be successful.

5.1 THE EXISTENCE OF MAXIMUM MIGRATION M_{\max}

It has been observed that in most flume tests the erosion rate becomes smaller and smaller and comes close to zero in the end, as shown in Figure 5.1. The migration M versus time t curve looks like a hyperbola. Two parameters can determine a hyperbolic curve. Curve fitting was performed to determine these two parameters for test data. The format of the hyperbolic equation was transformed so that a hyperbolic fitting was converted into a linear fitting, shown as follows:

$$M = \frac{t}{\frac{1}{\dot{M}_i} + \frac{t}{M_{\max}}}$$

$$M = \frac{t}{a + bt}$$

$$\frac{t}{M} = a + bt$$

$$\dot{M}_i = \frac{1}{a}, M_{\max} = \frac{1}{b}$$

Parameters a and b were obtained by doing least square fitting on a series of data points (M, t) from flume tests. The curve fitting process is shown in Table 5.1 and Figure 5.1. "Difference" on the table is relative difference between the original and fitted

migration expressed as $(M_{\text{original}} - M_{\text{fit}}) / M_{\text{original}}$. It can be seen from this table the original data and fitted curves are very close.

Table 5.1 Fitting hyperbolic curves for flume test data

Test 7: 51.6 GPM R/W=4					
T7-1R	a	b	$\dot{M}_i = \text{---}$	1.3	(cm/hour)
	0.7632	0.0595			
t (No.)	Time (hours)	Migration (cm)	t/M (hour/cm)	Hyperbola (cm)	Difference (%)
t0	0	0		0	
t1	4.2	4.13	1.0	4.1	0.4%
t2	14.2	8.69	1.6	8.8	1.6%
t3	22.8	11.02	2.1	10.8	-2.4%
t4	37.8	12.47	3.0	12.5	0.6%
			M_{max} (cm)=	16.8	
Test 8: 51.6GPM R/W=8					
T8-1R	a	b	$\dot{M}_i = \text{---}$	1.8	(cm/hour)
	0.5528	0.0573			
t (No.)	Time (hours)	Migration (cm)	t/M (hour/cm)	Hyperbola (cm)	Difference (%)
t0	0	0		0	
t1	5.6	8.21	0.7	6.4	-21.9%
t2	15.2	10.14	1.5	10.7	5.3%
t3	24.6	11.22	2.2	12.5	11.7%
t4	50.9	15.15	3.4	14.7	-3.2%
			M_{max} (cm)=	17.5	
Test 9: 26GPM R/W=4					
T9-1R	a	b	$\dot{M}_i = \text{---}$	0.3	(cm/hour)
	3.1396	0.0854			
t (No.)	Time (hours)	Migration (cm)	t/M (hour/cm)	Hyperbola (cm)	Difference (%)
t0	0	0		0	
t1	10.3	3.23	3.2	2.6	-20.7%
t2	20	3.85	5.2	4.1	7.2%
t3	32.8	4.86	6.7	5.5	13.6%
t4	43	6.17	7.0	6.3	2.3%
t5	66	7.95	8.3	7.5	-5.4%
			M_{max} (cm)=	11.7	

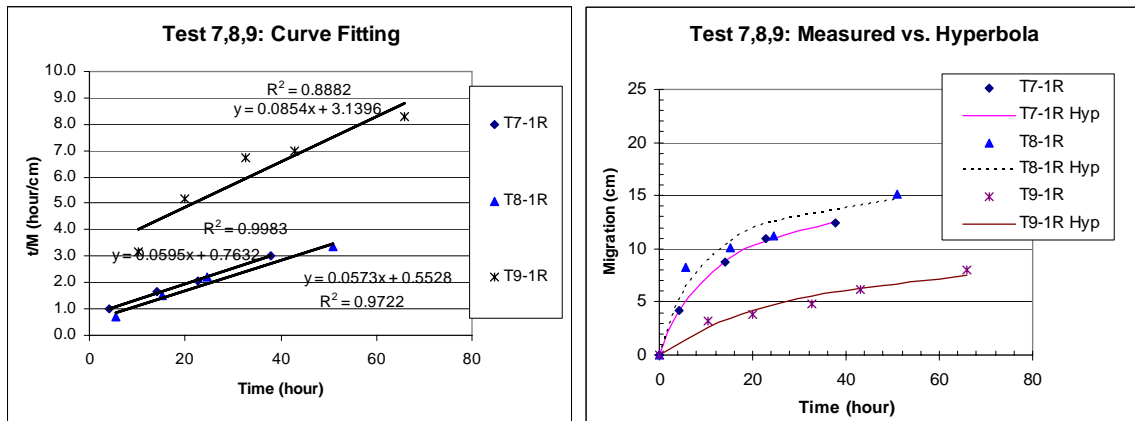


Figure 5.1 Fitting hyperbolic curves for flume test data

Data from the field also shows similar trend. Table 5.2 and Figure 5.2 show the original data and curve fitting process for Des Moines River.

Table 5.2 Fitting hyperbolic curve for Des Moines River

	$\frac{a}{0.0603}$	$\frac{b}{0.0009}$	$\dot{M}_i = 16.6$	(m/year)	
t	Time	Migration	t/M	Hyperbola	Difference
(No.)	(Year)	(m)	(year/m)	(m)	(%)
t0	0	0		0	
t1	3	57	0.05	47.6	-16.5%
t2	19	243	0.08	245.5	1.0%
t3	30	314	0.10	343.6	9.4%
t4	43	421	0.10	434.3	
t5	58	514	0.11	515.6	0.3%
t6	70	558	0.13	567.7	
t7	80	639	0.13	604.7	
t8	87	660	0.13	627.7	-4.9%
t0=1880, t8=1967			M_{max} (m)=	1111.1	

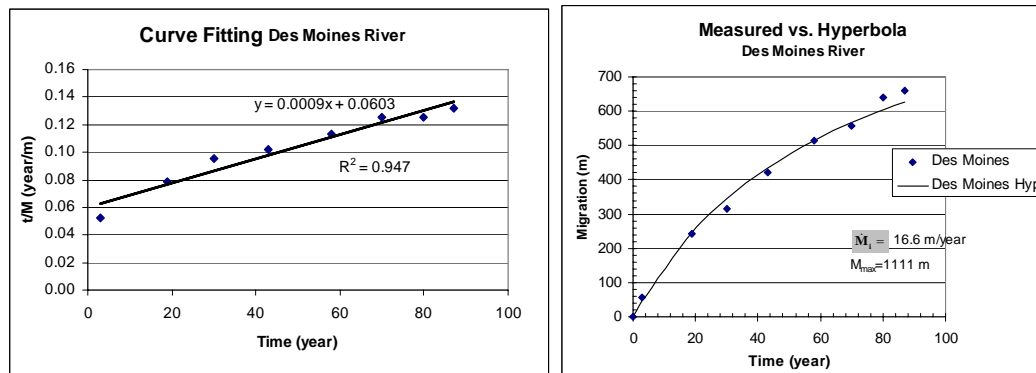


Figure 5.2 Fitting hyperbolic curve for Des Moines River

5.2 DEVELOPING EQUATIONS FOR M_{\max}

In the Pier Scour project, the equation for maximum scour depth z_{\max} was developed from flume test data. A bunch of parameters were involved with the development of z_{\max} . But it was found that the most important one is Reynolds number. Reynolds number is a function of velocity and pier diameter. The same type of work is also being done for the prediction of meander migration but with much more complexity. Po-Hung Yeh is developing M_{\max} equations for sand. Namgyu Park will develop M_{\max} equations for clay. The final equations can be found in their Ph.D. dissertations respectively which are supposed to come out in 2006.

A set of preliminary M_{\max} equations for sand has been developed by Po-Hung Yeh based on regression of flume test data. Further improvement is still underway. The general form can be expressed as:

$$\frac{M_{\max}}{W} = f\left(\frac{R}{W}, \phi, \theta, Fr\right)$$

Where,

- M_{\max} : Maximum migration distance;
- W : Average channel width;
- R : Radius of curvature of the channel;
- ϕ : Bend angle;

θ : Location angle starting from inflection point to the point of interest;

$$Fr: = \frac{v}{\sqrt{gy}}, \text{ Froude number;}$$

v : Velocity of flow;

g : Acceleration of gravity;

y : Water depth.

5.3 DEVELOPING EQUATIONS FOR τ_{\max}

Before explaining maximum shear stress τ_{\max} , it is important to explain critical shear stress τ_c . Critical shear stress is a stress level at which soil starts to erode. Below this shear stress soil particles are not moved by water, above this shear stress soil particles are moved away and a certain erosion rate is established. The velocity that produces critical shear stress is called critical velocity. A method to test critical velocity is to observe soil erosion condition under different velocity in a flume. The velocity that causes soil start to erode is critical velocity. This method is good for sand. For clay, the erosion is hardly noticeable since the particle size is so tiny. In EFA test, critical shear stress is defined as corresponding to a standardized erosion rate of 1 mm/hr. If shear stress is smaller than critical shear stress, it is considered no erosion occurs.

In Pier Scour project, maximum shear stress τ_{\max} is the maximum shear stress that occurs on soil-water interface when erosion starts (Chen, H.-C., 2002). With erosion progressing, scour depth increases gradually and shear stress decreases accordingly until it reaches critical shear stress τ_c . Since it is not convenient to measure shear stress on the interface, numerical simulation is needed to develop the τ_{\max} equation. Numerical simulation can be considered as a computer version of “flume test”. A variety of parameters can be chosen and varied at small steps. Many more cases can be conducted than in actual flume tests. For pier scour, data reduction showed that the most influential factors that affect τ_{\max} are velocity and Reynolds number.

The same concept and procedure can be applied to the prediction of meander migration. The maximum shear stress corresponds to a certain flow condition, soil property, and channel geometry. Numerical simulation calculates the maximum shear stress along the channel under different flow and geometry conditions. The influence of these factors on maximum shear stress is quantitatively analyzed and an empirical formula is thus developed.

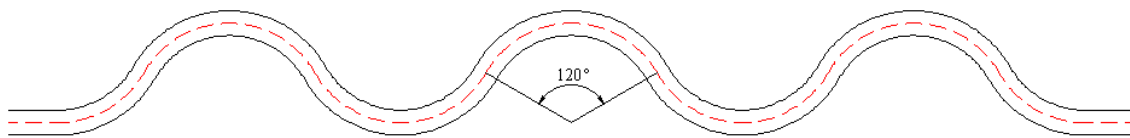


Figure 5.3 $R/W=4$ $\phi=120^\circ$

Figure 5.3 shows the geometry of a typical model channel in which $R/W=4$, $\phi=120^\circ$. There are seven bends between two straight segments. The bend angle for the first one and the last one is only 60° . By varying these parameters, a number of cases have been simulated numerically.

Figure 5.4 is the result of a simulation case where $R/W=4$ and $\phi=180^\circ$. The shear stress near the inlet is very high since the inflow is assumed to be uniform without boundary layer. The shear stress on the first and last bend is quite different from that on the bends in the middle. This is because they are close to the inlet or the exit and the bends in the middle are least affected by boundary conditions. The left bank and the right bank have the same shear stress distribution except that there is a phase difference of ϕ angle. Shear stress distribution on a bend occurs almost periodically. τ_{\max} on the graph is the maximum shear stress on the cross section of a bank. τ_{avg} is depth average shear stress. Both τ_{\max} and τ_{avg} take similar shape. The line segments corresponding to $\theta/\phi=[0 \ 1]$ indicate the location of occurrence of a shear stress on a bend. θ denotes the relative location of a point on the bend. It is the angle between the inflection point and

the point of interest. When $\theta=0$, it indicates the inflection point. When $\theta=\phi/2$, it indicates the middle point of the bend. It is observed from Figure 5.4 and other graphs, the peaks of τ_{\max} and τ_{avg} curves often occur at a location close to $\theta/\phi=1$. A peak means the maximum τ_{\max} of a bend, also called τ_{\max_max} .

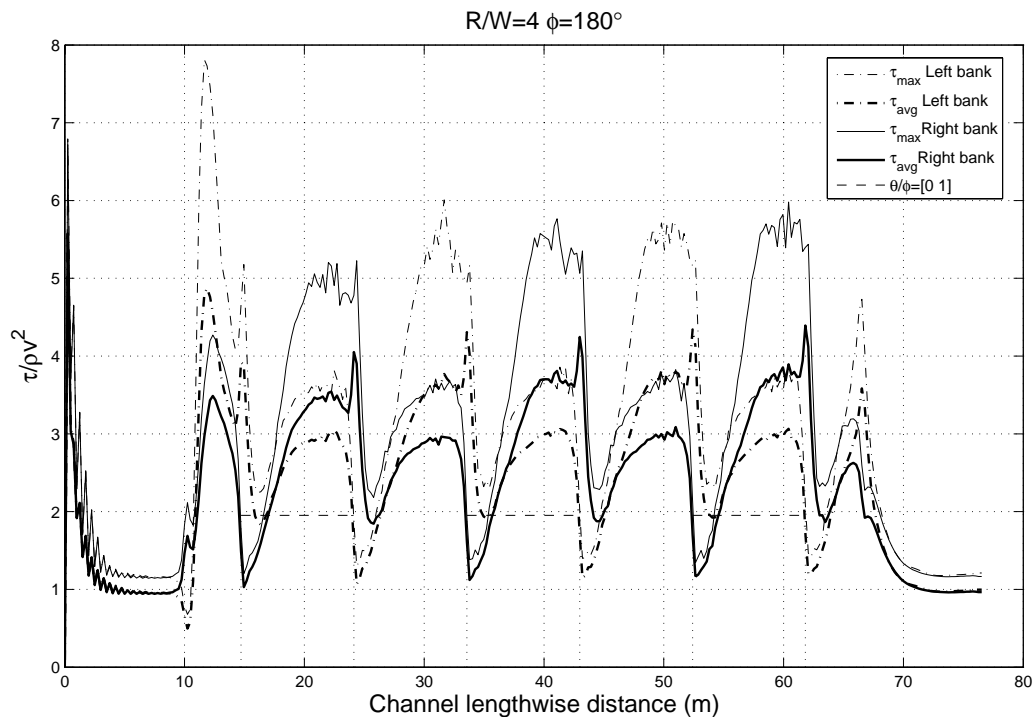


Figure 5.4 Simulated shear stress along a channel

The multiple regression involves variables v , R/W , ϕ , and θ . From dimensional analysis and past experience, it can be reasonably assumed that the non-dimensionalized shear stress $\tau/(\rho v^2)$ is independent of velocity. So the term $\tau/(\rho v^2)$ is a function of R/W , ϕ and θ . When the influence of one parameter is being studied, others are kept constant. The whole regression process can be divided into these steps:

1. Find the relationship between τ_{\max_max} and R/W , $\tau_{\max_max}/(\rho v^2)=f_1(R/W)$;
2. Find the relationship between τ_{\max_max} and ϕ , $\tau_{\max_max}/(\rho v^2)=f_2(\phi)$;

3. Find the relationship between the relative location of occurrence of τ_{\max_max} and R/W , $\theta_{\max}/\phi=f_3(R/W)$;
4. Find the relationship between the relative location of occurrence of τ_{\max_max} and ϕ , $\theta_{\max}/\phi=f_4(\phi)$;
5. Fit the shear stress distribution of a bend, $\tau_{\max}/(\rho v^2)=f_5(\theta/\phi, \theta_{\max}/\phi)$;
6. Get the final equation: $\tau_{\max}/(\rho v^2)=f_1(R/W) \times f_2(\phi) \times f_5(\theta/\phi, f_3 \times f_4)$.

Figure 5.5 shows the relationship between τ_{\max_max} and R/W at a constant ϕ angle of 180° and a velocity of 0.2 m/s. A curve fitting was done for the maximum shear stress. When $R/W \geq 2$, the fitted and the original curves are very close. When $R/W < 2$, the fitted curve is far off the original data. As a matter of fact, geometry of $R/W < 2$ doesn't often happen in nature. The expression of the fitted curve is:

$$\frac{\tau_{\max_max}}{\rho v^2} = f_1\left(\frac{R}{W}\right) = \frac{1}{400\left(\frac{R}{W}\right)}$$

Figure 5.6 shows the relationship between τ_{\max_max} and ϕ angle at a constant radius to width ratio of 4 and a velocity of 0.2 m/s. A parabolic curve can be fitted to the data. Since the difference between the maximum and minimum τ_{\max_max} values is about 25%, a straight line was used instead. So τ_{\max_max} was treated as independent of ϕ . The expression of function f_2 is: $f_2(\phi)=1$.

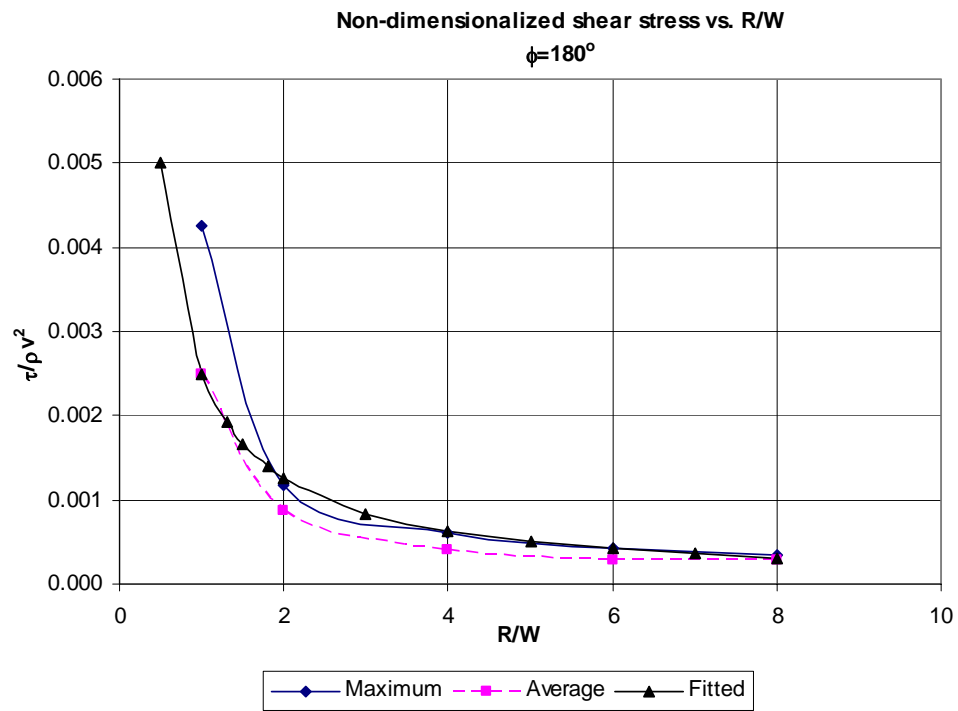


Figure 5.5 Influence of R/W

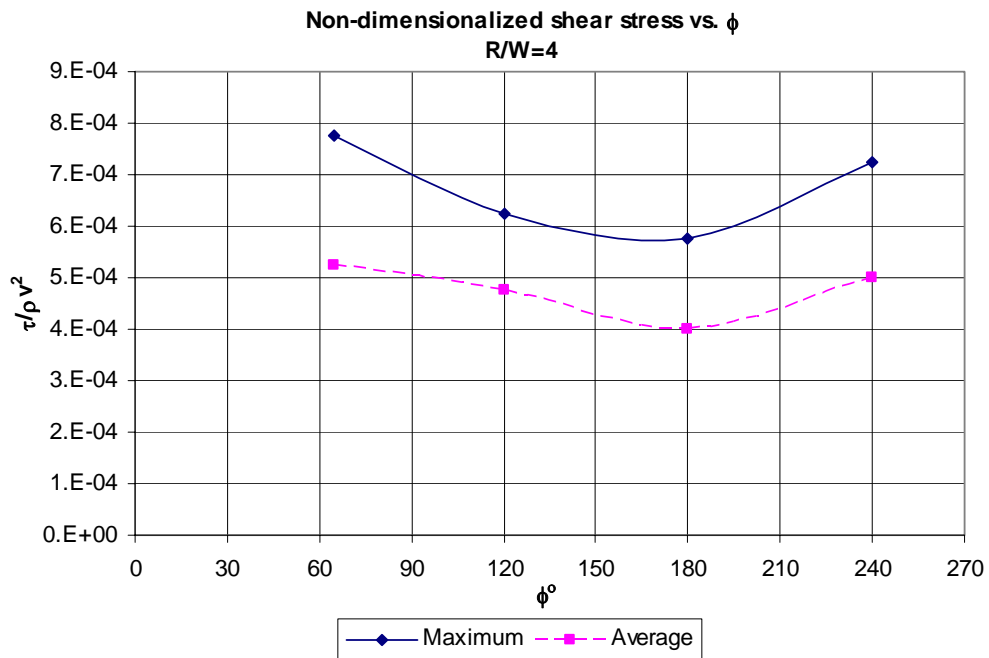


Figure 5.6 Influence of ϕ angle

The shear stress distribution of one bend mostly has similar shapes when the influential factors change. It is possible and desirable to find an analytical curve which fits the distribution well. Based on comparison of different type of analytical curves, the probability density function (PDF) of *extreme value distribution* is found to be a good match for the calculated shear stress. The expression is:

$$y = f(x|\mu, \sigma) = \frac{1}{\sigma} e^{\left(\frac{x-\mu}{\sigma}\right)} e^{-\left(e^{\left(\frac{x-\mu}{\sigma}\right)}\right)}$$

Where, μ is the location parameter and σ is the scale parameter. The indicator of relative location on the bend is $x=\theta/\phi$. When $x=\mu$, y reaches maximum value. σ determines the maximum value and the degree of spreading of the curve. Figure 5.7 displays an example of extreme value distribution with $\mu=1$, and $\sigma=0.37$.

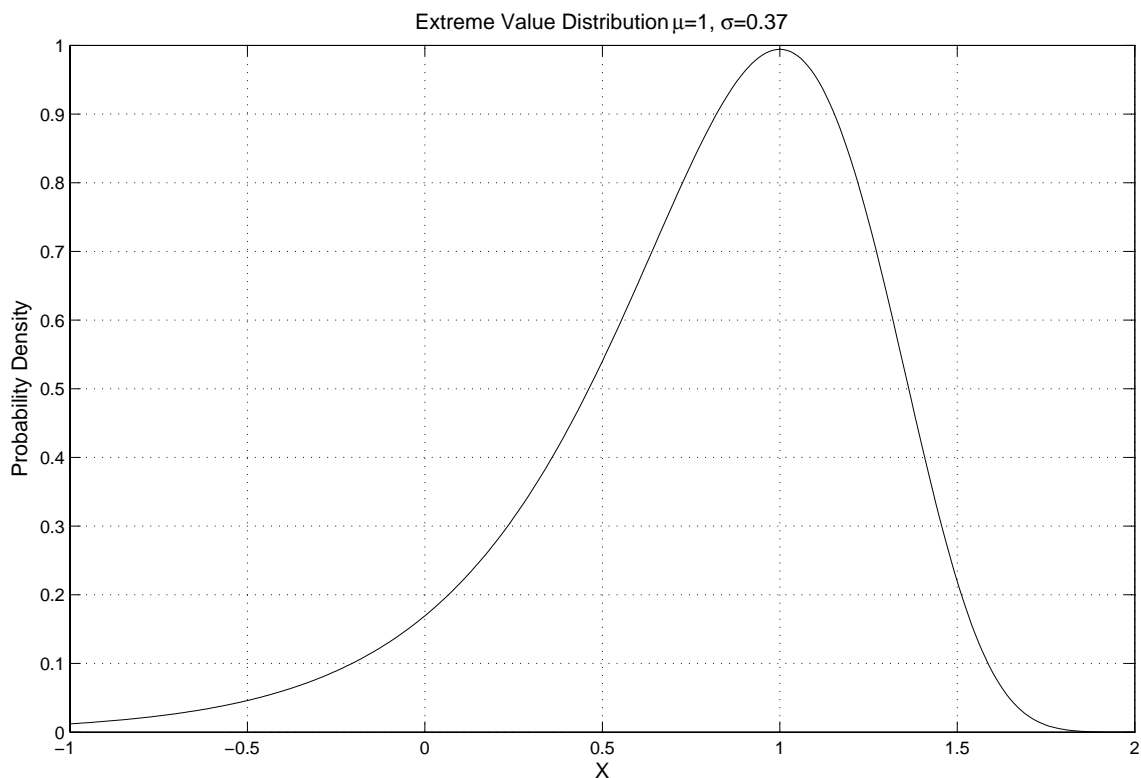


Figure 5.7 Extreme value distribution

Theoretically σ should also be a function of R/W and ϕ angle. When $\sigma=0.37$ the maximum value is close to 1 and the degree of spreading is similar to most cases of shear stress distribution. A constant value of $\sigma=0.37$ is used for the purpose of simplicity while necessary precision is maintained. Figure 5.8 shows the calculated shear stress and fitted curves. A good fitting can be observed. The shear stress distribution equation can be expressed as:

$$f_s\left(\frac{\theta}{\phi}, \frac{\theta_{\max}}{\phi}\right) = \frac{1}{\sigma} e^{\left(\frac{x-\mu}{\sigma}\right)} e^{-\left(e^{\left(\frac{x-\mu}{\sigma}\right)}\right)}$$

$$\sigma = 0.37, \mu = \frac{\theta_{\max}}{\phi}, x = \frac{\theta}{\phi}$$

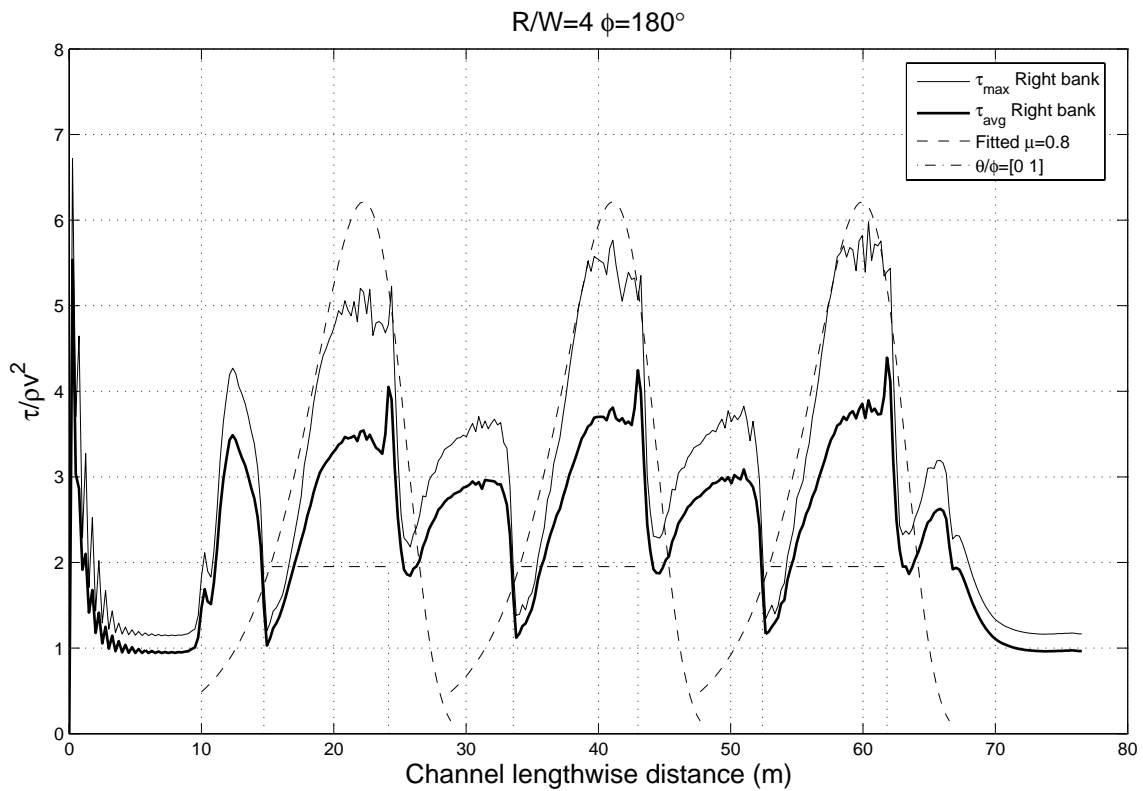


Figure 5.8 Simulated shear stress and fitted curves

Next step is to determine the location parameter μ which is a function of R/W and ϕ angle. The influence of R/W on location parameter μ is shown in Figure 5.9. θ_{\max}/ϕ indicates relative location of the peak values. “tao_max” denotes the original data. For the original data, when there is a plateau or there are several peak values of similar magnitude on the same bend, the location of the peak value of the average shear stress curve is chosen. Curve fitting is a type of simplification. The fitted curve will stand for the actual data in the prediction of meander migration. The relative location of the peak values of fitted curves is of greater interest. The relationship between this relative location and R/W is indicated by “Fitted” in Figure 5.9. The θ_{\max}/ϕ value corresponds to the peak of the *extreme value distribution* in Figure 5.8. A straight line is fitted with an $R^2=0.9156$. The expression for the location parameter is:

$$\mu = \frac{\theta_{\max}}{\phi} = f_3\left(\frac{R}{W}\right) = -0.047\left(\frac{R}{W}\right) + 1.05$$

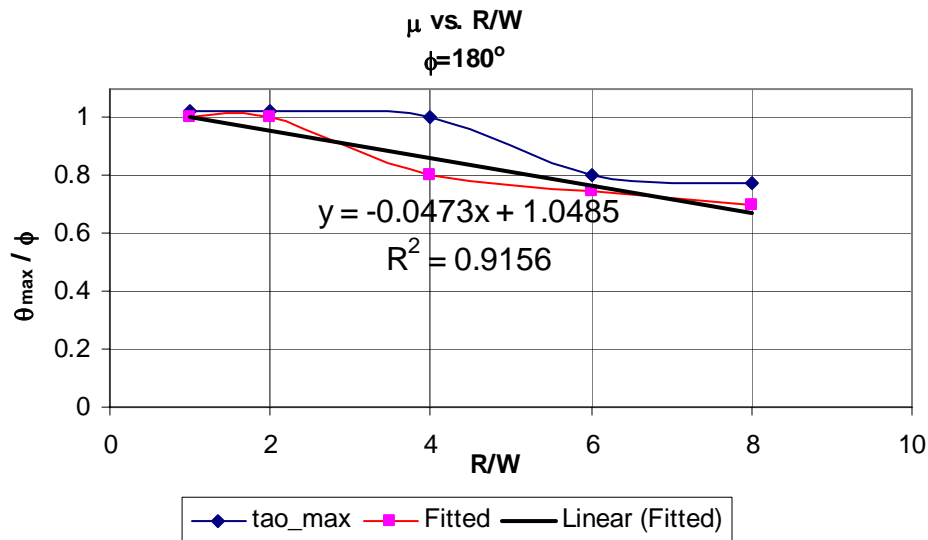


Figure 5.9 Influence of R/W on location parameter

Figure 5.10 shows the relationship between θ_{\max}/ϕ and ϕ angle. The curve for the original data (“tao_max”) is obtained with the same method as shown in Figure 5.9. The relative location for the peaks of the fitted extreme value distribution changes very little when ϕ angle is less than 120° . When ϕ angle is larger than or equal to 120° , it doesn’t change at all. Thus the location parameter is considered as independent of ϕ angle. The expression of function f_4 is $f_4(\phi)=1$.

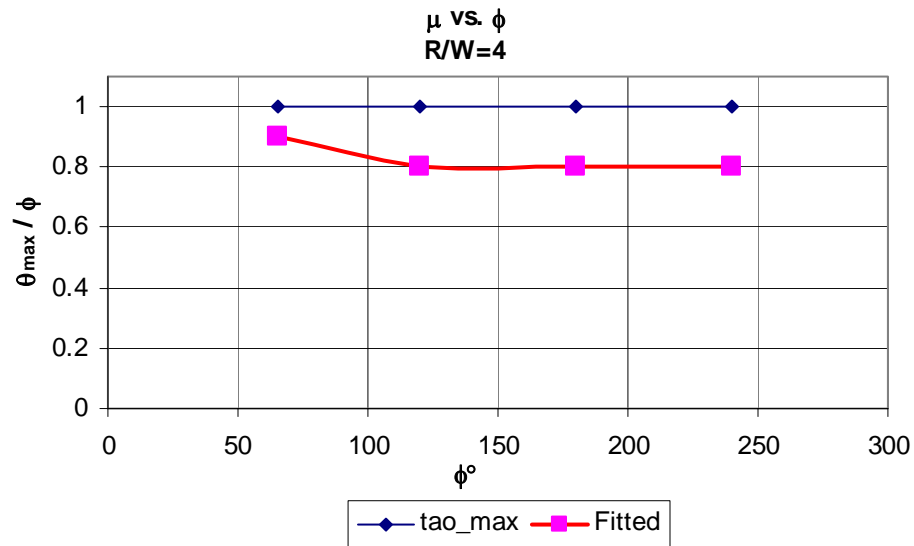


Figure 5.10 Influence of ϕ angle on location parameter

In numerical simulation, the bank is considered as perfectly smooth. To account for the roughness in reality, a constant coefficient of $c_1=8$ is used. It was also found a geometry dependent coefficient c_2 will improve the prediction. Both c_1 and c_2 are determined by verification study. Details are discussed in Chapter IX.

The last step is to combine the individual relationships and develop unified equation for $\tau_{\max}/(\rho v^2)$ as a function of R/W , ϕ and θ . The combined form is:

$$\frac{\tau_{\max}}{\rho v^2} = f_1\left(\frac{R}{W}\right) \times f_2(\phi) \times f_5\left(\frac{\theta}{\phi}, f_3 \times f_4\right)$$

By plugging in corresponding equations and the final equation is expressed as:

$$\frac{\tau_{\max}}{\rho v^2} = \frac{c_1 c_2}{400 \left(\frac{R}{W}\right)} \times \frac{1}{\sigma} e^{\left(\frac{x-\mu}{\sigma}\right)} e^{-\left(e^{\left(\frac{x-\mu}{\sigma}\right)}\right)}$$

$$\sigma = 0.37, \mu = \frac{\theta_{\max}}{\phi} = -0.047 \left(\frac{R}{W}\right) + 1.05, x = \frac{\theta}{\phi}, c_1 = 8$$

$$c_2 = 0.25 \frac{R}{W} - 0.5 \dots \dots \dots \left(\frac{R}{W} > 6\right)$$

$$c_2 = 1 \dots \dots \dots \left(\frac{R}{W} \leq 6\right)$$

The regression is done on a limited number of simulation cases. A systematic matrix will generate maximum shear stress under all possible combination of conditions. *Extreme value distribution* is a good match for the shear stress distributions. In some cases the difference between the original data and the fitted curve is quite big. A better fitting is desired. In general this is the first version of τ_{\max} equation. Further improvement might be done in the near future.

5.4 APPLICATION OF THE SRICOS-EFA METHOD

When the equations for M_{\max} , τ_{\max} are ready, SRICOS-EFA method can be applied with reference to the steps implemented in Pier Scour project. The major difference in the implementation is that for pier scour velocity is the only changing parameter, whereas for meander migration changing parameters also include channel geometry and water depth. If the velocity is constant, there is only one scour depth z versus time t curve for the prediction of pier scour. Scour depth after a certain time can be calculated in one step. For the prediction of meander migration even though velocity is constant, the geometry and possibly water depth change with time. These parameters need to be updated after each time step. A long time constant velocity needs to be divided into a number of time steps. The migration distance of each step is accumulated which is similar to using a hydrograph.

A channel curve is simplified into circles and straight lines for easy analysis. The bends and the parts with reasonably large curvature are fitted with circles. The rest are

treated as line segments. There might be tens of thousands of time steps for one hydrograph. Manual fitting would make a practical solution not possible. A huge amount of efforts have been put into developing automatic fitting techniques and a computer program. The program can't give perfect solutions to all cases but it can give reasonable solutions to many cases.

Water depth changes with discharge or velocity. Even there exists field data for water depth, it may be for only one location. The prediction literally needs the water depth of each time step for each point on the channel. It would be too complicated to obtain all the true field data. Assumptions are made to simplify the problem. The cross section of the channel is assumed to be trapezoidal and remain the same in the migration process. The average velocity is treated as being the same throughout the channel. With these assumptions, the discharge or velocity versus water depth curve can be developed by running HEC-RAS. Then all the needed water depth can be provided numerically.

Aerial photographs are the source of channel geometry. A hardcopy photograph can be converted into a raster image file by a scanner. WinDIG, the free program mentioned before, can digitize the raster images and obtain coordinates of the river banks.

With all these issues being addressed, the implementation of the Hyperbolic Model can be carried out based on the procedure described below, which is also an adapted version of the SRICOS-EFA method:

1. Collect Shelby tube samples near the bends of interest (sampling direction should be perpendicular to the surface of bank slope);
2. Test them in the EFA to obtain the erosion rate \dot{M} (mm/hr) versus hydraulic shear stress τ (N/m²) curve;
3. Prepare hydrograph and coordinates of channel curves;
4. Choose a time step t_i ;
5. Fit circles and calculating R , ϕ , and θ ;
6. Choose a point on the channel P_j ;
7. Calculate M_{\max} and τ_{\max} ;

8. Read the initial erosion rate \dot{M}_i (mm/hr) corresponding to τ_{\max} on the \dot{M} vs. τ curve;
9. Construct the migration distance M versus time t curve using a hyperbolic model;
10. Calculate equivalent time t_e ;
11. Calculate incremental migration distance of this point for this time step;
12. Calculate the accumulated migration distance and go to step 6 for next point;
13. Go to step 4 for next time step when the calculation of the last point is finished;
14. Prepare graphic output after the last time step.

CHAPTER VI

GEOMETRY STUDY

The purpose of geometry study is to automatically simplify a curvy channel into circles and straight lines. The geometry of a river plays an important role on its migration. Circles are chosen to represent the bend shape due to its simplicity and its claimed effectiveness in literature. The fitting process needs to be done automatically because the fitting process will be repeated thousands of times when a hydrograph is applied. The geometry study consists of these steps:

1. Develop a method to fit a circle for a given group of points;
2. Calculate radius of curvature of each point on the curve;
3. Identify bends for which circles should be fitted;
4. Find the best fit circle at a certain bend;
5. Calculate bend angle.

6.1 FIT A CIRCLE FOR A GIVEN GROUP OF POINTS

6.1.1 A traditional method

The equation of a circle is:

$$(x - x_c)^2 + (y - y_c)^2 = R^2 \quad (6.1)$$

To fit a circle for n given points, a traditional least square method would tend to minimize one of the following error terms:

$$E_1 = \sum_{i=1}^n (R_i - R^*)^2 \quad (6.2)$$

$$E_2 = \sum_{i=1}^n |R_i - R^*| \quad (6.3)$$

$$E_3 = \sum_{i=1}^n |R_i^2 - R^{*2}| \quad (6.4)$$

In these terms, x_c , y_c , R^* are unknowns and $R_i = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}$. To achieve the minimum value of an error term, the following equations must be solved:

$$\begin{cases} \frac{\partial E}{\partial x_c} = 0 \\ \frac{\partial E}{\partial y_c} = 0 \\ \frac{\partial E}{\partial R^*} = 0 \end{cases} \quad (6.5)$$

Error term E_1 gives the simplest form among the three, as shown in what follows:

$$\begin{cases} \sum_{i=1}^n (x_i - x_c) \left(\frac{R^*}{\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}} - 1 \right) = 0 \\ \sum_{i=1}^n (y_i - y_c) \left(\frac{R^*}{\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}} - 1 \right) = 0 \\ R^* = \frac{1}{n} \sum_{i=1}^n \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \end{cases} \quad (6.6)$$

Since there are three unknowns and three equations, these equations are theoretically solvable. This method is called a direct solution (Moura and Kitney 1991). The advantage of this method is that it is straightforward. The disadvantage is that it is not easy to obtain a solution and the computation is time consuming.

6.1.2 Apply the optimization toolbox of Matlab

The search of the best circle can also be treated as an optimization problem. The target is to find a circle which gives minimum error. The object function can be any one of Equation from (6.2) to (6.4). Equation (6.2) was proved to be the most efficient. The optimization toolbox of Matlab has a good solution. Function $x = \text{fminunc}(\text{fun}, x_0, \text{options})$ was used to find a minimum of an unconstrained multivariable function “fun” whose variables were valid in the range of $(-\infty, +\infty)$. x_0 here indicates the coordinates of the starting point for searching the center of the circle. No searching range needs to be specified. Function $x = \text{fmincon}(\text{fun}, x_0, A, b, Aeq, Beq, lb, ub)$ was used to find a minimum of a constrained multivariable function “fun”. x_0 has the same meaning as above. The ranges or constraints of the variables (lb =lower boundary, ub =upper boundary) need to be set which also consist a searching range. Very precise solutions have been given by these two functions. They can also fit a very large circle to a group of points in a straight line, as shown in Figure 6.1. Straight line segments are not unusual in river banks. It is not

surprising that a regular program would collapse in trying to fit a circle to a straight line. Matlab can handle this problem very well. The difference between the solutions of these two functions is tiny. The advantage of this approach is Matlab is widely available and no complicated programming is needed. However, it is not fast enough. These functions were used until a much better method was found.

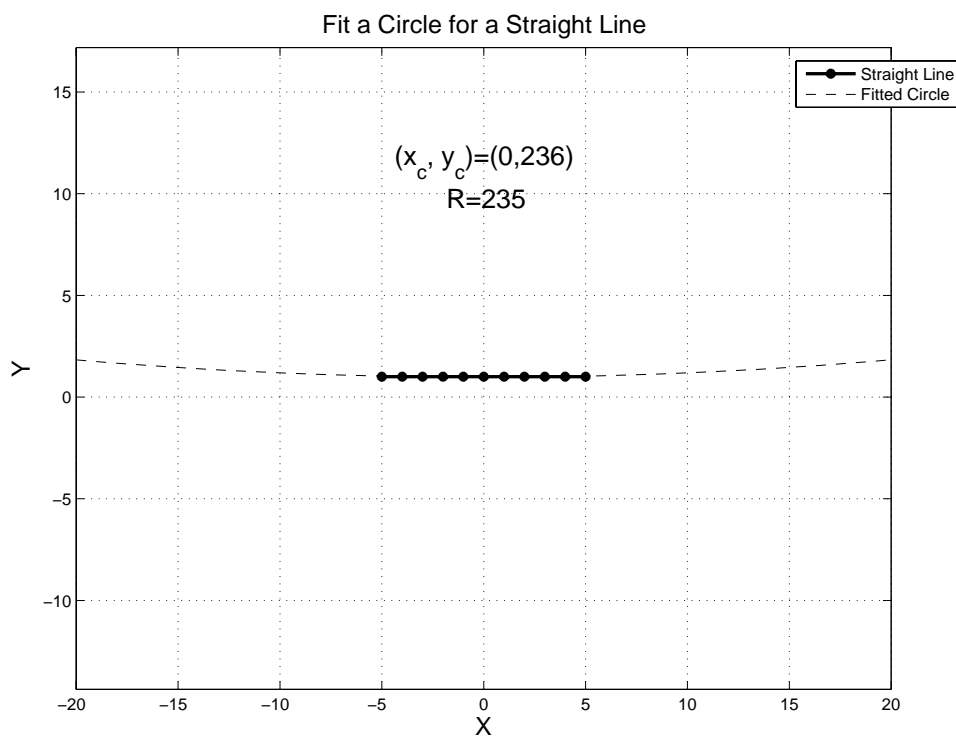


Figure 6.1 Using Matlab to fit a circle for a straight line

6.1.3 Linear least square fitting

Rewrite the equation of a circle Eqn. (6.1) in this form:

$$2x_c x + 2y_c y + R^2 - x_c^2 - y_c^2 = x^2 + y^2 \quad (6.7)$$

Let $a=2x_c$, $b=2y_c$, $c = R^2 - x_c^2 - y_c^2$, $z=x^2+y^2$, the equation can be simplified as:

$$ax+by+c=z \quad (6.8)$$

Plug in the coordinates of the n given points:

$$\begin{cases} ax_1 + by_1 + c \approx z_1 \\ ax_2 + by_2 + c \approx z_2 \\ \dots \\ ax_n + by_n + c \approx z_n \end{cases} \quad (6.9)$$

There are three unknowns and n equations. This is a typical linear least square or linear optimization problem. The equations can be expressed in a matrix form:

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \end{Bmatrix} \approx \begin{Bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{Bmatrix}, \text{ or} \quad (6.10)$$

$$XA \approx Z \quad (6.11)$$

The sign “ \approx ” indicates it is not an exact linear equation set but an optimization problem. A typical way to solve this problem is firstly time the transpose of matrix X to the left side of XA and Z. Then $X^T X$ is a 3 by 3 matrix, and $X^T Z$ is a 3 by 1 vector. Thus $A = \{a, b, c\}^T$ can be solved:

$$A = \begin{Bmatrix} a \\ b \\ c \end{Bmatrix} = \frac{X^T Z}{X^T X} \quad (6.12)$$

Once we solve for a, b, c, we can get $x_c = a/2$, $y_c = b/2$, $R = \sqrt{c + (a^2 + b^2)/4}$. This method is used in the program.

A straightforward way to solve this linear optimization problem is to explicitly express the error terms and to find the minimum sum square.

$$\begin{cases} ax_1 + by_1 + c = z_1 + e_1 \\ ax_2 + by_2 + c = z_2 + e_2 \\ \dots \\ ax_n + by_n + c = z_n + e_n \end{cases}, \text{ or} \quad (6.13)$$

$$\mathbf{e} = XA - Z = [e_1, e_2, \dots, e_n]^T \quad (6.14)$$

T denotes matrix transpose operation. The sum square of the error terms is expressed as:

$$E = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (ax_i + by_i + c - z_i)^2 \quad (6.15)$$

Parameters a, b and c will be obtained by solving equations $\frac{\partial E}{\partial a} = 0, \frac{\partial E}{\partial b} = 0, \frac{\partial E}{\partial c} = 0$. A simpler solution can be achieved by applying Gauss's *principal of least squares* which states that the parameters $A = [a \ b \ c]^T$ that make term (6.15) reach minimum value can also make this term reach minimum value:

$$\mathbf{J} = \frac{1}{2} \mathbf{e}^T \mathbf{e} \quad (6.16)$$

Substituting Equation (6.14) for \mathbf{e} into Equation (6.16) yields:

$$\mathbf{J} = \mathbf{J}(A) = \frac{1}{2} (A^T X^T (XA) - 2Z^T XA + Z^T Z) \quad (6.17)$$

In Equation (6.17), $Z^T XA = (Z^T XA)^T = A^T X^T Z$, because they are scalars. If there exist the required parameters $A = [a \ b \ c]^T$, these requirements need to be satisfied:

1. Necessary condition

$$\nabla_A \mathbf{J} = \begin{bmatrix} \frac{\partial \mathbf{J}}{\partial a} \\ \frac{\partial \mathbf{J}}{\partial b} \\ \frac{\partial \mathbf{J}}{\partial c} \end{bmatrix} = X^T XA - X^T Z = 0 \quad (6.18)$$

2. Sufficient condition

$$\nabla_A^2 \mathbf{J} = \frac{\partial^2 \mathbf{J}}{\partial A \partial A^T} = X^T X \text{ must be positive definite} \quad (6.19)$$

$\nabla_A \mathbf{J}$ is the Jacobian and $\nabla_A^2 \mathbf{J}$ is the Hessian. Any matrix B which satisfies

$$\mathbf{z}^T \mathbf{B} \mathbf{z} \geq 0 \quad (6.20)$$

for all $\mathbf{z} \neq 0$ is called positive semi-definite. Let $\mathbf{h} = X\mathbf{z}$ be a column vector. It can be easily obtained the scalar $\mathbf{h}^2 = \mathbf{h}^T \mathbf{h} = \mathbf{z}^T X^T X \mathbf{z} \geq 0$. So $X^T X$ is always positive semi-definite. It becomes positive definite when its n column vectors are independent which is not satisfied here. This explains why circles can't be fit for some cases, say points are on a straight line.

From the necessary condition, we can obtain the solution

$$A = (X^T X)^{-1} X^T Z \quad (6.21)$$

which is exactly the same as solution (6.12). The above clearly shows the correctness of the simply but efficient method used in the program and also explains why the method fails for some cases. When condition like this occurs, the program was designed to report error and go on the next fitting.

In deriving the equations listed above, the following matrix calculus differentiation rules (Crassidis and Junkins, 2004) were used:

$$\frac{\partial}{\partial \mathbf{x}} (\mathbf{Ax}) = \mathbf{A} \quad (6.22)$$

$$\frac{\partial}{\partial \mathbf{A}} (\mathbf{a}^T \mathbf{Ab}) = \mathbf{ab}^T \quad (6.23)$$

$$\frac{\partial}{\partial \mathbf{A}} (\mathbf{a}^T \mathbf{A}^T \mathbf{b}) = \mathbf{ba}^T \quad (6.24)$$

$$\frac{\partial}{\partial \mathbf{x}} (\mathbf{Ax} + \mathbf{b})^T \mathbf{C} (\mathbf{Dx} + \mathbf{e}) = \mathbf{A}^T \mathbf{C} (\mathbf{Dx} + \mathbf{e}) + \mathbf{D}^T \mathbf{C}^T (\mathbf{Ax} + \mathbf{b}) \quad (6.25)$$

$$\frac{\partial}{\partial \mathbf{x}} (\mathbf{x}^T \mathbf{Cx}) = (\mathbf{C} + \mathbf{C}^T) \mathbf{x} \quad (6.26)$$

6.2 CALCULATE RADIUS OF CURVATURE OF EACH POINT ON A CURVE

6.2.1 Obtaining coordinates of the banks

The channel curves come from digitized aerial photos and maps. Coordinates are obtained by using the free program WinDIG to digitize the scanned photos and maps. These scanned files are the best clue one can get about the geometry of the channel. The user is encouraged to capture detailed geometry feature retained on the electronic image files. The spacing of the digitized points is often uneven and the curves are not smooth. A zigzag can have a big impact on the curvature of the points around it which may cause fluctuation of curvature in that region. This problem is partially solved by evenly distributing the points. Figure 6.2 shows the smoothing effect.

The principal of distribution is to maintain a constant total length of the original line segments between two adjacent new points. First calculate total length (L_0) of the original curve. Then determine the number of segments (N) the original curve will be divided into. Start from the first point and travel along the original curve. Stop at distance

L_0/N and record the coordinates of this point which is the second point of the new curve. Then travel a distance of L_0/N again and the third point will be obtained. Repeat this procedure until the end of the original curve. Since L_0/N is the length along the original curve and may be the summation of several segments, it is not the spacing of the new curve in normal cases.

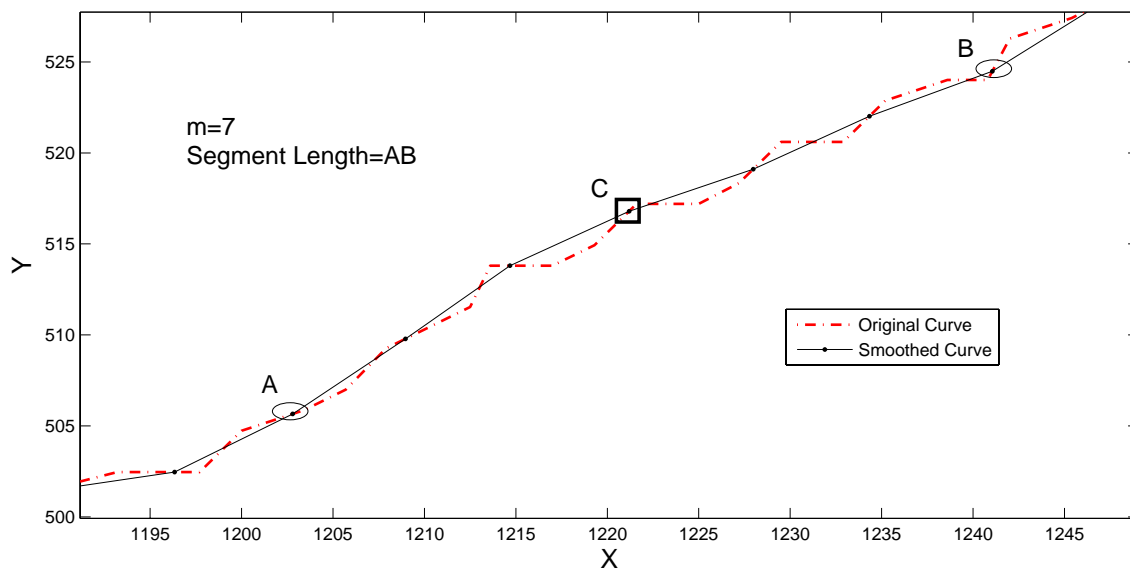


Figure 6.2 Comparison between original curve and smoothed curve

6.2.2 Equations

For continuous curve, the equations for radius of curvature and curvature are:

$$R = \frac{\sqrt{(1+y'^2)^3}}{y''}, c = \frac{1}{R} = \frac{y''}{\sqrt{(1+y'^2)^3}} \quad (6.27)$$

Where,

- R: Radius of curvature
- c: Curvature
- y: Ordinate of the curve

$$y': \frac{dy}{dx}$$

$$y'': \frac{d^2y}{dx^2}$$

In civil engineering, when the deflection of a structural element is very small, y' is close to zero. Thus radius of curvature and curvature can be reasonably approximated as:

$$R = \frac{1}{y''}, c = \frac{1}{R} = y'' \quad (6.28)$$

For a bank curve consisting of discrete points, these simplified equations do not apply because the first derivative y' can't be ignored and sometimes may reach infinity.

6.2.3 Curve fitting and important parameters

The discrete points can be fitted with an analytical curve whose curvature is treated as that of the corresponding points. A long river bank can be fitted with one high order parabolic curve. The fitting error can be small and an analytical expression of curvature is available for the whole curve. Figure 6.3 shows an initial bank of flume test 8 and two fitted parabolic curves of 5th and 7th orders. It is obvious that the higher the order is, the closer the fitting will be. However, if the order is higher than 7, the improvement in closeness is hardly noticeable. This can also be seen in Figure 6.4 which shows how the average fitting error goes down with the increase of fitting order. But the decreasing trend almost stops after order 7.

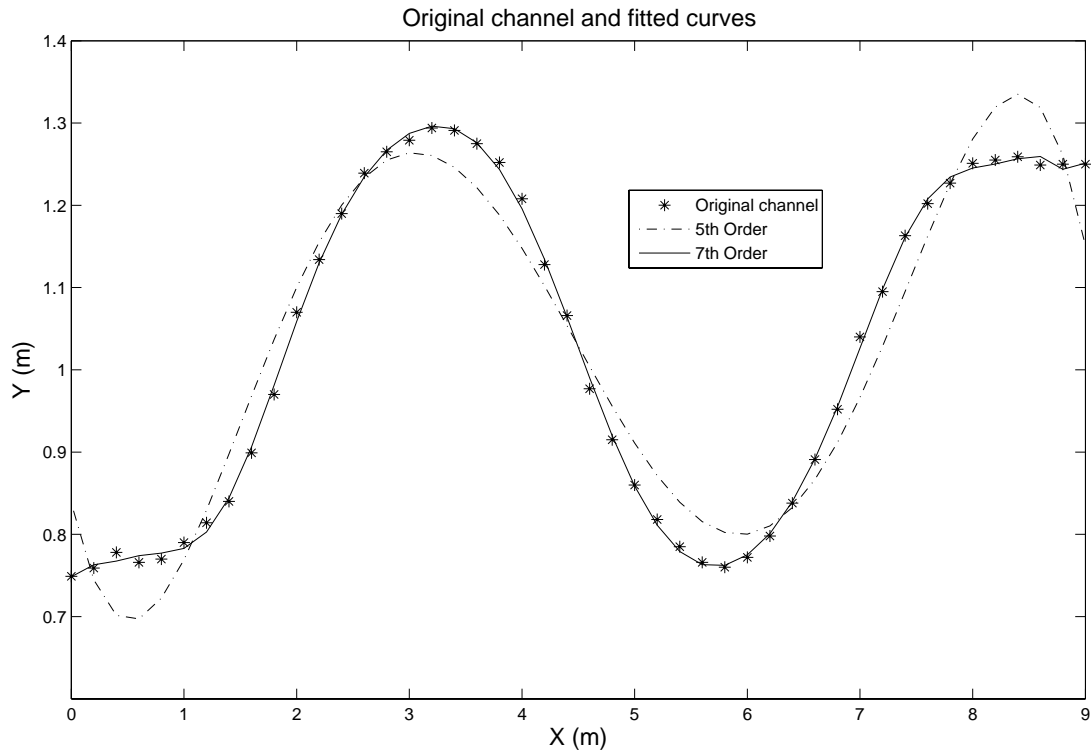


Figure 6.3 Comparison of parabolic fittings of different orders

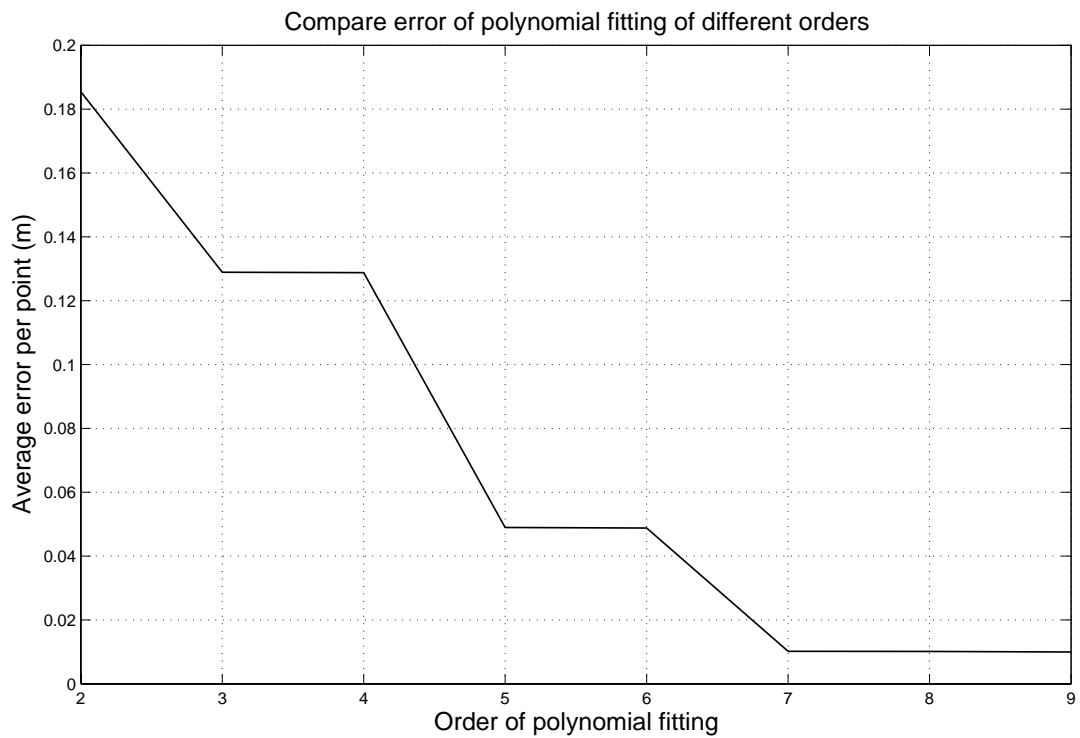


Figure 6.4 Comparison of errors of polynomial fitting of different orders

The idea of using one analytical expression to describe a long channel is tempting. If this goal was achieved, most geometric features of a curve would have been obtained and the task of geometry study can be considered as completed. Figure 6.5 displays a troublesome picture which may turn down this hope. Several orders of parabolic fittings were done to a perfect arc. Although the closeness increases with the order, the fluctuation of radius of curvature increases with the order too. According to the graph, quadratic fitting (2nd order) gives the most stable and closest result. More data points are needed for a high order parabolic fitting than a quadratic fitting. It means quadratic fitting focuses more on local curvature which is more suitable to a point in the middle. When the channel curve is not a single-value function of x coordinate, it has to be broken into several segments in order to be fitted with parabolic curves. Since what is needed is curvature of each point and high order parabolic fitting doesn't provide good result, quadratic fitting is chosen for the calculation.

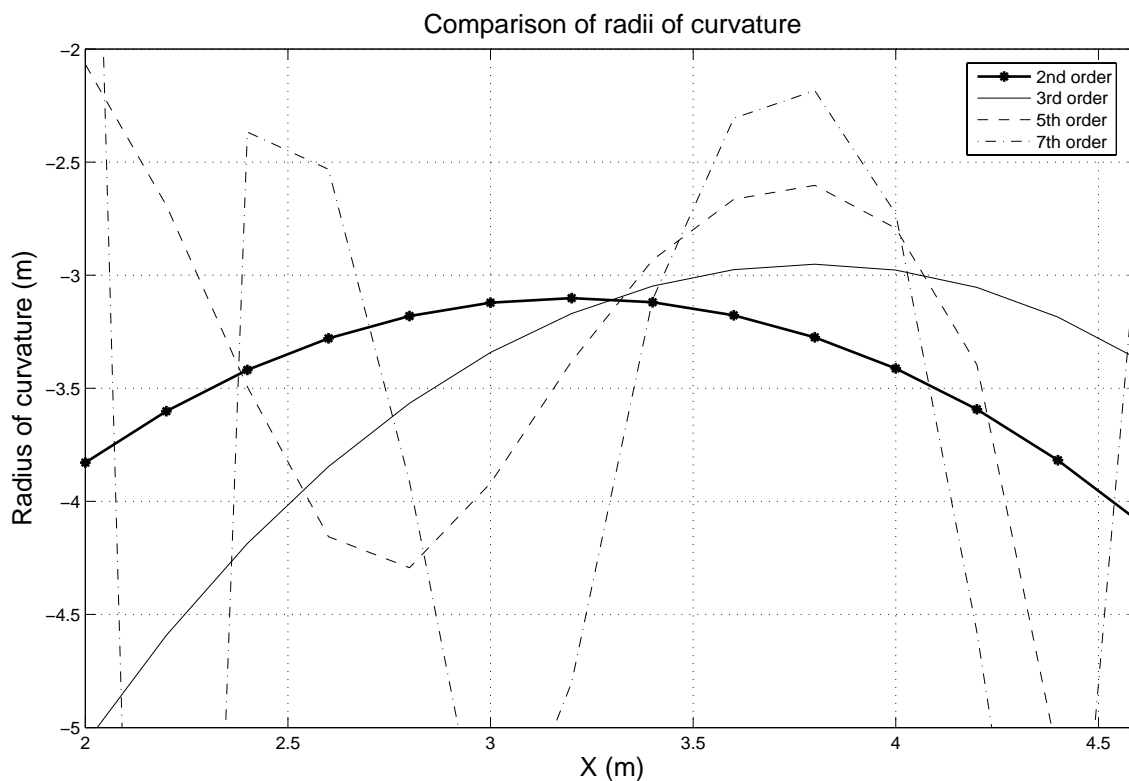


Figure 6.5 Comparison of Rs of different fitting orders

A single point alone doesn't have curvature. Its curvature is determined by a segment that is extended from the point in both directions by a certain length (from point A to point B in Figure 6.2). A quadratic curve is fitted to this chosen segment. The curvature of the corresponding point on the fitted curve is treated as that of the point of interest. This process is repeated for each point on the bank.

Before fitting a quadratic curve, two parameters need to be determined. The first one is the spacing used to distribute the points. The other one is the length of the curve segment for which a quadratic curve is to be fitted, called segment length. The curvature of a river is often related to its average width. These parameters are non-dimensionalized by river width, which can make them independent of a certain case. The spacing coefficient (spacing/width) and segment length coefficient (segment length/width) are therefore used.

With spacing and segment length ready, the number of points "m" contained in that segment is determined accordingly. If m is not an odd number, use m+1 in the program. One segment is defined for each point with $(m-1)/2$ points to the left and $(m-1)/2$ points to the right. For the first and last $(m-1)/2$ points, this fitting cannot be carried out. Normally there is a straight segment at the beginning and in the end. The nearest available radius of curvature can be reasonably assigned to these points. Then each point has a radius of curvature and a curve of the ratio of radius of curvature to channel width (R/W) versus channel lengthwise distance can be drawn.

