



## AN ABSTRACT OF THE THESIS OF

Travis A. Grohman for the degree of Master of Science in Water Resources Engineering presented on September 14, 2017.

Title: Proposal of an Automated Workflow for Appropriate Tile Drainage Simulation in Denitrifying Bioreactor Planning and Design

Abstract approved: \_\_\_\_\_

Frank W.R. Chaplen

Denitrifying bioreactors are an edge-of-field treatment technology particularly well suited for nitrate removal from subsurface (tile) drainage effluent of agricultural lands. The effectiveness of a proposed denitrifying bioreactor is site-specific and depends upon the complicated interplay of field and environmental parameters controlling discharge events, nitrate concentrations, subsurface temperatures, dissolved oxygen levels, pH levels, and the relative timing of each. Furthermore, denitrifying bioreactors will perform poorly still at sites well suited to their incorporation if they are not designed and operated using appropriate parameters for the drainage discharge and nitrate levels they intercept. This thesis discusses an automated workflow for the assisted creation and calibration of subsurface drainage simulations appropriate for use in planning the construction and operation of denitrifying bioreactors. The workflow was designed such that users of flexible backgrounds may 1) quickly

build first-guess feasibility simulations for denitrifying bioreactor performance at a given site, 2) Calibrate the first-guess simulations using measurements of discharge and nitrate concentration until satisfied with the simulation performance, and 3) proceed to determine efficient design and operation parameters for a denitrifying bioreactor using the results of the calibrated simulation. To demonstrate and test the automated workflow concept, it was used to create simulations for four fields in the Oregon Willamette Valley. Results of the first guess feasibility simulations, 12-point calibrated simulations, and 20-point calibrated simulations were compared with measurements of tile drainage and nitrate concentrations collected in a previous study.

©Copyright by Travis A. Grohman  
September 14, 2017  
All Rights Reserved

Proposal of an Automated Workflow for Appropriate Tile Drainage  
Simulation in Denitrifying Bioreactor Planning and Design

by

Travis A. Grohman

A THESIS

submitted to

Oregon State University

in partial fulfillment of  
the requirements for the  
degree of

Master of Science

Presented September 14, 2017  
Commencement June 2018

Master of Science thesis of Travis A. Grohman presented on September 14, 2017.

APPROVED:

---

Major Professor, representing Water Resources Engineering

---

Director of the Water Resources Graduate Program

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

---

Travis A. Grohman, Author

## ACKNOWLEDGEMENTS

# TABLE OF CONTENTS

	<u>Page</u>
1 Introduction	1
2 Background and Literature Review	8
2.1 Subsurface Drainage . . . . .	8
2.1.1 Subsurface Drainage Overview . . . . .	8
2.2 Denitrifying Bioreactors . . . . .	12
2.2.1 Denitrification Kinetics in Denitrifying Bioreactors . . . . .	14
2.3 Software and Previous Models . . . . .	17
2.3.1 Drainmod . . . . .	17
2.3.2 Rosetta . . . . .	27
2.3.3 PEST . . . . .	28
2.3.4 Protocol and Interactive Routine for the Design of Subsurface Bioreactors (PIRDSB) . . . . .	29
3 Materials and Methods	31
3.1 Model Overview . . . . .	31
3.1.1 User Interface . . . . .	31
3.2 The Field Component . . . . .	35
3.2.1 Soils . . . . .	35
3.2.2 Weather . . . . .	40
3.2.3 Drainage Parameters . . . . .	44
3.2.4 Crops, Fertilizer, and Irrigation . . . . .	46
3.2.5 Automated Calibration With PEST . . . . .	53
4 Results	64
4.1 Validation Experiment Overview . . . . .	64
4.1.1 Site Description and Model Building . . . . .	67
4.1.2 Results and Discussion . . . . .	75
5 Conclusions And Future Directions	93
Bibliography	97
Appendix	107



# LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
2.1 Manual Installation of a Clay Tile Drainage System in the Early 20th Century [Powers and Teeter, 1916] . . . . .	9
2.2 A Tractor-Pulled Trenching Machine. (image taken from [AFT Trenchers Ltd, 2017]) . . . . .	9
2.3 Some Common Subsurface Drainage Layouts [U.S. Natural Resources Conservation Service, 1985] . . . . .	10
2.4 Diagram of the Hydraulic Gradient Across a Bioreactor . . . . .	14
2.5 The Drainmod Simulation Scenario. Taken from Skaggs [1982b] . . .	22
2.6 Screenshot of the Drainmod 6 Native Interface . . . . .	26
2.7 Screen Captures of the PIRDSB Model . . . . .	30
3.1 Graphical Representation of the two major model components. The field component models daily discharge rates and nitrate concentrations in subsurface (tile) flow. The denitrifying bioreactor component (PIRDSB) models the daily mass of nitrate removed and the daily mass of nitrate exiting the system untreated. . . . .	32
3.2 Screen Capture of the Automated Workflow Start Page . . . . .	34
3.3 A flowchart describing the Online Soil Survey/Rosetta Import Procedure	39
3.4 The Weather_Inputs worksheet Interface for entry and quality check of weather values. . . . .	42
3.5 The PRISM climate group import User Form. . . . .	43
3.6 Diagram Illustrating Drainage Parameters. Taken from Skaggs et al. [2012] . . . . .	44
3.7 Drainage Design Input Form . . . . .	45
3.8 Screen Capture of Explicit Crop Parameters Required of Drainmod .	47
3.9 The User Interface to the Highly Simplified Grassland/Generic Crop Procedure . . . . .	50
3.10 An Example Sinusoidal Root Growth for 3 Max Height Months . . .	50

## LIST OF FIGURES (Continued)

<u>Figure</u>	<u>Page</u>
3.11 Screen Capture of Crop Irrigation Scheduler . . . . .	54
3.12 Screen Capture of Automated Calibration Interface . . . . .	55
3.13 Table of Drainmod Input Parameters Incuding thier Relative Utility in Calibration. Taken from Skaggs et al. [2012] . . . . .	58
3.14 Hydrology Calibration Routine Flowchart . . . . .	59
4.1 The Experimental Set Up of Warren,2002 for Flow Measurement and Nitrate Sampling . . . . .	66
4.2 First year (left) and Established (right) Tall Fescue of Fields 1 and 3 Respectively. Taken from Warren [2002] . . . . .	72
4.3 Field 1 First Guess Uncalibrated Simulation Vs Observed Discharge .	76
4.4 Field 2 First Guess Uncalibrated Simulation Vs Observed Discharge .	77
4.5 Field 3 Precipitation Data and Measured Instantaneous Discharge Rates Over the Period of Measurement . . . . .	78
4.6 Field 4 Daily Precipitation Data, Daily Measured Drainage Discharge Volumes, and Measured Nitrate Concentrations . . . . .	79
4.7 Lateral Conductivity (top) and Lateral + Matching Point Conductiv- ity (bottom) Calibrations for Field 2 with Ko as the Initial Guess for Lateral Conductivities . . . . .	82
4.8 Leftover Runoff Forced by the Simulation for Field 2 when Calibrated for the Peak Flow Event and Comparison to Uncalibrated Runoff . .	83
4.9 Lateral Conductivity (top) and Lateral + Matching Point Conductiv- ity (bottom) Calibrations for Field 2 with Ko as the Initial Guess for Lateral Conductivities . . . . .	84
4.10 Field 3 Discharge and Residuals After 12 point Calibration: December 28th 2001- January 8th 2002 . . . . .	86
4.11 Field 3 Discharge After 24 point Calibration: December 28th 2001- January 8th 2002 . . . . .	87

## LIST OF FIGURES (Continued)

<u>Figure</u>	<u>Page</u>
4.12 Field 4 Discharge and Residuals After 12 point Calibration: December 28th 2001- January 8th 2002 . . . . .	88
4.13 Initial Guess and 10 point Calibration for Nitrate Simulations of Field 3	90
4.14 Initial Guess and 20 point Calibration for Nitrate Simulations of Field 3	91
4.15 Initial Guess and Calibrated Nitrate Simulations for Field 4 . . . . .	92
5.1 General Schematics for the Full Scale Denitrifying Bioreactor Built at the OSU Dairy Farm . . . . .	95
5.2 Construction of the Full Scale Denitrifying Bioreactor for Subsequent Validation and Improvement of the Model . . . . .	96

## LIST OF TABLES

<u>Table</u>		<u>Page</u>
3.1	General Parameters used in PEST calibrations . . . . .	63
4.1	General Drainage System Information for Experimental Sites as Reported by Warren [2002] . . . . .	68
4.2	Web Soil Survey Parameters for Soils used in Validation Experiment .	69
4.3	Aggregated Rosetta Predictions Used in Simulation . . . . .	70
4.4	Crop Rotations for Each Field in Validation Study . . . . .	72
4.5	Meteorological Data Sources Used for Validation Experiments . . . .	74

## LIST OF APPENDICES

	<u>Page</u>
A VBA Code	108

# Proposal of an Automated Workflow for Appropriate Tile Drainage Simulation in Denitrifying Bioreactor Planning and Design

## 1 Introduction

Global changes in the nitrogen content of environmental waters have occurred over the last century, resulting from increased anthropogenic production and leeching. Galloway et al. [2003] estimated this anthropogenic leeching as 160 million tons per year, with approximately two thirds sourced from agricultural fertilizer application. Nitrogen applied to soils as fertilizer is susceptible to being washed out from the soil, predominantly as nitrate which is highly mobile in water and does not experience sorption to most sediments [Smil, 2004]. Heavy discharges of nitrate into an aquatic ecosystem cause eutrophication and, subsequently, hypoxia. Eutrophication of aquatic ecosystems involves unintended fertilization of algae, resulting in algae blooms that are larger than a given aquatic ecosystem can sustain. The subsequent hypoxia occurs when the waters are robbed of oxygen during the surge of algae growth and the following surge of algae death and decomposition that occurs as the fertilizer (nitrogen) passes through ecosystem [Vitousek et al., 1997, Mosier et al., 2004]. Because of the potential adverse impacts on aquatic ecosystems, nitrate is regulated federally as a nutrient pollutant and a permit for point discharge is required through the National Pollutant Discharge Elimination System (NPDES) of the United States Environmental Protection Agency (EPA). The Clean Water Act section 402 and Code of Federal Regulations (CFR) 122.1(b) establishes the

framework for the NPDES program.

It has been demonstrated that denitrifying bioreactors located at a drainage system or watershed outlet are an effective tool in the field-scale remediation of excess nitrate. Denitrifying bioreactors involve the filtering of waste water through a solid-state carbon media (most often wood chips). Denitrifying bioreactors reduce nitrate to nitrogen gasses via denitrifying bacteria cultivated within the system. Because denitrifying bacteria are ubiquitous in soils across the world, the systems are populated naturally during operation with no need to artificially inoculate. Denitrifying bacteria utilize carbon released from the solid-state media as a primary electron donor and nitrate from influent waste water as a terminal electron acceptor, thus obtaining cellular energy from the reduction of nitrate. The favorable conditions created by the denitrifying bioreactors allow denitrifying bacteria populations to thrive and effectively serve as a nitrate removal filter.

Denitrifying bioreactors are a promising treatment option for several reasons including cost efficiency, simplicity of construction, no requirements to alter field practices, and minimal required maintenance. They have seen an explosion in popularity over the past decade, particularly in the Midwestern United States, and have recently been acknowledged by the USDA NRCS as an accepted conservation practice (NRCS Conservation Practice Standard code 605, September 2015). There is, however, a significant amount of variability in the effectiveness of a proposed denitrifying bioreactor depending upon environmental and field parameters of a treatment site. Such parameters include local temperatures, precipitation, drainage system design,

soil hydrology, soil chemical parameters (pH, organic matter, dissolved oxygen distribution, etc), fertilizer application rates/timing, and seasonal growth of crops (with associated uptake of water and nitrogen). The interplay of these parameters at a drainage system outlet can be difficult to predict if measurements have not been taken, and scattered observations may be misleading. This can impede the adoption of this conservation practice if it is avoided for fear of failure at test site. Even worse is the potential waste of resources towards building these systems in adverse conditions and having them fail in operation.

Even applied to propitious treatment sites, bioreactors must be sized and designed appropriately to handle the discharge rates and nitrate concentrations they are intended to treat throughout the year for them to be effective. If a system is too small or designed with too steep a hydraulic gradient for the discharge it is intended to treat, much of said discharge will bypass the system untreated and/or the residence time within the bioreactor will not be sufficient to remove nitrate effectively. If a system is designed too large or with excessively long hydraulic residence times not only will they be costlier than necessary, they may actually cause environmental harm through the production of methyl-mercury in extremely reducing situations [Hudson and Cooke, 2011, Shih et al., 2011].

Thus, as support for denitrifying bioreactors grows, so too does the need for sophisticated design tools that allow denitrifying bioreactors to be constructed and operated efficiently. One such tool is the Protocol and Interactive Routine for the Design of Subsurface Bioreactors (PIRDSB) proposed by Cooke and Bell [2014]. This routine was created to model flow and residence time in a bioreactor given a



set of tile drain discharge rates and design parameters. It allows users to optimize size and control structure settings given a target residence time. The routine also allows the user to apply a constant nitrate concentration to subsurface flow and a set of expected denitrification performance information to set goals in terms of nitrate reduction rather than residence time. This routine operates in two modes: a steady state mode in which the user specifies a constant tile drain discharge rate, and a simulation mode which accepts daily tile drain discharge volumes from the vadose zone transport and subsurface drainage simulation software Drainmod [Skaggs, 1978, 1980b, Skaggs et al., 2012]. PIRDSB is an exceptionally well-thought-out tool for designing efficient bioreactors in a wide range of situations given a set of reliable subsurface flow and nitrate concentration data. If this flow rate and/or nitrate concentration data is not available or incomplete, however, choosing replacement input data is difficult and can be an impediment to proper application of the tool.

Steady state flow data is useful for hypothesis testing and a rough initial guesses but it can be difficult to choose steady state flow data appropriately for estimating long term denitrifying bioreactor performance. Even estimating an average or range of performance values on a monthly basis using steady state flows can be difficult with performance depending upon temperature, discharge rate, and nitrate concentrations (which are, in turn, all dependent upon the interplay of a potentially complex set of further variables). If an average flow is used to set the optimum parameters, peak flows that may carry a significant portion of the monthly nitrate pollution are not accounted for and may bypass the treatment system as overflow. If peak flows are used, a converse fate may arise in missing treatment for the majority of the nitrate

by setting parameters for a non-occurring or scarcely occurring nitrate peak flow. Furthermore, the use of average or peak flows depends upon the existence of reliable flow rate estimates and estimates based upon anecdotal or scattered observations of drainage flow may be misleading. It can also be difficult to estimate when and how a fertilizer or manure application will affect nitrate concentrations in the drainage effluent without simulation. For example, Bjorneberg et al. [1996] and Cambardella et al. [1999] found that nitrate concentrations do not always increase in the subsurface flow events directly following a fertilizer application.

The alternative to steady state flow is subsurface drainage simulation with Drainmod. The creation of a working Drainmod simulation from scratch is a daunting task to those familiar and unfamiliar with the software alike. Drainmod is an exceptionally robust tool for vadose zone transport especially well suited at simulating subsurface drainage. The robust nature of Drainmod is due to its flexibility and its ability to consider a wide range of processes in simulation. Unfortunately, the unavoidable cost for Drainmod's ability to consider so many processes is a dizzying number of variables that must be defined for a given simulation run. Most of these variables can be estimated using common resources (such as soil databases), assigned standard values or values within a standard range, assigned unreasonably high or low values so that the process they control is ignored in the simulation, or may be ignored themselves if it is known they will not have any significant impact on the current simulation. To those unfamiliar with the software and its underlying principles however, choosing how to handle the variables is an unreasonably tall order. This is especially true if the user in question is only seeking in initial feasibility estimate.

For those familiar with the software, creation of a feasibility simulation not so great a task, however those intimately familiar with Drainmod are likely to be engineers or professional consultants who need to be able to trust their simulations beyond the level of a first-guess feasibility study. In this case calibration is necessary, which may also be burdensome with no built-in calibration tool available.

The major objective for this model was complete field to edge-of-field treatment simulation allowing users of varying backgrounds (e.g. CAFO operators, Extension Agents, Contractors, Engineers, etc.) to design, construct, and operate denitrifying bioreactors at the optimum efficiency for their site. To accomplish this task, a thoughtfully simplified automated workflow for the creation of subsurface drainage simulations was amended to the previously developed PIRDSB model. The automated workflow uses a combination of commonly available sources to build a Drainmod simulation for subsurface drainage flow and an associated Drainmod NII [Youssef et al., 2005] simulation for nitrate concentration. Soil input files are created using a combination of values reported by SSURGO/STATSGO2 and the pedotransfer, neural-network soil parameter prediction tool Rosetta [Schaap et al., 2001]. SSURGO/STATSGO2 published values for bulk density, water content at -1/3 bar and at -15 bar as well as sand, silt, and clay percentages were passed to Rosetta (version 1.2) which in turn predicted Van Genuchten-Mualem hydrologic soil parameters [Van Genuchten, 1980] for all unique major component soil horizon layers. Commonly used values for fertilizer applications and crop parameters were also included in the model. Additionally, an automated routine for the calibration of a simulation using subsurface drainage discharge and nitrate measurements is included by

incorporation of an a PEST parameter estimation routine [Doherty et al., 1994].

## 2 Background and Literature Review

### 2.1 Subsurface Drainage

#### 2.1.1 Subsurface Drainage Overview

Subsurface drainage, otherwise known as ‘tile’ drainage, involves the installation of water-permeable drainage pipes below the surface of a field in order to improve aeration and keep the water table below the root zone. Subsurface drainage systems are often installed in poorly drained agricultural land in order to prevent flooding, increase trafficability, and (most importantly) reduce excess soil water stresses a shallow water table depth may pose to vegetation on a poorly drained field. Until plastics became a cost effective alternative around the late 1950’s, the most common form of subsurface drainage were permeable ceramic cylindrical sections, or ‘tiles’ placed end-to-end within trenches manually dug to a prescribed grade (see figure 2.1). Today, subsurface drainage lines are almost always perforated plastic piping and may be installed quickly and efficiently using trenching machines (figure 2.2)



Figure 2.1: Manual Installation of a Clay Tile Drainage System in the Early 20th Century [Powers and Teeter, 1916]



Figure 2.2: A Tractor-Pulled Trenching Machine. (image taken from [AFT Trenchers Ltd, 2017])

Drain lines are typically 4 - 6 inches in diameter and installed 0.5 - 2 meters below the surface at a gently sloping gradient of 0.01% to 0.1% towards the outlet point(s). Horizontal spacing between drainage lines can range from about 5 meters to 50 meters and is usually set at 15-20 meters. Deeper drainage lines installed at larger grades and smaller horizontal spacing generally drain more water, however if a drainage line is installed within or below an aquitard (i.e. a layer of heavy clay relative to soil layers above) it will drain less efficiently. Four common subsurface drainage layouts are show in figure 2.3.

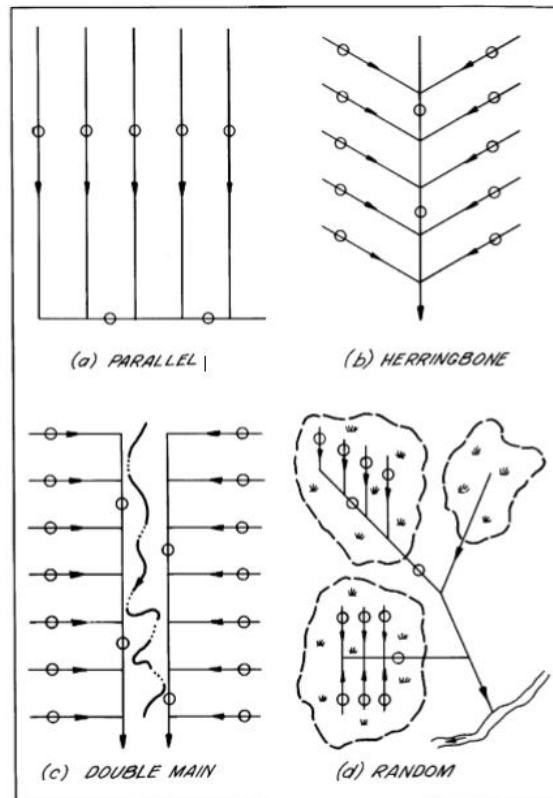


Figure 2.3: Some Common Subsurface Drainage Layouts [U.S. Natural Resources Conservation Service, 1985]

The choice of layout depends upon the needs of the field(s) drained. For example, a rectangular field that needed to be uniformly drained may use a parallel design. A herringbone design gives more flexibility in installing drains selectively where needed. The most flexibility in selective drainage is a random system when may be employed if the field layout is complex. All four of these designs incorporate at least one main or ‘stem’ drainage line that carries a bulk of the drainage off the field to an outlet point. These main drainage lines are typically larger than the lateral drainage lines (6 - 12 inches) to account for the increased discharge rates. Main lines are typically designing for conveyance of flow drained by the lateral lines rather than drainage themselves (though they may, of course, contribute to drainage of the field if permeable).

### **Tile Drains and Nitrate**

Artificial drainage has been shown to modify the nitrogen cycle as well as the hydrology of agricultural fields. The relatively rapid transport of drainage water into tile lines decreases the time for natural processes like denitrification and root uptake to occur in the root zone of the soil [Kellman, 2005]. This quantity of nitrate lost to tile drainage has often been correlated with problems at the scale of watersheds [Mulla and Strock, 2008]. Specific studies indicating this include Fenelon and Moore [1998] who identified subsurface drainage as a major contributor to  $NO_3^-$  pollution of Sugar Creek in Indiana over a four year study, and David et al. [1997] who found similar results studying the Embarras River Watershed in Illinois. As nitrate is highly soluble in water and venerable to being washed out of the root zone with water fluxes,



the amount of nitrate entering tile drains on fertilized agricultural fields generally increases with precipitation and flow rate [Cambardella et al., 1999, Jaynes et al., 1999, Bolton et al., 1970, Bakhsh et al., 2007]. Although the nitrate loss generally shows a positive correlation with the subsurface drainage flow, the interplay between the two can be complicated and inconstant. For example, de Vos et al.[2000] found a correlation between nitrate concentration and flow events while Jaynes et al. [1999] found no such correlation. Bjorneberg et al, [1996] and Cambardella et al, [1999] found that nitrate concentrations do not always increase in the subsurface flow events directly following a fertilizer application.

## 2.2 Denitrifying Bioreactors

Denitrifying bioreactors are one option for treatment of excess nitrate in agricultural runoff. Denitrifying bioreactors are an edge-of-field treatment technology typically designed to treat 1.6 - 4.4 acres of field per 100 square feet of denitrifying bioreactors surface area [Schipper et al., 2010]. Although still considered to be an emerging technology (with the first published suggestion credited to Blowes et al. [1994]) denitrifying bioreactors are a federally recognized conservation practice. Federal funding has been made available for building these systems with the USDA Natural Resources Conservation Service (NRCS) Conservation Practice Standard No. 605 published in September of 2015. These systems filter waste water through a solid substrate that acts as a source of labile carbon. As waste water is filtered through the solid substrate, dissolved organic carbon (DOC) is released into the waste water.

Heterotrophic denitrifying bacteria are able to utilize the dissolved carbon as an electron donor and the nitrate from receiving waters as an electron acceptor, thus obtaining cellular energy from the reduction of nitrate into nitrogen gas. The most common carbon substrate for use in denitrifying bioreactors are wood chips as they are cheap, commonly available, and provide an adequate (non-limiting) amount of labile carbon for a substantial lifetime. Previous studies have suggested that wood chips offer a non-limiting supply of carbon supporting denitrification for 5-15 years [Robertson et al., 2000, 2008, 2009, Schipper and Vojvodić-Vuković, 2001] The NRCS conservation practice standard No. 605 recommends a lifetime of ten years.

Denitrifying bioreactors can generally be expected to remove between 222 g Nitrate m<sup>3</sup> day<sup>-1</sup> of the influent they intercept if they are built and operated correctly [Schipper et al., 2010]. Design parameters for denitrifying bioreactors include the media type (i.e. woodchip effective porosity and hydraulic conductivity) and the volume of the bioreactor unit. It is generally suggested that clean woodchips of 13-25 mm mean diameter are used in construction in order to avoid flow-resistant compression and preferential flow path formation [Christianson et al., 2012]. Hydraulic conductivity values for woodchips of this size have been found to range between 1.2 and 11 cm/s with a tendency for conductivity to decrease over time [Robertson et al., 2005, Driel et al., 2006]. A bioreactor is generally sized to hold a given percentage of a target flow for a target hydraulic residence time. The nitrate removed in a bioreactor is directly proportional to the residence time within the system. Residence times generally range between 3 and 24 hours. Residence times may be altered within the

Figure 2.4: Diagram of the Hydraulic Gradient Across a Bioreactor

system through the use of control structures (or stoplogs) at the inlet and outlet of a bioreactor. The differences in stoplog heights between the inlet and outlet control structures determines the hydraulic gradient across the system (see figure ??). Flow through a bioreactor is generally assumed to follow Darcy's Law

$$\frac{Q}{A} = -K \left( \frac{\Delta H}{\Delta x} \right) \quad (2.1)$$

where  $A$  is the cross sectional area through which flow occurs,  $\Delta H$  is the change in hydraulic head between two points, and  $\Delta x$  is the linear distance between them.

At high velocities or hydraulic gradients, an intrinsic permeability term must be included (Darcy-Forcheimer) [Ghane et al., 2014]. The Darcy Forcheimer equation takes the form

$$-i = \frac{1}{k_F} q + \omega q^2 \quad (2.2)$$

where  $i$  is the hydraulic gradient (cm/cm),  $k_F$  is the Forcheimer hydraulic conductivity (cm/s),  $\omega$  is constant, and  $q$  is the specific discharge (cm/s).

### 2.2.1 Denitrification Kinetics in Denitrifying Bioreactors

Nitrate removal rates in denitrifying bioreactors range generally from 2-22 mg nitrate L<sup>-1</sup> Day<sup>-1</sup> [Shipper, 2010]. Heterotrophic denitrification is an enzyme catalyzed

reaction and is therefore described by Michaelis-Menten kinetics.

$$r_{NO_3} = \frac{v_{max}[NO_3]_{in}}{K_m + [NO_3]} \quad (2.3)$$

Equation 2.3 is the Michaelis-Menten equation where  $r_{NO_3}$  is the reaction rate in mg nitrate  $L^{-1}hour^{-1}$ ,  $v_{max}$  is the maximum reaction rate (mg nitrate  $L^{-1}hour^{-1}$ ,  $K_M$  is the rate constant (mg nitrate  $L^{-1}$ ) equivalent to the concentration of nitrate at which  $r_{NO_3} = \frac{1}{2}v_{max}$ , and  $[NO_3]_{in}$  is the influent concentration of nitrate (mg nitrate  $L^{-1}$ )

To determine the reaction rate constants  $v_{max}$  and  $K_m$  the inverse of nitrate removal rate (  $1/r_{NO_3}$  ) is plotted against the inverse of the influent nitrate concentration ( $1/[NO_3]_{in}$  ). The result is a linear function with a slope of  $K_m/v_{max}$  and intercept of  $1/v_{max}$  (Lineweaver-Burke form of the above Michaelis-Menten equation). When nitrate concentrations are high enough to be functionally non-limiting, the reaction rate can be considered operationally zero-order.

$$r_{NO_3} \approx v_{max} \quad (2.4)$$

Under most conditions of interest (nitrate concentrations greater than 3.5 mg Nitrate L<sup>-1</sup>) it has been suggested that reaction rates may be effectively modeled as zero order [Robertson et al., 2000; Schipper et al., 2005] When nitrate concentrations are considered rate-limiting, the reaction rate is considered first-order

$$f + NO_3 = \frac{v_{max}[NO_3]_{in}}{K_m} \quad (2.5)$$

First order reaction rates have been measured starting at minimum threshold nitrate concentrations of 1 - 3.5 mg Nitrate  $L^{-1}$  [Robertson 2010, Chun et al., 2009, Leverenz et al., 2010.]

The rate at which nitrate is reduced is controlled by a number of environmental factors besides reactant concentrations including oxygen availability, pH, temperature, and residence time [Shipper 2010, Christianson et al 2011]. Oxygen concentrations inhibit the rate of denitrification through direct competition (Oxygen being a more energetically favorable terminal electron donor) as well as through repression of the nitrate reductase enzyme at high concentrations [Metcalf & Eddy, Inc., 2003]. Because naturally ubiquitous aerobic organisms reduce oxygen provided an adequate feed source of organic carbon, the increased concentration of dissolved organic carbon in a denitrifying bioreactors serves the secondary role of promoting anaerobic conditions in which denitrification may occur. Robertson, 2010 showed that a column study mimicking a woodchip denitrifying bioreactors depleted dissolved oxygen in oxygen-saturated influent within one hour.  $pH$  is determined by soil properties and heterotrophic denitrification is considered ineffective at  $pH$  4. Temperature dependencies and hydraulic residence times are controlled by meteorological conditions on-site. Temperature effects on denitrification reaction rates can be modeled using the Arrhenius equation

## 2.3 Software and Previous Models

### 2.3.1 Drainmod

Drainmod [Skaggs, 1978, 1980b, Skaggs et al., 2012] is one of the most widely used and validated hydrologic models for the simulation of subsurface (tile) drainage systems and has been adopted by the USDA NRCS for design and evaluation of agricultural subsurface water management systems . Drainmod is a field-scale, process-based, distributed software model for vadose zone hydrology simulation of poorly drained or artificially drained sites. Simulations of field hydrology are performed through 1-D discretized computation of the following water balances:

- 1) A surface water balance taking the form

$$P = F + \Delta S + RO \quad (2.6)$$

where  $P$  is precipitation (cm),  $F$  is infiltration (cm),  $\Delta S$  is the change in water volume stored (pooled) at the surface , and  $RO$  is runoff (cm).

- 2) A subsurface water balance taking the form

$$\Delta V_a = D + ET + \Delta LS - F \quad (2.7)$$

where, for a given volume of soil,  $\Delta V_a$  is the change in the water-free pore space (or air volume) in the soil (cm),  $D$  is the artificial drainage from (or sub-irrigation into) the soil (cm),  $ET$  is evapotranspiration leaving the system

(cm),  $\Delta LS$  is the loss of water to deep and lateral seepage outside of the system boundaries (cm), and  $F$  is infiltration (cm) entering the soil.

Drainmod's handling of each of these terms in the water balance is discussed below.

## **Infiltration**

Infiltration parameters set the rate at which water may enter from the surface and fill the pore space of unsaturated soils. Like all environmental unsaturated porous flow phenomenon, infiltration is a complex processes with a laborsome set of parameters to be considered. Besides the hydraulic conductivity function of a soil as it would be measured in the lab, the infiltration rates depend upon initial water content and its distribution, surface compaction by recent field activities, swelling and shrinking of clays, plant cover, root depth, activity of burrowing creatures, and any other processes which serve to disturb the surface layers of soil from the ideal conditions of undeformable, deeply drained, and uniform. Additionally, it stands to reason that the intensity, duration, and distribution of rainfall determine infiltration rates. Temperature also plays an important role – both in terms of evapotranspiration rates and matrix changes resulting from freezing of the soil layers.

Generally speaking, a complete representation of infiltration would require the system to first be modeled as matrix flow in a well behaved soil by solving for the Richard's equation. Supplemental equations which account for violations of the the Richard's equation's assumption (a fixed and undeformable matrix with well-defined hydraulic conductivity and soil water characteristic functions and no macro-pore or preferential flow paths) would then need to be added. The Richard's equation for

vertical water movement takes the form

$$C(h)\frac{\partial h}{\partial t} = \frac{\partial}{\partial z} \left[ K(h)\frac{\partial h}{\partial z} \right] - \frac{\partial K(h)}{\partial z} \quad (2.8)$$

where  $h$  is the soil water pressure head,  $z$  is the distance below the soil surface,  $t$  is time,  $K(h)$  is hydraulic conductivity as a function of soil water pressure head, and  $C(h)$  is the 'specific moisture capacity'  $[1/L]$  which defines the rate of change of saturation with respect the matric head. In other words

$$C(h) = \frac{\partial \theta}{\partial h} \quad (2.9)$$

where  $\theta$  is the volumetric moisture content. Direct application of the Richard's equation is generally not a feasible option for simulation as it is computationally expensive, unpredictable in convergence, and requires a burdensome amount of input data. Moreover, applying the Richard's equation in this way assumes the soil system to be . In practice, infiltration is therefore modeled using one of many available simplifying numerical schemes. For a full discussion of soil physics and infiltration theory, see Morel-Seytoux and Kahanji [1974]. Additionally, imnek and Grdens [2003] discusses techniques for describing non-equilibrium and preferential flow in infiltration.

Drainmod uses the infiltration model developed by Green and Ampt [1911] and extended by Philip [1954], Mein and Larson [1973], and Reeves and Miller [1975] among others. The original model was developed for the highly idealized case of treating soil as a bundle of variably sized capillary tubes with a well defined 'wetting



front' (i.e. a well defined boundary between full saturation and desiccated soil or soil at the wilting point). Although the original conception is simple, the Green and Ampt model has empirically demonstrated adequate prediction power under a highly flexible range of situations including profiles that become denser with depth [Childs and Bybordi, 1969], partially sealed (crusted) surfaces [Hillel and Gardner, 1969], and soils with a nonuniform initial water content distribution [Bouwer, 1969]. The form of the Green and Ampt equation used in the Drainmod simulation is as follows

$$f = k_{vs} + \left( \frac{k_{vs} M S_{av}}{F} \right) \quad (2.10)$$

where  $f$  is the infiltration rate (the downward flux of water),  $k_{vs}$  is the effective vertical saturated conductivity,  $M$  is the initial air volume in the porespace,  $S_{av}$  is the effective suction at the wetting front, and  $F$  is the accumulated infiltration. Drainmod assumes infiltration rate to be equal to infiltration rate until the infiltration capacity (the rate  $f$ ) is exceeded, at which time it calculates infiltration with this Green and Ampt equation. It is important to note that infiltration affects the time step at which the Drainmod simulation proceeds. The surface water balance is performed on a variable time step ( $t$ ) in Drainmod – using  $t = 1$  hour as the standard time step,  $t = 1$  day during periods of no precipitation with slow drainage and evapotranspiration rates, and  $t = 0.05$  hours when calculating infiltration over periods in which rainfall exceeds infiltration capacity.

