

## ABSTRACT

JONES, MATTHEW PAUL. Effect of Urban Stormwater BMPs on Runoff Temperature in Trout Sensitive Regions. (Under the direction of William F. Hunt and Daniel H. Willits.)

While the negative impact of warm urban stormwater runoff on coldwater stream environments has been studied, little is known about the effect of urban stormwater best management practices (BMPs) on runoff temperature. A monitoring study was conducted from May through October of 2005, 2006, and 2007 in western North Carolina, along the southeastern extent of United States trout populations, to examine the effect of urban stormwater BMPs on runoff temperature. The monitoring sites consisted of a stormwater wetland, wet pond, and four bioretention areas. Runoff temperatures at all monitoring locations significantly ( $p < 0.05$ ) exceeded the 21°C trout temperature threshold from June through September. Monitored runoff temperatures at a parking lot surrounded by a mature tree canopy and a parking lot covered with a light-colored chip seal were cooler than nearby unshaded and standard asphalt parking lots. Both the stormwater wetland and wet pond increased water temperatures significantly. Effluent temperatures from the wet pond were significantly warmer than flows from the stormwater wetland from June through September. At both sites, water temperatures were coolest at the bottom depths, and water was cooler than 21°C at the bottom of the stormwater wetland during the early summer and early fall, indicating the thermal benefit of an outlet structure that would draw water from these bottom depths. Water was significantly cooler after conveyance in buried pipes when discharged into the stormwater wetland and wet pond.

All of the bioretention areas monitored during the course of this study significantly reduced maximum stormwater temperatures; however, only bioretention areas smaller than 10% of their contributing watershed significantly reduced median stormwater temperatures. The larger bioretention areas provided evidence of

substantial reductions in runoff volume, which would reduce effluent thermal loads. Despite temperature reductions, all bioretention areas discharged effluent significantly warmer than 21°C during the summer months. Evaluation of bioretention temperature profiles showed that the coolest effluent temperatures could be obtained from bioretention areas with a soil depth between 90 and 120 cm. Due to its ability to reduce runoff temperatures and flows, bioretention areas are considered to be an effective treatment option for mitigating thermal pollution from urban stormwater runoff.

A computer model was developed to simulate the thermal dynamics of a bioretention area. The model used a Green-Ampt based approach to simulate bioretention hydraulics. Pavement and runoff temperature were calculated using a finite difference solution for thermal conduction within the pavement profile in conjunction with a surface heat balance. A number of analytical and empirical methods were used to estimate weather parameters and the antecedent soil temperature profile. Soil and water temperature profiles during infiltration were simulated using a model for conduction and convection in porous media that utilized separate energy equations for the fluid and solid phases. The bioretention thermal model was validated by comparing simulation results with temperature data collected during the course of the monitoring study. The majority of simulated storm events had a root mean squared error less than 2.0°C for bioretention effluent estimates. Predicted effluent temperatures were typically warmer than measured values between 10:00 and 17:00 hours, and cooler for the remainder of the day. A sensitivity analysis showed that effluent temperatures were most affected by input parameters related to the soil and pavement surface heat balances. Simulation results suggested that volume reductions had a larger impact on effluent thermal loads than temperature reductions. Overall, the bioretention thermal model was considered to serve as a valuable tool for predicting bioretention effluent temperatures in trout sensitive regions.

Effect of Urban Stormwater BMPs on Runoff Temperature  
in Trout Sensitive Regions

by  
Matthew Paul Jones

A dissertation submitted to the Graduate Faculty of  
North Carolina State University  
in partial fulfillment of the  
requirements for the Degree of  
Doctor of Philosophy

Biological and Agricultural Engineering

Raleigh, North Carolina

2008

APPROVED BY:

---

William F. Hunt  
Co-Chair of Advisory Committee

---

Daniel H. Willits  
Co-Chair of Advisory Committee

---

Garry L. Grabow

---

Aziz Amoozegar

## **BIOGRAPHY**

Matthew Paul Jones was born in Gainesville, Florida, on April 16, 1983. He is the son of Paul and Kathy Jones and has two younger brothers, Michael and Tyler. When he was 2 years old, his family moved to Dublin, Ohio, where they spent the next 9 years. After living in Newark, Delaware, for two years, Matthew's family moved to Mechanicsburg, Pennsylvania, where he attended high school. While attending Cumberland Valley High School, Matthew developed a strong interest in biology through his work with Randy Cassell, a biology teacher at Cumberland Valley. Matthew's interest in biology and problem solving, as well as his family's connections to North Carolina, led him to the Biological and Agricultural Engineering (BAE) Department at North Carolina State University in 2001. Matthew completed his Bachelor of Science in biological engineering in May of 2005 and began to pursue his graduate degree shortly thereafter under the direction of Dr. Bill Hunt, focusing on stormwater management. Later that year, Matthew married Lindsay Harrell, a fellow BAE graduate. During his graduate studies, Matthew decided to pursue his PhD under the direction of Dr. Bill Hunt and Dr. Dan Willits. Upon graduation, Matthew plans to use his knowledge and experience related to stormwater and environmental engineering to develop innovative stormwater treatment systems and contribute towards sustainable development.

## **ACKNOWLEDGEMENTS**

The author would like to thank many people who have contributed towards his research and supported his goals. Dr. Bill Hunt has been an integral part of the author's achievements in graduate school, providing invaluable support, advice, and friendship. The author would like to thank Dr. Dan Willits for his wisdom and advice related to the author's computer modelling efforts. The author would also like to thank Dr. Garry Grabow and Dr. Aziz Amoozegar for their support throughout his graduate studies. Jonathan Smith was valuable in sharing his knowledge of stormwater monitoring projects with the author. The author would like to thank Jon Calabria, Allen Caldwell, Eric Caldwell, Seth Nagy, and Jason Zink for their assistance in selecting research sites and support throughout the project. The North Carolina Department of Environment and Natural Resources, Division of Water Quality deserves recognition for funding the author's research.

The author would like to extend a special thanks to his family for always being there to support him and help him achieve his goals. The author is especially grateful for the love, support, and understanding of his wife, Lindsay, throughout his time in graduate school. The author's achievements are largely due to her encouragement and guidance.

# TABLE OF CONTENTS

List of Tables	viii
List of Figures	x
Nomenclature	xv
<b>Chapter 1:</b>	<b>1</b>
<b>Effect of Urban Stormwater BMPs on Runoff Temperature: Introduction and Overview</b>	
<b>Chapter 2:</b>	<b>6</b>
<b>Effect of Stormwater Wetlands and Wet Ponds on Runoff Temperature in Trout Sensitive Waters</b>	
Introduction	7
Materials and Methods	11
Site Description	12
Results and Discussion	17
Effect of Conveyance in Buried Pipes	17
Wet Pond Results and Discussion	19
Stormwater Wetland Results and Discussion	24
Comparison of Wetland and Wet Pond Results and Discussion	27
Alternative Outlet Structure Designs	28
Summary and Conclusions	29
Acknowledgements	31
References	32
Nomenclature	33
<b>Chapter 3:</b>	<b>34</b>
<b>Bioretention Impact on Runoff Temperature in Trout Sensitive Waters</b>	
Abstract	35
Introduction	36
Materials and Methods	39

Site Description	41
Results and Discussion	44
Conclusions and Summary	56
Acknowledgements	59
References	59

**Chapter 4: \_\_\_\_\_ 61**

**Effect of Urban Stormwater BMPs on Runoff Temperature: Additional  
Monitoring Study Observations**

Introduction	62
Methods and Materials	62
Flow Monitoring Equipment	62
Estimation of Vegetative Shading	68
Results and Discussion	69
Effect of Runoff Temperature Measurement Location	69
Effect of Tree Canopy on Runoff Temperature	71
Effect of Pavement Material on Runoff Temperature	73
Stormwater Wetland and Wet Pond Observations from 2007 Monitoring	76
References	78

**Chapter 5: \_\_\_\_\_ 79**

**Bioretention Thermal Model: Background and Development**

Introduction	80
Assessment of Need	80
Bioretention Overview	80
Literature Review	83
Modelling Soil Water Movement	83
Modelling Runoff Temperatures	88
Modelling Air Temperature	89
Modelling Solar Radiation	90

Modelling Soil Temperature Profiles	90
Soil Temperature during Infiltration	92
Estimating Soil Thermal Properties	95
Model Methodology	96
Programming Language	96
Model Procedure Overview	97
Rainfall Input File	98
Watershed Hydrology	100
Bioretention Hydraulics / Hydrology	100
Pavement and Runoff Temperatures	104
Soil Thermal Properties	109
Antecedent Soil Temperatures	110
Storm Soil and Water Temperature Profiles	111
Model Outputs	113
Additional Model Applications	114
Future Work	114
Code Optimization	116
User Interface	117
References	119
<b>Chapter 6:</b>	<b>125</b>
<b>Bioretention Thermal Model: Validation and Sensitivity Analysis</b>	
Model Validation	126
Validation Procedure	126
Validation Results	130
Sensitivity Analysis	138
Procedure	138
Results	140
References	151

## Appendices

Appendix A: Bioretention Thermal Model: Equations and Derivations_____	153
Appendix B: Bioretention Thermal Model: Visual Basic Code_____	176
Appendix C: Bioretention Thermal Model: Sample Save File and Rainfall Input File _____	303
Appendix D: Bioretention Thermal Model: User Interface Screenshots_____	307
Appendix E: Bioretention Thermal Model: Validation and Sensitivity Analysis Plots _____	311
Appendix F: Effect of Urban Stormwater BMPs on Runoff Temperature: Sample SAS Code and Analysis Output_____	338

# LIST OF TABLES

**Chapter 1:**  
**Effect of Urban Stormwater BMPs on Runoff Temperature: Introduction and Overview**

**Chapter 2:**  
**The Effect of Stormwater Wetlands and Wet Ponds on Runoff Temperature in Trout Sensitive Waters**

<b>Table 1:</b> 30-year normal (1976-2006) weather data for Asheville, NC (NC Climate Office, 2007)	14
<b>Table 2:</b> 30-year normal (1976-2006) weather data for Lenoir, NC (NC Climate Office, 2007)	16
<b>Table 3:</b> Lenoir wet pond temperature summary	20
<b>Table 4:</b> Asheville stormwater wetland temperature summary for 2005 and 2006	25

**Chapter 3:**  
**Bioretention Impact on Runoff Temperature in Trout Sensitive Waters**

<b>Table 1:</b> Bioretention flow monitoring strategy for each monitoring location	40
<b>Table 2:</b> Bioretention site characteristics	44
<b>Table 3:</b> Median summary statistics for the bioretention area at each location	45
<b>Table 4:</b> Observed and 30-year average (1971-2000) rainfall depths near bioretention locations	45
<b>Table 5:</b> Measurable inflow and outflow events at each bioretention area	51

**Chapter 4:**  
**Effect of Urban Stormwater BMPs on Runoff Temperature: Additional Monitoring Study Observations**

<b>Table 4.1</b> Measured and 30-year normal (1971-2000) rainfall depths (cm) for weather stations near monitoring locations	77
--	----

**Chapter 5:**  
**Bioretention Thermal Model: Background and Development**

<b>Table 5.1</b> Asphalt and gravel physical and thermal properties (Barber, 1957)	108
--	-----

<b>Table 5.2</b>	Soil thermal properties included in the model interface_____	109
<b>Table 5.3</b>	Input parameters for antecedent soil temperature prediction_____	110
<b>Table 5.4</b>	Description of model outputs_____	114
<b>Table 5.5</b>	Grid spacing, time step, and storage interval for model calculations____	117

### **Chapter 6:**

#### **Bioretention Thermal Model: Validation and Sensitivity Analysis**

<b>Table 6.1:</b>	Rainfall depths required for validation_____	127
<b>Table 6.2</b>	Model inputs used for validation_____	129
<b>Table 6.3</b>	Validation simulation results for storm events at each site_____	133
<b>Table 6.4</b>	Simulation outputs for events used in sensitivity analysis before any input adjustments were made_____	140
<b>Table 6.5</b>	Summary of input parameters with low sensitivity_____	140
<b>Table 6.6</b>	Summary of sensitivity analysis results_____	151

# LIST OF FIGURES

## Chapter 1:

### Effect of Urban Stormwater BMPs on Runoff Temperature: Introduction and Overview

## Chapter 2:

### The Effect of Stormwater Wetlands and Wet Ponds on Runoff Temperature in Trout Sensitive Waters

<b>Figure 1:</b> Site photo and simplified equipment layout for Asheville stormwater wetland	13
<b>Figure 2:</b> Site photo and simplified equipment layout for Lenoir wet pond	15
<b>Figure 3:</b> Temperature of runoff exiting parking lot and entering the wetland	18
<b>Figure 4:</b> Influent and effluent temperatures at the wet pond during July	21
<b>Figure 5:</b> Mean diurnal temperature range at specified depths in the wet pond	22
<b>Figure 6:</b> Temperature distribution within the wet pond water column near the outlet	23
<b>Figure 7:</b> Temperature distribution within the wetland water column near the outlet	27
<b>Figure 8:</b> Illustration of modified outlet structure drawing from bottom waters	29

## Chapter 3:

### Bioretention Impact on Runoff Temperature in Trout Sensitive Waters

<b>Fig. 1:</b> Cross-section of a typical bioretention area	38
<b>Fig. 2:</b> Photo of Asheville bioretention and diagram of equipment layout	42
<b>Fig. 3:</b> Photo of Lenoir bioretention and diagram of equipment layout	43
<b>Fig. 4:</b> Photo of Brevard east bioretention and diagram of equipment layout	43
<b>Fig. 5:</b> Photo of Brevard west bioretention and diagram of equipment layout	44
<b>Fig. 6:</b> Influent and effluent temperatures at the Asheville bioretention area	46
<b>Fig. 7:</b> Influent and effluent temperatures at the Lenoir bioretention area	48
<b>Fig. 8:</b> Influent and effluent temperatures at the Brevard East bioretention area	49
<b>Fig. 9:</b> Influent and effluent temperatures at the Brevard West bioretention area	50

**Fig. 12:** Brevard East soil temperature at a depth of 40 cm during 2007 \_\_\_\_\_ 54

#### **Chapter 4:**

#### **Effect of Urban Stormwater BMPs on Runoff Temperature: Additional Monitoring Study Observations**

**Figure 4.1** V-notch weir installed at the Asheville stormwater wetland outlet \_\_\_\_\_ 63

**Figure 4.2** Pulley-float water level recorder (left) and shaft connection (right) \_\_\_\_\_ 64

**Figure 4.3** Metal frame installed to protect inflow monitoring equipment at Brevard west bioretention (left), and damaged inflow monitoring equipment before protective frames were installed (right) \_\_\_\_\_ 65

**Figure 4.4** Water level recorded within outlet weir box at Lenoir bioretention, illustrating shifts in depths measurements, which should be 0 between storm events \_\_\_\_\_ 66

**Figure 4.5** Composite overhead image of Brevard east bioretention (left) and camera setup (right) \_\_\_\_\_ 69

**Figure 4.6** Painted overhead photographs used to evaluate vegetative shading at the Brevard east bioretention site \_\_\_\_\_ 69

**Figure 4.7** Apparatus installed to capture runoff directly from wet pond parking lot before installation (left) and during a storm event (right) \_\_\_\_\_ 70

**Figure 4.8** Temperature measurements collected within the surface runoff capture apparatus and culvert \_\_\_\_\_ 71

**Figure 4.9** Contributing watersheds for Lenoir bioretention area (left) and wet pond (right) \_\_\_\_\_ 72

**Figure 4.10** Measured runoff temperatures at the Lenoir bioretention area and wet pond \_\_\_\_\_ 73

**Figure 4.11** Contributing watershed for Asheville bioretention before (left) and after (right) application of light colored chip seal \_\_\_\_\_ 74

**Figure 4.12** Recorded runoff temperatures at the Asheville bioretention area and stormwater wetland during the 2006 monitoring period \_\_\_\_\_ 75

<b>Figure 4.13</b> Aerial photograph showing location of Asheville stormwater wetland and bioretention area (Parking surfaces shaded red)	76
<b>Figure 4.14</b> Photograph of stormwater wetland in June (left) and wet pond in July (right) during the drought of 2007	77

## Chapter 5:

### Bioretention Thermal Model: Background and Development

<b>Figure 5.1</b> Typical bioretention cross-section	81
<b>Figure 5.2</b> Overflow structure at Asheville bioretention recently after construction	82
<b>Figure 5.3</b> Flow chart of overall model procedure	98
<b>Figure 5.4</b> Soil water characteristic curve for the Brevard east bioretention area with the point of saturation and field capacity identified	102
<b>Figure 5.5</b> Collection apparatus for rainfall temperature measurements	107
<b>Fig 5.6</b> Measured rainfall temperatures vs. ambient air temperatures at the Asheville stormwater wetland	108
<b>Figure 5.7</b> Screenshot of model interface	118

## Chapter 6:

### Bioretention Thermal Model: Validation and Sensitivity Analysis

<b>Figure 6.1</b> Hyetograph for a storm that would not meet conditions for validation due to extended periods without rainfall at 12:15, 13:35, 16:10, and 17:15	127
<b>Figure 6.2</b> Comparison of simulated (Sim) and measured (Meas) runoff and effluent temperatures for a storm on 06-14-07 at the Brevard west bioretention area	131
<b>Figure 6.3</b> Portion of validation simulations exceeding each root mean squared error value	132
<b>Figure 6.4</b> Distribution of RMSE for storms at different times during the day	134
<b>Figure 6.5</b> Distribution of mean error for storms at different times during the day	135
<b>Figure 6.6</b> Distribution of RMSE for different storm rainfall depths	136
<b>Figure 6.7</b> Distribution of mean error for different storm rainfall depths	136

<b>Figure 6.8</b> Comparison of errors in mean effluent temperature results when using simulated runoff temperatures and measured runoff temperatures for model calculations_____	137
<b>Figure 6.9</b> Simulated (Sim) and measured (Meas) runoff and effluent temperatures for the Lenoir storm used in the sensitivity analysis_____	138
<b>Figure 6.10</b> Simulated (Sim) and measured (Meas) runoff and effluent temperatures for the Brevard west storm used in the sensitivity analysis_____	139
<b>Figure 6.11</b> Sensitivity of effluent temperature to transmittance adjustments for the Brevard west location_____	141
<b>Figure 6.12</b> Effect of bioretention shading on effluent temperatures for the Brevard west location_____	142
<b>Figure 6.13</b> Sensitivity of effluent temperature to radiation adjustments for the Brevard west location_____	142
<b>Figure 6.14</b> Sensitivity of effluent temperature to air temperature adjustments for the Lenoir location_____	143
<b>Figure 6.15</b> Sensitivity of effluent temperature to wind speed adjustments for the Lenoir location_____	144
<b>Figure 6.16</b> Sensitivity of effluent temperature to relative humidity adjustments for the Lenoir location_____	144
<b>Figure 6.17</b> Sensitivity of effluent temperature to watershed area adjustments for the Lenoir location_____	145
<b>Figure 6.18</b> Sensitivity of effluent temperature to drain depth adjustments for the Lenoir location_____	146
<b>Figure 6.19</b> Sensitivity of effluent temperature to soil thermal conductivity adjustments at the Lenoir bioretention area_____	147
<b>Figure 6.20</b> Sensitivity of effluent temperature to subsoil saturated hydraulic conductivity adjustments for the Lenoir location_____	148

**Figure 6.21** Cross section of a bioretention area with an internal water storage layer

---

149

## Nomenclature

\* Denotes significance at 5% level

\*\* Denotes significance at 1% level

$T$  : Temperature ( $^{\circ}\text{C}$ , K)

$\alpha$  : Thermal diffusivity ( $\text{m}^2/\text{s}$ )

$t$  : Time (s, hr)

$z$  : Depth (m)

$k$  : Thermal conductivity (W/m K)

$q$  : Heat flux ( $\text{W}/\text{m}^2$ )

$h_c$  : Convective coefficient ( $\text{W}/\text{m}^2 \text{K}$ )

$\rho$  : Density ( $\text{kg}/\text{m}^3$ )

$C_p$  : Specific heat (kJ/kg K)

$\varepsilon$  : Porosity

$\vec{u}$  : Darcian velocity (m/s)

$G$  : Tortuosity parameter

$\sigma$  : Thermal conductivity ratio

$h_{sf}$  : Interfacial heat transfer coefficient ( $\text{W}/\text{m}^2 \text{K}$ )

### Subscripts

$m$  : Depth index

$F$  : Fluid phase

$S$  : Solid phase

$stg$  : Combined soil and stagnant fluid

### Superscripts

$t$  : Time index

$F$  : Fluid phase

$S$  : Solid phase

## **Chapter 1**

# **Effect of Urban Stormwater BMPs on Runoff Temperature: Introduction and Overview**

As advances in aquatic ecology have coincided with increased urbanization, it has become evident that warm stormwater runoff can negatively impact coldwater stream environments. While increased water temperatures can affect a variety of aquatic organisms, trout and salmon often receive special attention due to their economic and ecological importance. Specific effects of increased water temperatures on trout include increased feeding, disorientation, increased metabolism, reduced reproduction, and possible mortality. Thermal impacts of urban stormwater runoff are particularly a concern in western North Carolina, since this region of the state lies along the southeastern extent of United States trout populations. As a result, relatively small thermal impacts can cause water temperatures to exceed trout temperature thresholds. Stormwater best management practices (BMPs), such as stormwater wetlands, wet ponds, and bioretention areas, have been installed throughout the United States and around the world to reduce the negative impacts of urban stormwater on the aquatic ecosystem and decrease the risk of flooding; however, the effect of these systems on runoff temperature is largely unknown. Because these systems are designed to capture stormwater runoff from an impervious surface and often discharge that water into a stream, the potential impact of urban stormwater BMPs on runoff temperature is substantial.

A monitoring study was conducted at 6 stormwater BMP locations in western North Carolina in order to better understand the effect of urban stormwater BMPs on runoff temperature. The monitoring sites consisted of 4 bioretention areas, a stormwater wetland, and a wet pond. The monitoring study was conducted to examine the actual temperature dynamics of stormwater BMPs in the field, accounting for the effects of seasonal weather patterns, individual storm characteristics, and many other parameters that would be difficult to evaluate in a lab study. The objectives of the monitoring study were to (1) quantify the effect of urban stormwater BMPs on runoff temperature, (2) evaluate which BMPs may be best suited for use in trout sensitive regions, and (3) identify any design parameters

that could be modified to reduce thermal impacts or urban stormwater. This dissertation contains three chapters that pertain to the monitoring study. Chapter 2 discusses the results of the stormwater wetland and wet pond monitoring study in the form of a journal article. Chapter 3, also in journal article format, presents the results of the bioretention monitoring study. Chapter 4 discusses additional observations from the monitoring study, which were not included in Chapters 3 and 4, including the effect of watershed characteristics on runoff temperature and additional detail on some of the monitoring study methods.

Results of the BMP monitoring study demonstrated that bioretention could be beneficial for use in trout sensitive regions due to its ability to reduce runoff temperatures and effluent flows. However, monitoring results also suggested that a number of design parameters could affect effluent temperatures. In order to evaluate the effect of bioretention design parameters on effluent temperatures for a variety of system locations and weather patterns, a computer model was developed to simulate heat exchanges within a bioretention area. The bioretention thermal model was intended to allow designers to simulate the effluent temperatures discharged from various bioretention designs, allowing them to determine the design for their site that will result in the coolest effluent temperatures. Background on the model and related research, as well as a description of the calculation procedure and individual model components can be found in Chapter 5. Chapter 6 contains an analysis of the accuracy of the model, comparing simulation results with monitoring data, as well as an examination of the sensitivity of model outputs to various input parameters. Details of the model code, interface, validation, and sensitivity analysis can be found in the appendices.

The results of this research are intended to provide a thorough evaluation of the effect of urban stormwater BMPs on runoff temperature in trout sensitive regions. As concern over the preservation of coldwater aquatic ecosystems continues to develop, these findings could be applied to future stormwater regulations or other

mechanisms to help further reduce the negative impacts of urban stormwater on aquatic environments. There are a number of opportunities for future work, including pairing results of the monitoring study and modelling effort with studies on coldwater stream environments to quantify specific impacts to the aquatic ecosystem. Using the information presented in this dissertation, it should be possible to limit the negative thermal impacts associated with urban stormwater, preserving the coldwater stream environment and the valuable organisms that reside there.

## **Chapter 2**

# **Effect of Stormwater Wetlands and Wet Ponds on Runoff Temperature in Trout Sensitive Waters**

## **Effect of Stormwater Wetlands and Wet Ponds on Runoff Temperature in Trout Sensitive Waters**

**Abstract.** With increasing development in areas of trout sensitive waters, the effect of urban stormwater runoff temperature on the aquatic ecosystem has become a reason for concern. A study was conducted in western North Carolina, near the southeastern limit of trout populations, to determine the effect of stormwater wetlands and wet ponds on the temperature of urban stormwater runoff. Data were collected at a stormwater wetland in Asheville, NC, and a wet pond in Lenoir, NC, and included temperature measurements at the inlets, outlets, and at several depths within the best management practices (BMPs). Parking lot runoff temperatures were significantly ( $p < 0.05$ ) higher than the 21°C temperature threshold for trout during peak summer months and water temperatures consistently increased from the inlet to the outlet in the stormwater wetland and wet pond, implicating these BMPs as sources of thermal pollution. Despite similar inflows, effluent temperatures from the wet pond were significantly ( $p < 0.05$ ) warmer than those from the stormwater wetland for the period from June through September. Substantial cooling was observed as runoff was conveyed through buried corrugated metal and reinforced concrete pipes, which could be incorporated into BMP design to achieve thermal pollution mitigation goals. Temperatures at the bottom of the water columns were cooler than water leaving the current outlet structures, providing support for the installation of modified outlet structures in regions with cold water fisheries. Large temperature fluctuations near the water surfaces, where conventional outlet structures draw water, in both the wetland and wet pond indicate the difficulty in evaluating these systems as part of a temperature TMDL (Total Maximum Daily Load) program. Effluent temperatures decreased significantly ( $p < 0.05$ ) as storms progressed, emphasizing the importance of limiting effluent flows early in a storm for existing outlet structures.

## **Introduction**

Thermal pollution of surface waters began to receive much attention during the 1950's as advances in aquatic ecology coincided with an increased demand for power plants and industrial facilities (Langford, 1990). Studies determined that discharges of heated water, resulting from cooling processes within the facilities, could have substantial impacts on the aquatic ecosystem, reducing the abundance and diversity of aquatic organisms (Hocutt et al., 1980). Especially during the summer months, paved surfaces elevate runoff temperatures by capturing solar radiation and transferring this stored energy to runoff during rainfall events. Due to the low thermal conductivity and reflectivity of asphalt, heat from solar radiation concentrates near the surface and can lead to asphalt surface temperatures in excess of 60°C (Asaeda et al., 1996).

There are several heat transfer mechanisms that are altered in an urbanized environment, leading to higher runoff temperatures. Replacing native vegetation with pavement or a rooftop decreases the solar reflectivity of the ground surface and removes the tree canopy, causing more thermal energy to be captured at the anthropogenic surface (Akbari and Konopacki, 2005). Another impact is the decrease in evapotranspiration, which is an important cooling mechanism. Finally, increased runoff flowrates and volume, resulting from an increase in impervious area, enhances the potential for runoff to alter the temperature of a receiving body of water.

Streams exhibit natural diurnal temperature fluctuations, typically with a 0.6-0.8°C increase in water temperature for every 1°C increase in air temperature (Morrill et al., 2005). While diurnal stream temperature fluctuations are substantial, the direct discharge of thermally enriched runoff can lead to temperature increases in water bodies well above normal levels (Kieser et al., 2004). Even though temperature spikes are greatest in the afternoon, runoff can be discharged at

temperatures substantially higher than stream temperatures during the night and early morning (Lieb and Carline, 2000).

Thermally enriched runoff is a concern because of the negative effects it can have on an aquatic ecosystem. Many aquatic organisms, such as fish, amphibians, and macro-invertebrates, are ectotherms and require suitable habitat temperatures to maintain crucial biological functions. In these organisms, temperature plays an important role in dictating metabolic rates and reproductive processes, as well as other behavioral and physiological traits. While most fish species can tolerate slow, seasonal changes in temperature, rapid changes have been proven to be lethal (Agersborg, 1930). Trout and salmon species are among the fish most sensitive to water temperature changes and serve as important game fish in many parts of the country (U.S. Environmental Protection Agency, 2003). While the temperature preference ranges of many aquatic organisms have been examined, inconsistencies between laboratory and field research data have indicated that there are a variety of factors in a natural environment that make it difficult to predict actual fish behavior (Hocutt et al., 1980). In general, trout and salmon have been found to avoid water temperatures in excess of 21°C (Coutant, 1977).

In response to elevated temperatures, fish are believed to be able to detect temperature variations and relocate to an area within their preferred temperature range, if possible (Hocutt et al., 1980). Several studies have shown that although fish may be able to temporarily survive elevated water temperatures, irreparable damage may ultimately lead to death after temperature reductions (Hocutt et al., 1980). Substantial impacts on macro-invertebrate communities have also been associated with thermal discharges. A study at the Martins Creek Power Station, located near Easton, PA, on the Delaware River, found that during the warmest part of the year, the reduction in macroinvertebrate biomass downstream from a thermal discharge could exceed 99% (Coutant, 1962).

The full effect of elevated stream temperatures on aquatic ecosystems is unknown due in part to complex interactions between organisms in the same thermal niche (Huff et al. 2005). For example, a study of interactions between juvenile Colorado River cutthroat trout (*Oncorhynchus clarkii pleuriticus*) and brook trout (*Salvelinus fontinalis*) in Wyoming found that brook trout were more aggressive, consumed more food, and appeared more dominant than the cutthroat trout at warmer water temperatures (De Staso and Rahel, 1994). Another concern with elevated water temperatures is associated with parasites and disease. Fish may be more susceptible to diseases and parasites due to increased parasite and pathogen growth and reproduction associated with elevated water temperatures (Langford, 1990). An indirect effect on the aquatic ecosystem is a reduction in dissolved oxygen content corresponding to increased water temperature. While short periods of low dissolved oxygen content do not normally have a significant effect on aquatic organisms, periods lasting several days can be lethal (Nebeker, 1972).

North Carolina contains approximately 4,000 miles of streams capable of supporting trout, the most of any state in the Southeast USA. Due to topography and climate, most of these trout waters are located in the western part of the state and represent a portion of the southeastern extent of trout populations in the United States. Fishing activities are important to the North Carolina economy, with an estimated 1.3 million anglers fishing in North Carolina alone, spending more than \$1 billion a year (Dept. of Int. NC Report, 2001).

The Clean Water Act of 1972 and subsequent legislation have led to the installation of stormwater best management practices (BMPs) throughout the country. Even though most BMPs were not designed to mitigate thermal pollution, their role in capturing and treating stormwater runoff before it is discharged into a creek or river can affect the temperature of stormwater effluent. Research has found that the effluent water temperature can be greater than the temperature of the

incoming runoff in a wet pond (Kieser et al., 2004). The suspected reason for this increase is that most wet ponds are not adequately shaded and incoming solar radiation heats the water above the temperature of the ambient air. Increasing a wet pond's surface area, relative to the drainage area, reduces nutrient loading; however, a larger surface area may result in increased heating (Wu et al., 1996). While in many cases vegetative shading can lead to cooler water temperatures, it has been shown that when a pond is covered by vegetation in contact with the water surface, such as *Lemnaceae* (duckweed), water temperatures can be highly variable near the surface due to conduction of heat captured by the vegetation (Dale and Gillespie, 1976). A study of the thermal balance of an on-stream wet pond found that under calm weather conditions, water at the surface was on average 3.6°C warmer than the water 1 meter below the surface. It was also noted that moderate winds and inflows to the wet pond led to vertical water circulation within the pond and reduced the temperature difference (Van Buren et al., 2000). It has been observed that a stormwater wetland can mitigate thermal loading when well-shaded, with the net heat reduction attributed to evapotranspiration and infiltration (Kieser et al., 2004).

One limitation with mitigating thermal pollution through the use of stormwater wetlands and wet ponds is that the stormwater effluent typically cannot be cooled below the ambient air temperature (Kieser et al., 2004). During the summer months, the ambient air temperature is usually higher than the temperature of a cold water stream, indicating that even under favorable conditions, effluent from a stormwater wetland or wet pond would increase stream temperatures.

Transporting fluids through buried pipes for heating or cooling purposes has been used in industrial and HVAC processes for many years (Baxter, 1994). Due to its large heat capacity and insulation from surface weather fluctuations, the relatively constant temperature of soil at some depths can be used to accomplish heating and cooling goals. In addition to conventional roles in HVAC systems, research has

identified the potential of cooling stormwater runoff through conveyance in buried pipes, primarily through numerical modeling (Kieser et al., 2004).

Research has indicated that stormwater runoff can increase the temperature of coldwater streams and that this temperature increase has a direct impact on the aquatic ecosystem. The effectiveness of stormwater wetlands and wet ponds in mitigating thermal pollution has been tied primarily to vegetative cover; however, further research into the variability of water temperature with depth holds promise for better control techniques. Previous thermal pollution research has focused on BMPs designed to treat runoff from large urban drainage networks, leaving the effect of wetlands and wet ponds sized for individual developments on runoff temperatures unknown. By monitoring temperatures throughout a stormwater wetland and wet pond, it should be possible to identify which of these BMPs can effectively reduce runoff temperatures, as well as identify any design parameters that can be modified to improve the temperature reduction capacity of these systems. Additionally, the impact of conveyance systems on runoff temperature reduction will be investigated.

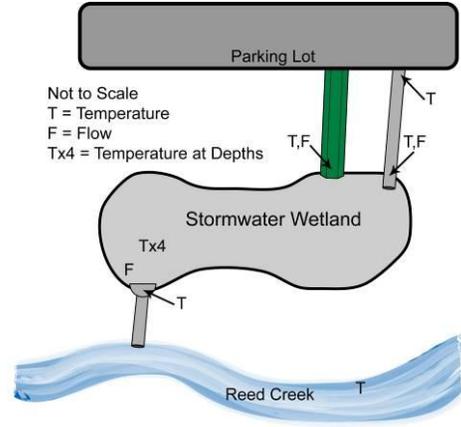
### **Materials and Methods**

Monitoring was conducted at a stormwater wetland in Asheville, NC, and a wet pond in Lenoir, NC. Temperature, flow, and rainfall were monitored at both sites, with measurements of temperature and flow logged at 5-minute intervals. Data were collected for the stormwater wetland from July through mid-October, 2005, and at both the stormwater wetland and wet pond from May through mid-October, 2006. For the stormwater wetland, flow was measured at all major inlets and outlets using v-notch weirs in conjunction with pulley-float stage recorders. Only stage was recorded within the wet pond. Rainfall data were collected at each site using tipping bucket rain gauges with a resolution of 0.25 mm (0.01"). Temperature measurements were collected with a combination of HOBO™ Water Temp Pro (H20-001) and HOBO™ 4 channel loggers (H08-008-04 & U12-008) with

temperature sensors attached (TMCX-HD). All temperature loggers were manufactured by Onset Computer Corporation.

### **Site Description**

The stormwater wetland was located in Asheville, NC (35°36'52"N, 82°33'48"W), on the campus of the University of North Carolina at Asheville, and received runoff primarily from an estimated 7,350 m<sup>2</sup> asphalt parking lot. The parking area was used routinely by faculty and staff of the university and was partially surrounded by mature trees. Beginning in spring of 2006, the uppermost section of the parking lot (3,820 m<sup>2</sup>) underwent construction. The stormwater wetland covered a 724 m<sup>2</sup> area, of which an estimated 70% was covered by vegetation during mid-summer of 2006. Vegetative cover was estimated by examining a series of 8 random overhead images and digitally comparing areas of open water to the total area. Water was conveyed from the parking surface to the stormwater wetland through a 168 m long, 71 cm (28 in) diameter buried corrugated metal pipe and a separate vegetated channel. Water was discharged from the wetland through a flashboard riser structure and corrugated metal pipe into Reed Creek, which is approximately 10 m from the wetland. Temperature probes were located inside a drop inlet at the parking lot, the corrugated metal pipe inlet to the wetland, the vegetated channel inlet, and the outlet structure. Additionally, 4 probes were spaced at 30 cm depths from the normal pool elevation to the base of a deep pool near the outlet structure (Figure 1). Throughout this text, depth refers to the distance below the static normal pool elevation. Temperature probes within the water column were housed inside a perforated PVC pipe to limit the effect of direct solar radiation on the probes during the summer of 2006, but were not shielded during the summer of 2005. Temperature probes were also used to measure the ambient air temperature and the temperature of Reed Creek upstream from the stormwater wetland outlet for reference.



**Figure 1:** Site photo and simplified equipment layout for Asheville stormwater wetland

The weather in Asheville was typical for this region of North Carolina, with diurnal air temperature fluctuations around 12° C during the peak summer months (Table 1). Mean ambient air temperatures exceeded the 21° C water temperature threshold for trout during the months of June through August in both 2005 and 2006.

**Table 1:** 30-year normal (1976-2006) weather data for Asheville, NC (NC Climate Office, 2007)

	<b>Apr</b>	<b>May</b>	<b>June</b>	<b>Jul</b>	<b>Aug</b>	<b>Sept</b>	<b>Oct</b>	<b>Annual</b>
<b>Normal Monthly Max Temperature (°C)</b>	19.3	23.5	27.1	29.1	28.3	24.9	19.8	19.1 <sup>a</sup>
<b>Normal Monthly Min Temperature (°C)</b>	6.6	11.2	15.3	17.5	16.9	13.6	7.2	7.3 <sup>a</sup>
<b>Normal Monthly Mean Temperature (°C)</b>	12.9	17.3	21.2	23.3	22.6	19.3	13.6	13.2 <sup>a</sup>
<b>2005 Monthly Mean Temperature (°C)</b>	13.4	16.8	21.7	24.4	24.1	20.8	14.8	13.35 <sup>c</sup>
<b>2006 Monthly Mean Temperature (°C)</b>	15.9	16.9	21.5	24.2	24.6	18.4	12.4	13.53 <sup>c</sup>
<b>Normal Monthly Precipitation (cm)</b>	8.0	9.0	8.2	7.5	8.5	7.6	6.1	94.7 <sup>b</sup>

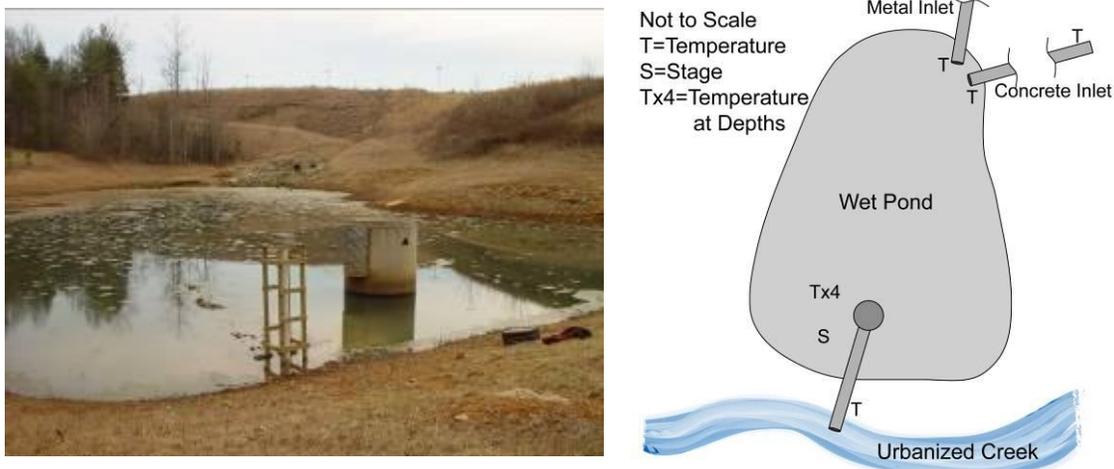
a: Mean

b: Total

c: Obtained from different weather station due to missing data

The wet pond was located in Lenoir, NC (35°54'1"N, 81°31'18"W), and captured runoff from an estimated 56,500 m<sup>2</sup> of rooftop and asphalt parking lot. The wet pond's surface area was 2,545 m<sup>2</sup>, and it received water from a 122 cm (48 in) diameter corrugated metal pipe and a 122 cm (48 in) diameter reinforced concrete pipe. The corrugated metal pipe had a perennial discharge into the wet pond, which likely originated from groundwater interception. The outlet structure was designed for water to exit the wet pond through two 60-cm holes in the concrete structure; however, poor seals lowered the normal pool elevation approximately 120 cm below these outlet holes. There was a consistent flow through the 25-cm diameter

corrugated plastic tubing at the base of the outlet structure, likely due to a leak in the emergency drawdown pipe. This wet pond discharged water directly into a heavily urbanized creek. There was a substantial amount of algae covering the pond for much of the year, in part because the wet pond received essentially no shading from vegetation. Temperature probes were located inside a drop inlet at the parking lot, inside the corrugated metal pipe inlet to the wet pond, inside the reinforced concrete pipe inlet, inside the outlet pipe, as well as 4 depths near the outlet structure (Figure 2). Stormwater runoff entering at the monitored drop inlet traveled through 320 m of buried pipe and was combined with flow from a number of additional drop inlets before entering the wet pond through the reinforced concrete pipe inlet. A temperature probe, housed within a naturally aspirated solar radiation shield, was used to measure the ambient air temperature for reference.



**Figure 2:** Site photo and simplified equipment layout for Lenoir wet pond

Normal air temperatures at the Lenoir site were slightly warmer than the Asheville location, and the annual mean air temperature for 2006 was warmer than the 30-year normal (Table 2). Like the Asheville site, mean ambient air temperatures in Lenoir exceeded 21° C during the months of June through August.

**Table 2:** 30-year normal (1976-2006) weather data for Lenoir, NC (NC Climate Office, 2007)

	<b>Apr</b>	<b>May</b>	<b>June</b>	<b>Jul</b>	<b>Aug</b>	<b>Sept</b>	<b>Oct</b>	<b>Annual</b>
<b>Normal Monthly Max Temperature (°C)</b>	21.1	25.2	28.7	30.9	30.0	26.7	21.6	20.7 <sup>a</sup>
<b>Normal Monthly Min Temperature (°C)</b>	5.7	11.4	16.1	18.6	17.7	14.0	6.9	7.1 <sup>a</sup>
<b>Normal Monthly Mean Temperature (°C)</b>	13.4	18.3	22.4	24.7	23.9	20.4	14.3	13.9 <sup>a</sup>
<b>2006 Monthly Mean Temperature (°C)</b>	16.4	17.2	22.9	25.2	25.6	19.6	13.4	14.9 <sup>a</sup>
<b>Normal Monthly Precipitation (cm.)</b>	10.4	11.9	11.3	11.2	9.8	11.3	9.2	125.0 <sup>b</sup>

a: Mean

b: Total

Statistical analysis was conducted using SAS<sup>®</sup> software, Version 9 (SAS Institute Inc., Cary, NC; Appendix F). The potential impact of water temperature on trout populations at various stages in the runoff conveyance and treatment system was ascertained by comparing water temperatures to 21° C, the temperature at which trout begin to experience thermal stress, using a signed rank test.

Comparisons of water temperatures at different locations in the BMPs were conducted using the Wilcoxon Rank Sum test (Wilcoxon, 1945). Data from the stage monitors in each BMP were used to discard temperature measurements within the water column when the probes were exposed to air. Although there was always discharge from the wet pond, comparisons between inflow and outflow temperatures were only conducted during periods of rainfall. Comparisons between the wet pond

and wetland were limited to data collected during 2006. Unless otherwise noted, analysis was conducted using monthly storm median and maximum temperatures.

Linear regression was used to determine the effect of storm duration and rainfall depth on effluent temperature. For the storm duration analysis, the difference in temperature from the storm mean served as the dependent variable with the percentage of storm duration being the independent variable. These normalized variables were selected to account for variations in temperature due to factors such as weather patterns, as well as make comparisons practical for storms of different durations.

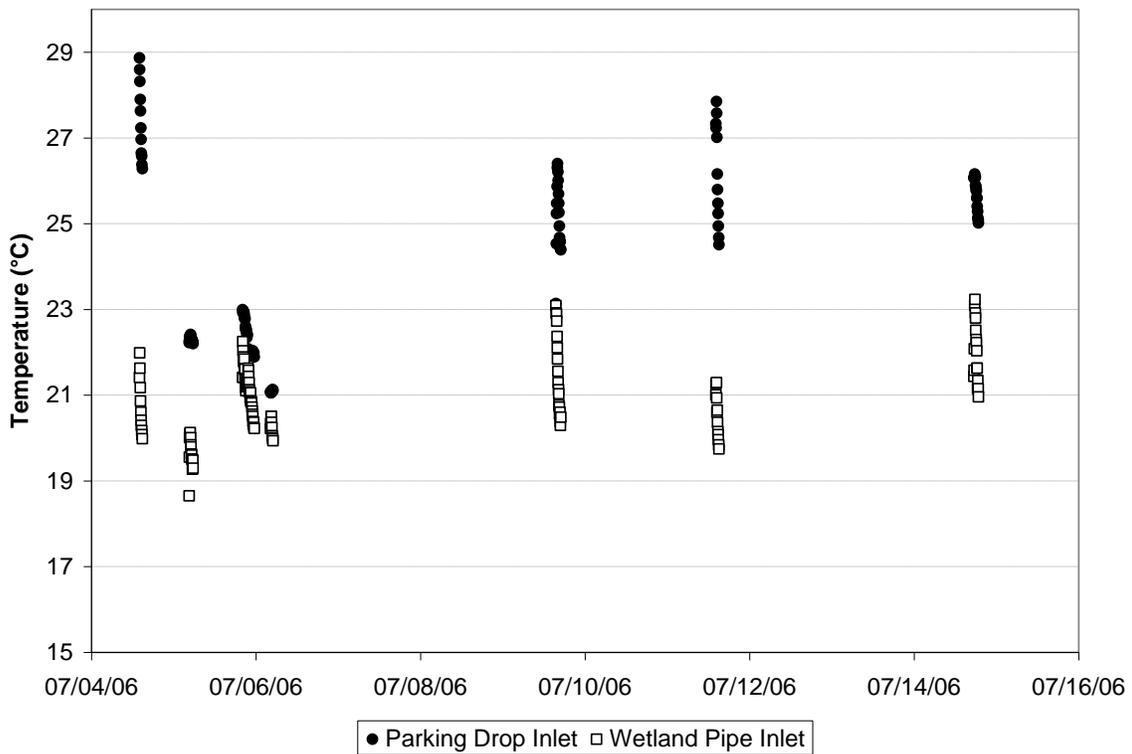
### **Results and Discussion**

The temperature of runoff leaving the parking surface at the Lenoir wet pond site was significantly\*\* warmer than 21° C for June through September (Table 3). The maximum runoff temperature of 32.7°C at the Lenoir site occurred on 07/14/2006 at 2:05 PM. At the Asheville stormwater wetland site, runoff temperatures leaving the parking surface were significantly\* warmer than 21° C for June through August (Table 4), with the maximum runoff temperature of 30.4° C occurring on 07/11/2006 at 1:05 PM. These elevated runoff temperatures indicate that there is the potential for runoff directly from these parking surfaces to impact trout populations, provided flowrates are substantial when compared to those of the receiving water body. In general, runoff temperatures were warmest near the beginning of a storm and cooled as rainfall progressed and heat stored in the asphalt diminished.

### **Effect of Conveyance in Buried Pipes**

Both median and maximum water temperatures were significantly\*\* cooler after traveling through the buried corrugated metal pipe from the parking lot drop inlet to the wetland inlet for the period from May through August. Cooling in the buried pipe generally persisted throughout the duration of a storm event. A

decrease in runoff temperatures as storms progressed was also observed at the wetland inlet, suggesting that the water may not have reached thermal equilibrium with the soil surrounding the pipe over the 168 m distance (Figure 3). At times, water was cooled by more than 7° C and runoff temperatures in excess of 21° C were cooled below the trout temperature threshold before water entered the wetland or wet pond. During the month of October, 2006, at the stormwater wetland, water temperatures were warmer after traveling through the pipe due to the same insulating soil properties that caused cooler temperatures during the summer months.



**Figure 3:** Temperature of runoff exiting parking lot and entering the wetland

Runoff temperatures were also significantly\*\* cooler after traveling from the monitored drop inlet at the Lenoir wet pond to the concrete pipe inlet for the period from May through October. Stormwater runoff entering the wet pond from the

concrete pipe was significantly\*\* warmer than runoff entering from the metal pipe at the same site for the months of June through September (Table 3). The differences in temperature may be attributed to the higher thermal conductivity of the metal pipe, but cannot be verified due to additional differences in watershed composition and pipe network configurations.

### **Wet Pond Results and Discussion**

The effluent temperature from the wet pond never dropped below 21° C for the months of June through August, with the maximum recorded effluent temperature of 29.2° C logged on 7/20/2006 (Table 3). Effluent temperatures exhibited substantial diurnal fluctuations corresponding to diurnal temperature fluctuations observed within the water column (Figure 4). For the entire monitoring period, there was no significant\* difference between the temperature of the wet pond effluent during periods of rainfall and direct runoff from the parking lot surface. Median effluent temperatures were significantly\*\* warmer than water entering the pond from the concrete and metal pipe inlets for the entire monitoring period, with temperature differences sometimes exceeding 10° C (Figure 4). Although significant\*\* differences in storm temperature maximums between the metal pipe inlet and pond outlet were evident for the entire monitoring period, there was no significant\* difference in storm maximums between the concrete pipe inlet and pond outlet. The difference in influent and effluent temperatures implies that the pond was a consistent source of thermal pollution.

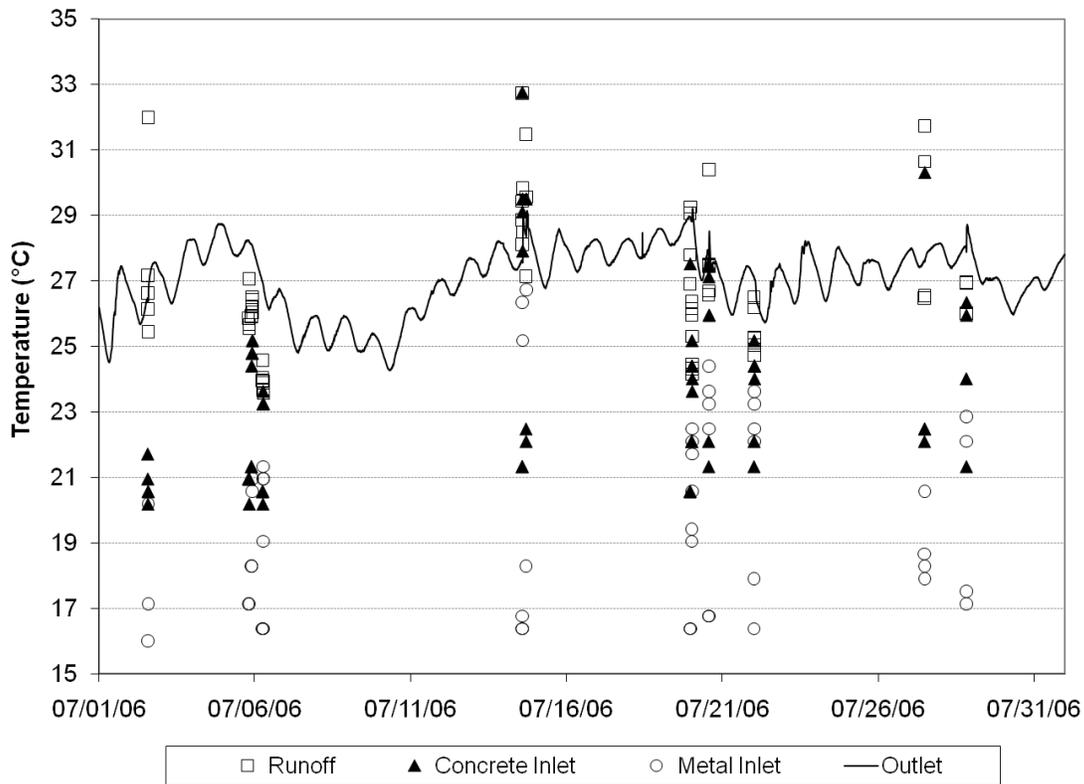
Reported effluent temperatures from the wet pond are likely cooler than what would be observed with a standard wet pond due to leaks in the outlet structure of the monitored system. These leaks caused the effluent water during a storm to be a mixture of warm surface water from the designed outlet orifices with cooler water from leaks deep within the water column. Also, effluent temperatures were recorded after water had entered the outlet structure and traveled through 35 m of buried

reinforced concrete pipe, which would likely decrease water temperatures below that of water entering the outlet structure.

**Table 3:** Lenoir wet pond temperature summary

<b>2006</b>	<b>May</b>	<b>June</b>	<b>July</b>	<b>Aug</b>	<b>Sept</b>	<b>Oct</b>
<b>Median Runoff Temperature (°C)</b>	18.03	24.70	26.55	25.72	22.61	19.94 <sup>†</sup>
<b>Median Metal Inlet Temperature (°C)</b>	15.23	18.66	18.76	21.71	18.28	15.90 <sup>†</sup>
<b>Median Concrete Inlet Temperature (°C)</b>	16.38	22.19	23.05	24.61	21.52	17.43 <sup>†</sup>
<b>Median Effluent Temperature (°C)</b>	19.39	24.94	27.74	26.18	23.26	18.45 <sup>†</sup>
<b>Median Temperature at 120 cm Depth (°C)</b>	17.90 <sup>†</sup>	23.24 <sup>†</sup>	25.56	25.56	21.71	17.90 <sup>†</sup>
<b>Median Temperature at 80 cm Depth (°C)</b>	18.28 <sup>†</sup>	23.63 <sup>†</sup>	25.95	25.95	21.33	17.14 <sup>†</sup>
<b>Median Temperature at 40 cm Depth (°C)</b>	20.18 <sup>†</sup>	22.86 <sup>†</sup>	27.12	26.34 <sup>†</sup>	22.09	17.52 <sup>†</sup>
<b>Median Temperature at Normal Pool Elev. (°C)</b>	--	24.40 <sup>†</sup>	27.91 <sup>†</sup>	24.01 <sup>†</sup>	22.0 <sup>†</sup>	16.38 <sup>†</sup>

<sup>†</sup> Dataset not complete for entire month due to equipment malfunction or removal

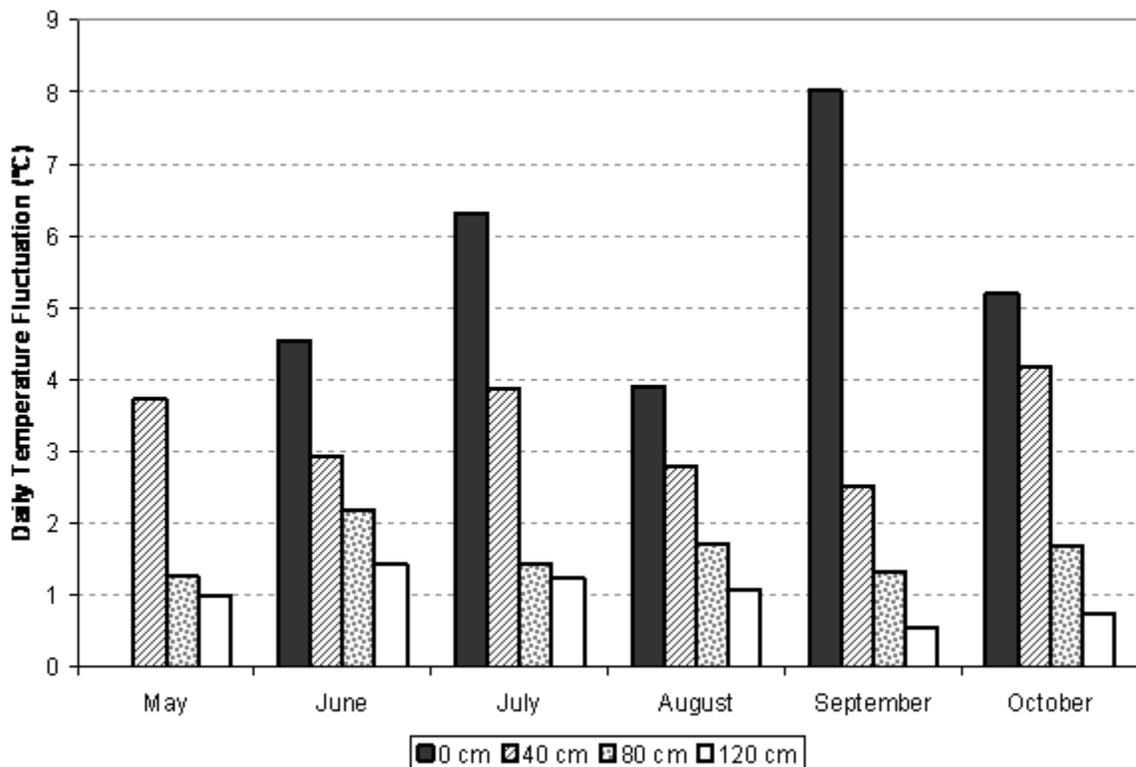


**Figure 4:** Influent and effluent temperatures at the wet pond during July

Effluent temperatures from the wet pond decreased significantly\* over time through the duration of storms in June, August, September, and October. The decrease in effluent temperatures is likely attributed to the effects of mixing within the wet pond, cooled influent from pipe conveyance, and cooler influent temperatures as storms progressed. The magnitude of the effluent temperature change during a storm was significantly\*\* correlated with the rainfall depth, which was expected, since larger storms correspond to more mixing within the pond and lower minimum runoff temperatures.

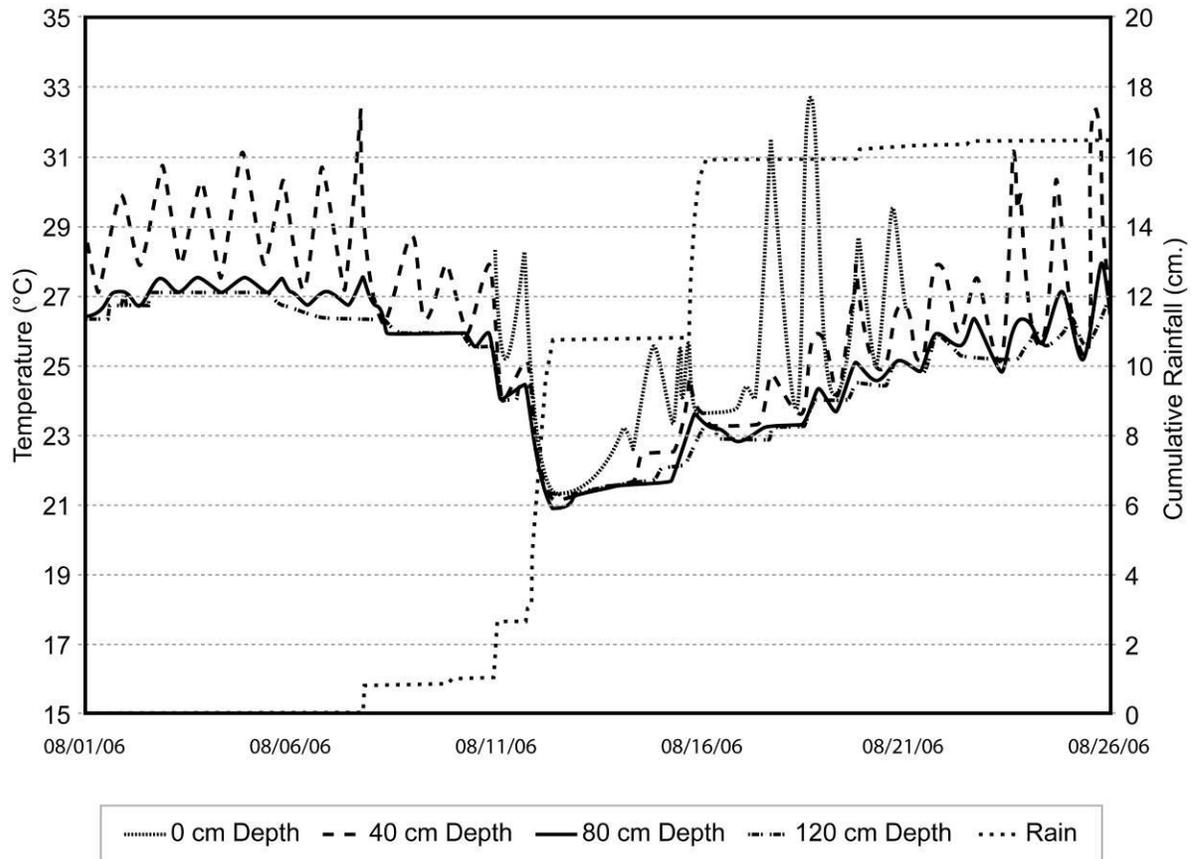
Due to thermal stratification within a pond, one potential mechanism to reduce effluent temperatures is to modify the outlet structure to draw water from the bottom of the water column. Temperatures at all depths (0 cm, 40 cm, 80 cm, and 120 cm) were significantly\*\* different from each other for the entire monitoring period, with the

warmest temperatures near the surface. Differences in water temperature were greatest in the late evening, due to the cumulative effect of heating from warmer air temperatures and solar radiation throughout the day. Diurnal fluctuations in water temperature were greatest at the normal pool elevation (Figure 5), which is the static water level between storms, but there were no significant\* differences in temperature fluctuations between depths of 80 and 120 cm observed for the months of May through August. Diurnal temperature fluctuations were significant\* at all depths for the entire monitoring period. The large fluctuations in water temperature near the surface are indicative of the unpredictable nature of effluent temperatures early in a storm. Mean diurnal temperature fluctuations at the normal pool elevation were lower for the month of August because most measurements were recorded subsequent to a large storm event (81 mm) that mixed water within the pond and raised the water level substantially higher than the normal pool elevation.



**Figure 5:** Mean diurnal temperature range at specified depths in the wet pond

During storm events, such as the storm on August 11, 2006, water temperatures at all depths generally cooled and approached the temperature of the deepest water in the pond (Figure 6). Water temperatures within the pond sometimes required several days to return to antecedent conditions. For example, the water in the wet pond did not approach temperatures antecedent to the August 11, 2006, storm until August 21, 2006 (Figure 6). It is possible that these relatively long recovery periods are partially attributed to the insulating effects of elevated water levels on the fixed temperature probes. Because cooling was primarily associated with the reduction in runoff temperatures as storms progressed, the pond did little to reduce temperatures itself, but was able to convey the benefits of cooler influent.



**Figure 6:** Temperature distribution within the wet pond water column near the outlet

The temperature of water leaving the current outlet was significantly\*\* warmer than the water temperature at a depth of 120 cm for June through October (with a maximum difference in monthly medians of 2.18° C), suggesting that a modified outlet would be beneficial for thermal pollution mitigation. At the same time, the water temperature at a depth of 120 cm was still significantly\*\* warmer than 21° C for the months of June through September (with a maximum difference in monthly medians of 4.56° C), indicating that even under favorable conditions where the coolest water at the base of the wet pond is discharged, the effluent still had the potential to impact trout populations. Furthermore, the water temperature at a depth of 120 cm was significantly\* warmer than the metal pipe influent for the entire monitoring period. Based on these results, the wet pond is expected to increase runoff temperature regardless of outlet structure configuration.

### **Stormwater Wetland Results and Discussion**

Median effluent temperatures from the stormwater wetland were significantly\*\* warmer than 21°C for the months of June through September. Additionally, median effluent temperatures were significantly\*\* warmer than piped inflow temperatures, and maximum effluent temperatures were significantly warmer than runoff directly leaving the parking surface for the period from June through September (Table 4), suggesting that the wetland was a source of thermal pollution. It is expected that flowrate reduction through attenuation in the stormwater wetland reduced the thermal loading to Reed Creek; however, complications with the flow monitoring equipment made it difficult to quantify exact reductions. A number of storms during the monitoring period were captured entirely by the wetland without generating outflow, inherently mitigating the thermal load to Reed Creek.

**Table 4:** Asheville stormwater wetland temperature summary for 2005 and 2006

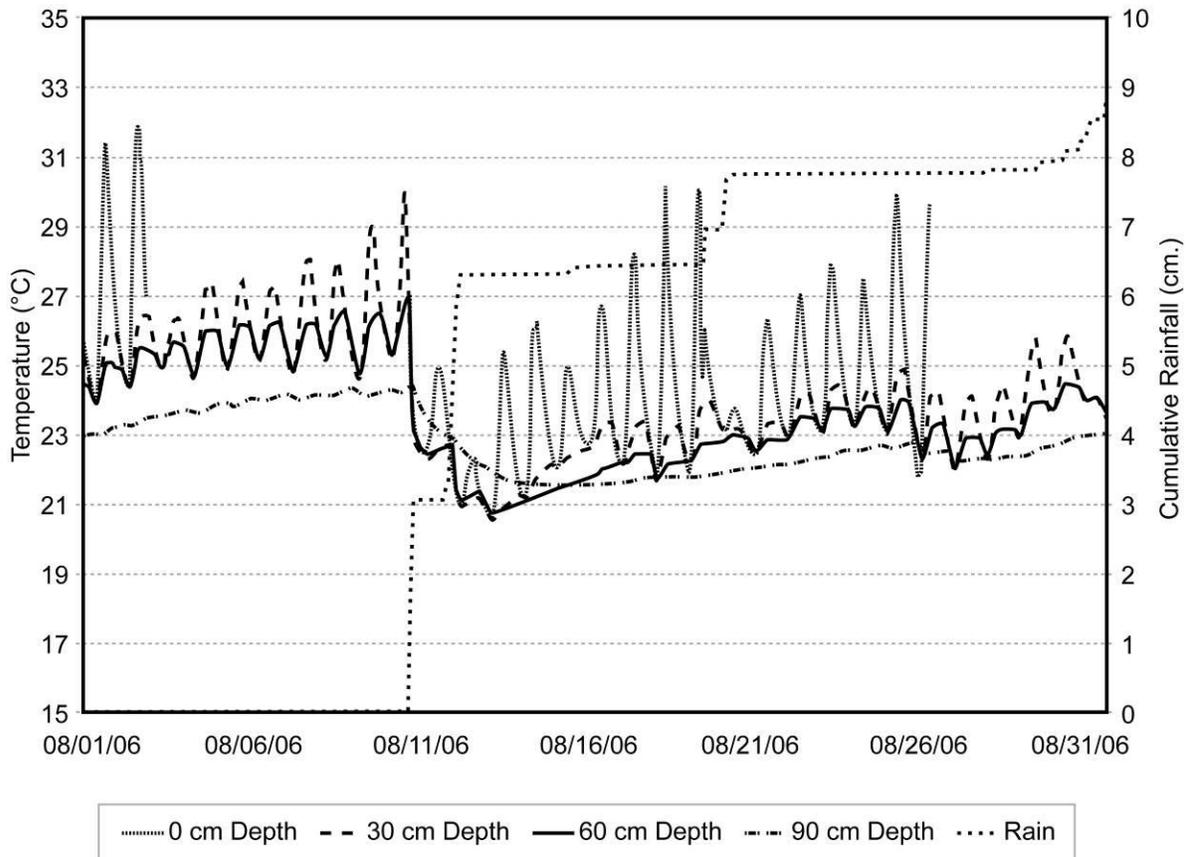
<b>2005</b>	<b>May</b>	<b>June</b>	<b>July</b>	<b>Aug</b>	<b>Sept</b>	<b>Oct</b>
<b>Median Pipe Temperature (°C)</b>	--	--	--	22.20	20.39	20.29 <sup>†</sup>
<b>Median Channel Temperature (°C)</b>	--	--	--	23.02	--	--
<b>Median Effluent Temperature (°C)</b>	--	--	--	23.06	--	--
<b>Median Temperature at 90 cm Depth (°C)</b>	--	--	23.74	22.75 <sup>†</sup>	20.96	19.46 <sup>†</sup>
<b>Median Temperature at 60 cm Depth (°C)</b>	--	--	24.17	23.16 <sup>†</sup>	21.15	19.37 <sup>†</sup>
<b>Median Temperature at 30 cm Depth (°C)</b>	--	--	24.24	23.86 <sup>†</sup>	21.68 <sup>†</sup>	19.89 <sup>†</sup>
<b>Median Temperature at Normal Pool Elev. (°C)</b>	--	--	24.17 <sup>†</sup>	24.73 <sup>†</sup>	--	19.67 <sup>†</sup>
<b>2006</b>	<b>May</b>	<b>June</b>	<b>July</b>	<b>Aug</b>	<b>Sept</b>	<b>Oct</b>
<b>Median Runoff Temperature (°C)</b>	18.85	24.67	25.00	23.34	20.40	16.25 <sup>†</sup>
<b>Median Channel Temperature (°C)</b>	18.15	20.25	22.60	21.29	--	--
<b>Median Pipe Temperature (°C)</b>	17.53	20.63	21.19	22.83	21.12	16.57 <sup>†</sup>
<b>Median Effluent Temperature (°C)</b>	18.34	22.14	23.76	23.28	21.30	15.47 <sup>†</sup>
<b>Median Temperature at 90 cm Depth (°C)</b>	15.37	20.41 <sup>†</sup>	22.39 <sup>†</sup>	22.56	19.75	16.23 <sup>†</sup>
<b>Median Temperature at 60 cm Depth (°C)</b>	16.30	21.53	23.52	23.33	19.32	13.64 <sup>†</sup>
<b>Median Temperature at 30 cm Depth (°C)</b>	17.56	22.23	23.88	23.83	19.56	13.93 <sup>†</sup>
<b>Median Temperature at Normal Pool Elev. (°C)</b>	19.27	22.92 <sup>†</sup>	24.41 <sup>†</sup>	23.88 <sup>†</sup>	20.70 <sup>†</sup>	13.62 <sup>†</sup>

† Dataset not complete for entire month due to equipment malfunction or removal

Water temperatures at the bottom of the stormwater wetland were the coolest and also exhibited the smallest diurnal fluctuations (Figure 7). Water at the bottom of the wetland was significantly\*\* cooler than 21°C for the months of May, June, September, and October, 2006, and with the exception of several storms in August, water temperatures during storms at the 90 cm depth were consistently cooler than the current effluent. These low temperatures are likely attributed to not

only the distance from the water surface and effects of differences in water density, but also the proximity to the interface between the water within the wetland and the saturated soil below. There was no significant\* difference in temperature between the piped influent and water at a depth of 90 cm for every month in the monitoring period. These results suggest that a modified outlet structure may discharge temperatures suitable for trout; however, because inflow temperatures were already cool, it is unlikely that substantial temperature reductions will result from wetland treatment.

For the entire monitoring period, temperatures at all depths exhibited significant\* diurnal fluctuations. Although daily divergence existed, there was no significant\*\* difference in median water temperatures between depths of 30 cm and 60 cm during the months of September and October, 2006. In response to a substantial storm event, such as the storm on August 10, 2006, water temperatures within the wetland decreased for reasons similar to the wet pond (Figure 7). Following storms on August 10 and 11, 2006, water temperatures did not return to their antecedent conditions, which is likely attributed to the prevalence of cooler weather conditions.



**Figure 7:** Temperature distribution within the wetland water column near the outlet

### Comparison of Wetland and Wet Pond Results and Discussion

Effluent temperatures from the wet pond were significantly\* warmer than those from the stormwater wetland for the months from June through September. There were no significant\* differences in the temperature of water entering the wetland and wet pond during the entire monitoring period, suggesting that higher effluent temperatures from the wet pond cannot be attributed to higher inflow temperatures. Additionally, mean water temperatures just 30 cm below the normal pool elevation of the stormwater wetland were significantly\*\* cooler than mean water temperatures measured at the bottom of the wet pond during the months of June through October. Diurnal temperature fluctuations within the water column were also significantly\* greater in the wet pond than the stormwater wetland for the months of July through October. Differences in water temperatures between the

wetland and wet pond are likely attributed to the presence of vegetation in the wetland and its associated cooling through shading and evapotranspiration. The wetland and wet pond sites were separated by approximately 100 km, which may account for some of the differences observed.

### **Alternative Outlet Structure Designs**

Although currently used for reservoirs and other large water bodies, an outlet structure that draws from the deepest point in the water column has not been previously recommended for wetlands and wet ponds. With a modified outlet structure, effluent temperatures significantly\*\* cooler than the 21°C temperature threshold for trout appear to be attainable for stormwater wetlands, but unlikely for wet ponds in borderline trout regions. Implementation of a modified outlet structure could consist of a section of perforated plastic tubing along the bottom of the wetland or pond surrounded by a gravel envelope and connected by non-perforated tubing to the outlet structure at the normal pool elevation (Figure 8). However, there are several potential concerns involved with the implementation of this type of outlet. First, maintenance is critical to ensure that the pipe does not clog with sediment or gross solids, preventing proper outflow from the system. Another concern is the potential for pollutant concentrations to be higher at the bottom of these treatment systems than they are at the surface. Dissolved oxygen stratification has been observed in wetland systems (Chimney, 2006) and one would expect higher suspended solids concentrations near the bottom of the water column; however, additional research is needed to quantify specific differences. If a standard outlet configuration is used, effluent flowrates should be limited during the early periods of a storm, since monitoring results presented herein indicated that effluent temperatures decrease with time.



**Figure 8:** Illustration of modified outlet structure drawing from bottom waters

### **Summary and Conclusions**

With a standard outlet configuration, neither the stormwater wetland nor wet pond were capable of consistently reducing runoff temperatures and often served as sources of thermal pollution. This finding for a stormwater wetland differs from that reported by Kieser et al. (2004). The reason for this discrepancy may be because the wetland examined by Kieser treated the warm effluent from a wet pond. Due to the large fluctuations in water temperature near the surface, outlet temperatures were not only elevated, but subject to large fluctuations, making the estimation of effluent impact difficult. Because water is discharged from the top of the water column with most conventional outlet structures, monitoring results suggest that effluent temperatures could exceed 35°C and vary by more than 8°C depending upon time of day and weather conditions. These large fluctuations make it difficult to evaluate the role of a stormwater wetland or wet pond in a temperature TMDL (Total Maximum Daily Load) program. Continuous measurement of temperatures within the wetland and wet pond support Van Buren's (2000) discrete measurements that find reduced thermal stratification in response to high inflows. Unlike Van Buren's results, thermal stratification did not appear to be affected by moderate winds, which could be attributed to the larger pond area and suspected higher base flows in Van

Buren's study. The wetland and wet pond examined herein were appreciably smaller, but are typical for small to medium sized catchments.

The most probable reason for differences between water temperatures of the two systems pertains to the amount of vegetative shading within the BMPs. Any shading of water within the Lenoir wet pond was provided by algae covering the pond surface. While the algae shielded deeper water within the system from radiation, much of this radiation was captured by the algae itself and the water near the surface was consequently heated through conduction, reducing any benefits of shading. This phenomenon is similar to the one observed by Dale (1976) in a pond covered by *Lemnaceae* (duckweed). In addition to cooling through transpiration, the broad leaf plants covering the stormwater wetland likely intercepted and reflected substantial amounts of solar radiation above the water surface, insulating the water from this heat. The cooling aspects associated with the presence of vegetation provide a benefit to the stormwater wetland that the wet pond lacks, indicating a stormwater wetland may inherently be better suited for regions of cold water fisheries. Although the composition of the watersheds was similar, the stormwater wetland and wet pond monitoring sites were located approximately 100 km apart, which could account for some of the temperature differences observed.

While complications with the flow monitoring equipment made it difficult to quantify the exact flowrate reduction between the inlet and outlet, flowrate reduction is a key design component of stormwater wetlands and has been verified in numerous research efforts. Because thermal energy is dependent upon temperature and flow, the stormwater wetland should inherently reduce the thermal impact of stormwater runoff as long as temperatures are not increased as water travels through the wetland. While flowrate reductions would also benefit wet ponds, evidence of water temperature increases may outweigh the benefit of reduced flowrates.

Unlike some BMPs, such as bioretention areas, there is often a substantial distance between the parking surface and a wetland or wet pond, due to the relatively large size and specific topographic requirements of these BMPs. Because runoff is frequently piped underground to overcome these distances, reduced inflow temperatures appear to be attainable for many wetland and wet pond installations. Since increased water temperature can potentially enhance pollutant removal mechanisms, such as metal uptake by plants, it is important to identify any potential concerns associated with cooling water before it reaches a wetland or wet pond. Although there is evidence that cooler influent to a wetland or wet pond will result in cooler effluent, the benefit of cooling runoff before it enters these BMPs is substantially reduced when the water within these systems is warmer than the original runoff. Consequently, conveying water through buried pipes should be incorporated after treatment in a wetland or wet pond for substantial temperature reductions to be realized.

Because North Carolina trout waters are located along the southeastern extent of trout populations, it is important to minimize the thermal impacts associated with urbanization and stormwater treatment, since small changes in temperature can have substantial impacts on the aquatic ecosystem. With proper BMP design, implementation of modified outlet structures, emergent vegetation, and conveyance in buried pipes when practical, it should be possible for these systems to achieve sediment, nutrient, and metal removal goals, while minimizing, but not eliminating, thermal impacts to trout waters.

### **Acknowledgements**

This research was funded by the North Carolina Department of Environment and Natural Resources (NCDENR), Division of Water Quality. The authors would like to thank Jonathan Smith, Dan Willits, Garry Grabow, Aziz Amoozegar, Jon Calabria, Allen Caldwell, Eric Caldwell, Seth Nagy, and Jason Zink, all of whom are

currently or formerly of NC State University, for their assistance in selecting research sites and support throughout the project.

## **References**

- Agersborg, H. P. (1930). "The Influence of Temperature on Fish." *Ecology*, 11 136-144.
- Akbari, H., and Konopacki, S. (2005). "Calculating Energy-Saving Potentials of Heat-Island Reduction Strategies." *Energy Policy*, 33(6), 721-756.
- Asaeda, T., Ca, V. T., and Wake, A. (1996). "Heat Storage of Pavement and its Effect on the Lower Atmosphere." *Atmospheric Environment*, 30(3), 413-427.
- Baxter, D. O. (1994). "Energy exchanges and related temperatures of an earth-tube heat exchanger in the cooling mode." *Transactions of the ASAE*, 37(1), 257.
- Chimney, M. J. (2006). "Patterns of vertical stratification in a subtropical constructed wetland in south Florida (USA)." *Ecological Engineering*, 27(4), 322.
- Coutant, C. C. (1977). "Compilation of Temperature Preference Data." *Journal of the Fisheries Research Board of Canada*, 34 740-745.
- Coutant, C. C. (1962). "The Effect of a Heated Water Effluent Upon the Macroinvertebrate Riffle Fauna of the Delaware River." *Proceedings of the Pennsylvania Academy of Science*, 36 58-71.
- Dale, H. M., and Gillespie, T. (1976). "The Influence of Floating Vascular Plants on the Diurnal Fluctuations of Temperature Near the Water Surface in Early Spring." *Hydrobiologia*, 49(3), 245-256.
- De Staso III, James, and Rahel, F. J. (1994). "Influence of Water Temperature on Interactions Between Juvenile Colorado River Cutthroat Trout and Brook Trout in a Laboratory System." *Transactions of the American Fisheries Society*, 123(3), 289-297.
- Hocutt, C. H., Stauffer, Jay R., Jr., Edinger, J. E., Hall, Lenwood W., Jr., and Morgan, Raymond P., II. (1980). *Power Plants: Effects on Fish and Shellfish Behavior*.
- Huff, D. D., Hubler, S. L., and Borisenko, A. N. (2005). "Using Field Data to Estimate the Realized Thermal Niche of Aquatic Vertebrates." *North American Journal of Fisheries Management*, 25(1), 346-360.

- Kieser, M. S., Spoelstra, J. A., Feng Feng, A., James, W., and Li, Y. (2004). *Stormwater Thermal Enrichment in Urban Watersheds*.
- Langford, T. E. (1990). *Ecological Effects of Thermal Discharges*. Elsevier, New York.
- Lieb, D. A., and Carline, R. F. (2000). "Effects of Urban Runoff from a Detention Pond on Water Quality, Temperature and Caged *Gammarus minus* (Say) (Amphipoda) in a Headwater Stream." *Hydrobiologia*, 441(1), 107-116.
- Morrill, J. C., Bales, R. C., and Conklin, M. H. (2005). "Estimating Stream Temperature from Air Temperature: Implications for Future Water Quality." *Journal of Environmental Engineering*, 131(1), 139-146.
- Nebeker, A. V. (1972). "Effect of Low Oxygen Concentration on Survival and Emergence of Aquatic Insects." *Transaction of the American Fisheries Society*, (4), 675-679.
- North Carolina Climate Office. 2007. Website: <http://www.nc-climate.ncsu.edu/cronos>. Accessed 05 Oct 07.
- U.S. Department of the Interior, Fish and Wildlife Service and U.S. Department of Commerce, U.S. Census Bureau. (2001). *National Survey of Fishing, Hunting, and Wildlife-Associated Recreation*.
- U.S. Environmental Protection Agency. (2003). "EPA Region 10 Guidance for Pacific Northwest State and Tribal Temperature Water Quality Standards."
- Van Buren, M. A., Watt, W. E., Marsalek, J., and Anderson, B. C. (2000). "Thermal Balance of an On-Stream Storm-Water Management Pond." *Journal of Environmental Engineering*, 126(6), 509-517.
- Wilcoxon, F. (1945). "Individual Comparisons by Ranking Methods." *Biometrics*, 1(6), 80-83.
- Wu, J. S., Holman, R. E., and Dorney, J. R. (1996). "Systematic Evaluation of Pollutant Removal by Urban Wet Detention Ponds." *Journal of Environmental Engineering*, 122(11), 983-988.

### **Nomenclature**

\* Denotes significance at 5% level

\*\* Denotes significance at 1% level

## **Chapter 3**

# **Bioretention Impact on Runoff Temperature in Trout Sensitive Waters**

Submitted to:

*Journal of Environmental Engineering*  
"In Press" (With Permission from ASCE)

# Bioretention Impact on Runoff Temperature in Trout Sensitive Waters

Matthew P. Jones, S.M.ASCE<sup>1</sup> and William F. Hunt, M.ASCE, P.E.<sup>2</sup>

## **Abstract**

A study was conducted in western North Carolina, along the southeastern extent of United States trout populations, to examine the effect of bioretention areas on runoff temperature. Four bioretention areas were monitored during the summers of 2006 and 2007. It was found that smaller bioretention areas, with respect to the size of their contributing watershed, were able to significantly reduce both maximum and median water temperatures between the inlet and outlet. The proportionately larger bioretention areas were only able to significantly reduce maximum water temperatures between the inlet and outlet; however, these systems showed evidence of substantial reductions in outflow quantity, effectively reducing the thermal impact. Despite temperature reductions, effluent temperatures still posed a potential threat to coldwater streams during the peak summer months. During the summer months, effluent temperatures were generally coolest at the greatest soil depths, supporting evidence of an optimum drain depth between 90 and 120 cm. The ability of bioretention areas to reduce stormwater temperature and flows supports their application to reduce the thermal impacts of urban stormwater runoff.

## **Subject Headings:**

Stormwater Management; Temperature; Aquatic Habitats; Ecology; Infiltration; Urban Development

## **Introduction**

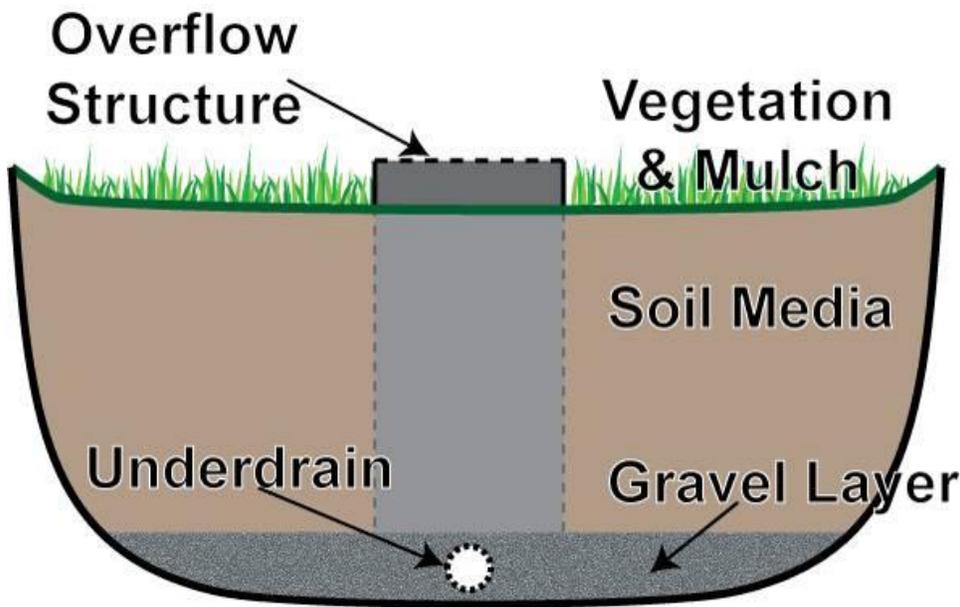
Water temperature is an important habitat constraint in aquatic environments, directly impacting the metabolism, behavior, enzyme function, and reproduction of many aquatic organisms. Although water temperatures exhibit natural daily and seasonal temperature fluctuations, it has been observed that heat from anthropogenic discharges can have a substantial impact on the aquatic ecosystem, reducing both the abundance and diversity of aquatic organisms (Hocutt et al. 1980). The thermal impact of industrial discharges has been reduced through the use of better heat exchange processes; however, there are few mechanisms available to mitigate non-point sources of thermal pollution.

One major non-point source of thermal pollution is urban stormwater runoff. Paved surfaces elevate runoff temperatures by capturing solar radiation and transferring this stored energy to runoff during rainfall events, which is especially a concern during the summer months. Asphalt typically has a low thermal conductivity and reflectivity, causing heat from solar radiation to concentrate near the surface, which can lead to asphalt surface temperatures in excess of 60°C (Asaeda et al. 1996). Because heat is concentrated near the surface, runoff temperatures typically exhibit a short term temperature spike, cooling as a storm progresses. Heated runoff from urban watersheds has been shown to increase the temperature of water bodies above their normal levels (Kieser et al. 2004).

Although temperature affects many aquatic organisms, trout and other salmonids are among the fish most sensitive to water temperature changes. They are important game fish in many parts of the country (U.S. Environmental Protection Agency 2003). Due to a variety of complex factors, it is difficult to predict actual fish behavior in response to elevated temperatures, which is evidenced by inconsistencies between laboratory and field research data (Hocutt et al. 1980). Despite these complexities, trout and salmon have been observed to generally avoid

water temperatures in excess of 21°C (Coutant 1977). In many states where trout and salmon reside, temperature is listed as a pollutant of concern within lists of impaired waters required by section 303(d) of the Clean Water Act; however, the high variability of natural water temperatures has made implementation of total maximum daily load (TMDL) programs for temperature control difficult. Although it is likely that many North Carolina streams have been negatively impacted by elevated temperatures from anthropogenic sources, criteria to assess thermal impacts on coldwater stream environments have not been incorporated into the North Carolina Index of Biotic Integrity (North Carolina Department of Environment and Natural Resources 2007).

One urban stormwater best management practice (BMP) that has gained popularity due to its ability to simultaneously satisfy stormwater and landscaping requirements is bioretention. In North Carolina, a bioretention area typically consists of an underdrain system surrounded by a gravel envelope and overlain by 0.7 to 1.2 m of fill soil media, all of which is contained in an excavated basin (Fig. 1). In locations where the hydraulic conductivity of the underlying soil is substantial, underdrains may not be required. Generally, the fill soil media is predominantly sand, with a small percentage of fine particles and organic matter. These systems are typically mulched and vegetated and often located immediately adjacent to their contributing watershed. During moderate inflows, runoff infiltrates into the soil media and leaves the system either through the underdrains, seepage into the underlying soil, or evapotranspiration. These systems are generally designed to bypass additional water after collecting runoff from approximately 25 mm of rainfall. In North Carolina, bioretention areas are commonly designed to occupy an area between 5 and 7 percent of the contributing watershed.



**Fig. 1:** Cross-section of a typical bioretention area

The effect of bioretention areas on stormwater runoff temperature has not been thoroughly examined. Because soil temperature is largely regulated by radiation and convective heat exchanges at the surface and heat conduction within a soil column is relatively slow, deeper layers of soil are able to maintain a relatively stable temperature over long periods of time (Mohseni et al. 2002). At a shallow (60 cm) bioretention area in the northeastern United States, seasonal mean water temperatures were not significantly different between rooftop runoff and underdrain outflow (Dietz and Clausen 2005). In a study of soil temperature profiles during irrigation with warm and cool water, it was noted that water temperature approaches that of the soil as it infiltrates, causing little variation in soil temperature at depths greater than 100 cm (Wierenga et al. 1970). The researchers also noted that irrigation with both warm and cool water led to soil temperatures lower than a non-irrigated plot due to the cooling associated with evaporation and higher heat capacity of the saturated soil. Bioretention areas also facilitate infiltration into the shallow groundwater, which serves as a major water source for many coldwater streams and helps maintain water temperatures below that of the ambient air during the warmer

portions of the year. Research has shown that the effluent volume from bioretention areas can be less than 50% of the influent volume, with the greatest reductions evident during the warmer times of the year (Hunt et al. 2006).

There has been some previous research into the effect of stormwater wetlands and wet ponds on runoff temperature. A study in western North Carolina showed that wetlands and wet ponds increased the temperature of the water they received, with large temperature fluctuations near the surface making it difficult to predict effluent temperatures (Jones and Hunt, unpublished monitoring results, 2008). Kieser et al. (2004) demonstrated that a well-shaded stormwater wetland could reduce the thermal load of urban stormwater runoff; however, cooling was limited to the temperature of the ambient air. Because stormwater wetlands and wet ponds do not generally decrease runoff volumes, substantial temperature reductions are required to mitigate the impact of thermal pollution from urban stormwater runoff.

Although the impact of urban stormwater runoff on stream temperatures has been shown, there are few mechanisms available to limit these thermal impacts. Stormwater BMPs, such as bioretention areas, are being installed throughout the country to satisfy stormwater regulations and present an opportunity to limit thermal pollution from urban watersheds. By examining bioretention areas in western North Carolina, which constitutes the southeastern extent of United States trout populations, it should be possible to evaluate the effect of bioretention areas on runoff temperature and identify any design criteria that can be modified to better mitigate thermal pollution.

### **Materials and Methods**

In order to investigate the effect of bioretention on runoff temperature, a monitoring study was conducted at four bioretention areas in western North Carolina during the summers of 2006 and 2007. Water temperatures were measured at all inlets and underdrain outlets. Additionally, temperatures were measured at evenly

spaced depths within the soil column at some bioretention areas. Pulley-float stage recorders were used in conjunction with v-notch weirs to measure outflow at all sites. Inflow measurements were obtained using pulley-float stage recorders with v-notch weirs or estimated using rainfall data (Table 1). Due to complications with the flow monitoring equipment, flow data were used primarily to identify periods of flow over the temperature sensors and not to measure specific flowrates. Measurements from all temperature and flow monitoring equipment were logged at 5-minute intervals. Vegetative cover of each bioretention area was estimated by creating a composite overhead image of the site and digitally comparing areas of exposed mulch or soil to vegetated areas.

**Table 1:** Bioretention flow monitoring strategy for each monitoring location

<b>Bioretention Location</b>	<b>Inflow Monitoring</b>	<b>Outflow Monitoring</b>
<b>Asheville</b>	Rainfall	Weir Box
<b>Lenoir</b>	Rainfall	Weir Box
<b>Brevard East</b>	Weir Box	Weir Box
<b>Brevard West</b>	Weir Box	Weir Box

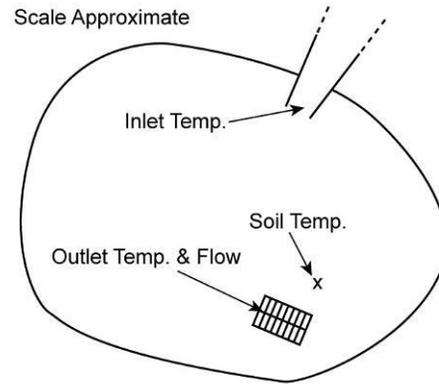
Rainfall data were collected at each site using tipping bucket rain gauges with a resolution of 0.25 mm (0.01”). Temperature measurements were collected with a combination of HOBO™ Water Temp Pro (H20-001) and HOBO™ 4 channel loggers (H08-008-04 & U12-008) with temperature sensors attached (TMCX-HD). All temperature loggers were manufactured by Onset Computer Corporation.