It can be imagined that large spacing in redistribution can cause the curve to lose local curvature, while very small spacing hasn't been found to cause any problem except that it takes more computing time. In cases where an optimum spacing can not be determined, it would be on the conservative side to pick a small number. In here, 0.1 times of width (0.1W) is suggested if no experience is available.

Segment length directly affects the quality of the calculated curvature. If it is too short, local curvature dominates, which is sometimes drastically different from the curvature of a reasonably larger range. If it is too long, global curvature dominates and some local curvature may be lost. Then the radius of curvature of an inflection point might be only several times larger than that of other points, although theoretically it should be infinity. From the discussion followed we'll see that the major function of R/W

vs. Channel Length curve is to help identify the approximate range of a bend. A good choice of segment length should make the bends stand out on the R/W curve.

The case of Guadalupe River, Texas is used for demonstration. The coordinates of a section of the river is processed by using the procedures described above. The corresponding average river width is 37.7 meters. As it can be seen in Figure 6.6, when segment length is too short ($1.1W$), it is hard to tell where the bends are. Figure 6.7 is the result of using an appropriate segment length $4.0W$. Figure 6.8 shows when segment length is very long ($12.1W$) the R/W vs. Channel Length curve is much smoother and flatter. But the individual bends are still recognizable. Comparing Figure 6.7 with the original bank geometry shown in the last figure of this chapter, it can be observed that each significant bend is represented by an arc-shaped curve segment of different size. Therefore, a relatively large segment length can be chosen when not enough knowledge is available for making a decision. In here, $4.0W$ is suggested.

In these figures, MBL stands for minimum bend length which means only when the identified bend is longer than MBL will it be considered as a bend.

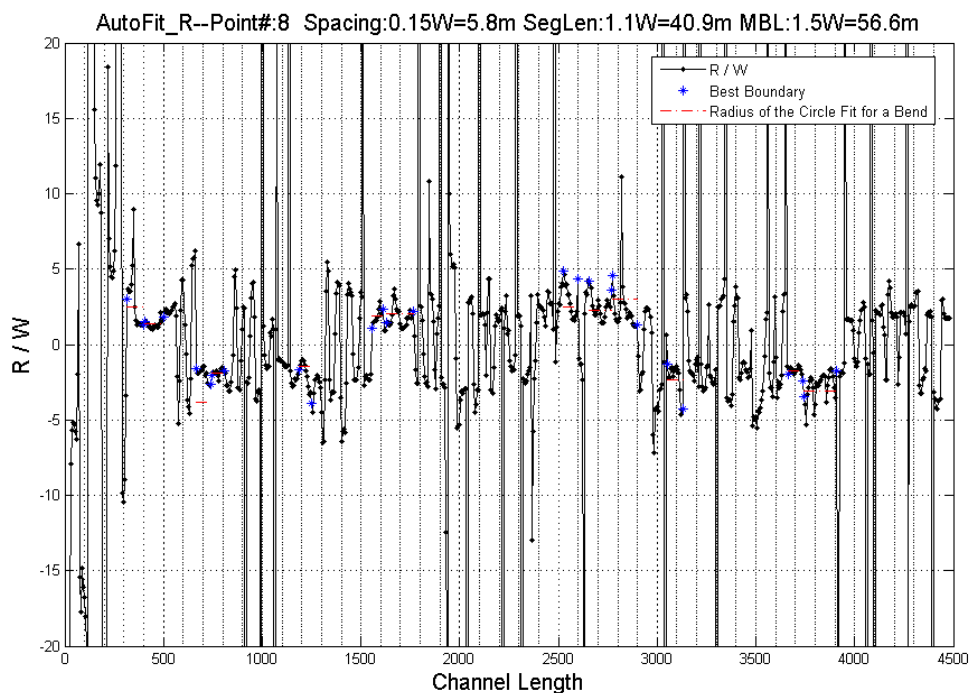


Figure 6.6 R/W vs. Channel length, segment length= $1.1W$

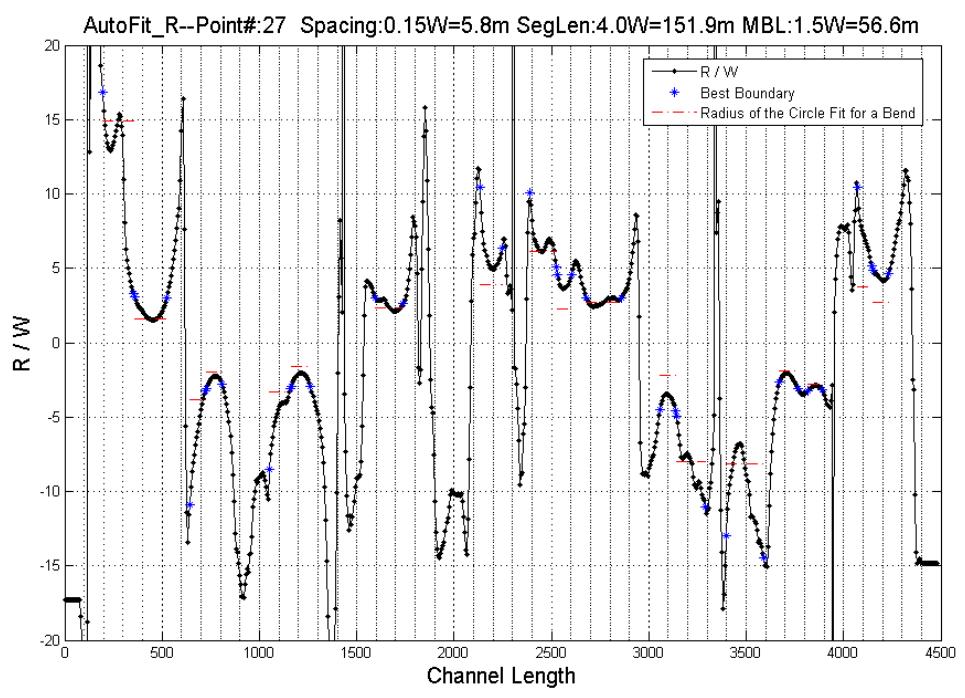


Figure 6.7 R/W vs. Channel length, segment length=4.0W

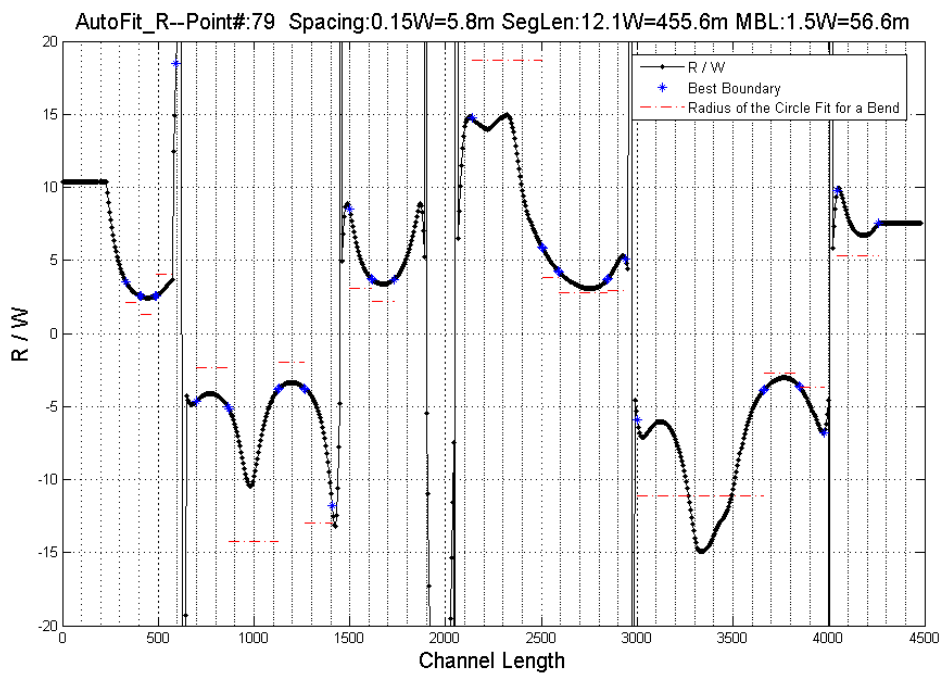


Figure 6.8 R/W vs. Channel length, segment length=12.1W

6.3 IDENTIFY BENDS FOR WHICH CIRCLES WILL BE FITTED

It is so easy for human eyes to tell whether a curve segment is curvy or straight. A computer can't "see" things this way. Mathematical features about the curve need to be discovered and criteria need to be developed for the computer to make a judgment based on its simple logic.

The R/W vs. Channel Length curve shows clear features which are related to the bends on the original bank. It is desired to fit circles for those bends. The choice of the portion of a bend for fitting the circle is critical. Before pinpointing the exact segments for fitting those circles, the approximate ranges of the bell-shaped curves need to be identified first. Three methods are tried.

6.3.1 Manual method with AutoCAD

The user directly specifies the region of a bend in AutoCAD. In the early stage of developing this method, it was thought the bank needed to be fitted with circles only once in the beginning. Later the research team decided to do curve fitting for each time step to take into account the importance of geometry. This method was abandoned because the work needs to be repeated thousands of times for a long hydrograph. A human being is not capable of this.

6.3.2 Criterion line method

This method was recommended by Professor Hamn-Ching Chen. The R vs. length curve for an arc is a horizontal line. When a bend on a bank can be reasonably fitted with a circle, its radius of curvature vs. length curve is smooth and of bell shape. The closer the bend is to an arc, the closer the bell-shaped curve is to a horizontal line. Therefore, if a curve segment is found to be of bell shape facing up or down, it most likely corresponds to an obvious bend on the river bank.

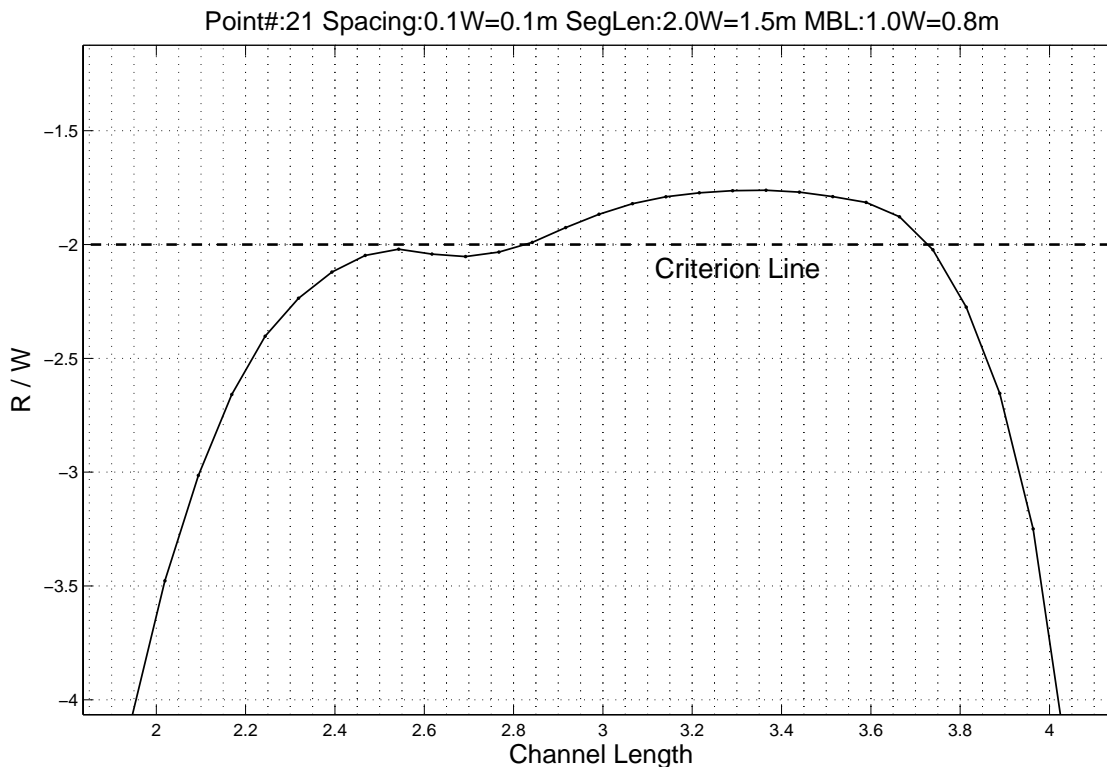


Figure 6.9 A case showing sensitivity of criterion lines

This method uses three criterion lines (six symmetric lines as a matter of fact) to identify the bell-shaped segments on the R/W vs. length curve. The curve segment in Figure 6.9 indicates a bend to be identified. As shown in Figure 6.10, six lines of $R/W = \pm 3, \pm 5, \pm 8$ were drawn on the graph. In the first loop, all continuous segments between $R/W = 0$ and $R/W = \pm 3$ were identified. These segments should be of bell shape and indicate the existence of bends. These segments were marked so that they will not be looked at again in the next loop. In the second loop, the continuous segments left between $R/W = 0$ and $R/W = \pm 5$ were identified. This procedure was repeated for $R/W = \pm 8$. For each loop there might be new bends identified. The outcome is sometimes very sensitive to the choice of these criterion lines. Figure 6.9 shows a tiny change in the criterion line could lead to quite different result. What is shown is the R/W vs. length curve of a well behaved bend. But the criterion line of $R/W = -2$ divides it into two arcs, which would cause a problem. If $R/W = -2.2$, this problem can be avoided. So far no method has been developed to decide on the appropriate numbers. In order to guarantee the inclusion of

the desired boundary, the range should be extended by a certain percentage of the bend length.

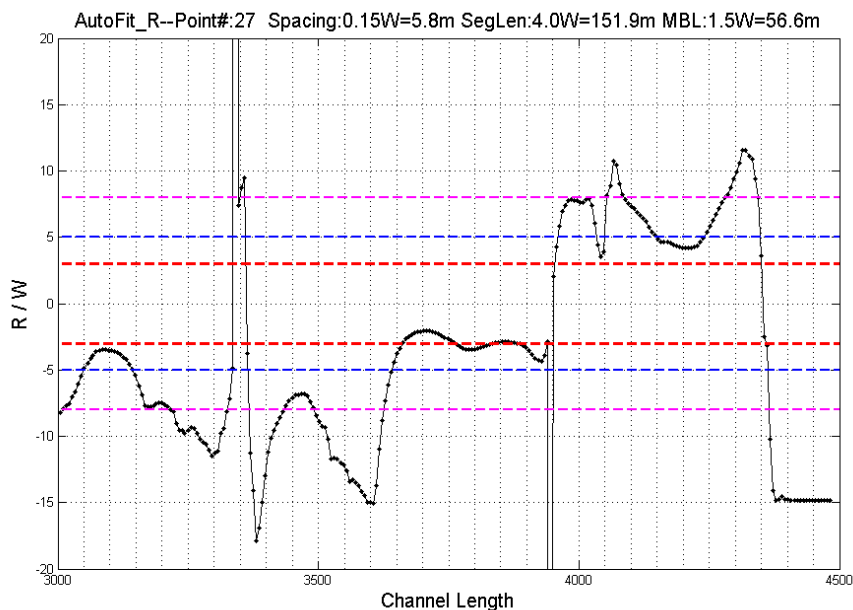


Figure 6.10 Criterion line method

6.3.3 Second derivative method

This method identifies the bends by using the curve d^2R/ds^2 vs. channel length, as shown in Figure 6.11. A continuous segment very close to x axis is considered as a bend. The logic behind this method is that for each bend, the R vs. channel length curve segment is of parabolic shape (bell shape). In parabolic equation $y=ax^2+bx+c$, 'a' value is extremely small for these curves. So d^2R/ds^2 value is close to zero. The range obtained this way also needs to be extended. When two bends of different radius are right next to each other, this method will fail because there is almost no change in d^2R/ds^2 at the intersection point.

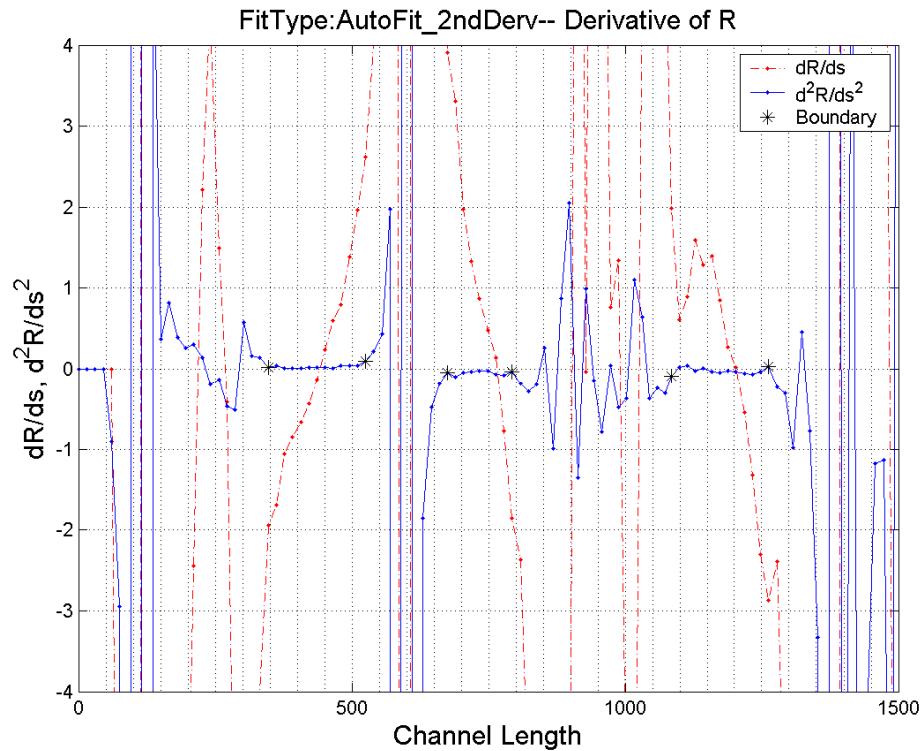


Figure 6.11 The second derivative of R vs. channel length

6.3.4 Change of sign method

The initial channel of flume tests consists of tangent perfect arcs, while the migrated channels can be fitted with tangent arcs. On the R/W versus channel length curve, large values occur once or twice around the inflection point. The change of sign can clearly indicate the separation of bends. This method is applicable only when the bends are tangent to each other.

All methods can produce good results for the purpose of this step. Criterion line method is suggested for its efficiency.

The identified range is a two-point boundary. A circle can be fitted to these points. But there is no guarantee that this circle fits the bend best. Further work is needed to pick the best one.

6.4 AT A CERTAIN BEND FIND THE BEST FIT CIRCLE

6.4.1 Producing a set of circles for making a choice

For a certain bend, dozens of circles can be fitted by using different curve segments. Among them, only one or a few circles really fit the bend. Any of these good circles is called the best fit circle. The arc segment of a best fit circle is supposed to ideally represent the geometry of the bend. To achieve this goal, one should be able to know which circle is the best before asking the computer to make the same decision. Due to the indistinct boundaries of the bend, there is not such an easy formula ready that can tell which circle is the best. To see is to believe. Before a sound criterion is established, human vision provides the only judgment that whether a fitting is good or not. The criteria to be developed will match the visual judgment so that a computer can come up with the same or very close result. It is possible that different people tell different best circles. But normal people give surprisingly close if not the same estimation about the global closeness between a fitted circle and the original bend.

In Figure 6.12, four out of many circles for that bend were picked for comparison. Circle (a) is a very close fitting regarding the segment used. But only a portion of the bend is used for fitting. It doesn't reflect the global geometry feature of the bend. Circle (d) reflects more global feature but the closeness is not good. Circle (b) and (c) maintain a good balance between bend coverage and closeness. The two circles are so close that it is hard to tell which one is better than the other. Since they are identified by using human vision, any one of these two can be called the visually best circle.

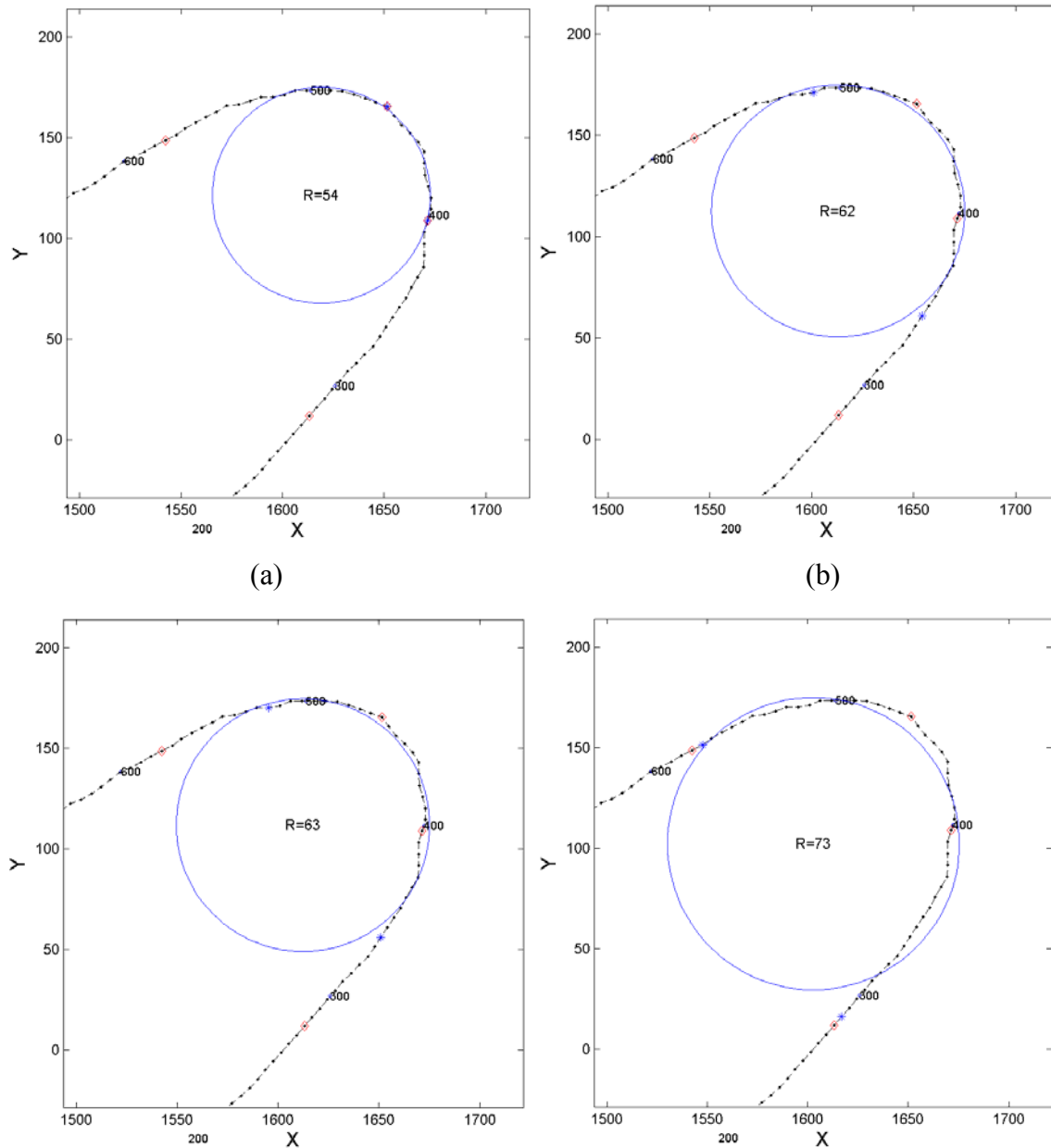


Figure 6.12 Visual comparison of fittings

Figure 6.13 is the R/W vs. length curve for the bend shown in Figure 6.12. It helps explain how the set of circles for the bend were obtained. Point A and B were identified by Criterion Line Method. Then they were extended in both directions by a certain percentage of curve length AB and R/W shouldn't be larger than 10. Thus point A was extended to the region from C to E in which the first correct boundary point was

supposed to fall. The second correct boundary point would fall in the region from D to F. Next step was to pick one point from each region and fit a circle. There are 15 points from C to E and 14 points from D to F. So there are $15 \times 14 = 210$ circles which must include the best fit circle. The R/W vs. length curve is almost symmetric to the vertical line passing the lowest point, which indicates the bend can also be symmetric to the point corresponding to the lowest point in Figure 6.13. In the process of producing the set of circles, it is reasonable to pick approximately symmetric points. By starting from point C and D and moving up one point on each side one time, only 15 circles were produced. The computing time was thus greatly shortened without compromising the precision.

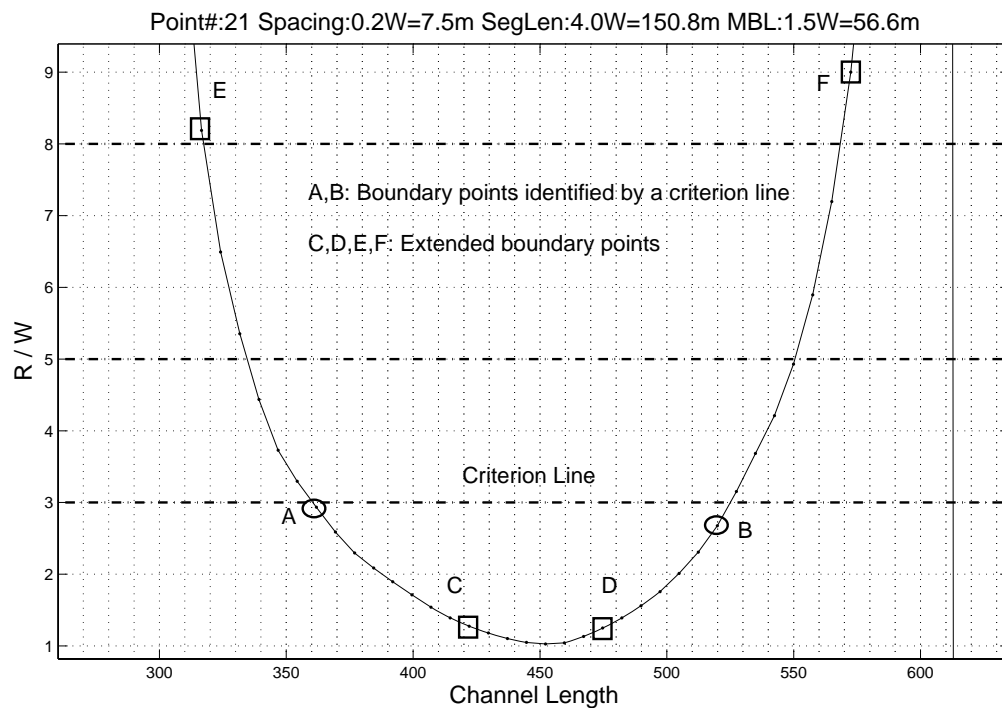


Figure 6.13 Extension of bend boundaries

Finding the best fit circle is the ultimate goal of geometry study. The challenge is how to numerically describe the visually best circles so that they can be identified by a computer. Several methods have been developed for this goal.

6.4.2 Least square error method

The first method is to make a judgment based on least square error. The most common error term is:

$$ER = \sqrt{\frac{\sum (R^* - R_i)^2}{n}} \quad (6.29)$$

- R^* : Radius of the fitted circle;
 R_i : Distance between the center of the fitted circle and point i on the bend;
 n : Number of points used to fit the circle, also a measure of bend angle.

This error term always tends to pick up the shortest possible segment, very likely segments of three points. A circle like the one in Figure 6.12(a) or smaller ones is very likely to be selected. An ideal circle should have a small error (high degree of closeness) and a large n value (for accommodating global geometric feature of the bend). The smaller the error is and the larger n value is, the better the circle will be. However, there is a trade off between small error and large n value. That is to say if n value is increased, the error will go up accordingly. The degree of closeness should be well balanced against n value so that a good circle can be produced. Equation (6.29) is not a well balanced expression, where too much weight is given to error value. By considering giving some weight to n , this error term is generated:

$$ER = \sqrt{\frac{\sum (R^* - R_i)^2}{n^\alpha}}, \alpha > 1 \quad (6.30)$$

If this error term works, there is an optimum α value associated with each bend and it may vary with bends. Unfortunately, in many cases no α value is available that could produce a best fit circle.

6.4.3 Criterion line method

The second method is to use the criterion lines. As described in previous section, three well placed criterion lines can help identify most of the bends. In some cases good circles can be fitted to the identified bends. The good results normally come from trial

and error. A criterion line good for this river may not be suitable for the next river or for the next migrated river. Right now there is no way to calculate the right criterion lines for a river. If this method has to be used, the criterion lines are chosen based on experience and applied to migrated channels. A popular R/W range is from 3 to 8.

6.4.4 Second derivative method

The third method is to apply the d^2R/ds^2 vs. channel length curve to identify the exact boundaries. Similar as the Criterion Line Method, this method can identify both region of the bend and the boundaries for best fit circle. In Figure 6.11, the second derivative of R is close to x axis for a bend. At the two ends of the bend, the second derivative doesn't change gradually. There is a sudden jump. The turning points here happen to be the boundary points that are looked for. The way to find these segments is the same as using criterion lines. But the criterion lines here have fixed values which are good for most rivers. Two criterion lines are set with $d^2R/ds^2=0.3$ and $d^2R/ds^2=0.8$. This method works well for the case shown in Figure 6.11. The inherent drawbacks would make it fail for other cases. When the channel curve is not so smooth, there is a lot of fluctuation in d^2R/ds^2 which causes the result unreliable. When two bends of different radius are next to each other or are very close, the method can hardly tell which is which.

6.4.5 Balanced method

The fourth method is to balance least error against bend angle with a numerical expression. The expression is:

$$\alpha = \frac{1}{\varphi} + b \sqrt{\frac{\sum_{i=1}^n (R_i - R^*)^2}{n}} / R^* \quad (6.31)$$

Where,

α : Target term.

φ : Bend angle.

b: Empirical coefficient.

R_i : The distance between point i and the center.

R^* : Radius of the fitted circle

n : Number of points used to fit the circle

α is a combination of the bend angle term and the term of fitting error. The larger the bend angle is, the smaller α is; the smaller the fitting error is, the smaller α is. The circle with the smallest α will be treated by Equation (6.31) as the best circle and it can be called the numerically best circle. The coefficient b plays a critical role in balancing bend angle and fitting error. A good choice of b would make the numerically best circle the same or very close to the visually best circle.

Figure 6.14 are graphs for the relationship between α value and R . Different b values make the minimum α value to occur at different R . According to visual observation of Figure 6.12 and additional graphs, there are several best fitting circles. The range of radius is from 61 to 64. The b values that can make minimum α value to fall in this range are the values being looked for. For this bend, $b=30$ and $b=50$ make this happen. If Equation (6.31) does work at some bends, the best b value will differ from bend to bend. So far no other properties of the bend have been associated with the best b values. The determination of b is empirical at this stage. It was found $b=100$ gives a relatively good result for many cases. In Figure 6.14, $b=100$ produces a result of $R=54$ ($R/W=54/37.7=1.4$). The ideal R/W is $62/37.7=1.6$. The difference in R/W ratio is $(1.6-1.4)/1.6=12.5\%$.

For some bends, either term of α or α itself has more than one values corresponding to a single R . Although this makes it hard to find the minimum value, the outcome is not bad. This situation can be avoided by using the ordinal number of the circles as abscissa. Further work still needs to be done to optimize the solution. An application of this method on the Guadalupe River is shown in Figure 6.15.

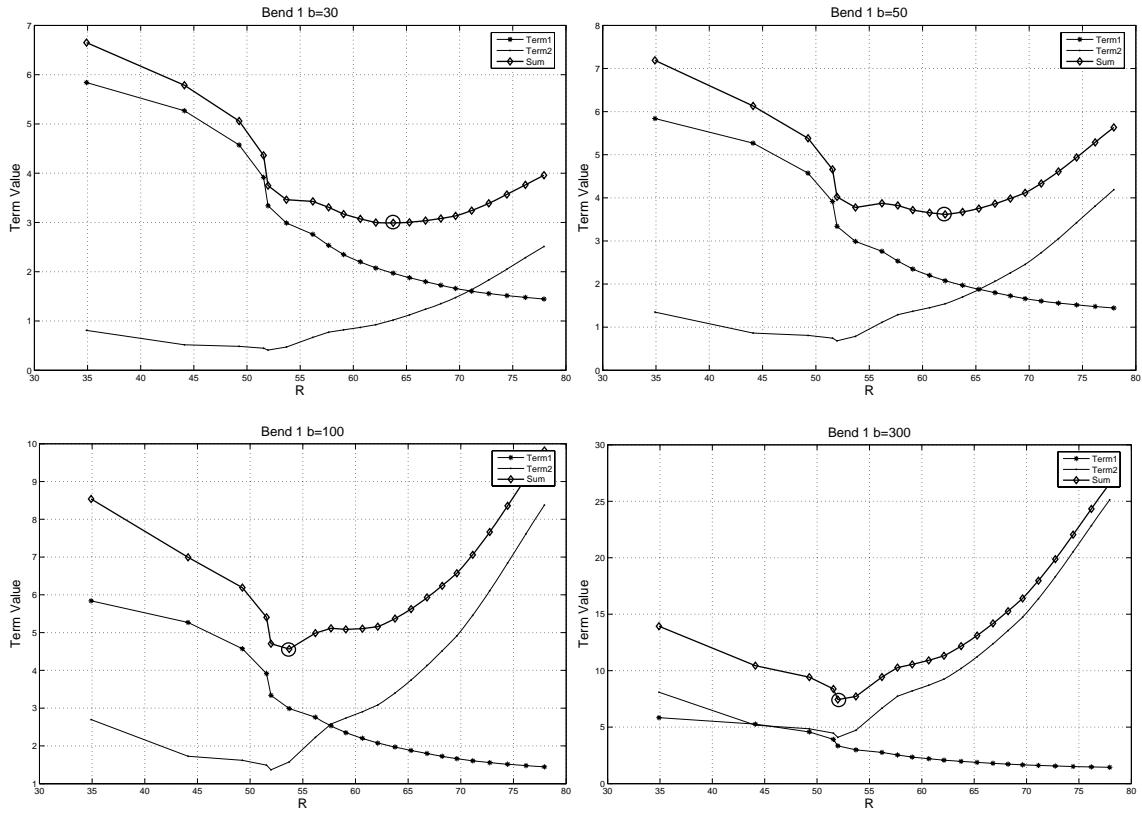


Figure 6.14 The influence of parameter b

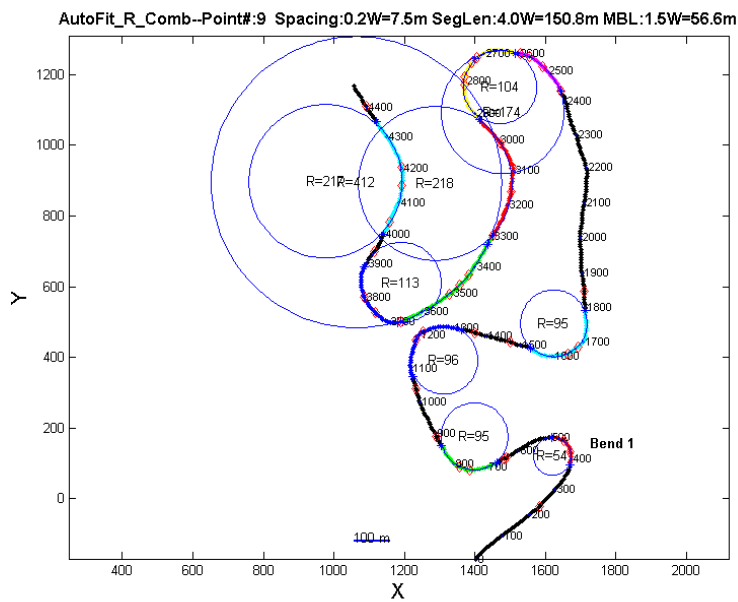


Figure 6.15 Same b coefficient applied to all bends

6.5 CALCULATE BEND ANGLE

When the best fit circle is available, the bend angle can be calculated based on the two boundary points and the center of the circle. In some cases, the bend angle calculated based on the two straight lines connected with the bend looks more appropriate. The former method was used in the program.

CHAPTER VII FUTURE HYDROGRAPHS

7.1 INTRODUCTION

Compared to using the flow obtained by numerical simulation in other research efforts, the application of hydrograph in the prediction of meander migration is a significant advantage of this research, where the influence of each single daily flow is accounted for and summed up. The prediction is always about the future locations of a river. But the flow causing that change hasn't occurred yet. So a future hydrograph needs to be generated based on existing hydrograph data. Will the predicted hydrograph stand for the flow conditions that will happen? It's hard to tell. What can be reasonably assumed is that the hydrograph for the next 100 years has the same or similar statistic properties as the existing 100-year hydrograph. Based on this idea, a method is proposed to generate statistically equal future hydrographs. Dr. Paolo D'odorico provided helpful advices to this work.

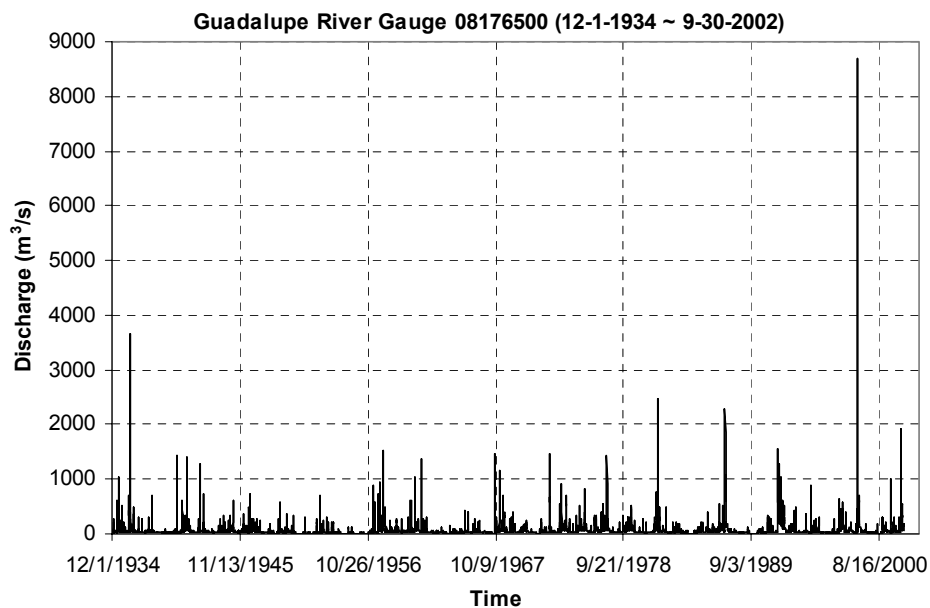


Figure 7.1 Hydrograph of Guadalupe River Gauge Station 08176500

7.2 THE DISTRIBUTION OF DAILY DISCHARGE

In order to extract the statistical properties of an existing hydrograph, the type of distribution of the hydrograph should be determined first. It has been observed that a hydrograph mostly follows lognormal distribution. Figure 7.1 is the hydrograph of Guadalupe River gauge station 08176500 at Victoria, TX from 12-1-1934 to 9-30-2002. Figure 7.2 is the probability density function (PDF) of the original data and fitted distribution. Figure 7.3 is the cumulative density function (CDF) of the original data and fitted distribution. These curves show the daily discharge data can be roughly considered as a lognormally distributed random variable.

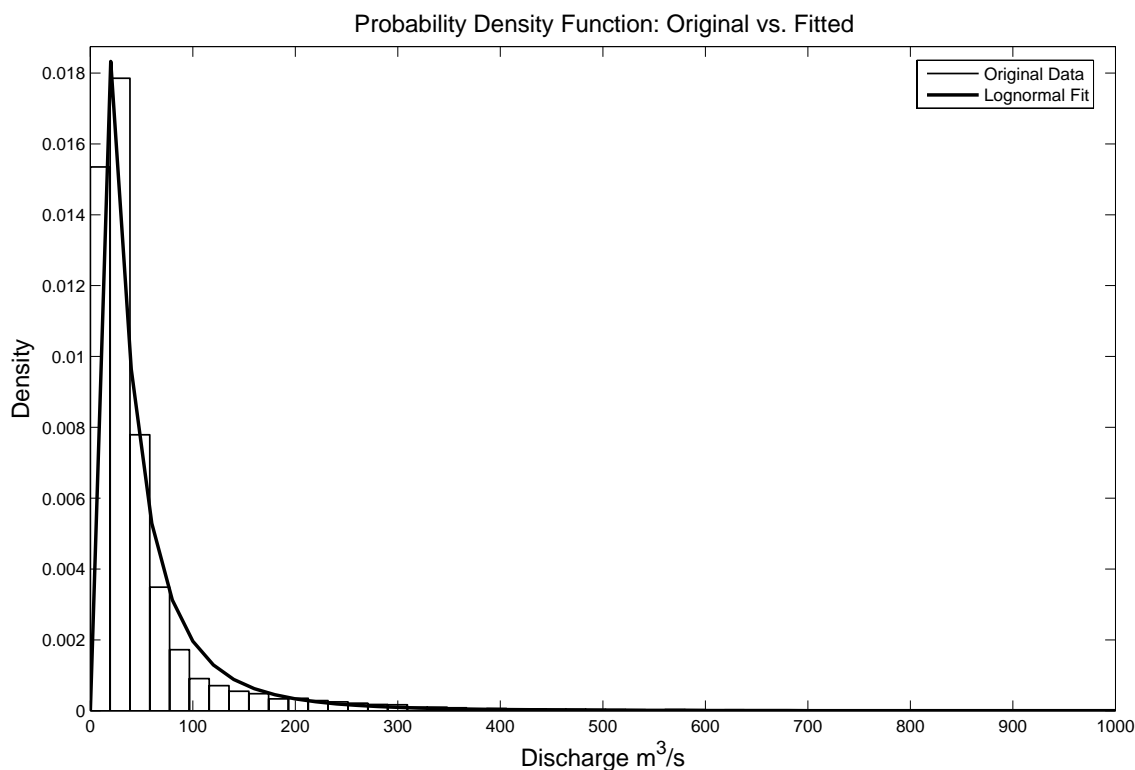


Figure 7.2 PDF of original data and fitted distribution – Guadalupe River

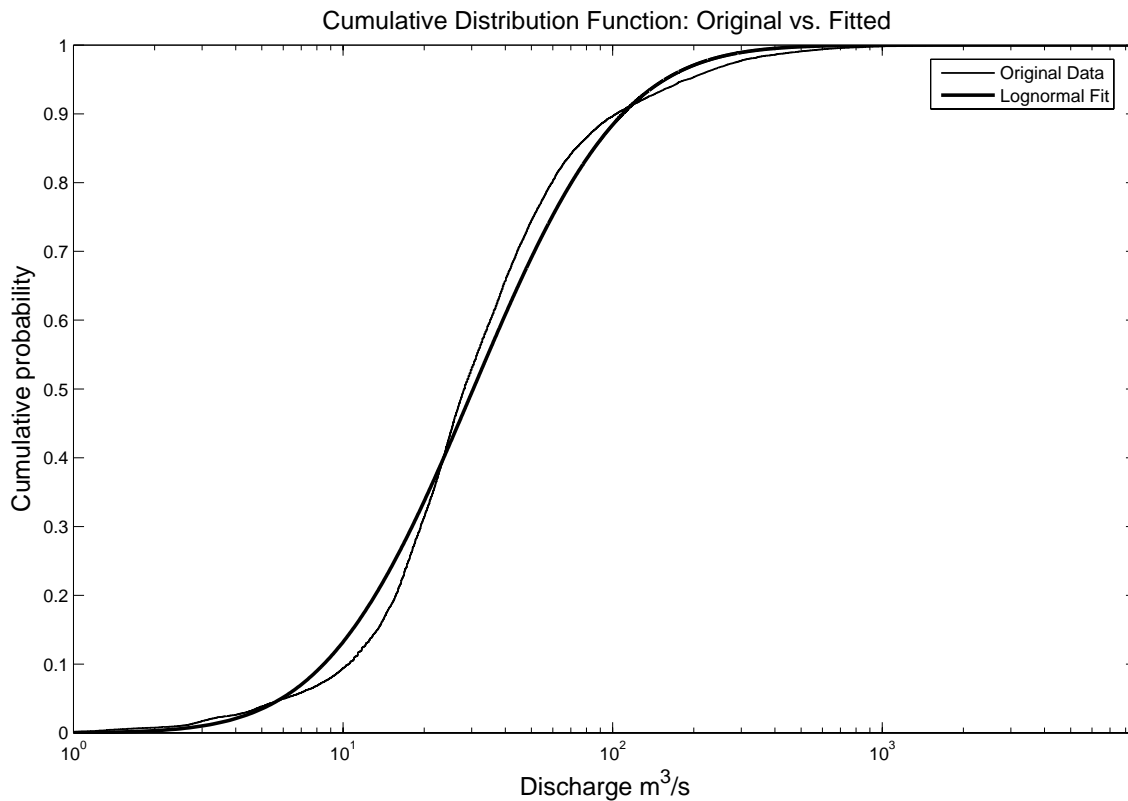


Figure 7.3 CDF of original data and fitted distribution – Guadalupe River

Figure 7.4 is the hydrograph from Potomac River near Washington D.C. Little Falls Pump Station, USGS gage number 01646500. In order to consider the discharge through Woodrow Wilson Bridge not far downstream, the original flow rate was timed by 1.03. The recorded daily flow is from March 1, 1930 to September 30, 2003. Figure 7.5 and Figure 7.6 demonstrate that lognormal distribution fits these data better than the Guadalupe River.

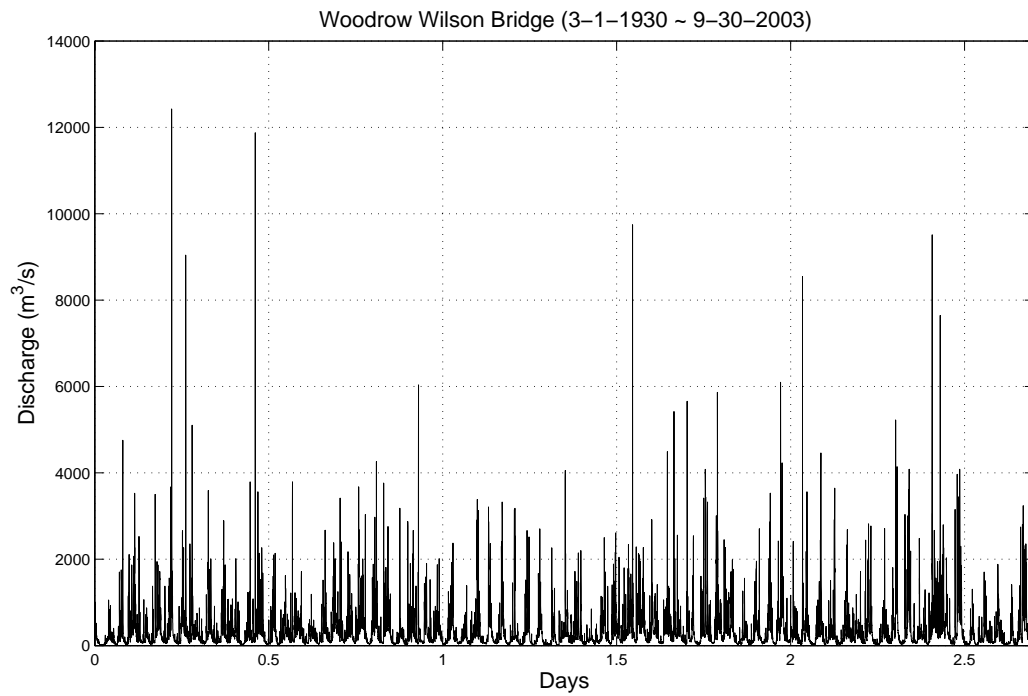


Figure 7.4 Hydrograph of Woodrow Wilson Bridge

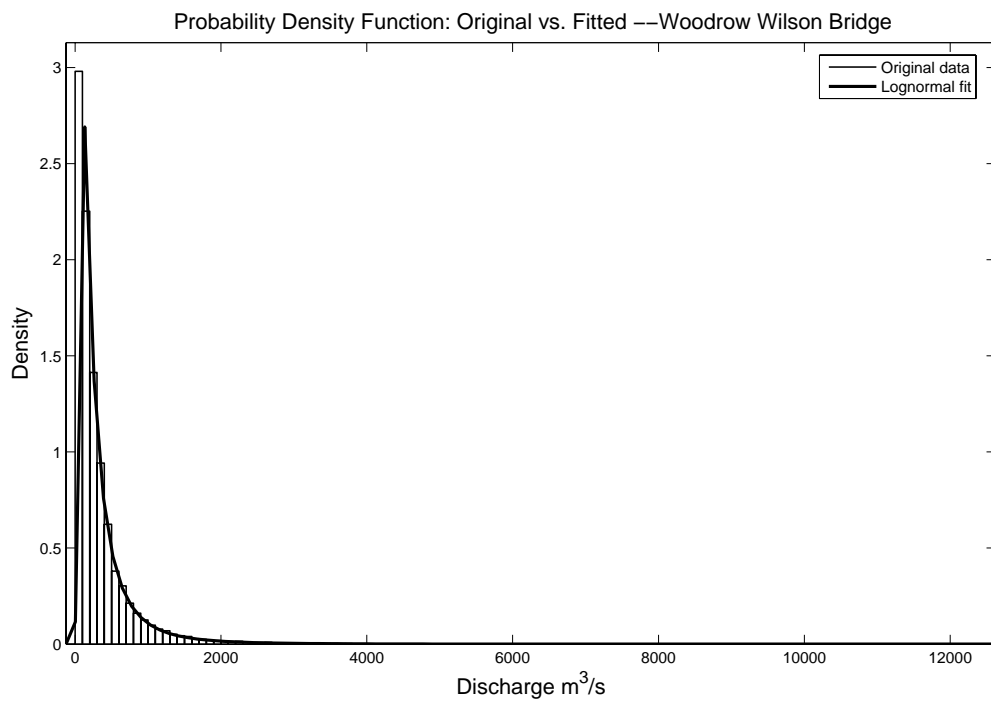


Figure 7.5 PDF of original and fitted distribution

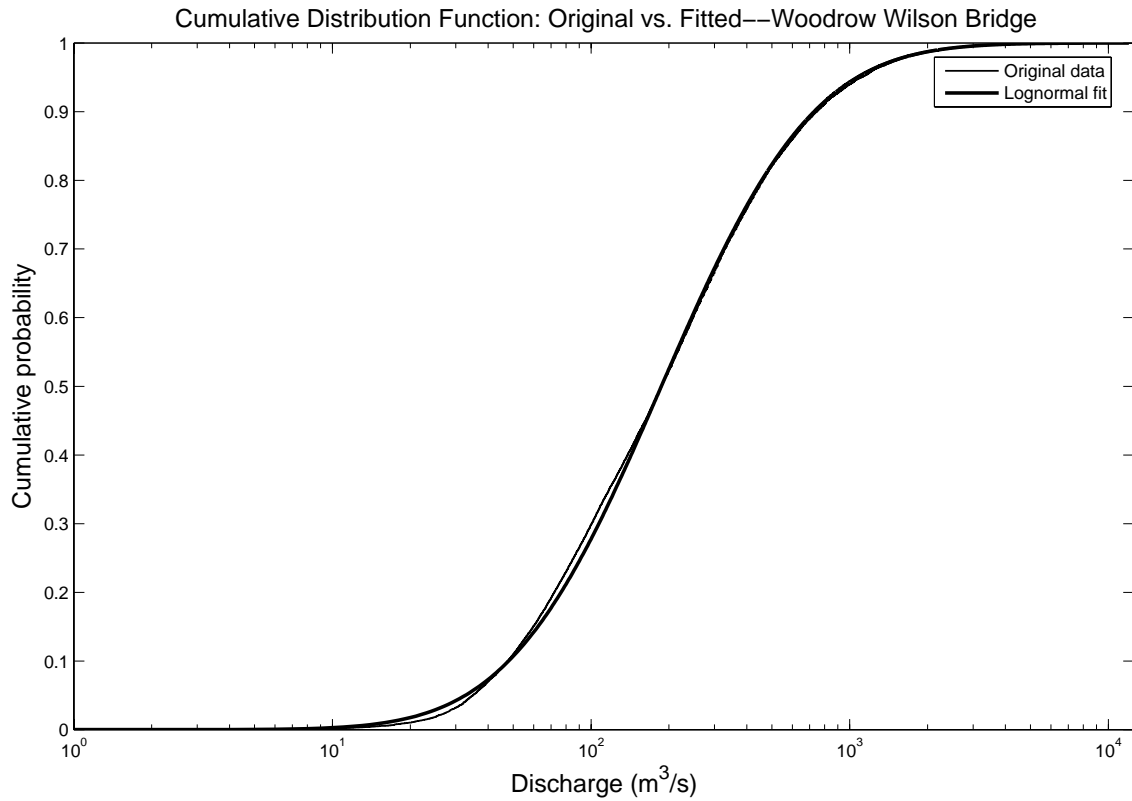


Figure 7.6 CDF of original data and fitted distribution

Assume Y is a normal random variable. Then $Q=e^Y$ has a lognormal distribution. The probability density function of Q , the flow rate, is:

$$f(q|\mu, \sigma) = \frac{1}{q\sigma\sqrt{2\pi}} e^{-\frac{(\ln q - \mu)^2}{2\sigma^2}} \quad (7.1)$$

Where μ, σ are the mean and standard deviation of Y , the normal random variable. Let m, s be the mean and standard deviation of Q , the lognormal random variable. The following formulas can be derived:

$$m = e^{\mu + \frac{\sigma^2}{2}} \quad (7.2)$$

$$s = e^{\mu + \frac{\sigma^2}{2}} \sqrt{e^{\sigma^2} - 1} \quad (7.3)$$

$$\mu = \ln\left(\frac{m^2}{\sqrt{m^2 + s^2}}\right) \quad (7.4)$$

$$\sigma = \sqrt{\ln\left(\left(\frac{s}{m}\right)^2 + 1\right)} \quad (7.5)$$

The lognormal fitting is performed by using Matlab version 7.0.1. The fitted means and standard deviations of the normal random variable Y and lognormal random variable Q for both cases are listed in Table 7.1. For a confidence level of 99%, the ranges of these two parameters for Guadalupe River are:

$$\mu_Y \in [3.399 \ 3.432], \sigma_Y \in [0.982 \ 1.008].$$

The ranges for Woodrow Wilson Bridge are:

$$\mu_Y \in [5.214 \ 5.248], \sigma_Y \in [1.049 \ 1.073].$$

This shows the fitting criteria are well met. Table 7.1 has a detailed comparison of the mean and standard deviation of the original data and fitted distribution for these two cases.

Table 7.1 Comparison of statistical parameters (Unit: m³/s)

	Guadalupe River		Woodrow Wilson Bridge	
	Original Data	Fitted Distribution	Original Data	Fitted Distribution
μ_Y	3.087	3.415	5.244	5.231
σ_Y	1.358	0.996	1.057	1.061
m_Q	55.1	49.9	331.330	328.255
s_Q	127.2	65.1	475.347	473.746

μ_Y and σ_Y are the mean and standard deviation for the normal variate Y. m_Q and s_Q are the mean and standard deviation for the lognormal variate Q. m_Q and s_Q for the original

data are calculated by using standard formulas. μ_Y and σ_Y for the original data are calculated by using Formulas (7.4) and (7.5). The difference of standard deviation s_Q for Guadalupe River is about 50%, for Woodrow Wilson Bridge is about 0.3%. The hydrograph of Woodrow Wilson Bridge is closer to a lognormal distribution. The fitted parameters for Guadalupe River should be used with caution. The fitting results show not all hydrographs are in well-behaved lognormal distribution. The fitting is not perfect but in general the statistical features are close to the original data. The compromise in perfection is compensated by the simplicity of the process. In the MEANDER program, m_Q and s_Q for the original data are used to generate future hydrographs.

7.3 DEFINITION OF THE 100/500-YEAR FLOOD

The term “100-year flood” is somewhat misleading. Many people mistakenly think it is a description of the flood that occurs once in every 100 years. Instead, it describes a flood elevation that has a 1-percent chance of being equaled or exceeded in one year. Therefore, “500-year flood” is a flood elevation that has a 0.2-percent (1/500) chance of being equaled or exceeded in one year. To calculate the probability for a 100-year flood to be equaled or exceeded within a certain period of time, say N years, this formula can be used:

$$p = 1 - \left(1 - \frac{1}{100}\right)^N \quad (7.6)$$

$\left(1 - \frac{1}{100}\right)$ is the probability for a 100-year flood not to occur in one year. $\left(1 - \frac{1}{100}\right)^N$ is the probability for a 100-year flood not to occur in N year. So $p = 1 - \left(1 - \frac{1}{100}\right)^N$ is the probability for a 100-year flood to be equaled or exceeded in N years. The design life of a new bridge is about 75 years during which the probability for a 100-year flood to occur is $p=52.9\%$. If $N=100$ years, $p=63.4\%$.

What’s the probability for a 100-year flood to be equaled or exceeded in one day? Assume one year has 365 days. Let p_x denote the unknown. This problem can be solved by using Equation (7.6). The probability for the flood to be equaled or exceeded in one

day is p_x . The probability for a 100-year flood to be equaled or exceeded in 365 days (one year) is $1/100$. The equation can be written as:

$$\frac{1}{100} = 1 - (1 - p_x)^{365} \quad (7.7)$$

The solution is $p_x = \frac{1}{100} \times \frac{1}{363.2}$, very close to $\frac{1}{100} \times \frac{1}{365}$. The probability for a 500-year flood to occur in one day is $p_x = \frac{1}{500} \times \frac{1}{364.6}$, very close to $\frac{1}{500} \times \frac{1}{365}$. With this comparison in mind, the approximate values can be used.

7.4 COMPUTE DISCHARGE DISTRIBUTION BASED ON 100/500-YEAR FLOOD

If an existing hydrograph is given, the distribution of daily discharge can be obtained by two methods. One is to do a lognormal fitting and obtain the parameters of mean and standard deviation. The other one is to calculate the mean and standard deviation of the hydrograph with the assumption that it follows a lognormal distribution. The latter is used for simplicity and reasonable precision.

In some other cases a hydrograph is not available and only 100-year flood and 500-year flood are provided. The lognormal distribution (μ, σ) needs to be solved based on this information. The two corresponding probabilities of exceedance for a 100-year flood and a 500-year flood have been given in the previous section. For the CDF equation of an analytical lognormal distribution, there are two unknowns and two equations. So the unknowns can be solved as shown below.

The CDF of Q can be expressed as:

$$\begin{aligned} P(Q \leq q) &= F\langle q | \mu, \sigma \rangle = \frac{1}{\sigma\sqrt{2\pi}} \int_0^q \frac{1}{t} e^{-\frac{(\ln t - \mu)^2}{2\sigma^2}} dt = \frac{1}{\sqrt{2\pi}} \int_0^q e^{-\frac{(\ln t - \mu)^2}{2\sigma^2}} d\left(\frac{\ln t - \mu}{\sqrt{2\sigma}}\right)\sqrt{2} \\ &= \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\frac{\ln q - \mu}{\sqrt{2\sigma}}} e^{-z^2} dz = \frac{1}{2} + \frac{1}{2} \operatorname{erf}\left(\frac{\ln q - \mu}{\sqrt{2\sigma}}\right) \end{aligned} \quad (7.8)$$

The term erf indicates *error function*. Definition of error function and inverse error function is as follows:

$$y = \operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \cdots (-\infty < x < +\infty) \quad (7.9)$$

$$\operatorname{erf}(-\infty) = -1, \operatorname{erf}(\infty) = 1 \quad (7.10)$$

The inverse error function is:

$$x = \operatorname{erfinv}(y) \cdots (-1 < y < 1) \quad (7.11)$$

Thus the cumulative distribution function of a lognormal distribution can be written as:

$$\operatorname{erfinv}(2P-1) = \frac{\ln q - \mu}{\sqrt{2}\sigma} \quad (7.12)$$

There are two applications for this formula. 1. Given the distribution (μ, σ) and the probabilities, $P_1(Q \leq q_1)$, and $P_2(Q \leq q_2)$, the corresponding discharges q_1 and q_2 can be found. This can be used to find Q_{100} and Q_{500} which will be discussed in the next section. 2. Given two points (q_1, P_1) , (q_2, P_2) on the CDF curve, the lognormal distribution (μ, σ) can be solved. Here (Q_{100}, P_{100}) , (Q_{500}, P_{500}) are given. The data of the distribution are daily average discharge. The CDF curve based on this data tells the probability for a certain level of discharge not to be exceeded in one day. It makes more sense to use the probability for a 100-year flood (500-year flood) to occur in one day in stead of one year. So in the above equations: $P_{100} = 1 - \frac{1}{100} \times \frac{1}{365}$, $P_{500} = 1 - \frac{1}{500} \times \frac{1}{365}$.

Define:

$$u_{100} = \frac{\ln Q_{100} - \mu}{\sigma} = \sqrt{2} \times \operatorname{erfinv}(2P_{100} - 1) = 4.03417 \quad (7.13)$$

Likewise,

$$u_{500} = \frac{\ln Q_{500} - \mu}{\sigma} = \sqrt{2} \times \operatorname{erfinv}(2P_{500} - 1) = 4.39733 \quad (7.14)$$

The inverse error function can be solved by using any pertinent numeric recipe or Matlab.

$$\sigma = \frac{1}{u_{500} - u_{100}} \ln\left(\frac{Q_{500}}{Q_{100}}\right) \quad (7.15)$$

$$\mu = \ln(Q_{100}) - u_{100} \cdot \sigma \quad (7.16)$$

Once the mean and standard deviation of the normal variate Y are obtained, the lognormal random variable is determined to be $Q=e^Y$.

7.5 THE DETERMINATION OF 100/500-YEAR FLOOD

100-year flood and 500-year flood reflect the risk level a river experiences. The determination can be based on an existing hydrograph which contains sufficient data. The more data is available, the more accurate the result will be.

Continuous PDF and CDF curves for a hydrograph can be drawn as shown in Figure 7.2 and Figure 7.3. The probability of exceedance curve, which is also called survivor function, is obtained by subtracting CDF from 1, as shown in Figure 7.7. A point (Q, P_Q) means the probability for discharge Q to be equaled or exceeded during a unit of time is P_Q . The unit of time is the duration of a single Q value. If Q is daily average discharge, the unit of time is one day. The probability for a 100-year flood to be equaled or exceeded in one day is $1/(100 \times 365)$ and the probability for a 500-year flood to be equaled or exceeded in one day is $1/(500 \times 365)$. On an ideal daily flow probability of exceedance curve, the corresponding discharges Q_{100} , Q_{500} are what to be sought.

A probability of exceedance curve can come from a hydrograph and can also be determined by a given distribution (μ, σ) . Lognormal fitting generates μ and σ . The original data also has μ and σ which are normally different from the fitted results. Figure 7.7 shows these three probability of exceedance curves of Guadalupe River. Any point on the curve of the original denotes a concept of percentile rather than probability since everything has happened. The other two curves predict the probability for the occurrence of a certain event. Thus two sets of 100/500-year flood can be obtained and compared.