### Depression Storage

The change in depression storage ( $S$ ) is the change in volume of ponded water not

able to leave the system as runoff due to potholes, craters, crevices, and other local small-scale topography (see figure 2.5 below). Surface stored water may only exit the surface boundary condition (may only be subtracted from the surface storage mass balance) through infiltration or evapotranspiration. The amount of surface storage that occurs at a site is defined by two user defined parameters: maximum surface storage ( $S_m$ , and minimum storage for lateral surface distribution  $S_l$  (otherwise referred to as Kirkham's depth for flow to drains). Maximum surface storage is a user input parameter that represents the total depth of ponding that may occur at the surface before initiating runoff. Kirkham's depth represents the depth at which the majority of ponded water is no longer free to move laterally across a surface to fill space.

**Runoff** Runoff (RO) leaving the system is calculated as the total amount of water entering the system as precipitation (including irrigation) while both infiltration capacity and surface storage capacity are exceeded. Precipitation data are taken from a designated input file containing hourly precipitation measurements. Drainmod 6 also includes the possibility of considering contributing runoff. If contributing runoff is selected in the simulation options, input files listing the amount of water expected to enter the modeled system as runoff from land outside the simulated field are accepted. This is useful if the land of interest is at a local topological depression with runoff acting as a considerable input in the field water balance. These files can also be used to combine the simulation of several local fields into simulation of a larger watershed.

## Artificial Drainage Loss and Subirrigation

Finally, the real star of the Drainmod simulation software is its simplified ,but consistently accurate, scheme to model artificial drainage in the subsurface (tile drainage flow) and/or subirrigation through the flooding of these subsurface drains. In order to model the flow into/from subsurface drains in one dimension, the Drainmod simulation performs water balances at the midpoint between two theoretically parallel drains spaced a (user defined) length of  $L$  apart. The configuration considered by Drainmod is shown in figure 2.5.

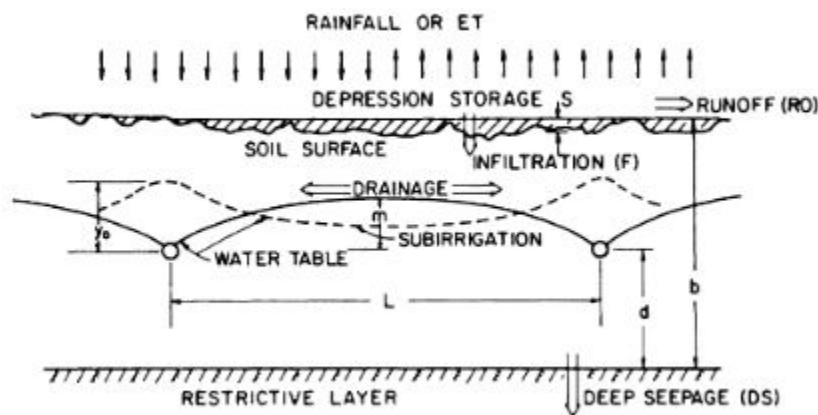


Figure 2.5: The Drainmod Simulation Scenario. Taken from Skaggs [1982b]

The vadose zone water balance is evaluated for water table elevation and moisture content using a one-dimensional vertical section at a point ( $m$ ) located midway between two subsurface drainage lines (see figure 2.5). When the water table is above the surface of the soil + the Kirkham's depth for flow to drains (the minimum depth for lateral surface movement of ponded water discussed previously), Drainmod calculates subsurface drainage using the equations developed by Kirkham [1958]. Namely,

$$q = \frac{4\pi k(t + b - r)}{gL} \quad (2.11)$$

where  $t$  is the average depression storage depth (cm),  $b$  is the depth of the tile drain from the soil surface (cm),  $r$  is the radius of the tile drain (cm), and  $g$  is a dimensionless factor.

When the water table drops below the soil surface, drainage is estimated using the Hooghoudt equation [Bouwer and van Schilfgaarde, 1963].

$$q = \frac{4K_e m(2d_e + m)}{L^2} \quad (2.12)$$

where  $q$  is the drainage flux in cm,  $m$  is the midpoint water table height above the drain (cm),  $K_e$  is the effective lateral saturated hydraulic conductivity (cm/h),  $L$  is the distance between drains (cm), and  $d_e$  is the equivalent depth of the impermeable layer below the tile drains (cm). Although the drainage rate violates the steady-state assumption of the Hooghoudt equation as the water table falls from the surface of the soil to the drain depth, Skaggs and Tang [1976] demonstrate that this process is

usually slow enough to be adequately estimated by the Hoodoudt equation. If the water table drops to the drain depth or below, drainage rate is taken to be zero.

**Drainmod NII** In addition to hydrology, the simulation of nitrogen species transformation and transport may be included in a Drainmod simulation using the model Drainmod-NII [Youssef et al., 2005]. Drainmod-NII is included with the Drainmod 6 distribution and can be used as a separate utility or integrated within the Drainmod 6 user interface. It is an expansion on the previously developed Drainmod-N Breve [1994], Breve et al. [1997] with the main upgrade being the consideration of a mineralized ammonium-nitrogen pool. The Drainmod-N and NII nitrogen simulations use a combination of transport equations and discretized mass balances of nitrogen species and carbon cycling to model the nitrogen cycle in a field. These simulations are able to consider common mineral fertilizer applications (urea, nitrogen/ammonium salts, and anhydrous ammonia), animal waste fertilization, plant uptake, organic carbon cycling, atmospheric deposition, natural nitrification/denitrification,  $NH_3$  volatilization and more [Youssef et al., 2005]. The Drainmod nitrogen modules use Michaelis-Menten based rates laws for nitrification and denitrification considering environmental constraints (temperature, pH, carbon content). Transport of nitrogen species is modeled using the one dimensional advection-dispersion equation. For the remainder of this thesis, the term Drainmod refers to both the Drainmod hydrology model and the Drainmod NII model unless otherwise specified.

**Conclusion** The mark of an exceptional environmental model is the ability to take a natural environmental process, always brimming with complexity, and choose

a set of highly simplifying assumptions that make the model attainable without loosing the accuracy necessary for planning nor the robustness needed for use under a reasonably wide array of conditions. In this respect, the one dimensional Drainmod is an exceptional model. Since its introduction in 1978, the Drainmod scheme for simulating flow into subsurface drainage has shown repeated success in a variety of different soils and locations [Skaggs et al., 1981, Skaggs, 1982a, McMahon et al., 1988, Broadhead and Skaggs, 1989, Cox et al., 1994, Singh et al., 1994, Gupta et al., 1993, Shukla et al., 1994, Fouss et al., 1987a, Dean and DeBoer, 1988, Skaggs et al., 1991, Skaggs, 1980a, Evans and Skaggs, 1989, Madramootoo, 1990, Shirmohammadi et al., 1991, Cris et al., 1993], a variety of drainage conditions and irrigation systems [Fipps and Skaggs, 1989, Konyha and Skaggs, 1992, Kandil et al., 1993, Chang et al., 1983, Lorre et al., 1994, Chescheir et al., 1992, Amatya et al., 1995, Skaggs, 1976, Massey et al., 1983, Fouss et al., 1987b] and at predicting hydrology for a number of row crop needs [Ravelo et al., 1982, Skaggs et al., 1982, Evans et al., 1990, Evans and Skaggs, 1993, Seymour et al., 1992, Gayle et al., 1985, Abdel-Dayem and Skaggs, 1991, Skaggs, 1990, Madramootoo et al., 1995], nitrate transport and water quality modeling [Skaggs et al., 1993, Kandil et al., 1992, Parsons et al., Helwig et al., 2002, Skaggs and Gilliam, 1981, Skaggs and Nassehzadeh-Tabrizi, 1982, Skaggs and Gilliam, 1986, Deal et al., 1986]

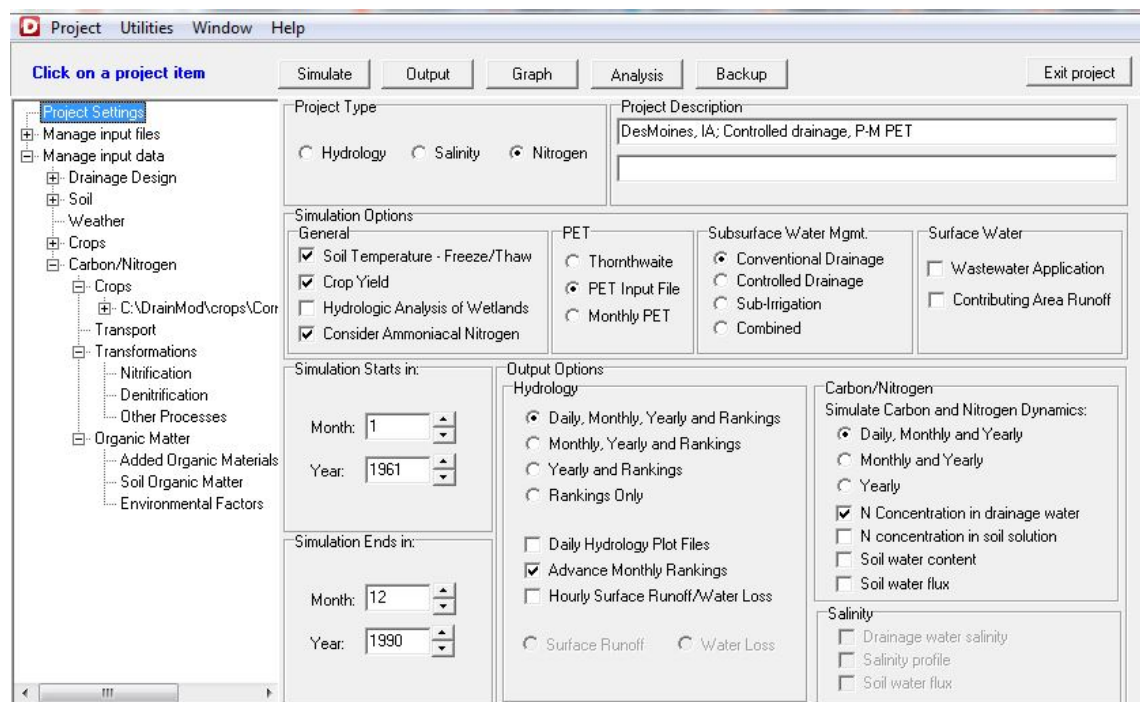


Figure 2.6: Screenshot of the Drainmod 6 Native Interface

### 2.3.2 Rosetta

Rosetta [Schaap et al., 2001] is a computer program that estimates soil hydraulic parameters based upon pedotransfer functions. Rosetta is supported by the USDA NRCS and is highly popular in the united states for large scale studies or first guess approximations where limited soil data is available. Rosetta predicts the Van Genuchten Mualem soil hydraulic parameter using five hierarchical pedotransfer functions (PTFs) for estimating water retention, saturated conductivity, and unsaturated hydraulic conductivity functions basted upon the Van Genuchten-Mualem [1980] equation. The Van Genuchten-Mualem model is a is a widely used approximate solution to the Richard's equation which takes the following form

$$\theta(h) = \theta_r + \theta_s - \theta_r [1 + (\alpha h)^n]^{1-1/n} \quad (2.13)$$

where  $\theta(h)$  is the measured volumetric water content (cm<sup>3</sup> cm<sup>3</sup>) at the suction  $h$  (cm, taken positive for increasing suctions). The parameters  $\theta_r$  and  $\theta_s$  are residual and saturated water contents, respectively, (cm<sup>3</sup> cm<sup>3</sup>);  $\alpha$  ( $>0$ , in cm<sup>-1</sup>) is related to the inverse of the air entry suction, and  $n$  ( $>1$ ) is a measure of the pore-size distribution. Rosetta is based on neural network analyses combined with the bootstrap method, thus allowing the program to provide uncertainty estimates of the predicted hydraulic parameters. Drainmod includes a utility for using the Van Genuchten predictions of Rosetta in order to build the majority of soil required inputs required. The combination of Rosetta and Drainmod have shown continuous success Osvaldo Salazar [2008], Qi et al. [2015b], Abdelbaki and Youssef [2010], Singh et al. [1994]



### 2.3.3 PEST

PEST is a model-independent parameter estimation and uncertainty analysis tool using, at its core, the Levenberg-Marquart method for minimizing residual error [Doherty et al., 1994]. It works by generically interfacing with models through template files (which allow it to alter input parameters) and instruction files (which tell it how to read output files). The full capabilities of PEST and its associated utilities is continually growing and beyond the scope of this paper. Interested readers should consult the PEST website [pes], the most current version of the PEST and PEST utility manuals.

For its flexibility and robust calibration procedures, PEST has been integrated into several popular models including Root Zone Water Quality Model [Ma et al., 2001], Modflow Flex (Waterloo HydroGeologic Inc.), and the Soil and Water Assessment Tool [Arnold et al., 2012]. PEST has also been used successfully to improve Drainmod models [Singh et al., 2006, Qi et al., 2015a]. Singh et al. [2006] used a combination of the the Iowa Soil Properties and Interpretations Database [Miller et al., 2010] and Rosetta predictions to build a Drainmod simulation for Webster and Canisteo soils in Iowa under Corn and Corn/Soybean Rotation. They went on to validate to calibrate the model using PEST for optimal settings of the Van Genuchten fitting parameter  $\alpha$ , saturated conductivity, and lateral saturated conductivity. 27 monthly subsurface drainage measurements were calibrated against, and 31 additional monthly subsurface drainage measurements outside the calibration period were used as validation. Results of the validation period revealed strong correlation; total root mean square errors (RMSE) were 2.02 and 2.21 with Nash Sutchliffe efficiency

(NSE) values of 0.56 and 0.67 for the Webster and Canisto soils respectively.

### 2.3.4 Protocol and Interactive Routine for the Design of Subsurface Bioreactors (PIRDSB)

The denitrifying bioreactor component included with this model has been adapted from a standalone model previously developed by Cooke and Bell [2014]. The Cooke and Bell model simulates physical flow through a proposed bioreactor and reports values for residence time within the bioreactor and the percentage of flow that bypasses the system untreated. The rate of discharge into the system can be set as a constant (steady state) or can be set to accept daily subsurface drainage discharge values from a Drainmod simulation output file. The Cooke and Bell model allows for optimization of design parameters (area) and operation parameters (adjustable stop log heights of inlet and outlet control structures) by setting user defined constraints on residence time and/or maximum allowed percentage of discharge bypassing the system at peak flow events. The Cooke and Bell model also contains helpful subroutines for cost analysis and report generation. The model has been slightly modified to accept daily nitrate concentration values from Drainmod-NII simulation output files.

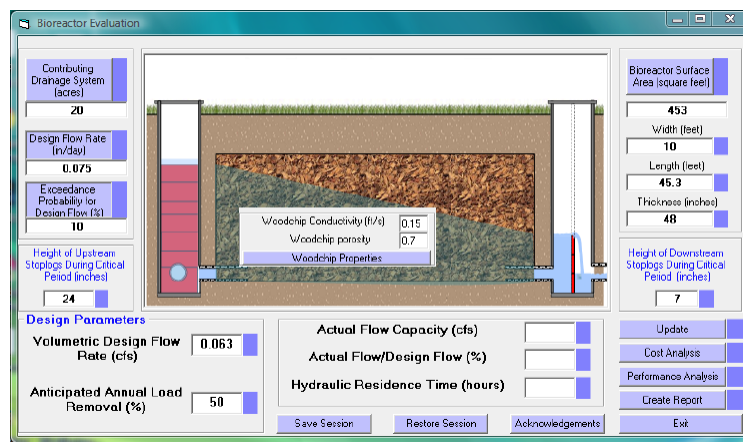


Figure 2.7: Screen Captures of the PIRDSB Model

## 3 Materials and Methods

### 3.1 Model Overview

The developed model can be separated into two major components: a field model component and a denitrifying bioreactor model component (see figure 3.1). The field component predicts daily discharge rates and nitrate concentrations in a subsurface (or tile) drainage system and is the major concern of this project. After the field component simulation has been completed, the predicted daily discharge and nitrate concentrations of the subsurface drainage effluent are used as input data for the denitrifying bioreactor component. The denitrifying bioreactor component predicts the amount of nitrate removed by the bioreactor as well as the amount of nitrate exiting the system untreated. Optionally, the denitrifying bioreactor component may also optimize design and/or operation parameters for maximum efficiency. The automated workflow currently uses the PIRDSB model as the bioreactor component of the modeling framework.

#### 3.1.1 User Interface

The user interface takes the form of a macro-enabled Microsoft Excel file. The workflow is automated by event-oriented subroutines written in Microsoft Visual Basic for Applications (VBA) and included with the Excel Interface file. The user

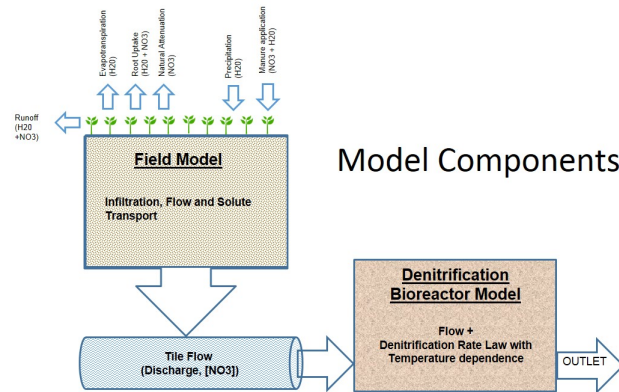


Figure 3.1: Graphical Representation of the two major model components. The field component models daily discharge rates and nitrate concentrations in subsurface (tile) flow. The denitrifying bioreactor component (PIRDSB) models the daily mass of nitrate removed and the daily mass of nitrate exiting the system untreated.

interacts with the VBA automated workflow from the Excel worksheet view via VBA form controls and text entry into designated worksheet cells. The VBA scripts are compatible with both VBA 6 and VBA 7 and the interface is compatible with both 64 and 32-bit releases of Microsoft Excel (only tested on 2010 or later). The model was designed to be run on Microsoft Windows operating systems (Vista and later). The workflow is designed to allow a flexible amount of automation. The user workflow is as near fully automated by default, which allows users to seek a first guess or rough estimate of a proposed denitrifying bioreactors performance with minimal site data and effort. This is useful if the user is inexperienced, if the user has little data on hydraulic parameters for a given site, and/or if the user is only performing an initial feasibility study. At various points in the main routine, the workflow is halted and the input data / options for the following subroutine(s) are displayed on the active worksheet. At these breakpoints, the user may optionally

edit input information and options as they see fit (for example, if the user would like to substitute empirically measured soil hydraulic parameters). Finally, the workflow includes an optional semi-automated field component calibration procedure. This calibration procedure uses any empirical measurements of discharge and/or nitrate concentrations in the subsurface drainage outlet to calibrate the most sensitive input parameters of the field component. The semi-automated calibration procedure was included with the following five-step scenario in mind: 1) The user creates a first guess feasibility simulation by running through the entire field and denitrifying bioreactor components using the default automated workflow. 2) Based upon results of the first guess feasibility simulation, the user decides if a denitrification bioreactor seems like an attractive treatment option they would like to explore for their site. 3) If so, the user collects discharge and/or nitrate concentration measurements over a period of time which includes at least one data point with subsurface drainage flow. 4) The user runs the semi-automated calibration of the field component using their collected measurements until satisfied with the predicted level of uncertainty. 5) The user proceeds again to the denitrification bioreactor component of the model using the calibrated field component results and determines the design and operation parameters that are predicted to be the most efficient for their needs.

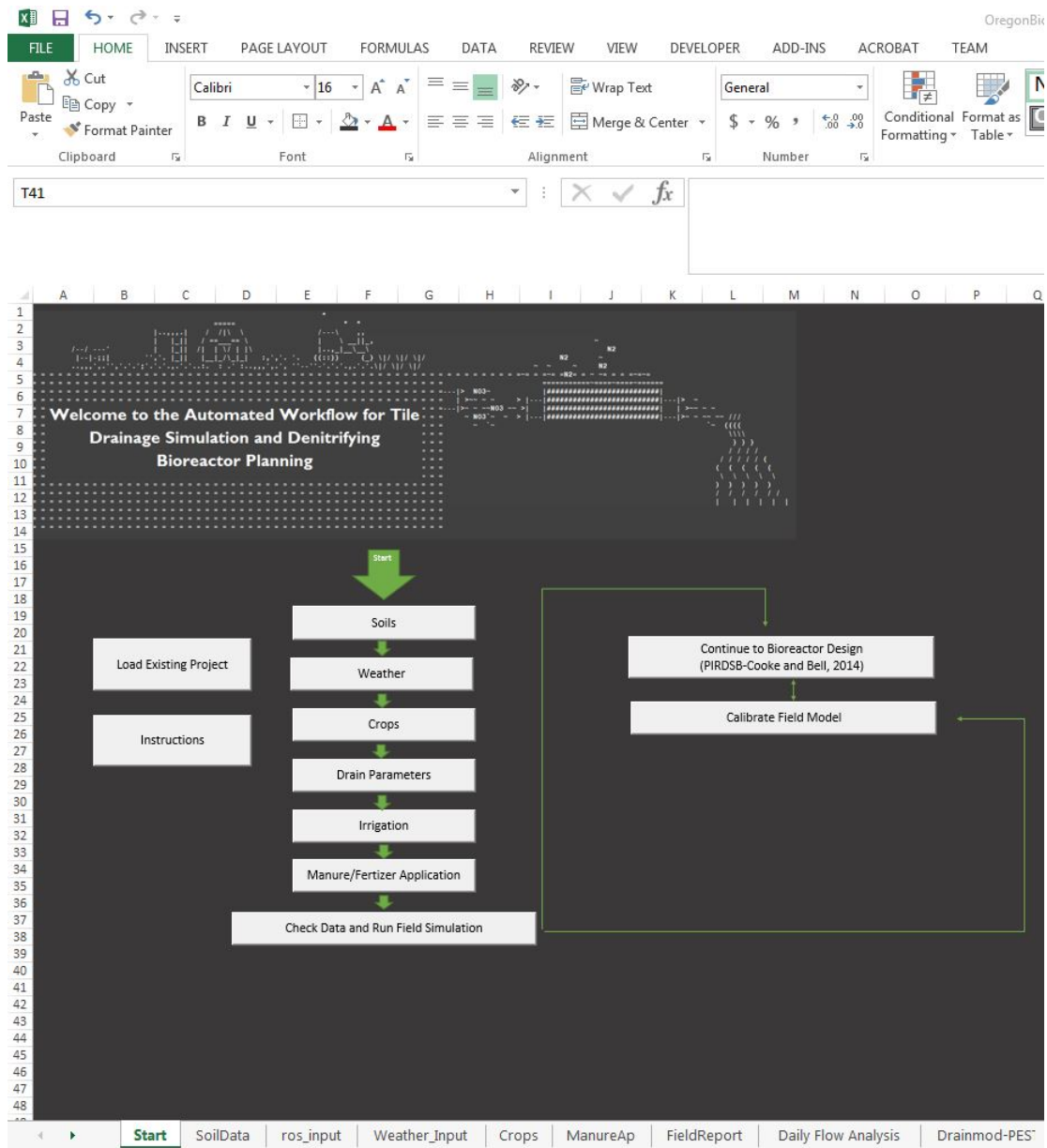


Figure 3.2: Screen Capture of the Automated Workflow Start Page

## 3.2 The Field Component

### 3.2.1 Soils

Soils data is a logical place to start in creating a simulation because soil parameters represent the most site-specific class of Drainmod input data at the field scale. Drainmod accepts three types of soil input files for simulation. The first, distinguished by the extension `.SIN`, is a general soil file that is required in all Drainmod hydrology simulations. The SIN file stores general soil data including soil water characteristic curve data, the water table vs volume drained relationship, upward flux relationships, and Green-Ampt infiltration parameter values. The second soil file, distinguished by extension `.MIS`, stores more detailed information on the soil water characteristic curve. Finally, a `.WDV` soil file stores information on the relationship between the water table elevation and the volume drained. Skaggs [1978] recommended the use of soil survey data supplemented with physical measurements of soil parameters in the creation of soil input files. Due to the complexity and spatial heterogeneity of the soil parameters, even extensive testing of soil parameters may fail to correctly predict the hydrology of a site if the input parameters are not calibrated [Skaggs et al., 2012]. In the case where a tile drainage system exists prior to the simulation, we propose that the tedious and costly process of physically measuring soil parameters may be omitted completely in the simulation. Instead, soil survey parameters are used to create a set of initial guess parameters and the model is calibrated using discharge measurements taken at the outlet of the subsurface drainage system. All parameters included in the `.SIN`, `MIS`, and `.WDM` input files may be



estimated using the Van Genuchten-Mualem parameters. Drainmod 6 includes a utility program (DMSOILP.EXE) which is capable of creating properly formatted SIN, MIS and WDM files using the Van Genuchten-Mualem parameters. The utility accepts a text file, distinguished by the extension .ROS, which contains the number of layers, the depth of each layer, and the Van Genuchten-Mualem parameters for each layer. Although soil surveys rarely contain values for all parameters required by the Van Genuchten-Mualem method, common characteristics reported in soil surveys may be used to predict Van Genuchten-Mualem parameters using the hierarchical pedotransfer and neural network software Rosetta [Schaap et al. 1998, 2001].

Included with the folder containing the automated workflow model is a folder containing Rosetta estimated Van Genuchten parameters for each unique soil layer in the United States that has all required entries published in the Soil Data Access online soils database. The Rosetta predictions are separated into files by state named by the two digit Soil Data Access state identifier (e.g. OR for Oregon, TX for Texas, etc.). The procedure for creating these Rosetta predictions files was as follows: 1) The Soil Data Access online database was queried by state for the input parameters used by Rosetta to predict Van Genuchten parameters. These include representative %sand %clay %silt bulk density volumetric water content at 1/3 bar and at 15 bar. The chkey was also included in the query as a layer identifiers. 2) The queried data were pre-filtered for entries containing missing parameter data. Often layers missing data for a parameter are layers that are not traditional soils. All layers that contain missing data on the parameters necessary for Rosetta prediction were deleted from the file in the pre-filter stage 3) The file was formatted for Rosetta input and Rosetta

was run 4) The Rosetta output data was post-filtered, removing all layers with null value predictions from Rosetta (flagged as values of -99.9 by Rosetta).

To begin the process of creating soil input files for a site, the end user navigates to the Web Soil Survey online GIS tool. The user navigates the GIS mapping tool to locate the site for which a proposed bioreactor is (to be) installed. Using the Area of Interest (AOI) selection tools, the user outlines a best guess for the spatial extent of the subsurface drained area contributing to the subsurface drainage outlet at which the (proposed) bioreactor is (to be) installed. Once satisfied with the selected AOI, the user downloads the soils information for the AOI as a Microsoft database (.mdb file) from the WSS website. The Excel VBA automated workflow connects to the download WSS database and runs an SQL query for information relevant to the model as well as information relevant for quality control and report creation. For each unique chkey value contained in the WSS database, a corresponding Rosetta prediction is searched for in the included Rosetta prediction files. Chkey is simply a soil layer identifier used by the online soil surveys. Each unique soil layer in the soil survey database has a unique chkey associated with it. Rosetta inputs and layer widths are then used to create a ROS file for each unique soil series in the WSS database and run it then the soil utility to create all required Drainmod soil input files.

### **Handling Drainmod's 5 Layer Limit**

Drainmod is able to run simulations on a maximum of 5 soil layers at a time beyond which no memory is assigned to hold layer properties. Soil series , or Soil 'Map Units', reported in the SSURGO/STATSGO online databases often consists of

more than 5 layers, however, and hence there is a need to systematically aggregate reported layers into a 5 layer simulation. The aggregation scheme in this routine simply performs a depth-weighted average of all values between the two consecutive layers with the least difference (smallest absolute residual value) in matching point conductivity and repeats the process until there are 5 layers for each soil series. Matching point conductivity was chosen to determine which layers are aggregated due to its strong influence on initial simulations (it influences infiltration and acts as a first guess for lateral conductivity).

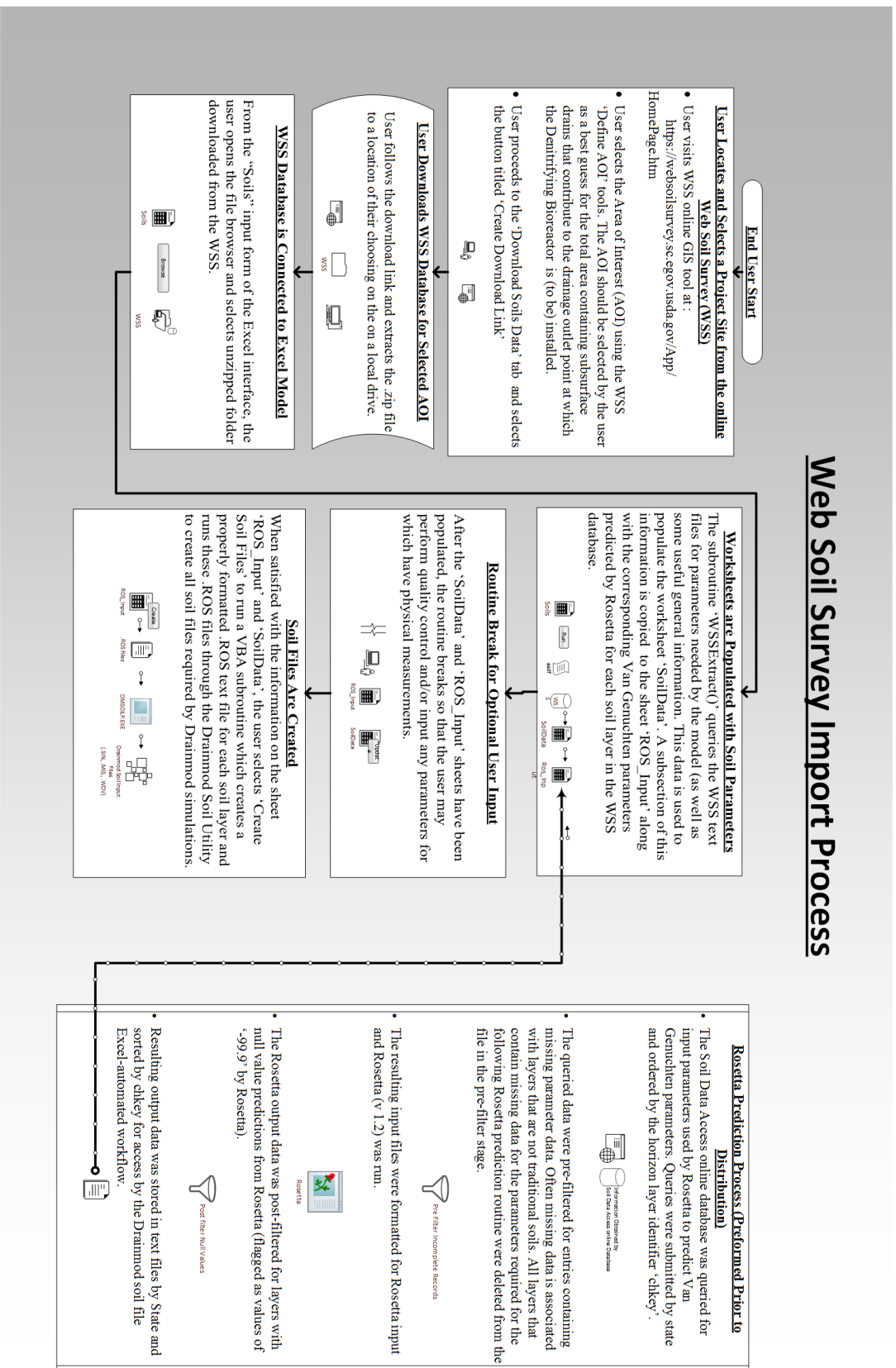


Figure 3.3: A flowchart describing the Online Soil Survey/Rosetta Import Procedure

### 3.2.2 Weather

The weather class of Drainmod input files consist of mandatory temperature (.TEM) and precipitation (.RAI) files with an optional potential evapotranspiration (.PET) file. The TEM file consists of daily minimum and maximum air temperatures. The .RAI file consists of hourly precipitation values (not including irrigation). The PET file consists of daily potential evapotranspiration values. If a PET file is not included, a latitude and longitude is requested and potential evapotranspiration is estimated using the Thornthwaite method [Thornthwaite, 1948, Thornthwaite and Mather, 1957] which is the Drainmod default potential evapotranspiration routine. The Thornthwaite method is an empirically derived model based upon mean monthly temperatures and takes the following form:

$$PET = 1.6 \left( \frac{L_{day}}{12} \right) \left( \frac{N}{30} \right) \left( \frac{10T_{avg}}{I} \right)^\alpha$$

*where*

$$I = \sum_{i=1}^{12} \left( \frac{T_{avg}}{5} \right)^{1.514} \tag{3.1}$$

*and*

$$\alpha = (6.75E^{-7})I^3 - (7.71E^{-5})I^2 + (1.79E^{-2})I + 0.49239$$

where PET is potential evapotranspiration on a daily basis (cm) is on a daily basis,  $L_{day}$  is actual day length in hours, N is the length of the month being calculated in days, I is a heat index which depends upon 12 month mean temperature, and alpha is a fitting parameter dependent upon I.