Statistical analysis was conducted using SAS® software, Version 9 (SAS Institute Inc., Cary, NC). The potential thermal impact to trout populations was ascertained by comparing water temperatures to 21° C, the temperature at which trout begin to experience thermal stress, using a signed rank test. Comparisons of influent, effluent, and soil temperatures were conducted using the Wilcoxon Rank Sum test (Wilcoxon, 1945). Individual storm medians and maximums were used in the analysis of BMP influent and effluent temperatures and only storms that

generated measurable outflow were used in the comparison of influent and effluent temperatures. During the course of the monitoring study, runoff from many relatively small storms was able to infiltrate into the bioretention areas and underlying soils without generating measurable outflow. Linear regression was used to examine the correlation between the time of day and influent or effluent temperatures. In order for a linear relationship to exist, the timescale was centered about noon, where the dependent variable used in the regression analysis was the length of time from noon to the beginning of the storm event each day. Linear regression was also used to examine the correlation between influent and effluent temperatures. Statistical significance was established within a 99% confidence interval ( $p < 0.01$ ), unless otherwise noted.

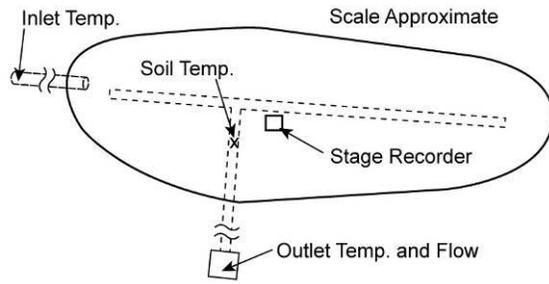
### **Site Description**

The 45 m<sup>2</sup> Asheville bioretention area was located on the campus of the University of North Carolina at Asheville (35°36'46"N, 82°33'54"W) and received runoff from 280 m<sup>2</sup> of asphalt parking lot (Fig. 2). The bioretention area and contributing parking lot were constructed during the summer of 2005, at which time a light colored chip seal was applied to the parking surface in an attempt to reduce pavement temperatures. Runoff was routed by a speed bump into a 5.75 m long asphalt channel which led directly into the bioretention area. The bioretention area was drained by 10 cm perforated PVC pipes and outflow was discharged directly through a 15 m length of 38 cm smooth-walled corrugated plastic tubing (CPT) into Reed Creek. The bioretention area was not mulched and naturally progressed from no vegetative cover in June of 2005 to an estimated 55% vegetative cover in late August 2006 (Table 2). Temperature probes were installed at the inlet channel, inside the outlet weir box, and at 5 evenly spaced depths within the soil column from the surface to the underdrain depth.



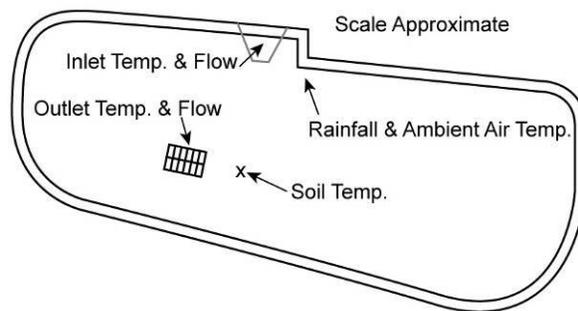
**Fig. 2:** Photo of Asheville bioretention and diagram of equipment layout

The 30 m<sup>2</sup> Lenoir bioretention area (35°55'20"N, 81°31'24"W) received runoff from a 674 m<sup>2</sup> asphalt parking lot and a 95 m<sup>2</sup> area of concrete sidewalk, comprised of two separate sections (Fig. 3). The bioretention area and adjacent parking surface were shaded by a mature tree canopy. The surface of the bioretention area was covered by hardwood mulch and vegetation within the system provided shading for 79% of the surface. Runoff entered the system through a 4.9 m length of buried 22 cm PVC pipe. Effluent was collected by a network of 10 cm perforated CPT underdrains and discharged onto the adjacent street curb, located approximately 10 m away, where it entered the municipal stormwater drainage network. Temperature probes were installed in the inlet pipe, outlet weir box, and at 4 evenly spaced depths within the soil column from the surface to the underdrain depth. Soil temperatures were only monitored at this site during the summer of 2007.



**Fig. 3:** Photo of Lenoir bioretention and diagram of equipment layout

Adjacent bioretention areas were monitored in Brevard, NC ( $35^{\circ}14'20''N$ ,  $82^{\circ}43'52''W$ ). Both systems were mulched with pine needles. Water from these bioretention areas, as well as others in the parking lot, was routed by an underground drainage network to Kings Creek, which was adjacent to the property. The Brevard east bioretention area covered a  $37\text{ m}^2$  area and received runoff from approximately  $525\text{ m}^2$  of asphalt parking lot (Fig. 4). Water left the bioretention area through a 10 cm perforated CPT underdrain network located approximately 48 cm below the soil surface. An estimated 43% of this bioretention area was shaded by low-lying plants. Temperature probes were located in the inlet weir box and outlet pipe and soil temperature was measured at a depth of 48 cm during the summer of 2007.



**Fig. 4:** Photo of Brevard east bioretention and diagram of equipment layout

The Brevard west bioretention area covered a 36 m<sup>2</sup> area and received runoff from approximately 325 m<sup>2</sup> of asphalt parking lot (Fig. 5). Water left the bioretention area through a 10 cm perforated CPT underdrain network located approximately 43 cm below the soil surface. An estimated 43% of this bioretention area was shaded, primarily by two maple trees. Temperature probes were located within the inlet and outlet weir boxes.



**Fig. 5:** Photo of Brevard west bioretention and diagram of equipment layout

**Table 2:** Bioretention site characteristics

<b>Bioretention Location</b>	<b>Underdrain Depth</b>	<b>% Vegetative Cover</b>	<b>% of Watershed Area</b>
<b>Asheville</b>	135 cm	55%	16%
<b>Lenoir</b>	95 cm	79%	4%
<b>Brevard East</b>	48 cm	43%	7%
<b>Brevard West</b>	43 cm	43%	11%

### **Results and Discussion**

Both influent and effluent temperatures were warmest at the Brevard east bioretention site, while the coolest temperatures were observed at the Asheville location (Table 3). The maximum influent temperature reading of 39.2°C was made at the Brevard east bioretention area at 2:00 pm on 07/01/2006. The maximum effluent temperature reading of 30.3°C was also recorded at the Brevard east site at 2:25 pm on 06/24/2006. During 2007, rainfall depths at all sites were substantially lower than the 30-year normal depths (Table 4).

**Table 3:** Median summary statistics for the bioretention area at each location

	<b>Asheville</b>	<b>Lenoir</b>	<b>Brevard East</b>	<b>Brevard West</b>
<b>Median Influent (°C)</b>	20.6	26.0	27.9	23.3
<b>Median Effluent (°C)</b>	19.8	22.3	23.7	22.5
<b>Max. Influent (°C)</b>	23.2	26.9	30.3	27.1
<b>Max. Effluent (°C)</b>	20.2	23.0	24.9	23.8
<b>Inlet Variance (°C)</b>	1.9	0.9	0.9	1.0
<b>Outlet Variance (°C)</b>	0.2	0.2	0.2	0.3

**Table 4:** Observed and 30-year average (1971-2000) rainfall depths near bioretention locations

	<b>Asheville <sup>a</sup> (cm)</b>		<b>Lenoir <sup>b</sup> (cm)</b>		<b>Brevard <sup>c</sup> (cm)</b>	
	Observed	Average	Observed	Average	Observed	Average
<b>May 2006</b>	7.2	9.0	3.1	11.9	11.8	15.0
<b>June 2006</b>	10.2	8.2	9.2	11.3	10.4	14.6
<b>July 2006</b>	7.7	7.5	10.4	11.2	22.4	13.0
<b>August 2006</b>	9.2	8.5	12.5	9.8	21.3	13.7
<b>September 2006</b>	9.6	7.6	13.8	11.3	18.1	13.0
<b>October 2006</b>	6.1	6.1	10.5	9.2	16.5	12.3
<b>2006 Annual</b>	98.5	95.7	92.2	125.0	167.5	168.1
<b>May 2007</b>	1.9	9.0	1.5	11.9	4.3	15.0
<b>June 2007</b>	3.8	8.2	8.4	11.3	12.5	14.6
<b>July 2007</b>	9.3	7.5	9.6	11.2	8.2	13.0
<b>August 2007</b>	2.8	8.5	6.0	9.8	7.4	13.7
<b>September 2007</b>	6.8	7.6	7.4	11.3	8.4	13.0
<b>October 2007</b>	0.6	6.1	0.0	9.2	0.5	12.3
<b>2007 Annual</b>	53.3	95.7	78.7	125.0	92.03	168.1

a: National Weather Service Coop Station # 310301 (35°35'43"N, 82°33'24"W)

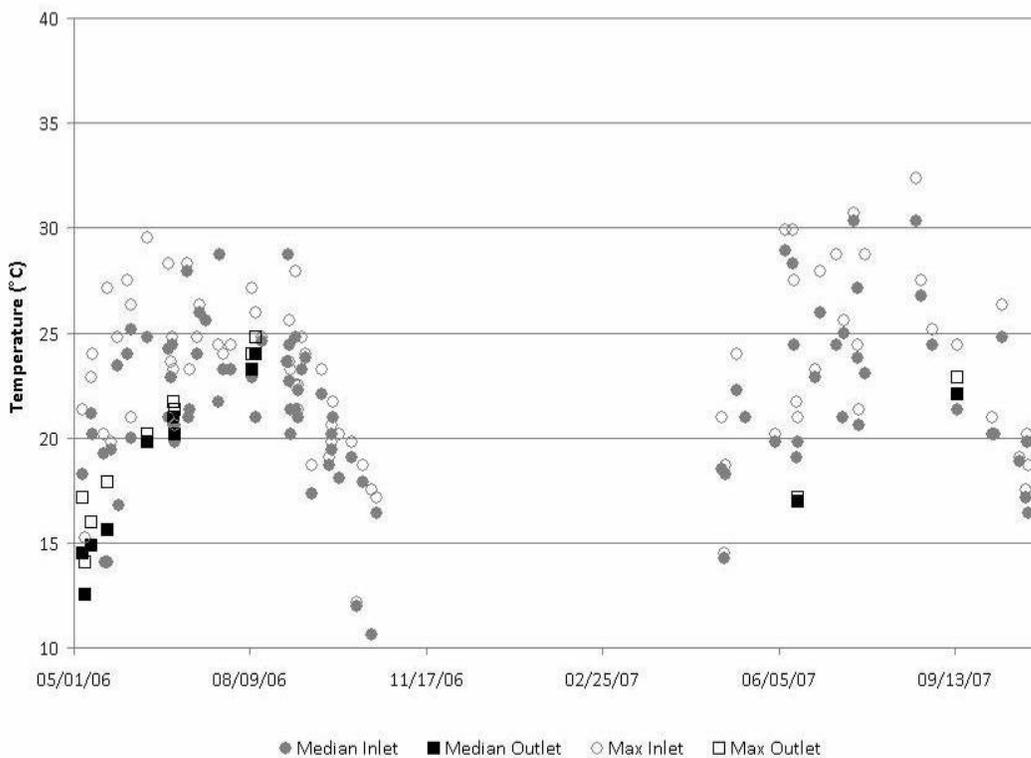
b: National Weather Service Coop Station # 314938 (35°54'42"N, 81°32'2"W)

c: National Weather Service Coop Station # 311055 (35°16'6"N, 82°42'11"W)

Source: NC Climate Office 2008

At the Asheville bioretention area, maximum runoff temperatures were significantly warmer than 21°C for the entire monitoring period, while median runoff temperatures were significantly warmer than 21°C only when the cooler months of May and October were excluded from analysis. Since 21°C is considered the upper

avoidance temperature for North Carolina trout species, there was evidence that direct runoff from this site would negatively impact the thermal environment of nearby Reed Creek. There was no significant ( $p>0.05$ ) difference between median effluent temperatures and the 21°C threshold when examining the entire monitoring period; however, median effluent temperatures were significantly ( $p<0.05$ ) cooler than 21°C when the month of August was excluded. Both influent and effluent temperatures exhibited substantial seasonal variations with the warmest influent and effluent temperatures during the month of August (Fig. 6).

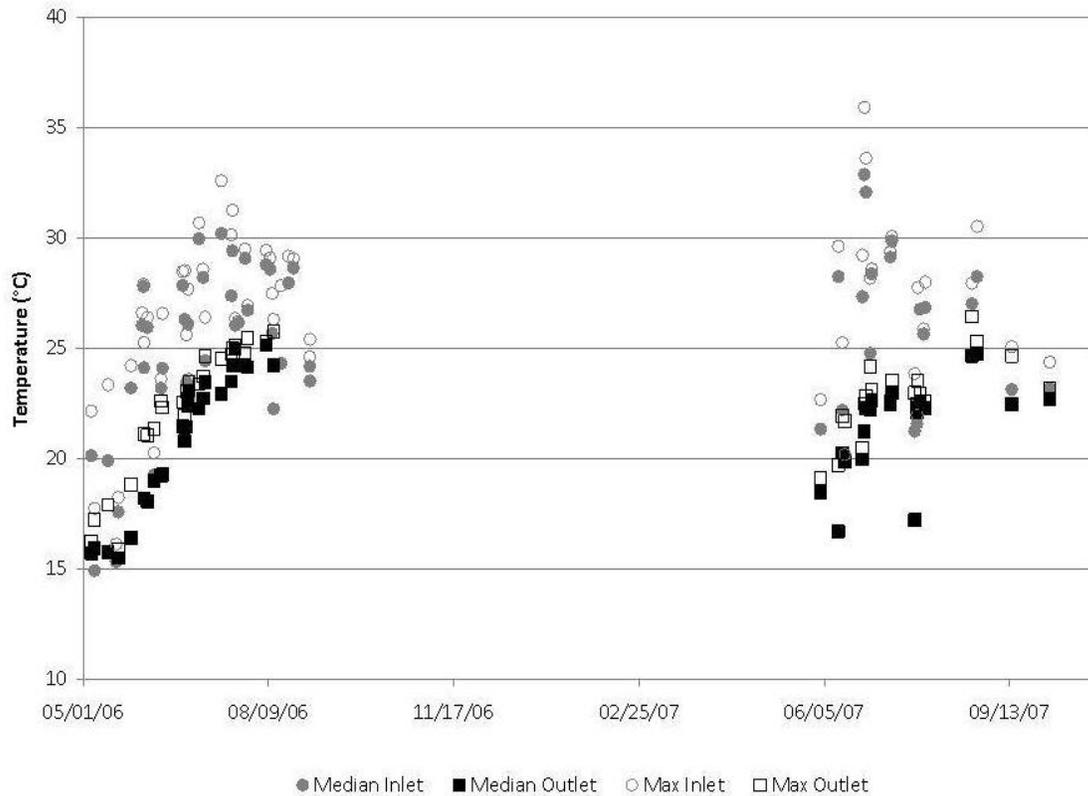


**Fig. 6:** Influent and effluent temperatures at the Asheville bioretention area

There was no significant ( $p>0.05$ ) difference between median influent and effluent temperatures at the Asheville bioretention area, indicating that the bioretention area was not able to consistently reduce runoff temperatures over the course of an entire storm. Maximum effluent temperatures were significantly ( $p<0.05$ ) cooler than the maximum influent; however, maximum effluent

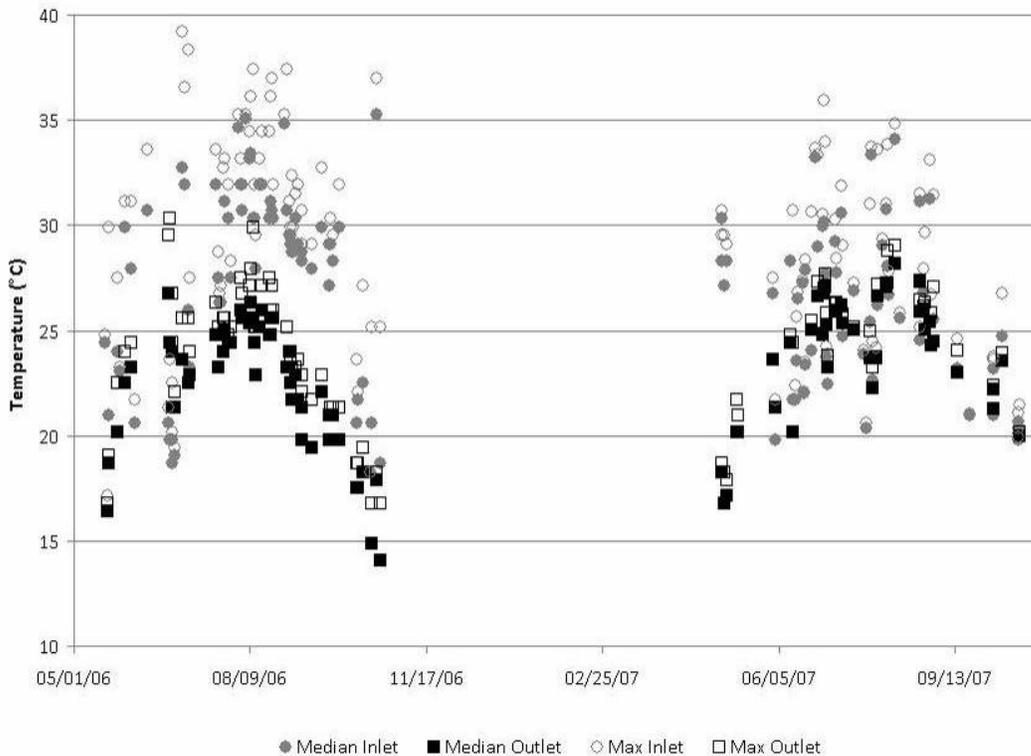
temperatures were not significantly ( $p>0.05$ ) different from the 21°C threshold. A reduction in maximum water temperatures suggests that the bioretention area was able to reduce the initial runoff temperature spike, but unable to adapt to the cooler runoff as a storm progresses.

At the Lenoir bioretention area, median and maximum influent temperatures were significantly warmer than 21°C for the entire monitoring period, suggesting potentially negative impacts if runoff were directly discharged into trout waters. Median and maximum effluent temperatures were both significantly lower than influent temperatures at the Lenoir bioretention area, which indicates that the bioretention area was able to reduce the thermal impact associated with the stormwater runoff. Despite the reduction in water temperature resulting from bioretention treatment, median effluent temperatures were not significantly ( $p>0.05$ ) different from 21°C and maximum effluent temperatures were significantly warmer than 21°C. Although thermal impacts were reduced, the effluent from this bioretention area still posed some risk to the thermal environment of trout waters. Similar to the Asheville bioretention area, influent and effluent temperatures varied seasonally (Fig. 7).



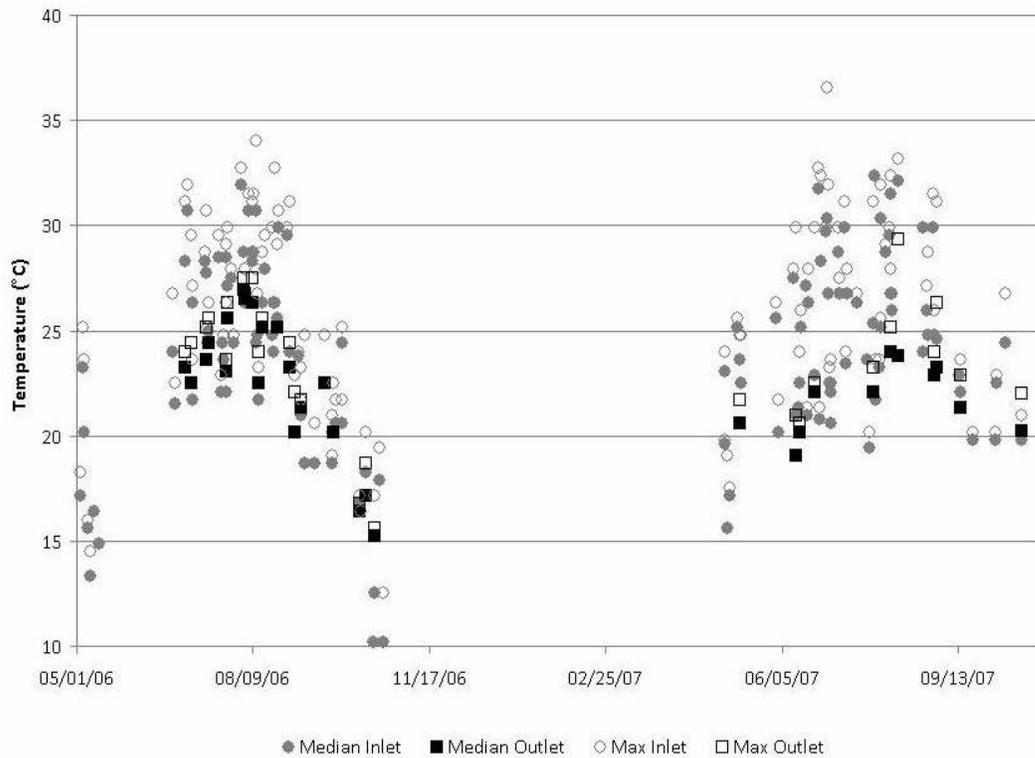
**Fig. 7:** Influent and effluent temperatures at the Lenoir bioretention area

At the Brevard east bioretention, median and maximum inlet temperatures were significantly warmer than 21°C, meaning direct runoff would have been a source of thermal pollution. Median and maximum effluent temperatures were significantly cooler than inlet temperatures; however, effluent temperatures were also significantly warmer than 21°C. Similar to the Lenoir bioretention, the Brevard east bioretention was able to reduce, but not eliminate the thermal impact to a coldwater stream environment. Also similar to other sites, influent and effluent temperatures were coolest during the spring and fall, which corresponds to spawning seasons for North Carolina trout species (Fig. 8).



**Fig. 8:** Influent and effluent temperatures at the Brevard East bioretention area

At the Brevard west bioretention, median and maximum inlet temperatures were significantly warmer than 21°C. Although there was no significant ( $p>0.05$ ) difference between median influent and effluent temperatures, there was a significant difference between maximum influent and effluent temperatures. These temperature results indicate that the bioretention area was likely able to reduce the initial spike in runoff temperatures, but could not adapt to the cooler runoff temperatures later in a storm. Median and maximum effluent temperatures were both significantly warmer than 21°C, indicating potential thermal impacts to a coldwater stream environment. Seasonal trends in influent and effluent temperatures were not as well defined during 2007, possibly due to the impact of the drought throughout the region (Fig. 9).



**Fig. 9:** Influent and effluent temperatures at the Brevard West bioretention area

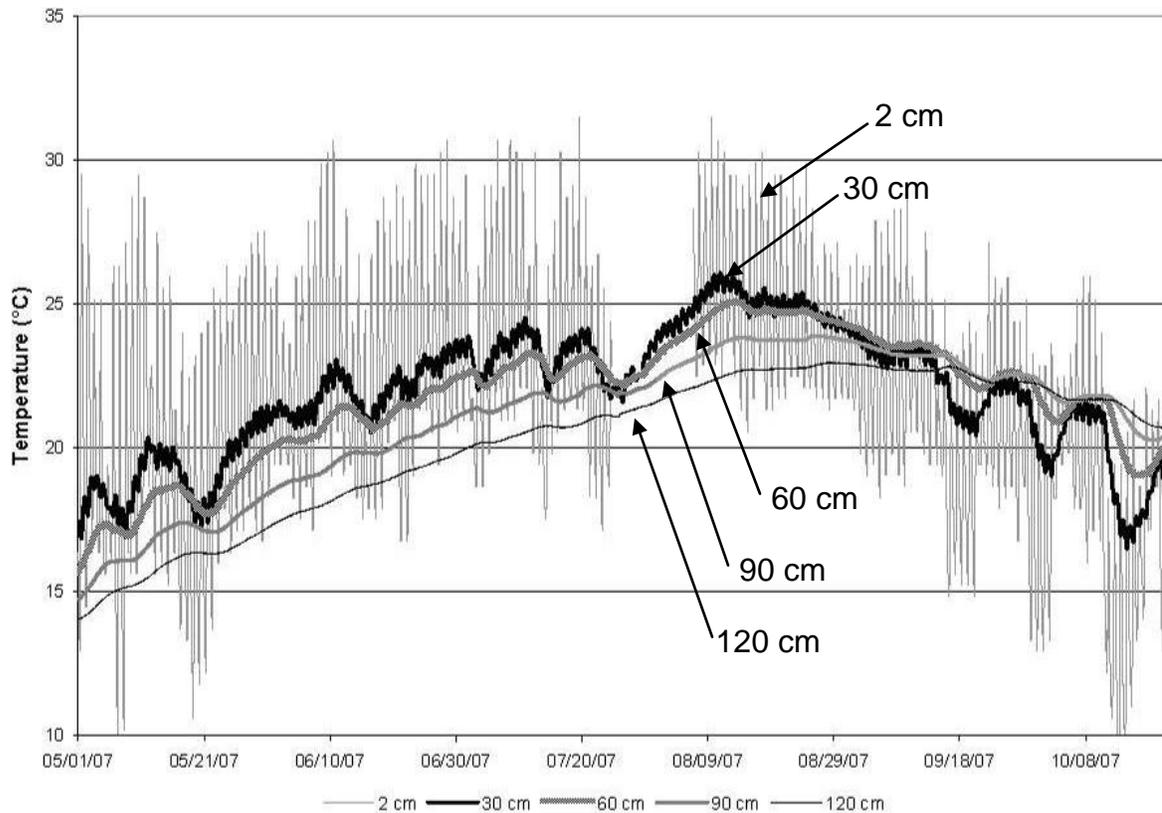
The median influent temperature at the Brevard east bioretention area was significantly warmer than inflow into the Brevard west bioretention area; however, there was no significant ( $p > 0.05$ ) temperature difference in median effluent temperature between the two systems. With immediately adjacent locations, similar soil properties, drain depth, and bioretention size, the primary difference between these systems was the size of their contributing watershed. The area occupied by the Brevard east bioretention was equal to 7% of the contributing watershed, while the area of the Brevard west bioretention was equivalent to 11% of the contributing watershed. Temperature results from this pair of bioretention areas indicate that there were minimal benefits of moderately oversizing a bioretention area with regards to thermal pollution. However, one benefit of a proportionally larger bioretention area that is not evident directly from temperature measurements was the ability of the larger system to infiltrate runoff without generating outflow. When

seepage occurs and water leaves the bioretention area through the underlying soil and not the drainage pipes, the thermal impact from the runoff is effectively eliminated. Despite receiving the same rainfall, measurable outflow occurred in response to 76% of rainfall events at the Brevard east bioretention area, while only 27% of rainfall events at the Brevard west bioretention area generated outflow (Table 5). Even during storms that generate outflow, substantial reductions in runoff volume and consequently thermal load are likely, due to seepage and evapotranspiration.

**Table 5:** Measurable inflow and outflow events at each bioretention area

	<b>Inflow Events</b>	<b>Outflow Events</b>	<b>Inflow Events with Outflow</b>
<b>Asheville</b>	89	11	12%
<b>Lenoir</b>	58	46	79%
<b>Brevard East</b>	127	96	76%
<b>Brevard West</b>	128	34	27%

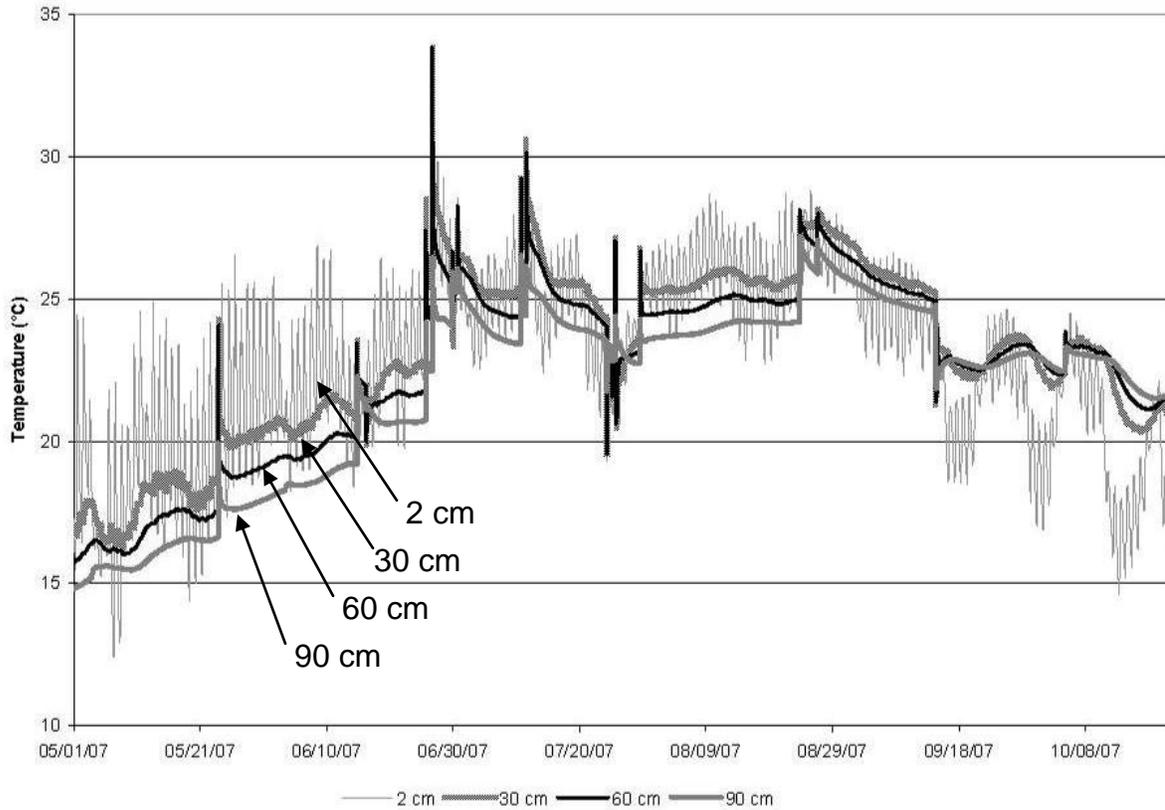
Storm events generally had a negligible effect on soil temperatures at the Asheville bioretention area (Fig. 10). Even during the largest storms, runoff temperatures appeared to reach thermal equilibrium with the surrounding soil after infiltrating only 60 cm. The relative stability of soil temperatures likely had an impact on the ability of the Asheville bioretention area to mitigate thermal pollution. A significant ( $p < 0.05$ ) difference between maximum influent and effluent temperatures was evident because initial runoff temperatures were warmer than soil deep within the bioretention area. When pavement temperatures cooled as a storm progressed, runoff became cooler than soils deep within the bioretention area, causing the bioretention area to raise the temperature of infiltrating water above that of the influent during the later portions of a storm. Although a large bioretention area, such as the one at Asheville, may result in predictable effluent temperatures, the inability of soils to cool in response to cooler runoff is a substantial disadvantage towards their role in mitigating thermal pollution.



**Fig. 10:** Soil temperatures within the Asheville bioretention area during 2007

Soil temperatures below the surface at the Lenoir bioretention area were much more dynamic than those at the Asheville bioretention area (Fig. 11). Soil temperatures generally increased at all depths in response to a storm event. Median influent temperatures at the Lenoir bioretention were significantly warmer than those at the Asheville bioretention, which may have been responsible for some of the differences in soil temperature responses. Another substantial difference between the two systems was the area they occupied in relation to their contributing watershed. The Lenoir bioretention covered an area equivalent to 4% of the contributing watershed, while the Asheville bioretention occupied the equivalent of 16% of the contributing watershed. With approximately four times more soil relatively available to absorb heat at the Asheville site, it is reasonable that soil media temperature changes during a storm would be limited to shallow soil depths. Diurnal soil temperature fluctuations near the surface were larger at the Asheville

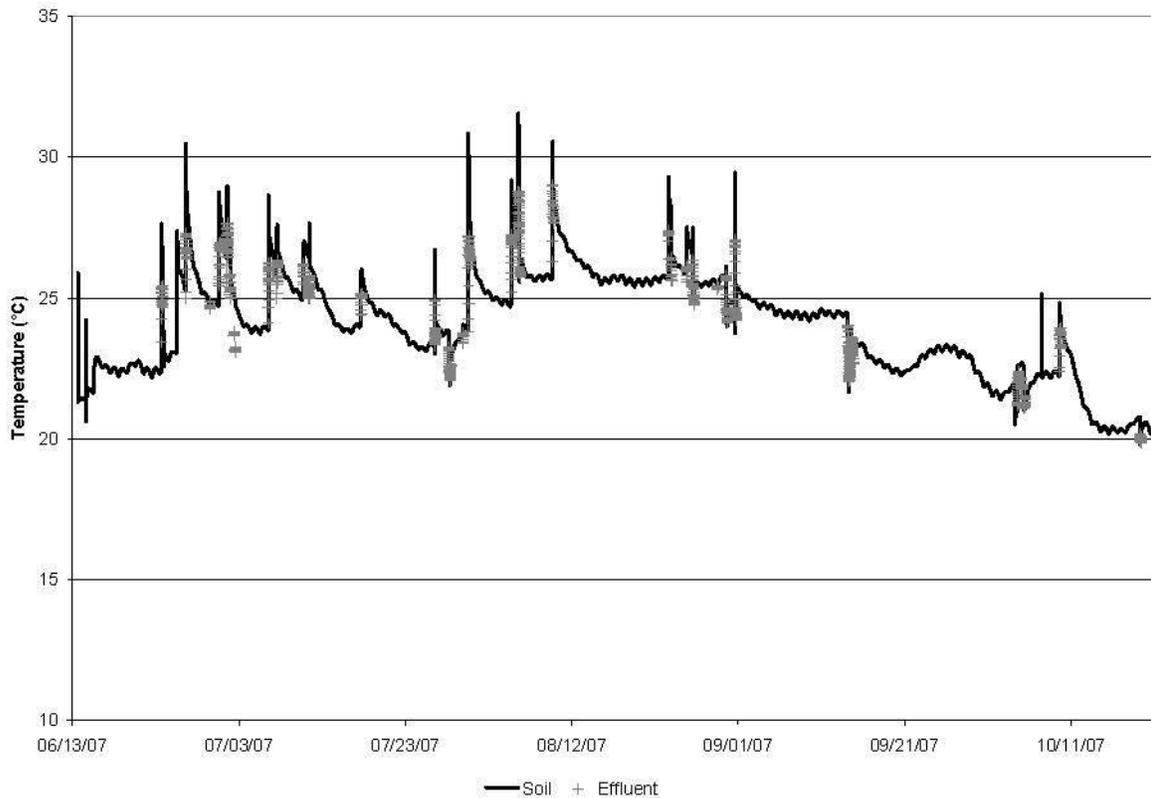
bioretention area, which was likely due to the relatively smaller amount of vegetative shading that site received. Soil temperatures near the surface at both bioretention areas often cooled following storm events, probably due to heat losses from evaporation and cooler air temperatures.



**Fig. 11:** Soil temperatures within the Lenoir bioretention area during 2007

Similar to the Lenoir bioretention area, soil temperature fluctuations were also observed at the Brevard east bioretention area in response to storm events (Fig. 12). Although the depth of the underdrains was relatively shallow at 48 cm, the size of the bioretention area fell within general sizing guidelines at 7% of the contributing watershed. At times, the effect of storms on soil temperatures was prolonged, taking several days for the soil to cool to antecedent temperatures. Differences between soil and effluent temperatures at all three sites where soil temperature was

monitored suggest that additional cooling occurs as water is collected by the underdrain network and discharged.



**Fig. 12:** Brevard East soil temperature at a depth of 40 cm during 2007

Analysis of temperature trends at specific soil depths provided insight into optimum underdrain depths. With the exception of nighttime surface temperatures, soil temperatures were coolest at the bottom of the bioretention soil column during the summer months. Soil temperatures at the bottom of the bioretention areas also exhibited the smallest fluctuations in response to storms or diurnal and seasonal temperature changes. However, the stability of soil temperatures at greater depths poses a potential risk of increasing the temperature of infiltrating water during later portions of a storm when cooler runoff prevails. Although there were several exceptions during late night storms, soils at a depth of 90 cm or greater remained cooler than soils at shallower depths for most storms. Beginning in September, soil

temperatures were often warmest at the bottom of the bioretention areas, which raises potential concerns of warm effluent from deep bioretention areas during spawning seasons in the fall when preferred temperatures are cooler.

The time of the beginning of each storm bore a significant ( $p < 0.05$ ) negative linear correlation with median and maximum influent and effluent temperatures at both the Brevard east and Brevard west bioretention areas. The correlation between time and effluent temperature may be attributed to the warmer influent temperatures or warmer soil temperatures due to the shallow drain depth. Overall, the correlation may indicate the inability of relatively shallow bioretention areas to buffer temperature changes near the surface. There was a significant negative linear correlation when time was compared to median and maximum influent temperatures at the Lenoir bioretention area and Asheville bioretention area; however, there was no significant ( $p > 0.05$ ) linear correlation between time and median or maximum effluent temperatures at those sites. It is not surprising that time of day did not have a significant effect on effluent temperatures at the deeper bioretention areas of Asheville and Lenoir, since the greater soil depths should buffer any diurnal effects due to the insulating properties of the soil. The greater soil depths also allowed for runoff temperatures to equilibrate with the surrounding soil, reducing the effects of varying influent temperatures.

There was a significant positive correlation between median and maximum influent and effluent temperatures at the Lenoir, Brevard east, and Brevard west sites, with warm influent indicating warm effluent. This correlation is supported by changes in soil temperatures in response to storm events. No significant ( $p > 0.05$ ) correlation between maximum influent and effluent temperatures was observed at the Asheville bioretention area. There was a small, but significant ( $p < 0.05$ ) positive correlation between median influent and effluent temperatures at the Asheville site. Because soils deep within the Asheville bioretention area did not exhibit temperature changes in response to storm events, it is not surprising that influent and effluent

temperatures were not strongly correlated. The variance of influent temperature within each storm was significantly greater than the variance in effluent temperature at all four bioretention areas. The reduced variance at the outlet may be indicative of the ability for deeper soils within the bioretention areas to buffer changes in both water and soil temperature experienced near the surface during a storm.

### **Conclusions and Summary**

Monitoring results suggest that bioretention areas are a viable option for reducing the thermal impacts of urban stormwater runoff. The two bioretention cells that covered a smaller area with respect to their watershed (Lenoir and Brevard east) were able to significantly reduce both maximum and median water temperatures between the inlet and outlet. Because it is not possible for runoff flows to increase as a result of bioretention treatment, these systems clearly reduced the thermal load associated with urban stormwater runoff. This reduction in temperature differs from the results Dietz and Clausen (2005) obtained at a bioretention area in the northeastern United States, which may be attributed to the warmer influent temperatures observed at the North Carolina locations and increased measurement frequency. Although the two bioretention cells that covered a larger area with respect to their watershed (Asheville and Brevard west) were only able to significantly ( $p < 0.05$ ) reduce maximum water temperatures between the inlet and outlet, there was evidence that these systems generated less outflow, effectively reducing the thermal load. The ability of bioretention areas to decrease runoff temperatures and reduce runoff volumes through seepage suggests that they may be better suited for coldwater stream environments than stormwater wetlands and wet ponds. Despite these temperature reductions, effluent temperatures were still warmer than the 21°C temperature threshold for trout, indicating additional cooling may be necessary to completely eliminate the thermal impact from urban stormwater runoff.

The ability of bioretention areas to reduce runoff volumes was considered to be a major benefit of their use in trout sensitive regions. When runoff volumes are reduced, the thermal impact to the receiving stream is decreased, as long as there are not large increases in temperature resulting from bioretention treatment. Furthermore, bioretention areas mimic predevelopment hydrology by recharging shallow groundwater supplies, which constitute a major water source for many coldwater fisheries. The ability of bioretention areas to mimic predevelopment hydrology is distinctly different from other stormwater BMPs, such as stormwater wetlands and wet ponds, where water largely remains on the surface and volume reductions are relatively minimal.

The largest runoff volume reductions are expected for locations where the hydraulic conductivity of the soil underlying the bioretention area is substantial. In some of these locations, underdrains may not be required; however, when underdrains are not incorporated into bioretention design, there is an increased risk of generating overflow during a storm event. When the hydraulic conductivity of the underlying soil is high enough to completely drain the bioretention area between storm events, the thermal impact of overflow is likely minimal since overflow would occur later in a storm when runoff temperatures have cooled. If the bioretention area has not been adequately drained between storm events, overflow may begin early in a storm event when the warmest runoff temperatures were observed, negating the temperature reductions of the bioretention area. An alternative design where a storage layer is included below the underdrains or an upturned elbow is used to create an internal water storage zone would allow for increased seepage into the underlying soil, while also minimizing the risk of surface overflow.

The behavior of soil temperatures within the monitored bioretention areas provided insight into how the systems functioned. At the Asheville bioretention area, proportionately the largest system studied, soil temperature trends were in general agreement with results Wierenga et al. (1970) obtained when irrigating soil with

warm water, with water temperature equilibrating with the surrounding soil after infiltrating through less than one meter of soil. At bioretention areas falling within conventional sizing guidelines, soil temperatures fluctuated throughout the entire soil column in response to storm events, which can be attributed to higher mass transfer rates per unit of soil area. Although fluctuations existed, the magnitude of temperature change was reduced deep within all bioretention areas, increasing the predictability of effluent temperatures. Effluent temperatures also followed seasonal patterns corresponding to soil and pavement temperatures. The ability to predict effluent temperatures from a bioretention area has important implications for the development of temperature TMDL programs.

Bioretention design parameters appear to play an important role in the effectiveness of bioretention areas in mitigating thermal pollution from urban stormwater runoff. Installation of larger bioretention areas, with respect to the watershed size, does not seem to have substantial benefits with regards to temperature reduction; however, greater reductions in runoff volume appear to have major implications for reducing the thermal load to coldwater stream environments. Bioretention areas should not be lined and should be sited in locations with high underlying soil hydraulic conductivities when possible to encourage movement of stormwater runoff into the shallow groundwater. When compared with stormwater wetlands and wet ponds, the ability of bioretention areas to reduce runoff volumes may be the BMP's greatest asset. Monitoring results indicate that during the summer months, water temperatures were typically coolest after reaching greater soil depths indicating that deeper bioretention areas may be better suited in regions where thermal pollution is a concern. It is possible for soil depth to be too great for temperature reduction, since the temperature of deep soils does not decrease in response to cooler runoff temperatures and temperatures at greater depths are warmer than shallower depths during trout spawning seasons in the fall. Despite these concerns, underdrain depths between 90 and 120 cm appear to be practical for most applications.

Due to the wide variety of possible bioretention configurations, additional research is needed to examine the effects of these varying designs. Additionally, results should be confirmed for other regions around the world where coldwater fisheries are a concern, since North Carolina lies along the southeastern extent of United States trout populations. Although the thermal impacts of bioretention treatment were assessed in the current study, there are a number of complex factors that affect the thermal impact of stormwater discharges on the temperature of coldwater stream environments. Specifically, detailed measurements of stormwater and stream flows and temperatures are required. Additional monitoring and modelling efforts are needed to better understand the effect of direct stormwater discharges and BMP effluents on these waters and evaluate ecological impacts. Based on the results of this study, it is evident that with careful consideration in BMP design, bioretention areas should serve as suitable treatment mechanisms for thermal pollution from urban stormwater runoff.

### **Acknowledgements**

This research was funded by the North Carolina Department of Environment and Natural Resources (NCDENR), Division of Water Quality. The authors would like to thank Jonathan Smith, Dr. Dan Willits, Dr. Garry Grabow, Dr. Aziz Amoozegar, Jon Calabria, Allen Caldwell, Eric Caldwell, Seth Nagy, and Jason Zink, all of whom are currently or formerly of NC State University, for their assistance in selecting research sites and support throughout the project.

### **References**

- Asaeda, T., Ca, V. T., and Wake, A. (1996). "Heat Storage of Pavement and its Effect on the Lower Atmosphere." *Atmospheric Environment*, 30(3), 413-427.
- Dietz, M. E., and Clausen, J. C. (2005). "A Field Evaluation of Rain Garden Flow and Pollutant Treatment." *Water, Air, and Soil Pollution*, 167 123-138.
- Hocutt, C. H., Stauffer, Jay R., Jr., Edinger, J. E., Hall, Lenwood W., Jr., and Morgan, Raymond P., II. (1980). *Power Plants: Effects on Fish and Shellfish Behavior*, Academic Press, New York, NY.

- Hunt, W. F., Jarrett, A. R., Smith, J. T., and Sharkey, L. J. (2006). "Evaluating Bioretention Hydrology and Nutrient Removal at Three Field Sites in North Carolina." *Journal of Irrigation and Drainage Engineering*, 132(6), 600-608.
- Jones, M. P., and Hunt, W. F. (2008). "The Effect of Stormwater Wetlands and Wet Ponds on Runoff Temperature in Trout Sensitive Waters." *Biological Engineering*, Submitted Oct. 07.
- Kieser, M. S., Spoelstra, J. A., Feng Feng, A., James, W., and Li, Y. (2004). *Stormwater Thermal Enrichment in Urban Watersheds*. Water Environment Research Foundation, Alexandria, Va.
- Mohseni, O., Erickson, T. R., and Stefan, H. G. (2002). "Upper Bounds for Stream Temperatures in the Contiguous United States." *Journal of Environmental Engineering*, 128(1), 4-11.
- North Carolina Department of Environment and Natural Resources (2007). *North Carolina Water Quality Assessment and Impaired Waters List (2006 Integrated 305(b) and 303(d) Report)*, Raleigh, NC, 28.
- North Carolina Climate Office. (2008). "NC Climate Retrieval and Observations Network Of the Southeast Database." <<http://www.nc-climate.ncsu.edu/cronos>> (Jan 08, 2008).
- Wierenga, P. J., Hagan, R. M., and Nielsen, D. R. (1970). "Soil Temperature Profiles During Infiltration and Redistribution of Cool and Warm Irrigation Water." *Water Resources Research*, 6(1), 230-238.
- Wilcoxon, F. (1945). "Individual Comparisons by Ranking Methods." *Biometrics*, 1(6), 80-83.

## **Chapter 4**

### **Effect of Urban Stormwater BMPs on Runoff Temperature: Additional Monitoring Study Observations**

## **Introduction**

Due to the preparation timeframe and concise nature of the journal article submissions presented in chapters 2 and 3, there are several components of the monitoring study that merit further explanation. These topics include details of the flow monitoring equipment, assessment of vegetative shading, dynamics of runoff temperature measurements, and stormwater wetland and wet pond monitoring results from 2007. This discussion provides a better understanding of thermal pollution from urban stormwater runoff, as well as an indication of several mitigating factors that can be incorporated into urban watershed designs.

## **Methods and Materials**

### **Flow Monitoring Equipment**

The greatest problem encountered during the course of the monitoring study was related to the flow monitoring equipment. A common method of measuring flow in stormwater monitoring applications is to use a sharp-crested weir in conjunction with a water level recorder. Sharp-crested weirs are a valuable tool for assessing stormwater flow because water flowing over these devices passes a critical depth. As a result, water level in advance of the weir can be correlated to the flow rate over the weir (Equation 1). A v-notch weir is often used in stormwater monitoring due to its ability to accurately describe flow over a relatively wide range of flow rates (Figure 4.1). All weir plates for this monitoring study were constructed in the NCSU Biological and Agricultural Engineering (BAE) research shop. Most weir plates were calibrated over the full range of potential water levels in a teaching flume; however, weir plates used to monitor BMP inflow rates were too large to be calibrated with the teaching flume. When possible, larger weir plates were calibrated by supplying water from an outdoor faucet at a variety of flow rates.

**Equation 1:** Discharge equation for 30° v-notch weir where Q is flow (cfs) and H is head above weir crest (ft)

$$Q = 0.4369 \cdot H^{2.5}$$



**Figure 4.1** V-notch weir installed at the Asheville stormwater wetland outlet

In order to obtain a record of flow at all monitoring locations, a device that measures water level upstream of the weir and stores those measurements must be used. Typically, these measurements are obtained from either a pressure transducer or pulley-float system in conjunction with a data logger that records at a regular interval. Before the author was involved in the monitoring project, a number of pulley-float systems were obtained to monitor flow rates at the BMP monitoring sites due to their reduced cost and an expected increase in accuracy (Figure 4.2). These pulley-float systems were constructed by BAE staff. The pulley-float systems consisted of a float that was counterbalanced by a weight. Both the float and weight were attached to pulleys fixed on the same shaft. This shaft was connected to a potentiometer (Figure 4.2). A voltage logging circuit supplied a constant voltage to the potentiometer and recorded the voltage output from the potentiometer at a

regular interval. Before deployment in the field, voltage measurements recorded by the logger were correlated to the position of the float.



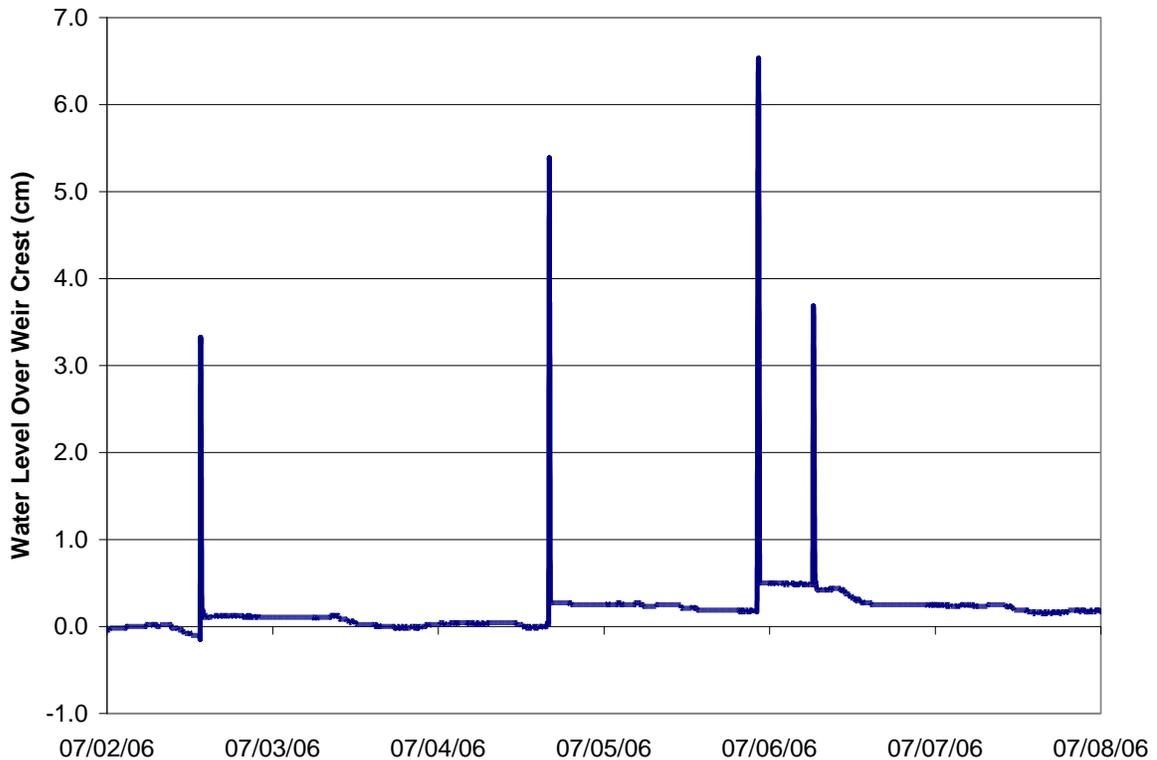
**Figure 4.2** Pulley-float water level recorder (left) and shaft connection (right)

In theory, the pulley-float water level recorders were expected to be more accurate than pressure transducer systems, since water level measurements would not be affected by changes in pressure due to temperature and other factors. In practice however, the pulley-float systems were fraught with errors and difficulties. First, because the systems were substantially larger than most pressure transducers, they were relatively difficult to install. The larger size of these systems had to be accounted for in the design of weir boxes and installation of the systems within pipes or culvert boxes. Also, each pulley-float system had to have a separate structure installed to support the device. At the Brevard monitoring locations, metal frames had to be installed over the pulley-float systems to prevent vehicles from crushing the devices (Figure 4.3).



**Figure 4.3** Metal frame installed to protect inflow monitoring equipment at Brevard west bioretention (left), and damaged inflow monitoring equipment before protective frames were installed (right)

Early in the monitoring project, modifications to the logger software were necessary to prevent data loss while downloading from the logger. This problem was attributed to the lack of flow control in the serial communication. Although difficulties with equipment installation and logger download were inconveniences, the largest problems were associated with the accuracy of the monitoring devices. Errors in water level measurements were observed early in the monitoring study when the water level recorded in a weir box before a storm would not equal the water level after a storm event (Figure 4.4). Errors in water level measurements were confirmed when flow calculations resulted in effluent volumes greater than influent volumes for a number of storm events.



**Figure 4.4** Water level recorded within outlet weir box at Lenoir bioretention, illustrating shifts in depths measurements, which should be 0 between storm events

There were several factors that appeared to contribute to the inaccuracy of the pulley-float stage recorders. At times, the float would become stuck within the surrounding perforated PVC pipe as it was descending with the receding water level. The insides of all perforated PVC pipes were sanded and smoothed during the summer of 2005, which largely eliminated these problems. Another factor that introduced error into water level measurements was the connection between the pulley shaft and the potentiometer. To allow for adjustment of the potentiometer position during installation and subsequent deployment, a length of vinyl tubing was used to connect the two shafts. Because this tubing was flexible, some small movements of the pulley shaft would twist the tubing, but not move the potentiometer shaft. A larger concern was that the vinyl tubing would sometimes allow the potentiometer shaft to slip, preventing the potentiometer from registering

any movement of the pulley shaft. In order to address this problem, silicone adhesive was applied to the connection of the two shafts during the 2006 monitoring period. The silicone adhesive was intended to solidify the connection, while still allowing for the shafts to be separated for adjustments. At the beginning of the 2007 monitoring season, a rigid epoxy was used to attach the two shafts. After both applications of adhesive, all pulleys were rotated several times through the full range of potential water levels while taking numerous level recordings with the voltage logger. Visual inspection of the shaft connection and analysis of the recorded data indicated level measurements were correct. Despite the success of these manual tests, errors in water level measurements continued to appear in monitoring data during actual storm events.

There were several additional factors that may have contributed to the inaccuracy of the pulley-float systems. One potential problem was that the floats may have absorbed water over time, causing their buoyancy to change and introducing error into the water level recording. The bottoms of all floats were coated with an impermeable polymer for the duration of the study, but small amounts of water absorption may have persisted. Another factor that may have introduced error was the presence of moisture on the potentiometer and voltage logger circuitry. Although a vent was installed in all monitoring enclosures, condensation was often observed within the systems because they were positioned over standing water in the weir boxes. Creating an airtight enclosure for the potentiometer and voltage logger would have required a major change in system design and was not practical to implement during the course of this monitoring study.

Overall, the custom-built nature of the pulley-float systems made it such that numerous components had to work and interface precisely in order for accurate water level measurements to be realized. In applications where pulley-float water level recorders have been commonly used, such as water table monitoring, these relatively small errors are probably not even noticed. When used for monitoring

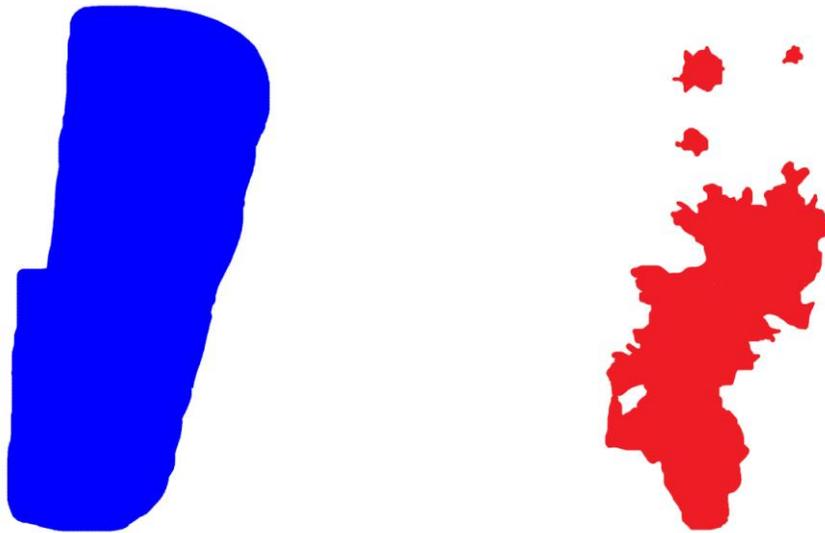
stormwater flows, however, water level measurement inaccuracies as small as several millimeters could result in large errors in runoff volume calculations. Unfortunately, errors introduced by the monitoring equipment made it impossible to reliably evaluate the effect of the monitored BMPs on runoff volumes and consequently thermal loads. Water level data recorded by the pulley-float systems were used to identify periods of flow by visually inspecting graphs of water level measurements and making adjustments for observed shifts in water levels.

### **Estimation of Vegetative Shading**

The amount of shading provided by vegetation on the surface of each BMP was estimated using a series of overhead digital photographs in late August of 2006. An aluminum angle bracket was fabricated to attach a digital camera to the top of a survey rod. The self-timer was set on the digital camera and the survey rod was raised to a height of approximately 3 m. For the bioretention areas, a series of photographs were taken to provide imagery for the entire BMP surface. Due to the size of the stormwater wetland, it was impractical to take photographs of the entire wetland surface; therefore, a series of photographs were taken at various locations within the stormwater wetland to provide an estimate of vegetative cover. For the bioretention areas, the overhead photographs were assembled in Adobe Photoshop™ to provide a composite image of the entire bioretention surface (Figure 4.5). All areas covered by vegetation were painted based upon visual inspection with a red brush in Photoshop. The entire bioretention surface was painted with a blue brush (Figure 4.6). The number of pixels of each color was counted using the image analysis program ImageJ™, provided by the National Institutes of Health (National Institutes of Health, 2008). The ratio of pixels representing vegetation, to the total number of pixels contained within the bioretention area was calculated to determine the percentage of total BMP area that was shaded by vegetation.



**Figure 4.5** Composite overhead image of Brevard east bioretention (left) and camera setup (right)



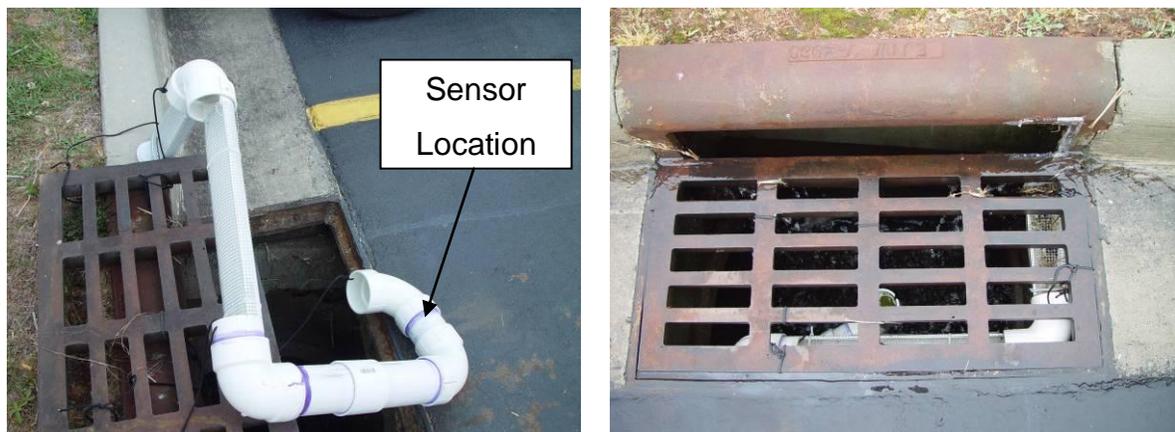
**Figure 4.6** Painted overhead photographs used to evaluate vegetative shading at the Brevard east bioretention site

## **Results and Discussion**

### **Effect of Runoff Temperature Measurement Location**

Because runoff was observed to cool substantially while conveyed through buried pipes, it was assumed that runoff temperatures measured within a culvert below a drop inlet may be substantially cooler than runoff directly leaving the parking

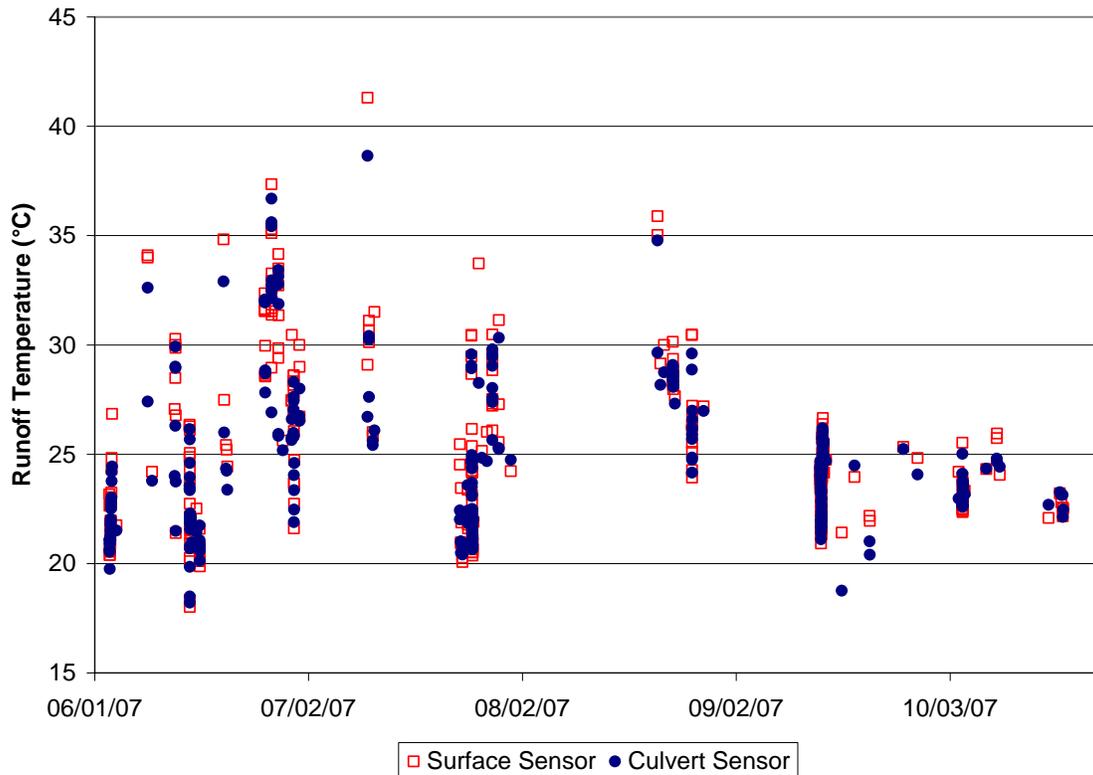
surface. Due to the configuration of the parking lot drainage, runoff temperature sensors at the Lenoir wet pond and Asheville stormwater wetland measured the temperature of water falling from the inlet above, as well as water flowing underground from other drop inlets. To evaluate the effect of the temperature probe location on runoff temperature measurements, an apparatus was installed within a drop inlet at the Lenoir wet pond to capture runoff directly from the parking surface (Figure 4.7). Water was collected by two open sections of PVC pipe which were covered with wire mesh to reduce clogging. The temperature logger was installed within an upturned elbow on the lower portion of the apparatus that detained a relatively small volume of runoff before discharging water into the culvert below. The temperature logger used to monitor runoff temperatures during the 2006 monitoring period and used for comparison during the 2007 monitoring period was installed within the culvert directly below this drop inlet.



**Figure 4.7** Apparatus installed to capture runoff directly from wet pond parking lot before installation (left) and during a storm event (right)

Results of a Wilcoxon signed rank test (Wilcoxon, 1945) showed that maximum runoff temperatures measured within the collection apparatus were significantly ( $p < 0.01$ ) warmer than runoff measured in the culvert below; however, there was no significant ( $p < 0.05$ ) difference in median runoff temperatures (Figure 4.8). This analysis suggests that measurements used in evaluating the effects of

stormwater wetlands and wet ponds on runoff temperature in Chapter 2 accurately represented the overall temperature of runoff leaving the site during a storm by measuring runoff at the bottom of the drop inlet structure; however, maximum runoff temperatures were likely underrepresented.



**Figure 4.8** Temperature measurements collected within the surface runoff capture apparatus and culvert

### Effect of Tree Canopy on Runoff Temperature

One mechanism for reducing the thermal impact of urban stormwater runoff is to decrease the surface temperature of the contributing watershed. An evaluation of the effect of a mature tree canopy surrounding the parking surface was conducted at the Lenoir bioretention area and wet pond, located 2.4 km apart from each other. The asphalt parking lot contributing runoff to the bioretention area was surrounded

by a mature tree canopy, while the wet pond asphalt parking lot received essentially no shading from vegetation (Figure 4.9).



**Figure 4.9** Contributing watersheds for Lenoir bioretention area (left) and wet pond (right)

According to a Wilcoxon signed rank test, the median and maximum temperature of runoff leaving the un-shaded wet pond parking lot was significantly ( $p < 0.05$ ) warmer than runoff leaving the shaded bioretention parking lot during the 2007 monitoring period (Figure 4.10). During the 2006 monitoring period, there was a significant ( $p < 0.05$ ) difference in storm maximum temperatures, but not storm median temperatures. Over the entire monitoring period, median runoff temperatures leaving the shaded parking lot were  $0.43\text{ }^{\circ}\text{C}$  cooler than runoff leaving the parking lot without shading. These results suggest that shading an impermeable surface may indirectly minimize the thermal impact of treated stormwater runoff by reducing BMP influent temperatures; however, further research is needed to quantify specific temperature reductions.



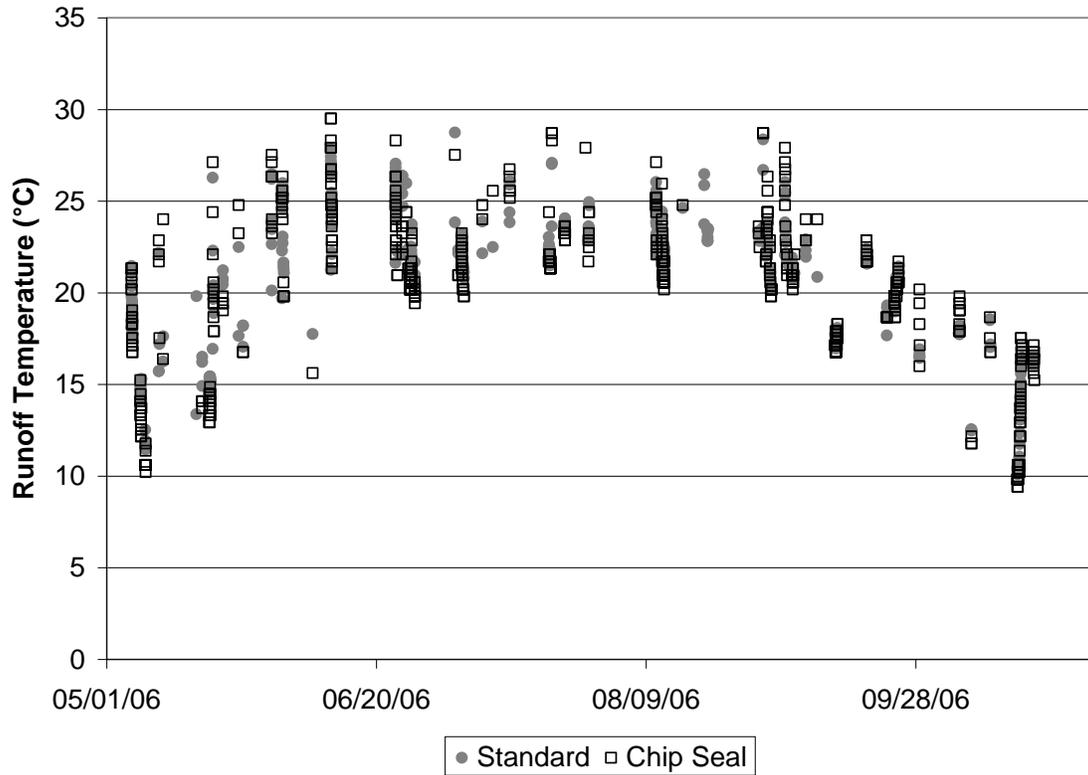


**Figure 4.11** Contributing watershed for Asheville bioretention before (left) and after (right) application of light colored chip seal

The light colored chip seal was installed in an effort to reduce the contribution of the asphalt surface to the urban heat island effect. Because the nearby asphalt parking lot contributing water to the Asheville stormwater wetland did not have a chip seal applied to its surface, it was possible to compare runoff temperatures from these sites to evaluate the impact of the light colored chip seal on runoff temperatures.

Over the course of the 2006 monitoring period, a Wilcoxon rank sum test showed that there was no significant ( $p < 0.05$ ) difference between median and maximum runoff temperatures at the two locations (Figure 4.12). Median runoff temperatures were  $0.73^{\circ}\text{C}$  cooler at the chip seal parking lot over the course of the monitoring period. Despite the lack of a significant difference, there were several factors that likely contributed to cooler runoff temperatures at the conventional asphalt location that were not present at the chip seal location. The wetland parking lot was surrounded by a canopy of mature trees, which provided shading for the parking surface and likely cooler asphalt temperatures (Figure 4.13). The parking surface at the wetland parking lot was also lighter in color than the newly paved surface at the bioretention location before the chip seal was applied, due to normal aging processes. Although there was no significant ( $p < 0.05$ ) difference in runoff temperatures between the two monitoring locations, it is possible that the light

colored chip seal had similar cooling effects as a mature tree canopy and aged parking surface. Certainly more tests are necessary to definitively evaluate the effect of a light colored chip seal on runoff temperature.



**Figure 4.12** Recorded runoff temperatures at the Asheville bioretention area and stormwater wetland during the 2006 monitoring period



**Figure 4.13** Aerial photograph showing location of Asheville stormwater wetland and bioretention area (Parking surfaces shaded red)

#### **Stormwater Wetland and Wet Pond Observations from 2007 Monitoring**

Because the journal article submission presented in Chapter 2 was prepared before the conclusion of the 2007 monitoring period, it does not contain any monitoring results from 2007. Overall, the trends observed during the 2005 and 2006 monitoring periods persisted in 2007. One important difference between the monitoring periods was the lack of rainfall during the summer of 2007 (Table 4.1). On numerous occasions during the 2007 monitoring period, substantial areas of the stormwater wetland and wet pond remained dry (Figure 4.14). These dry conditions made it difficult to compare monitoring results collected in 2007 with those from previous years. Because water levels were often more than 20 cm below the normal pool elevation, additional storage capacity was available within the stormwater wetland and wet pond. This additional storage capacity allowed these systems to capture runoff from entire storm events without generating much outflow, effectively eliminating the thermal load of the urban stormwater runoff. This impact was best

observed at the stormwater wetland, since leaks in the outlet structure for the wet pond resulted in constant outflow, regardless of water level.

**Table 4.1** Measured and 30-year normal (1971-2000) rainfall depths (cm) for weather stations near monitoring locations

<b>Asheville <sup>a</sup></b>	<b>2005</b>	<b>2006</b>	<b>2007</b>	<b>30-year Normal</b>
<b>May</b>	5.8	7.2	1.9	9.0
<b>June</b>	16.0	10.2	3.8	8.2
<b>July</b>	16.7	7.7	9.3	7.5
<b>August</b>	13.9	9.2	2.8	8.5
<b>September</b>	1.3	9.6	6.8	7.7
<b>October</b>	2.8	6.1	0.6	6.1
<b>Annual</b>	103	100	54.3	95.7
<b>Lenoir <sup>b</sup></b>	<b>2005</b>	<b>2006</b>	<b>2007</b>	<b>30-year Normal</b>
<b>May</b>	9.6	3.1	1.5	11.9
<b>June</b>	21.8	9.2	8.4	11.3
<b>July</b>	16.1	10.4	9.6	11.2
<b>August</b>	15.0	12.5	6.0	9.8
<b>September</b>	0.4	13.8	7.4	11.3
<b>October</b>	13.0	10.5	0.0	9.2
<b>Annual</b>	123	93.6	80.0	125

a: National Weather Service Coop Station # 310301 (35°35'43"N, 82°33'24"W)

b: National Weather Service Coop Station # 314938 (35°54'42"N, 81°32'2"W)

Source: NC Climate Office, 2008



**Figure 4.14** Photograph of stormwater wetland in June (left) and wet pond in July (right) during the drought of 2007

Due to lower water depths in the stormwater wetland and wet pond during the 2007 monitoring period, it was anticipated that water temperatures at the bottom depths of these systems would be warmer. Increased temperatures were expected due to the limited thermal buffering capacity of the reduced water depths. According to a Wilcoxon rank sum test, there was no significant ( $p < 0.05$ ) difference between the 2006 and 2007 monitoring periods for water temperatures at a depth of 60 cm or 90 cm at the stormwater wetland. Similarly, there was no significant ( $p < 0.05$ ) difference between the 2006 and 2007 monitoring periods for water temperatures at a depth of 80 cm or 120 cm at the wet pond. These analyses indicates that a modified outlet structure that draws from the bottom depths of these systems would be beneficial, even during times of substantial drought.

### **References**

- National Institutes of Health. (2008). "ImageJ: Image Processing and Analysis in Java." <<http://rsbweb.nih.gov/ij/>> (Sep 23, 2008).
- North Carolina Climate Office. (2008). "NC Climate Retrieval and Observations Network Of the Southeast Database." <<http://www.nc-climate.ncsu.edu/cronos>> (Sep 22, 2008).
- Wilcoxon, F. (1945). "Individual Comparisons by Ranking Methods." *Biometrics*, 1(6), 80-83.

## **Chapter 5**

### **Bioretention Thermal Model: Background and Development**

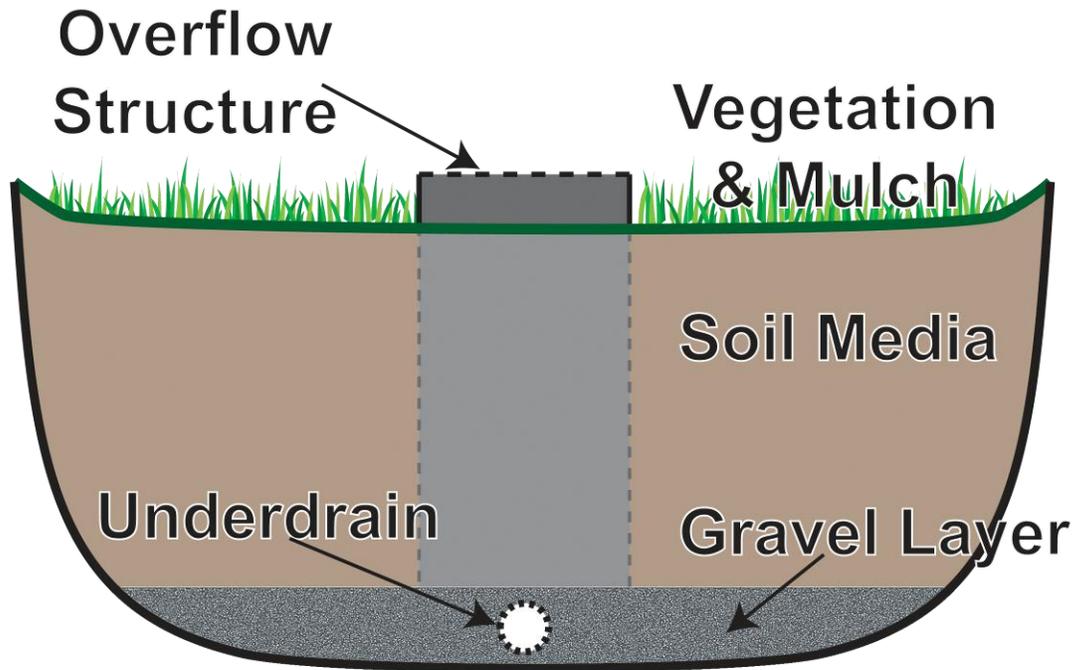
## **Introduction**

### **Assessment of Need**

Although the negative effects of thermal discharges on coldwater stream environments have been well documented, there are few mechanisms available for reducing the thermal impacts of heated urban stormwater runoff. Monitoring results have indicated that bioretention cells can substantially reduce runoff temperatures and volumes; however, specific reductions appear to be based upon a number of design parameters. Due to the numerous potential design configurations, it is impractical to make conclusive design recommendations based upon monitoring results alone. Consequently, a computer model that simulates bioretention temperature and flow would serve as a valuable tool for designers, allowing them to experiment with numerous design configurations for any specific location. Ideally, the ability to simulate the thermal behavior of a bioretention configuration should allow designers to determine the optimum bioretention design parameters for reducing the thermal impacts of urban stormwater runoff on coldwater stream environments.

### **Bioretention Overview**

Bioretention areas; also known as bioretention cells, rain gardens, and bioinfiltration basins; are a popular BMP in North Carolina due to their ability to meet stormwater and landscape requirements. Bioretention is often classified as a low impact development (LID) practice because it facilitates infiltration of stormwater runoff into the shallow groundwater, mimicking pre-development hydrology (Dietz, 2007). Bioretention cells function by allowing stormwater runoff to flow into a shallow basin and infiltrate through an engineered soil media. A bioretention cell typically consists of an engineered soil media, gravel drainage layer, underdrain pipes, overflow structure, vegetation, and mulch (Figure 5.1). In locations where the hydraulic conductivity of the subsoil is substantial, underdrains can be omitted to encourage groundwater recharge.



**Figure 5.1** Typical bioretention cross-section

The soil media constitutes the bulk of the system and is where most of the treatment takes place. Bioretention cells typically contain an engineered soil media. These engineered soils are predominantly sand with some fraction of fine and organic particles. The soil mix recommended for use in North Carolina consists of 85-88% sand, 3-5% organic matter, and 8-12% fine particles (Hunt and Lord, 2006; North Carolina Division of Water Quality, 2007). The sand fraction of the soil media is necessary to maintain adequate infiltration rates, while the organic matter supports plant and microbial growth. Bioretention cells are designed to drain within 24 to 48 hours after a storm event in order to make storage available for the next storm event and prevent the development of mosquitoes and other undesirable organisms. The nutrient content of the soil media, particularly phosphorus content, is also specified for the purpose of pollutant removal, but is not relevant to the thermal behavior of the system.

The overflow structure is designed to bypass runoff from larger storm events, such as those that exceed the first flush depth. The first flush in North Carolina is typically between 2.5 and 3.0 cm. The overflow structure often consists of an iron grate installed over a concrete box that extends to the bottom of the bioretention cell (Figure 5.2). The underdrain conveys water that has infiltrated through the soil media, but does not infiltrate into the bioretention subsoil. Underdrains typically consist of a section of perforated PVC pipe. Underdrains are often connected to the concrete box of the overflow structure. A gravel layer encases the underdrain to convey water to the individual drain pipes. This gravel layer often consists of washed 57 stone (North Carolina Division of Water Quality, 2007). Overflow and water from the underdrains typically leave the system via the same conveyance mechanism, discharging into the nearest drainage way. The surface of a bioretention cell is typically mulched and vegetated. Hardwood mulch is often applied to the soil surface in order to prevent the growth of invasive plants. Vegetation coverage is widely variable and can include shrubs, trees, and other plants.



**Figure 5.2** Overflow structure at Asheville bioretention recently after construction

## **Literature Review**

### **Modelling Soil Water Movement**

Modelling the flow of water through soil systems is often divided into separate approaches for saturated and unsaturated conditions. Darcy's law is commonly used to describe the flow of water through a soil system under saturated conditions (Jury and Horton, 2004). Darcy's law states that the flow of liquid through porous media is directly proportional to the hydraulic gradient and the hydraulic conductivity of the soil. The hydraulic conductivity is a soil property that can be measured in the laboratory using standard procedures. Because hydraulic conductivity is a function of soil water content, the property is often measured under saturated conditions. For bioretention, Darcy's law would apply when the wetting front reaches the gravel drainage layer. The Richards equation is generally considered to be the most accurate equation for describing unsaturated flow through porous media (Richards, 1931). Because the Richards equation is a non-linear partial differential equation that requires a numerical solution, it is not commonly used due to the detailed input and calculation requirements.

A frequently used approach for modelling the infiltration of water into an unsaturated soil profile was developed by Green and Ampt (1911). The Green-Ampt approach assumes that there is a well defined wetting front during the infiltration process. The Green-Ampt model accounts for the effect of the suction head in advance of the wetting front on the infiltrating water, effectively employing a modified form of Darcy's law. As the wetting front moves away from the surface, the effect of suction pressure on the advancing wetting front diminishes and the infiltration rate approaches the saturated hydraulic conductivity of the soil. The modelling approach originally developed by Green and Ampt has been modified numerous times to account for different situations encountered when describing infiltration processes. When modelling infiltration for a bioretention area, the Green-Ampt model would apply to the initial infiltration process, as the wetting front migrates towards the gravel drainage layer.

Because the original Green-Ampt approach assumed a uniform soil profile, modifications are necessary to account for layered soils. Bouwer (1969) used a tabular calculation approach to account for changes in water content and hydraulic conductivity with depth. Childs and Bybordi (1969) developed an infiltration modelling approach for stratified soils based on the work of Green and Ampt by accounting for the saturated hydraulic conductivity and suction pressure of each soil layer. This approach was only deemed appropriate for situations where hydraulic conductivity of successive layers decreased with depth, since water flowing into a layer of increased conductivity would violate the assumptions of a well defined wetting front progression for the original Green-Ampt model. Selker et al. (1999) developed a Green-Ampt model for stratified soils that focused on the variability in pore sizes. Because bioretention areas are engineered systems, a uniform soil profile is often a reasonable assumption; however, there is the possibility of soil stratification near the surface due to the incorporation of organic matter to support plant growth or sediment captured by the system.

Modelling the movement of water from a fine textured layer with low hydraulic conductivity into a coarse soil layer with a higher hydraulic conductivity is difficult because assumptions about wetting front progression from the original Green-Ampt approach do not apply. This phenomenon, commonly referred to as fingering, relates to the formation of preferential flow paths when water flows into a layer with increased hydraulic conductivity. Hillel and Baker (1988) proposed that this behavior is due to the flow path constriction necessary to satisfy the conservation of mass where the flow accelerates. The formation of preferential flow paths is also related to the pressure buildup required in the upper layer to overcome the lack of suction pressure in the lower, coarser material. Ahuja (1983) developed a modified Green-Ampt approach to account for a restrictive layer at the soil surface by introducing a time dependent pressure head at the layer interface and a profile shape correction factor.

Another difficulty in modelling infiltration is encountered when addressing a limited surface water supply or unsteady rainfall, which is frequently associated with natural storm events. Most infiltration models assume that there is an excess water supply at the soil surface and infiltration is limited by the ability of the wetting front to progress through the soil profile. Mein and Larson (1973) developed a two-stage model that accounts for situations where infiltration capacity exceeds rainfall intensity. The model assumed that rainfall was steady. The first stage of the model corresponds to the time during a storm event when the infiltration capacity exceeds the rainfall rate. During this phase, infiltration into the soil profile was assumed to occur at a rate equal to the rainfall rate. The duration of the first phase was determined by calculating the cumulative infiltration volume when the infiltration capacity predicted by the Green-Ampt model was equal to the rainfall intensity. The second phase relates to the period when the rainfall rate exceeds the infiltration capacity of the soil. During this phase, the infiltration rate was restricted to the infiltration capacity calculated by the Green-Ampt model and water ponds at the surface. The researchers found good agreement between their model predictions and those generated by a numerical solution to the Richards equation for a variety of soil types.

Chu (1978) built upon the work of Mein and Larson (1973) by developing an infiltration model to address unsteady rainfall, where the ponding status at the soil surface may alternate numerous times. Chu addressed unsteady rainfall by calculating the time that ponding occurs and a shift in time scale to account for the effect of cumulative infiltration at the ponding time. These parameters were calculated each time surface ponding occurred in simulations. Indicator variables were used to evaluate the ponding status at the soil surface. Chu's approach assumed that the soil profile was homogeneous and the antecedent soil moisture content was uniform. Chu compared the results of the infiltration model on a watershed scale to runoff volume measurements collected by the Agricultural

Research Service for three storm events. The rainfall excess calculated by the infiltration model agreed with runoff measurements for 2 of the 3 storms analyzed.

Chu and Mariño (2005) subsequently improved upon the work of Chu (1978) by modelling infiltration of unsteady rainfall into a layered soil profile. Chu and Mariño accounted for layered soil profiles by assuming an instantaneous hydraulic equilibrium at the interface of soil layers. Although there is likely a region near the soil layer interface where water flow adjusts to the new hydraulic conditions and wetting front becomes distorted, the researchers felt this transition region was small and could be neglected based on the Green-Ampt assumption of piston flow. The soil profile was discretized to account for changes in water content, suction head, and fillable porosity. The time domain of model calculations was also discretized to account for unsteady rainfall conditions. When the surface was not ponded, the potential progression of the wetting front was calculated based on the rainfall intensity and soil properties used in infiltration capacity calculations were updated to account for this progression. If the rainfall rate exceeded the infiltration capacity of the soil, surface ponding was assumed to occur and the specific time of ponding and location of the wetting front at that time were calculated. The equivalent time required for the wetting front to reach that depth under an initially ponded condition was also calculated and subsequently used for calculations during the ponded phase. The infiltration rate and location of the wetting front under ponded conditions were calculated using the Newton-Raphson iteration method in conjunction with a modified Green-Ampt equation. To validate their infiltration model, Chu and Mariño compared model results to other infiltration models and monitoring data. Chu and Mariño's modelling results showed good agreement for the cases of a homogeneous soil with unsteady rainfall, layered soil with steady rainfall, and layered soil with unsteady rainfall.

In recent years, modelling the infiltration process for a bioretention area has received interest as researchers seek tools to evaluate bioretention hydrology.

Heasom et al. (2006) developed a hydrologic model for bioretention utilizing HEC-HMS. A Green-Ampt approach was used within HEC-HMS to model infiltration into the bioretention soil, assuming an excess water supply at the soil surface for the duration of the event. A layer of decreased hydraulic conductivity was added to the top of the simulated soil profile to account for clogging by fine particles.

A numerical model based on the Richards equation, known as RECHARGE, has been developed to evaluate bioretention hydrology (Dussailant et al., 2004). The bioretention soil profile was divided into 3 layers corresponding to a root zone, storage zone, and subsoil. All water entering the bioretention area was assumed to be lost to evaporation or seepage into the subsoil, with no drain pipes considered. The model used a water balance at the rain garden surface to account for direct rainfall, inflow, infiltration, and runoff. Inflow to the rain garden was calculated based on a simple abstraction due to depression storage. The numerical solution of the Richards equation was based upon a Crank-Nicholson finite difference scheme. The model was validated using published literature results pertinent to bioretention and showed good agreement. Simulations showed that maximum groundwater recharge was attained when the area of the rain garden was between 10 and 20% of the contributing watershed, with evaporation having a greater impact at bioretention areas larger than 20% (Dussailant et al., 2004).

Another bioretention hydraulic model, known as RECARGA, was developed based upon the Green-Ampt equation (Dussailant et al., 2003; Dussailant et al., 2005). The Green-Ampt model was used because it required fewer soil inputs and less demanding computations than the Richards equation model, making the model more accessible. The suction pressure ahead of the wetting front was approximated based on the bubbling pressure of the soil using an equation presented by Brakensiek and Onstad (1977). The model used a water balance at the rain garden surface and inflow calculation similar to the RECHARGE model. The rain garden profile was divided into 3 homogeneous soil layers. Drainage between each layer

was approximated as the unsaturated hydraulic conductivity estimated by the van Genuchten relationship (van Genuchten, 1980). Analysis showed that results of the RECARGA simulations were generally similar to those from the more complex RECHARGE model, with most differences associated with simulations for low saturated hydraulic conductivities of the soil underlying the bioretention area.

### **Modelling Runoff Temperatures**

In recent years, numerous research efforts have focused on modelling pavement temperatures. Most of these models have sought pavement temperature predictions for use in structural simulations of flexible pavements like asphalt. Because the details of temperature fluctuations do not play a major role in pavement structural evaluations, many pavement temperature models have focused on predicting annual minimum and maximum temperatures without directly accounting for daily and seasonal temperature fluctuations (Barber, 1957). Diefenderfer et al. (2006) developed a pair of linear regression models to predict daily minimum and maximum pavement temperature profiles. The regression models were based upon pavement temperature monitoring data collected in Virginia and adapted for other locations using maximum and minimum ambient air temperatures and solar radiation.

James and Verspagen (1997) developed a regression model to predict runoff temperatures based on measurements collected at asphalt test plots for a range of rainfall intensities. Inputs required by the regression model were rainfall intensity, thermal conductivity of the surface, initial runoff temperature, and initial rainfall temperature. Although runoff temperature predictions generally agreed with test plot measurements, the researchers found a substantially larger margin of error during the first 10 minutes of a rainfall event. The model generally underestimated runoff temperatures during the early portion of a storm.

Van Buren et al. (2000) developed a model to predict both pavement and runoff temperatures to evaluate the thermal impacts of urban stormwater (Van Buren et al., 2000). Heat transfer was modeled within the asphalt using an explicit finite difference solution of a transient, one-dimensional heat equation. To calculate the temperature at the surface node, the author assumed negligible internal resistance to heat transfer and equated change in internal energy to the heat flux from convection, radiation, and evaporation. Runoff temperature was assumed to be the average of the pavement surface temperature and rainfall temperature. Separate heat balances were used for wet and dry conditions. Convection and radiation heat fluxes were considered under dry conditions, while convection, radiation, and evaporation heat fluxes were taken into account for wet conditions. The latent heat of vaporization, evaporation rate, and sensible heat transfer were calculated using procedures published in the literature. During dry weather simulations, the convective heat transfer coefficient was estimated based on wind speed with an empirical equation presented by Barber (1957). During wet weather simulations, the convective heat transfer coefficient was approximated based on rainfall intensity using an empirical relationship developed by the author from monitoring data collected at asphalt test plots in Ontario, Canada.

### **Modelling Air Temperature**

In order to calculate a surface heat balance for a pavement or soil surface, it is often necessary to have knowledge of the ambient air temperature. While an increasing number of weather stations record hourly air temperatures, these measurements are not always widely available or easily obtained, making it valuable to predict air temperature based on limited measurements. There are several models that have been produced to predict diurnal temperature patterns based on daily minimum and maximum air temperatures. Reicosky et al. (1989) evaluated 5 diurnal air temperature models that used daily maximum and minimum air temperatures as their primary inputs and found that all of the evaluated models provided reasonable air temperature estimates. Sadler and Schroll (1997)

developed an air temperature model where the diurnal pattern was not restricted to a predefined sine wave or similar curve. They utilized several temperature normalization procedures in conjunction with a beta distribution to predict air temperatures.

Satyamurty and Sarath Babu (1999) developed an ambient air temperature model based on hourly measurements collected over a period of 21 to 30 years at 30 Indian locations and 55 locations in the United States. The air temperature model utilized daily maximum and minimum air temperatures as inputs. Separate equations were used to describe the temperature before and after 17:00 each day. The maximum air temperature was assumed to occur at 14:30, while the minimum air temperature was assumed to take place at 6:30. The root mean squared error of temperature predictions was determined to be 0.27%, with an absolute error of 0.79°C.