For the hydrograph of Guadalupe River gage station 08176500, if the fitted distribution is used, the 100-year flood and 500-year flood are found to be:

$$Q_{100} = 1692 \text{ m}^3/\text{s}, Q_{500} = 2429 \text{ m}^3/\text{s}.$$

If the lognormal distribution based on the original μ and σ is used, the 100-year flood and 500-year flood are:

$$Q'_{100} = 5251 \text{ m}^3/\text{s}, Q'_{500} = 8601 \text{ m}^3/\text{s}.$$

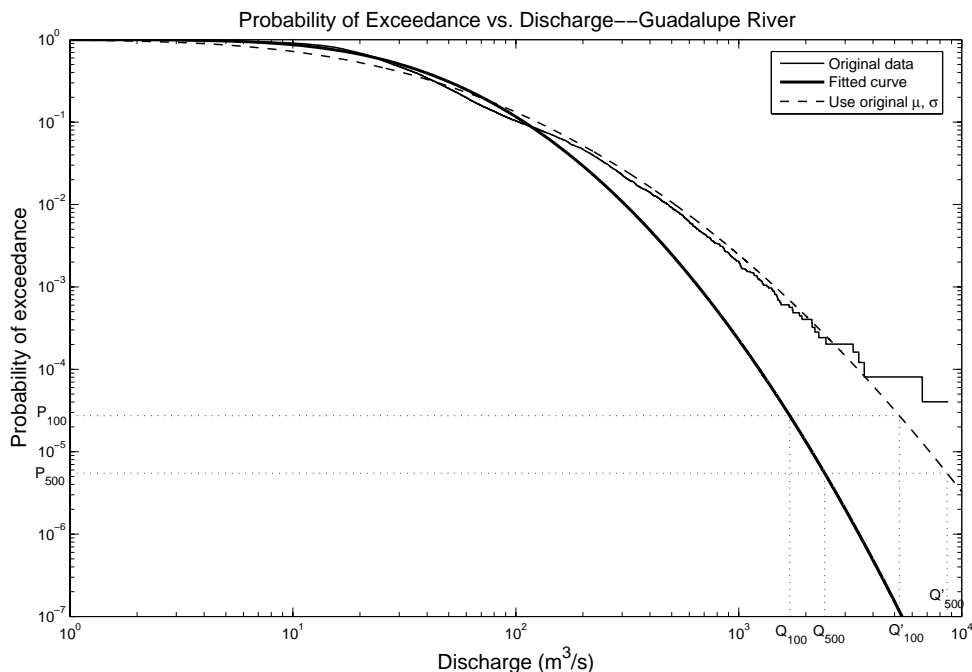


Figure 7.7 Probability of exceedance curves - Guadalupe River

The maximum recorded discharge is $8693 \text{ m}^3/\text{s}$. According to the fitted curve, the probability for this flow to be equaled or exceeded is $6.8 \times 10^{-9} = \frac{1}{400,782} \times \frac{1}{365}$ which is a 400,782-year flood. According to the distribution based on the original μ and σ , the probability for the maximum recorded discharge to be equaled or exceeded is $5.3 \times 10^{-6} = \frac{1}{518} \times \frac{1}{365}$ which is a 518-year flood. It can be seen in Figure 7.7 the curve based on the original μ and σ fits the data record much better when the discharge is larger than $200 \text{ m}^3/\text{s}$. Its Q_{100} and Q_{500} are more reasonable.

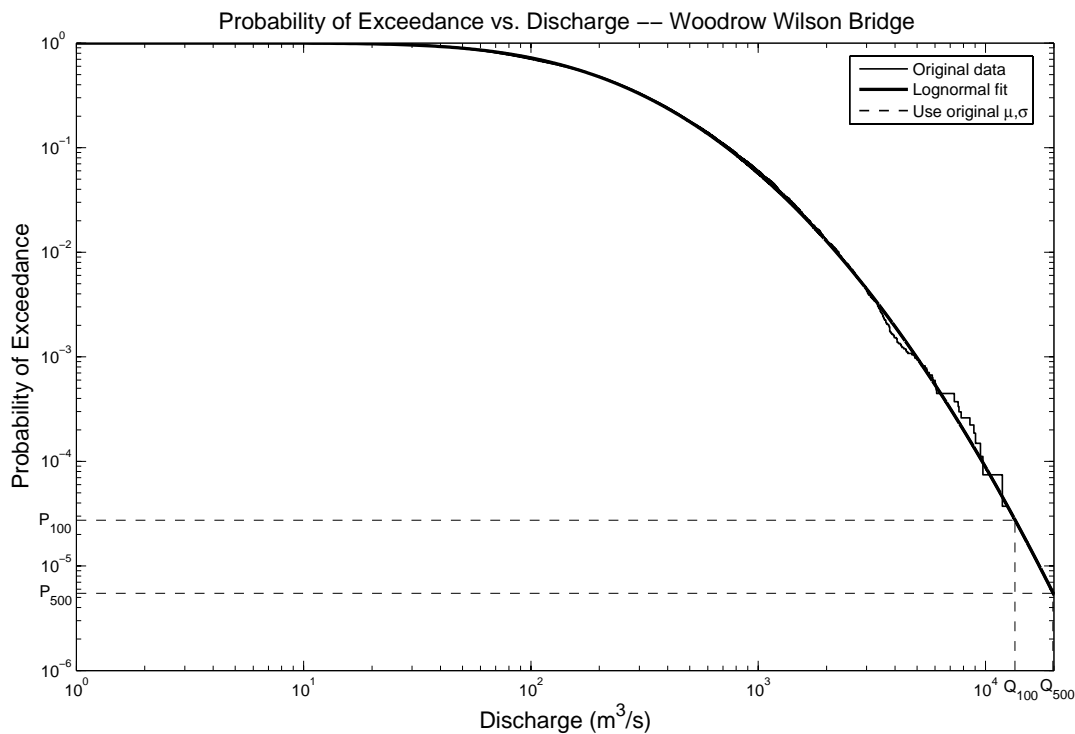


Figure 7.8 Probability of exceedance curves – Woodrow Wilson Bridge

For the case of Guadalupe River the method of using fitted distribution drastically underestimates high risk floods, which makes this method undesirable. On the hydrograph plot, the maximum flood is a single spike with a magnitude much larger than other peak floods. Spikes like this are critical in estimating 100/500-year flood of the river but are of tiny importance to the fitting process which equally treats each daily flow. This explains why good results were not produced by this method.

If the fitted distribution is used for the hydrograph of Woodrow Wilson Bridge, the 100-year flood and 500-year flood are found to be:

$$Q_{100} = 13513 \text{ m}^3/\text{s}, Q_{500} = 19865 \text{ m}^3/\text{s}.$$

If the lognormal distribution based on the original μ and σ is used, the 100-year flood and 500-year flood are:

$$Q'_{100} = 13487 \text{ m}^3/\text{s}, Q'_{500} = 19799 \text{ m}^3/\text{s}.$$

The maximum recorded discharge is 12425 m³/s. According to the fitted distribution, the probability for this flow to occur is $3.826 \times 10^{-5} = \frac{1}{71.6} \times \frac{1}{365}$. If the distribution based on the original μ and σ is considered, the probability for this flow to occur is $3.801 \times 10^{-5} = \frac{1}{72.1} \times \frac{1}{365}$. The closeness of this comparison can also be seen in Figure 7.8. These two curves almost overlap and they well fit the original data. This shows this hydrograph strictly follows lognormal distribution.

It can be seen from above, the fitted distribution works well only when the original data is a well-behaved lognormal distribution. The lognormal distribution based on the original μ and σ is always better than the fitted one regarding the prediction of high risk floods. When a hydrograph is given, it is reasonable to treat it as a lognormal distribution with the same mean and standard deviation as the original data.

7.6 RANDOM NUMBER GENERATION

With the analytical lognormal distribution function ready, the extraction of the statistical properties of the original data is finished. The next step is to produce future hydrographs by using random number generation (RNG) technique.

The generation of lognormal random numbers can be a sampling process done on a CDF curve of a lognormal distribution. First generate uniformly distributed random numbers in the range of 0 to 1. Then apply these uniform random numbers to the y axis (cumulative probability) of the CDF curve and the corresponding x values (discharge) are the random numbers to be found. An analytical solution can be obtained from Equation (7.12). Given μ , σ , P, solve for q. This method directly solves the lognormal CDF function and can be easily understood. The drawback is that an inverse *error function* needs to be solved for each random number which makes it not efficient even with high performance computers. Alternatives have been developed to generate random numbers for normal distribution Y. Lognormal random numbers are simply e^Y ,

$e=2.71828$. The following is a description of the methods used in this research for generating uniform random numbers and normal random numbers.

The linear congruential generators are most commonly used for generating random numbers of uniform distribution. The form is:

$$X_n = a * x_{n-1} + b \text{ mod } M, n=1, 2, 3 \dots, \text{ given } x_0.$$

“mod M” means the right hand side is first divided by M, then is replaced by the remainder. Let $a=351$, $b=42$, $M=100$, $x_1=81$. A series of random numbers can be obtained: 81, 73, 65, 57, 49, 41, 33, 25, 17, 9, 1, 93, 85, 77, 69, 61, 53, 45, 37, 29, 21, 13, 5, 97, 89, 81... The random numbers start to repeat at a certain point. These numbers are in the range of [0 100]. To obtain random numbers in a given range, transformation is needed. For example, divide each of the random numbers by $M=100$, the resulted random numbers will be in the range of [0 1]. Parameters a , b , and M determines the characteristics of the random number generator. The choice of x_0 determines the particular sequence of random numbers that is generated. In the MEANDER program, the values used are: $a=663608941$, $b=0$, $M=2^{32}$.

Normal random number generators are mostly based on Box-Muller transformation. Box-Muller transformation transforms a two-dimensional continuous uniform distribution into a two-dimensional bivariate normal distribution. If x_1 and x_2 are uniformly and independently distributed between 0 and 1, then z_1 and z_2 as defined below have a normal distribution with mean $\mu=0$ and variance $\sigma^2=1$. Detailed information can be found at <http://mathworld.wolfram.com/Box-MullerTransformation.html>.

$$z_1 = \sqrt{-2 \ln x_1} \cos(2\pi x_2) \quad (7.17)$$

$$z_2 = \sqrt{-2 \ln x_1} \sin(2\pi x_2) \quad (7.18)$$

Given two uniform variates generated by the uniform random number generator, two normal variates can be obtained. Free implementation code in C or Fortran language can be found on the Internet.

For the two random number generators (RNG) introduced here, the C RNG (RNG written in C language) is better than the one that directly solves lognormal CDF.

Matlab also has a lognormal random number generator which applies the cutting edge technique in this field. It is used as a benchmark to test the reliability of other two random number generators. Table 7.2 compares the results of 4 runs of these random number generators. The maximum difference for μ_Q and σ_Q between C RNG and Matlab RNG are only 2.45% and 4.57% respectively. C RNG is used in the program to avoid calling Matlab code.

7.7 RISK ANALYSIS

Tens of thousands of hydrographs can be generated based on the computer capacity and the user's needs. The more hydrographs are used, the more reliable the risk analysis will be. For each hydrograph, a final location of the river is calculated. Probabilistic predictions are performed based on the thousands of possible river locations. A program is to be written by Namgyu Park which can tell something like in 75 years, the probability for the river to reach a certain line and further is 10%.

Table 7.2 Comparison of random number generators

Methods	Items	Run 1		Run 2		Run 3		Run 4	
		Result	Error	Result	Error	Result	Error	Result	Error
C RNG	μ_Y	4.9894	0.20%	4.998	0.04%	4.9908	0.18%	4.9919	0.16%
	σ_Y	0.9998	0.02%	1.0012	0.12%	1.0027	0.27%	0.9991	0.09%
	μ_Q	241.3	2.43%	245.1	2.43%	242.5	0.82%	241.7	1.02%
	σ_Q	312.6	4.11%	319.3	0.66%	308.8	4.57%	303.3	1.17%
Solve CDF RNG	μ_Y	4.9994	-0.01%	5.0117	0.23%	5.002	0.04%	4.976	-0.48%
	σ_Y	0.9958	-0.42%	1.0047	0.47%	1.0036	0.36%	0.9954	-0.46%
	μ_Q	244.5	-1.96%	249.2	1.44%	244.9	-1.11%	236.5	-3.04%
	σ_Q	332.8	0.28%	321.4	-4.95%	310.4	-2.74%	311.7	4.22%
Matlab RNG	μ_Y	5.0011	0.02%	5.0009	0.02%	5.0101	0.20%	4.9851	0.30%
	σ_Y	1.0011	0.11%	1.0049	0.49%	1.0061	0.61%	1.0024	0.24%
	μ_Q	247.3		251.2		244.5		244.2	
	σ_Q	326.0		317.2		295.3		306.9	

Notes:

- $$\mu_{Y-error} = (\mu_{Y-RNG} - \mu_{Y-Input}) / \mu_{Y-Input}$$

$$\mu_{Q-error} = (\mu_{Q-CRNG} - \mu_{Q-MTRNG}) / \mu_{Q-MTRNG}$$
- Input $\mu_Y=5$, $\sigma_Y=1$; Result is the μ_Y , σ_Y , μ_Q , σ_Q of generated random numbers;
- $Q=e^Y$, Q is the flow, Y is the normally distributed variate.
- 10,000 random numbers are generated.
- In "Solve CDF RNG", the most current μ_Q , σ_Q from Matlab RNG are used to calculate the error term for μ_Q , σ_Q , not the values listed in this table.

CHAPTER VIII

THE MEANDER PROGRAM

8.1 INTRODUCTION

The MEANDER computer program implements all the methods developed by the team and is to be presented to practice engineers to predict meander migration. It consists of two major components: graphic user interface (GUI) and numerical implementation. The graphic user interface was adapted from that of SRICOS-EFA program the GUI of which was written by Jinming Xu. In SRICOS-EFA program, the GUI takes user input and forms a source data file. Then a FORTRAN program reads the data file, does the computation and generates output data file. The GUI reads the output file and shows the result graphically. The MEANDER program does all the things in a more efficient way. The GUI and the implementation of the model are written in C++ and are seamlessly linked together. The part of geometry study and graphic output are written in Matlab. Matlab program is compiled and linked to the C++ program so that one executable is made out of different parts. Transferring data from one module to another is not through hard drive but through much faster computer memory. Implementation of the model is the kernel of the MEANDER program. Each member of the team has contributed to different parts of the progress. The program is a most important product ready for practical use.

In this chapter a general view of the MEANDER program is given regarding graphic user interface, some programming techniques, and implemented modules. The part of Risk Analysis is being implemented by Namgyu Park which will be addressed in detail in his dissertation. These items will be introduced in order:

- Input and output user interfaces
- Mixed language programming—C++ and Matlab
- Overview of the whole program
 - Implementation of Geometry Study
 - Implementation of the Hyperbolic Model

8.2 GRAPHIC USER INTERFACE (GUI)

Graphic user interface is indispensable to modern computer program. A lot of resources have been invested on the development of the GUI of SRICOS-EFA. The engine of this code is reused with some moderate modifications and additions. A consistent appearance is maintained and programming time is reduced. Figure 8.1 is the main interface of the MEANDER program which has a similar look to that of the SRICOS-EFA. All the user interfaces and functions can be accessed either through the menu or the buttons. The user buttons from left to right correspond to these interfaces: Units, Geometry input, Soil input, Water input, Table input, Plot input, Run function, Plot output.

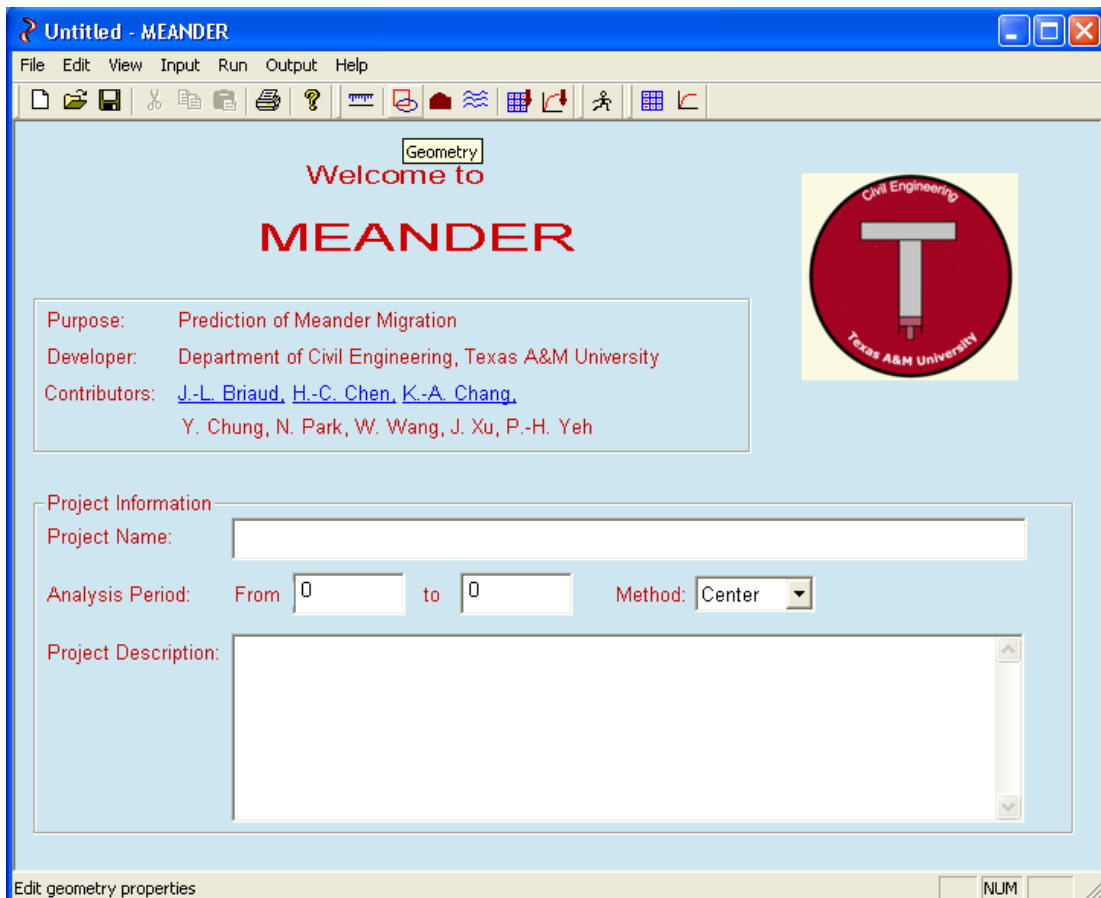


Figure 8.1 The main interface of the MEANDER program

Figure 8.2 shows the unit input dialogue. It allows the user to choose a unit system for the computation. There are two unit systems: metric system and US Customary System (English system). The user should be consistent in the units being used. If metric units are chosen, all the input data are implied to have metric units and output is the same. There is an exception for English units. That is that the EFA curve is always in metric units.

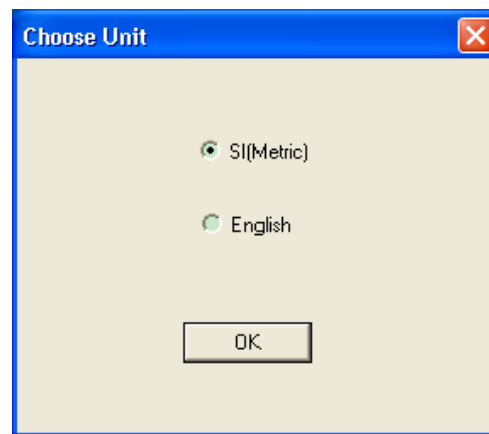


Figure 8.2 Choose a unit

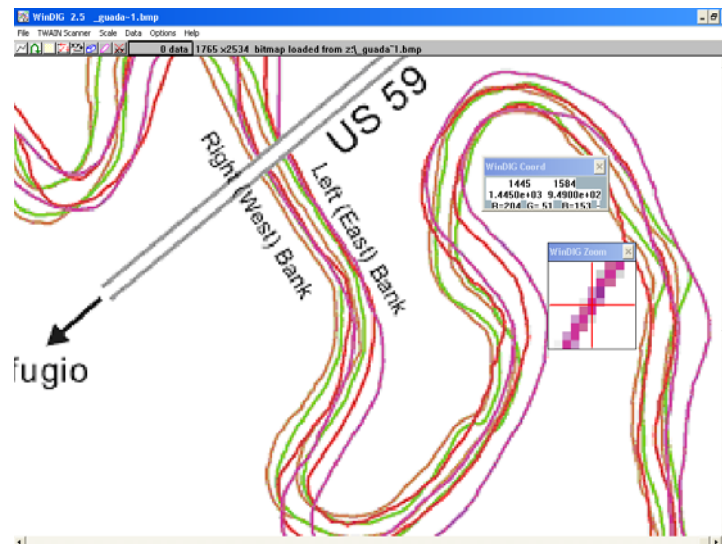


Figure 8.3 Digitize river banks with WinDIG program

Figure 8.4 is the Geometry Input dialog. The data for geometry study is entered here. The user needs to input average river width and the path of coordinate file. Coordinate files can be obtained by digitizing river images with WinDIG program, as shown in Figure 8.3. The channel curve will be drawn immediately after the coordinate file is loaded. The given default values for the parameters are based on experience. Adjustments are needed for certain cases. If the “Fit Circles” button is clicked, circles will be fitted and drawn on the dialog. A better version of figures is shown in two new windows. One is the original channel and fitted circles as shown in Figure 8.5. The other is the R/W versus channel lengthwise distance curve as shown in Figure 8.6. These two graphs are drawn by compiled Matlab code which has a lot of advantages over graphs drawn by C++ code. Besides excellent quality of figures, the user can zoom and pan the graph. It can be exported to any format of raster image or vector image. The coordinates of any point can be displayed at the user’s choice.

Before curve fitting is done, the center line or bank of the river is evenly divided into many segments. Spacing is the distance between two adjacent points. *Spacing Coefficient* is the ratio of spacing to the river width. A default value of 0.2 is given. Reducing the spacing coefficient will smooth the R/W vs. channel length curve. But the computation time will increase as a result.

If Segment length is the length used to calculate the radius of curvature of the middle point (length AB in Figure 6.2). *Segment Length Coefficient* is the ratio of segment length to the river width. It is recommended to use 5. If the segment length coefficient is too small, the bends may not be distinguishable on the R/W versus channel length curve. If this value is increased, the bends will become more obvious but the computation time will become longer.

When the length of an identified bend is shorter than a certain value, no circle will be fitted. This value is called minimum bend length. *Minimum Bend Length Coefficient* is the ratio of minimum bend length to the river width. A value of 2 is suggested here. The smaller the value is, the more circles will be generated. If the value is too large, many significant bends will be excluded.

Another way to eliminate an identified bend with small curvature is to calculate the average distance from a point on the bend to the baseline which connects the first and last point of the bend. If this average distance is smaller than a certain limit, it will be treated as a straight line. The limit is called *Straightness Limit*. If a value of 0 is given, this parameter will be ignored. If the user wants to eliminate some unwanted circles, a number around 5 can be picked. The final desired value can be determined by trial and error.

It's nice to show channel lengthwise distance on the channel. Then the user needs to specify *Tick Spacing*. There is a detailed explanation for *Criterion Line* and *Fitting Method* in the chapter of Geometry Study. The given default values come from experience and may not apply to other cases. The values of the criterion line 1 to 3 should be in ascendance order, or it will be ignored. *X0* and *Y0* on the dialogue are the coordinates of the point for which a migration versus time curve will be drawn. It is chosen by the user on the graph of fitted circles as shown in the third circle of Figure 8.5.

If default settings don't produce good fitting results, the user can first adjust criterion lines based on Figure 8.6. On the R/W versus channel length curve, a well behaved bend appears to be a plateau or a parabolic curve with its vertex close to the horizontal axis ($R/W=0$). The criterion lines should be adjusted in such a way that the segment between a criterion line and the horizontal axis is longer than the minimum bend length.

Further adjustments that can improve fitting quality include reducing spacing and increasing segment length. Smaller spacing leads to longer computation time, but the R/W versus channel length curve becomes smoother. Increasing segment length has the same effect and can effectively reduce sudden jumps on the curve. A very small bend could be better fitted as a part in a larger bend. Setting a reasonable minimum bend length coefficient will help the fitting of the larger bend. Perfect straight lines are rare in river channels, however, not all segments with a curvature is good to be fitted with a circle. Choosing a straightness limit can help eliminate the segments that have a reasonable curvature but look straight.

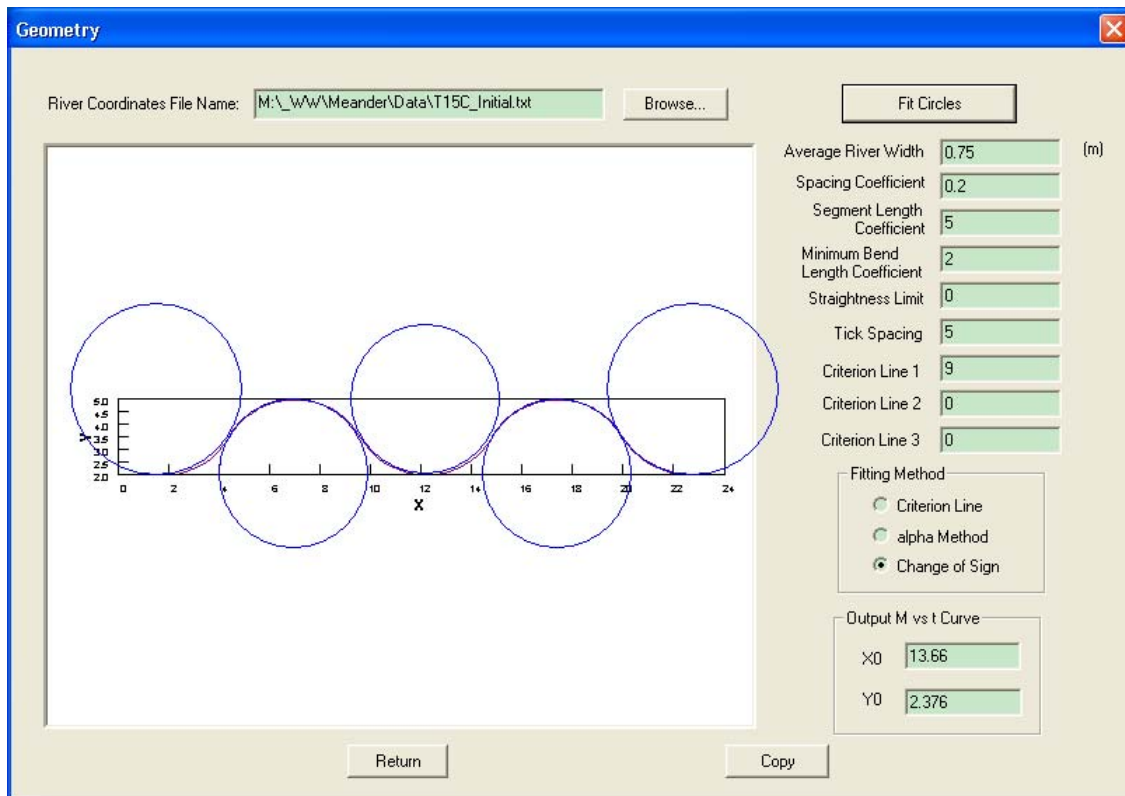


Figure 8.4 Geometry input

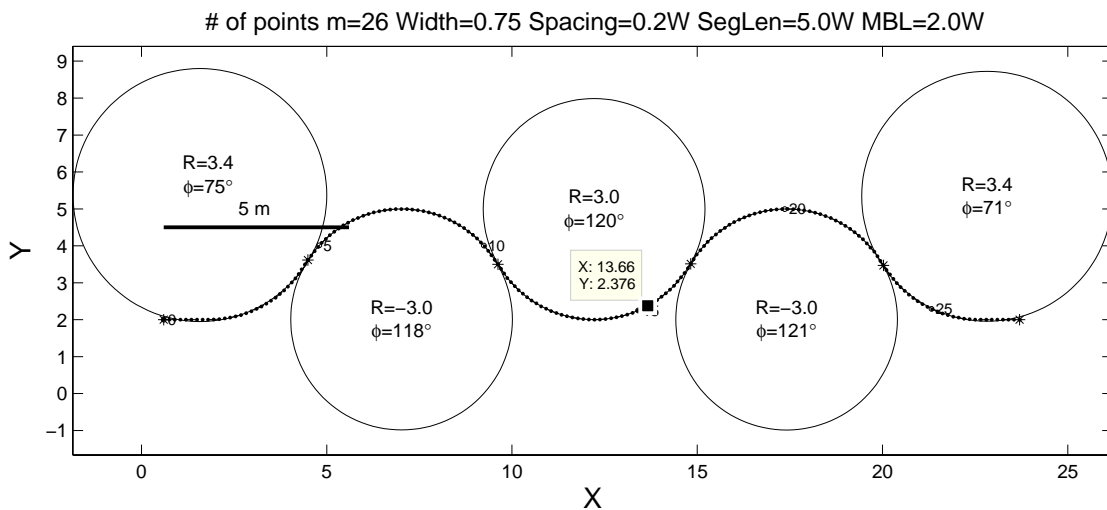


Figure 8.5 Original channel and fitted circles for the center line of flume test 15

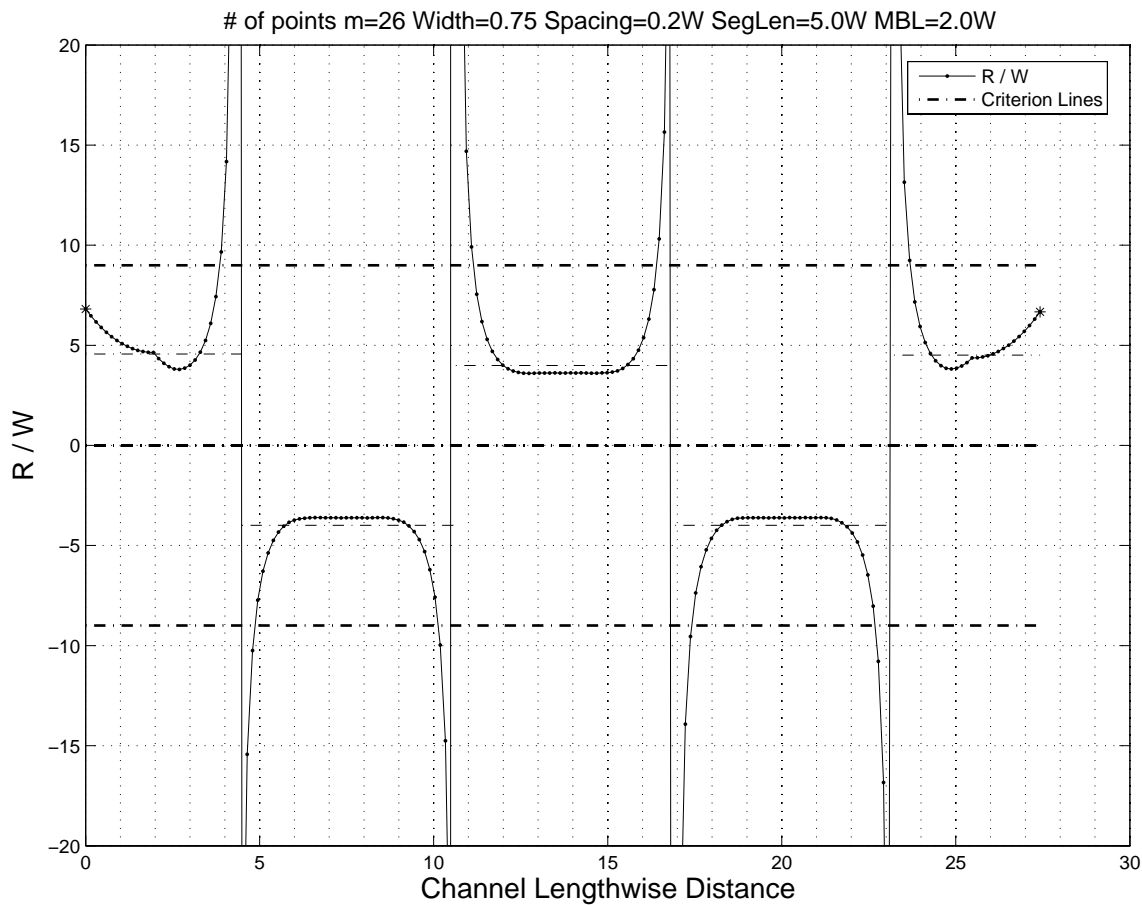


Figure 8.6 R/W versus channel lengthwise distance for the center line of flume test 15

Figure 8.7 is the soil data input interface. The first item is critical shear stress which corresponds to a scour rate of 1 mm/hour. The number of points on an EFA curve needs to be specified so that enough space will be allocated on the table. Conventionally, scour rate is in metric unit. When looking up a scour rate based on shear stress, linear interpolation is used. Data shown on the dialogue is for the sand used in the new Coastal Engineering Laboratory.

Soil Data

Erodibility Properties

Critical Shear Stress: 0.04 N / m²

Points on EFA Curve: 8

Point No	Shear Stress (N/m ²)	Scour Rate (mm/hr)
1	0.012	0.1
2	0.04	1
3	0.12	4
4	0.2	12
5	0.28	70
6	0.5	700
7	0.9	1200
8	2.1	13000

OK Cancel

Figure 8.7 Soil data input

The interface for entering flow conditions is shown in Figure 8.8. The flow can be in discharge or velocity. For doing a prediction, three types of analyses are available: constant flow, hydrograph and risk analysis. The hydrograph used here can be a file directly downloaded from USGS website www.usgs.gov or an edited single column data file. Each row in a hydrograph has the average flow during one time step which is specified in the dialog and is normally one day (24 hours). Risk analysis takes as input either a hydrograph or a 100-year and 500-year flood. Both choices can be used to calculate the risk level a river underwent during a certain period of time in history. It is assumed that the river will be at the same risk level during the predicted period in the future. The Velocity versus Water depth table is used to find the water depth corresponding to a velocity. If the input is discharge the Discharge versus Velocity table and Discharge versus Water depth table are required. All these tables are obtained from HEC-RAS simulation.

Water Data

Critical Froude Number: s/m^{1/3}

Time Step: hr

Input Hydrologic Data

Discharge vs. Time Velocity vs. Time

Constant Hydrograph Risk Analysis

Velocity: m/s

Time:

No. of Points on Curve

Velocity vs. Water Depth:

Velocity vs. Water Depth

Point No	Velocity (m/s)	Water Depth (m)
1	0	0.08
2	10	0.08

OK Cancel

Figure 8.8 Water data input

Figure 8.9 is a dialogue showing previously entered data in a tabular format for easy check. If a mistake is found, the user can go back to make corrections. Figure 8.10 provides an easier way to check the correctness of the data. Abnormal data can be easily identified on a graph. After the data is entered and checked, the computation process can be started. Upon hitting the “Run” button, the program starts the prediction process.

Input Tables

Choose one type

- Scour Rate -- Shear Stress
- Discharge -- Velocity
- Discharge -- Water Depth
- Velocity -- Water Depth

Point No	Shear Stress (N/m ²)	Scour Rate (mm/hr)
1	0.012	0.1
2	0.04	1
3	0.12	4
4	0.2	12
5	0.28	70
6	0.5	700
7	0.9	1200
8	2.1	13000

Return Save to File

Figure 8.9 Input tables

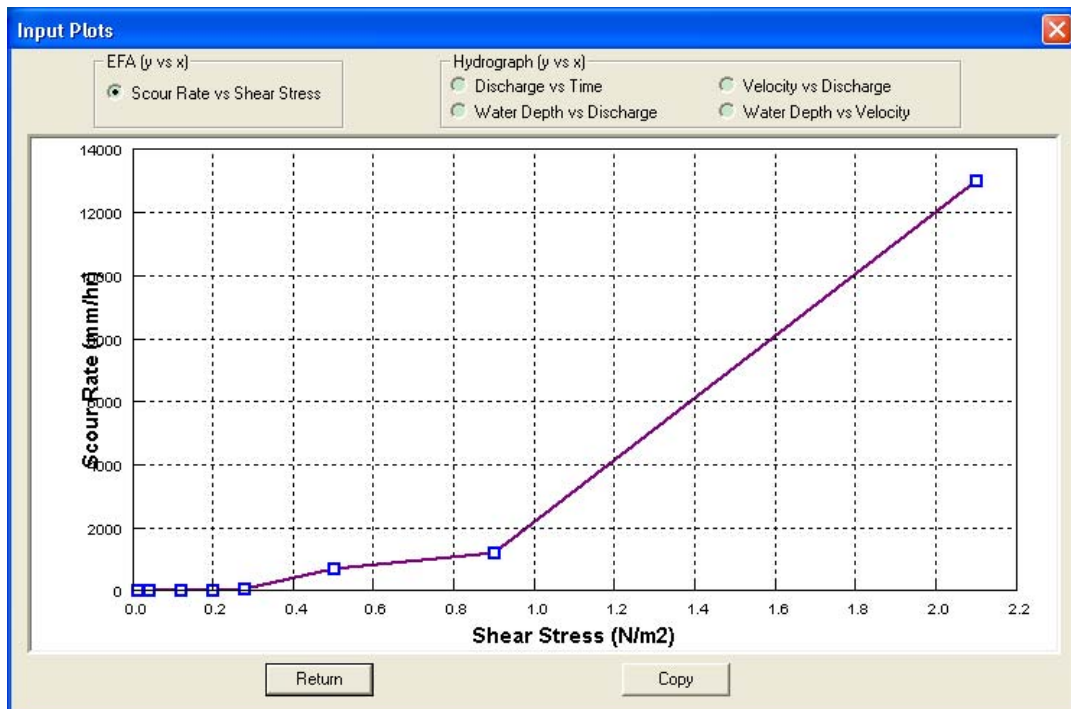


Figure 8.10 Input plots

Project Name: 15C

Point No	X0 (m)	Y0 (m)	Xt (m)	Yt (m)	Migration (m)
1	0.6	2.0	0.6	2.0	0.018
2	0.7	2.0	0.8	2.0	0.021
3	0.9	2.0	0.9	2.0	0.024
4	1.0	2.0	1.1	2.0	0.028
5	1.2	2.0	1.3	2.0	0.032
6	1.3	2.0	1.4	2.0	0.037
7	1.5	2.0	1.6	2.0	0.043
8	1.6	2.0	1.8	1.9	0.051
9	1.8	2.0	1.9	1.9	0.060
10	1.9	2.0	2.1	1.9	0.070
11	2.1	2.0	2.3	2.0	0.080
12	2.2	2.0	2.4	2.0	0.091
13	2.4	2.1	2.6	2.0	0.104
14	2.5	2.1	2.8	2.0	0.132
15	2.7	2.1	2.9	2.0	0.175

Return Save...

Figure 8.11 Output table

Figure 8.11 is the output table that shows the coordinates of the initial channel and the final channel. Accumulated migration distance of each point is on the last column.

The Output Plots dialog Figure 8.12 contains four buttons: *Center Line or One Bank*, *Both Banks*, *Risk Analysis*, and *M vs. t for one point*. All these functions were realized in Matlab. The first button is for showing the migrated channel of each step for the center line or a bank. The second button is to show initial banks, predicted final banks and measured final banks if data is available.

The trace of a migrating process is shown in Figure 8.13. The black dash line is the initial center line. Figure 8.14 shows a comparison between predicted and measured final banks. The *Center Line Method* mentioned in the title will be discussed in detail in the next section. *Risk Analysis* is being implemented by Namgyu Park in Matlab. It is integrated together under the same interface. For details please refer to Namgyu Park's Ph.D. dissertation which is likely to come out in 2006.

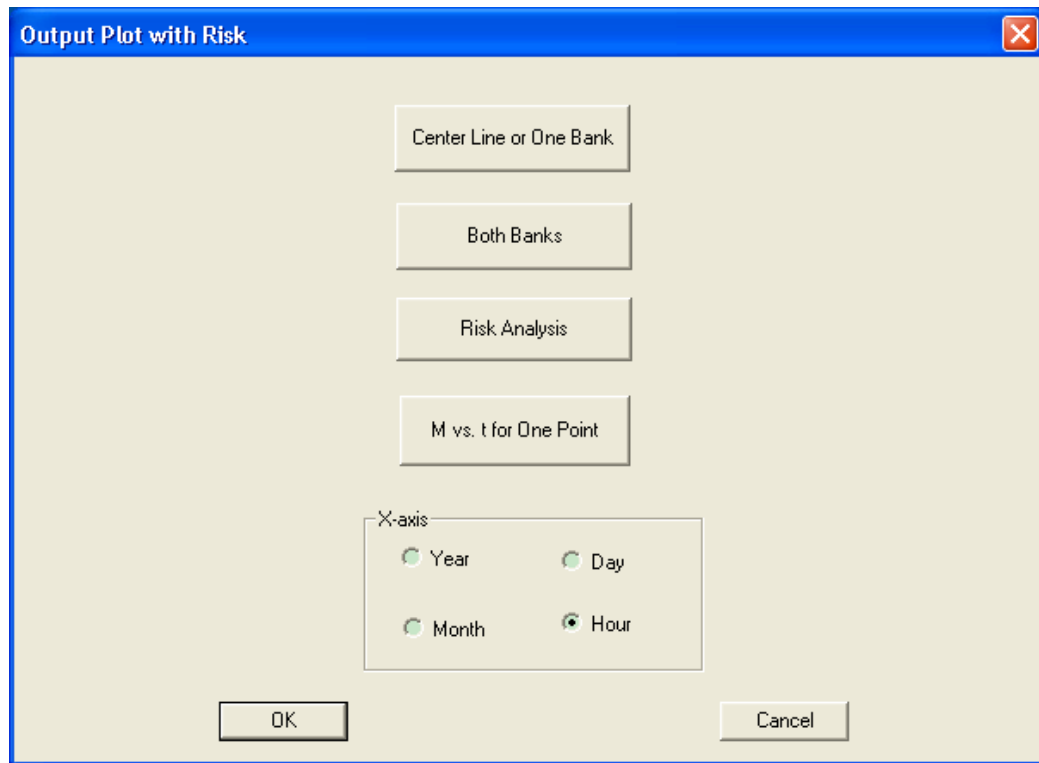


Figure 8.12 Output plots dialogue

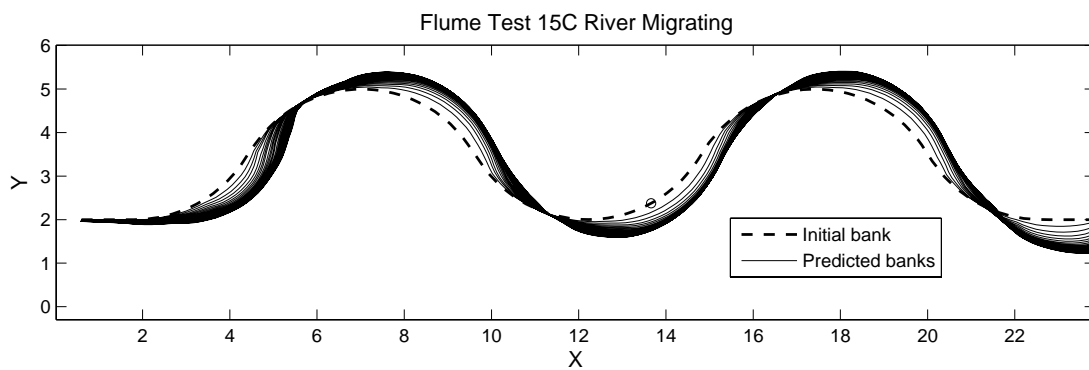


Figure 8.13 Trace of a migrating center line

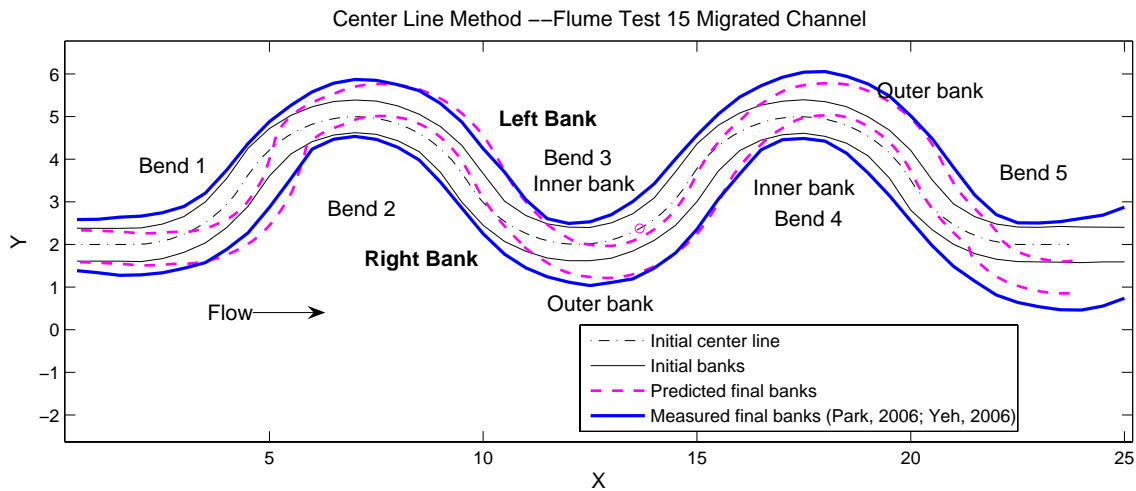


Figure 8.14 Predicted and measured banks by using Center Line Method

The migration process of an important point is of great interest. The coordinate of this point is entered on the Geometry Input dialogue. The user can obtain the coordinate of a point by clicking the “Data Cursor” button of Matlab figure and then clicking on the point of interest in any one of Figure 8.5, Figure 8.13, or Figure 8.14. On the Output Plot dialogue, the user first chooses the right unit for time by clicking appropriate radio button. Then hit the “M vs. t for One Point” button. A curve like Figure 8.15 will pop up. The location of the chosen point is indicated by a tiny circle in Figure 8.14. At this stage, if the Migration vs. Time curve is needed for another point, the calculation has to be done all over again.

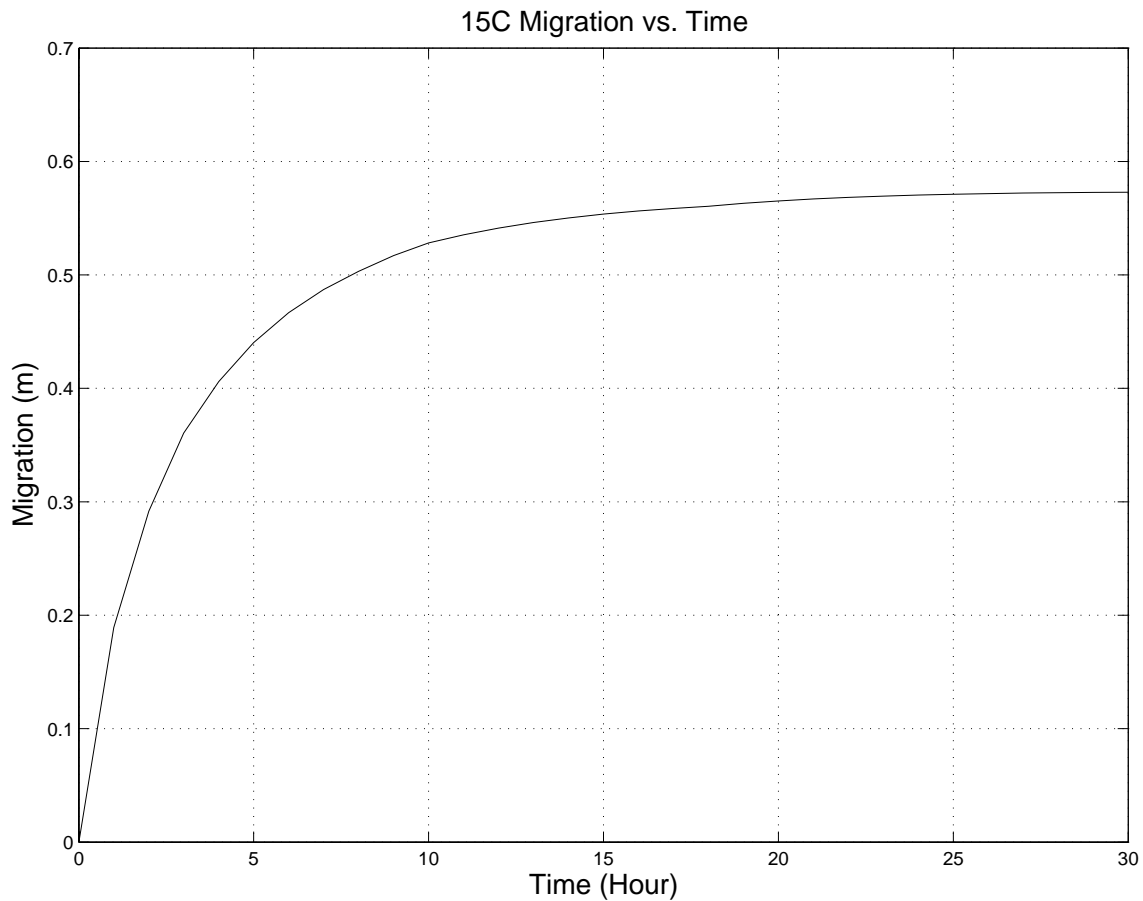


Figure 8.15 Migration versus time plot for a certain point

8.3 CENTER LINE METHOD VERSUS BANK METHOD

The initial channel of all flume tests has a well defined center line. The geometry of a center line is referred as that of the channel because of its constant radius to width ratio for all bends. It's natural to use the center line to represent the channel and to do the prediction of migration. However the model doesn't have any difficulty in predicting the migration of a single bank. Two approaches are thus available to the user. To distinguish one from the other, when the center line is used it is called Center Line Method. The other method is called Bank Method.

For Center Line Method river width has to be assumed as constant since the equations can't predict change of river width. Channel widening is a common phenomenon in flume test. The migration of a bend often goes beyond its two inflection points. The migration of a point on a center line is considered as the superposition of the migration vectors of two corresponding bank points. In calculation, a point on the center line can have two or more than two migration vectors. One is caused by the bend it is on. The others can be caused by the bend preceding it or after it. The advantage of this method is it can simulate Flume Test 11 (also called Stolpa Experiment by the team) in which a straight channel can migrate into a sinusoidal channel. The disadvantage of this method is that sometimes superposition can cause a false appearance of the state that the maximum migration is almost reached. The truth might be the difference of the magnitude of two large opposite migration vectors is small. If two consecutive bends are too close, the migrations at the two sides of the inflection point are of opposition direction which causes the transition part more and more distorted from a perfect arc. Thus the effectiveness of curve fitting goes down with the increase of migration. Another reason that makes this method unfavorable to practice engineers is that it is very hard to obtain the center line of a real river. Figure 8.13 shows the trace of predicted center lines coming from an acceptable application of Center Line Method. The predicted banks in Figure 8.14 were obtained by offsetting corresponding center line by a half channel width in both directions.

Bank Method only calculates migration of outer bends. The calculation for one bank is normally done on every other bend of the bank. The influence of an outer bend usually doesn't go beyond adjacent inner bends. Superposition is not likely to occur for this method. Figure 8.16 shows migrated banks at each time step. Figure 8.17 is a simplified version of Figure 8.16 where only banks of the last time step are kept and highlighted. For the left bank in Figure 8.16, the first, third and fifth bends are considered as active cause of migration. Migration of the second and fourth bend is passively caused by the outer bends next to them. For the right bank in Figure 8.16, the second, fourth bends are considered as active cause of migration. Migration of the first,

third and fifth bends is passively caused by the second or fourth bend. Migration of a channel is the result of migration of both banks and not of an imaginary center line. This method simulates migration phenomenon that really happens in experimental tests and in real rivers. The widening effect can be verified or predicted as shown in Figure 8.17. Without superposition to occur, the calculation can produce more stable results. The disadvantage of this method is that two times of computation time is needed to do a complete prediction.

The development of the M_{\max} equation is a multiple regression process. The migration of the outer bank and the geometry of the initial center line are used for developing the M_{\max} equation. Since the migration of an outer bank can be treated as that of the corresponding center line, the M_{\max} equation is the right one for Center Line Method. In preparing the geometry for a flume test, a bank is obtained by offsetting the center line by half the channel width. The left bank, center line and the right bank of a bend share the same center but have different radius. The difference from one to the next is half the channel width. Figure 8.18 shows the geometry of the left bank of flume test 15 which can be compared to Figure 8.5 for the center line of the same test. For ideal situation, the radius of outer banks is 3.5 and the radius of inner banks is 2.5. The fitting is not perfect but is very close. To apply the same M_{\max} equation for outer banks, the radius to width ratio should be subtracted by 0.5. The inner banks can be ignored which have no active migration.

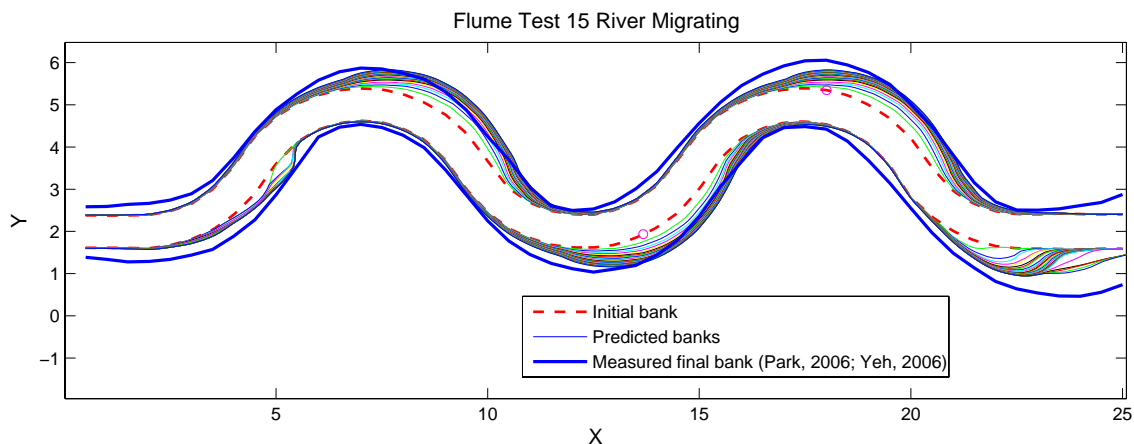


Figure 8.16 Trace of migrating banks

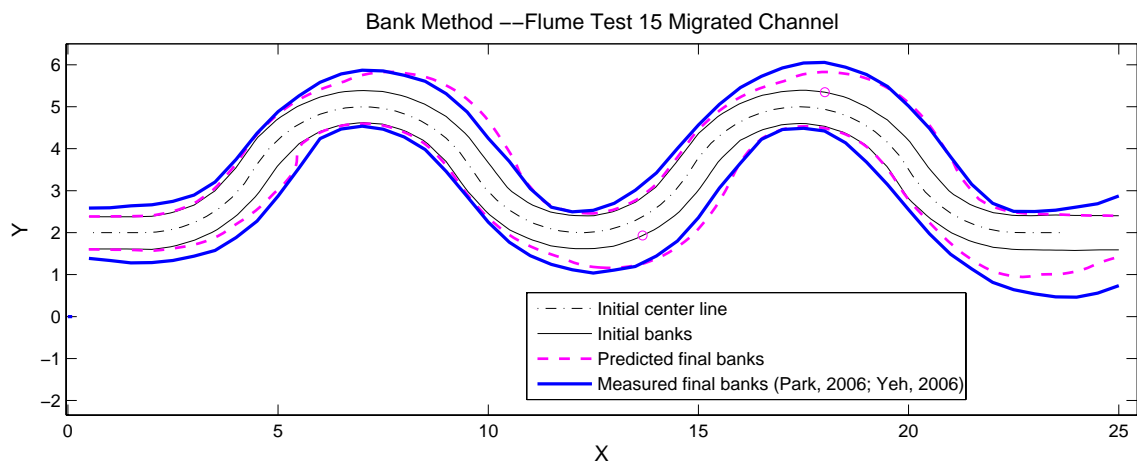


Figure 8.17 Predicted and measured banks by using Bank Method

8.4 MIXED LANGUAGE PROGRAMMING—C++ AND MATLAB

This program could have been written completely in Matlab. Then the Graphic User Interface (GUI) code from SRICOS-EFA could not be used and a new GUI had to be written in Matlab. It could also have been written completely in C++. Then the code for Geometry Study which had been written in Matlab could not be used. The advantages of Matlab such as strong functionality in graphic output, debugging, and

numerous built-in numerical recipes, etc. wouldn't be utilized. For the MEANDER program, the GUI is in C++ with the application Microsoft Foundation Class (MFC). The implementation of the Hyperbolic Model is in regular C++. The implementation of Geometry Study, Risk Analysis, and graphic output are in Matlab. The program development environment is Visual C++ 6.0 and Matlab 7.0.4 (service pack 2).

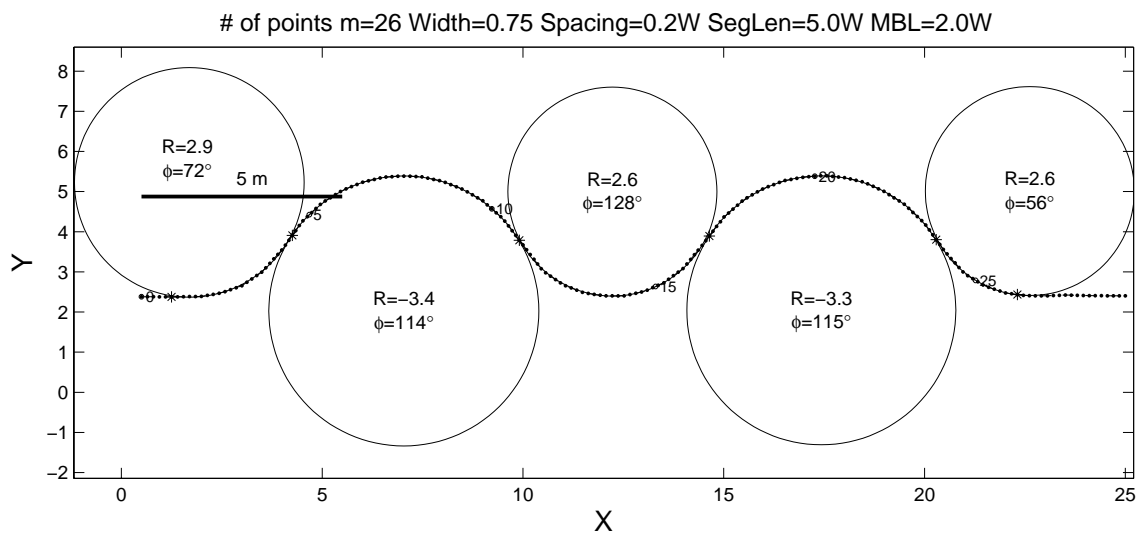


Figure 8.18 Original channel and fitted circles for the left bank of flume test 15

The integration of Matlab with C++ is not like routine C++ programming or Matlab programming. A good understanding of both languages is extremely helpful. Lots of time was also spent on solving abnormal problems due to incomplete documentation of Matlab on this issue. Matlab Compiler is an independent module which differs from version to version. The functionality and documentation are also progressing. It's worthwhile to give an explanation from a user's point of view so that later programmers can compile complicated Matlab programs more smoothly.

Matlab is a type of interpreting language. Unlike compiling languages like C, C++ or FORTRAN, it is executed line by line on running and has a lower efficiency.

Matlab file has an extension of “m”, also called “M-file”. M-files cannot run without Matlab environment. Matlab Compiler takes M-files as input and generates redistributable, stand-alone applications or software components. The resulting applications can be independent of Matlab environment. The latest version (Version 4, going with Matlab 7.0.4) of Matlab Compiler can support all functionalities of Matlab. The Matlab Compiler can generate these kinds of applications or components:

1. Stand-alone applications;
2. C and C++ shared libraries (dynamically linked libraries, or DLLs, on Microsoft Windows);
3. Excel add-ins and Com objects.

C++ shared libraries are generated for the MEANDER program. Besides generating the DLL files, The Matlab Compiler also produces an interface function which passes on the input data for the Matlab program and returns the results. A special data type class `mwArray` was created to do this job. The programmer first converts C++ variables into `mwArray` type and then converts the results in `mwArray` type back into C++ type. This process is often written as an user-defined C++ interface function. This function can be called and shared as a regular C++ function. The following is an example from the MEANDER program which calls the compiled Matlab program to fit circles. The sentences following double slash “//” are comments.

```
// Excerpted from GeoRndUI.cpp.
// Interface between Visual C++ & the DLL compiled from Matlab.
// Count the times this function is called.
// For the 1st time a special initialization function should be called.
static int init=0; // local static variable.
double ArcIdxDbl[nMaxArc][2]; // Receive data from Matlab
int GeoInterface(double (*xy_in)[2], int* nNum, double *Arg, \
                 double (*xyRF)[4], int (*ArcIdx)[2])
// xy_in, coordinates of the channel; nNum[0], number of points
// Arg, an array containing parameters needed for fitting circles.
// xyRF, coordinates, radii, bend angle(fe) of the fitted arcs
// ArcIdx, boundary indices of the bends
// Avoid matrix transpose here. Do it inside Matlab code.
{
    if(init==0) // The 1st time to call this function.
    {
        if(!mclInitializeApplication(NULL,0) || !libGeoRndInitialize())
            return -1;
    }
    else
        init++;
}
```

```

int i,j,nPt=nNum[0],nArc,errCode;
double nNumDbl[3];
CString erMsg,erPlace;
// Create a 2-row nPt-column Matlab array. Default: mxREAL
mwArray mw_xy(2,nPt,mxDOUBLE_CLASS,mxREAL);
// Since the size is fixed, it can be static.
static mwArray mw_arg(nArgSize,1,mxDOUBLE_CLASS,mxREAL);
// Corresponds to the names with prefix "mw_".
mwArray mw_xyRF,mw_ArcIdx,mw_nNum;
// SetData treats the input C array as the ONE-D array it is in
physical memory.
// Then assigns the 1-D array to mwArray element by element
according to the defined dimensions and Matlab convention.
try
{
// Assign values of C++ variables to mwArray type variables.
mw_xy.SetData((double*)xy_in,nPt*2);
mw_arg.SetData(Arg,nArgSize);
}
catch (const mwException& e)
{
erMsg=e.what();
erPlace="\nError in mwArray.SetData(...).";
erMsg+=erPlace;
throw erMsg;
return -1;
}
try
{
// 2 input arguments: mw_xy,mw_arg;
// 4 output arguments: mw_xyRF,mw_ArcIdx,mw_nNum,mw_xy.
AutoFit_Run_InVC(4,mw_xyRF,mw_ArcIdx,mw_nNum,mw_xy,mw_xy,mw_arg);
}
catch (const mwException& e)
{
erMsg=e.what();
erPlace="\nError in autofit_run_invc(...).";
erMsg+=erPlace;
throw erMsg;
return -1;
}

// Transfer values from mwArray type to C/C++ type.
mw_nNum.GetData((double*)nNumDbl,4);
for(i=0;i<4;i++)
nNum[i]=(int)nNumDbl[i];
nPt=nNum[0];
nArc=nNum[1];
errCode=nNum[3];

// Refer to CGeometry::OnFitCircles(...)
if(errCode!=-2 && errCode<=0)
return -1;
// Arg[10]: if 0, don't EvenlyDivide the curve, if otherwise,
divide it.
if(fabs(Arg[10])>1e-3)
{
// nPt=mw_nPt(1,1);
mw_xy.GetData((double*)xy_in,nPt*2);
}

```

```

//      nArc=mw_nNums(2,1); // or (1,2), neither one works with
nNums=zeros(2,1) stated in the m code.

mw_ArcIdx.GetData((double*)ArcIdxDb1,nArc*2);
mw_xyRF.GetData((double*)xyRF,nArc*4);

for(i=0;i<nArc;i++)
    for(j=0;j<2;j++) // "-1" make it from 1-based to 0-based.
        ArcIdx[i][j]=(int)ArcIdxDb1[i][j]-1;
return 0;
}

```

Conversion of data from one type to the other is the major task of this function. For a scalar or a vector, the conversion is straightforward. For a 2-D or 3-D array, caution should be taken because C/C++ and Matlab store multiple-dimensional array differently. In physical memory a multiple-dimensional array is stored in a column of continuous memory cells. There are two methods to map the 1-D physical index to the multiple-dimensional array index at computer language level. The first method is to store data from row to row, as what C/C++ does. The other is to store data from column to column, which is used by Matlab. Class `mwArray` has a method `SetData` to read data from a C/C++ variable and pass it to a `mwArray` variable. The data is read from a single column physical memory. Then the Matlab mapping method is used to assign it to an array of Matlab type. A 3-D array of dimensions $M \times N \times L$ in C/C++ will be converted into a 3-D array of dimensions $L \times N \times M$ in Matlab. However their storage in the physical memory is of the same order. The `GetData` method of class `mwArray` also uses the Matlab mapping method to assign data from a Matlab array to a physical memory which is for a C/C++ variable.