If daily rather than hourly precipitation values are provided, the Drainmod 6 weather utility will distribute the daily precipitation over a user-provided time interval centered upon a user-defined time of day. For the sake of simplicity and minimizing the number of user input parameters, the automated work flow presented here always distributes daily precipitation across a 12 hour period beginning at hour 3:00. This choice of precipitation period and starting hour was not made thoughtlessly. If the dispersion period is too short (say 6 hours), large continuous rainfall events will be forced to occur within a short period of time. This situation has potential to overestimate runoff and underestimate drainage as infiltration capacity is reached in the simulation without allowing time for field drainage to occur. Because these large continuous rainfall events often lead to large peaks in discharge rate (and potentially nitrates) choosing such a small precipitation distribution period is unacceptable. Conversely, choosing too large of a distribution period will spread smaller, periodic precipitation into a low amplitude continuous pattern that can alter infiltration rates and overestimate the losses to evapotranspiration. The compromise chosen here is substantial precipitation distribution period of 12 hours with the bulk of the distribution period (3am-3pm) occurring in the cooler morning hours of the day. Of course, using reliable hourly precipitation is suggested where available.

Depending upon the location and the users needs, weather data may be available from a variety of sources and in a variety of formats. The automated workflow program allows users to either input data manually into the excel worksheet `Weather_Inputs` or to import data from a comma delimited weather file downloaded from the PRISM climate group (<http://prism.oregonstate.edu>) into the

### Weather\_Inputs worksheet.

Day of year (1-365)	Date	ppt (inches)	tmin (degrees F)	tmax (degrees F)
1	January 1, 1982	0.65	27.1	30
2	January 2, 1982	0.39	26.4	29.6
3	January 3, 1982	0.85	28.5	30.9
4	January 4, 1982	2.48	24.9	30.2
5	January 5, 1982	0.73	19.9	28.2
6	January 6, 1982	0.10	16.2	24.7
7	January 7, 1982	0.00	18.3	27.4
8	January 8, 1982	0.00	24.2	32.2
9	January 9, 1982	0.00	30.6	36.4
10	January 10, 1982	0.00	31.2	37
11	January 11, 1982	0.10	29.1	34.7
12	January 12, 1982	0.10	29.4	34.3
13	January 13, 1982	0.04	30.3	37.4
14	January 14, 1982	0.00	31.2	38.1
15	January 15, 1982	0.06	31.6	35.5
16	January 16, 1982	0.16	32.2	36.6
17	January 17, 1982	1.52	32.2	35.7
18	January 18, 1982	1.00	29.4	33.9
19	January 19, 1982	0.49	27.6	32.1

Figure 3.4: The Weather\_Inputs worksheet Interface for entry and quality check of weather values.

When the user is satisfied with the weather data on sheet Weather\_Inputs, the "Continue with This Weather Data" button calls the subroutine Create\_Weather\_Inputs. Create\_Weather\_Inputs writes the weather data into intermediary text files and runs these text files through the Drainmod 6 weather utility DMWEA.EXE. Intermediary text files are helpful here because, unlike the Drainmod input files themselves, the intermediary text files are not fixed-width formatted and therefore have less potential for errors caused by errant characters and whitespaces that may corrupt the ability of Drainmod to read correct values.

Much as the soils data defines the spatial extent of the simulation, the weather data defines the temporal extend of the simulation. As it runs,the VBA subroutine Create\_Weather\_Inputs records the beginning and ending dates for the weather data

Import Data from the PRISM climate Group

**PRISM Climate Group** **PRISM Weather Import** **Instructions**

This page imports daily values for precipitation, maximum temperature, and minimum temperature from a .csv file downloaded from the PRISM climate group. If this is your first time using this page, please see the instructions (button above). The website for the PRISM climate group data explorer is here:

<http://www.prism.oregonstate.edu/explorer/>

Browse for Downloaded PRISM .csv File

Import Data From PRISM .CSV File Above

Go Back to Main Weather Page

Figure 3.5: The PRISM climate group import User Form.

in worksheet Weather\_Inputs, writes (or updates) their value in the associated general hydrology (.GEN), and stores the starting and ending dates in the automated workflow tagfile (TAG). It is recommended that users input at least 10 years worth of weather data for simulations in order to minimize the impacts of any initial condition errors (for example, errors in initial water table depth). Because it is often that available daily precipitation historical data predates available hourly precipitation historical data, and because of the potential adverse effects artificial dispersion of rainfall data discussed previously, the automated workflow includes a tool "Append Weather Data" that allows weather data to be appended to the end of existing RAI and TEM files. Using this tool, hourly precipitation data may be appended to the



end of existing artificially dispersed daily precipitation data.

### 3.2.3 Drainage Parameters

Once again, the figure considered in drainage design is shown if figure 3.6. With this system in mind, there are a few basic parameters that we need to collect from the user in order to build a simulation including: drain depth, drain spacing, drain type (effective radius), and a first guess for storage depths and initial water table depth. These are collected through the user form ‘Drainage Parameters’ shown in figure 3.7.

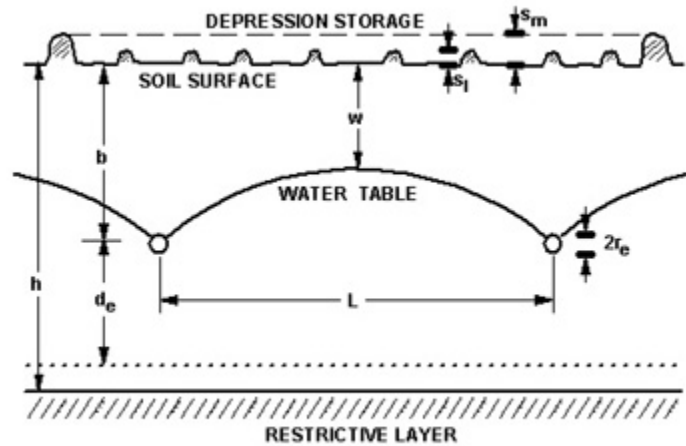


Figure 3.6: Diagram Illustrating Drainage Parameters. Taken from Skaggs et al. [2012]

Drainage Parameters ☰

Please enter the following drainage parameters for your site.

Drainage Depth (cm)

Drainage line Spacing (cm)

Drain Type / Radius (cm)

Maximum Surface Storage (cm)

Threshold depth for local storage water movement(cm)

Best Guess for Initial Water Table Depth (cm) on January 1




Figure 3.7: Drainage Design Input Form

### 3.2.4 Crops, Fertilizer, and Irrigation

Drainmod was originally developed for drainage system planning in row crop production. Drainmod therefore incorporates a crop yield component that allows users to predict how a given drainage system will effect crop production. Building a crop yield input file from scratch in Drainmod may be a tedious task due to the amount of parameters required to be explicitly defined. Figure 3.8 shows the many crop parameters that require explicit definition in the native Drainmod 6 interface. The crop yield simulation in Drainmod is not a robust predictor of crop production itself, rather the model simply applies user defined stresses to crops when it predicts the conditions for those stresses (also user-defined) to be met. Stresses considered by the hydrology component are limiting and excess water stress within the root zone. Stresses considered by the Drainmod NII model are availability of fixed nitrogen in the root zone. Both  $NO_3$  and  $NH_4$  in the root zone are considered available to plants.



Figure 3.8: Screen Capture of Explicit Crop Parameters Required of Drainmod

In order to streamline the process, it was decided that the quality of crop yield simulations would be sacrificed somewhat in exchange for fewer parameters required of the user. A rough simulation of crop yield is necessary in order to account for the effects of crops on the nitrogen and water balances, but this rough simulation does not necessitate the number of variables requested by the native Drainmod 6 crop yield utility. Users seeking a subsurface drainage simulation for denitrifying bioreactor planning that are unfamiliar with the sensitivities associated with each crop variable in the Drainmod model may over- or under-estimate the importance of a crop variable on drainage discharge / nitrate concentrations; this may lead to hesitation and abandonment of the simulation or poor simulation results. The focus of the crop routine in this automated workflow is to aid in the situation just described by properly estimating important parameters for simulating drainage discharge rate / nitrate concentrations and ignoring or providing little attention to the nuances of crop production which have little to no observable effect on these variables of interest. If the user requires detailed crop yield estimations, they may supplement a simulation created through the automated workflow by opening the project in the Drainmod 6 native interface and adding or modifying crop yield input files (distinguished by the extension \*.CIN) to include all relevant parameters as shown in figure 3.8. Alternatively, the ‘crops’ user form of the automated workflow includes a subroutine for adding previously created Drainmod crop yield input files rather than creating a new one.

Seven crop types are currently considered by the automated workflow: corn, soybean, corn/soybean rotations, wheat, pasture, grassland/generic crops, and legumes.

Parameters for the row crops mentioned (corn, soybean, corn/soybean rotations, and wheat) were taken from previous studies and the only user information requested for crops are the planting and harvest dates. The planting and harvest dates provided by the user are used to adjust time dependent variables (namely root depth, seedbed preparation period, and first/last possible dates to plant crops without loss to yield). The crop parameters for these row crops are otherwise unchanged. Routines for pasture, grassland, generic crops, and pasture-incorporated legumes are created through the workflow with the aim of reasonable estimates of root depth and dry matter yield throughout the year.

**Grassland and Generic Crops** The default case for modeling crop water and nitrate uptake is the ‘Grassland and Generic Crop’ option shown in figure 3.9. In this case, a single ‘crop’ is modeled on a yearly rotation with a highly simplified sinusoidal rooting depth equation as demonstrated in figure 3.10. In addition to the maximum and average root depth estimates (required by all simulations to define a root zone) the Grassland/generic crop model also asks for a month of maximum growth (to set the sinusoidal amplitude), the root to shoot ratio (to set the total dry matter yield based upon the root growth), and an optional % of dry matter harvested in a year. The % of dry matter harvested is simply added to the dry matter yield equation. This simplified model is the only one verified in this study.

Crops

- ☐ Pasture
- ☐ Corn
- ☐ Soybean
- ☐ Corn/Soybean
- ☒ Grassland/ Generic Crop

Month of Max Yield

Root to Shoot Ratio

% Yield Harvested in a year?

Continue With These Values

Figure 3.9: The User Interface to the Highly Simplified Grassland/Generic Crop Procedure

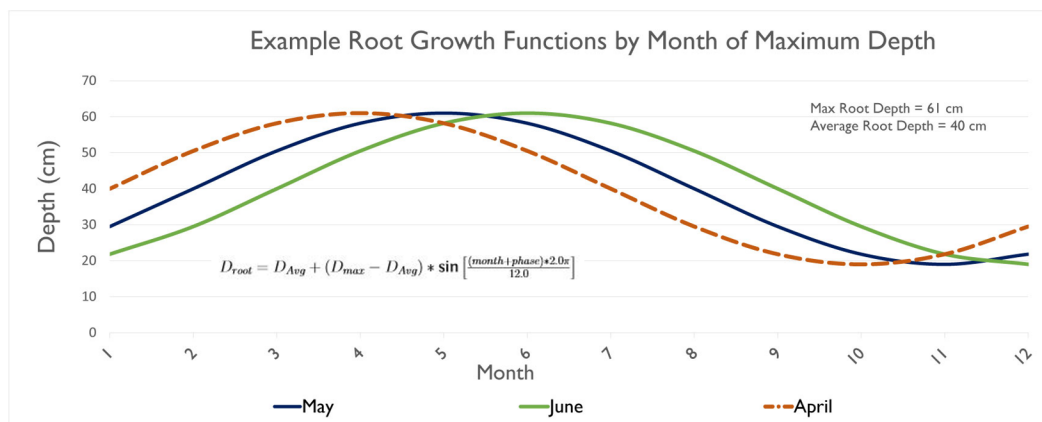


Figure 3.10: An Example Sinusoidal Root Growth for 3 Max Height Months

**Pasture** Pasture models are an important case for CAFO operators spreading animal waste fertilizers. In general, pasture does an excellent job of nitrate uptake when established and well maintained, however denitrifying bioreactors may act as a defense against unexpected rainfall events washing out nitrate too quickly for uptake by pasture. The water stress model included with Drainmod is only equipped to model row crops and is not adequate to model pasture directly. An experimental work-around procedure was adapted as part of this automated workflow from a study by Bechtold et al. [2007] in which pasture uptake of nitrogen was simulated by treating each cut of the pasture as a separate crop and highly simplifying hydrology parameters by setting them constant (or near-constant) through each cut period. Yield is set based upon on the user knowledge of expected height of pasture at cutting times and the commonly used value of 200 lb per acre per inch of dry matter is used. Also included a grazing term which acts as a sink of dry matter yield using the USDA livestock coefficients based on livestock type, the number of grazing animals, and the length of grazing. In this case, the amount of dry matter yield expected to have been consumed by livestock is added to the dry matter yield equation. Legumes are often incorporated in to pasture for reasons including increased pasture health and better foraging quality. Because Legumes have the ability to fix nitrogen, their influence needs to be taken in to account if they are well established. In the automated workflow, a generic legume file was modeled after clover to have the option of incorporation into pasture. The rooting depths are taken to be constant and the legume yield is not considered in grazing or cuts. This pasture routine is entirely theoretical in nature and has not be verified in this study.



**Irrigation and Manure/Fertilizer Applications** The Automated workflow follows the Drainmod standard interface procedure for defining irrigation schedules and manure applications. The automated workflow interfaces for these tasks are shown in figure 3.11

### 3.2.5 Automated Calibration With PEST

After an initial first-guess field simulation has been created, the user is encouraged to proceed to the PIRDSB [Cooke and Bell, 2014] with simulation results for an initial feasibility study on how a denitrifying bioreactor will work for their situation. An excel version of the PIRDSB made available by Dr. Richard Cooke of The University of Illinois at Urbana-Champaign is included with automated workflow. It has been slightly modified to accept nitrate concentrations from the automated workflow's field model. This interface is discussed in some detail in the following section.

If the end user finds that the results of the first-guess feasibility simulation satisfactory and is still interested in pursuing a denitrifying bioreactor at their site, the simulation will need calibration if any trust is to be placed in its results. Calibrations are performed by the automated workflow through the creation and running of a PEST calibration using pre-chosen settings and variables. Calibration data accepted for use in the calibration procedure are daily subsurface discharge measurements (for hydrology) and measured nitrate concentrations in subsurface drainage (for nitrate). The automated calibration user interface is shown in figure 3.12.

A	B	C	D	E	F
<b>Fertilizer/Manure Applications</b>				<b>Continue with this data</b>	
Please enter fertilizer and manure applications below. You may enter up to 50. Leave unused applications blank. Do not skip applications (e.g. if you enter info for Application 1, you must enter info for Application 2, etc.).					
The exact date of fertilizer application is only used in calibration. Otherwise, the model finds the next period with no considerable rainfall in 2 dry weeks after the application.					
Enter month as a number (1-12). Enter 4 digit year.					
<b>Application 1</b>					
Month	Day	Year (yyyy)	Type	Amount (kg/hectare)	
1			Ammonium		
<b>Application 2</b>					
Month	Day	Year	Type	Amount (kg/hectare)	
			Ammonium		
<b>Application 3</b>					
Month	Day	Year	Type	Amount (kg/hectare)	
			Ammonium		
<b>Application 4</b>					
Month	Day	Year	Type	Amount (kg/hectare)	
<b>Application 5</b>					
Month	Day	Year	Type	Amount (kg/hectare)	
			Anhydrous Ammonium		
<b>Application 6</b>					
Month	Day	Year	Type	Amount (kg/hectare)	
<b>Application 7</b>					

Start SoilData ros\_input Weather\_Input Crops **Fertilizer** FieldReport Daily Flow Analysis Drainmod-PEST Sheet1

---

**Irrigation**

Please enter information for up to three distinct annual irrigation periods. If a period is not used, leave it blank or set the application rate to 0 cm/hr.

**Approx. Irrigation Rate (cm/hour)**

Jan	0.4
Feb	0.4
March	0.4
April	0.4
May	0.4
June	0.4
July	0.4
Aug	0.4
Sept	0.4
Oct	0.4
Nov	0.4
Dec	0.4

**Irrigation Start Date**

Month Day

1

**Irrigation End Date**

Month Day

**Irrigation Period # 1**

**Irrigation Period # 2**

Minimum Rainfall to Delay Irrigation (cm):

Approx. Number Days Between each Irrigation:

Typically irrigate from to

**EXIT**

**Continue with this Data**

Figure 3.11: Screen Capture of Crop Irrigation Scheduler

Please insert the observations (empirical measurements) you have obtained to calibrate (a) Drainmod Simulation(s) below. You may enter however many you like. You may leave Drain flow or nitrate concentration fields blank if you do not have one or the other for a given date but DO NOT SKIP ROWS IN THE DATE COLUMNS. Dates must begin on cell C3 and any columns skipped in column C will cause the program to break. Additionally, if a date is listed in column C either a Drainflow or Nitrate measurement must be entered or the program will return an error.

Launch PEST File Utility for Drainmod

Date (#M/#D/YYYY)	Drain flow (cm of water/day)	Nitrate Concentration (mg/L)
1/16/01	0.10378382	
1/17/01	0.23444626	
1/18/01	0.229097229	
1/19/01	0.224909431	
1/20/01	0.235242835	
1/21/01	0.206609228	
1/22/01	0.235418334	
1/23/01	0.258128072	
1/24/01	0.198747518	
1/25/01	0.33292939	
1/26/01	0.325734995	
1/27/01	0.311571096	
1/28/01	0.298970922	
1/29/01	0.23824733	
1/30/01	0.24177622	
1/31/01	0.252879926	
2/1/01	0.259531011	
2/2/01	0.238265196	

Calibrate Nitrate?

SoilData | ros\_input | Weather\_Input | Crops | ManureAp | FieldReport | Daily Flow Analysis | **Drainmod-PEST** | She ... (+) :

Figure 3.12: Screen Capture of Automated Calibration Interface

When calibrating a Drainmod simulation, it is important to heed the warning of Skaggs [1982b] that calibrating too many parameters at once is dangerous and will not necessarily correlate with an increased prediction power of the model. The equations in Drainmod represent a physical system and mathematically fitting a number of variables to match observations can easily (and insidiously) lead to relative variable choices that are not physically feasible or possible. Additionally, the choice of calibration parameters must be done with care so that the calibration favors (as much as possible) a global minimum of error rather than a highly local minimum. The effectiveness of a calibration will always ultimately depend upon the quality, quantity, and breadth of calibration data used—however we can set reasonable limits and choose our calibrated terms carefully as to avoid excessive failures due to bad calibration points.

Skaggs et al. [2012] compiled an exhaustive list of Drainmod simulation variables, representative values, and their relative utility in calibration which has been reproduced in figure 3.13. Skaggs [1980b] tested the sensitivities of Drainmod simulations to hydraulic conductivity, water content at wilting point, upward flux-water table relationships, drainage volume to water table relationships, root depth and potential evapotranspiration. He found that the model was most sensitive to conductivity values and values of potential evapotranspiration used. As discussed previously, Singh et al. [2006] demonstrated excellent results calibrating their Drainmod simulations for saturated conductivity, alpha, and lateral conductivity using PEST. Through a combination of previous recommendations, trial, and error it was decided that the calibrated parameters for hydrology simulation would be lateral conductivity, matching

point conductivity, and maximum surface storage. The calibrated parameters for nitrogen were chosen to be Michaelis-Menten Parameters (maximum rate and half-saturation constant) for denitrification and well as the advection-dispersion constants (longitudinal dispersivity, and tortuosity).

Input	Independent Measurement or Calculation	Adjusted During Calibration	Expected Range
<b>Weather data</b>			
Hourly precipitation	Required	Never <sup>[a]</sup>	Measured
Daily maximum and minimum air temperature	Required	Never	Measured
Daily potential evapotranspiration (PET)	Preferred	Never	0.0 to 2 cm d <sup>-1</sup>
Evapotranspiration correction factors	Preferred	Rare <sup>[b]</sup>	0.5 to 3.0
<b>Soil property inputs</b>			
Profile layer depths	Preferred	Seldom	5 to 200 cm
Hydraulic conductivity by layer	Possible	Often	0.05 to 100 cm h <sup>-1</sup>
Soil water characteristic by layer	Preferred	Often	Function, see DRR <sup>[c]</sup>
Drainage volume versus water table depth ( $V_d$ )	Preferred	Often	Function, see DRR
Maximum upward flux versus water table depth (Upflux)	Possible	Often	Function, see DRR
Infiltration parameters versus water table depth	Possible	Possible	Table, see DRR
Soil water content at saturation ( $\theta_s$ )	Preferred	Possible	0.3-0.9 cm <sup>3</sup> cm <sup>-3</sup>
Soil water content at lower limit ( $\theta_l$ )	Possible	Often	( $\theta_s - 0.10$ ) to ( $\theta_s - 0.26$ ) cm <sup>3</sup> cm <sup>-3</sup>
<b>Seepage inputs</b>			
Thickness of restrictive layer	Possible	Often	2 to 500 cm
Hydraulic conductivity of restrictive layer	Difficult	Often	0.0 to 1 cm h <sup>-1</sup>
Hydraulic head at bottom of restrictive layer	Possible	Often	0 to 1000 cm
Distance to lateral sink	Preferred	Rare	1000 to 20000 cm
Water elevation at lateral sink	Preferred	Possible	0 to 300 cm
Effective lateral hydraulic conductivity	Possible	Often	0.1 to 100 cm h <sup>-1</sup>
<b>Site and drainage system inputs</b>			
Drain depth	Preferred <sup>[d]</sup>	Rare <sup>[d]</sup>	50 to 300 cm
Drain spacing	Preferred <sup>[d]</sup>	Rare <sup>[d]</sup>	5 to 200 m
Drainage coefficient ( $DC$ )	Preferred	Rare	0.5 to 10 cm d <sup>-1</sup>
Surface depressional storage ( $S_1$ )	Possible	Often	0.25 to 10 cm
Minor surface depressional storage ( $S_2$ )	Difficult	Often	0.25 to 10 cm
Depth to restrictive layer	Possible	Rare	50 to 1000 cm
Weir depth	Preferred <sup>[d]</sup>	Rare <sup>[d]</sup>	0 to drain depth
Crop or vegetation inputs	Possible	Rare	See Evans and Skaggs (1993)
Root depth versus time (cm vs. days)	Possible	Possible	Table, 0 to 100 cm
<b>Data needed for calibration and validation</b>			
Daily subsurface drainage outflow	Required <sup>[e]</sup>		0 to 4 cm
Daily surface runoff	Preferred <sup>[e]</sup>		0 to 8 cm
Water table depth at end of day	Required		0 to 300 cm

<sup>[a]</sup> Measured precipitation data are never adjusted for calibration. When hourly values are estimated from measured daily values, some calibration could occur as long as the daily amounts remain unchanged.

<sup>[b]</sup> PET correction factors should rarely be changed during calibration. If used, adjustments should be 15% or less.

<sup>[c]</sup> DRR is the DRAINMOD Reference Report ([www.bae.ncsu.edu/soil\\_water/drainmod/manuals.html](http://www.bae.ncsu.edu/soil_water/drainmod/manuals.html)).

<sup>[d]</sup> These inputs are usually known or are design variables. If they are known, they should not be calibrated. If they are design variables, they will be varied accordingly. In some cases, including non-parallel, poorly defined drains (typical in some wetland cases), these inputs may not be readily known and are obtained by calibration.

<sup>[e]</sup> While separate surface runoff and subsurface flow measurements are preferred, the model can be calibrated with only subsurface outflow data, or with a combination of surface and subsurface flow data if necessary.

Figure 3.13: Table of Drainmod Input Parameters Including their Relative Utility in Calibration. Taken from Skaggs et al. [2012]

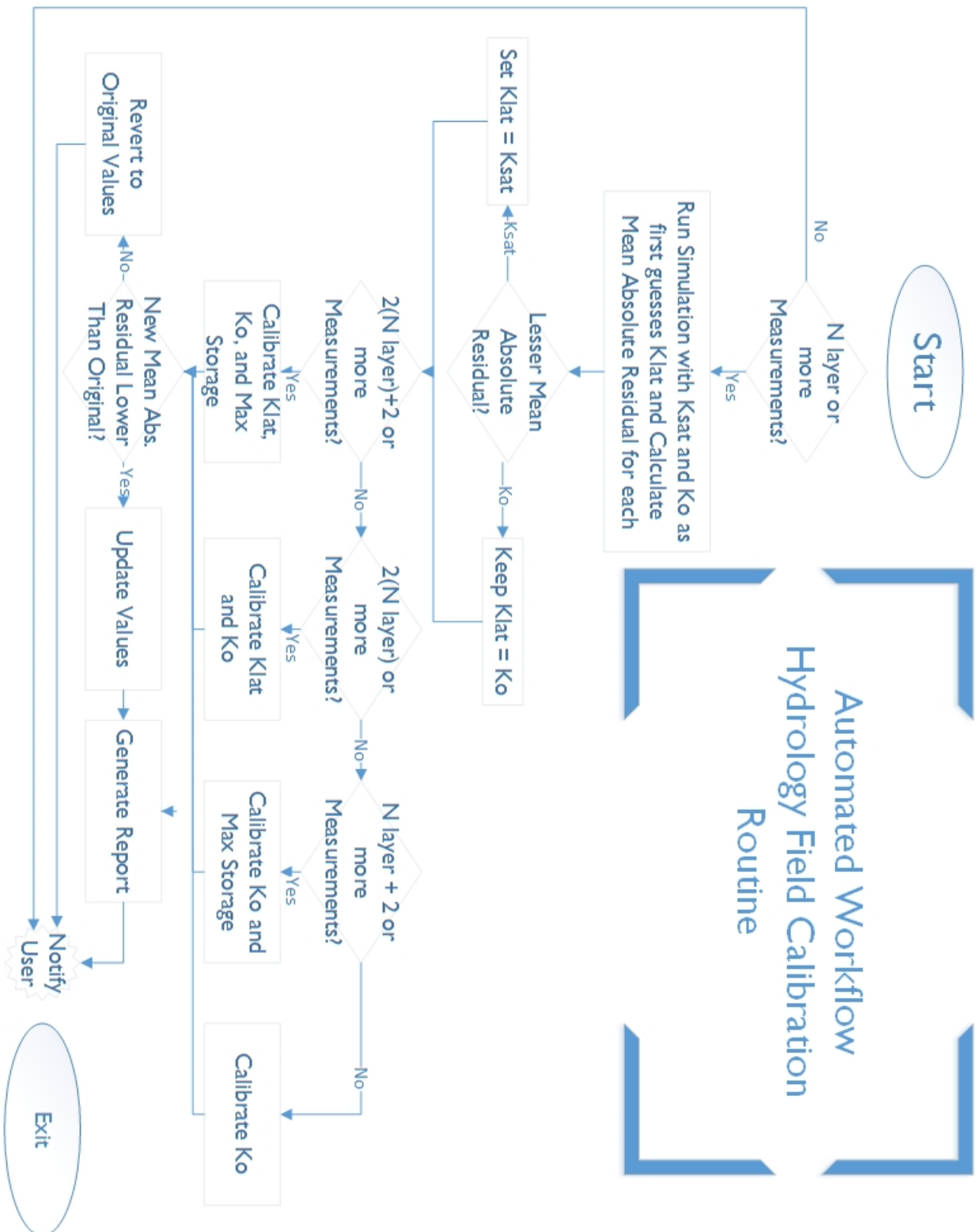


Figure 3.14: Hydrology Calibration Routine Flowchart



**Matching Point Hydraulic Conductivity** Matching point hydraulic conductivity ( $K_o$ ) is an important parameter in that Drainmod uses this value in order to calculate Green and Ampt parameters  $A$  and  $B$  when creating soil files from Rosetta output. Recall that Drainmod calculates infiltration rate using the rate of precipitation when infiltration capacity has not been met and the Green and Ampt equation for infiltration after infiltration capacity has been met. The form of the Green and Ampt equation used by Drainmod is

$$f = \frac{A}{F} + B \quad (3.2)$$

where  $f$  is the infiltration rate and  $F$  is the accumulative infiltration. The Green and Ampt parameters  $A$  and  $B$  are

$$\begin{aligned} A &= (k_{vs})(\theta_{sat})(S_{av}) \\ B &= (k_{vs}) \end{aligned} \quad (3.3)$$

where  $k_{vs}$  is the vertical saturated conductivity,  $\theta_{sat}$  is the saturated moisture content and  $S_{av}$  is the suction at the wetting which is a function of the relationship of matric head and volumetric moisture content (the characteristic curve). When Drainmod accepts soil file inputs from a Rosetta prediction, it sets  $k_{vs}$  equal to  $k_o$ . Drainmod automatically calculates  $S_{av}$  based upon the characteristic curve determined by the Van Genuchten equation and uses the  $\theta_{sat}$  predicted by Rosetta.  $A$  and  $B$  are not held constant in the simulation, but are a function of the water table depth (since the water table depth determines the extent to which each layer

plays a role in infiltration). Therefore, Drainmod calculates  $A$  and  $B$  for ten water table depths (0, 10, 20, 40, 60, 80, 100, 150, 200, and 1000 cm below the surface) by weighting the contribution of each layer (by depth) and interpolates to find the  $A$  and  $B$  values for any given water table depth at each iteration of the simulation.

Since  $A$  and  $B$  are both heavily dependent upon  $k_o$ , and the  $k_o$  value for each layer sets the water table vs Green and Ampt relationship, calibrating  $k_o$  effectively calibrates the infiltration parameters without altering the characteristic curve.  $k_o$  is a good choice for calibration when the simulation is over-predicting or under-predicting the rate of infiltration while the water table is still well below the surface – a valuable option to have considering the plethora of factors influencing infiltration rates (e.g. compaction compaction and preferential flow in the surface layers). The allowed values for  $k_o$  are those recommended by [Skaggs et al., 2012]: 0.05 to 100 cm/hr.

**Lateral Conductivity** Lateral conductivity is another important conductivity value which shows great control over the rate of flow to drains. Lateral conductivity is the saturated hydraulic conductivity experienced in radial flow towards the subsurface drains (including any resistance to entering the drain) and is related to, but not necessarily equal to, vertical and horizontal conductivity. Together Lateral Conductivity and the Green and Ampt parameters seem to show the greatest influence over the percentage of precipitation captured by the system vs that which is forced to leave as runoff– with the Green and Ampt parameters controlling the input allowed at the surface and the lateral conductivity controlling output rates allowed at the drain. In the first-guess feasibility study, the lateral conductivity is

set to the matching point conductivity ( $k_o$ ) predicted by Rosetta—which puts a large weight to this value when combined with its previously mentioned influence on the Green and Ampt parameters. Although used as an initial guess, lateral conductivity is independent of  $k_o$  in the Drainmod routine – thus calibrating lateral conductivity separately from matching point conductivity takes weight off the the  $k_o$  value and allows the simulation to calibrate flow controls at both ends (surface and drain) so that it may set limits properly. As shown in the hydrology calibration workflow (and as will be discussed in some detail in the results section of this thesis) the first step in calibrating lateral conductivity is to compare the use of the  $k_o$  parameter predicted by Rosetta to the use of the  $k_{sat}$  parameter predicted by Rosetta as a first-guess for lateral conductivity. This was done because it was found that in systems which allow for short, high-intensity drainage events, lateral saturated conductivity is generally closer to the horizontal saturated conductivity value predicted by Rosetta (usually about one order of magnitude higher) and that beginning with this value in this case lead to better predictions by avoiding local minimums in calibration. Thus, in any calibration procedure, the automated workflow first runs a simulation using each of these initial values, calculates the mean absolute residual for the two simulations, and proceeds with the value that yielded the lower mean absolute residual. Once again, allowed values for  $k_o$  in calibration are those recommended by [Skaggs et al., 2012]: 0.05 to 100 cm/hr.

**PEST Specifics for Automated Calibration** PEST requires three types of input files for each run: template files (one for each model input file which PEST must write to prior to a model run), control files (one for each model output file

which PEST must read after a model run), and a control file (one per run). The template files mark the places in an input file where the calibrated parameters appear so that PEST may write its iterated estimates to the input files before each run. The instruction file performs a similar tasks by telling PEST where to look for results in the output files. The control file contains all the setting for a PEST run as well as the initial values for the parameters being calibrated and the observed values it is calibrating against. An identical set of PEST calibration settings was used for each calibration; that is, each PEST control file used in the automated calibration routine is identical except for the parameters, observations, and necessary values attached to them.

Run Mode	: Parameter Estimation
Initial $\lambda$	: 10
$\lambda$ adjustment factor	: iteration-dependent
Value of variable governing $\lambda$ adjustment	: -3.00
Sufficient value of $\Phi_{new}/\Phi_{old}$ per iteration	: 0.30
Limiting relative $\phi$ reduction between $\lambda$ s	: $3 \times 10^{-3}$
Maximum trial lambdas per iteration	: 20

Table 3.1: General Parameters used in PEST calibrations

## 4 Results

### 4.1 Validation Experiment Overview

A validation experiment was conducted to test the automated workflow’s prediction of subsurface drainage and nitrate concentrations. For this experiment, a data set of tile drain discharge and nitrate concentrations from four fields in the Oregon Willamette Valley were compared to automated workflow simulations. The data set was collected between October 2000 and May 2002 in a previous study by Warren [2002]. The flow rates for all four fields were measured using turbine flow meters (TX101, submersible, brass model, Seametrics, Kent, Washington). Flow measurements were logged in 5 and 15 minute intervals which were summed to yield total daily discharge volumes in units of cm of water per day. Nitrate concentrations were determined through time-weighted auto-sampling with a model 2900 ISCO sampler (ISCO Inc., Lincoln, Nebraska) followed by analysis for nitrate-nitrogen using an Astoria-Pacific International Alpkem Nitrogen Autoanalyzer (Astoria-Pacific, Inc., Clackamas, Oregon). The experimental set up for the experiment is shown in figure 4.1. First guess and calibrated simulation results were compared to reported results using 3 metrics: Root Mean Square Error (RMSE), average absolute residual ( $RES_{avg}$ ), and Nash Sutcliffe Efficiency (NSE) [Nash and Sutcliffe, 1970].

$$RMSE = \sqrt{\frac{\sum (x_{Obs} - x_{Sim})^2}{N}} \quad (4.1)$$

$$RES_{avg} = \frac{\sum |x_{Obs} - x_{Sim}|}{N} \quad (4.2)$$

$$NSE = 1 - \frac{\sum (x_{Obs} - x_{Sim})^2}{\sum (x_{Obs} - \overline{x_{obs}})^2} \quad (4.3)$$

where  $x_{Obs}$  is an reported observation,  $x_{Sim}$  is a corresponding simulated value,  $N$  is the total number of data points, and  $\overline{x_{Obs}}$  is the mean reported observation over the entirety of the simulation period. A 12 point calibration was used as the standard calibration period in hydrology simulations for all four fields as this is the minimum calibration period allowed by the automated workflow for a full calibration routine (lateral and hydraulic conductivities as well as maximum storage). Finally, representative nitrate simulations and automated calibrations were carried out on fields 3 and 4 with estimated values.