### **Modelling Solar Radiation**

Solar radiation is another component of a surface energy balance that must be considered in a number of pavement and soil temperature models. Because detailed solar radiation measurements are not widely available, a number of models have been developed to predict solar radiation based upon predictable astronomical patterns and common meteorological measurements. Campbell and Norman (1998) presented an empirical model to predict solar radiation based primarily upon the desired time and location. Spokas and Forcella (2006) expanded upon the model presented by Campbell and Norman by including a decision matrix to estimate the atmospheric transmissivity based upon the occurrence and time of precipitation.

### **Modelling Soil Temperature Profiles**

A number of models have been developed to simulate soil temperature profiles over time. Knowledge of soil temperature is beneficial to the understanding of water movement, plant growth, chemical transformations, microbial activity, and

other processes. Most soil temperature models account for both diurnal and seasonal temperature changes throughout the soil profile.

Van Wijk and de Vries (1966) developed a soil temperature model based on the sum of two sinusoids, accounting for daily and seasonal temperature fluctuations. The researchers solved the one-dimensional heat equation analytically with the upper boundary described by the sinusoidal relationships and the lower boundary set at a constant temperature. The model did not account for the effects of individual weather patterns on soil temperatures.

Elias et al. (2004) expanded upon the efforts of van Wijk and de Vries (1966) by including a time-dependent daily amplitude term to better account for seasonal fluctuations. The model was calibrated with soil temperature measurements collected over a 10 year period at a location in Australia and two locations in Brazil. It was determined that introduction of the time-dependent daily amplitude term improved soil temperature estimates.

A number of soil temperature models have taken a more detailed approach by using a surface heat balance to calculate the upper boundary condition instead of an empirical relationship (Chung and Horton, 1987; Mahrer, 1979; Ochsner et al., 2007). Timlin et al. (2002) analyzed the effect of errors in estimation of hourly solar radiation and air temperature values on soil temperature profiles produced by a two-dimensional finite element model. Simulated soil temperatures using estimated weather data instead of measured weather data were cooler by about 2°C over all soil depths.

Wierenga and de Wit (1970) presented a model that predicted subsoil temperatures based on the temperature at the soil surface and the apparent thermal diffusivity of the soil. The effects of temperature and soil moisture content on the apparent thermal diffusivity were also considered. The researchers determined that

vapor movement within a partially saturated soil could have a substantial impact on the apparent thermal diffusivity of the soil.

### **Soil Temperature during Infiltration**

As the thermal dynamics of dry soil systems have been analyzed, some researchers have examined the effect of infiltration and water redistribution on soil temperature profiles. Modelling heat transfer during infiltration and redistribution of water can be fairly complex because changes in the soil thermal properties that relate to water content must be accounted for in addition to heat transfer via conduction and convection. Nonetheless, modelling heat transfer during infiltration has important implications for the understanding of thermal pollution, ground water movement, and better describing soil temperature profiles by accounting for the effects of individual weather patterns.

Research by de Vries (1958) focused on the heat transfer due to moisture fluxes within a soil profile. Heat transfer mechanisms accounted for by de Vries included heat conduction, latent heat transfer by vapor movement, and sensible heat transfer via liquid and vapor phases. This approach assumed that heat transfer via convection was negligible. While de Vries' approach has been incorporated into a number of subsequent models, it primarily pertains to internal redistribution of moisture due to thermal or moisture gradients.

Wierenga et al. (1970) examined the effect of irrigation with warm and cool water on soil temperature profiles. Temperature differences between soil profiles irrigated with warm and cool water persisted at a depth of 50 cm for more than 5 days, illustrating the potential extended impact of stormwater runoff on bioretention soil temperatures. Wierenga et al. calculated the impact of irrigation on soil temperatures assuming that mass movement of water was the only heat transfer mechanism. Soil temperature profiles were computed based on the soil and water heat capacity and measured temperatures before and after thermal equilibrium for

discretized soil layers. Measurements after irrigation showed that the thermal impact of evaporation on the soil profile was substantial, with the maximum irrigated soil surface temperature more than 20°C cooler than non-irrigated soil surface during the day after irrigation. Wierenga et al. demonstrated that modelling heat transfer during infiltration by only accounting for conduction was not adequate.

Shao et al. (1998) developed a one-equation model that accounts for conduction and convection when water is moving through a soil profile. The model assumed a ponded surface condition for the duration of the simulation and predicted surface temperatures based on a sine function, assuming the soil and infiltrating water were locally in thermal equilibrium. The temperature predictions of their analytical model were generally within 2°C of values measured in the field. The researchers also noted that the hydraulic conductivity of the soil changed in response to temperature oscillations, corresponding to changes in fluid viscosity.

Historically, soil infiltration models have assumed that the process is effectively isothermal. Research has shown that the process of wetting soil can result in temperature changes, especially for very dry, clayey soils (Prunty and Bell, 2005). These temperature changes are largely associated with latent heat exchanges in the vicinity of the wetting front and are commonly referred to as the heat of wetting or heat of immersion. Despite substantial impacts in some cases, Collis-George and Lal (1973) found that the heat of wetting could be ignored for soils with moisture contents near the permanent wilting point. Due to the complex thermal interactions associated with wetting a soil particle, modelling these heat exchanges can be relatively difficult and the overall effect of these temperature changes on soil and water temperatures during the overall infiltration process are likely minimal for bioretention areas due to the relatively large soil particle size and moist soil conditions.

A number of one-equation models have been developed to describe thermal dispersion in porous media (Moyné et al., 2000; Shao et al., 1998). These one-equation models generally assume that the solid and liquid phases are locally in thermal equilibrium. The assumption of local thermal equilibrium is advantageous because heat transport in the solid and liquid phases can be combined into one equation that uses an effective thermal conductivity. The assumption of local thermal equilibrium is invalid for transient heat conduction in porous media (Hsu, 1999; Kaviany, 1991; Quintard and Whitaker, 1995). While the assumption of local thermal equilibrium is prevalent among soil infiltration models, models for heat transfer in packed bed reactors and similar porous media systems have considered situations where thermal equilibrium assumptions do not apply.

Hsu (1999) derived a set of governing equations for transient heat conduction in porous media under non-thermal equilibrium conditions. Modelling non-thermal equilibrium conditions involves separate equations to describe heat transport in the solid and liquid phases and requires some type of closure modelling to reduce the number of unknown parameters in these equations. Quintard and Whitaker (1993) described a closure modelling scheme for heat transport in porous media based upon the concept of volume averaging. Hsu utilized a volume averaging scheme similar to Quintard and Whitaker, but modeled the tortuosity effect analytically and assumed quasi-steady conditions for heat conduction across the interfacial areas. The closure modelling scheme presented by Hsu (1999) was advantageous because it resulted in a more concise set of equations than previous efforts, reducing the number of coefficients that must be estimated.

Nakayama et al. (2001) developed a two equation model to account for both conduction and convection in a porous media without local thermal equilibrium conditions. Their approach was based on the closure model of Hsu (1999), but accounted for convection in addition to conduction terms. Parameters included in the model that must be determined include the interfacial convective heat transfer

coefficient, thermal dispersion tensor, and effective thermal conductivity of the saturated stagnant porous medium. Wakao and Kaguei (1982) provide empirical estimations of the interfacial convective heat transfer coefficient and thermal dispersion tensor based primarily on the fluid velocity and physical properties of the fluid and solid. Hsu et al. (1995) developed an algebraic expression for the stagnant thermal conductivities based upon the media porosity and thermal conductivity of the fluid and solid phases by assuming theoretical particle geometries. Nakayama et al. (2001) used the governing equations they presented to develop an analytical model for steady conditions. The researchers noted that validation of a two-phase, two-equation model is difficult because the temperature of the fluid and solid phases must be measured independently.

### **Estimating Soil Thermal Properties**

Due to limited applications, soil thermal properties are not typically available from routine soil analyses. Probes have been developed to measure soil thermal conductivity and specific heat; however, the availability of these probes is somewhat limited (Bristow, 1998; Bristow et al. 2001; Campbell et al. 1991). Thermal properties of the solid fraction of soil media have been well researched. De Vries (1966) reviewed specific heat and thermal conductivity measurements from the literature to determine an average value for both the mineral and organic matter components. Usowicz et al. (2006) showed that the thermal conductivity of quartz could be described as a function of temperature. Campbell and Norman (1998) presented specific heat and thermal conductivity values representative of general soil minerals. Ochsner et al. (2001) estimated thermal conductivities of soil solids by fitting measured results to de Vries model. Ochsner et al. calculated soil heat capacities based on heat pulse measurements and a procedure described by Campbell et al. (1991) that accounted for the heat capacity of each soil constituent.

While the thermal conductivity and specific heat of soil minerals can be estimated with reasonable accuracy based upon measurements in the literature, the

thermal properties of the entire soil matrix are widely variable due to differences in mineral composition, water content, bulk density, temperature, and porosity. De Vries (1966) provided a method for estimating the specific heat capacity of a soil based upon the volume fraction of soil minerals, organic matter, and water and the measured specific heat of each component. Abu-Hamdeh (2003) compared measured soil thermal properties with those predicted by equations developed by de Vries (1966) and Bristow (1998) and found good agreement between laboratory measurements and predictions. Usowicz et al. (2006) developed a regression model that predicted soil thermal conductivity based upon penetration resistance and air-filled porosity with reasonable accuracy. Becker et al. (1992) used a set of 4 correlation coefficients to predict thermal conductivity for several types of soil, based upon measured conductivities published in the literature. Lu et al. (2007) presented a model that predicted soil thermal conductivity based upon bulk density, quartz fraction, and water content, finding good agreement between measured values and model predictions over a wide range of soil textures, including those associated with bioretention areas. Hsu et al. (1995) developed a more general model to predict the stagnant thermal conductivity of porous media with an analytical model based on ideal particle geometries.

### **Model Methodology**

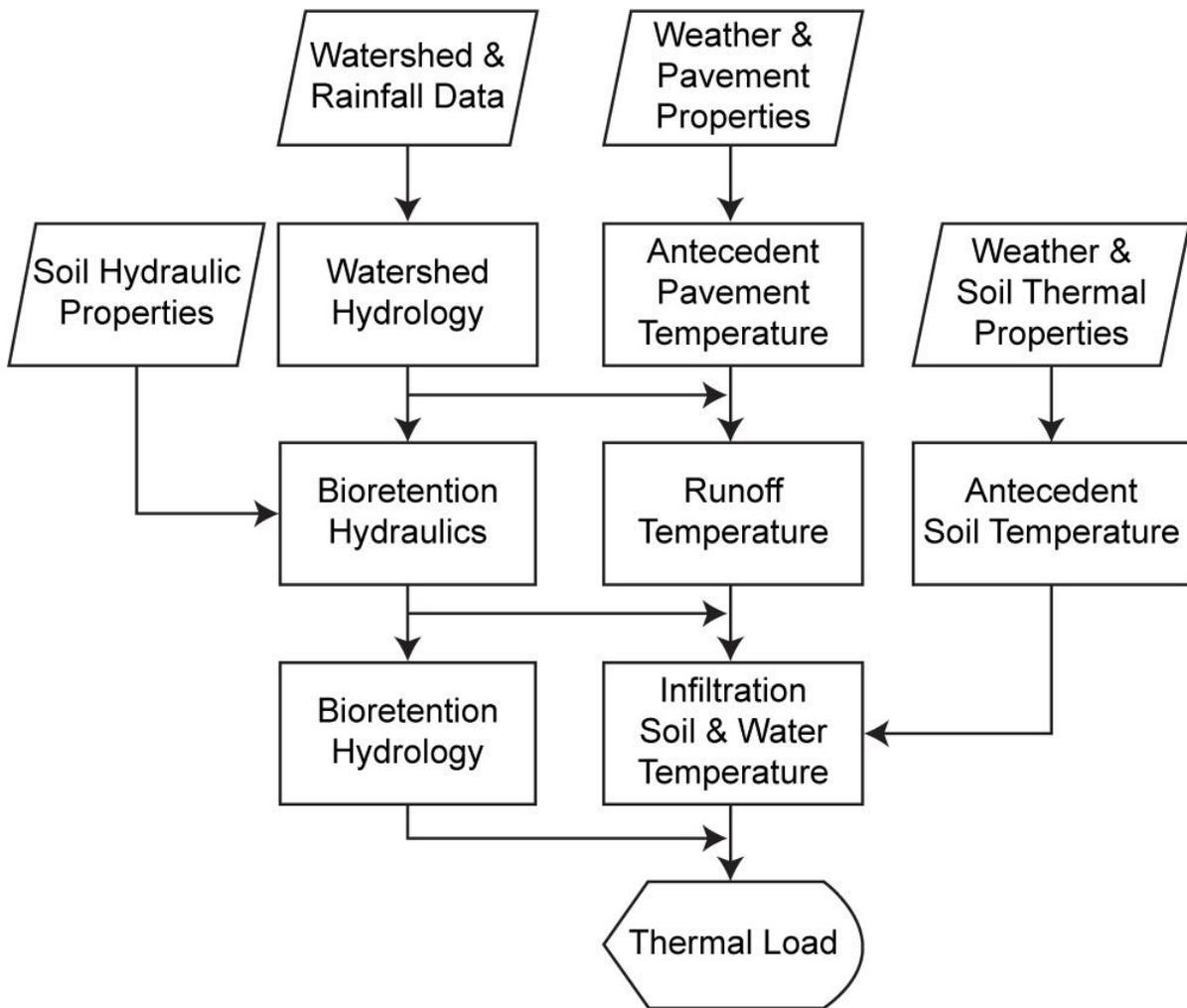
#### **Programming Language**

Visual Basic 2008, also known as Visual Basic 9.0, was used for all model development. Visual Basic 2008 is an object-oriented programming language and operates on the Microsoft .NET Framework. The program code was developed using the Visual Basic 2008 Express Edition editor and compiler, which was available as a free download from Microsoft. The free distribution of the development software, relative simplicity of the programming language, and wide range of instructional and support materials available was expected to make it easier for future researchers to modify or expand upon the bioretention thermal model code.

There were some disadvantages associated with using Visual Basic 2008 for the bioretention thermal model. Visual Basic 2008 was built around the Microsoft .NET framework, which was only compatible with Microsoft Windows operating systems. The .NET framework must also be installed on the computer where the model would be used. The .NET framework was often distributed via routine Windows updates, but may require manual installation in some instances. It was also possible that the use of other programming languages may result in shorter computation times; however, anticipated differences were not large enough to justify use of another programming language.

### **Model Procedure Overview**

The calculation of soil and water temperature profiles during infiltration was preceded by three main branches of calculations, corresponding to system hydraulics, influent temperature, and initial soil temperature profile (Figure 5.3). Each model component was calculated separately for the duration of the simulation period, with data typically passed between model components using arrays.



**Figure 5.3** Flow chart of overall model procedure

### Rainfall Input File

Stormwater models can be designed to analyze single storm events or long-term weather data. Early stormwater models were almost entirely storm based, due to their relative simplicity and limited computational requirements. In a storm based model, computations are performed for an individual storm event, often a synthetic design storm. Design storms are often based upon reference hyetographs published by the United States Department of Agriculture Natural Resources Conservation Service (USDA-NRCS) for different regions of the United States (Kent, 1973). The NRCS design storms were originally developed for agricultural

watersheds, but are commonly considered standard engineering practice for urban stormwater designs, despite their age and questionable applicability. As computational power has increased, more emphasis has been placed on developing continuous models that analyze long-term weather data. Although more computationally intensive, continuous models generally provide a better approximation of how a system will actually perform over a long period of time because they are based on actual weather data and account for weather variability.

The bioretention thermal model was originally intended to be a continuous model, accounting for seasonal weather changes and the variety of storm characteristics a site may encounter. Due to the computational requirements of the model, it became necessary to employ a storm based approach in order to maintain reasonable execution times. Although a storm based approach was utilized, it was possible to run model simulations for both synthetic and historical rainfall data. Analyzing historical storm events provides an indication of how the system might actually perform, while design-based storms can be utilized to maintain consistency and satisfy regulatory requirements. Because the model was originally intended to be a continuous model, the code was structured such that simulations could be conducted on a continuous basis with minimal additional programming in the future.

Rainfall input files were loaded in the form of comma separated variable files. This format allowed for complete customization of the rainfall data (Appendix C). Each row consisted of a date and time, and the depth of rainfall accumulated between the current time and the previous record. Rainfall data could be based upon a design storm, tipping bucket data, or other rainfall measurements. The model did not require a constant time step for rainfall data, since rainfall data were disaggregated to meet the required time step for calculations. The model did require that rainfall was continuous during a storm event. This requirement was necessary because the model assumed that the soil profile was completely drained before a storm occurred and did not account for the latent heat exchanges associated with

evaporation from the soil after a storm event. The time and duration of the storm were extracted from the rainfall input file and used throughout the model to load or estimate other weather parameters. A sample rainfall input file can be found in Appendix C.

### **Watershed Hydrology**

Due to the model's exclusive focus on impermeable surfaces, a simple approach was utilized to evaluate watershed hydrology. At the beginning of a storm event, rainfall was first used to satisfy an initial abstraction depth. The initial abstraction depth was a function of the surface roughness and was used as an input. After the initial abstraction was satisfied, all remaining rainfall was treated as direct runoff. The simplified watershed modelling approach was similar to the one utilized by Dussailant et al. (2004); however, a discrete initial abstraction was used instead of assuming a percentage of rainfall was lost to depressional storage. Factors such as watershed slope and shape were not considered because the model was intended for bioretention cells that receive runoff directly from small urban catchments. The scope of the model did not include large watersheds or situations where water must be conveyed from the impermeable surface to the bioretention by a pipe, swale, or other mechanism because it would be impractical to account for all of the thermal effects associated with those factors. Similarly, the model was intended for use only in situations where the watershed was entirely impervious because it was difficult to anticipate the thermal load derived from the variety of permeable surfaces that may be encountered. The time step for the bioretention inflow volume calculations was equivalent to the time step used for Green-Ampt simulations. Watershed area, initial abstraction depth, and bioretention surface area were all input by the model user.

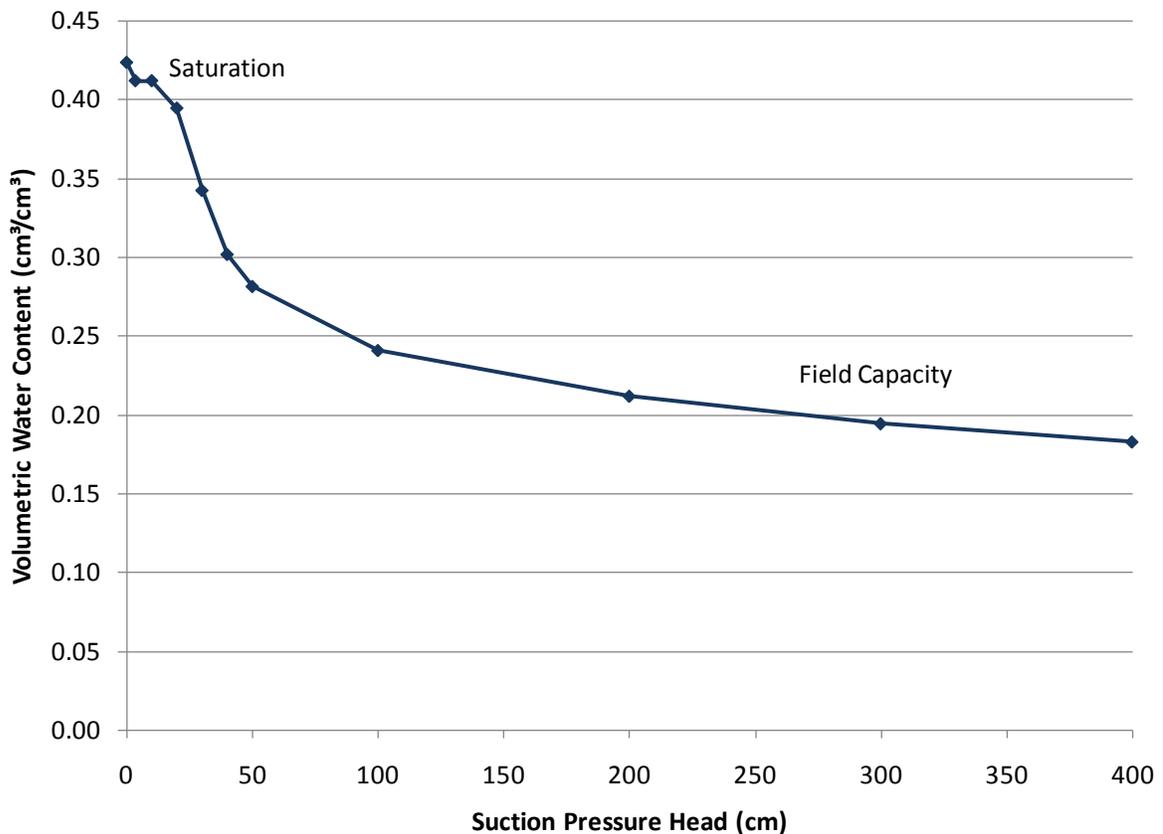
### **Bioretention Hydraulics / Hydrology**

A modified Green-Ampt approach presented by Chu and Mariño (2005) was used to simulate bioretention hydraulics during unsaturated flow (Appendix A). This

approach was selected because it accounted for unsteady rainfall and layered soil profiles, which are both conditions relevant to modelling infiltration for bioretention areas. The Green-Ampt approach presented by Chu and Mariño was intended for rainfall over a soil surface. To make the algorithm applicable to bioretention, inflow into the bioretention cell was converted to an effective rainfall rate by dividing the inflow volume by the area of the bioretention cell. Using this approach had the same effect as increasing the rainfall intensity of a storm event and was not expected to have an impact on the accuracy of simulation results. It was assumed that the sides of the bioretention area were vertical. A time step of 1 second and grid spacing of 1 mm were used for all Green-Ampt calculations.

Soil properties considered in the Green-Ampt infiltration equations were saturated hydraulic conductivity, pressure head in advance of the wetting front, and fillable porosity. The saturated hydraulic conductivity could be measured in the laboratory for an intact soil sample using standard soil analysis methods. Saturated hydraulic conductivity is often measured in the laboratory by measuring the water flow rate required to maintain a static hydraulic head or measuring the flow of water leaving a soil sample subjected to a static hydraulic head. Pressure head in advance of the wetting front refers to the suction force exerted by the dry soil just below the wetting front. Because this pressure head is difficult to measure in practice, it is frequently approximated by the bubbling pressure of the intact soil sample, which has been shown to serve as a reasonable estimate (Morel-Seytoux and Khanji, 1974; Rubin and Steinhardt, 1964). The bubbling pressure is often easily identified on a soil water characteristic curve by noting the suction required to remove water from the soil sample. The fillable porosity is defined as the amount of water that could be stored within the soil during saturation. Fillable porosity can generally be estimated as the difference between the moisture content at the point of saturation and the moisture content at field capacity. Both of these parameters could be estimated based upon a soil water characteristic curve (Figure 5.4). Variation in the initial soil water content and the amount of air entrapped in the

saturated soil could lead to errors in the estimate of fillable porosity. Although the Green-Ampt algorithm presented by Chu and Mariño accounted for layered soil profiles, layered profiles were not fully implemented in the bioretention thermal model. Layered profiles were not considered in the present version due to the wide variety of potential user inputs and the unpredictable effect they may have on thermal calculations. The framework for accounting for layered soil profiles was included in the model code and may be fully implemented at a later date. Drain depth, surface storage capacity, gravel porosity, drain height, and subsoil saturated hydraulic conductivity were additional model inputs required from the user.



**Figure 5.4** Soil water characteristic curve for the Brevard east bioretention area with the point of saturation and field capacity identified

One disadvantage of using Chu and Mariño’s (2005) approach to model bioretention infiltration was that the hydraulic impact of water ponded at the soil

surface was not considered during the initial saturation of the soil profile. Because bioretention areas are typically designed to store up to 20 cm of water above the soil surface before overflowing, the hydraulic impact of surface ponding could be substantial. Warrick et al. (2005) presented a Green-Ampt model that accounts for variable ponding depths at the surface; however, this approach does not address the situation where infiltration is limited by rainfall intensity and was not used in the bioretention model as a result. The hydraulic impact of surface ponding was accounted for during saturated flow through the soil profile. Because surface ponding was most likely to occur during saturated flow, due to the lower infiltration rate, negating the hydraulic impact of surface ponding during the initial infiltration process was not expected to have a major impact on overall bioretention hydrology for most storms.

After the wetting front advanced to the depth of the gravel drainage layer, water movement through the soil column was calculated according to Darcy's law. The model contained code to account for layered soil properties by calculating an effective hydraulic conductivity for the soil profile (Jury and Horton, 2004). Differences in the approximations of the Green-Ampt method and Darcy's law may result in a discontinuity in the infiltration rate. Discontinuities were expected to be most evident for situations where there was substantial surface ponding at the time the soil profile becomes completely saturated, since the hydraulic impact was accounted for in Darcy's Law calculations, but neglected in the Green-Ampt approach.

Infiltration of water from the gravel layer into the underlying soil was calculated using Darcy's law. Although infiltration into the underlying soil likely occurs as unsaturated flow, Darcy's law was considered to serve as a conservative approximation, since calculated flows would be slower than unsaturated flows. The same Green-Ampt approach that was utilized for the bioretention soil could be applied to the underlying soil; however, information on underlying soil properties,

location of the water table, and the presence of confining layers were expected to be scarce and could have substantial impacts on infiltration rate errors. For the Green-Ampt approach to be accurately implemented for the subsoil, detailed knowledge of physical soil properties would be needed for all soil layers within at least 1 m of the base of the bioretention area.

After simulated runoff had ceased, drainage of the soil profile was calculated using Darcy's law. The water table was assumed to recede along a well defined front and the hydraulic gradient used in Darcy's law calculations was constantly updated at each time step. Soil above the water table was assumed to be drained to field capacity, neglecting the variation in suction pressure above the water table. Although this assumption has a major impact on the water distribution in the soil after a storm event, it was not expected to have a substantial influence on water flow rates or bioretention hydrology, which are more important for this modelling effort. Assumptions regarding water distribution within the soil would have the greatest impact at the very beginning of a storm event and during the relatively low flow rates as the soil profile drains after a storm. For most moderate storm events, such as those likely to be used when designing bioretention areas, these periods of flow would represent a small portion of the overall storm volume.

A surface water balance and soil water balance were calculated at each time step and the resultant volumes were stored in an array for later use in thermal load estimations. Similarly, the flux of water through the soil profile was stored in an array for use in evaluating the soil and water temperature profiles during infiltration.

### **Pavement and Runoff Temperatures**

Runoff temperatures were simulated using the algorithm presented by Van Buren et al. (2000) (Appendix A). Heat transfer through the pavement profile was calculated using a transient heat conduction approach:

$$T_m^{t+\Delta t} = T_m^t \left( 1 - \frac{2\alpha\Delta t}{\Delta z^2} \right) + \frac{\alpha\Delta t}{\Delta z^2} (T_{m+1}^t + T_{m-1}^t) \quad (1)$$

where  $T$  represents temperature within the pavement ( $^{\circ}\text{C}$ ),  $\alpha$  is the thermal diffusivity of the pavement material ( $\text{m}^2/\text{s}$ ),  $\Delta t$  is the calculation time step (s),  $z$  is depth below the pavement surface (m), the  $m$  subscript refers to a depth index, and the  $t$  superscript refers to a time index. The temperature at the pavement surface was calculated during dry weather conditions based on the following surface heat balance:

$$\left. \frac{\partial T}{\partial z} \right|_{z=0} = \frac{h_c}{k} (T_s - T_{air}) - \frac{q_{rad}}{k} \quad (2)$$

where  $k$  is thermal conductivity of the pavement ( $\text{W}/\text{m K}$ ),  $T_s$  is the pavement surface temperature ( $^{\circ}\text{C}$ ),  $T_{air}$  is the air temperature above the pavement surface ( $^{\circ}\text{C}$ ), and  $q_{rad}$  is the solar radiation heat flux ( $\text{W}/\text{m}^2$ ). During a storm event, temperature at the pavement surface was calculated based on the following surface heat balance:

$$\left. \frac{\partial T}{\partial z} \right|_{z=0} = \frac{h_c}{k} (T_s - T_{ro}) - \frac{q_{rad}}{k} + \frac{q_{evap}}{k} \quad (3)$$

where  $h_c$  is the convective heat transfer coefficient at the pavement surface ( $\text{W}/\text{m}^2 \text{K}$ ),  $T_{ro}$  is the runoff temperature ( $^{\circ}\text{C}$ ), and  $q_{evap}$  is the evaporation heat flux ( $\text{W}/\text{m}^2$ ).

The model simulated dry pavement conditions for 10 days before the storm event began to establish the pavement temperature profile before the storm event and reduce any bias of the initial temperature profile estimate. It was assumed that the length of time between the simulated storm event and any previous storm events was sufficiently long, such that the previous storm did not impact the pavement temperature profile. A time step of 15 seconds and grid spacing of 1 cm were used for both dry and wet weather pavement and runoff temperature calculations. To account for future improvements in runoff temperature modelling, the model was

configured to allow runoff temperatures to be loaded from an xml file for the duration of the storm event.

Weather data such as wind speed and air temperatures were loaded based upon monthly 30-year normal values obtained from the North Carolina State Climate Office. The air temperature at the current time in the simulation was estimated based upon the procedure presented by Satyamurty and Babu (1999) (Appendix A). This procedure required daily minimum and maximum air temperatures and the times when those temperatures occurred and was selected due to its relative simplicity and minimal input requirements. Because air temperatures were only used to calculate the initial pavement temperature profile, any errors associated with this simplified procedure were deemed reasonable.

Solar radiation was calculated using the procedure presented by Spokas and Forcella (2006) (Appendix A). The decision matrix described by the researchers was not fully implemented because daily weather data was not loaded into the model; however, this decision matrix could easily be implemented for continuous simulations in the future. The time of sunrise, solar noon, and sunset were calculated according to algorithms presented by Meeus (1998) and were used in calculating solar radiation values. These algorithms required date, time, latitude, longitude, and elevation as the primary inputs.

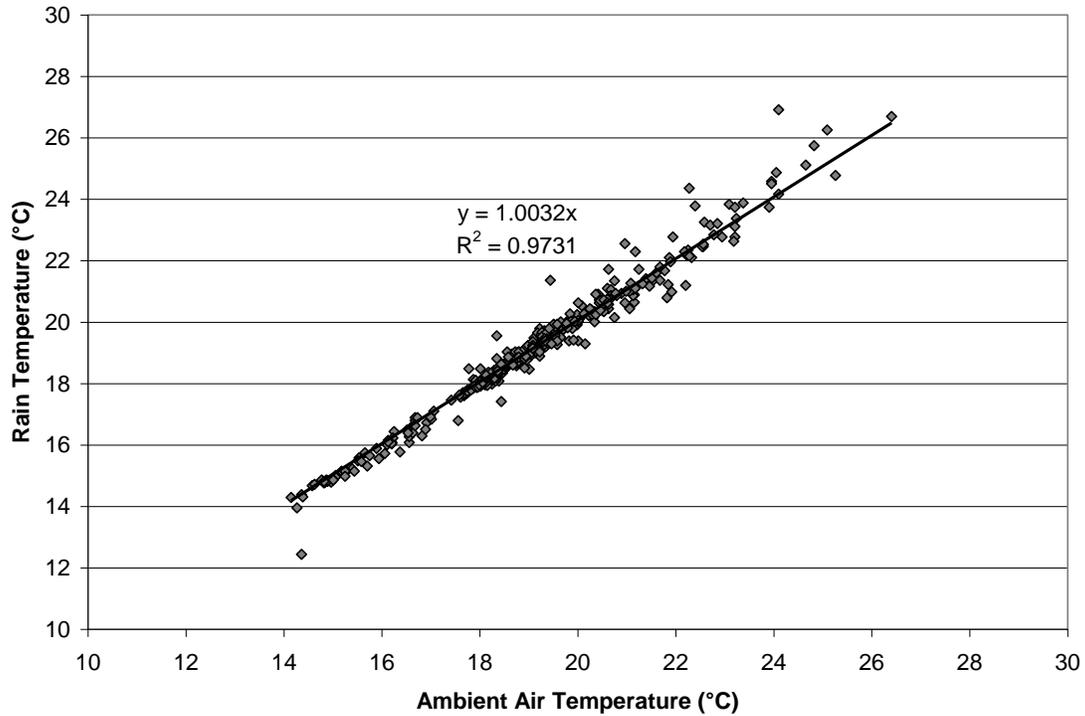
One of the complexities of modelling stormwater runoff temperatures was approximating weather parameters during a storm event. The methods used to predict daily and seasonal variations for parameters such as wind speed, air temperature, and solar radiation often do not apply to the dynamic changes observed during storm events. A number of factors well beyond the scope of this model could influence wind speed, air temperature, rainfall temperature, cloud cover, and other parameters that could affect pavement and runoff temperatures. Due to the potential variability of weather conditions during a storm event, air temperature,

wind speed, solar radiation, and relative humidity were required inputs from the model user and were not calculated using the previously described estimation techniques. When used for design purposes, it is possible that synthetic or standard weather parameters would be used in model simulations for consistency.

Based on monitoring data collected at the Asheville stormwater wetland location during the summer of 2007, rainfall temperature was assumed to be equivalent to air temperature. Rainfall was collected by the apparatus shown in Figure 5.5 and the water temperature was recorded by a TMCX-HD thermistor connected to a U-12 4-channel logger. Also connected to the 4-channel logger was a TMCX-HD thermistor installed within an RS1 radiation shield to measure ambient air temperature. The temperature sensors, logger, and radiation shield were all obtained from Onset Computer Corporation. The rainfall collection apparatus was designed such that water was flushed from the collection tube with every 0.2 mm of rainfall, minimizing the effect of stagnant water on the temperature measurements. The collection surface was also transparent to minimize the effects of solar radiation on the rainfall temperature measurements. Monitoring results showed that ambient air temperature was a good predictor of rainfall temperature (Figure 5.6).



**Figure 5.5** Collection apparatus for rainfall temperature measurements



**Fig 5.6** Measured rainfall temperatures vs. ambient air temperatures at the Asheville stormwater wetland

Estimation of pavement and runoff temperatures required information on depth, density, thermal conductivity, and specific heat of both the pavement and underlying gravel layer. The asphalt thickness was specified at 10 cm. This asphalt thickness was calculated using standard design procedures for a parking lot with 50-100 spaces on a moderate class subgrade with an untreated aggregate base (NCDOT, 2004). The depth of the underlying gravel layer was set artificially high at 50 cm to achieve a constant temperature condition at the lower boundary for the purpose of the finite difference solutions. Physical and thermal properties of the asphalt and gravel (Table 5.1) were taken from Barber (1957).

**Table 5.1** Asphalt and gravel physical and thermal properties (Barber, 1957)

	<b>Asphalt</b>	<b>Gravel</b>
<b>Density</b>	2250 kg/m <sup>3</sup>	1760 kg/m <sup>3</sup>
<b>Thermal Conductivity</b>	1.21 W/m K	1.30 W/m K
<b>Specific Heat</b>	921 J/kg K	837 J/kg K

## Soil Thermal Properties

There were several mechanisms incorporated into the model to account for soil thermal property inputs. Soil properties directly related to thermal infiltration calculations were soil specific heat of the solid phase, porosity, thermal conductivity of the solid phase, interfacial area, particle diameter, and the stagnant thermal conductivity of the saturated soil media. Because knowledge of soil thermal properties is often limited, estimation mechanisms were provided for several of these properties. Soil specific heat, thermal conductivity, and interfacial area could be selected from a series of values reported in the literature (Abu-Hamdeh, 2003; Campbell and Norman, 1998; de Vries, 1966; Ochsner et al. 2001) (Table 5.2). Several procedures were included to calculate the effective stagnant thermal conductivity of the saturated soil media (Abu-Hamdeh, 2003; Becker et al., 1992; Bristow, 1998; de Vries, 1966; Hsu et al., 1995; Lu et al., 2007; Usowicz et al., 2006). Text boxes were incorporated into the user interface portion of the model to allow for direct input of soil thermal properties since these parameters could be measured or estimated using techniques beyond those included in the model.

**Table 5.2** Soil thermal properties included in the model interface

<b>Specific Heat (kJ / kg K)</b>	<b>Notes</b>	<b>Source</b>
0.71	Soil Minerals	(de Vries, 1966)
1.9	Organic Matter	(de Vries, 1966)
0.87	Soil Minerals	(Campbell and Norman, 1998)
0.80	Low Estimate	(Ochsner et al., 2001)
0.90	High Estimate	(Ochsner et al., 2001)
<b>Thermal Conductivity (W / m K)</b>	<b>Notes</b>	<b>Source</b>
8.5	Quartz	(Usowicz et al., 2006)
2.5	Soil Minerals	(Campbell and Norman, 1998)
2.93	Soil Minerals	(de Vries, 1966)
0.251	Organic Matter	(de Vries, 1966)
3.06	Low Estimate	(Ochsner et al., 2001)
3.72	High Estimate	(Ochsner et al., 2001)

## Antecedent Soil Temperatures

Before the thermal exchanges between the bioretention soil and infiltrating water could be evaluated, it was necessary to determine the soil temperature profile before the storm began. The antecedent soil temperature profile was modeled using an analytical soil-temperature model developed by Elias et al. (2004) (Appendix A). The lower boundary temperature and parameters required for the sinusoidal surface temperature relationships were originally obtained from long-term weather data collected at three locations around the world. These parameters were localized for western North Carolina, based upon soil temperature data collected at the Lenoir and Asheville bioretention areas during the monitoring phase of the study (Table 5.3). Model parameters that required localization included annual average surface temperature, as well as annual and diurnal amplitude and damping depth.

**Table 5.3** Input parameters for antecedent soil temperature prediction

Parameter	Value
Annual Average Surface Temperature	20°C
Annual Amplitude	7°C
Annual Damping Depth	2.36 m
Daily Amplitude	4°C
Daily Damping Depth	0.12 m

One disadvantage of this soil temperature modelling approach was that it was entirely dependent upon time of year and time of day. As a result, differences in air temperature, radiation, rainfall, wind speeds, and other weather parameters were not directly accounted for in model results. Neglecting these factors would pose serious questions about the accuracy of the overall results from the bioretention thermal model if the model was implemented as a continuous simulation, since specific weather patterns can have substantial impacts on soil temperatures. Because storm based approaches inherently neglect long-term weather conditions, this assumption was deemed reasonable for the current version of the bioretention thermal model.

## Storm Soil and Water Temperature Profiles

The soil and water temperature profiles were calculated using a numerical solution to a two-phase, two-equation model for heat transfer in a fluid moving through porous media, presented by Nakayama et al. (2001) (Appendix A). Separate equations were used to describe the temperature of the fluid and solid phases, based on the following differential equations:

$$\rho_F C_{pF} \left[ \varepsilon \frac{\partial \langle T \rangle^F}{\partial t} + \langle \vec{u} \rangle \frac{\partial \langle T \rangle^F}{\partial z} \right] =$$

$$(\varepsilon + G(1 - \sigma)) k_F \frac{\partial^2 \langle T \rangle^F}{\partial z^2} + \bar{k}_{dis} \frac{\partial^2 \langle T \rangle^F}{\partial z^2} + a_{sf} h_{sf} (\langle T \rangle^S - \langle T \rangle^F)$$

$$- k_S G \left( \frac{\partial^2 \langle T \rangle^S}{\partial z^2} - \frac{\partial^2 \langle T \rangle^F}{\partial z^2} \right) \quad (4)$$

$$(1 - \varepsilon) \rho_S C_S \frac{\partial \langle T \rangle^S}{\partial t} =$$

$$(1 - \varepsilon + G(\sigma - 1)) k_S \frac{\partial^2 \langle T \rangle^S}{\partial z^2} - a_{sf} h_{sf} (\langle T \rangle^S - \langle T \rangle^F) + k_S G \left( \frac{\partial^2 \langle T \rangle^S}{\partial z^2} - \frac{\partial^2 \langle T \rangle^F}{\partial z^2} \right) \quad (5)$$

$$\sigma = \frac{k_S}{k_F} \quad (6)$$

$$G = \frac{\left( \frac{k_{stg}}{k_f} \right) - \varepsilon - (1 - \varepsilon) \sigma}{(\sigma - 1)^2} \quad (7)$$

$$\bar{k}_{dis} = k_f \left[ \frac{\rho_F C_{pF} \left| \langle \vec{u} \rangle \right| d_p}{2k_F} \right] \quad (8)$$

$$h_{sf} = \frac{k_F}{d_p} \left[ 2 + 1.1 \text{Pr}_F^{1/3} \left( \frac{\rho_F \left| \langle \vec{u} \rangle \right| d_p}{\mu_F} \right)^{0.6} \right] \quad (9)$$

where  $\rho$  represents density (kg/m<sup>3</sup>),  $C_p$  represents specific heat capacity (kJ/kg K),  $\varepsilon$  is soil porosity,  $T$  is temperature (K),  $t$  is time (s),  $\langle \vec{u} \rangle$  is the Darcian velocity of infiltrating water (m/s),  $z$  is depth below the soil surface (m),  $G$  is a tortuosity parameter,  $\sigma$  is the thermal conductivity ratio,  $k$  is the thermal conductivity (W/m K),  $\bar{k}_{dis}$  is the thermal dispersion tensor,  $a_{sf}$  is the interfacial area (m<sup>2</sup>/m<sup>3</sup>),  $h_{sf}$  is the interfacial heat transfer coefficient (W/m<sup>2</sup> K),  $d_p$  is the particle diameter (m),  $\text{Pr}$  is the Prandtl number, and  $\mu$  is the dynamic viscosity (N s/m<sup>2</sup>). In equations 3 and 4, the  $S$  subscripts apply to the solid phase, the  $F$  subscripts apply to the fluid phase, and the  $stg$  subscript applies to the combined solid and stagnant fluid.

A no flux boundary condition was applied to the lower boundary of the soil profile for both the solid and fluid phases. The soil profile was set artificially deep at 2 meters to make the boundary condition assumption reasonable. Only soil and water temperatures at or above the drain depth were included in model outputs. The upper boundary for the fluid phase was set equal to the runoff temperature during the current time step. The upper boundary for the solid phase was calculated based on the assumption that there was negligible internal resistance to heat transfer using an approach similar to the surface boundary of the pavement profile. The surface boundary calculations accounted for convective transfer between the soil surface and ponded runoff and incoming solar radiation, but did not consider evaporation or heat exchanges with the atmosphere. To account for vegetative shading of the

bioretention area, the solar radiation flux was multiplied by a ratio corresponding to the percentage of the bioretention area covered by vegetation. After the simulated storm had ended, the temperature of the upper boundaries for both the fluid and solid phases were held constant. A grid spacing of 5 cm and time step of 0.1 seconds were used for thermal infiltration calculations.

The governing equations were discretized using a finite difference scheme presented by Leonard (1979). This procedure, known as quadratic upstream interpolation for convective kinematics (QUICK), uses a three-point upstream-weighted quadratic interpolation to discretize convective terms. The QUICK scheme was advantageous because it avoided the inaccuracies associated with using upstream differencing and stability issues with using central differencing to describe convection terms. A derivation of the thermal infiltration equations and their discretization are shown in Appendix A.

### **Model Outputs**

The bioretention model provided several forms of output data to evaluate the thermal impact of a bioretention area (Table 5.4). Maximum and mean temperatures were calculated for the runoff, effluent, overflow, drainage, and seepage. The effluent term corresponded to the combination of overflow and drainage. Thermal loads were calculated by combining the results of hydrology calculations with temperature predictions. Based on the combination of these terms, it was possible to evaluate the effect of the bioretention area on the overall thermal load of the storm event. In addition to numerical summaries displayed in the model interface, temperature profiles were provided in graphical and tabular format for the antecedent pavement, pavement and runoff during the storm, fluid phase during infiltration, and solid phase during infiltration. The heat flux and cumulative thermal load were also presented over the duration of the storm event.

**Table 5.4** Description of model outputs

<b>Output</b>	<b>Description</b>
Runoff, Overflow, Underdrain, and Seepage Total Energy	Cumulative thermal energy for flow at each location (MJ)
Effluent Total Energy	Cumulative thermal energy for overflow and drainage (MJ)
Thermal Load Reduction	Reduction in cumulative thermal energy between runoff and effluent (%)
Volume Reduction	Reduction in cumulative runoff volume between inlet and outlet (%)
Runoff, Overflow, Underdrain, and Seepage Maximum Temperature	Maximum water temperature simulated at each location (°C)
Effluent Maximum Temperature	Maximum water temperature simulated for overflow or drainage (°C)
Runoff, Overflow, Underdrain, and Seepage Average Temperature	Average water temperature simulated at each location (°C)
Effluent Average Temperature	Average water temperature simulated for overflow or drainage (°C)

### **Additional Model Applications**

Although designed to evaluate the thermal impact of bioretention on urban stormwater runoff temperatures, there are additional potential uses for the bioretention thermal model. The bioretention thermal model could be used to describe a number of transient infiltration processes with little or no modification. Some of these applications include irrigation with industrial thermal effluent, infiltration of warm runoff into natural soil profiles, and general infiltration or rainfall into a soil with substantial temperature differences. The bioretention thermal model could also serve as a standalone hydrology model with additional refinement and validation.

### **Future Work**

Due to the numerous components involved in evaluating the thermal behavior of a bioretention area, there are many opportunities for future improvements to the bioretention thermal model. The algorithms used for the various model components were carefully selected based upon current research; however, there will

undoubtedly be potential improvements as work continues in these fields. It became necessary to limit the scope of the model and include a number of simplified assumptions throughout the model development in order to complete development work within a reasonable time frame. When possible, efforts were made throughout the model code to indicate areas for future improvements. Similarly, the model was programmed to be flexible for future improvements wherever possible. Arrays were generally used to transfer the results of one model component to another. The program could easily be adapted to either calculate array contents by some other means or load array values from an external source. For example, runoff flows were calculated within the model via the methodology discussed earlier and results were stored into an array, but future model improvements could implement more advanced watershed hydrology methods, such as kinematic wave theory. Similarly, an external stormwater modelling program could be used to generate watershed hydrology results and import those results directly into the bioretention thermal model.

Another future improvement that could be incorporated into the model is to account for thermal exchanges during the initial wetting of the soil profile. Although unsaturated conditions were incorporated into hydraulic calculations, the thermal component of the model assumes that the entire soil profile is saturated for the duration of the storm. The progression of the wetting front through the soil profile may have substantial impacts on the soil temperature profile as the warm water moves through the soil. The effects of the wetting front advancement on the soil temperature profile were not expected to have a major impact on runoff temperatures since the soil profile was saturated before there was any effluent thermal load. The latent heat exchanges associated with the initial saturation of the soil profile could have an impact on effluent temperatures. Although latent heat exchanges were expected to have minimal impacts on the overall thermal load when there was substantial saturated flow, accounting for these effects should improve the accuracy of the model.

## Code Optimization

The results of many calculations throughout the model were stored into arrays. Arrays were a convenient data structure because many calculations referenced the results of other calculations both temporally and spatially due to their discretization. When calculation results were stored in array format with rows representing each time step and column representing each spatial location, referencing simulation results was relatively easy. Because many of these calculations were performed on a fine grid with a short time step, the arrays storing calculation results had the potential to become quite large. Since these arrays were stored in computer RAM, the size of the arrays could potentially exceed the available memory capacity of the computer executing the model.

Mechanisms were incorporated into the model code to reduce total array size. Most model calculations only required data from the previous time step and immediately adjacent locations as inputs. As a result, calculations arrays were used that stored calculation results for all grid locations, but only the current and previous time step. When all of the calculations for a given time step were completed, the array row containing current calculation results was copied to the previous array row and calculations proceeded. A storage interval was set for each set of calculations, typically some multiple of the calculation time step (Table 5.3). When this storage interval was met or exceeded, the contents of the calculation array were copied to a larger array containing results for the entire simulation at the less frequent time interval. This larger array was subsequently used to provide data for other model components or output files and displays. In theory, some model resolution was lost in the process of aggregating these data; however, storage intervals were selected such that these effects were minimal.

The time step and grid spacing used in model calculations had a substantial impact on model accuracy and function, as well as execution time. A trial and error procedure was used whereby the grid spacing and time step for each model

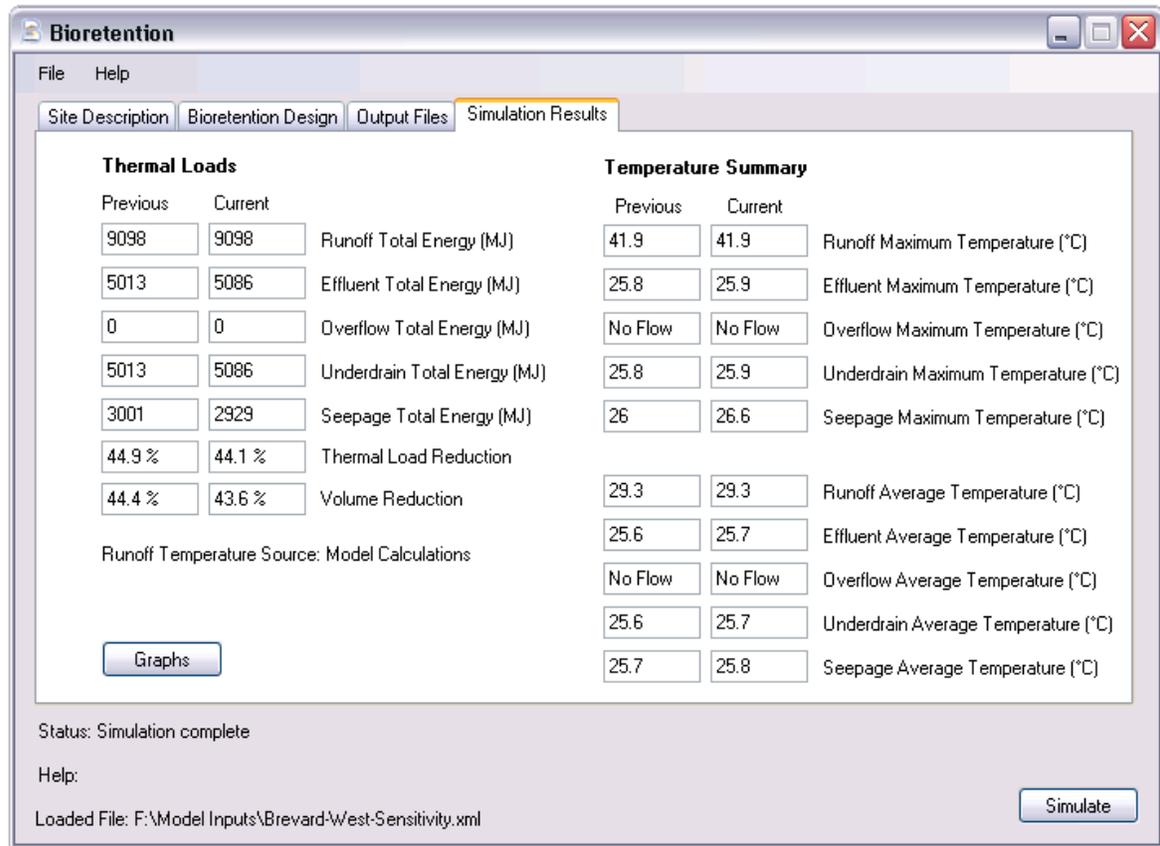
component was adjusted to ensure stability for the expected ranges of model inputs, while minimizing the required computation time (Table 5.5). Timers were incorporated into the model code to evaluate the specific impact of code changes on the computation time of each component.

**Table 5.5** Grid spacing, time step, and storage interval for model calculations

<b>Model Component</b>	<b>Grid Spacing (m)</b>	<b>Time Step (s)</b>	<b>Storage Interval (s)</b>
Bioretention Hydraulics	0.0001	1	1
Dry Pavement Temperatures	0.01	15	3600
Wet Pavement Temperatures	0.01	15	15
Infiltration Temperature Profiles	0.05	0.1	10

### **User Interface**

The overall user interface for the bioretention thermal model was organized into a series of tabs, with each tab providing options for entering required input data (Figure 5.7, Appendix D). Input data were grouped into the categories of site description, bioretention design, output files, and simulation results. Each tab contained a series of text boxes to input the required model parameters. The bioretention thermal model allowed users to load or save the contents of all text boxes as an xml file. The tag structure of the xml file made it possible to directly understand and edit the contents of the save file in a text editor. This organized file structure could facilitate the batch preparation of simulation inputs. An example save file can be found in Appendix C.



**Figure 5.7** Screenshot of model interface

## **References**

- Abu-Hamdeh, N. H. 2003. Thermal Properties of Soils as affected by Density and Water Content. *Biosystems Engineering* 86(1): 97-102.
- Ahuja, L. R. 1983. Modeling Infiltration into Crusted Soils by the Green-Ampt Approach. *Soil Science Society of America Journal* 47(3): 412-418.
- Barber, E. S. 1957. Calculation of Maximum Pavement Temperatures from Weather Reports. *Highway Research Board Bulletin* 168: 1-8.
- Becker, B. R., A. Misra and B. A. Fricke. 1992. Development of Correlations for Soil Thermal Conductivity. *International Communications in Heat and Mass Transfer* 19(1): 59-68.
- Bouwer, H. 1969. Infiltration of Water into Nonuniform Soil. *Journal of the Irrigation and Drainage Division, Proceedings of the American Society of Civil Engineers* 95(IR4): 451-462.
- Brakensiek, D. L. and C. A. Onstad. 1977. Parameter Estimation of the Green and Ampt Infiltration Equation. *Water Resources Research* 13(6): 1009-1012.
- Bristow, K. L. 1998. Measurement of Thermal Properties and Water Content of Unsaturated Sandy Soil using Dual-Probe Heat-Pulse Probes. *Agricultural and Forest Meteorology* 89(2): 75-84.
- Bristow, K. L., G. J. Kluitenberg, C. J. Goding and T. S. Fitzgerald. 2001. A Small Multi-Needle Probe for Measuring Soil Thermal Properties, Water Content, and Electrical Conductivity. *Computers and Electronics in Agriculture* 31(3): 265-280.
- Campbell, G. S., C. Calissendorff and J. H. Williams. 1991. Probe for Measuring Soil Specific Heat Using a Heat-Pulse Method. *Soil Science Society of America Journal* 55(1): 291-293.
- Campbell, G. S. and J. M. Norman. 1998. Chapter 11: Radiation Fluxes in Natural Environments. In *Introduction to Environmental Biophysics*, 167-184. New York: Springer.
- Campbell, G. S. and J. M. Norman. 1998. Chapter 8: Heat Flow in the Soil. In *An Introduction to Environmental Biophysics*, 113-128. New York: Springer.
- Childs, E. C. and M. Bybordi. 1969. The Vertical Movement of Water in Stratified Porous Material. 1. Infiltration. *Water Resources Research* 5(2): 446-459.

- Chu, S. T. 1978. Infiltration During an Unsteady Rain. *Water Resources Research* 14(3): 461-466.
- Chu, X. and M. A. Mariño. 2005. Determination of Ponding Condition and Infiltration into Layered Soils under Unsteady Rainfall. *Journal of Hydrology* 313(3-4): 195-207.
- Chung, S. and R. Horton. 1987. Soil Heat and Water Flow with a Partial Surface Mulch. *Water Resources Research* 23(12): 2175-2186.
- Collis-George, N. and R. Lal. 1973. The Temperature Profiles of Soil Columns During Infiltration. *Australian Journal of Soil Research* 11(1): 93-105.
- de Vries, D. A. 1966. Chapter 7: Thermal Properties of Soils. In *Physics of Plant Environment*, 210-235. ed. W. R. Van Wijk, Amsterdam: North-Holland Publishing Company.
- de Vries, D. A. 1958. Simultaneous Transfer of Heat and Moisture in Porous Media. *Transaction, American Geophysical Union* 39(5): 909-916.
- Diefenderfer, B. K., I. L. Al-Qadi and S. D. Diefenderfer. 2006. Model to Predict Pavement Temperature Profile: Development and Validation. *Journal of Transportation Engineering* 132(2): 162-167.
- Dietz, M.E. 2007. Low Impact Development Practices: A Review of Current Research and Recommendations for Future Directions. *Water, Air, and Soil Pollution* 186(1): 351-163.
- Dussaillant, A., K. Cozzetto, K. Brander and K. Potter. 2003. Green-Ampt Model of a Rain Garden and Comparison to Richards Equation Model. *Sustainable World* 6891.
- Dussaillant, A. R., C. H. Wu and K. W. Potter. 2004. Richards Equation Model of a Rain Garden. *Journal of Hydrologic Engineering* 9(3): 219-225.
- Dussaillant, A.R., A. Cuevas and K.W. Potter. 2005. Raingardens for Stormwater Infiltration and Focused Groundwater Recharge: Simulations for Different World Climates. *Water Science and Technology: Water Supply* 5(3-4): 173-179.
- Elias, E. A., R. Cichota, H. H. Torriani and De Jong Van Lier, Quirijn. 2004. Analytical Soil-Temperature Model: Correction for Temporal Variation of Daily Amplitude. *Soil Science Society of America Journal* 68(3): 784-788.

- Green, W. H. and G. A. Ampt. 1911. Studies on Soil Physics, 1, The flow of air and water through soils. *Journal of Agricultural Science* 4(1): 1-24.
- Heasom, W., R. G. Traver and A. Welker. 2006. Hydrologic Modeling of a Bioinfiltration Best Management Practice. *Journal of the American Water Resources Association* 42(5): 1329-1347.
- Hillel, D. and R. S. Baker. 1988. A Descriptive Theory of Fingering during Infiltration into Layered Soils. *Soil Science* 146(1): 51-56.
- Hsu, C. T. 1999. A Closure Model for Transient Heat Conduction in Porous Media. *Journal of Heat Transfer* 121(3): 733-739.
- Hsu, C. T., P. Cheng and K. W. Wong. 1995. A Lumped-Parameter Model for Stagnant Thermal Conductivity of Spatially Periodic Porous Media. *Journal of Heat Transfer* 117(2): 264-269.
- Hunt, W. F. and W. G. Lord. 2006. Urban Waterways: Bioretention Performance, Design, Construction, and Maintenance. AGW-588-051-9.
- James, W. and B. Verspagen. 1997. Chapter 8: Thermal Enrichment of Stormwater by Urban Pavement. In *Advances in Modeling the Management of Stormwater Impacts: Volume 5*, 155-177. ed. W. James, Guelph, Ontario: Computational Hydraulics International.
- Jury, W. A. and R. Horton. 2004. *Soil Physics*. 6th ed. Hoboken, NJ: John Wiley.
- Kaviany, M. 1991. *Principles of Heat Transfer in Porous Media*. 1st ed. New York: Springer-Verlag.
- Kent, K. M. 1973. A Method for Estimating Volume and Rate of Runoff in Small Watersheds. *Soil Conservation Service SCS-TP-149*.
- Leonard, B. P. 1979. A Stable and Accurate Convective Modelling Procedure Based on Quadratic Upstream Interpolation. *Computer Methods in Applied Mechanics and Engineering* 19(1): 59-98.
- Lu, S., T. Ren, Y. Gong and R. Horton. 2007. An Improved Model for Predicting Soil Thermal Conductivity from Water Content at Room Temperature. *Soil Science Society of America Journal* 71(1): 8-14.
- Mahrer, Y. 1979. Prediction of Soil Temperatures of a Soil Mulched with Transparent Polyethylene. *Journal of Applied Meteorology* 18(10): 1263-1267.

- Meeus, J. 1998. *Astronomical Algorithms*. Second Edition ed. Richmond: Willmann-Bell, Inc.
- Mein, R. G. and C. L. Larson. 1973. Modeling Infiltration during a Steady Rain. *Water Resources Research* 9(2): 384-394.
- Morel-Seytoux, H. J. and J. Khanji. 1974. Derivation of an Equation of Infiltration. *Water Resources Research* 10(4): 795-800.
- Moyne, C., S. Didierjean, H. P. Amaral Souto and O. T. da Silveira. 2000. Thermal Dispersion in Porous Media: One-Equation Model. *International Journal of Heat and Mass Transfer* 43(20): 3853-3867.
- Nakayama, A., F. Kuwahara, M. Sugiyama and G. Xu. 2001. A Two-Energy Equation Model for Conduction and Convection in Porous Media. *International Journal of Heat and Mass Transfer* 44(22): 4375-4379.
- North Carolina Climate Office. (2008). "NC Climate Retrieval and Observations Network of the Southeast Database." <<http://www.nc-climate.ncsu.edu/cronos>> (Sep 22, 2008).
- North Carolina Department of Transportation (NCDOT). 2004. Hot Mix Asphalt Quality Management System Manual: Section 3: Design of Asphalt Pavement Structures and Mixtures. *North Carolina Department of Transportation*.
- North Carolina Division of Water Quality. (2007). Stormwater Best Management Practices Manual. North Carolina Department of Environment and Natural Resources.
- Ochsner, T. E., R. Horton and T. Ren. 2001. A New Perspective on Soil Thermal Properties. *Soil Science Society of America Journal* 65(6): 1641-1647.
- Prunty, L. and J. Bell. 2005. Soil Temperature Change over Time during Infiltration. *Soil Science Society of America Journal* 69(3): 766-775.
- Quintard, M. and S. Whitaker. 1993. One- and Two-Equation Models for Transient Diffusion Processes in Two-Phase Systems. In *Advances in Heat Transfer*, 369. New York: Academic Press.
- Quintard, M. and S. Whitaker. 1995. Local Thermal Equilibrium for Transient Heat Conduction: Theory and Comparison with Numerical Experiments. *International Journal of Heat and Mass Transfer* 38(15): 2779-2796.

- Reicosky, D. C., L. J. Winelman, J. M. Baker and D. G. Baker. 1989. Accuracy of Hourly Air Temperatures Calculated from Daily Minima and Maxima. *Agricultural and Forest Meteorology* 46(3): 193-209.
- Richards, L.A. 1931. Capillary Conduction of Liquids in Porous Media. *Physics* 1: 318-333.
- Rubin, J. and R. Steinhardt. 1964. Soil Water Relations During Rain Infiltration: III. Water Uptake at Incipient Ponding. *Soil Science Society of America Journal* 28(5): 614-619.
- Sadler, E. J. and R. W. Schroll. 1997. An Empirical Model of Diurnal Temperature Patterns. *Agronomy Journal* 89(4): 542-548.
- Satyamurty, V. V. and K. Sarath Babu. 1999. Relative Performance of Correlations to Estimate Hourly Ambient Air Temperature and Development of a General Correlation. *International Journal of Energy Research* 23(8): 663-673.
- Selker, J. S., J. Duan and J. Parlange. 1999. Green and Ampt Infiltration into Soils of Variable Pore Size with Depth. *Water Resources Research* 35(5): 1685-1688.
- Shao, M., R. Horton and D. B. Jaynes. 1998. Analytical Solution for One-Dimensional Heat Conduction-Convection Equation. *Soil Science Society of America Journal* 62(1): 123-128.
- Spokas, K. and F. Forcella. 2006. Estimating Hourly Incoming Solar Radiation from Limited Meteorological Data. *Weed Science* 54(1): 182-189.
- Timlin, D. J., Y. Pachepsky, B. A. Acock, J. Simunek, G. Ferchinger and F. Whisler. 2002. Error Analysis of Soil Temperature Simulations Using Measured and Estimated Hourly Weather Data with 2DSOIL. *Agricultural Systems* 72(3): 215-239.
- Usovicz, B., J. Lipiec and A. Ferrero. 2006. Prediction of Soil Thermal Conductivity Based on Penetration Resistance and Water Content or Air-Filled Porosity. *International Journal of Heat and Mass Transfer* 29(25-26): 5010-5017.
- Van Buren, M.A., W.E. Watt, J. Marsalek and B.C. Anderson. 2000. Thermal Enhancement of Stormwater Runoff by Paved Surfaces. *Water Research* 34(4): 1359-1371.
- van Genuchten, M. T. 1980. A Closed-Form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils. *Soil Science Society of America Journal* 44(5): 892-898.

- van Wijk, W. R. and D. A. de Vries. 1966. Chapter 4: Periodic Temperature Variations in a Homogeneous Soil. In *Physics of Plant Environment*, 102-140. ed. W. R. van Wijk, Amsterdam: North-Holland Publishing Company.
- Wakao, N. and S. Kaguei. 1982. *Heat and Mass Transfer in Packed Beds*. New York: Gordon and Breach Science Publishers.
- Warrick, A. W., D. Zerihun, C. A. Sanchez and A. Furman. 2005. Infiltration under Variable Ponding Depths of Water. *Journal of Irrigation and Drainage Engineering* 131(4): 358-363.
- Wierenga, P. J. and C. T. de Wit. 1970. Simulation of Heat Transfer in Soils. *Soil Science Society of America Journal* 34(6): 845-848.
- Wierenga, P. J., R. M. Hagan and D. R. Nielsen. 1970. Soil Temperature Profiles during Infiltration and Redistribution of Cool and Warm Irrigation Water. *Water Resources Research* 6(1): 230-238.

## **Chapter 6**

### **Bioretention Thermal Model: Validation and Sensitivity Analysis**

## **Model Validation**

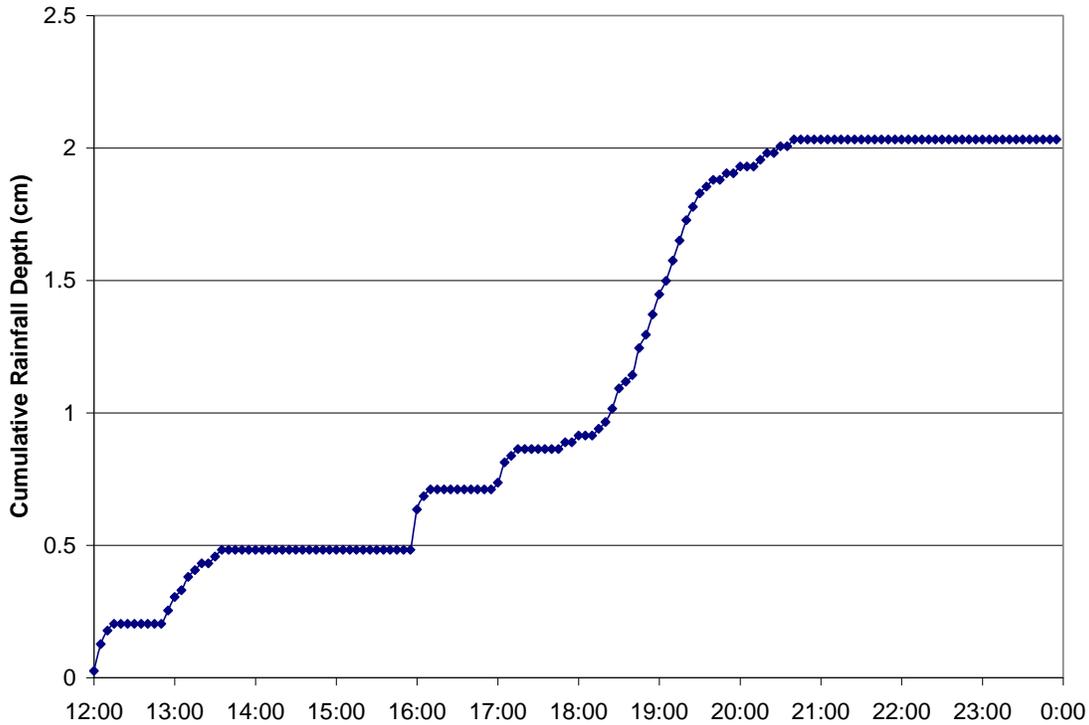
### **Validation Procedure**

The bioretention thermal model was validated by comparing simulation results with monitoring data collected at 4 bioretention areas in western North Carolina during the 2006 and 2007 monitoring periods. A detailed site description of the monitoring locations can be found in Chapter 3. Due to model assumptions, not all storm events that were analyzed during the course of the monitoring study could be used in validating the model. Comparing simulation results with monitoring data was expected to evaluate the cumulative effect of errors from the individual model components and establish the validity of simulation results.

Because the thermal calculations of the bioretention model assumed saturated flow conditions, only storms that had substantial periods of saturated flow were used for model validation. These storms were identified by calculating the amount of rainfall required over the watershed to fill the available soil porosity for each bioretention area (Table 6.1). Furthermore, storms were negated from validation efforts if there was a period of 15 minutes or more with less than 0.25 mm of rainfall, since the model assumed rainfall was continuous (Figure 6.1). In some cases, an isolated portion of a larger storm event that satisfied these criteria was used in model validation. Storms were not validated if the portion of rainfall satisfying validation criteria was preceded by more than 0.25 mm of rainfall within 6 hours, since this situation would violate the model assumption of dry conditions before a storm event. It is important to clarify that storms which failed to meet these criteria could still generate measurable outflow and be used in monitoring study analysis; however, they violated integral assumptions of the bioretention thermal model which were discussed in more detail in Chapter 5. Similarly, these validation criteria are not expected to limit use of the model since most storms typically used for stormwater designs would meet these criteria.

**Table 6.1:** Rainfall depths required for validation

Location	Required Rainfall (cm)	Number of Storms Validated
Asheville	3.80	0
Brevard East	0.89	13
Brevard West	1.13	8
Lenoir	1.26	5



**Figure 6.1** Hyetograph for a storm that would not meet conditions for validation due to extended periods without rainfall at 12:15, 13:35, 16:10, and 17:15

Rainfall input files were generated from tipping bucket rain gauge data collected at a location near each monitoring site (See Chapter 3). The tipping bucket rain gauges recorded the time of each bucket tip, which corresponded to 0.25 mm of accumulated rainfall. During the course of analyzing monitoring data, rainfall was aggregated into 5 minute time intervals and these aggregated data were used to generate rainfall input files. Because the model code did not require a constant rainfall time step, tipping bucket data could have been loaded directly into the model. However, this increased resolution was not deemed necessary for model validation.

For instances where there were no bucket tips recorded in a 5 minute interval, the time step was removed from the input data file. Removing a time step from the input data file evenly distributed the rainfall amount recorded in the next time step over the entire interval.

Model inputs were determined based on a combination of onsite measurements and estimation procedures (Table 6.2). The watershed area and bioretention surface area were determined from survey data at each site. The latitude, longitude, and elevation were determined using a geographic information systems (GIS) program in conjunction with aerial photography. An atmospheric transmittance value of 0.6 was selected based upon recommendations presented by Spokas and Forcella (2006). The drain depth and surface storage capacity were based upon site measurements and survey data. Shading at each bioretention area was based upon analysis of overhead digital photographs. The suction head, fillable porosity, and total porosity were based upon the moisture release curves measured for soil samples collected from each site using a pressure plate apparatus. Two intact soil samples were collected from each site just below the soil surface. The saturated hydraulic conductivity was also based on measurements from soil samples collected at each site. The subsoil saturated hydraulic conductivity was estimated based upon data published in the Natural Resources Conservation Service (NRCS) soil survey for the site locations (Soil Survey Staff, 2008). The soil specific heat and thermal conductivity were estimated based on procedures presented by de Vries (1966) and Ochsner et al. (2001). Stagnant thermal conductivities used in model validation were calculated using the analytical procedure described by Hsu (1999). The particle diameter of the soil at each site was based upon typical diameters reported in the literature for sandy soils. The air temperature during the storm was input as the average air temperature observed from monitoring data. The radiation during the storm was based upon average radiation measurements during rainfall for the duration of the 2007 monitoring period that were collected at the Asheville stormwater wetland as part of this study (Chapter 2).

**Table 6.2** Model inputs used for validation

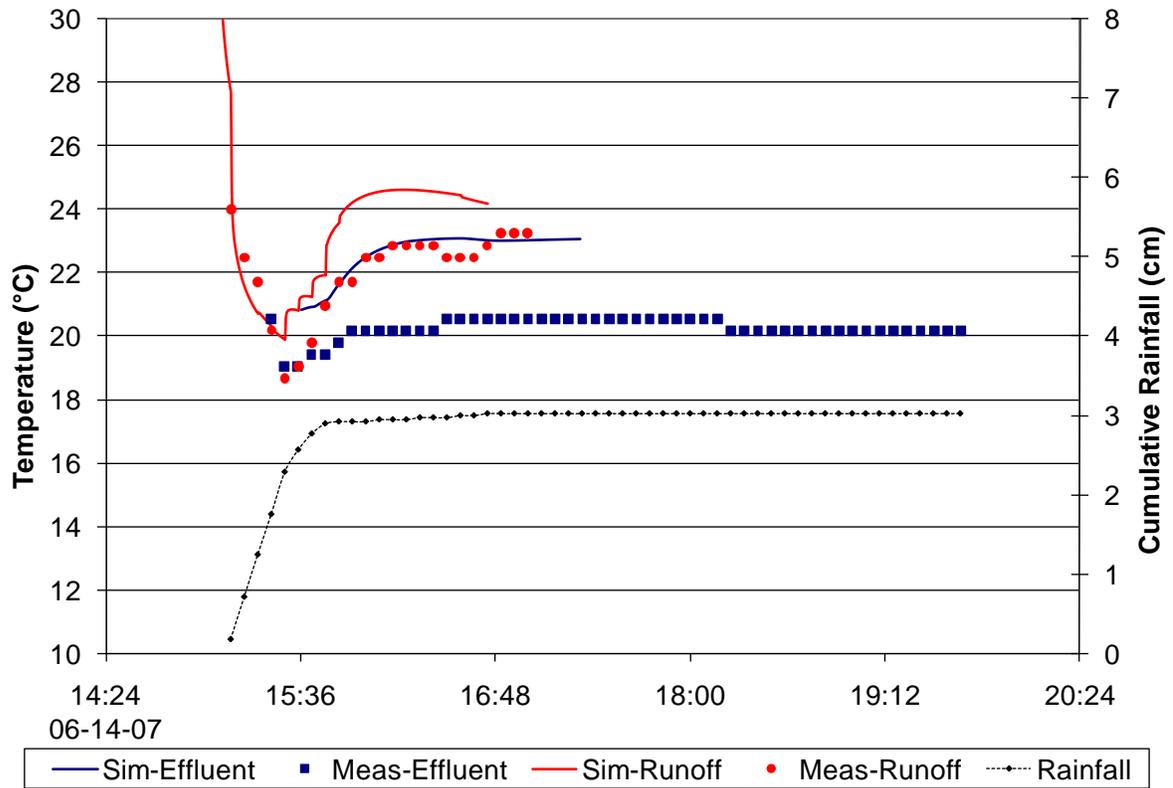
	<b>Asheville</b>	<b>Brevard East</b>	<b>Brevard West</b>	<b>Lenoir</b>
<b>Watershed Area</b>	280 m <sup>2</sup>	525 m <sup>2</sup>	325 m <sup>2</sup>	769 m <sup>2</sup>
<b>Initial Abstraction</b>	0.5 mm	0.5 mm	0.5 mm	0.5 mm
<b>Bioretention Area</b>	45 m <sup>2</sup>	37 m <sup>2</sup>	36 m <sup>2</sup>	30 m <sup>2</sup>
<b>Latitude</b>	35.613°	35.239°	35.239°	35.922°
<b>Longitude</b>	82.565°	82.731°	82.731°	81.523°
<b>Transmittance</b>	0.6	0.6	0.6	0.6
<b>Elevation</b>	615 m	656 m	656 m	343 m
<b>Time Zone</b>	-5	-5	-5	-5
<b>Shading</b>	0.55	0.43	0.43	0.79
<b>Drain Depth</b>	1.35 m	0.48 m	0.43 m	0.95 m
<b>Suction Head</b>	0.138 m	0.138 m	0.138 m	0.138 m
<b>Hydraulic K<sub>sat</sub></b>	67.8 cm/hr	14.8 cm/hr	17.9 cm/hr	108 cm/hr
<b>Fillable Porosity</b>	0.175 cm <sup>3</sup> /cm <sup>3</sup>	0.264 cm <sup>3</sup> /cm <sup>3</sup>	0.238 cm <sup>3</sup> /cm <sup>3</sup>	0.339 cm <sup>3</sup> /cm <sup>3</sup>
<b>Surface Storage Capacity</b>	20.0 cm	16.8 cm	15.5 cm	30.8 cm
<b>Gravel Porosity</b>	0.30	0.30	0.30	0.30
<b>Drain height</b>	75 mm	75 mm	75 mm	75 mm
<b>Subsoil Hydraulic K<sub>sat</sub></b>	1 cm/hr	6 cm/hr	6 cm/hr	3 cm/hr
<b>Soil Specific Heat</b>	0.85 kJ/kg K	0.85 kJ/kg K	0.85 kJ/kg K	0.85 kJ/kg K
<b>Soil Porosity</b>	0.625	0.523	0.516	0.525
<b>Soil Thermal Conductivity</b>	3.0 W/m K	3.0 W/m K	3.0 W/m K	3.0 W/m K
<b>Interfacial Area</b>	25	25	25	25
<b>Stagnant Thermal Conductivity</b>	1.18 W/m K	1.18 W/m K	1.18 W/m K	1.18 W/m K
<b>Particle Diameter</b>	0.5 mm	0.5 mm	0.5 mm	0.5 mm
<b>Storm Radiation</b>	78 W/m <sup>2</sup>	78 W/m <sup>2</sup>	78 W/m <sup>2</sup>	78 W/m <sup>2</sup>

Effluent temperatures calculated by the model and measured at the monitoring sites were used for validation. Because monitoring data were collected at a 5 minute interval, only model results that matched the time of temperature measurements were used for comparison, with the intermediate simulated temperatures discarded. The root mean squared error (RMSE), mean error (ME), and mean absolute error (MAE) were calculated over the duration of the storm event, comparing simulated and measured effluent temperatures. The RMSE calculations provided an indication of the weighted model error, mean error calculations accounted for the cancelling effect of overestimations and underestimations, while the mean absolute error calculation described the overall magnitude of model errors. When the duration of outflow differed between measured values and simulation results, the outflow times recorded during the monitoring study were given precedent.

In order to eliminate the influence of runoff temperature estimates on the overall model performance, simulations were also conducted based on monitored runoff temperatures. For each storm event, an xml file was created containing runoff temperatures recorded at the monitoring sites. The runoff temperatures were loaded at a 5 minute interval, assuming runoff temperature was constant during the 5 minute period. The mean effluent temperature calculated by the model and mean effluent temperature observed during monitored storms was used for comparison.

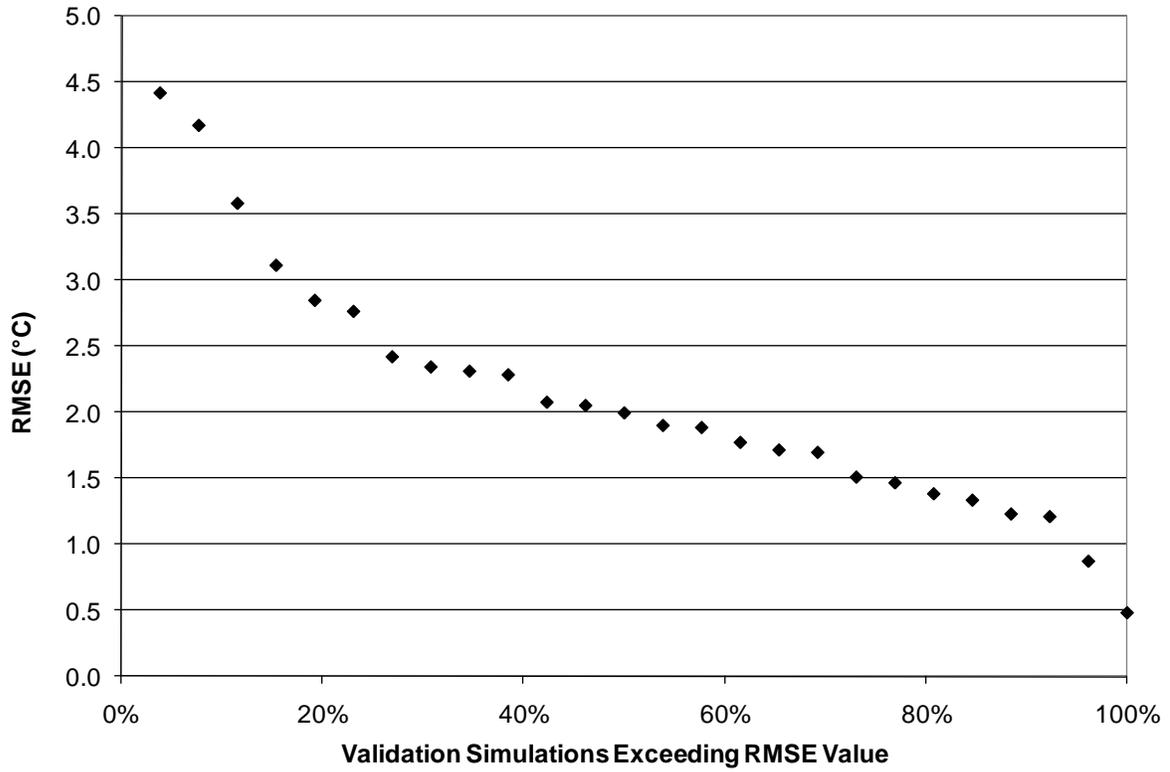
### **Validation Results**

Simulated runoff and effluent temperatures often followed the same trends as measured values, with some differences in temperature magnitude and timescale (Figure 6.2). Runoff and effluent temperature plots for all validation simulations can be found in Appendix E. Effluent flows often persisted longer in monitoring data than what was predicted by the bioretention thermal model. These differences in flow duration were likely associated with the model's uniform water content assumptions before and after a storm event.



**Figure 6.2** Comparison of simulated (Sim) and measured (Meas) runoff and effluent temperatures for a storm on 06-14-07 at the Brevard west bioretention area

The majority of storm events analyzed for the model validation had an RMSE of 2.0°C or less, which was substantially less than observed variation in monitored effluent temperatures (Figure 6.3). The largest errors were associated with comparisons at the Lenoir bioretention area, while simulations for the Brevard east location were the most accurate (Table 6.3).



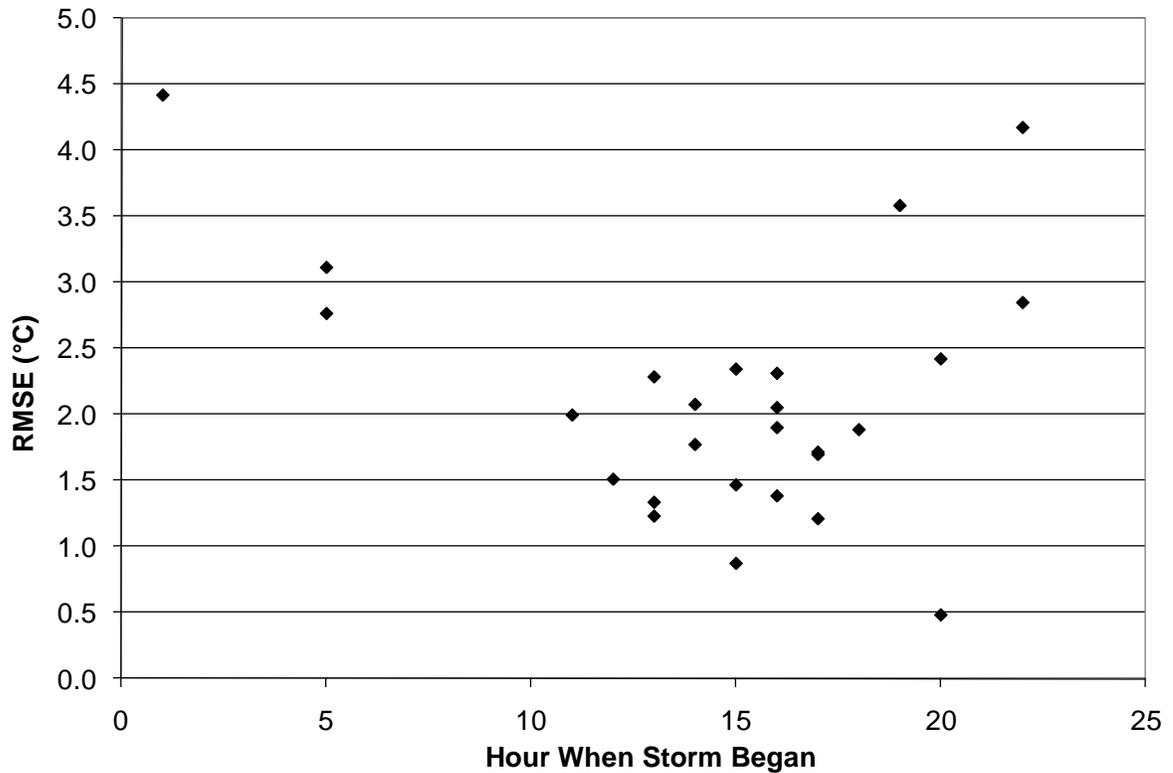
**Figure 6.3** Portion of validation simulations exceeding each root mean squared error value

**Table 6.3** Validation simulation results for storm events at each site

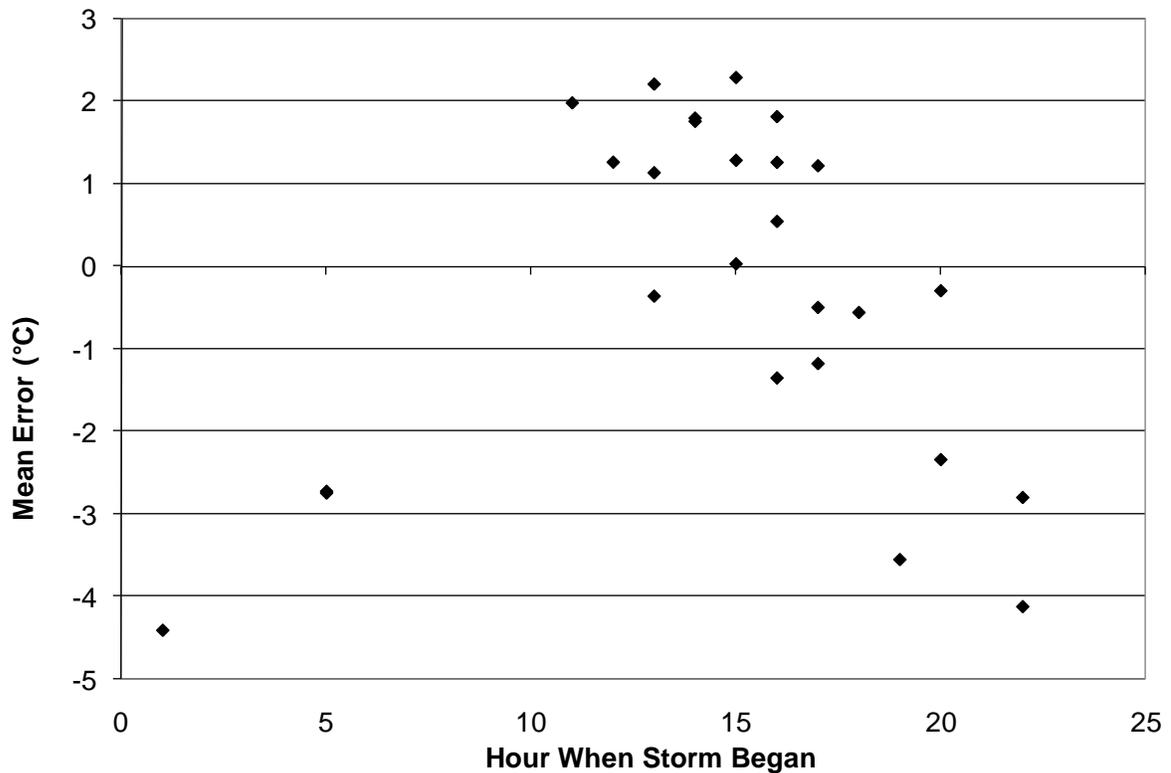
<b>Storm Start Date and Time</b>	<b>Mean Error (°C)</b>	<b>Mean Absolute Error (°C)</b>	<b>RMSE (°C)</b>	<b>Rainfall Total (cm)</b>
<b>Lenoir</b>				
6/14/2007 18:30	-0.56	1.70	1.88	2.82
7/25/2007 16:25	0.55	1.16	1.38	3.25
8/10/2006 22:25	-4.12	4.12	4.17	1.63
8/15/2006 17:10	-0.50	1.46	1.71	5.11
10/4/2007 19:50	-3.55	3.55	3.58	2.77
<b>Mean</b>	-1.63	2.40	2.54	3.11
<b>Brevard West</b>				
6/14/2007 15:25	2.29	2.29	2.34	3.02
6/24/2006 14:45	1.80	1.80	2.07	3.00
6/26/2006 5:15	-2.74	2.74	2.76	3.05
7/25/2006 16:20	1.26	1.47	2.31	2.26
8/29/2006 17:15	1.22	1.22	1.69	5.74
8/30/2007 14:05	1.76	1.76	1.77	1.80
9/4/2006 22:45	-2.80	2.80	2.84	2.36
9/18/2006 20:45	-0.29	0.45	0.48	2.34
<b>Mean</b>	0.31	1.82	2.03	2.95
<b>Brevard East</b>				
5/11/2007 16:05	1.82	1.82	1.90	1.35
6/2/2006 13:10	-0.36	1.08	1.23	2.16
6/12/2007 15:20	0.03	0.77	0.87	2.46
6/14/2007 15:20	1.29	1.32	1.46	3.02
6/26/2006 5:00	-2.72	2.74	3.11	3.05
7/25/2006 16:30	-1.35	1.93	2.05	2.26
7/26/2007 12:55	1.26	1.26	1.51	2.87
8/12/2006 1:10	-4.41	4.41	4.42	2.95
8/14/2006 13:50	2.21	2.21	2.28	1.50
8/22/2006 11:55	1.98	1.98	1.99	1.17
8/29/2006 17:50	-1.18	1.18	1.21	5.74
8/30/2007 13:50	1.14	1.16	1.33	1.80
9/18/2006 20:25	-2.34	2.34	2.42	2.34
<b>Mean</b>	-0.20	1.86	1.98	2.51
<b>Grand Mean</b>	-0.32	1.95	2.11	2.76

There was no evidence of correlation between time of day and model error; however, some of the largest errors were associated with late evening and nighttime

storms (Figure 6.4). Predicted effluent temperatures were typically warmer than measured values between 10:00 and 17:00, and cooler than measured effluent temperatures for the remainder of the day (Figure 6.5). The distribution of mean errors with time of day indicated that errors in calculation of diurnal fluctuations in pavement and soil profiles were likely responsible for effluent temperature errors.