It is helpful to list a step by step procedure explaining the process of integration. Assume Matlab 7 and Visual C++ 6 are installed. The main MFC project is in folder “Z:_WW\MEANDER\”. The subproject in VC++ 6 for compiled Matlab program is called `GeoRndPrj` which is in folder “Z:_WW\MEANDER\GeoRndPrj\”.

If it's the first time to compile Matlab program, run the command `mbuild -setup` in Matlab command window first and choose appropriate C compiler. In here VC++ 6 is used. Then follow the steps listed below.

1. Change Matlab work directory to the folder holding the m files. If a file named "mccpath" exists, delete it.

2. Run this command: `mcc -W cpplib:libGeoRnd -T link:lib AutoFit_Run_InVC.m mLognrnd.m SolveCDF.m -v -d "M:_WW\Meander\GeoRndPrj"`. Words in italic characters are names decided by the user. “-v” means verbose which outputs intermediate results in detail. “-d” switch sends output files to the folder after it.
3. Add a static library project to MFC application (MEANDER), GeoRndPrj in here. If “-d” option is not used in step 2, copy generated files libGeoRnd.* to folder “.\MEANDER\GeoRndPrj”.
4. Copy/Move libGeoRnd.ctf and libGeoRnd.dll to folder “.\MEANDER\”.
5. Add two files to the subproject “GeoRndPrj”. They are "libGeoRnd.lib" and "<Matlab Root>\extern\lib\win32\microsoft\msvc60\mclmcrtr.lib". Build subproject “GeoRndPrj” and GeorndPrj.lib will be generated.
6. Add GeoRndPrj.lib to MEANDER project. In VC++ 6 environment, add "<Matlab Root>\extern\include\" to INCLUDE folder and add "<Matlab Root>\extern\WIN32\MICROSOFT\MSVC60" to LIBRARY folder.
7. Build MEANDER project. The linking process will be done automatically.

8.5 AN OVERVIEW OF THE IMPLEMENTATION OF THE PROGRAM

All the methods have been introduced in previous chapters. The implementation is a description of the same idea in computer languages. A flow chart is an overview of the computer code. About 14,000 lines of code has been written and checked line by line over a long period of time. It is not realistic to introduce the code by going through all the lines. Flow charts help introduce important modules, steps and their functionalities in natural language. Figure 8.19 is the flow chart of the entire MEANDER program.

The MEANDER Program

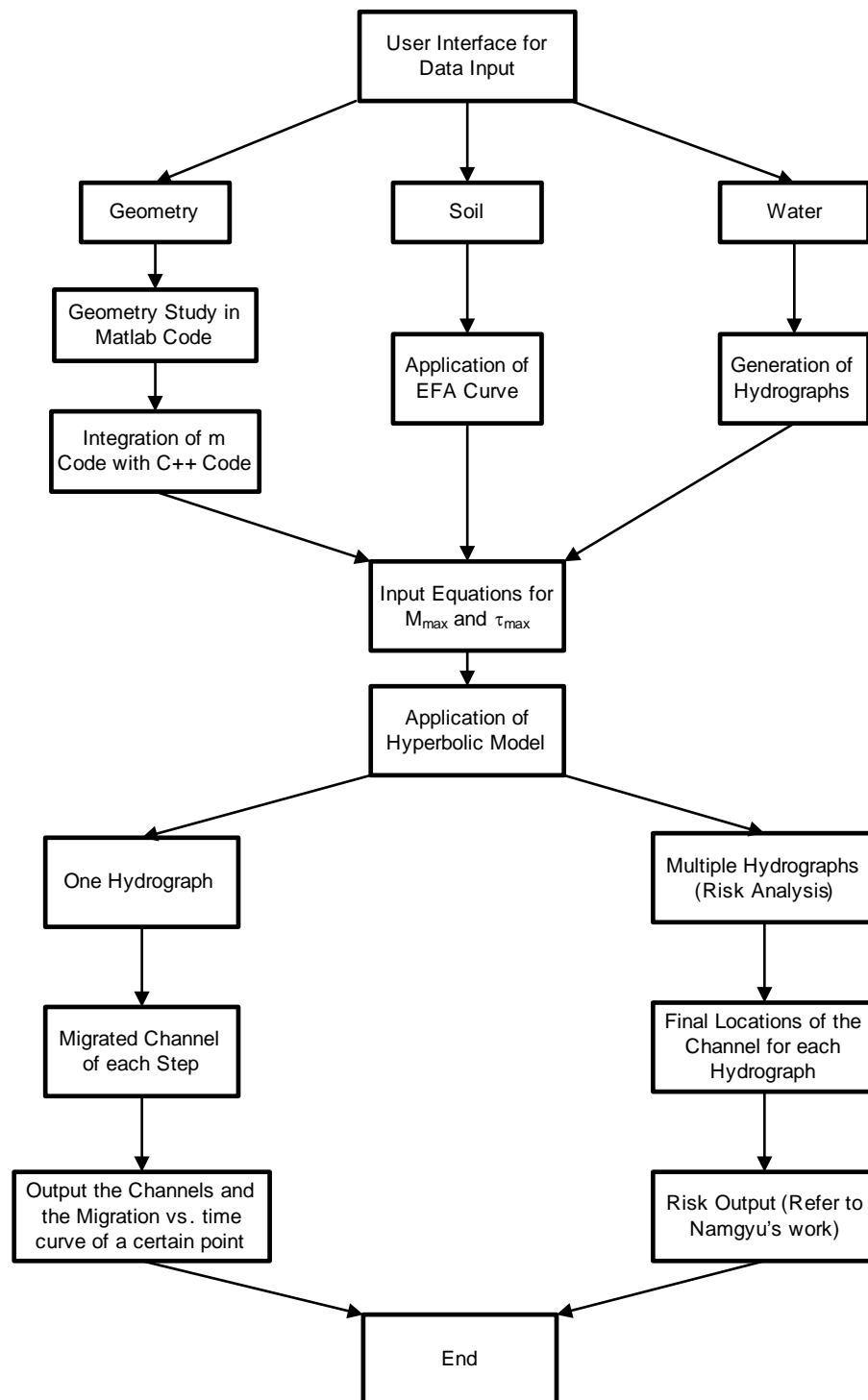


Figure 8.19 Flow chart of the MEANDER program

Figure 8.19 gives a good idea about what the project is about and what the program can do. To further understand how a specific task is implemented, a more detailed flow chart is needed. Figure 8.20 is the flow chart of Geometry Study.

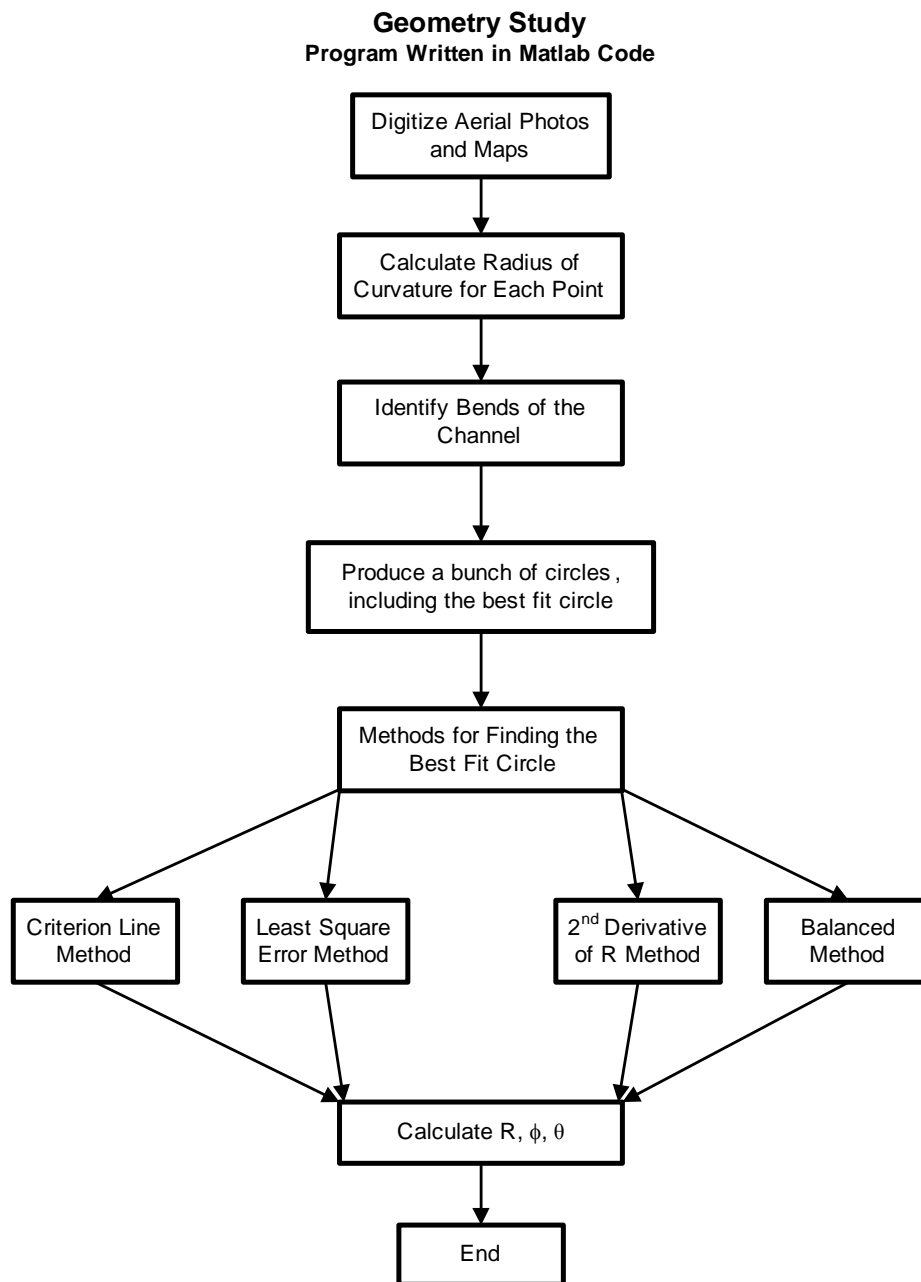


Figure 8.20 Flow chart of Geometry Study

Figure 8.21 also explains the task of Geometry Study but from a different perspective of view. The task is outlined in a problem-solving manner. The major difficulties of certain steps are briefly described.

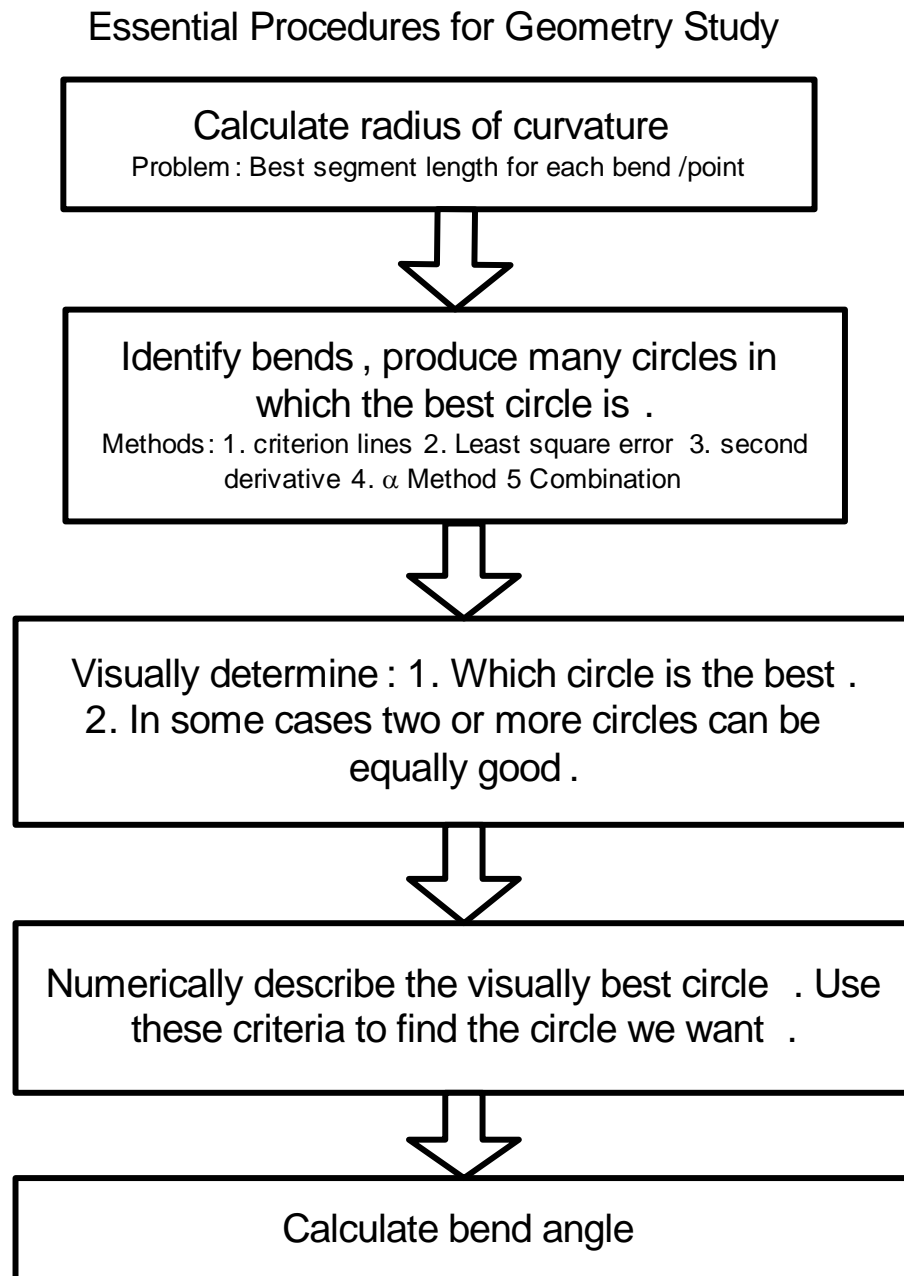


Figure 8.21 Essential procedures for Geometry Study

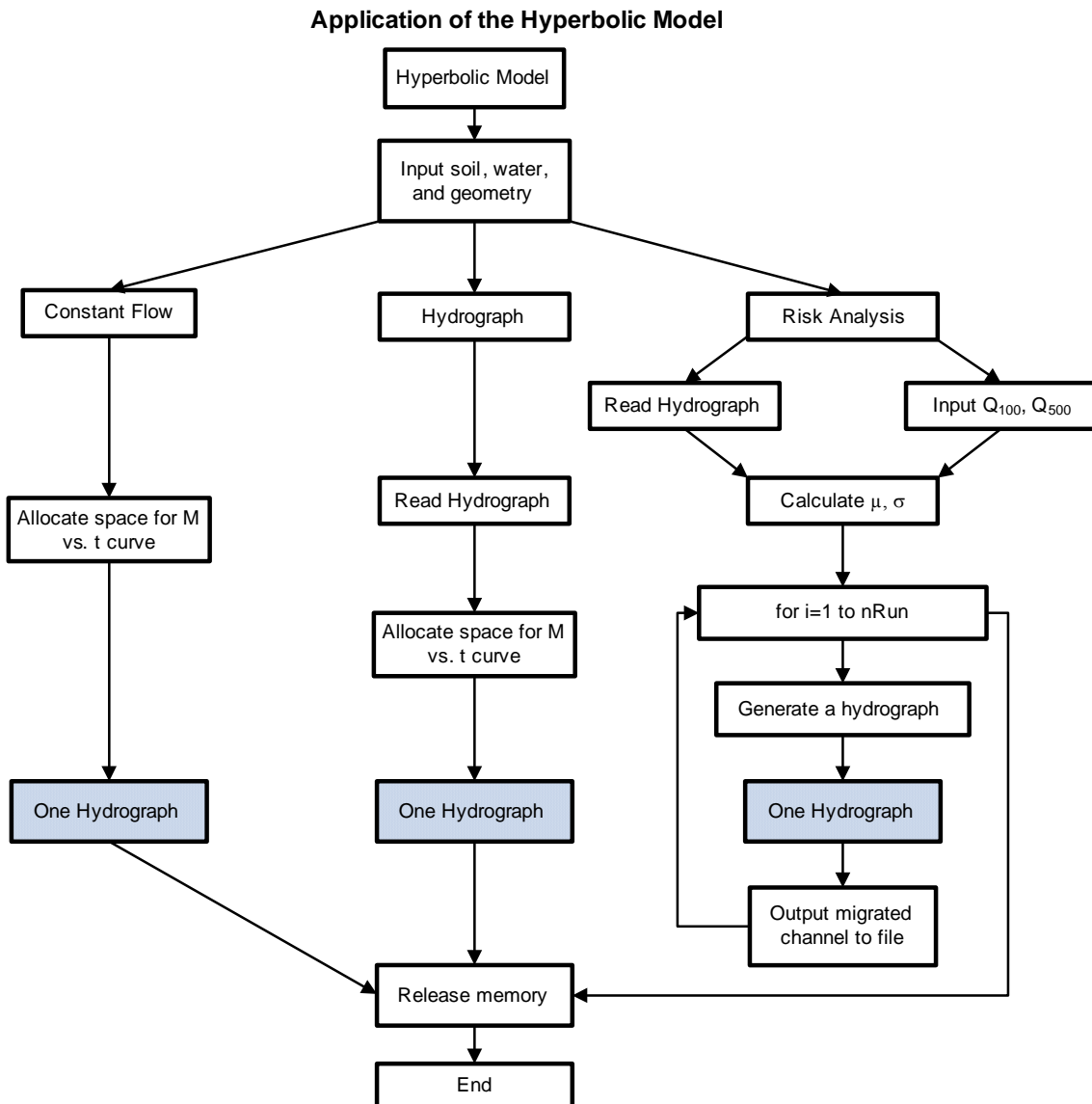
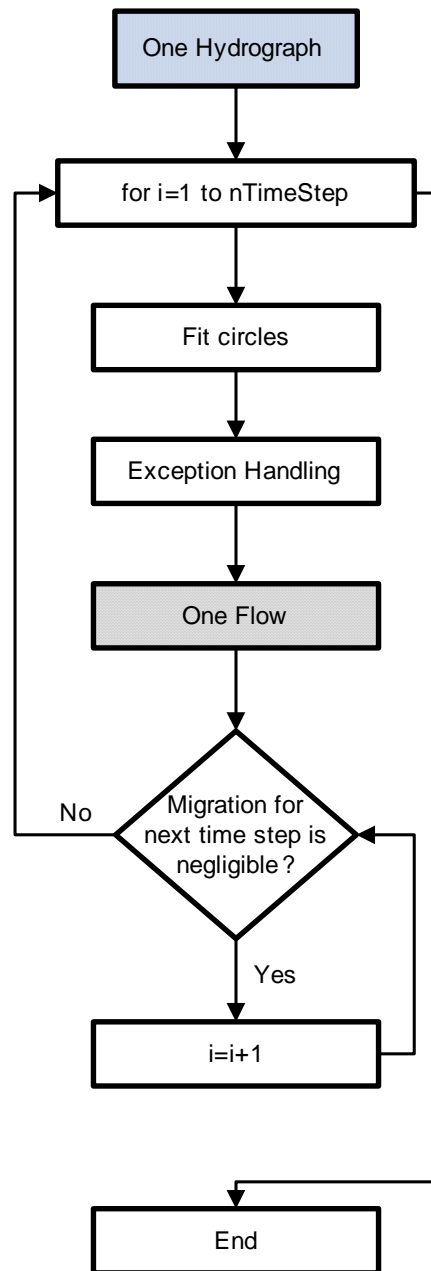


Figure 8.22 Flow chart of the implementation of the Hyperbolic Model

The code that applies the Hyperbolic Model is based on Figure 8.22. The shaded item “One Hydrograph” is a sub-function whose flow chart is Figure 8.23. Function OneHydrograph () calculates the migration caused by each time step of a hydrograph and comes up with an accumulated distance.

Implementation of Function OneHydrograph()**Figure 8.23 Flow chart of function OneHydrograph**

The shaded item "One Flow" in Figure 8.23 is a sub-function of function OneHydrograph().

“One Flow” calculates the migration of the whole channel caused by the flow of one time step. The flow chart of function OneFlow() is shown in Figure 8.24.

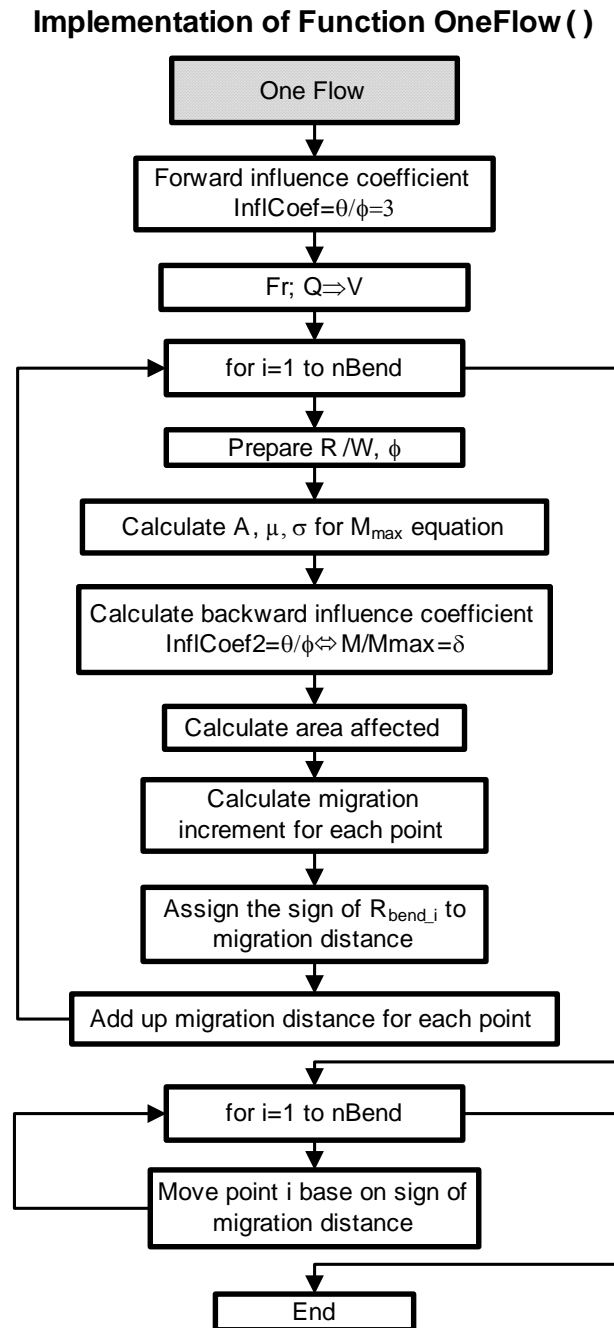


Figure 8.24 Flow chart of function OneFlow

CHAPTER IX

VERIFICATION AND PARAMETRIC STUDY

Although a tremendous amount of time has been spent on developing the program, it is useless unless it performs as it was expected to. Only when the program is a faithful implementation of the model can the results from the program be used to evaluate the model. Verification of flume tests is the first step to prove the soundness of the program and the model. The verification process is also a process of testing the program and adjusting the model so that a better match of prediction with measured data can be achieved. Due to the lack of soil data from the field, verification on real rivers was done only for the Brazos River at State Highway 105 (Texas) with hypothetical soil properties. A more detailed verification on real cases will be done later on. The flume test on clay is underway on the second floor of the old Hydro Lab. The model incorporating clay test results will better reflect the behavior of real rivers. At this stage verification is done for flume tests on sand.

A parametric study is a comprehensive test of the performance of the program and the model. Important parameters are varied and the effects are observed. The relations between migration and some influential factors should appeal to common sense. If not, something may be wrong and further research needs to be done. If it works well, a parametric study can predict migration behavior for a river with any combination of major factors, for which flume test is not capable of.

9.1 VERIFICATION OF FLUME TESTS

Eighteen flume tests have been done in the new Coastal Engineering Laboratory as shown in Table 9.1. Test 1 can be considered as a preliminary test from which some good parameters were obtained for later tests. Test 5 was a repeatability test of Test 2. Some tests have obvious straightening effects which add difficulty to data reduction and verification. In the process of developing M_{\max} equation, the data of 10 flume tests were used due to good correlation and easiness in data reduction. For each test used, migration of the second bend was analyzed. Initial geometry and initial average velocity were used

and treated as constant. A detailed explanation of the process will be provided in Po-Hung Yeh and Namgyu Park's doctoral dissertations.

A comparison between the predicted and measured banks is done for most of the flume tests. Center Line Method and Bank Method produce close results when both of them are applicable. There are some cases for which Bank Method works well but Center Line Method cannot make reasonable predictions due to large migration. For the examples listed in this chapter, the Center Line Method is used only for Test 11. The Bank Method is used for the other cases.

**Table 9.1 Flume tests done in the Coastal Engineering Lab
(Park, 2006; Yeh, 2006)**

TEST MATRIX IN THE NEW LAB (Sand)					
				- revised in July 21, 2004	
R/W	Bend Angle	Flowrate (GPM)	Test No	Remark	Used for M_{max}
4	120°	226	1		No
	180°		6		Yes
	220°		7		Yes
8	65°	226	4		Yes
6			18	Extra Test	Yes
4			2	Reference	No
4			5	Repeatability	Yes
3			16	Extra Test	Yes
2			3	Straightening	Yes
4			14	20cm/s	Yes
2	120° ²	185	13	20cm/s	No
1			12	Straightening	No
4			15		Yes
4	120° ⁴	260	17	redo Test 15	Yes
2		178	8	Straightening	No
n/a	n/a	226	9	Brazos River	No
n/a	n/a	226	10	Brazos River	No
n/a	n/a	257	11	Stolpa Experiment	No

The definition of a bend in a channel is shown in Figure 9.1. In flume tests, two adjacent bends are tangent to each other at the beginning of a test. An inflection point

separates two bends. Bend 1 is the first bend from the upstream. Flow direction is from left to right if not otherwise indicated. Left Bank is defined as the bank at the left hand side when a person faces downstream. The one at the right hand side is defined as the Right Bank.

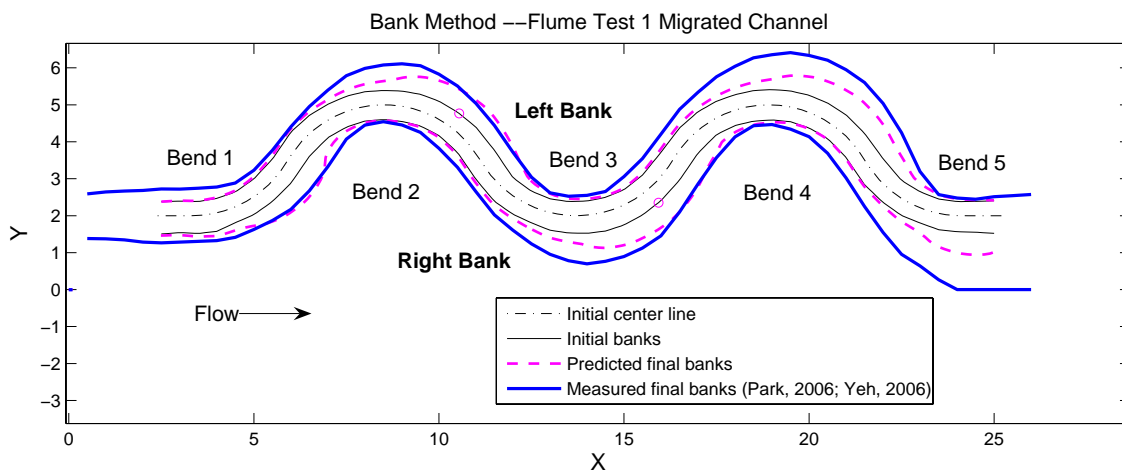


Figure 9.1 Verification of Flume Test 1 $R/W=4$, $\phi=120^\circ$, $v=0.25\text{m/s}$

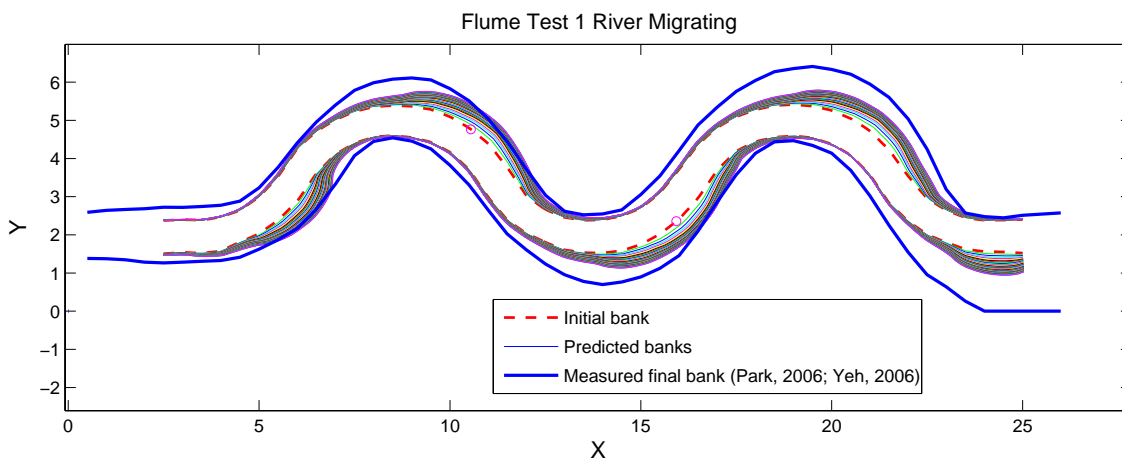


Figure 9.2 Verification of Flume Test 1 with migrated banks at each time step

There are two ways to present prediction results. One is shown in Figure 9.1 which shows the initial banks, initial center line and the predicted banks after the duration of the test. Final measured banks are superimposed for comparison. Besides what Figure 9.1 shows, Figure 9.2 includes migrated bank at each time step. It provides more information about the predicted migration process. But sometimes the process is only represented by a black block. Figure 9.1 is more focused on comparing the predicted channel with the measured one. This type of presentation is used in this chapter for clarity. All the measured banks come from flume tests conducted by Namgyu Park and Po-Hung Yeh.

It can be seen from Figure 9.1, the maximum migration of Bend 2, 3, and 4 is not well matched. The match is good around inflection points. The measured migration tends to increase from one bend to the next along the channel, while the predicted migration is periodic. So the difference between measured and predicted migration also increases along the channel. Compared to Flume Test 15 the initial velocity of Flume Test 1 is 0.02 m/s smaller, but the measured migration is much larger. Since this is the first test in the new Coastal Engineering Lab, testers' lack of experience with the new environment may have accounted for the discrepancy. The data of this test was not used for developing the M_{\max} Equation.

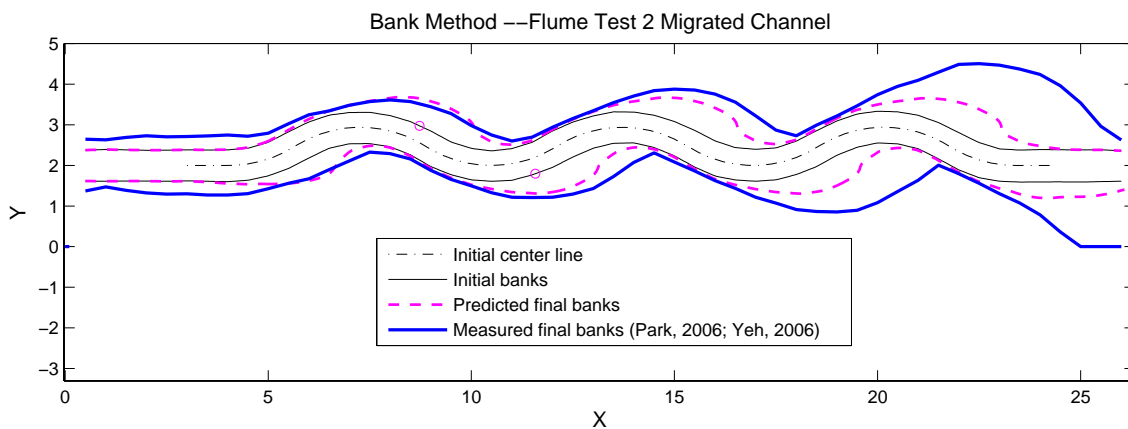


Figure 9.3 Verification of Flume Test 2 $R/W=4$, $\phi=65^\circ$, $v=0.25$ m/s

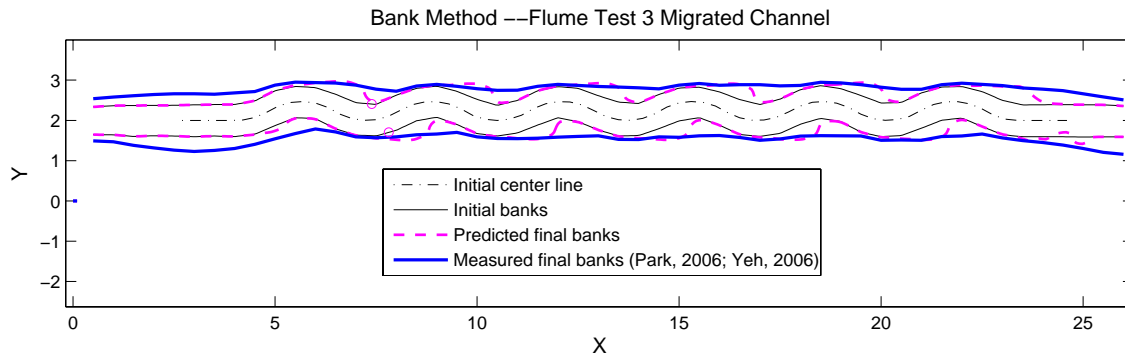


Figure 9.4 Verification of Flume Test 3 $R/W=2$, $\phi=65^\circ$, $v=0.25$ m/s

The phenomenon that migration increases along the bends is most obvious in Flume Test 2 as shown in Figure 9.3. Secondary flow makes significant contribution to scour and sediment transport. The development and increase of secondary flow along the channel can be a partial cause for this phenomenon. The flow condition at the exit is far different from any that of any part in the middle. The change in material from soil to wood at exit contributes to this problem to some degree. Free fall of water at the exit makes the velocity much larger than normal condition. Faster erosion at a certain spot leads to irregular bank shape which in turn leads to turbulence. Turbulence can cause extreme erosion. This kind of situation rarely happens in real rivers and is not accounted for in the prediction model.

Figure 9.4 shows the verification for Flume Test 3. For rivers with small R/W the channel tends to straighten up. In this case, the original curvy banks became almost straight at the end of the test. The model heavily relies on curve fitting which cannot be effectively done on curves with small or zero curvature. A fair match is still achieved for this case. If radius to width ratio decreases to one, the performance of the program will suffer. The good news is that geometry like this is rarely seen in real rivers.

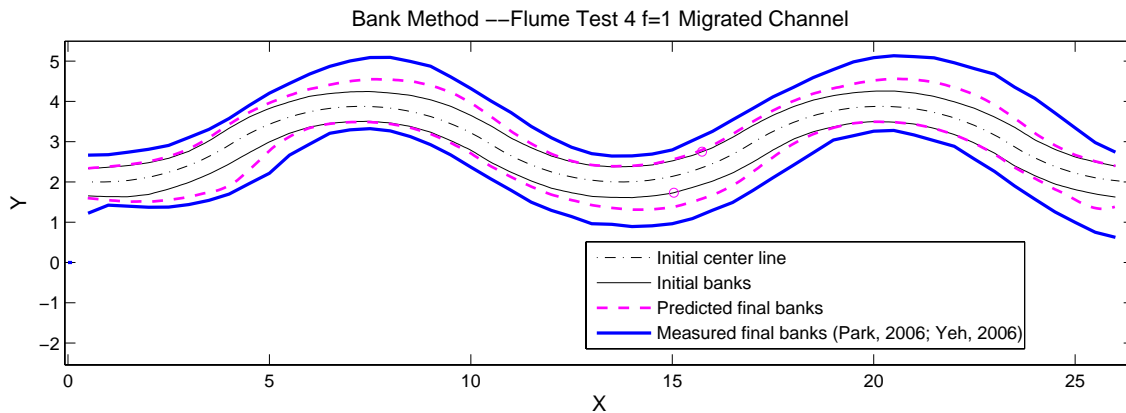


Figure 9.5 Verification of Flume Test 4 with small τ_{\max} $R/W=8$, $\phi=65^\circ$, $v=0.24$ m/s

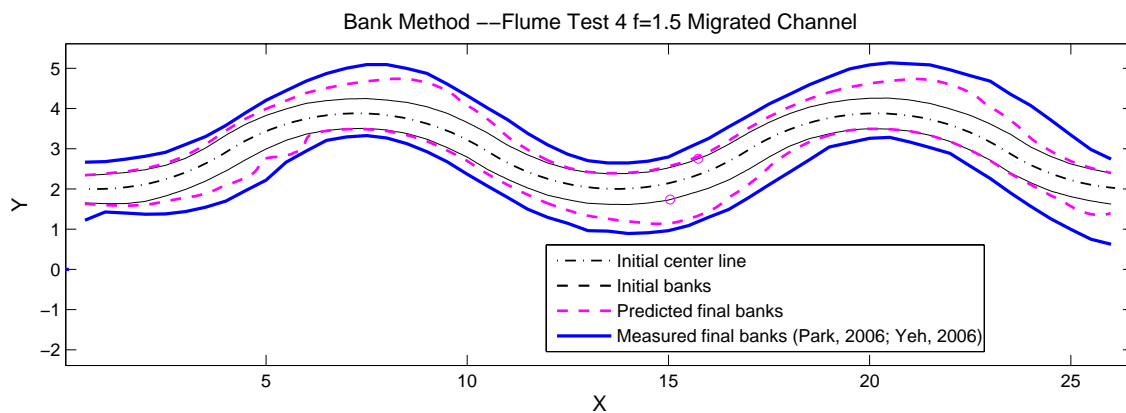


Figure 9.6 Verification of Flume Test 4 with large τ_{\max} $R/W=8$, $\phi=65^\circ$, $v=0.24$ m/s

In developing the τ_{\max} equation, magnitude is scaled up by a factor of $c_1=8$ to account for roughness of the bank. This, however, does not help for two flume test cases. Figure 9.5 indicates a bad match for Flume Test 4, compared to Figure 9.6. Figure 9.7 shows another bad match for the last bend of Flume Test 9. A large radius to width ratio is a common characteristic of these two cases. In Flume Test 4, $R/W=8$. In Flume Test 9 the last bend has a radius to width ratio of 9.1 as shown in Figure 9.9. It suggests that the scale factor should be a function of radius to width ratio. The following equation is proposed:

$$c_2 = 0.25(R / W) - 0.5 \dots\dots\dots (R / W > 6)$$

$$c_2 = 1 \dots\dots\dots (R / W \leq 6)$$

Figure 9.6 and Figure 9.8 show the improvement after applying this new factor.

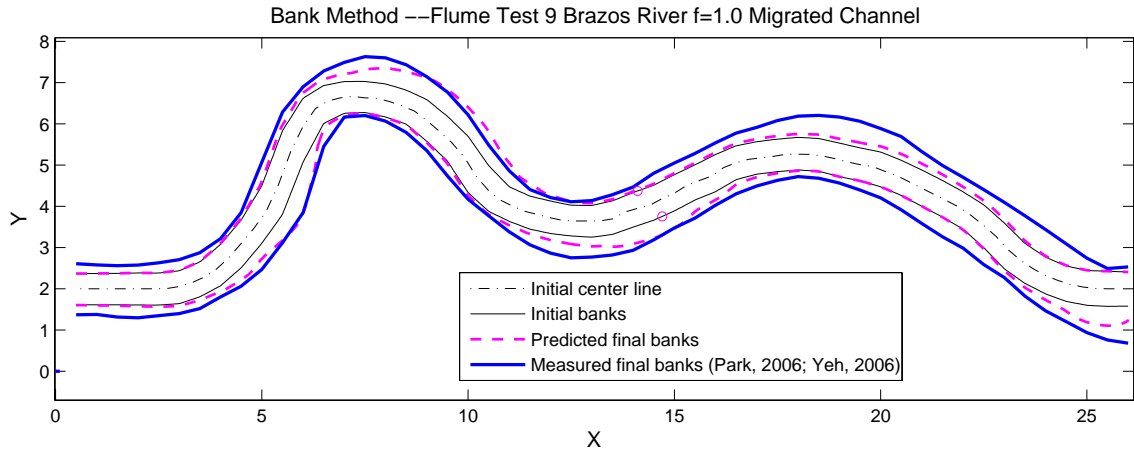


Figure 9.7 Verification of Flume Test 9 with small τ_{max} , $v=0.25$ m/s

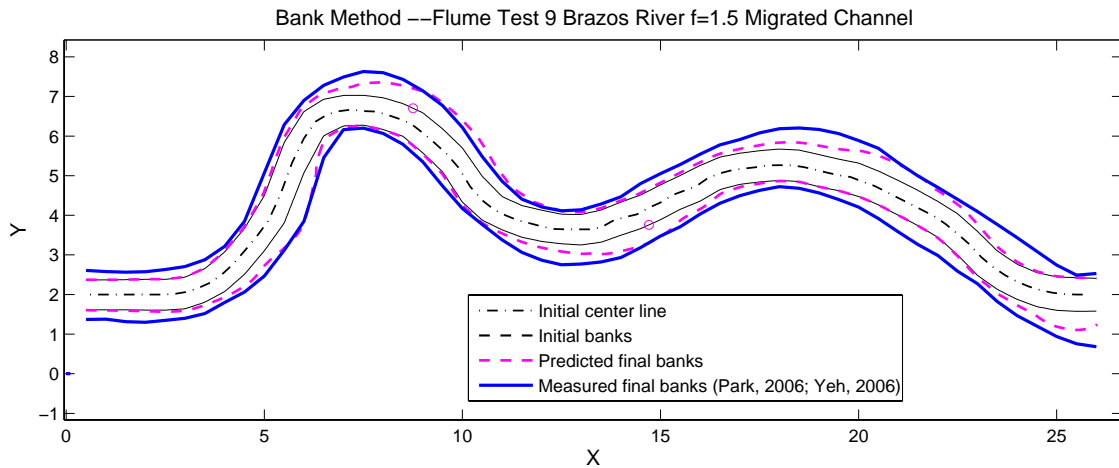


Figure 9.8 Verification of Flume Test 9 with large τ_{max} , $v=0.25$ m/s

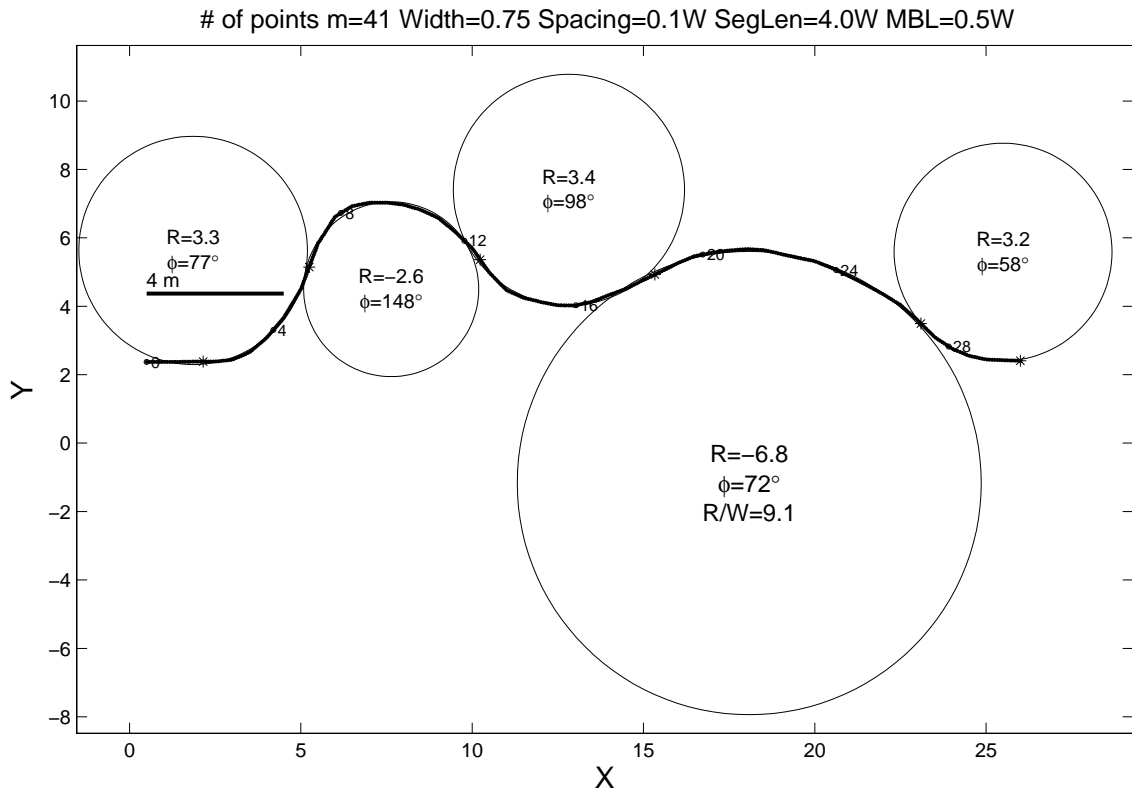


Figure 9.9 Flume Test 9, fitted circles of the left bank

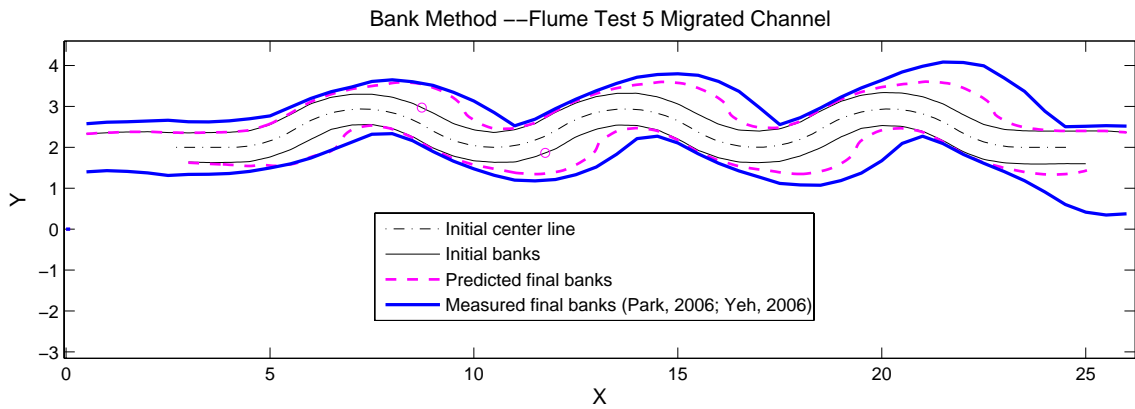


Figure 9.10 Verification of Flume Test 5 $R/W=4$, $\phi=65^\circ$, $v=0.27$ m/s

Figure 9.10 is the verification of Flume Test 5 which is a repeatability test of Flume Test 2. Due to the experience accumulated from previous tests, the flow condition at the exit was better controlled. As a result, migration at the exit became smaller than before.

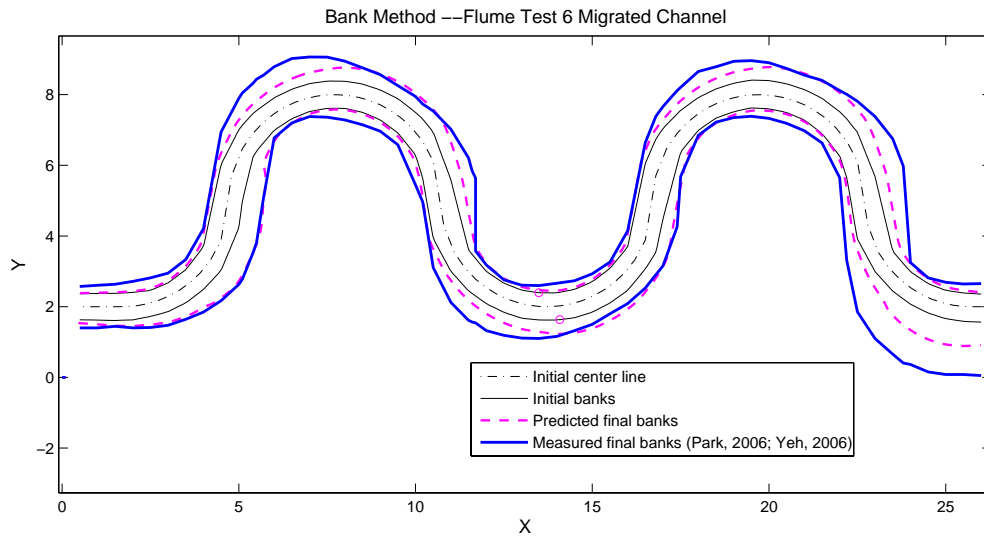


Figure 9.11 Verification of Flume Test 6 $R/W=4$, $\phi=180^\circ$, $v=0.24$ m/s

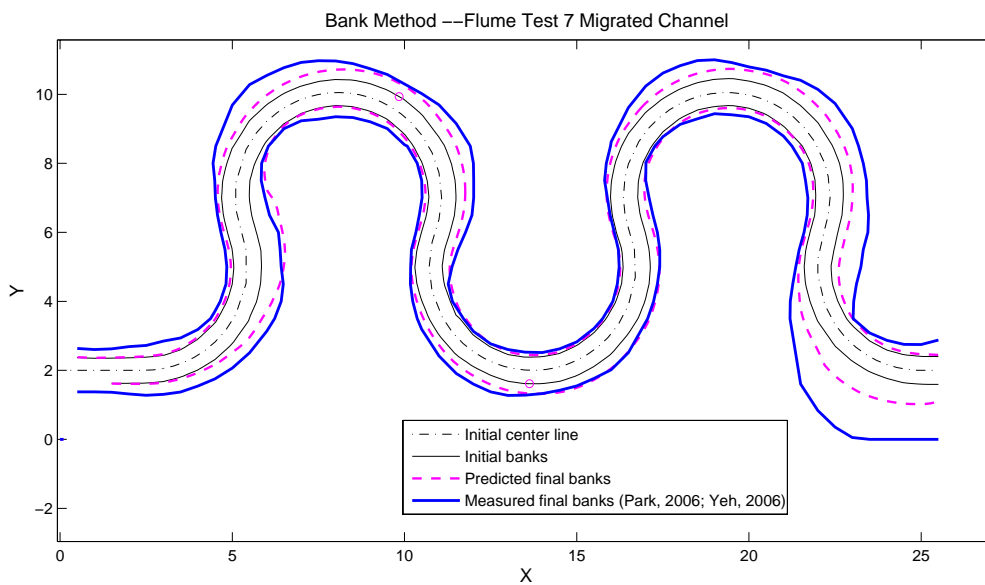


Figure 9.12 Verification of Flume Test 7 $R/W=4$, $\phi=220^\circ$, $v=0.24$ m/s

Figure 9.11 and Figure 9.12 show the two largest flume tests. The two peak migrations that were observable at the second bend of both tests were missed by the prediction. Po-Hung Yeh is developing a better version of M_{\max} equation to solve this problem. Detailed explanation will be seen in his doctoral dissertation. The double-peak phenomenon does not appear on the third bend which compound the complexity of migration problem. The current method has a good match for the third bend.

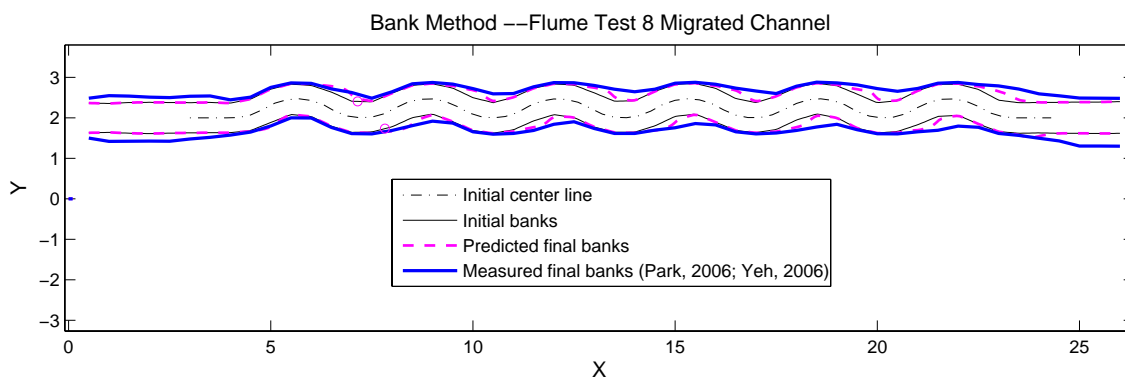


Figure 9.13 Verification of Flume Test 8 $R/W=2$, $\phi=65^\circ$, $v=0.19$ m/s

Flume Test 8 as shown in Figure 9.13 has the same geometry as Flume Test 3 but has a smaller velocity. The straightening effect is still obvious but less severe. The prediction partially matches the measured data.

The purpose of Flume Test 11 (Figure 9.14) was to see whether a straight channel can develop into a meander. The test was a success. A meandering channel with periodic geometry was formed in the end. The wave length was almost constant, 7 meters. Since there is only one bend at the entry for each bank, the Bank Method does not work here. By applying the model on the center line, the straight channel turns into a sinusoidal channel with a wavelength of about 7 meters. There is a phase lag of half period from the measured data.

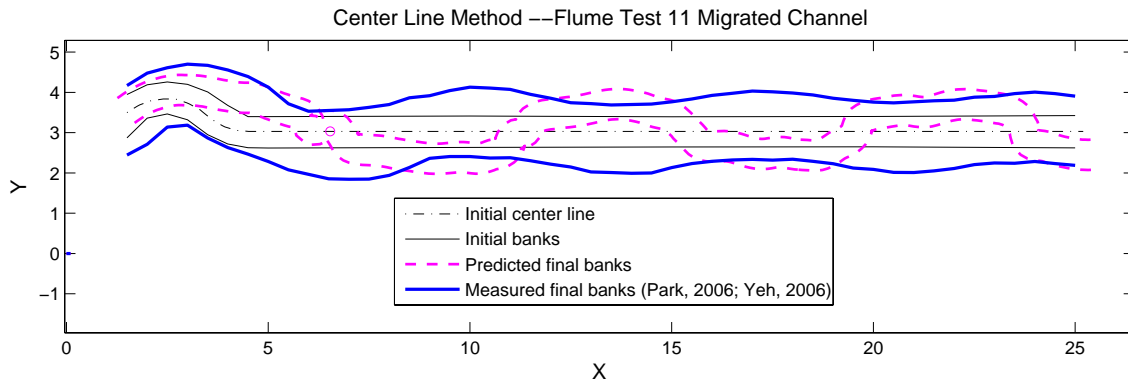


Figure 9.14 Verification of Flume Test 11, $v=0.30$ m/s

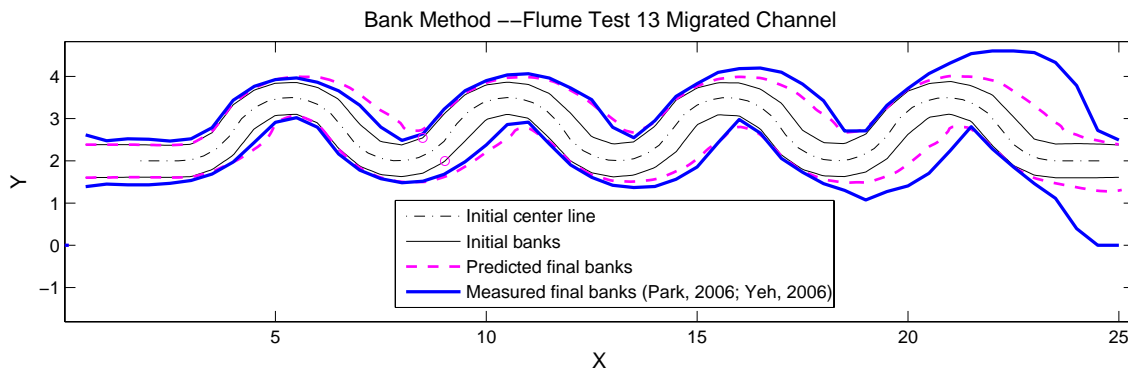


Figure 9.15 Verification of Flume Test 13 $R/W=2$, $\phi=120^\circ$, $v=0.20$ m/s

Flume Test 13 is also a case of small radius to width ratio. Due to large ϕ angle, the straightening effect is not so obvious. As can be seen in Figure 9.15, large migration at the exit still existed. The prediction for the first several bends is satisfactory.

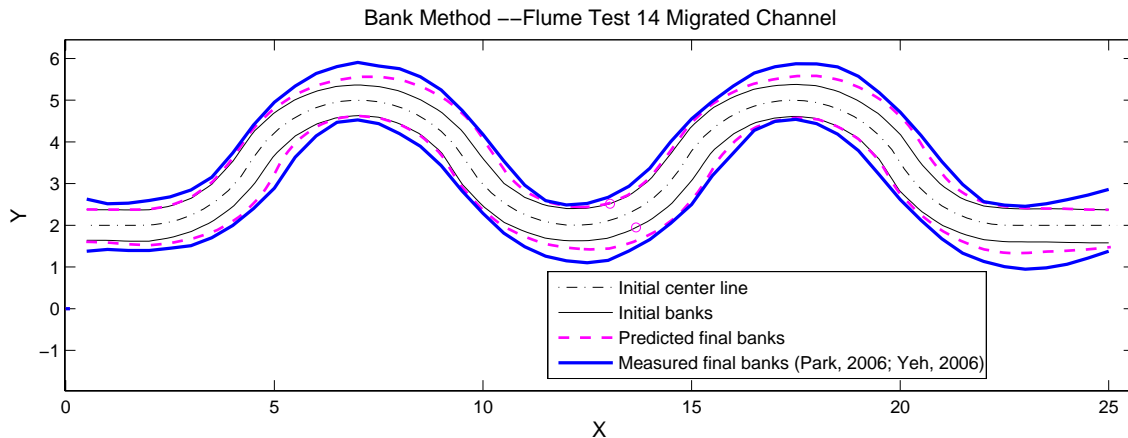


Figure 9.16 Verification of Flume Test 14 $R/W=4$, $\phi=120^\circ$, $v=0.20$ m/s

Figure 9.16 is the verification of Flume Test 14 which has the same geometry as Flume Test 1, 15 and 17. The difference is in velocity. The prediction missed the maximum migration of the bends. A good match occurs when the ratio of location angle to bend angle (θ/ϕ) is in the range of 0.7 to 1.0. Flume Test 15 has a larger velocity than Flume Test 14. Figure 9.17 displays a better prediction result for this case. Flume Test 16 is another case of small radius to width ratio. A phenomenon similar to Flume Test 3 can be seen in Figure 9.18. Flume Test 17 as shown in Figure 9.19 has the largest initial velocity of all the flume tests. For the third bend the prediction almost overlaps the measured data. The second bend was slightly overpredicted.

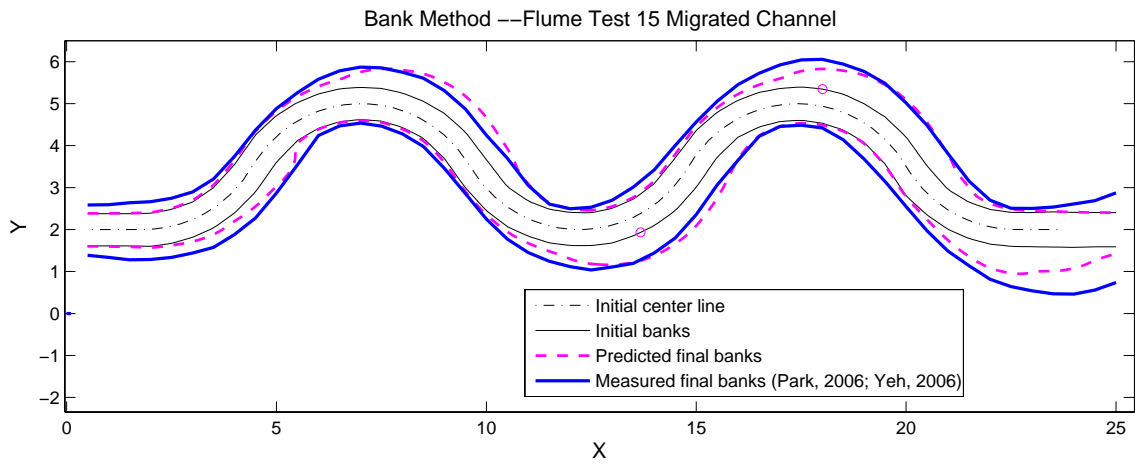


Figure 9.17 Verification of Flume Test 15 $R/W=4$, $\phi=120^\circ$, $v=0.27$ m/s

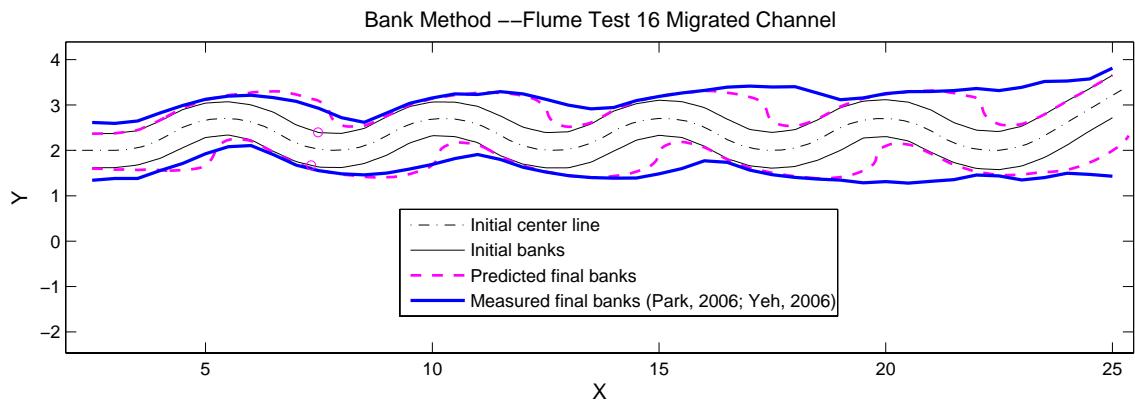


Figure 9.18 Verification of Flume Test 16 $R/W=3$, $\phi=65^\circ$, $v=0.25$ m/s

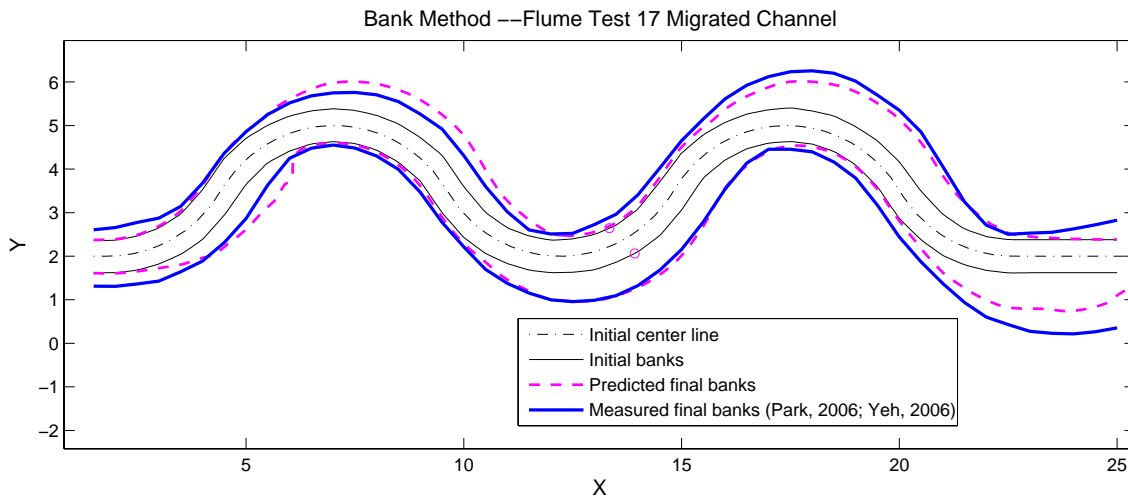


Figure 9.19 Verification of Flume Test 17 $R/W=4$, $\phi=120^\circ$, $v=0.32$ m/s

Flume Test 18 is the last experimental test on sand. Figure 9.20 shows a good match of the prediction with measured data.