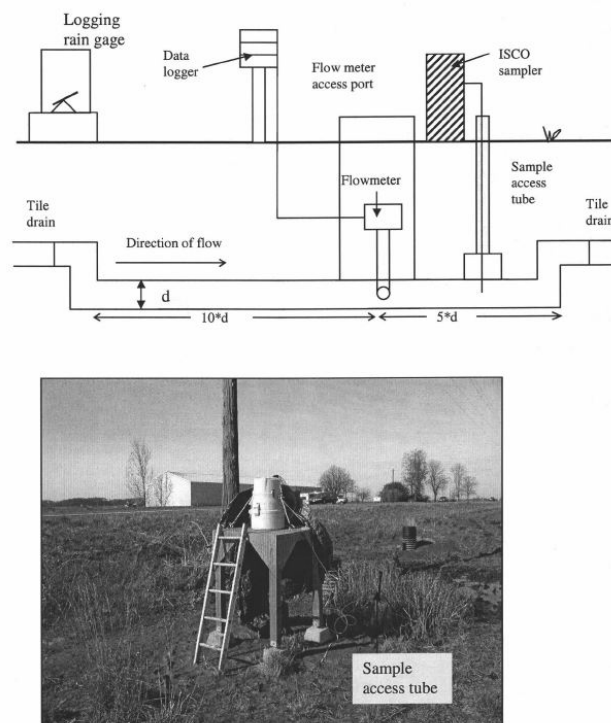


Figure 4.1: The Experimental Set Up of Warren,2002 for Flow Measurement and Nitrate Sampling

### 4.1.1 Site Description and Model Building

The soils represented in the study were silt loam textures including Amity Silt Loam, Dayton Silt Loam, and Woodburn silt loam. Because the exact locations of the tiled fields were not apparent from the literature, the Web Soil Survey was used to download a data set from the general location containing all three soils and the relative areas were manually partitioned in the simulation based upon the values reported by Warren [2002]. Table 4.1 shows general information for the sites as described by Warren [2002] while table 4.2 show the web soil survey (SSURGO/STATSGO2) data used in building simulations (i.e. the values passed to Rosetta). The web soil survey results were run through the automated workflow soil routine to match them with pre-calculated Rosetta predictions and aggregate them to 5 layers (all three soil series consisted of more than 5 layers). The final Rosetta predicted, 5 layer aggregated data used in the simulations are shown in table 4.3.



Field Name	Area (Acre)	Soil Name	Location	Drain Type	Drain Spacing (m)	Drain Depth (m)
Field 1	2.9	Dayton silt loam	Dayton,OR.	4" Corr.	15	1.2-1.5
Field 2	2.9	Dayton Silt Loam/ Amity Silt Loam	Dayton,OR.	4" Corr.	15	1.2-1.5
Field 3	29.7	80% Woodburn Silt Loam 20% Amity Silt Loam	16km east of Corvallis,Or.	6" Corr.	15	1.2-1.5
Field 4	20.0	80% Woodbun Silt Loam 20% Amity Silt Loam	16km east of Corvallis,Or.	6" Corr.	15	1.2-1.5

Table 4.1: General Drainage System Information for Experimental Sites as Reported by Warren [2002]

### Amity Silt Loam, 0 to 3 Percent Slopes

MapUnit: 2301A

Layer	chkey	Depth to Bottom (cm)	Bulk Density (g/cm <sup>3</sup> )	$\theta_{33}$ (cm <sup>3</sup> /cm <sup>3</sup> )	$\theta_{1500}$ (cm <sup>3</sup> /cm <sup>3</sup> )	% Sand	% Silt	% Clay
1	37862413	18	1.35	30.5	16.40	6	72	22
2	37862414	41	1.35	29.9	15.40	6	71	23
3	37862415	56	1.45	29.6	15.1	6	70	24
4	37862416	71	1.42	31.7	17.9	4.5	65.5	30
5	37862417	89	1.41	33.0	20.1	4.5	60.5	35
6	37862418	183	1.35	29.2	14.2	5	69	26

### Dayton silt loam, 0 to 2 percent slopes

MapUnit: 2306A

Layer	chkey	Depth to Bottom (cm)	Bulk Density (g/cm <sup>3</sup> )	$\theta_{33}$ (cm <sup>3</sup> /cm <sup>3</sup> )	$\theta_{1500}$ (cm <sup>3</sup> /cm <sup>3</sup> )	% Sand	% Silt	% Clay
1	37862428	23	1.35	29.9	15.5	5	74	21
2	37862429	30	1.44	28.7	13.8	5	74	21
3	37862430	38	1.44	28.3	13.3	5	74	21
4	37862431	56	1.30	31.9	24.3	3	52	45
5	37862432	74	1.30	30.9	22.5	3	55	42
6	37862433	102	1.30	32.1	18.5	5	60	35
7	37862434	135	1.35	29.2	14.2	5	69	26
8	37862435	163	1.35	27.9	12.5	15	62	23
9	37862436	193	1.35	27.9	12.5	15	62	23

### Woodburn silt loam, 0 to 3 percent slopes

MapUnit: 2310A

Layer	chkey	Depth to Bottom (cm)	Bulk Density (g/cm <sup>3</sup> )	$\theta_{33}$ (cm <sup>3</sup> /cm <sup>3</sup> )	$\theta_{1500}$ (cm <sup>3</sup> /cm <sup>3</sup> )	% Sand	% Silt	% Clay
1	37862395	23	1.35	30.6	16.5	5	73	22
2	37862396	43	1.35	29.8	15.3	5	72	23
3	37862397	64	1.42	31.4	17.5	4.5	67.5	28
4	37862398	81	1.42	30.9	16.7	4.5	67.5	28
5	37862399	99	1.35	29.3	14.4	5	69	26
6	37862400	137	1.35	29.3	14.3	5	69	26
7	37862401	173	1.35	27.9	12.5	15	62	23
8	37862402	203	1.40	24.4	8.5	55	30	15
9	37862403	234	1.40	27.3	8.5	55	30	15

Table 4.2: Web Soil Survey Parameters for Soils used in Validation Experiment

**Amity Silt Loam, 0 to 3 Percent Slopes**  
**MapUnit: 2301A**

Depth to Bottom (cm)	$\theta_{sat}$ ( $cm^3/cm^3$ )	$\theta_{res}$ ( $cm^3/cm^3$ )	$\alpha$ (1/cm)	$n$ (—)	$k_{sat}$ (cm/day)	$k_o$ (cm/day)	$L$ (—)
41	$6.18E^{-2}$	0.439	$1.36E^{-2}$	1.34	31.1	4.74	-0.621
56	$5.84E^{-2}$	0.415	$1.10E^{-2}$	1.36	18.31	3.68	-0.416
71	$7.08E^{-2}$	0.437	$1.28E^{-2}$	1.32	19.98	3.77	-0.739
89	$7.93E^{-2}$	0.449	$1.55E^{-2}$	1.28	11.12	4.22	-0.181
183	$5.75E^{-2}$	0.433	$1.12E^{-2}$	1.38	21.79	4.10	-0.274

**Dayton silt loam, 0 to 2 percent slopes**  
**MapUnit: 2306A**

Depth to Bottom (cm)	$\theta_{sat}$ ( $cm^3/cm^3$ )	$\theta_{res}$ ( $cm^3/cm^3$ )	$\alpha$ (1/cm)	$n$ (—)	$k_{sat}$ (cm/day)	$k_o$ (cm/day)	$L$ (—)
23	$5.98E^{-2}$	0.435	$1.29E^{-2}$	1.36	32.0	4.70	-0.490
38	$5.17E^{-2}$	0.410	$1.05E^{-2}$	1.37	22.9	3.84	-0.286
74	$1.1E^{-2}$	0.483	$4.92E^{-2}$	1.27	15.9	10.3	-2.86
102	$7.82E^{-2}$	0.468	$1.54E^{-2}$	1.31	17.5	4.77	-0.88
193	$5.22E^{-2}$	0.394	$1.09E^{-2}$	1.38	22.4	4.10	-0.2.70

**Woodburn silt loam, 0 to 3 percent slopes**  
**MapUnit: 2310A**

Depth to Bottom (cm)	$\theta_{sat}$ ( $cm^3/cm^3$ )	$\theta_{res}$ ( $cm^3/cm^3$ )	$\alpha$ (1/cm)	$n$ (—)	$k_{sat}$ (cm/day)	$k_o$ (cm/day)	$L$ (—)
43	$6.22E^{-2}$	0.438	$1.31E^{-2}$	1.35	29.7	4.626	-0.541
81	$6.76E^{-2}$	0.433	$1.23E^{-2}$	1.33	15.94	3.77	-0.636
137	$5.79E^{-2}$	0.433	$1.11E^{-2}$	1.38	21.94	4.10	-0.287
173	$4.92E^{-2}$	0.418	$1.02E^{-2}$	1.41	23.91	4.08	-0.120
234	$3.18E^{-2}$	0.394	$1.03E^{-2}$	1.42	30.77	5.14	-0.014

Table 4.3: Aggregated Rosetta Predictions Used in Simulation

Fields 1 and 2 were characterized by rapid, short duration drainage with peak instantaneous drainage rates of roughly  $2.5 \text{ cm day}^{-1}$ . Monitoring well measurements showed the water table remaining below 50 cm of the soil surface for the duration of the study and dropping below the depth of the tiles within a 24 hour period following precipitation events. Fields 3 and 4 were characterized by low-intensity drainage rates of substantial duration. Instantaneous drainage rates were between  $0.5\text{-}0.9 \text{ cm day}^{-1}$ . The minimum water table depth was recorded as 10 cm the soil surface and it took 1-2 weeks for the water table to drop below the tile drains following major precipitation events.

All fields were planted with perennial grass seed. Field 1 was planted with first-year Tall Fescue immediately following a corn growing season. Field 2 was planted with Perennial Ryegrass for the duration of the study. Fields 3 and 4 were both planted with established Tall Fescue. All crops were modeled using the grassland/-generic crop routine of the automated workflow. For all fields but field 1, the maximum root depth was set at 61 cm and the average root depth of 40 cm. For field 1 (it being a first-year crop) the maximum root depth was set at 45 cm and the average root depth was set at 25 cm. Root-to-shoot ratio was set to be 2 and the month of maximum yield was set to be May for all fields.

Season	Field 1	Field 2	Field 3	Field 4
1999-2000	Beets	Corn	Tall Fescue	Tall Fescue
2000-2001	Corn	Ryegrass	Tall Fescue	Tall Fescue
2001-2002	Tall Fescue	Ryegrass	Tall Fescue	Tall Fescue

Table 4.4: Crop Rotations for Each Field in Validation Study



Figure 4.2: First year (left) and Established (right) Tall Fescue of Fields 1 and 3 Respectively. Taken from Warren [2002]

All simulations were run for approximately 17 years with simulations beginning on January 1st, 1985 and ending at the last month for which discharge measurements were available. During the period for which discharge measurements were taken, daily and/or 15 minute precipitation data collected on site using a tipping bucket method by Warren, 2002 was used where available. Failures in tipping bucket equipment were reported for by Warren [2002] for 3 periods during discharge measurements. For these periods, 15 minute precipitation data from nearby weather stations was used. Outside of the measurement period, PRISM daily precipitation for the counties of each respective field were used. PRISM daily maximum/minimum temperature values were used for the duration of the simulation. Potential Evapotranspiration values were calculated using the Thornthwaite method (the standard method of the automated workflow). Table 4.5 details the source for all meteorological data used in the experiment.

**Fields 1 & 2**

<b>Date Range</b>	<b>Type</b>	<b>Source</b>
1 Jan,1985- 27 March, 2002	Max/Min Daily Temperature	PRISM Climate Grp. Yamhill Co.
1 Jan,1985- 9 Oct,2001	Daily Precipitation	PRISM Climate Grp. Yamhill Co.
10 Oct,2001- 20 Mar,2002	On-Site Tipping Bucket Precip.	Warren,2002
21 Mar,2002- 27 Mar,2002	Daily Precipitation	McMinnville Munic. weather station

**Fields 3 & 4**

<b>Date Range</b>	<b>Type</b>	<b>Source</b>
1 Jan,1985- 11 May, 2002	Max/Min Daily Temperature	PRISM Climate Grp. Linn Co.
1 Jan,1985- 30 June,2000	Daily Precipitation	PRISM Climate Grp. Linn Co.
1 July - 14 Jan,2000	On-Site Tipping Bucket Precip.	Warren,2002
15 Jan - 30 Jan,2000	15 min Precipitation	Jefferson Munic. weather station
1 Feb - 25 Feb,2002	On-Site Tipping Bucket Precip.	Warren,2002
26 Feb - 27 Mar,2002	15 min Precipitation	Jefferson Munic. weather station

Table 4.5: Meteorological Data Sources Used for Validation Experiments

### 4.1.2 Results and Discussion

#### **First-Guess Hydrology Simulations**

After a simulation was created for each of the field sites using the data and descriptions of [Warren, 2002], the ‘first-guess’ uncalibrated simulations were run for each field and compared to observed values of discharge. Figures 4.3 through 4.6 show uncalibrated simulation predictions of daily drainage discharge and observed measurements for each of the four fields. In order to distinguish zero measurements from absence of measurements (as there were gaps in data collection for all four fields) dates with no measurement are marked with a grey vertical line below the axis.



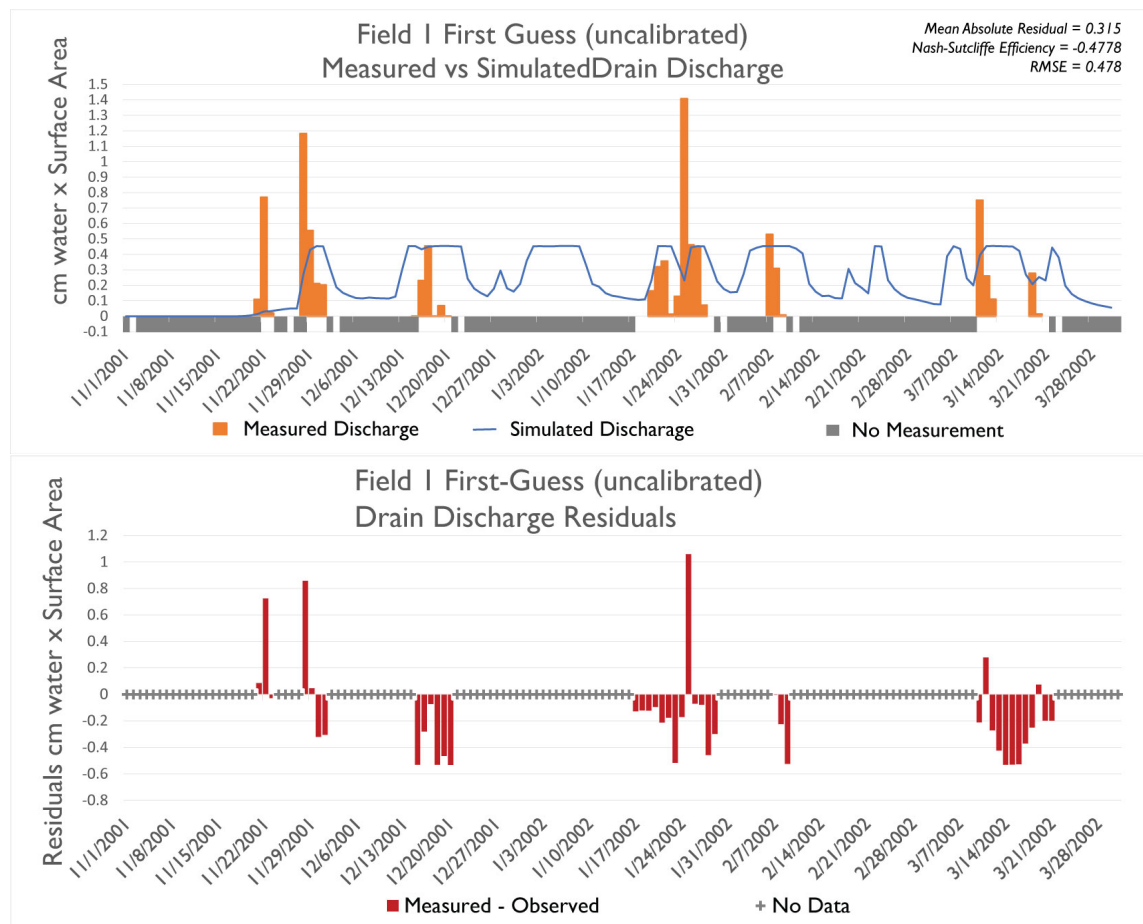


Figure 4.3: Field 1 First Guess Uncalibrated Simulation Vs Observed Discharge

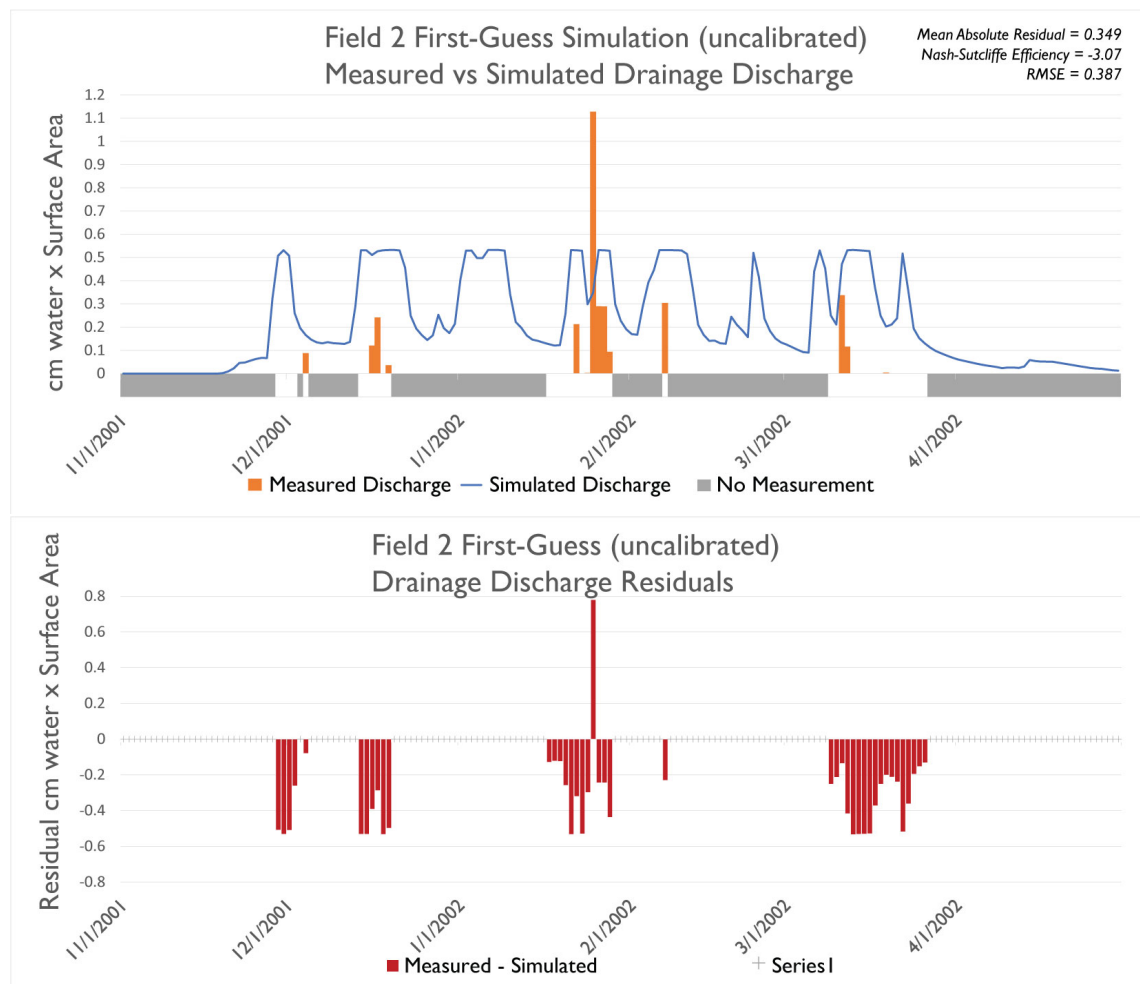


Figure 4.4: Field 2 First Guess Uncalibrated Simulation Vs Observed Discharge

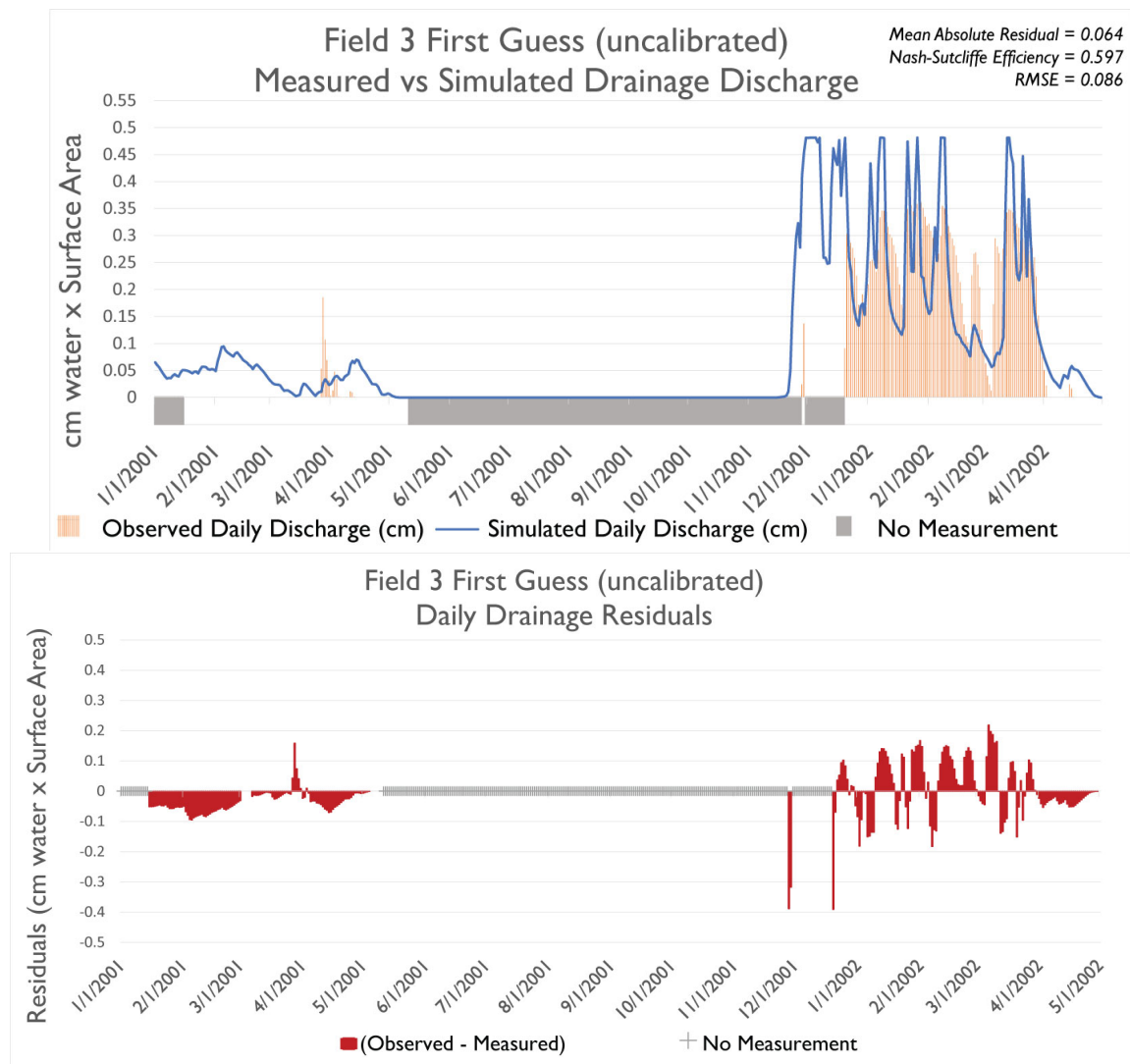


Figure 4.5: Field 3 Precipitation Data and Measured Instantaneous Discharge Rates Over the Period of Measurement

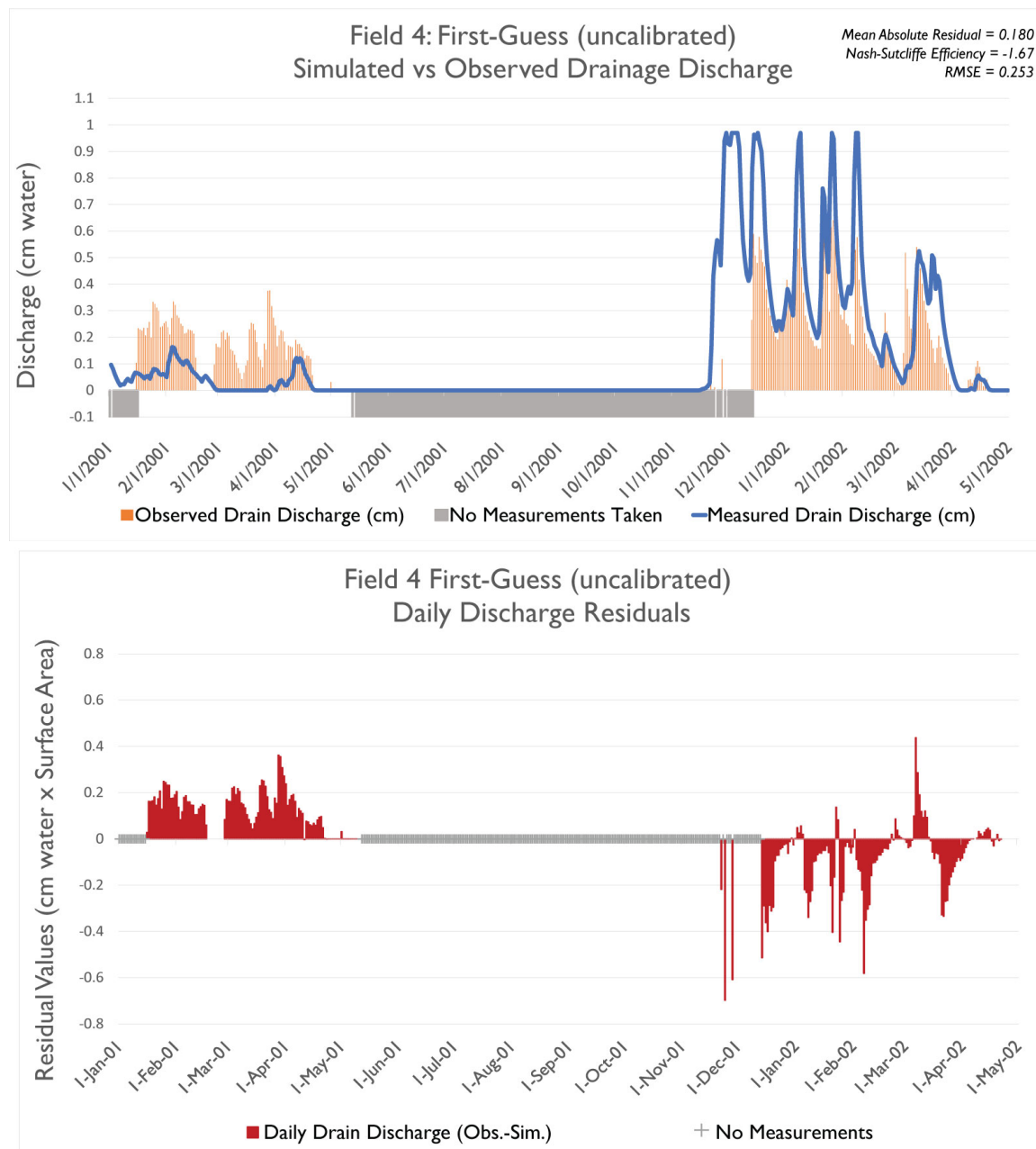


Figure 4.6: Field 4 Daily Precipitation Data, Daily Measured Drainage Discharge Volumes, and Measured Nitrate Concentrations

For the first-guess uncalibrated simulations, only field 3 showed a positive Nash Sutcliffe Efficiency (indicating a simulation fit better than the mean). Fields 1 and 2 showed poor correlation with the first-guess simulations largely due to an inability to capture peak flow events and sharp drop-off (rapid lowering of the water table to below the tile drain) following a flow event. Fields 3 and 4 showed an acceptable fit to the shape of the drainage curve for the second year of data, but both had difficulty estimating discharge in the dryer first year. Upon closer inspection it was discovered that the values reported in the dryer first year were unable to be accounted for by the controls set on mass balances in the drainage model. In other words, simulations with infiltration and lateral conductivity parameters that allowed complete capture with zero runoff could not produce the flow rate quantities reported for field 4 or the lone peak reported at field 3 in the first year (January - December 2001). Several explanations could explain this including noise in the low measurements, a shallower water table than predicted, less root uptake due to water stresses (since the model assumes crops are managed to avoid extreme water stresses), and/or erroneous values for evapotranspiration.

### **Discharge Calibrations: Fields 1 and 2**

Fields 1 and 2 demonstrate why it is important to include a pre-PEST calibration routine for estimating lateral conductivity values. As discussed previously, the first step in the automated calibration procedure is to run a simulation using both the saturated and matching point conductivities predicted by Rosetta as starting values for lateral conductivities, calculating the sum of the (absolute) residuals, and choosing that value which yields the lower sum. The reason this is necessary is due to the

existence of local optima which PEST (using the conservative settings and relatively small sample size we are using here) is prone to settle upon unless the initial value is within about an order of magnitude of the actual value. Because  $K_{sat}$  and  $K_o$  are usually about an order of magnitude apart and both represent physical values, these were the values chosen for dual-first-guess.

Figure 4.7 shows the calibration of lateral and lateral + matching point conductivities for field 2 using the initial value of matching point conductivity as the value for lateral conductivity as well (the default settings for a ‘first-guess’ simulation. The calibration period was chosen to be the 10 point peak reported flow event (maximum storage was not calibrated). Under both conditions, the simulation settles on an local optima and is unable to capture the peak flow event. Figure 4.8, plots corresponding runoff and demonstrates that there is still a considerable amount off runoff forced by the system at the peak event. By comparison, figure ?? shows fields one and two using the Rosetta-predicted saturated conductivity value as a first guess for lateral conductivity. In this case, the peak flow event of field 2 is finally allowed to be captured by the calibration procedure.