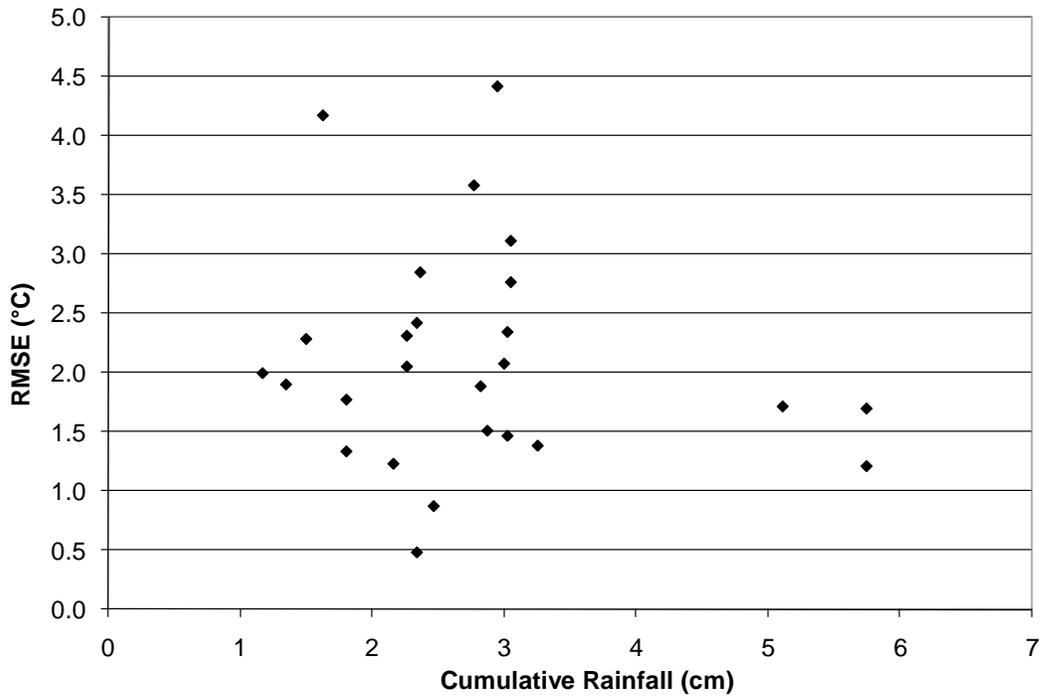


**Figure 6.4** Distribution of RMSE for storms at different times during the day

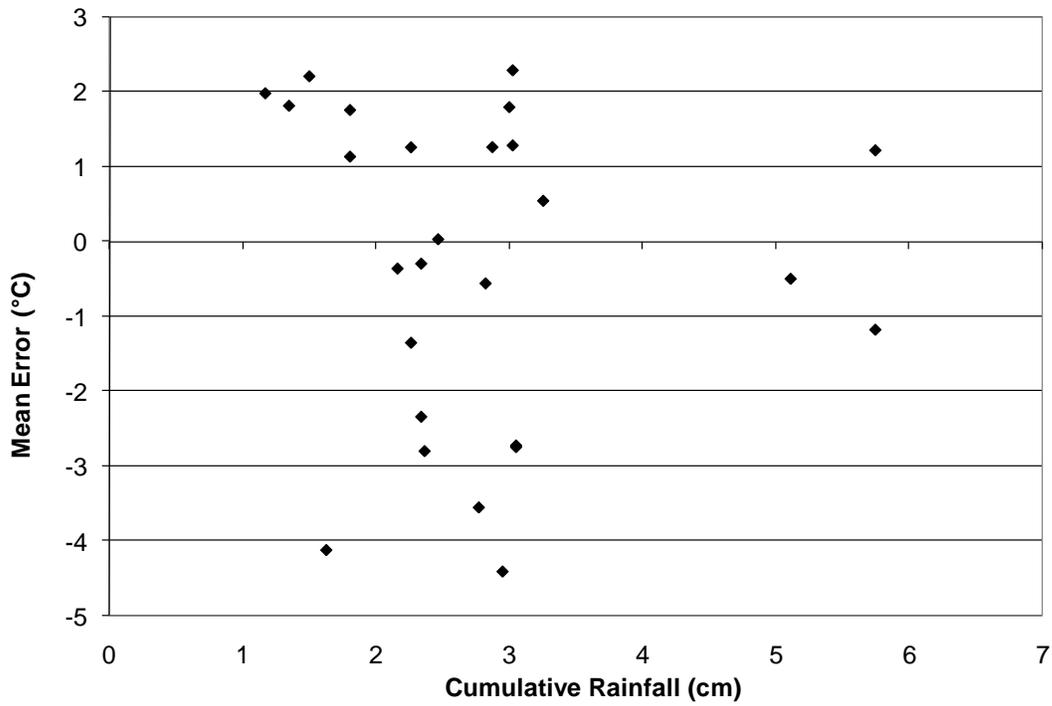


**Figure 6.5** Distribution of mean error for storms at different times during the day

The largest errors were associated with rainfall events less than 3 cm (Figure 6.6). It is reasonable to expect that accuracy would be improved for larger storm events because the overall effect of uncertainties early in a storm would be diminished. Because there were relatively few rainfall events greater than 3 cm, it was not possible to definitively conclude that simulation results were more accurate for larger storm events. Effluent temperatures were typically under predicted for rainfall depths between 2 and 3 cm; however, the lack of a consistent trend suggested that under prediction was likely coincidental with the time of these rainfall events (Figure 6.7).

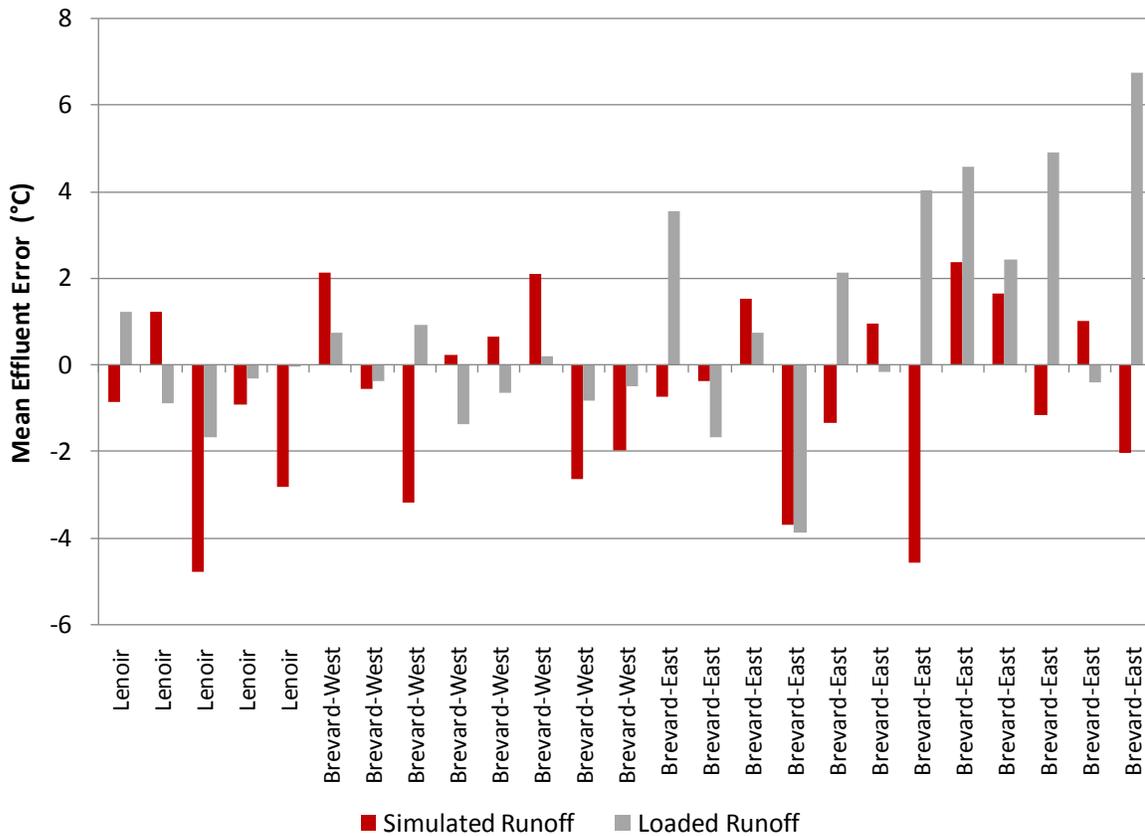


**Figure 6.6** Distribution of RMSE for different storm rainfall depths



**Figure 6.7** Distribution of mean error for different storm rainfall depths

A comparison of model outputs using simulated runoff temperatures and loading monitored runoff temperatures showed no consistent improvement in model predictions when using loaded runoff temperatures (Figure 6.8). Mean effluent errors were substantially larger for loaded runoff temperatures at the Brevard east location. During the course of the monitoring study, the warmest runoff temperatures were often observed at the Brevard east site, which may be responsible for the increased error.

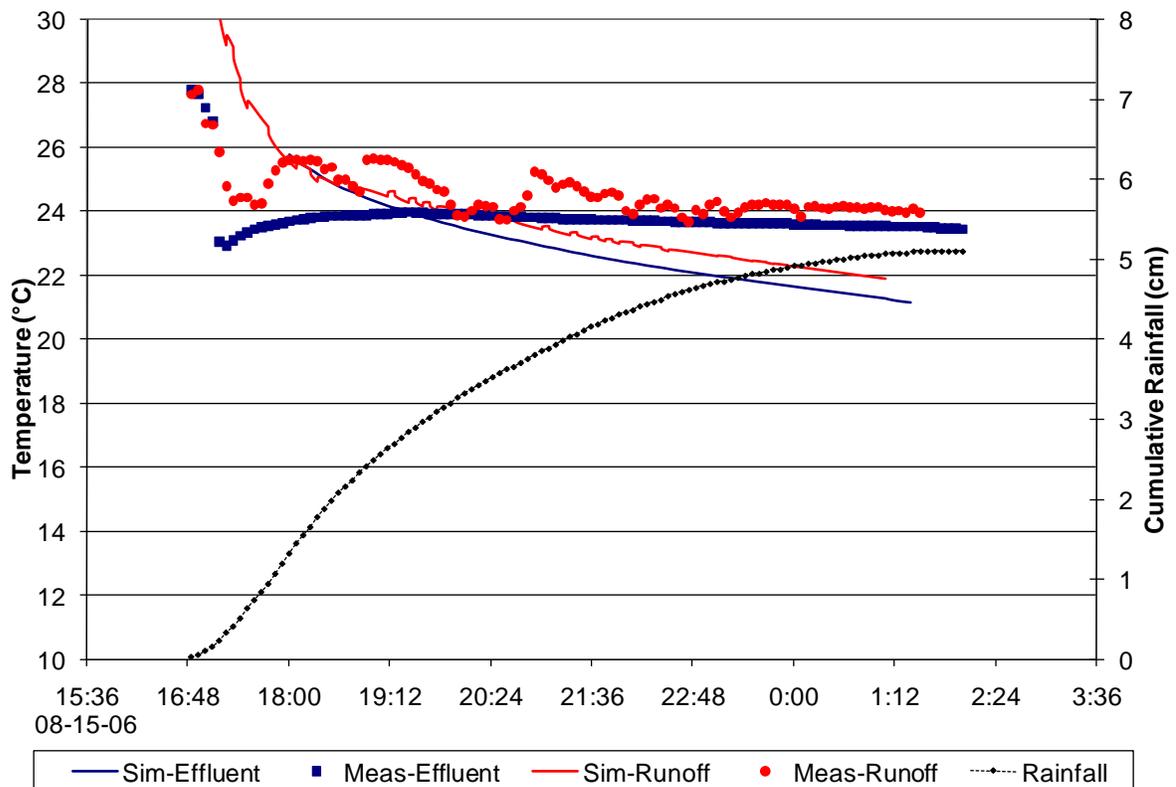


**Figure 6.8** Comparison of errors in mean effluent temperature results when using simulated runoff temperatures and measured runoff temperatures for model calculations

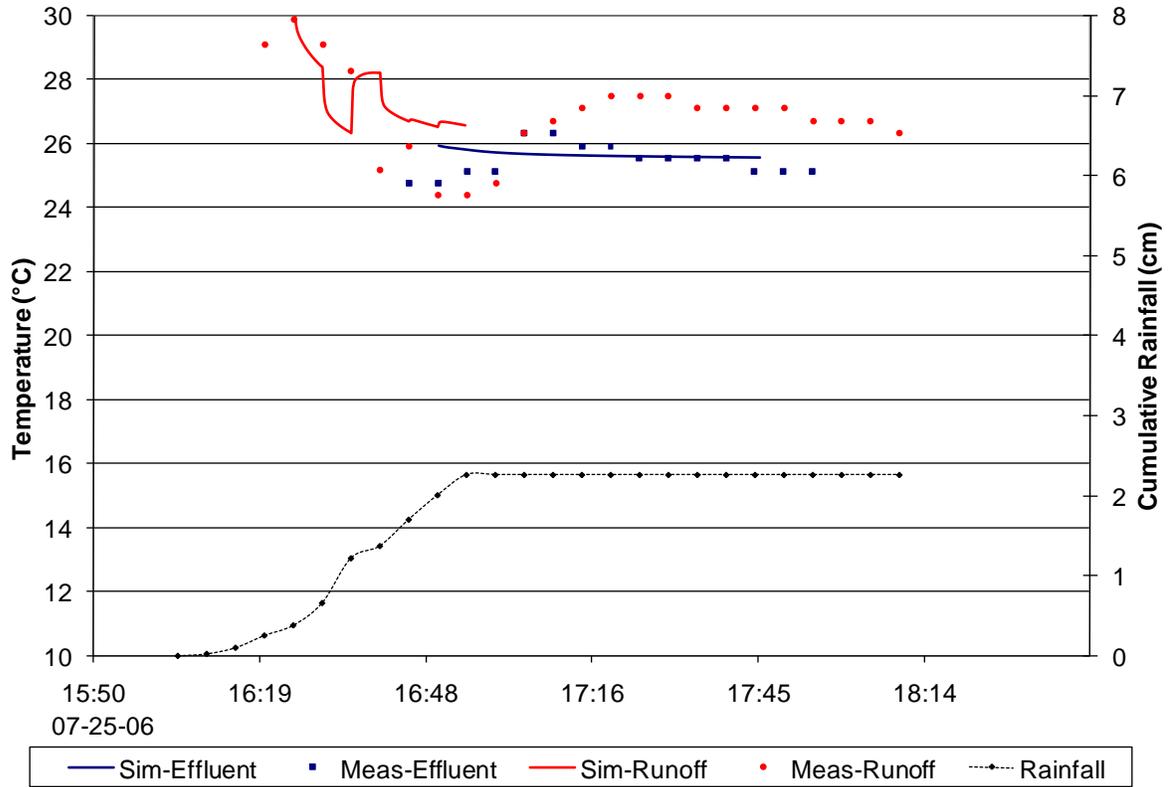
## Sensitivity Analysis

### Procedure

A sensitivity analysis was conducted to determine the influence of model inputs on effluent temperatures and evaluate the potential impact of input errors on model performance. The sensitivity analysis was conducted using a monitored storm event at the Lenoir bioretention area and Brevard west bioretention area. The storm event used for the Lenoir bioretention area analysis was selected due to its large rainfall depth and relatively steady rainfall intensity (Figure 6.9). The storm event used for the Brevard west bioretention area analysis was selected because it had intense rainfall and the total rainfall depth was better representative of a typical storm event (Figure 6.10).



**Figure 6.9** Simulated (Sim) and measured (Meas) runoff and effluent temperatures for the Lenoir storm used in the sensitivity analysis



**Figure 6.10** Simulated (Sim) and measured (Meas) runoff and effluent temperatures for the Brevard west storm used in the sensitivity analysis

For each model parameter analyzed, the input value of interest was adjusted and the mean and maximum runoff, effluent, overflow, drainage, and seepage temperatures were recorded from the model output. Model parameters were adjusted from their initial value based on a specified percentage. Typically, model parameters were adjusted at an increment of 10% for the range from -50% to 50%. When anticipated ranges were available from the literature, parameters were adjusted to ensure they encompassed the specified range. Base input parameters can be found earlier in the text in Table 6.2. Mean and maximum effluent temperatures were evaluated to establish model sensitivity for most parameters (Table 6.4).

**Table 6.4** Simulation outputs for events used in sensitivity analysis before any input adjustments were made

<b>Simulation Location</b>	<b>Maximum Effluent Temperature</b>	<b>Mean Effluent Temperature</b>
<b>Lenoir</b>	25.9 °C	25.7 °C
<b>Brevard West</b>	25.8 °C	22.9 °C

## Results

Analysis showed that effluent temperature outputs from the bioretention thermal model were not sensitive for expected errors or variation in particle diameter and interfacial area. The lack of sensitivity to interfacial area was advantageous, since this parameter was difficult to measure or accurately predict. A number of input parameters changed effluent temperatures by less than 1% when adjusted within their expected ranges (Table 6.5). Plots of specific impacts for all input parameters can be found in Appendix E.

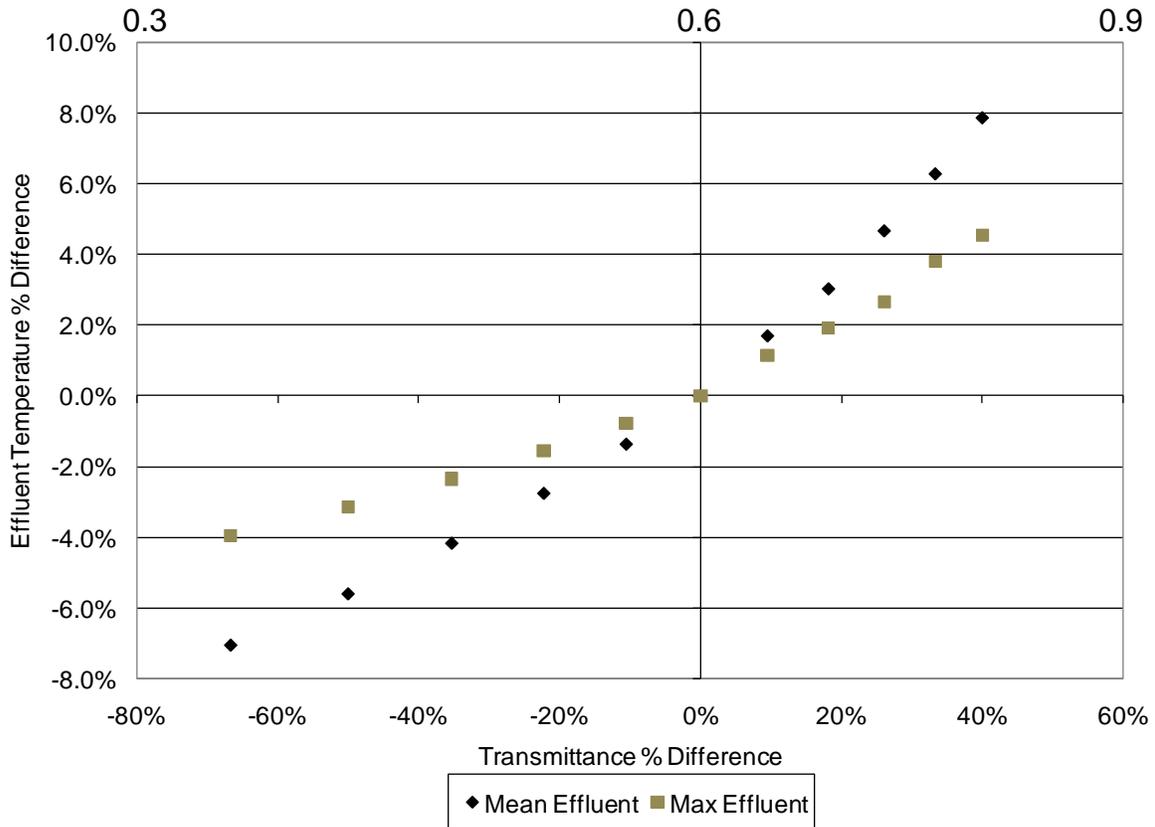
**Table 6.5** Summary of input parameters with low sensitivity

<b>Input Parameter</b>	<b>Effect of Input Increase</b>		<b>Effect of Input Decrease</b>	
	<b>Maximum Effluent</b>	<b>Mean Effluent</b>	<b>Maximum Effluent</b>	<b>Mean Effluent</b>
Elevation	Increase	No Change	Decrease	Decrease
Saturated Hydraulic Conductivity	No Change	No Change	Decrease	Decrease
Specific Heat	Varied	Varied	Varied	Varied
Porosity	Varied	Varied	Varied	Varied

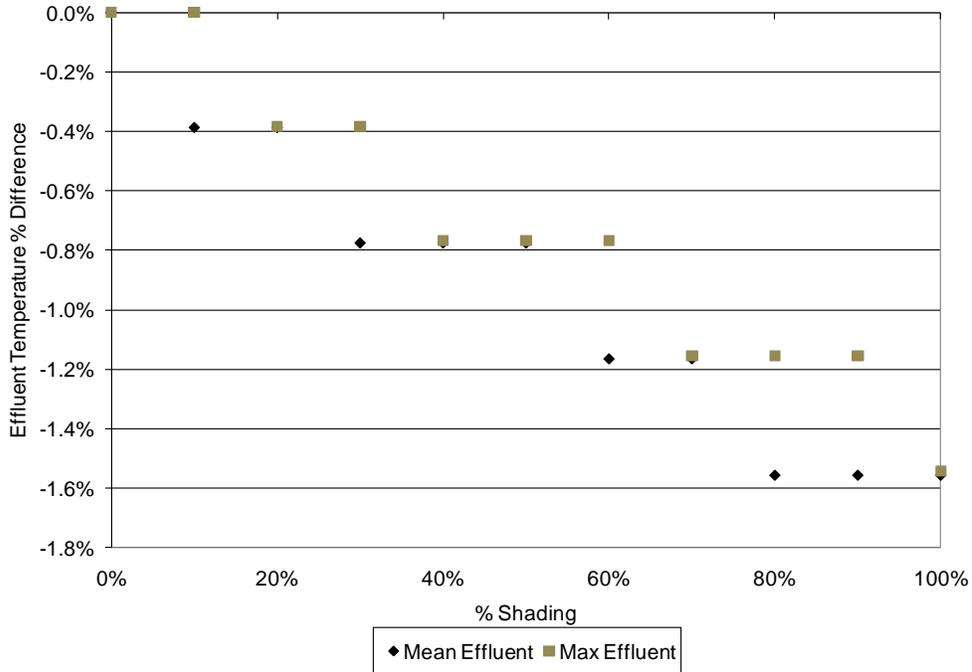
Changing the soil specific heat and porosity had different effects at the Lenoir and Brevard west locations. Decreasing the specific heat increased effluent temperatures at the Brevard west location, while decreasing effluent temperatures at the Lenoir location. Decreasing the specific heat of the soil should decrease the effective buffering capacity of the soil system, with heat fluxes from the incoming runoff having a more substantial impact on soil temperatures. Due to the larger storm depth for the Lenoir simulations, cooler runoff temperatures persisted for much of the storm event and cooler effluent temperatures were obtained because

the soil did not retain heat stored before the storm and during the early portions with warm runoff.

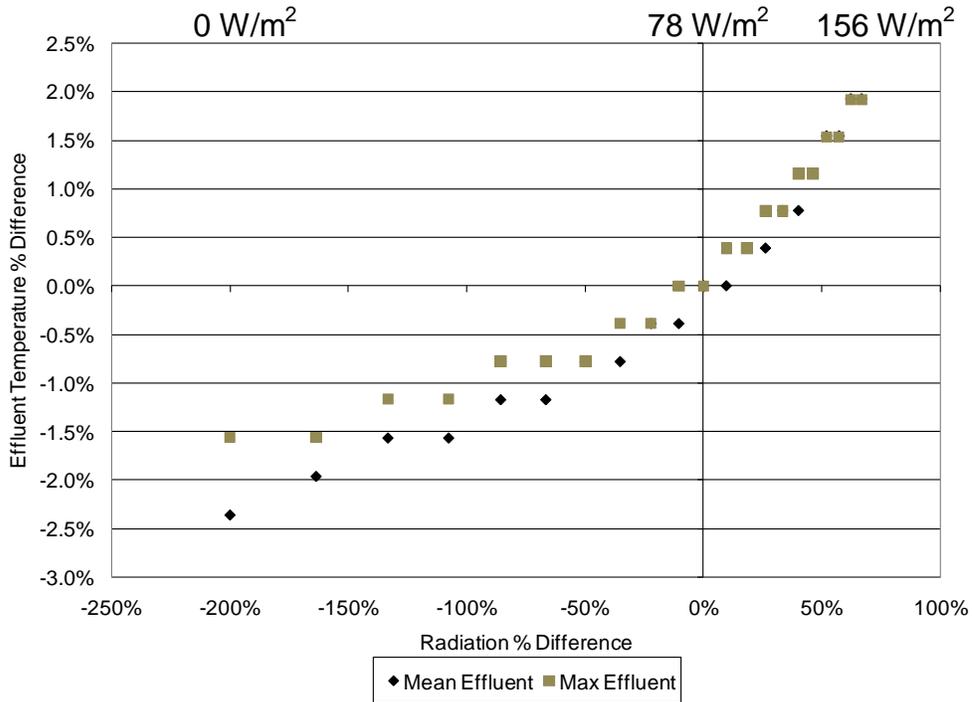
There were a number of input parameters that affected solar radiation calculations, consequently impacting the soil and pavement surface heat balance. Increasing the transmittance and elevation resulted in warmer effluent temperatures due to the increased radiation heat flux at the soil and pavement surface (Figure 6.11, Table 6.5). Similarly, increasing shading of the bioretention areas reduced the surface radiation heat flux for the soil surface, reducing effluent temperatures (Figure 6.12). Finally, increasing solar radiation directly during a storm event resulted in increased effluent temperature (Figure 6.13).



**Figure 6.11** Sensitivity of effluent temperature to transmittance adjustments for the Brevard west location

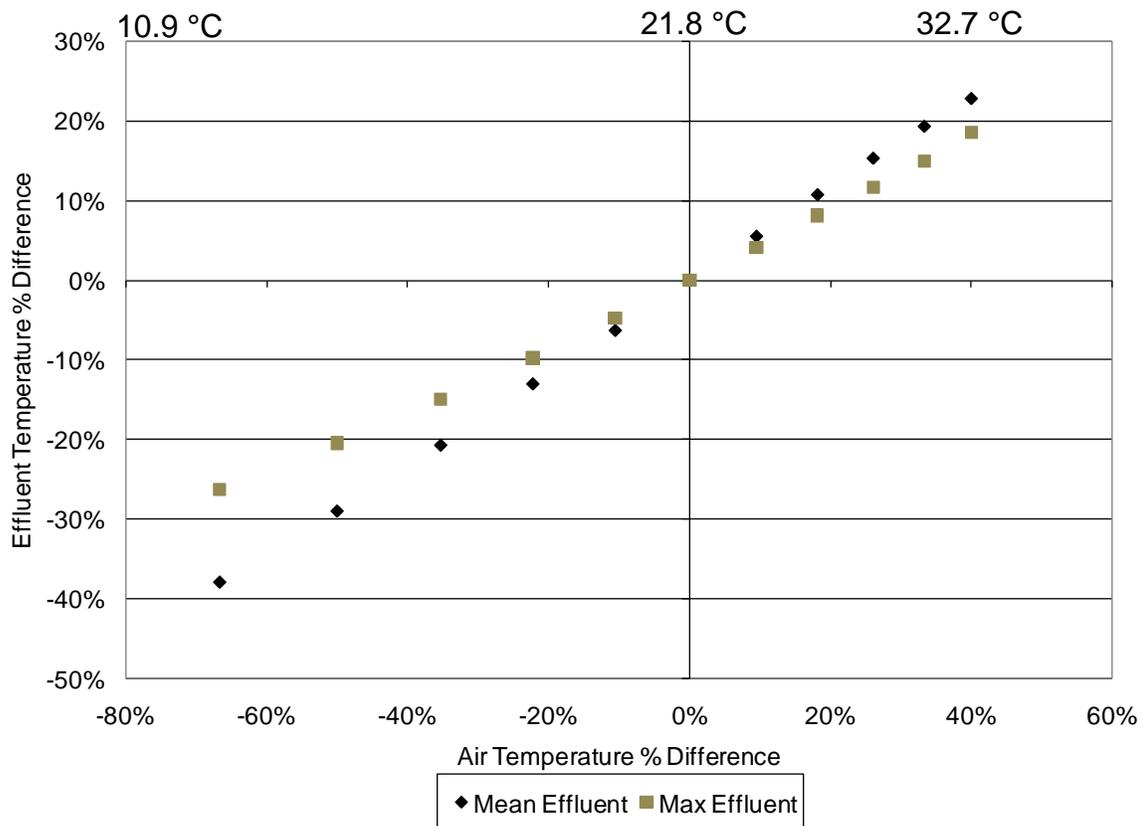


**Figure 6.12** Effect of bioretention shading on effluent temperatures for the Brevard west location

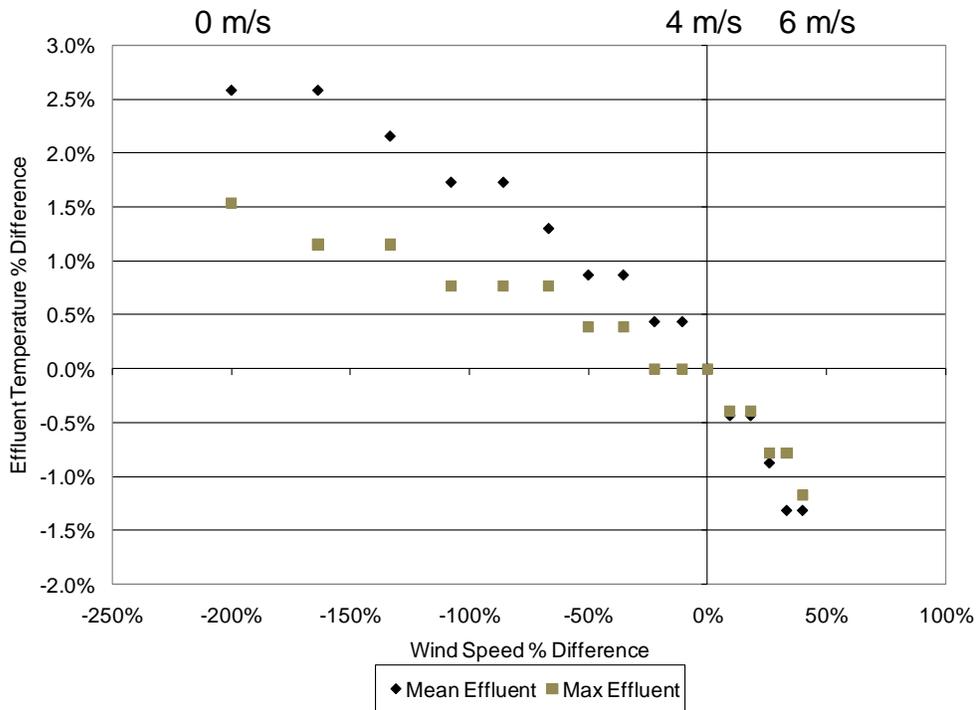


**Figure 6.13** Sensitivity of effluent temperature to radiation adjustments for the Brevard west location

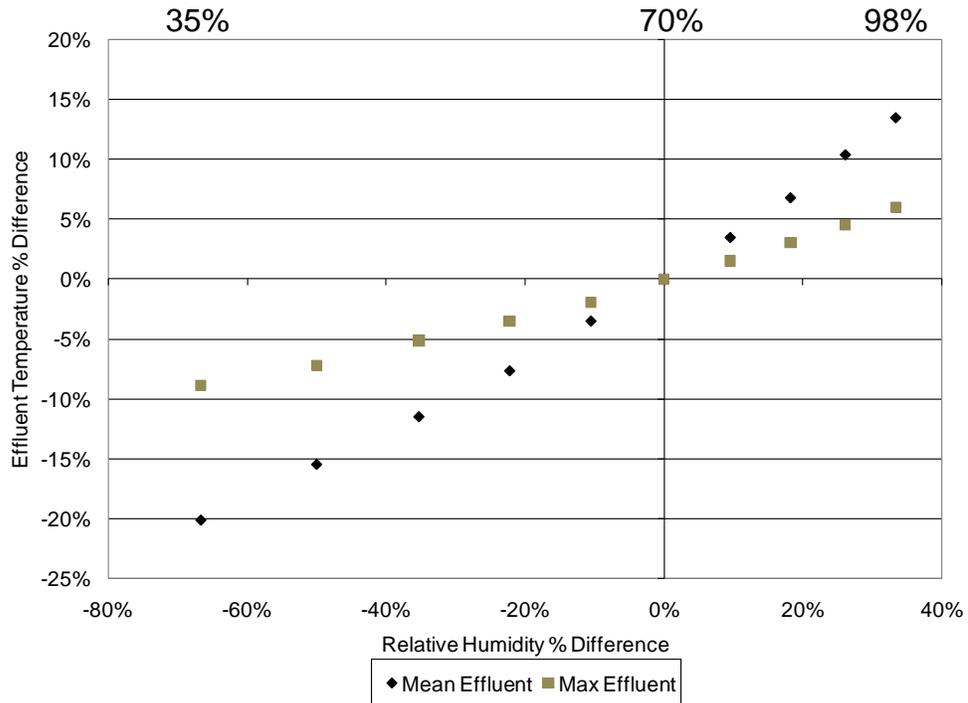
Air temperature, wind speed, and relative humidity during the storm event also affected the calculation of runoff and soil surface temperatures. Because runoff temperatures were calculated as the average of air temperature and pavement surface temperature, increasing the air temperature increased runoff temperatures and consequently effluent temperatures (Figure 6.14). Increasing the wind speed and decreasing the relative humidity resulted in decreased effluent temperatures, corresponding to an increase in evaporative cooling at the soil and pavement surface during the storm events (Figure 6.15, Figure 6.16).



**Figure 6.14** Sensitivity of effluent temperature to air temperature adjustments for the Lenoir location

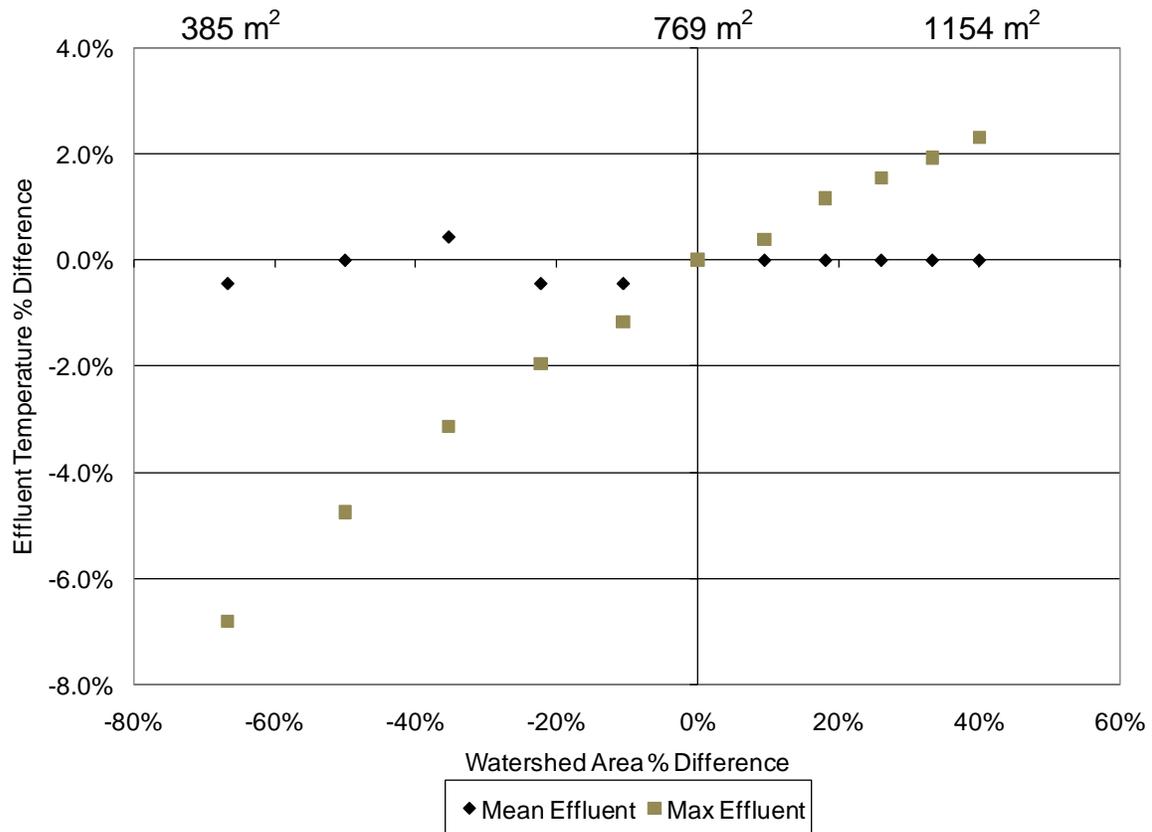


**Figure 6.15** Sensitivity of effluent temperature to wind speed adjustments for the Lenoir location



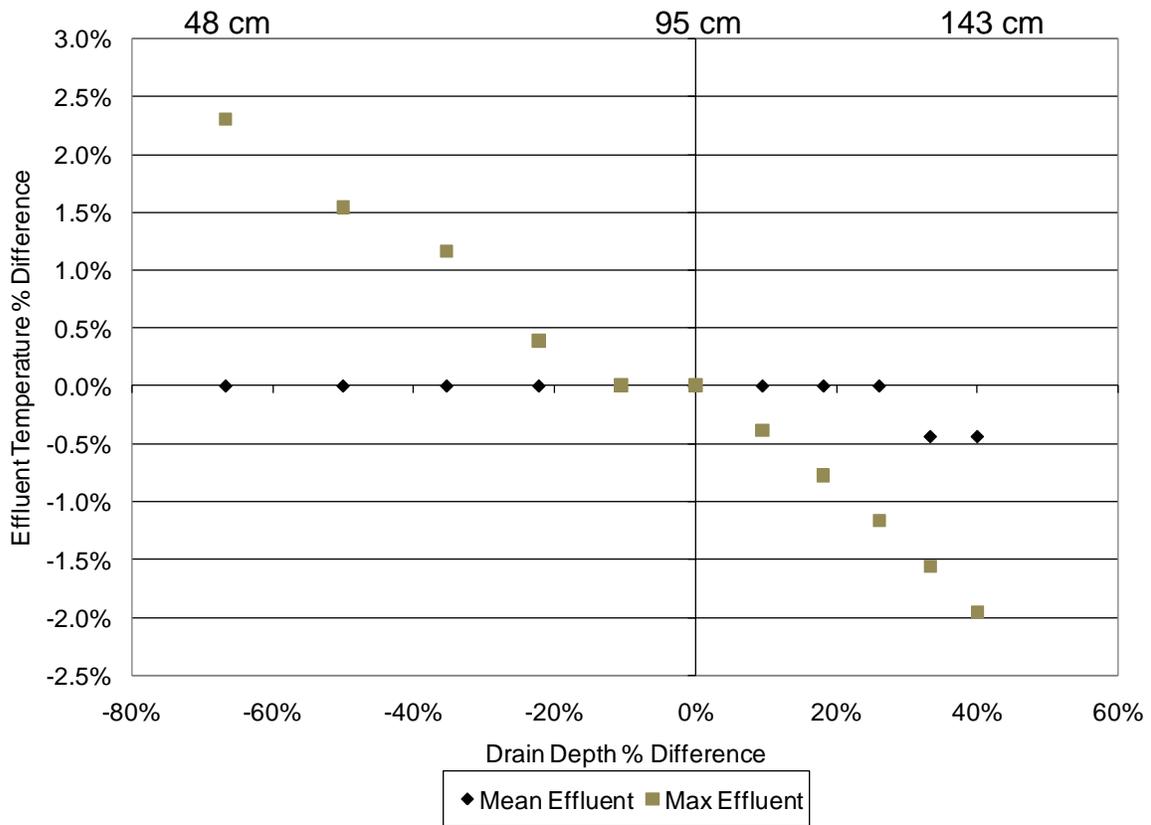
**Figure 6.16** Sensitivity of effluent temperature to relative humidity adjustments for the Lenoir location

The effect of watershed area on effluent temperatures was substantial at the Lenoir location, but minimal for the Brevard bioretention area. Changes in maximum effluent temperatures were greater than changes in mean effluent temperatures (Figure 6.17). There were likely two reasons for the differences in sensitivity between these two locations. Increasing the watershed area should increase flow through the bioretention area, effectively reducing the thermal buffering capacity of the bioretention area during a storm event. Because the storm event from the Lenoir simulation had a greater rainfall depth, the increase in runoff flow was proportionally greater than the Brevard west location. Also, the thermal buffering capacity of the Lenoir bioretention area was already greater than the Brevard west location due to the increased soil depth.



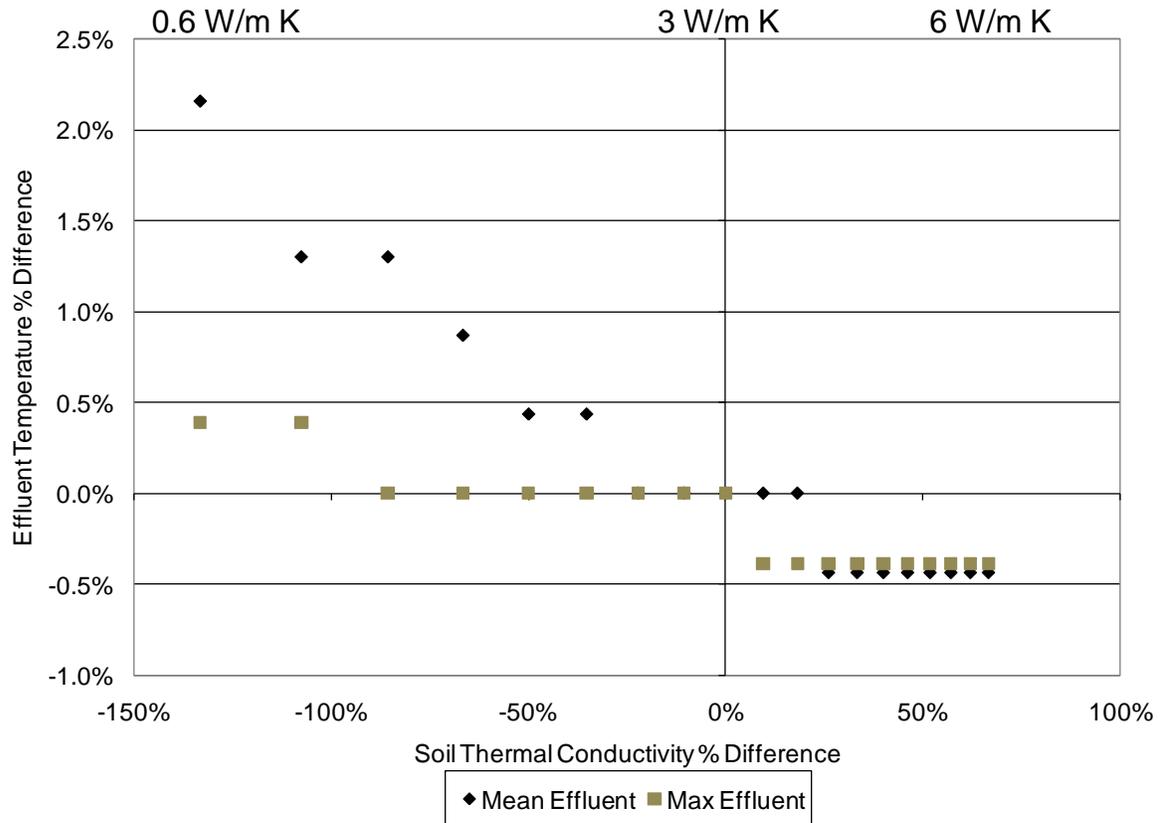
**Figure 6.17** Sensitivity of effluent temperature to watershed area adjustments for the Lenoir location

For both the Lenoir and Brevard west simulations, increasing the drain depth resulted in cooler effluent temperatures (Figure 6.18). Drain depth changes had a more pronounced impact on maximum effluent temperatures than mean values. Typically, the warmest runoff and coolest soil temperatures were observed during the beginning of a storm event. Because soil temperatures generally decreased with depth, a deep system was most effective at reducing maximum storm temperatures. Over the course of the storm, the bioretention soil and infiltrating runoff approached thermal equilibrium at all depths, reducing the impact of drain depth on mean effluent temperatures. These sensitivity analysis results corresponded to results from the bioretention monitoring study.



**Figure 6.18** Sensitivity of effluent temperature to drain depth adjustments for the Lenoir location

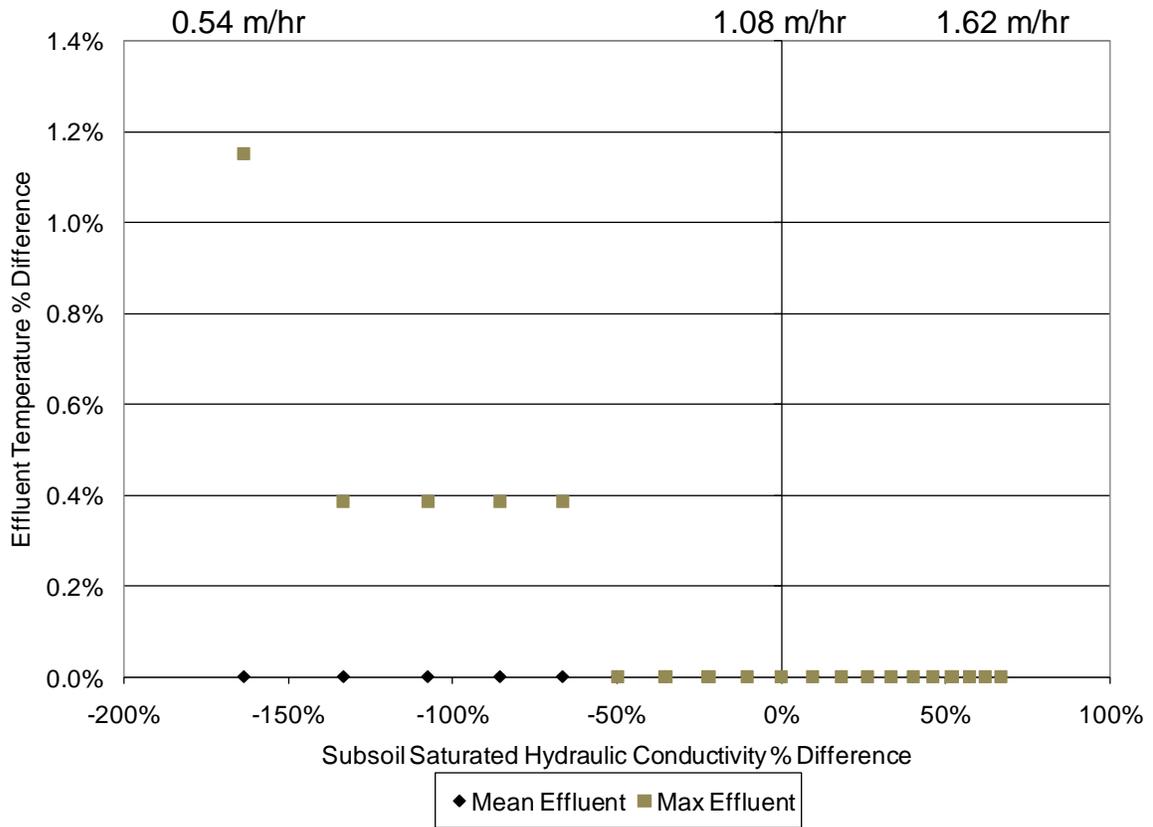
For both sites, decreased soil thermal conductivity resulted in increased effluent temperatures (Figure 6.19). Increased effluent temperatures were likely the result of the reduced effect of evaporative cooling on soil temperatures throughout the profile.



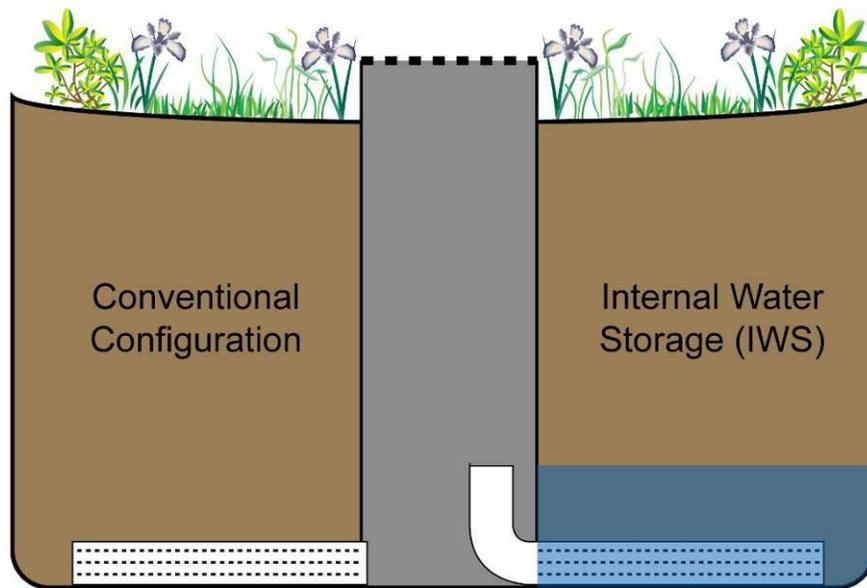
**Figure 6.19** Sensitivity of effluent temperature to soil thermal conductivity adjustments at the Lenoir bioretention area

Decreasing the saturated hydraulic conductivity of the subsoil resulted in increased maximum effluent temperatures (Figure 6.20). When subsoil saturated hydraulic conductivity was substantial, the warmest infiltrating water at the beginning of a storm event was able to seep into the subsoil before the water level within the gravel drainage layer reached the drain pipes. An internal water storage layer, created by an upturned elbow on the underdrain pipes, was expected to have a similar result (Figure 6.21). With an internal water storage layer, the warmest runoff

at the beginning of the storm event would either seep from the bottom of the system or mix with cooler water before being discharged.



**Figure 6.20** Sensitivity of effluent temperature to subsoil saturated hydraulic conductivity adjustments for the Lenoir location



**Figure 6.21** Cross section of a bioretention area with an internal water storage layer

In general, simulations showed that the ability of a bioretention area to reduce runoff volumes had a much larger impact on thermal load reductions than the ability of a bioretention area to reduce runoff temperatures. At times, thermal load reductions due to effluent volume decreases exceeded 40%, while thermal load reductions due to temperature decreases were often below 5%. Volume reductions were greatest at large bioretention areas, which was exemplified when none of the monitored storm events at the Asheville bioretention area, the largest system monitored, met validation criteria due to the lack of saturated flow. As expected, storms with low rainfall intensities or rainfall depths did not generate substantial outflow in simulations, effectively eliminating the thermal load as water seeped into the bioretention subsoil. Increasing the saturated hydraulic conductivity of the underlying soil and height of the underdrain above the subsoil in model simulations also resulted in less bioretention effluent. For example, increasing the elevation of the underdrain from 7 cm to 15 cm for the Brevard west sensitivity simulation resulted in a 10% further reduction in the effluent thermal load. Similarly doubling the saturated hydraulic conductivity of the subsoil from 6 cm/hr to 12 cm/hr for the same sensitivity simulation increased the thermal load reduction from 44% to 76%,

which was directly related to the increased volume reduction. These results indicate the important influence of design parameters related to bioretention volume reductions on effluent thermal loads and the need for further research to quantify these impacts. The saturated hydraulic conductivity of the underlying soil cannot be easily specified by a designer, but could be considered when selecting the location of a bioretention area. A designer could increase the elevation of the underdrains within the gravel drainage layer, providing more storage volume for water to seep into the subsoil. Similarly, implementing an internal water storage layer would likely decrease effluent volumes by effectively increasing the storage capacity within the system before water is discharged from the underdrain. Due to problems with the flow monitoring equipment discussed in Chapters 2, 3, and 4, it was not possible to validate runoff volume reductions. The hydraulic modelling used in this effort was similar to the approaches utilized by Heasom et al. (2006) and Dussailant et al. (2003), who both concluded that Green-Ampt based approaches described bioretention hydraulics with reasonable accuracy. All monitored rainfall events that failed to generate measurable outflow likewise failed to generate outflow in model simulations.

It is important to note that thermal load reductions alone are not necessarily the best indication of bioretention performance, since effluents cooler than the trout temperature threshold would still have a large thermal load, but would not negatively impact the coldwater stream environment. In order to better understand the effect of urban stormwater runoff on coldwater stream environments, it is necessary to simulate the effect of BMP effluent on stream temperatures. This effort requires an understanding of stream flows and temperature distributions beyond the scope of the current modelling effort, but does merit further investigation.

Analysis has shown that a number of model input parameters can have substantial impacts on effluent temperature calculations (Table 6.6). The sensitive model inputs include parameters that can be controlled by a bioretention designer,

such as contributing watershed size and drain depth, as well as parameters that cannot be easily controlled, such as weather parameters and soil properties. The bioretention thermal model should allow designers to determine the optimum system configuration to reduce effluent temperatures by adjusting input parameters they can specify, while accounting for factors they have no control over. The high sensitivity of model outputs to a number of weather related input parameters indicated that simulations should be conducted for a variety of storm events and weather characteristics to better estimate the true performance of a bioretention system.

**Table 6.6** Summary of sensitivity analysis results

<b>Input Parameter</b>	<b>Effect of Input Increase on Effluent Temperature</b>	<b>Sensitivity of Effluent Temperature Outputs<sup>1</sup></b>
Transmittance	Increase	5-10%
Shading	Decrease	0-5%
Radiation	Increase	0-5%
Air Temperature	Increase	40-50%
Wind Speed	Increase	0-5%
Relative Humidity	Increase	10-20%
Watershed Area	Increase	5-10%
Drain Depth	Decrease	0-5%
Soil Thermal Conductivity	Decrease	0-5%

1. Range of maximum percent difference in effluent temperature when input parameter was adjusted within the expected range

### **References**

- de Vries, D. A. 1966. Chapter 7: Thermal Properties of Soils. In *Physics of Plant Environment*, 210-235. ed. W. R. Van Wijk, Amsterdam: North-Holland Publishing Company.
- Dussailant, A., K. Cozzetto, K. Brander and K. Potter. 2003. Green-Ampt Model of a Rain Garden and Comparison to Richards Equation Model. *Sustainable World* 6891.
- Heasom, W., R. G. Traver and A. Welker. 2006. Hydrologic Modeling of a Bioinfiltration Best Management Practice. *Journal of the American Water Resources Association* 42(5): 1329-1347.
- Hsu, C. T. 1999. A Closure Model for Transient Heat Conduction in Porous Media. *Journal of Heat Transfer* 121(3): 733-739.

Ochsner, T. E., R. Horton and T. Ren. 2001. A New Perspective on Soil Thermal Properties. *Soil Science Society of America Journal* 65(6): 1641-1647.

Soil Survey Staff, Natural Resources Conservation Service, United States Department of Agriculture. (2008). "Web Soil Survey."  
<<http://websoilsurvey.nrcs.usda.gov/>> (09/28/2008).

Spokas, K. and F. Forcella. 2006. Estimating Hourly Incoming Solar Radiation from Limited Meteorological Data. *Weed Science* 54(1): 182-189.

## **APPENDIX A**

### **Bioretention Thermal Model: Equations and Derivations**

## **Green-Ampt Infiltration Equations (Chu and Mariño, 2005)**

### **Pre-Ponding Conditions**

The depth of the wetting front was calculated based on the previous wetting front depth and rate of inflow.

$$z = z_{m-1} + \frac{r\Delta t}{\theta_f} \quad (\text{A-1})$$

At each time step, the infiltration capacity was calculated based on the wetting front position and surrounding soil properties to determine if the rainfall rate exceeded the infiltration capacity of the soil, which would trigger ponded conditions.

$$i = \frac{z + h_{sn}}{\sum_{j=1}^{n-1} \frac{z_j - z_{j-1}}{K_j} + \frac{z - z_{n-1}}{K_n}} \quad (\text{A-2})$$

If  $r_k > i$ , then begin ponded calculations

### **Conditions at Ponding**

The time for the wetting front to reach a specified depth under ponded conditions could be calculated based on the infiltration capacity of the soil.

$$t_z = t_{z_{n-1}} + \frac{\theta_{fn}}{K_n} (z - z_{n-1}) + \theta_{fn} \left[ \sum_{j=1}^{n-1} z_j \left( \frac{1}{K_j} - \frac{1}{K_{j+1}} \right) - \frac{h_{sn}}{K_n} \right] \ln \left( \frac{z + h_{sn}}{z_{n-1} + h_{sn}} \right) \quad (\text{A-3})$$

The specific position of the wetting front at the time of ponding was calculated by finding the position of the wetting front necessary for the infiltration capacity to equal the rainfall rate.

$$z_p = \frac{K_{n_p} h_{sn_p} + r_{m_p} z_{n_p-1} - r_{m_p} K_{n_p} \sum_{j=1}^{n_p-1} \frac{z_j - z_{j-1}}{K_j}}{r_{m_p} - K_{n_p}} \quad (\text{A-4})$$

The specific time when ponding occurred was calculated based on the cumulative infiltration and current rainfall rate.

$$t_p = \frac{1}{r_{m_p}} \left( I_p - \sum_{k=1}^{m_p-1} r_k \Delta t \right) + (m_p - 1) \Delta t \quad (\text{A-5})$$

The equivalent time for the wetting front to reach its current position under ponded conditions was calculated using Equation A-3 in conjunction with the wetting front position at the time of ponding.

$$t_{pd} = t_{z_{n_p-1}} + \frac{\theta_{fn_p}}{K_{n_p}} (z_p - z_{n_p-1}) + \theta_{fn_p} \left[ \sum_{j=1}^{n_p-1} z_j \left( \frac{1}{K_j} - \frac{1}{K_{j+1}} \right) - \frac{h_{sn_p}}{K_{n_p}} \right] \times \ln \left( \frac{z_p + h_{sn_p}}{z_{n_p-1} + h_{sn_p}} \right) \quad (\text{A-6})$$

### Post-Ponding Conditions

The position of the wetting front was calculated based on the previous position of the wetting front, the infiltration capacity of the soil, and a time shift to account for the infiltration before ponding occurred.

$$z_b = z_{b-1} - \frac{t_{z_{n-1}} - (t - t_p + t_{pd}) + \frac{\theta_{fn}}{K_n} (z_{b-1} - z_{n-1}) + \theta_{fn} \left[ \sum_{j=1}^{n-1} z_j \left( \frac{1}{K_j} - \frac{1}{K_{j+1}} \right) - \frac{h_{sn}}{K_n} \right] \ln \left( \frac{z_{b-1} + h_{sn}}{z_{n-1} + h_{sn}} \right)}{\frac{\theta_{fn} \left[ \sum_{j=1}^{n-1} z_j \left( \frac{1}{K_j} - \frac{1}{K_{j+1}} \right) - \frac{h_{sn}}{K_n} \right]}{z_{b-1} + h_{sn}} + \frac{\theta_{fn}}{K_n}} \quad (\text{A-7})$$

The infiltration capacity of the soil was calculated by itself and compared to the inflow rate to determine if ponding conditions persisted.

$$i = \frac{z_b + h_{sn}}{\sum_{j=1}^{n-1} \frac{z_j - z_{j-1}}{K_j} + \frac{z_b - z_{n-1}}{K_n}} \quad (\text{A-8})$$

### Notation

$z$  = wetting front position (m)

$r$  = rainfall rate (m/hr)

$\Delta t$  = timestep (hr)

$\theta_f$  = fillable porosity (cm<sup>3</sup>/cm<sup>3</sup>)

$i$  = infiltration capacity (m/hr)

$h_s$  = suction head in advance of the wetting front (m)

$K$  = saturated hydraulic conductivity (m/hr)

$z_p$  = wetting front position at time of ponding

$t_p$  = ponding time (hr)

$t_{pd}$  = equivalent time for wetting front to reach ponding depth under initially ponded conditions

$t_0$  = initial time when ponded condition is assumed

$k$  = time index

$n$  = soil layer index

$m$  = wetting front position index

$b$  = iteration index

## **Runoff Temperature Calculations (Van Buren et al., 2000)**

### **Dry Weather Conditions**

#### **Finite Difference Equation for Conduction in Pavement Profile**

Temperatures within the pavement profile were calculated at each simulation time step based on a finite difference solution to an equation for transient heat conduction.

$$T_m^{t+\Delta t} = T_m^t \left( 1 - \frac{2\alpha\Delta t}{\Delta z^2} \right) + \frac{\alpha\Delta t}{\Delta z^2} (T_{m+1}^t + T_{m-1}^t) \quad (\text{A-9})$$

#### **Surface Boundary Governing Equation**

A surface heat balance was used to calculate the surface boundary condition, accounting for convective heat transfer with the overlying air and incoming solar radiation.

$$\left. \frac{\partial T}{\partial z} \right|_{z=0} = \frac{h_c}{k} (T_s - T_{air}) - \frac{q_{rad}}{k} \quad (\text{A-10})$$

#### **Surface Boundary Equation**

Discretizing and solving for  $T_s$  yields the surface boundary equation used in model calculations.

$$T_s = \frac{\alpha T_{bs} + (q_{rad} + h_c T_{air}) \Delta z}{\alpha + h_c \Delta z} \quad (\text{A-11})$$

### Convective Coefficient Calculation (Barber, 1957)

The convective heat transfer coefficient was calculated based on wind speed using an empirical equation developed by Barber (1957) for asphalt pavement surfaces.

$$h_c = 5.68(1.3 + 3.35U_w^{0.75}) \quad (A-12)$$

### Wet Weather Conditions

#### Surface Boundary Governing Equation

A surface heat balance was used to calculate the surface boundary condition, accounting for convective heat transfer with the overlying runoff, incoming solar radiation, and evaporation from the surface.

$$\left. \frac{\partial T}{\partial z} \right|_{z=0} = \frac{h_c}{k} (T_s - T_{ro}) - \frac{q_{rad}}{k} + \frac{q_{evap}}{k} \quad (A-13)$$

#### Surface Boundary Equation

Discretizing and solving for  $T_s$  yields the surface boundary equation used in model calculations.

$$T_s = \frac{\alpha T_{bs} + (q_{rad} - q_{evap} + h_c T_{ro}) \Delta z}{\alpha + h_c \Delta z} \quad (A-14)$$

### Runoff Temperature Equation

Runoff temperature was calculated as the average of the pavement surface temperature and temperature of the incoming rainfall, which was approximated as the air temperature.

$$T_{ro} = \frac{T_S + T_{rain}}{2} \quad (A-15)$$

### Convective Coefficient Calculation

The convective coefficient during a storm event was calculated based on an empirical relationship developed by Van Buren et al. (2000) for runoff flow over asphalt pavement based on monitoring data.

$$h_c = 3.46(i)^{1.1} \quad (A-16)$$

### Evaporation Heat Flux Calculation

The evaporation heat flux was calculated as a combination of the latent heat and sensible heat transfer.

$$q_{evap} = q_v + q_{sen} \quad (A-17)$$

### Latent Heat Flux Calculation

The latent heat flux was calculated based on the density of water, latent heat of vaporization, and evaporation rate.

$$q_v = \rho H_v EV \quad (A-18)$$

### Latent Heat of Vaporization Calculation

Van Buren et al. (2000) calculated the latent heat of vaporization based on an expression presented by Linsley et al. (1958).

$$H_v = 862 \left[ 597 - 0.56(T_{ro}) \right] \quad (\text{A-19})$$

### Evaporation Rate Calculation

Van Buren et al. (2000) estimated the evaporation rate using Meyer's equation, which was presented in Chow (1964).

$$EV = 3.59 \times 10^{-7} (1 + 0.0447U_w) e_{sa} (1 - f) \quad (\text{A-20})$$

### Sensible Heat Flux Calculation

The sensible heat flux was calculated based on the latent heat flux and the Bowen ratio (Bowen, 1926).

$$q_{sen} = q_v \left[ 6.1 \times 10^{-4} p_{atm} \left( \frac{T_{ro} - T_{air}}{e_{sa} (1 - f)} \right) \right] \quad (\text{A-21})$$

### Notation

$T$  = temperature (°C)

$z$  = depth from surface (m)

$h_c$  = convective coefficient (W/m<sup>2</sup> °C)

$k$  = solid thermal conductivity (W/m °C)

$T_s$  = temperature of asphalt surface (°C)

$T_{air}$  = temperature of air above asphalt (°C)

$q_{rad}$  = radiation heat flux (W/m<sup>2</sup>)

$t$  = time index

$m$  = depth index

$\alpha$  = thermal diffusivity of the solid (m<sup>2</sup>/s)

$U_w$  = wind speed (m/s)

$q_{evap}$  = evaporation heat flux (W/m<sup>2</sup>)

$T_{ro}$  = temperature of runoff (°C)

$T_{rain}$  = temperature of rainfall (°C)

$i$  = rainfall intensity (mm/hr)

$q_v$  = latent heat flux (W/m<sup>2</sup>)

$q_{sen}$  = sensible heat flux (W/m<sup>2</sup>)

$\rho$  = density of water (kg/m<sup>3</sup>)

$H_v$  = latent heat of vaporization (J/kg)

$EV$  = rate of evaporation (m/s)

$e_{sa}$  = saturation vapor pressure (kPa)

$f$  = relative humidity

$P_{atm}$  = atmospheric pressure (kPa)

### **Air Temperature Calculations (Satyamurty and Babu, 1999)**

Air temperatures were estimated using separate empirical equations for different periods of the day.

#### **Air Temperature Before 17:00**

$$T = \frac{1}{2}(T_{\max} + T_{\min}) - \frac{1}{2}(T_{\max} - T_{\min}) \text{Cos}\left(2\pi\bar{T}\right)^{\frac{\ln(2)}{\ln\left(\frac{24}{T_{\max} - T_{\min}}\right)}} - 0.22 \quad (\text{A-22})$$

#### **Air Temperature After 17:00**

$$T = \frac{1}{2}(T_{\max} + T_{\min}) - \frac{1}{2}(T_{\max} - T_{\min}) \text{Cos}\left(2\pi\bar{T}\right)^{\frac{\ln(2)}{\ln\left(\frac{24}{T_{\max} - T_{\min}}\right)}} - 1.25 \quad (\text{A-23})$$

$$\bar{T} = \frac{t - t_{\min temp}}{24} \quad (\text{A-24})$$

#### **Notation**

$T$  = air temperature (°C)

$\bar{T}$  = adjusted time

$t$  = time (hr)

### **Solar Radiation Calculations (Spokas and Forcella, 2006)**

The radiation heat flux was a combination of direct and diffuse radiation.

$$R = S_b + S_d \quad (\text{A-25})$$

The zenith and solar declination angles were calculated based on the predictable movement of the sun, presented by Campbell and Norman (1998).

$$\Psi = \arccos\left(\sin(\Phi)\sin(\delta_{SD}) + \cos(\Phi)\cos(\delta_{SD})\cos\left[0.0833\pi(t - t_{sn})\right]\right) \quad (\text{A-26})$$

$$\delta_{SD} = \arcsin\left(0.39785 \sin\left[4.869 + 0.0172J + 0.03345 \sin(6.2238 + 0.0172J)\right]\right) \quad (\text{A-27})$$

The beam radiation was calculated using a model presented by Liu and Jordan (1960) that accounted for atmospheric transmittance.

$$S_b = S_{b_0} \tau^m \quad (\text{A-28})$$

The optical mass number, atmospheric pressure, and diffuse radiation were estimated based on relationships presented in Campbell and Norman (1998).

$$m = \frac{P_a}{101.3(\cos \Psi)} \quad (\text{A-29})$$

$$P_a = 101.3e^{-(a/8200)} \quad (\text{A-30})$$

$$S_d = 0.30(1 - \tau^m) S_{p_0} \cos \Psi \quad (\text{A-31})$$

### Notation

$R$  = radiation heat flux (W/m<sup>2</sup>)

$S_b$  = direct beam radiation (W/m<sup>2</sup>)

$S_d$  = diffuse solar radiation (W/m<sup>2</sup>)

$\Psi$  = zenith angle (radians)

$\Phi$  = site latitude (radians)

$\delta_{SD}$  = solar declination angle (radians)

$t$  = current time (hr)

$t_{sn}$  = time of solar noon (hr)

$J$  = calendar day (January 1 = 1, December 31 = 365 or 366)

$S_{b_0}$  = solar constant (1360 W/m<sup>2</sup>)

$\tau$  = atmospheric transmittance

$m$  = optical mass number

$P_a$  = atmospheric pressure (kPa)

$a$  = elevation of the site (m)

### **Antecedent Soil Temperature Profile Calculations (Elias et al., 2004)**

The antecedent soil temperature profile was calculated using an exponential-sinusoidal one-dimensional analytical model.

$$T(z, t) = T_{ay} + A_y \exp\left(\frac{-z}{D_y}\right) \sin\left(\omega_y t - \frac{z}{D_y} + \phi_d\right) + A_d \exp\left(\frac{-z}{D_d}\right) \sin\left(\omega_d t - \frac{z}{D_d} + \phi_d\right) + \left(\frac{B}{2}\right) \exp\left(\frac{-z}{D'}\right) \sin\left(\omega' t - \frac{z}{D'} + \phi_d - \beta + \frac{\pi}{2}\right) - \left(\frac{B}{2}\right) \exp\left(\frac{-z}{D''}\right) \sin\left(\omega'' t - \frac{z}{D''} + \phi_d + \beta + \frac{\pi}{2}\right) \quad (\text{A-32})$$

#### Notation

$T$  = soil temperature (°C)

$z$  = depth below soil surface (m)

$t$  = length of time between beginning of year and current time (s)

$T_a$  = average surface temperature (°C)

$A$  = temperature amplitude (°C)

$D$  = damping depth (m)

$\omega$  = radial frequency (rad/s)

$\phi$  = phase constant (rad)

$B$  = constant related to temporal variation of daily amplitude

$D'$ ,  $D''$  = damping depths for additional temperature waves (m)

$\omega'$ ,  $\omega''$  = radial frequencies for additional temperature waves (rad/s)

$\beta$  = phase constant for temporal variation of daily amplitude (rad)

Subscripts:

$y$  = annual

$d$  = daily

### **Thermal Infiltration Calculations (Nakayama et al., 2001)**

#### **Governing Equations**

The thermal infiltration equations accounted for conduction and convection heat transfer mechanisms as well as the mass movement of fluid through the soil system.

#### **Fluid Energy Equation**

$$\rho_F C_{pF} \left[ \varepsilon \frac{\partial \langle T \rangle^F}{\partial t} + \langle \vec{u} \rangle \frac{\partial \langle T \rangle^F}{\partial z} \right] =$$
$$(\varepsilon + G(1 - \sigma)) k_F \frac{\partial^2 \langle T \rangle^F}{\partial z^2} + \bar{k}_{dis} \frac{\partial^2 \langle T \rangle^F}{\partial z^2} + a_{sf} h_{sf} (\langle T \rangle^S - \langle T \rangle^F) - k_S G \left( \frac{\partial^2 \langle T \rangle^S}{\partial z^2} - \frac{\partial^2 \langle T \rangle^F}{\partial z^2} \right) \quad (\text{A-33})$$

### Solid Energy Equation

$$(1-\varepsilon)\rho_s C_s \frac{\partial \langle T \rangle^s}{\partial t} =$$

$$(1-\varepsilon + G(\sigma-1))k_s \frac{\partial^2 \langle T \rangle^s}{\partial z^2} - a_{sf} h_{sf} (\langle T \rangle^s - \langle T \rangle^F) + k_s G \left( \frac{\partial^2 \langle T \rangle^s}{\partial z^2} - \frac{\partial^2 \langle T \rangle^F}{\partial z^2} \right) \quad (\text{A-34})$$

$$\sigma = \frac{k_s}{k_f} \quad (\text{A-35})$$

The tortuosity parameter effectively reduced the conduction through the soil profile due to the structure of the porous media and was calculated based on work presented by Hsu (1999).

$$G = \frac{\left( \frac{k_{stg}}{k_f} \right) - \varepsilon - (1-\varepsilon)\sigma}{(\sigma-1)^2} \quad (\text{A-36})$$

The thermal dispersion tensor and interfacial heat transfer coefficient were calculated based on empirical expressions presented by Wakao and Kaguei (1982).

$$\bar{k}_{dis} = k_f \left[ \frac{\rho_F C_{pF} \left| \langle \vec{u} \rangle \right| d_p}{2k_F} \right] \quad (\text{A-37})$$

$$h_{sf} = \frac{k_F}{d_p} \left[ 2 + 1.1 \text{Pr}_F^{1/3} \left( \frac{\rho_F \langle \vec{u} \rangle d_p}{\mu_F} \right)^{0.6} \right] \quad (\text{A-38})$$

### Discretized Thermal Infiltration Equations

The governing equations were discretized using a finite difference scheme presented by Leonard (1979).

### Fluid Energy Equation

$$\begin{aligned} \rho_S C_{P_F} \varepsilon \frac{\langle T_{t+1}^F \rangle - \langle T \rangle^F}{\Delta t} + \rho_F C_{P_F} \langle \vec{u} \rangle \frac{(T_r^F - T_l^F)}{\Delta z_0} = \\ (\varepsilon + G(1-\sigma)) k_F \left( \frac{T_L^F + T_R^F - 2T_C^F}{\Delta z_0^2} \right) + k_{dis} \left( \frac{T_L^F + T_R^F - 2T_C^F}{\Delta z_0^2} \right) + a_{sf} h_{sf} (\langle T \rangle^S - \langle T \rangle^F) \\ - k_S G \left( \frac{T_L^S + T_R^S - 2T_C^S}{\Delta z_0^2} - \frac{T_L^F + T_R^F - 2T_C^F}{\Delta z_0^2} \right) \end{aligned} \quad (\text{A-39})$$

$$T_r = \frac{1}{2}(T_C + T_R) - \frac{1}{8}(T_L + T_R - 2T_C) \quad (\text{A-40})$$

$$T_l = \frac{1}{2}(T_L + T_C) - \frac{1}{8}(T_L + T_R - 2T_C) \quad (\text{A-41})$$

### Solid Energy Equation

$$\rho_S C_{P_S} (1 - \varepsilon) \frac{\langle T_{t+1} \rangle^S - \langle T \rangle^S}{\Delta t} = \quad (A-42)$$
$$(1 - \varepsilon + G(\sigma - 1)) k_S \left( \frac{T_L^S + T_R^S - 2T_C^S}{\Delta z_0^2} \right) - a_{sf} h_{sf} (\langle T \rangle^S - \langle T \rangle^F) + k_S G \left( \frac{T_L^S + T_R^S - 2T_C^S}{\Delta z_0^2} - \frac{T_L^F + T_R^F - 2T_C^F}{\Delta z_0^2} \right)$$

### Notation

$\rho$  = density (kg/m<sup>3</sup>)

$C_p$  = specific heat capacity (kJ/kg K)

$\varepsilon$  = porosity

$T$  = temperature (K)

$t$  = time (s)

$\langle \vec{u} \rangle$  = fluid darcian velocity (m/s)

$z$  = depth below soil surface (m)

$G$  = tortuosity parameter

$\sigma$  = thermal conductivity ratio

$k$  = thermal conductivity (W/m K)

$\bar{k}_{dis}$  = thermal dispersion tensor

$a_{sf}$  = interfacial area ( $\text{m}^2/\text{m}^3$ )

$h_{sf}$  = interfacial heat transfer coefficient ( $\text{W}/\text{m}^2 \text{K}$ )

$k_{stg}$  = thermal conductivity of stagnant saturated soil ( $\text{W}/\text{m K}$ )

$d_p$  = particle diameter (m)

$\text{Pr}$  = prandtl number

$\mu$  = dynamic viscosity ( $\text{N s}/\text{m}^2$ )

Superscripts and Subscripts:

$F$  = fluid phase

$S$  = solid phase

$t$  = time index

$r$  = interpolated downstream node

$l$  = interpolated upstream node

$L$  = upstream node

$R$  = downstream node

$C$  = current node

## Thermal Infiltration Equations Derivation

### Solid Energy Equation

$$\rho_s C_{ps} \frac{\partial T_s}{\partial t} = \nabla \cdot (k_s \nabla T) \quad (\text{A-43})$$

Integrate over an elementary control volume following procedure described by Nakayama (1995):

$$\rho_s C_{ps} \frac{\partial (1-\varepsilon) \langle T \rangle^s}{\partial t} = \nabla \cdot [k_s \nabla (1-\varepsilon) \langle T \rangle^s - \frac{1}{V} \int_{A_{\text{int}}} k_s T dA] - \frac{1}{V} \int_{A_{\text{int}}} k_f \nabla T dA \quad (\text{A-44})$$

Based on closure modelling presented by Hsu (1999)

$$G(\nabla \langle T \rangle^F - \sigma \nabla \langle T \rangle^S) = \frac{1}{V} \int_{A_{\text{int}}} T_F dA = \frac{1}{V} \int_{A_{\text{int}}} T_S dA \quad (\text{A-45})$$

$$h_{fs} a_{fs} (\langle T \rangle^S - \langle T \rangle^F) = \frac{1}{V} \int_{A_{\text{int}}} k_f \nabla T_F dA = \frac{1}{V} \int_{A_{\text{int}}} k_s \nabla T_S dA \quad (\text{A-46})$$

Substituting yields:

$$(1-\varepsilon) \rho_s C_{ps} \frac{\partial \langle T \rangle^S}{\partial t} = k_s \nabla^2 \langle T \rangle^S - \varepsilon k_s \nabla^2 \langle T \rangle^S - \nabla k_s G(\nabla T_f - \sigma \nabla T_s) - h_{fs} a_{fs} (\langle T \rangle^S - \langle T \rangle^F) \quad (\text{A-47})$$

Rearranging terms yields:

$$(1-\varepsilon)\rho_s C_{ps} \frac{\partial \langle T \rangle^S}{\partial t} = \nabla[(1-\varepsilon + (\sigma-1)G)k_s \nabla \langle T \rangle^S] - (a_{sf} h_{sf} - k_s G \nabla^2)(\langle T \rangle^S - \langle T \rangle^F) \quad (\text{A-48})$$

### Fluid Energy Equation

$$\rho_f C_{pf} \frac{\partial T_f}{\partial t} + \nabla(\rho_f C_{pf} T_f u) = \nabla(k_f \nabla T) \quad (\text{A-49})$$

Integrate over an elementary control volume following procedure described by Nakayama (1995):

$$\begin{aligned} \rho_f C_{pf} \left( \frac{\partial \varepsilon \langle T \rangle^F}{\partial t} + \langle u \rangle \nabla \langle T \rangle^F \right) &= \nabla(k_f \nabla \varepsilon \langle T \rangle^F) + \frac{1}{V} \int_{A_{\text{int}}} k_f T dA - \rho_f C_{pf} \langle T' u' \rangle \\ &+ \frac{1}{V} \int_{A_{\text{int}}} k_f \nabla T dA \end{aligned} \quad (\text{A-50})$$

Substituting results from closure modelling yields:

$$\begin{aligned} \rho_f C_{pf} \left( \frac{\partial \varepsilon \langle T \rangle^F}{\partial t} + \langle u \rangle \nabla \langle T \rangle^F \right) &= \\ k_f \varepsilon \nabla^2 \langle T \rangle^F + \nabla k_f G (\nabla \langle T \rangle^F - \sigma \nabla \langle T \rangle^S) - \nabla \rho_f C_{pf} \langle T' u' \rangle &+ h_{fs} a_{fs} (\langle T \rangle^S - \langle T \rangle^F) \end{aligned} \quad (\text{A-51})$$

Based on porous media heat transfer modelling work presented by Nakayama (1995)

$$\rho_f C_{pf} \langle T' u' \rangle = -\bar{k}_{dis} \nabla \langle T \rangle^F \quad (\text{A-52})$$

Substitution yields:

$$\rho_f C_{pf} \left( \frac{\partial \varepsilon \langle T \rangle^F}{\partial t} + \langle u \rangle \nabla \langle T \rangle^F \right) =$$

$$k_f \varepsilon \nabla^2 \langle T \rangle^F + \nabla k_f G (\nabla \langle T \rangle^F - \sigma \nabla \langle T \rangle^S) + \bar{k}_{dis} \nabla \langle T \rangle^F + h_{fs} a_{fs} (\langle T \rangle^S - \langle T \rangle^F) \quad (\text{A-53})$$

### **References**

- Bowen, I. S. 1926. The Ratio of Heat Loss by Conduction and by Evaporation from any Water Surface. *Physical Review* 27(6): 779-787.
- Campbell, G. S. and J. M. Norman. 1998. Chapter 11: Radiation Fluxes in Natural Environments. In *Introduction to Environmental Biophysics*, 167-184. New York: Springer.
- Chow, V. T. 1964. *Handbook of Applied Hydrology*. New York: McGraw-Hill Inc.
- Chu, X. and M. A. Mariño. 2005. Determination of Ponding Condition and Infiltration into Layered Soils under Unsteady Rainfall. *Journal of Hydrology* 313(3-4): 195-207.
- Elias, E. A., R. Cichota, H. H. Torriani and De Jong Van Lier, Quirijn. 2004. Analytical Soil-Temperature Model: Correction for Temporal Variation of Daily Amplitude. *Soil Science Society of America Journal* 68(3): 784-788.
- Hsu, C. T. 1999. A Closure Model for Transient Heat Conduction in Porous Media. *Journal of Heat Transfer* 121(3): 733-739.
- Leonard, B. P. 1979. A Stable and Accurate Convective Modelling Procedure Based on Quadratic Upstream Interpolation. *Computer Methods in Applied Mechanics and Engineering* 19(1): 59-98.

- Linsley, R. K., M. A. Kohler, and J. L. H. Paulhus. 1958. *Hydrology for Engineers*. New York: McGraw-Hill Inc.
- Liu, B. Y., R. C. Jordan. 1960. The Interrelationship and Characteristic Distribution of Direct, Diffuse, and Total Solar Radiation. *Solar Energy* 4(3): 1-19.
- Nakayama, A. 1995. Chapter 5: Convective Flows in Porous Media. In *PC-Aided Numerical Heat Transfer and Convective Flow*, 103-177. Boca Raton: CRC Press.
- Nakayama, A., F. Kuwahara, M. Sugiyama and G. Xu. 2001. A Two-Energy Equation Model for Conduction and Convection in Porous Media. *International Journal of Heat and Mass Transfer* 44(22): 4375-4379.
- Satyamurty, V. V. and K. Sarath Babu. 1999. Relative Performance of Correlations to Estimate Hourly Ambient Air Temperature and Development of a General Correlation. *International Journal of Energy Research* 23(8): 663-673.
- Spokas, K. and F. Forcella. 2006. Estimating Hourly Incoming Solar Radiation from Limited Meteorological Data. *Weed Science* 54(1): 182-189.
- Van Buren, M.A., W.E. Watt, J. Marsalek and B.C. Anderson. 2000. Thermal Enhancement of Stormwater Runoff by Paved Surfaces. *Water Research* 34(4): 1359-1371.
- Wakao, N. and S. Kaguei. 1982. *Heat and Mass Transfer in Packed Beds*. New York: Gordon and Breach Science Publishers.