The prediction matches the measured data reasonably well for all the listed cases. However, the good matches are not done in one step. It is an interactive process of testing and modifying the program and the model. The prediction model was proposed based on observation of the test data. Matching the test data step by step is a gradual process of debugging the program and improving the model until an overall good match is made. Verification of flume tests is only the first step. The program can be used only after verification with real rivers.

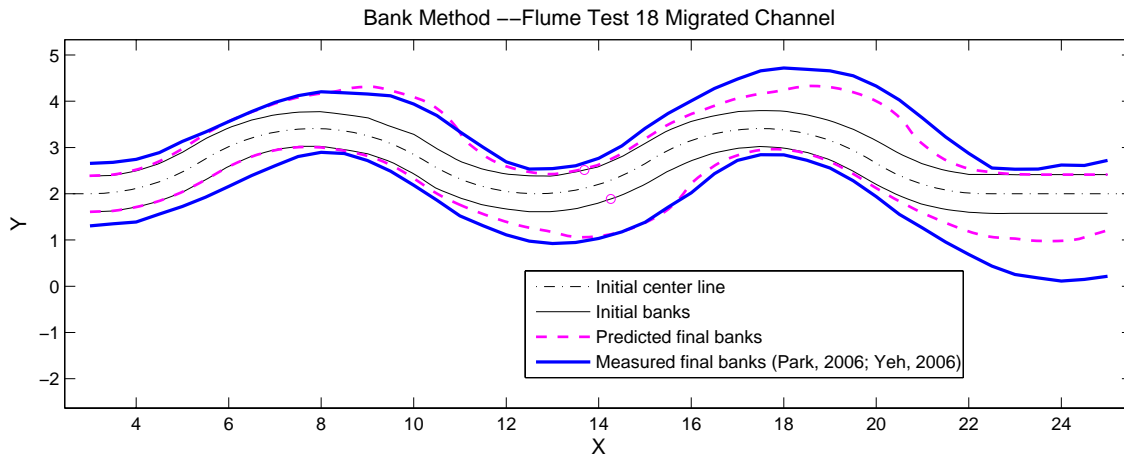


Figure 9.20 Verification of Flume Test 18 $R/W=6$, $\phi=65^\circ$, $v=0.26$ m/s

9.2 PRELIMINARY VERIFICATION OF THE CASE OF THE BRAZOS RIVER

The Brazos River runs 840 miles across Texas to its mouth on the Gulf of Mexico. It is the longest river in Texas and the one with the largest discharge. The migration of this river at bridge site SH105 near Navasota has previously been studied (Briaud, et al. 2001a, 2001b). Figure 9.21 shows the location of the river at different times. The migration from 1910 to 1958 at the crossing of the bridge contrasts to what the model would predict. Two periods were selected for verification: 1958 to 1981 and 1981 to 1993.

Samples from field sites will be taken on a later date, therefore soil properties for this case are not available yet. Based on numerous EFA test results, an assumed EFA curve is shown in Figure 9.22.

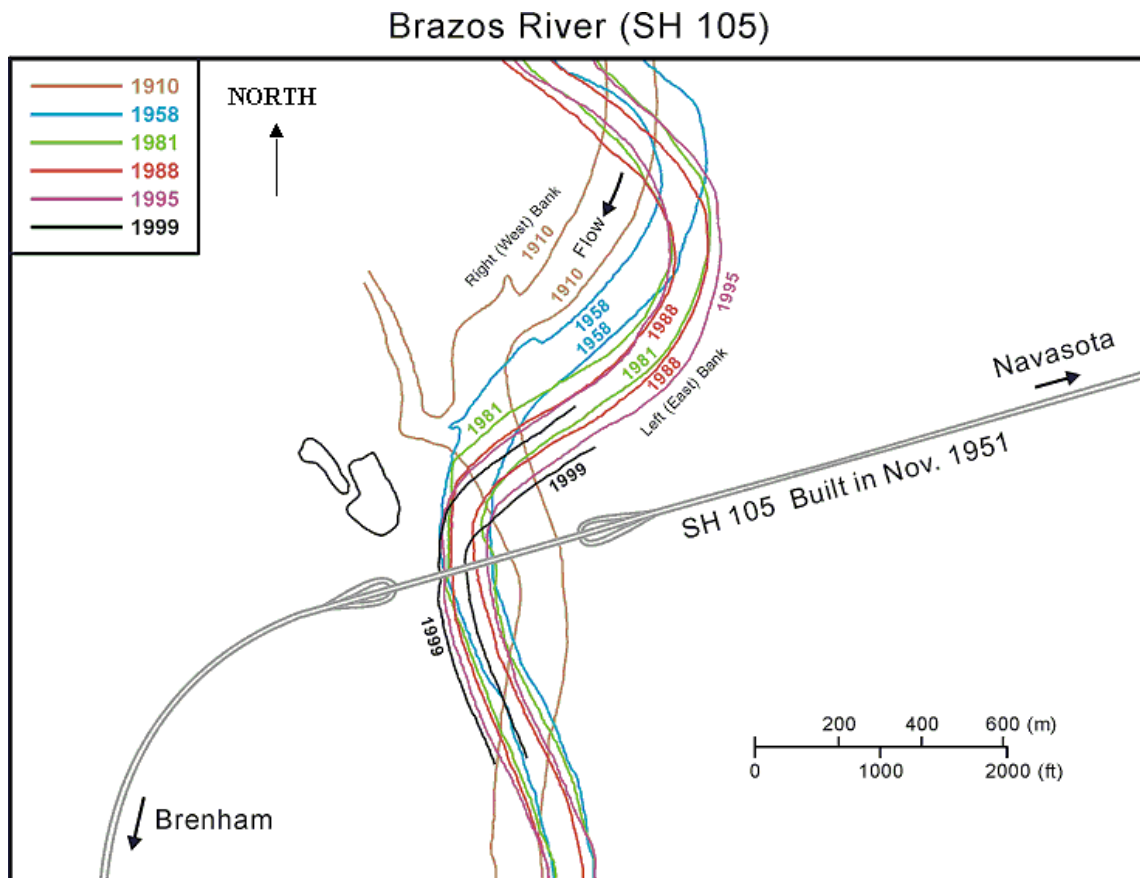


Figure 9.21 Migration of the Brazos River (Park, 2001)

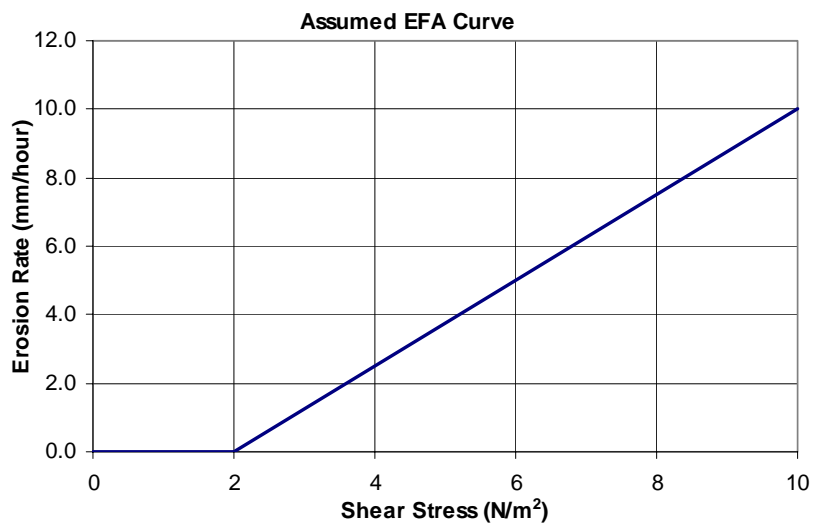


Figure 9.22 Assume EFA Curve for the verification of the Brazos River

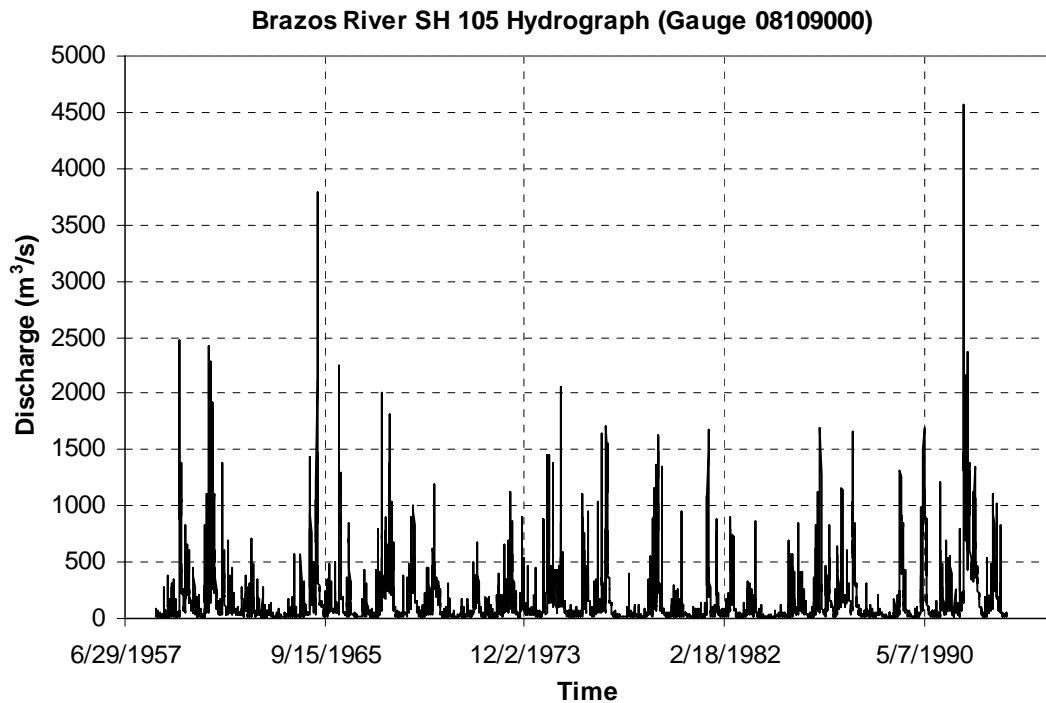


Figure 9.23 Hydrograph of the Brazos River SH105 1958-1993 (www.usgs.gov)

Figure 9.23 is the hydrograph of this site from 1958 to 1993. The file downloaded from USGS website contains data only up to September 30, 1993. The unit of discharge is converted from ft^3/s to m^3/s . Since velocity is used in the prediction, the curve relating discharge to velocity as shown in Figure 9.24 is needed. Figure 9.25 indicates the relation between discharge and water depth. Both relations came from a HEC-RAS simulation. Given the cross section, the discharge, and other needed parameters of the river, HEC-RAS can calculate the velocity, water depth, and water surface elevation. Soil and water data is ready. Geometry data is obtained by digitizing the graph in Figure 9.22. It is time to proceed to the step of prediction.

After 8400 steps of calculation, a predicted channel of 1981 was obtained as shown in Figure 9.26 with the dash line. The second bend where the bridge passes has been reinforced. The migration of this part during the 23 years is almost negligible in regard to possible measurement errors. The first bend's migration is mainly in a

downstream direction while the predicted migration is mostly in radial direction. Predicted migration distance is very close to that of the measured for the selected point.

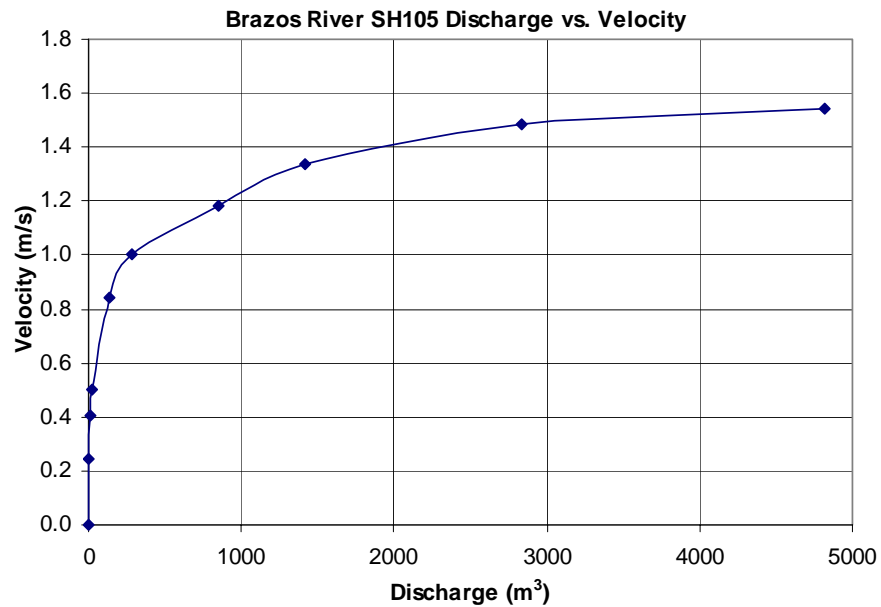


Figure 9.24 Discharge vs. velocity (Park, 2001)

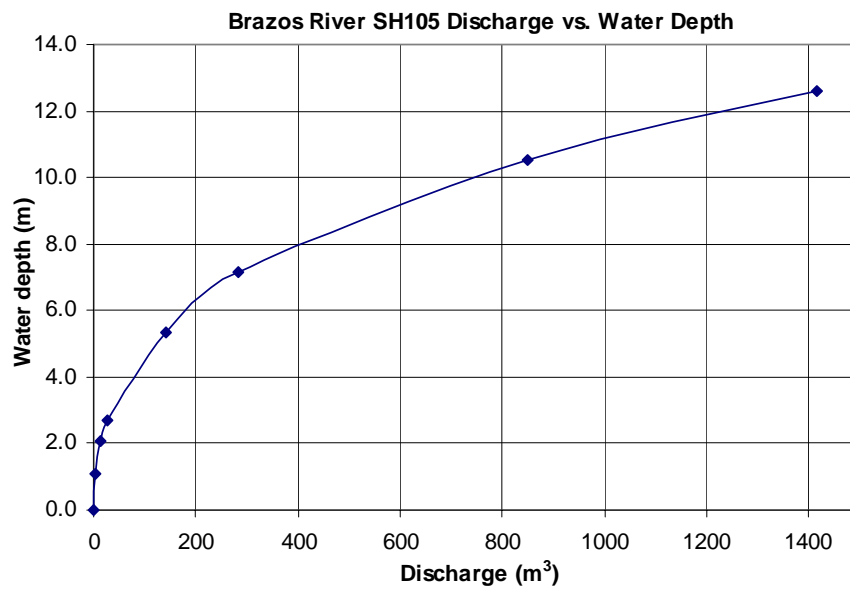


Figure 9.25 Discharge vs. water depth (Park, 2001)

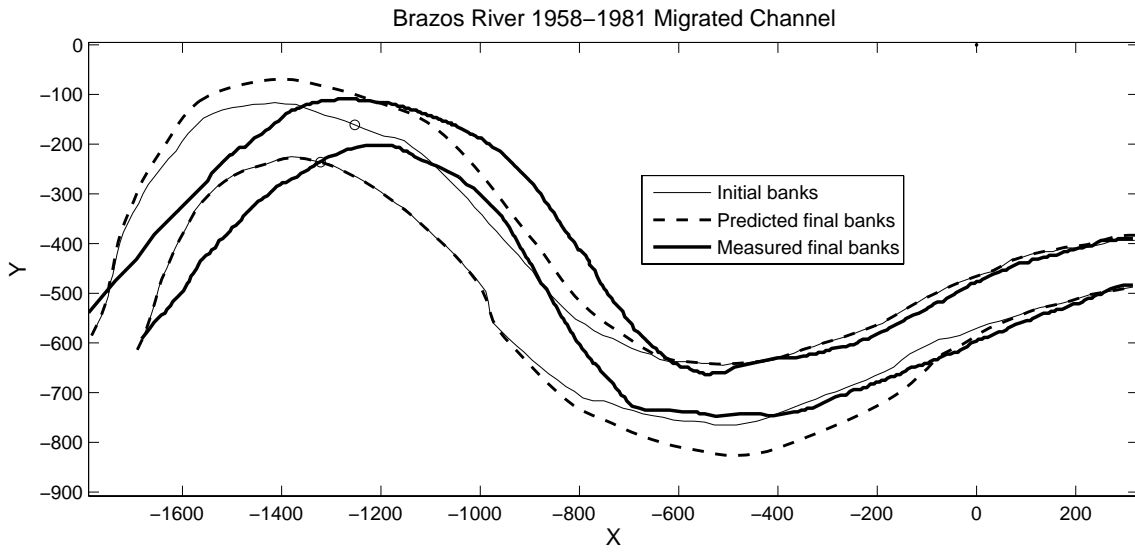


Figure 9.26 The Brazos River SH105 1958-1981 measured vs. predicted

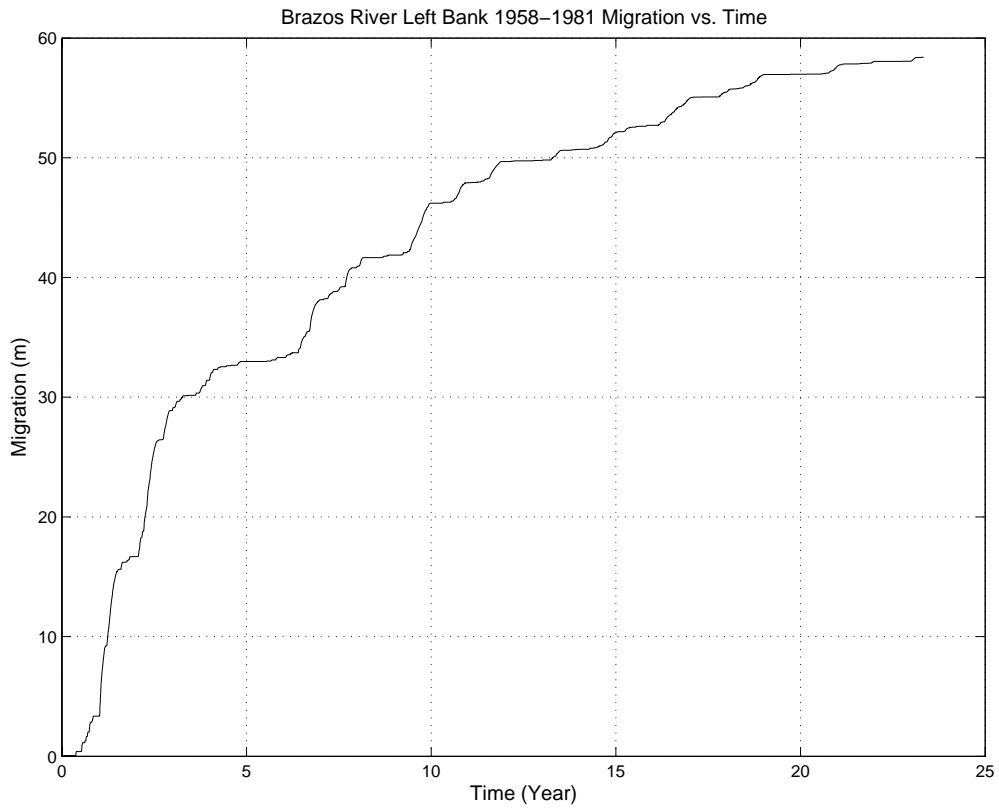


Figure 9.27 The Brazos River SH105 1958-1981 predicted migration vs. time

The predicted migration versus time curve for a selected point on the left bank is shown in Figure 9.27. The rate of migration is in accordance with the hydrograph. A large discharge on a certain day leads to a large migration distance for that day. A small discharge can only cause a small jump on the curve. If the discharge is small enough to where the velocity is below critical velocity, no migration will occur. In this case it appears as a horizontal line segment on the curve.

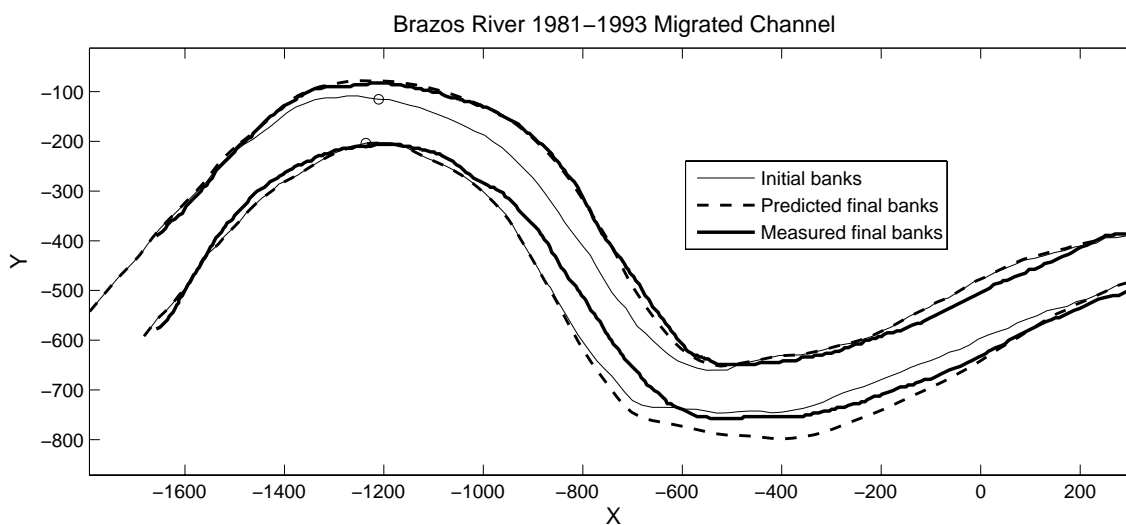


Figure 9.28 The Brazos River SH105 1981-1993 measured vs. predicted

The measured channel of 1993 is not available and the hydrograph for the period of 1993 to 1995 is not available either. The prediction therefore can only be made through 1993. The comparison is made between the predicted channel of 1993 and the measured channel of 1995. It can be reasonably assumed the migration that occurred from 1993 to 1995 is small enough that the measured channel of 1995 can be used to judge the accuracy of the prediction. Figure 9.28 shows a wonderful match for the first bend. The second bend is still overpredicted due to reinforcement of the bank. The prediction takes 4382 steps. Figure 9.29 is the migration versus time curve for a selected

point on the left bank. The effect of a hydrograph is more obvious in this figure. The plateaus indicate numerous small discharges in the hydrograph.

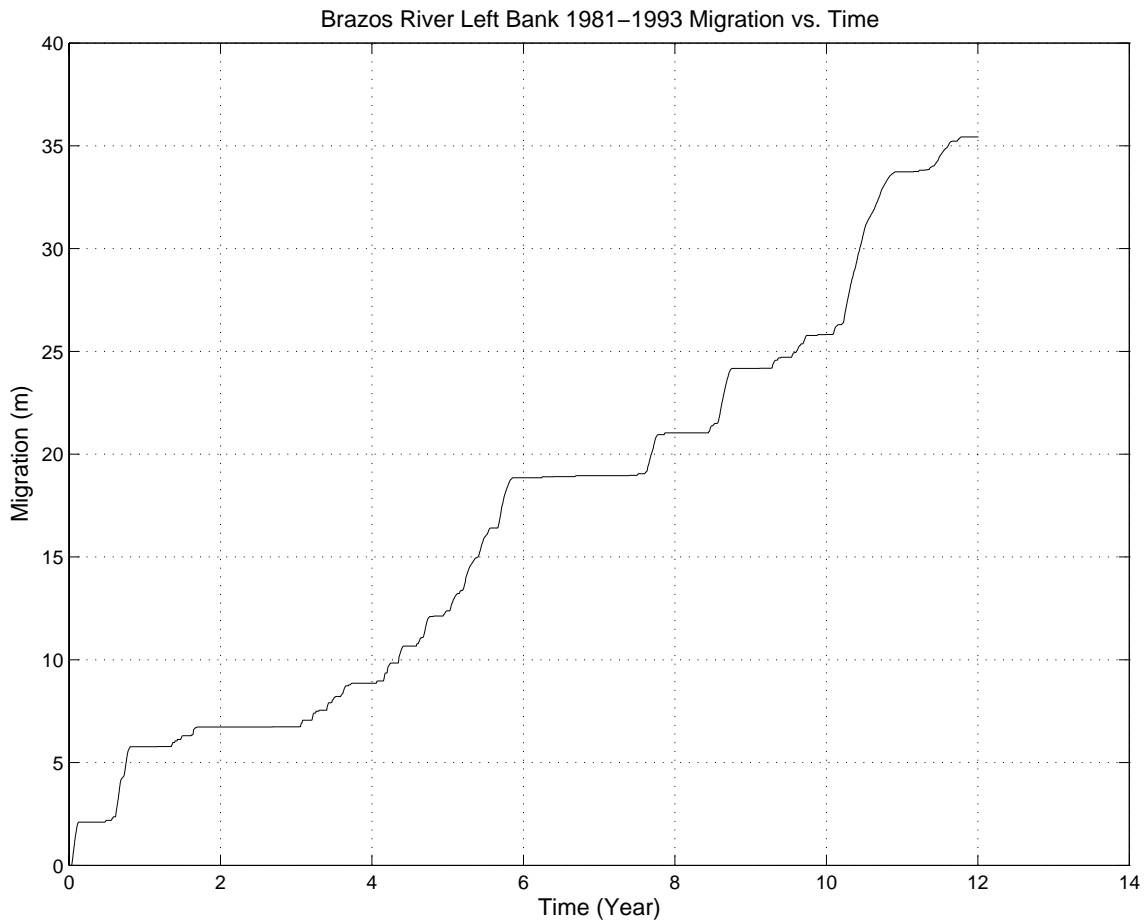


Figure 9.29 The Brazos River SH105 1981-1993 predicted migration vs. time

9.3 PARAMETRIC STUDY

The first step in a parametric study is to choose influential parameters and to decide on a reference case. The parameters chosen include the following: the ratio of radius of curvature to river width (R/W), bend angle (ϕ°), velocity (V), duration of the

flow (t), and the slope of the assumed EFA curve (S_i). The reference case is numbered Case 1 and has the combination: $R/W=4$, $\phi=180^\circ$, $V=1.5$ m/s, $t=360$ days, $S_i=1.25$ mm/hour/Pa. A list of cases is generated by varying only one parameter from the reference case. Thus the influence of a single parameter on the migration can be studied. Table 9.2 shows all the cases in groups where each group has only one changing parameter. The maximum value of migration at time t for the fourth bend of the left bank is also listed.

Table 9.2 Matrix of parametric study for the Brazos River SH105

Matrix of Parametric Study							
Case No.	R/W	ϕ ($^\circ$)	V(m/s)	t (days)	EFA Slope (mm/hr/Pa)	$M_{t,max}$ (m)	Note
2	4	60	1.5	360	1.25	32.29	Vary ϕ
3	4	120	1.5	360	1.25	31.97	
1	4	180	1.5	360	1.25	23.41	
4	4	240	1.5	360	1.25	16.56	
5	4	270	1.5	360	1.25	10.79	
6	1	180	1.5	360	1.25	19.21	Vary R/W
7	2	180	1.5	360	1.25	27.91	
1	4	180	1.5	360	1.25	23.41	
8	6	180	1.5	360	1.25	19.77	
9	8	180	1.5	360	1.25	17.95	
10	10	180	1.5	360	1.25	14.36	Vary t
11	4	180	1.5	90	1.25	13.38	
12	4	180	1.5	180	1.25	18.99	
1	4	180	1.5	360	1.25	23.41	
13	4	180	1.5	720	1.25	27.53	
14	4	180	1.5	1440	1.25	29.66	Vary slope of EFA curve
15	4	180	1.5	2880	1.25	30.85	
16	4	180	1.5	360	0.125	7.07	
17	4	180	1.5	360	0.625	18.98	
1	4	180	1.5	360	1.25	23.41	
18	4	180	1.5	360	2.5	27.54	Vary V
19	4	180	1.5	360	12.5	30.98	
20	4	180	0.66	360	1.25	0.16	
21	4	180	0.7	360	1.25	2.52	
22	4	180	0.8	360	1.25	7.31	
23	4	180	0.9	360	1.25	11.09	
24	4	180	1	360	1.25	14.16	
25	4	180	1.2	360	1.25	18.83	
1	4	180	1.5	360	1.25	23.41	
26	4	180	2	360	1.25	30.41	
27	4	180	3	360	1.25	39.19	
Note:							
1. Critical shear stress is $\tau_c=2$ N/m ² , Froude number is $F_r=0.3$;							
2. $M_{t,max}$ is the maximum value of migration at time t for the 4th bend of the left bank;							
3. River width is 40 meters.							

The planar migration of the reference case is shown in Figure 9.30. Figure 9.31 is the migration versus time curve for a point on the 4th bend of the left bank. The same plots for other cases are omitted here. Due to the limitations associated with the Bank Method, an inner bank either moves inward or does not move at all. An outer bank will move outward. So the predicted channel widens. The migration versus time curve indicates that the predicted channel is still quite far from M_{\max} at the end of 360 days.

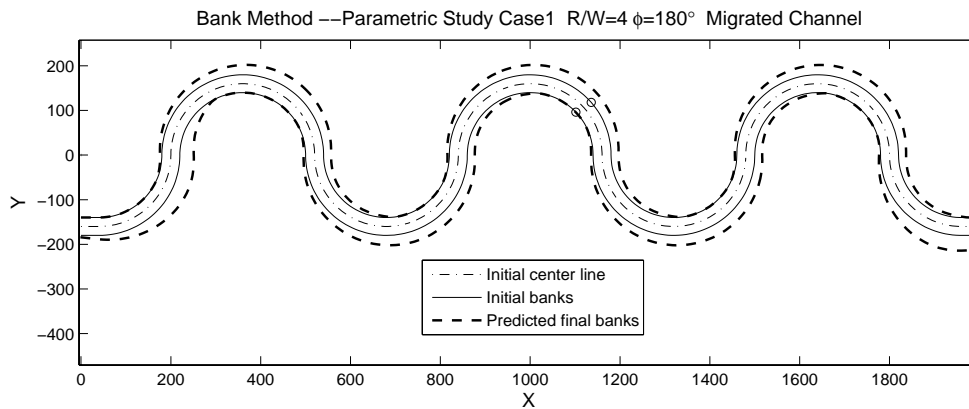


Figure 9.30 Parametric study reference case, migration of channel

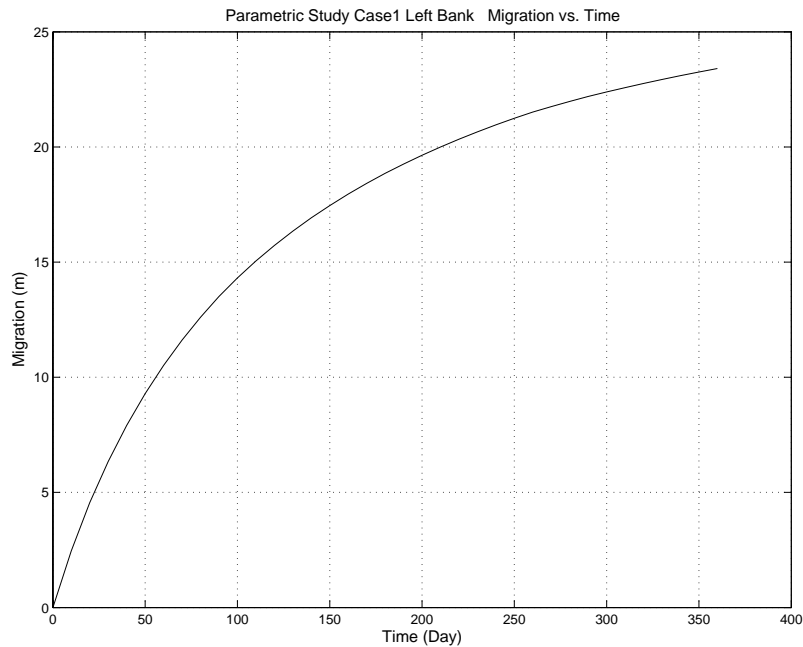


Figure 9.31 Parametric study reference case, migration vs. time

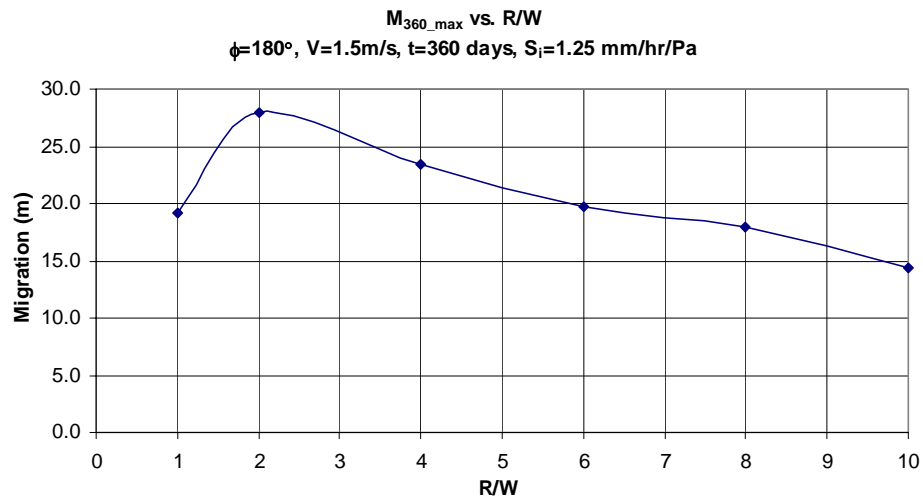


Figure 9.32 Parametric study migration versus R/W

The important parameters are isolated and studied one by one. Their influence on migration distance is shown respectively in Figure 9.32 to Figure 9.36. Figure 9.32 shows how migration distance can be affected by radius to width ratio. The curve takes the same shape as the one proposed by Nanson and Hickin (1983). Here the maximum value is reached at $R/W=2$, while Nanson and Hickin suggested $R/W=2.5$. The peak will occur at a different R/W value if other parameters like ϕ or V is changed.

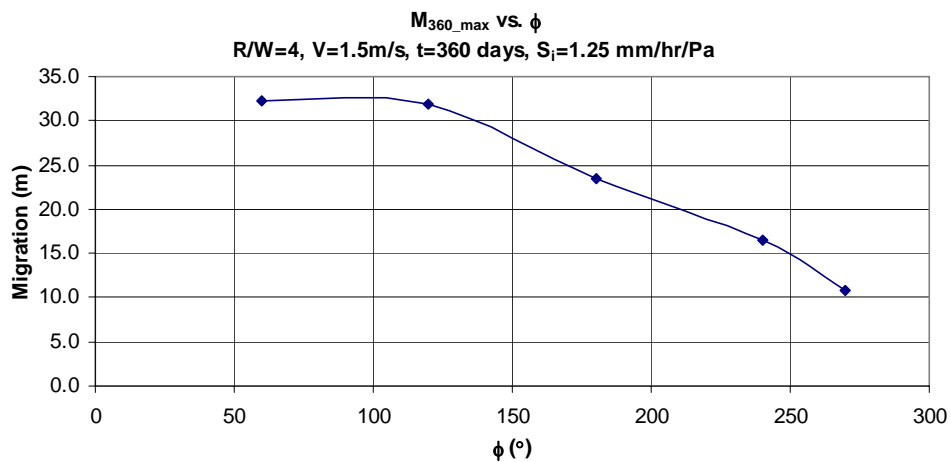


Figure 9.33 Parametric study migration versus bend angle

Figure 9.33 demonstrates the relationship between migration distance and bend angle predicted by this model. When the bend angle is larger than 120 degrees, migration distance decreases with the increase of bend angle. This phenomenon has been observed in flume tests. When the bend angle increases the channel length increases. The slope of the channel bed decreases accordingly and the flow slows down as a result. Field data is needed to prove that the relation shown in Figure 9.33 actually happens in real rivers. Further work needs to be done on this issue.

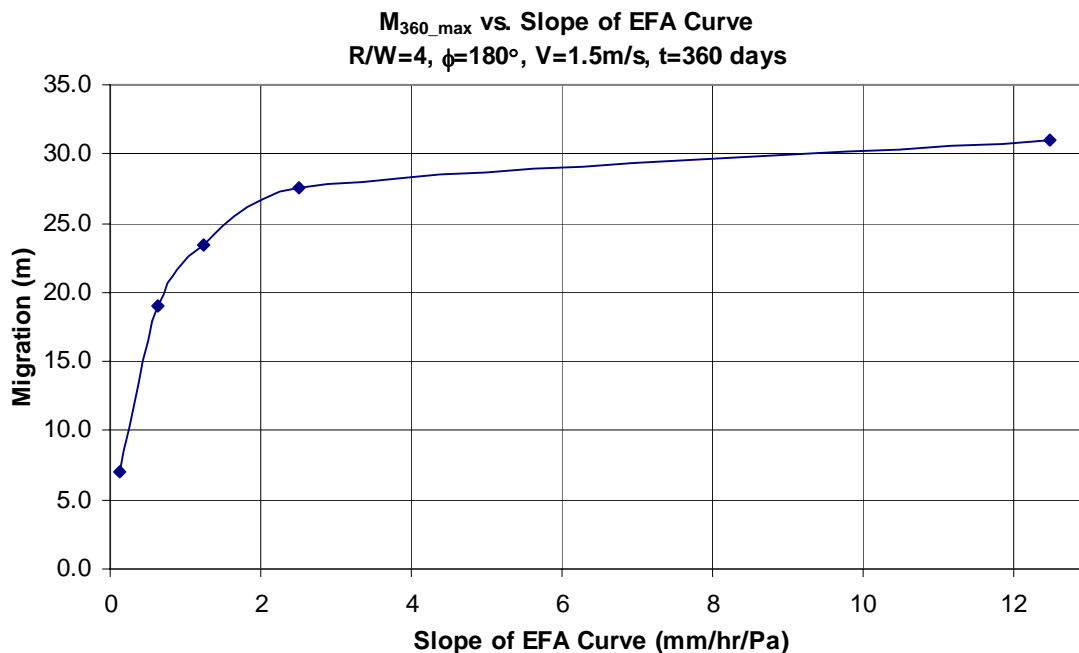


Figure 9.34 Parametric study migration versus slope of EFA curve

The topic of this research project is “Soil Properties-Based Prediction of Meander Migration Rate.” A focus on soil properties is the major difference between this research and others. The prediction model takes into account the factor of soil properties. Due to the cost and time associated with experimentation, only sand and clay are used in flume tests. A large variety of soil properties can be applied in a parametric

study. With this study the slope of EFA curve is varied from 0.125 mm/hour/Pa to 12.5 mm/hour/Pa. The effect can be seen in Figure 9.34. The larger the slope the more erodible the soil is. First, migration distance increases sharply with the increase of erodibility. Then migration becomes less sensitive to soil erodibility. This is because the maximum migration is almost reached.

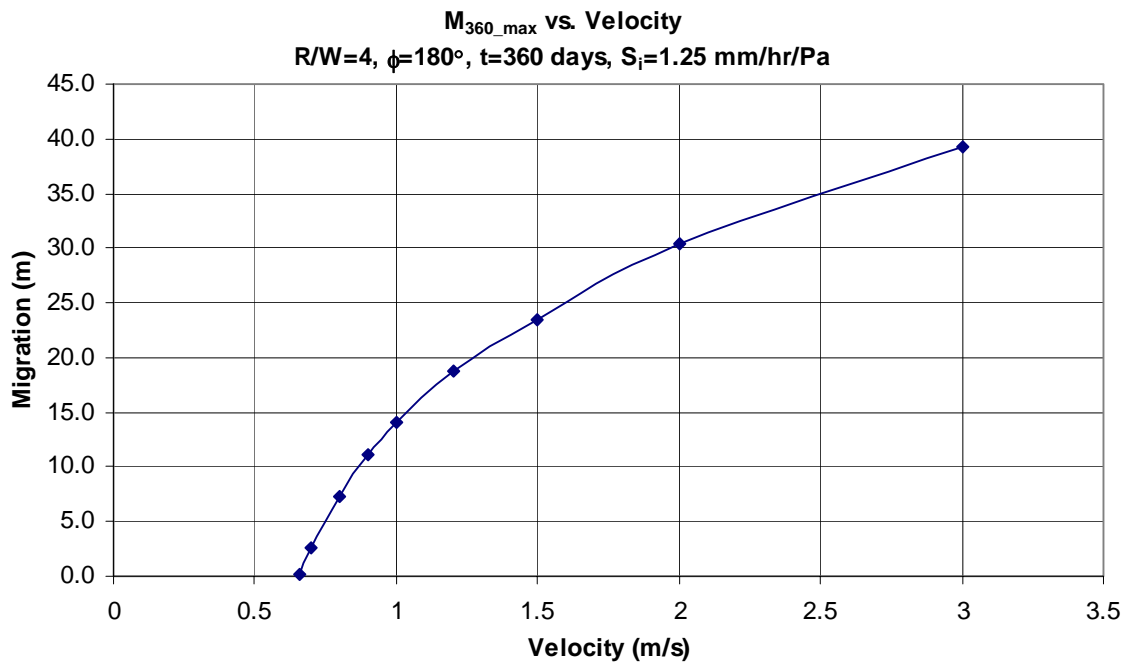


Figure 9.35 Parametric study migration versus velocity

Velocity of flow is a direct cause of soil erosion. A velocity vector can be decomposed into two perpendicular components: longitudinal flow and secondary flow. Secondary flow plays an important role in the erosion process and is responsible for transporting sediment from one bank to the other. The effect of longitudinal flow in eroding soil can be directly observed in flume tests. The velocity of secondary flow is in proportion to that of longitudinal flow. Due to the difficulties in data acquisition of flow field, only average velocity in the longitudinal direction is measured in flume test on

sand. This data has proved to be very useful in the prediction. Figure 9.35 shows how a change in longitudinal velocity can cause a change in migration distance. The intersection of the curve and horizontal axis is the critical velocity V_c , which is 0.66 m/s here. If velocity is less than V_c , no migration will occur.

Figure 9.36 shows how the channel migrates with time in a hyperbolic shape. At the end of eight years the channel will come close to maximum migration.

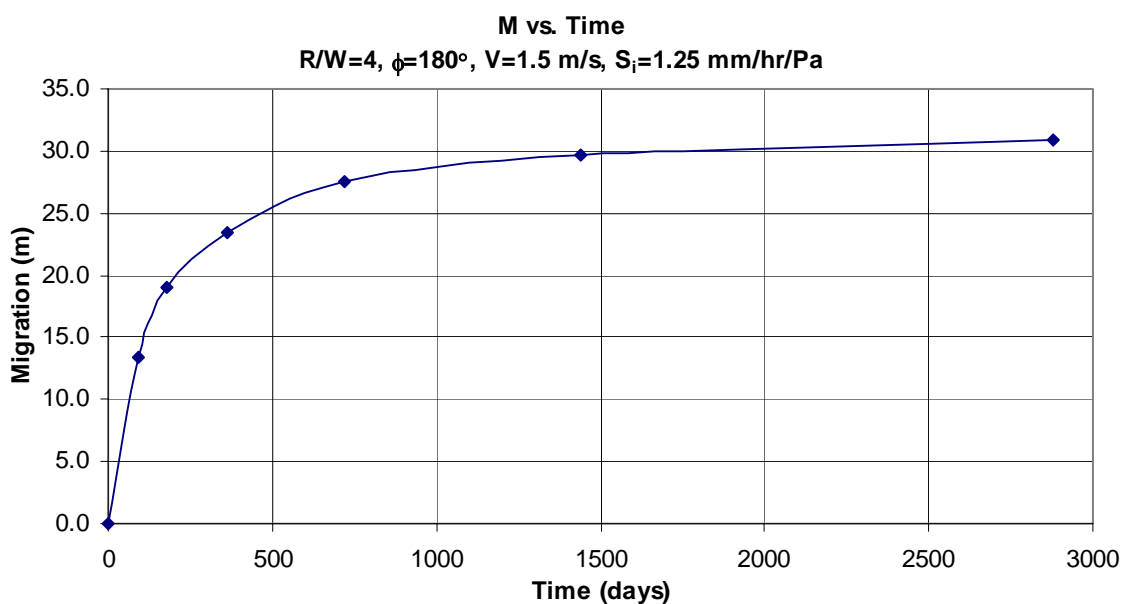


Figure 9.36 Parametric study migration versus time

9.4 CONCLUSION

The M_{\max} equation comes from the data of these flume tests and the τ_{\max} equation is obtained by fitting numerical simulation data. M_{\max} equation and τ_{\max} equation are the two major components of the Hyperbolic Model. Verification was done for 16 out of 18 flume tests on sand in the new Coastal Engineering Laboratory. The closeness of the prediction to the measured data verifies the effectiveness of both the Hyperbolic Model

and the M_{\max} and τ_{\max} equations. The validity of the implementation of the whole methodology as well as the validity of the MEANDER program is proved.

Eventually the model will be used to predict the migration of real rivers. The verification of the Brazos River case is a further proof to the applicability of the model and the program in the real world. At this stage, flume tests cannot produce soil erosion at one bank, sediment transport to the other side, and sediment deposition on the other bank. In flume tests, the channel always widens. As a result the prediction model inherits the same limitation. However, practice engineers are more concerned about which part of the bank will be eroded than which part of the bank will be filled with deposition.

In conclusion, the results of parametric study comply with the behavior of meander migration that has been observed. This demonstrates that the model can deal with not only individual cases but also rivers of all types.

CHAPTER X

CONCLUSIONS AND RECOMMENDATIONS

10.1 CONCLUSIONS

10.1.1 Introduction

The problem of meander migration has been troubling mankind for centuries. The erosion of banks endangers bridge abutments and highway foundations. The shifting of the channels causes loss of land and problems to navigation. Meandering rivers can be classified into three categories: single-thread channel with various sinuosity, braided channels, and anabranching channels. The most influential factors affecting meander migration are found to be: soil, water, and geometry of the channel.

10.1.2 Existing knowledge about meander migration

The solutions to the problem of meander migration proposed by numerous researchers can be classified into three categories: time sequence extrapolation, empirical equations, and fundamental modeling. Existing empirical equations consider only one or at most two of the most influential factors. Some equations are so simple that they are below the bar. The numerical methods involve solutions to the constitutive equations, development of sediment transport model and bank erosion model. None of the method can deal with changing water level and a hydrograph. The consideration of soil erosion was oversimplified.

10.1.3 Research objectives and methodology

Our approach can be considered as a combination of empirical equations and fundamental modeling. The flume test allows the team to study the influential factors respectively. EFA test describes erodibility as a fundamental property of soil which distinguishes this research from all others found in the literature. It was found that the Hyperbolic Model matches the migration process in flume tests and in real rivers. Predicting meander migration based on a hydrograph has never been done before. It is another key feature of this research. If written in a report, the whole methodology could

be hundreds of pages long. If the application of the methodology was done by manual labor, it would take months of time to do one prediction. But the MEANDER computer program assembles everything together and presents the model to practice engineers with user friendly graphic interface.

10.1.4 Flume test

The author conducted the first 13 flume tests in the old Hydro Lab. The preparation of the experimental setup includes purchasing the piping system and sand, designing false bottom, moving sand, calibration of the flow meter, and making measuring tools. A new flume test started at the end of previous one. The first step was to level the sand and dig a channel according to predetermined geometry. Measurements followed the start of a test. Bank migration was measured with a tape measure. Water depth was measured with point gauge and a ruler. Cross section profile was measured about every one meter along the channel. A paper boat was once used to measure surface velocity. The migration status of each time step was also recorded in digital photos. The results first indicated the possible existence of maximum migration. Due to the limitation of flume size, the tests of a much larger scale in the new Coastal Engineering Laboratory proved to be more valuable.

10.1.5 The application of the SRICOS-EFA method

The decreasing migration rate observed in flume tests led to the idea of applying the Hyperbolic Model. Maximum migration M_{\max} and initial migration rate \dot{M}_i are the two major components of this model. The derivation of the M_{\max} equation was based on flume test data (Details will be discussed in Po-Hung Yeh and Namgyu Park's doctoral dissertations which will likely come out in 2006). Erosion rate \dot{Z} on the EFA curve is treated as migration rate \dot{M} . The erosion rate corresponding to maximum shear stress τ_{\max} is called \dot{M}_i . A similar matrix like that of the flume test was set up for numerical simulation. The maximum shear stress τ_{\max} along the banks was calculated and plotted. The curve of *extreme value distribution* was chosen to fit the distribution of τ_{\max} along a

bend. Multiple regression was done to determine τ_{\max} as a function of other parameters. τ_{\max} equation was generated by combining all these functions together. With the equations for M_{\max} and τ_{\max} ready, the Hyperbolic Model for the prediction of meander migration is realized.

10.1.6 Geometry study

It was a challenge to have a computer to reduce complicated channel geometry into a form with acceptable complexity. Arcs and straight lines were chosen to represent actual channel geometry. Fitting circles and identifying the best fit circle are the core of geometry study. The first step is to calculate radius of curvature for each point on the bank and plot it versus channel lengthwise distance. Points on a bend normally have a small radius of curvature. The points with radii of curvature smaller than a certain value are roughly identified as a bend. Each of the two boundary points of the bend is extended in both directions into a segment. Each time one point is chosen from each segment. A circle is fitted to these two points and the points in between. In this way, a bunch of circles are produced which should include the best fit circle. The best fit circle tends to have more points but less fitting error at the same time. A method is developed to balance these two contradictory criteria. It is not perfect, but it works well for many cases.

10.1.7 Future hydrographs

Doing a prediction based on a hydrograph has little practical meaning except for verification purposes, because the hydrograph for the predicted period does not yet exist. The hydrograph for the future needs to be generated based on equivalent risk level of the existing hydrograph. It has been proved that a hydrograph reasonably follows a lognormal distribution. Based on its mean and standard deviation, a random number generator can generate tens of thousands of hydrographs of any desired length which also follows lognormal distribution and have the same mean and standard deviation. One hundred year flood Q_{100} and five hundred year flood Q_{500} are also a measure of risk level

that has existed in history. The mean and standard deviation can be analytically solved from Q_{100} and Q_{500} . Thus the same random number generator can be used again.

10.1.8 The MEANDER program

So far the MEANDER program has consisted of these main modules: graphic user interface (GUI), generation of future hydrographs in C++ and Matlab, geometry study in Matlab, implementation of hyperbolic model in C++ and Matlab, graphic output in Matlab. The module of risk analysis will be added by Namgyu Park. As far as programming technique is concerned, the MEANDER program requires skills in C++ language, Microsoft Foundation Class (MFC), Matlab, and a technique to integrate Matlab with C++. The result of all these is a user friendly input interface and powerful graphic output. Once getting familiar with the program, the user can virtually make use of all the findings the team has made.

10.1.9 Verification and parametric study

Verification was done one by one for most of the flume tests in the Coastal Engineering Lab. It is not realistic to have a prediction matching test data at each point. Good matches have been achieved for at least 40% of the bend length. A positive verification of Brazos River case demonstrates the program's applicability to real rivers. Reasonable relationships between migration distance and influential parameters were reproduced in parametric study, which further proves the reliability of the program and the prediction model.

10.2 RECOMMENDATIONS FOR FUTURE RESEARCH

The team can deliver a practical and useful product at this stage. When the project finishes in 2006 a better result will be achieved. However hard the team works, there are things the team cannot do due to the limit of budget and time. These things would become recommendations for future research.

The realization of the importance of channel bed slope came from analysis of the failure of the first flume test done in 2002 where gravel of $d_{50}=3\text{mm}$ was used and no

erosion occurred. Although initial velocity contributes to the momentum of flow, slope of channel bed plays a critical role, as can be seen in real rivers. It works the same way in flume tests. During a flume test the velocity gradually goes down. This is partially because eroded sand from the bank accumulates in the channel and reduces the initial slope. If the slope is increased, a larger magnitude and smaller variation of velocity are more likely to be achieved.

In real rivers, soil erodes on one bank and deposits on the opposite bank. So the whole channel migrates. But in flume tests, the deposition on the opposite bank is so little that only widening effect can be observed. Soil eroded from one bank can hardly travel to the opposite bank. In designing the flume test, velocity is scaled down from that of real rivers but not the soil particle size. The ratio of soil particle size to flow velocity is much larger in flume test than in real rivers. Moderately increasing the velocity will help sediment travel farther. Soil loss plays a role in the widening effect. For the flume tests in the old Hydro Lab, the author observed a pile of sand behind the exit (Appendix B). The larger the velocity the more sand will be lost. If the lost sand can be fed at the entry, a balance of sediment can be maintained. If a balance between sediment supply and sediment loss can be maintained, it is more likely to reproduce what happens in nature (Parker and Wilcock 1993; Friedkin, 1945).

The shear stress provided by numerical simulation appears to be periodic from one bend to the next, however, the measured migration distance increases along the channel. Developing secondary flow can partially account for this phenomenon. It is recommended to measure how the secondary flow changes from bend to bend. At the exit, there is a free fall of water and a sudden change of material from soil to wood. Excessive erosion often occurs here and moves upstream. Eliminating the free fall (Figure B.20) and strengthening banks close to the exit will help create a normal flow condition in this area.

Although the velocity varies from entry to exit and changes from the beginning to the end and the geometry is different for each time step, only the initial average velocity and initial geometry are used to develop M_{\max} equation. It is a good

approximation since test data does not follow the Hyperbolic Model perfectly. The change of velocity in time leaves some room for improving the M_{\max} -based prediction. The migration should be linked to instantaneous velocity and geometry instead of initial velocity and geometry. At this stage, it is hard to analyze the effect of changing velocity and geometry on migration based the Hyperbolic Model. Future researchers may develop a method that can make full use of the measured data.

When the Hyperbolic Model is being used, at any time step a point has two properties, existing migration distance and a migration versus time curve which is determined by instantaneous soil, water and geometry properties. The existing migration distance positions the point on the right location on the migration versus time curve. Then the program moves the point one time step forward along the curve. Since the existing migration distance is accumulated from time zero, if the calculation starts at an earlier time, the existing migration distance will be larger and the calculated migration for the current step will be smaller. A hypothetical case can better explain this limitation. Assume the change of geometry, soil, and flow properties at three different times is so small that the M versus t curves are very similar. Two predictions are made to predict the location of a point at time t_3 , as shown in Figure 10.1. The first prediction starts from t_1 and the predicted migration distance at time t_2 is M_1 , and at time t_3 is M_2 . The second prediction starts from t_2 . At time t_2 , the existence migration distance for prediction 1 is M_1 and it follows curve 1; the existence migration distance for prediction 2 is zero and it follows curve 2. The new location indicated by prediction 2 is M_3 . There is a difference of $M_3 - M_2$.

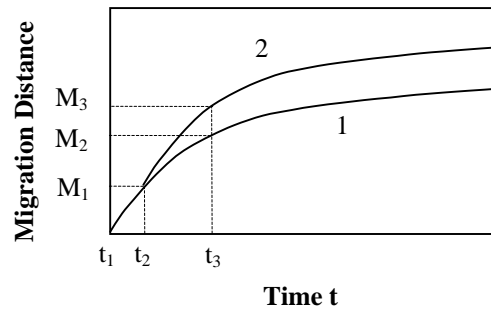


Figure 10.1 Start the prediction from different times

The migration versus time curve should correspond to a constant geometry. But the reality is that geometry changes with time. These limitations pose a challenge to a wider application of the Hyperbolic Model. An alternative will be ignoring what happened in history and use instantaneous properties only. Instead of using M_{\max} and τ_{\max} , future researchers may want to develop and use migration rate \dot{M} as a function of instantaneous soil, water and geometry properties.

The distribution of maximum shear stress τ_{\max} generated by numerical simulation approximately follows *extreme value distribution* which is like a skewed *Gaussian distribution* in shape. The location of the maximum τ_{\max} along a bend is normally close to that of maximum migration distance. The numerical simulation is partially verified by flume test. It is desirable to run more numerical simulation cases to discover the behavior of some cases that haven't been run in flume tests.

Geometry study is a major task of this project and is very time-consuming. Although a working solution has been produced, a better solution would be extremely helpful. In computer science this problem can be classified into the category of *computer vision* or *pattern recognition*. The solution to a much more complicated problem, Optical Character Recognition (OCR), is already out there. Handwriting recognition program for some complex characters like Chinese has been in the market for some time. After the user trains the program for a while it will recognize a non-regular style of handwriting. The training process is called *computer learning*. It should be much easier to recognize

arcs and straight lines than to recognize handwriting in Chinese. If the project budget allows, it would be better to subcontract this task to related experts in computer science.

A large amount of time was saved by adapting the engine of the GUI of SRICOS-EFA program for the MEANDER program. The most difficult parts of the GUI include tabular data input and graphic data output. The former programmer might have spent months of time writing the code. But a commercial tabular data input module costs only about \$150 without mentioning the difference in quality of code. Graphic data output module is also available, however, Matlab is a better choice for the demanding requirement of the MEANDER project and it is free to the client. Matlab is well known for its power of graphic output. With a good understanding of the Matlab Compiler, a programmer can make full use of this software to reduce programming work.

REFERENCES

- Abad, J., and Garcia, M. H. (2004). "Conceptual and mathematical model for evolution of meandering rivers in naturalization processes." *Proceedings of the 2004 World Water and Environmental Resources Congress: Critical Transitions in Water and Environmental Resources Management*, Salt Lake City, UT, 2048-2057.
- Blondeaux, P., and Seminara, G. (1985). "A unified bar-bend theory of river meanders." *Journal of Fluid Mechanics*, 157, 449-470.
- Briaud J.-L., Ting, F., Chen, H. C., Gudavalli, R., Perugu, S., and Wei, G. (1999). "SRICOS: Prediction of scour rate in cohesive soils at bridge piers." *Journal of Geotechnical and Environmental Engineering*, ASCE, 125(4), 237-246.
- Briaud J.-L., Ting, F., Chen, H. C., Cao, Y., Han, S. W., and Kwak, K. W. (2001a). "Erosion function apparatus for scour rate predictions." *Journal of Geotechnical and Geoenvironmental Engineering*, ASCE, 127(2), 105-113.
- Briaud J.-L., Chen H.-C., and Park S. (2001b). "Predicting meander migration: Evaluation of some existing techniques." *Texas Transportation Institute Report No. 2105-1 for Texas Department of Transportation*, The Texas A&M University System, College Station, TX.
- Briaud J.-L., Chen H.-C., Edge W., Park S., and Shaw A. (2001c). "Guidelines for bridges over degrading and migrating streams, part 1: Synthesis of existing knowledge." *Texas Transportation Institute Report No. 2105-2 for the Texas Department of Transportation*, The Texas A&M University System, College Station, TX.
- Briaud J.-L., Chen H.-C., Li, Y., Nurtjahyo, P, and Wang, J. (2003). "Complex pier scour and contraction scour in cohesive soils." *Texas Transportation Institute Report No. 24-15 for National Cooperative Highway Research Program*, The Texas A&M University System, College Station, TX.
- Brice, J. C. (1975). "Airphoto interpretation of the form and behavior of alluvial rivers." *Final Report to the U.S. Army Research Office*, Washington, D.C.
- Brice, J. C., and Blodgett, J. C. (1978). "Countermeasures for hydraulic problems at bridges, vol. I. analysis and assessment. (Final report. Jun 75 – Sep 78)." *Report no. FHWA/RD-78/162*, Federal Highway Administration, Washington, D.C.
- Brice, J. C. (1982). "Stream channel stability assessment." *Report no. FHWA/RD-82/021*, Federal Highway Administration, Washington, D.C.
- Chen, H.-C. (2002). "Numerical simulation of scour around complex piers in cohesive soil." *First International Conference on Scour of Foundations, ICSF-1*, vol. 1, Texas A&M University, College Station, TX, 14-33.

Crassidis, J. L., and Junkins, J. L. (2004). *Optimal Estimation of Dynamic Systems*, Chapman & Hall/CRC, Boca Raton, FL.

Darby, S. E., and Thorne, C. R. (1994). "Physically-based model of channel widening." *Proceedings - National Conference on Hydraulic Engineering*, Buffalo, NY, 944-948.

Darby, S. E., and Thorne, C. R. (1996a). "Numerical simulation of widening and bed deformation of straight sand-bed rivers. I: Model development." *Journal of Hydraulic Engineering*, 122(4), 184-193.

Darby, S. E., Thorne, C. R. and Simon, A. (1996b). "Numerical simulation of widening and bed deformation of straight sand-bed rivers. II: Model evaluation." *Journal of Hydraulic Engineering*, 122(4), 194-202.

Darby, S. E., Alabyan, A. M., and Van de Wiel, M. J. (2002). "Numerical simulation of bank erosion and channel migration in meandering rivers." *Water Resources Research*, 38(9), 21-221.

Duan, J. G., Jia, Y., and Wang, S. S. Y. (1997). "Meandering process simulation with a two dimensional model." *Proc., Conf. on Mgmt. of Landscapes Disturbed by Channel Incision*, University of Mississippi, Oxford, MS, 389-394.

Duan, J. G. (1998). "Simulation of alluvial channel migration processes with a two-dimensional numerical model." Ph.D. dissertation, The Center for Computational Hydroscience and Engineering, the University of Mississippi, Oxford, MS.

Duan, J. G., Wang, S. S. Y., and Jia, Y. (2001). "The applications of the enhanced CCHE2D model to study the alluvial channel migration processes." *Journal of Hydraulic Research/De Recherches Hydrauliques*, 39(5), 469-480.

Duan, J. G., and Julien, P. Y. (2005). "Numerical simulation of the inception of channel meandering." *Earth Surface Processes and Landforms*, 30(9), 1093-1110.

Duan, J. G. (2005). "Analytical approach to calculate rate of bank erosion." *Journal of Hydraulic Engineering*, 131(11), 980-990.

Friedkin, J. (1945). "A laboratory study of the meandering of alluvial rivers." *Technical Report*, U.S. Waterways Experimentation, Vicksburg, MS.

Hasegawa, K. (1981). "Bank-erosion discharge based on a non-equilibrium theory." *Proc. JSCE*, Tokyo, 316, 37-50 (in Japanese).

Hooke, J. M. (1980). "Magnitude and distribution of rates of river bank erosion." *Earth Surface Processes*, 5(2), 143-157.

Howard, A. D. (1996). "Modelling channel evolution and floodplain morphology." *Floodplain Processes* (M. G. Anderson, D. E. Walling, and P. D. Bates, eds), Chichester, John Wiley & Sons, 15-62.

Hudson, P. F., and Kesel, R. H. (2000). "Channel migration and meander-bend curvature in the lower Mississippi River prior to major human modification." *Geology*, 28(6), 531-534.

Ikeda, S., Parker, G., and Sawi, K. (1981). "Bend theory of river meanders. I: Linear development." *Journal of Fluid Mechanics*, 112, 363-377.

Jang, C.-L., and Shimizu, Y. (2005). "Numerical simulation of relatively wide, shallow channels with erodible banks." *Journal of Hydraulic Engineering*, 131(7), 565-575.

Johannesson, H., and Parker, G. (1989). "Linear theory of river meanders." *River Meandering, Water Resources Monograph*, vol. 12, edited by S. Ikeda and G. Parker, AGU, DC, 181-213.

Keady, D. M., and Priest, M. S. (1977). "The downstream migration rate of river meandering patterns." *Proceedings, Mississippi Water Resources Conference*, Meeting 12th Mississippi Water Resources Conference, Jackson, MS, 29-34.