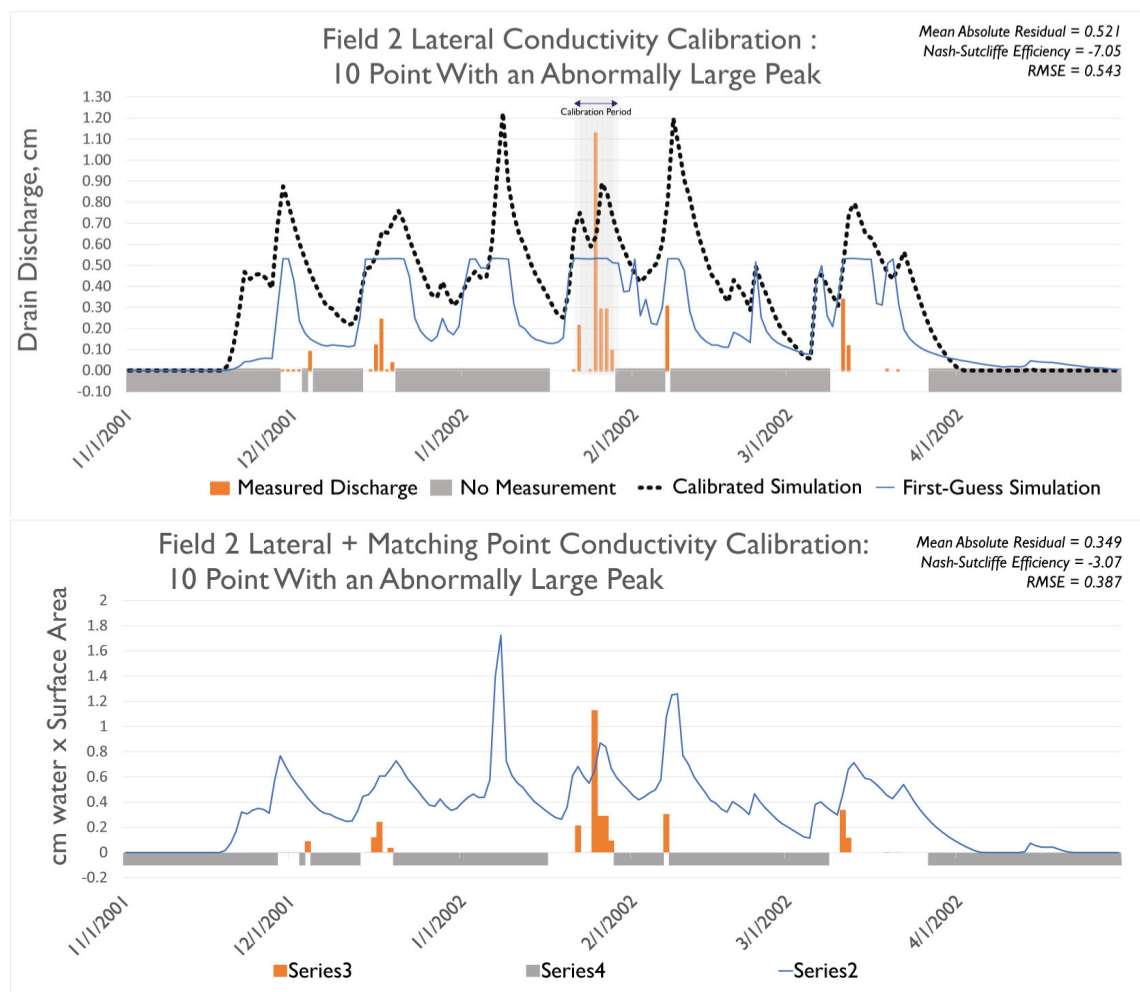


Figure 4.7: Lateral Conductivity (top) and Lateral + Matching Point Conductivity (bottom) Calibrations for Field 2 with  $K_0$  as the Initial Guess for Lateral Conductivities

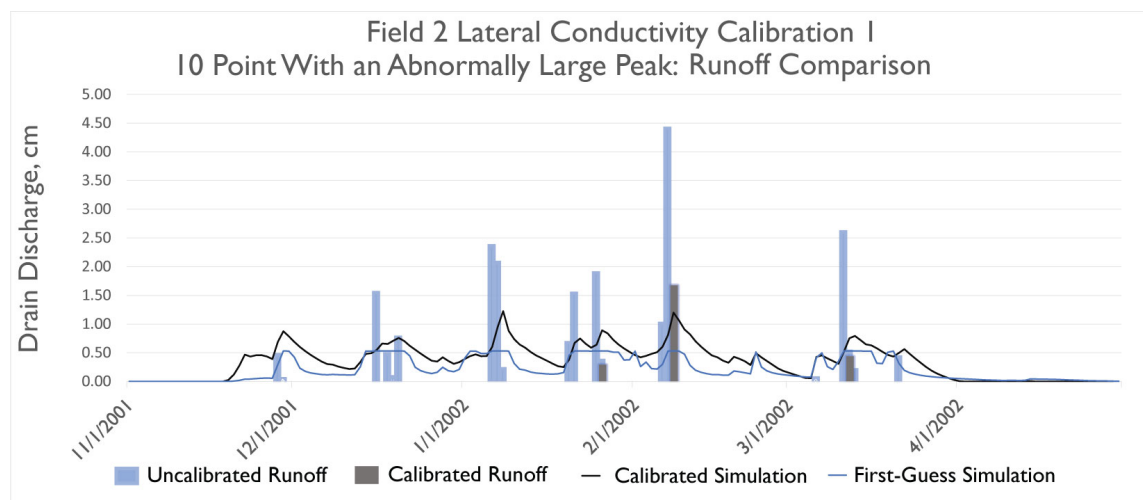


Figure 4.8: Leftover Runoff Forced by the Simulation for Field 2 when Calibrated for the Peak Flow Event and Comparison to Uncalibrated Runoff



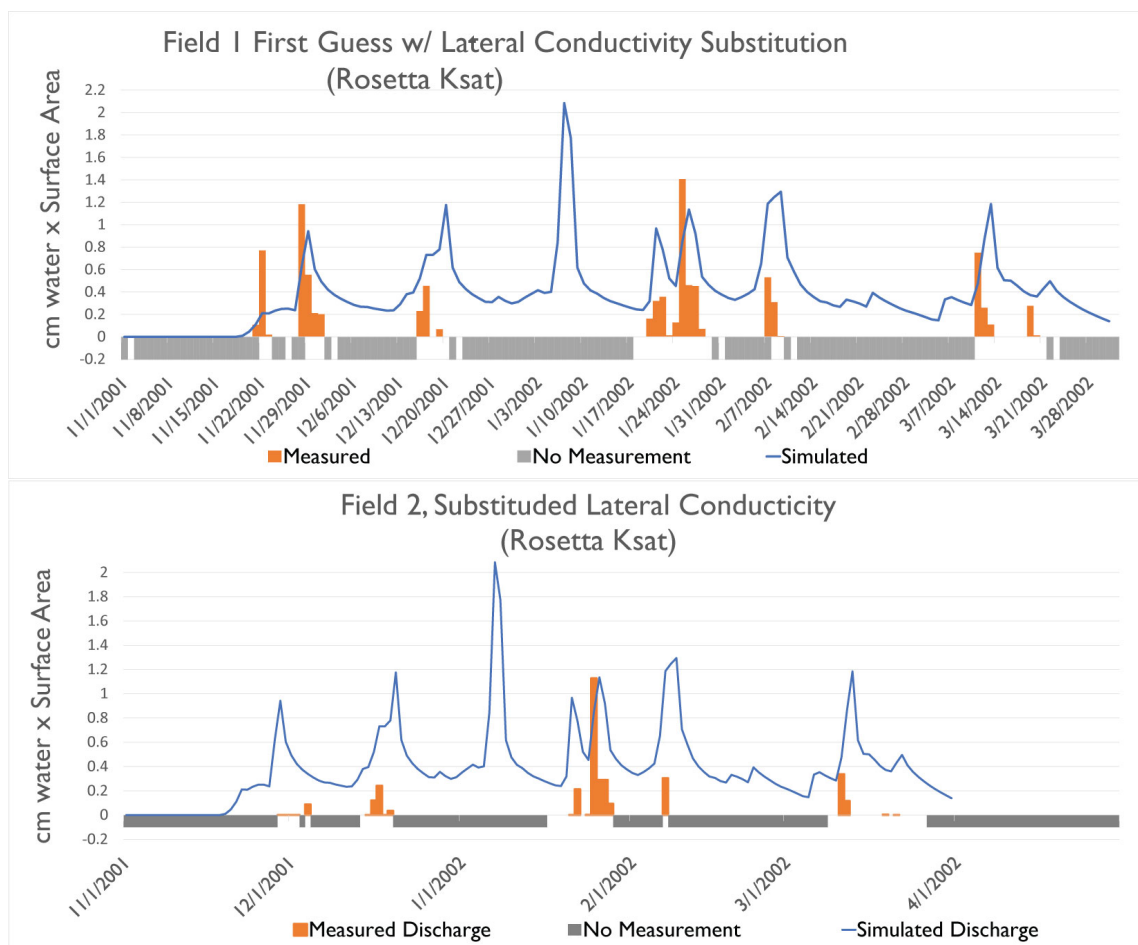


Figure 4.9: Lateral Conductivity (top) and Lateral + Matching Point Conductivity (bottom) Calibrations for Field 2 with  $K_o$  as the Initial Guess for Lateral Conductivities

**Discharge Calibrations: Fields 3 and 4**

Due to the aforementioned trouble with year one, calibrations for fields 3 and 4 were run using exclusively calibration points from year 2. Once again, field 3 performed best in simulation. A 12 point calibration of field 3 brought the Nash Sutcliffe efficiency to 0.744 (figure 4.10). This is an impressive and correlation considering the small amount of observable data used in its achievement. A second calibration for field 3 using 24 points is shown in figure 4.11. The additional 12 points showed no further improvement on predictive power by any of the metrics used in this study. A 12 point calibration of field 4 is shown in figure 4.12. The first year measurements have been left in all goodness-of-fit metric calculations since the cause is not certain. Ignoring year 1 in the field 4, 12 point calibration does succeeded in bringing the Nash Sutcliffe efficiency slightly above 0 to a value of 0.073.

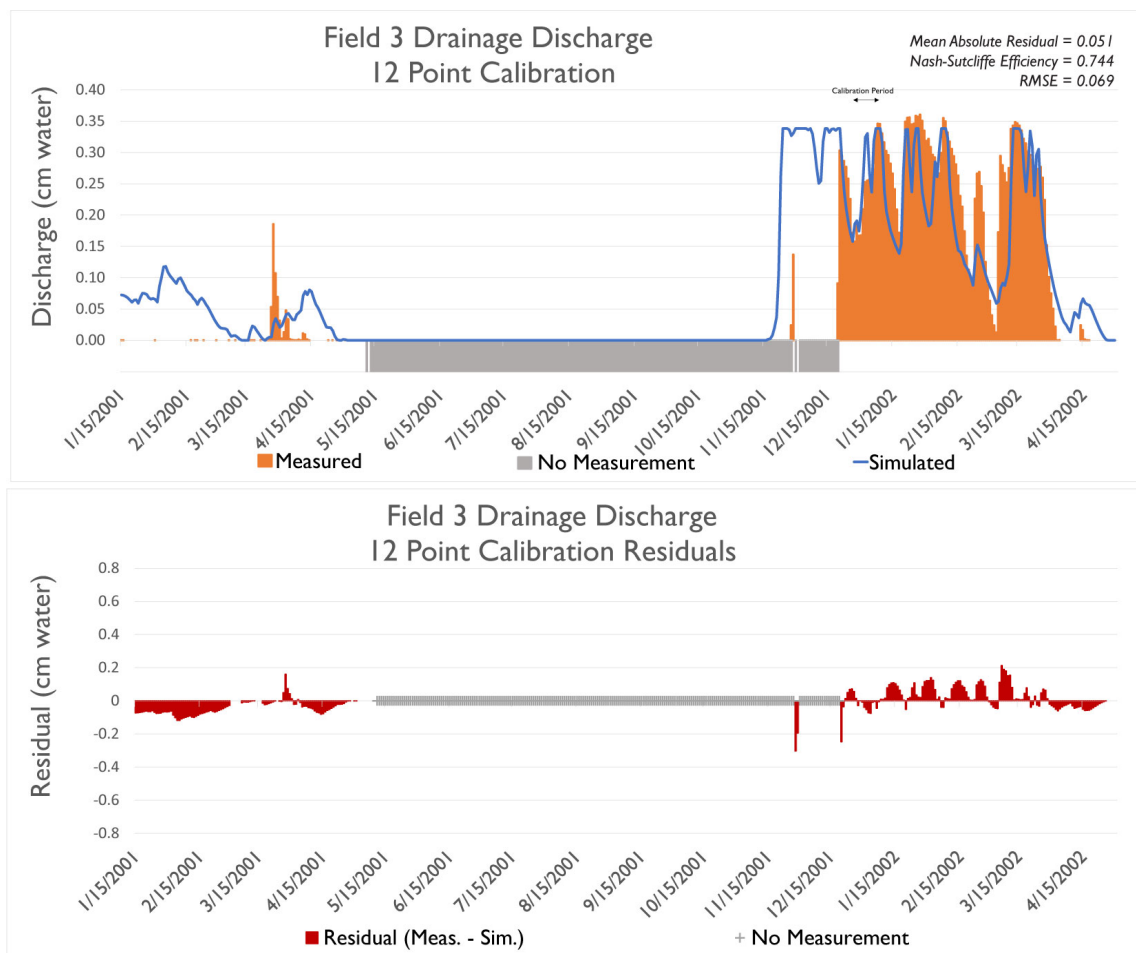


Figure 4.10: Field 3 Discharge and Residuals After 12 point Calibration: December 28th 2001- January 8th 2002

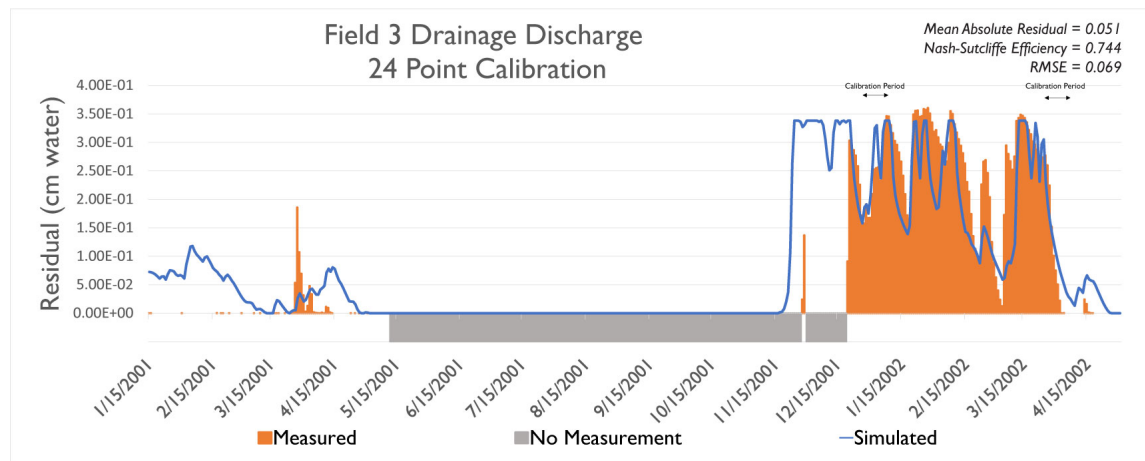


Figure 4.11: Field 3 Discharge After 24 point Calibration: December 28th 2001-January 8th 2002

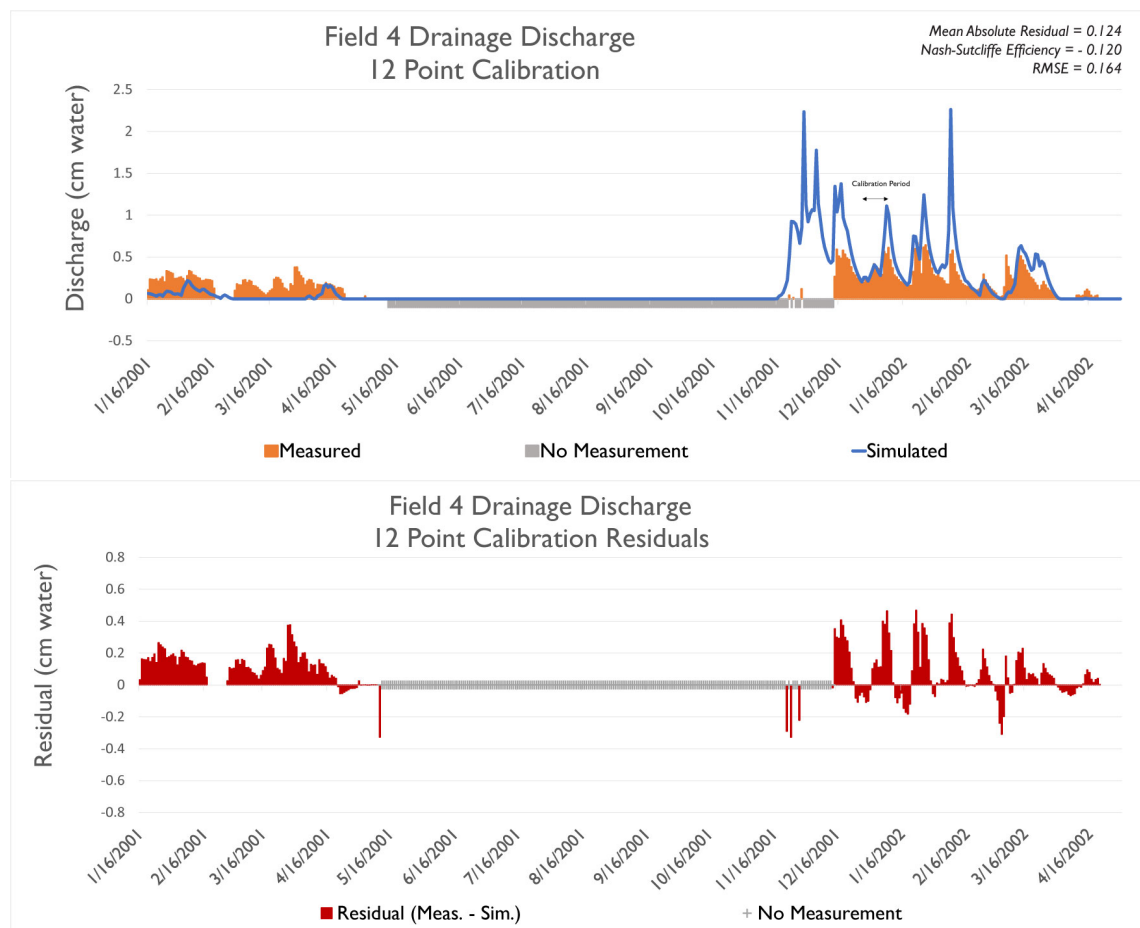


Figure 4.12: Field 4 Discharge and Residuals After 12 point Calibration: December 28th 2001- January 8th 2002

### **Nitrate Simulation: Fields 3 and 4**

For both fields 3 and 4, Warren [2002] mineral nitrogen fertilizer was applied in both years of simulation. Fertilizer was applied in March and April of each year at application rates of about 60 and 90 lbs/acre. With these values placed into the a Drainmod nitrogen file and using calibrated hydrology outputs for 2001-2002, the model consistently overestimated nitrate in both fields 3 and 4. First guess and 12 point calibrations of nitrate for field 3 are show in figure 4.13. The calibration period was from December 28th 2001 - January 9th 2002. Because this period was also the period over which hydrology was calibrated, the nitrate data fit well to observations over this initial 12 pt calibration period and the calibrations follow the ‘first guess’ simulation. A second calibration of 20 points (including the previous calibration as well as the peak nitrate flow period between March 30th and April 5th, 2001) is shown in figure 4.14.

Field 3 represented the best hydrology simulation of the four fields and nitrate simulations are heavily dependent upon their corresponding hydrology simulations. For comparison, figure 4.15 shows a ‘first-guess’ and calibrated simulation of field 4. The calibration procedure serves to scale nitrate levels down to reasonable levels, but is unable to simulate nitrogen movement effectively.

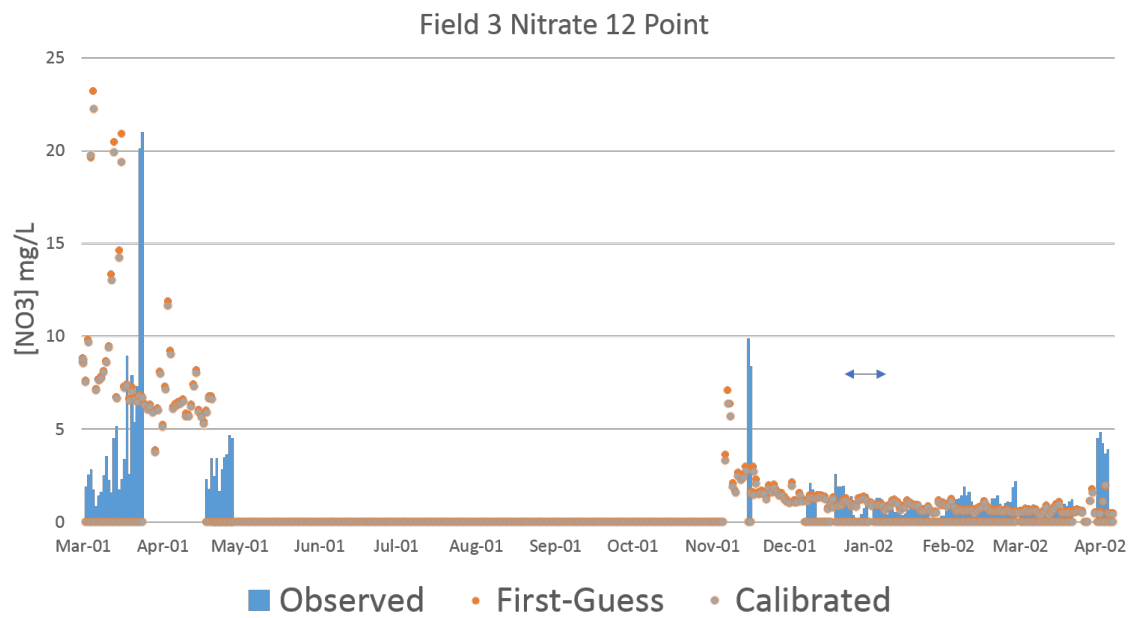


Figure 4.13: Initial Guess and 10 point Calibration for Nitrate Simulations of Field 3

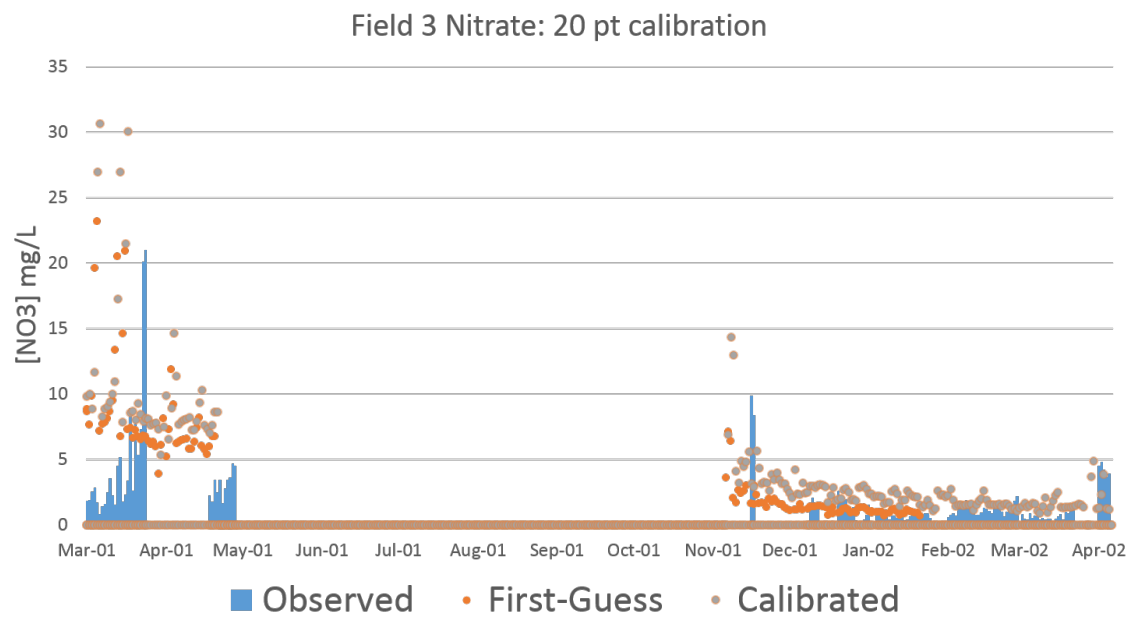


Figure 4.14: Initial Guess and 20 point Calibration for Nitrate Simulations of Field 3



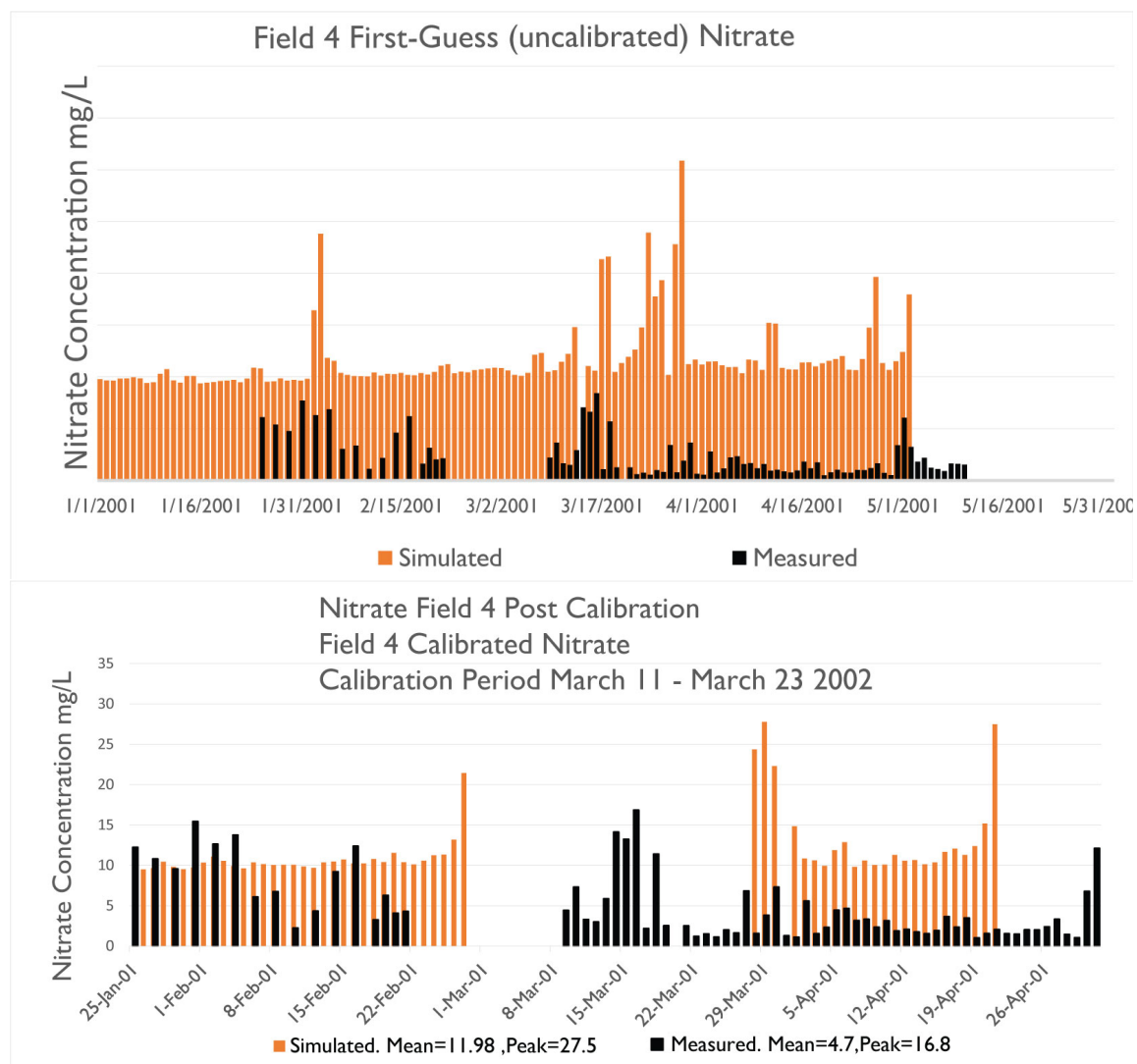


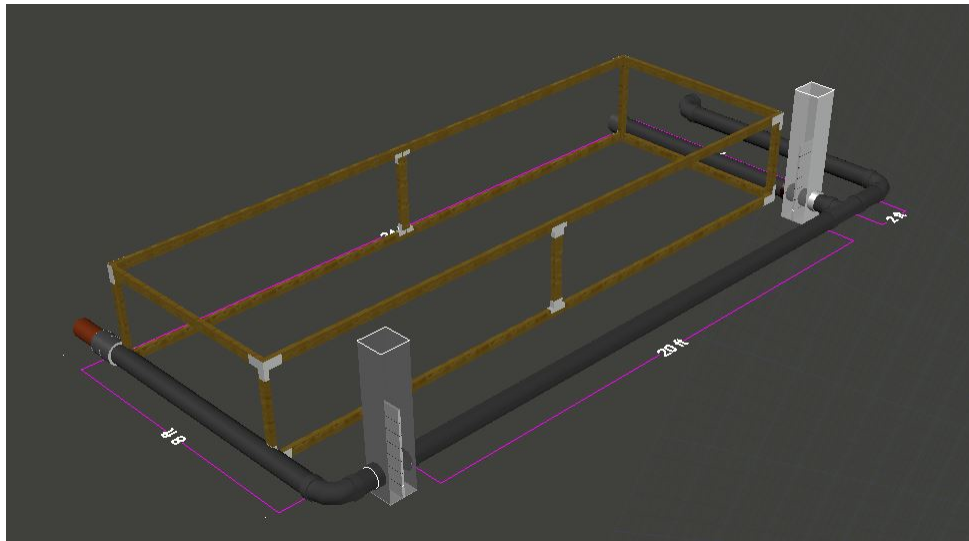
Figure 4.15: Initial Guess and Calibrated Nitrate Simulations for Field 4

## 5 Conclusions And Future Directions

In conclusion, this study presented and demonstrated an automated workflow and calibration procedure built for subsurface drainage simulation to be used in planning and design of denitrifying bioreactors. The simulation was applied to a two-year data set of tile drainage discharge and nitrate concentrations. The simulation was shown to work well for slowly draining fields (3 and 4) with positive Nash Sutcliffe values achieved using only minimal user input to build the simulations and only twelve calibrating observations. The rapid, high intensity drainage of fields 1 and 2 was not captured well by the model in validation, but a pre-calibration routine based upon dual-estimation of the lateral hydraulic conductivity was included in order to make reasonable predictions in this case. Further studies are needed with larger samples sizes in order to improve and further validate the model in the case of rapid drainage. Nitrate simulations were applied to fields 3 and 4. The model was unable to capture the exact timing of nitrogen discharge, but showed reasonable simulation in the case of field 3 in which the hydrology simulation showed the greatest agreement with observations. A more controlled set of studies are needed in order to test hypotheses for better nitrogen simulation. A larger sample size and duration would also be helpful in order to test whether the nitrate simulation consistently predicts statistically relevant leaching over long periods (i.e. 10 years or the expected life of a denitrifying bioreactor). Combination of this model with large datasets or

databases such as the MANAGE Drain Load Database for drainage nutrient studies [Christianson and Harmel, 2015] could be beneficial for improving the model itself as well improving regional denitrifying bioreactor design guidelines in regions for which few studies have been undertaken on denitrifying bioreactor efficiency.

For future studies and further calibration of the model, a denitrifying bioreactor was constructed at the Oregon State Experimental Dairy Farm. The bioreactor schematics and construction are shown below. Recommendations in an operation manual will be made based upon a first guess simulation of the dairy. Drainage and tile measurements may be used to further calibrate the model and correlate predictions with the reduction rates seen in operation.



OSU Dairy Farm Upstream Bioreactor General Layout

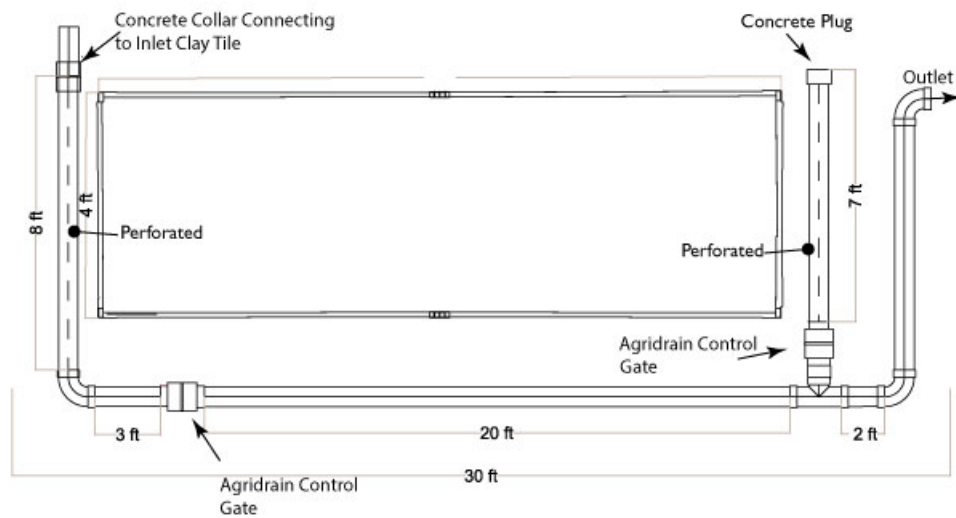


Figure 5.1: General Schematics for the Full Scale Denitrifying Bioreactor Built at the OSU Dairy Farm



Figure 5.2: Construction of the Full Scale Denitrifying Bioreactor for Subsequent Validation and Improvement of the Model

## Bibliography

- PESTmodel-independent parameter estimation and uncertainty analysis. <http://www.pesthomepage.org/>. Accessed: 2017-08-30.
- S. Abdel-Dayem and R. W. Skaggs. Effect of residual entrapped air on predicting soil-water-crop relationships. *Water Science (Cairo, Egypt)*:, pages 19–25, 1991.
- Ahmed Mohamed Abdelbaki and Mohamed A Youssef. Assessing the feasibility of drainmod application using soil hydraulic properties estimated by pedotransfer functions. In *9th International Drainage Symposium held jointly with CIGR and CSBE/SCGAB Proceedings, 13-16 June 2010, Québec City Convention Centre, Quebec City, Canada*, page 1. American Society of Agricultural and Biological Engineers, 2010.
- AFT Trenchers Ltd. Tractor pulled trencher, 2017. URL <https://www.trenchers.co.uk/agri-aft-100.html>. [Online; accessed August 25th, 2017].
- D. M. Amatya, G. M. Chescheir, and R. W. Skaggs. Hydrologic effects of wetland location and size in an agricultural landscape. In *In:Proc. of the 1995 AWRA/ASAE Intl Conf. on Versatility of Wetlands in an Agricultural Landscape*, FL, Sept, 1995. Application of DRAINMOD, 1995. Tampa.
- Jeffrey G Arnold, Daniel N Moriasi, Philip W Gassman, Karim C Abbaspour, Michael J White, Raghavan Srinivasan, Chinnasamy Santhi, RD Harmel, Ann Van Griensven, Michael W Van Liew, et al. Swat: Model use, calibration, and validation. *Transactions of the ASABE*, 55(4):1491–1508, 2012.
- A Bakhsh, RS Kanwar, C Pederson, and TB Bailey. N-source effects on temporal distribution of no<sub>3</sub>-n leaching losses to subsurface drainage water. *Water, Air, and Soil Pollution*, 181(1-4):35–50, 2007.
- Iris Bechtold, Sigrid Köhne, Mohamed A Youssef, Bernd Lennartz, and R Wayne Skaggs. Simulating nitrogen leaching and turnover in a subsurface-drained grassland receiving animal manure in northern germany using drainmod-n ii. *Agricultural water management*, 93(1):30–44, 2007.