## **APPENDIX B**

### **Bioretention Thermal Model Visual Basic Code**

## Main Program Code

```
1 Imports System
2 Imports System.IO
3 Imports System.Text
4 Imports System.Math
5 Imports System.Diagnostics
6 Imports System.Xml
7 Imports Microsoft.VisualBasic
8 Imports Microsoft.VisualBasic.FileIO
9 Imports ZedGraph
10
11 '*** Table of Contents ***
12 'Line Number - Contents
13 ' 31 - Model initialization and file selection
14 ' 300 - Prepare rainfall and calculate inflow
15 ' 496 - Green-Ampt infiltration pre-ponded calculations
16 ' 762 - Green-Ampt infiltration post-ponded calculations
17 ' 902 - Saturated flow calculations
18 ' 1100 - Soil drainage calculations
19 ' 1484 - Simulation of dry pavement temperature profiles
20 ' 1846 - Solar radiation calculations
21 ' 1903 - Air temperature calculations
22 ' 2032 - Simulation of runoff temperature and pavement temperature profile during the storm
23 ' 2441 - Calculation of antecedent soil temperature profile
24 ' 2560 - Simulation of soil and water temperature profiles during infiltration
25 ' 2925 - Simulation of soil and water temperature profiles during drainage
26 ' 3230 - Calculation of thermal loads and storm summaries
27
28
29 Public Class Bioretention
```

```

30
31     'Dimension user input variables that are used in several places
32     Public InputRainFileName As String 'Filename for loading rainfall data
33     Public StormDate As Date 'Date of simulated storm
34     Public XMLOpenWeatherFileName As String 'Filename for loading xml weather data
35     Public RunoffTempFileName As String 'Filename for loading runoff temperatures
36     Public OnScreenHelpVisible As Boolean 'Describes whether help should be turned on or off
37     'Dimension output filenames
38     Public HydraulicOutFileName As String
39     Public DryPaveOutFileName As String
40     Public WetPaveOutFileName As String
41     Public FluidOutFileName As String
42     Public SoilOutFileName As String
43     Public ThermalLoadOutFileName As String
44
45     'Make arrays public here so they can be referenced after the simulation is finished
46     Public Shared TRMStoreArray(,) As String 'Array for storing runoff and pavement
temperature simulation results
47     'TRMStoreArray format: Time (date), RunoffTemp(°C), SurfaceTemp(°C), Depth1(°C),
Depth2(°C), ...
48     Public Shared TRMStoreArrayLength As Integer 'Number of rows in TRMStoreArray for use in
plotting
49
50     Public Shared HydraulicComboArray(,) As String 'Array containing both sets of hydraulic
calculations
51     Public Shared HydraulicComboArrayLength As Integer 'Length of array
52
53     Public Shared WetSoilComboArray_F(,) As Double 'Combination of both sets of wetsoil
simulations for fluid phase
54     Public Shared WetSoilComboArray_S(,) As Double 'Combination of both sets of wetsoil
simulations for solid phase
55     Public Shared WetSoilComboTimeArray(,) As Date 'Date array for combination of wetsoil
simulations
56     Public Shared WetSoilComboArrayLength As Integer 'Number of rows in WetSoilOut Arrays for
use in plotting

```

```

57     Public Shared WetSoilComboArray_NumCols As Integer 'Number of columns of depth data for
wet soil output array, used in deciding which lines to graph
58     Public Shared SoilDepth As Double 'Distance from soil surface to beginning of gravel layer
(m)
59     Public Shared GridSpacing As Double 'Grid spacing along z axis (m)
60
61     Public Shared ThermalLoadArray(,) As String 'Array for storing thermal load from runoff,
to drain, and underlying soil
62     Public Shared ThermalLoadArrayLength As Integer 'Length of array, accounting for all
runoff and drain times
63     Public Shared ThermalEnergyArray(,) As String 'Array for storing cumulative thermal energy
64
65
66 #Region "Set help file information"
67     'Executes when the form loads
68     Private Sub Bioretention_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
69         'Sets the form icon to the 16 pixel version
70         Me.Icon = My.Resources.SIXTEEN
71
72         'Setup on-screen help information
73         OnScreenHelpVisible = True
74
75         'Create tooltips containing information on model inputs
76         Dim HelpToolTip As New ToolTip()
77         HelpToolTip.SetToolTip(HBUT_RainInputFile, "Load comma separated variable file
containing storm date and rainfall data")
78         HelpToolTip.SetToolTip(HBUT_AirTemp, "Air temperature during the storm event")
79         HelpToolTip.SetToolTip(HBUT_WindSpeed, "Wind speed during the storm event")
80         HelpToolTip.SetToolTip(HBUT_Radiation, "Solar radiation during the storm event")
81         HelpToolTip.SetToolTip(HBUT_RelHumidity, "Relative humidity during the storm event")
82         HelpToolTip.SetToolTip(HBUT_WeatherInputFile, "Location of file containing normal
monthly air temperatures and wind speeds")
83         HelpToolTip.SetToolTip(HBUT_RunoffInputFile, "Location of file containing runoff
temperatures if external runoff temperatures are to be used")

```

```

84         HelpToolTip.SetToolTip(HBUT_WatershedArea, "Size of the watershed contributing runoff
to the bioretention area")
85         HelpToolTip.SetToolTip(HBUT_InitialAbstraction, "Depth of rainfall stored within
depressions on the watershed surface before runoff")
86         HelpToolTip.SetToolTip(HBUT_BioretentionArea, "Size of the bioretention footprint")
87         HelpToolTip.SetToolTip(HBUT_Latitude, "Latitude of the bioretention location")
88         HelpToolTip.SetToolTip(HBUT_Longitude, "Longitude of the bioretention location")
89         HelpToolTip.SetToolTip(HBUT_Transmittance, "Portion of solar radiation that penetrates
the atmosphere and reaches the ground surface")
90         HelpToolTip.SetToolTip(HBUT_Elevation, "Elevation of the bioretention location")
91         HelpToolTip.SetToolTip(HBUT_Timezone, "Number of hours from Greenwich Mean Time")
92         HelpToolTip.SetToolTip(HBUT_Shading, "Portion of the bioretention surface covered by
vegetation")
93         HelpToolTip.SetToolTip(HBUT_DrainDepth, "Distance from soil surface to the bottom of
underdrain pipes")
94         HelpToolTip.SetToolTip(HBUT_Suction, "Suction head in advance of the wetting front")
95         HelpToolTip.SetToolTip(HBUT_KSat, "Saturated hydraulic conductivity of the
bioretention soil")
96         HelpToolTip.SetToolTip(HBUT_ThetaFilled, "Soil pore space that can be filled during
infiltration")
97         HelpToolTip.SetToolTip(HBUT_SurfaceStorage, "Distance from soil surface to overflow
crest")
98         HelpToolTip.SetToolTip(HBUT_GravelPorosity, "Porosity of the gravel drainage layer")
99         HelpToolTip.SetToolTip(HBUT_DrainHeight, "Distance from subsoil to bottom of
underdrain pipes")
100        HelpToolTip.SetToolTip(HBUT_SubsoilKSat, "Saturated hydraulic conductivity of soil
beneath the bioretention area")
101        HelpToolTip.SetToolTip(HBUT_CpSoil, "Specific heat capacity of the bioretention soil
solids")
102        HelpToolTip.SetToolTip(HBUT_SoilPorosity, "Porosity of the bioretention soil")
103        HelpToolTip.SetToolTip(HBUT_ThermalKSoil, "Thermal conductivity of the bioretention
soil solids")
104        HelpToolTip.SetToolTip(HBUT_InterfaceArea, "Contact area between soil and water")
105        HelpToolTip.SetToolTip(HBUT_StagThermalK, "Thermal conductivity of the saturated
soil")

```

```
1106         HelpToolTip.SetToolTip(HBUT_ParticleDia, "Typical particle diameter of the
1107         bioretention soil")
1108         HelpToolTip.SetToolTip(HBUT_HydraulicOutput, "Location of file for storing results of
1109         hydraulic calculations")
1110         HelpToolTip.SetToolTip(HBUT_DryPaveOutput, "Location of file for storing temperature
1111         profiles of the pavement before the storm event")
1112         HelpToolTip.SetToolTip(HBUT_WetPaveOutput, "Location of file for storing runoff
1113         temperatures and temperature profiles of the pavement during the storm event")
1114         HelpToolTip.SetToolTip(HBUT_FluidTempOutput, "Location of file for storing fluid
1115         temperature profiles within the bioretention area during the storm event")
1116         HelpToolTip.SetToolTip(HBUT_SoilTempOutput, "Location of file for storing soil
1117         temperature profiles within the bioretention area during the storm event")
1118         HelpToolTip.SetToolTip(HBUT_ThermalLoadOutput, "Location of file for storing thermal
1119         load calculations for the duration of the storm event")
1120         HelpToolTip.SetToolTip(HBUT_RunoffEnergy, "Total amount of thermal energy associated
1121         with water leaving the watershed surface")
1122         HelpToolTip.SetToolTip(HBUT_EffluentEnergy, "Total amount of thermal energy associated
1123         with water leaving via the bioretention underdrains and overflow")
1124         HelpToolTip.SetToolTip(HBUT_OverflowEnergy, "Total amount of thermal energy associated
1125         with water leaving via the bioretention overflow structure")
1126         HelpToolTip.SetToolTip(HBUT_UnderdrainEnergy, "Total amount of thermal energy
1127         associated with water leaving via the bioretention underdrains")
1128         HelpToolTip.SetToolTip(HBUT_SeepageEnergy, "Total amount of thermal energy associated
1129         with water seeping into the underlying soil")
1130         HelpToolTip.SetToolTip(HBUT_ThermalLoadReduction, "Reduction between total runoff
1131         energy and total effluent energy")
1132         HelpToolTip.SetToolTip(HBUT_VolReduction, "Reduction between total runoff volume and
1133         total effluent volume")
1134         HelpToolTip.SetToolTip(HBUT_RunoffMax, "Maximum runoff temperature during simulated
1135         storm")
1136         HelpToolTip.SetToolTip(HBUT_EffluentMax, "Maximum effluent temperature during
1137         simulated storm")
1138         HelpToolTip.SetToolTip(HBUT_OverflowMax, "Maximum overflow temperature during
1139         simulated storm")
1140         HelpToolTip.SetToolTip(HBUT_UnderdrainMax, "Maximum underdrain temperature during
1141         simulated storm")
```

```

124         HelpToolTip.SetToolTip(HBUT_SeepageMax, "Maximum seepage temperature during simulated
storm")
125         HelpToolTip.SetToolTip(HBUT_RunoffAVG, "Average runoff temperature for the simulated
storm")
126         HelpToolTip.SetToolTip(HBUT_EffluentAVG, "Average effluent temperature for the
simulated storm")
127         HelpToolTip.SetToolTip(HBUT_OverflowAVG, "Average overflow temperature for the
simulated storm")
128         HelpToolTip.SetToolTip(HBUT_UnderdrainAVG, "Average underdrain temperature for the
simulated storm")
129         HelpToolTip.SetToolTip(HBUT_SeepageAVG, "Average seepage temperature for the simulated
storm")
130
131     End Sub
132 #End Region
133
134
135 #Region "Rainfall File Select"
136     'Executes when the rainfall file browse button is clicked
137     Private Sub BUT_RainFileBrowse_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_RainFileBrowse.Click
138         ' Filters requested files
139         RainOpenFileDialog.Filter = "rai files (*.rai)|*.rai|csv files (*.csv)|*.csv|All files
(*.*)|*.*"
140         If RainOpenFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
141             End If
142             ' Sets filename variable and updates main page
143             InputRainFileName = RainOpenFileDialog.FileName
144             TB_RainFileName.Text = InputRainFileName
145             TB_RainFileName.SelectionStart = Len(TB_RainFileName.Text)
146         End Sub
147 #End Region
148
149
150 #Region "Runoff Temperature File Select"

```

```

151     'Executes when the Runoff Temperature file browse button is clicked
152     Private Sub BUT_RunoffTemp_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_RunoffTemp.Click
153         ' Filters requested files
154         RunoffTempOpenFileDialog.Filter = "csv files (*.csv)|*.csv|All files (*.*)|*.*"
155         If RunoffTempOpenFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
156             End If
157         ' Sets filename variable and updates main page
158         RunoffTempFileName = RunoffTempOpenFileDialog.FileName
159         TB_RunoffTempFileName.Text = RunoffTempFileName
160         TB_RunoffTempFileName.SelectionStart = Len(TB_RunoffTempFileName.Text)
161     End Sub
162 #End Region
163
164
165 #Region "Weather Data File Select"
166     'Code for loading weather data from an XML file
167     Private Sub BUT_Weather_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles BUT_Weather.Click
168         ' Filters requested files
169         XMLOpenWeatherFileDialog.Filter = "xml files (*.xml)|*.xml|All files (*.*)|*.*"
170         ' if the user did not click on the OK button, return
171         If XMLOpenWeatherFileDialog.ShowDialog(Me) <> Windows.Forms.DialogResult.OK Then
172             Return
173         End If
174         ' Sets filename variable and updates main page
175         XMLOpenWeatherFileName = XMLOpenWeatherFileDialog.FileName
176         TB_WeatherFileName.Text = XMLOpenWeatherFileName
177         TB_WeatherFileName.SelectionStart = Len(TB_WeatherFileName.Text)
178     End Sub
179 #End Region
180
181
182     '-----
183     'Executes when Simulate button is clicked

```

```

184     Private Sub BUT_Simulate_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles BUT_Simulate.Click
185
186         'DEBUG MATERIAL
187         System.Diagnostics.Debug.WriteLine(Now()) 'Write current time to output window
188         'Starts stopwatch to observe total execution time
189         Dim stopWatchTotalTime As New Stopwatch()
190         stopWatchTotalTime.Start()
191         'END DEBUG MATERIAL
192
193         'Copy results from previous simulations into previous text boxes
194         'Copy energy results into previous text boxes
195         TB_Prev_RunoffEnergy.Text = TB_RunoffEnergy.Text
196         TB_Prev_EffluentEnergy.Text = TB_EffluentEnergy.Text
197         TB_Prev_OverflowEnergy.Text = TB_OverflowEnergy.Text
198         TB_Prev_PipeEnergy.Text = TB_PipeEnergy.Text
199         TB_Prev_SeepageEnergy.Text = TB_SeepageEnergy.Text
200         TB_Prev_ThermalReduction.Text = TB_ThermalReduction.Text
201         TB_Prev_VolReduction.Text = TB_VolReduction.Text
202         'Copy average temperature results into previous text boxes
203         TB_Prev_RunoffAvgTemp.Text = TB_RunoffAvgTemp.Text
204         TB_Prev_EffluentAvgTemp.Text = TB_EffluentAvgTemp.Text
205         TB_Prev_OverflowAvgTemp.Text = TB_OverflowAvgTemp.Text
206         TB_Prev_PipeAvgTemp.Text = TB_PipeAvgTemp.Text
207         TB_Prev_SeepageAvgTemp.Text = TB_SeepageAvgTemp.Text
208         'Copy maximum temperature results into previous text boxes
209         TB_Prev_RunoffMaxTemp.Text = TB_RunoffMaxTemp.Text
210         TB_Prev_EffluentMaxTemp.Text = TB_EffluentMaxTemp.Text
211         TB_Prev_OverflowMaxTemp.Text = TB_OverflowMaxTemp.Text
212         TB_Prev_PipeMaxTemp.Text = TB_PipeMaxTemp.Text
213         TB_Prev_SeepageMaxTemp.Text = TB_SeepageMaxTemp.Text
214
215
216         'DEBUG MATERIAL
217         'Starts stopwatch to observe time to load rainfall data

```

```

218 Dim stopWatchLoadRain As New Stopwatch()
219 stopWatchLoadRain.Start()
220 'END DEBUG MATERIAL
221
222 'Update status label
223 Label_Status.Text = "Status: Loading rainfall file"
224 Label_Status.Update()
225
226 'Dimension variables for loading rainfall data
227 'Initialize array for storing rain data read from csv file
228 Dim RainDataArray(,) As String
229 'RainDataArray format: Date and Time, Rainfall Depth (m), Rainfall Intensity (m/s)
230 'Only Date & Time, and Rainfall Depth are loaded from csv file
231 'Initial rainfall depth loaded must be 0 to correctly calculate intensities
232
233
234 'Routine to load Rainfall input file
235 'Load filename from text box
236 InputRainFileName = TB_RainFileName.Text
237 'Error check that the input file exists
238 If File.Exists(InputRainFileName) Then
239     'Simulation continues if file exists
240 Else
241     'Simulation stops and displays error message if file does not exist
242     MessageBox.Show("Input File Does Not Exist")
243     Exit Sub
244 End If
245
246
247 ' Indexes for storing values to correct position in array
248 Dim Rain_R As Integer = -1
249 Dim Rain_C As Integer = -1
250 Dim Rain_NumOfRows As Integer = -1
251
252 'Code to load rainfall .csv file into array

```

```

253 Dim CSVReader As StreamReader = File.OpenText(InputRainFileName)
254 Dim csvRow As String
255 Dim Rain_memFile As System.Collections.ArrayList
256 Rain_memFile = New System.Collections.ArrayList
257 csvRow = CSVReader.ReadLine()
258 Do While csvRow <> Nothing 'Perform the loop as long as there is something stored in
the row
259     Rain_NumOfRows = Rain_NumOfRows + 1 'Keep track of length of file
260     ' load each record to the array of records
261     Rain_memFile.Add(csvRow.Split(",")) 'Sets the delimiter as comma
262     csvRow = CSVReader.ReadLine() 'Sets each row as a record
263 Loop
264 'Redimension the array based on number of rows in csv file
265 ReDim RainDataArray(Rain_NumOfRows, 2)
266 'Read in each row
267 For Each Rain_record As Object In Rain_memFile
268     Rain_R = Rain_R + 1
269
270     'Read in each column and store it into the array
271     For Each Rain_field As Object In Rain_record
272         Rain_C = Rain_C + 1
273         If Rain_C = 2 Then
274             Rain_C = 0
275         End If
276         RainDataArray(Rain_R, Rain_C) = Rain_field
277     Next
278 Next
279
280 'Converts date and hour string data to Date format
281 For Rain_R = 0 To Rain_NumOfRows
282     RainDataArray(Rain_R, 0) = CDate(RainDataArray(Rain_R, 0))
283 Next Rain_R
284 'Converts rainfall String data to Double
285 For Rain_R = 0 To Rain_NumOfRows
286     RainDataArray(Rain_R, 1) = Double.Parse(RainDataArray(Rain_R, 1))

```

```

287     Next Rain_R
288
289
290     'DEBUG MATERIAL
291     'Stops rainfall load time stopwatch and prints time in Debug line
292     stopWatchLoadRain.Stop()
293     Debug.Print("=====")
294     System.Diagnostics.Debug.WriteLine("Load Time: " & stopWatchLoadRain.Elapsed.Seconds &
    ". " & stopWatchLoadRain.Elapsed.Milliseconds)
295     'END DEBUG MATERIAL
296
297
298
299     '*****
300     'Prepare rainfall array
301
302     'DEBUG LINES
303     'Starts stopwatch to observe execution time for rainfall disaggregation
304     Dim stopWatchDisaggregate As New Stopwatch()
305     stopWatchDisaggregate.Start()
306     'END DEBUG LINES
307
308     'Update status label
309     Label_Status.Text = "Status: Calculating watershed hydrology"
310     Label_Status.Update()
311
312     'Determine date and time parameters describing storm
313     Dim StormStopDate As Date 'Date and time of end of storm
314     'Loads storm date
315     StormDate = RainDataArray(0, 0)
316     'Loads StormStopDate as last date in rainfall input file
317     StormStopDate = RainDataArray(Rain_NumOfRows, 0)
318
319     'Convert rainfall depths from inches to m
320     For Rain_R = 0 To Rain_NumOfRows

```

```

321         RainDataArray(Rain_R, 1) = RainDataArray(Rain_R, 1) * 0.0254
322     Next Rain_R
323
324     'Calculate rainfall rate
325     Dim RainStepLength As Double '(s) Time between previous rainfall measurement time and
current rainfall measurement time
326     Dim CurRainTime As Date 'Date and time of current rainfall measurement
327     Dim PrevRainTime As Date 'Date and time of previous rainfall measurement
328     'Initial rainfall depth loaded must be 0 to correctly calculate intensities
329     'Manually set initial rainfall rate as 0
330     RainDataArray(0, 2) = 0
331     For Rain_R = 1 To Rain_NumOfRows
332         CurRainTime = RainDataArray(Rain_R, 0)
333         PrevRainTime = RainDataArray(Rain_R - 1, 0)
334         'Calculate rain step length
335         RainStepLength = DateDiff(DateInterval.Second, PrevRainTime, CurRainTime)
336         'Divide by step length to get average rainfall rate
337         RainDataArray(Rain_R - 1, 2) = RainDataArray(Rain_R, 1) / RainStepLength
338         'Rainfall rate is stored to previous timestep since it represents rate needed to
accumulate measured rainfall depth at the end of the timestep
339     Next
340
341
342     'Calculate bioretention inflow
343     'Currently removes initial abstraction, assumes rest of rainfall is converted entirely
to runoff
344     'Assumes instantaneous travel time
345     'NOTE: In the future, try to use kinematic wave or other rainfall-runoff model to
better simulate real system
346     'NOTE: Code could also be adapted to load hydrology data into the array from another
model, such as SWMM
347
348     'Initialize inflow variables
349     Dim InflowArray(,) As String
350     Dim InflowIndexLength As Integer

```

```

351         InflowIndexLength = Rain_NumOfRows + 1 'Add a row to array to account for the added
time slot for when runoff begins
352     ReDim InflowArray(InflowIndexLength, 2)
353     'Format: Date & Time, Inflow (m³/s), RainRate (m/s)
354     Dim InitialAbstraction As Double 'Initial abstraction of pavement surface (m)
355     Dim WatershedArea As Double 'Area of contributing watershed (m²)
356     Dim BioretentionArea As Double 'Area of bioretention cell (m²)
357     Dim PavementStorage As Double = 0 'Amount of rainfall stored in pavement surface (m)
358     Dim RunoffStartDate As Date 'Time when initial abstraction is satisfied and runoff
flow begins
359
360     'Load watershed information from GUI
361     WatershedArea = TB_WatershedArea.Text
362     InitialAbstraction = TB_InitialAbstraction.Text
363     BioretentionArea = TB_BioretentionArea.Text
364
365     'Determine time when initial abstraction is satisfied
366     Rain_R = -1
367     Dim SecondsToRunoff As Double 'Number of seconds from previous rain timestep to runoff
flow beginning
368     While Rain_R < Rain_NumOfRows And PavementStorage < InitialAbstraction
369         'Advance rainfall row
370         Rain_R = Rain_R + 1
371         'If rainfall depth in current timestep plus previous storage exceeds initial
abstraction
372         If RaindataArray(Rain_R, 1) + PavementStorage > InitialAbstraction Then
373             SecondsToRunoff = (InitialAbstraction - PavementStorage) /
RaindataArray(Rain_R, 2) 'Determine how many seconds it would take with current rainfall rate
to satisfy initial abstraction
374             PrevRainTime = RaindataArray(Rain_R - 1, 0) 'Load time for beginning of
current measurement
375             RunoffStartDate = PrevRainTime.AddSeconds(SecondsToRunoff) 'Add
SecondsToRunoff to date to get time when runoff flow begins
376             PavementStorage = InitialAbstraction 'Fill pavement storage to initial
abstraction

```

```

377         Else
378             'If initial abstraction is not exceeded, add to pavement storage and move to
next timestep
379             PavementStorage = PavementStorage + RainDataArray(Rain_R, 1)
380         End If
381     End While
382
383
384     'Convert rainfall to runoff
385     Rain_R = 0
386     PrevRainTime = RainDataArray(Rain_R, 0)
387     CurRainTime = RainDataArray(Rain_R + 1, 0)
388     'If Runoff does NOT begin during current timestep
389     While CurRainTime < RunoffStartDate
390         InflowArray(Rain_R, 0) = RainDataArray(Rain_R, 0) 'Copy date
391         InflowArray(Rain_R, 1) = 0 'No inflow
392         InflowArray(Rain_R, 2) = RainDataArray(Rain_R, 2) 'Copy rainfall rate
393         Rain_R = Rain_R + 1
394         PrevRainTime = RainDataArray(Rain_R, 0)
395         CurRainTime = RainDataArray(Rain_R + 1, 0)
396     End While
397
398     'If runoff DOES begin during current timestep
399     InflowArray(Rain_R, 0) = RainDataArray(Rain_R, 0) 'Copy date
400     InflowArray(Rain_R, 1) = 0 'No inflow
401     InflowArray(Rain_R, 2) = RainDataArray(Rain_R, 2) 'Copy rainfall rate
402     InflowArray(Rain_R + 1, 0) = RunoffStartDate 'Copy runoffstartdate
403     InflowArray(Rain_R + 1, 1) = RainDataArray(Rain_R, 2) * WatershedArea 'Calculate
inflow rate
404     InflowArray(Rain_R + 1, 2) = RainDataArray(Rain_R, 2) 'Copy rainfall rate
405     Rain_R = Rain_R + 1
406
407     'Calculate runoff rate and copy rainfall rate for times after runoff has begun
408     While Rain_R < Rain_NumOfRows
409         InflowArray(Rain_R + 1, 0) = RainDataArray(Rain_R, 0) 'Copy date

```

```

410         InflowArray(Rain_R + 1, 1) = RainDataArray(Rain_R, 2) * WatershedArea 'Calculate
inflow rate
411         InflowArray(Rain_R + 1, 2) = RainDataArray(Rain_R, 2) 'Copy rainfall rate
412         Rain_R = Rain_R + 1
413     End While
414
415     'Copy last date of measured rainfall with no runoff
416     InflowArray(InflowIndexLength, 0) = RainDataArray(Rain_NumOfRows, 0) 'Copy date
417     InflowArray(InflowIndexLength, 1) = 0 'Specify no inflow since inflow stops after this
time
418     InflowArray(InflowIndexLength, 2) = 0 'No rainfall rate since rain has stopped
419
420     'Calculate the duration of the loaded storm event in seconds
421     Dim StormDuration As Double 'Duration of the loaded storm event (sec)
422     StormDuration = DateDiff(DateInterval.Second, StormDate, StormStopDate)
423     'Calculate duration of runoff
424     Dim RunoffDuration As Double '(s) Length of time that runoff lasts
425     RunoffDuration = DateDiff(DateInterval.Second, RunoffStartDate, StormStopDate)
426
427     'Initialize Green-Ampt Array
428     Dim GreenAmptArray(,) As String 'Main array for Green-Ampt calculations
429     Dim GATimeStep As Double 'Timestep for G-A calculations (hr)
430     GATimeStep = 1 / 60 / 60 '(hr)
431     Dim GAIndexLength As Integer
432     GAIndexLength = RunoffDuration / (GATimeStep * 60 * 60) 'Calculate # of rows in
GreenAmptArray
433     ReDim GreenAmptArray(GAIndexLength, 3)
434     'GreenAmptArray Contents: Time (Date), WettingFront (m), Infil (m/hr), EffRainRate
(m/hr)
435     Dim GAIndex As Integer 'Index for GreenAmptArray
436     Dim CurGADate As Date 'Current date of Green Ampt simulations
437     Dim NexRainTime 'Date of upcoming change in rainfall rate
438
439     Dim HydrologyArray(,) As String 'Same structure as Green-Ampt array but applies to
gravel layer and overflow

```

```

440         'HydrologyArray Contents: Time (Date), Overflow (m/hr), Underlying Soil Infiltration
(m/hr), Pipe Flow (m/hr)
441         ReDim HydrologyArray(GAIndexLength, 3)
442
443         'NOTE: Current dimensioning of GreenAmptArray would cause a problem if wetting front
did not reach drain layer during storm and there was still ponding on surface
444         'NOTE: This issue is probably irrelevant because thermal calculations require
saturated profile
445         'NOTE: Model now displays an error message if wetting front does not reach the drain
layer during a storm event
446
447         'Convert inflow to an equivalent rainfall rate over the bioretention area
448         'Set initial row manually
449         CurGADate = RunoffStartDate
450         GreenAmptArray(0, 0) = CurGADate
451
452         'Advances through times before inflow begins
453         Rain_R = 0
454         While CurGADate < InflowArray(Rain_R, 0)
455             Rain_R = Rain_R + 1
456         End While
457
458         'Load initial rain times
459         CurRainTime = InflowArray(Rain_R, 0)
460         NexRainTime = InflowArray(Rain_R + 1, 0)
461         GAIndex = 0
462
463         'Copy rainfall rates to match requirements of GreenAmptArray
464         While GAIndex < GAIndexLength
465             'Rainfall rate for green ampt array equals inflow from impervious area, evenly
distributed over the bioretention area
466             'Load rain times
467             CurRainTime = InflowArray(Rain_R, 0)
468             NexRainTime = InflowArray(Rain_R + 1, 0)
469

```

```

470         While CurGADate < NexRainTime
471             GreenAmptArray(GAIndex, 0) = CurGADate 'Copy date
472             'Calculate effective rainfall rate over bioretention, including runoff and
direct rainfall, and convert from m/s to (m/hr)
473             'GreenAmptArray(GAIndex, 3) = (InflowArray(Rain_R, 1) / BioretentionArea +
InflowArray(Rain_R, 2)) * 60 * 60
474
475             'Calculate effective rainfall rate over bioretention, including only runoff,
and convert from m/s to (m/hr)
476             GreenAmptArray(GAIndex, 3) = (InflowArray(Rain_R, 1) / BioretentionArea) * 60
* 60
477
478             GAIndex = GAIndex + 1 'Advance GAIndex
479             CurGADate = CurGADate.AddSeconds(GATimeStep * 60 * 60) 'Advance date
480         End While
481         'When GA Date passes next rainfall time, advance index for inflow and rain rate
482         Rain_R = Rain_R + 1
483
484     End While
485
486     'DEBUG MATERIAL
487     'Stops execution time stopwatch and prints execution time in Debug line
488     stopWatchDisaggregate.Stop()
489     Debug.Print("=====")
490     System.Diagnostics.Debug.WriteLine("Dissagregation and Watershed Hydrology: " &
stopWatchDisaggregate.Elapsed.Seconds & "." & stopWatchDisaggregate.Elapsed.Milliseconds)
491     'END DEBUG MATERIAL
492
493
494
495     '**_**_**_**_**_**
496     'Green-Ampt Calculations
497     'Based on work of Chu and Marino (2005)
498
499     Dim GASoilArray(,) As Double 'Array containing soil properties for use in green-ampt

```

```

calculations
500     'Dim SoilDepth As Double 'Distance from soil surface to beginning of gravel layer (m)
501     Dim GATime As Double 'Time used in Green-Ampt calculations (hr)
502     Dim GADate As Date 'Current date in Green-Ampt calculations
503     Dim RainRate As Double 'Rainfall rate used in G-A calculations (m/hr)
504     Dim Suction As Double 'Pressure head (suctions) used in G-A calculations (m)
505     Dim TotalRain As Double 'Cumulative rainfall (m)
506     Dim KSat As Double 'Saturated hydraulic conductivity (m/hr)
507     Dim ThetaFilled As Double 'Difference between intial and saturated water content
      (cm/cm)
508     Dim WettingFront As Double 'Position of the wetting front (m)
509     Dim InfilCap As Double 'Infiltration capactiy (m/hr)
510     'Dim TotalStorage As Double 'Water stored above in soil profile
511
512     Dim TotalZjKj As Double 'Sum of ( z(j) - z(j-1) ) / K(j)
513     Dim PondingDepth As Double 'Specific ponding depth (m)
514     Dim PondingTime As Double 'tp from equation 27
515     'Dim PondingInfilTotal As Double 'Cumulative infiltration at the ponding condition
516     Dim TimePseudo As Double 'Equivalent time for wetting front to advance if under ponded
condition
517     Dim TimePD As Double 'TimePD from equation 28
518     Dim TZero As Double 'Initial time at which a ponded condition is assumed
519     Dim GASoilIndex As Integer 'Index for loading soil properties
520     Dim Poned As Boolean = False 'True if surface is under ponded conditions
521     Dim SoilStep As Double 'Soil property discretization increment, 0.001 = every mm
522     Dim GASoilArraySize As Integer 'Necessary number of rows for GASoilArray
523     Dim PseudoIndex As Integer 'Index for use in calculating pseudo time
524     Dim NonPonedStart As Integer = 1 'Index value when non-ponded conditions begin
525
526     'Post-Ponding Variables
527     Dim CumulativeInf As Double 'Cumulative infiltration at time step k
528     Dim SurfaceStorage As Double '(m) Surface storage at time step k
529     Dim CumulativeOverflow As Double 'cumulative overflow (termed runoff in Chu's paper)
530     Dim Infiltrated As Double 'Depth infiltrated during current timestep (m)
531     Dim Overflowed As Double 'Depth overflowed during current timestep (m)

```

```

532 Dim StorageCapacity As Double 'Storage depths where overflow begins (m)
533 Dim InfilRate As Double 'Infiltration rate (m/hr)
534
535 'Variables for Newton-Raphson Iteration
536 Dim ZError As Double 'Error in depth estimate during each iteration step
537 Dim ZGuess As Double 'Initial guess for wetting front depth
538 Dim ZResult As Double 'Result of current iteration estimate
539 Dim IterationCount As Integer 'Number of iterations to reach convergence
540 Dim TotalIterationCount As Integer = 0 'Total number of iterations for all timesteps
541 Dim TShifted As Double 'Current time (shifted for the ponded calculations)
542 Dim TPrev As Double 'Previous time
543 Dim ZPrev As Double 'Wetting front depth at previous timestep
544 Dim ZKTot As Double 'Cumulative term from equation 16 and 28
545
546 Dim WettingFrontShift As Double 'Wetting front position at end of ponded conditions,
used to correct error associated with switching from ponded to pre-ponded
547 Dim WettingFrontAdv As Double 'Distance wetting front advanced during current timestep
(m)
548
549 'Update status label
550 Label_Status.Text = "Status: Performing Green-Ampt calculations"
551 Label_Status.Update()
552
553 'Load soil depth from GUI
554 SoilDepth = TB_SoilDepth.Text
555
556 'Round soil depth to 3 decimal places to prevent indexing error
557 SoilDepth = Round(SoilDepth, 3)
558 SoilStep = 0.001 'Sets soil property discretization increment, 0.001 = every mm
559 'NOTE: If soil step is changed, need to change rounding when Soil properties are
loaded based on wetting front position
560 GASoilArraySize = SoilDepth / SoilStep
561
562 ReDim GASoilArray(GASoilArraySize, 3)
563 'GASoilArray Contents: Depth, KSat, Suction, ThetaFilled

```

```

564
565     'Load soil property values from GUI
566     Suction = TB_Suction.Text
567     KSat = TB_KSat.Text
568     ThetaFilled = TB_ThetaFilled.Text
569
570     'TEMPORARY DEBUG LINE
571     Dim VolCheck As Double
572     'END TEMPORARY LINE
573
574     'Setup soil spatial properties array: GASoilArray
575     'Copy constant KSat, Suction, and ThetaFilled values into GASoilArray
576     'NOTE: Current model does not use layered soil properties; however, these properties
are considered in the Green-Ampt algorithm and could be implemented with minimal programming
changes
577     'Manually copy first row
578     GASoilIndex = 0
579     GASoilArray(GASoilIndex, 0) = 0
580     GASoilArray(GASoilIndex, 1) = KSat
581     GASoilArray(GASoilIndex, 2) = Suction
582     GASoilArray(GASoilIndex, 3) = ThetaFilled
583     'Copy remaining rows
584     For GASoilIndex = 1 To GASoilArraySize
585         GASoilArray(GASoilIndex, 0) = GASoilArray(GASoilIndex - 1, 0) + SoilStep
586         GASoilArray(GASoilIndex, 0) = Round(GASoilArray(GASoilIndex, 0), 3) 'Round to
prevent truncation error
587         GASoilArray(GASoilIndex, 1) = KSat
588         GASoilArray(GASoilIndex, 2) = Suction
589         GASoilArray(GASoilIndex, 3) = ThetaFilled
590     Next GASoilIndex
591
592
593     'Set initial Green-Ampt Array values
594     GAIndex = 0
595     GATime = 0

```

```

596     WettingFront = 0
597     CumulativeOverflow = 0
598     CumulativeInf = 0
599     GADate = RunoffStartDate
600
601     'Manually perform first timestep for pre-ponded conditions
602     GreenAmptArray(0, 0) = GADate 'Store initial time
603     GreenAmptArray(0, 1) = 0 'Initial wetting front is zero
604     HydrologyArray(0, 0) = GADate 'Store intial time for indexing purposes only
605
606     'Manually perform second timestep
607     GAIndex = GAIndex + 1
608     GATime = GATime + GATimeStep 'Time advanced
609     GATime = Round(GATime, 3) 'Round to prevent truncation error
610     GADate = GADate.AddSeconds(GATimeStep * 60 * 60) 'Date advanced
611     GreenAmptArray(1, 0) = GADate 'Date stored in array
612     HydrologyArray(1, 0) = GADate 'Date stored in array for indexing purposes only
613     RainRate = GreenAmptArray(GAIndex, 3) 'Load rainfall rate
614     GASoilIndex = Round(WettingFront, 3) / SoilStep 'Specifices position in soil for
loading soil properties
615     ThetaFilled = GASoilArray(GASoilIndex, 3) 'Load fillable porosity for current position
616     KSat = GASoilArray(GASoilIndex, 1) 'Load KSat for current position
617     TotalRain = TotalRain + RainRate * GATimeStep 'Update cumulative rainfall
618     WettingFrontAdv = (RainRate * GATimeStep) / ThetaFilled
619     WettingFront = WettingFront + WettingFrontAdv 'Wetting front calculated based on
current rain rate instead of comparing total rain and total storage
620     GreenAmptArray(1, 1) = WettingFront 'Stores wetting front position in array
621     GAIndex = GAIndex + 1 'Advance index before entering loop
622
623
624     '-----
625     'While loop for overall infiltration
626     'Allows for switching back and forth between ponded and unponded
627     While WettingFront < SoilDepth And GAIndex < GAIndexLength
628

```

```

629         '-----
630         'PRE-PONDED CONDITION CALCULATIONS
631
632         'DEBUG LINES
633         'Starts stopwatch to observe execution time for pre-ponding
634         Dim stopWatchPrePonded As New Stopwatch()
635         stopWatchPrePonded.Start()
636         'END DEBUG LINES
637
638         'Set infiltration rate extraordinarily high to enter while loop
639         'Note: Could manually calculate infiltration capacity for initial timestep;
however, overall error is minimal due to short time step
640         InfilCap = 100000
641
642         'Runs loop for pre-ponding conditions
643         While Ponded = False And WettingFront < SoilDepth And GAIndex < GAIndexLength
644             GATime = GATime + GATimeStep 'Advance time
645             GATime = Round(GATime, 3) 'Round to prevent truncation error
646             GADate = GADate.AddSeconds(GATimeStep * 60 * 60) 'Date advanced
647             GreenAmptArray(GAIndex, 0) = GADate 'Date stored in array
648             HydrologyArray(GAIndex, 0) = GADate 'Date stored in array for indexing
purposes only
649             RainRate = GreenAmptArray(GAIndex, 3) 'Load rainfall rate
650             GASoilIndex = Round(WettingFront, 3) / SoilStep 'Specifices position in soil
for loading soil properties
651             ThetaFilled = GASoilArray(GASoilIndex, 3) 'Load fillable porosity for current
position
652             KSat = GASoilArray(GASoilIndex, 1) 'Load KSat for current position
653             Suction = GASoilArray(GASoilIndex, 2) 'Load suction for current position
654             TotalRain = TotalRain + RainRate * GATimeStep 'Update cumulative rainfall
655             'WettingFront = WettingFront + (TotalRain - TotalStorage) / ThetaFilled
'Wetting front determined by comparing total rain to total storage
656             WettingFrontAdv = (RainRate * GATimeStep) / ThetaFilled
657             WettingFront = WettingFront + WettingFrontAdv 'Wetting front calculated based
on current rain rate instead of comparing total rain and total storage

```

```

658      GreenAmptArray(GAIndex, 1) = WettingFront 'Stores wetting front position in
array
659
660      'Calculate cumulative infiltration for use in next timestep
661      CumulativeInf = CumulativeInf + WettingFrontAdv * ThetaFilled
662
663      'Calculate infiltration capacity
664      'TotalZjKj is intermediate step in calculating infiltration capacity
665      'TotalZjKj represents the effective conductivity of the soil profile up to the
current point
666      TotalZjKj = TotalZjKj + (GreenAmptArray(GAIndex - 1, 1) -
GreenAmptArray(GAIndex - 2, 1)) / KSat
667      'Infiltration capacity calculated based on equation 13
668      InfilCap = (WettingFront + Suction) / (TotalZjKj + (GreenAmptArray(GAIndex, 1)
- GreenAmptArray(GAIndex - 1, 1)) / KSat)
669
670      'Store infiltration rate into array
671      GreenAmptArray(GAIndex, 2) = WettingFrontAdv * ThetaFilled / GATimeStep
672
673      'TEMPORARY DEBUG LINE
674      VolCheck = TotalRain / CumulativeInf
675      'System.Diagnostics.Debug.WriteLine(VolCheck)
676      'END TEMPORARY LINE
677
678      'System.Diagnostics.Debug.WriteLine(GAIndex & "," & GATime & "," &
WettingFront & "," & InfilCap)
679
680      'Check ponding status
681      If RainRate > InfilCap Then
682          Poned = True
683      End If
684
685      GAIndex = GAIndex + 1
686  End While
687

```

```

688         'If storm doesn't generate outflow
689         'Displays an error message because thermal calculations are based upon saturated
flow
690         If GAIndex >= GAIndexLength And WettingFront < SoilDepth Then
691             MessageBox.Show("Storm does not generate substantial outflow")
692             Exit Sub
693         End If
694
695         'Corrects index since loop was exited
696         GAIndex = GAIndex - 1
697         'Corrects cumulative infiltration since loop was exited
698         CumulativeInf = CumulativeInf - WettingFrontAdv * ThetaFilled
699
700
701         'Calculate specifics of moment when ponding occurs
702         If Poned = True Then
703             'Calculate total for equation 28 and pseudo time
704             'Run loop up until depth right before ponding
705             For PseudoIndex = NonPonedStart To GAIndex - 1
706                 WettingFront = GreenAmptArray(PseudoIndex, 1) 'Load wetting front position
707                 GASoilIndex = Round(WettingFront, 3) / SoilStep 'Specifices position in
soil for loading soil properties
708                 ThetaFilled = GASoilArray(GASoilIndex, 3) 'Load fillable porosity for
current position
709                 KSat = GASoilArray(GASoilIndex, 1) 'Load KSat for current position
710                 Suction = GASoilArray(GASoilIndex, 2) 'Load suction for current position
711
712                 'Calculate ZKTot for use in equation 16
713                 'If wetting front is 0, prevent indexing error
714                 If GASoilIndex = 0 Then
715                     GASoilIndex = 1
716                 End If
717                 ZKTot = ZKTot + WettingFront * (1 / GASoilArray(GASoilIndex - 1, 1) - 1 /
(GASoilArray(GASoilIndex, 1)))
718                 'TimePseudo calculated from equation 16

```

```

719             'Equal to the time for wetting front to reach current position under
ponded conditions
720         TimePseudo = TimePseudo + ThetaFilled / KSat *
(GreenAmptArray(PseudoIndex, 1) - GreenAmptArray(PseudoIndex - 1, 1)) + ThetaFilled * (ZKTot -
Suction / KSat) * Log((GreenAmptArray(PseudoIndex, 1) + Suction) / (GreenAmptArray(PseudoIndex
- 1, 1) + Suction))
721
722         Next PseudoIndex
723
724
725         'Solve for specific ponding depth using equation 24
726         PondingDepth = WettingFrontShift + (KSat * Suction + RainRate *
GreenAmptArray(GAIndex - 1, 1) - RainRate * KSat * TotalZjKj) / (RainRate - KSat)
727         'Calculate cumulative infiltration at the time ponding condition is met
728         'Uses cumulative infiltration from before, but adds in the extra amount for
the partial interval from the previous timestep to the actual ponding depth
729         CumulativeInf = CumulativeInf + (PondingDepth - GreenAmptArray(GAIndex - 2,
1)) * ThetaFilled
730         'Calculation of ponding time tp using equation 27
731         PondingTime = (1 / RainRate) * (CumulativeInf - (TotalRain - RainRate *
GATimeStep)) + GATime - GATimeStep
732         'Calculation of TimePD using equation 28
733         'TimePD is the equivalent time for the wetting front to reach the current
position under ponded conditions
734         TimePD = TimePseudo + (ThetaFilled / KSat) * (PondingDepth -
GreenAmptArray(GAIndex - 1, 1)) + ThetaFilled * (ZKTot - Suction / KSat) * Log((PondingDepth +
Suction) / (GreenAmptArray(GAIndex - 1, 1) + Suction))
735         'Calculate time at which a ponded conditions are assumed using equation 29
736         TZero = PondingTime - TimePD
737         End If
738
739         'DEBUG LINES
740         'Display estimated and specific wetting front depths
741         'MessageBox.Show("Estimated: " & WettingFront.ToString & ", " & "Actual: " &
PondingDepth)

```

```

742         'END DEBUG LINES
743
744         'DEBUG MATERIAL
745         'Stops execution time stopwatch and prints execution time in Debug line
746         stopWatchPrePonded.Stop()
747         System.Diagnostics.Debug.WriteLine("Pre-Ponding Time: " &
stopWatchPrePonded.Elapsed.Seconds & "." & stopWatchPrePonded.Elapsed.Milliseconds)
748         System.Diagnostics.Debug.WriteLine("GATime: " & GATime)
749         System.Diagnostics.Debug.WriteLine("Wetting Front: " & WettingFront)
750         'END DEBUG MATERIAL
751
752         '----
753         'DEBUG LINES
754         'Starts stopwatch to observe execution time for ponded condition
755         Dim stopWatchPostPonded As New Stopwatch()
756         stopWatchPostPonded.Start()
757         'END DEBUG LINES
758
759
760
761         '-----
762         'POST PONDING COMPUTATION
763
764         'Set intial surface storage at zero
765         SurfaceStorage = 0
766
767         'Load values from GUI
768         StorageCapacity = TB_StorageCapacity.Text
769
770         'Update variables with results of pre-ponding condition
771         TShifted = GATime - PonderingTime + TimePD
772         TPrev = TShifted - GATimeStep
773         ZPrev = PonderingDepth
774         ZResult = PonderingDepth
775         ZGuess = PonderingDepth 'Set initial guess at previous ponding depth

```

```

776
777     'Perform Newton-Raphson iteration to solve equation 16
778     While Poneded = True And WettingFront < SoilDepth And GAIndex < GAIndexLength
779
780         'Initialize error high to enter the while loop
781         ZError = 100000
782         'Update total iteration count
783         TotalIterationCount = TotalIterationCount + IterationCount
784         'Reset iteration count
785         IterationCount = 0
786
787         'Load rainfall rate for current time and convert to m/hr
788         RainRate = GreenAmptArray(GAIndex, 3)
789
790         'Load soil properties for current depth
791         GASoilIndex = Round(ZResult, 3) / SoilStep 'Specifices position in soil for
loading soil properties
792         ThetaFilled = GASoilArray(GASoilIndex, 3) 'Load fillable porosity for current
position
793         KSat = GASoilArray(GASoilIndex, 1) 'Load KSat for current position
794         Suction = GASoilArray(GASoilIndex, 2) 'Load suction for current position
795
796         'Calculate ZKTot for use in equation 16
797         ZKTot = ZKTot + WettingFront * (1 / GASoilArray(GASoilIndex - 1, 1) - 1 /
(GASoilArray(GASoilIndex, 1)))
798
799         'ITERATE TO SOLVE FOR PONDED CONDITIONS
800         'Sets error tolerance at 0.0001 (m) and iterate to solve for ZResult
801         While Abs(ZError) > 0.00001
802             IterationCount = IterationCount + 1
803             'Apply Newton-Raphson iteration method
804             ZResult = ZGuess - (TPrev - TShifted + (ThetaFilled / KSat) * (ZGuess -
ZPrev) + ThetaFilled * (ZKTot - Suction / KSat) * Log((ZGuess + Suction) / (ZPrev + Suction)))
/ ((ZKTot - Suction / KSat) * ThetaFilled / (ZGuess + Suction) + ThetaFilled / KSat)
805             ZError = ZResult - ZGuess 'Calculate error between previous guess and

```

```

current result
806         ZGuess = ZResult 'Set next guess equal to current iteration result
807     End While
808
809
810     'Advance GreenAmptArray and store time and wetting front depth
811     GAIndex = GAIndex + 1
812     GATime = GATime + GATimeStep 'Advance time
813     GATime = Round(GATime, 3) 'Round to prevent truncation error
814     GADate = GADate.AddSeconds(GATimeStep * 60 * 60) 'Date advanced
815     GreenAmptArray(GAIndex, 0) = GADate 'Date stored in array
816     HydrologyArray(GAIndex, 0) = GADate 'Date stored for indexing purposes
817     GreenAmptArray(GAIndex, 1) = ZResult 'Store wetting front
818
819
820     'Calculate infiltration rate using equation 13
821     TotalZjKj = TotalZjKj + (GreenAmptArray(GAIndex - 1, 1) -
GreenAmptArray(GAIndex - 2, 1)) / GASoilArray(GASoilIndex - 1, 1)
822     InfilRate = (ZResult + Suction) / (TotalZjKj + (ZResult - ZPrev) / KSat)
823
824     'Calculate depth infiltrated during current timestep
825     'NOTE: This calculation is for homogeneous soil and doesn't have the layer
detail contained in equation 32
826     Infiltrated = (ZResult - ZPrev) * ThetaFilled
827
828     'Store infiltration rate into GreenAmptArray
829     GreenAmptArray(GAIndex, 2) = Infiltrated / GATimeStep
830
831     'Calculate total amount of rain
832     TotalRain = TotalRain + RainRate * GATimeStep
833
834     'Update cumulative infiltration depth
835     CumulativeInf = CumulativeInf + Infiltrated
836
837     'Calculate current surface storage depth and overflow

```

```

838         If RainRate * GATimeStep <= Infiltrated + (StorageCapacity - SurfaceStorage)
839     Then
840         'If surface storage depth will not increase above capacity during current
841     timestep
842         SurfaceStorage = SurfaceStorage + RainRate * GATimeStep - Infiltrated
843         Overflowed = 0
844     Else
845         'If storage depth will increase above capacity during current timestep
846     Overflowed = RainRate * GATimeStep - Infiltrated - (StorageCapacity -
847     SurfaceStorage)
848     HydrologyArray(GAIndex, 1) = Overflowed / GATimeStep 'Store overflow rate
849     into HydrologyArray
850     CumulativeOverflow = CumulativeOverflow + Overflowed
851     SurfaceStorage = StorageCapacity
852     End If
853
854     'Determine ponding conditions
855     'NOTE: Different approach from Chu's paper
856     If SurfaceStorage <= 0 Then
857         Poned = False
858         NonPonedStart = GAIndex
859         WettingFrontShift = ZResult
860     End If
861
862     'Prepare for next timestep
863     WettingFront = ZResult 'Store iteration result to wetting front
864     TPrev = TShifted 'Update previous time
865     TShifted = TShifted + GATimeStep 'Advance shifted time
866     ZPrev = ZResult 'Stores iteration result into previous depth for next timestep
867     ZGuess = ZResult 'Set intial guess for next step as current result
868
869     'TEMPORARY DEBUG LINE
870     VolCheck = TotalRain / (SurfaceStorage + CumulativeInf)
871     'System.Diagnostics.Debug.WriteLine (VolCheck)

```

```

869         'END TEMPORARY LINE
870
871         'System.Diagnostics.Debug.WriteLine("Time: " & GATime & " Wetting Front: " &
WettingFront)
872
873         End While 'While loop for ponded conditions
874
875         'If storm doesn't generate outflow
876         If GAIndex >= GAIndexLength And WettingFront < SoilDepth Then
877             MessageBox.Show("Storm does not generate substantial outflow")
878             Exit Sub
879         End If
880
881         'DEBUG MATERIAL
882         'Stops execution time stopwatch and prints execution time in Debug line
883         stopwatchPostPonded.Stop()
884         System.Diagnostics.Debug.WriteLine("Post-Ponding Time: " &
stopwatchPostPonded.Elapsed.Seconds & "." & stopwatchPostPonded.Elapsed.Milliseconds)
885         System.Diagnostics.Debug.WriteLine("GATime: " & GATime)
886         System.Diagnostics.Debug.WriteLine("Wetting Front: " & WettingFront)
887         'END DEBUG MATERIAL
888
889         'DEBUG MATERIAL
890         System.Diagnostics.Debug.WriteLine("=====")
891         'END DEBUG MATERIAL
892
893
894         End While 'While loop for overall Green-Ampt infiltration simulation
895
896         'DEBUG MATERIAL
897         System.Diagnostics.Debug.WriteLine("=====")
898         'END DEBUG MATERIAL
899
900
901

```

```

902     '*****DARCY'S*****
903     'Calculates infiltration when soil is saturated based on Darcy's Law
904     'Based on equivalent conductivity for layered soils
905     Dim Keq As Double 'Equivalent hydraulic conductivity of layered soil
906     Dim DKtot As Double 'Running total of layer depth divided by conductivity for
calculating equivalent conductivity
907     Dim DrainageFlux As Double '(m/hr) Rate of water movement through the soil
908
909     Dim GravelPorosity As Double 'Porosity of gravel drainage layer
910     Dim GravelStorage As Double '(m) Depth of water in gravel drainage layer
911     Dim DrainHeight As Double '(m) Height of drain bottom above the gravel storage layer
912     Dim SubSoilKSat As Double '(m/hr) Saturated hydraulic conductivity of underlying soil
913     Dim PipeFlow As Double '(m) Volume of water leaving through the drain pipe
914     Dim Seepage As Double '(m) Volume of water leaving through underlying soil
915
916     'Update status label
917     Label_Status.Text = "Status: Calculating for saturated conditions"
918     Label_Status.Update()
919
920     'Load values from GUI
921     GravelPorosity = TB_GravelPorosity.Text
922     DrainHeight = TB_DrainHeight.Text
923     SubSoilKSat = TB_SubSoilKSat.Text
924
925     'Initial gravel storage is zero
926     GravelStorage = 0
927
928     Dim DarcyTime As Double
929     DarcyTime = GATime
930
931     'DEBUG LINES
932     'Starts stopwatch to observe execution time for Darcy's
933     Dim stopWatchDarcys As New Stopwatch()
934     stopWatchDarcys.Start()
935     'END DEBUG LINES

```

```

936
937     Dim CumulativeSeepage As Double 'Total seepage depth
938     Dim CumulativePipe As Double 'Total pipe depth
939
940
941     'Divides soil step depth by hydraulic conductivity at current position and adds to
total
942     'Intermediate step for calculating Keq
943     For GASoilIndex = 0 To GASoilArraySize
944         DKtot = DKtot + SoilStep / GASoilArray(GASoilIndex, 1)
945     Next GASoilIndex
946
947     'Calculate equivalent hydraulic conductivity for layered soil
948     Keq = SoilDepth / DKtot
949
950     'Performs Darcy's calculations until inflow ends
951     While GAIndex <= GAIndexLength
952
953         'Load rainfall rate
954         RainRate = GreenAmpArray(GAIndex, 3)
955
956         'Update total rainfall depth
957         TotalRain = TotalRain + RainRate * GATimeStep
958
959         'Calculate drainage flux based on Darcy's Law
960         'Assumes head at soil surface is equal to drain depth plus ponded depth
961         'Assumes zero pressure head at drain depth
962         DrainageFlux = Keq * (SoilDepth + SurfaceStorage) / SoilDepth
963
964         'If rainfall rate is less than drainage flux, restricts drainageflux to rainfall
rate
965         If RainRate < DrainageFlux Then
966
967             'If there is no surface storage at beginning of timestep
968             If SurfaceStorage <= 0 Then

```

```

969         DrainageFlux = RainRate
970         SurfaceStorage = 0
971     End If
972
973     'If surface ponding will be depleted during current timestep
974     If SurfaceStorage + (RainRate - DrainageFlux) * GATimeStep < 0 Then
975         DrainageFlux = RainRate + SurfaceStorage / GATimeStep
976         SurfaceStorage = 0
977     End If
978
979     'If surface ponding still exists
980     DrainageFlux = DrainageFlux
981     SurfaceStorage = SurfaceStorage + (RainRate - DrainageFlux) * GATimeStep
982
983     'If rainfall rate is greater than or equal to drainage flux
984 Else
985     DrainageFlux = DrainageFlux
986     'Update surface storage
987     SurfaceStorage = SurfaceStorage + (RainRate - DrainageFlux) * GATimeStep
988 End If
989
990
991 'Updates cumulative infiltration amount
992 CumulativeInf = CumulativeInf + DrainageFlux * GATimeStep
993
994 'Determines if there is overflow
995 If SurfaceStorage > StorageCapacity Then
996     'If there is overflow, calculate overflow depth and set surface storage at
997     capacity level
998     Overflowed = (RainRate - DrainageFlux) * GATimeStep - (SurfaceStorage -
999     StorageCapacity)
1000     HydrologyArray(GAIndex, 1) = Overflowed / GATimeStep
1001     CumulativeOverflow = CumulativeOverflow + Overflowed
1002     SurfaceStorage = StorageCapacity
1003 End If

```

```

1002
1003     'Store DrainageFlux to GreenAmptArray
1004     GreenAmptArray(GAIndex, 2) = DrainageFlux
1005
1006     GreenAmptArray(GAIndex, 0) = GADate 'Date stored in array
1007     HydrologyArray(GAIndex, 0) = GADate 'Date stored in array for indexing
1008
1009     'Store Wetting Front Depth to GreenAmptArray
1010     GreenAmptArray(GAIndex, 1) = WettingFront
1011
1012
1013     '-----***-----
1014     'Track where outflow from soil goes
1015
1016     'If drainage flux exceeds infiltration capacity of subsoil
1017     If DrainageFlux > SubSoilKSat Then
1018         'Seepage limited by SubSoilKSat
1019         Seepage = SubSoilKSat
1020         HydrologyArray(GAIndex, 2) = Seepage 'Store seepage to output array
1021         'Update total seepage depth
1022         CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1023
1024         'Calculates change in water level within gravel storage layer
1025         GravelStorage = GravelStorage + ((DrainageFlux - SubSoilKSat) * GATimeStep) /
GravelPorosity
1026     'If water level in gravel exceeds drain height, all water above drain height
leaves through the drain
1027     If GravelStorage > DrainHeight Then
1028         PipeFlow = (GravelStorage - DrainHeight) * GravelPorosity
1029         HydrologyArray(GAIndex, 3) = PipeFlow / GATimeStep 'Store pipe flow to
output array
1030         'Update total pipe drainage depth
1031         CumulativePipe = CumulativePipe + PipeFlow
1032         'Set water level in gravel storage at drain height
1033         GravelStorage = DrainHeight

```

```

1034         End If
1035
1036         'If drainage flux does not exceed infiltration capacity of subsoil
1037 Else
1038
1039         If GravelStorage = 0 Then
1040             'All water is lost to seepage
1041             Seepage = DrainageFlux
1042             CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1043         End If
1044
1045         'If seepage would deplete gravel layer
1046         If GravelStorage + ((DrainageFlux - SubSoilKSat) * GATimeStep) /
GravelPorosity <= 0 Then
1047             'Calculate seepage to account for incoming drainage flux and depletion of
storage
1048             Seepage = DrainageFlux + GravelStorage / GATimeStep
1049             HydrologyArray(GAIndex, 2) = Seepage 'Store seepage to output array
1050             'Update cumulative seepage depth
1051             CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1052             GravelStorage = 0 'Gravel storage layer is depleted
1053         End If
1054
1055         'If change in storage does not completely deplete water level in storage layer
1056         If GravelStorage + ((DrainageFlux - SubSoilKSat) * GATimeStep) /
GravelPorosity > 0 Then
1057             GravelStorage = GravelStorage + ((DrainageFlux - SubSoilKSat) *
GATimeStep) / GravelPorosity
1058
1059             'Seepage limited by SubSoilKSat since storage will not be depleted
1060             Seepage = DrainageFlux
1061             HydrologyArray(GAIndex, 2) = Seepage 'Store seepage to output array
1062             'Update cumulative seepage depth
1063             CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1064         End If

```

```

1065
1066     End If
1067
1068     'Advance time variables
1069     DarcyTime = DarcyTime + GATimeStep
1070     DarcyTime = Round(DarcyTime, 3) 'Round to prevent truncation error
1071     GAdate = GAdate.AddSeconds(GATimeStep * 60 * 60) 'Date advanced
1072     GAIndex = GAIndex + 1
1073
1074     'TEMPORARY DEBUG LINE
1075     VolCheck = TotalRain / (CumulativeInf + SurfaceStorage)
1076     'System.Diagnostics.Debug.WriteLine(VolCheck)
1077     'END TEMPORARY LINE
1078
1079     End While
1080
1081
1082     'DEBUG MATERIAL
1083     'Stops execution time stopwatch and prints execution time in Debug line
1084     stopWatchDarcys.Stop()
1085     System.Diagnostics.Debug.WriteLine("Darcy's RunTime: " &
stopWatchDarcys.Elapsed.Seconds & "." & stopWatchDarcys.Elapsed.Milliseconds)
1086     'END DEBUG MATERIAL
1087
1088
1089     'DEBUG LINES
1090     'Starts stopwatch to observe drainage phase execution time
1091     Dim stopWatchDrainPhase As New Stopwatch()
1092     stopWatchDrainPhase.Start()
1093     'END DEBUG LINES
1094
1095     'Update status label
1096     Label_Status.Text = "Status: Draining water from soil profile"
1097     Label_Status.Update()
1098

```

```

1099      '**-----**
1100      'Calculate hydraulics after rainfall has ended
1101      'Assume profile is entirely saturated
1102      Dim SurfaceDrainTime As Double '(hr) Time needed to drain surface storage
1103      Dim DrainIndex As Integer 'Index for storing to DrainArray
1104      Dim DrainArray(,) As String 'Main array for profile drainage calculations
1105      ReDim DrainArray(2, 1000)
1106      'DrainArray Contents: Time (Date), DrainFront (m), Infil (m/hr)
1107      'Rows and Column are switched for this array to preserve data for dynamic sizing
1108      DrainIndex = -1
1109      Dim DrainArrayLength As Integer 'Length of DrainArray, used during redim and
referenced later in code
1110      Dim DrainFront As Double '(m) Distance of receding water above the drainage layer
1111      DrainFront = SoilDepth 'Set initial DrainFront at the top of the soil profile
1112      Dim DrainSoilIndex As Integer 'Location of drain front in soil profile for referencing
layered properties
1113
1114      Dim DrainHydrologyArray(,) As String 'Same structure as Green-Ampt array but applies
to gravel layer and overflow
1115      'HydrologyArray Contents: Time (Date), Overflow (m/hr), Underlying Soil Infiltration
(m/hr), Pipe Flow (m/hr)
1116      ReDim DrainHydrologyArray(3, 1000)
1117      'Array is transposed from earlier hydrology array due to redimensioning
1118
1119      'Initial DrainArrayLength is 1000
1120      DrainArrayLength = 1000
1121
1122      'If water is ponded on the surface, drain ponded water first following Darcy's
1123      While SurfaceStorage > 0
1124
1125          'Advance DrainIndex
1126          DrainIndex = DrainIndex + 1
1127
1128          'If array size needs to be increased
1129          If DrainIndex > DrainArrayLength Then

```

```

1130         'Increase DrainArrayLength size
1131         'DrainArray is resized in this fashion to reduce memory demands of copying
entire array for each timestep
1132         DrainArrayLength = DrainArrayLength + 1000
1133         'Resize DrainArray and DrainHydrologyArray
1134         ReDim Preserve DrainArray(2, DrainArrayLength)
1135         ReDim Preserve DrainHydrologyArray(3, DrainArrayLength)
1136     End If
1137
1138     'Store current date into DrainHydrologyArray
1139     DrainHydrologyArray(0, DrainIndex) = GADate
1140
1141     'Calculate drainage flux based on Darcy's Law
1142     'Assumes head at soil surface is equal to drain depth plus ponded depth
1143     'Assumes zero pressure head at drain depth
1144     DrainageFlux = Keq * (SoilDepth + SurfaceStorage) / SoilDepth
1145
1146     'If DrainageFlux will result in negative surface storage
1147     If SurfaceStorage - DrainageFlux * GATimeStep < 0 Then
1148         'Determine time (hrs) needed to deplete surface storage
1149         SurfaceDrainTime = SurfaceStorage / DrainageFlux
1150         SurfaceStorage = 0
1151         'Store results to array
1152         DrainArray(0, DrainIndex) = GADate
1153         DrainArray(1, DrainIndex) = SoilDepth
1154         DrainArray(2, DrainIndex) = DrainageFlux
1155         'Advance date based on time required to drain surface
1156         GADate = GADate.AddSeconds(SurfaceDrainTime * 60 * 60)
1157
1158     'If DrainageFlux does not deplete surface storage
1159 Else
1160     'Calculate new surface storage
1161     SurfaceStorage = SurfaceStorage - DrainageFlux * GATimeStep
1162     'Store results to array
1163     DrainArray(0, DrainIndex) = GADate

```

```

1164         DrainArray(1, DrainIndex) = SoilDepth + SurfaceStorage
1165         DrainArray(2, DrainIndex) = DrainageFlux
1166         'Advance times
1167         DarcyTime = DarcyTime + GATimeStep
1168         DarcyTime = Round(DarcyTime, 3) 'Round to prevent truncation error
1169         GADate = GADate.AddSeconds(GATimeStep * 60 * 60) 'Date advanced
1170     End If
1171
1172     'Track where outflow from soil goes
1173     'If drainage flux exceeds infiltration capacity of subsoil
1174     If DrainageFlux > SubSoilKSat Then
1175         'Seepage limited by SubSoilKSat
1176         Seepage = SubSoilKSat
1177         DrainHydrologyArray(2, DrainIndex) = Seepage 'Store seepage to output array
1178         'Update cumulative seepage depth
1179         CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1180
1181         'Calculates change in water level within gravel storage layer
1182         GravelStorage = GravelStorage + (DrainageFlux - SubSoilKSat) * GATimeStep /
GravelPorosity
1183         'If water level in gravel exceeds drain height, all water above drain height
leaves through the drain
1184         If GravelStorage > DrainHeight Then
1185             PipeFlow = (GravelStorage - DrainHeight) * GravelPorosity
1186             DrainHydrologyArray(3, DrainIndex) = PipeFlow / GATimeStep 'Store pipe
flow to output array
1187             'Update cumulative seepage depth
1188             CumulativePipe = CumulativePipe + PipeFlow
1189
1190             GravelStorage = DrainHeight
1191         End If
1192         'If drainage flux does not exceed infiltration capacity of subsoil
1193     Else
1194         'If change in storage does not completely deplete water level in storage layer
1195         If GravelStorage + (DrainageFlux - SubSoilKSat) * GATimeStep / GravelPorosity

```

```

1196 > 0 Then
1196         GravelStorage = GravelStorage + (DrainageFlux - SubSoilKSat) * GATimeStep
1197         / GravelPorosity
1197         'Seepage limited by SubSoilKSat since storage will not be depleted
1198         Seepage = SubSoilKSat
1199         DrainHydrologyArray(2, DrainIndex) = Seepage 'Store seepage to output
1200     array
1200         'Update cumulative seepage depth
1201         CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1202
1203         'If gravel storage layer would be depleted
1204     Else
1205         'Calculate seepage to account for incoming drainage flux and depletion of
1206     storage
1206         Seepage = DrainageFlux + GravelStorage / GATimeStep
1207         DrainHydrologyArray(2, DrainIndex) = Seepage 'Store seepage to output
1208     array
1208         'Update cumulative seepage depth
1209         CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1210
1211         GravelStorage = 0 'Gravel storage layer is depleted
1212     End If
1213 End If
1214
1215 End While 'While surface storage is being depleted
1216
1217
1218 'While water still remains in the soil profile but surface ponding has been removed
1219 While DrainFront > 0
1220
1221     'Advance DrainIndex
1222     DrainIndex = DrainIndex + 1
1223
1224     'If array size needs to be increased
1225     If DrainIndex > DrainArrayLength Then

```

```

1226         'Increase DrainArrayLength size
1227         'DrainArray is resized in this fashion to reduce memory demands of copying
entire array for each timestep
1228         DrainArrayLength = DrainArrayLength + 1000
1229         'Resize DrainArray and DrainHydrologyArray
1230         ReDim Preserve DrainArray(2, DrainArrayLength)
1231         ReDim Preserve DrainHydrologyArray(3, DrainArrayLength)
1232     End If
1233
1234         'Store current date into DrainHydrologyArray
1235         DrainHydrologyArray(0, DrainIndex) = GADate
1236
1237         'NOTE: If soil step is changed, need to change rounding when Soil properties are
loaded based on drain front position
1238         DrainSoilIndex = Round(SoilDepth - DrainFront, 3) / SoilStep
1239
1240         'NOTE: To use layered soil properties, need to calculate Keq for each timestep
based on position of drainage front
1241
1242         'DKtot = 0 'Reset DKtot on each loop to calculate new Keq
1243
1244         ''Divides soil step depth by hydraulic conductivity at current position and adds
to total
1245         ''Intermediate step for calculating Keq
1246         'For GASoilIndex = DrainSoilIndex To GASoilArraySize
1247         '     DKtot = DKtot + SoilStep / GASoilArray(DrainSoilIndex, 1)
1248         'Next GASoilIndex
1249
1250         ''Calculate equivalent hydraulic conductivity for layered soil
1251         'Keq = DrainFront / DKtot
1252
1253         'Calculate drainage flux based on Darcy's Law
1254         'Assumes head is equal to depth of water above the drain
1255         'Assumes zero pressure head at drain depth
1256         'NOTE: Currently uses Keq calculated for saturated profile

```

```

1257 DrainageFlux = Keq
1258
1259 ThetaFilled = GASoilArray(DrainSoilIndex, 3) 'Load fillable porosity for current
position
1260
1261 'Update DrainFront based on DrainageFlux
1262 DrainFront = DrainFront - (DrainageFlux * GATimeStep) / ThetaFilled
1263
1264 'Store results to array
1265 DrainArray(0, DrainIndex) = GADate
1266 DrainArray(1, DrainIndex) = DrainFront
1267 DrainArray(2, DrainIndex) = DrainageFlux
1268
1269 'Advance times
1270 DarcyTime = DarcyTime + GATimeStep
1271 DarcyTime = Round(DarcyTime, 3) 'Round to prevent truncation error
1272 GADate = GADate.AddSeconds(GATimeStep * 60 * 60) 'Date advanced
1273
1274
1275 'Track where outflow from soil goes
1276 'If drainage flux exceeds infiltration capacity of subsoil
1277 If DrainageFlux > SubSoilKSat Then
1278     'Seepage limited by SubSoilKSat
1279     Seepage = SubSoilKSat
1280     DrainHydrologyArray(2, DrainIndex) = Seepage 'Store seepage to output array
1281     'Update cumulative seepage depth
1282     CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1283
1284     'Calculates change in water level within gravel storage layer
1285     GravelStorage = GravelStorage + (DrainageFlux - SubSoilKSat) * GATimeStep /
GravelPorosity
1286     'If water level in gravel exceeds drain height, all water above drain height
leaves through the drain
1287     If GravelStorage > DrainHeight Then
1288         PipeFlow = (GravelStorage - DrainHeight) * GravelPorosity

```

```

1289             DrainHydrologyArray(3, DrainIndex) = PipeFlow / GATimeStep 'Store pipe
flow to output array
1290             'Update cumulative pipe drainage depth
1291             CumulativePipe = CumulativePipe + PipeFlow
1292
1293             GravelStorage = DrainHeight
1294         End If
1295         'If drainage flux does not exceed infiltration capacity of subsoil
1296     Else
1297         'If change in storage does not completely deplete water level in storage layer
1298         If GravelStorage + (DrainageFlux - SubSoilKSat) * GATimeStep / GravelPorosity
> 0 Then
1299             GravelStorage = GravelStorage + (DrainageFlux - SubSoilKSat) * GATimeStep
/ GravelPorosity
1300             'Seepage limited by SubSoilKSat since storage will not be depleted
1301             Seepage = SubSoilKSat
1302             DrainHydrologyArray(2, DrainIndex) = Seepage 'Store seepage to output
array
1303             'Update cumulative seepage depth
1304             CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1305
1306             'If gravel storage layer would be depleted
1307         Else
1308             'Calculate seepage to account for incoming drainage flux and depletion of
storage
1309             Seepage = DrainageFlux + GravelStorage / GATimeStep
1310             DrainHydrologyArray(2, DrainIndex) = Seepage 'Store seepage to output
array
1311             'Update cumulative seepage depth
1312             CumulativeSeepage = CumulativeSeepage + Seepage * GATimeStep
1313             GravelStorage = 0 'Gravel storage layer is depleted
1314         End If
1315     End If
1316
1317 End While 'Removing remaining water from soil profile

```

```

1318
1319     'Calculate final length of DrainArray
1320     DrainArrayLength = DrainIndex
1321
1322     'Resize DrainArray and DrainHydrologyArray to reflect final array size
1323     ReDim Preserve DrainArray(2, DrainArrayLength)
1324     ReDim Preserve DrainHydrologyArray(3, DrainArrayLength)
1325
1326     'Find time when drainage stops
1327     Dim DrainTimeStop As Date
1328     DrainTimeStop = GADate.AddSeconds(-GATimeStep * 60 * 60) 'Substract timestep to
account for advance at tne do loop
1329
1330     'Find time length of drainage phase
1331     Dim DrainTimeLength As Double 'Length of time for drainage wet soil simulations (s)
1332     DrainTimeLength = DrainArrayLength * GATimeStep * 60 * 60
1333
1334
1335     'DEBUG MATERIAL
1336     'Stops execution time stopwatch and prints execution time in Debug line
1337     stopWatchDrainPhase.Stop()
1338     System.Diagnostics.Debug.WriteLine("Drain Phase RunTime: " &
stopWatchDrainPhase.Elapsed.Seconds & "." & stopWatchDrainPhase.Elapsed.Milliseconds)
1339     'END DEBUG MATERIAL
1340
1341
1342     'DEBUG LINES
1343     'Starts stopwatch to observe hydrology combo execution time
1344     Dim stopWatchHydrologyCombo As New Stopwatch()
1345     stopWatchHydrologyCombo.Start()
1346     'END DEBUG LINES
1347
1348     '-----*****-----
1349     'Build array containing all hydraulic information for during and after storm
1350     'Dim HydraulicComboArray(,) As String 'Array containing both sets of hydraulic

```

```

calculations
1351 'HydraulicComboArray Contents: Date, Drainage Flux (m/hr)
1352 'Dim HydraulicComboArrayLength As Integer 'Length of array
1353 HydraulicComboArrayLength = GAIndexLength + DrainArrayLength
1354 'Redimension HydraulicComboArray
1355 ReDim HydraulicComboArray(HydraulicComboArrayLength, 1)
1356 Dim HydraulicComboStoreIndex As Integer 'Index for storing values into Combo arrays
1357 Dim HydraulicComboLoadIndex As Integer 'Index for loading values from Hydraulic arrays
1358
1359 ''Set Index values at 1 to prevent copying first value, which is zero
1360 'HydraulicComboLoadIndex = 1
1361
1362 'Update status label
1363 Label_Status.Text = "Status: Combining hydrology arrays"
1364 Label_Status.Update()
1365
1366 'While copying values from array during storm event
1367 While HydraulicComboStoreIndex <= GAIndexLength
1368
1369     'Copy date
1370     HydraulicComboArray(HydraulicComboStoreIndex, 0) =
GreenAmptArray(HydraulicComboLoadIndex, 0)
1371     'Copy infiltration rate
1372     HydraulicComboArray(HydraulicComboStoreIndex, 1) =
GreenAmptArray(HydraulicComboLoadIndex, 2)
1373
1374     'Advance both index values
1375     HydraulicComboStoreIndex = HydraulicComboStoreIndex + 1
1376     HydraulicComboLoadIndex = HydraulicComboLoadIndex + 1
1377 End While
1378
1379
1380 'Reset load index because about to pull from new array
1381 HydraulicComboLoadIndex = 1
1382 While HydraulicComboStoreIndex <= HydraulicComboArrayLength

```

```

1383
1384         'Copy date
1385         HydraulicComboArray(HydraulicComboStoreIndex, 0) = DrainArray(0,
HydraulicComboLoadIndex)
1386         'Copy infiltration rate
1387         HydraulicComboArray(HydraulicComboStoreIndex, 1) = DrainArray(2,
HydraulicComboLoadIndex)
1388
1389         'Advance both index values
1390         HydraulicComboStoreIndex = HydraulicComboStoreIndex + 1
1391         HydraulicComboLoadIndex = HydraulicComboLoadIndex + 1
1392     End While
1393
1394
1395     '-----
1396     'Write Hydraulic results to an output file
1397     Dim theWriterGA As System.IO.StreamWriter
1398     theWriterGA = My.Computer.FileSystem.OpenTextFileWriter(TB_HydraulicOutFile.Text,
False)
1399     'Write summary data
1400     theWriterGA.WriteLine("Dry Pavement Simulation")
1401     theWriterGA.WriteLine("")
1402     theWriterGA.WriteLine("Model Outputs:")
1403     theWriterGA.WriteLine("")
1404
1405     'Write array data
1406     theWriterGA.WriteLine("Date" & "," & "Wetting Front (m)" & "," & "Infil Rate (m/hr)" &
", " & "Eff. Rain Rate (m/hr)")
1407     Dim OutRowGA As Integer
1408     Dim OutColGA As Integer
1409
1410     For OutRowGA = 0 To GAIndexLength - 1
1411         'Writes GA-array contents output table
1412         For OutColGA = 0 To 3
1413             theWriterGA.Write(GreenAmptArray(OutRowGA, OutColGA) & ",")

```

```

1414         Next OutColGA
1415         theWriterGA.WriteLine()
1416     Next OutRowGA
1417
1418     'Writes contents for drainage of soil profile after rainfall
1419     For OutColGA = 0 To DrainIndex
1420         For OutRowGA = 0 To 2
1421             theWriterGA.Write(DrainArray(OutRowGA, OutColGA) & ",")
1422         Next OutRowGA
1423         theWriterGA.Write("0") 'Write zero for effective rainfall rate
1424         theWriterGA.WriteLine()
1425     Next OutColGA
1426
1427     theWriterGA.Close()
1428
1429
1430
1431     '-----*****-----
1432     'Build array containing all hydrology information for during and after storm
1433     Dim HydrologyComboArray(,) As String 'Array containing both sets of hydrology
calculations
1434     'HydrologyArray Contents: Time (Date), Overflow (m/hr), Underlying Soil Infiltration
(m/hr), Pipe Flow (m/hr)
1435     Dim HydrologyComboArrayLength As Integer 'Length of array
1436     HydrologyComboArrayLength = GAIndexLength + DrainArrayLength
1437     'Redimension HydrologyComboArray
1438     ReDim HydrologyComboArray(HydrologyComboArrayLength, 3)
1439     Dim HydrologyComboStoreIndex As Integer 'Index for storing values into Combo arrays
1440     Dim HydrologyComboLoadIndex As Integer 'Index for loading values from Hydrology arrays
1441
1442     'While copying values from array during storm event
1443     While HydrologyComboStoreIndex <= GAIndexLength
1444         'Copy date
1445         HydrologyComboArray(HydrologyComboStoreIndex, 0) =
HydrologyArray(HydrologyComboLoadIndex, 0)

```

```

1446         'Copy overflow rate
1447         HydrologyComboArray(HydrologyComboStoreIndex, 1) =
HydrologyArray(HydrologyComboLoadIndex, 1)
1448         'Copy seepage rate
1449         HydrologyComboArray(HydrologyComboStoreIndex, 2) =
HydrologyArray(HydrologyComboLoadIndex, 2)
1450         'Copy pipe rate
1451         HydrologyComboArray(HydrologyComboStoreIndex, 3) =
HydrologyArray(HydrologyComboLoadIndex, 3)
1452
1453         'Advance both index values
1454         HydrologyComboStoreIndex = HydrologyComboStoreIndex + 1
1455         HydrologyComboLoadIndex = HydrologyComboLoadIndex + 1
1456     End While
1457
1458     'Reset load index because about to pull from new array
1459     HydrologyComboLoadIndex = 0
1460     While HydrologyComboStoreIndex <= HydrologyComboArrayLength
1461         'Copy date
1462         HydrologyComboArray(HydrologyComboStoreIndex, 0) = DrainHydrologyArray(0,
HydrologyComboLoadIndex)
1463         'Copy overflow rate
1464         HydrologyComboArray(HydrologyComboStoreIndex, 1) = DrainHydrologyArray(1,
HydrologyComboLoadIndex)
1465         'Copy seepage rate
1466         HydrologyComboArray(HydrologyComboStoreIndex, 2) = DrainHydrologyArray(2,
HydrologyComboLoadIndex)
1467         'Copy pipe rate
1468         HydrologyComboArray(HydrologyComboStoreIndex, 3) = DrainHydrologyArray(3,
HydrologyComboLoadIndex)
1469
1470         'Advance both index values
1471         HydrologyComboStoreIndex = HydrologyComboStoreIndex + 1
1472         HydrologyComboLoadIndex = HydrologyComboLoadIndex + 1
1473     End While

```

```

1474
1475     'DEBUG MATERIAL
1476     'Stops execution time stopwatch and prints execution time in Debug line
1477     stopWatchHydrologyCombo.Stop()
1478     System.Diagnostics.Debug.WriteLine("Hydrology Combo Arrays: " &
stopWatchHydrologyCombo.Elapsed.Seconds & "." & stopWatchHydrologyCombo.Elapsed.Milliseconds)
1479     'END DEBUG MATERIAL
1480
1481
1482
1483     '**_**_**_**
1484     'Calculate dry pavement temperature
1485     'Based on work of Van Buren et al. (2000)
1486
1487     'DEBUG LINES
1488     'Starts stopwatch to observe execution time for dry pavement temperature
1489     Dim stopWatchDryPave As New Stopwatch()
1490     stopWatchDryPave.Start()
1491     'END DEBUG LINES
1492
1493     'Update status label
1494     Label_Status.Text = "Status: Calculating initial pavement temperatures"
1495     Label_Status.Update()
1496
1497     'Calculate day to start dry pavement temperature simulations
1498     Dim DryPaveStartDate As Date 'Date to start dry pavement simulations
1499     'Subtracts time from StormDate to start dry pavement simulations
1500     'Extra time for dry pavement simulations reduces bias of initial temperature profile
estimate
1501     DryPaveStartDate = DateAdd(DateInterval.Day, -10, StormDate)
1502
1503     'Initiate date variables for loading weather data
1504     Dim SimDate As Date 'Current date in simulation in date format
1505     Dim SimMonth As Integer 'Current month in simulation
1506     Dim SimHour As Double 'Current hour of simulation including fraction

```

```

1507 Dim SimDay As Integer 'Current day in the year
1508 Dim DryTRMTimestep As Double '(sec) Timestep for dry pavement simulation
1509
1510 'Set SimDate as DryPaveStartDate
1511 SimDate = DryPaveStartDate
1512
1513 'Set timestep for calculating weater data and dry pavement temperatures
1514 'Increasing timestep can lead to stability issues
1515 DryTRMTimestep = 15 'seconds
1516
1517 '-----
1518 'Configure weather data for use in calculating pavement temperatures
1519
1520 'Wind speed data
1521 Dim WindSpeedArray(,) As Double
1522 ReDim WindSpeedArray(12, 0)
1523 'First row of array is blank
1524 'Each row is mean windspeed for the month of that row (m/s)
1525
1526 'Load weather data from xml file
1527 XMLOpenWeatherFileName = TB_WeatherFileName.Text
1528
1529 ' load the xml document from the given path
1530 Dim xml = New XmlDocument()
1531 xml.Load(XMLOpenWeatherFileName)
1532
1533 ' get the different nodes from the xml file and put them in the array
1534 ' NOTE: add a check for the values being Nothing.
1535 Dim L_WindSpeedJan = xml.SelectSingleNode("//WindSpeedJan")
1536 WindSpeedArray(1, 0) = L_WindSpeedJan.InnerText
1537
1538 Dim L_WindSpeedFeb = xml.SelectSingleNode("//WindSpeedFeb")
1539 WindSpeedArray(2, 0) = L_WindSpeedFeb.InnerText
1540
1541 Dim L_WindSpeedMar = xml.SelectSingleNode("//WindSpeedMar")

```

```

1542 WindSpeedArray(3, 0) = L_WindSpeedMar.InnerText
1543
1544 Dim L_WindSpeedApr = xml.SelectSingleNode("//WindSpeedApr")
1545 WindSpeedArray(4, 0) = L_WindSpeedApr.InnerText
1546
1547 Dim L_WindSpeedMay = xml.SelectSingleNode("//WindSpeedMay")
1548 WindSpeedArray(5, 0) = L_WindSpeedMay.InnerText
1549
1550 Dim L_WindSpeedJun = xml.SelectSingleNode("//WindSpeedJun")
1551 WindSpeedArray(6, 0) = L_WindSpeedJun.InnerText
1552
1553 Dim L_WindSpeedJul = xml.SelectSingleNode("//WindSpeedJul")
1554 WindSpeedArray(7, 0) = L_WindSpeedJul.InnerText
1555
1556 Dim L_WindSpeedAug = xml.SelectSingleNode("//WindSpeedAug")
1557 WindSpeedArray(8, 0) = L_WindSpeedAug.InnerText
1558
1559 Dim L_WindSpeedSep = xml.SelectSingleNode("//WindSpeedSep")
1560 WindSpeedArray(9, 0) = L_WindSpeedSep.InnerText
1561
1562 Dim L_WindSpeedOct = xml.SelectSingleNode("//WindSpeedOct")
1563 WindSpeedArray(10, 0) = L_WindSpeedOct.InnerText
1564
1565 Dim L_WindSpeedNov = xml.SelectSingleNode("//WindSpeedNov")
1566 WindSpeedArray(11, 0) = L_WindSpeedNov.InnerText
1567
1568 Dim L_WindSpeedDec = xml.SelectSingleNode("//WindSpeedDec")
1569 WindSpeedArray(12, 0) = L_WindSpeedDec.InnerText
1570
1571
1572 'Monthly air temperature data
1573 Dim MonthAirTempArray(,) As Double
1574 ReDim MonthAirTempArray(12, 2)
1575 'First row of array is blank
1576 'Each row is air temperature for the month of that row

```

```
1577 'Format: Min Temp (°C), Max Temp (°C), Avg Temp (°C)
1578
1579 'Load air temperatures from xml file
1580 'Minimum daily air temperatures
1581 Dim L_MinAirTempJan = xml.SelectSingleNode("//MinAirTempJan")
1582 MonthAirTempArray(1, 0) = L_MinAirTempJan.InnerText
1583
1584 Dim L_MinAirTempFeb = xml.SelectSingleNode("//MinAirTempFeb")
1585 MonthAirTempArray(2, 0) = L_MinAirTempFeb.InnerText
1586
1587 Dim L_MinAirTempMar = xml.SelectSingleNode("//MinAirTempMar")
1588 MonthAirTempArray(3, 0) = L_MinAirTempMar.InnerText
1589
1590 Dim L_MinAirTempApr = xml.SelectSingleNode("//MinAirTempApr")
1591 MonthAirTempArray(4, 0) = L_MinAirTempApr.InnerText
1592
1593 Dim L_MinAirTempMay = xml.SelectSingleNode("//MinAirTempMay")
1594 MonthAirTempArray(5, 0) = L_MinAirTempMay.InnerText
1595
1596 Dim L_MinAirTempJun = xml.SelectSingleNode("//MinAirTempJun")
1597 MonthAirTempArray(6, 0) = L_MinAirTempJun.InnerText
1598
1599 Dim L_MinAirTempJul = xml.SelectSingleNode("//MinAirTempJul")
1600 MonthAirTempArray(7, 0) = L_MinAirTempJul.InnerText
1601
1602 Dim L_MinAirTempAug = xml.SelectSingleNode("//MinAirTempAug")
1603 MonthAirTempArray(8, 0) = L_MinAirTempAug.InnerText
1604
1605 Dim L_MinAirTempSep = xml.SelectSingleNode("//MinAirTempSep")
1606 MonthAirTempArray(9, 0) = L_MinAirTempSep.InnerText
1607
1608 Dim L_MinAirTempOct = xml.SelectSingleNode("//MinAirTempOct")
1609 MonthAirTempArray(10, 0) = L_MinAirTempOct.InnerText
1610
1611 Dim L_MinAirTempNov = xml.SelectSingleNode("//MinAirTempNov")
```

```

1612     MonthAirTempArray(11, 0) = L_MinAirTempNov.InnerText
1613
1614     Dim L_MinAirTempDec = xml.SelectSingleNode("//MinAirTempDec")
1615     MonthAirTempArray(12, 0) = L_MinAirTempDec.InnerText
1616
1617     'Maximum daily temperatures
1618     Dim L_MaxAirTempJan = xml.SelectSingleNode("//MaxAirTempJan")
1619     MonthAirTempArray(1, 1) = L_MaxAirTempJan.InnerText
1620
1621     Dim L_MaxAirTempFeb = xml.SelectSingleNode("//MaxAirTempFeb")
1622     MonthAirTempArray(2, 1) = L_MaxAirTempFeb.InnerText
1623
1624     Dim L_MaxAirTempMar = xml.SelectSingleNode("//MaxAirTempMar")
1625     MonthAirTempArray(3, 1) = L_MaxAirTempMar.InnerText
1626
1627     Dim L_MaxAirTempApr = xml.SelectSingleNode("//MaxAirTempApr")
1628     MonthAirTempArray(4, 1) = L_MaxAirTempApr.InnerText
1629
1630     Dim L_MaxAirTempMay = xml.SelectSingleNode("//MaxAirTempMay")
1631     MonthAirTempArray(5, 1) = L_MaxAirTempMay.InnerText
1632
1633     Dim L_MaxAirTempJun = xml.SelectSingleNode("//MaxAirTempJun")
1634     MonthAirTempArray(6, 1) = L_MaxAirTempJun.InnerText
1635
1636     Dim L_MaxAirTempJul = xml.SelectSingleNode("//MaxAirTempJul")
1637     MonthAirTempArray(7, 1) = L_MaxAirTempJul.InnerText
1638
1639     Dim L_MaxAirTempAug = xml.SelectSingleNode("//MaxAirTempAug")
1640     MonthAirTempArray(8, 1) = L_MaxAirTempAug.InnerText
1641
1642     Dim L_MaxAirTempSep = xml.SelectSingleNode("//MaxAirTempSep")
1643     MonthAirTempArray(9, 1) = L_MaxAirTempSep.InnerText
1644
1645     Dim L_MaxAirTempOct = xml.SelectSingleNode("//MaxAirTempOct")
1646     MonthAirTempArray(10, 1) = L_MaxAirTempOct.InnerText

```

```
1647
1648 Dim L_MaxAirTempNov = xml.SelectSingleNode("//MaxAirTempNov")
1649 MonthAirTempArray(11, 1) = L_MaxAirTempNov.InnerText
1650
1651 Dim L_MaxAirTempDec = xml.SelectSingleNode("//MaxAirTempDec")
1652 MonthAirTempArray(12, 1) = L_MaxAirTempDec.InnerText
1653
1654 'Average daily temperatures
1655 Dim L_MeanAirTempJan = xml.SelectSingleNode("//MeanAirTempJan")
1656 MonthAirTempArray(1, 2) = L_MeanAirTempJan.InnerText
1657
1658 Dim L_MeanAirTempFeb = xml.SelectSingleNode("//MeanAirTempFeb")
1659 MonthAirTempArray(2, 2) = L_MeanAirTempFeb.InnerText
1660
1661 Dim L_MeanAirTempMar = xml.SelectSingleNode("//MeanAirTempMar")
1662 MonthAirTempArray(3, 2) = L_MeanAirTempMar.InnerText
1663
1664 Dim L_MeanAirTempApr = xml.SelectSingleNode("//MeanAirTempApr")
1665 MonthAirTempArray(4, 2) = L_MeanAirTempApr.InnerText
1666
1667 Dim L_MeanAirTempMay = xml.SelectSingleNode("//MeanAirTempMay")
1668 MonthAirTempArray(5, 2) = L_MeanAirTempMay.InnerText
1669
1670 Dim L_MeanAirTempJun = xml.SelectSingleNode("//MeanAirTempJun")
1671 MonthAirTempArray(6, 2) = L_MeanAirTempJun.InnerText
1672
1673 Dim L_MeanAirTempJul = xml.SelectSingleNode("//MeanAirTempJul")
1674 MonthAirTempArray(7, 2) = L_MeanAirTempJul.InnerText
1675
1676 Dim L_MeanAirTempAug = xml.SelectSingleNode("//MeanAirTempAug")
1677 MonthAirTempArray(8, 2) = L_MeanAirTempAug.InnerText
1678
1679 Dim L_MeanAirTempSep = xml.SelectSingleNode("//MeanAirTempSep")
1680 MonthAirTempArray(9, 2) = L_MeanAirTempSep.InnerText
1681
```

```

1682 Dim L_MeanAirTempOct = xml.SelectSingleNode("//MeanAirTempOct")
1683 MonthAirTempArray(10, 2) = L_MeanAirTempOct.InnerText
1684
1685 Dim L_MeanAirTempNov = xml.SelectSingleNode("//MeanAirTempNov")
1686 MonthAirTempArray(11, 2) = L_MeanAirTempNov.InnerText
1687
1688 Dim L_MeanAirTempDec = xml.SelectSingleNode("//MeanAirTempDec")
1689 MonthAirTempArray(12, 2) = L_MeanAirTempDec.InnerText
1690
1691
1692 '-----
1693 'Calculate solar radiation for use in pavement temperature calculations
1694 Dim SolarDec As Double 'Solar declination (radians)
1695 Dim Latitude As Double 'Latitude of site (radians)
1696 Dim Longitude As Double 'Longitude of site location (radians)
1697 Dim LatDeg As Double 'Latitude of site (degrees)
1698 Dim LonDeg As Double 'Longitude of site (degrees)
1699 Dim SolarNoon As Double 'Time of solar noon (hours)
1700 Dim ZenithAngle As Double 'Zenith angle of the sun (radians)
1701 Dim SolarConstant As Double 'Solar constant (W/m2)
1702 Dim Transmittance As Double 'Transmittance of atmosphere (W/m2)
1703 Dim Elevation As Double 'Elevation of site (m)
1704 Dim BaroPress As Double 'Average barometric pressure (kPa)
1705 Dim OptiMassNum As Double 'm term
1706 Dim BeamRadiation As Double ' Sp term, direct radiation (W/m2)
1707 Dim DiffRadiation As Double ' Sd term, diffuse radiation (W/m2)
1708 Dim SolarRadiation As Double ' Rs term, total solar radiation (W/m2)
1709
1710 'Initialize variables for calculating sunrise, sunset, and solar noon
1711 Dim FractionalYear As Double 'Fractional year (radians)
1712 Dim EquationOfTime As Double 'Equation of time (minutes)
1713 Dim SolarTimeDec As Double 'Solar declination (radians) for sunrise, sunset
calculations
1714 Dim HourAngle As Double 'Hour angle used for sunrise and sunset
1715 Dim SunriseTime As Double 'Hours until sunrise (hours)

```

```

1716 Dim SunsetTime As Double 'Hours until sunset (hours)
1717 Dim Timezone As Double 'Correction for timezone (hours)
1718
1719 'Set solar constant
1720 SolarConstant = 1360
1721
1722 'Load variables from GUI
1723 Transmittance = TB_Transmittance.Text
1724 LatDeg = TB_LatDeg.Text
1725 LonDeg = TB_LonDeg.Text
1726 Elevation = TB_Elevation.Text
1727 Timezone = TB_Timezone.Text
1728
1729 'Calculate static variables for solar radiation calculations
1730 BaroPress = 101.3 * Exp(-Elevation / 8200)
1731
1732 'Convert latitude and longitude from degrees to radians
1733 Latitude = LatDeg * PI / 180
1734 Longitude = LonDeg * PI / 180
1735
1736 'Initialize variables for calculating air temperature
1737 Dim MaxAirTemp As Double 'Daily maximum air temperature (°C)
1738 Dim MinAirTemp As Double 'Daily minimum air temperature (°C)
1739 Dim AirTemp As Double 'Current air temperature in simulation (°C)
1740 Dim MaxAirTempTime As Double 'Time when daily maximum air temperature occurs (hrs)
1741 Dim MinAirTempTime As Double 'Time when daily minimum air temperature occurs (hrs)
1742 Dim AirTempTbar As Double 'Intermediate variable associated with time
1743
1744 'Set Max and Min AirTempTime based on recommendations in Satyamurty and Babu, 1999
1745 MaxAirTempTime = 14.5
1746 MinAirTempTime = 6.5
1747
1748 'Dimension static variables that are used for both dry and wet simulations
1749 Dim Pave_Depth As Double '(m) Depth of asphalt
1750 Dim Pave_Density As Double '(kg/m³)

```

```

1751 Dim Pave_ThermalK As Double '(W/m °C) Thermal conductivity
1752 Dim Pave_Cp As Double '(J/kg °C) Pavement specific heat
1753 Dim Pave_ThermalAlpha As Double '(m²/s) Pavement thermal diffusivity
1754 Dim Gravel_Depth As Double '(m) Depth of gravel base
1755 Dim Gravel_Density As Double '(kg/m³)
1756 Dim Gravel_ThermalK As Double '(W/m °C) Thermal conductivity
1757 Dim Gravel_Cp As Double '(J/kg °C) Gravel specific heat
1758 Dim Gravel_ThermalAlpha As Double '(m²/s) Gravel thermal diffusivity
1759 Dim Water_Density As Double '(kg/m³)
1760
1761 'Dimension static variables
1762 Dim DryTRMGridSpacing As Double '(m) Depth increment for finite difference
calculations
1763 Dim DryTRMDepthIndex As Integer 'Location in pavement at current step
1764
1765 'Dimension weather variables
1766 'Dim DryAirTemperatureK As Double '(K) Air temperature in Kelvin
1767 Dim DryWindSpeed As Double 'Wind speed (m/s)
1768
1769 'Dimension intermediate variables
1770 Dim DryPaveConvCoeff As Double '(W/m² °C) Convective heat transfer coefficient
1771
1772 'Dimension result variables
1773 Dim DrySurfaceTemp As Double '(°C) Asphalt surface temperature
1774 Dim DryBelowSurfaceTemp As Double '(°C) Temperature at first node below surface
1775 Dim DryTRMOutputArray(,) As Double
1776 'DryTRMOutputArray format: SurfaceTemp(°C), Depth1(°C), Depth2(°C), ...
1777 'First row is previous timestep, second row is current timestep
1778
1779 'Dimension variables for output storage array
1780 Dim DryPaveStoreDate As Date 'Date of next save point
1781 Dim DryPaveStoreFreq As Double 'Frequency of save points (sec)
1782 Dim DryPaveStoreIndex As Integer 'Index for storing variables to output storage array
1783 Dim DryPaveStoreArray(,) As String 'Array for storing dry pavement temperature
simulation results

```

```

1784         'DryPaveStoreArray format: Time (date), Surface temp (°C), Depth 1 (°C), Depth 2
(°C),...
1785
1786         'Set storage frequency and initial storage date
1787         DryPaveStoreFreq = 3600
1788         DryPaveStoreDate = SimDate
1789
1790         'Set variable values
1791         'Thermal properties are from a table in Barber
1792         Pave_Depth = 0.1 'Asphalt thickness for 50-100 spaces on moderate class subgrade with
untreated aggregate base
1793         Pave_Density = 2250
1794         Pave_ThermalK = 1.21
1795         Pave_Cp = 921
1796         Pave_ThermalAlpha = Pave_ThermalK / (Pave_Density * Pave_Cp)
1797         Water_Density = 1000
1798         Gravel_Depth = 0.5 'Gravel thickness for 50-100 spaces on moderate class subgrade is
0.15m
1799         'Gravel Depth set artificially high so that constant temperature condition can be
achieved
1800         Gravel_Density = 1760
1801         Gravel_ThermalK = 1.3
1802         Gravel_Cp = 837
1803         Gravel_ThermalAlpha = Gravel_ThermalK / (Gravel_Density * Gravel_Cp)
1804
1805         DryTRMGridSpacing = 0.01
1806         'NOTE: Setting the grid spacing too small causes stability issues
1807
1808
1809         'Redimension output array
1810         'NOTE: May want to change the way this is calculated eventually to be more precise
1811         ReDim DryTRMOutputArray(1, ((Pave_Depth + Gravel_Depth) / DryTRMGridSpacing))
1812         'Redimension output storage array
1813         ReDim DryPaveStoreArray(DateDiff(DateInterval.Second, SimDate, StormDate) /
DryPaveStoreFreq, ((Pave_Depth + Gravel_Depth) / DryTRMGridSpacing) + 1)

```

```

1814
1815
1816     'Manually perform calculations for first timestep
1817     DrySurfaceTemp = 20 'Set initial pavement surface temperature
1818     DryTRMOutputArray(0, 0) = DrySurfaceTemp 'Store initial pavement surface temperature
1819
1820     'Set initial temperature profile in pavement
1821     For DryTRMDepthIndex = 1 To ((Pave_Depth + Gravel_Depth) / DryTRMGridSpacing - 1)
1822         'Basic linear initial temperature profile
1823         'Decreases temperature at every depth step
1824         DryTRMOutputArray(0, DryTRMDepthIndex) = DryTRMOutputArray(0, DryTRMDepthIndex -
1) - 0
1825     Next DryTRMDepthIndex
1826
1827     'Set temperature of bottom depth
1828     DryTRMOutputArray(0, (Pave_Depth + Gravel_Depth) / DryTRMGridSpacing) = 15
1829
1830     'Manually advance time for first step
1831     SimDate = SimDate.AddSeconds(DryTRMTimestep)
1832
1833
1834     '**_**_**_**_**_**
1835     'Start of loop for calculating solar radiation, air temperature, and pavement
temperature
1836     While SimDate < StormDate
1837
1838         'Load current date and time variables
1839         'Pull current month from SimDate
1840         SimMonth = DatePart(DateInterval.Month, SimDate)
1841         'Pull current hour from SimDate
1842         SimHour = DatePart(DateInterval.Hour, SimDate) + DatePart(DateInterval.Minute,
SimDate) / 60
1843         'Pull current day of year from SimDate
1844         SimDay = DatePart(DateInterval.DayOfYear, SimDate)
1845