Kitanidis, P. K., and Kennedy, J. F. (1984). "Secondary currents and river-meander formation." *Journal of Fluid Mechanics*, 144, 217-229.

Kovacs, A., and Parker, G. (1994). "A new vectorial bedload formulation and its application to the time evolution of straight river channels." *Journal of Fluid Mechanics*, 267, 153-183.

Lagasse, P. F., Spitz, W. J., Zevenbergen, L. W., and ZachMann, D. W. (2004a), "Methodology for predicting channel migration." *Report for National Cooperative Highway Research Program Project 24-16*, Owen Ayres & Associates, Inc, Fort Collins, CO.

Lagasse, P.F., Spitz, W.J., Zevenbergen, L.W., and ZachMann, D.W. (2004b). "Handbook for predicting stream meander migration using aerial photographs and maps." *Report for National Cooperative Highway Research Program Project 24-16*, Owen Ayres & Associates, Inc, Fort Collins, CO.

Lane, E. W. (1957). "A study of the shape of channels formed by natural streams flowing in erodible materials." *M.R.D. Sediment Series No. 9*, U.S. Army Engineers Division, Missouri River Corps of Engineers, Omaha, NE.

Lane, S. N., Chandler, J. H., and Porfiri, K. (2001). "Monitoring river channel and flume surface with digital photogrammetry." *Journal of Hydraulic Engineering*, ASCE, 127(10), 871-877.

Larsen, E. (1995). "Mechanics and modeling of river meander migration." Ph.D. dissertation, University of California at Berkeley.

Leopold, L. B., and Wolman, M. G. (1960). "River meanders." *Geological Society America Bulletin*, U.S. Geological Society, vol. 71, New York, NY.

Li, Y. (2002). "Bridge pier scour and contraction scour in cohesive soils on the basis of flume tests." Ph.D. dissertation, Texas A&M University, College Station.

Meyer-Peter, E. and Muller, R. (1948), "Formulas for bedload transport." *Proceedings of the 2nd Meeting International Association of Hydraulic Research*, Stockholm, Sweden, 39-64.

Moody, L. F. (1944). "Friction factors for pipe flow." *Trans. Am. Soc. of Mech. Engrs.*, 66.

Mosselman, E. (1998). "Morphological modelling of rivers with erodible banks." *Hydrological Processes*, 12(8), 1357-1370.

Moura, L., and Kitney, R. (1991). "A direct method for least-squares circle fitting." *Computer Physics Communications*, 64, 57-63.

Munson, B. R., Young, D. F., and Okiishi, T. H. (1990). *Fundamentals of Fluid Mechanics*, Wiley, New York.

Nagata, N., Hosoda, T., Muramoto, Y., and Rahman, M. M. (1996). "Numerical analysis of unsteady open channel flows with channel processes." *Hydraulic Engineering Software VI. Sixth International Conference on Hydraulic Engineering Software, HYDROSOFT 96*, Penang, Malaysia, 233-42.

Nagata, N., Hosoda, T., and Muramoto, Y. (2000). "Numerical analysis of river channel processes with bank erosion." *Journal of Hydraulic Engineering*, ASCE, 126(4), 243-252.

Nakagawa, H., and Tsujimoto, T. (1980). "Sand bed instability due to bed load motion." *Journal of Hydraulic Division*, ASCE, 106(12), 2029-2051.

Nanson, G. C., and Hickin, E. J. (1983). "Channel migration and incision on the beatton river." *Journal Hydraulic Engineering*, ASCE, 109(3), 327-337.

Odgaard, A. J. (1986). "Meander flow model. I: Development." *Journal of Hydraulic Engineering*, ASCE, 112(12), 1117-1136.

Odgaard, A. J. (1987). "Streambank erosion along two rivers in Iowa." *Water Resources Research*, 23(7), 1225-1236.

Odgarrd, A. J. (1989a), "River-meander model. I: Development." *Journal of Hydraulic Engineering*, ASCE, 115(11), 1433-1450.

Odgarrd, A. J. (1989b), "River-meander model. II: Applications." *Journal of Hydraulic Engineering*, ASCE, 115(11), 1451-1464.

Olsen, N. R. B. (2003). "Three-dimensional CFD modeling of self-forming meandering channel." *Journal of Hydraulic Engineering*, 129(5), 366-372.

Park, N. (2006). "Prediction of meander migration." Ph.D. dissertation, Texas A&M University, College Station.

Park, S. (2001). "Measured and predicted meander migration near four highway crossings in Texas." M.S. thesis, Texas A&M University, College Station.

Parker, G., Sawai, K., and Ikeda, S. (1982). "Bend theory of river meanders. II. Nonlinear deformation of finite-amplitude bends." *Journal of Fluid Mechanics*, 115, 303-314.

Parker, G., and Andrews, E. D. (1985) "Sorting of bed load sediment by flow in meander bends." *Water Resources Research*, 21, 1361-1373.

Parker, G., and Wilcock, P. (1993) "Sediment feed and recirculating flumes: Fundamental difference." *Journal of Hydraulic Engineering*, ASCE, 119(11), 1192-1204.

Pizzuto, J. E. (1990). "Numerical simulation of gravel river widening." *Water Resources Research*, 26(9), 1971-1980.

Richardson, E. V., Simons, D. B., and Julien, P. Y. (1990). "Highways in the river environment." *Prepared for the Federal Highway Administration*, Washington, D.C. by the Department of Civil Engineering, Colorado State University, Fort Collins, CO.

Rodriguez, J. F., Bombardelli, F. A., Garcia, M. H., Frothingham, K. M., Rhoads, B. L., and Abad, J. D. (2004). "High-resolution numerical simulation of flow through a highly sinuous river reach." *Water Resources Management*, 18(3), 177-199.

Rouse, H. (1938). "Experiments on the mechanics of sediment suspension." *Proceedings of the 5th International Congress for Applied Mechanics*, Cambridge, MA, 550-554.

Seminara, G., Zolezzi, G., Tubino, M., and Zardi, D. (2001). "Downstream and upstream influence in river meandering. Part 2. Planimetric development." *Journal of Fluid Mechanics*, 438, 213-230.

Shen, H. W., Schumm, S. A., Nelson, J. D., Doehring, D. O., Skinner, M. M., and Smith, G. L. (1981). "Methods for assessment of stream-related hazards of highways and bridges." *Report no. FHWA-RD-80-160; FCP 35H1-062*. Colorado State University, Engineering Research Center, Fort Collins, CO.

Sun, T., Meakin, P., and Jossang, T. (2001a). "A computer model for meandering rivers with multiple bed load sediment sizes 1. Theory." *Water Resources Research*, 37(8), 2227-2241.

Sun, T., Meakin, P., and Jossang, T. (2001b). "A computer model for meandering rivers with multiple bed load sediment sizes 2. Computer simulations." *Water Resources Research*, 37(8), 2243-2258.

Van Rijn, L. C. (1989). "Sediment transport by currents and waves." *Report H461*, Delft Hydraulics, The Netherlands.

Yeh, P.-H. (2006). "Prediction of meander migration." Ph.D. dissertation, Texas A&M University, College Station.

Zolezzi, G., and Seminara, G. (2001). "Downstream and upstream influence in river meandering. Part 1. General theory and application overdeepening." *Journal of Fluid Mechanics*, 438, 183-211.

APPENDIX A

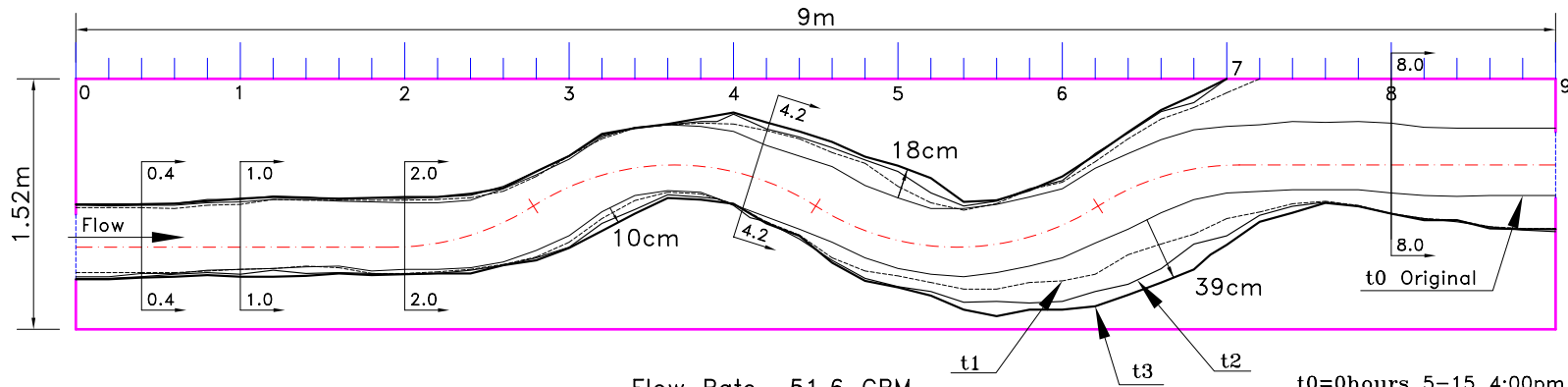
FLUME TESTS CONDUCTED BY THE AUTHOR IN THE OLD HYDRO LAB

Table A.1 List of flume tests done by the author in the old Hydro Lab

Test No.	R/W	Bend Angle	Flow Rate (GPM)
1	4	65°	51.6
2	4	65°	51.6
3	8	45.6°	80.0
4	8	45.6°	51.6
5	2	93.1°	51.6
6	1	136°	51.6
7	4	65°	51.6
8	8	45.6°	51.6
9	4	65°	26
10	4	65°	51.6
11	4	65°	64.8
12	2	65°	51.6
13	4	65°	50

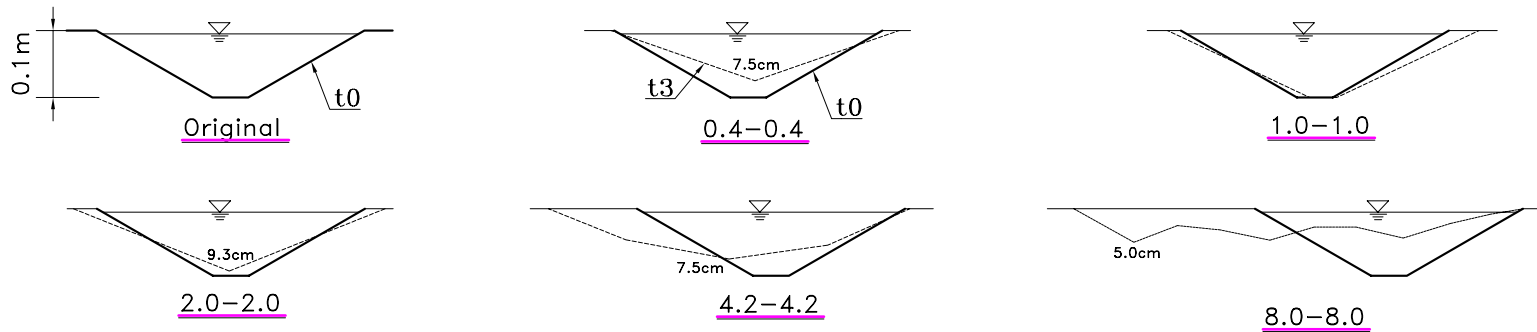
Migration of the Meander

Test 1 May 15 – May 17 2002



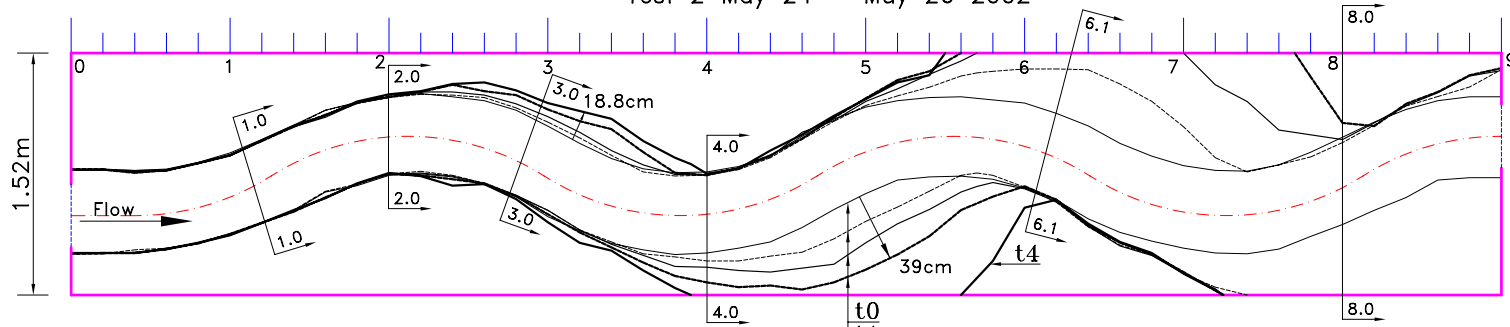
Flow Rate 51.6 GPM
 Top View (R/W=4 w=0.4m) Channel width=0.4m

t0=0hours 5-15 4:00pm
 t1=19 5-16 11:00am
 t2=31 5-16 11:00pm
 t3=48 5-17 4:00pm



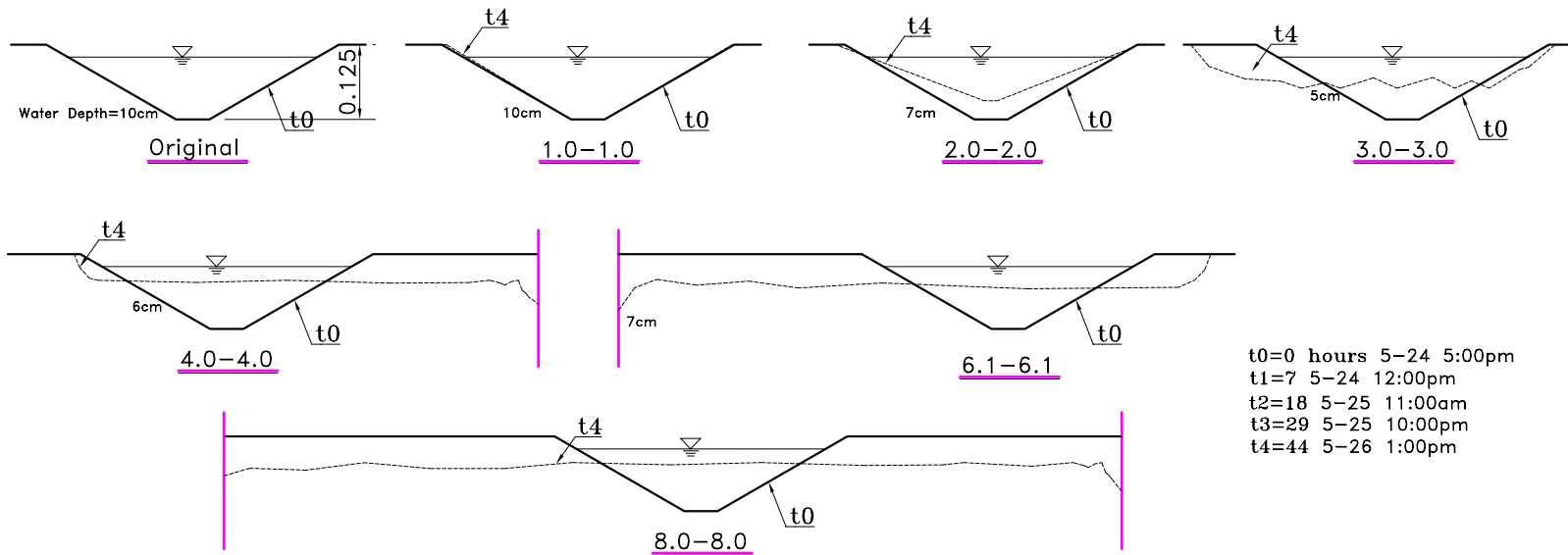
Migration of the Meander

Test 2 May 24 – May 26 2002



Flow Rate 51.6 GPM

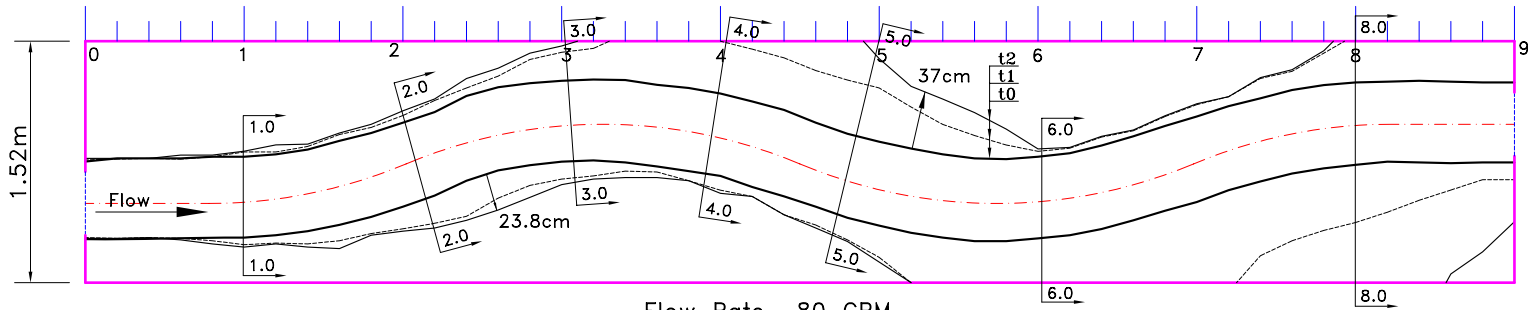
R=1.6m Water surface width= 0.4 m Channel width=0.5m



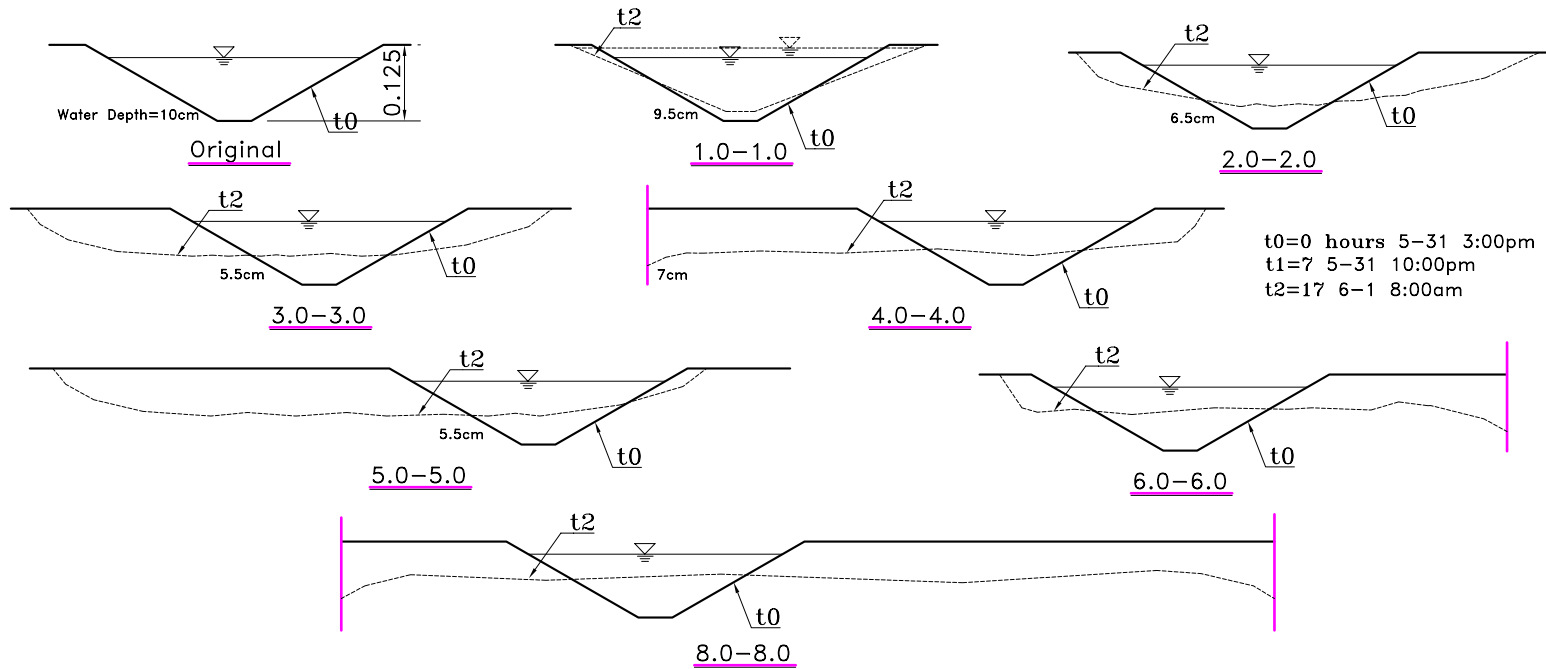
t0=0 hours 5-24 5:00pm
 t1=7 5-24 12:00pm
 t2=18 5-25 11:00am
 t3=29 5-25 10:00pm
 t4=44 5-26 1:00pm

Migration of the Meander

Test 3 May 31 - Jun 1 2002

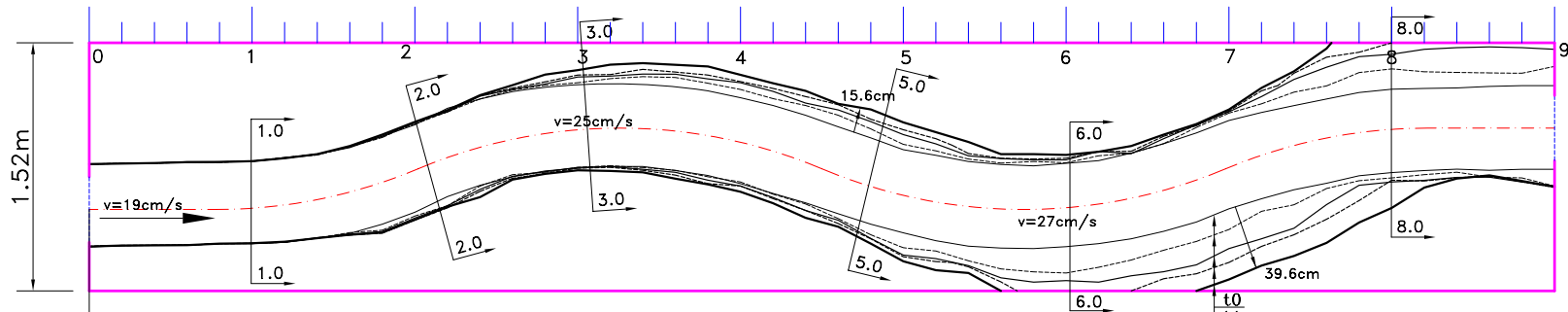


Flow Rate 80 GPM
 $R=3.2\text{m}$ Water surface width=0.4m $R/W=8$ Channel width=0.5m



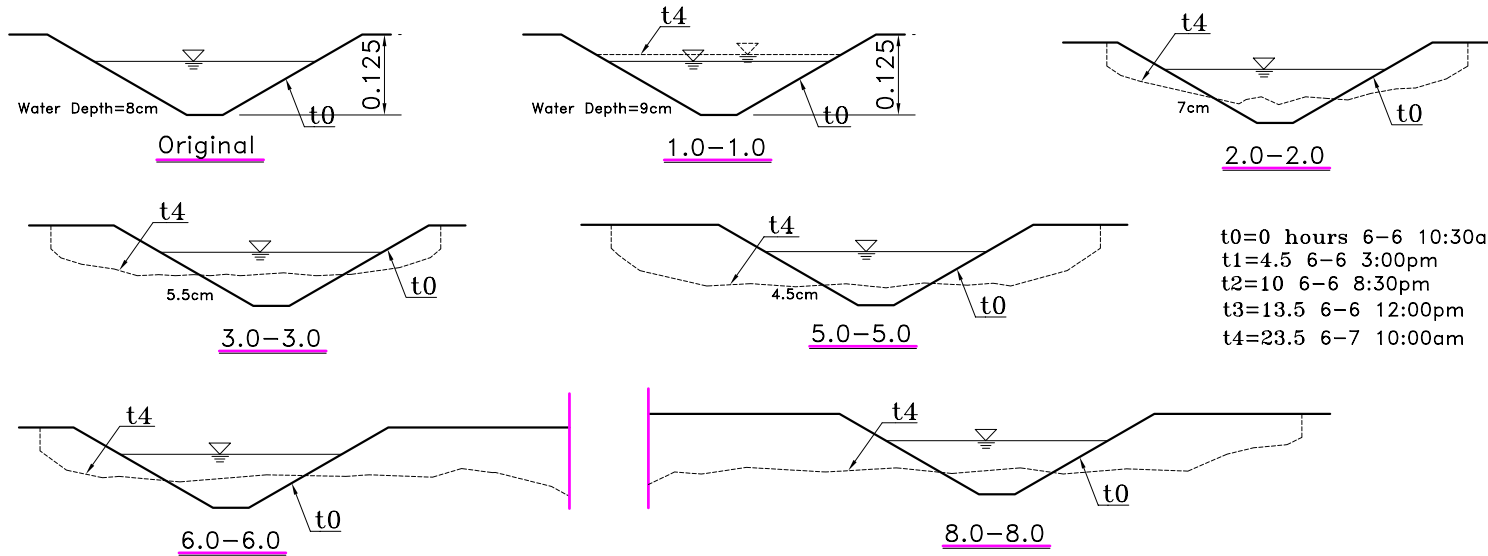
Migration of the Meander

Test 4 Jun 6 - Jun 7 2002



Flow Rate 51.6 GPM

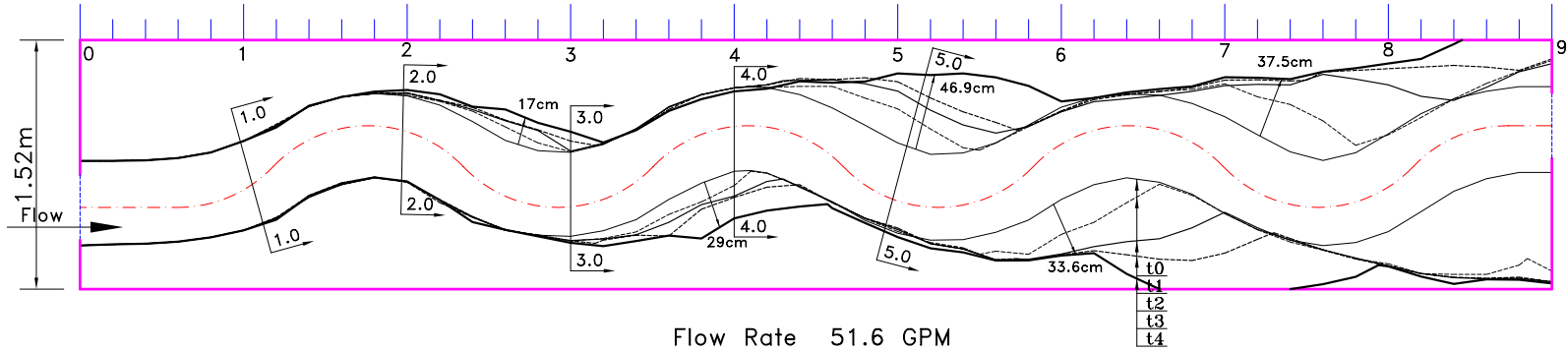
R=3.2m Water surface width=0.35m Channel width=0.5m



$t_0=0$ hours 6-6 10:30am
 $t_1=4.5$ 6-6 3:00pm
 $t_2=10$ 6-6 8:30pm
 $t_3=13.5$ 6-6 12:00pm
 $t_4=23.5$ 6-7 10:00am

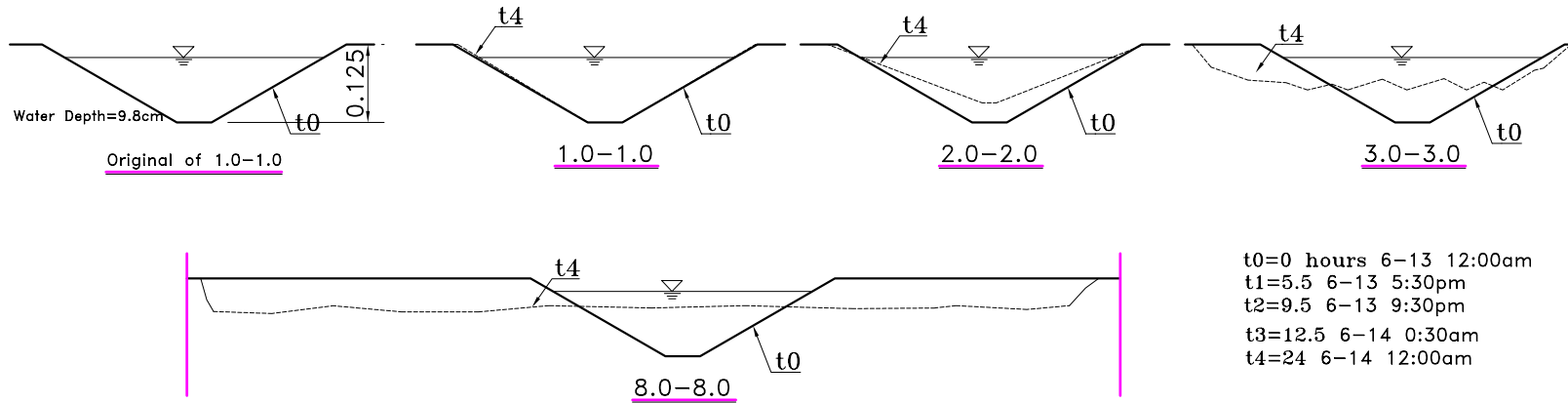
Migration of the Meander

Test 5 Jun 13-14 2002



Flow Rate 51.6 GPM

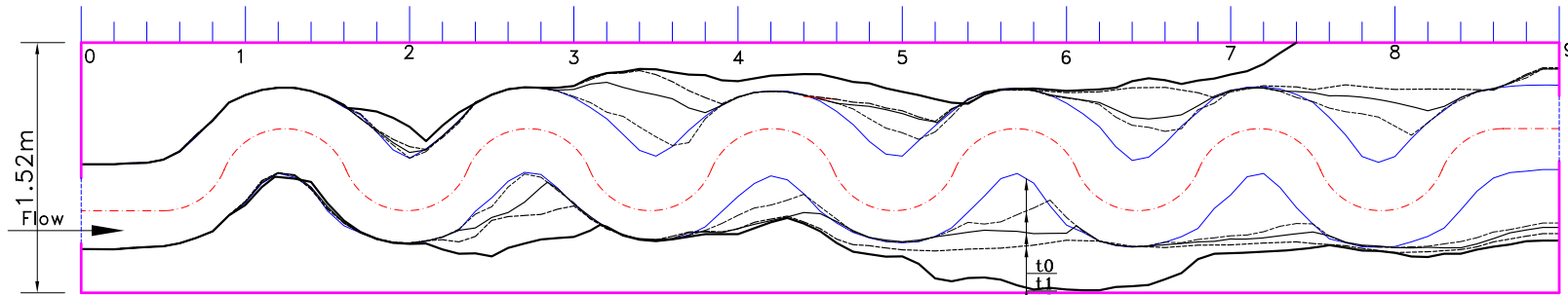
R=0.8m Water Surface Width=0.4m R/W=2 Channel width=0.5m



t0=0 hours 6-13 12:00am
 t1=5.5 6-13 5:30pm
 t2=9.5 6-13 9:30pm
 t3=12.5 6-14 0:30am
 t4=24 6-14 12:00am

Migration of the Meander

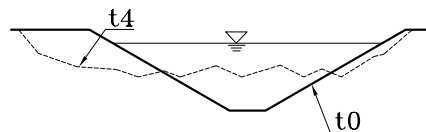
Test 6 July 4-6 2002



Flow Rate 51.6 GPM

R=0.4m water surface width=0.4m Channel width=0.5m

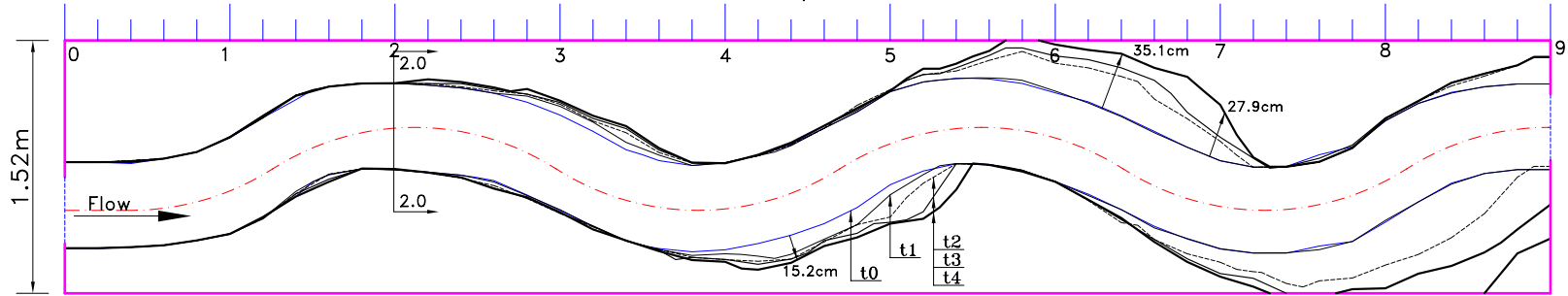
t0=0 hours
t1=2.0 hours
t2=5.2 hours
t3=12.4 hours
t4=35 hours



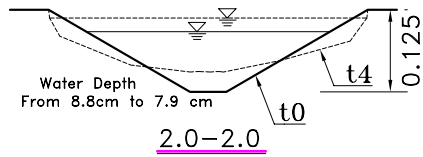
Typical Profile

Migration of the Meander

Test 7 July 11-13 2002



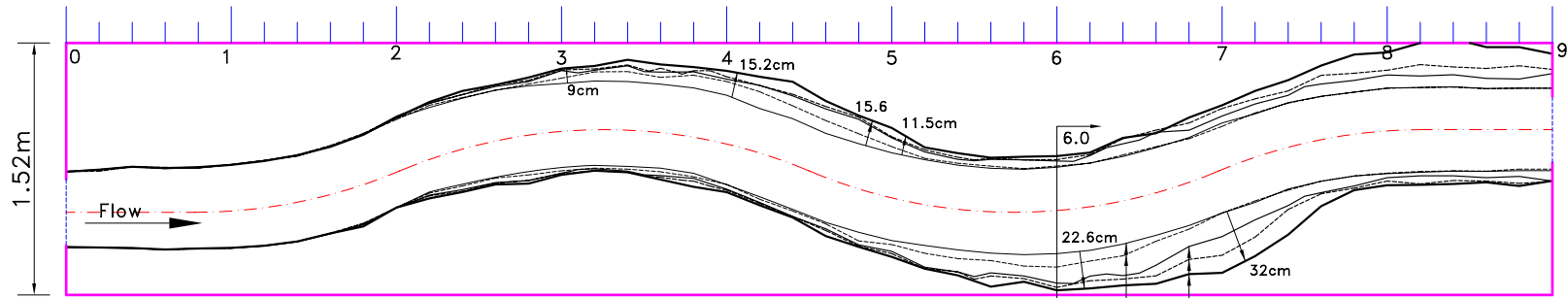
Flow Rate 51.6 GPM Weir height=3.4cm
 R=1.6m Channel width=0.5m



- t0=0 hours
- t1=4.2 hours
- t2=14.2 hours
- t3=22.8 hours
- t4=37.8 hours

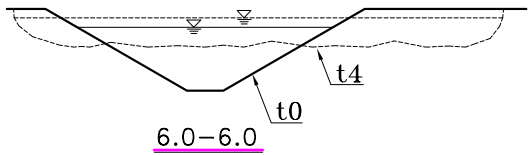
Migration of the Meander

Test 8 July 19–22 2002



Flow Rate 51.6 GPM Weir height=3.3cm

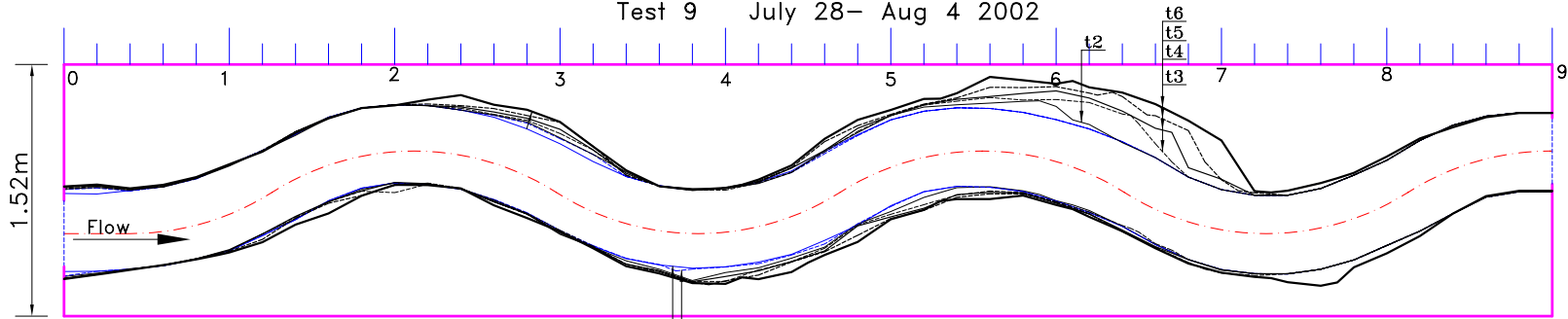
R=3.2m Water surface width= 0.4 m R/W=8 Channel width=0.5m



t0=0 hours
 t1=5.6 hours
 t2=15.2 hours
 t3=24.6 hours
 t4=50.9 hours

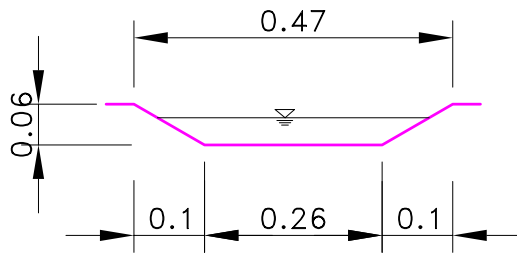
Migration of the Meander

Test 9 July 28– Aug 4 2002



Flow Rate 26 GPM Weir height= 0.0 cm

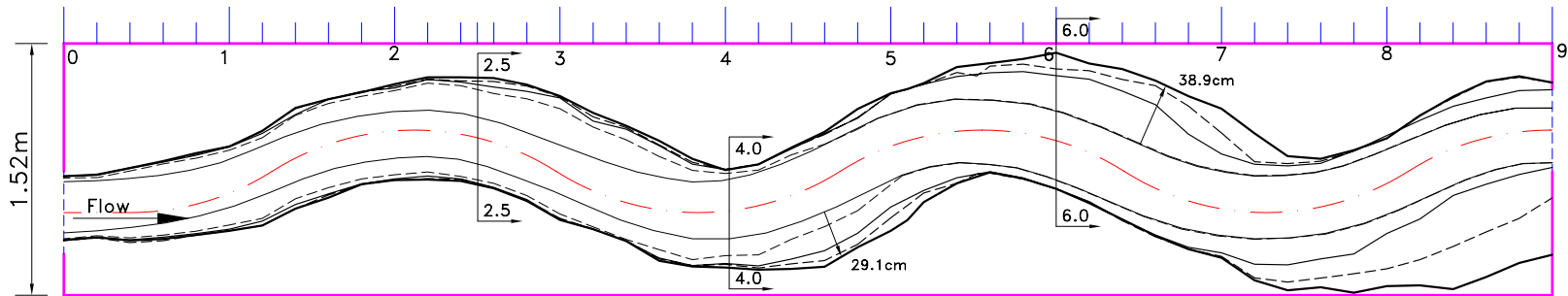
R=1.6m Channel width=0.47m



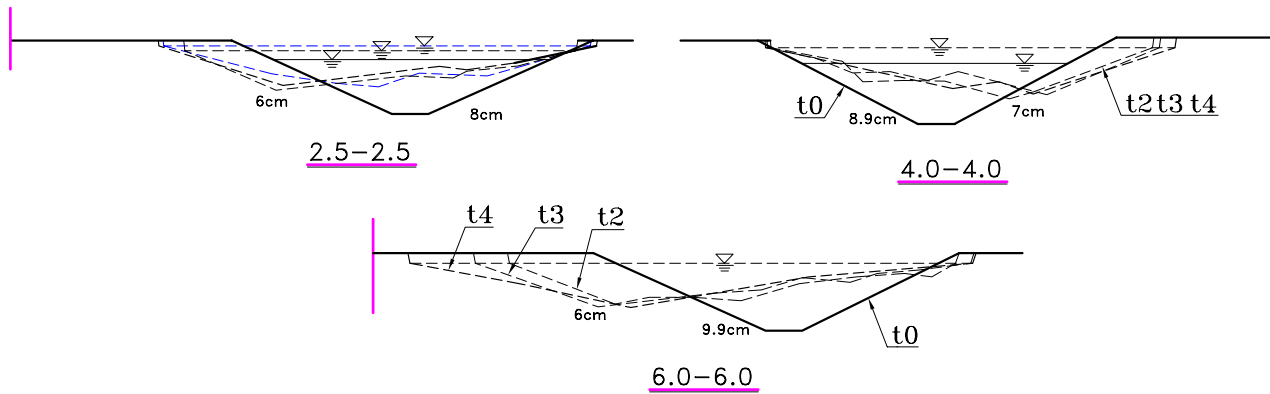
- t0=0 hours
- t1=10.3 hours
- t2=20 hours
- t3=32.8 hours
- t4=43 hours
- t5=66 hours
- t6=137.2 hours

Migration of the Meander

Test 10 Nov 9-11 2002



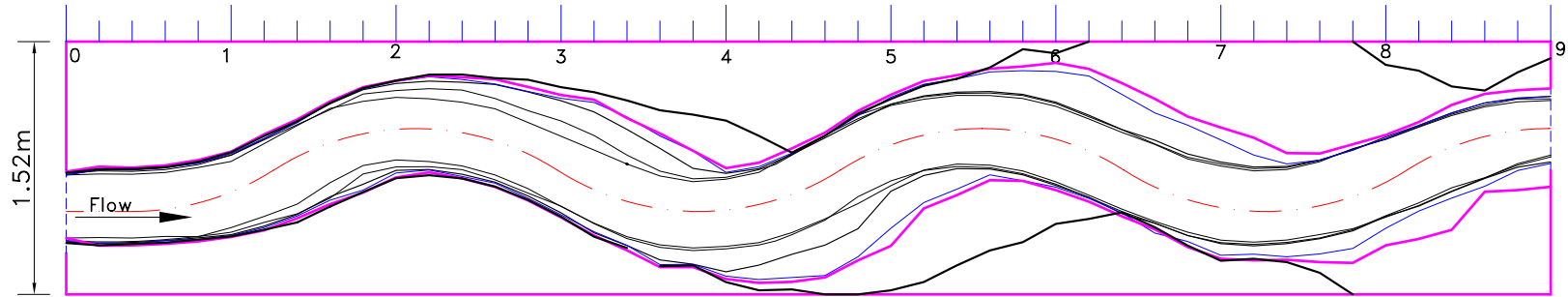
Flow Rate 51.6 GPM Weir height=0 cm
 R=1.6m Channel width=0.5m Water surface width is measured



t0=0 hours Nov 9 19:50
 t1=4.5 Nov 10 0:20
 t2=13.5 Nov 10 9:20
 t3=26.8 Nov 10 22:40
 t4=41 Nov 11 12:50

Migration of the Meander

Test 11 Nov 23-25 2002

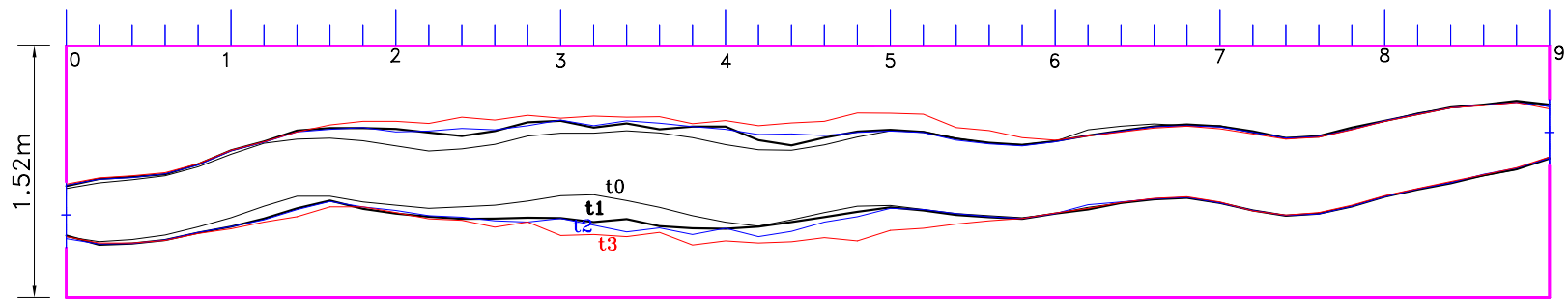


Flow Rate 64.8 GPM Weir height=0 cm
R=1.6m Channel width=0.5m

t0=0 hours Nov 23 17:20
t1=6 Nov 23 23:20
t2=22.3 Nov 24 15:40
t3=47.7 Nov 25 17:00
t4=77.7 Nov 26 23:00

Migration of the Meander

Test 12

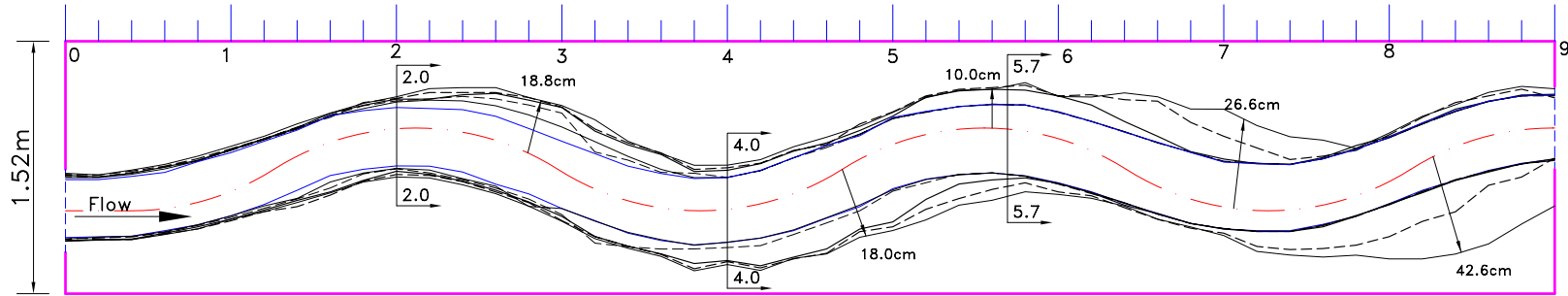


R/W=2 fei=65

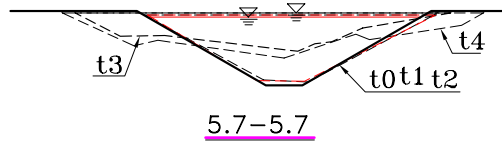
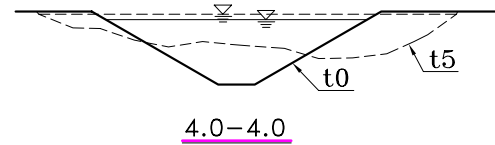
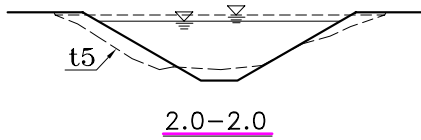
t0: Dec 26, 12:30 pm
t1: Dec 27, 10:00 am
t2: Dec 28, 11:25 am
t3: Dec 29, 8:25 am

Migration of the Meander

Test 13 Jan 6-23 2003



Flow Rate 50 GPM



- t0=0 hours
- t1=25 hours
- t2=46 hours
- t3=98 hours
- t4=194 hours
- t5=410 hours

APPENDIX B

A GALLERY OF PICTURES OF FLUME TESTS IN THE OLD HYDRO LAB



Figure B.1 Flume Test 2 at final stage, looking downstream



Figure B.2 Lost sand of Flume Test 2



Figure B.3 Wall is reached in Flume Test 3



Figure B.4 Smooth transition at the exit of Flume Test 3



Figure B.5 Flume Test 4 at final stage, looking downstream



Figure B.6 Flume Test 4 at final stage, looking upstream



Figure B.7 A section of Flume Test 5 at final stage



Figure B.8 Flume Test 5 at final stage, looking upstream



Figure B.9 Lost sand of Flume Test 5



Figure B.10 The exit of Flume Test 5



Figure B.11 Flume Test 6 is running, looking downstream



Figure B.12 Flume Test 6 at final stage, looking downstream



Figure B.13 Lost sand of Flume Test 6



Figure B.14 Flume Test 7 at final stage, looking upstream



Figure B.15 Flume Test 8 at final stage, looking downstream



Figure B.16 Flume Test 9 at final stage, looking upstream



Figure B.17 Flume Test 10 is running, looking upstream



Figure B.18 Flume Test 10 at final stage, looking upstream



Figure B.19 Flow transition at the exit of Flume Test 10



Figure B.20 Smooth flow transition at the exit of Flume Test 10



Figure B.21 Flume Test 11 with reinforcement at entry, looking downstream



Figure B.22 Flume Test 11 is running, looking upstream



Figure B.23 The end of Flume Test 12, looking downstream



Figure B.24 Flume Test 13 is running, looking downstream

APPENDIX C

SOURCE CODE OF THE COMPUTER PROGRAM

C.1 COMPUTER CODE WRITTEN IN MATLAB FOR GEOMETRY STUDY AND GRAPHIC OUTPUT

Table C.1 Dependency report of computer code for fitting circles

M Files	Children (Called Functions)	Parents (Calling Functions)
AutoFit_Run_InVC	ChannelLen	
	EvenlyDivide	
	AutoFit_R	
	AutoFit_R_Comb	
	AutoFit_RbySign	
AutoFit_R	ChannelLen	AutoFit_Run_InVC
	Rad_Cur_Curve	
	slope	
	findBendByCrtLine	
	sortIdx	
	plotCirRvsLen	
	fitcirlin	
	removeSmallBend	
AutoFit_R_Comb	AutoFit_R	AutoFit_Run_InVC
	plotCirRvsLen	
	ChannelLen	
	fitcirlin	
	removeSmallBend	
	shrinkArray	
AutoFit_RbySign	ChannelLen	AutoFit_Run_InVC
	Rad_Cur_Curve	
	slope	
	AutoFit_R	
	findBendByCrtLine	
	fitcirlin	
	removeSmallBend	
	plotCirRvsLen	
ChannelLen		AutoFit_Run_InVC
		AutoFit_R
		AutoFit_R_Comb
		AutoFit_RbySign
EvenlyDivide	ChannelLen	AutoFit_Run_InVC
findBendByCrtLine	straight_chk	AutoFit_R
		AutoFit_RbySign
fitcirlin		AutoFit_R
		AutoFit_R_Comb
		AutoFit_RbySign

Table C.1 Continued

M Files	Children (Called Functions)	Parents (Calling Functions)
Fixed_Spacing_Divide	ChannelLen	plotCirRvsLen
PlotCircle		plotCirRvsLen
plotCirRvsLen	PlotCircle	AutoFit_R
	Fixed_Spacing_Divide	AutoFit_R_Comb
	ChannelLen	AutoFit_RbySign
	Rad_Cur_Curve	
Rad_Cur_Cir	fitcirlin	
	Rad_Cur_Curve	
Rad_Cur_Curve	Rad_Cur_Sec	AutoFit_R
		AutoFit_RbySign
		plotCirRvsLen
Rad_Cur_Sec	rot_xy	Rad_Cur_Curve
removeSmallBend		AutoFit_R
		AutoFit_R_Comb
		AutoFit_RbySign
rot_xy		Rad_Cur_Sec
shrinkArray		AutoFit_R_Comb
slope		AutoFit_R
		AutoFit_RbySign
sortIdx		AutoFit_R
straight_chk		findBendByCrtLine

Table C.2 Dependency report of code for graphic output and utilities

M Files	Children (Called Functions)	Parents (Calling Functions)
GenRiverCoord		
GenRiverCoord_360deg		
GetAngle		
mergeVectors		
offsetCurve		PlotRiverCenOffset
PlotChenSimulationData	PlotChen_evDist	
PlotChen_evDist		PlotChenSimulationData
plotMgtForOneFlow		
PlotMvsT		
PlotRiver1Curve	read2Col3ColFile	PlotRiver2Banks
PlotRiver2Banks	PlotRiver1Curve	PlotRiverBanks_Main
	read2Col3ColFile	
PlotRiverBanks_Main	PlotRiverCenOffset	
	PlotRiver2Banks	
PlotRiverCenOffset	offsetCurve	
	read2Col3ColFile	
ReadMeanderCoord		
read2Col3ColFile		PlotRiver1Curve
		PlotRiver2Banks
		PlotRiverCenOffset

Index of Matlab Functions

function [xyRF,ArcIdx,nRet,curve]=AutoFit_Run_InVC(curve,arg)	225
function [xyRF,ArcIdx,Row]=AutoFit_R(xx,yy,m,arg,noxyRF,Row,bLast) ...	227
function [xyRF,ArcIdx,ArcIdx2]=AutoFit_R_Comb(xx,yy,m,arg)	230
function ar2=shrinkArray(ar,n)	233
function [xyRF,ArcIdx]=AutoFit_RbySign(xx,yy,m,arg)	234
function [len, Total_Len]=ChannelLen(x,y)	236
function [x1,y1]=EvenlyDivide(x,y,n)	236
function [ArcNum,ArcIdx]=findBendByCrtLine.....	237
function [x0,fval,fval2]=fitcirlin(x,y)	238
function [x1,y1]=Fixed_Spacing_Divide(x,y,spacing)	239
function PlotCircle(xyRF,LineSpec)	240
function plotCirRvsLen(xx,yy,m,xyRF,ArcIdx,arg)	241
function R=Rad_Cur_Cir(Curve,m,Norder,rotflag)	244
function R=Rad_Cur_Curve(xx,yy,m,Norder,rotflag)	245
function R=Rad_Cur_Sec(x,y,n,rotflag,index)	246
function [xyRF,ArcIdx]=removeSmallBend(xyRF,ArcIdx,len,arg)	247
function [x1,y1,sita1]=rot_xy(x,y,sita,translate)	248
function slp=slope(x,y,nPt)	249
function ArcIdx2=sortIdx(ArcIdx,n)	250
function DistAvg=straight_chk(x,y)	250
function GenRiverCoord()	251
function GenRiverCoord_360deg()	253
function [fei,sita]=GetAngle(x,y,xcyc)	254
function ar3=mergeVectors(ar1,ar2)	256
function cur2=offsetCurve(curl,dist)	257
Script PlotChenSimulationData.m.....	258
function f=PlotChen_evDist(x,miu)	259
function plotMgrtForOneFlow.....	260
function PlotMvsT(time,migration,xL,yL,sT)	262
function PlotRiver1Curve(fn,sTitle,bMov,bOverlap)	262
function PlotRiver2Banks(sT,fType,testNoStr)	266
function PlotRiverBanks_Main(sTitle,width)	270
function PlotRiverCenOffset(sTitle,width,fType,testNoStr)	271
function [cur,count,fid]=read2Col3ColFile(fn)	273
function ReadMeanderCoord()	274


```

function [xyRF,ArcIdx,nRet,curve]=AutoFit_Run_InVC(curve,arg)
% [xyRF,ArcIdx,sita,nArc]=AutoFit_Run_InVC(Curve,arg)
% 'curve' passed from Cpp has 2 rows, nPt columns;passed from
AutoFit_Run,
% it has nPt rows and 2 columns, transpose it.
% If the row number of 'curve' is not 2, transpose it.
% xyRF(1:ArcNum,1:4): xc,yc,Rc,Fei for a fitted bend.
% ArcIdx:stores the starting and ending point number of a curve section
% for which a circle is fitted.
% nArc: the number of circles fitted.
% ERROR CODE:
% nRet(4)==0: m>nPt/2,
% nRet(4)==-1: AutoFit_R(...) didn't produce any circle.
% nRet(4)==-2: In AutoFit_R_Comb(...), >=1 circle(s) is eaten by
preceding
% circle(s). Criterion line needs to be adjusted. Program can continue.

global nMaxArc;
nMaxArc=50;
nRet=zeros(10,1);
nPt=length(curve); % What if the curve has many elements not assigned a
number?

% Make curve 2 rows, nPt columns--as passed in by VC.
nRow=size(curve,1);
if nRow>2 % When called by AutoFit_Run(...), curve has nPt rows, 2
columns
    curve=curve';
end

%Output test XY data before evenly dividing the curve.
if abs(arg(19))>1e-5 % If arg(19)!=0, do it.
    fn0=num2str(arg(21));
    fn=['c:\temp\BankCoord_StepNo_' fn0 '.txt'];
    fid=fopen(fn,'wt');
    fprintf(fid,'%10d %10.3f %10.3f\n',[[1:nPt]' curve']');
    fclose(fid);
end

if arg(17)<=0 % Redistribute the points by spacingCoef.
    %width=37.7; %width=37.7; %for Guadalupe.dat let width=1, not
circle is fit,?
    %width=104; % width=104 for Brazos river.
    width=arg(1);
    spacingCoef=arg(2); % spacing=spacingCoef*width
    SameNum=false; % if 'true', use the same number as user input.
Then it will override spacingCoef.
    if spacingCoef<=0
        SameNum=true;
    end
    [len,totalLen]=ChannelLen(curve(1,:),curve(2,:));
    segLenCoef=arg(4); % for which a quadratic line will be fitted.

% Calculate nSpacing & m.
if SameNum==false
    spacing=spacingCoef*width;
    nSpacing=ceil(totalLen/spacing);
    m=round(segLenCoef/spacingCoef)+1;

```

```

    %Use same number of points as input. Recalculate
    spacing, spacingCoef, m.
    else % SameNum==true
        nSpacing=nPt-1; % # of spacing
        segLen=segLenCoef*width;
        spacing=totalLen/nSpacing;
        m=round(segLen/spacing)+1;
    end
else % Redistribute the points by a fixed number of spacing.
    nSpacing=arg(17)-1;
    m=arg(18);
end

if m>nSpacing/3.0
    xyRF=0;
    ArcIdx=0;
    nRet(4)=0; % ERROR CODE 0.
    return;
end

if abs(arg(11))>1e-5 % if arg(11)≠0, EvenlyDivide(Redistribute) it.
    [xx, yy]=EvenlyDivide(curve(1,:), curve(2,:), nSpacing);
    nPt=length(xx);
    curve=[xx;yy];
end

%arg(10): determine which function to use.
if abs(arg(10)-1)<1e-5 % if arg(10)==1
    [xyRF, ArcIdx]=AutoFit_R(curve(1,:), curve(2,:), m, arg);
elseif abs(arg(10)-2)<1e-5 % if arg(10)==2
    [xyRF, ArcIdx]=AutoFit_R_Comb(curve(1,:), curve(2,:), m, arg);
elseif abs(arg(10)-3)<1e-5
    [xyRF, ArcIdx]=AutoFit_RbySign(curve(1,:), curve(2,:), m, arg);
else
end

% Check Error Code
[n1, n2]=size(xyRF);
if (n1==1 && n2==1) || (n1==0 || n2==0)
    nArc=0;
elseif n1>=1 && n2>=1
    nArc=n1;
else
end
nRet(4)=ArcIdx(nArc+1, 1);

% arg(16), if 0, AutoFit_Run_InVC(...) is NOT called by VC; otherwise,
it's called VC.
% Transpose the matrix so that this work is not needed to be done in VC.
if abs(arg(16))>1e-4 % AutoFit_Run_InVC(...) is called by VC
    xyRF=xyRF';
    ArcIdx=ArcIdx';
end

nRet(1)=nPt; % Possibly to be used for the migrated channel.
nSpacing=nPt-1.
nRet(2)=nArc; % It stores ArcNum only and >=0.
nRet(3)=m; % Possibly to be used for the migrated channel.