- David L Bjorneberg, Ramesh S Kanwar, and Stewart W Melvin. Seasonal changes in flow and nitrate-n loss from subsurface drains. *Transactions of the ASAE*, 39(3):961–967, 1996.
- DW Blowes, WD Robertson, CJ Ptacek, and C Merkley. Removal of agricultural nitrate from tile-drainage effluent water using in-line bioreactors. *Journal of Contaminant Hydrology*, 15(3):207–221, 1994.
- EF Bolton, JW Aylesworth, and FR Hore. Nutrient losses through tile drains under three cropping systems and two fertility levels on a brookston clay soil. *Canadian journal of soil science*, 50(3):275–279, 1970.
- H. Bouwer. Infiltration of water into nonuniform soil. *J. Irrigation and Drainage Division, ASCE*, 95(IR4):451–462, 1969.
- H. Bouwer and J. van Schilfgaarde. Simplified method of predicting fall of water table in drained land. *Transactions of the ASAE*, 6(4):288–291, 1963.
- M. A. Breve. *Modeling movement and fate of nitrogen in poorly drained soils*. PhD thesis, North Carolina State University, Raleigh, 1994. PhD Thesis.
- M. A. Breve, R. W. Skaggs, J. E. Parsons, and J. W. Gilliam. Drainmod-n, a nitrogen model for artificially drained soils. *Trans. of the ASAE*, 40(4):1067–1075, 1997.
- R. G. Broadhead and R. W. Skaggs. Hydrologic model for the north carolina peatlands. In V. A. Dodd and P. M. Grace, editors, *Agricultural Engineering*, pages 61–70. I. Land and Water Use. Bolkema, Rottendam, The Netherlands, 1989.
- CA Cambardella, TB Moorman, DB Jaynes, JL Hatfield, TB Parkin, WW Simpkins, and DL Karlen. Water quality in walnut creek watershed: Nitrate-nitrogen in soils, subsurface drainage water, and shallow groundwater. *Journal of Environmental Quality*, 28(1):25–34, 1999.
- A. C. Chang, R. W. Skaggs, L. F. Hermsmeier, and W. R. Johnson. Evaluation of a water management model for irrigated agriculture. *Transactions of the ASAE*, 26(2):412–418, 1983.
- G. M. Chescheir, R. W. Skaggs, J. W. Gilliam, and R. G. Broadhead. Evaluation of wetland buffers for evaluating treatment of pumped agricultural drainage water. *Transactions of the ASAE*, 35(1):175–182, 1992.

- E. C. Childs and M. Bybordi. The vertical movement of water in stratified porous material 1. infiltration. *Water Resources Res.*, 5(2):446–459, 1969.
- Laura E. Christianson, Alok Bhandari, and Matthew J. Helmers. A practice-oriented review of woodchip bioreactors for subsurface agricultural drainage. *Appl. Eng*, 28(6):861–874, 2012.
- LE Christianson and RD Harmel. The manage drain load database: Review and compilation of more than fifty years of north american drainage nutrient studies. *Agricultural Water Management*, 159:277–289, 2015.
- R. A. Cooke and N. L. Bell. Protocol and interactive routine for the design of subsurface bioreactors. *Appl. Eng*, 30:761–771, 2014.
- J. W. Cox, D. J. McFarlane, and R. W. Skaggs. Field evaluation of drainmod for predicting waterlogging intensity and drain performance in south-western australia, australian journal of soil research. *Vol.*, 32:653–671, 1994.
- J. S. Cris, D. W. DeBoer, and H. D. Werner. Water table management evaluation on glacial till soils. *ASAE, St*, pages 1–13, 1993.
- D.M. David, G.E. Gentry, D.A. Kovacic, and K.M. Smith. Nitrogen balance in and export from an agricultural watershed. *J. Environ. Qual.*, 29:494–508, 1997.
- S. C. Deal, J. W. Gilliam, R. W. Skaggs, and K. D. Konyha. Prediction of nitrogen and phosphorus losses as related to agricultural drainage system design. *Agri. Ecosystems and Environment*, 18:37–51, 1986.
- I. Dean and D. W. DeBoer. Field evaluation of drainmod in south dakota. *ASAE, St*, pages 88–2571, 1988.
- John Doherty et al. Pest: a unique computer program for model-independent parameter optimisation. *Water Down Under 94: Groundwater/Surface Hydrology Common Interest Papers; Preprints of Papers*, page 551, 1994.
- Peter W. van Driel, William D. Robertson, and L. Craig Merkley. Upflow reactors for riparian zone denitrification. *Journal of Environment Quality*, 35:2, 2006. doi: 10.2134/jeq2005.0027.
- R. O. Evans and R. W. Skaggs. Design guidelines for water table management systems on coastal plains soils. *Applied Engineering in Agriculture*, 5(4):534–548, 1989.



- R. O. Evans and R. W. Skaggs. Stress day index models to predict corn and soybean yield response to water table management. In E. Lorre, editor, *Subsurface Drainage Simulation Models, Transactions of Workshop, 15th Congress ICID*,, pages 219–234. The Hague, 1993.
- R. O. Evans, R. W. Skaggs, and R. E. Sneed. Normalized crop susceptibility factors for corn and soybean to excess water stress. *Transactions of the ASAE*, 33(4): 1153–1161, 1990.
- J.M. Fenelon and R.C. Moore. Transport of agrichemicals to ground and surface water in a small central Indiana watershed. *J. Environ. Qual.*, 27:884–894, 1998.
- G. Fipps and R. W. Skaggs. Influence of a slope on subsurface drainage of hillsides. *Water Resources Research*, 25(7):1717–1726, 1989.
- J. L. Fouss, R. L. Bengtson, and C. E. Carter. Simulating subsurface drainage in the lower mississippi valley with drainmod. *Trans. of ASAE*, 30(6):1979–1688, 1987a.
- J. L. Fouss, R. W. Skaggs, and J. S. Rogers. Two-stage weir control for subsurface drainage in humid areas. *Transactions of the ASAE*, 30(6):1713–1719, 1987b.
- James N Galloway, John D Aber, Jan Willem Erisman, Sybil P Seitzinger, Robert W Howarth, Ellis B Cowling, and B Jack Cosby. The nitrogen cascade. *AIBS Bulletin*, 53(4):341–356, 2003.
- G. A. Gayle, R. W. Skaggs, and C. E. Carter. Evaluation of a water management model for a louisiana sugar cane field. *Journal of the American Society of Sugar Cane Technologists*, 4:18–28, 1985.
- Ehsan Ghane, Norman R. Fausey, and Larry C. Brown. Non-darcy flow of water through woodchip media. *Journal of Hydrology*, 519:3400–3409, November 2014. doi: 10.1016/j.jhydrol.2014.09.065.
- W.H. Green and G. Ampt. Studies of soil physics, part i – the flow of air and water through soils. *J. Ag. Sci.*, 4:1–24, 1911.
- G. P. Gupta, S. O. Prasher, S. T. Chieng, and I. N. Mathur. Application of drainmod under semi-arid conditions. *Agricultural Water Management*, 24:63–80, 1993.
- T. G. Helwig et al. *Modelling nitrate losses in drainage water using DRAINMOD*, 56(2):153–168, 2002.

- D. Hillel and W. R. Gardner. Steady infiltration into crust topped profiles. *Soil Science*, 108:137–142, 1969.
- RJM Hudson and RAC Cooke. methyl mercury export from denitrifying bioreactors in tile-drained fields of central illinois. In *2010 Symposium. Champaign, Ill.: Illinois Sustainable Technology Center*. doi: <http://www.istc.illinois.edu/research/092910symposium/1045.pdf>, 2011.
- D.B. Jaynes, J.L. Hatfield, and D.W. Meek. Water quality in Walnut Creek watershed: Herbicides and nitrate in surface waters. *J. Environ. Qual.*, 28:45–59, 1999.
- H. Kandil, C. T. Miller, and R. W. Skaggs. Modeling long-term solute transport in the unsaturated zone. *Water Resources Research*, 28(10):2799–2809, 1992.
- H. M. Kandil, R. W. Skaggs, S. Abdel-Dayem, and Y. Aiad. Drainmod-s: Water management model for irrigated lands, crop yield and application. In E. Lorre, editor, *Subsurface Drainage Simulation Models, Transactions of Workshop, 15th Congress ICID*,, pages 257–275. The Hague, 1993.
- L. M. Kellman. A study of tile drain nitrate -  $\delta^{15}\text{N}$  values as a tool for assessing nitrate sources in an agricultural region. *Nutrient Cycling in Agroecosystems*, 71(2):131–137, Feb 2005. ISSN 1573-0867. doi: 10.1007/s10705-004-1925-0. URL <https://doi.org/10.1007/s10705-004-1925-0>.
- D. Kirkham. Seepage of steady rainfall through soil into drains. *American Geophysical Union Transactions*, 39:892–908, 1958.
- K. D. Konyha and R. W. Skaggs. A coupled field hydrology open channel flow model: Theory. *Transactions of the ASAE*, 35(5):1431–1440, 1992.
- E. Lorre, B. Lesaffre, and R. W. Skaggs. Comparison of models for subsurface drainage in flat and sloping lands. *Journal of Irrigation and Drainage*, 120(2):166–177, 1994.
- L Ma, LR Ahuja, JC Ascough, MJ Shaffer, KW Rojas, RW Malone, and MR Cameira. Integrating system modeling with field research in agriculture: Applications of the root zone water quality model (rzwqm). *Advances in Agronomy*, 71:233–292, 2001.
- C. A. Madramootoo. Assessing drainage benefits on a heavy clay soil in quebec. *Trans. of the ASAE*, 33(4):1217–1223, 1990.

- C. A. Madramootoo, S. R. Broughton, and G. T. Dodds. Watertable management strategies for soybean production on a sandy loam soil. *Canadian Agricultural Engineering*, 37(1):1–7, 1995.
- F. C. Massey, R. W. Skaggs, and R. E. Sneed. Energy and water requirements for subirrigation versus sprinkler irrigation. *Transactions of the ASAE.*, 26(1): 126–133, 1983.
- P. C. McMahon, S. Mostaghimi, and Wright Fs. Simulation of corn yield by a water management model for a coastal plains soil. *Trans. ASAE*, 31(3):734–742, 1988.
- Russell G Mein and Curtis L Larson. Modeling infiltration during a steady rain]. *Water Resources Research*, 9(5):1478–1478, 1973.
- GA Miller, TE Fenton, BR Oneal, BJ Tiffany, and CL Burras. Iowa soil properties and interpretations database. *ISPAID Version*, 7, 2010.
- H.J. Morel-Seytoux and J. Kahanji. Derivation of an equation of infiltration. *Water Resources Research*, 10(4):794–800, 1974.
- Arvin Mosier, J Keith Syers, and John Raymond Freney. *Agriculture and the nitrogen cycle: assessing the impacts of fertilizer use on food production and the environment*, volume 65. Island Press, 2004.
- D.J. Mulla and J.S. Strock. *Nitrogen in Agricultural Systems*, chapter 10:Nitrogen Transport Processes in Soil. American Society of Agronomy Inc., Crop Science Society of America Inc., Soil Science Society of America Inc., 2008.
- J Eamonn Nash and Jonh V Sutcliffe. River flow forecasting through conceptual models part ia discussion of principles. *Journal of hydrology*, 10(3):282–290, 1970.
- Abraham Joel Osvaldo Salazar, Ingrid Westro. Evaluation of drainmod using saturated hydraulic conductivity estimated by a pedotransfer function model. *Agricultural Water Management*, 95(10):1135–1143, 2008.
- J. E. Parsons, R. W. Skaggs, and J. W. Gilliam. Pesticide fate with drainmod/creams. In *Proceedings of CREAMS Symposium*, pages 123–125, GA, p. Athens.
- John R Philip. An infiltration equation with physical significance. *Soil Science*, 77 (2):153–158, 1954.
- W.L. Powers and T.A.H. Teeter. The drainage of white land and other wet lands in oregon. *Oregon Agricultural College Experimental Station Bulletin*, 137, 1916.

- Zhiming Qi, Ranvir Singh, Matthew J Helmers, and Xiaobo Zhou. Evaluating the performance of drainmod using soil hydraulic parameters derived by various methods. *Agricultural Water Management*, 155:48–52, 2015a.
- Zhiming Qi, Ranvir Singh, Matthew J Helmers, and Xiaobo Zhou. Evaluating the performance of drainmod using soil hydraulic parameters derived by various methods. *Agricultural Water Management*, 155:48–52, 2015b.
- C. J. Ravelo, D. L. Reddell, E. A. Hiler, and R. W. Skaggs. Incorporating crop needs into drainage system design. *Transactions of the ASAE*, 25(3):623–629, 1982.
- Mark Reeves and Edward E Miller. Estimating infiltration for erratic rainfall. *Water Resources Research*, 11(1):102–110, 1975.
- W. D. Robertson, G. I. Ford, and P. S. Lombardo. Wood-based filter for nitrate removal in septic systems. *Transactions of the ASAE*, 48(1):121–128, 2005.
- W.D. Robertson, D.W. Blowes, C.J. Ptacek, and J.A. Cherry. *Ground Water*, 38: 689–695, 2000.
- W.D. Robertson, J.L. Vogan, and P. Lombardo. Nitrate removal rates in a 15-year old permeable reactive barrier treating septic system nitrate. *Ground Water Monit. Remediat.*, 28:65–74, 2008.
- W.D. Robertson, C.J. Ptacek, and S.J. Brown. Rates of nitrate and perchlorate removal in a 5-year-old wood particle reactor treating agricultural drainage. *Ground Water Monit. Remediat.*, 29:87–94, 2009.
- Marcel G Schaap, Feike J Leij, and Martinus Th Van Genuchten. Rosetta: a computer program for estimating soil hydraulic parameters with hierarchical pedo-transfer functions. *Journal of hydrology*, 251(3):163–176, 2001.
- Louis A. Schipper and Maja Vojvodić-Vuković. Five years of nitrate removal, denitrification and carbon dynamics in a denitrification wall. *Water Research*, 35(14): 3473–3477, 2001.
- Louis A. Schipper, Will D. Robertson, Arthur J. Gold, Dan B. Jaynes, and Stewart C. Cameron. Denitrifying bioreactors—an approach for reducing nitrate loads to receiving waters. *Ecological Engineering*, 36(11):1532–43, 2010. doi: 10.1016/j.ecoleng.2010.04.008.

- R. M. Seymour, R. W. Skaggs, and R. O. Evans. Corn yield response to planting date. *Transactions of the ASAE*, 35(3):865–870, 1992.
- Rita Shih, William D Robertson, Sherry L Schiff, and David L Rudolph. Nitrate controls methyl mercury production in a streambed bioreactor. *Journal of environmental quality*, 40(5):1586–1592, 2011.
- A. Shirmohammadi, D. L. Thomas, and M. C. Smith. Drainage-subirrigation design for pelham loamy sand. 1991. *Trans. of the ASAE*, 34(1):73–80, 1991.
- M. B. Shukla, S. O. Prahser, A. Madani, and G. P. Gupta. Field evaluation of drainmod in atlantic canada. *Canadian Agricultural Engineering*, 36(4):205–213, 1994.
- M. Singh, S. O. Prasher, C. S. Tan, and C. M. Tejawat. Evaluation of drainmod for southern ontario conditions. *Canadian Water Resources Journal*, 19(4):313–326, 1994.
- R. Singh, M. J. Helmers, and Zhiming Qi. Calibration and validation of drainmod to design subsurface drainage systems for iowa’s tile landscapes. *Agricultural Water Management*, 85(3):221–32, 2006. doi: 10.1016/j.agwat.2006.05.013.
- R. W. Skaggs. Evaluation of water table control systems using a water management model. *Proceedings, Third National Drainage Symposium, ASAE, St*, pages 61–68, 1976.
- R. W. Skaggs. Combination surface-subsurface drainage systems for humid regions. *Journal of Irrigation and Drainage Division, ASCE.*, 106:265–283, 1980a.
- R. W. Skaggs. Field evaluation of a water management simulation model. *Transactions of the ASAE*, 25(3):666–674, 1982a.
- R. W. Skaggs. Simulation of drainage system performance as affected by irrigation management. In *Keynote*, pages 61–77, Vol. I. Proceedings of Symposium on Land Drainage for Salinity Control in Arid and Semi-arid Regions, Drainage Research Institute, WRC, Cairo, Egypt, 1990.
- R. W. Skaggs and J. W. Gilliam. Effect of drainage system design and operation on nitrate transport. *Transactions of the ASAE.*, 24(4):929–934, 1981.
- R. W. Skaggs and J. W. Gilliam. Modeling subsurface drainage and water management systems to alleviate water quality problems. pages 295–319. Elsevier, Amsterdam, 1986.

- R. W. Skaggs and A. Nassehzadeh-Tabrizi. Design and drainage systems for land treatment of wastewater. *SCE J. Irr. and Drainage Div*, 108:196–211, 1982.
- R. W. Skaggs, N. R. Fausey, and B. N. Nolte. Water management model evaluation for north central ohio. *Transactions of the ASAE*, 24(4):922–928, 1981.
- R. W. Skaggs, S. Hardjoamidjojo, E. H. Wiser, and E. A. Hiler. Simulation of crop response to surface and subsurface drainage systems. *Transactions of the ASAE*, 25(6):1673–1678, 1982.
- R. W. Skaggs, J. W. Gilliam, and R. O. Evans. A computer simulation study of pocosin hydrology. *Wetlands*, 11:399–416, 1991.
- R. W. Skaggs, M. A. Breve, A. Mohammad, J. E. Parsons, and J. W. Gilliam. Simulation of drainage water quality with drainmod. In E. Lorre, editor, *Subsurface Drainage Simulation Models, Transactions of Workshop, 15th Congress ICID*,, pages 183–220. The Hague, 1993.
- R. W. Skaggs, M. A. Youssef, and G. M. Chescheir. *DRAINMOD: Model Use, Calibration, and Validation. Trans. of the ASABE 55(4):1509-1522*. Application of DRAINMOD for Analysis of Wetland Hydrology, 2012.
- R. Wayne Skaggs and Yau-K Tang. Saturated and unsaturated flow to parallel drains. *Journal of the Irrigation and Drainage Division*, 102(2):221–238, 1976.
- R.W. Skaggs. A water management model for shallow water table soil. *Technical report No. 134 of the Water Resources Research Institute of the University of North Carolina, NC State University, Raleigh, NC, USA*, page 329 pp., 1978.
- R.W. Skaggs. Drainmod reference report. methods for design and evaluation of drainage-water management systems for soils with high water tables. *USDA-SCS*, page 329, 1980b.
- R.W. Skaggs. Field evaluation of a water management simulation model. *Trans. ASAE*, pages 666–674, 1982b.
- Vaclav Smil. *Enriching the earth: Fritz Haber, Carl Bosch, and the transformation of world food production*. MIT press, 2004.
- Charles Warren Thornthwaite. An approach toward a rational classification of climate. *Geographical review*, 38(1):55–94, 1948.

- CW Thornthwaite and JR Mather. Introduction and tables for computing potential evapotranspiration and water balanced. *Publ In Climatology. Drexel Institute Of Technology. Laboratory Of Climatology*, 10(31), 1957.
- U.S. Natural Resources Conservation Service. National engineering handbook. volume 16:Drainage of Agricultural Land, chapter 4:Subsurface Drainage. Washington, D.C., 1985.
- M Th Van Genuchten. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil science society of America journal*, 44(5):892–898, 1980.
- Peter M Vitousek, John D Aber, Robert W Howarth, Gene E Likens, Pamela A Matson, David W Schindler, William H Schlesinger, and David G Tilman. Human alteration of the global nitrogen cycle: sources and consequences. *Ecological applications*, 7(3):737–750, 1997.
- Kristina Warren. *Transport of water and solutes from tile-drained fields in the Willamette Valley, Oregon: A Field scale study*. PhD thesis, 2002.
- M. Y. Youssef, R. W. Skaggs, G. M. Chescheir, and J. W. Gilliam. The nitrogen simulation model, drainmod-n ii. *Trans. ASAE*, 48(2):611–626, 2005.
- Jarvis N.J. Van Genuchten M.T. imnek, J. and A. Grdens. Review and comparison of models for describing non-equilibrium and preferential flow and transport in the vadose zone. *Journal of Hydrology*, 272:14–35, 2003.
- Schipper, L.A., Barkle, G.F., Vojvodic-Vukovic, M., 2005. Maximum rates of nitrate removal in a denitrification wall. *J. Environ. Qual.* 34, 12701276.
- Metcalf Eddy, Inc. Wastewater Engineering: Treatment and Reuse -Fourth Edition. New York: McGraw-Hill. 2003.
- Miner, Ronald J, Humenik, Frank J, Overcash, Michael R. 2000. Managing Livestock Wastes to Preserve Environmental Quality. Ames, Iowa: Iowa University State Press.

## APPENDIX



## A VBA Code

### Option Explicit

```
'#####
'#####
'The following "Public" variables are global variables. They can be accessed anywhere within the
'module.
'A number of these are stored in a TAG file associated with the project. When the project is
'reopened, these variables are re-declared in memory by reading them from the TAG file
'(see sub TAG_initialize)
'Travis A Grohman, travis.a.grohman@gmail.com, May 2017
'#####
Private Declare PtrSafe Function GetWindowsDirectory Lib "kernel32" ()
```

Public folderpath As Variant *'global path to model's local folder*

Public projectname As String *'this is the user defined name of the project. All files*  
*'(weather,soil,gen,prj,dmn>tag) are named with this string at*  
*'the base (e.g. if there are 4 soil files created the name of*  
*'the 4th SIN file would be projectname\_4.SIN)*

Public seriesnumber As Integer *'global # of distinct soil series in the area of interest*

Public simstart\_month As String *'starting month of simulation (defined by weather input) as*  
*'string (e.g. "January")*

Public simstart\_day As Integer *'starting day-of-month (e.g. 1-31) for simulation*  
*'(defined by weather input).*

Public simstart\_year As String *'4 digit starting year of simulation defined by weather input*

Public simend\_month As String *'ending month of simulation (defined by weather input) as*  
*'string (e.g. "January")*

Public simend\_day As Integer

Public simend\_year As String

```

Public LAT_deg As Double 'latitude minutes
Public LAT_min As Double 'latitude seconds
Public LONG_deg As Double 'longitude minutes
Public LONG_min As Double 'Longitude seconds
Public ncrops As Integer 'number of crop yield files

Public GENarray(1 To 100) As String
Public PRJarray(1 To 62) As String
Public DMNarray(1 To 100) As String
Public TAGarray(0 To 50) As String
Public TAGheader() As String 'this is an array which holds elements the fist line of
                                'tag file (or TAGarray(0)) separated by commas.

Public weather_unit_warning As Boolean 'this flag is used to warn the user if they
                                        'change units after importing data weather data

'#####
'The Subroutine Clear_Globals simply clears all values currently declared for global variables.
'#####
Public Sub Clear_Globals()
folderpath = Empty
projectname = Empty
seriesnumber = Empty
simstart_month = Empty
simstart_day = Empty

```

```

simstart_year = Empty
simend_month = Empty
simend_day = Empty
simend_year = Empty
LAT_deg = Empty
LAT_min = Empty
LONG_deg = Empty
LONG_min = Empty
ncrops = Empty
Erase GENarray
Erase PRJarray
Erase DMNarray
Erase TAGarray
Erase TAGheader
weather_unit_warning = False
End Sub

```

```

'#####

```

```

'This subroutine uses a .TAG file created in previous project to set the global variables.

```

```

'In this way, the TAG file allows users to pick up where they left off in a previous project.

```

```

'#####

```

```

Public Sub TAG_initialize(TAG_path As String)
Dim filenumber As Integer 'a numeric alias for the TAG file used for Line Input routines
Dim i As Integer 'a generic counter
Dim str As String 'a generic string
On Error GoTo errormsg 'Set error handle

```

```

Call GetFolderPath

filenumber = FreeFile 'Set the numeric alias of the TAG file to be the next available integer

Open TAG_path For Input As #filenumber 'set TAG file's alias as filenumber, open it for
    ↪ reading
i = 0 'initialize counter at 0
Do While Not EOF(filenumber) ' Loop until end of file.
    Line Input #filenumber, TAGarray(i) ' read line i from file and set
                                                'it to element i of TAGarray

    i = i + 1 'count
Loop
Close #filenumber

Call TAG_header_read 'splits element 0 (headerline) of TAGarray by commas and sets this
                        'header element array as TAGheader

If TAGheader(12) <> " " Then
    ncrops = TAGheader(12)
Else: Sheets("crops").Activate
End If

If TAGheader(11) <> " " Then
    LONG_min = TAGheader(11)
Else: Sheets("Weather_Input").Activate
End If

If TAGheader(10) <> " " Then
    LONG_deg = TAGheader(10)

```

```
Else: Sheets("weather_input").Activate
```

```
End If
```

```
If TAGheader(9) <> " " Then
```

```
    LAT_min = TAGheader(9)
```

```
Else: Sheets("Weather.Input").Activate
```

```
End If
```

```
If TAGheader(8) <> " " Then
```

```
    LAT_deg = TAGheader(8)
```

```
Else: Sheets("Weather.Input").Activate
```

```
End If
```

```
If TAGheader(7) <> " " Then
```

```
    simend_year = TAGheader(7)
```

```
Else: Sheets("Weather.Input").Activate
```

```
End If
```

```
If TAGheader(6) <> " " Then
```

```
    simend_day = TAGheader(6)
```

```
Else: Sheets("Weather.Input").Activate
```

```
End If
```

```
If TAGheader(5) <> " " Then
```

```
    simend_month = TAGheader(5)
```

```
Else: Sheets("Weather.Input").Activate
```

```
End If
```

```

If TAGheader(4) <> " " Then
    simstart_year = TAGheader(4)
Else: Sheets("Weather_Input").Activate
End If

If TAGheader(3) <> " " Then
    simstart_day = TAGheader(3)
Else: Sheets("Weather_Input").Activate
End If

If TAGheader(2) <> " " Then
    simstart_month = TAGheader(2)
Else: Sheets("Weather_Input").Activate
End If

If TAGheader(1) <> " " Then
    seriesnumber = TAGheader(1)
Else: Sheets("ros_input").Activate
End If

If TAGheader(0) <> " " Then
    projectname = TAGheader(0)
Else: Sheets("ros_input").Activate
End If

weather_unit_warning = False

Exit Sub

```

```
errmsg: MsgBox "Could not initialize from selected TAG file."
```

```
End Sub
```

```
'#####
```

```
'This sub sets the global variable "folderpath" as the path to the active folder
```

```
'(the folder containing this workbook).
```

```
'locations of input and output files to the model are referenced by this folderpath.
```

```
'Travis Grohman, travis.a.grohman@gmail.com, May 2017
```

```
'#####
```

```
Public Sub GetFolderPath()
```

```
    folderpath = ActiveWorkbook.Path
```

```
End Sub
```

```
'#####
```

```
'This Subroutine accepts a user-defined project name for a new project.
```

```
'#####
```

```
Public Sub GetFirstProjectName()
```

```
On Error GoTo errmsg
```

```
Dim str As String 'generic string
```

```
Call GetFolderPath
```

```
projectname = InputBox("What would you like to name this project?")
```

```
projectname = "_" & projectname
```



```

str = folderpath & "\" & projectname & ".TAG"
If Len(str) > 65 Then

    MsgBox "Error: The path to the TAG file being created is too long. It must be less" & _
    " than 65 charaters to avoid fixed-width errors. The path you have chosen is: " & """" & _
    folderpath & "\" & projectname & ".TAG" & """" & " Which is " & Len(str) & _
    " characters. Please choose a different projectname , " & _
    "rename the project folder, or move the project folder closer to the directory. " & _
    " (The projectfolder is: " & """" & folderpath & """" & " ). Sorry for the inconvenience."
projectname = InputBox("What would you like to name this project?")
Else

Call TAG_create_file(projectname)
End If
Exit Sub

errormsg: MsgBox "Failure: A project name was not set. Please try again by running the " & _
"Soils routine on worksheet Ros_Inputs."
End Sub


Public Sub getprojectname()
Dim str As String
Dim TAGfilepath As String

If Len(projectname) < 1 Then

    ' On Error GoTo errormsg

```

Call GetFolderPath

If Dir(folderpath & "\INPUTS\") = Empty Then

MsgBox "ERROR: Could not find an *'inputs' folder in the current directory. This*

→ *worksheet should be kept in the original project folder it was downloaded with*

→ *and, while the project folder name may be changed," & \_*

*"the folders within it should not be renamed."*

Exit Sub

Else

ChDir folderpath & "\inputs\"

End If

MsgBox ("Please choose the TAG file associated with your current project. If there you are

→ starting a new project, you must run the soils routine first.")

TAGfilepath = Application.GetOpenFilename("Den.Bioreactor file,\*.TAG")

If Len(TAGfilepath) > 65 Then

MsgBox ("Error: The path to your TAG file for this project is too long—it must be

→ less than 65 characters to avoid fixed width errors. The full base path to your

→ project is: " & TAGfilepath & \_

" Please move the project folder closer to the root directory and/or renaming the

→ project folder and try again. Do not rename the TAG file (it is used to find all

→ associated files). Sorry for the inconvenience.")

Exit Sub

End If

Call TAG\_initialize(TAGfilepath)

End If

Exit Sub

errmsg:

'MsgBox "TAG file was not selected."

'End If

End Sub

'#####

'#This sub allows the user to find their downloaded database (.mdb) file using the windows file  
 ➞ explorer.

'#The .mdb file is meant to be a database downloaded from the web soil survey website.

'#Travis Grohman May 2017

'#####

Public Sub browse\_wss\_mdb.button()

Dim wssfilepath As Variant

    wssfilepath = Application.GetOpenFilename("Access Databases,\*.MDB")

    Sheets("soils").Range("wss\_mdb\_file\_path").Value = wssfilepath

End Sub

```
'#####
'Author: Travis A. Grohman,Oregon State University, travis.a.grohman@gmail.com, 5/22/17

'The purpose of the subroutine "SoilSQL()" is to Query the database (.mdb) file downloaded
    ↳ from the NRCS Web Soil Survey
'for information relevent to rosetta (.ros) Drainmod Input files.
'After the query, lines rosetta predicated values corresponding to each chkey returned by the
    ↳ query are called from a textfile
'containing all rosetta predictions for a given state.
'A combination the rosetta predictions( theta_sat, theta_res, alpha, n ,k_sat, tortosity) and the
    ↳ queried info (number of layers in each soil series,depth to each layer)
'are written to the sheet "ros_inputs".
,
'#####
```

Sub SoilSQL(soilDatabasePath As String)

```
Dim cnt As New ADODB.Connection "declare in memory connection to database object
Dim rst As New Recordset "declare in memory our recordset as rst
Dim strConnectStr As String "declare in memory our connection string
Dim Qry As String "declare in memory our query string
Dim n_lastrow_in_soildata As Integer
Dim r As Range 'generic range
Dim i As Integer 'generic counter
```

```
Dim doub As Double 'generic double
```

```
Dim str As String 'generic string
```

```
Dim n_SoilSeries As Integer 'Total number of soil series in area of interest we need to  

    ↪ drainmod files for
```

```
'#####
```

```
'#Set application.calculation to manual.
```

```
'This keeps excel from recalculating the entire workbook anytime something is changed.
```

```
'It drastically speeds up calc time when there are workbook operations involved.
```

```
'#####
```

```
Application.Calculation = xlCalculationManual
```

```
'#####
```

```
Call GetFolderPath 'the path to the Bioreactor_Model folder is set the the active workbook
```

→ path. This workbook needs to be placed in the Bioreactor\_Model folder

```
'#####
'the database provider to use in running an SQL query on the access .mdb file is architecture
    → specific. If the version of excel
'being run is 32 bit, we use the Jet provider. If the version of excel being run is 64 bit, we use
    → the ACE provider.
'This next part runs a pre-compile c# routine asking if the excel architecture is Win64 or
    → Win32. The boolean Is64Bit is set to true for
'Win64 and false for Win32. The results are used to connect to the correct provider.
'#####
Dim Is64Bit As Boolean

#If Win64 Then
    Is64Bit = True
#End If

Dim OLEDBprovider As String

If Is64Bit = True Then
    OLEDBprovider = "Microsoft.ACE.OLEDB.12.0"
    str = "You are running a 64 bit version of excel.Database query will be performed using
```

```

        ↪ database interface 'Microsoft.ACE.OLEDB.12.0'
Else
OLEDBprovider = "Microsoft.Jet.OLEDB.4.0"
str = "You are running a 32 bit version of excel.Database query will be performed using
        ↪ database interface 'Microsoft.Jet.OLEDB.4.0'"
End If

MsgBox "64 32 test results: " & str

strConnectStr = "Provider = " & OLEDBprovider & ";Data Source=" & """" &
        ↪ soilDatabasePath & """" & ","

'#####

```





cnt.Open strConnectStr 'open the connection through provider

rst.Open Qry, cnt 'put results of query in recordset

Worksheets("SoilData").Activate 'bring the user to the sheet "SoilData"

Worksheets("SoilData").Range("C2:S2").CopyFromRecordset rst 'copy info from recordset

↪ into SoilData workbook

'###Enter Headers

Sheets("SoilData").Range("C1") = "Map Unit (MU)Symbol"

Sheets("SoilData").Range("D1") = "chkey (horizion id)"

Sheets("SoilData").Range("E1") = "Area Symbol"

Sheets("SoilData").Range("F1") = "Area Name"

Sheets("SoilData").Range("G1") = "Series (MU) name"

Sheets("SoilData").Range("H1") = "Component %"

Sheets("SoilData").Range("I1") = "depth to top of layer (cm)"

Sheets("SoilData").Range("J1") = "Depth to bottom of layer (cm)"

Sheets("SoilData").Range("K1") = "Slope"

Sheets("SoilData").Range("L1") = "Total Project Area(Acres)"

Sheets("SoilData").Range("M1") = "Total Series (MU) Area (Acres)"

Sheets("SoilData").Range("N1") = "Saturated Conductivity (cm/day)"

Sheets("SoilData").Range("O1") = "% Sand"

Sheets("SoilData").Range("P1") = "% Silt"

Sheets("SoilData").Range("Q1") = "% Clay"

Sheets("SoilData").Range("R1") = "Bulk Density (g/cm3)"

Sheets("SoilData").Range("S1") = "pH"

Sheets("SoilData").Range("T1") = "Theta 1/3 bar (cm3/cm3)"

Sheets("SoilData").Range("U1") = "Theta 15 bar (cm3/cm3)"

```

#####
'##Great now lets count how many rows of data we imported so we know what to set ranges to
    ↪ in following
'## procedures
#####
Set r = Worksheets("SoilData").Range("C2:C10000")
'^set the range to the columns where we keep mapunit symbol (musym)
'(arbitrary which column really, but I don't plan on moving the musym as I am writing this)

n_lastrow_in_soildata = Application.WorksheetFunction.CountA(r) + 1

```

```

Worksheets("SoilData").Range("B1") = "Layer Number"
Worksheets("SoilData").Range("A2").Value = 1
Worksheets("SoilData").Range("B2").Value = 1

For i = 3 To n_lastrow_in_soildata

    str = "B" & CStr(i)

    Sheets("SoilData").Range(str).FormulaR1C1 = "=IF(RC3=R[-1]C3,R[-1]C+1,1)"
    doub = Sheets("SoilData").Range(str).Value
    Sheets("SoilData").Range(str) = doub

    str = "A" & CStr(i)
    Sheets("SoilData").Range(str).FormulaR1C1 = "=IF(RC2=1,R[-1]C+1,R[-1]C)"
    doub = Sheets("SoilData").Range(str).Value
    Sheets("SoilData").Range(str) = doub

Next i

''this for loop just goes through and counts number of layers per series (column B) and the
    ↪ total number of soil series in the file (column A)

''counting layers works by going down the musym (column C) which should be sorted by the
    ↪ SQL query.