```

```

1846         'Calculate Sunrise, Sunset, and Solar Noon
1847         'Based on algorithms presented in Astronomical Algorithms by Jean Meeus, 1998
1848
1849         'Calculate fractional year
1850         FractionalYear = (2 * PI) / 365 * (SimDay - 1 + (SimHour - 12) / 24)
1851
1852         'Calculate equation of time
1853         EquationOfTime = 229.18 * (0.000075 + 0.001868 * Cos(FractionalYear) - 0.032077 *
Sin(FractionalYear) - 0.014615 * Cos(2 * FractionalYear) - 0.040849 * Sin(2 * FractionalYear))
1854
1855         'Calculate solar declination for use in sunrise, sunset, and solar noon
1856         SolarTimeDec = 0.006918 - 0.399912 * Cos(FractionalYear) + 0.070257 *
Sin(FractionalYear) - 0.006758 * Cos(2 * FractionalYear) + 0.000907 * Sin(2 * FractionalYear)
- 0.002697 * Cos(3 * FractionalYear) + 0.00148 * Sin(3 * FractionalYear)
1857
1858         'Calculate hour angle
1859         HourAngle = Acos(Cos(PI * 90.833 / 180) / (Cos(Latitude) * Cos(SolarTimeDec)) -
Tan(Latitude) * Tan(SolarTimeDec))
1860
1861         'Calculate sunrise and sunset time in hours
1862         SunriseTime = (720 + 4 * (LonDeg - (HourAngle * 180 / PI)) - EquationOfTime) / 60
+ Timezone
1863         SunsetTime = (720 + 4 * (LonDeg + (HourAngle * 180 / PI)) - EquationOfTime) / 60 +
Timezone
1864
1865         'Calculate solar noon in hours
1866         SolarNoon = (720 + 4 * LonDeg - EquationOfTime) / 60 + Timezone
1867
1868
1869         'Beginning of Solar Radiation Algorithm
1870         'Calculate solar declination
1871         SolarDec = Asin(0.39785 * Sin(4.869 + 0.0172 * SimDay + 0.03345 * Sin(6.2238 +
0.0172 * SimDay)))
1872
1873         'Calculate zenith angle

```

```

1874          ZenithAngle = Acos(Sin(Latitude) * Sin(SolarDec) + Cos(Latitude) * Cos(SolarDec) *
Cos(0.0833 * PI * (SimHour - SolarNoon)))
1875
1876          'Calculate optical mass number
1877          OptiMassNum = BaroPress / (101.3 * Cos(ZenithAngle))
1878
1879          'Calculate beam solar radiation
1880          BeamRadiation = SolarConstant * Transmittance ^ OptiMassNum
1881
1882          'Calculate diffuse solar radiation
1883          DiffRadiation = 0.3 * (1 - Transmittance ^ OptiMassNum) * SolarConstant *
Cos(ZenithAngle)
1884
1885          'Calculate total solar radiation
1886          'Sets solar radiation to zero if time is before sunrise or after sunset
1887          'Commented out because different methodologies may lead to a negative optimass
number and instability
1888          'If SimHour < SunsetTime Or SimHour > SunriseTime Then
1889          '    SolarRadiation = BeamRadiation + DiffRadiation
1890          'Else
1891          '    SolarRadiation = 0
1892          'End If
1893
1894          'Calculate total solar radiation
1895          'Sets solar radiation to zero if OptiMassNum is negative
1896          If OptiMassNum > 0 Then
1897              SolarRadiation = BeamRadiation + DiffRadiation
1898          Else
1899              SolarRadiation = 0
1900          End If
1901
1902          '---
1903          'Calculate air temperature for current time based on daily maximum and minimum
values
1904          'Load Max and Min AirTemp based on current month

```

```

1905     MinAirTemp = MonthAirTempArray(SimMonth, 0)
1906     MaxAirTemp = MonthAirTempArray(SimMonth, 1)
1907
1908     'Calculate AirTempTbar based on current time
1909     If SimHour < MinAirTempTime Then
1910         AirTempTbar = (SimHour + 24 - MinAirTempTime) / 24
1911     Else
1912         AirTempTbar = (SimHour - MinAirTempTime) / 24
1913     End If
1914
1915     'Calculate current air temperature in simulation
1916     If SimHour <= 17 Then
1917         AirTemp = 0.5 * (MaxAirTemp + MinAirTemp) - 0.5 * (MaxAirTemp - MinAirTemp) *
Cos(2 * PI * AirTempTbar ^ (Log(2) / Log(24 / (MaxAirTempTime - MinAirTempTime)))) - 0.22
1918     Else
1919         AirTemp = 0.5 * (MaxAirTemp + MinAirTemp) - 0.5 * (MaxAirTemp - MinAirTemp) *
Cos(2 * PI * AirTempTbar ^ (Log(2) / Log(24 / (MaxAirTempTime - MinAirTempTime)))) - 1.25
1920     End If
1921
1922
1923     '-----
1924     'Calculate pavement temperature profile before storm using TRMPave algorithm
1925     'Dry simulations run for a certain period of time before storm to minimize the
effect of initial temperature profile guess
1926
1927     'Load weather variables
1928     DryWindSpeed = WindSpeedArray(SimMonth, 0)
1929
1930     'Loop for solving finite difference for depths below surface
1931     'Solve for asphalt layer
1932     For DryTRMDepthIndex = 1 To (Pave_Depth / DryTRMGridSpacing)
1933         DryTRMOutputArray(1, DryTRMDepthIndex) = DryTRMOutputArray(0,
DryTRMDepthIndex) * (1 - (2 * Pave_ThermalAlpha * DryTRMTimestep) / DryTRMGridSpacing ^ 2) +
(Pave_ThermalAlpha * DryTRMTimestep / DryTRMGridSpacing ^ 2) * (DryTRMOutputArray(0,
DryTRMDepthIndex - 1) + DryTRMOutputArray(0, DryTRMDepthIndex + 1))

```

```

1934         Next DryTRMDepthIndex
1935         'Solve for gravel layer
1936         For DryTRMDepthIndex = (Pave_Depth / DryTRMGridSpacing + 1) To ((Pave_Depth +
Gravel_Depth) / DryTRMGridSpacing - 1)
1937             DryTRMOutputArray(1, DryTRMDepthIndex) = DryTRMOutputArray(0,
DryTRMDepthIndex) * (1 - (2 * Gravel_ThermalAlpha * DryTRMTimestep) / DryTRMGridSpacing ^ 2) +
(Gravel_ThermalAlpha * DryTRMTimestep / DryTRMGridSpacing ^ 2) * (DryTRMOutputArray(0,
DryTRMDepthIndex - 1) + DryTRMOutputArray(0, DryTRMDepthIndex + 1))
1938         Next DryTRMDepthIndex
1939
1940         'Van Buren assumes that gravel depth is large enough to cause a constant
temperature at the bottom depth
1941         'The total depth Van Buren uses is 0.6 m
1942
1943         'Specify constant temperature at bottom depth
1944         DryTRMOutputArray(1, (Pave_Depth + Gravel_Depth) / DryTRMGridSpacing) =
DryTRMOutputArray(0, (Pave_Depth + Gravel_Depth) / DryTRMGridSpacing)
1945
1946         'Calculate convective heat transfer coefficient
1947         'Calculated using equation from Van Buren, 2000
1948         'Equation is an empirical relationship developed by Van Buren using his test plots
1949         DryPaveConvCoeff = 5.68 * (1.3 + 3.35 * DryWindSpeed ^ 0.75)
1950
1951         'Load below surface temperature from output array
1952         'Loads value from current timestep
1953         DryBelowSurfaceTemp = DryTRMOutputArray(1, 1)
1954
1955         'Calculate surface temperature
1956         'Calculated using equation 11 from Van Buren, 2000
1957         'Van Buren's algebraic manipulation (equation 12) is incorrect, so I'm using my
own manipulation of his original equation
1958         'Surface temperature calculation works with values from current timestep
1959         DrySurfaceTemp = (Pave_ThermalAlpha * DryBelowSurfaceTemp + (SolarRadiation +
DryPaveConvCoeff * AirTemp) * DryTRMGridSpacing) / (Pave_ThermalAlpha + DryPaveConvCoeff *
DryTRMGridSpacing)

```

```

1960      DryTRMOutputArray(1, 0) = DrySurfaceTemp 'Store surface temperature into output
array
1961
1962
1963      'Store results to storage array if storage interval is met
1964      If SimDate >= DryPaveStoreDate Then
1965
1966          'Copy current time to time output array
1967          DryPaveStoreArray(DryPaveStoreIndex, 0) = SimDate.ToString()
1968
1969          For DryTRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / DryTRMGridSpacing)
1970              'Store pavement temperatures
1971              DryPaveStoreArray(DryPaveStoreIndex, DryTRMDepthIndex + 1) =
DryTRMOutputArray(1, DryTRMDepthIndex).ToString()
1972          Next DryTRMDepthIndex
1973
1974          'Advances time for next storage interval
1975          DryPaveStoreDate = SimDate.AddSeconds(DryPaveStoreFreq)
1976
1977          'Advance WetSoilStoreIndex
1978          DryPaveStoreIndex = DryPaveStoreIndex + 1
1979      End If
1980
1981      'Copy temperature results from current timestep to previous row for next timestep
1982      For DryTRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / DryTRMGridSpacing)
1983          DryTRMOutputArray(0, DryTRMDepthIndex) = DryTRMOutputArray(1,
DryTRMDepthIndex)
1984      Next DryTRMDepthIndex
1985
1986      'Advance Time
1987      SimDate = SimDate.AddSeconds(DryTRMTimestep)
1988
1989      End While
1990
1991      'DEBUG MATERIAL

```

```

1992         'Stops execution time stopwatch and prints execution time in Debug line
1993         stopWatchDryPave.Stop()
1994         System.Diagnostics.Debug.WriteLine("Dry Pave RunTime: " &
stopWatchDryPave.Elapsed.Seconds & "." & stopWatchDryPave.Elapsed.Milliseconds)
1995         'END DEBUG MATERIAL
1996
1997
1998
1999         'Write dry pavement results to an output file
2000         Dim theWriterDryPave As System.IO.StreamWriter
2001         theWriterDryPave = My.Computer.FileSystem.OpenTextFileWriter(TB_DryPaveOutFile.Text,
False)
2002         'Write summary data
2003         theWriterDryPave.WriteLine("Dry Pavement Simulation")
2004         theWriterDryPave.WriteLine("")
2005         theWriterDryPave.WriteLine("Model Outputs:")
2006         theWriterDryPave.WriteLine("")
2007
2008         'Write array data
2009         Dim OutRowDryPave As Integer
2010         Dim OutColDryPave As Integer
2011         'Write depth to first row of output table
2012         theWriterDryPave.Write(",")
2013         For DryTRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / DryTRMGridSpacing)
2014             theWriterDryPave.Write((DryTRMDepthIndex * DryTRMGridSpacing).ToString() & ",")
2015         Next DryTRMDepthIndex
2016         theWriterDryPave.WriteLine()
2017
2018         For OutRowDryPave = 0 To DryPaveStoreIndex - 1
2019             'Write current time for each row
2020             theWriterDryPave.Write(DryPaveStoreArray(OutRowDryPave, 0).ToString() & ",")
2021             'Writes temperature at all depths to output table
2022             For OutColDryPave = 1 To ((Pave_Depth + Gravel_Depth) / DryTRMGridSpacing) + 1
2023                 theWriterDryPave.Write(DryPaveStoreArray(OutRowDryPave,
OutColDryPave).ToString() & ",")

```

```

2024         Next OutColDryPave
2025         theWriterDryPave.WriteLine()
2026     Next OutRowDryPave
2027     theWriterDryPave.Close()
2028
2029
2030
2031     '**-----**
2032     'Calculate pavement and runoff temperatures DURING STORM using TRMPave algorithm
2033
2034     'DEBUG LINES
2035     'Starts stopwatch to observe execution time for dry pavement temperature
2036     Dim stopWatchWetTRM As New Stopwatch()
2037     stopWatchWetTRM.Start()
2038     'END DEBUG LINES
2039
2040     'Update status label
2041     Label_Status.Text = "Status: Calculating runoff temperatures"
2042     Label_Status.Update()
2043
2044     'Dimension static variables
2045     Dim AtmPress As Double '(kPa) Atmospheric pressure; Currently assumed to be constant
for sea level elevation, but could incorporate effects of elevation and temperature changes
2046     Dim TRMTimestep As Double '(seconds) Timestep for simulation
2047     Dim TRMGridSpacing As Double '(m) Depth increment for finite difference calculations
2048     Dim TRMDepthIndex As Integer 'Location in pavement at current step
2049
2050     'NOTE: Weather data during the storm could be loaded from an input array
2051     'Dimension weather variables loaded from an input array
2052     'Dim TRMWeatherArray(,) As Double
2053     'TRMWeatherArray format: Time(sec), AirTemperature(°C), Radiation(W/m²),
WindSpeed(m/s), RelHumidity(fraction)
2054     'Dim WeatherIndex As Integer = -1 'Index for loading weather parameters
2055     Dim TRMRainRate As Double '(mm/hr) Rainfall intensity
2056     Dim Radiation As Double '(W/m²) Incoming radiation

```

```

2057 Dim WindSpeed As Double '(m/s) Wind speed over pavement surface
2058 Dim RelHumidity As Double '(fraction) Relative humidity of the air
2059 Dim AirTemperature As Double '(°C) Air temperature
2060 Dim AirTemperatureK As Double '(K) Air temperature in Kelvin
2061 Dim RainTemperature As Double '(°C) Temperature of falling rainfall
2062
2063 'Dimension intermediate variables
2064 Dim EvapRate As Double '(m/s) Rate of evaporation from pavement surface
2065 Dim SatVapPress As Double '(kPa) Saturation vapor pressure of air
2066 Dim qv As Double '(W/m²) Rate of energy used for latent heat of vaporization
2067 Dim Hv As Double '(J/kg) Latent heat of vaporization
2068 Dim BowenRatio As Double '(dimensionless) Used to calculate qsens from qv
2069 Dim qsens As Double '(W/m²) Sensible heat, rate of energy conducted and convected
between water surface and atmosphere
2070 Dim qevap As Double '(W/m²) Sum of sensible and latent heat transfer associated with
evaporation
2071 Dim PaveConvCoeff As Double '(W/m² °C) Convective heat transfer coefficient
2072
2073 'Dimension result variables
2074 Dim RunoffTemp As Double '(°C) Runoff temperature
2075 Dim SurfaceTemp As Double '(°C) Asphalt surface temperature
2076 Dim BelowSurfaceTemp As Double '(°C) Temperature at first node below surface
2077 Dim TRMOutputArray(,) As Double
2078 'TRMOutputArray format: RunoffTemp(°C), SurfaceTemp(°C), Depth1(°C), Depth2(°C), ...
2079 'Row 0 contains results from previous timestep, row 1 contains results from current
timestep
2080 Dim TRMRunoffArray(,) As Double 'Array for storing current and previous runoff
temperature
2081 ReDim TRMRunoffArray(1, 0) 'Row 0 is previous temperature, row 1 is current
temperature
2082
2083 'Dimension variables for output storage array
2084 Dim TRMStoreDate As Date 'Date of next save point
2085 Dim TRMStoreFreq As Double 'Frequency of save point (sec)
2086 Dim TRMStoreIndex As Integer 'Index for storing variables to output storage array

```

```

2087         'Dim TRMStoreArray(,) As String 'Array for storing runoff and pavement temperature
simulation results
2088         'TRMStoreArray format: Time (date), RunoffTemp(°C), SurfaceTemp(°C), Depth1(°C),
Depth2(°C), ...
2089
2090         'Set storage frequency and initial storage date
2091         TRMStoreFreq = 15 'seconds
2092         TRMStoreDate = RunoffStartDate
2093         'Set simdate to beginning of runoff
2094         SimDate = RunoffStartDate
2095
2096         'Set variable values
2097         'Most variables are set above for dry simulations
2098         AtmPress = 100 'May want to account for elevation or weather
2099
2100         'Probably want to keep these the same as dry simulations
2101         TRMTimestep = 15
2102         TRMGridSpacing = 0.01
2103         'Setting the grid spacing at 1mm causes major stability issues
2104
2105         'Load values from GUI
2106         AirTemperature = TB_AirTemperature.Text
2107         WindSpeed = TB_WindSpeed.Text
2108         Radiation = TB_Radiation.Text
2109         RelHumidity = TB_RelHumidity.Text
2110
2111         'Redimension output array
2112         'May want to change the way this is calculated eventually to be more precise
2113         ReDim TRMOutputArray(1, ((Pave_Depth + Gravel_Depth) / TRMGridSpacing))
2114         'Allows an extra column to store runoff temperature
2115         'Redimension output storage array
2116         ReDim TRMStoreArray(Round(RunoffDuration / TRMStoreFreq, 0), ((Pave_Depth +
Gravel_Depth) / TRMGridSpacing) + 2)
2117         'Calculate TRMStoreArrayLength for use in plotting array contents
2118         TRMStoreArrayLength = Round(RunoffDuration / TRMStoreFreq, 0)

```

```

2119
2120     'Set rainfall temperature during the storm
2121     'NOTE: Monitoring data shows that rainfall temperature can be reasonably approximated
by air temperature
2122     'NOTE: Both air and rainfall temperatures decrease during a storm event, which is not
currently accounted for
2123     RainTemperature = AirTemperature
2124
2125     'Manually perform calculations for first timestep
2126     'Load pavement temperatures from dry simulations into TRM calculations
2127     For TRMDepthIndex = 0 To (Pave_Depth + Gravel_Depth) / TRMGridSpacing
2128         TRMOutputArray(0, TRMDepthIndex) = DryTRMOutputArray(1, TRMDepthIndex)
2129     Next TRMDepthIndex
2130     SurfaceTemp = TRMOutputArray(0, 0) 'Load initial pavement surface temperature
2131     RunoffTemp = (RainTemperature + SurfaceTemp) / 2 'Calculate initial runoff temperature
2132     TRMRunoffArray(0, 0) = RunoffTemp 'Store initial runoff temperature
2133
2134
2135     'Load initial rainfall rate
2136     'Advances through times before inflow begins
2137     Rain_R = 0
2138     While SimDate < InflowArray(Rain_R, 0)
2139         Rain_R = Rain_R + 1
2140     End While
2141     TRMRainRate = InflowArray(Rain_R, 2) * 60 * 60 * 1000 'Load rainfall rate and convert
from m/s to (mm/hr)
2142     NexRainTime = InflowArray(Rain_R + 1, 0) 'Load time of next change in rainfall rate
2143
2144
2145     'Copy initial results to output array
2146     'Copy current time to time output array
2147     TRMStoreArray(TRMStoreIndex, 0) = SimDate.ToString()
2148     'Copy runoff temperature to output storage array
2149     TRMStoreArray(TRMStoreIndex, 1) = TRMRunoffArray(0, 0).ToString()
2150     For TRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / TRMGridSpacing)

```

```

2151         'Store pavement temperatures
2152         TRMStoreArray(TRMStoreIndex, TRMDepthIndex + 2) = TRMOutputArray(0,
TRMDepthIndex).ToString()
2153     Next TRMDepthIndex
2154     'Advances time for next storage interval
2155     TRMStoreDate = SimDate.AddSeconds(TRMStoreFreq)
2156     'Advance TRMStoreIndex
2157     TRMStoreIndex = TRMStoreIndex + 1
2158
2159     'Advance SimDate by timestep
2160     SimDate = SimDate.AddSeconds(TRMTimestep)
2161
2162
2163     'BEGIN MAIN TRM SIMULATION LOOP BASED ON TIME
2164     While SimDate < StormStopDate
2165
2166         'Load rainfall intensity
2167         'If SimDate passes a change in rainfall intensity, update rainfall intensity
2168         If SimDate > NexRainTime Then
2169             Rain_R = Rain_R + 1 'Advance index
2170             TRMRainRate = InflowArray(Rain_R, 2) * 60 * 60 * 1000 'Load rainfall rate and
convert from m/s to (mm/hr)
2171             NexRainTime = InflowArray(Rain_R + 1, 0) 'Load time of next change in rainfall
rate
2172         End If
2173
2174
2175         'Calculate runoff temperature
2176         'Assumed to be average of rainfall temperature and surface temperature based on
Van Buren, 2000
2177         'Runoff temperature calculation works with values from previous timestep
2178         'NEEDS TO OCCUR BEFORE ALL OTHER CALCULATIONS FOR CURRENT TIMESTEP
2179         RunoffTemp = (RainTemperature + SurfaceTemp) / 2
2180         TRMRunoffArray(1, 0) = RunoffTemp 'Store runoff temperature to output array
2181

```

```

2182         'Load weather variables from input array
2183         'NOTE: If storm weather input array is included, weather parameters need to be
updated here for current time
2184
2185
2186         'Loop for solving finite difference for depths below surface
2187         'Solve for asphalt layer
2188         For TRMDepthIndex = 1 To (Pave_Depth / TRMGridSpacing)
2189             TRMOutputArray(1, TRMDepthIndex) = TRMOutputArray(0, TRMDepthIndex) * (1 - 2 *
(Pave_ThermalAlpha * TRMTimestep / TRMGridSpacing ^ 2)) + (Pave_ThermalAlpha * TRMTimestep /
TRMGridSpacing ^ 2) * (TRMOutputArray(0, TRMDepthIndex - 1) + TRMOutputArray(0, TRMDepthIndex
+ 1))
2190         Next TRMDepthIndex
2191         'Solve for gravel layer
2192         For TRMDepthIndex = (Pave_Depth / TRMGridSpacing + 1) To ((Pave_Depth +
Gravel_Depth) / TRMGridSpacing - 1)
2193             TRMOutputArray(1, TRMDepthIndex) = TRMOutputArray(0, TRMDepthIndex) * (1 - 2 *
(Gravel_ThermalAlpha * TRMTimestep / TRMGridSpacing ^ 2)) + (Gravel_ThermalAlpha * TRMTimestep
/ TRMGridSpacing ^ 2) * (TRMOutputArray(0, TRMDepthIndex - 1) + TRMOutputArray(0,
TRMDepthIndex + 1))
2194         Next TRMDepthIndex
2195
2196         'Van Buren assumes that gravel depth is large enough to cause a constant
temperature at the bottom depth
2197         'The total depth Van Buren uses is 0.6 m
2198
2199         'Specify constant temperature at bottom depth
2200         TRMOutputArray(1, (Pave_Depth + Gravel_Depth) / TRMGridSpacing) =
TRMOutputArray(0, ((Pave_Depth + Gravel_Depth) / TRMGridSpacing))
2201
2202         'NOTE: If using variable rainfall temperature, need to update rainfall temperature
here
2203
2204         'Convert air temperature to Kelvin for saturated vapor pressure calculations
2205         AirTemperatureK = AirTemperature + 273.15

```

```

2206
2207     'Saturated vapor pressure calculated using Goff-Gratch equation
2208     'Goff-Gratch is generally considered to be the most reliable estimation of
saturated vapor pressure
2209     'Unit of vapor pressure used here is hectoPascals (100 pascals)
2210     SatVapPress = 10 ^ (-7.90298 * (373.15 / AirTemperatureK - 1) + 5.02808 *
Log10(373.15 / AirTemperatureK) - 0.00000013816 * (1011.344 * (1 - AirTemperatureK / 373.15) -
1) + 0.0081328 * (10 ^ (-3.49149 * (373.15 / AirTemperatureK - 1)) - 1) + Log10(1013.25))
2211     'Convert SatVapPress from hectoPascals to kPa
2212     SatVapPress = SatVapPress / 10
2213
2214     'Calculate evaporation rate
2215     'Calculated using equation 8 from Van Buren, 2000
2216     'Equation is called Meyer's equation and is presented in Chow, 1964
2217     EvapRate = 0.000000359 * (1 + 0.0447 * WindSpeed) * SatVapPress * (1 -
RelHumidity)
2218
2219     'Calculate convective heat transfer coefficient
2220     'Calculated using equation 16 from Van Buren, 2000
2221     'Equation is an empirical relationship developed by Van Buren using his test plots
2222     PaveConvCoeff = 3.46 * TRMRainRate ^ 1.1
2223
2224
2225     'Calculate Hv
2226     'Needs to be located after runoff temperature is calculated for current timestep
2227     'Calculated using equation 7 from Van Buren, 2000
2228     'Equation originally presented in Linsley, et al., 1958
2229     Hv = 862 * (597 - 0.56 * RunoffTemp)
2230
2231     'Calculate latent heat transfer
2232     'Calculated using equation 6 from Van Buren, 2000
2233     qv = Water_Density * Hv * EvapRate
2234
2235     'Calculate Bowen ratio
2236     'Calculated using equation 9 from Van Buren, 2000

```

```

2237         'Equation originally presented in Bowen, 1926
2238         BowenRatio = 0.00061 * AtmPress * (RunoffTemp - AirTemperature) / (SatVapPress *
(1 - RelHumidity))
2239
2240         'Calculate sensible heat transfer based on latent heat transfer and bowen ratio
2241         qsen = qv * BowenRatio
2242
2243         'Load below surface temperature from output array
2244         BelowSurfaceTemp = TRMOutputArray(1, 1)
2245
2246         'Calculate qevap
2247         qevap = qsen + qv
2248
2249         'Calculate surface temperature
2250         'Calculated using equation 11 from Van Buren, 2000
2251         'Van Buren's algebraic manipulation (equation 12) is incorrect, so I'm using my
own manipulation of his original equation
2252         'Surface temperature calculation works with values from current timestep
2253         SurfaceTemp = (Pave_ThermalK * BelowSurfaceTemp + (Radiation - qevap +
PaveConvCoeff * RunoffTemp) * TRMGridSpacing) / (Pave_ThermalK + PaveConvCoeff *
TRMGridSpacing)
2254         TRMOutputArray(1, 0) = SurfaceTemp 'Store surface temperature into output array
2255
2256
2257         'Store results to storage array if storage interval is met
2258         If SimDate >= TRMStoreDate Then
2259
2260             'Copy current time to time output array
2261             TRMStoreArray(TRMStoreIndex, 0) = SimDate.ToString()
2262
2263             'Copy runoff temperature to output storage array
2264             TRMStoreArray(TRMStoreIndex, 1) = TRMRunoffArray(1, 0).ToString()
2265
2266             For TRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / TRMGridSpacing)
2267                 'Store pavement temperatures

```

```

2268             TRMStoreArray(TRMStoreIndex, TRMDepthIndex + 2) = TRMOutputArray(1,
TRMDepthIndex).ToString()
2269         Next TRMDepthIndex
2270
2271         'Advances time for next storage interval
2272         TRMStoreDate = SimDate.AddSeconds(TRMStoreFreq)
2273
2274         'Advance TRMStoreIndex
2275         TRMStoreIndex = TRMStoreIndex + 1
2276     End If
2277
2278
2279     'Copy temperature results from current timestep to previous row for next timestep
2280     For TRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / TRMGridSpacing)
2281         TRMOutputArray(0, TRMDepthIndex) = TRMOutputArray(1, TRMDepthIndex)
2282     Next TRMDepthIndex
2283
2284
2285     'Advance SimDate by timestep
2286     SimDate = SimDate.AddSeconds(TRMTimestep)
2287
2288     End While
2289
2290     'Copy final runoff temperature to RunoffStopDat as last row of array
2291     'Copy current time to time output array
2292     TRMStoreArray(TRMStoreArrayLength, 0) = StormStopDate.ToString()
2293
2294     'Copy runoff temperature to output storage array
2295     TRMStoreArray(TRMStoreArrayLength, 1) = TRMRunoffArray(1, 0).ToString()
2296
2297     For TRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / TRMGridSpacing)
2298         'Store pavement temperatures
2299         TRMStoreArray(TRMStoreArrayLength, TRMDepthIndex + 2) = TRMOutputArray(1,
TRMDepthIndex).ToString()
2300     Next TRMDepthIndex

```

```

2301
2302
2303     'DEBUG MATERIAL
2304     'Stops execution time stopwatch and prints execution time in Debug line
2305     stopWatchWetTRM.Stop()
2306     System.Diagnostics.Debug.WriteLine("Wet TRM RunTime: " &
stopWatchWetTRM.Elapsed.Seconds & "." & stopWatchWetTRM.Elapsed.Milliseconds)
2307     'END DEBUG MATERIAL
2308
2309
2310     'Write wet pavement results to an output file
2311     Dim theWriterTRMPave As System.IO.StreamWriter
2312     theWriterTRMPave = My.Computer.FileSystem.OpenTextFileWriter(TB_WetPaveOutFile.Text,
False)
2313     'Write summary data
2314     theWriterTRMPave.WriteLine("Dry Pavement Simulation")
2315     theWriterTRMPave.WriteLine("")
2316     theWriterTRMPave.WriteLine("Model Outputs:")
2317     theWriterTRMPave.WriteLine("")
2318
2319     'Write array data
2320     Dim OutRowTRMPave As Integer
2321     Dim OutColTRMPave As Integer
2322     'Write depth to first row of output table
2323     theWriterTRMPave.Write(",")
2324     theWriterTRMPave.Write("Runoff,")
2325     For TRMDepthIndex = 0 To ((Pave_Depth + Gravel_Depth) / TRMGridSpacing)
2326         theWriterTRMPave.Write((TRMDepthIndex * TRMGridSpacing).ToString() & ",")
2327     Next TRMDepthIndex
2328     theWriterTRMPave.WriteLine()
2329
2330     For OutRowTRMPave = 0 To TRMStoreIndex - 1
2331         'Write current time for each row
2332         theWriterTRMPave.Write(TRMStoreArray(OutRowTRMPave, 0).ToString() & ",")
2333         'Write runoff temperature for each row

```

```

2334         theWriterTRMPave.Write (TRMStoreArray(OutRowTRMPave, 1).ToString() & ",")
2335         'Writes temperature at all depths to output table
2336         For OutColTRMPave = 2 To ((Pave_Depth + Gravel_Depth) / TRMGridSpacing) + 2
2337             theWriterTRMPave.Write (TRMStoreArray(OutRowTRMPave, OutColTRMPave).ToString()
& ",")
2338         Next OutColTRMPave
2339         theWriterTRMPave.WriteLine()
2340     Next OutRowTRMPave
2341     theWriterTRMPave.Close()
2342
2343     '----**----
2344     'Load values from runoff temperature input file
2345
2346     'Initialize array for storing rain data read from csv file
2347     Dim RunoffTempDataArray(,) As String
2348     'RunoffTempDataArray format: Date and Time, Runoff Temperature (C)
2349
2350     'Load filename from text box
2351     RunoffTempFileName = TB_RunoffTempFileName.Text
2352     'Error check that the input file exists
2353     If File.Exists(RunoffTempFileName) Then
2354         'If file exists, load values into the runoff temperature array
2355
2356         'Update runoff temperature source label
2357         Label_RunoffTempSource.Text = "Runoff Temperature Source: Input File"
2358         Label_RunoffTempSource.Update()
2359
2360         ' Indexes for storing values to correct position in array
2361         Dim RunoffTempFile_R As Integer = -1
2362         Dim RunoffTempFile_C As Integer = -1
2363         Dim RunoffTemp_NumOfRows As Integer = -1
2364
2365         'Code to load runoff temperature .csv file into array
2366         Dim RunoffTempCSVReader As StreamReader = File.OpenText(RunoffTempFileName)
2367         Dim RunoffTempcsvRow As String

```

```

2368         Dim RunoffTemp_memFile As System.Collections.ArrayList
2369         RunoffTemp_memFile = New System.Collections.ArrayList
2370         RunoffTempcsvRow = RunoffTempCSVReader.ReadLine()
2371         Do While RunoffTempcsvRow <> Nothing 'Perform the loop as long as there is
something stored in the row
2372             RunoffTemp_NumOfRows = RunoffTemp_NumOfRows + 1 'Keep track of length of file
2373             ' load each record to the array of records
2374             RunoffTemp_memFile.Add(RunoffTempcsvRow.Split(",")) 'Sets the delimiter as
comma
2375             RunoffTempcsvRow = RunoffTempCSVReader.ReadLine() 'Sets each row as a record
2376         Loop
2377         'Redimension the array based on number of rows in csv file
2378         ReDim RunoffTempDataArray(RunoffTemp_NumOfRows, 1)
2379         'Read in each row
2380         For Each RunoffTemp_record As Object In RunoffTemp_memFile
2381             RunoffTempFile_R = RunoffTempFile_R + 1
2382
2383             'Read in each column and store it into the array
2384             For Each RunoffTemp_field As Object In RunoffTemp_record
2385                 RunoffTempFile_C = RunoffTempFile_C + 1
2386                 If RunoffTempFile_C = 2 Then
2387                     RunoffTempFile_C = 0
2388                 End If
2389                 RunoffTempDataArray(RunoffTempFile_R, RunoffTempFile_C) = RunoffTemp_field
2390             Next
2391         Next
2392
2393         'Converts date and hour string data to Date format
2394         For RunoffTempFile_R = 0 To RunoffTemp_NumOfRows
2395             RunoffTempDataArray(RunoffTempFile_R, 0) =
CDate(RunoffTempDataArray(RunoffTempFile_R, 0))
2396         Next RunoffTempFile_R
2397         'Converts runoff temperature String data to Double
2398         For RunoffTempFile_R = 0 To RunoffTemp_NumOfRows
2399             RunoffTempDataArray(RunoffTempFile_R, 1) =

```

```

2400 Double.Parse(RunoffTempDataArray(RunoffTempFile_R, 1))
2401     Next RunoffTempFile_R
2402     'Copy runoff temperatures into array used for later calculations
2403
2404     Dim RunoffTempCopyIndex As Integer 'Index for copying runoff temperatures
2405     Dim CurRunoffTempTime As Date 'Date of current runoff temperature measurement
2406     Dim NexRunoffTempTime As Date 'Date of next runoff temperature measurement
2407     Dim CurRODate As Date 'Current date within the TRM Array
2408     RunoffTempFile_R = 0
2409
2410     While RunoffTempCopyIndex < TRMStoreArrayLength
2411         'Lookup times stored in runoff temp file
2412         CurRunoffTempTime = RunoffTempDataArray(RunoffTempFile_R, 0)
2413         NexRunoffTempTime = RunoffTempDataArray(RunoffTempFile_R + 1, 0)
2414
2415         'Lookup current date in TRMStoreArray
2416         CurRODate = TRMStoreArray(RunoffTempCopyIndex, 0)
2417
2418         'While runoff temperature does not change
2419         While CurRODate < NexRunoffTempTime
2420             'Copy runoff temperature
2421             TRMStoreArray(RunoffTempCopyIndex, 1) =
2422 RunoffTempDataArray(RunoffTempFile_R, 1)
2423
2424             RunoffTempCopyIndex = RunoffTempCopyIndex + 1 'Advance RunoffTempCopyIndex
2425             'Lookup current date in TRMStoreArray
2426             CurRODate = TRMStoreArray(RunoffTempCopyIndex, 0)
2427         End While
2428         'When Date passes next runoff temperature measurement, advance index for
2429 loading runoff temperatures
2430         RunoffTempFile_R = RunoffTempFile_R + 1
2431     End While

```

```

2432 Else
2433     'If runoff temperature file does not exist, use calculated values
2434     'Update runoff temperature source label
2435     Label_RunoffTempSource.Text = "Runoff Temperature Source: Model Calculations"
2436     Label_RunoffTempSource.Update()
2437 End If
2438
2439
2440     '**---**---**
2441     'Calculate initial soil temperature profile based on time of year
2442     'Follows procedure described in Elias, et al. 2004. Analytical Soil-Temperature Model:
Correction for Temporal Variation of Daily Amplitude. Soil Science Society of America Journal.
68: 784-788.
2443     'DEBUG LINES
2444     'Starts stopwatch to observe execution time
2445     Dim stopWatchDrySoil As New Stopwatch()
2446     stopWatchDrySoil.Start()
2447     'END DEBUG LINES
2448
2449     'Update status label
2450     Label_Status.Text = "Status: Calculating initial soil temperature profile"
2451     Label_Status.Update()
2452
2453     'Determines how many seconds have elapsed from the beginning of the year to date and
time input by the user
2454     Dim InputDate As Date
2455     Dim InputYear As Integer
2456     Dim YearBeginDate As Date
2457     Dim DrySoilTime As Double
2458
2459     'Set InputDate to StormDate
2460     InputDate = StormDate
2461
2462     'Determines year of input date and converts to string
2463     'Used to calculate elapsed seconds during current year

```

```

2464     InputYear = DatePart(DateInterval.Year, InputDate)
2465     InputYear = Str(InputYear)
2466
2467     'Determines date at the beginning of the year
2468     YearBeginDate = "1, 1, " & InputYear
2469     YearBeginDate = DateTime.Parse(YearBeginDate)
2470
2471     'Calculates number of seconds between the 2 dates
2472     DrySoilTime = DateDiff(DateInterval.Second, YearBeginDate, InputDate)
2473     'Subtract 12 hours from time to account for a phase shift that causes max temperature
at midnight if no adjustment
2474     DrySoilTime = DrySoilTime - 43200
2475
2476
2477     'Calculate soil temperature profile for selected time
2478
2479     'Set soil depth
2480     Dim DrySoilTotalDepth As Double
2481     DrySoilTotalDepth = 2 'Set at 2 m to make zero flux boundary condition applicable for
wetsoil calculations
2482
2483
2484     ''Grid spacing used for Two-Phase calculations
2485     'Dim GridSpacing As Double 'Grid spacing along z axis (m)
2486
2487     'Set Two-Phase Grid Spacing
2488     GridSpacing = 0.05 '(m)
2489
2490     Dim DrySoilDepthStep As Double
2491     'This depth step must be the same as the one used in Two-Phase calculations below
2492     DrySoilDepthStep = GridSpacing
2493
2494     Dim DrySoilDepth As Double 'Depth below surface (m)
2495     Dim DrySoilArray(,) As Double 'Array for storing soil temperature at each depth
2496     Dim DrySoilTemp As Double 'Temperature of soil at given depth and time (°C)

```

```

2497     Dim DrySoilRowIndex As Integer = 0
2498
2499     ReDim DrySoilArray(DrySoilTotalDepth / DrySoilDepthStep, 1)
2500
2501     Dim Tay As Double 'Annual average surface temperature (°C)
2502     Dim Ay As Double 'Annual amplitude (°C)
2503     Dim Dy As Double 'Annual damping depth (m)
2504     Dim wy As Double 'Annual radial frequency (rad/s)
2505     Dim Phiy As Double 'Annual phase constant (rad)
2506     Dim Ad As Double 'Daily amplitude (°C)
2507     Dim Dd As Double 'Daily damping depth (m)
2508     Dim wd As Double 'Daily radial frequency (rad/s)
2509     Dim Phid As Double 'Daily phase constant (rad)
2510     Dim B As Double 'Constant related to temporal variation of daily amplitude around its
average value of Ad (°C)
2511     Dim Dprime As Double 'Damping depth of temperature wave with w'
2512     Dim wprime As Double 'wd-wb
2513     Dim Beta As Double 'Phase constant related to change in amplitude (rad)
2514     Dim Dprimeprime As Double 'Damping depth for temperature wave with w''
2515     Dim wprimeprime As Double 'wd+wb
2516     Dim wb As Double 'Radial frequency related to change in daily amplitude (rad/s)
2517
2518     'NOTE: Need to calibrate these variables with those at monitoring sites
2519     'Set values for each parameter
2520     Tay = 20 'Annual average surface temperature (°C)
2521     Ay = 7 'Annual amplitude (°C)
2522     Dy = 2.36 'Annual damping depth (m)
2523     wy = 0.000000199 'Annual radial frequency (rad/s)
2524     Phiy = 4.5 'Annual phase constant (rad)
2525     Ad = 4 'Daily amplitude (°C)
2526     Dd = 0.12 'Daily damping depth (m)
2527     wd = 0.0000727 'Daily radial frequency (rad/s)
2528     Phid = 1.85 'Daily phase constant (rad)
2529     B = 0.95 'Constant related to temporal variation of daily amplitude around its average
value of Ad (°C)

```

```

2530     Dprime = 0.12 'Damping depth of temperature wave with w'
2531     wprime = 0.0000725 'wd-wb
2532     Beta = 1.73 'Phase constant related to change in amplitude (rad)
2533     Dprimeprime = 0.12 'Damping depth for temperature wave with w''
2534     wprimeprime = 0.0000729 'wd+wb
2535     wb = 0.000000199 'Radial frequency related to change in daily amplitude (rad/s)
2536
2537     'Because of round-off error, can't use <= to ensure that all depths are accounted for
2538     While DrySoilDepth < DrySoilTotalDepth + DrySoilDepthStep
2539
2540         DrySoilTemp = Tay + Ay * Exp(-DrySoilDepth / Dy) * Sin(wy * DrySoilTime -
DrySoilDepth / Dy + Phiy) + Ad * Exp(-DrySoilDepth / Dd) * Sin(wd * DrySoilTime - DrySoilDepth
/ Dd + Phid) + (B / 2) * Exp(-DrySoilDepth / Dprime) * Sin(wprime * DrySoilTime - DrySoilDepth
/ Dprime + Phid - Beta + PI / 2) - (B / 2) * Exp(-DrySoilDepth / Dprimeprime) *
Sin(wprimeprime * DrySoilTime - DrySoilDepth / Dprimeprime + Phid + Beta + PI / 2)
2541
2542         DrySoilArray(DrySoilRowIndex, 0) = DrySoilDepth
2543         DrySoilArray(DrySoilRowIndex, 1) = DrySoilTemp
2544
2545         DrySoilDepth = DrySoilDepth + DrySoilDepthStep
2546         DrySoilDepth = Round(DrySoilDepth, 3) 'Round to prevent precision error
2547         DrySoilRowIndex = DrySoilRowIndex + 1
2548
2549     End While
2550
2551     'DEBUG MATERIAL
2552     'Stops execution time stopwatch and prints execution time in Debug line
2553     stopWatchDrySoil.Stop()
2554     System.Diagnostics.Debug.WriteLine("Dry Soil RunTime: " &
stopWatchDrySoil.Elapsed.Seconds & "." & stopWatchDrySoil.Elapsed.Milliseconds)
2555     'END DEBUG MATERIAL
2556
2557
2558
2559     '**-----**

```

```

2560 'Perform calculations for 2-phase, 2-temperature model of soil and water temperature
2561 'DEBUG LINES
2562 'Starts stopwatch to observe execution time for ponded condition
2563 Dim stopWatchWetSoil As New Stopwatch()
2564 stopWatchWetSoil.Start()
2565 'END DEBUG LINES
2566
2567 'Update status label
2568 Label_Status.Text = "Status: Calculating infiltration temperature profiles"
2569 Label_Status.Update()
2570
2571 Dim WetSoilTime As Double 'Current time of simulation (s)
2572 Dim WetSoilTimeStep As Double 'Timestep for wet soil calculations (s)
2573 Dim WetSoilTimeLength As Double 'Length of time for wet soil simulations (s)
2574 Dim WetSoilTimeIndex As Integer 'Time index for wet soil simulations
2575 Dim WetSoilDepthIndex As Integer 'Depth index for wet soil simulations
2576 Dim WetSoilDepth As Double 'Depth of soil profile
2577
2578 'Initialize variables
2579 Dim WetSoilDate As Date 'Current date in wet soil simulations
2580 Dim NexTRMDate As Date 'Next date of change in runoff temperature
2581 Dim WetSoilCalcArray_F(,) As Double 'Array to store fluid temperature for calc
purposes
2582 Dim WetSoilCalcArray_S(,) As Double 'Array to store solid temperature for calc
purposes
2583 Dim WetSoilSurfArray(,) As Double 'Array to store solid and fluid surface temperatures
2584 Dim Density_S As Double 'Solid density
2585 Dim Cp_S As Double 'Solid specific heat
2586 Dim Porosity As Double 'Porosity of soil material
2587 Dim G As Double 'Combination term
2588 Dim Sigma As Double 'Ks/Kf combination term
2589 Dim ThermalK_S As Double 'Solid thermal conductivity
2590 Dim InterfaceArea As Double 'Interfacial surface area - asf
2591 Dim ConvCoeff As Double 'Convective coefficient - hsf
2592 Dim Density_F As Double 'Fluid density

```

```

2593     Dim Cp_F As Double 'Fluid specific heat
2594     Dim ThermalK_F As Double 'Fluid thermal conductivity
2595     Dim Vel_F As Double 'Fluid velocity
2596     Dim KDis As Double 'Thermal diffusivity
2597
2598     'Initialize variables for storing array data
2599     Dim WetSoilOutArray_F(,) As Double 'Array for storing fluid temperature calculation
results
2600     Dim WetSoilOutArray_S(,) As Double 'Array for storing soil temperature calculation
results
2601     Dim WetSoilStoreFreq As Double 'Time period for storing temperature (s)
2602     Dim WetSoilTimeStored As Double = 0 'Time when last set of array data was stored (s)
2603     Dim WetSoilStoreIndex As Integer = 0 'Index for storing output to array
2604     Dim WetSoilOutTimeArray(,) As Date 'Array for storing times for each row exported to
output array
2605
2606     'Initialize variables for specific nodes
2607     Dim T_S_dt As Double 'Temperature of solid at downstream node in time
2608     Dim T_S_ct As Double 'Temperature of solid at current node in time
2609     Dim T_F_dt As Double 'Temperature of fluid at downstream node in time
2610     Dim T_F_ct As Double 'Temperature of fluid at current node in time
2611     Dim T_S_ds As Double 'Temperature of solid at downstream node in space
2612     Dim T_S_cs As Double 'Temperature of solid at current node in space
2613     Dim T_S_us As Double 'Temperature of solid at upstream node in space
2614     Dim T_F_ds As Double 'Temperature of fluid at downstream node in space
2615     Dim T_F_cs As Double 'Temperature of fluid at current node in space
2616     Dim T_F_us As Double 'Temperature of fluid at upstream node in space
2617     Dim Tr As Double 'Combination term for QUICK scheme
2618     Dim Tl As Double 'Combination term for QUICK scheme
2619
2620     'Initialize components of combination terms
2621     Dim ThermalK_Stg As Double 'Thermal conductivity of stagnant soil-water system
2622     Dim ParticleDia As Double 'Soil particle diameter
2623     Dim Visc_F As Double 'Fluid viscosity (Mu)
2624     Dim Prandtl As Double 'Prandtl number

```

```

2625
2626 Dim SurfConvCoeff As Double 'Convective coefficient at surface (W/m^2 K)
2627 Dim Shading As Double 'Percent of bioretention surface shaded from solar radiation
2628
2629 'Set static input parameters
2630 Density_S = 1650 '(kg/m³)
2631 Density_F = 998 '(kg/m³)
2632 Cp_F = 4.183 '(kJ/kg K)
2633 ThermalK_F = 0.58 '(W/m K)
2634 Visc_F = 0.001002 '(N s/m²)
2635 Prandtl = 7 'Typical number for water
2636
2637
2638 'Load values from GUI
2639 Cp_S = TB_Cp_S.Text
2640 Porosity = TB_Porosity.Text
2641 ThermalK_S = TB_ThermalK_S.Text
2642 InterfaceArea = TB_InterfaceArea.Text
2643 ThermalK_Stg = TB_ThermalK_Stg.Text
2644 ParticleDia = TB_ParticleDia.Text
2645 Shading = TB_Shading.Text
2646
2647 'Set timestep for simulation and storage
2648 WetSoilTimeStep = 0.1 '(s)
2649 WetSoilStoreFreq = 10 '(s)
2650 'Grid spacing is set above with dry soil simulations
2651
2652 'Set time length of simulations
2653 WetSoilTimeLength = RunoffDuration
2654
2655 'Set depth of soil
2656 WetSoilDepth = 2
2657 'Set depth at 2 meters to make zero flux boundary condition appropriate
2658
2659 'Dimension calculation arrays

```

```

2660     'For WetSoilCalcArrays, row 0 is previous timestep, row 1 is current timestep
2661     ReDim WetSoilCalcArray_F(1, WetSoilDepth / GridSpacing)
2662     ReDim WetSoilCalcArray_S(1, WetSoilDepth / GridSpacing)
2663     'For WetSoilSurfArray, rows are time, column 0 is solid, column 1 is fluid
2664     ReDim WetSoilSurfArray(WetSoilTimeLength / WetSoilTimeStep, 1)
2665
2666     'Dimension WetSoilArray for output storage
2667     'Rows are times, columns are depths
2668     'First column is current time (s)
2669     ReDim WetSoilOutArray_F(WetSoilTimeLength / WetSoilStoreFreq, WetSoilDepth /
GridSpacing)
2670     ReDim WetSoilOutArray_S(WetSoilTimeLength / WetSoilStoreFreq, WetSoilDepth /
GridSpacing)
2671     ReDim WetSoilOutTimeArray(WetSoilTimeLength / WetSoilStoreFreq, 0)
2672
2673     'Calculate array length for use in plotting
2674     Dim WetSoilOutArrayLength As Integer
2675     WetSoilOutArrayLength = WetSoilTimeLength / WetSoilStoreFreq
2676
2677     'Calculate static combination terms
2678     Sigma = ThermalK_S / ThermalK_F
2679     G = ((ThermalK_Stg / ThermalK_F) - Porosity - (1 - Porosity) * Sigma) / (Sigma - 1) ^
2
2680
2681
2682     'DEBUG LINES
2683     'Starts stopwatch to observe execution time for setting wet soil initial conditions
2684     Dim stopWatchWetSoilIC As New Stopwatch()
2685     stopWatchWetSoilIC.Start()
2686     'END DEBUG LINES
2687
2688     'Set initial conditions
2689     WetSoilTime = 0
2690     WetSoilDate = RunoffStartDate
2691     SimDate = RunoffStartDate

```

```

2692         'WetSoilSurfArray(0, 0) = 30 'Set initial surface soil temperature
2693         WetSoilCalcArray_S(0, 0) = WetSoilSurfArray(0, 0)
2694         TRMStoreIndex = 0
2695         WetSoilSurfArray(0, 1) = TRMStoreArray(TRMStoreIndex, 1) 'Load initial surface fluid
temperature
2696         WetSoilCalcArray_F(0, 0) = WetSoilSurfArray(0, 1) 'Store initial surface fluid
temperature to calculation array
2697
2698
2699         'Set initial conditions for all depths based on dry soil simulations
2700         For WetSoilDepthIndex = 0 To (WetSoilDepth / GridSpacing)
2701             WetSoilCalcArray_F(0, WetSoilDepthIndex) = DrySoilArray(WetSoilDepthIndex, 1)
2702             WetSoilCalcArray_S(0, WetSoilDepthIndex) = DrySoilArray(WetSoilDepthIndex, 1)
2703         Next WetSoilDepthIndex
2704
2705
2706         'Write initial conditions to output array
2707         'Copy current time to time output array
2708         WetSoilOutTimeArray(WetSoilStoreIndex, 0) = SimDate
2709         For WetSoilDepthIndex = 0 To (WetSoilDepth / GridSpacing)
2710             'Store soil temperatures
2711             WetSoilOutArray_S(0, WetSoilDepthIndex) = WetSoilCalcArray_S(0, WetSoilDepthIndex)
2712             'Store fluid temperatures
2713             WetSoilOutArray_F(0, WetSoilDepthIndex) = WetSoilCalcArray_F(0, WetSoilDepthIndex)
2714         Next WetSoilDepthIndex
2715         'Store current time to TimeStored for use in next interval
2716         WetSoilTimeStored = WetSoilTime
2717         'Advance WetSoilStoreIndex
2718         WetSoilStoreIndex = 1
2719
2720         'Advance WetSoilDate
2721         WetSoilDate = WetSoilDate.AddSeconds(WetSoilTimeStep)
2722
2723         'Load date of next change in runoff temperature
2724         NexTRMDate = TRMStoreArray(TRMStoreIndex + 1, 0)

```

```

2725
2726
2727     'Set surface conditions for all times except initial time
2728     For WetSoilTimeIndex = 1 To (WetSoilTimeLength / WetSoilTimeStep)
2729
2730         'Load fluid temperatures
2731         If WetSoilDate > NexTRMDate Then
2732             TRMStoreIndex = TRMStoreIndex + 1 'Advance TRM store index
2733             NexTRMDate = TRMStoreArray(TRMStoreIndex + 1, 0) 'Load new date of next change
in runoff temperature
2734         End If
2735
2736         'Load runoff temperature
2737         WetSoilSurfArray(WetSoilTimeIndex, 1) = TRMStoreArray(TRMStoreIndex, 1)
2738
2739         'Advance WetSoilDate
2740         WetSoilDate = WetSoilDate.AddSeconds(WetSoilTimeStep)
2741
2742     Next WetSoilTimeIndex
2743
2744
2745     'DEBUG MATERIAL
2746     'Stops execution time stopwatch and prints execution time in Debug line
2747     stopWatchWetSoilIC.Stop()
2748     System.Diagnostics.Debug.WriteLine("Wet Soil IC RunTime: " &
stopWatchWetSoilIC.Elapsed.Seconds & "." & stopWatchWetSoilIC.Elapsed.Milliseconds)
2749     'END DEBUG MATERIAL
2750
2751
2752     'Manually advance time and timeindex variables for beginning of calculations
2753     WetSoilTime = WetSoilTimeStep
2754     WetSoilTimeIndex = 1
2755
2756     'Set SimDate for beginning of calculations
2757     SimDate = RunoffStartDate

```

```

2758
2759     'Set GAIndex to zero for loading fluid velocities
2760     GAIndex = 0
2761
2762     Dim NexGADate As Date 'Date of next change in fluid velocity from Green-Ampt
2763     NexGADate = GreenAmptArray(GAIndex + 1, 0)
2764
2765     'Load initial fluid velocity and convert from m/hr to (m/s)
2766     Vel_F = GreenAmptArray(GAIndex, 2) / 60 / 60
2767
2768
2769     'MAIN LOOP FOR WETSOIL CALCULATIONS
2770     While WetSoilTimeIndex <= (WetSoilTimeLength / WetSoilTimeStep)
2771
2772         'Advance SimDate
2773         SimDate = SimDate.AddSeconds(WetSoilTimeStep)
2774
2775         'Load fluid velocity
2776         If SimDate > NexGADate Then
2777             GAIndex = GAIndex + 1 'Advance GAIndex
2778
2779             'Workaround for problem where tries to call index past the end of the array
2780             If GAIndex + 1 < GAIndexLength Then
2781                 Vel_F = GreenAmptArray(GAIndex, 2) / 60 / 60 'Load fluid velocity and
2782                 NexGADate = GreenAmptArray(GAIndex + 1, 0) 'Load date of next change in
2783                 fluid velocity
2784             End If
2785         End If
2786
2787         'Calculate combination terms
2788         KDis = 0.5 * Density_F * Cp_F * Vel_F * ParticleDia
2789         ConvCoeff = (2 + 1.1 * Prandtl ^ (1 / 3) * (Density_F * Vel_F * ParticleDia /
2790         Visc_F) ^ 0.6) * ThermalK_F

```

```

2790      'Copy surface data to calculation array
2791      'WetSoilCalcArray_S(0, 0) = WetSoilSurfArray(WetSoilTimeIndex, 0)
2792      WetSoilCalcArray_F(0, 0) = WetSoilSurfArray(WetSoilTimeIndex, 1)
2793
2794      'Calculate terms for evaporation estimate
2795      'Convert air temperature to Kelvin for saturated vapor pressure calculations
2796      AirTemperatureK = AirTemperature + 273.15
2797
2798      'Saturated vapor pressure calculated using Goff-Gratch equation
2799      'Goff-Gratch is generally considered to be the most reliable estimation of
saturated vapor pressure
2800      'Unit of vapor pressure used here is hectoPascals (100 pascals)
2801      SatVapPress = 10 ^ (-7.90298 * (373.15 / AirTemperatureK - 1) + 5.02808 *
Log10(373.15 / AirTemperatureK) - 0.00000013816 * (1011.344 * (1 - AirTemperatureK / 373.15) -
1) + 0.0081328 * (10 ^ (-3.49149 * (373.15 / AirTemperatureK - 1)) - 1) + Log10(1013.25))
2802      'Convert SatVapPress from hectoPascals to kPa
2803      SatVapPress = SatVapPress / 10
2804
2805      'Calculate evaporation rate
2806      'Calculated using equation 8 from Van Buren, 2000
2807      'Equation is called Meyer's equation and is presented in Chow, 1964
2808      EvapRate = 0.000000359 * (1 + 0.0447 * WindSpeed) * SatVapPress * (1 -
RelHumidity)
2809
2810      'Calculate Hv
2811      'Calculated using equation 7 from Van Buren, 2000
2812      'Equation originally presented in Linsley, et al., 1958
2813      Hv = 862 * (597 - 0.56 * WetSoilCalcArray_F(0, 0))
2814
2815      'Calculate latent heat transfer
2816      'Calculated using equation 6 from Van Buren, 2000
2817      qv = Water_Density * Hv * EvapRate
2818
2819      'Calculate Bowen ratio
2820      'Calculated using equation 9 from Van Buren, 2000

```

```

2821         'Equation originally presented in Bowen, 1926
2822         BowenRatio = 0.00061 * AtmPress * (WetSoilCalcArray_F(0, 0) - AirTemperature) /
(SatVapPress * (1 - RelHumidity))
2823
2824         'Calculate sensible heat transfer based on latent heat transfer and bowen ratio
2825         qsen = qv * BowenRatio
2826
2827         'Calculate qevap
2828         qevap = qsen + qv
2829
2830         'Calculate soil surface temperature
2831         'Note: Trying a different approach for calculating surface temperatures
2832         'Convective coefficient set manually
2833         SurfConvCoeff = 100
2834         WetSoilCalcArray_S(0, 0) = (ThermalK_Stg * WetSoilCalcArray_S(0, 1) + ((1 -
Shading) * Radiation - qevap + SurfConvCoeff * WetSoilCalcArray_F(0, 0)) * GridSpacing) /
(ThermalK_Stg + SurfConvCoeff * GridSpacing)
2835
2836
2837         'Run loop for remaining depths
2838         For WetSoilDepthIndex = 1 To (WetSoilDepth / GridSpacing - 1)
2839
2840             'Check for stability errors and exit sub if unstable
2841             If T_S_dt > 75 Or T_S_dt < 0 Then
2842                 MessageBox.Show("Stability error encountered")
2843                 Exit Sub
2844                 Exit Sub
2845                 Exit Sub
2846             End If
2847
2848             'Load soil and water temperatures from array
2849             T_S_cs = WetSoilCalcArray_S(0, WetSoilDepthIndex)
2850             T_S_ct = T_S_cs
2851             T_S_us = WetSoilCalcArray_S(0, WetSoilDepthIndex - 1)
2852             T_S_ds = WetSoilCalcArray_S(0, WetSoilDepthIndex + 1)

```

```

2853         T_F_cs = WetSoilCalcArray_F(0, WetSoilDepthIndex)
2854         T_F_ct = T_F_cs
2855         T_F_us = WetSoilCalcArray_F(0, WetSoilDepthIndex - 1)
2856         T_F_ds = WetSoilCalcArray_F(0, WetSoilDepthIndex + 1)
2857
2858         'Solid
2859         T_S_dt = (Density_S * Cp_S * (1 - Porosity) * T_S_ct / WetSoilTimeStep + (1 -
Porosity + G * (Sigma - 1)) * ThermalK_S * ((T_S_us + T_S_ds - 2 * T_S_cs) / GridSpacing ^ 2)
- InterfaceArea * ConvCoeff * (T_S_cs - T_F_cs) + ThermalK_S * G * ((T_S_us + T_S_ds - 2 *
T_S_cs) / GridSpacing ^ 2 - (T_F_us + T_F_ds - 2 * T_F_cs) / GridSpacing ^ 2)) / (Density_S *
Cp_S * (1 - Porosity) / WetSoilTimeStep)
2860
2861         'Fluid
2862         Tr = (1 / 2) * (T_F_cs + T_F_ds) - (1 / 8) * (T_F_us + T_F_ds - 2 * T_F_cs)
2863         Tl = (1 / 2) * (T_F_us + T_F_cs) - (1 / 8) * (T_F_us + T_F_ds - 2 * T_F_cs)
2864         T_F_dt = (Density_F * Cp_F * Porosity * T_F_ct / WetSoilTimeStep - (Density_F
* Cp_F * Vel_F * (Tr - Tl)) / GridSpacing + (Porosity + G * (1 - Sigma)) * ThermalK_F *
(T_F_us + T_F_ds - 2 * T_F_cs) / GridSpacing ^ 2 + KDis * (T_F_us + T_F_ds - 2 * T_F_cs) /
GridSpacing ^ 2 + InterfaceArea * ConvCoeff * (T_S_cs - T_F_cs) - ThermalK_S * G * ((T_S_us +
T_S_ds - 2 * T_S_cs) / GridSpacing ^ 2 - (T_F_us + T_F_ds - 2 * T_F_cs) / GridSpacing ^ 2)) /
(Density_F * Cp_F * Porosity / WetSoilTimeStep)
2865
2866         'Write soil and water temperature to calc array for next timestep
2867         WetSoilCalcArray_S(1, WetSoilDepthIndex) = T_S_dt
2868         WetSoilCalcArray_F(1, WetSoilDepthIndex) = T_F_dt
2869
2870         Next WetSoilDepthIndex
2871
2872         'Copy temperatures from last calculated depth to bottom soil depth
2873         WetSoilCalcArray_S(1, WetSoilDepth / GridSpacing) = WetSoilCalcArray_S(1,
WetSoilDepth / GridSpacing - 1)
2874         WetSoilCalcArray_F(1, WetSoilDepth / GridSpacing) = WetSoilCalcArray_F(1,
WetSoilDepth / GridSpacing - 1)
2875
2876         'Copy surface temperatures into array

```

```

2877     WetSoilCalcArray_F(1, 0) = WetSoilCalcArray_F(0, 0)
2878     WetSoilCalcArray_S(1, 0) = WetSoilCalcArray_S(0, 0)
2879
2880
2881     'Store results to storage array if storage interval is met
2882     If (WetSoilTime - WetSoilTimeStored) >= WetSoilStoreFreq Then
2883         'Copy current time to time output array
2884         WetSoilOutTimeArray(WetSoilStoreIndex, 0) = SimDate
2885
2886         For WetSoilDepthIndex = 0 To (WetSoilDepth / GridSpacing)
2887             'Store soil temperatures
2888             WetSoilOutArray_S(WetSoilStoreIndex, WetSoilDepthIndex) =
WetSoilCalcArray_S(1, WetSoilDepthIndex)
2889             'Store fluid temperatures
2890             WetSoilOutArray_F(WetSoilStoreIndex, WetSoilDepthIndex) =
WetSoilCalcArray_F(1, WetSoilDepthIndex)
2891         Next WetSoilDepthIndex
2892
2893         'Store current time to TimeStored for use in next interval
2894         WetSoilTimeStored = WetSoilTime
2895
2896         'Advance WetSoilStoreIndex
2897         WetSoilStoreIndex = WetSoilStoreIndex + 1
2898
2899     End If
2900
2901
2902     'Copy temperatures from current timestep row to previous timestep row
2903     For WetSoilDepthIndex = 1 To (WetSoilDepth / GridSpacing)
2904         WetSoilCalcArray_S(0, WetSoilDepthIndex) = WetSoilCalcArray_S(1,
WetSoilDepthIndex)
2905         WetSoilCalcArray_F(0, WetSoilDepthIndex) = WetSoilCalcArray_F(1,
WetSoilDepthIndex)
2906     Next WetSoilDepthIndex
2907

```

```

2908         'Advance time and TimeIndex
2909         WetSoilTime = WetSoilTime + WetSoilTimeStep
2910         WetSoilTime = Round(WetSoilTime, 3) 'Round to prevent truncation error
2911         WetSoilTimeIndex = WetSoilTimeIndex + 1
2912
2913     End While
2914
2915
2916     'DEBUG MATERIAL
2917     'Stops execution time stopwatch and prints execution time in Debug line
2918     stopWatchWetSoil.Stop()
2919     System.Diagnostics.Debug.WriteLine("Wet Soil Total RunTime: " &
stopWatchWetSoil.Elapsed.Seconds & "." & stopWatchWetSoil.Elapsed.Milliseconds)
2920     'END DEBUG MATERIAL
2921
2922
2923
2924     '**-----**
2925     'Calculate two-phase temperatures for DRAINAGE phase
2926
2927     'Initialize variables
2928     Dim WetSoilDrainTime As Double 'Time in current simulation (s) used for determining
when to store array data
2929     Dim WetSoilDrainCalcArray_F(,) As Double 'Array to store fluid temperature for calc
purposes
2930     Dim WetSoilDrainCalcArray_S(,) As Double 'Array to store solid temperature for calc
purposes
2931     Dim WetSoilDrainSurfArray(,) As Double 'Array to store solid and fluid surface
temperatures
2932
2933     'Initialize variables for storing array data
2934     Dim WetSoilDrainOutArray_F(,) As Double 'Array for storing fluid temperature
calculation results
2935     Dim WetSoilDrainOutArray_S(,) As Double 'Array for storing soil temperature
calculation results

```

```

2936         Dim WetSoilDrainStoreIndex As Integer = 0 'Index for storing output to array
2937         Dim WetSoilOutDrainTimeArray(,) As Date 'Array for storing times for each row exported
to output array
2938
2939         'Dimension calculation arrays
2940         'For WetSoilCalcArrays, row 0 is previous timestep, row 1 is current timestep
2941         ReDim WetSoilDrainCalcArray_F(1, WetSoilDepth / GridSpacing)
2942         ReDim WetSoilDrainCalcArray_S(1, WetSoilDepth / GridSpacing)
2943         'For WetSoilSurfArray, rows are time, column 0 is solid, column 1 is fluid
2944         ReDim WetSoilDrainSurfArray(DrainTimeLength / WetSoilTimeStep, 1)
2945
2946         'Dimension arrays for output storage
2947         'Rows are times, columns are depths
2948         'First column is current time (s)
2949         'Calculate array length
2950         Dim WetSoilDrainOutArrayLength As Integer 'Length of array
2951         WetSoilDrainOutArrayLength = DrainTimeLength / WetSoilStoreFreq
2952         ReDim WetSoilDrainOutArray_F(WetSoilDrainOutArrayLength, WetSoilDepth / GridSpacing)
2953         ReDim WetSoilDrainOutArray_S(WetSoilDrainOutArrayLength, WetSoilDepth / GridSpacing)
2954         ReDim WetSoilOutDrainTimeArray(WetSoilDrainOutArrayLength, 0)
2955
2956         'Set initial conditions
2957         WetSoilDrainTime = 0
2958         SimDate = StormStopDate
2959
2960         'Set surface conditions for all times
2961         For WetSoilTimeIndex = 0 To (DrainTimeLength / WetSoilTimeStep)
2962             'Constant surface fluid temperature loaded from previous wetsoil simulations
2963             WetSoilDrainSurfArray(WetSoilTimeIndex, 1) = WetSoilSurfArray(WetSoilTimeLength /
WetSoilTimeStep, 1)
2964             'Constant surface solid temperature loaded from previous wetsoil simulations
2965             WetSoilDrainSurfArray(WetSoilTimeIndex, 0) = WetSoilCalcArray_S(1, 0)
2966         Next WetSoilTimeIndex
2967         'WetSoilDepth / GridSpacing
2968

```

```

2969         'Set initial conditions at all depths based on previous wet soil simulations
2970         For WetSoilDepthIndex = 0 To (WetSoilDepth / GridSpacing)
2971             WetSoilDrainCalcArray_F(0, WetSoilDepthIndex) = WetSoilCalcArray_F(1,
WetSoilDepthIndex)
2972             WetSoilDrainCalcArray_S(0, WetSoilDepthIndex) = WetSoilCalcArray_S(1,
WetSoilDepthIndex)
2973         Next WetSoilDepthIndex
2974
2975
2976         'Write initial conditions to output array
2977         'Copy current time to time output array
2978         WetSoilOutDrainTimeArray(WetSoilDrainStoreIndex, 0) = SimDate
2979         For WetSoilDepthIndex = 0 To (WetSoilDepth / GridSpacing)
2980             'Store soil temperatures
2981             WetSoilDrainOutArray_S(0, WetSoilDepthIndex) = WetSoilDrainCalcArray_S(0,
WetSoilDepthIndex)
2982             'Store fluid temperatures
2983             WetSoilDrainOutArray_F(0, WetSoilDepthIndex) = WetSoilDrainCalcArray_F(0,
WetSoilDepthIndex)
2984         Next WetSoilDepthIndex
2985         'Store current time to TimeStored for use in next interval
2986         WetSoilTimeStored = WetSoilDrainTime
2987         'Advance WetSoilStoreIndex
2988         WetSoilDrainStoreIndex = 1
2989
2990         'Manually advance time and timeindex variables for beginning of calculations
2991         WetSoilDrainTime = WetSoilTimeStep
2992         WetSoilTimeIndex = 1
2993
2994         'Set GAIndex to zero for loading fluid velocities
2995         DrainIndex = 0
2996
2997         'Load next date for change in fluid velocity
2998         NexGADate = DrainArray(0, DrainIndex + 1)
2999

```

```

3000     'Load initial fluid velocity and convert from m/hr to (m/s)
3001     Vel_F = DrainArray(2, DrainIndex) / 60 / 60
3002
3003
3004     'MAIN LOOP FOR WETSOIL CALCULATIONS
3005     While WetSoilTimeIndex <= (DrainTimeLength / WetSoilTimeStep)
3006
3007         'Advance SimDate
3008         SimDate = SimDate.AddSeconds(WetSoilTimeStep)
3009
3010         'Load fluid velocity
3011         If SimDate > NexGADate Then
3012             DrainIndex = DrainIndex + 1 'Advance DrainIndex
3013
3014             'Workaround for problem where tries to call index past the end of the array
3015             If DrainIndex + 1 < DrainArrayLength Then
3016                 Vel_F = DrainArray(2, DrainIndex) / 60 / 60 'Load fluid velocity and
convert from m/hr to (m/s)
3017                 NexGADate = DrainArray(0, DrainIndex + 1) 'Load date of next change in
fluid velocity
3018             End If
3019
3020         End If
3021
3022         'Calculate combination terms
3023         KDis = 0.5 * Density_F * Cp_F * Vel_F * ParticleDia
3024         ConvCoeff = (2 + 1.1 * Prandtl ^ (1 / 3) * (Density_F * Vel_F * ParticleDia /
Visc_F) ^ 0.6) * ThermalK_F
3025
3026         'Copy surface data to calculation array
3027         WetSoilDrainCalcArray_S(0, 0) = WetSoilDrainSurfArray(WetSoilTimeIndex, 0)
3028         WetSoilDrainCalcArray_F(0, 0) = WetSoilDrainSurfArray(WetSoilTimeIndex, 1)
3029
3030         'Run loop for remaining depths
3031         For WetSoilDepthIndex = 1 To (WetSoilDepth / GridSpacing - 1)

```

```

3032         'Load soil and water temperatures from array
3033         T_S_cs = WetSoilDrainCalcArray_S(0, WetSoilDepthIndex)
3034         T_S_ct = T_S_cs
3035         T_S_us = WetSoilDrainCalcArray_S(0, WetSoilDepthIndex - 1)
3036         T_S_ds = WetSoilDrainCalcArray_S(0, WetSoilDepthIndex + 1)
3037         T_F_cs = WetSoilDrainCalcArray_F(0, WetSoilDepthIndex)
3038         T_F_ct = T_F_cs
3039         T_F_us = WetSoilDrainCalcArray_F(0, WetSoilDepthIndex - 1)
3040         T_F_ds = WetSoilDrainCalcArray_F(0, WetSoilDepthIndex + 1)
3041
3042         'Solid
3043         T_S_dt = (Density_S * Cp_S * (1 - Porosity) * T_S_ct / WetSoilTimeStep + (1 -
Porosity + G * (Sigma - 1)) * ThermalK_S * ((T_S_us + T_S_ds - 2 * T_S_cs) / GridSpacing ^ 2)
- InterfaceArea * ConvCoeff * (T_S_cs - T_F_cs) + ThermalK_S * G * ((T_S_us + T_S_ds - 2 *
T_S_cs) / GridSpacing ^ 2 - (T_F_us + T_F_ds - 2 * T_F_cs) / GridSpacing ^ 2)) / (Density_S *
Cp_S * (1 - Porosity) / WetSoilTimeStep)
3044
3045         'Fluid
3046         Tr = (1 / 2) * (T_F_cs + T_F_ds) - (1 / 8) * (T_F_us + T_F_ds - 2 * T_F_cs)
3047         Tl = (1 / 2) * (T_F_us + T_F_cs) - (1 / 8) * (T_F_us + T_F_ds - 2 * T_F_cs)
3048         T_F_dt = (Density_F * Cp_F * Porosity * T_F_ct / WetSoilTimeStep - (Density_F
* Cp_F * Vel_F * (Tr - Tl)) / GridSpacing + (Porosity + G * (1 - Sigma)) * ThermalK_F *
(T_F_us + T_F_ds - 2 * T_F_cs) / GridSpacing ^ 2 + KDis * (T_F_us + T_F_ds - 2 * T_F_cs) /
GridSpacing ^ 2 + InterfaceArea * ConvCoeff * (T_S_cs - T_F_cs) - ThermalK_S * G * ((T_S_us +
T_S_ds - 2 * T_S_cs) / GridSpacing ^ 2 - (T_F_us + T_F_ds - 2 * T_F_cs) / GridSpacing ^ 2)) /
(Density_F * Cp_F * Porosity / WetSoilTimeStep)
3049
3050         'Write soil and water temperature to calc array for next timestep
3051         WetSoilDrainCalcArray_S(1, WetSoilDepthIndex) = T_S_dt
3052         WetSoilDrainCalcArray_F(1, WetSoilDepthIndex) = T_F_dt
3053
3054         Next WetSoilDepthIndex
3055
3056         'Copy temperatures from last calculated depth to bottom soil depth
3057         WetSoilDrainCalcArray_S(1, WetSoilDepth / GridSpacing) =

```

```

WetSoilDrainCalcArray_S(1, WetSoilDepth / GridSpacing - 1)
3058     WetSoilDrainCalcArray_F(1, WetSoilDepth / GridSpacing) =
WetSoilDrainCalcArray_F(1, WetSoilDepth / GridSpacing - 1)
3059
3060     'Copy surface temperatures into array
3061     WetSoilDrainCalcArray_F(1, 0) = WetSoilDrainCalcArray_F(0, 0)
3062     WetSoilDrainCalcArray_S(1, 0) = WetSoilDrainCalcArray_S(0, 0)
3063
3064
3065     'Store results to storage array if storage interval is met
3066     If (WetSoilDrainTime - WetSoilTimeStored) >= WetSoilStoreFreq Then
3067         'Copy current time to time output array
3068         WetSoilOutDrainTimeArray(WetSoilDrainStoreIndex, 0) = SimDate
3069
3070         For WetSoilDepthIndex = 0 To (WetSoilDepth / GridSpacing)
3071             'Store soil temperatures
3072             WetSoilDrainOutArray_S(WetSoilDrainStoreIndex, WetSoilDepthIndex) =
WetSoilDrainCalcArray_S(1, WetSoilDepthIndex)
3073             'Store fluid temperatures
3074             WetSoilDrainOutArray_F(WetSoilDrainStoreIndex, WetSoilDepthIndex) =
WetSoilDrainCalcArray_F(1, WetSoilDepthIndex)
3075         Next WetSoilDepthIndex
3076
3077         'Store current time to TimeStored for use in next interval
3078         WetSoilTimeStored = WetSoilDrainTime
3079
3080         'Advance WetSoilStoreIndex
3081         WetSoilDrainStoreIndex = WetSoilDrainStoreIndex + 1
3082
3083     End If
3084
3085     'Copy temperatures from current timestep row to previous timestep row
3086     For WetSoilDepthIndex = 1 To (WetSoilDepth / GridSpacing)
3087         WetSoilDrainCalcArray_S(0, WetSoilDepthIndex) = WetSoilDrainCalcArray_S(1,
WetSoilDepthIndex)

```

```

3088         WetSoilDrainCalcArray_F(0, WetSoilDepthIndex) = WetSoilDrainCalcArray_F(1,
WetSoilDepthIndex)
3089     Next WetSoilDepthIndex
3090
3091     'Advance time and TimeIndex
3092     WetSoilDrainTime = WetSoilDrainTime + WetSoilTimeStep
3093     WetSoilDrainTime = Round(WetSoilDrainTime, 3) 'Round to prevent truncation error
3094     WetSoilTimeIndex = WetSoilTimeIndex + 1
3095
3096 End While
3097
3098
3099     '-----*****-----
3100     'Build array containing all wet soil information for during and after storm
3101     'Dim WetSoilComboArray_F(,) As Double 'Combination of both sets of wetsoil simulations
for fluid phase
3102     'Dim WetSoilComboArray_S(,) As Double 'Combination of both sets of wetsoil simulations
for solid phase
3103     'Dim WetSoilComboTimeArray(,) As Date 'Date array for combination of wetsoil
simulations
3104     'Dim WetSoilComboArrayLength As Integer 'Length of combo arrays
3105     Dim WetSoilComboStoreIndex As Integer 'Index for storing values into Combo arrays
3106     Dim WetSoilComboLoadIndex As Integer 'Index for loading values from WetSoil arrays
3107
3108     'Update status label
3109     Label_Status.Text = "Status: Configuring temperature profile output arrays"
3110     Label_Status.Update()
3111
3112     'Calculate length of combo array
3113     WetSoilComboArrayLength = WetSoilOutArrayLength + WetSoilDrainOutArrayLength
3114
3115     'Calculate number of columns for use in determining which lines to plot
3116     WetSoilComboArray_NumCols = Round(SoilDepth / GridSpacing, 0)
3117
3118     'NOTE: WetSoilComboArrays only copy depths above drain, not all depths used in

```

```

calculations
3119
3120     'Redimension arrays
3121     ReDim WetSoilComboTimeArray(WetSoilComboArrayLength, 0)
3122     ReDim WetSoilComboArray_F(WetSoilComboArrayLength, SoilDepth / GridSpacing)
3123     ReDim WetSoilComboArray_S(WetSoilComboArrayLength, SoilDepth / GridSpacing)
3124
3125     'While copying values from WetSoil array during storm event
3126     While WetSoilComboStoreIndex < WetSoilOutArrayLength
3127
3128         'Copy time information
3129         WetSoilComboTimeArray(WetSoilComboStoreIndex, 0) =
WetSoilOutTimeArray(WetSoilComboLoadIndex, 0)
3130
3131         'Copy all depth information
3132         For WetSoilDepthIndex = 0 To SoilDepth / GridSpacing
3133             WetSoilComboArray_F(WetSoilComboStoreIndex, WetSoilDepthIndex) =
WetSoilOutArray_F(WetSoilComboLoadIndex, WetSoilDepthIndex)
3134             WetSoilComboArray_S(WetSoilComboStoreIndex, WetSoilDepthIndex) =
WetSoilOutArray_S(WetSoilComboLoadIndex, WetSoilDepthIndex)
3135         Next WetSoilDepthIndex
3136
3137         'Advance both index values
3138         WetSoilComboStoreIndex = WetSoilComboStoreIndex + 1
3139         WetSoilComboLoadIndex = WetSoilComboLoadIndex + 1
3140     End While
3141
3142
3143     'Reset load index because about to pull from new array
3144     WetSoilComboLoadIndex = 0
3145     While WetSoilComboStoreIndex <= WetSoilComboArrayLength
3146
3147         'Copy time information
3148         WetSoilComboTimeArray(WetSoilComboStoreIndex, 0) =
WetSoilOutDrainTimeArray(WetSoilComboLoadIndex, 0)

```

```

3149
3150     'Copy all depth information
3151     For WetSoilDepthIndex = 0 To SoilDepth / GridSpacing
3152         WetSoilComboArray_F(WetSoilComboStoreIndex, WetSoilDepthIndex) =
WetSoilDrainOutArray_F(WetSoilComboLoadIndex, WetSoilDepthIndex)
3153         WetSoilComboArray_S(WetSoilComboStoreIndex, WetSoilDepthIndex) =
WetSoilDrainOutArray_S(WetSoilComboLoadIndex, WetSoilDepthIndex)
3154     Next WetSoilDepthIndex
3155
3156     'Advance both index values
3157     WetSoilComboStoreIndex = WetSoilComboStoreIndex + 1
3158     WetSoilComboLoadIndex = WetSoilComboLoadIndex + 1
3159 End While
3160
3161
3162     '-----
3163     'Write Fluid results to an output array
3164     Dim theWriterF As System.IO.StreamWriter
3165     theWriterF = My.Computer.FileSystem.OpenTextFileWriter(TB_WetSoilFluidOutFile.Text,
False)
3166     'Write summary data
3167     theWriterF.WriteLine("Wet Soil Simulation")
3168     theWriterF.WriteLine("")
3169     'theWriter.WriteLine("Model Inputs:")
3170     'theWriter.WriteLine("Input Data File: " & InputRainFileName.ToString())
3171     theWriterF.WriteLine("Model Outputs:")
3172     theWriterF.WriteLine("")
3173
3174     'Write array data
3175     Dim OutRowF As Integer
3176     Dim OutColF As Integer
3177     'Write depth to first row of output table
3178     theWriterF.Write(",")
3179     For WetSoilDepthIndex = 0 To SoilDepth / GridSpacing
3180         theWriterF.Write((WetSoilDepthIndex * GridSpacing).ToString() & ",")

```

```

3181     Next WetSoilDepthIndex
3182     theWriterF.WriteLine()
3183
3184     For OutRowF = 0 To WetSoilComboArrayLength
3185         'Write current time for each row
3186         theWriterF.Write(WetSoilComboTimeArray(OutRowF, 0).ToString() & ",")
3187
3188         For OutColF = 0 To SoilDepth / GridSpacing
3189             theWriterF.Write(WetSoilComboArray_F(OutRowF, OutColF).ToString() & ",")
3190         Next OutColF
3191         theWriterF.WriteLine()
3192     Next OutRowF
3193     theWriterF.Close()
3194
3195
3196     'Write Soil results to an output array
3197     Dim theWriterS As System.IO.StreamWriter
3198     theWriterS = My.Computer.FileSystem.OpenTextFileWriter(TB_WetSoilSolidOutFile.Text,
False)
3199     'Write summary data
3200     theWriterS.WriteLine("Wet Soil Simulation")
3201     theWriterS.WriteLine("")
3202     'theWriter.WriteLine("Model Inputs:")
3203     theWriterS.WriteLine("Model Outputs:")
3204     theWriterS.WriteLine("")
3205
3206     'Write array data
3207     Dim OutRowS As Integer = 0
3208     Dim OutColS As Integer
3209     'Write depth to first row of output table
3210     theWriterS.Write(",")
3211     For WetSoilDepthIndex = 0 To SoilDepth / GridSpacing
3212         theWriterS.Write((WetSoilDepthIndex * GridSpacing).ToString() & ",")
3213     Next WetSoilDepthIndex
3214     theWriterS.WriteLine()

```

```

3215
3216     For OutRowS = 0 To WetSoilComboArrayLength
3217         'Write current time for each row
3218         theWriterS.Write(WetSoilComboTimeArray(OutRowS, 0).ToString() & ",")
3219
3220         For OutColS = 0 To SoilDepth / GridSpacing
3221             theWriterS.Write(WetSoilComboArray_S(OutRowS, OutColS).ToString() & ",")
3222         Next OutColS
3223         theWriterS.WriteLine()
3224     Next OutRowS
3225     theWriterS.Close()
3226
3227
3228
3229     '-----*****-----
3230     'Calculate thermal loads
3231
3232     'Update status label
3233     Label_Status.Text = "Status: Calculating thermal loads"
3234     Label_Status.Update()
3235
3236     'Dim ThermalLoadArray(,) As String 'Array for storing thermal load from runoff, to
drain, and underlying soil
3237     'ThermalLoadArray Contents: Date, Runoff load (W), Overflow load (W), Pipe load (W),
Soil load (W), Effluent load (W)
3238     'Dim ThermalLoadArrayLength As Integer 'Length of array, accounting for all runoff and
drain times
3239     Dim ThermalLoadTimestep As Double '(s) Timestep for performing thermal load
calculations
3240     Dim ThermalLoadDate As Date 'Current date in thermal load calculations
3241     Dim ThermalLoadIndex As Integer 'Index for storing information into ThermalLoadArray
3242
3243     'Dim ThermalEnergyArray(,) As String 'Array for storing cumulative thermal energy
3244     'ThermalEnergyArray Contents: Date, Runoff Energy (MJ), Overflow Energy (MJ), Pipe
Energy (MJ), Soil Energy (MJ), Effluent Energy (MJ)

```

```

3245
3246 'Initialize variables to store information loaded from arrays
3247 Dim RunoffMassRate As Double 'Mass flowrate of runoff (kg/s)
3248 Dim OverflowMassRate As Double 'Mass flowrate of bioretention overflow (kg/s)
3249 Dim PipeMassRate As Double 'Mass flowrate of pipe drainage (kg/s)
3250 Dim SeepageMassRate As Double 'Mass flowrate of seepage (kg/s)
3251 Dim SurfaceWaterTemp As Double 'Water temperature at bioretention surface (°C)
3252 Dim BottomWaterTemp As Double 'Water temperature at bottom of bioretention (°C)
3253
3254 'Initialize variables to reference array load dates
3255 Dim NextInflowArrayDate As Date
3256 Dim NextHydrologyArrayDate As Date
3257 Dim NextWetSoilComboArrayDate As Date
3258
3259 'Initialize variables to calculate loads
3260 Dim RunoffThermalLoad As Double 'Thermal load of runoff (W)
3261 Dim OverflowThermalLoad As Double 'Thermal load of overflow (W)
3262 Dim PipeThermalLoad As Double 'Thermal load of pipe flow (W)
3263 Dim SeepageThermalLoad As Double 'Thermal load of seepage (W)
3264 Dim EffluentThermalLoad As Double 'Combined thermal load of overflow and pipe (W)
3265
3266 'Initialize variables to calculate total energy
3267 Dim RunoffTotalEnergy As Double 'Cumulative energy (J)
3268 Dim OverflowTotalEnergy As Double 'Cumulative energy (J)
3269 Dim PipeTotalEnergy As Double 'Cumulative energy (J)
3270 Dim SeepageTotalEnergy As Double 'Cumulative energy (J)
3271 Dim EffluentTotalEnergy As Double 'Combined energy of overflow and pipe (J)
3272 Dim ThermalReduction As Double 'Reduction of thermal load (%)
3273 Dim VolReduction As Double 'Reduction of runoff volume (%)
3274
3275 Dim RunoffVolCheck As Double
3276 Dim OverflowVolCheck As Double
3277 Dim PipeVolCheck As Double
3278 Dim SeepageVolCheck As Double
3279

```

```

3280 'Initialize variables for average temperature
3281 Dim RunoffTempTotal As Double 'Total of all runoff temperatures
3282 Dim OverflowTempTotal As Double
3283 Dim PipeTempTotal As Double
3284 Dim SeepageTempTotal As Double
3285 Dim EffluentTempTotal As Double
3286 Dim RunoffTempAVG As Double 'Average of all runoff temperatures
3287 Dim OverflowTempAVG As Double
3288 Dim PipeTempAVG As Double
3289 Dim SeepageTempAVG As Double
3290 Dim EffluentTempAVG As Double
3291 Dim RunoffAVGIndex As Integer 'Total number of temperature measurements
3292 Dim OverflowAVGIndex As Integer
3293 Dim PipeAVGIndex As Integer
3294 Dim SeepageAVGIndex As Integer
3295 Dim EffluentAVGIndex As Integer
3296
3297 'Initialize variables to maximum temperature
3298 Dim RunoffTempMax As Double 'Maximum runoff temperature
3299 Dim OverflowTempMax As Double
3300 Dim PipeTempMax As Double
3301 Dim SeepageTempMax As Double
3302 Dim EffluentTempMax As Double
3303
3304
3305 'Set timestep
3306 'NOTE: timestep can't be larger than the most precise timestep of loaded data in order
to load correct array data
3307 'Currently most precise timestep is from HydrologyArray
3308 ThermalLoadTimestep = 1
3309 'Calculate array length based on length of time
3310 ThermalLoadArrayLength = (RunoffDuration + DrainTimeLength) / ThermalLoadTimestep
3311 'Redimension array
3312 ReDim ThermalLoadArray(ThermalLoadArrayLength, 5)
3313 ReDim ThermalEnergyArray(ThermalLoadArrayLength, 5)

```

```

3314
3315     'Initialize ThermalLoadDate at the start of runoff
3316 ThermalLoadDate = RunoffStartDate
3317
3318     'Reset various indices that will be used to load array data
3319 Rain_R = 0
3320 GAIndex = 0
3321 WetSoilComboStoreIndex = 0
3322
3323     'Set new load dates for all arrays
3324 NextInflowArrayDate = InflowArray(Rain_R + 1, 0)
3325 NextHydrologyArrayDate = HydrologyComboArray(GAIndex + 1, 0)
3326 NextWetSoilComboArrayDate = WetSoilComboTimeArray(GAIndex + 1, 0)
3327
3328     '*****
3329     'Main loop for thermal load calculations
3330 While ThermalLoadDate < DrainTimeStop
3331
3332         'If runoff rate has changed
3333         If ThermalLoadDate >= NextInflowArrayDate Then
3334             'If rainfall is over, no runoff
3335             If ThermalLoadDate > StormStopDate Then
3336                 RunoffMassRate = 0
3337                 'If rainfall is not over
3338             Else
3339                 'Advance Rain_R
3340                 Rain_R = Rain_R + 1
3341                 'Load new RunoffMassRate
3342                 RunoffMassRate = InflowArray(Rain_R, 1) * 1000 'Load inflow as (m³/s) and
convert to (kg/s)
3343                 'RunoffMassRate = GreenAmptArray(Rain_R, 2) * BioretentionArea * 1000 / 60
/ 60 'Load inflow as (m³/s) and convert to (kg/s)
3344
3345                 'Set new load date
3346                 'If statement prevents indexing past the end of the array

```

```

3347         If Rain_R + 1 <= InflowIndexLength Then
3348             NextInflowArrayDate = InflowArray(Rain_R + 1, 0)
3349         End If
3350     End If
3351 End If
3352
3353     'If HydrologyComboArray contents have changed
3354     If ThermalLoadDate >= NextHydrologyArrayDate Then
3355         'Advance GAIndex
3356         GAIndex = GAIndex + 1
3357
3358         'Load OverflowMassRate
3359         OverflowMassRate = HydrologyComboArray(GAIndex, 1) * BioretentionArea * 1000 /
60 / 60 'Load overflow as (m/hr) and convert to (kg/s)
3360         'Load SeepageMassRate
3361         SeepageMassRate = HydrologyComboArray(GAIndex, 2) * BioretentionArea * 1000 /
60 / 60 'Load seepage as (m/hr) and convert to (kg/s)
3362         'LoadPipeMassRate
3363         PipeMassRate = HydrologyComboArray(GAIndex, 3) * BioretentionArea * 1000 / 60
/ 60 'Load pipe flow as (m/hr) and convert to (kg/s)
3364
3365         'Set new load date
3366         'If statement prevents indexing past the end of the array
3367         If GAIndex + 1 < GAIndexLength Then
3368             NextHydrologyArrayDate = HydrologyComboArray(GAIndex + 1, 0)
3369         End If
3370     End If
3371
3372     'If WetSoilComboArray contents have changed
3373     If ThermalLoadDate >= NextWetSoilComboArrayDate And WetSoilComboStoreIndex + 1 <=
WetSoilComboArrayLength Then
3374         'Advance WetSoilComboStoreIndex
3375         WetSoilComboStoreIndex = WetSoilComboStoreIndex + 1
3376
3377         'Load SurfaceWaterTemp