```

```

function [xyRF,ArcIdx,Row]=AutoFit_R(xx,yy,m,arg,noxyRF,Row0,bLast)

% [xyRF,ArcIdx]=AutoFit_R(xx,yy,m,arg,noxyRF,Row0,bLast)
% xyRF(1:ArcNum,1:4): xc,yc,Rc,Fei for a fitted bend.
% Space for ArcIdx is allocated in findBendByCrtLine() and resized to
ArcIdx(1:ArcNum+1,1:2).
% Fei is ignored here and calculated in VC.
% In calling functions, ArcNum is determined by xyRF.
% Walking from the 1st point to the last point:
% Rc>0(y">0) if the center is to the left side of the channel;
% Rc<0(y"<0) if the center is to the right side.
% ArcIdx(ArcNum+1,2): the indices of two boundary points of a bend
% Element ArcIdx(ArcNum+1,1) carries Error Code.
% noxyRF, if true, don't calculate xyRF&sita. And xyRF=ArcNum.
% Leave the calculation of m outside of this function, or parallel
% functions will need the same code for calculating m.
% width, average width of the channel, used to calculate R/W for
function findBendByCrtLine()
% In AutoFit_Cur(), curve W/R vs. Len is used; here R(R/W) vs. Len is
used.
% minBenLen: a segment will be fitted with a circle only if its length
is larger than minBenLen
% Row0: When called by AutoFit_RbySign(), (xx,yy) may be a segment of
the
% whole channel. Pass along corresponding R/W.
% bLast: if true this segment contains the last point of the whole bank.
for findBendByCrtLine()
% If R goes up or down monotonously && bLast==true, treat it as a bend.
Default: true

%global nMaxArc;% len;
if nargin<=4
    noxyRF=false; % if true, don't calculate xyRF&sita
end
if nargin<=6, bLast=true; end

width=arg(1);
lmt(1:3)=arg(7:9);
%minBendLen=arg(3)*width;
minBendLen=[]; % or 0. Remove small bend length later.
distAvgLmt=arg(5);

Norder=2; % Default is 2.
rotflag='Y'; % Default is 'N' (No).
nPt=length(xx); % total number of points
%hlfm=floor(m/2);

len=ChannelLen(xx,yy); %len obtained here will be used.
if nargin<=5
    Row=Rad_Cur_Curve(xx,yy,m,Norder,rotflag); % Radius of curvature
    Row=Row./width; % R/W
else
    Row=Row0;
end
slp=slope(len,Row,3);

%Apply the 1st criterion line, pick sections below and intersect the
criterion line
%n1=hlfm; n2=nPt-hlfm; % Are the first and last hlfm points harmful?
n1=1;n2=nPt; % try to remove hlfm and consider all points

```

```

[ArcNum,ArcIdx]=findBendByCrtLine(xx(n1:n2),yy(n1:n2),RoW(n1:n2), ...
len(n1:n2)-len(n1),slp(n1:n2),lmt(1),minBendLen,distAvgLmt,bLast);

% Loops to pick sections below criteria lines.
% Intersection is not required.
for kk=2:3
    ArcNumOfPrevFit=ArcNum;    % Number of arcs obtained from previous
    criterion lines.
    for jj=1:ArcNumOfPrevFit+1 %ArcNumofPrevFit can be zero.
        if jj==1              %First pick up curve segments between
those that have been fitted with circles
            %first=hlfm;
            first=1;
            if ArcNumOfPrevFit==0
                %last=nPt-hlfm;
                last=nPt;
            else
                last=ArcIdx(jj,1)-1;
            end
            elseif jj==ArcNumOfPrevFit+1 % when ArcNumOfPrevFit=0, both
"if" and "elseif" hold, only "if" will be executed!
                first=ArcIdx(jj-1,2)+1;
                %last=nPt-hlfm;
                last=nPt;
            else
                first=ArcIdx(jj-1,2)+1;
                last=ArcIdx(jj,1)-1;
            end
            if last-first<3, continue; end
            if bLast
                if last<nPt, bLast=false; end
            end

            [AN2,AI2]=findBendByCrtLine(xx(first:last),yy(first:last), ...
RoW(first:last),len(first:last)-len(first),slp(first:last),lmt(kk), ...
minBendLen,distAvgLmt,bLast);
            for N=1:AN2 %AN2 is ArcNum2; AI2 is ArcIdx2
                AI2(N,:)=AI2(N,:)+first-1;
            end

            tmp=ArcNum;
            ArcNum=ArcNum+AN2; % Append newly fitted circles.
            ArcIdx(tmp+1:ArcNum,1:2)=AI2(1:AN2,1:2);
        end
        if ArcNum>0 % If ArcNum==0, a 0-by-2 matrix will be returned.
            ArcIdx=sortIdx(ArcIdx,ArcNum); % The dimension of ArcIdx is not
changed.
        end
    end
end

if ArcNum==0
    xyRF=0;
    ArcIdx=-1; % ERROR CODE -1, AutoFit_R(...) doesn't produce any
circle.
    if arg(12)>=1
        arg(12)=2; % Plot R/W vs Len only
        plotCirRvsLen(xx,yy,m,xyRF,ArcIdx,arg);
    end
    return;
end
end

```

```

ArcIdx=ArcIdx(1:ArcNum+1,:); % Shrink the array and reserve a slot for
Error Code.
xyRF=zeros(ArcNum,4);
ArcIdx(ArcNum+1,1)=ArcNum; % Error Code, normal situation.

for kk=1:ArcNum % Calculate radii of the circles.
    idx1=ArcIdx(kk,1); idx2=ArcIdx(kk,2);
    if noxyRF==false
        xyR=fitcirlin(xx(idx1:idx2),yy(idx1:idx2));
    else
        xyR=[0 0 1]; % The sign of R is needed for AutoFit_R_Comb(...)
    end
    bendLen=len(idx2)-len(idx1);

    % fei,sita are also calculated in C code. Here if for showing fei
    together with R.
    fei=bendLen/xyR(3)*180/pi;
    % fei=GetAngle(xx(idx1:idx2),yy(idx1:idx2),xyR); % Only the exact
    fei angle is calculated.

    idx=floor((idx1+idx2)/2); % Assign signs(+/-) to R. By default it
    is +.
    if Row(idx)<0, xyR(3)=-xyR(3); end
    xyRF(kk,:)=[xyR fei];
end

[xyRF,ArcIdx]=removeSmallBend(xyRF,ArcIdx,len,arg);

bPlot=arg(12);
if bPlot>0
    plotCirRvsLen(xx,yy,m,xyRF,ArcIdx,arg);
end

```

```

function [xyRF,ArcIdx,ArcIdx2]=AutoFit_R_Comb(xx,yy,m,arg)

% [xyRF,ArcIdx,ArcIdx2]=AutoFit_R_Comb(xx,yy,m,arg)
% First run AutoFit_R() to identify bends
% Then give a range for starting point and ending point and find the
% best circle based on a combination of ArcLength and miu, std
% ArcIdx(ArcNum,2): The actual boundary obtained
% ArcIdx2(ArcNum,4): The range searched; 2&3 inner; 1&4 outer.
% The number of circles will keep the same or go down.
% ArcIdx(ArcNum+1,1) is Error Code. If ErrorCode==-2, ArcIdx(ArcNum+1,2)
% is the # of circles eaten by preceding circles.

bHist=false; % Plot histogram of term1 & term2
bPlot=false; % Plot term1,term2 and sum for each bend
%byR=true; % alpha method, bPlot, x axis is R
byR=false; % true: x axis is Ri(1:n), Ri is the radii of circles
searched in order
% Ri normally goes up but may go up and down
% false: x axis is the ordinal No. of circles searched, [1:n]

% ext=0.40; % The range of the starting and ending point is +(-)ext
% *(length of the identified bend)
ext=arg(14);
tn=length(xx);

bPlotCirR=arg(12);
arg(12)=-1; % If <0, don't plot any curve.
% If =1, plot original channel and circles only.
% If =2, also plot curve R vs. Len.

noxyRF=true;
% xyRF0 tells the signs of the radii.
% ArcIdx has ArcNum rows.
[xyRF0,ArcIdx,RoW]=AutoFit_R(xx,yy,m,arg,noxyRF);
arg(12)=bPlotCirR;

nDim=size(xyRF0);
if nDim(1,1)==1 && nDim(1,2)==1 % ArcNum==0
    xyRF=0; % AutoFit_R(...) didn't produce any circle.
    if arg(12)>=1
        arg(12)=2; % Plot R/W vs Len only
        plotCirRvsLen(xx,yy,m,xyRF,ArcIdx,arg);
    end
    return;
end

ArcNum=nDim(1,1);
% Index of the circles eaten by the preceding circle.
missingCirIdx=zeros(ArcNum-1,1);
nMissingCir=0;

cLen=ChannelLen(xx,yy); % channel length, 1-D array

xyRF=zeros(ArcNum,4); % dynamic memory allocation, how to avoid this?
ArcIdx2=zeros(ArcNum,4);
for i=1:ArcNum
    len=ArcIdx(i,2)-ArcIdx(i,1);
    ext2=floor(ext*len);
    ArcIdx2(i,1)=floor(ArcIdx(i,1)-1.4*ext2);
    ArcIdx2(i,2)=ArcIdx(i,1)+ext2;

```

```

ArcIdx2(i,3)=ArcIdx(i,2)-ext2;
ArcIdx2(i,4)=floor(ArcIdx(i,2)+1.4*ext2);

% If the two adjacent bends are of opposite sign, divide at the
inflection point.
if i<ArcNum && ArcIdx2(i,4)>ArcIdx(i+1,1)
    if RoW(ArcIdx(i,2))*RoW(ArcIdx(i+1,1))<0
        k=ArcIdx(i,2);
        while RoW(k)*RoW(ArcIdx(i+1,1))<0 && k<=ArcIdx(i+1,1)
            k=k+1;
        end
        ArcIdx2(i,4)=k-1;
    end
end
end

if ArcIdx2(1,1)<1, ArcIdx2(1,1)=1; end
if ArcIdx2(ArcNum,4)>tn, ArcIdx2(ArcNum,4)=tn; end

ArcIdx=zeros(ArcNum+1,2);
ArcIdx(ArcNum+1,1)=ArcNum; % Error Code, normal situation.

a=1;
% Reducing b will lead to larger bend angle
b=arg(15); % Important coefficient, balancing bend angle and fitting
error
%dist=zeros(500,1);
if bPlot==true
    term1=zeros(100,1);
    term2=zeros(100,1);
    Ri=zeros(100,1);
end
if bHist==true
    term1Hist=zeros(1000,1);
    term2Hist=zeros(1000,1);
end
nHist=0;

for i=1:ArcNum
    % beta=0; % to find maximum value
    beta=1e7; % to find minimum value
    j=ArcIdx2(i,2)+1;
    k=ArcIdx2(i,3)-1;
    n=0;
    nMinAlpha=0;
    AI1=ArcIdx2(i,1); % 1st outer boundary.
    AI4=ArcIdx2(i,4); % last outer boundary.
    % At most one point on the already fitted bend will be used.
    if i>1 && AI1<ArcIdx(i-1,2), AI1=ArcIdx(i-1,2); end % overlap check

    while true
        j=j-1; % j goes from 1st inner boundary to 1st outer boundary.
        k=k+1; % k goes from 2nd inner boundary to 2nd outer boundary.
        if j<AI1 && k>AI4, break; end
        if j<AI1, j=AI1; end
        if k>AI4, k=AI4; end % This step is important.

        n=n+1; nHist=nHist+1;
        [xyR,fval]=fitcirlin(xx(j:k),yy(j:k));%fval is sum of squares.
    end
end

```

```

t1=a*(2*pi*xyR(3))/(cLen(k)-cLen(j)); % alpha
fval=sqrt(fval/(k-j+1));
t2=b*(fval/xyR(3)); % alpha
if bPlot==true
    term1(n)=t1;
    term2(n)=t2;
    Ri(n)=xyR(3);
end
if bHist==true
    term1Hist(nHist)=t1;
    term2Hist(nHist)=t2/b;
end

sum=t1+t2;
% if beta<sum % Find maximum value
if beta>sum % Find minimum value
    beta=sum;
    xyRF(i,1:3)=xyR;
    ArcIdx(i,1)=j; ArcIdx(i,2)=k;%There should be overlap check.
    nMinAlpha=n;
end
end
% Plot alpha(beta) vs. R(# of circles), term1 term2 sum
if bPlot==true
    figure
    if byR==true
        plot(Ri(1:n),term1(1:n),'b*- ',Ri(1:n),term2(1:n),'k.- ',
Ri(1:n),term1(1:n)+term2(1:n),'gd-')
        hold on
    end
    plot(Ri(nMinAlpha),term1(nMinAlpha)+term2(nMinAlpha),'ro','MarkerSize',
8)
    xlabel('R','FontSize',15)
else % by # of circles
    Nv=[1:n];
    plot(Nv,term1(1:n),'b*- ',Nv,term2(1:n),'k.- ', Nv,
term1(1:n)+term2(1:n),'gd-')
    hold on
end
plot(nMinAlpha,term1(nMinAlpha)+term2(nMinAlpha),'ro','MarkerSize',8)
xlabel('Ordinal Number of Circles','FontSize',15)
end
ylabel('Target Term','FontSize',15)
legend('1/\phi','b\times(Err/R)','Sum')
tstr=['Bend ' num2str(i) ' b=' num2str(b)];
title(tstr,'FontSize',15)
grid on
end
end

if bHist==true
    Hist1=shrinkArray(term1Hist,nHist);
    Hist2=shrinkArray(term2Hist,nHist);
    figure
    hist(Hist1,50)
    figure
    hist(Hist2,50)
end

```



```

for i=1:ArcNum
    if xyRF0(i,3)<0, xyRF(i,3)=-xyRF(i,3);end
end

% Check whether a circle is eaten by the preceding circle.
for i=1:ArcNum
    if ArcIdx(i,1)==0 || ArcIdx(i,2)==0
        nMissingCir=nMissingCir+1;
        missingCirIdx(nMissingCir)=i;
    end
end

% Error Code -2:A circle is eaten by the preceding circle in
% AutoFit_R Comb. A criterion line needs to be adjusted.
if nMissingCir>0
    xyRF1=zeros(ArcNum-nMissingCir,4);
    ArcIdx1=zeros(ArcNum-nMissingCir+1,2);
    idx=0;
    for i=1:ArcNum % Remove the circles eaten
        if ArcIdx(i,1)~=0 && ArcIdx(i,2)~=0
            idx=idx+1;
            xyRF1(idx,:)=xyRF(i,:);
            ArcIdx1(idx,:)=ArcIdx(i,:);
        end
    end
    ArcNum=ArcNum-nMissingCir;
    xyRF=xyRF1;
    ArcIdx=ArcIdx1;
    ArcIdx(ArcNum+1,1)=-2; % Error Code
    ArcIdx(ArcNum+1,2)=nMissingCir; % Number of circles eaten
end

% Calculate fei angle for showing it in fitted circles.
for i=1:ArcNum
    idx1=ArcIdx(i,1); idx2=ArcIdx(i,2);
    fei=(cLen(idx2)-cLen(idx1))/abs(xyRF(i,3))*180/pi;
    xyRF(i,4)=fei;
end

[xyRF,ArcIdx]=removeSmallBend(xyRF,ArcIdx,cLen,arg);

if bPlotCirR>0
    plotCirRvsLen(xx,yy,m,xyRF,ArcIdx,arg);
end

function ar2=shrinkArray(ar,n)

% keep the first n elements of 1-D array ar
len=length(ar);
if len<=n
    ar2=ar;
    return
end
ar2=zeros(n,1);
for i=1:n
    ar2(i)=ar(i);
end

```

```

function [xyRF,ArcIdx]=AutoFit_RbySign(xx,yy,m,arg)

% [xyRF,ArcIdx]=AutoFit_RbySign(xx,yy,m,arg)
% Designed for flume test geometry whose bends share inflection points.
% When the sign of R changes, it's considered a switch to next bend.
% The first and last bend are identified by criterion lines.
% This method is not good for a single 360-degree circle.
global nMaxArc;
ArcIdx=zeros(nMaxArc,2);%The whole dimension(nMaxArc*2)will be returned.

width=arg(1);
%lmt(1:3)=arg(7:9);
minBendLen=arg(3)*width;
distAvgLmt=arg(5);

Norder=2;      % Default is 2.
rotflag='Y';   % Default is 'N'(No).
nPt=length(xx); % total number of points

len=ChannellLen(xx,yy); %len obtained here will be used.
RoW=Rad_Cur_Curve(xx,yy,m,Norder,rotflag); % Radius of curvature

RoW=RoW./width; % R/W
slp=slope(len,RoW,3);

% Use criterion lines to find the first bend.
i=2;
% If the curve is a perfect 360-degree circle, there is a problem here.
while RoW(1)*RoW(i)>0 % It's possible RoW(i)<0 (or >0) for i=[1 nPt]
    if i==nPt
        disp('In AutoFit_RbySign(), RoW doesn't change sign.')
        break;
    end
    i=i+1;
end % It's possible range identified here doesn't contain a bend.
bPlot=arg(12);
arg(12)=-1; % Don't plot anything.
bLast=false; % This segment doesn't contain the last point of the bank.
[xyR1,AI1]=AutoFit_R(xx(1:i),yy(1:i),m,arg,true,RoW(1:i),bLast);
[n1,m1]=size(xyR1);
if n1==1 && m1==1 % n1: # of bends identified. Most likely just 1 bend.
    n1=0; % No bend was identified.
else
    AI1(n1,2)=i; % Directly use inflection point as boundary point.
end
first=i;

for j=1:n1, ArcIdx(j,:)=AI1(j,:); end

i=nPt-1;
while RoW(i)*RoW(nPt)>0
    i=i-1;
end

[xyR2,AI2]=AutoFit_R(xx(i:nPt),yy(i:nPt),m,arg,true,RoW(i:nPt));
arg(12)=bPlot; % Restore original value.

[n2,m2]=size(xyR2);
if n2==1 && m2==1 % No circle was fitted.

```

```

        n2=0;
    else
        AI2(1,1)=1+1; %If there are more than one, extend backward the
        boundary of the 1st one.
    end
    for k=1:n2, AI2(k,:)=AI2(k,:)+i-1; end
    last=i+1;

    % The point on which R changes sign is treated as shared boundary point.
    ArcNum=n1;
    i=first;
    maxLmt=max([arg(7) arg(8) arg(9)]);

    while i<last
        j=i+1;
        while RoW(i)*RoW(j)>0
            j=j+1;
        end
        minBendLen=[]; % Ignore this criterion. For all bends?
        [AN3,AI3]=findBendByCrtLine(xx(i:j),yy(i:j),RoW(i:j),len(i:j)-
        len(i),...
            slp(i:j),maxLmt,minBendLen,distAvgLmt,false); %arg(9) is the
        largest criterion line.
        %use maxLmt*20,so that straight line segments(R/W>1e4...)can be removed.
        % 20 may not be the best number.
        if AN3>0
            ArcNum=ArcNum+1;
            ArcIdx(ArcNum,1)=i;
            ArcIdx(ArcNum,2)=j;%AI3(AN3,2)+i-1;%If the two share an
            inflection point ArcIdx(ArcNum,2)=j.
        end
        i=j;
    end
    ArcIdx(ArcNum+1:ArcNum+n2,:)=AI2(1:n2,:);

    ArcNum=ArcNum+n2;
    ArcIdx(ArcNum+1,1)=ArcNum;
    ArcIdx=ArcIdx(1:ArcNum+1,:);
    xyRF=zeros(ArcNum,4);

    for kk=1:ArcNum % Calculate radii of the circles.
        idx1=ArcIdx(kk,1); idx2=ArcIdx(kk,2);
        xyR=fitcirlin(xx(idx1:idx2),yy(idx1:idx2));
        bendLen=len(idx2)-len(idx1);
        % fei,sita are also calculated in C code. Here if for showing fei
        together with R.
        fei=bendLen/xyR(3)*180/pi;

        idx=floor((idx1+idx2)/2); % Assign signs(+/-) to R. By default it
        is +.
        if RoW(idx)<0, xyR(3)=-xyR(3); end
        xyRF(kk,:)=[xyR fei];
    end

    [xyRF,ArcIdx]=removeSmallBend(xyRF,ArcIdx,len,arg);
    bPlot=arg(12);
    if bPlot>0
        plotCirRvsLen(xx,yy,m,xyRF,ArcIdx,arg);
    end
end

```

```

function [len, Total_Len]=ChannelLen(x,y)

% function [len,Total_Len]=ChannelLen(x,y)
% Given coordinates of a bank, calculate the curve length of a channel
% len is a vector. len(i) is corresponding to the i-th point.
% Total_Len is the total length of the curve.
% There is a C version of this function.

n=length(x);
len=zeros(n,1);
for i=2:n % Avoid exponential function.
    v=sqrt((x(i)-x(i-1))*(x(i)-x(i-1))+(y(i)-y(i-1))*(y(i)-y(i-1)));
    len(i)=v+len(i-1);
end
Total_Len=len(n);

function [x1,y1]=EvenlyDivide(x,y,n)

% Evenly divide a curve described by x,y into n segments
% The length of each segment (total length of the channel)/n is
measured
% along the CURVE and is not the distance between adjacent two points
of
% x1,y1 which is a straight line distance.
% x1,y1 have the same vector direction as x,y(row/column).
% There are n+1 points for the result
% If x is a row/column vector, the result is of the same direction.
% There is a C version of this function
tn=length(x);

len=ChannelLen(x,y); %element: 1 to tn
seclen=len(tn)/n;

nRow=size(x,1);
if nRow==tn
    x1=zeros(n+1,1); y1=zeros(n+1,1);
else
    x1=zeros(1,n+1); y1=zeros(1,n+1);
end

x1(1)=x(1);    y1(1)=y(1);
x1(n+1)=x(tn); y1(n+1)=y(tn);

len_sum=0;
for i=2:n    %From the 2nd point to the n-th point to be solved.
    len_sum=len_sum+seclen;
    for j=1:tn-1 %From the 1st point to the (tn-1)-th of the original
curve.
        if len_sum<len(j)
            continue
        end
        if len_sum<len(j+1) % len_sum belongs to [len(j) len(j+1))
            slope=(len_sum-len(j))/(len(j+1)-len(j));
            x1(i)=x(j)+(x(j+1)-x(j))*slope;
            y1(i)=y(j)+(y(j+1)-y(j))*slope;
            break
        end
    end
end
end

```

```

function [ArcNum,ArcIdx]=findBendByCrtLine
(xx,yy,RC,len,slp,lmt,minBendLen,distAvgLmt,bLast)

% [ArcNum,ArcIdx]=findBendByCrtLine(xx,yy,RC,len,slp,lmt,minBendLen,
distAvgLmt,bLast)
% Identify bends by a criterion line. The portion of cruve RC vs.Length
% between lmt and X axis is considered a bend.
% xx,yy: coordinates of the section of interest, for straight_chk(...)
% Here are the differences from previous version.
% RC: R or R/W or d^2R/ds^2, with positive and negative values
% len(1:length(xx)): length vector of the segment.
% slp:slope, Make sure the identified segment doesn't go up/down
monotonously
% lmt: the criterion line, scalar
% minBendLen: if the length of a curve segment is smaller than this, no
% circle will be fit.
% distAvgLmt: a measure of straightness, obtained from function
straight_chk()
% If the user don't want to use these two criteria, set
minBendLen&distAvgLmt to <=0
% bLast: if true this segment contains the last point of the whole bank.
% If R goes up or down monotonously && bLast==true, treat it as a bend.
If not present: true
% Pass a lot of parameters so that they don't need to be calculated
again.

global nMaxArc;

if isempty(minBendLen), minBendLen=0; end % Ignore this criterion.
if isempty(distAvgLmt), distAvgLmt=0; end

if nargin<=8, bLast=true; end % Default value for bLast.

tn=length(xx);
ArcIdx=zeros(nMaxArc,2); % The whole dimension(nMaxArc*2) will be
returned.
ArcNum=0;
i=1;
while i<=tn
    if abs(RC(i))>lmt
        i=i+1;
        continue;
    end
    j=i; i=i+1;
    while i<=tn
        if abs(RC(i))<=lmt && RC(i)*RC(j)>0 % When R changes sign, a
bend ends
            i=i+1; % For d^2R/ds^2, change of
sign doesn't matter.
            continue
        end
        break
    end % To this step there must be RC(i)>lmt and RC(i-1)<=lmt.

    if len(i-1)-len(j)<minBendLen || i-j<4 % at least 4 points below
criterion line
        continue
    end

```

```

    sign=0;
    for k=j+1:i-1 % Make sure the identified segment doesn't go
up/down monotonously (*)
        if slp(j)*slp(k)<0
            sign=-1;
            break
        end
    end
    % If i-1==tn(the last point of the whole bank), ignore this rule
(*) .
    % (1:tn) can be only a small segment of this bank, then...
    if sign==0 && (i-1<tn || ~bLast),continue,end;

    distAvg=straight_chk(xx(j:i-1),yy(j:i-1));
    if abs(distAvg)<distAvgLmt
        continue
    end

    ArcNum=ArcNum+1;
    ArcIdx(ArcNum,1)=j; ArcIdx(ArcNum,2)=i-1;
End

```

```
function [x0,fval,fval2]=fitcirlin(x,y)
```

```

%Make the problem a linear least square one
%General circle function:  $x^2+y^2-2ax-2by-c=0$ 
%or:  $2ax+2by+c=x^2+y^2$ 
%xc=a, yc=b, r=sqrt(c+a^2+b^2)
%x y can be row or column vectors
%x0=[xc yc r], fval=sum(squares), fval2=sum(|R'-R|)/lenX
%Construct over-determined equations  $ax=b$ . Solve  $(a'a)x=a'b$ 

lenX=size(x);
if lenX(1)==1 % x,y are row vectors
    x=x'; y=y';
    n=lenX(2);
elseif lenX(2)==1 % x,y are column vectors
    n=lenX(1);
else
    x0=[inf inf inf];
end
a=[2*x 2*y ones(n,1)];
b=x.^2+y.^2;

ap=a';
A=ap*a; % a'*a doesn't work after compiling in R13.
B=ap*b;

coef=A\B;
x0(1)=coef(1); x0(2)=coef(2); % x0 is a row vector by default.
x0(3)=sqrt(x0(1)^2+x0(2)^2+coef(3));
if nargout==1
    return
end

Rp=sqrt((x-x0(1)).^2+(y-x0(2)).^2);
fval=sum((Rp-x0(3)).^2);
if nargout<=2

```

```

    return
end

fval2=sum(abs(Rp-x0(3)))/lenX;
er=fval2/x0(3);

if x0(3)==-Inf
    x0(3)=-1e7;
elseif x0(3)==Inf
    x0(3)=1e7;
elseif er>0.2 % Even coef=A\B has a solution, if er is too large,
consider the fit a failure
    x0(3)=1e7*(x0(3)/abs(x0(3)));
end

function [x1,y1]=Fixed_Spacing_Divide(x,y,spacing)

%Given a curve, evenly divide it according to a constant spacing.
%Dividing points (x1,y1) will be returned.

if spacing<=0
    x1=x;
    y1=y;
    return;
end

tn=length(x);
len=ChannelLen(x,y); %element: 1 to tn

n=floor(len(tn)/spacing)+1; %number of points that will be used to
divide the curve
x1=zeros(n,1); y1=zeros(n,1);
x1(1)=x(1); y1(1)=y(1);

len_sum=0;
for i=2:n
    len_sum=len_sum+spacing;

    for j=1:tn-1
        if len_sum<len(j)
            continue
        end
        if len_sum>len(j) & len_sum<len(j+1)
            slope=(len_sum-len(j))/(len(j+1)-len(j));
            x1(i)=slope*(x(j+1)-x(j))+x(j);
            y1(i)=slope*(y(j+1)-y(j))+y(j);
        end
    end
end
end

```

```

function PlotCircle(xyRF,LineStyle)

% xyRF can have 3 elements: [xc yc R] or 4 [xc yc R fei]
% Given xc yc R (fei), plot the circle and show the radius (fei angle)

if nargin==1
    LineSpec=['k-'];
end

xc=xyRF(1);
yc=xyRF(2);
R=xyRF(3);
fei=-500;
if length(xyRF)==4
    fei=xyRF(4);
end

sita=linspace(0,2*pi,121);
x=R*cos(sita)+xc;
y=R*sin(sita)+yc;
hold on
plot(x,y,LineStyle)

if R<10
    fmt='%3.1f';
else
    fmt='%4.0f';
end
rStr=['R=' num2str(R,fmt)];
if fei>-361
    fStr=['\phi=' num2str(fei,'%3.0f') '\circ'];
    Str={rStr,fStr};
else
    Str=rStr;
end

hold on
text(xc,yc,Str,'HorizontalAlignment','center')

```



```

function plotCirRvsLen(xx,yy,m,xyRF,ArcIdx, arg)

% plotCirRvsLen.m, function version of Plot_Cir_RvsLen_Script.m
% Can be called by AutoFit_Run(), AutoFit_R(), AutoFit_R_Comb()
% xyRF has 4 columns, the last column is fei angle.

bPlot=arg(12); % If <=0, don't plot any curve and return, otherwise,
plot curve R vs. Len.
                % If =1, plot original channel and circles only.
                % If =2, plot curve R vs. Len only.
                % If =3, plot both.

if bPlot<=0
    return;
end

wid=arg(1);
spacingCoef=arg(2);
minBendLenCoef=arg(3);
segLenCoef=arg(4);
scrsz=get(0, 'ScreenSize'); % Let the plot fill the whole screen

timeStr=['Time step=' num2str(arg(20)) ' hour(s) ' 'Current step No.='
num2str(arg(21))];
str0=['# of points m=' num2str(m) ' Width=' num2str(wid) ' Spacing='
num2str(spacingCoef,2) 'W SegLen='...
num2str(segLenCoef, '%2.1f') 'W'];
str=[str0 ' MBL=' num2str(minBendLenCoef, '%2.1f') 'W'];

nPt=length(xx);
if bPlot==1 || bPlot==3
    ArcNum=size(xyRF,1);
    % Identify two boundary points of each circle.
    xbd=zeros(ArcNum*2,1);
    ybd=zeros(ArcNum*2,1);

    for i=1:ArcNum % Find x, y coordinates of the boundaries
        idx1=ArcIdx(i,1); idx2=ArcIdx(i,2);
        xbd(2*i-1)=xx(idx1); xbd(2*i)=xx(idx2);
        ybd(2*i-1)=yy(idx1); ybd(2*i)=yy(idx2);
    end

    %*****
    %plot the river bank, distance marker, the circles and their boundaries
    %*****
    figure('Position',scrsz);
    % Plot different segments with different colors.
    linSpec=cell(7,1);
    linSpec(1)=cellstr('r--');
    linSpec(2)=cellstr('g--');
    linSpec(3)=cellstr('b--');
    linSpec(4)=cellstr('c--');
    linSpec(5)=cellstr('m--');
    linSpec(6)=cellstr('y--');
    linSpec(7)=cellstr('k--');

    n2=nPt+1;
    for i=1:ArcNum
        n1=ArcIdx(i,1); n2=ArcIdx(i,2);
        if i==1, start=1;

```

```

else
    start=ArcIdx(i-1,2);
end

if start<n1
    plot(xx(start:n1),yy(start:n1),char(linSpec(7)))
    hold on
end
num=rem(i,6);
if num==0, num=6; end
plot(xx(n1:n2),yy(n1:n2),char(linSpec(num)))
hold on
end
if n2<=nPt % if ArcNum==0, the clause will not be executed.
    plot(xx(n2:nPt),yy(n2:nPt),char(linSpec(7)))
end

for i=1:ArcNum
    PlotCircle(xyRF(i,1:4),'b-') % If it's 1:3, fei angle will not
be shown.
end

hold on
plot(xbd,ybd,'*','MarkerSize',6)
axis equal
hold on

%plot distance for each marker
tickspacing=arg(6);
if tickspacing>0
    [markX, markY]=Fixed_Spacing_Divide(xx,yy,tickspacing);
    if tickspacing>0
        plot(markX,markY,'o','MarkerSize',2.5) % Plot marker points
    end
    tickLabel=arg(13); % If 0, don't plot marker point distance,
otherwise, plot it.

    if abs(tickLabel)>1e-5 % Not 0, show distance of the marker
points.
        leng=length(markX);
        %txt=zeros(len,length(num2str(tickspacing))+3);
        txt=cell(leng,1);
        for i=1:leng
            % txt(1,:)=strcpy(txt(1,:),num2str(i*tickspacing));
            s=num2str((i-1)*tickspacing);
            s=[' ' s]; % add a space before the number. It seems
doesn't work.
            txt(i)=cellstr(s);
        end
        text(markX,markY,txt,'FontSize',8)
    end

    vx=min(xx); vy=min(yy)+tickspacing/2;
    hold on
    plot([vx vx+tickspacing],[vy vy],'LineWidth',2)
    tickstr=sprintf('%d m',tickspacing);
    text(vx,vy+tickspacing/10,tickstr)
end

```

```

        xlabel('X','FontSize',15)
        ylabel('Y','FontSize',15)
    %   str='Original Channel and Fitted Circles';
    if arg(20)>0
        tleStr={str,timeStr}
    else
        tleStr=str;
    end
    title(tleStr,'FontSize',13)%,'FontWeight','bold')
    %   grid on

end

%*****
%plot R/W vs. Len and criterion lines
%*****
if bPlot~=2 && bPlot~=3
    return;
end
len=ChannelLen(xx,yy);
Norder=2; rotflag='Y';
R=Rad_Cur_Curve(xx,yy,m,Norder,rotflag);

figure('Position',scrsz);
plot([0 0.01],[0 0],'.b-', 'MarkerSize',6) % For generating legend.
plot([0 0.01],[0 0], 'r-.', 'LineWidth',1.4)

plot(len,R./wid, '.b-', 'MarkerSize',6)
hold on
xLen=[0 len(nPt)]; % Plot criterion lines
yR=[arg(7) arg(7);-arg(7) -arg(7);arg(8) arg(8);-arg(8) -arg(8);arg(9)
arg(9);-arg(9) -arg(9)];
plot(xLen,yR, 'r-.', 'LineWidth',1.4)
grid on

ylim([-20 20])
xlabel('Channel Lengthwise Distance','FontSize',15)
ylabel('R / W','FontSize',15)
legend('R / W','Criterion Lines')

if bPlot==2 % When xyRF & ArcIdx are not available.
    title(str,'FontSize',13)
    return;
end

% lenbd & Rbd/Rcbd are for the boundaries on the R/W vs. Len curve.
% lenbd:X, Rbd/Rcbd:Y
lenbd=zeros(ArcNum*2,1); % for R/W?
% R radius of curvature, bd-boundary, for identifying points on R/W
% vs. Len curve.
Rbd=zeros(ArcNum*2,1);
Rcbd=zeros(ArcNum*2,1); % Rc radius of circle,

for i=1:ArcNum % Find x, y coordinates of the boundaries
    idx1=ArcIdx(i,1); idx2=ArcIdx(i,2);
    lenbd(2*i-1)=len(idx1);
    lenbd(2*i)=len(idx2); % Find X coordinates on the R/W vs.Len curve

    Rbd(2*i-1)=R(idx1);

```

```

        Rbd(2*i)=R(idx2);           % Rbd:  radius of curvature

        Rcbd(2*i-1)=xyRF(i,3);
        Rcbd(2*i)=Rcbd(2*i-1);    % Rcbd:  radius of circles
    end

    plot(lenbd,Rbd./wid,'*r','MarkerSize',6)

    set(gca,'XMinorGrid','On') % Turns on X minor grid only.
    % grid minor % It turns on the minor grids for both X & Y.

    for j=1:ArcNum
        hold on
        plot(lenbd(2*j-1:2*j),Rcbd(2*j-1:2*j)./wid,'m-.')
    end

    if arg(20)>0
        tleStr={str,timeStr}
    else
        tleStr=str;
    end
    title(tleStr,'FontSize',13)

function R=Rad_Cur_Cir(Curve,m,Norder,rotflag)

    % Calculate radius of curvature by fitting a circle for each segment as
    % opposed to Rad_Cur_Curve() which fits a parabolic line for each
    % segment.
    % Use the result of Rad_Cur_Curve() to determine the sign of R and
    % where no
    % circle can be fitted.

    if nargin<4
        rotflag='';
    end
    a=rem(m,2); %remainder of integer m.
    if a==0
        m=m+1;
    end

    CurSize=length(Curve); %Number of points on the curve.
    if CurSize<=2
        ErrorReport=['There is only 1 or 2 points on the cuve?']
        %   Msgbox('There is only 1 or 2 points on the cuve?') % Give it
        % up for compiling purpose.
    end
    R=zeros(CurSize,1);

    hlfm=floor(m/2); %m=2*hlfm+1
    idx1st=hlfm+1; %First point to be evaluated.
    idxLst=CurSize-hlfm; %Last point to be evaluated.

    R=zeros(CurSize,1);
    index=hlfm+1;
    for i=idx1st:idxLst
        Lbound=i-hlfm; Rbound=i+hlfm;
        Section=Curve(Lbound:Rbound,:);
        if i==idx1st

```

```

        Rt=fitcirlin(Section(:,1),Section(:,2));
        R(1:i)=Rt(3);
    elseif i==idxLst
        Rt=fitcirlin(Section(:,1),Section(:,2));
        R(i:CurSize)=Rt(3);
    else
        Rt=fitcirlin(Section(:,1),Section(:,2));
        R(i)=Rt(3);
    end
end

R1=Rad_Cur_Curve(Curve,m,Norder,rotflag);
for i=1:CurSize
    if R1(i)<0, R(i)=-R(i); end
    if abs(R(i))>1e6, R(i)=R1(i); end % If no circle can be fit, use
that of Rad_Cur_Curve()
end

function R=Rad_Cur_Curve(xx, yy, m, Norder, rotflag)

% R=Rad_Cur_Curve(xx,yy, m, Norder, rotflag)
% xx,yy can be row/column vectors.
% R, radius of each point, column vector
% m: all fittings are done on m points. m is an odd number. If not, use
m+1.
% Default value for Norder is 2.
% Fit MANY Norder-th order polynomial curves for a series of points
%which are divided into many segments. One quadratic line is fitted for
one segment.
% The R value for the center point of each segment is kept.
% Use moving quadratic fit. m points are fitted a time.
% If 'rotflag' is present(='Y'), the coordinate system is required to
be rotated.
% If 'rotflag' is not present, no rotation will be performed.
% X' axis is in the direction of linearly fitted line. Y' axis points
to the positive direction of Y axis.
% For the first m/2 and last m/2 points, the "R" of corresponding
segments are used.

if nargin<=4
    rotflag='';
end
if nargin<=3
    Norder=2;
end
a=rem(m,2); %remainder of integer m.
if a==0
    m=m+1;
end
CurSize=length(xx); %Number of points on the curve.
if CurSize<=2
    ErrorReport=['There is only 1 or 2 points on the cuve?, --
Rad_Cur_Curve(...)']
    %   MsgBox('There is only 1 or 2 points on the cuve?') % Give it
up for compiling purpose.
end

% hlfm=int16(m/2); %int16() cut everything after the decimal point.
% No math operations except for sum are defined for int16 etc.

```

```

hlfm=floor(m/2);      %m=2*hlfm+1
idx1st=hlfm+1;      %First point to be evaluated.
idxLst=CurSize-hlfm; %Last point to be evaluated.

R=zeros(CurSize,1);
index=hlfm+1;
for i=idx1st:idxLst
    Lbound=i-hlfm; Rbound=i+hlfm;
    x=xx(Lbound:Rbound); y=yy(Lbound:Rbound);
    if i==idx1st
        Rt=Rad_Cur_Sec(x, y, Norder, rotflag);
        R(1:i)=Rt(1:i);
    elseif i==idxLst
        Rt=Rad_Cur_Sec(x, y, Norder, rotflag);
        R(i:CurSize)=Rt(index:m);
    else
        R(i)=Rad_Cur_Sec(x, y, Norder, rotflag, index);
    end
end

function R=Rad_Cur_Sec(x,y,n,rotflag,index)

% R=Rad_Cur_Sec(x,y,n,rotflag,index)
% Fit ONE n-th order polynomial curve and calculate radius of curvature
for
% a group of points.
% x,y can be row/column vector.
% If the 4th input argument 'index' is present, R of a certain point is
% calculated(R is a scalar), or R for all points are calculated(R is a
vector).
% If 'rotflag' is not empty and ='Y', the coordinate system will be
rotated.
% Fit a line to these points. Then rotate x axis to this line.

%x=section(:,1); y=section(:,2); % To save memory, don't use it.
% (nargin>=3) && (upper(rotflag)=='Y') doesn't work why?
if (nargin>=4) & (upper(rotflag)=='Y')
    translate='Y'; %If rotation is not required, no translation will
be done.
    %For quadratic fit, translation doesn't make a difference for R &
Curvature.

    [x,y]=rot_xy(x,y,[],translate);
end

a=polyfit(x,y,n);
ap=polyder(a);    app=polyder(ap);
ypp=polyval(ap,x); ypp=polyval(app,x);

if nargin==3 || nargin==4    %R for each point is required.
    R=(1+ypp.^2).^ (1.5) ./ypp; % What if ypp(i)==0, how to effectively
find all of them out?
elseif nargin==5
    if ypp(index)==0
        ypp(index)=1e-40;
    end
    R=(1+ypp(index)^2)^ (1.5) /ypp(index);    %Only one R referred by
'index' is required.
else

```

```

        ErrorReport=['The number of input arguments is wrong for function
"Rad_Cur_Sec() "']
%   MsgBox('The number of input arguments is wrong for function
"Rad_Cur_Sec() "')
        %only when nargin==1
End

function [xyRF,ArcIdx]=removeSmallBend(xyRF,ArcIdx,len,arg)

% Remove bends with small bend length or fei angle
% Call this function after all possible bends are identified.

RowLarge=40; % If the fitted R/W of the last bend > RowLarge, discard
this bend.

width=arg(1);
ArcNum=size(xyRF,1);

minBendLen=arg(3)*width;
nSmallBend=0;
smallBendIdx=zeros(ArcNum,1);
for kk=1:ArcNum % Identify bends with small fei or bend length
    idx1=ArcIdx(kk,1); idx2=ArcIdx(kk,2);
    bendLen=len(idx2)-len(idx1);

    % Allow the last bend to have small length
    if (bendLen<minBendLen || abs(xyRF(kk,4))<arg(22)) && kk<ArcNum
        nSmallBend=nSmallBend+1;
        smallBendIdx(nSmallBend)=kk;
    end
    if kk==ArcNum && xyRF(kk,3)/arg(1)>RowLarge
        nSmallBend=nSmallBend+1;
        smallBendIdx(nSmallBend)=kk;
    end
    % if abs(xyRF(kk,4))<arg(22) && kk<ArcNum % Allow the last bend to
have small fei angle
    %     nSmallFei=nSmallFei+1;
    %     smallFeiIdx(nSmallFei)=kk;
    %     end
end
if nSmallBend==0, return; end

xyRF1=zeros(ArcNum-nSmallBend,4);
ArcIdx1=zeros(ArcNum-nSmallBend+1,2);
nLongBend=0;
for i=1:ArcNum
    small=false;
    for j=1:nSmallBend
        if i==smallBendIdx(j)
            small=true;
            break;
        end
    end
    if ~small
        nLongBend=nLongBend+1;
        xyRF1(nLongBend,:)=xyRF(i,:);
        ArcIdx1(nLongBend,:)=ArcIdx(i,:);
    end
end
end

```

```

ArcIdx1(nLongBend+1,:) = ArcIdx(ArcNum+1,:);
ArcIdx1(nLongBend+1,1) = nLongBend; % New ArcNum

xyRF=xyRF1;
ArcIdx=ArcIdx1;

function [x1,y1,sita1]=rot_xy(x,y,sita,translate)

% [x1,y1]=rot_xy(x,y)
% [x1,y1]=rot_xy(x,y,sita)
% [x1,y1]=rot_xy(x,y,sita,translate)
% [x1,y1,sita1]=rot_xy(...)
% If 'sita' is not present, fit a line to these points first, then
% rotate x axis to this line.
% The positive direction of the new X axis is from points of smaller
% number to larger number
% x y can be row or column vectors. If they are row vectors, 1-D
% transpose
% operation can be avoided.
% If 'sita' is present, this angle is used. And sita1=sita.
% x1 y1 are coordinates in this new coordinate system. Vector
% direction(row/column) is the same as x y.
% sita1, unit: radian, is the rotation angle of the axes (-pi pi].
% sita, a scalar unit: radian (-pi pi], if specified, rotate axis
% system by this angle.
% translate, ='Y' translate origin to point(mean(x), mean(y)); ='N' or
% not present do nothing.

len=length(x);

if len==size(x,1)
    rc='c'; x=x'; y=y'; %Make x y row vectors
else
    rc='r';
end

if nargin==2 || (nargin>2 && isempty(sita)) % sita can be '[]'.
% ws=warning; warning off warning(ws);

%An important step. Rotate the axes first.
alpha=atan2(y(len)-y(1), x(len)-x(1)); % (-pi, pi]
RotMat=[cos(alpha) sin(alpha); -sin(alpha) cos(alpha)];

xryr=RotMat*[x; y];
xr=xryr(1,:); yr=xryr(2,:);
% polyfit(x,y,1) may produce unpredictable results when alpha==(+/-
)pi/2.
a=polyfit(xr,yr,1);
if a(1)==inf | a(1)==-inf %If those points form an almost perfect
vertical line.
    sita=pi/2; %The sign will be determined based on
alpha
    disp('A rare situation occurred in function rot_xy(). The
fitted line segment is vertical after the 1st rotation')
else
    sita=atan(a(1)); % [-pi/2 pi/2] Angle of the fitted line
with respect to the rotated axes(by angle of alpha)
end

```



```

        sita=alpha+sita;    %The angle by which the original axes need to be
rotated.
end

if nargin>3 && translate=='Y'
    x0=mean(x);    %This is the origin of the new coordinate system.
    y0=mean(y);
else
    x0=0; y0=0;
end

RotMat=[cos(sita) sin(sita);-sin(sita) cos(sita)]; %Rotation matrix
xnyn=RotMat*[x-x0; y-y0];    %Rotate from original coordinates to new
coordinates
%The 1st row of xnyn is x vector

if rc=='c'
    x1=xnyn(1,:); y1=xnyn(2,:);
else
    x1=xnyn(1,:); y1=xnyn(2,:);
end

if nargout>2
    sital=sita;
end

function slp=slope(x,y,nPt)

% slp=slope(x,y,nPt)
% x,y are coordinates of the curve
% nPt points will be used to calculate the slope in the middle. nPt>=2
% Calculate slope for a curve given by x,y
% First fit a line for nPt points, use the slope of this line.
% If nPt is an odd number, the slope is for the point in the middle.
% If nPt is an even number, the slope is for point No. nPt/2.

tn=length(x);    % total number

if mod(nPt,2)==1
    oe=1;    % Odd or even number. Odd number.
else
    oe=0;    % Even number
end

slp=zeros(tn,1);
nhlf=floor(nPt/2);
for i=nhlf+oe:tn-nhlf
    xv=x(i-(nhlf-(1-oe)):i+nhlf);
    yv=y(i-(nhlf-(1-oe)):i+nhlf);
    p=polyfit(xv,yv,1);
    slp(i)=p(1);
end

for i=1:nhlf+oe-1
    slp(i)=slp(nhlf+oe);
end
for i=tn-nhlf+1:tn

```

```

        slp(i)=slp(tn-nhlf);
end

function ArcIdx2=sortIdx(ArcIdx,n)

% Sort ArcIdx so that the indices of point number are in ascending
order.
% n is the ArcNum
% ArcIdx2 is of the same size of ArcIdx.
% Don't shrink ArcIdx. Or it will be dynamically expanded in next loop
of the calling function.
% This function is called by AutoFit_R().
nRow=length(ArcIdx(:,1));

idx=zeros(n,1); % Not necessarily efficient.
ArcIdx2=zeros(nRow,2); % Only the size of ArcIdx is wanted here.

for i=1:n
    idx(i)=i;
end

for i=1:n-1
    for j=i:n
        if ArcIdx(idx(j),1)<ArcIdx(idx(i),1)
            v=idx(j); idx(j)=idx(i); idx(i)=v;
        end
    end
end

for k=1:n
    ArcIdx2(k,:)=ArcIdx(idx(k),:);
end

function DistAvg=straight_chk(x,y)

% DistAvg=straight_ckh(x,y)
% to be called by AutoFit_2ndDerv(), findBendByCrtLine()
% check how straight a curve segment is.
% draw a line from the first point to the last point
% rotate x axis to this line
% get the summation of distance of each point to this line
% if the point is below the line, the distance is <0

len=length(x);
alpha=atan2(y(len)-y(1), x(len)-x(1)); % (-pi, pi]
RotMat=[cos(alpha) sin(alpha); -sin(alpha) cos(alpha)];

if len==size(x,1)
    x=x'; % convert it to row vector
    y=y';
end

x0=(x(1)+x(len))/2;
y0=(y(1)+y(len))/2;

xryr=RotMat*[x-x0; y-y0];

```

```
xr=xryr(1,:); yr=xryr(2,:);
DistAvg=sum(yr)/len;
```

```
function GenRiverCoord()
```

```
% Given R, W, fei, # of bends, generate coordinates of center line,
right and left banks
% The first and last bends have a bend angle of fei/2.
% The X axis(Y=0) passes through all inflection points

caseNo=82; % Generate filenames for the 3 files: C##R_Initial.txt
C##L_Initial.txt, C##C_Initial.txt
sNo=num2str(caseNo);
if caseNo<=9, sNo=['00' sNo];
elseif caseNo<=99, sNo=['0' sNo]; end

myPath=['M:\_WW\Meander\Data\'];
%myPath=['c:\temp\'];

R=120;
W=40;
fei=180; % In degrees here
nBend=7;
RoW=R/W;
spacingCoef=0.1; % Use this to determine nPerBend.
nPerBend=-100; % # of segments on a bend. If nPerBend<=0, use
spacingCoef, or use nPerBend.
LoW=1; % The straight line preceding and after the bends, length/width

sRoWF=['_RoW' num2str(RoW) 'F' num2str(fei) 'B' num2str(nBend)];
fei=fei*pi/180; % convert fei to radians

sRoWF=[]; % Remove this information, so that the file name can be
easily handled in VC & PlotRiver
fnR=['P' sNo 'R' sRoWF '_Initial.txt']; % P stands for parametric study
fnC=['P' sNo 'C' sRoWF '_Initial.txt'];
fnL=['P' sNo 'L' sRoWF '_Initial.txt'];
fn={fnR;fnC;fnL};

if nPerBend<=0
    spacing=spacingCoef*W;
    nPerBend=round(R*fei/spacing);
else
    spacingCoef=R*fei/nPerBend/W;
    spacing=spacingCoef*W;
end
if rem(nPerBend,2)==1 % Make it an even number
    nPerBend=nPerBend+1;
end
straightNo=round(LoW/spacingCoef); % # of segments on the two straight
line segments
straightLen=straightNo*spacing;

nPt=straightNo*2+(nBend-1)*nPerBend+1;
xy=zeros(nPt,6); % Right column 1&2; Center column 3&4; Left column
5&6.

% Calculate coordinates of the centers
```

```

Xc=zeros(nBend,1); Yc=zeros(nBend,1);
Xc(1)=straightLen; Yc(1)=R*cos(fei/2);
for i=2:nBend
    if rem(i,2)==0, Yc(i)=-Yc(1);
    else, Yc(i)=Yc(1); end
    Xc(i)=Xc(1)+2*R*sin(fei/2)*(i-1);
end

% Coordinates of the preceding straight line segments
y1=Yc(1)-R;
xlin=0:spacing:(straightLen-spacing); % Horizontal line
xy(1:straightNo,1:2:5)=[xlin',xlin',xlin']; % The tangent point is not
included here.
xy(1:straightNo,2)=y1-W/2;
xy(1:straightNo,4)=y1;
xy(1:straightNo,6)=y1+W/2;

% Range: sita=[0 fei]
sita1=linspace(3*pi/2,3*pi/2+fei/2,nPerBend/2+1); % The first bend
with bend angle of fei/2
sita2=linspace(pi/2+fei/2,pi/2-fei/2,nPerBend+1); % The 2k-th bend
(k=1,2,3,...)
sita3=linspace(3*pi/2-fei/2,3*pi/2+fei/2,nPerBend+1); % The (2k+1)-th
bend (k=1,2,3,...)
sita4=linspace(3*pi/2-fei/2,3*pi/2,nPerBend/2+1); % The last bend
with bend angle of fei/2
sita={sita1';sita2';sita3';sita4'};

count=straightNo+1; % 1st point of the 1st bend. It points to the
current point.
Roci=[R+W/2 R R-W/2]; % Radius of outer bank, center line, inner bank.
for i=1:nBend
    if i==1, nS=1; % subscript for sita
    elseif i==nBend, nS=4;
    else, nS=rem(i,2)+2; end

    if i==1 || i==nBend, nInc=nPerBend/2; % # of segments on this bend.
    else, nInc=nPerBend; end

    for j=1:3 % from right bank to center line to left bank
        if rem(i,2)==1, nR=j; % subscript for R, order is 1,2,3
        else, nR=4-j; end % order is 3,2,1

        xy(count:count+nInc,(j-1)*2+1)=Roci(nR)*cos(sita{nS})+Xc(i);
        xy(count:count+nInc,(j-1)*2+2)=Roci(nR)*sin(sita{nS})+Yc(i);
    end % The coordinates of the inflection points are calculated for
two times.
    count=count+nInc; % Next inflection point
end
xlin=xy(count,1):spacing:xy(count,1)+(straightLen-spacing);
xy(count+1:count+straightNo,1:2:5)=[xlin',xlin',xlin'];
xy(count+1:count+straightNo,2)=y1-W/2;
xy(count+1:count+straightNo,4)=y1;
xy(count+1:count+straightNo,6)=y1+W/2;
count=count+straightNo; % Total number of points. There should be
count==nPt

figure
linespec=cell(3,1);

```

```

linespec(1)=cellstr('b-');
linespec(2)=cellstr('r-.');
linespec(3)=cellstr('b-');
for i=1:3
    plot(xy(:,2*i-1),xy(:,2*i),linespec{i}); % linespec{i} or
char(linespec(i))
    hold on
end
xlabel('X','FontSize',15)
ylabel('Y','FontSize',15)
title('Initial Geometry','FontSize',15)
axis equal
grid on

for i=1:3
    fullFN=[myPath fn{i}];
    fid=fopen(fullFN,'wt');
    xyout=[ [1:count] ',xy(:,2*i-1:2*i)]';
    fprintf(fid,'%7d %10.4f %10.4f\n',xyout');
    fclose(fid);
end

function GenRiverCoord_360deg()

% Follow Dr. Briaud's requirement, generate a bend of 360 degrees

caseNo=6; % Generate filenames for the 3 files: C##R_Initial.txt
C##L_Initial.txt, C##C_Initial.txt
sNo=num2str(caseNo);
if caseNo<=9, sNo=['00' sNo];
elseif caseNo<=99, sNo=['0' sNo]; end

R=160;
W=40;
fei=360; % In degrees here
nBend=1;
RoW=R/W;
spacingCoef=0.1; % Use this to determine nPerBend.
nPerBend=-100; % # of segments on a bend. If nPerBend<=0, use
spacingCoef, or use nPerBend.
LoW=1; % The straight line preceding and after the bends, length/width

sRoWF=['_RoW' num2str(RoW) 'F' num2str(fei) 'B' num2str(nBend)];
fei=fei*pi/180; % convert fei to radians

sRoWF=[]; % Remove this information, so that the file name can be
easily handled in VC & PlotRiver
fnR=['P' sNo 'R' sRoWF '_Initial.txt']; % P stands for parametric study
fnC=['P' sNo 'C' sRoWF '_Initial.txt'];
fnL=['P' sNo 'L' sRoWF '_Initial.txt'];
fn={fnR;fnC;fnL};

if nPerBend<=0
    spacing=spacingCoef*W;
    nPerBend=round(R*fei/spacing);
else
    spacingCoef=R*fei/nPerBend/W;
    spacing=spacingCoef*W;

```

```

end
if rem(nPerBend,2)==1 % Make it an even number
    nPerBend=nPerBend+1;
end

nPt=nPerBend+1;
xy=zeros(nPt,6);
Xc=0; Yc=0;

Roci=[R+W/2 R R-W/2]; % Radius of outer bank, center line, inner bank.
sita=linspace(pi/2+fei/2,pi/2-fei/2,nPerBend+1);

for i=1:3
    xy(:,2*i-1)=Roci(i)*cos(sita)+Xc;
    xy(:,2*i)=Roci(i)*sin(sita)+Yc;
end

figure
linespec=cell(3,1);
linespec(1)=cellstr('b-');
linespec(2)=cellstr('r-.');
linespec(3)=cellstr('b-');
for i=1:3
    plot(xy(:,2*i-1),xy(:,2*i),linespec{i}); % linespec{i} or
char(linespec(i))
    hold on
end
axis equal
grid on

for i=1:3
    fullFN=['M:\_WW\Meander\Data\' fn{i}];
    fid=fopen(fullFN,'wt');
    xyout=[1:nPt]',xy(:,2*i-1:2*i)];
    fprintf(fid,'%7d %10.4f %10.4f\n',xyout');
    fclose(fid);
end

function [fei,sita]=GetAngle(x,y,xcyc)

% function [fei,sita]=GetAngle(x,y,xcyc)
% Calculate fei and sita angle in radian.
% fei: angle formed by P1,center,P3 (P1 is the 1st point, P3 is the
last)
% sita: angle formed by P1 center P2 (P2 is any point in between)
% x,y coordinates of that bend. % previously only three points.
% Based on the center, the angle starts from the 1st point to the 2nd
point.
% P1(x(1),y(1)) P3(x(n),y(n)) (n=lengthx), are at the two ends
% P2(x(2),y(2)) is the reference point for determining the fei angle
direction and sita angle.
% xcyc is the coordinate of the center
% If the reference point overlaps an end point, negative fei and sita
will be returned.

TOL=1e-6;
n=length(x);

```

```

a1=atan2(y(1)-xcyc(2),x(1)-xcyc(1)); % a1 range: [-pi pi]
if a1<0
    a1=a1+2*pi; % Change angles to range [0 2*Pi]
end

a2=atan2(y(2)-xcyc(2),x(2)-xcyc(1)); % a2 is the reference point
if a2<0, a2=a2+2*pi; end

a3=atan2(y(n)-xcyc(2),x(n)-xcyc(1));
if a3<0, a3=a3+2*pi; end

if abs(a1-a2)<TOL || abs(a2-a3)<TOL
    disp('Reference point overlaps an end point in function GetAngle()')
    fei=-99; sita=-1;
    return
end

if a1<a3
    if a2>a1 && a2<a3
        clkWise=false;
    else
        clkWise=true;
    end
else
    if a2>a3 && a2<a1
        clkWise=true;
    else
        clkWise=false;
    end
end

if clkWise==true
    fei=a1-a3;
    if fei<0, fei=fei+2*pi; end
else
    fei=a3-a1;
    if fei<0, fei=fei+2*pi; end
end

if nargout==2
    for i=2:n-1
        ai=atan2(y(i)-xcyc(2),x(i)-xcyc(1));
        if ai<0, ai=ai+2*pi; end

        if clkWise==true
            sita(i)=a1-ai;
        else
            sita(i)=ai-a1;
        end

        if sita(i)<0, sita(i)=sita(i)+2*pi; end
    end
    sita(1)=0;
    sita(n)=fei;
end

```

```

function ar3=mergeVectors(ar1,ar2)

% ar3=mergeVectors(ar1,ar2)
% Merge vector ar1 into ar2. Both ar1 & ar2 are in ascending order.
% It would be more efficient to let length(ar1)<=length(ar2)
% If ar1(i)==ar2(j), just keep one in ar3
% length(ar3)<=length(ar1)+length(ar2)

%ar1=[1 2 4 5 7 10 13 14 18];
%ar1=[1 2 14];
%ar2=[2 3 5 9 11 13];

n1=length(ar1);
n2=length(ar2);
ar3=zeros(n1+n2,1);

count=0;
j=1; % It points to ar2
for i=1:n1 % The elements of ar1 divide ar2 into several segments
    k=j; % k is the start point of the segment, j goes to the end.
    if j>n2 % element ar1(i) is larger than the last element of ar2.
        count=count+1;
        ar3(count)=ar1(i);
        continue;
    end
    while j<=n2 && ar1(i)>=ar2(j)
        j=j+1;
    end % if j=3, it's better to see j points to the end of element 2,
    the beginning of element 3.

    for n=k:j-1
        count=count+1;
        ar3(count)=ar2(n);
    end
    if j==1
        count=count+1;
        ar3(count)=ar1(i);
    end
    if j>1 && ar1(i)~=ar2(j-1) % Then ar1(i)>ar2(j-1)
        count=count+1;
        ar3(count)=ar1(i);
    end
end

for i=j:n2
    count=count+1;
    ar3(count)=ar2(i);
end

ar3=ar3(1:count);

```



```

function cur2=offsetCurve(cur1,dist)

% offset a curve in both directions by dist
% dist should be relatively small.
% cur1 is a column vector
% cur2(:,1:2) is the offset curve above the original
% cur2(:,3:4) is the offset curve below the original
% unlike what AutoCAD offset command does, m=3 points are used here to
% determine the direction of middle point
% Two offsetted curves will be returned

m=3; %# of points are used to determine the direction of the middle
point
nPt=length(cur1);
cur2=zeros(nPt,4);
if rem(m,2)==0, m=m+1; end
hlfm=floor(m/2);
for i=hlfm+1:nPt-hlfm
    % when alpha==(+/-)pi/2, p(1) can be Inf or -7e-10, unpredictable.
    p=polyfit(cur1(i-hlfm:i+hlfm,1),cur1(i-hlfm:i+hlfm,2),1);
    k=-1/p(1); % p(1)==Inf => k=0; p(1)=0 => k=Inf
    alpha=atan2(cur1(i+hlfm,2)-cur1(i-hlfm,2),cur1(i+hlfm,1)-cur1(i-
hlfm,1));
    if alpha==pi/2 || alpha==-pi/2
        k=0;
    end
    dy=dist/sqrt(1+1/k^2);
    if (alpha>=-pi && alpha<=-pi/2) || (alpha>=pi/2 && alpha<=pi)
        dy=-dy; % The first-last point vector is in 2nd/3rd quadrant.
    end

    if alpha==-pi/2, dx=dist;
    elseif alpha==pi/2, dx=-dist;
    else, dx=1/k*dy; end

    if i==hlfm+1
        cur2(1:hlfm,1)=cur1(1:hlfm,1)+dx; % above -x
        cur2(1:hlfm,2)=cur1(1:hlfm,2)+dy; % above -y
        cur2(1:hlfm,3)=cur1(1:hlfm,1)-dx; % below -x
        cur2(1:hlfm,4)=cur1(1:hlfm,2)-dy; % below -y
    elseif i==nPt-hlfm
        cur2(nPt-hlfm+1:nPt,1)=cur1(nPt-hlfm+1:nPt,1)+dx;
        cur2(nPt-hlfm+1:nPt,2)=cur1(nPt-hlfm+1:nPt,2)+dy;
        cur2(nPt-hlfm+1:nPt,3)=cur1(nPt-hlfm+1:nPt,1)-dx;
        cur2(nPt-hlfm+1:nPt,4)=cur1(nPt-hlfm+1:nPt,2)-dy;
    end
    cur2(i,1)=cur1(i,1)+dx;
    cur2(i,2)=cur1(i,2)+dy;
    cur2(i,3)=cur1(i,1)-dx;
    cur2(i,4)=cur1(i,2)-dy;
end
end

```

Script PlotChenSimulationData.m

```

% Plot Dr. Chen's simulation data based on sita over fei
% Works with ChenSimulationData.mat

% 1st col, point #
% 2nd col, x coordinate
% 3rd col, channel lengthwise distance
% 4th col, maximum shear stress at left bank
% 5th col, maximum shear stress at right bank
% 6th col, depth average shear stress at left bank
% 7th col, depth average shear stress at right bank
ar=r4f240(:,3:7);
RoverW=4;
fei=240;

miu=0.8;    % For Extreme Value Distribution

rou=1000;   % density of water
v=0.2;      % velocity at entrance
rouVs=rou*v^2; % rou v square, to non-dimensionalize tao.

ar(:,2:5)=ar(:,2:5)/rouVs;

sL=ar(:,1); % Channel lengthwise distance
ar(1,2:5)=0; % Shear stress is too large.

n0=41; % First and last 41 points are on straight lines.

nPt=length(ar);

nArc=6;

n1Arc=(nPt-n0*2+1)/nArc;

n1Arc=round(n1Arc);

maxS=max(ar(:,2));

ytmp=maxS/4;

figure

%subplot(2,1,1)
% plot(ar(:,1),ar(:,2),'k-') % tao_max left bank
% hold on
% plot(ar(:,1),ar(:,4),'k-','LineWidth',1.5) %tao_avg, left bank
% hold on
plot(ar(:,1),ar(:,3),'r-') % tao_max right bank
hold on
plot(ar(:,1),ar(:,5),'k-','LineWidth',1.5) %tao_avg, right bank
hold on

xlabel('Channel lengthwise distance (m)','FontSize',15)
ylabel('Shear stress (N/m^2)','FontSize',15)
ylabel('\tau/\rho v^2','FontSize',15)
str=['R/W=',num2str(RoverW),' \phi=',num2str(fei),'\circ'];
title(str,'FontSize',15)
grid on

```

```

for i=1:2:nArc
    n1=round(n0+n1Arc/2);
    n2=n1+(i-1)*n1Arc;
    n3=n1+i*n1Arc;
    x1=sL(n2);
    x2=sL(n3);

    m1=round(n2-n1Arc/2);
    m2=round(n3+n1Arc/2);
    SoF=zeros(m2-m1+1,1); % It's in workspace and needs to be
initialized.
    for j=m1:1:m2
        SoF(j-m1+1)=(j-n2)/(n3-n2);
    end
    f=1/400/RoverW*PlotChen_evDist(SoF,miu); % Non-dimensionalized
shear stress

    plot(sL(m1:m2),f,'b--')
    hold on

    plot([x1 x2],[ytmp ytmp],'b-.')
    hold on
    plot([x1 x1],[0 ytmp],'k:')
    hold on
    plot([x2 x2],[0 ytmp],'k:')
    hold on

end
str=['Fitted',' \mu=',num2str(miu)];
%legend('\tau_m_a_x Left bank','\tau_a_v_g Left bank','\tau_m_a_xRight
bank','\tau_a_v_gRight bank','\theta/\phi=[0 1]')
%legend('\tau_m_a_x Left bank','\tau_a_v_g Left
bank','Fitted','\theta/\phi=[0 1]')
legend('\tau_m_a_x Right bank','\tau_a_v_g Right
bank',str,'\theta/\phi=[0 1]')

function f=PlotChen_evDist(x,miu)

% Extreme value distribution

sigma=0.37;

x2=(x-miu)/sigma;

f=1/sigma*exp(x2).*exp(-exp(x2));

```

```

function plotMgrtForOneFlow
(Mm1,M1dt,M0,M1idx,cXY,cLen,nArcIdx,nPt,nArc,tStep)

% For C++ debug. Plot intermediate migration for C++ function OneFlow()
% Variable names are the same as the ones in C++ function.
% Mm1(nPt,nArc),Mmax; M1dt(nPt,nArc), 1-time
migration;M1idx(2,nArc),index for Mm1 & M1dt
% cXY(4,nMaxPt), coordinates of the channel before & after migration.
% cLen(nPt), channel length
% nPt, # of points on the curve. nArc, # of identified bends
% tStep(1): time step in hours, tStep(2),current step No.

rSign=-1; % Make a mirror image about X axis.
scrsz=get(0,'ScreenSize');

linSpec=cell(8,1); % For Mmax
linSpec(1)=cellstr('k--');
linSpec(2)=cellstr('k--');
linSpec(3)=cellstr('k--');
linSpec(4)=cellstr('k--');
linSpec(5)=cellstr('k--');
linSpec(6)=cellstr('k--');
linSpec(7)=cellstr('k--');
linSpec(8)=cellstr('k--');

linSpec2=cell(8,1); % For M1dt
linSpec2(1)=cellstr('r-.');
linSpec2(2)=cellstr('r-.');
linSpec2(3)=cellstr('r-.');
linSpec2(4)=cellstr('r-.');
linSpec2(5)=cellstr('r-.');
linSpec2(6)=cellstr('r-.');
linSpec2(7)=cellstr('r-.');
linSpec2(8)=cellstr('k--');

timeStr=['Time step=' num2str(tStep(1)) ' hour(s) ' 'Current step No.='
num2str(tStep(2)) ' (1=Initial)'];

figure('Position',scrsz) % Maximize the wondow
% to produce the right legend
plot([0 0.01],[0 0],'r-.') % Mmax
hold on
plot([0 0.01],[0 0],'k--') % M1dist
hold on
plot([0 0.01],[0 0],'k--','LineWidth',1.8) % Mcombined
hold on
plot([0 0.01],[0 0],'k-','LineWidth',1.8) % Mtotal
hold on
plot([0 0.01],[0 0],'b*','MarkerSize',5) % Arc boundary
hold on
set(gca,'XMinorGrid','On');

for i=1:nArc % Plot Mmax
    idx1=M1idx(1,i); % It's a transpose of the one in C++
    idx2=M1idx(2,i);
    if idx1<=0 || idx2<=0, continue; end % If calculation is not done
    on a bend, this will happen.
    nStyle=rem(i,7);
    if nStyle==0, nStyle=7; end

```

```

        plot(cLen(idx1:idx2),rSign*Mm1(idx1:idx2,i),char(linSpec2(nStyle)))
        hold on
    end

    for i=1:nArc+1 % Plot M1dt and accumulated migration for one flow
        idx1=M1idx(1,i);
        idx2=M1idx(2,i);
        if idx1<=0 || idx2<=0, continue; end

        nStyle=rem(i,7);
        if nStyle==0, nStyle=7; end

        if i==nArc+1 % It is the accumulated migration of this flow.
            %idx1=1; idx2=nPt % already done in C++ code.
            nStyle=8;
        end
        if i<nArc+1

        plot(cLen(idx1:idx2),rSign*M1dt(idx1:idx2,i),char(linSpec(nStyle)))
        else

        plot(cLen(idx1:idx2),rSign*M1dt(idx1:idx2,i),char(linSpec(nStyle)),'Lin
eWidth',1.8)
        end
        hold on
    end

    % Plot total migration up to the end of this step.
    plot(cLen(1:nPt),rSign*M0(1:nPt),'k-','LineWidth',1.8)
    hold on

    mSize=[5 8 11]; %Marker Size
    for i=1:nArc % Plot Arc boundaries
        idx1=nArcIdx(1,i);
        idx2=nArcIdx(2,i);
        if idx1<=0 || idx2<=0, continue; end

        x1=cLen(idx1); x2=cLen(idx2);
        y1=rSign*M1dt(idx1,i); y2=rSign*M1dt(idx2,i);
        sIdx=rem(i,3);
        if sIdx==0, sIdx=3; end
        plot([x1 x2],[y1 y2],'b*','MarkerSize',mSize(sIdx))
        hold on
    end

    xlabel('Channel Lengthwise Distance (m)','FontSize',14)
    ylabel('Migration (mm)','FontSize',14)
    s1=['Migration Output for Debug'];
    if tStep(1)>0
        tleStr={s1,timeStr}; % title string
    else
        tleStr=s1;
    end
    title(tleStr,'FontSize',14)
    legend('M_m_a_x','M_l_b_e_n_d','M_c_o_m_b_i_n_e_d','M_t_o_t_a_l',
'Boundary')
    grid on
    ylim([-2000 2000]) % For parametric study where Mmax is about 4e5 mm

```

```

%*****
% Plot the initial channel, the channel before migration and the
channel after migration
figure('Position',scrsz)
plot(cXY(1,1:nPt),cXY(2,1:nPt),'r--','LineWidth',1.5)
hold on
plot(cXY(3,1:nPt),cXY(4,1:nPt),'m-')
hold on
plot(cXY(5,1:nPt),cXY(6,1:nPt),'b--')

xlabel('X','FontSize',14)
ylabel('Y','FontSize',14)

s1=['Original and Migrated Channel'];
if tStep(1)>0
    tleStr={s1,timeStr};
else
    tleStr=s1;
end
title(tleStr,'FontSize',14)
legend('Initial','Before migration','After migration')
grid on
axis equal

function PlotMvsT(time,migration,xL,yL,sT)

% Plot migration vs. time curve for one point.
% time: x axis, time
% xL: xlabel
% sT: title

scrsz=get(0,'ScreenSize');

testNo=str2num(sT);
if ~isempty(testNo)
    sT=['Flume Test ' sT ' Migration vs. Time'];
else
    sT=[sT ' Migration vs. Time'];
end

figure('Position',scrsz); % Maximize the window
time=[0; time];
migration=[0; migration];
plot(time,migration,'b-')
grid on

xlabel(xL,'FontSize',12)
ylabel(yL,'FontSize',12)
title(sT,'FontSize',12)

function PlotRiver1Curve(fn,sTitle,bMov,bOverlap)

% Plot original channel and migrated channel in VC program
% Plot only one curve, either center line or one bank
% If required, superimpose the final measured curve(flume test, field
data).