```

*''it counts until there is a new value in the C column; on a new value it restarts count. it puts*

*↪ these in the B column and changes the formulas to values (double)*

*'' total number of soil series count just starts at one and everytime there is a layer number*

*↪ value of one in column B, it adds one.*

i = 2

For i = 2 To n\_lastrow\_in\_soildata

str = "N" & CStr(i)

doub = Sheets("SoilData").Range(str).Value \* 8.64

Sheets("SoilData").Range(str) = Format(doub, "##,##0.00")

Next i

*'^ this loop converts saturated conductivity from um/s to cm/day*

Set r = Worksheets("SoilData").Range("A2:A" & CStr(n\_lastrow\_in\_soildata))

```

'#####
'##now get the total the number of distinct soil series we have present in our NRCS WSS area
    ↳ of interest
'#####
n_SoilSeries = CountDistinct(r)
'^call the funciton "CountDistint" with input r as Range. Subtract one for the blank distinct.
'^ n_SoilSeries is the name of an integer representing the number of distict soil profiles in the
'NRCS WSS downloaded database we are using for an input

Worksheets("SoilData").Range("A1") = "Soil Series number of " & CStr(n_SoilSeries) & "
    ↳ Total"
'^we will go ahead and tell the user what we counted in the header

Dim filenumber As Integer
Dim rowi As Integer
Dim rowcount As Integer
Dim nextrow As Integer
Dim fs As Object
Dim a As Object
Dim filepath As String
Dim rosstring As String
Dim botdepthstring As String
Dim textstring As String

```

```
'#####
```

' This part finds the lines of rosetta predictions corresponding to each layer (each chkey) in the

→ Area of interest (in the selected .mdb file) and writes them to the sheet "ros\_input".

'it also copies some info from the sheet "soildata" over to the sheet ros\_inputs

```
'#####
```

Dim arrstr() As String 'array string. the line of data is split at tabs and each value is assigned

→ to a position in the array.

Erase arrstr() 'clear arrstr if there is something lingering in there

Call GetFolderPath 'make sure folder path is current

Worksheets("ros\_input").Cells.ClearContents 'clear whatever is in the ros\_input worksheet to

→ make room for our current project rosetta inputs

'Now lets add headers so we can tell what is what

Sheets("ros\_input").Range("A1").Value = Sheets("soildata").Range("A1").Value

```

Sheets("ros_input").Range("B1").Value = Sheets("soildata").Range("B1").Value
Sheets("ros_input").Range("C1").Value = "chkey"
Sheets("ros_input").Range("D1").Value = "bottom depth of layer (cm)"
Sheets("ros_input").Range("E1").Value = "theta_residual"
Sheets("ros_input").Range("E2").Value = "cm3/cm3"
Sheets("ros_input").Range("F1").Value = "theta_sat"
Sheets("ros_input").Range("F2").Value = "cm3/cm3"
Sheets("ros_input").Range("G1").Value = "alpha"
Sheets("ros_input").Range("G2").Value = "1/cm"
Sheets("ros_input").Range("H1").Value = "n (van genuchten mulaem fitting param)"
Sheets("ros_input").Range("H2").Value = "(unitless)"
Sheets("ros_input").Range("I1").Value = "ksat"
Sheets("ros_input").Range("I2").Value = "cm/day"
Sheets("ros_input").Range("J1").Value = "Ko(matching point at saturation)"
Sheets("ros_input").Range("J2").Value = "cm/day"
Sheets("ros_input").Range("K1").Value = "tortosity(L)"
Sheets("ros_input").Range("K2").Value = "(unitless)"
Sheets("ros_input").Range("L1").Value = "theta_res uncertainty"
Sheets("ros_input").Range("L2").Value = "cm3/cm3"
Sheets("ros_input").Range("M1").Value = "theta_sat uncertainty"
Sheets("ros_input").Range("M2").Value = "cm3/cm3"
Sheets("ros_input").Range("N1").Value = "alpha uncertainty"
Sheets("ros_input").Range("N2").Value = "1/cm"
Sheets("ros_input").Range("O1").Value = "n uncertainty"
Sheets("ros_input").Range("O2").Value = "(unitless)"
Sheets("ros_input").Range("P1").Value = "ksat uncertainty"
Sheets("ros_input").Range("P2").Value = "cm/day"
Sheets("ros_input").Range("Q1").Value = "ko uncertainty"

```

```

Sheets("ros_input").Range("Q2").Value = "cm/day"
Sheets("ros_input").Range("R1").Value = "tortuosity uncertainty"
Sheets("ros_input").Range("R2").Value = "(unitless)"

```

*'Now populate sheet ros\_input with rosetta information.*

*'Read info from rosetta predictions text file and copy query info from sheet "Soildata"*

```

For i = 3 To (n_lastrow_in_soildata + 1)
    str = Sheets("SoilData").Range("D" & CStr(i - 1)).Value

```

```

    Sheets("ros_input").Activate

```

```

    'MsgBox "looking rosetta predictions of chkey " & str

```

```

    If find_line_with_chkey(str) = -99.9 Then ' -99.9 is the error flag I set for a missing

```

```

        ↪ chkey.

```

*'This block of code has the routine for what do*

```

        ↪ do in the case of a missing chkey.

```

*'If the top layer in a series is missing (most*

```

        ↪ common case):

```

*'the user is notified, ignored, bottom depth*

```

        ↪ of missing chkey layer subtracted

```

```

        ↪ from top depth of

```

*'next layer in series.*

*'If the bottom layer in a series is missing:*

```

        'user notified, it is deleted completely.

```

*'If a middle layer in a series is missing:*



*'user notified, depth can be split between*

*↪ top and bottom or caN Be*

*↪ incorporated totally into either.*

Else

End If

arrstr() = Split(find\_line\_with\_chkey(str), vbTab)

Sheets("ros\_input").Range("E" & CStr(i)).Value = arrstr(1) *'residual moisture content(theta*

*↪ res) column E*

Sheets("ros\_input").Range("F" & CStr(i)).Value = arrstr(2) *' saturated moisture content(theta*

*↪ sat) in column F*

Sheets("ros\_input").Range("G" & CStr(i)).Value = arrstr(3) *' alpha column G*

Sheets("ros\_input").Range("H" & CStr(i)).Value = arrstr(4) *'n column H*

Sheets("ros\_input").Range("I" & CStr(i)).Value = arrstr(5) *'Ksat column I*

Sheets("ros\_input").Range("J" & CStr(i)).Value = arrstr(6) *'Ko (matching point) column J*

Sheets("ros\_input").Range("K" & CStr(i)).Value = arrstr(7) *'tortuosit L in column K*

Sheets("ros\_input").Range("L" & CStr(i)).Value = arrstr(8) *'theta res uncertainty column L*

Sheets("ros\_input").Range("M" & CStr(i)).Value = arrstr(9) *' theta sat uncertainty in column M*

Sheets("ros\_input").Range("N" & CStr(i)).Value = arrstr(10) *' alpha uncertainty in column N*

Sheets("ros\_input").Range("O" & CStr(i)).Value = arrstr(11) *'n uncertainty in column O*

Sheets("ros\_input").Range("P" & CStr(i)).Value = arrstr(12) *'ksat uncertainty to column P*

Sheets("ros\_input").Range("Q" & CStr(i)).Value = arrstr(13) *'Ko uncertainty to column Q*

Sheets("ros\_input").Range("R" & CStr(i)).Value = arrstr(14) *'L uncertainty to column R*

```

'#####
'copy over infomation from worksheet "soildata"
'#####

    Sheets("ros_input").Range("C" & CStr(i)).Value = str 'copy chkey from sheet "soil
        ↳ data" column D to sheet "ros_input" column c

    Sheets("ros_input").Range("A" & CStr(i)).Value = Sheets("SoilData").Range("A" &
        ↳ CStr(i - 1)).Value 'copy soil series number to column A

    Sheets("ros_input").Range("B" & CStr(i)).Value = Sheets("SoilData").Range("B" &
        ↳ CStr(i - 1)).Value 'copy soil layer number copy soil layer number to column B

    Sheets("ros_input").Range("D" & CStr(i)).Value = Sheets("SoilData").Range("J" &
        ↳ CStr(i - 1)).Value 'copy over bottom depth of layer to column D

'On Error GoTo missingchkey(str,i)

Next i

Erase arrstr() 'clean up arrstring
If Sheets("ros_input").Shapes("5layer_suppress").ControlFormat.Value = -4146 Then
Call make5layer
End If
Application.Calculation = xlCalculationAutomatic ' go back to default (Calculating entire
    ↳ workbook when something is changed)

```

```

End Sub

Sub layer_suppress.Click()
Dim yesno As Integer
If Sheets("ros_input").Shapes("5layer_suppress").ControlFormat.Value = 1 Then

yesno = MsgBox("WARNING! By not condensing to 5 layers, Drainmod will completely ignore
    ↪ any" & _
" layers after the fifth. Do you still want to suppress 5-layer" & _
" condensing?", vbYesNo + vbQuestion, "5 Layer Suppress")
If yesno = vbNo Then Sheets("ros_input").Shapes("5layer_suppress").ControlFormat.Value =
    ↪ -4146
End If
End Sub

'#####
'This section finds the most similar sequential layers (by k_lat) and joins them using the
    ↪ depth-weighted average until there are only 5 soil layers (the max Drainmod allocates
    ↪ memory for).
'#####

Public Sub make5layer()
Dim row, lastrow, nseries, i, nlayers As Integer
Dim count, nlayerdelete, n, p, rowtodelete As Integer
Dim depth1, depth2, val1, val2, newval As Double
Dim residual, minresidual As Double
row = 3

```

```

Do Until IsEmpty(Sheets("ros_input").Range("A" & CStr(row)))
row = row + 1
Loop
lastrow = row - 1
'MsgBox lastrow
nseries = CountDistinct(Sheets("ros_input").Range("A3:A" & CStr(lastrow)))
'MsgBox nseries

For i = 1 To nseries
'MsgBox "Series number" & i
nlayers = 0
    For count = 3 To lastrow
        If i = CInt(Sheets("ros_input").Range("A" & CStr(count))) Then
            nlayers = nlayers + 1
        End If
    Next count
'MsgBox "nlayers: " & nlayers
    If nlayers >= 6 Then
        nlayerdelete = nlayers - 5
        'MsgBox "removing " & nlayerdelete & " layers"
        minresidual = 1000 'ridiculously large intializing min. residual to beat
        For n = 1 To nlayerdelete
            minresidual = 1000 'ridiculously large intializing min. residual to beat

            For p = 3 To lastrow
                If CInt(Sheets("ros_input").Range("A" & CStr(p))) = i And _
                    CInt(Sheets("ros_input").Range("A" & CStr(p + 1))) = i Then

```

```

residual = CDb1(Sheets("ros_input").Range("J" & CStr(p))) - _
CDbl(Sheets("ros_input").Range("J" & CStr(p + 1)))
residual = Abs(residual)
'MsgBox "row " & p & " residual= " & residual
If residual < minresidual Then
    minresidual = residual
    rowtodelete = p + 1
    'MsgBox "new row to delete = " & rowtodelete
End If

End If

Next p
'MsgBox "deleting row " & rowtodelete
'now the row marked for deleting is merged with the row below it
'by taking depth-averaged weights of each value.
If CInt(Sheets("ros_input").Range("B" & CStr(rowtodelete - 1))) = 1
    ↪ Then
        depth1 = CDb1(Sheets("ros_input").Range("D" & CStr(rowtodelete
            ↪ - 1)).Value)
    Else
        depth1 = CDb1(Sheets("ros_input").Range("D" & CStr(rowtodelete
            ↪ - 1)).Value) -
        - CDb1(Sheets("ros_input").Range("D" & CStr(rowtodelete - 2)))
    End If
    depth2 = CDb1(Sheets("ros_input").Range("D" &
        ↪ CStr(rowtodelete)).Value) -
        - CDb1(Sheets("ros_input").Range("D" & CStr(rowtodelete - 1)))

```

*'bottom depth of layer*

```
Sheets("ros_input").Range("D" & CStr(rowtodelete - 1)) = _
```

```
Sheets("ros_input").Range("D" & CStr(rowtodelete))
```

*'theta res*

```
val1 = CDBl(Sheets("ros_input").Range("E" & CStr(rowtodelete)).Value)
```

```
val2 = CDBl(Sheets("ros_input").Range("E" & CStr(rowtodelete - 1)).Value)
```

```
newval = ((val1 * depth1) + (val2 * depth2)) / (depth1 + depth2)
```

```
Sheets("ros_input").Range("E" & CStr(rowtodelete - 1)) = newval
```

*'theta sat*

```
val1 = CDBl(Sheets("ros_input").Range("F" & CStr(rowtodelete)).Value)
```

```
val2 = CDBl(Sheets("ros_input").Range("F" & CStr(rowtodelete - 1)).Value)
```

```
newval = ((val1 * depth1) + (val2 * depth2)) / (depth1 + depth2)
```

```
Sheets("ros_input").Range("F" & CStr(rowtodelete - 1)) = newval
```

*'Alpha*

```
val1 = CDBl(Sheets("ros_input").Range("G" & CStr(rowtodelete)).Value)
```

```
val2 = CDBl(Sheets("ros_input").Range("G" & CStr(rowtodelete - 1)).Value)
```

```
newval = ((val1 * depth1) + (val2 * depth2)) / (depth1 + depth2)
```

```
Sheets("ros_input").Range("G" & CStr(rowtodelete - 1)) = newval
```

*'N*

```
val1 = CDBl(Sheets("ros_input").Range("H" & CStr(rowtodelete)).Value)
```

```
val2 = CDBl(Sheets("ros_input").Range("H" & CStr(rowtodelete - 1)).Value)
```

```
newval = ((val1 * depth1) + (val2 * depth2)) / (depth1 + depth2)
```

```
Sheets("ros_input").Range("H" & CStr(rowtodelete - 1)) = newval
```

*'KSAT*

```
val1 = CDBl(Sheets("ros_input").Range("I" & CStr(rowtodelete)).Value)
```

```
val2 = CDBl(Sheets("ros_input").Range("I" & CStr(rowtodelete - 1)).Value)
```

```
newval = ((val1 * depth1) + (val2 * depth2)) / (depth1 + depth2)
```

```
Sheets("ros_input").Range("I" & CStr(rowtodelete - 1)) = newval
```

```

'matching point conductivity
val1 = CDbl(Sheets("ros_input").Range("J" & CStr(rowtodelete)).Value)
val2 = CDbl(Sheets("ros_input").Range("J" & CStr(rowtodelete - 1)).Value)
newval = ((val1 * depth1) + (val2 * depth2)) / (depth1 + depth2)
Sheets("ros_input").Range("J" & CStr(rowtodelete - 1)) = newval

'tortosity
val1 = CDbl(Sheets("ros_input").Range("K" & CStr(rowtodelete)).Value)
val2 = CDbl(Sheets("ros_input").Range("K" & CStr(rowtodelete - 1)).Value)
newval = ((val1 * depth1) + (val2 * depth2)) / (depth1 + depth2)
Sheets("ros_input").Range("K" & CStr(rowtodelete - 1)) = newval


        Sheets("ros_input").Rows(rowtodelete).Delete
    Next n
End If

Next i

'Now one last loop to re-order the layer numbers
i = 3 're-initialize the i counter
Do Until IsEmpty(Sheets("ros_input").Range("A" & CStr(i)))
    'for each row with data
    If Sheets("ros_input").Range("A" & CStr(i)) <> -
        Sheets("ros_input").Range("A" & CStr(i - 1)) Then
        'if it is the first layer of the series
        Sheets("ros_input").Range("B" & CStr(i)) = 1
        'layer number is set to one
    Else

```

```

'otherwise
Sheets("ros_input").Range("B" & CStr(i)) = _
CInt(Sheets("ros_input").Range("B" & CStr(i - 1))) + 1
'layer number is the previous layernumber + 1
End If
i = i + 1
Loop

End Sub

'This code counts the number of distinct values in a given range passed to it.
'Adapted from
    ↪ https://stackoverflow.com/questions/26980877/function-to-count-distinct-values-in-
    ↪ a-column-range
'user2362937, Dec 16th, 20016.

Public Function CountDistinct(r As Range) As Long
'' DIM = declare in memory

Dim col As Collection
Dim arr As Variant
Dim x As Long
Dim y As Long
Set col = New Collection

'' setting a Variant = Range will fill the Variant with a 2 dimensional array of the values of the

```



```

    ➡ range!
arr = r
'' skip the errors that are raised
On Error Resume Next
'' loop over all of the elements.
'' UBound is a built in VBA Function that gives you the largest value of an array.
    For x = 1 To UBound(arr, 1)
        For y = 1 To UBound(arr, 2)
            '' try to add the value in arr to the collection
            col.Add 0, CStr(arr(x, y))

            '' every time the collection runs into a value it has already added,
            '' it will raise an error.

            'uncomment the below to see why we are turning off errors
            'Debug.Print Err.Number, Err.Description
        Next
    Next
Next
'' turn errors back on.
On Error GoTo 0
''set the function name to the value you want the formula to return
CountDistinct = col.count
'' The next parts should be handled by VBA automatically but it is good to explicitly clean up.
Set col = Nothing
Set arr = Nothing
Set r = Nothing
End Function

```

```

'#####
'Author Travis Grohman. June 2017. Oregon State Univeristy.
'This function returns line beginning with the chkey passed to it from a text file.
'#
'#It first finds the text file named after the two letter state identifier (e.g. OR.txt for Oregon)
'#It then opens that file and reads each line until the first 8 characters match the chkey string
    → passed to it.
'#Upon a match, it exits and returns the line with the matching chkey.
'# The input file is meant to be a text file containing rosetta predictions for each unique soil
    → layer.
'# Use the included workbook "onestateperfile_rosoutput.xlsm" or the included module
    → "exportfile.bas" to create input text file from raw rosetta output.
'# They are both in the folder "vba_sheets"
'#####

```

```
Public Function find_line_with_chkey(chkey As String)
```

```

Dim areasymp As String
areasymp = Sheets("Soildata").Range("E3") ''Sets cell E3 of worksheet soildata as the location
    ↪ identifier
Dim stringfilename As String
Dim errorflag As Boolean
errorflag = True

    stringfilename = folderpath & "\soils\soil_input_files\" & Left(areasymp, 2) & ".txt"
    Dim number, i As Long
    number = FreeFile 'set number as the next free number available to set as a numbered file
        ↪ location.
    Dim str1, str2 As String
    Open stringfilename For Input As number 'The file set as stringfilename (state soils file) is
        ↪ now input "#number"
    i = 0 'intialize i at zero
Do While Not EOF(number) 'search until the end of the file
    Line Input #number, str1
    str2 = Left(str1, Len(chkey))
    i = i + 1
    If chkey = str2 Then
        errorflag = False
        'MsgBox ("found chkey " & str2 & "on line number " & CStr(i)) 'de-bugging messages
        'MsgBox (str1)
        find_line_with_chkey = str1
        Exit Do

    End If
Loop

```

```

    If errorflag = True Then
        find_line_with_chkey = -99.9 ' This value is being returned as an error flag
    End If
    Close number

End Function

'#####
'#Author Travis Grohman. May 2017. Oregon State Univeristy.
'# This uses the information contained in the sheet "ros_inputs" to create a properly formatted
    ↳ .ros file for each distinct soil series in the given area of interest (the .mdb file used.)
'# The program DMSOILP.EXE is then run in the terminal for each .ros file. DMSOILP.EXE
    ↳ creates all soil files needed for Drainmod.
'#####

Public Sub ros_dmsoil()

```

Call GetFolderPath 'the path to the Bioreactor\_Model folder is set the the active workbook path.

Call GetFirstProjectName ' user enters projectname, a TAG file is created

```
Dim i As Long 'generic Long counter (long so we can use large numbers than allowed for integer
    ↪ type)
```

```
Dim count As Integer
```

```
Dim drain_coeff As Double
```

```
Dim layercount As Integer
```

```
Dim filepath, filepathpros, filepathpros_tpl As String
```

```
Dim filenumber, filenumberpros, filenumberpros_tpl As Integer
```

```
Dim nrows As Double
```

```
Dim str As String 'generic string
```

```
Dim fs As Object
```

```
Dim a, b, c As Object
```

```
Dim strshell As String
```

```
Dim retval As Long
```

```
Dim maxrootdepth, avgrootdepth As Double
```

```
nrows = 0
```

```
For i = 3 To 100000000
```

```
    str = "A" & CStr(i)
```

```
        If IsEmpty(Sheets("ros_input").Range(str)) Then Exit For Else nrows = nrows + 1
```

```
Next i
```

```

MsgBox "Total number of data lines is :" & CStr(nrows) & " (Note that the maximum is
    ↪ 1000000000)"
Call getprojectname

seriesnumber = 0 'intialize seriesnumber

'#####
'# create .ROS files AND .PROS files for each soil series using the data in sheet "ros_inputs"

'#####
    avgrootdepth = InputBox("Average Root Depth (cm)?") 'get avg and max root
        ↪ depth from user
    maxrootdepth = InputBox("Maximum Root Depth (cm)?")

    'create the singular 'TAG' file for the project (See TAG_create_file Sub for details
        ↪ on the TAG file)
    Call TAG_read_file(projectname) 'Put TAG file lines into TAGarray

    Call TAG_header_read 'split the first element in TAGarray ( TAGarray(0) ) by
        ↪ comma delimiters

                                'and put split elements in the dynamic
                                ↪ array TAGheader

TAGheader(14) = maxrootdepth
TAGheader(15) = avgrootdepth

```

```
For i = 3 To (nrows + 2)
```

```
    str = Sheets("ros_input").Range("B" & CStr(i)).Value
```

```
'<><> Begin Special Case: If first layer of a series (MU)
```

```
    If str = 1 Then
```

```
        seriesnumber = seriesnumber + 1 'count the number of series in the project
```

```
        filepath = folderpath & "\soils\" & projectname & CStr(seriesnumber) &
```

```
            ↪ ".ros"
```

```
        Set fs = CreateObject("scripting.filesystemobject")
```

```
        Set a = fs.createTextFile(filepath, True) 'create a .ROS file for this series
```

```
        a.Close 'close because we want to print to it, nto write to it (avoids
```

```
            ↪ problematic mark up added by write function)
```

```
        filepathpros = folderpath & "\soils\" & projectname & CStr(seriesnumber)
```

```
            ↪ & ".pros"
```

```
        Set fs = CreateObject("scripting.filesystemobject")
```

```
        Set b = fs.createTextFile(filepathpros, True) 'create a .PROS file for this
```

```
            ↪ series
```

```
        b.Close 'close because we want to print to it, nto write to it (avoids
```

```
            ↪ problematic mark up added by write function)
```

```
        filepathpros_tpl = folderpath & "\soils\" & "pros" & projectname &
```

```
            ↪ CStr(seriesnumber) & ".tpl"
```

```
        Set fs = CreateObject("scripting.filesystemobject")
```

```
        Set c = fs.createTextFile(filepathpros_tpl, True) 'create a .PROS file for
```

```
            ↪ this series
```

```
        c.Close 'close because we want to print to it, nto write to it (avoids
```

```
            ↪ problematic mark up added by write function)
```

```

Call DMN_create_file(projectname, seriesnumber) 'create DMN file
Call DMN_read_file(projectname, seriesnumber) 'Put it in DMNArray

```

```

Call PRJ_create_file(projectname, seriesnumber) 'Create PRJ file
Call PRJ_read_file(projectname, seriesnumber) 'put it in PRJarray

```

```

Call GEN_create_file(projectname, seriesnumber) 'Create GEN file
Call GEN_read_file(projectname, seriesnumber) 'put it in GENarray

```

```

GENArray(19) = "*** Soils *** "
GENArray(22) = "99 .00"

```

```

filenumber = FreeFile() 'filenumber alias is set as next available integer
Open filepath For Output As #filenumber
filenumberpros = FreeFile()
Open filepathpros For Output As #filenumberpros
filenumberpros_tpl = FreeFile()
Open filepathpros_tpl For Output As #filenumberpros_tpl
Print #filenumber, projectname & CStr(seriesnumber)
Print #filenumberpros, projectname & CStr(seriesnumber)
Print #filenumberpros_tpl, "ptf $"
Print #filenumberpros_tpl, projectname & CStr(seriesnumber)

```

```

layercount = 0

```



```

For count = 3 To (nrows + 2)
    str = Sheets("ros_input").Range("A" & CStr(count)).Value
    If str = seriesnumber Then
        '<><> count number of layers in this Series (MU) <><>'
        layercount = layercount + 1

        '<><> write DMN line from associated column in Sheet
        ↪ 'SOILDATA' for each layer in the following order:
        ' layer number
        'depth to bottom of layer (cm), column J
        'Saturated Hydraulic conductivity (cm/hour),column N
        ↪ [**!!note this is converted from cm/day in this
        ↪ part of the routine!!**],
        'Wilt Point (cm3/cm3),*constant at .30
        '% clay (decimal form, converted from percentage in
        ↪ this routine),column Q
        '% silt (decimal form, converted from percentage in this
        ↪ routine),column P
        'Bulk Density (gm/cm3), column R
        'pH, column S
        'AMMONIA NH4+ Distribution Coefficient (Kd),
        ↪ simple 3 value (2.6,3.0,3.5) based upon clay
        ↪ fraction of soil

        DMNArray(24 + layercount) = " " & CStr(layercount) & _
        Space(9 -
        ↪ Len(Left(Round(Sheets("SoilData").Range("J"

```

```

    ↪ & CStr(count - 1).Value, 1), 5))) &
    ↪ Left(Round(Sheets("SoilData").Range("J" &
    ↪ CStr(count - 1).Value, 1), 5) & _
Space(9 -
    ↪ Len(Left(Round(Sheets("SoilData").Range("N"
    ↪ & CStr(count - 1).Value / 24, 2), 4))) &
    ↪ Left(Round(Sheets("SoilData").Range("n" &
    ↪ CStr(count - 1).Value / 24, 2), 4) & _
Space(9 - 5) & ("0.300") & _
Space(9 -
    ↪ Len(Left(Round(Sheets("SoilData").Range("Q"
    ↪ & CStr(count - 1).Value / 100, 3), 5))) &
    ↪ Left(Round(Sheets("SoilData").Range("Q" &
    ↪ CStr(count - 1).Value / 100, 3), 5) & _
Space(9 -
    ↪ Len(Left(Round(Sheets("SoilData").Range("P"
    ↪ & CStr(count - 1).Value / 100, 3), 5))) &
    ↪ Left(Round(Sheets("SoilData").Range("P" &
    ↪ CStr(count - 1).Value / 100, 3), 5) & _
Space(9 -
    ↪ Len(Left(Round(Sheets("SoilData").Range("R"
    ↪ & CStr(count - 1).Value, 2), 4))) &
    ↪ Left(Round(Sheets("SoilData").Range("R" &
    ↪ CStr(count - 1).Value, 2), 4) & _
Space(9 -
    ↪ Len(Left(Round(Sheets("SoilData").Range("S"
    ↪ & CStr(count - 1).Value, 4), 4))) &
    ↪ Left(Round(Sheets("SoilData").Range("S" &

```

↪ CStr(count - 1)).Value, 2), 4) & \_  
 Space(9 - 4) & "0.30"

'The following line writes the bottom depth of layers and

↪ the lateral conductivities (taken to be Ko or  
 ↪ matching point conductivity) to the gen file, line 21

GENArray(21) = GENArray(21) & Space(5 -

↪ Len(Left(Round(Sheets("ros\_input").Range("D" &  
 ↪ CStr(count)).Value, 2), 5))) & Left(Round(Sheets(  
 ↪ "ros\_input").Range("D" & CStr(count)).Value, 2),  
 ↪ 5) & \_

Space(5 - Len(Left(Round(Sheets(  
 ↪ "ros\_input").Range("J" &  
 ↪ CStr(count)).Value / 24, 2), 5))) &  
 ↪ Left(Round(Sheets(  
 ↪ "ros\_input").Range("J" &  
 ↪ CStr(count)).Value / 24, 2), 5)

'for K\_D NH4 we use a low, mid, and high value based

↪ upon literatrue review.

,

'If Sheets("SoilData").Range("Q" & CStr(count -

↪ 1)).Value < 5 Then

'ElseIf Sheets("SoilData").Range("Q" & CStr(count -

↪ 1)).Value >= 5 And Sheets("SoilData").Range("Q"

↪ & CStr(count - 1)).Value < 50 Then

'ElseIf Sheets("SoilData").Range("Q" & CStr(count -

```

        ↪ 1)).Value >= 5 And Sheets("SoilData").Range("Q"
        ↪ & CStr(count - 1)).Value < 50 Then
    'End If
End If

Next count

End If
str = Sheets("ros_input").Range("B" & CStr(i)).Value
If str = 1 Then
    Print #filenumber, CStr(layercount) & vbTab & "1"
    Print #filenumberpros, CStr(layercount) & vbTab & "1"
    Print #filenumberpros_tpl, CStr(layercount) & vbTab & "1"
    TAGarray(seriesnumber) = layercount

    drain_coeff = 10 'drainage coefficient set to 6cm/day by default.
        'drainage coefficient works to set an upper limit on the amount
        ↪ of drainage that
        'may occur in a day.
        'this is value of 10 is meant to be too large a drainage coefficient
        ↪ to interfere with
        'drainage rates.
        'by setting drain_coeff=10 we are effectively ignoring drainage
        ↪ coefficient.
        'a much better estimate comes from a limiting drainage rate
        ↪ being set by conductivities.

End If

```

```
'<><> End Special Case Tasks for first layer of series
```

```
'<><><>Begin Main Routine
```

```
'print lines of .ROS file (bottom depth of layer (cm), residual and saturated
```

```
    ↪ vol.moisture content cm3/cm3, van genuchten parameters n(unitless) and
```

```
    ↪ alpha(1/cm),
```

```
'Saturated hydraulic conductivity and matching point conductivity (cm/day), and
```

```
    ↪ tortosity (unitless)
```

```
Print #filenumber, (Sheets("ros_input").Range("D" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("E" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("F" & CStr(i)).Value _
```

```
& vbTab & Sheets("ros_input").Range("G" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("H" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("I" & CStr(i)).Value _
```

```
& vbTab & Sheets("ros_input").Range("J" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("K" & CStr(i)).Value & vbTab)
```

```
'print same line to a .pros file, but with conductivities in units of cm/hour
```

```
Print #filenumberpros, (Sheets("ros_input").Range("D" & CStr(i)).Value & vbTab
```

```
    ↪ & Sheets("ros_input").Range("E" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("F" & CStr(i)).Value _
```

```
& vbTab & Sheets("ros_input").Range("G" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("H" & CStr(i)).Value & vbTab &
```

```
    ↪ (Sheets("ros_input").Range("I" & CStr(i)).Value / 24) _
```

```
& vbTab & ((Sheets("ros_input").Range("J" & CStr(i)).Value) / 24) & vbTab &
    ↪ Sheets("ros_input").Range("K" & CStr(i)).Value & vbTab)
```

'print same line to a pros template file (.tpl), but with conductivities replaced with

```
    ↪ Pest conductivity markers $ KS'SERIESNUMBER''layernumber'$ and $
    ↪ Ko'seriesnumber''layernumber'$
```

```
Print #filenumberpros_tpl, (Sheets("ros_input").Range("D" & CStr(i)).Value &
```

```
    ↪ vbTab & Sheets("ros_input").Range("E" & CStr(i)).Value & vbTab &
    ↪ Sheets("ros_input").Range("F" & CStr(i)).Value _
```

```
& vbTab & Sheets("ros_input").Range("G" & CStr(i)).Value & vbTab &
```

```
    ↪ Sheets("ros_input").Range("H" & CStr(i)).Value & vbTab & "$ KS" & _
```

```
Sheets("ros_input").Range("A" & CStr(i)).Value & Sheets("ros_input").Range("B"
```

```
    ↪ & CStr(i)).Value & " $" & vbTab & "$ Ko" & _
```

```
Sheets("ros_input").Range("A" & CStr(i)).Value & Sheets("ros_input").Range("B"
```

```
    ↪ & CStr(i)).Value & " $" & vbTab & Sheets("ros_input").Range("K" &
```

```
    ↪ CStr(i)).Value & vbTab)
```

```
str = Sheets("ros_input").Range("B" & CStr(i + 1)).Value 'look at the following
```

```
    ↪ line of data
```

```
'<><><> End Main Routine
```

'<><>Begin Special Case Tasks for the final layer in a series

'if the last series in the project do the following

If IsEmpty(Sheets("ros\_input").Range("B" & CStr(i + 1))) Then 'if the next line is

↪ blank (end of data) do the following

Print #filenumber, CStr(maxrootdepth) & vbTab & CStr(avgrootdepth) &

↪ vbTab & "1"

Print #filenumberpros, CStr(maxrootdepth) & vbTab &

↪ CStr(avgrootdepth) & vbTab & "1"

Print #filenumberpros\_tpl, CStr(maxrootdepth) & vbTab &

↪ CStr(avgrootdepth) & vbTab & "1"

Close #filenumber

Close #filenumberpros

Close #filenumberpros\_tpl

TAGheader(1) = seriesnumber

TAGarray(seriesnumber) = TAGarray(seriesnumber) & "," &

↪ Sheets("soildata").Range("J" & CStr(i - 1)).Value & "," &

↪ drain\_coeff

DMNarray(16) = "&1" & Space(8 -

↪ Len(Left(Round(Sheets("soildata").Range("D" & CStr(i -

↪ 1)).Value, 1), 5))) & -

CStr(Left(Round(Sheets("soildata").Range("D" & CStr(i - 1)).Value, 1),

↪ 5)) & " 5.00"

GENarray(20) = Space(10 - Len(Left(Round(Sheets("soildata").Range("D"

↪ & CStr(i - 1)).Value, 2), 6))) & -

```
CStr(Left(Round(Sheets("soildata").Range("D" & CStr(i -
↪ 1)).Value, 2), 6))
```

Call TAG\_header.write 'write TAGheader back into TAGarray(0)

Call TAG\_write\_file(projectname) 'write the TAGarray back into the TAG  
↪ file.