```

```

3378         SurfaceWaterTemp = WetSoilComboArray_F(WetSoilComboStoreIndex, 0)
3379         'Load BottomWaterTemp
3380         BottomWaterTemp = WetSoilComboArray_F(WetSoilComboStoreIndex, SoilDepth /
GridSpacing)
3381
3382         'Set new load date
3383         'If statement prevents indexing past the end of the array
3384         If WetSoilComboStoreIndex + 1 < WetSoilComboArrayLength Then
3385             NextWetSoilComboArrayDate = WetSoilComboTimeArray(WetSoilComboStoreIndex +
1, 0)
3386         End If
3387
3388     End If
3389
3390     'Thermal Load: heat (W) = mass flow rate (kg/s) * Specific Heat (J/kg*K) *
Temperature (K)
3391
3392     'Calculate thermal loads
3393     RunoffThermalLoad = RunoffMassRate * 4180 * (SurfaceWaterTemp + 273.15)
3394     OverflowThermalLoad = OverflowMassRate * 4180 * (SurfaceWaterTemp + 273.15)
3395     PipeThermalLoad = PipeMassRate * 4180 * (BottomWaterTemp + 273.15)
3396     SeepageThermalLoad = SeepageMassRate * 4180 * (BottomWaterTemp + 273.15)
3397     EffluentThermalLoad = PipeThermalLoad + OverflowThermalLoad
3398
3399     'Store thermal load data to array
3400     'ThermalLoadArray Contents: Date, Runoff load (W), Overflow load (W), Pipe load
(W), Soil load (W)
3401     'Copy Date
3402     ThermalLoadArray(ThermalLoadIndex, 0) = ThermalLoadDate
3403     'Copy Runoff load
3404     ThermalLoadArray(ThermalLoadIndex, 1) = RunoffThermalLoad
3405     'Copy Overflow load
3406     ThermalLoadArray(ThermalLoadIndex, 2) = OverflowThermalLoad
3407     'Copy Pipe load
3408     ThermalLoadArray(ThermalLoadIndex, 3) = PipeThermalLoad

```

```

3409     'Copy Seepage load
3410     ThermalLoadArray(ThermalLoadIndex, 4) = SeepageThermalLoad
3411     'Copy Effluent load
3412     ThermalLoadArray(ThermalLoadIndex, 5) = EffluentThermalLoad
3413
3414
3415     'Update total energy values
3416     RunoffTotalEnergy = RunoffTotalEnergy + RunoffThermalLoad * ThermalLoadTimestep
3417     OverflowTotalEnergy = OverflowTotalEnergy + OverflowThermalLoad *
ThermalLoadTimestep
3418     PipeTotalEnergy = PipeTotalEnergy + PipeThermalLoad * ThermalLoadTimestep
3419     SeepageTotalEnergy = SeepageTotalEnergy + SeepageThermalLoad * ThermalLoadTimestep
3420     EffluentTotalEnergy = OverflowTotalEnergy + PipeTotalEnergy
3421
3422     'Store thermal energy data to array
3423     'Convert from (J) to (MJ) and round
3424     'Copy Date
3425     ThermalEnergyArray(ThermalLoadIndex, 0) = ThermalLoadDate
3426     'Copy Runoff load
3427     ThermalEnergyArray(ThermalLoadIndex, 1) = Round(RunoffTotalEnergy / 10 ^ 6, 0)
3428     'Copy Overflow load
3429     ThermalEnergyArray(ThermalLoadIndex, 2) = Round(OverflowTotalEnergy / 10 ^ 6, 0)
3430     'Copy Pipe load
3431     ThermalEnergyArray(ThermalLoadIndex, 3) = Round(PipeTotalEnergy / 10 ^ 6, 0)
3432     'Copy Seepage load
3433     ThermalEnergyArray(ThermalLoadIndex, 4) = Round(SeepageTotalEnergy / 10 ^ 6, 0)
3434     'Copy Effluent load
3435     ThermalEnergyArray(ThermalLoadIndex, 5) = Round(EffluentTotalEnergy / 10 ^ 6, 0)
3436
3437     'Update total temperature values
3438     If RunoffMassRate > 0 Then
3439         RunoffTempTotal = RunoffTempTotal + SurfaceWaterTemp
3440         RunoffAVGIndex = RunoffAVGIndex + 1
3441         'Check max temperature
3442         If SurfaceWaterTemp > RunoffTempMax Then

```

```

3443         RunoffTempMax = SurfaceWaterTemp
3444     End If
3445 End If
3446 If OverflowMassRate > 0 Then
3447     OverflowTempTotal = OverflowTempTotal + SurfaceWaterTemp
3448     OverflowAVGIndex = OverflowAVGIndex + 1
3449     'Check max temperature
3450     If SurfaceWaterTemp > OverflowTempMax Then
3451         OverflowTempMax = SurfaceWaterTemp
3452     End If
3453 End If
3454 If PipeMassRate > 0 Then
3455     PipeTempTotal = PipeTempTotal + BottomWaterTemp
3456     PipeAVGIndex = PipeAVGIndex + 1
3457     'Check max temperature
3458     If BottomWaterTemp > PipeTempMax Then
3459         PipeTempMax = BottomWaterTemp
3460     End If
3461 End If
3462 If SeepageMassRate > 0 Then
3463     SeepageTempTotal = SeepageTempTotal + BottomWaterTemp
3464     SeepageAVGIndex = SeepageAVGIndex + 1
3465     'Check max temperature
3466     If BottomWaterTemp > SeepageTempMax Then
3467         SeepageTempMax = BottomWaterTemp
3468     End If
3469 End If
3470 If EffluentThermalLoad > 0 Then
3471     EffluentTempTotal = EffluentTempTotal + BottomWaterTemp
3472     EffluentAVGIndex = EffluentAVGIndex + 1
3473 End If
3474
3475 RunoffVolCheck = RunoffVolCheck + RunoffMassRate * ThermalLoadTimestep
3476 OverflowVolCheck = OverflowVolCheck + OverflowMassRate * ThermalLoadTimestep
3477 PipeVolCheck = PipeVolCheck + PipeMassRate * ThermalLoadTimestep

```

```

3478         SeepageVolCheck = SeepageVolCheck + SeepageMassRate * ThermalLoadTimestep
3479
3480
3481         'Advance ThermalLoadDate and ThermalLoadIndex
3482         ThermalLoadDate = ThermalLoadDate.AddSeconds(ThermalLoadTimestep)
3483         ThermalLoadIndex = ThermalLoadIndex + 1
3484
3485     End While
3486
3487
3488     'Write thermal load results to an output file
3489     Dim theWriterThermal As System.IO.StreamWriter
3490     theWriterThermal =
3491     My.Computer.FileSystem.OpenTextFileWriter(TB_ThermalLoadOutFile.Text, False)
3492     'Write summary data
3493     theWriterThermal.WriteLine("Thermal Load Results")
3494     theWriterThermal.WriteLine("")
3495     theWriterThermal.WriteLine("Model Outputs:")
3496     theWriterThermal.WriteLine("")
3497     theWriterThermal.WriteLine("Date, Runoff load (W), Overflow load (W), Pipe load (W),
3498     Soil load (W), Effluent load (W)")
3499
3500     'Write array data
3501     Dim OutRowThermal As Integer
3502     Dim OutColThermal As Integer
3503     For OutRowThermal = 0 To ThermalLoadArrayLength - 1
3504         For OutColThermal = 0 To 5
3505             theWriterThermal.Write(ThermalLoadArray(OutRowThermal,
3506             OutColThermal).ToString() & ",")
3507         Next OutColThermal
3508         theWriterThermal.WriteLine()
3509     Next OutRowThermal
3510     theWriterThermal.Close()

```

```

3510 'Calculate average temperatures
3511 RunoffTempAVG = Round(RunoffTempTotal / RunoffAVGIndex, 1)
3512 EffluentTempAVG = Round(EffluentTempTotal / EffluentAVGIndex, 1)
3513 OverflowTempAVG = Round(OverflowTempTotal / OverflowAVGIndex, 1)
3514 PipeTempAVG = Round(PipeTempTotal / PipeAVGIndex, 1)
3515 SeepageTempAVG = Round(SeepageTempTotal / SeepageAVGIndex, 1)
3516
3517 'Write average temperatures to output page of GUI
3518 TB_RunoffAvgTemp.Text = RunoffTempAVG.ToString
3519 TB_EffluentAvgTemp.Text = EffluentTempAVG.ToString
3520 TB_OverflowAvgTemp.Text = OverflowTempAVG.ToString
3521 TB_PipeAvgTemp.Text = PipeTempAVG.ToString
3522 TB_SeepageAvgTemp.Text = SeepageTempAVG.ToString
3523
3524 'Write total energy amounts to output page of GUI
3525 'Convert from (J) to (MJ)
3526 TB_RunoffEnergy.Text = Round((RunoffTotalEnergy / 10 ^ 6), 0).ToString
3527 TB_EffluentEnergy.Text = Round((EffluentTotalEnergy / 10 ^ 6), 0).ToString
3528 TB_OverflowEnergy.Text = Round((OverflowTotalEnergy / 10 ^ 6), 0).ToString
3529 TB_PipeEnergy.Text = Round((PipeTotalEnergy / 10 ^ 6), 0).ToString
3530 TB_SeepageEnergy.Text = Round((SeepageTotalEnergy / 10 ^ 6), 0).ToString
3531 'Calculate thermal and volume reductions
3532 ThermalReduction = 100 - EffluentTotalEnergy / RunoffTotalEnergy * 100
3533 VolReduction = 100 - (PipeVolCheck + OverflowVolCheck) / RunoffVolCheck * 100
3534 'Write results to GUI
3535 TB_ThermalReduction.Text = (Round(ThermalReduction, 1) & " %")
3536 TB_VolReduction.Text = (Round(VolReduction, 1) & " %")
3537
3538
3539 'Calculate maximum effluent temperature
3540 If OverflowTempMax > PipeTempMax Then
3541     EffluentTempMax = OverflowTempMax
3542 Else
3543     EffluentTempMax = PipeTempMax
3544 End If

```

```

3545
3546     'Write maximum temperatures to output page of GUI
3547     TB_RunoffMaxTemp.Text = Round(RunoffTempMax, 1)
3548     TB_EffluentMaxTemp.Text = Round(EffluentTempMax, 1)
3549     TB_OverflowMaxTemp.Text = Round(OverflowTempMax, 1)
3550     TB_PipeMaxTemp.Text = Round(PipeTempMax, 1)
3551     TB_SeepageMaxTemp.Text = Round(SeepageTempMax, 1)
3552
3553
3554     'DEBUG MATERIAL
3555     'Stops total execution time stopwatch and prints execution time in Debug line
3556     stopWatchTotalTime.Stop()
3557     System.Diagnostics.Debug.WriteLine("Total Execution Time: " &
stopWatchTotalTime.Elapsed.Seconds & "." & stopWatchTotalTime.Elapsed.Milliseconds)
3558     'END DEBUG MATERIAL
3559
3560     'Update status label
3561     Label_Status.Text = "Status: Simulation complete"
3562     Label_Status.Update()
3563
3564     'Switch tab to outputs page
3565     MainTabControl.SelectedTab = TabOutput
3566
3567     'DEBUG MATERIAL
3568     System.Diagnostics.Debug.WriteLine("END=====END")
3569     'END DEBUG MATERIAL
3570
3571     End Sub
3572
3573
3574
3575
3576 #Region "Load XML User Data"
3577     'Code for loading user data from an XML file
3578     Private Sub OpenFileDialogStripMenuItem_Click(ByVal sender As System.Object, ByVal e As

```

```

System.EventArgs) Handles OpenFileDialog.Click
3579     ' Filters requested files
3580     XMLOpenFileDialog.Filter = "xml files (*.xml)|*.xml|All files (*.*)|*.*"
3581     ' if the user did not click on the OK button, return
3582     If XMLOpenFileDialog.ShowDialog(Me) <> Windows.Forms.DialogResult.OK Then
3583         Return
3584     End If
3585     ' Sets filename variable and updates main page
3586     Dim XMLOpenFileName = XMLOpenFileDialog.FileName
3587
3588     'Update file label with name of xml file
3589     Label_LoadedXMLFile.Visible = True 'Show label on GUI
3590     Label_LoadedXMLFile.Text = XMLOpenFileName
3591
3592     ' load the xml document from the given path
3593     Dim xml = New XmlDocument()
3594     xml.Load(XMLOpenFileName)
3595
3596     ' get the different nodes from the xml file and put them in the text boxes.
3597     ' NOTE: add a check for the values being Nothing.
3598     Dim L_RainFileName = xml.SelectSingleNode("//RainFileName")
3599     TB_RainFileName.Text = L_RainFileName.InnerText
3600     TB_RainFileName.SelectionStart = Len(TB_RainFileName.Text) 'Moves filename view to see
the end
3601
3602     Dim L_WeatherFileName = xml.SelectSingleNode("//WeatherFileName")
3603     TB_WeatherFileName.Text = L_WeatherFileName.InnerText
3604     TB_WeatherFileName.SelectionStart = Len(TB_WeatherFileName.Text) 'Moves filename view
to see the end
3605
3606     Dim L_HydraulicOutFile = xml.SelectSingleNode("//HydraulicOutFile")
3607     TB_HydraulicOutFile.Text = L_HydraulicOutFile.InnerText
3608     TB_HydraulicOutFile.SelectionStart = Len(TB_HydraulicOutFile.Text) 'Moves filename
view to see the end
3609

```

```

3610     Dim L_DryPaveOutFile = xml.SelectSingleNode("//DryPaveOutFile")
3611     TB_DryPaveOutFile.Text = L_DryPaveOutFile.InnerText
3612     TB_DryPaveOutFile.SelectionStart = Len(TB_DryPaveOutFile.Text) 'Moves filename view to
see the end
3613
3614     Dim L_WetPaveOutFile = xml.SelectSingleNode("//WetPaveOutFile")
3615     TB_WetPaveOutFile.Text = L_WetPaveOutFile.InnerText
3616     TB_WetPaveOutFile.SelectionStart = Len(TB_WetPaveOutFile.Text) 'Moves filename view to
see the end
3617
3618     Dim L_WetSoilFluidOutFile = xml.SelectSingleNode("//WetSoilFluidOutFile")
3619     TB_WetSoilFluidOutFile.Text = L_WetSoilFluidOutFile.InnerText
3620     TB_WetSoilFluidOutFile.SelectionStart = Len(TB_WetSoilFluidOutFile.Text) 'Moves
filename view to see the end
3621
3622     Dim L_WetSoilSolidOutFile = xml.SelectSingleNode("//WetSoilSolidOutFile")
3623     TB_WetSoilSolidOutFile.Text = L_WetSoilSolidOutFile.InnerText
3624     TB_WetSoilSolidOutFile.SelectionStart = Len(TB_WetSoilSolidOutFile.Text) 'Moves
filename view to see the end
3625
3626     Dim L_ThermalLoadOutFile = xml.SelectSingleNode("//ThermalLoadOutFile")
3627     TB_ThermalLoadOutFile.Text = L_ThermalLoadOutFile.InnerText
3628     TB_ThermalLoadOutFile.SelectionStart = Len(TB_ThermalLoadOutFile.Text) 'Moves filename
view to see the end
3629
3630     Dim L_WatershedArea = xml.SelectSingleNode("//WatershedArea")
3631     TB_WatershedArea.Text = L_WatershedArea.InnerText
3632
3633     Dim L_InitialAbstraction = xml.SelectSingleNode("//InitialAbstraction")
3634     TB_InitialAbstraction.Text = L_InitialAbstraction.InnerText
3635
3636     Dim L_BioretentionArea = xml.SelectSingleNode("//BioretentionArea")
3637     TB_BioretentionArea.Text = L_BioretentionArea.InnerText
3638
3639     Dim L_SoilDepth = xml.SelectSingleNode("//SoilDepth")

```

```
3640 TB_SoilDepth.Text = L_SoilDepth.InnerText
3641
3642 Dim L_Suction = xml.SelectSingleNode("//Suction")
3643 TB_Suction.Text = L_Suction.InnerText
3644
3645 Dim L_KSat = xml.SelectSingleNode("//KSat")
3646 TB_KSat.Text = L_KSat.InnerText
3647
3648 Dim L_ThetaFilled = xml.SelectSingleNode("//ThetaFilled")
3649 TB_ThetaFilled.Text = L_ThetaFilled.InnerText
3650
3651 Dim L_StorageCapacity = xml.SelectSingleNode("//StorageCapacity")
3652 TB_StorageCapacity.Text = L_StorageCapacity.InnerText
3653
3654 Dim L_GravelPorosity = xml.SelectSingleNode("//GravelPorosity")
3655 TB_GravelPorosity.Text = L_GravelPorosity.InnerText
3656
3657 Dim L_DrainHeight = xml.SelectSingleNode("//DrainHeight")
3658 TB_DrainHeight.Text = L_DrainHeight.InnerText
3659
3660 Dim L_SubSoilKSat = xml.SelectSingleNode("//SubSoilKSat")
3661 TB_SubSoilKSat.Text = L_SubSoilKSat.InnerText
3662
3663 Dim L_LatDeg = xml.SelectSingleNode("//LatDeg")
3664 TB_LatDeg.Text = L_LatDeg.InnerText
3665
3666 Dim L_LonDeg = xml.SelectSingleNode("//LonDeg")
3667 TB_LonDeg.Text = L_LonDeg.InnerText
3668
3669 Dim L_Transmittance = xml.SelectSingleNode("//Transmittance")
3670 TB_Transmittance.Text = L_Transmittance.InnerText
3671
3672 Dim L_Elevation = xml.SelectSingleNode("//Elevation")
3673 TB_Elevation.Text = L_Elevation.InnerText
3674
```

```
3675 Dim L_Timezone = xml.SelectSingleNode("//Timezone")
3676 TB_Timezone.Text = L_Timezone.InnerText
3677
3678 Dim L_Shading = xml.SelectSingleNode("//Shading")
3679 TB_Shading.Text = L_Shading.InnerText
3680
3681 Dim L_AirTemperature = xml.SelectSingleNode("//AirTemperature")
3682 TB_AirTemperature.Text = L_AirTemperature.InnerText
3683
3684 Dim L_WindSpeed = xml.SelectSingleNode("//WindSpeed")
3685 TB_WindSpeed.Text = L_WindSpeed.InnerText
3686
3687 Dim L_Radiation = xml.SelectSingleNode("//Radiation")
3688 TB_Radiation.Text = L_Radiation.InnerText
3689
3690 Dim L_RelHumidity = xml.SelectSingleNode("//RelHumidity")
3691 TB_RelHumidity.Text = L_RelHumidity.InnerText
3692
3693 Dim L_Cp_S = xml.SelectSingleNode("//Cp_S")
3694 TB_Cp_S.Text = L_Cp_S.InnerText
3695
3696 Dim L_Porosity = xml.SelectSingleNode("//Porosity")
3697 TB_Porosity.Text = L_Porosity.InnerText
3698
3699 Dim L_ThermalK_S = xml.SelectSingleNode("//ThermalK_S")
3700 TB_ThermalK_S.Text = L_ThermalK_S.InnerText
3701
3702 Dim L_InterfaceArea = xml.SelectSingleNode("//InterfaceArea")
3703 TB_InterfaceArea.Text = L_InterfaceArea.InnerText
3704
3705 Dim L_ThermalK_Stg = xml.SelectSingleNode("//ThermalK_Stg")
3706 TB_ThermalK_Stg.Text = L_ThermalK_Stg.InnerText
3707
3708 Dim L_ParticleDia = xml.SelectSingleNode("//ParticleDia")
3709 TB_ParticleDia.Text = L_ParticleDia.InnerText
```

```

3710
3711     End Sub
3712
3713 #End Region
3714
3715
3716
3717 #Region "Save XML User Data"
3718     'Code for saving information stored in the form to an XML file
3719     Private Sub SaveFileToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles SaveFileToolStripMenuItem.Click
3720         'Filters possible output file formats
3721         XMLSaveFileDialog.Filter = "xml files (*.xml)|*.xml|All files (*.*)|*.*"
3722         ' if the user did not click on the OK button, return
3723         If XMLSaveFileDialog.ShowDialog(Me) <> Windows.Forms.DialogResult.OK Then
3724             Return
3725         End If
3726
3727         ' get the file name.
3728         Dim XMLSaveFileName = XMLSaveFileDialog.FileName
3729
3730         ' compile the xml.
3731         Dim xml = <?xml version="1.0" encoding="utf-8"?>
3732             <SavedData>
3733                 <RainFileName><%= TB_RainFileName.Text %></RainFileName>
3734                 <WeatherFileName><%= TB_WeatherFileName.Text %></WeatherFileName>
3735                 <HydraulicOutFile><%= TB_HydraulicOutFile.Text %></HydraulicOutFile>
3736                 <DryPaveOutFile><%= TB_DryPaveOutFile.Text %></DryPaveOutFile>
3737                 <WetPaveOutFile><%= TB_WetPaveOutFile.Text %></WetPaveOutFile>
3738                 <WetSoilFluidOutFile><%= TB_WetSoilFluidOutFile.Text
%></WetSoilFluidOutFile>
3739                 <WetSoilSolidOutFile><%= TB_WetSoilSolidOutFile.Text
%></WetSoilSolidOutFile>
3740                 <ThermalLoadOutFile><%= TB_ThermalLoadOutFile.Text
%></ThermalLoadOutFile>

```

```

3741     <WatershedArea><%= TB_WatershedArea.Text %></WatershedArea>
3742     <InitialAbstraction><%= TB_InitialAbstraction.Text
%></InitialAbstraction>
3743     <BioretentionArea><%= TB_BioretentionArea.Text %></BioretentionArea>
3744     <SoilDepth><%= TB_SoilDepth.Text %></SoilDepth>
3745     <Suction><%= TB_Suction.Text %></Suction>
3746     <KSat><%= TB_KSat.Text %></KSat>
3747     <ThetaFilled><%= TB_ThetaFilled.Text %></ThetaFilled>
3748     <StorageCapacity><%= TB_StorageCapacity.Text %></StorageCapacity>
3749     <GravelPorosity><%= TB_GravelPorosity.Text %></GravelPorosity>
3750     <DrainHeight><%= TB_DrainHeight.Text %></DrainHeight>
3751     <SubSoilKSat><%= TB_SubSoilKSat.Text %></SubSoilKSat>
3752     <LatDeg><%= TB_LatDeg.Text %></LatDeg>
3753     <LonDeg><%= TB_LonDeg.Text %></LonDeg>
3754     <Transmittance><%= TB_Transmittance.Text %></Transmittance>
3755     <Elevation><%= TB_Elevation.Text %></Elevation>
3756     <Timezone><%= TB_Timezone.Text %></Timezone>
3757     <Shading><%= TB_Shading.Text %></Shading>
3758     <AirTemperature><%= TB_AirTemperature.Text %></AirTemperature>
3759     <WindSpeed><%= TB_WindSpeed.Text %></WindSpeed>
3760     <Radiation><%= TB_Radiation.Text %></Radiation>
3761     <RelHumidity><%= TB_RelHumidity.Text %></RelHumidity>
3762     <Cp_S><%= TB_Cp_S.Text %></Cp_S>
3763     <Porosity><%= TB_Porosity.Text %></Porosity>
3764     <ThermalK_S><%= TB_ThermalK_S.Text %></ThermalK_S>
3765     <InterfaceArea><%= TB_InterfaceArea.Text %></InterfaceArea>
3766     <ThermalK_Stg><%= TB_ThermalK_Stg.Text %></ThermalK_Stg>
3767     <ParticleDia><%= TB_ParticleDia.Text %></ParticleDia>
3768
3769     </SavedData>
3770
3771     ' write the xml to the file.
3772     xml.Save (XMLSaveFileName)
3773 End Sub
3774

```

```

3775 #End Region
3776
3777
3778
3779 #Region "HydraulicOutFile Select"
3780     'Executes when the HydraulicOutFile browse button is clicked
3781     Private Sub BUT_BrowseHydraulicOutFile_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_BrowseHydraulicOutFile.Click
3782         ' Filters requested files
3783         HydraulicOutSaveFileDialog.Filter = "csv files (*.csv)|*.csv|All files (*.*)|*.*"
3784         If HydraulicOutSaveFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
3785             End If
3786         ' Sets filename variable and updates main page
3787         HydraulicOutFileName = HydraulicOutSaveFileDialog.FileName
3788         TB_HydraulicOutFile.Text = HydraulicOutFileName
3789         TB_HydraulicOutFile.SelectionStart = Len(TB_HydraulicOutFile.Text)
3790     End Sub
3791 #End Region
3792
3793
3794 #Region "DryPaveOutFile Select"
3795     'Executes when the DryPaveOutFile browse button is clicked
3796     Private Sub BUT_BrowseDryPaveOutFile_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_BrowseDryPaveOutFile.Click
3797         ' Filters requested files
3798         DryPaveOutSaveFileDialog.Filter = "csv files (*.csv)|*.csv|All files (*.*)|*.*"
3799         If DryPaveOutSaveFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
3800             End If
3801         ' Sets filename variable and updates main page
3802         DryPaveOutFileName = DryPaveOutSaveFileDialog.FileName
3803         TB_DryPaveOutFile.Text = DryPaveOutFileName
3804         TB_DryPaveOutFile.SelectionStart = Len(TB_DryPaveOutFile.Text)
3805     End Sub
3806 #End Region
3807

```

```

3808
3809 #Region "WetPaveOutFile Select"
3810     'Executes when the HydraulicOutFile browse button is clicked
3811     Private Sub BUT_BrowseWetPaveOutFile_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_BrowseWetPaveOutFile.Click
3812         ' Filters requested files
3813         WetPaveOutSaveFileDialog.Filter = "csv files (*.csv)|*.csv|All files (*.*)|*.*"
3814         If WetPaveOutSaveFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
3815             End If
3816         ' Sets filename variable and updates main page
3817         WetPaveOutFileName = WetPaveOutSaveFileDialog.FileName
3818         TB_WetPaveOutFile.Text = WetPaveOutFileName
3819         TB_WetPaveOutFile.SelectionStart = Len(TB_WetPaveOutFile.Text)
3820     End Sub
3821 #End Region
3822
3823
3824 #Region "FluidOutFile Select"
3825     'Executes when the FluidOutFile browse button is clicked
3826     Private Sub BUT_BrowseFluidOutFile_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_BrowseFluidOutFile.Click
3827         ' Filters requested files
3828         FluidOutSaveFileDialog.Filter = "csv files (*.csv)|*.csv|All files (*.*)|*.*"
3829         If FluidOutSaveFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
3830             End If
3831         ' Sets filename variable and updates main page
3832         FluidOutFileName = FluidOutSaveFileDialog.FileName
3833         TB_WetSoilFluidOutFile.Text = FluidOutFileName
3834         TB_WetSoilFluidOutFile.SelectionStart = Len(TB_WetSoilFluidOutFile.Text)
3835     End Sub
3836 #End Region
3837
3838
3839 #Region "SoilOutFile Select"
3840     'Executes when the SoilOutFile browse button is clicked

```

```

3841     Private Sub BUT_BrowseSoilOutFile_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_BrowseSoilOutFile.Click
3842         ' Filters requested files
3843         SoilOutSaveFileDialog.Filter = "csv files (*.csv)|*.csv|All files (*.*)|*.*"
3844         If SoilOutSaveFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
3845             End If
3846         ' Sets filename variable and updates main page
3847         SoilOutFileName = SoilOutSaveFileDialog.FileName
3848         TB_WetSoilSolidOutFile.Text = SoilOutFileName
3849         TB_WetSoilSolidOutFile.SelectionStart = Len(TB_WetSoilSolidOutFile.Text)
3850     End Sub
3851 #End Region
3852
3853
3854 #Region "ThermalLoadOutFile Select"
3855     'Executes when the ThermalLoadOutFile browse button is clicked
3856     Private Sub BUT_BrowseThermalLoadOutFile_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_BrowseThermalLoadOutFile.Click
3857         ' Filters requested files
3858         ThermalLoadOutSaveFileDialog.Filter = "csv files (*.csv)|*.csv|All files (*.*)|*.*"
3859         If ThermalLoadOutSaveFileDialog.ShowDialog() = Windows.Forms.DialogResult.OK Then
3860             End If
3861         ' Sets filename variable and updates main page
3862         ThermalLoadOutFileName = ThermalLoadOutSaveFileDialog.FileName
3863         TB_ThermalLoadOutFile.Text = ThermalLoadOutFileName
3864         TB_ThermalLoadOutFile.SelectionStart = Len(TB_ThermalLoadOutFile.Text)
3865     End Sub
3866 #End Region
3867
3868
3869
3870     'On Graphs button click, open graphs form
3871     Private Sub BUT_Graphs_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles BUT_Graphs.Click
3872         Dim formGraphs As New Graphs

```

```

3873         formGraphs.ShowDialog()
3874     End Sub
3875
3876
3877
3878     'When generate button is clicked, open form to generate new rainfall file
3879     Private Sub BUT_GenerateRain_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BUT_GenerateRain.Click
3880         Dim formRainfall As New Rainfall
3881         formRainfall.ShowDialog()
3882     End Sub
3883
3884
3885     'Activate on-screen help buttons when menu item is clicked
3886     Private Sub OnScreenHelpToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles OnScreenHelpToolStripMenuItem.Click
3887
3888         'Makes help buttons visible
3889         HBUT_RainInputFile.Visible = OnScreenHelpVisible
3890         HBUT_AirTemp.Visible = OnScreenHelpVisible
3891         HBUT_Radiation.Visible = OnScreenHelpVisible
3892         HBUT_WindSpeed.Visible = OnScreenHelpVisible
3893         HBUT_RelHumidity.Visible = OnScreenHelpVisible
3894         HBUT_WeatherInputFile.Visible = OnScreenHelpVisible
3895         HBUT_RunoffInputFile.Visible = OnScreenHelpVisible
3896         HBUT_WatershedArea.Visible = OnScreenHelpVisible
3897         HBUT_InitialAbstraction.Visible = OnScreenHelpVisible
3898         HBUT_BioretenentionArea.Visible = OnScreenHelpVisible
3899         HBUT_Latitude.Visible = OnScreenHelpVisible
3900         HBUT_Longitude.Visible = OnScreenHelpVisible
3901         HBUT_Transmittance.Visible = OnScreenHelpVisible
3902         HBUT_Elevation.Visible = OnScreenHelpVisible
3903         HBUT_Timezone.Visible = OnScreenHelpVisible
3904         HBUT_Shading.Visible = OnScreenHelpVisible
3905         HBUT_DrainDepth.Visible = OnScreenHelpVisible

```

3906 HBUT\_Suction.Visible = OnScreenHelpVisible  
3907 HBUT\_KSat.Visible = OnScreenHelpVisible  
3908 HBUT\_ThetaFilled.Visible = OnScreenHelpVisible  
3909 HBUT\_SurfaceStorage.Visible = OnScreenHelpVisible  
3910 HBUT\_GravelPorosity.Visible = OnScreenHelpVisible  
3911 HBUT\_DrainHeight.Visible = OnScreenHelpVisible  
3912 HBUT\_SubsoilKSat.Visible = OnScreenHelpVisible  
3913 HBUT\_CpSoil.Visible = OnScreenHelpVisible  
3914 HBUT\_SoilPorosity.Visible = OnScreenHelpVisible  
3915 HBUT\_ThermalKSoil.Visible = OnScreenHelpVisible  
3916 HBUT\_InterfaceArea.Visible = OnScreenHelpVisible  
3917 HBUT\_StagThermalK.Visible = OnScreenHelpVisible  
3918 HBUT\_ParticleDia.Visible = OnScreenHelpVisible  
3919 HBUT\_HydraulicOutput.Visible = OnScreenHelpVisible  
3920 HBUT\_DryPaveOutput.Visible = OnScreenHelpVisible  
3921 HBUT\_WetPaveOutput.Visible = OnScreenHelpVisible  
3922 HBUT\_FluidTempOutput.Visible = OnScreenHelpVisible  
3923 HBUT\_SoilTempOutput.Visible = OnScreenHelpVisible  
3924 HBUT\_ThermalLoadOutput.Visible = OnScreenHelpVisible  
3925 HBUT\_RunoffEnergy.Visible = OnScreenHelpVisible  
3926 HBUT\_EffluentEnergy.Visible = OnScreenHelpVisible  
3927 HBUT\_OverflowEnergy.Visible = OnScreenHelpVisible  
3928 HBUT\_UnderdrainEnergy.Visible = OnScreenHelpVisible  
3929 HBUT\_SeepageEnergy.Visible = OnScreenHelpVisible  
3930 HBUT\_ThermalLoadReduction.Visible = OnScreenHelpVisible  
3931 HBUT\_VolReduction.Visible = OnScreenHelpVisible  
3932 HBUT\_RunoffMax.Visible = OnScreenHelpVisible  
3933 HBUT\_EffluentMax.Visible = OnScreenHelpVisible  
3934 HBUT\_OverflowMax.Visible = OnScreenHelpVisible  
3935 HBUT\_UnderdrainMax.Visible = OnScreenHelpVisible  
3936 HBUT\_SeepageMax.Visible = OnScreenHelpVisible  
3937 HBUT\_RunoffAVG.Visible = OnScreenHelpVisible  
3938 HBUT\_EffluentAVG.Visible = OnScreenHelpVisible  
3939 HBUT\_OverflowAVG.Visible = OnScreenHelpVisible  
3940 HBUT\_UnderdrainAVG.Visible = OnScreenHelpVisible

```
3941         HBUT_SeepageAVG.Visible = OnScreenHelpVisible
3942
3943
3944         'Set onscreenhelpvisible so that next time menu item is clicked, buttons visibility
changes
3945         If OnScreenHelpVisible = True Then
3946             OnScreenHelpVisible = False
3947             'Update label for menu item
3948             OnScreenHelpToolStripMenuItem.Text = "Hide On-Screen Help"
3949         Else
3950             OnScreenHelpVisible = True
3951             'Update label for menu item
3952             OnScreenHelpToolStripMenuItem.Text = "Show On-Screen Help"
3953         End If
3954
3955     End Sub
3956
3957
3958
3959
3960
3961
3962 End Class
```

## **APPENDIX C**

### **Bioretention Thermal Model: Sample Save File and Rainfall Input File**

### **Sample Save File Contents**

```
<?xml version="1.0" encoding="utf-8"?>
<SavedData>
  <RainFileName>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\Validation\Brevard West\BrevardWest_07-25-06.csv</RainFileName>
  <WeatherFileName>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\Brevard-Weather.xml</WeatherFileName>
  <HydraulicOutFile>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\GA-Output.csv</HydraulicOutFile>
  <DryPaveOutFile>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\DryPave-Output.csv</DryPaveOutFile>
  <WetPaveOutFile>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\WetPave-Output.csv</WetPaveOutFile>
  <WetSoilFluidOutFile>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\WetSoil-Fluid-Output.csv</WetSoilFluidOutFile>
  <WetSoilSolidOutFile>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\WetSoil-Soil-Output.csv</WetSoilSolidOutFile>
  <ThermalLoadOutFile>C:\Documents and Settings\Administrator\Desktop\Model
Inputs\ThermalLoad-Output.csv</ThermalLoadOutFile>
  <WatershedArea>325</WatershedArea>
  <InitialAbstraction>0.0005</InitialAbstraction>
  <BioretentionArea>36</BioretentionArea>
  <SoilDepth>0.43</SoilDepth>
  <Suction>0.13845</Suction>
  <KSat>0.179</KSat>
  <ThetaFilled>0.238</ThetaFilled>
  <StorageCapacity>0.23</StorageCapacity>
  <GravelPorosity>0.3</GravelPorosity>
  <DrainHeight>0.075</DrainHeight>
```

<SubSoilKSat>0.06</SubSoilKSat>  
<LatDeg>35.239</LatDeg>  
<LonDeg>82.731</LonDeg>  
<Transmittance>0.6</Transmittance>  
<Elevation>656</Elevation>  
<Timezone>-5</Timezone>  
<Shading>0.43</Shading>  
<AirTemperature>24.23</AirTemperature>  
<WindSpeed>4</WindSpeed>  
<Radiation>78</Radiation>  
<RelHumidity>0.7</RelHumidity>  
<Cp\_S>0.85</Cp\_S>  
<Porosity>0.516</Porosity>  
<ThermalK\_S>3.0</ThermalK\_S>  
<InterfaceArea>25</InterfaceArea>  
<ThermalK\_Stg>1.18</ThermalK\_Stg>  
<ParticleDia>0.0005</ParticleDia>  
</SavedData>

### **Sample Rainfall Input File Contents**

6/14/2007 18:10, 0  
6/14/2007 18:15, 0.01  
6/14/2007 18:25, 0.01  
6/14/2007 18:35, 0.01  
6/14/2007 18:40, 0.01  
6/14/2007 18:45, 0.01  
6/14/2007 18:50, 0.01  
6/14/2007 18:55, 0.01  
6/14/2007 19:00, 0.07  
6/14/2007 19:05, 0.22

6/14/2007 19:10, 0.32  
6/14/2007 19:15, 0.15  
6/14/2007 19:20, 0.03  
6/14/2007 19:25, 0.01  
6/14/2007 19:35, 0.01  
6/14/2007 19:40, 0.01  
6/14/2007 19:45, 0.02  
6/14/2007 19:50, 0.01  
6/14/2007 19:55, 0.01  
6/14/2007 20:00, 0.01  
6/14/2007 20:05, 0.01  
6/14/2007 20:10, 0.01  
6/14/2007 20:25, 0.01  
6/14/2007 20:40, 0.01  
6/14/2007 20:45, 0.01  
6/14/2007 20:50, 0.01  
6/14/2007 20:55, 0.01  
6/14/2007 21:00, 0.01  
6/14/2007 21:05, 0.01  
6/14/2007 21:10, 0.01  
6/14/2007 21:15, 0.01  
6/14/2007 21:20, 0.02  
6/14/2007 21:25, 0.02  
6/14/2007 21:40, 0.01

## **APPENDIX D**

### **Bioretention Thermal Model: User Interface Screenshots**

**Bioretention**

File Help

Site Description Bioretention Design Output Files Simulation Results

**Storm Weather**

Rainfall Input File

Air Temperature (°C)

Wind Speed (m/s)

Radiation (W/m²)

Relative Humidity

Annual Weather Input File

Runoff Temperature Input File

**Site Description**

Watershed Area (m²)

Initial Abstraction (m)

Bioretention Area (m²)

Latitude (Decimal Degrees)

Longitude (Decimal Degrees)

Transmittance

Elevation (m)

Timezone

Shading 

Status: Simulation complete

Help: Portion of the bioretention surface covered by vegetation

Loaded File: F:\Model Inputs\Brevard\West-Sensitivity.xml

**Bioretention**

File Help

Site Description Bioretention Design Output Files Simulation Results

**Hydraulic Properties**

Drain Depth (m)

Suction (m)

K Sat (m/hr)

Theta Filled

Surface Storage Capacity (m)

Gravel Porosity

Drain Height (m)

Sub-Soil K Sat (m/hr)

**Thermal Properties**

Cp Soil (kJ/kg K)

Soil Porosity

Thermal K Soil (W/m K)

Interface Area (m²/m²)

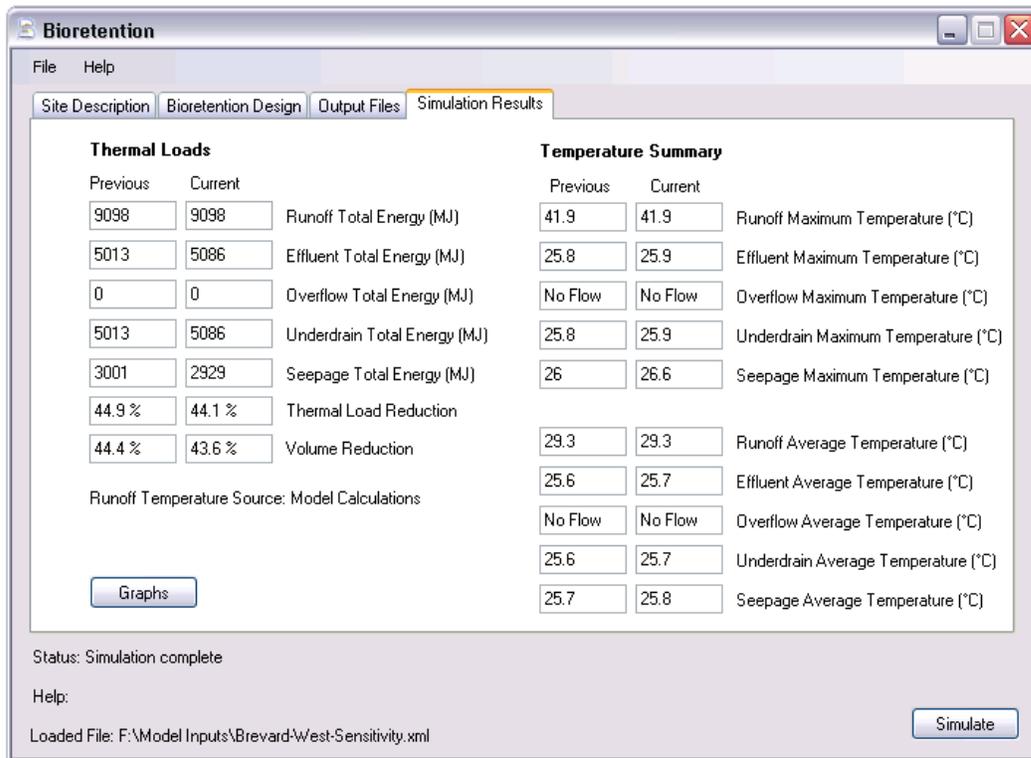
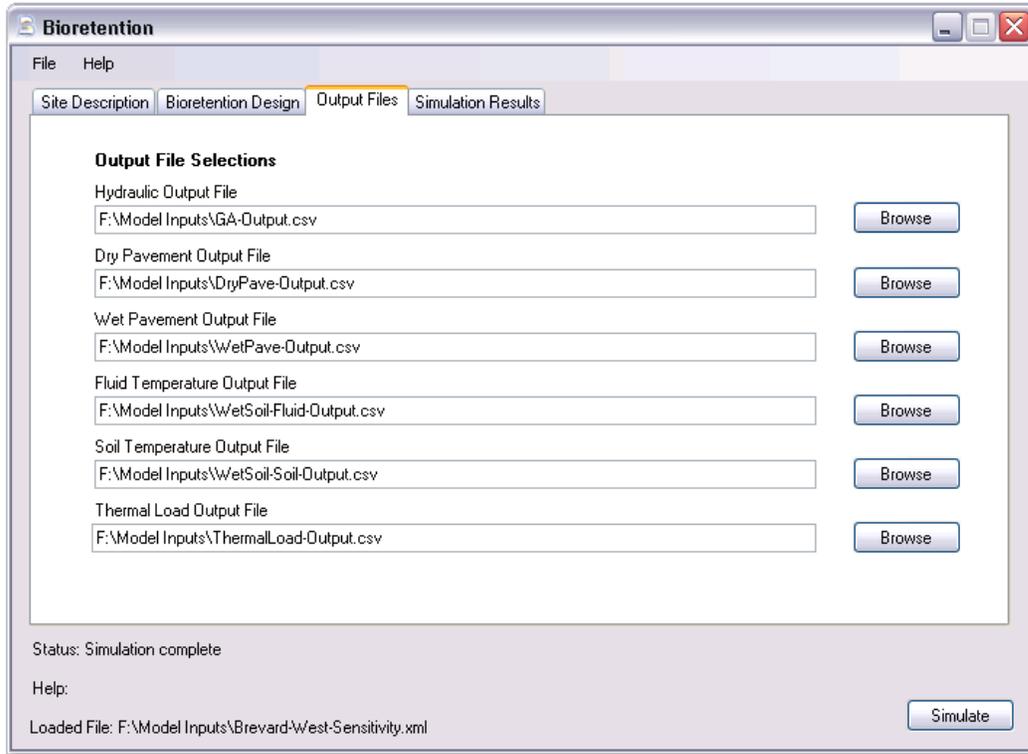
Stagnant Thermal K (W/m K)

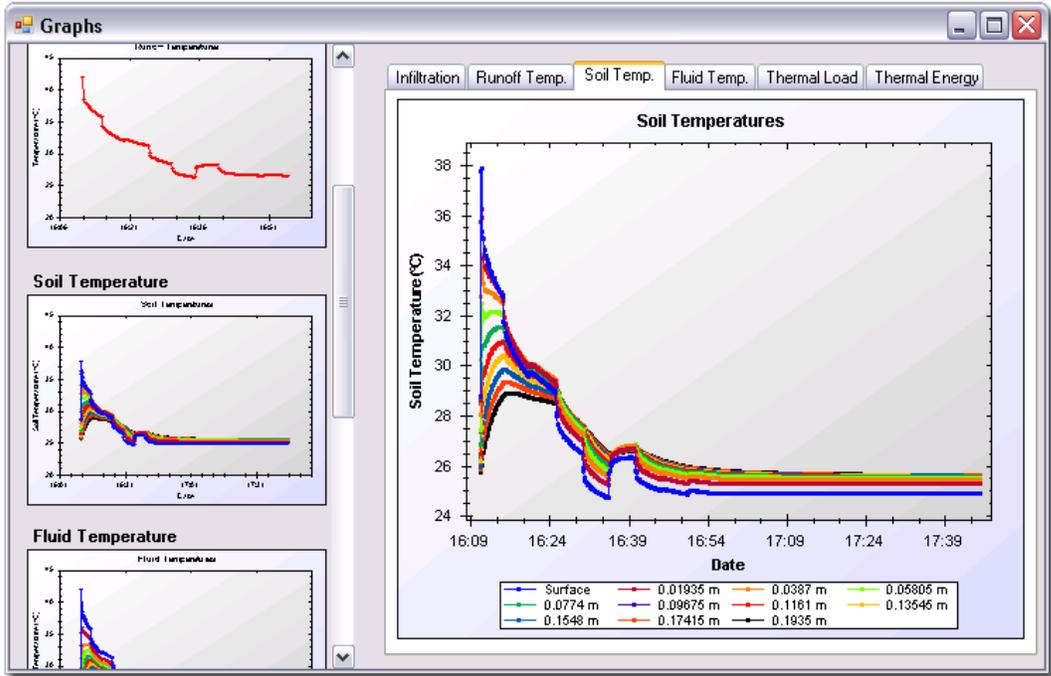
Particle Diameter (m)

Status: Simulation complete

Help:

Loaded File: F:\Model Inputs\Brevard\West-Sensitivity.xml

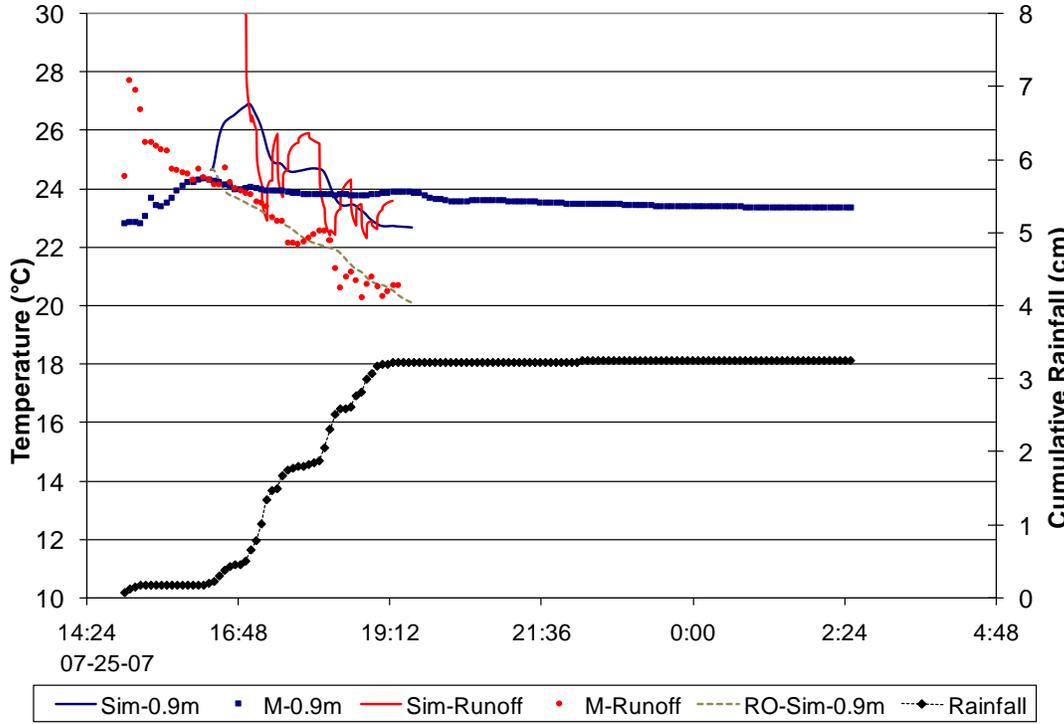
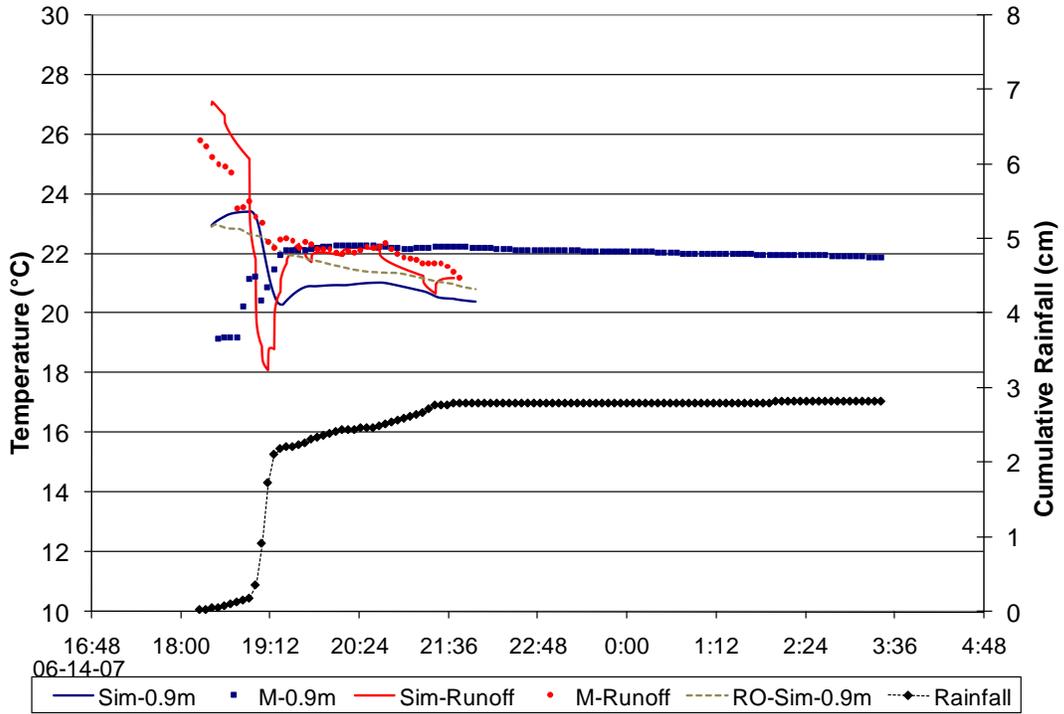


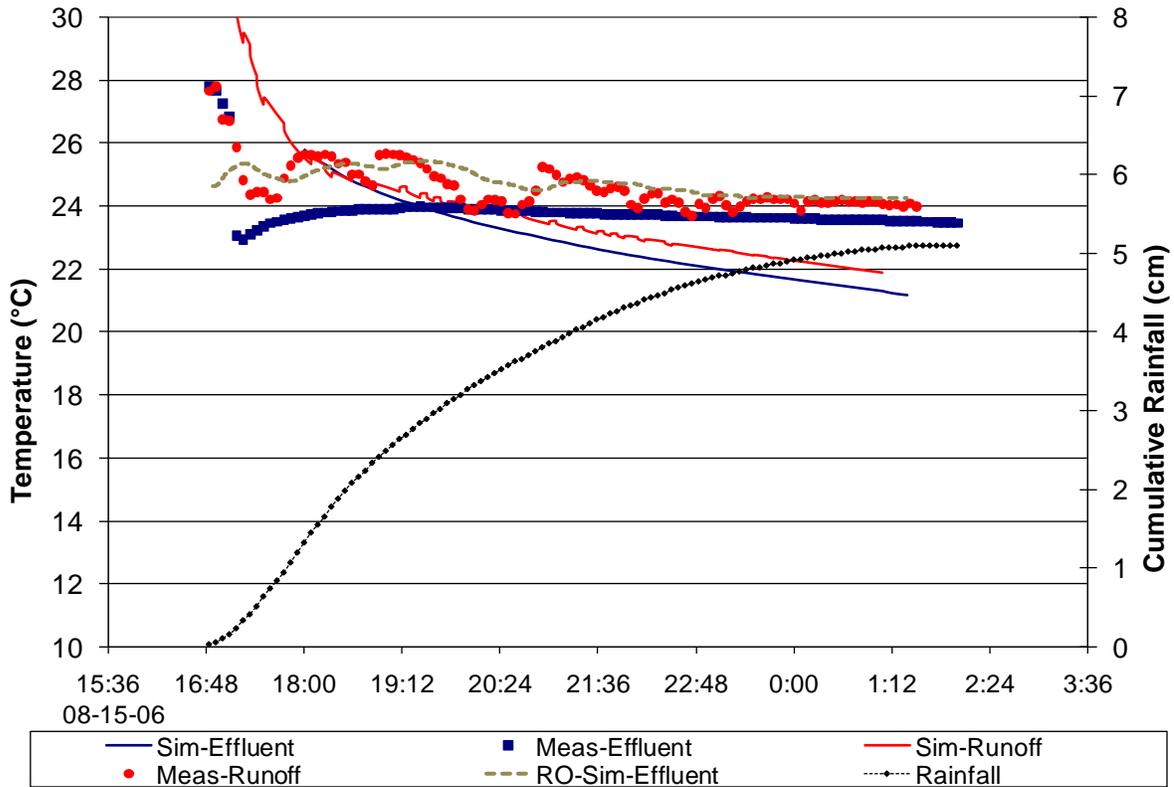
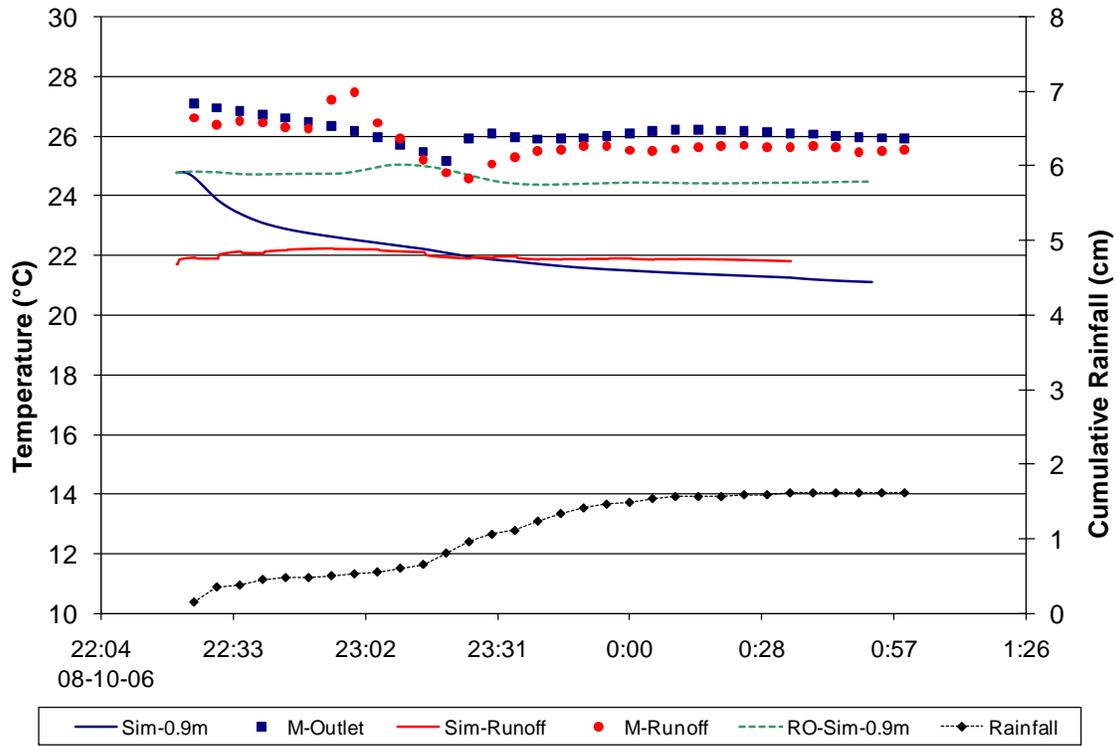


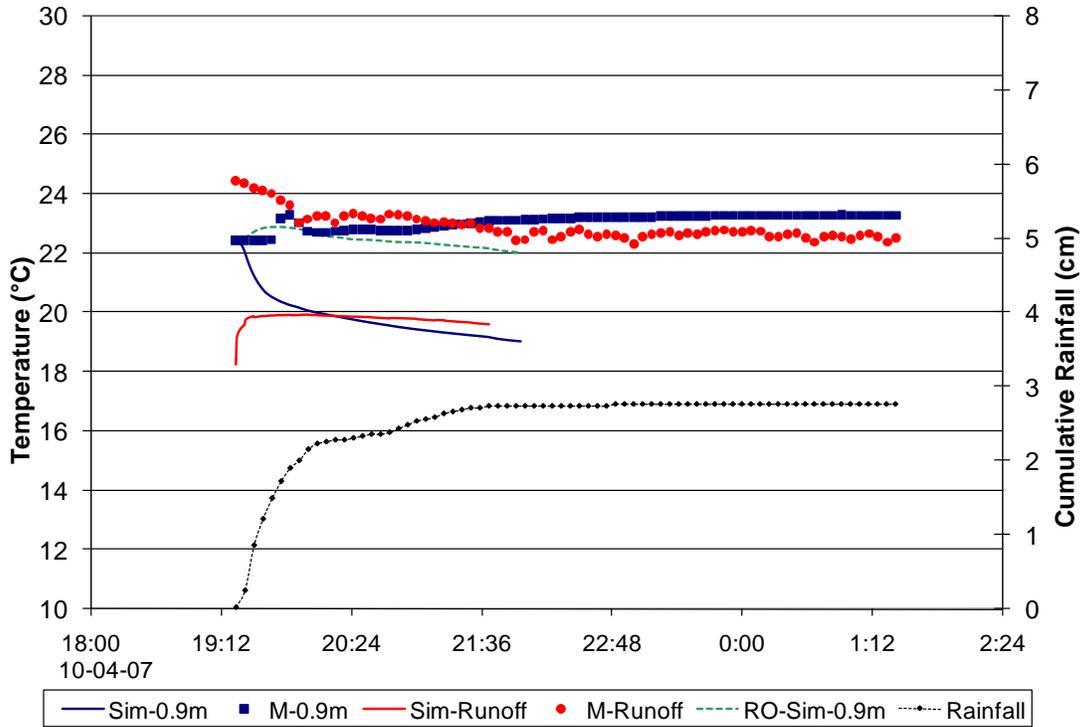
## **APPENDIX E**

### **Bioretention Thermal Model: Validation and Sensitivity Analysis Plots**

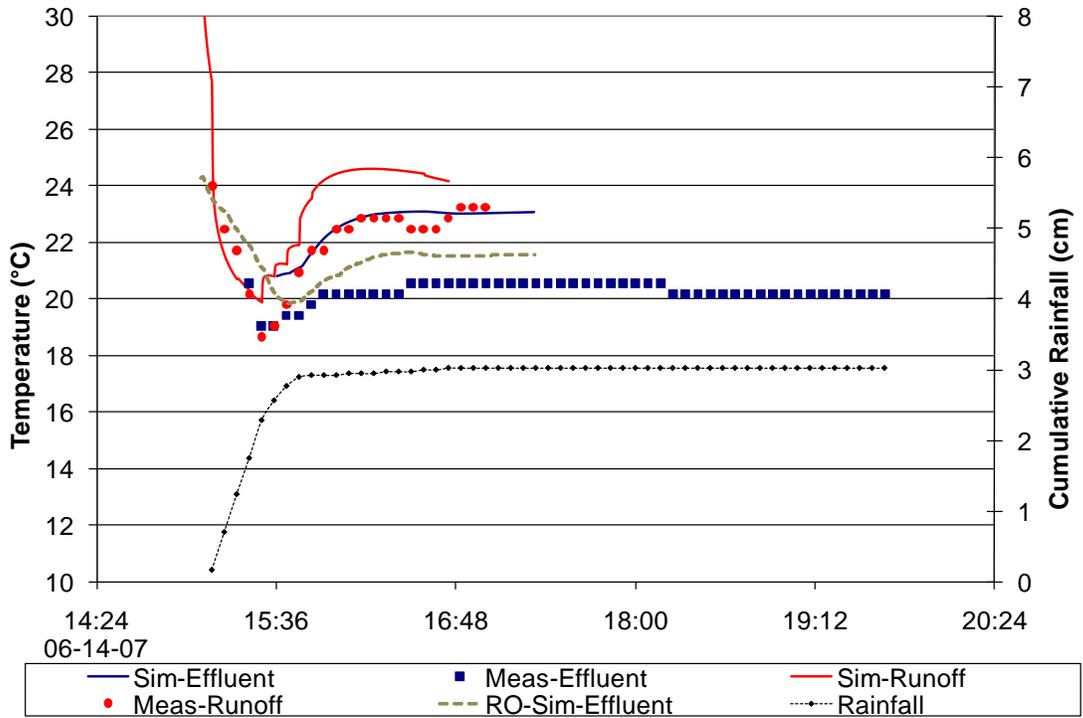
## Lenoir Validation

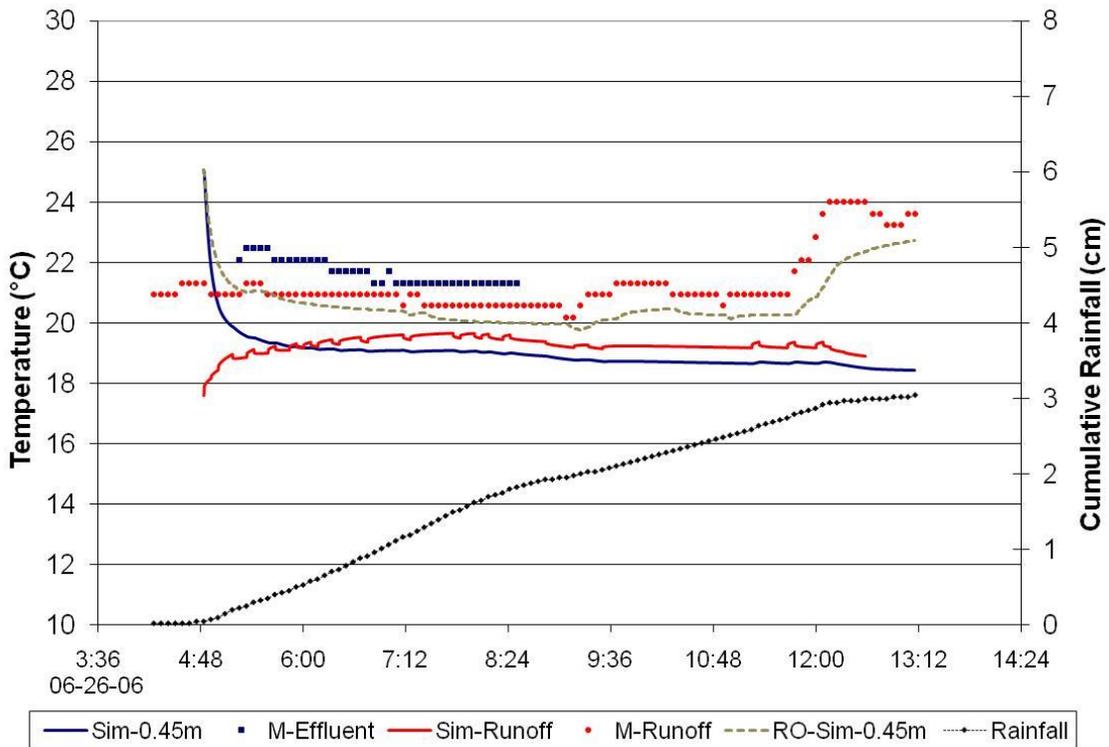
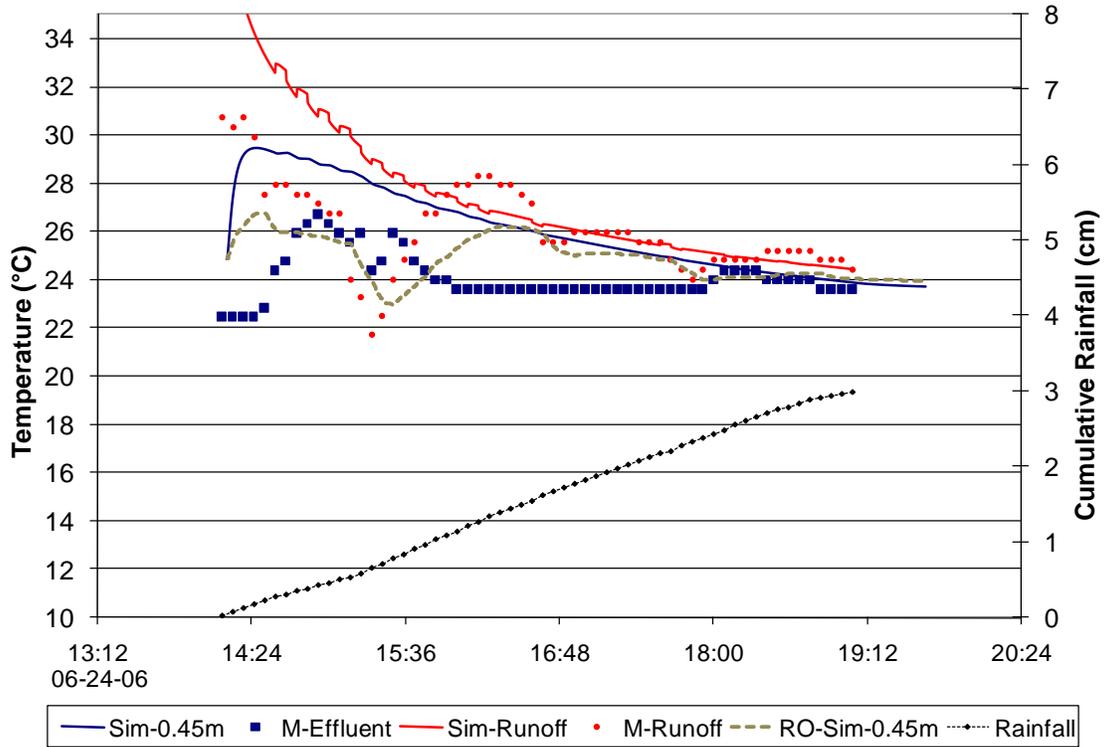


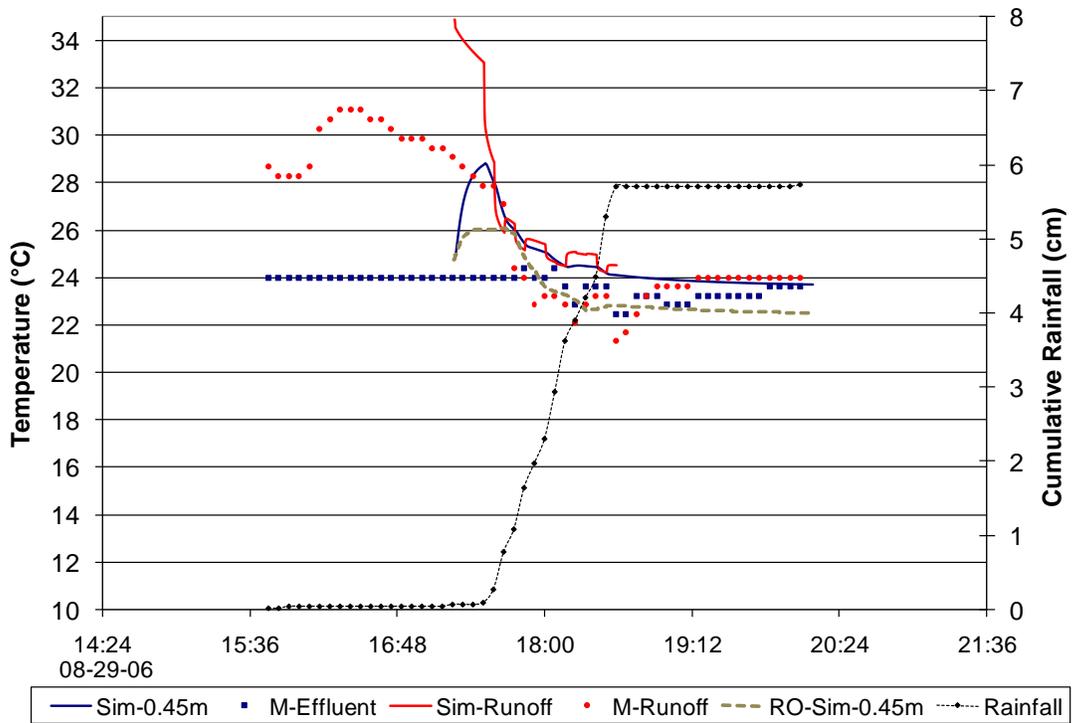
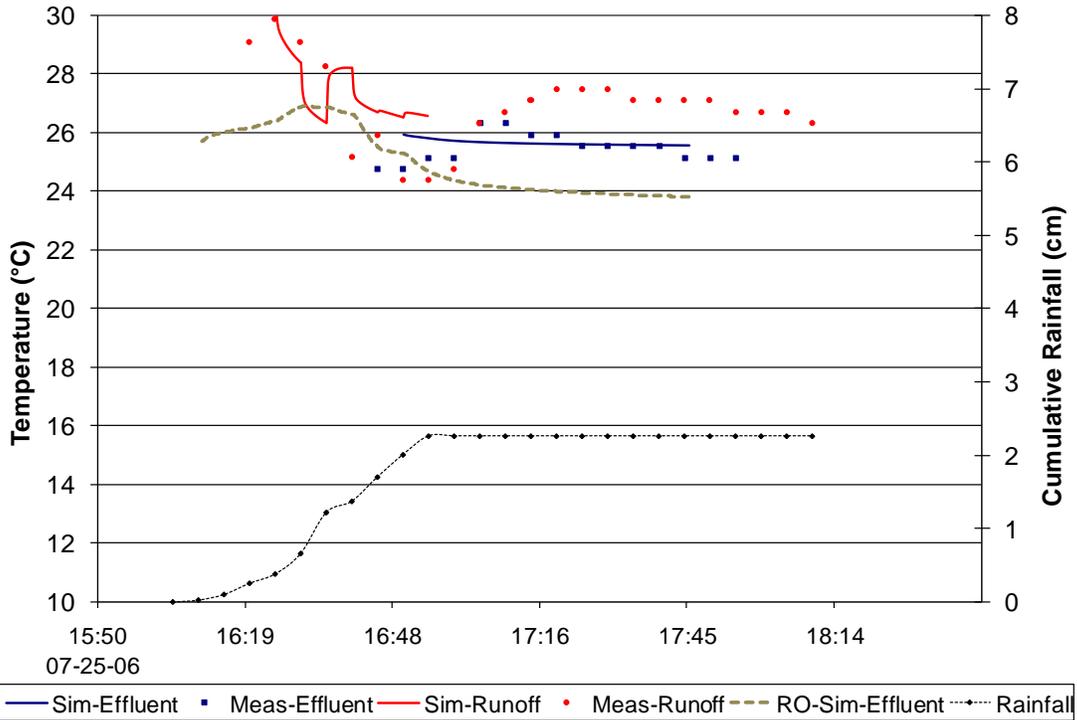


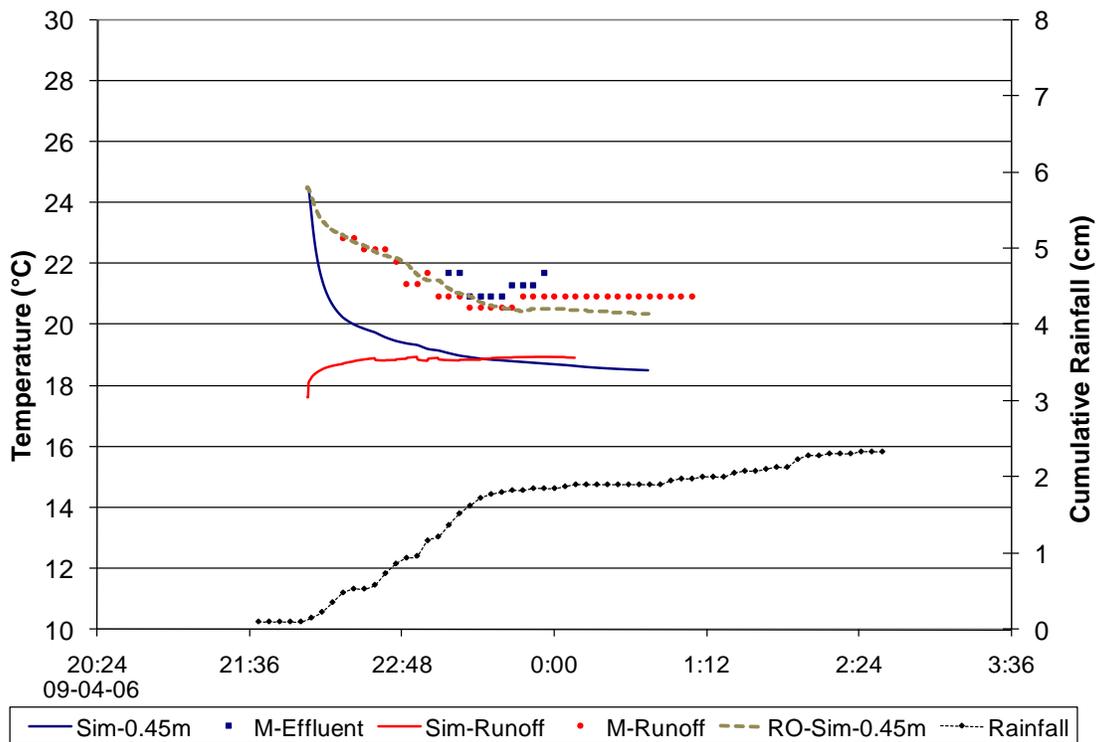
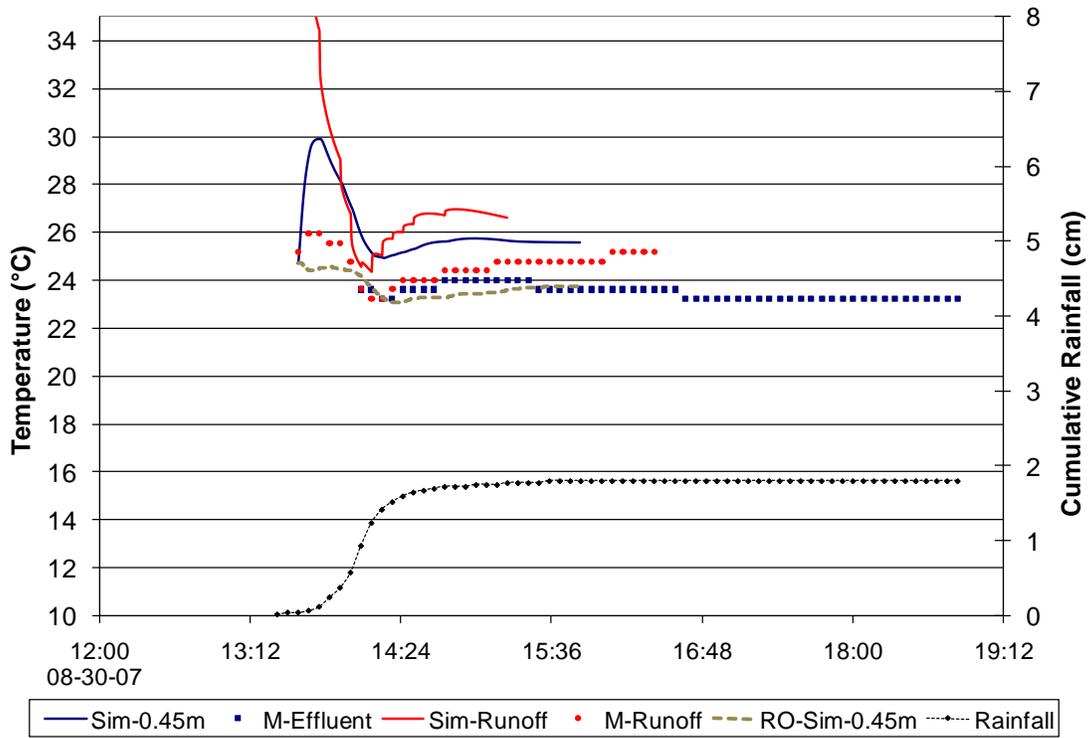


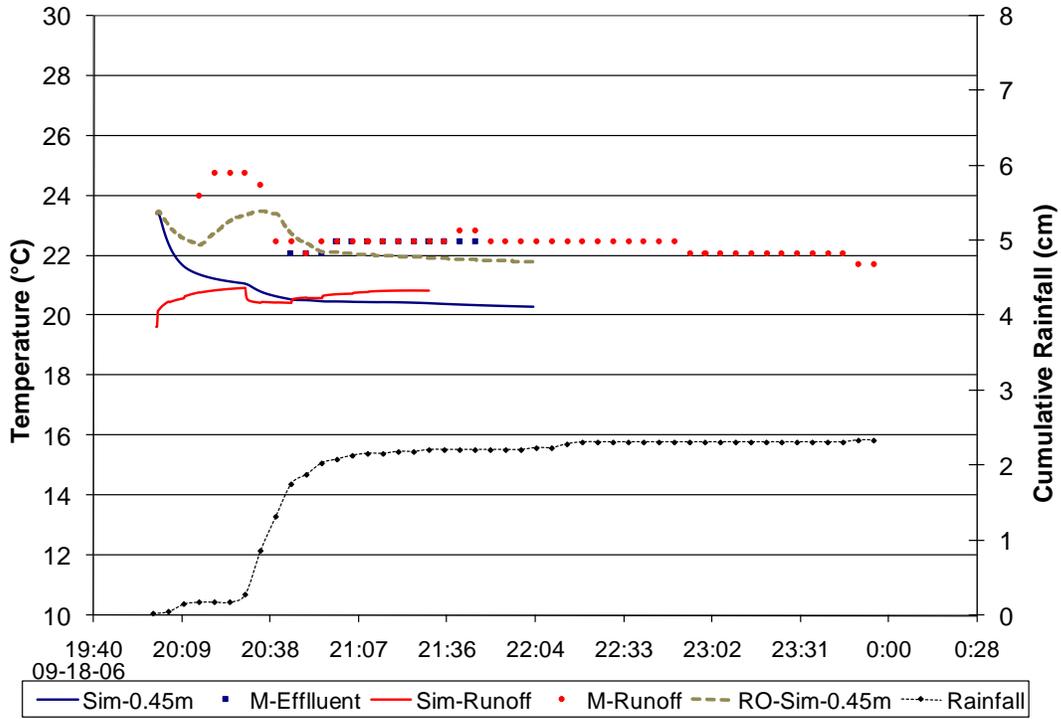
**Brevard West Validation**



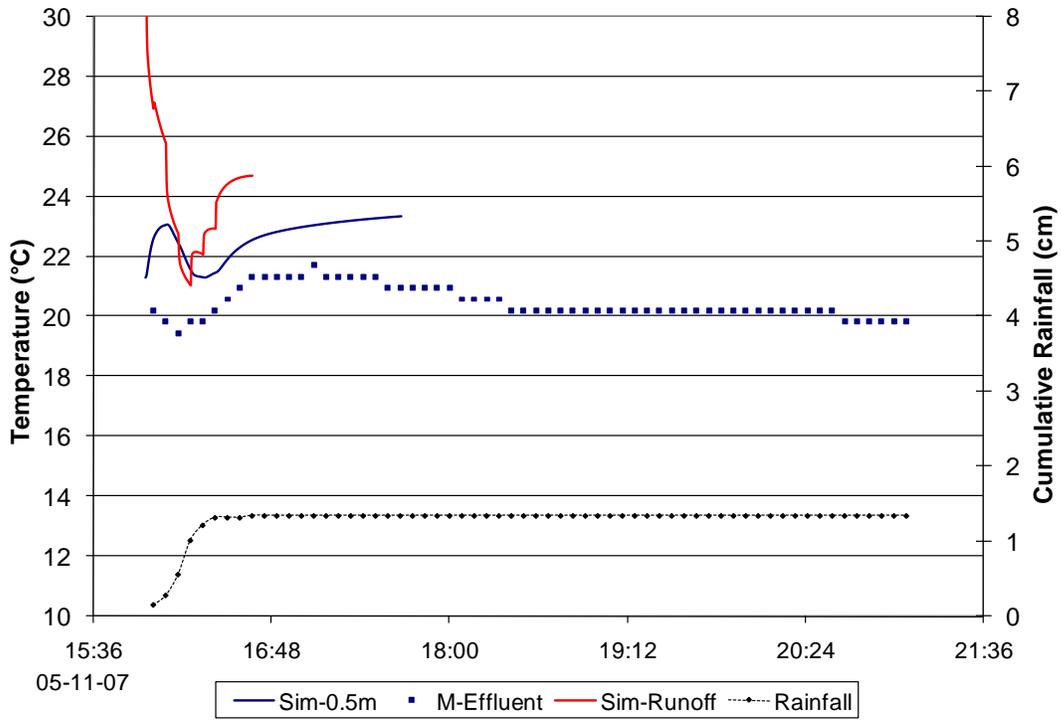


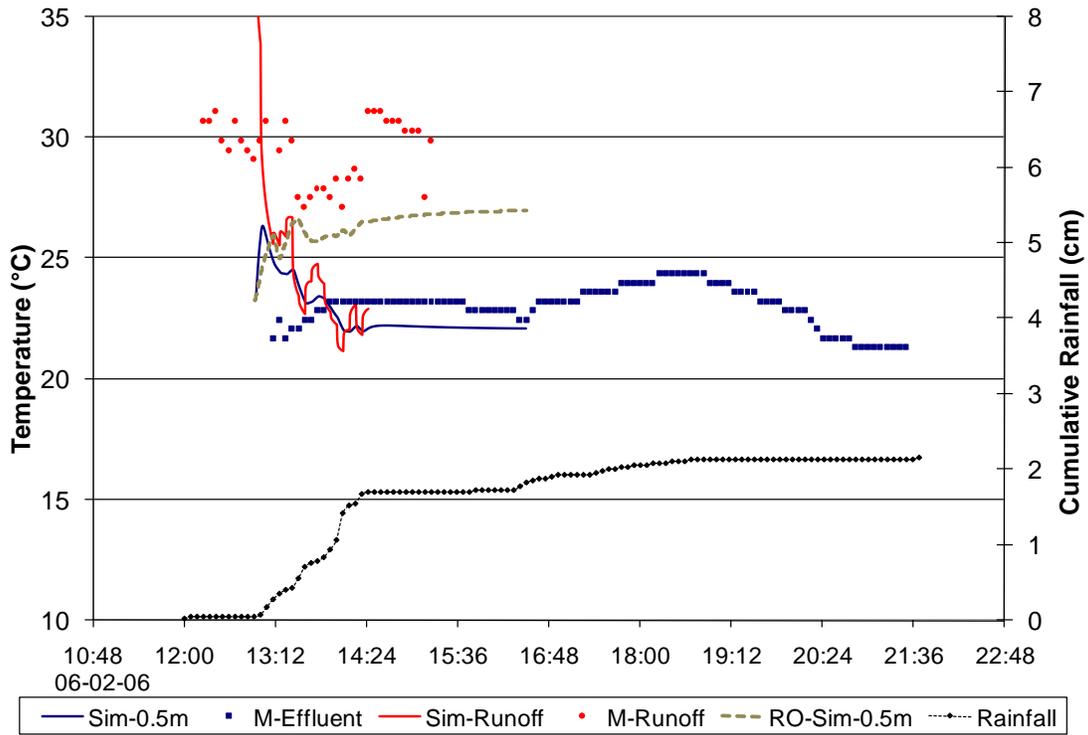
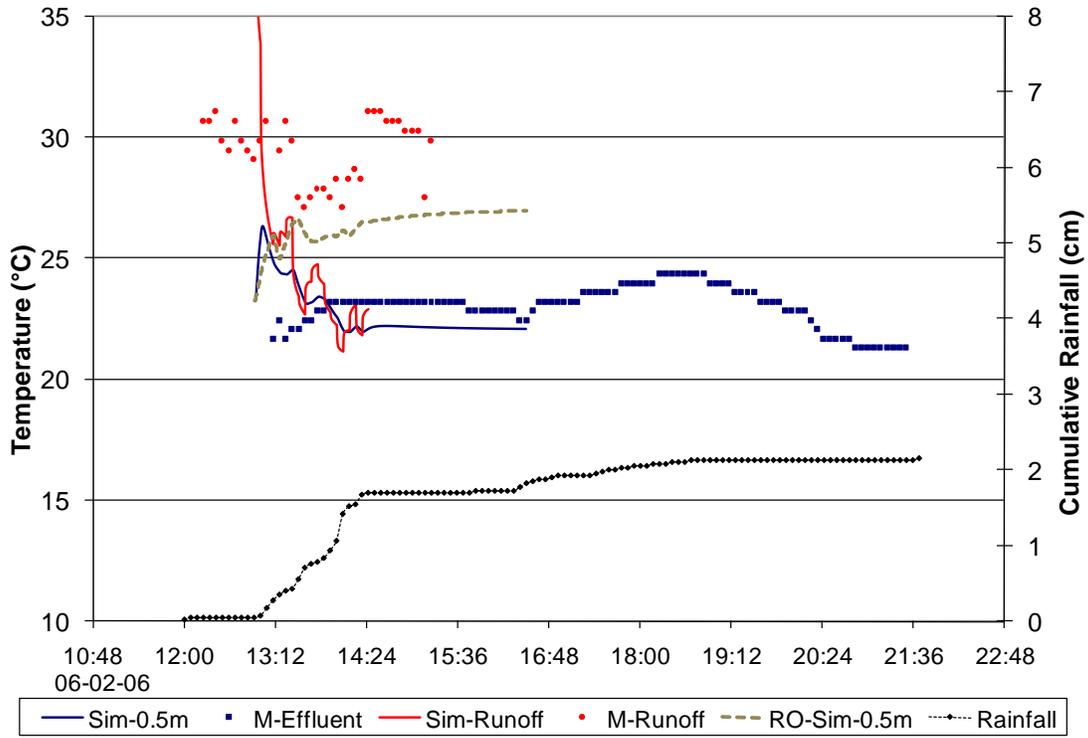


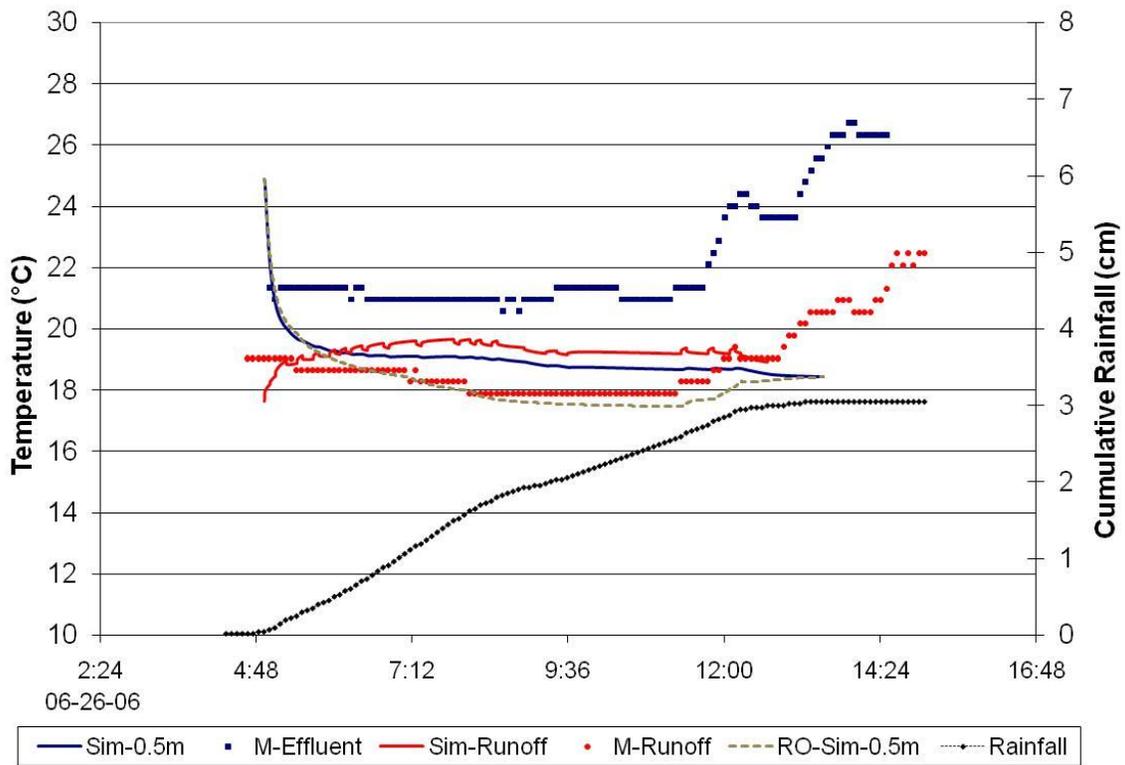
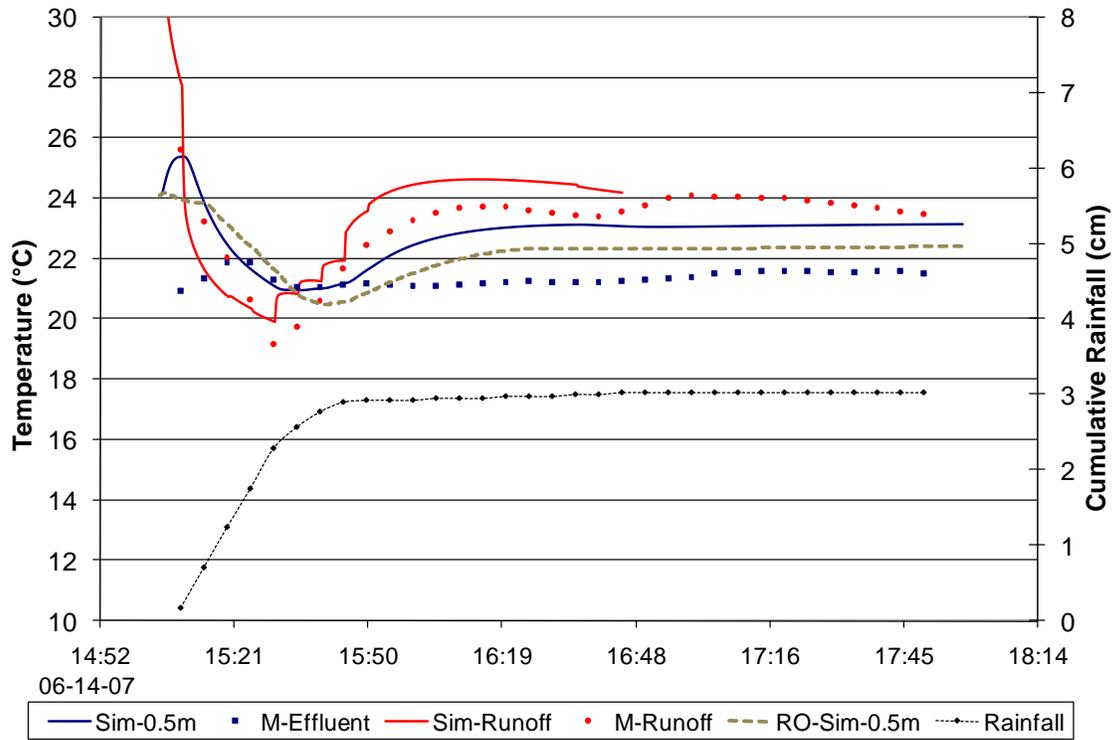