```

```

% The final measured center line is obtained by offsetting the banks by
W/2.
% bMov=0/1; 0: no movie will be shown. Otherwise 1, a movie will be
shown.
% sTitle is 'Prj Name' or flume test number;
% '_RLRLC_IF.txt' contains measured coordinates for initial banks,
final banks and center line.
% File 'T01_RLRLC_IF.txt' has 6 columns. Col 1: X coordinates for 5
other columns.
% Col 2: Y-Right initial; Col 3: Y-Left initial; Col 4: Y-Right final;
Col 5: Y-Left final
% Col 6: Y-Center final, average of col 4 &5.
% If '_RLRLC_IF.txt' is not available, 'R_Final.txt' or 'L_Final.txt'
should be there.

if nargin<=3, bOverlap=false; end

% fn=['M:\_WW\MDemo_Flume\Data\TMP_15C_C.dat']; % This is for debugging.
% sTitle=['15C'];
% bOverlap=true;

bMov=0;
superimpose=true; % Change it to false to disable debug code.
scrsz=get(0,'ScreenSize'); % Let the plot fill the whole screen

fid=fopen(fn,'rb');
nCur=fread(fid,1,'integer*4');
nPt=zeros(nCur,1);
xy=cell(nCur,1);
for i=1:nCur
    nPt(i)=fread(fid,1,'integer*4');
    cur=fread(fid,nPt(i)*2,'real*8');
    xy{i}=reshape(cur,2,[]);
end

%The point specified on Geometry dialogue. If no specification was made,
it is set to (0,0).
out_XY=fread(fid,2,'real*8');
fclose(fid);

% To superimpose the final center(bank) line on the graph. For
comparison of prediction and test result
%
% sTitle comes from "Project Name:" of the main GUI of MEANDER
% If simply a number is entered there, then it's treated as Flume Test
No.
% The final center line defined by Dr. Chen of this test will be
superimposed
% on the movie graph. A file with correct format like "T06C_Chen.txt"
must be there.
% The program will stop if either the file is not there or the format
is not correct.
% If a phrase like "Flume Test 15" is entered, don't plot the final
center line.
nT=length(sTitle);
if nT>6,
    disp('In PlotRiver1Curve(), Project name is not of the right
format.')
    return
end % Longest: P123R or 123L

```

```

testNo=str2num(sTitle);
if isempty(testNo)
    sT1=sTitle(1:nT-1); % Test No.
    sT2=upper(sTitle(nT)); % 'L' or 'R', can be in lower case.
else
    sT1=sTitle;
    sT2='C';
end

% Superimpose measured final center line/bank for flume test.
% curveXY stores the measured center line or bank.
testNo=str2num(sT1);
b2Files=0;
if length(testNo)>0 && superimpose==true
    testNoStr=sT1;
    if testNo<10, testNoStr=['0' testNoStr]; end
    pathStr=fileparts(fn); % Get the path of the executable.
    switch sT2
        case 'R'
            fn1=[pathStr '\\' 'T' testNoStr '_RLRLC_IF.txt'];
            fn2=[pathStr '\\' 'T' testNoStr '_R_Final.txt'];
            fid1=fopen(fn1,'rt'); % To test whether '_RLRLC_IF.txt' is
available
            fid2=fopen(fn2,'rt'); % If not, use '_R_Final.txt'
            if fid1~-1, b2Files=b2Files+1; fclose(fid1); end
            if fid2~-1, b2Files=b2Files+2; fclose(fid2); end
            if b2Files==0
                disp('In PlotRiver1Curve(), measured data is not
available.')
                return
            end
            if b2Files>=1 && b2Files~=2 % '_RLRLC_IF.txt' is available
                curveXY=dlmread(fn1);
                curveXY=curveXY(:,1:3:4);
            else % Only '_R_Final.txt' is available, like
Brazos River
                curveXY=read2Col3ColFile(fn2);
            end
            sTitle=['Flume Test ' sTitle ' River Migrating with Final
Right Bank'];
            case 'C'
                curveFN=['T' testNoStr '_C_Chen.txt'];
                curveFN=[pathStr '\\' curveFN]; % The files have to be in
the folder ..\Data\
                sTitle=['Flume Test ' sTitle ' River Migrating'];
                %
                curveXY=read2Col3ColFile(curveFN);
                %
                curveXY=curveXY(:,2:3);
            case 'L'
                fn1=[pathStr '\\' 'T' testNoStr '_RLRLC_IF.txt'];
                fn2=[pathStr '\\' 'T' testNoStr '_L_Final.txt'];
                fid1=fopen(fn1,'rt');
                fid2=fopen(fn2,'rt');
                if fid1~-1, b2Files=b2Files+1; fclose(fid1); end
                if fid2~-1, b2Files=b2Files+2; fclose(fid2); end
                if b2Files==0
                    disp('In PlotRiver1Curve(), measured data is not
available.')
                    return
                end
            end
        end
    end
end

```



```

        if b2Files>=1 && b2Files~=2 % '_RLRLC_IF.txt' is available
            curveXY=dlmread(fn1);
            curveXY=curveXY(:,1:4:5);
        else % Only 'L_Final.txt' is available, like
Brazos River
            curveXY=read2Col3ColFile(fn2);
        end
        sTitle=['Flume Test ' sTitle ' River Migrating with Final
Left Bank'];
        otherwise
            end
        end
%Plot the migrating channel in different colors
lnSpec=cell(7,1);
lnSpec(1)=cellstr('r-');
lnSpec(2)=cellstr('g-');
lnSpec(3)=cellstr('b-');
lnSpec(4)=cellstr('c-');
lnSpec(5)=cellstr('m-');
lnSpec(6)=cellstr('y-');
lnSpec(7)=cellstr('k-');

if bOverlap==false
    figure('Position',scrsz);
else
    hold on
end
plot([0 0.01],[0 0],'r--','LineWidth',1.5) % Measured initial
hold on
plot([0 0.01],[0 0],'b-') % Predicted at each time step
if sT2~='C' % Don't overlap Chen's final bank(offset the real one by
W/2).
    plot([0 0.01],[0 0],'b-','LineWidth',1.8) % Measured final
    hold on
end
if bMov==0 % No movie, for Risk Analysis
    plot(xy{1}(1,:),xy{1}(2:,:),'r--','LineWidth',1.5)
    hold on
    for i=2:nCur
        sNo=rem(i,7);
        if sNo==0, sNo=7; end
        plot(xy{i}(1,:),xy{i}(2,:),char(lnSpec(sNo)))
        hold on
    end
% grid on
axis equal
xlabel('X','FontSize',12)
ylabel('Y','FontSize',12)
title(sTitle,'FontSize',12)
else % Show a movie, Constant flow or one hydrograph
    set(fig,'DoubleBuffer','on');
    mov=avifile('..\Migrating.avi','fps',1);
    h=plot(xy{1}(1,:),xy{1}(2:,:),'r--','LineWidth',1.5);
    axis equal % This one has to be here to show the movie correctly
% grid on
xlabel('X','FontSize',12)
ylabel('Y','FontSize',12)
title(sTitle,'FontSize',12)

    set(h,'EraseMode','xor');

```

```

    frame=getframe(gca);
    mov = addframe(mov,frame);
    hold on
    for i=2:nCur
        sNo=rem(i,7);
        if sNo==0, sNo=7; end
        h=plot(xy{i}(1,:),xy{i}(2,:),char(lnSpec(sNo)))
        frame=getframe(gca);
        mov = addframe(mov,frame);
        hold on
    end
    mov=close(mov);
end
if length(testNo)>0 && superimpose==true && sT2~='C'
    hold on
    plot(curveXY(:,1),curveXY(:,2),'b-','LineWidth',1.8)
end
hold on % The point for M vs. t curve or risk analysis.
plot(out_XY(1),out_XY(2),'mo','MarkerSize',5)

if bOverlap==false
    if sT2~='C'
        legend('Initial bank','Predicted banks','Measured final bank')
    else
        legend('Initial bank','Predicted banks')
    end
end
% legend('boxoff')
end

if bOverlap==true % Overwrite previous title
    sTitle=['Flume Test ' num2str(testNo) ' River Migrating'];
    title(sTitle,'FontSize',12)
end

function PlotRiver2Banks (sT, fType, testNoStr)

% function PlotRiver2Banks(sT,fType,testNoStr)
% Called by PlotRiverBanks_Main().
% For Bank Method only. The calculated curve is a bank.
% This function plot initial 2 banks and predicted & measured final 2
banks
% For center line method,predicted final 2 banks are obtained by
offseting center line by W/2
% For both-bank method, predicted final 2 banks are calculated.
% sT{1} is the project name on SRICOSView dialogue for right bank
% sT{2} is the project name on SRICOSView dialogue for left bank
% File names of the data of both banks are generated here which is
consistent with VC.
% TMP_sTitle_R.dat is for the right bank; TMP_sTitle_L.dat is for the
left bank.
% Only when the results for both banks are present will the program
work.
% The user is responsible for providing up-to-date data.

bCenAvail=true; % If initial center line is not available, it's false.
Then don't plot it.
%*****
% Generate file names of initial and measured final data files.
datPath=['..\Data\'];

```

```

% datPath=['M:\_WW\Meander\Data\']; % For debugging
% datPath=['M:\_WW\MDemo_Brazos\Data\'];
if fType==1 % Flume test or real rivers (Measured final banks available)
    fnCI=[datPath 'T' testNoStr 'C_Initial.txt']; % Center line,
Initial file name
    fnmF=[datPath 'T' testNoStr '_RLRLC_IF.txt']; % file name of
'm'easured final banks
    fnmFR=[datPath 'T' testNoStr 'R_Final.txt']; % If '_RLRLC_IF.txt'
is not available,
    fnmFL=[datPath 'T' testNoStr 'L_Final.txt']; % use these two files.
elseif fType==2 % Parametric study (Measured data not available)
    fnCI=[datPath 'P' testNoStr 'C_Initial.txt'];
end

%*****
% Determine how many bank's data is available.
fnpFR=[datPath 'TMP_' sT{1} '_R.dat']; % file name of right bank,
predicted Final
fnpFL=[datPath 'TMP_' sT{2} '_L.dat']; % Actually contains migrated
channel of each step.

b2Banks=0; % Bit 1, Right bank; Bit 2 Left bank. If both banks are
available, b2Banks=3.
fidR=fopen(fnpFR,'rb');
fidL=fopen(fnpFL,'rb');
if fidR~-1, b2Banks=b2Banks+1; end % Bit 1
if fidL~-1, b2Banks=b2Banks+2; end % Bit 2
if b2Banks<3
    disp('In PlotRiver2Banks(),at least one bank is not ready.')
    return
end

% Read the initial and predicted final curves for the RIGHT bank.
nCurR=fread(fidR,1,'integer*4');
xyRIpF=cell(2,1); % Initial and predicted Final right bank
count=0; % xyRIpF{i} is a row major array. Row 1, X; Row 2, Y.
for i=1:nCurR
    nPt=fread(fidR,1,'integer*4');
    if i==1 || i==nCurR
        count=count+1;
        cur=fread(fidR,nPt*2,'real*8');
        xyRIpF{count}=reshape(cur,2,[]); % 1st row, X; 2nd row, Y.
    else
        fseek(fidR,nPt*2*8,'cof');
    end
end
%The point specified on Geometry dialogue. If no specification was made,
it is set to (0,0).
out_XYR=fread(fidR,2,'real*8');
fclose(fidR);

% Read the initial and predicted final curves for the LEFT bank.
nCurL=fread(fidL,1,'integer*4');
xyLIpF=cell(2,1); % Initial and predicted Final left bank
count=0;
for i=1:nCurL
    nPt=fread(fidL,1,'integer*4');
    if i==1 || i==nCurL
        count=count+1;
        cur=fread(fidL,nPt*2,'real*8');
    end
end

```

```

        xyLIpF{count}=reshape(cur,2,[]); % 1st row, X; 2nd row, Y.
    else
        fseek(fidL,nPt*2*8,'cof');
    end
end
out_XYL=fread(fidL,2,'real*8');
fclose(fidL);

% Read the initial center line of this case.
% xyCI is of 2 columns
[xyCI,count,flag]=read2Col3ColFile(fnCI); % If flag==-1, it failed to
read the file.
if flag==-1 % The initial center line doesn't exist or can't be read.
Just don't plot it.
%     xyCI=[0 0];
%     bCenAvail=false;
end
% For flume test read measured final banks.
if fType==1
    fidmF=fopen(fnmF,'rt');
    if fidmF~-1
        xymF=dlmread(fnmF); % fnmF is like 'T11_RLRLC_IF.txt'
        xymFR=xymF(:,[1,4]); % Col 1: X; Col 2: Y-Right initial; Col 3:
Y-Left initial
        xymFL=xymF(:,[1,5]);
    else
        xymFR=read2Col3ColFile(fnmFR);
        xymFL=read2Col3ColFile(fnmFL);
    end
end

%*****
% Plot initial and predicted final banks. Superimpose measured final
banks.
scrsz=get(0,'ScreenSize'); % Let the plot fill the whole screen
figure('Position',scrsz)
if bCenAvail==true % If initial center line is available
    plot([0 0.1],[0 0],'k-','LineWidth',0.5) % Initial center line.
    For producing right legend.
    hold on
end
plot([0 0.1],[0 0],'k-','LineWidth',0.5) % Initial banks
hold on
plot([0 0.1],[0 0],'m--','LineWidth',1.5) % Predicted final banks
hold on
plot([0 0.1],[0 0],'b-','LineWidth',1.8) % Predicted final banks
hold on

if bCenAvail
    plot(xyCI(:,1),xyCI(:,2),'k-','LineWidth',0.5) % Initial center
line
    hold on
end
plot(xyRIpF{1}(1,:),xyRIpF{1}(2:,:), 'k-', 'LineWidth',0.5) % Initial
right bank
hold on
plot(xyLIpF{1}(1,:),xyLIpF{1}(2:,:), 'k-', 'LineWidth',0.5) % Initial left
bank
hold on

```

```

plot(xyRipF{2}(1,:),xyRipF{2}(2:,:), 'm--', 'LineWidth',1.5) % Final
predicted right bank
hold on
plot(xyLipF{2}(1,:),xyLipF{2}(2:,:), 'm--', 'LineWidth',1.5) % Final
predicted left bank
hold on

if fType==1 % Plot measured final banks for flume tests
    plot(xymFR(:,1),xymFR(:,2), 'b-', 'LineWidth',1.8) % Measured final
right bank
    hold on
    plot(xymFL(:,1),xymFL(:,2), 'b-', 'LineWidth',1.8) % Measured final
left bank
end

hold on
plot(out_XYR(1),out_XYR(2), 'mo', 'MarkerSize',5)
hold on
plot(out_XYL(1),out_XYL(2), 'mo', 'MarkerSize',5)
axis equal
xlabel('X', 'FontSize',12)
ylabel('Y', 'FontSize',12)
testNoStr=num2str(str2num(testNoStr)); % Remove prefix '0'
if fType==1
    sTitle=['Bank Method --Flume Test ' testNoStr ' Migrated Channel'];
elseif fType==2
    sTitle=['Bank Method --Parametric Study ' testNoStr ' Migrated
Channel'];
end
title(sTitle, 'FontSize',12)

if fType==1
    if bCenAvail
        legend('Initial center line','Initial banks','Predicted final
banks','Measured final banks')
    else
        legend('Initial banks','Predicted final banks','Measured final
banks')
    end
elseif fType==2
    legend('Initial center line','Initial banks','Predicted final
banks')
end
%legend('boxoff')

%*****
% Plot predicted left and right banks at each time step. Superimpose
measured final banks.
bOverlap=false;
bMov=0;
PlotRiver1Curve(fnpFR, sT{1}, bMov, bOverlap);
bOverlap=true;
PlotRiver1Curve(fnpFL, sT{2}, bMov, bOverlap);

```

```

function PlotRiverBanks_Main(sTitle,width)

% It calls either PlotRiver2Banks() or PlotRiverCenOffset() based on
sTitle
% If the calculated curve is a bank, call PlotRiver2Banks()
% If the calculated curve is a center line, call PlotRiverCenOffset()
% width is used for offseting center line by width/2

% sTitle=['82L']; % Bank method: '2L' or '2R'; Center line method: '2C'
% width=0.75;

nT=length(sTitle);
if nT>6,
    disp('In PlotRiverBanks_Main(), Project name is not of the right
format.')
```

return

```

end % Longest: P123R

if sTitle(1)>='0' && sTitle(1)<='9', fType=1; % Flume Test
elseif upper(sTitle(1))=='P', fType=2; % Parametric study
else
    disp('In PlotRiverBanks_Main(),It''s neither a flume test nor a
parametric study case.')
```

return; % Consider other cases like field verification later.

```

end

if fType==1 % Flume test
    if nT<=3
        testNo=str2num(sTitle);
        if isempty(testNo)
            sT1=sTitle(1:nT-1); % Test No.
            sT2=upper(sTitle(nT)); % 'L' or 'R', can be in lower
case.
        else
            sT1=sTitle;
            sT2='C';
        end
    else
        disp('In PlotRiverBanks_Main(), fType=1 (flume test) but the
title is longer than 3.')
```

return;

```

end
    testNo=str2num(sT1);
    testNoStr=sT1;
    if testNo<10, testNoStr=['0' testNoStr]; end
elseif fType==2 % Parametric study
    if nT<=2, return; end
    sT1=sTitle(2:nT-1);
    sT2=upper(sTitle(nT));
    testNo=str2num(sT1);
    if isempty(testNo) || (sT2~='R' && sT2~='L' && sT2~='C')
        disp('In PlotRiverBanks_Main(), the case no of parametric study
is wrong.')
```

disp('Or the suffix (R,L,C) is missing')

return;

```

end
    testNoStr=sT1;
    if testNo<10, testNoStr=['00' testNoStr];
    elseif testNo<100, testNoStr=['0' testNoStr]; end
end
```

```

if sT2=='C'
    PlotRiverCenOffset(sTitle,width,fType,testNoStr)
elseif sT2=='R' || sT2=='L'
    sT=cell(2,1);
    sTitle=sTitle(1:nT-1); % Remove letter 'R' or 'L'.
    sT{1}=[sTitle 'R'];
    sT{2}=[sTitle 'L'];
    PlotRiver2Banks(sT,fType,testNoStr)
end

function PlotRiverCenOffset(sTitle,width,fType,testNoStr)

% function PlotRiverCenOffset(sTitle,width,fType,testNoStr)
% Called by PlotRiverBanks_Main(sTitle,width).
% For center line method only.
% Final predicted center line is offset by W/2 to get predicted final
banks.

%*****
%*****
% Generate file names of initial and measured final data files.
datPath=['..\Data\'];
%datPath=['M:\_WW\MDemo_Flume\Data\']; % For debugging
%datPath=['c:\Temp\Meander\Data\'];
if fType==1 % Flume test
    fnRI=[datPath 'T' testNoStr 'R_Initial.txt'];
    fnLI=[datPath 'T' testNoStr 'L_Initial.txt'];
    fnmF=[datPath 'T' testNoStr '_RLRLC_IF.txt']; % file name of
'm'easured final banks
elseif fType==2 % Parametric study
    fnRI=[datPath 'P' testNoStr 'R_Initial.txt'];
    fnLI=[datPath 'P' testNoStr 'L_Initial.txt'];
end

% Read the files
fnCpF=[datPath 'TMP_' sTitle '_C.dat'];
fidC=fopen(fnCpF,'rb');
if fidC==-1
    disp('In PlotRiverCenOffset(), can't open simulation data.')
    return
end
nCurC=fread(fidC,1,'integer*4');
xyCIpF=cell(2,1); % Initial and predicted Final center line
count=0;
for i=1:nCurC
    nPt=fread(fidC,1,'integer*4');
    if i==1 || i==nCurC
        count=count+1;
        cur=fread(fidC,nPt*2,'real*8');
        xyCIpF{count}=reshape(cur,2,[]); % 1st row, X; 2nd row, Y.
    else
        fseek(fidC,nPt*2*8,'cof');
    end
end
end
%The point specified on Geometry dialogue. If no specification was made,
it is set to (0,0).
out_XYC=fread(fidC,2,'real*8');
fclose(fidC);

```

```

% Offset predicted final center line to get predicted final banks
xyOff=offsetCurve(xyCIpF{2}',width/2); % xyOff(:,1:4)

% Read initial banks
[xyRI,count,fid]=read2Col3ColFile(fnRI); % 2 columns
[xyLI,count,fid]=read2Col3ColFile(fnLI);

% For flume test read measured final banks.
if fType==1 % Flume Test
    xymF=dlmread(fnmF); % fnmF is like 'T11_RLRLC_IF.txt'
    xymF=xymF(:, [1,4,5]); % Col 1: X; Col 2: Y-Right; Col 3: Y-Left
end

%*****
%*****
% Plot the curves
scrsz=get(0,'ScreenSize'); % Let the plot fill the whole screen
figure('Position',scrsz)
plot([0 0.1],[0 0],'k-','LineWidth',0.5) %Center line, For producing
right legend
hold on
plot([0 0.1],[0 0],'k-','LineWidth',0.5) %Initial banks
hold on
plot([0 0.1],[0 0],'m--','LineWidth',1.5) %Final banks by offsetting
predicted final center line
hold on
plot([0 0.1],[0 0],'b-','LineWidth',1.8) %Final measured banks
hold on

plot(xyCIpF{1}(1,:),xyCIpF{1}(2:,:), 'k-','LineWidth',0.5) % Initial
center line
hold on
plot(xyRI(:,1),xyRI(:,2), 'k-','LineWidth',0.5) % Initial right bank
hold on
plot(xyLI(:,1),xyLI(:,2), 'k-','LineWidth',0.5) % Initial left bank
hold on
plot(xyOff(:,1),xyOff(:,2), 'm--','LineWidth',1.5) % Predicted final
left bank.
hold on
plot(xyOff(:,3),xyOff(:,4), 'm--','LineWidth',1.5) % Predicted final
right bank.
hold on

if fType==1 % Plot measured final banks for flume tests
    plot(xymF(:,1),xymF(:,2), 'b-','LineWidth',1.8) % Measured final
right bank
    hold on
    plot(xymF(:,1),xymF(:,3), 'b-','LineWidth',1.8) % Measured final
left bank
end

hold on
plot(out_XYC(1),out_XYC(2), 'mo','MarkerSize',5)
axis equal

xlabel('X','FontSize',12)
ylabel('Y','FontSize',12)
testNoStr=num2str(str2num(testNoStr)); % Remove prefix '0'

```



```

if fType==1
    sTitle=['Center Line Method --Flume Test ' testNoStr ' Migrated
Channel'];
elseif fType==2
    sTitle=['Center Line Method --Parametric Study ' testNoStr '
Migrated Channel'];
end
title(sTitle,'FontSize',12)

if fType==1
    legend('Initial center line','Initial banks','Predicted final
banks','Measured final banks')
elseif fType==2
    legend('Initial center line','Initial banks','Predicted final
banks')
end
%legend('boxoff')

function [cur,count,fid]=read2Col3ColFile(fn)

% The file can be of 2 column or 3 columns. Delimiter can be
space,tab,or comma.
% The last two columns will be read and returned in column major matrix
% If failed, fid=-1.
% An alternative is to use dlmread() and judge the # of columns.

fid=fopen(fn,'rt'); %For the purpose of compiling.
if fid==-1
    cur=0;
    count=0;
    return
end
line1=fgetl(fid); % fgets() will return two more characters.
nComma=0;
nC=length(line1);
for i=1:nC
    if line1(i)==' ,'
        nComma=nComma+1;
    end
end
nNum=0;
i=1;
while i<=nC
    seeANum=false; % find a number
    while (line1(i)>='0' && line1(i)<='9') || line1(i)=='.' ||
line1(i)=='-' || upper(line1(i))=='E'
        i=i+1;
        seeANum=true;
        if i>nC, break; end
    end
    if seeANum,
        nNum=nNum+1;
    end
    i=i+1;
end

if nComma==0 && nNum==2
    fStr=['%g %g'];
elseif nComma==0 && nNum==3

```

```

        fStr=['%*d %g %g'];
elseif nComma==1 && nNum==2
    fStr=['%g, %g'];
elseif nComma==2 && nNum==3
    fStr=['%*d, %g, %g'];
else
    fStr=['Comma number=' num2str(nComma) ' nNum=' num2str(nNum)];
    disp(fStr)
    return
end
frewind(fid);
[cur,count]=fscanf(fid,fStr,[2 inf]);
cur=cur';
fclose(fid);

function ReadMeanderCoord()

fn='Z:\_WW\Meander\MeanderCoord.dat';
%fn='H:\Desktop\MeanderRun\MeanderCoord.dat';
%fn='C:\temp\MDemo\MeanderCoordRisk.dat';
fid=fopen(fn,'rb');

% num(1): # of points on the curve
% num(2): # of curves
nCur=fread(fid,1,'integer*4');
nPt=zeros(nCur,1);
xy=cell(nCur,1);
for i=1:nCur
    nPt(i)=fread(fid,1,'integer*4');
    cur=fread(fid,nPt(i)*2,'real*8');
    xy{i}=reshape(cur,2,[]);
end

fclose(fid);

figure
plot(xy{1}(1,:),xy{1}(2:,:),'r--','LineWidth',1.3)
hold on
for i=2:nCur
    plot(xy{i}(1,:),xy{i}(2:),'b-')
    hold on
end
grid on
axis equal

```

C.1 COMPUTER CODE WRITTEN IN C/C++ FOR IMPLEMENTING HYPERBOLIC MODEL (PARTIAL)

```

// SRICOSDoc.cpp : implementation of the CSRICOSDoc class
double CSRICOSDoc::Hyperbola(double M0,double Mmax,double MdotI,double
t)

// Implement the formula of hyperbolic model
// Given an existing migration distance(scour depth) and Migration vs.
Time curve,
// calculate the new accumulated migration.
// t, how long the situation(V, soil properties, R/W etc) of the new
// hyperbolic curve lasts. Unit: hour.
// Follow SRICOS-EFA, MdotI's unit is always mm/hr.
// For metric units there MUST be M0 mm, Mmax mm, t hour. Migration in
mm is returned
// For English units, migration in foot is returned.
// Called by OneVelocity(...)
{
if(M0<0) // If existing migration is a negative number, treat it as 0.
M0=0;
if(M0>=Mmax || fabs(MdotI)<1e-6) // If tao_max<tao_critical, let MdotI
=0. No erosion.
return M0;
if(!m_bSI) // If English unit. Assume MdotI is always in mm/hr.
MdotI/=304.8; // Convert from mm/hr to ft/hr,
double te=1/MdotI/(1/M0-1/Mmax);
return (te+t)/(1/MdotI+(te+t)/Mmax);
}

double CSRICOSDoc::CalMdotI(double v,double RoW,double SoF)
// Use the tao_max formula for Meander Migration
// RoW: R over W, SoF: sita over fei
{
// unit dependent
double taom,rou=1000; // rou, density of water, unit: kg/m^3

double MdotI,miu,sig,ep,fSoF,f1,f2;

if(!m_bSI)
rou=62.42796; // density of water: lbf/ft^3 or pcf

miu=-0.047*RoW+1.05;
if(miu<0)
miu=0;
sig=0.37;
ep=exp((SoF-miu)/sig);
fSoF=1/sig*ep*exp(-ep);

taom=rou*v*v/(400*RoW)*fSoF; // tao_max here is too small

```

```

//f1=-30*v+15; // Empirical coefficient. v=0.3, coef=9; v=0.2, coef=6.
f1=8;
if(RoW<=6)
    f2=1;
else
    f2=0.25*RoW-0.5; // R/W=8, f2=1.5; R/W=6, f2=1.
//          f2=0.5*RoW-2; // R/W=8, f2=2; R/W=6, f2=1
//          f2=0.4*RoW-1.4; // R/W=8, f2=1.8; R/W=6 f2=1
//          f2=1.5; // This one is for Flume Test 4, R/W=8 fei=65

taom*=f1*f2;

if(!m_bSI) // Taomax, if metric N/m^2, if English psf
    taom/=32.174; // gc=32.174 lbm-ft/(lbf-s^2)

if(taom<m_pSoil->m_dCriticalShearStress) // if tao_max<tao_critical,
no erosion occurs
return 0.0;

MdotI=Interp(m_pSoil->m_pLayer->m_dShearStress,m_pSoil->m_pLayer-
>m_dScourRate,\
    m_pSoil->m_pLayer->m_nPoints,taom,false); // Scour rate unit is
mm/hr even for English units

return MdotI;
}

void CSRICOSDoc::OnRun()
// If English units are used, the bank coordinates must also be in foot.
{
//Engine *ep; // Try to use Matlab engine
//ep=engOpen(NULL);

// In the constructor of CGeometry, m_pdxxy[0][0]=dLargeNum;
if(fabs(m_pGeometry->m_pdxxy[0][0]-dLargeNum)<1e-4)
{
    AfxMessageBox("Please input the channel coordinates!\n");
    return;
}
else if(fabs(m_pGeometry->m_dWidth)<1e-2)
{
    AfxMessageBox("Please input river width!\n");
    return;
}

m_pGeometry->RetrieveAssign();

// This file stores XY of migrated channels in binary format for
Matlab.Suffix & extension will be added.
// m_szOutFullFn=m_szInstallPath+m_szOutFn;
// m_szOutFullFn="..\Data\\"+m_szOutFn;

```

```

//On the View dialogue, if PeriodTo<=0, output one step specified by
PeriodFrom
//if PeriodTo>=1, output migration superposition graph for each step
//For debug purpose
if((int)m_dPeriodTo<=0)
    nDebugStep=(int)m_dPeriodFrom; // borrow the space for temporary
use,1-based.
else
    nDebugStep=nLargeInt+100; // disabled
if(m_iPeriodUnit==0) // for temporary use. No.
{
    bIsBank=false; // Unit: year, center line
    isRightBank=0;
    m_szOutFullFn="..\Data\TMP_"+m_szName+"_C";
}
else if(m_iPeriodUnit==1)
{
    bIsBank=true; // Unit: Month
    isRightBank=1; // It is the right bank.
    m_szOutFullFn="..\Data\TMP_"+m_szName+"_R";
} // Use "\" instead of "/". "\" becomes "\" when it is passed to
Matlab.
else if(m_iPeriodUnit==2)
{
    bIsBank=true; // Unit: Day
    isRightBank=-1; // It is the left bank.
    m_szOutFullFn="..\Data\TMP_"+m_szName+"_L";
}

m_nCurRun=1; // Used for reporting the location of error.
m_nTotalRun=1; // True for constant flow and one hydrograph
// CString erMsg;

double *Q=NULL,*X=NULL,*M0; // Allow this dynamic allocation. It occurs
only once.
double (*riskXY)[2];

int i,j,nEfDay=0,flg;
m_nPt=m_pGeometry->m_nPt;
// m_nP0=m_pGeometry->m_nP0; // The matching process is done here.

time_t sec1,sec2,dt;
char stime[255],tStr[500]; // tStr is a temporary string.
m_fitCirT=0;

//If redistributed only once. m_nPt is a constant after that.
memcpy(m_pdxym_pGeometry->m_pdxym_nPt*2*sizeof(double));

M0=m_dM0;
for(i=0;i<nMaxPt;i++,*M0+=0.0); // In the beginning, it's zero.
// Then it stores the accumulated migration distance of
the latest step.

if(m_pdM1p!=NULL)

```

```

        delete[] m_pdM1p;
    if(m_pdT1p!=NULL)
        delete[] m_pdT1p;

    if(m_pWater->m_iConstant==0) // Constant Q or V
    {
        //m_pWater->m_dTimeStep!=24, this can occur only when flow is constant.
        double timeStep=m_pWater->m_dTimeStep/24; // timeStep unit is DAY,
                                                    // m_pWater->m_dTimeStep unit
        is in HOUR
        //count of the number of time steps
        //when m_pWater->m_dTime's unit is day
        if(m_pWater->m_iTimeUnit==0)
            m_nTimeStep=int(m_pWater->m_dTime/timeStep);

        //when m_pWater->m_dTime's unit is hour
        else
            m_nTimeStep=int(m_pWater->m_dTime/m_pWater->m_dTimeStep);

        Q=new double[m_nTimeStep];
        m_pdM1p=new double[m_nTimeStep];
        m_pdT1p=new double[m_nTimeStep];
            //m_pdM1p[0]=0; // Initial point, zero migration. // don't
            do this.

        for(i=0;i<m_nTimeStep;i++)
        {
            Q[i]=m_pWater->m_dDischarge;
            m_pdT1p[i]=i+1; // time steps. unit: m_pWater->m_dTimeStep
(hours)
        }

        time(&sec1);
        OneHydrograph(m_dM0,Q,true);
        time(&sec2);

        dt=sec2-sec1;
        sprintf(stime,"Constant flow, Step number=%d, Total time=%d
sec\n\
            Time for fitting circles: %f sec\nTime for applying
hyperbola model: %f sec"\
            ,m_nTimeStep,dt,m_fitCirT,dt-m_fitCirT);
        AfxMessageBox(stime);
    }

    else if(m_pWater->m_iConstant==1) // Hydrograph
    {
        int sign=m_InputPlots.ReadHydroFileOf2Types(m_pWater-
>m_strFileName,\
            m_pWater->m_dTimeStep,X,Q,m_nTimeStep); // Space for X,Q is
allocated inside this function.

        // Send data to Matlab,test, failed
        //mxArray *mxQ;

```

```

//mxQ=mxCreateDoubleMatrix(1,m_nTimeStep,mxREAL);
//memcpy((char *)mxGetPr(mxQ), (char *)Q,m_nTimeStep*sizeof(double));
//engPutVariable(ep,"Q0",mxQ);

    m_pdM1p=new double[m_nTimeStep];
    m_pdT1p=new double[m_nTimeStep];

    for(i=0;i<m_nTimeStep;i++)
        m_pdT1p[i]=i+1; // time steps. unit: m_pWater->m_dTimeStep
(hours)

    if(sign==0) // If the hydrograph file is USGS format, time step
has to be 24 hours.
        m_pWater->m_dTimeStep=24;
    else if(sign==-1) // Failed to read the hydrograph file.
        return;
    time(&sec1);
    nEfDay=OneHydrograph(m_dM0,Q,true);
    time(&sec2);

    dt=sec2-sec1;
    sprintf(stime,"One Hydrograph, Number of total days=%d,Effective
days=%d Time=%d sec\n",\
        m_nTimeStep,nEfDay,dt);
    AfxMessageBox(stime);
}

else if(m_pWater->m_iConstant==2) // Risk Analysis
{
    double dMean,dStd;
    double nrc[2]={0,1}; // for Matlab RNG only.
    int flag,nRun=m_pWater->m_nRealization;
    m_nTotalRun=nRun;

    //Stores the # of points of the final migrated river for each run.
    int *nPtR=new int[nRun+1]; // R---Risk, # of points for each
channel.

    // Code changed. The # of
points is the same for each channel.
    nPtR[0]=m_nPt; // Initial channel

    flag=m_pWater->GetStatistics2(dMean,dStd); // deal with
hydrograph.txt first,
    if(flag<0) // Failed to read the hydrograph file. Error message
given.
        return;

    m_nTimeStep=(long)(m_pWater->m_dPredictTime*1); // 365 days/year
should be (*365)
    riskXY=new double[(nRun+1)*(int)(m_nPt*1.2)][2];
    // riskXY stores the final channel of each hydrograph.
    // If the points are redistributed for each flow, the maximum
number of points may be larger than nPt.

```

```

// If the # of points is fixed for redistribution, it's better to
use a 3-D array.

Q=new double[m_nTimeStep]; // Q/V of one generated hydrograph.
nrc[0]=m_nTimeStep;

// Assign initial channel to riskXY.
memcpy(riskXY,m_pdxym_nPt*2*sizeof(double));
time(&sec1);
srand((unsigned int)sec1); // set seed only once for each risk
analysis. for C RNG.

int nAcuPt=m_nPt; // Accumulated # of points.
for(i=0;i<nRun;i++)
{
    m_nCurRun=i+1;
    lognRnd(Q,dMean,dStd,m_nTimeStep); // Pure C code
    //CLognrnd(Q,dMean,dStd,nrc); // Compiled Matlab function

    m_nPt=m_pGeometry->m_nPt;

    M0=m_dM0;
    for(j=0;j<nMaxPt;j++,*M0+=0); //This step is time consuming.

    memcpy(m_pdxym_pGeometry->m_pdxym_nPt*2*sizeof(double));
    try
    {
        flg=OneHydrograph(m_dM0,Q,false);
        // Migrated curve is stored in m_pdxym
        nEfDay+=flg;
        if (flg<0) // no circle was fitted.
        {
            sprintf(tStr,"Error in OneHydrograph.
Hydrograph No. %d/%d; Step No. %d\n",i+1,nRun,m_nTimeStep);
            AfxMessageBox(tStr);
            return;
        }
    }
    catch(CString erMsg)
    {
        sprintf(tStr,"\nError in OneHydrograph.
Hydrograph No. %d/%d; Step No. %d\n", i+1, nRun, m_nTimeStep);
        AfxMessageBox(erMsg+tStr);
        return;
    }
    nPtr[i+1]=m_nPt;
    // m_nPt is the # of data points for this final migrated channel.

    memcpy(riskXY+nAcuPt,m_pdxym_nPt*2*sizeof(double));
    nAcuPt+=m_nPt;
}
time(&sec2);
dt=sec2-sec1;

```



```

        sprintf(stime,"Runs=%d, Days in 1 run=%d, Days calculated=%d,
Time=%d sec\n",\
            nRun,m_nTimeStep,nEfDay,dt);
        AfxMessageBox(stime);

int nt=nRun+1; // Total # of curves to be written to the file
    m_szOutFullFn=m_szInstallPath+m_szOutFn+"Risk.dat";
    FILE *fp=fopen(m_szOutFullFn,"wb");
    if(fp==NULL)
    {
        AfxMessageBox("Can't open file:"+ m_szOutFullFn +"for risk
analysis output");
        return;
    }
    nAcuPt=0; // Accumulated # of points
    fwrite(&nt,sizeof(int),1,fp);
    // Total # of curves to be written to the file
    int count1,count2;
    for(i=0;i<nRun+1;i++)
    {
        try
        {
            count1=fwrite(nPtr+i,sizeof(int),1,fp);
            count2=fwrite(riskXY+nAcuPt,sizeof(double),nPtr[i]*2,fp);
// Each curve might have different # of points.
        }
        catch(CString erMsg)
        {
            sprintf(tStr,"\nFunction OnRun(), error in writing
the file. Curve No.=%d\n",i+1);
            AfxMessageBox(erMsg+tStr);
            return;
        }
        nAcuPt+=nPtr[i];
    }
    fwrite(&m_pGeometry->m_dX0,sizeof(double),1,fp);
    fwrite(&m_pGeometry->m_dY0,sizeof(double),1,fp);
    fclose(fp);
}
m_bRunSuccess=true;

if(X!=NULL)
    delete[] X;
if(Q!=NULL)
    delete[] Q;
// if(riskXY!=NULL)
//     delete[] riskXY;
}

int CSRICOSDoc::OneHydrograph(double *M0,double *Q,bool bOutput)
// Used by MEANDER program only.
// M0 stores existing migration distance. M0=m_dM0.
// Q is the hydrograph

```

```

// if output==true, output intermediate result
// m_nTimeStep is the length of the hydrograph.
// No dynamic memory allocation inside this function.
// It will be called thousands of times.
// Before calling this function, allocate space for m_pdx..., cal
m_nTimeStep
// After calling this function, release the memory.
// If it succeeds, return 1; otherwise return 0.
{
    int nEfDay=0,m,nNum[5];

    double (*xyHolder)[2];
//    m_nArc=m_pGeometry->m_nArc; // not necessary

    FILE *fp=NULL;
    if(bOutput==true)
    {
        int nt=m_nTimeStep+1;
        // Total # of curves to be written to the file
        m_szOutFullFn+=".dat";
        fp=fopen(m_szOutFullFn,"wb");
        if(fp==NULL)
        {
            AfxMessageBox("Can't open file"+m_szOutFullFn+"for
writing coordinates of migrated channels.");
            return 0;
        }
        // sizeof(int) should be 4. Use "integer*4" in Matlab
        fwrite(&nt,sizeof(int),1,fp);
        // Total # of curves to be written to the file
    }

//    m_pGeometry->m_dArg[11]=-1; //<0:plot nothing; 1: plot channel &
circles 2: plot R/W 3: plot both
//Initially, might be: m_dArg[11]=3
//Move it inside the m_i loop
    m=0; // # of points for fitting a quadratic curve.

    m_pGeometry->m_dArg[19]=m_pWater->m_dTimeStep;
    // For plotting titles of Matlab graphs

    for(m_i=0;m_i<m_nTimeStep;m_i++)
    // global m_i, let OneFlow(..) know which flow it is.
    {
        nEfDay++; // # of channels for which circles are fitted.

//        if(m_i==nDebugStep-1 || nDebugStep>nLargeInt ||
((int)m_dPeriodFrom-1<=m_i && m_i<=(int)m_dPeriodTo-1))
            if(m_i==nDebugStep-1 || ((int)m_dPeriodFrom-1<=m_i &&
m_i<=(int)m_dPeriodTo-1))
                m_pGeometry->m_dArg[11]=3;
                // Plot fitted circles & R/W for debug
            else
                m_pGeometry->m_dArg[11]=-1;

```

```

        m_pGeometry->m_dArg[20]=m_i+1;
// Current step,1=Initial. For plotting titles of Matlab graphs.

// Output XY coordinates for a specified step.
if(m_i==nDebugStep-1)

        m_pGeometry->m_dArg[18]=1;
        // non-0, output bank coordinate file of this step.
else
        m_pGeometry->m_dArg[17]=0;
        // 0, don't output XY file for this step.

try
{
//         time_t sec1,sec2;
struct _timeb tBuf1,tBuf2;
//         time(&sec1);
        _ftime(&tBuf1);

if(m_i==0) // For the first fitting, use spacingCoef,
        m_pGeometry->m_dArg[16]=-1;
else
{ // For the rest, use fixed number of points.
        m_pGeometry->m_dArg[16]=m_nPt;
        m_pGeometry->m_dArg[17]=m;
}
        m_pGeometry->m_dArg[20]=m_i+1;
//Step No. 1-based. Matlab function uses it for file name&graph title.

        nNum[0]=m_nPt;
        GeoInterface(m_pdx, nNum, m_pGeometry-
>m_dArg, m_dxyRF, m_nArcIdx);
        m_nPt=nNum[0];
        m_nArc=nNum[1];
        m=nNum[2];
        if(m_i==0)
// Find the # of the point for which M vs. t curve will be plotted.
        m_nP0=m_pGeometry->matchPoint(m_pGeometry-
>m_dX0, m_pGeometry->m_dY0, m_pdx, m_nPt);

        if(bOutput==true && m_i==0)
// Write the redistribution points to file as the original curve
        {
                fwrite(&m_nPt, sizeof(int), 1, fp);
// Total # of points on one bank
                fwrite(m_pdx, sizeof(double), m_nPt*2, fp);
// The initial curve. count is for test only.
        }

        _ftime(&tBuf2);
//         time(&sec2);
//         m_fitCirT+=sec2-sec1;

```

```

        m_fitCirT+=(tBuf2.time+tBuf2.millitm/1000)-(tBuf1.time +
tBuf1.millitm/1000);
    }
    catch(CString erMsg)
    {
        char progress[400];
        // Try to show progress in the main dialog.
        sprintf(progress, "\nHydrograph No. %d of %d, Step No.
%d of %d\n", m_nCurRun, m_nTotalRun, m_i+1, m_nTimeStep);
        CString tip="\n\nIt is recommended to output the
channel coordinates of this step.\n";
        AfxMessageBox(erMsg+tip+progress);
        return -1; // Stop the program
    }
    if(nNum[3]!=-2 && nNum[3]<=0)
    {
        char err[400];
        sprintf(err, "Error Code: %d.\n", nNum[3]);
        AfxMessageBox("No circle was fitted."+CString(err));
        return -1;
    }

    m_pGeometry->ChannelLen(m_pdLen, m_pdxy, m_nPt);
    // for calculating fei, sita.

    OneFlow(M0, Q[m_i], false, bOutput);
    if(m_pWater->m_iConstant<=1)
    // This is only for constant flow and one hydrograph
    {
        m_pdM1p[m_i]=M0[m_nP0];
        // Trace the migration of point m_nP0
        if(m_bSI) // If English, foot
            m_pdM1p[m_i]/=1000; // Change from mm to meter.
    }
    if(bOutput==true)
    {
        //Write the migrated bank to file
        fwrite(&m_nPt, sizeof(int), 1, fp);
        fwrite(m_pdxyNew, sizeof(double), m_nPt*2, fp);
        // make it float later
    }

    xyHolder=m_pdxy;
    // Let m_pdxy point to m_pdxyNew and make it a bank to be moved
    m_pdxy=m_pdxyNew; // So m_pdxy also stores the final curve.
    m_pdxyNew=xyHolder;

    if(m_i==m_nTimeStep-1)
        continue;
}
if(fp!=NULL)
{
    fwrite(&m_pGeometry->m_dX0, sizeof(double), 1, fp);
    fwrite(&m_pGeometry->m_dY0, sizeof(double), 1, fp);
}

```

```

        fclose(fp);
    }
    return nEfDay;
}

int CSRICOSDoc::OneFlow(double *M0, double Qi, bool bSkip, bool& bOut)
// Given one flow(Qi or Vi), based on existing
// coordinates, RoverW, fei, sita, Fr
// to calculate migrated channel.
// M0[] stores existing migration distance and accumulates migration
// caused by this flow.
// M0=m_dM0;
//
// Sign of migration: (for m_dMdist, m_dM1dist)
// Walking from the 1st point to the last point:
// Rc>0(y">0) if the center is to the left side of the channel;
// Rc<0(y"<0) if the center is to the right side.
// If the Rc of the bend is >0, the outward migration is >0, the inward
// migration is <0;
// If the Rc of the bend is <0, the outward migration is <0, the inward
// migration is >0.
//
// If Rc>0, all the migration caused by this bend is >0
// If Rc<0, all the migration caused by this bend is <0
//
// Rc>0, M0>0: center of the circle is to the left side of the channel,
// migration is to the right side.
// Rc<0, M0<0: center of the circle is to the right side of the channel,
// migration is to the left side.
//
// bool bSkip: does the program try to skip this flow?
// Default value for bSkip is false(zero is returned).
// If bSkip==true, use this geometry to check whether the next flow can
// cause noticeable migration.
// Only when no bend/point has noticeable migration will this flow be
// ignored.
// Whenever a noticeable migration is observed, this function stops and
// returns 1.
//
// m_dMdist[], accumulated migration caused by the current bend and the
// bends in front of and after it.
// m_dMdist[] is the accumulation of m_dM1dist.
// m_dM1dist[], one time migration of the bend and the influenced part in
// front of and after it.
{
    int idx1, idx2, idxEnd, idx1st, j, k, m;
//    int nOnBendNo; // Point idxEnd is on this bend or the straight
// line after this bend.
    int nEfB=0; // # of effective bends which have significant
// migration. If nEfB=0, this flow can be ignored.

    double InflCoef=3.4; // Region of influence is assumed to be
// 0=<sita/fei<=InflCoef, forward influence

```

```

    double InflCoef2;
// backward influence, InflCoef2=sita/fei corresponds to Mmax/W=tiny
    double tiny=0.0001;
// Consider backward influence stops if Mmax/W<=tiny.

    double *M,*M1,dM0,dM;
    double vel,bendLen,sLen;
    double depth;
// meter, interpolated from the table on Water Dialogue, metric unit
first
    double A,miu,sigma,RoverW,fei,Fr;
    double MdotI,Mmax,Macu;
// Accumulated migration at the end of the day
    double ptc[2],pt0[2],ptN[2];
// center, point on curve, new position.
    double eps=1e-4;
// if(inc/Mmax>eps), consider noticeable migration occurs.
    eps=m_pWater->m_dManCoef;
    const static double gm=9.807,ge=32.174;
// Acceleration of gravity g in Metric: m/s^2, g in English: ft/s^2

#ifdef wwDEBUG // nMaxArc etc. are defined in GlobalConstVar.h
    static double Mmt[1000],Mat[1000]; // for debug only, Mmt=Mmax,
Mat=Maccumulated
    static double Mm1[nMaxArc][nMaxPt]; // like Mldt, it stores Mmax
    static double Mldt[nMaxArc][nMaxPt];
// It stores m_dMldist[] of each bend and m_dMdist[]
// X axis is channel length
    static double Mlidx[nMaxArc][2];
// Index of m_dMldist & Mm1 of each bend. Matlab accepts only double
type
    static double cXY[nMaxPt][6];
//Coordinates of original channel and new channel.Original 0,1;New 2,3.
    double tStep[2]; // tStep[0], time step in hours; tStep[0],
current step number(1=initial channel)
// Pass it to Matlab for the titles of the graphs.
    tStep[0]=m_pWater->m_dTimeStep;
    tStep[1]=m_i+1; // Make it 1-based.
#endif

M=m_dMdist;
M1=m_dMldist;
for(j=0;j<nMaxPt;j++,*M++=0.0,*M1++=0.0);

if(Qi>=0) // There is flow data for that day.
{
    if(m_pWater->m_iInputData==0)
// The input data is discharge, not velocity
        vel=Interp(m_pWater->m_pDV,m_pWater->m_nPointDV,Qi);
    else // It's a velocity hydrograph.
        vel=Qi;
}
else // When discharge is missing for the day, Q[i-1]<0
    vel=Qi;

```

```

if(m_pWater->m_iInputData==0) // The input data is discharge
    depth=Interp(m_pWater->m_pDW,m_pWater->m_nPointDW,Qi);
else if(m_pWater->m_iInputData==1) // The input data is velocity
    depth=Interp(m_pWater->m_pVW,m_pWater->m_nPointVW,Qi);

if(m_bSI)
    Fr=vel/sqrt(gm*depth); // Froude number, Metric units
else
    Fr=vel/sqrt(ge*depth); // English units

if(bSkip==false)
    memcpy(m_pdxNew,m_pdx,m_nPt*2*sizeof(double));
//If migration occurs, change it, or keep old ones.

// Calculate migration bend by bend.
for(j=0;j<m_nArc;j++)
{
    for(k=0;k<nMaxPt;m_dM1dist[k]=0,k++);
// not necessary,for debug only

    idx1=m_nArcIdx[j][0]; // Its index is already 0-based.
    idx2=m_nArcIdx[j][1];
    bendLen=m_pdLen[idx2]-m_pdLen[idx1];

    RoverW=fabs(m_dxyRF[j][2]/m_pGeometry->m_dWidth);
    fei=m_dxyRF[j][3];

    if(bIsBank==true && m_dxyRF[j][2]*isRightBank<0)
// Ignore inner bank.
        continue;
    if(bIsBank==true) //Reduce R/W by 0.5 since Po's equation
is based on geometry of center line.
        RoverW-=0.5;

//         fei=bendLen/fabs(m_dxyRF[j][2])*180/3.1415926;

//         if(bIsBank==true)
//             CalAmiuSigma_Po(A,miu,sigma,RoverW,fei,Fr);
//         else
//             CalAmiuSigma_Wei(A,miu,sigma,RoverW,fei,Fr);

    idxEnd=idx1;
// idxEnd corresponds to sita/fei=InflCoef. forward influence
// Calculate sita for the bend and the part it affects.
    while ((sLen=m_pdLen[idxEnd]-
m_pdLen[idx1])<=InflCoef*bendLen && idxEnd<m_nPt)
    {
        m_dSita[idxEnd]=sLen/fabs(m_dxyRF[j][2])*180/3.1415926;
        idxEnd++;
    }
    idxEnd--;

```

```

        InflCoef2=-sigma*sqrt(-2*log(tiny/A))+miu;
// backward influence, natural log
//InflCoef2=0; // for debug // Derived from Mmax/W=tiny
idxlst=idxl; // idxlst correspond to sita/fei=InflCoef2
if(InflCoef2<0)
{
    while((sLen=m_pdLen[idxlst]-m_pdLen[idxl]) >=
InflCoef2*bendLen && idxlst>=0)
    {
        m_dSita[idxlst]=sLen/fabs(m_dxyRF[j][2])*180/3.1415926; // sita<0
        idxlst--;
    }
    idxlst++;
}
// Calculate the migration for the bend and the part it affects.
for(k=idxlst;k<=idxEnd;k++)
{
    if(m_dxyRF[j][2]*M0[k]>0)
        dM0=fabs(M0[k]);
// Existing migration is in outward direction.
    else
        dM0=0;
        Mmax=CalMmax_Regr(A,miu,sigma,m_dSita[k],fei);
#ifdef wwDEBUG
        Mmt[k]=Mmax; // Mmt: maximum, debug
#endif
        // Manually decrease MdotI, for debug
        MdotI=CalMdotI(vel,RoverW,m_dSita[k]/fei);
        // Developed for MEANDER
//        MdotI=CalMdotI(vel)/5;
// Equation for pier scour
        Macu=Hyperbola(dM0,Mmax,MdotI,m_pWater->m_dTimeStep);
//in mm
#ifdef wwDEBUG
        Mat[k]=Macu; // Mat: accumulated, debug
#endif
        m_dMldist[k]=Macu-dM0;
        //It temporarily stores migration increment.
    }

    int rSign=1;
    // For debug output. It's the sign of the current arc.
    if(m_dxyRF[j][2]<0)
// If Rc<0, the sign of migration caused by this bend is defined as <0.
    {
        for(m=idxlst;m<=idxEnd;m_dMldist[m]=-
m_dMldist[m],m++);
        rSign=-1;
    }
    // Accumulated migration INCREMENTS.
    for(k=idxlst;k<=idxEnd;k++)
        m_dMdist[k]+=m_dMldist[k];

```



```

#ifdef wwDEBUG
    Mlidx[j][0]=idx1st+1;
    // Make it 1-based so that Matlab can use it directly.
    Mlidx[j][1]=idxEnd+1;
    for(k=idx1st;k<=idxEnd;k++)
    {
        Mm1[j][k]=Mmt[k]*rSign;
        Mldt[j][k]=m_dMldist[k];
    }
#endif
    idx1=-999; // for inserting a break point,debug
}
#ifdef wwDEBUG
Mlidx[m_nArc][0]=1;
// Pass it to matlab, for plotting accumulated migration
Mlidx[m_nArc][1]=m_nPt;

if(m_i==0) // To store the initial channel
    for(j=0;j<m_nPt;j++)
        //It will be kept and will not be changed for all steps.
        for(k=0;k<2;k++)
            cXY[j][k]=m_pdxxy[j][k];
for(j=0;j<m_nPt;j++)
{
    Mldt[m_nArc][j]=m_dMdist[j];
    // the accumulated migration caused by this flow
    for(k=0;k<2;k++)
        cXY[j][k+2]=m_pdxxy[j][k];
    // First save the channel before migration
}
#endif
// Move all the bends and the parts they affect.
for(j=0;j<m_nArc;j++)
{
    if(j==0)
        idx1=0;
    else
        idx1=m_nArcIdx[j][0]; // Its index is already 0-based.
    if(j<m_nArc-1)
        idx2=m_nArcIdx[j+1][0];
    else
        idx2=m_nPt;
// For Stolpa's Experiment, 1st bend's influence overtakes 2nd bend's
influence.
        //idx2=idxEnd+1;
// +1,so that the last affected point on the curve will be accounted
for. idxEnd is for the last bend, value not changed since last
assignment. Assume previous bend's influence doesn't overrun this
bend's influence.
    ptc[0]=m_dxyRF[j][0];
    ptc[1]=m_dxyRF[j][1];
    for(k=idx1;k<idx2;k++)
//Point No.idx2 is not to be included.Point with No.m_nPt doesn't exist.
    {

```

```

        if(fabs(m_dMdist[k])<1e-6)
            // No migration occurred at this point.
            continue;
        if(bSkip==true && fabs(m_dMdist[k])/Mmax>eps)
            // One point has noticeable migration.
            {
                nEfB+=1;
                return nEfB; // This flow cannot be ignored.
            }
        if(m_dxyRF[j][2]*m_dMdist[k]>0)
            dM0=fabs(m_dMdist[k]); // Migrate outward
        else
            dM0=-fabs(m_dMdist[k]); // Migrate inward

        if(m_bSI)
            dM=dM0/1000; // Change unit from mm to meter.
        else
            dM=dM0;
        pt0[0]=m_pdx[k][0];
        pt0[1]=m_pdx[k][1];
        extend(ptc,pt0,dM,ptN);
        m_pdxNew[k][0]=ptN[0];
        m_pdxNew[k][1]=ptN[1];

        M0[k]+=m_dMdist[k];
    }
}
#ifdef wwDEBUG
    for(j=0;j<m_nPt;j++) // Then save the channel after migration
        for(k=0;k<2;k++)
            cXY[j][k+4]=m_pdxNew[j][k];

    // if(m_i==nDebugStep-1 || nDebugStep>nLargeInt )
    // plot graphic output for debugging on demand.
    try
    {
        if( m_i==nDebugStep-1 || ((int)m_dPeriodFrom-1<=m_i &&
m_i<=(int)m_dPeriodTo-1) )

            cPlotMgrtForOneFlow(Mm1[0],M1dt[0],M0,M1idx[0],cXY[0],m_pdLen,m_n
ArcIdx,m_nPt,m_nArc,tStep);
    }
    catch(CString erMsg)
    {
        AfxMessageBox(erMsg);
        return nEfB;
    }
#endif

return nEfB;
}

```

VITA

Wei Wang was born in Anhui Province, China. In 1991 he enrolled in the Department of Geotechnical Engineering of Tongji University, Shanghai China and obtained his B.S. degree in July 1995. He then entered the Master program of the same university and worked under the guidance of Prof. Zhongde Tang as a research assistant for the Architectural Design & Research Institute of Tongji University, Division of Geotechnical Investigation & Design. After he obtained his M.S. degree in July 1998, he continued his work at the same company for three years. He designed numerous braced cuts for deep excavation in the urban areas of Shanghai. In September 2001, he became a doctoral student of Prof. Jean-Louis Briaud at Texas A&M University working on a project sponsored by TxDOT. He was also a teaching assistant for the course, *Introduction to Geotechnical Engineering*.

Mr. Wang can be reached at:

Wei Wang
Construction, Geotechnical, Structures
Department of Civil Engineering
Mail Stop 3136
Texas A&M University
College Station, TX, 77843