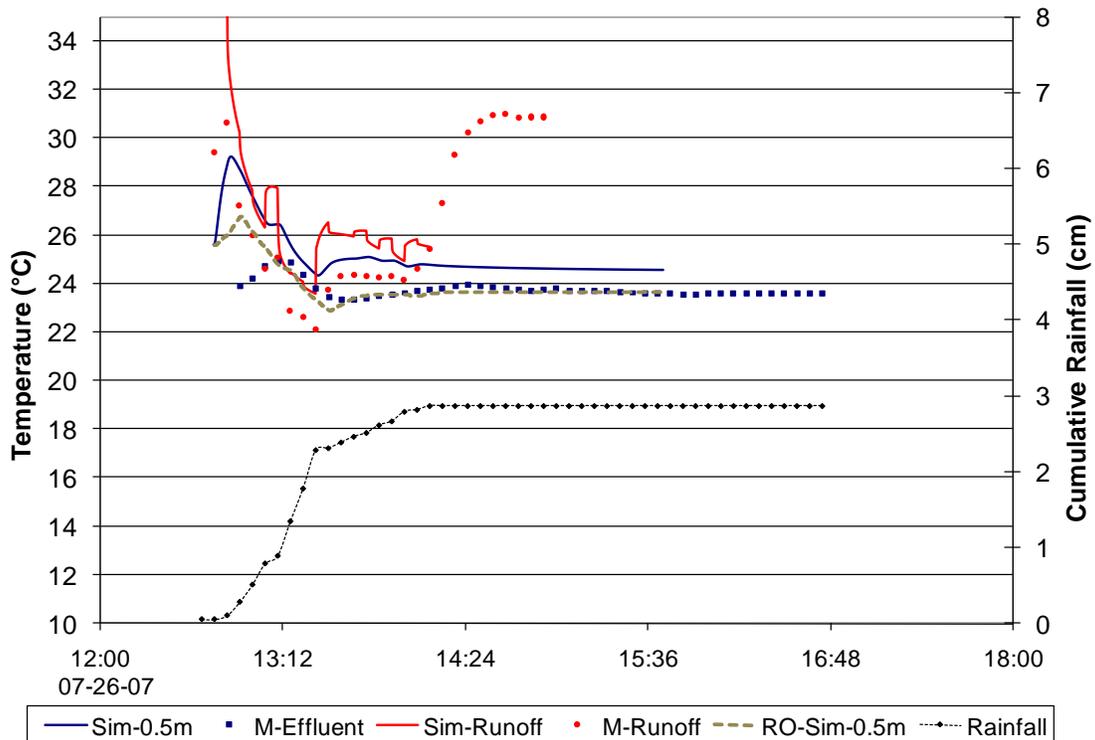
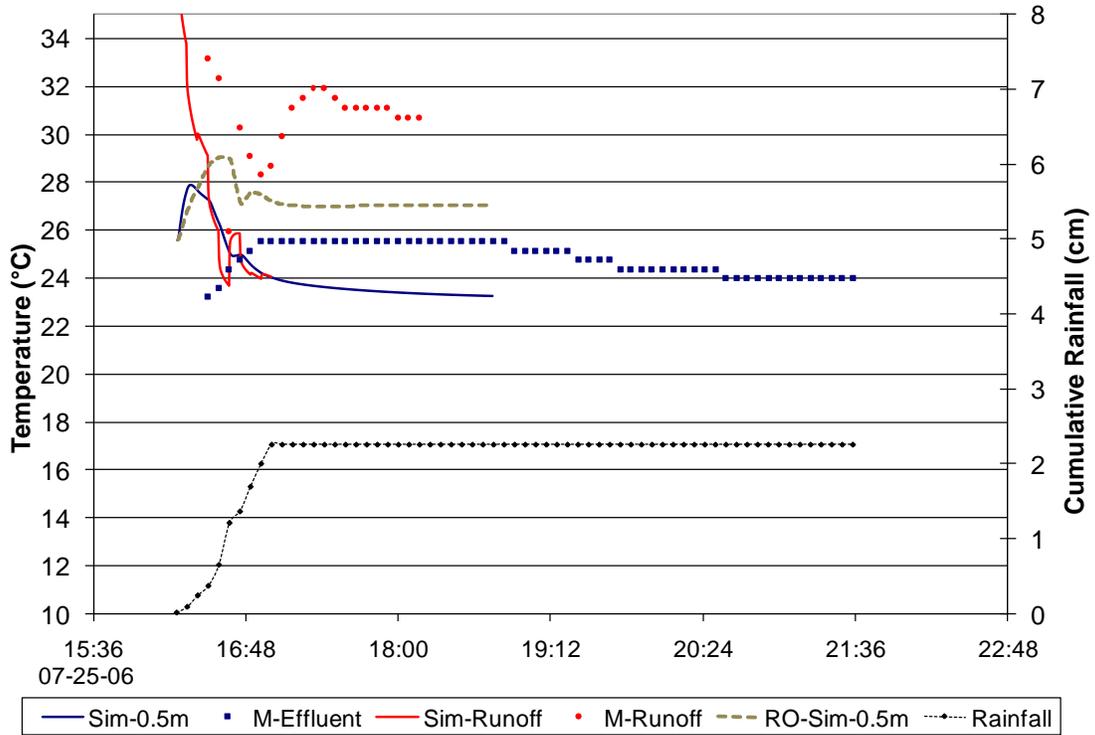


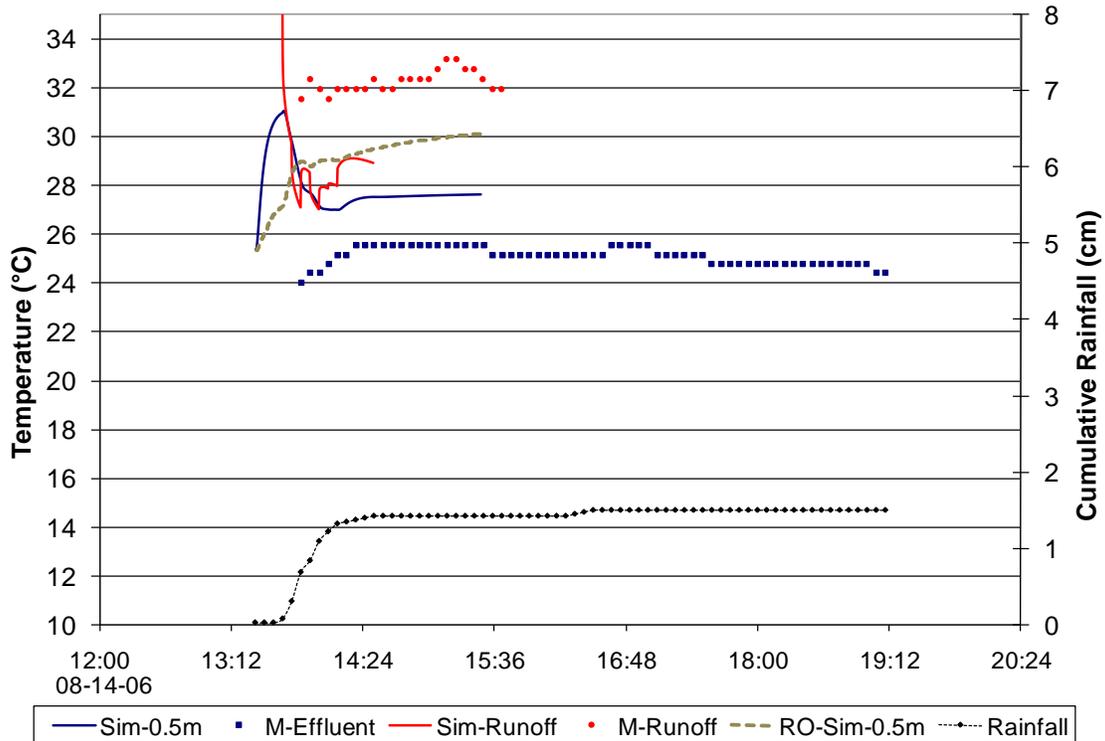
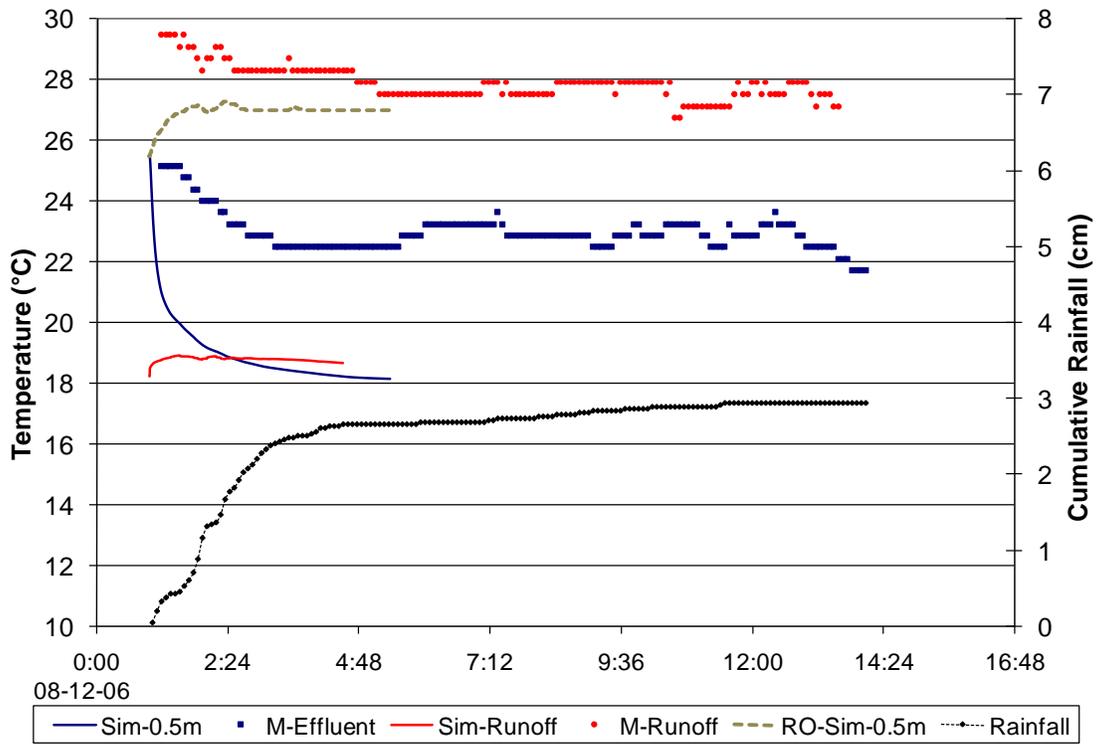
**Brevard East Validation**

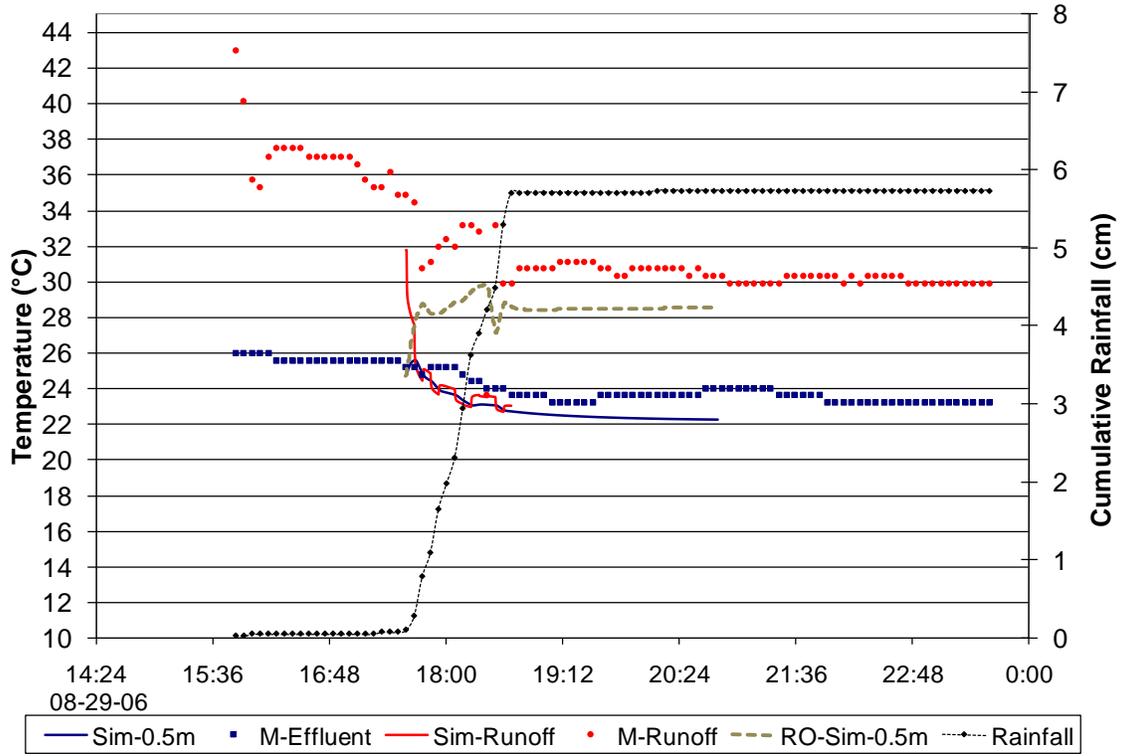
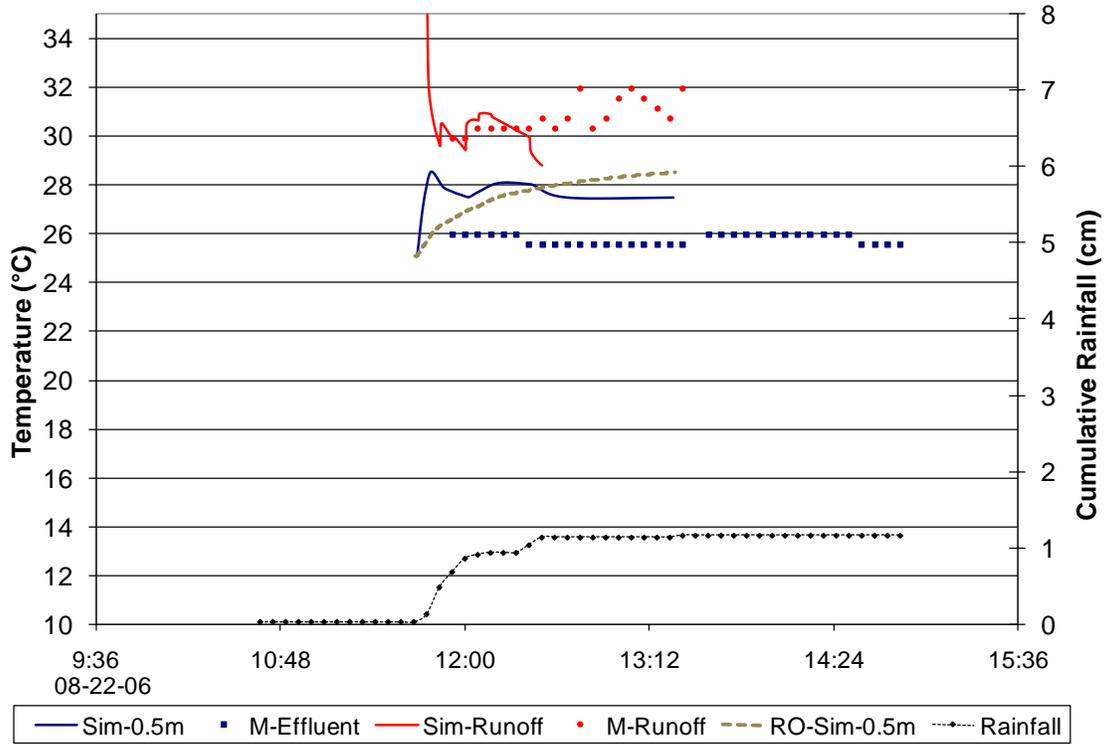


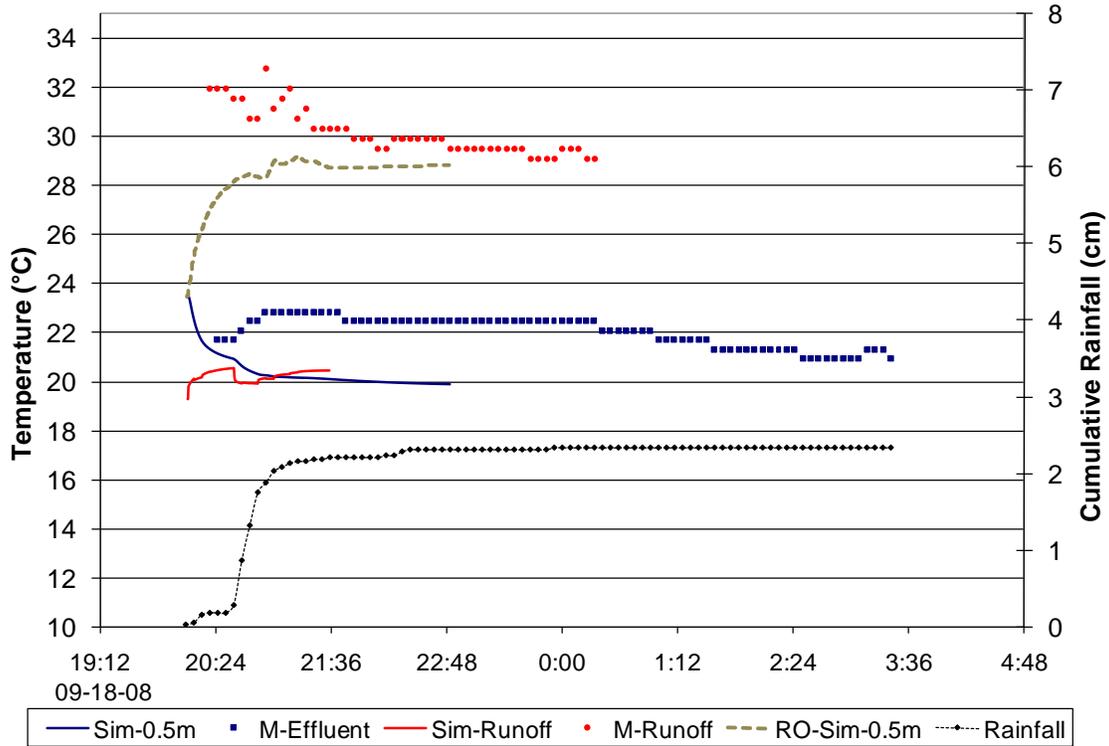
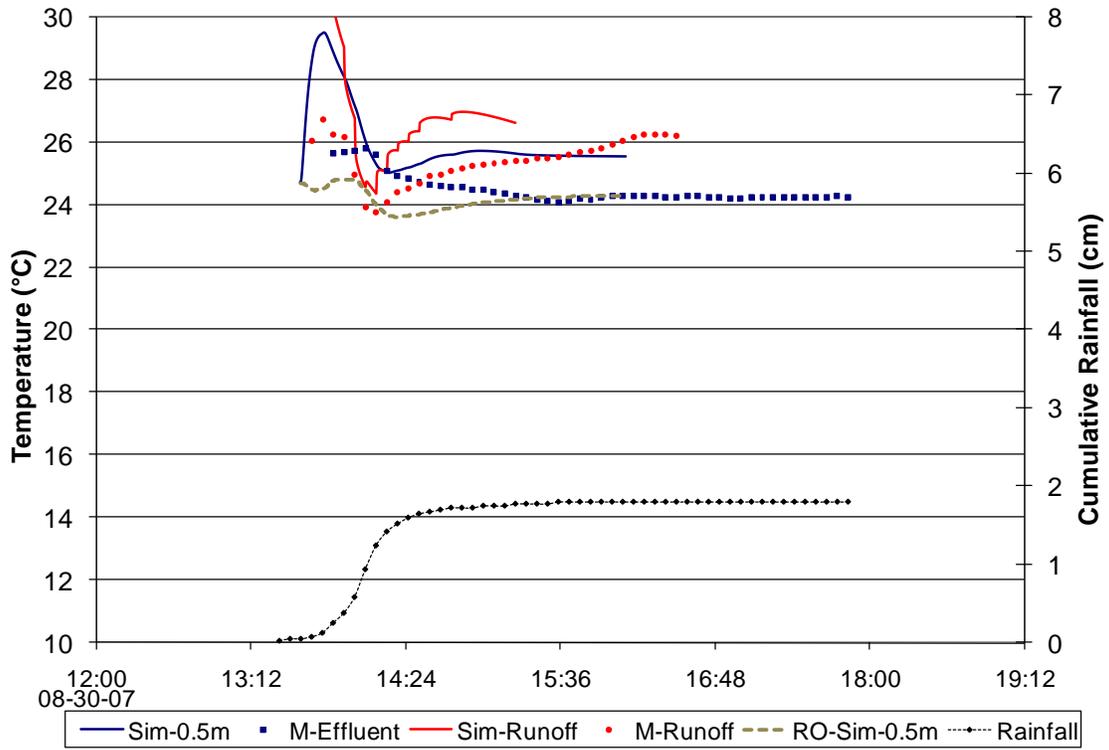




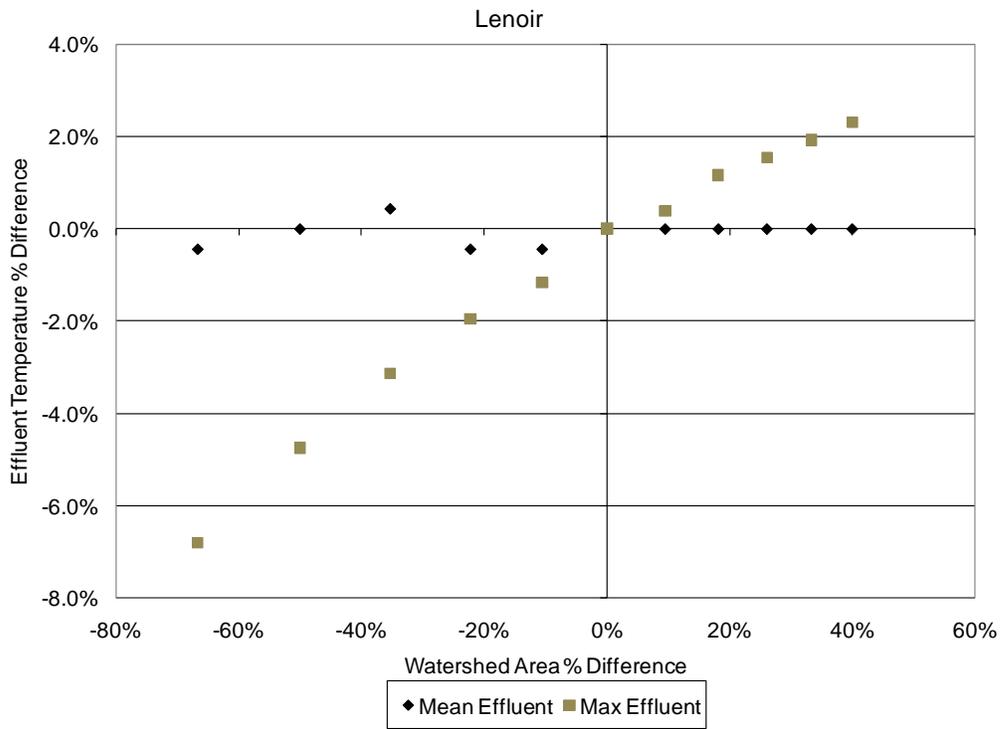
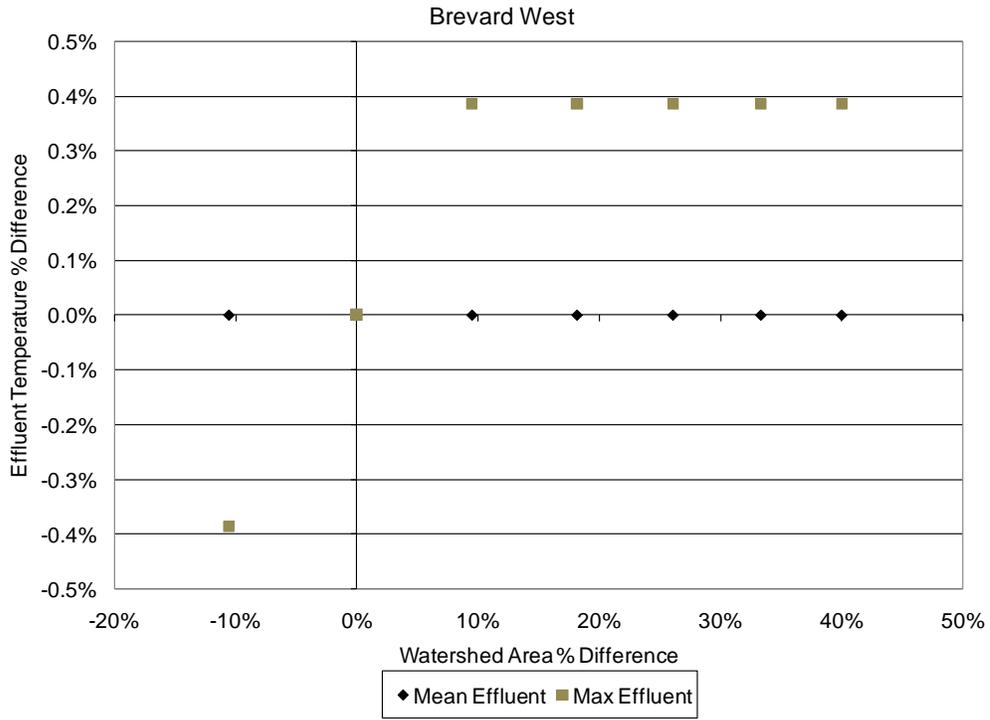


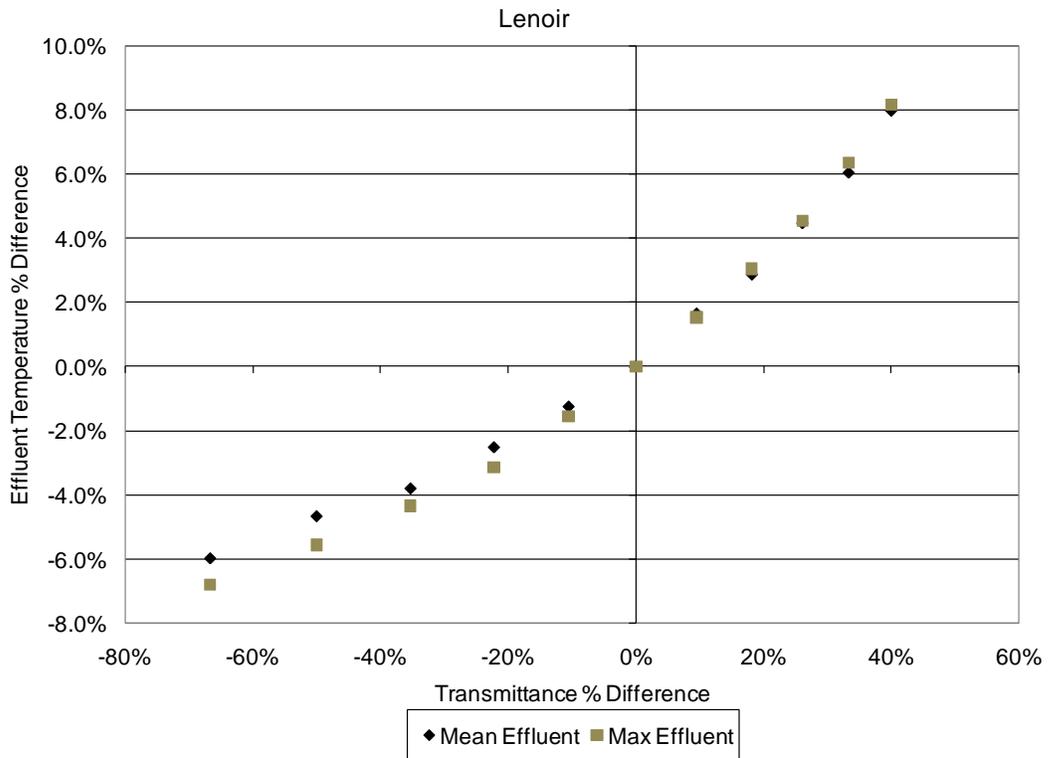
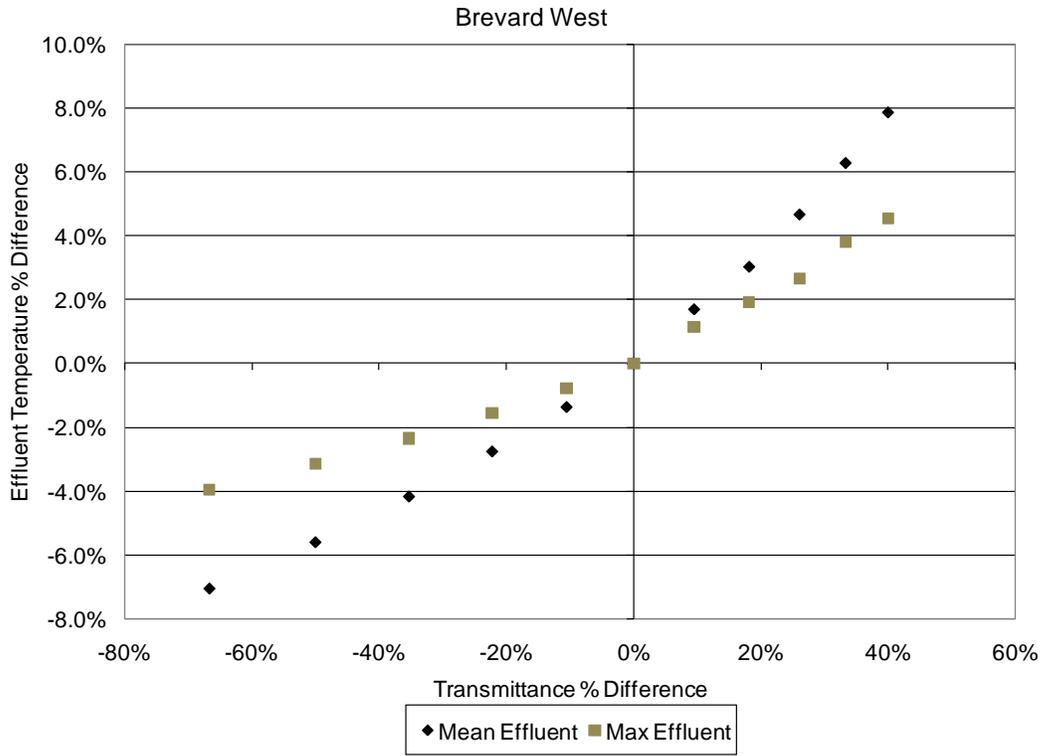


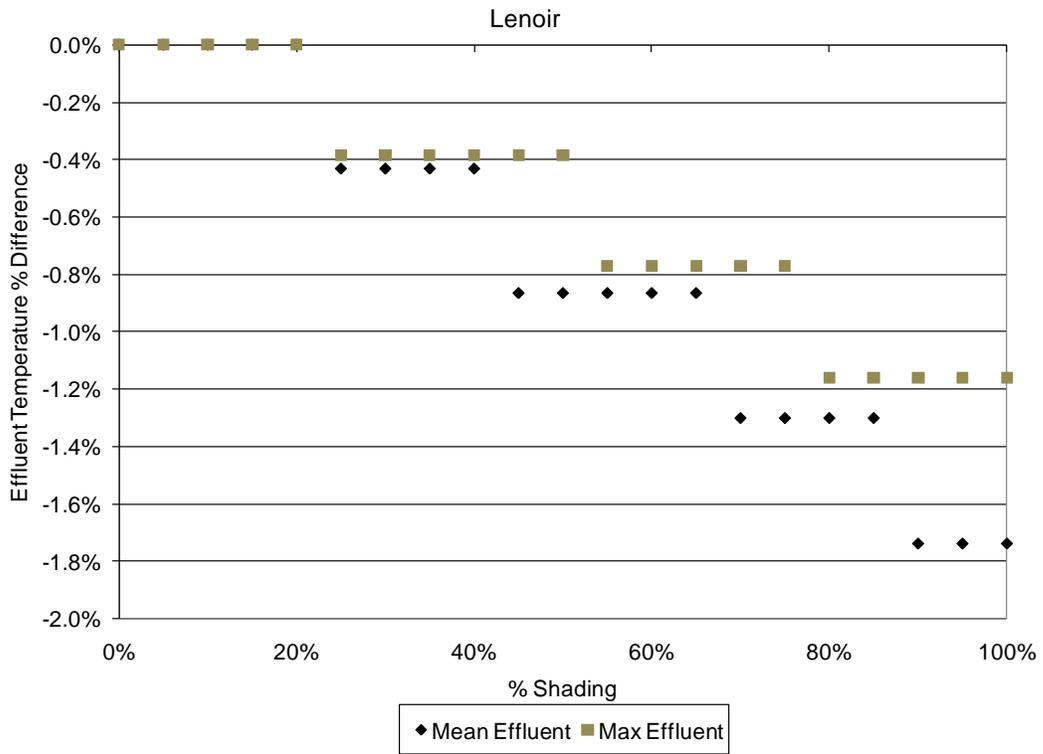
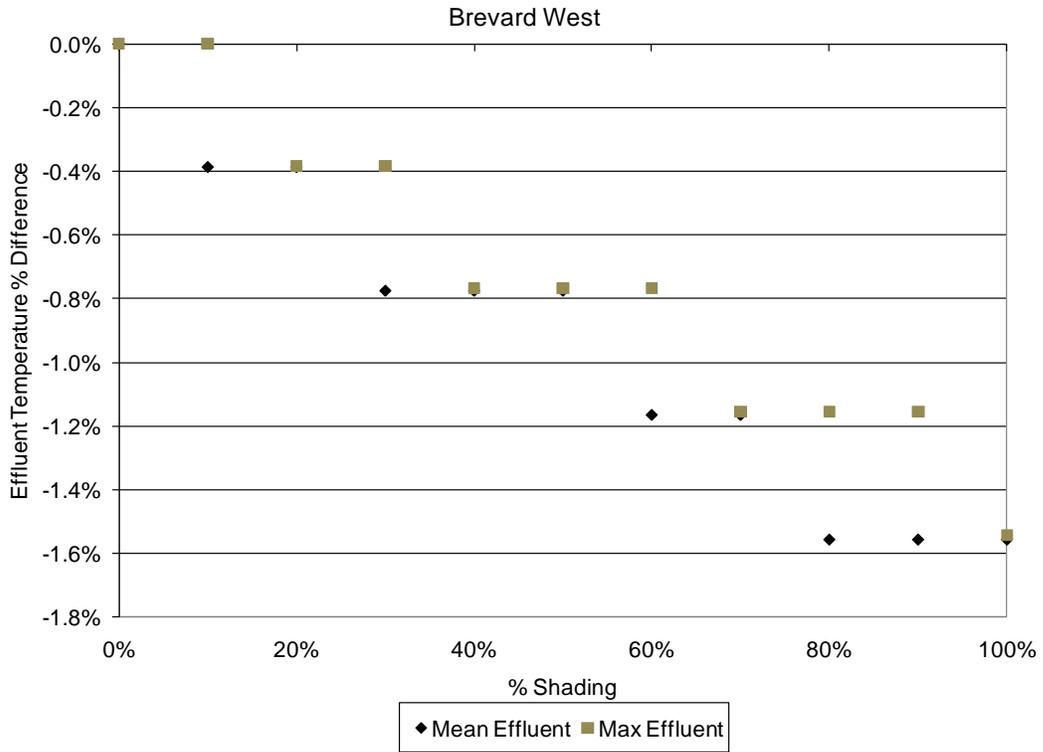


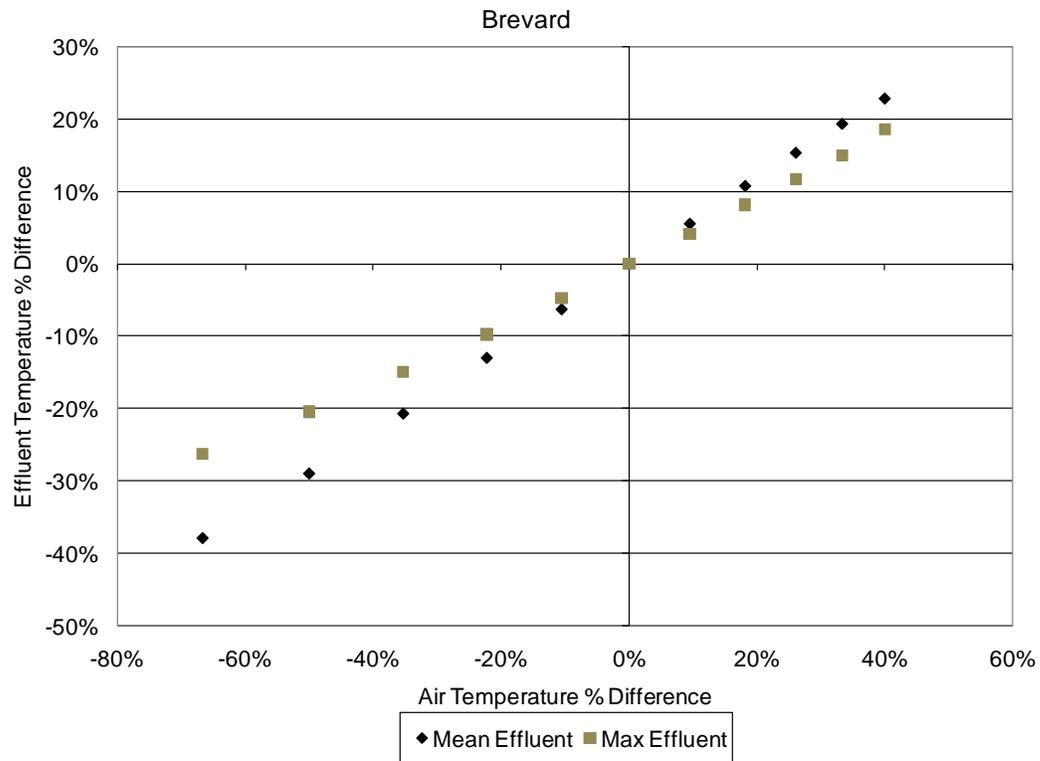
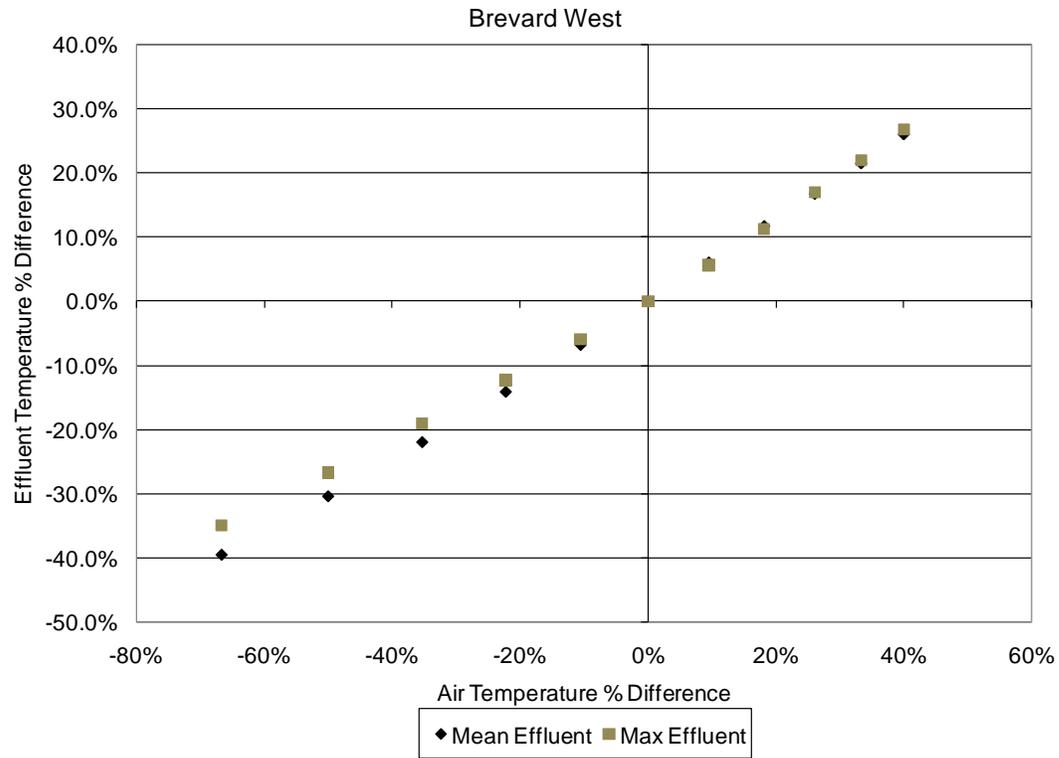


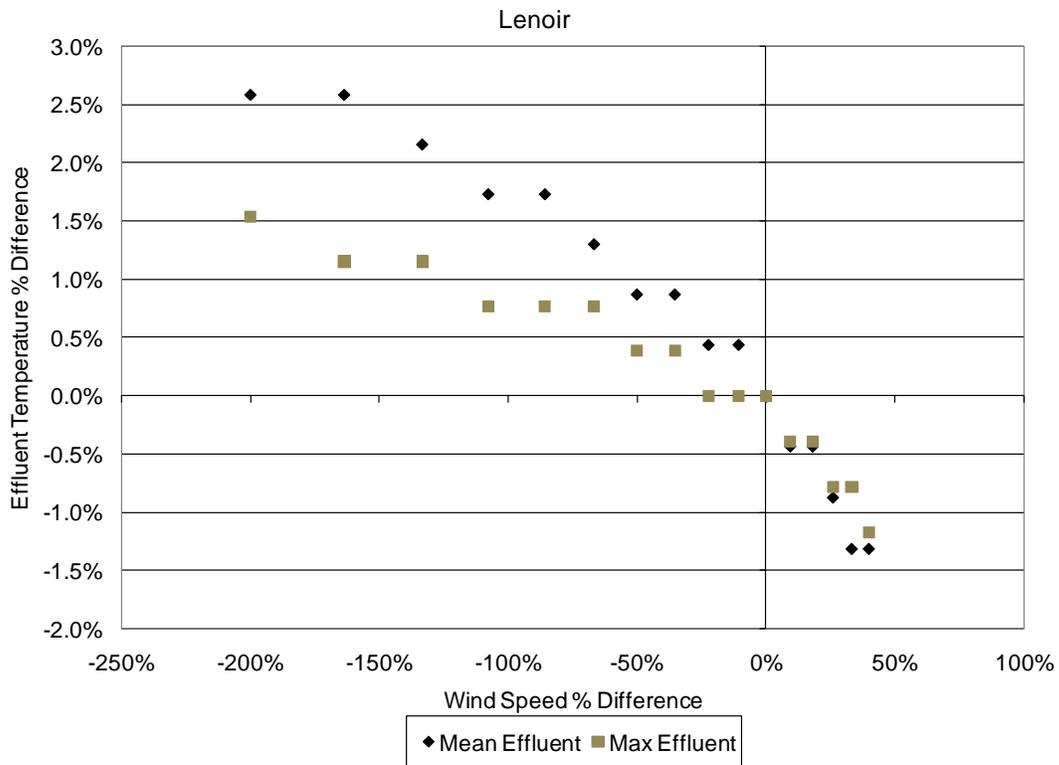
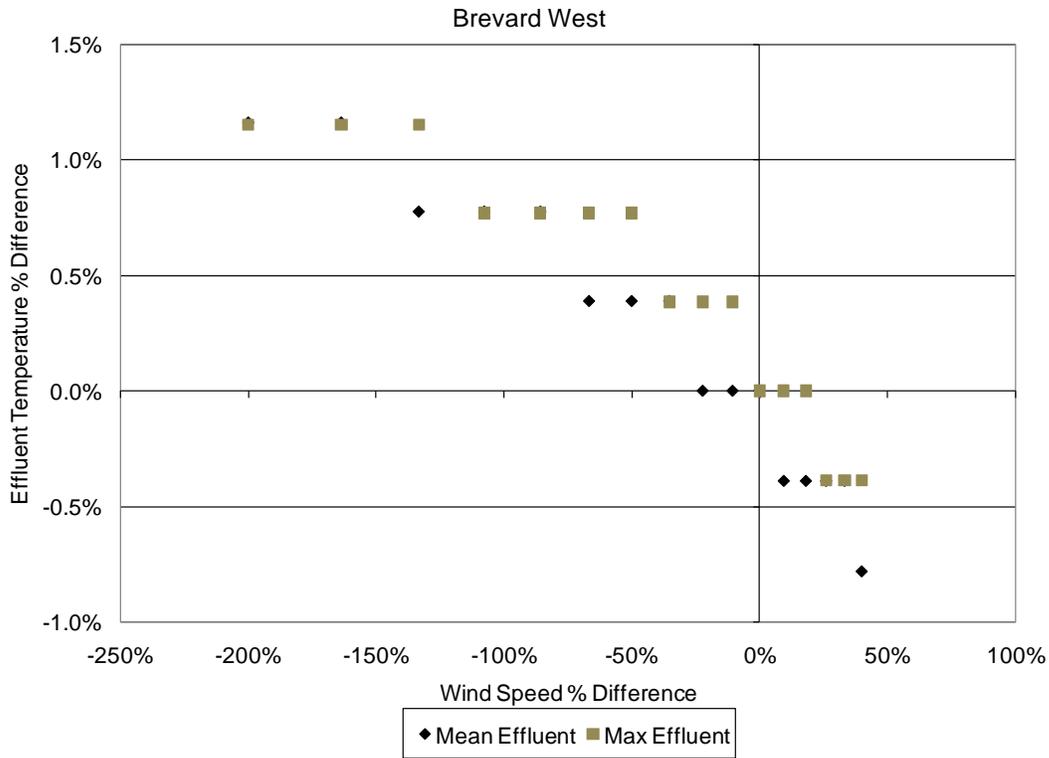
# Sensitivity Analysis

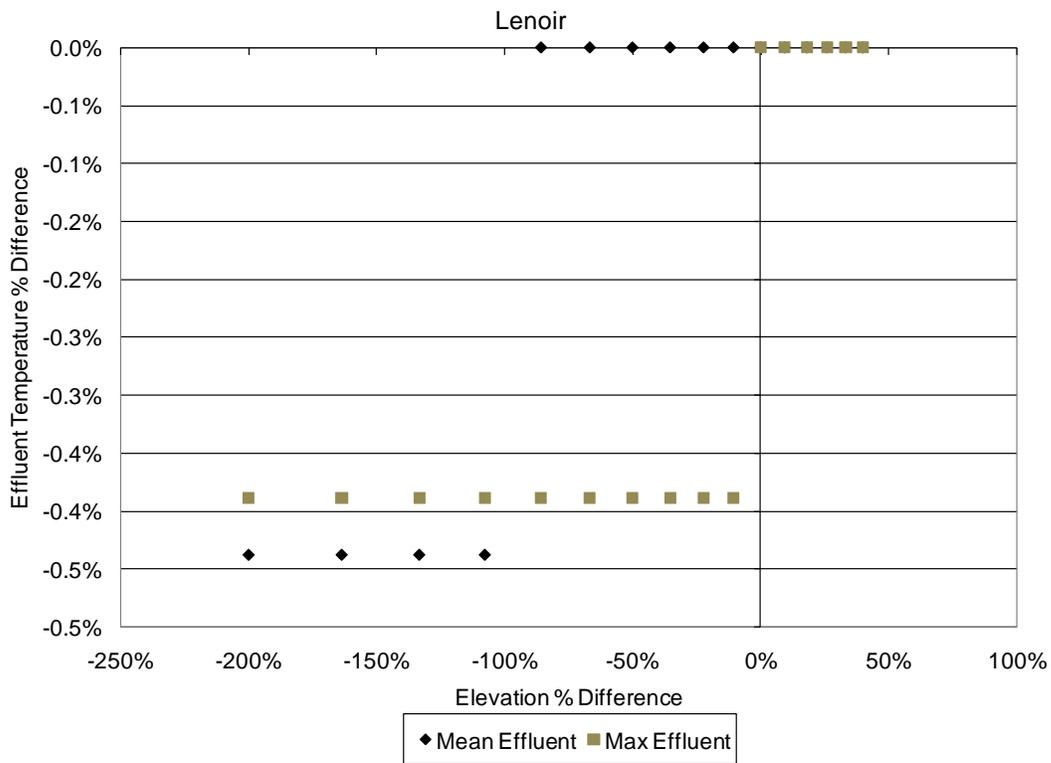
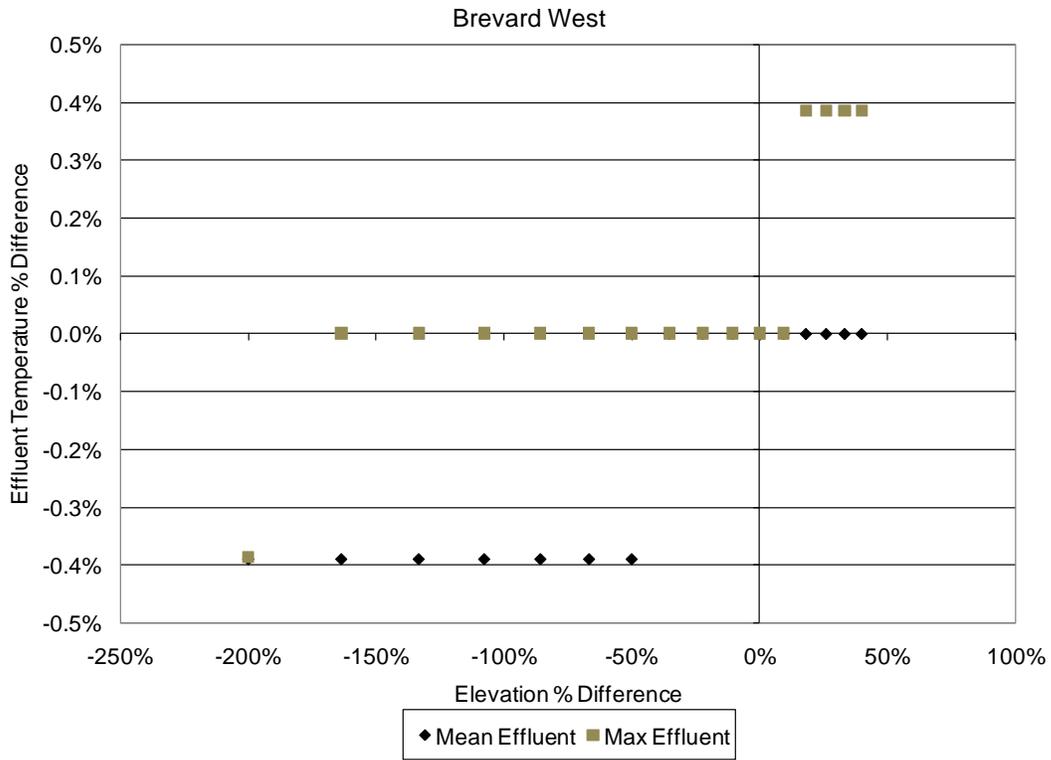


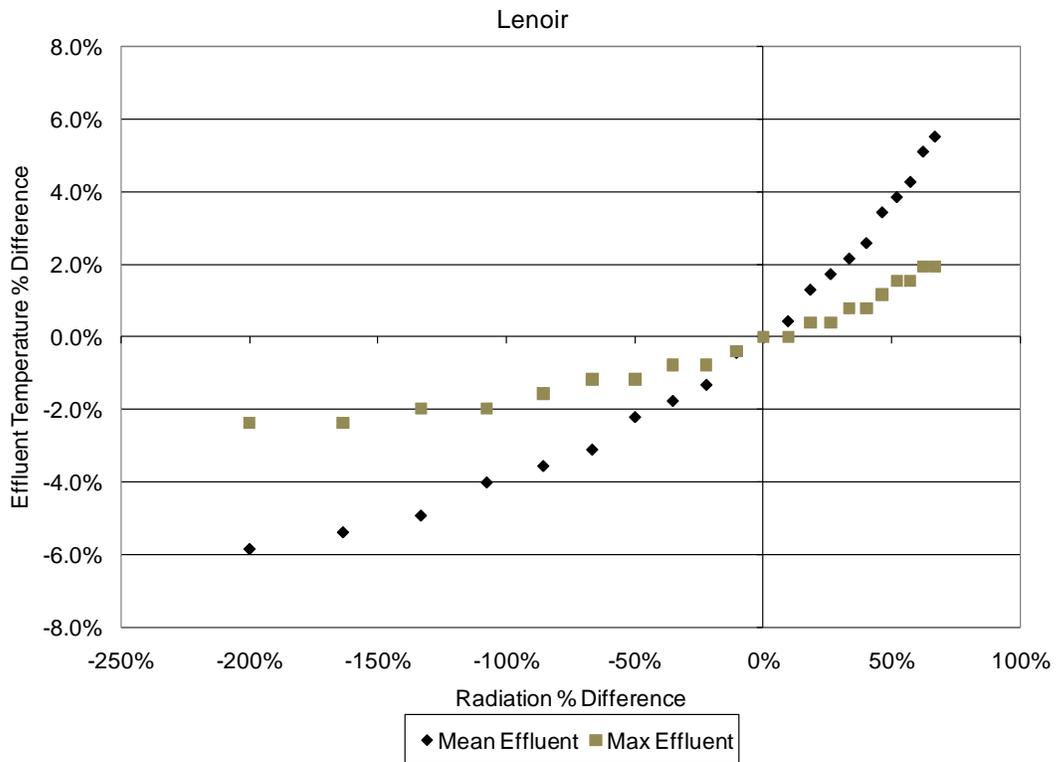
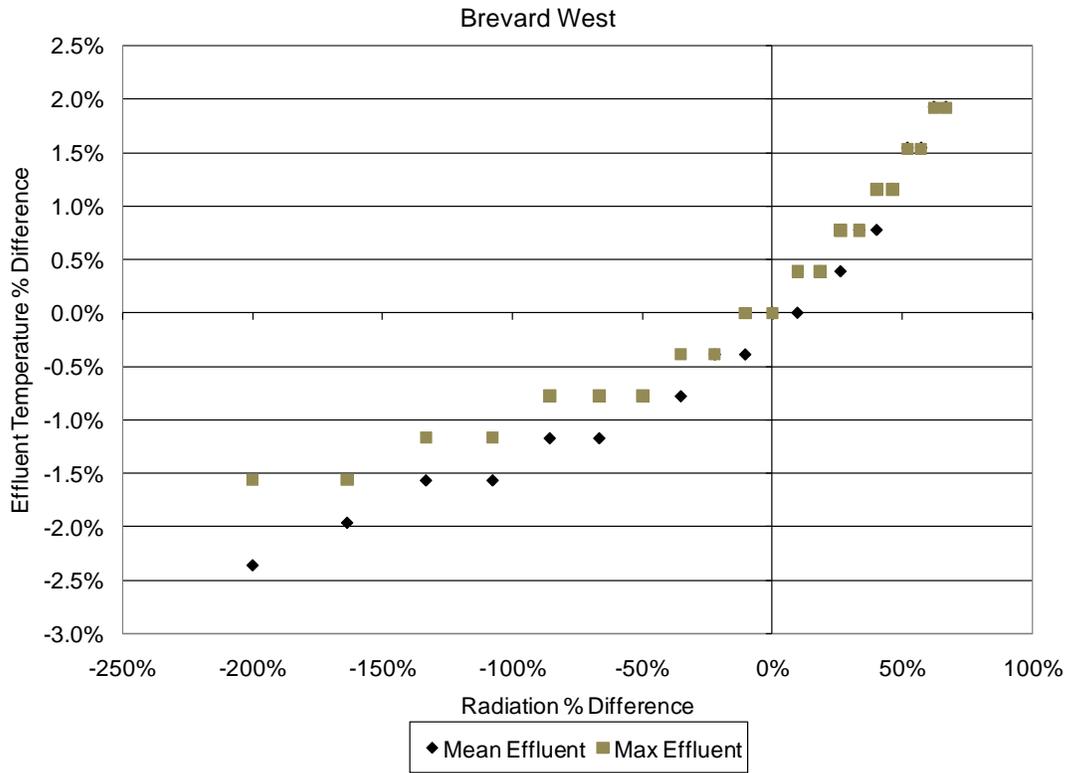


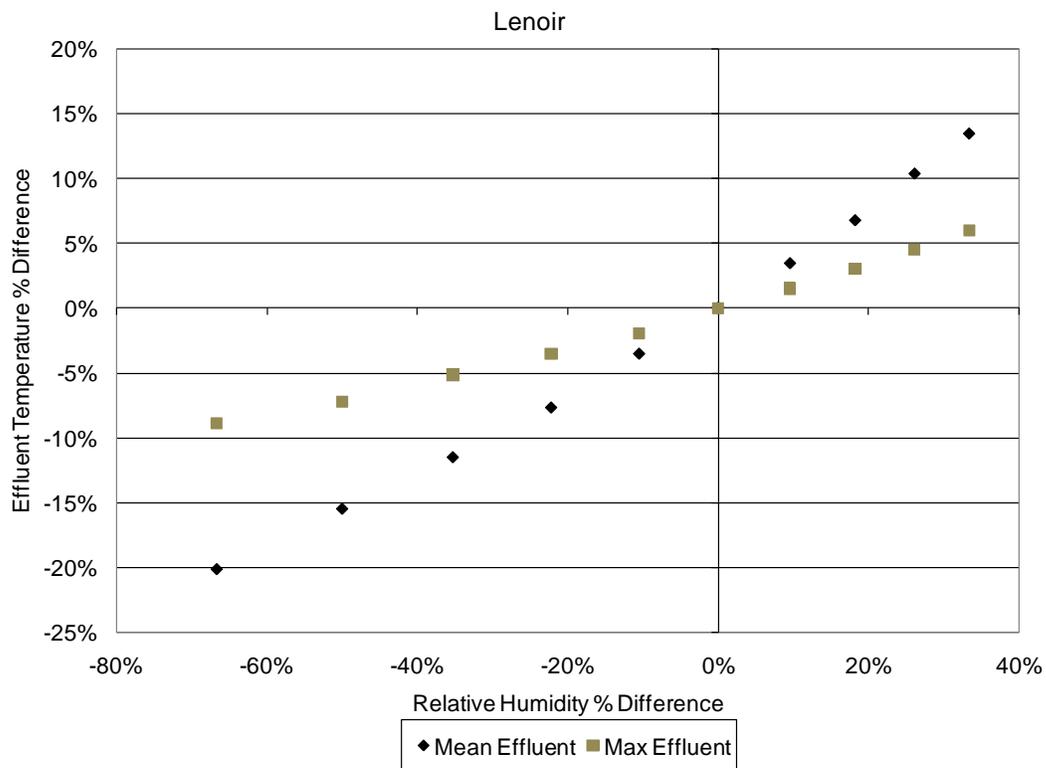
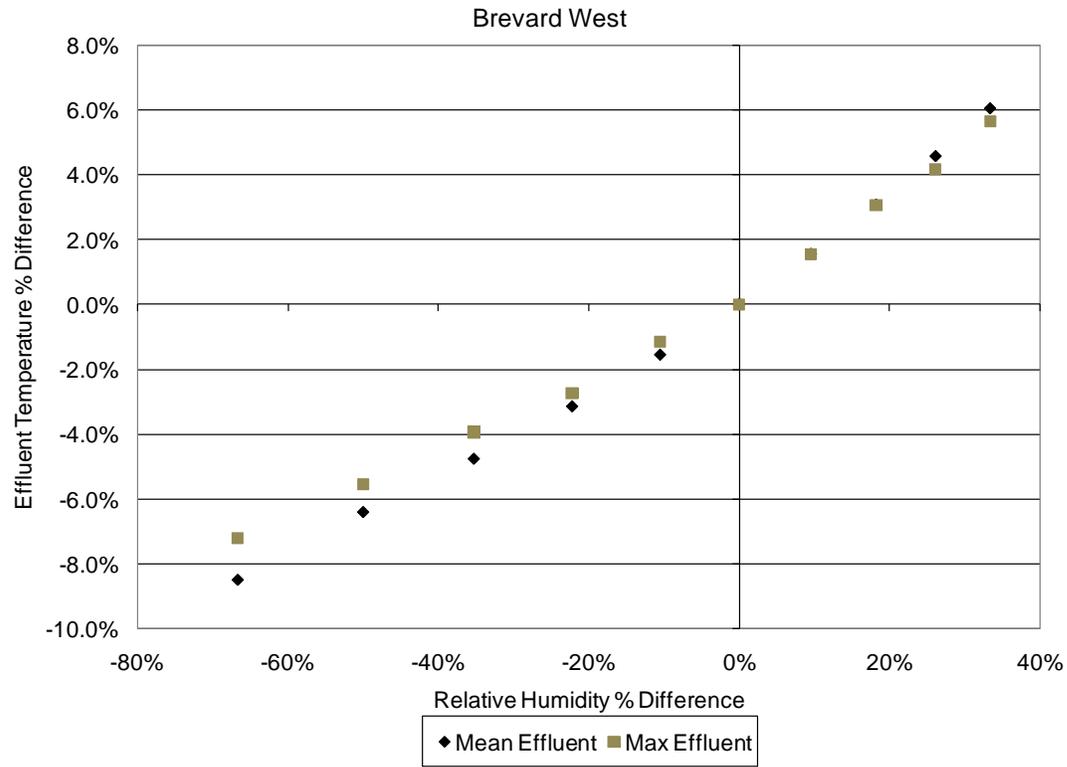


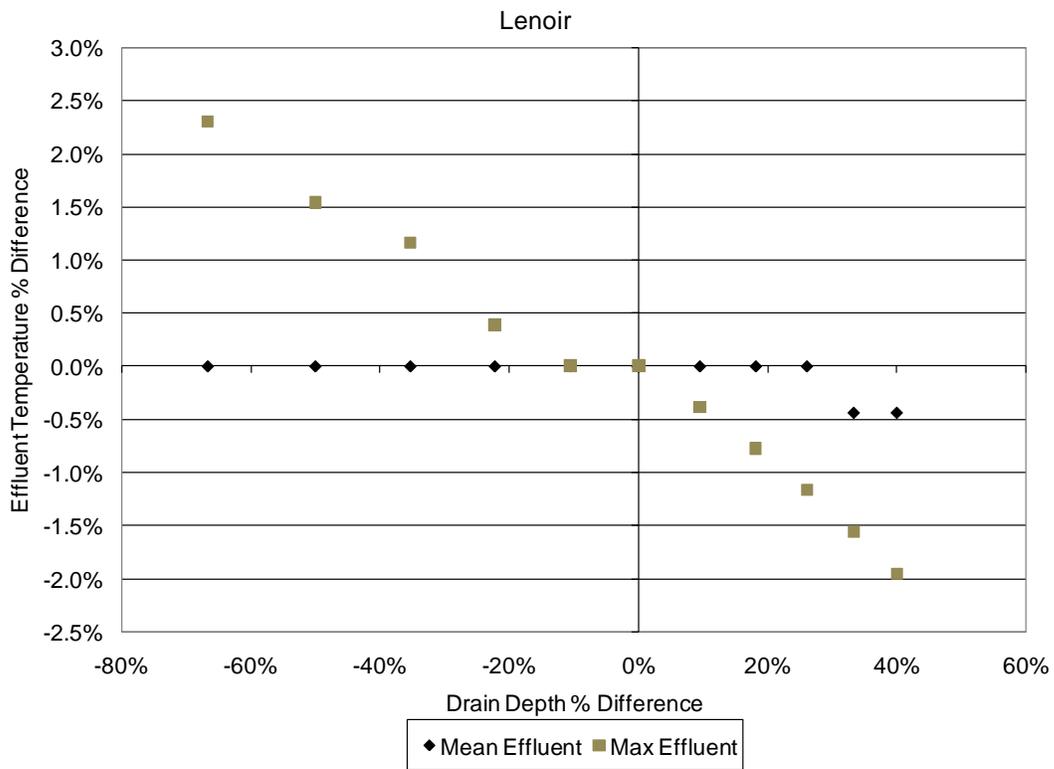
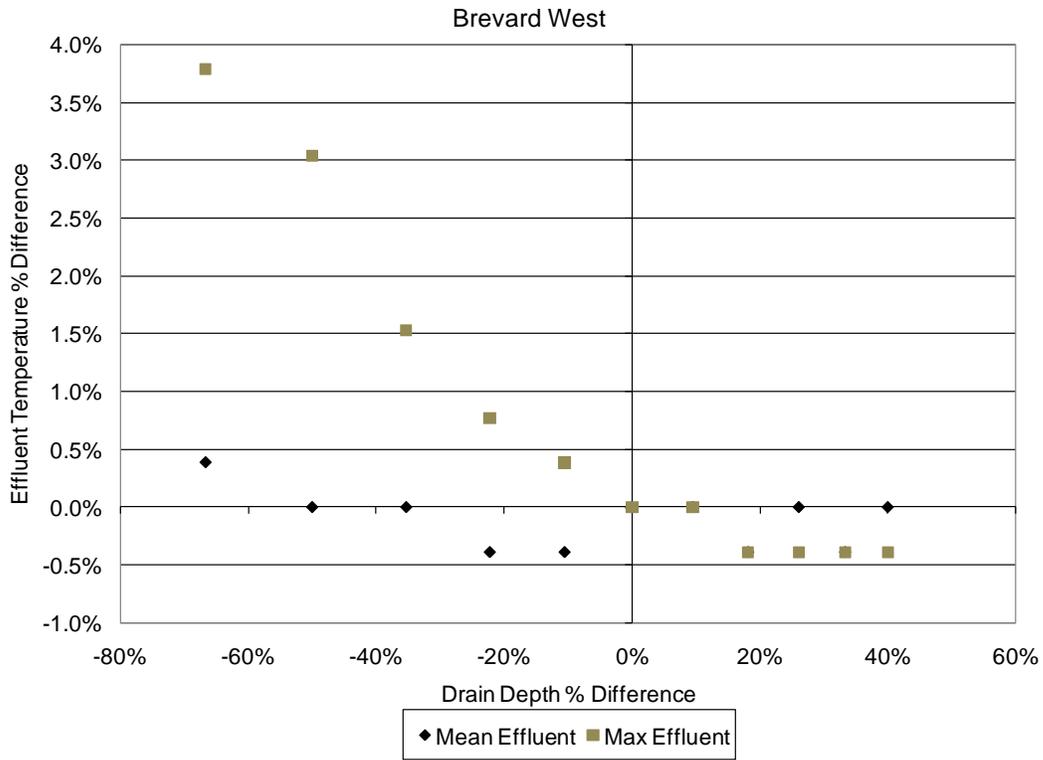


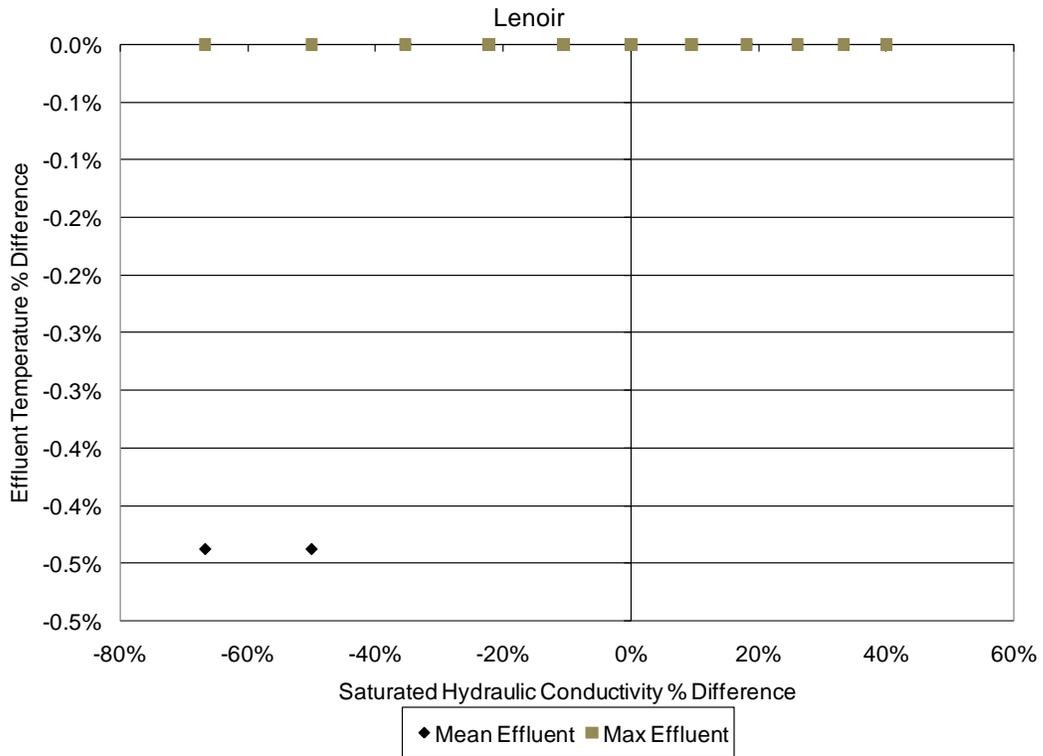
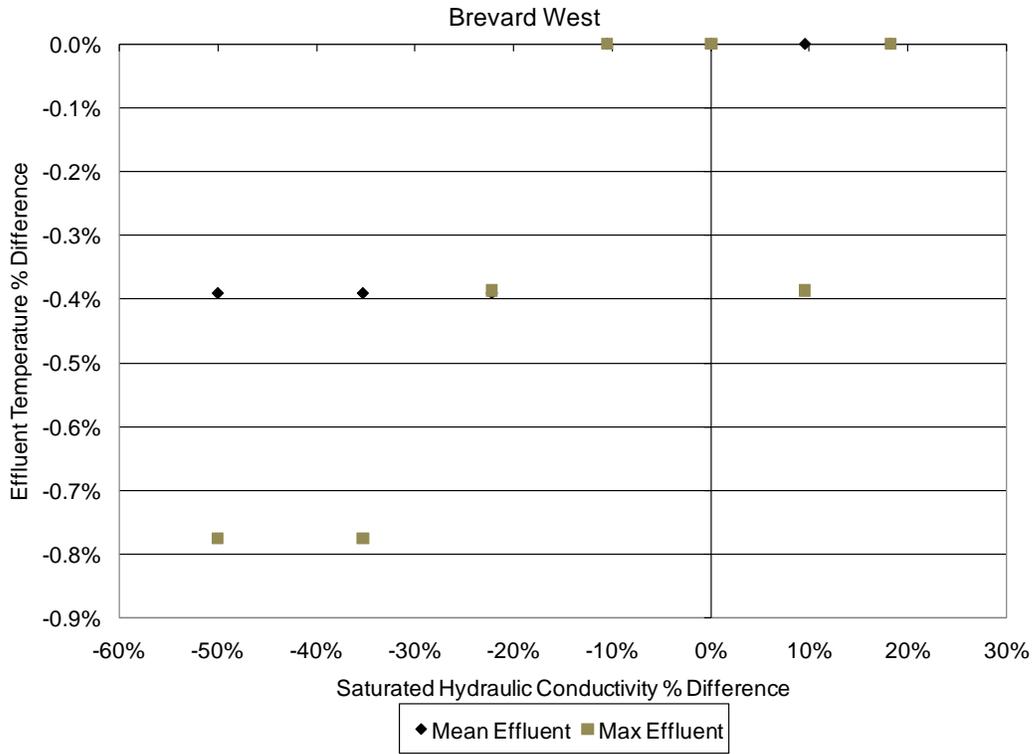


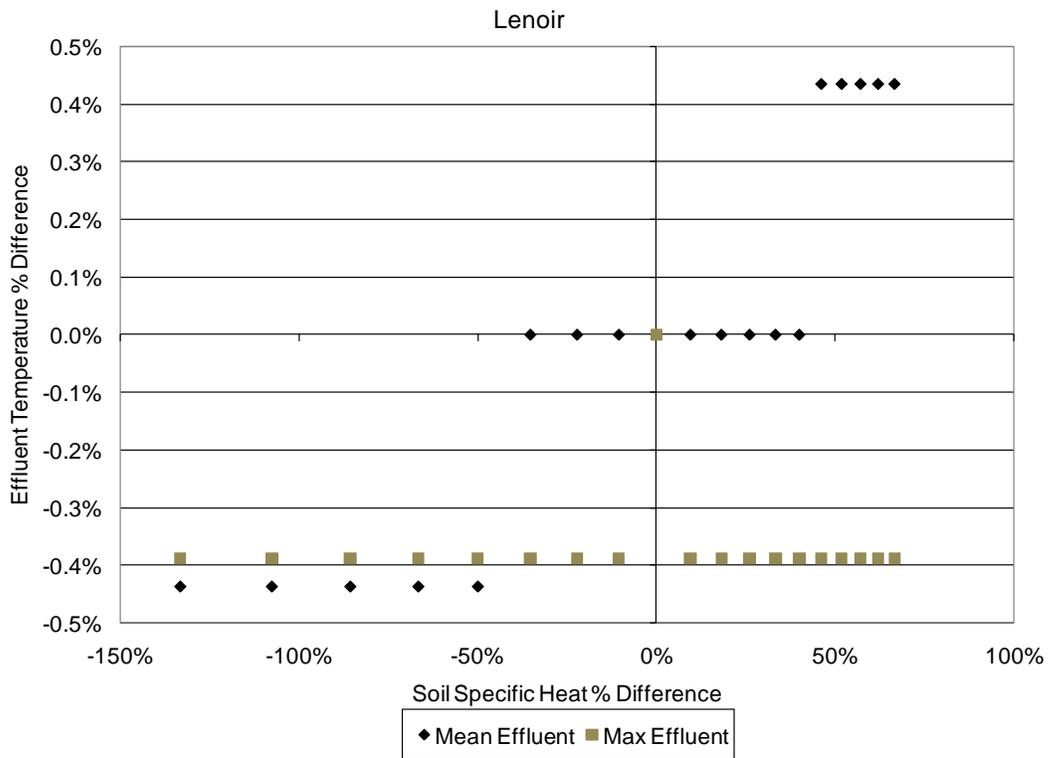
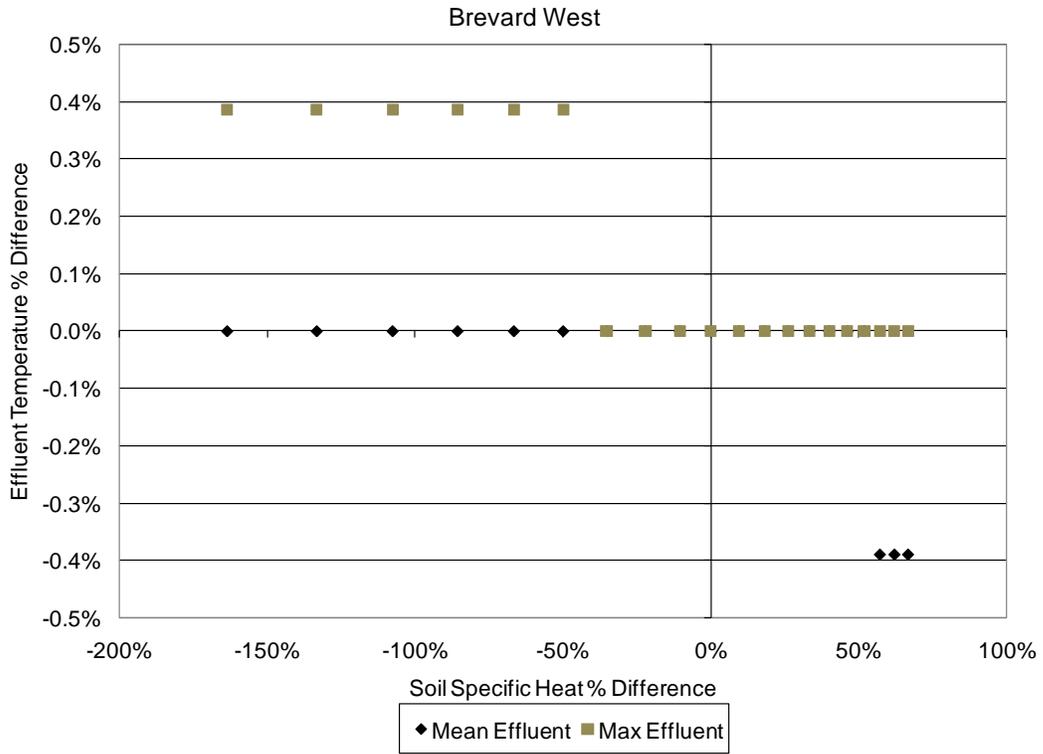


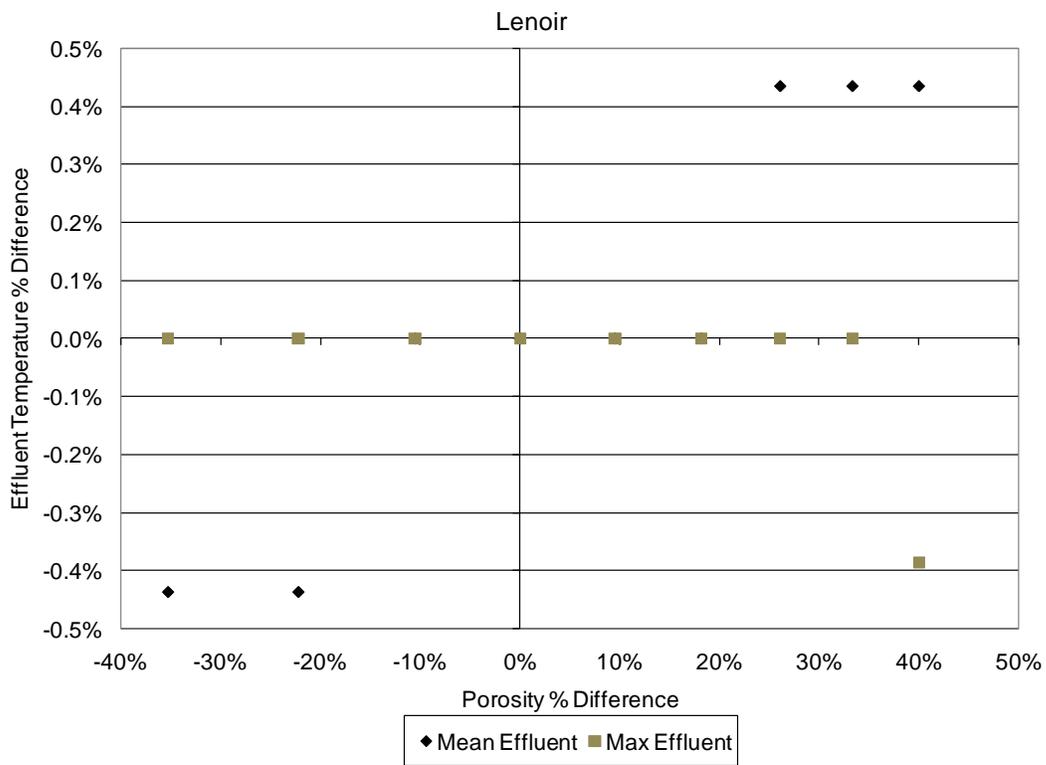
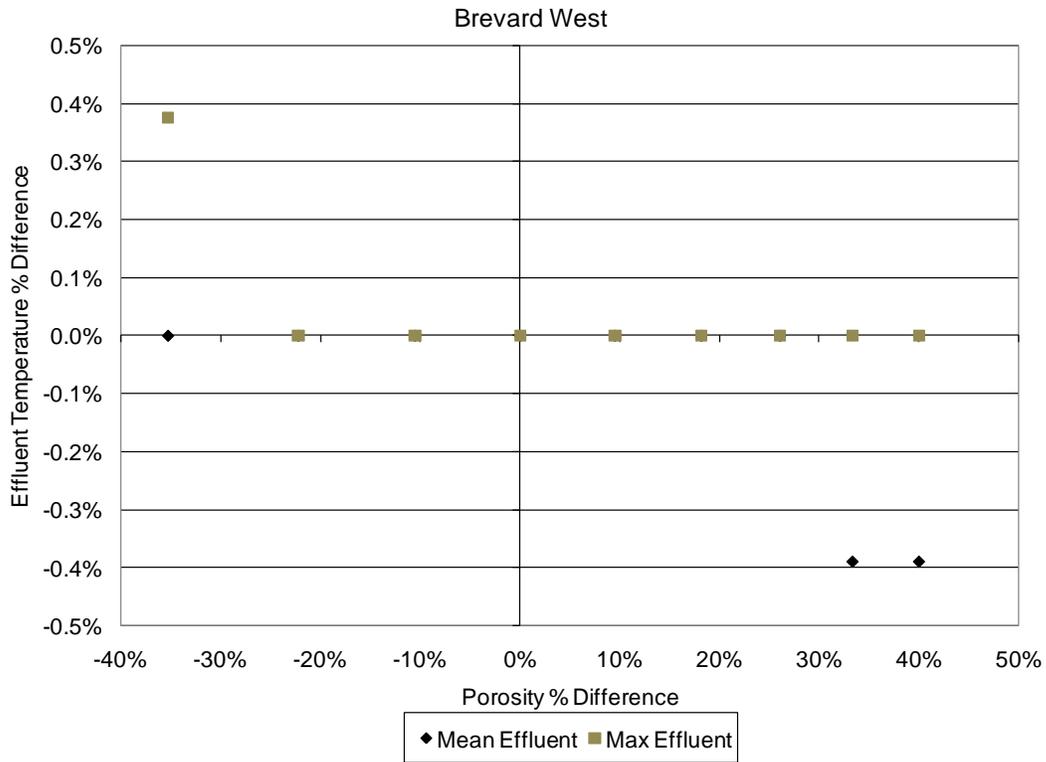


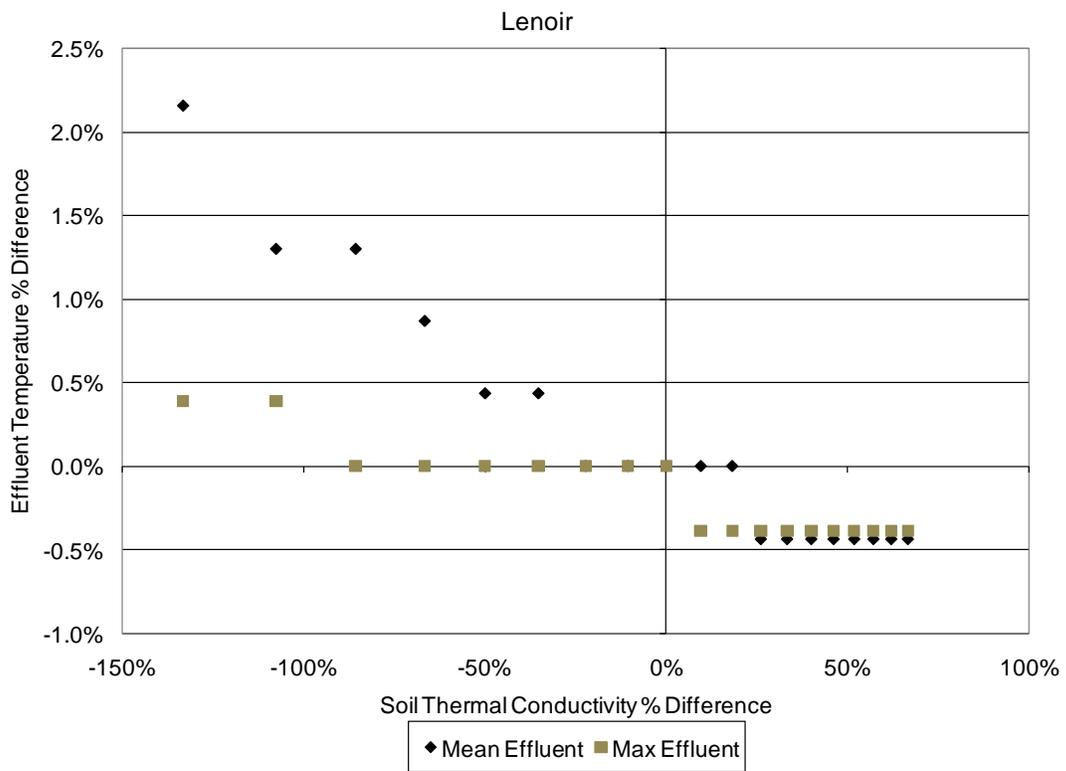
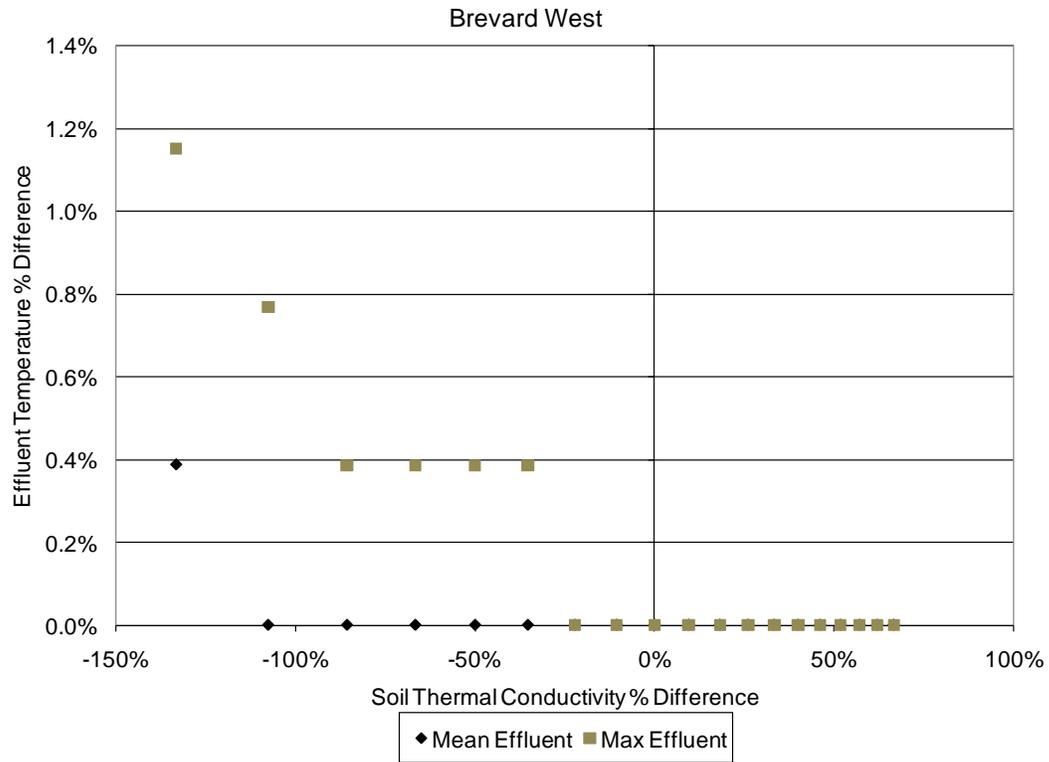












## **APPENDIX F**

### **Effect of Urban Stormwater BMPs on Runoff Temperature: Sample SAS Code and Analysis Output**

## Wilcoxon Rank Sum

Primarily used to compare temperature measurements from two different locations

### SAS Code

```
DATA MULTI_TEMP_BASE;
INFILE "E:\Documents and Settings\mpjones2\My
Documents\SAS\Bioretention\Caldwell-Locations-InandOut.csv" DSD;
input DATE $ LOCATION $ TYPE $ TEMPERATURE;
*Previous lines input the correct data;

*Prints original dataset;
proc print data=MULTI_TEMP_BASE;
run;

*Removes unwanted classes from the dataset;
DATA MULTI_TEMP;
SET MULTI_TEMP_BASE;
IF TYPE = 'MEDIAN' THEN DELETE;
RUN;

*Prints modified dataset;
proc print data=MULTI_TEMP;
run;

*Computes summary statistics;
proc means data=MULTI_TEMP N Mean StdDev Min Max Median;
class LOCATION;
run;

*Computes Wilcoxon rank sum test;
proc nparlway data = MULTI_TEMP wilcoxon;
class LOCATION;
var TEMPERATURE;
run;
Quit;
```

## Analysis Output

The SAS System

### The NPAR1WAY Procedure

Wilcoxon Scores (Rank Sums) for Variable TEMPERATURE  
Classified by Variable LOCATION

LOCATION	N	Sum of Scores	Expected Under H0	Std Dev Under H0	Mean Score
INLET	46	2802.50	2139.0	128.056607	60.923913
OUTLET	46	1475.50	2139.0	128.056607	32.076087

Average scores were used for ties.

### Wilcoxon Two-Sample Test

Statistic                    2802.5000

#### Normal Approximation

Z                            5.1774

One-Sided Pr > Z           <.0001

Two-Sided Pr > |Z|         <.0001

#### t Approximation

One-Sided Pr > Z           <.0001

Two-Sided Pr > |Z|         <.0001

Z includes a continuity correction of 0.5.

### Kruskal-Wallis Test

Chi-Square                   26.8459

DF                            1

Pr > Chi-Square             <.0001

## Wilcoxon Rank Sum Test

Primarily used to compare measured temperatures to the 21°C trout temperature threshold

### SAS Code

```
DATA MULTI_TEMP_BASE;
INFILE "E:\Documents and Settings\mpjones2\My
Documents\SAS\Bioretention\Botanical.csv" DSD;
input DATE $ MEDIANINLET MEDIANOUTLET MAXINLET MAXOUTLET;
*Previous lines input the correct data;

*Prints original dataset;
proc print data=MULTI_TEMP_BASE;
run;

*Removes unwanted classes from the dataset;
DATA MULTI_TEMP;
SET MULTI_TEMP_BASE;
*IF TYPE = 'Max' THEN DELETE;
RUN;

*Prints modified dataset;
proc print data=MULTI_TEMP;
run;

*Computes summary statistics;
proc means data=MULTI_TEMP N Mean StdDev Min Max Median;
run;

*Computes Wilcoxon signed rank test;
proc univariate data = MULTI_TEMP mu0 = 21;
var MAXINLET;
run;

Quit;
```

# Analysis Output

The SAS System

## The UNIVARIATE Procedure

Variable: MAXINLET

### Moments

N	89	Sum Weights	89
Mean	23.1140449	Sum Observations	2057.15
Std Deviation	4.03485291	Variance	16.280038
Skewness	-0.1625069	Kurtosis	-0.1671347
Uncorrected SS	48981.7009	Corrected SS	1432.64334
Coeff Variation	17.4562822	Std Error Mean	0.42769355

### Basic Statistical Measures

Location		Variability	
Mean	23.11404	Std Deviation	4.03485
Median	23.24000	Variance	16.28004
Mode	20.95000	Range	20.18000
		Interquartile Range	5.77000

### Tests for Location: Mu0=21

Test	-Statistic-	-----p Value-----	
Student's t	t 4.942896	Pr >  t	<.0001
Sign	M 11.5	Pr >=  M	0.0192
Signed Rank	S 1038.5	Pr >=  S	<.0001

### Quantiles (Definition 5)

Quantile	Estimate
100% Max	32.34
99%	32.34
95%	29.50
90%	28.71
75% Q3	25.95
50% Median	23.24
25% Q1	20.18
10%	18.66
5%	16.76
1%	12.16
0% Min	12.16

## **Regression Analysis**

Primarily used to identify trends in temperature changes over time

### **SAS Code**

```
DATA CORREL;
INFILE "E:\Documents and Settings\mpjones2\My
Documents\SAS\Bioretention\Caldwell-Hours.csv" DSD;
input DATE$ TIME TIMENOON MEDIANINLET MEDIANOUTLET MAXINLET MAXOUTLET;
*Previous lines input the correct data;

*Prints data input;
proc print data=CORREL;
run;

proc glm data=CORREL;
model MEDIANINLET = TIMENOON;
run;

proc reg data=CORREL;
model MEDIANINLET = TIMENOON;
plot MEDIANINLET*TIMENOON;
run;

Quit;
```

# Analysis Output

The SAS System

The REG Procedure

Model: MODEL1

Dependent Variable: MEDIANINLET

Number of Observations Read	59
Number of Observations Used	58
Number of Observations with Missing Values	1

## Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	124.76422	124.76422	10.04	0.0025
Error	56	696.21721	12.43245		
Corrected Total	57	820.98143			

Root MSE	3.52597	R-Square	0.1520
Dependent Mean	25.20552	Adj R-Sq	0.1368
Coeff Var	13.98887		

## Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr >  t
Intercept	1	27.65946	0.90245	30.65	<.0001
TIMENOON	1	-0.43784	0.13821	-3.17	0.0025