Erase TAGarray

Call DMN\_write\_file(projectname, seriesnumber)

Erase DMNarray

Call PRJ\_write\_file(projectname, seriesnumber)

Erase PRJarray

Call GEN\_write\_file(projectname, seriesnumber)

Erase GENarray

End If

'if there is another series to consider after the current sereis, do the following

If str = "1" Then 'if the next line is a new soil series (layer = 1) do the following

Print #filenumber, (CStr(maxrootdepth) & vbTab & CStr(avgrootdepth) &  
↪ vbTab & "1")

Print #filenumberpros, (CStr(maxrootdepth) & vbTab &  
↪ CStr(avgrootdepth) & vbTab & "1")

Print #filenumberpros.tpl, (CStr(maxrootdepth) & vbTab &  
↪ CStr(avgrootdepth) & vbTab & "1")

Close #filenumber

Close #filenumberpros

Close #filenumberpros.tpl



```

TAGarray(seriesnumber) = TAGarray(seriesnumber) & ", " &
    ↪ Sheets("soildata").Range("J" & CStr(i - 1)).Value & ", " &
    ↪ drain_coeff

DMNArray(16) = "&1" & Space(8 -
    ↪ Len(Left(Round(Sheets("soildata").Range("J" & CStr(i -
    ↪ 1)).Value, 1), 5))) & -
        CStr(Left(Round(Sheets("soildata").Range("J" & CStr(i
        ↪ - 1)).Value, 1), 5)) & " 5.00"

GENArray(20) = Space(10 - Len(Left(Round(Sheets("soildata").Range("J"
    ↪ & CStr(i - 1)).Value, 2), 6))) & -
        CStr(Left(Round(Sheets("soildata").Range("J" & CStr(i -
        ↪ 1)).Value, 2), 6))

Call DMN_write_file(projectname, seriesnumber)
Erase DMNArray
Call PRJ_write_file(projectname, seriesnumber)
Erase PRJarray
Call GEN_write_file(projectname, seriesnumber)
Erase GENArray

End If
'<><>END Special Case Tasks for the final layer in a series

```

Next i

```
'#####
'# Run DMSOILP.EXE on each .ros file.
'#####

For count = 1 To seriesnumber
    strshell = "" & folderpath & "\program\DMSOILP.EXE" & "" & " " & "" &
        ↪ folderpath & "\soils\" & projectname & CStr(count) & ".ros" & ""
    MsgBox ("Sending this command to the cmd terminal: " & vbNewLine & strshell)
    filepath = folderpath & "\soils\" & projectname & CStr(count) & ".ros"

    retval = ShellAndWait(strshell, 0, vbNormalFocus, PromptUser)
    If retval <> 0 Then
        MsgBox ("Error (code # " & retval & ") in creating Soil File (running dmsoilp.exe)
            ↪ for series number " & CStr(count))
```

```

        End If
    Next count

MsgBox ("Process completed. Your soil files should be located in " & folderpath & "\soils")

End Sub

'|<><><><><>|'
'|<><><><><>|'
'|Begin .GEN file Routines Section. |'
'| |'
'|1) create_GEN_file: creates a .GEN text file when called |'
'|2) read_GEN_file: Reads a GEN file into the global array GENarray for editing when called |'
'|3) write_GEN_file: Writes the information in GENarray back into the specified GEN file when
    ↳ called. |'
'| |'
'| Travis A. Grohman. July 2017. Oregon State University. travis.a.grohman@gmail.com |'
'|<><><><><><
'#####
'The sub is called from Soil file creation sub. Passed to it are the number of unique soil series'
'in the project and the project name. A GEN file for each unique soil series is created.
'Generic lines (lines that are the same for every project) are written here. Lines that change
    ↳ depending upon soil,crop,and manure/fertilizer applications

```

'are left blank. These lines will be filled throughout the workflow.

```
'#####
```

```
Public Sub GEN_create_file(projectname As String, nseries As Integer)
```

```
Dim count As Integer
```

```
Dim fs, a As Object
```

```
Dim filenumber As Long
```

```
Dim filepath, str As String
```

```
filenumber = FreeFile
```

```
filepath = folderpath & "\inputs\" & projectname & CStr(nseries) & ".GEN"
```

```
Set fs = CreateObject("scripting.filesystemobject")
```

```
Set a = fs.createTextFile(filepath, True)
```

```
a.Close
```

```
Open filepath For Output As #filenumber
```

```
Print #filenumber, "*** Job Title ***" 'line1
```

```
Print #filenumber, Space(2244) 'line2
```

```
Print #filenumber, Space(1971) 'line3
```

```
Print #filenumber, "*** Printout and Input Control ***" 'line4
```

```
str = folderpath & "\outputs"
```

```
Print #filenumber, " 1 101 " & str & Space(111) 'line 5
```

```
'write blank lines 6– 36 to fill in later
```

```
count = 6
```

```
For count = 6 To 36
```

```
    Print #filename, ""
```

```
Next count
```

'lines 73–65 (wetlands and COMBO drainage weir) are not used but we

→ must include information.

```
Print #filename, "WET *** Wetlands Information ***" 'Line 37
```

```
Print #filename, " 0" 'line 38
```

```
Print #filename, " 1 2" 'line 39
```

```
Print #filename, " 30.0 1" 'line 40
```

```
Print #filename, "COM *** Combo Drainage Weir Settings ***" 'line 41
```

'lines 42–65 are equivalent so we will write them with a loop

```
count = 42
```

```
For count = 42 To 65
```

```
    Print #filename, " 0 0 0 .0"
```

```
Next count
```

```
Print #filename, "FPE *** Fixed Monthly PET ***" 'line 66
```

```
Print #filename, " 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00" 'line
```

→ 67

```
Print #filename, "MRA *** Monthly Ranking ***" 'line 68
```

```
Print #filename, " 1" 'line 69
```

```
Print #filename, "FAC *** Daily PET Factors ***" 'line 70
```

```
Print #filename, " 0" 'line 71
```

```
Print #filename, "OVR *** Contributing Area Flow ***" 'line 72
```

```
Print #filename, " 0 .000 " ' Line 73
```

```
Print #filename, "STM *** Soil Temperature ***" 'line 74
```

```

Print #filenumber, " ZA ZB TKA TKB TB TLAG TSNOW TMELT CDEG
    ↪ CICE" 'line 75

Print #filenumber, " 2.50 1.21 0.39 1.33 9.11 8.00 0.00 1.00 5.00 0.20" 'line 76

Print #filenumber, "Initial Soil Temperature" 'line77

Print #filenumber, " 2" 'line 78

Print #filenumber, " 0.00 0.00" 'line 79

Print #filenumber, "299.00 9.11" 'line 80

Print #filenumber, "Initial snow depth(m) & density(kg/m3)" 'line 81

Print #filenumber, " 0.00 100.00" 'line 82

Print #filenumber, "Freezing characteristic curve" 'line 83

Print #filenumber, " 4" 'line 84

Print #filenumber, " 0.00 0.530" 'line 85

Print #filenumber, " -1.00 0.100" 'line 86

Print #filenumber, " -3.00 0.010" 'line 87

Print #filenumber, " -35.00 0.000" 'line 88

```

```

Close #filenumber

```

```

End Sub

```

```

'.....
'Read_GEN_file reads text in a .GEN file of the form 'projectname''nseries'.GEN
'.....

```

```

Public Sub GEN_read_file(projectname As String, nseries As Integer)
Dim str As String

```

```

Dim filenumber, n As Integer

str = folderpath & "\inputs\" & projectname & CStr(nseries) & ".GEN"
filenumber = FreeFile
Open str For Input As #filenumber
n = 1
Do While Not EOF(filenumber) ' Loop until end of file.
    Line Input #filenumber, GENarray(n) ' read next line from file and add text to the
        ↪ array
    n = n + 1
Loop
Close #filenumber

End Sub

', .....
'Write_GEN_file writes all values held in GENarray back into a gen file of the form
    ↪ 'projectname'nseries'.GEN
', .....

Public Sub GEN_write_file(projectname As String, nseries As Integer)
    Dim str As String
    Dim n, filenumber As Integer
    filenumber = FreeFile

    str = folderpath & "\inputs\" & projectname & CStr(nseries) & ".GEN"
    n = 1
    Open str For Output As #filenumber

```

```
        For n = 1 To 100
            Print #filenumber, GENarray(n)
        Next n
    Close #filenumber

End Sub

'|<><><><>|'
'|END .GEN file Routines Section. |'
'|<><><><>|'
```



```

'|<><><><><><>|
'|Begin.PRJ file Routines Section. '|
'| '|
'|1) create_PRJ_file: creates a .PRJ text file when called '|
'|2) read_PRJ_file: Reads a PRJ file into the global array PRJarray for editing when called '|
'|3) write_PRJ_file: Writes the information in PRJarray back into the specified PRJ file when
    ↪ called. '|
'| '|
'| Travis A. Grohman. July 2017. Oregon State University. travis.a.grohman@gmail.com '|
'|<><><><><><>|
Public Sub PRJ_create_file(projectname As String, nseries As Integer)

Dim i, count As Integer
Dim n As Long
Dim fs, a As Object
Dim filenumber As Long
Dim filepath, str As String
filenumber = FreeFile

    filepath = folderpath & "\inputs\" & projectname & CStr(nseries) & ".PRJ"
    Set fs = CreateObject("scripting.filesystemobject")
    Set a = fs.createTextFile(filepath, True)
    a.Close
    Open filepath For Output As #filenumber
    Print #filenumber, "[Options]" 'line 1

    Print #filenumber, "Project=2" & Space(1236) 'line 2

```

```

Print #filename, "Yield=1" & Space(12827) 'line 3
Print #filename, "PET=1" & Space(12827) 'line 4
Print #filename, "CREAMS=0" & Space(12827) 'line 5
Print #filename, "Runoff=0" & Space(12827) 'line6
Print #filename, "SoilTemp=1" & Space(12827) 'line 7
Print #filename, "[General]" 'line 8

    str = "Hydrology=" & folderpath & "\inputs\" & projectname & CStr(nseries)
    ↪ & ".GEN"

    'n = 8149 - Len(str)
Print #filename, str & Space(8106) 'line 9
Print #filename, "Runoff=" & Space(12928) 'line 10
Print #filename, "FieldRatio=0" & Space(12827) 'line 11
str = "Nitrogen=" & folderpath & "\inputs\" & projectname & CStr(nseries) &
    ↪ ".DMN"

'n = 12240 - Len(str)
Print #filename, str & Space(12198) 'line 12
Print #filename, "Salinity=" & Space(12928) 'line 13
Print #filename, "[Soils]" 'line 14

    str = "SoilFile=" & folderpath & "\soils\" & projectname & CStr(nseries) &
    ↪ ".SIN"

    'n = 8481 - Len(str)
Print #filename, str & Space(8444) 'line 15

    str = "SoilWater=" & folderpath & "\soils\" & projectname & CStr(nseries) &
    ↪ ".MIS"

    'n = 5990 - Len(str)

```

Print #filenumber, str & Space(5952) 'line 16

```
str = "VolDrained=" & folderpath & "\soils\" & projectname & CStr(nseries)
    ↪ & ".WDV"
'n = 9639 - Len(str)
```

Print #filenumber, str & Space(9600) 'line 17

Print #filenumber, Space(518) 'line 18

Print #filenumber, "[Weather]" 'line 19

```
str = "Rainfall=" & folderpath & "\weather\" & projectname & "precip.RAI"
'n = 8280 - Len(str)
```

Print #filenumber, str & Space(8223) 'line 20

```
str = "Temperature=" & folderpath & "\weather\" & projectname &
    ↪ "temp.TEM"
'n = 8283 - Len(str)
```

Print #filenumber, str & Space(8223) 'line 21

```
str = "PET=" & folderpath & "\weather\" & projectname & ".PET"
```

Print #filenumber, str & Space(12928) 'line 22

Print #filenumber, Space(249) 'line 23

Print #filenumber, "[Measured]" 'line 24

Print #filenumber, "WaterTable=" & Space(12800) 'line 25

Print #filenumber, "Drainage=" & Space(12800) 'line 26

Print #filenumber, "[State]" 'line 27

Print #filenumber, "stateset=true" 'line 28

Print #filenumber, "[Analysis]" 'line 29

```
Print #filenumber, "DrainMode=1" 'line 30
```

```
'blank lines 31–40 to be filled in with the drainage settings and crop routines
```

```
n = 31
```

```
For n = 31 To 40
```

```
    Print #filenumber, ""
```

```
Next n
```

```
Close #filenumber
```

```
End Sub
```

```
'.....
```

```
'PRJ_read_file reads text in a .PRJ file of the form 'projectname''nseries'.PRJ
```

```
'and stores it in the global array "PRJarray"
```

```
'.....
```

```
Public Sub PRJ_read_file(projectname As String, nseries As Integer)
```

```
    Dim str As String
```

```
    Dim filenumber, n As Integer
```

```

str = folderpath & "\inputs\" & projectname & CStr(nseries) & ".PRJ"
filenumber = FreeFile
Open str For Input As #filenumber
n = 1
Do While Not EOF(filenumbe) ' Loop until end of file.
    Line Input #filenumber, PRJarray(n) ' read next line from file and add text to the
        ↪ array
    n = n + 1
Loop
Close #filenumber

End Sub

'.....
'PRJ_write_file writes all values held in PRJarray back into a PRJ file of the form
    ↪ 'projectname'nseries'.PRJ
'.....

Public Sub PRJ_write_file(projectname As String, nseries As Integer)
    Dim str As String
    Dim n, filenumber As Integer
    filenumber = FreeFile

    str = folderpath & "\inputs\" & projectname & CStr(nseries) & ".PRJ"
    n = 1
    Open str For Output As #filenumber
        For n = 1 To 62

```

```

        Print #filenumber, PRJarray(n)
    Next n
Close #filenumber

```

```

End Sub

```

```

'|<><><><><><>|'
'|END .PRJ file Routines Section.
'|<><><><><><>|'

```

```

Public Sub DMN_create_file(projectname As String, nseries As Integer)

```

```

    Dim n, count, filenumber As Integer
    Dim filepath, str As String
    Dim fs, a As Object

```

```

filepath = folderpath & "\inputs\" & projectname & CStr(nseries) & ".DMN"
Set fs = CreateObject("scripting.filesystemobject")
Set a = fs.createTextFile(filepath, True)
a.Close
filenumber = FreeFile
Open filepath For Output As #filenumber
Print #filenumber, "*General" 'line1
Print #filenumber, "&======" 'line2
Print #filenumber, "&Nrot Ncrop" 'line3
Print #filenumber, "" 'line4 (filled in crops routine)
Print #filenumber, "&IsNH4 IsUniform IsYield IsTemp IsResidue IsManure" 'line5
Print #filenumber, ".T .T .T .T .T .T " 'line6
Print #filenumber, "&OutFlag OutPath" 'line7
Print #filenumber, " 1 " 'line8
Print #filenumber, "&IsDRN IsNIT IsFLX IsMCT" 'line9
Print #filenumber, ".T .F .F .F" 'line10
Print #filenumber, "*Grid " 'line11
Print #filenumber, "&=====" 'line12
Print #filenumber, "&DelZ" 'line13
Print #filenumber, " 5.00" 'line14
Print #filenumber, "&G DEPgrid DZgrid" 'line15
Print #filenumber, "" 'line16 (filled in soils routine)
Print #filenumber, "*Field" 'line17
Print #filenumber, "&=====" 'line18
Print #filenumber, "&Soil files " 'line19
str = folderpath & "\soils\" & projectname & nseries & ".MIS"
Print #filenumber, str & Space(80 - Len(str)) 'line20

```

```

str = folderpath & "\soils\" & projectname & nseries & ".WDV"
Print #filenumber, str & Space(80 - Len(str)) 'line21
Print #filenumber, "&ProfDepth YesWTD YesDDZ" 'line22
Print #filenumber, "" 'line23 '(filled in part in soil routine and part in drainage
    ↪ routine)
Print #filenumber, "&L DEPsoil HydrCond WltPnt ClyFrc SltFrc Rho.b SoilpH
    ↪ K_d" 'line24

```

```

Close filenumber

```

```

End Sub

```

```

Public Sub testdmndmn()

```

```

    Call getprojectname

```

```

    Call GetFolderPath

```

```

    Call DMN_create_file("TESTOUTPUT", 3)

```

```

End Sub

```

```

', .....

```

```

'DMN_read_file reads text in a .DMN file of the form 'projectname'nseries'.DMN

```

```

'and stores it in the global array "DMNarray"

```

```

', .....

```



```

Public Sub DMN_read_file(projectname As String, nseries As Integer)
    Dim str As String
    Dim filenumber, n As Integer

    str = folderpath & "\inputs\" & projectname & CStr(nseries) & ".DMN"
    filenumber = FreeFile
    Open str For Input As #filenumber
    n = 1
    Do While Not EOF(filenumber) ' Loop until end of file.
        Line Input #filenumber, DMNArray(n) ' read next line from file and add text to the
            ↪ array
        n = n + 1
    Loop
    Close #filenumber

End Sub

', .....
'DMN_write_file writes all values held in DMNArray back into a DMN file of the form
    ↪ 'projectname'nseries'.DMN
', .....

Public Sub DMN_write_file(projectname As String, nseries As Integer)
    Dim str As String
    Dim n, filenumber As Integer
    filenumber = FreeFile

```

```
str = folderpath & "\inputs\" & projectname & CStr(nseries) & ".DMN"
n = 1
Open str For Output As #filenumber
    For n = 1 To 41
        Print #filenumber, DMNArray(n)
    Next n
Close #filenumber

End Sub
```

'TAG file contains the following information

'line 0:

'line1-n, implayerdepth

'number of entries - 1= # soil sereis in the project :(i.e.

→  $[ubound(tagarray) - lbound(Tagarray) + 1] - 1] = \text{Seriesnumner}$

```
Public Sub TAG_create_file(projectname As String)
```

```
Dim filepath, str As String
```

```
Dim fs, a As Object
```

```
Dim filenumber As Long
```

```
filenumber = FreeFile
```

```
filepath = folderpath & "\inputs\" & projectname & ".TAG"
```

```
Set fs = CreateObject("scripting.filesystemobject")
```

```
Set a = fs.createTextFile(filepath, True)
```

```
a.Close
```

```
Open filepath For Output As #filenumber
```

'create first line of TAGfile. The first line of the TAG file contains

'the follwing variables delimited by commas:

'1)Project Name,2)Number of soil series,3)start month name, 4)start day-of-month

→ ,5) startyear,

'6) end month name, 7)end day-of-month, 8) end year



```

        ↪ array
        i = i + 1
    Loop
Close #filenumber
Exit Sub
errmsg: MsgBox "Could not read TAG file."
End Sub

', .....
'Write_TAG_file writes all values held in TAGarray back into a gen file of the form
    ↪ 'projectname'.TAG
', .....

Public Sub TAG_write_file(projectname As String)
    Dim str As String
    Dim n, filenumber As Integer
    filenumber = FreeFile

    str = folderpath & "\inputs\" & projectname & ".TAG"

    Open str For Output As #filenumber
        For n = 0 To seriesnumber
            Print #filenumber, TAGarray(n)
        Next
    Close #filenumber

End Sub

```

```
Public Sub testweather()
```

```
Dim startdate As Date
```

```
Dim startmonth As String
```

```
Dim arrone(1 To 5) As String
```

```
Dim arrtwo() As String
```

```

arrone(1) = "one,two,three,four,five"
arrone(2) = "arrone_two"
arrone(3) = "arrone_three"

arrtwo = Split(arrone(1), ",")
MsgBox arrtwo(0) & " " & arrtwo(1) & " " & arrtwo(2) & " " & arrtwo(3) & " " & arrtwo(4)

'arrone(1) = projectname & ", , , , , , "
'arrtwo = Split(arrone(1), ",")

'If arrtwo(7) = " " Then MsgBox "okie doke"
'arrtwo(7) = "."
'MsgBox (arrtwo(7))
'startdate = Sheets("Weather_Input").Range("D4")
'startmonth = MonthName(month(startdate), False)
'MsgBox startdate & " " & startmonth
'simstart_day = day(Sheets("Weather_Input").Range("D4"))
'MsgBox simstart_day
'simstart_year = Year(Sheets("Weather_Input").Range("D4"))
'MsgBox simstart_year
End Sub

```

```
'#####
'#Author Travis Grohman. July 2017. Oregon State Univeristy.
'# This uses the information contained in the sheet "Weather_Inputs" to create properly
    ↳ formatted precipitation, temperature, and (optionally) Potential Evapotranspiration files.
'# The program DMWEA.EXE is then run in the terminal for each .ros file. DMSOILP.EXE
    ↳ creates all soil files needed for Drainmod.
'# DMWEA.EXE is the weather utility included with Drainmod version six and was written by
    ↳ Dr. Skaggs ( Skaggs 1979, Skaggs et al 2012)
'#DMWEA.EXE creates the .TEM, .RAI, and .PET files used as inputs in the drainmod model.
    ↳ DMWEA.EXE converts daily precipitation into hourly precipitation in a .RAI file
    ↳ (distributes precipitation over 4 hours).
'#####
```

```
Public Sub Weather_Write()
```

```
'On Error GoTo errormsg
```

```
Dim i, count, precipfilenumber, tempfilenumber As Long 'count is generic counter.
```

```
    ↳ filenumbers for defining a file to print to.
```

```
Dim doub As Double 'generic double
```

```
Dim fs, a, b As Object 'objects create precip and temp files
```

```
Dim ITEM As Variant 'item is the control in the "FOR each in "array looping technique.
```

```
    ↳ (e.g. for each item in ARRAY...next)
```



doub = Weather\_Error.Check 'Checks there is data properly entered on the sheet

→ Weather\_Input, if not returns -99.9

If doub = -99.9 Then Exit Sub

Call GetFolderPath 'get the path to the main folder on this computer

Call getprojectname 'if a project name has not be assigned yet, ask the user to define one

→ now

Dim str As String 'generic string

Dim precip\_filepath, temp\_filepath As String 'strings of the filepath to the precip and

→ temperature output files

Dim arr() As String 'generic array

Call turnoff\_AUTOS 'turn off automathic worksheet functionality options that slow down

→ VBA script

Dim retval As Integer

precip\_filepath = folderpath & "\weather\" & projectname & "precip.txt"

temp\_filepath = folderpath & "\weather\" & projectname & "temp.txt"

Set fs = CreateObject("scripting.filesystemobject")

Set a = fs.createTextFile(precip\_filepath, True)

a.Close

precipfilenumber = FreeFile 'make the print function's number alias of precipitation

→ file the next available (unused)file number

Open precip\_filepath For Output As #CStr(precipfilenumber)

Set fs = CreateObject("scripting.filesystemobject")

```

Set b = fs.createTextFile(temp_filepath, True)
b.Close

tempfilenumber = FreeFile 'make the print function's number alias of pricip file the
    ↪ next available (unused)file number

Open temp_filepath For Output As #CStr(tempfilenumber)

,~~~~~

'<><> Start first line of data special tasks <><>'
'<><>Actual data starts at row 4 (import starts at D3, first row of import is a header
    ↪ row)<><>.

'<><>Before begining loop , do the following special tasks<><>'

count = 4 'initialize count at 4.

simstart_month = monthname(month(Sheets("Weather_Input").Range("D4")), False)
simstart_day = day(Sheets("Weather_Input").Range("D4"))
simstart_year = year(Sheets("Weather_Input").Range("D4"))

Call TAG_read_file(projectname) 'Put TAG file lines into TAGarray

Call TAG_header_read 'split TAGarray(0) by commas and store them in the dynamic array
    ↪ TAGheader

TAGheader(2) = simstart_month 'TAGheader(3) is the starting month of the simulation
    ↪ defined by date of first weather datapoint

TAGheader(3) = simstart_day 'TAGheader(4) is the starting day-of-month (e.g. 1-3) of
    ↪ the simulation " " " " " "

TAGheader(4) = simstart_year 'TAGheadder(5) is the starting year of simulation " " " " " "

```

Call TAG\_header\_write *'write TAGheader back into TAGarray(0)*

Call TAG\_write\_file(projectname) *'write the TAGarray back into the TAG file.*

count = 4 *'initiallize count at 4.(Actual data starts at row 4 on the Weather\_input sheet)*

*'<><> End first line of data special tasks <><>'*

*,^^^^^^^^^^^^^^^^^^,*

Do Until IsEmpty(Sheets("Weather\_Input").Range("D" & CStr(count))) *'loop until no data*

*↪ in column D (date)*

If count = 4 Then Sheets("Weather\_Input").Range("B3") = "Day of year (1-365)"

*↪ 'give the B column a header before it is populated*

*'extract day of year from date in column C and put in column B*

Sheets("Weather\_Input").Range("B" & CStr(count)) = "=D" & CStr(count) &

*↪ "-DATE(YEAR(D" & CStr(count) & "),1,0)"*

*'set to return only the values (not the formulas in the cells)*

Sheets("Weather\_Input").Range("B" & CStr(count)) =

*↪ Sheets("Weather\_Input").Range("B" & CStr(count)).Value*

```
str = year(Sheets("Weather_Input").Range("D" & CStr(count)).Value) & vbTab &
    ↳ Sheets("Weather_Input").Range("B" & CStr(count)).Value & vbTab &
    ↳ Sheets("Weather_Input").Range("E" & CStr(count)).Value
'string with year (column A), day of year (column B), and daily precipitation
    ↳ (column E) seperated by tab
Print #CStr(precipfilenumber), str 'print the above line to the precipitation file
```

```
str = year(Sheets("Weather_Input").Range("D" & CStr(count)).Value) & vbTab &
    ↳ Sheets("Weather_Input").Range("B" & CStr(count)).Value & vbTab & _
    Sheets("Weather_Input").Range("G" & CStr(count)).Value & vbTab &
    ↳ Sheets("Weather_Input").Range("F" & CStr(count)).Value
'^ after writing to precip file, redefine the generic string to be a line of the temp
    ↳ file. This includes:
'Year (column A), Day of year (Column B), Max temp (column G), and min temp
    ↳ (Column F) in that order
```

```
Print #tempfilenumber, str 'print the above line to the temp file
```

```
count = count + 1
```

```
Loop
```

```
Close #tempfilenumber 'close the files when done with them
```

```
Close #precipfilenumber
```

```
simend_month = monthname(month(Sheets("Weather_Input").Range("D" & CStr(count -
```

→ 1))), False)

simend\_day = day(Sheets("Weather\_Input").Range("D" & CStr(count - 1)))

simend\_year = year(Sheets("Weather\_Input").Range("D" & CStr(count - 1)))

Call TAG\_read\_file(projectname) *'Put TAG file lines into TAGarray*

Call TAG\_header\_read *'split the first element in TAGarray ( TAGarray(0) ) by comma*

→ *delimiters*

*'and put split elements in the dynamic array*

→ *TAGheader*

TAGheader(5) = simend\_month *'TAGheader(6) is the ending month of the simulation*

→ *defined by date of first weather datapoint*

TAGheader(6) = simend\_day *'TAGheader(7) is the ending day-of-month (e.g. 1-3) of the*

→ *simulation " " " " " "*

TAGheader(7) = simend\_year *'TAGheader(8) is the ending year of simulation " " " " " "*

Call TAG\_header\_write *'write the TAGheader elements back into TAGarray(0)*

Call TAG\_write\_file(projectname) *'write the TAGarray back into the TAG file.*

'#####

'#run DMWEA.EXE. get values from optionbuttons for which units to pass to DMWEA.EXE

'Proper use (taken directly from dmwea.exe help message)

""USAGE: DMWEEA <INFILE> <TYPE> <UNITS> <STA> <NHR> <BEGHR>

' where <INFILE> is the input filename

' <TYPE> is the input data type (TP=temperature,ET=potential ET, HR=hourly rain, DR =

→ *Daily rain*)

' <UNITS>: DF=degree far, DC =degree celcius, IN =inches, CM = centimeters, MM=

→ *millimeters*

```

' <STA> = station number (I am using the same station number "111111" for all in this
    ↪ program)
' <NHR> = hours to distribute daily rain (I am using 5 hours)
' <BEGHR> = begining hour (I am using 16)

'#####

'#####
'First create .RAI precipitation file.
'#####

str = vbNullString 'erase the generic string
'daily or hourly from option button on weather input
If Sheets("Weather_Input").Shapes("option_dailyprecip").ControlFormat.Value = 1 Then str =
    ↪ " DR "
If Sheets("Weather_Input").Shapes("option_hourlyprecip").ControlFormat.Value = 1 Then str =
    ↪ " HR "
'get units for precipitation from "Weather_Input" option buttons

If Sheets("Weather_Input").Shapes("option_inches").ControlFormat.Value = 1 Then str = str &
    ↪ "IN 111111 5 16"
If Sheets("Weather_Input").Shapes("option_mm").ControlFormat.Value = 1 Then str = str &
    ↪ "MM 111111 5 16"
If Sheets("Weather_Input").Shapes("option_Cm").ControlFormat.Value = 1 Then str = str &

```

```

    ➔ "CM 111111 5 16"

'run DMWEA.EXE in the terminal, wait for resut.
str = "" & folderpath & "\program\DMWEA.EXE" & "" & " " & "" & precip_filepath
    ➔ & "" & str

'Send it to the terminal with the shellandwait function
retval = ShellAndWait(str, 0, vbNormalFocus, PromptUser)

'#####
'Now create .TEM temperature file.
'#####

str = vbNullString 'erase the generic string

'get units for temp from "Weather_Input" option buttons

If Sheets("Weather_Input").Shapes("option_f").ControlFormat.Value = 1 Then str = str & " TP
    ➔ DF 111111"
If Sheets("Weather_Input").Shapes("option_c").ControlFormat.Value = 1 Then str = str & "
    ➔ TP DC 111111"

'run DMWEA.EXE in the terminal, wait for resut.
str = "" & folderpath & "\program\DMWEA.EXE" & "" & " " & "" & temp_filepath &
    ➔ "" & str

```

```

'Send it to the terminal with the shellandwait function
retval = ShellAndWait(str, 0, vbNormalFocus, PromptUser)

'#####
'now to delete the txt intermediary files:

'delete temp intermediary

str = temp_filepath

    If Dir(str) <> "" Then 'check if exists (it should).
        SetAttr str, vbNormal 'remove readonly attribute if it is there (shouldn't be)
        Kill str ' delete the file
    End If
'delete precip intermediary
str = precip_filepath

    If Dir(str) <> "" Then 'check if exists (should)
        SetAttr str, vbNormal 'remove readonly attribute if it is there (shouldn't be)
        Kill str ' Then delete the file
    End If

,~~~~~

'if latitude/longitude wasnt entered, get it now (call error_or_manual which 'loads

```



```

    ↪ FORM_LATLONGERROR)><><><><><><><><><><>
'<><>otherwise, set the global lat/long values to the values in the sheet Weather_input.
If IsNull(Sheets("weather_input").Range("latitude")) Then
    Call error_or_manual(False)
ElseIf IsNull(Sheets("weather_input").Range("longitude")) Then
    Call error_or_manual(False)
Else
    LAT_deg = Int(Sheets("weather_input").Range("latitude"))
    LAT_min = Minute((Sheets("weather_input").Range("latitude") - LAT_deg) / 24)
    LONG_deg = Int(Sheets("weather_input").Range("longitude"))
    LONG_min = Minute((Sheets("weather_input").Range("longitude") - LONG_deg) / 24)
End If

,~~~~~;

'<><>Begin Writing GEN lines related to weather<><><><>

count = 1
For count = 1 To seriesnumber
    Call GEN_read_file(projectname, CInt(count))

GENarray(6) = "*** Climate *** "

str = "111111 " & folderpath & "\weather\" & projectname & "precip.RAI"

```

```
GENarray(7) = str & Space(135 - Len(str))
```

```
str = "111111 " & folderpath & "\weather\" & projectname & "temp.TEM"
```

```
GENarray(8) = str & Space(135 - Len(str))
```

```
GENarray(9) = simstart_year & Space(3 - Len(Get_month_number(simstart_month))) &
```

```
    ↪ Get_month_number(simstart_month) & " " & simend_year & Space(3 -
```

```
    ↪ Len(Get_month_number(simend_month))) & _
```

```
    Get_month_number(simend_month) & Space(3 - Len(Left(CStr(LAT_deg), 3))) &
```

```
    ↪ Left(CStr(LAT_deg), 3) & Space(2 - Len(Left(CStr(LAT_min), 2))) &
```

```
    ↪ Left(CStr(LAT_min), 2) & " 0 51 0"
```

```
GENarray(10) = " 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00"
```

```
Call GEN_write_file(projectname, CInt(count))
```

Next count

Call turnon\_AUTOS 'turn worksheet functionality options back on

Exit Sub

errmsg: MsgBox "Failure:Unforseen errors creating weather files ."

End Sub

‘ $\langle \rangle \langle \rangle \langle \rangle \langle \rangle$ ’,

➤ This function checks for data in the first row holding weather data (row 4) in the  
➤ worksheet "Weather Input".

<><><> It is used in the Weather\_Write subroutine; called before writing temperature and  
 ↪ rainfall files.

'<><><> If any necessary data is missing, it returns "-99.9". otherwise it returns 0

'<><><> Created on :July 28th 2017 By:Travis A. Grohman, Oregon State University, Water  
 ↪ Resources Engineering.

,  $\langle \rangle \langle \rangle \langle \rangle \langle \rangle \langle \rangle \langle \rangle$

Public Function Weather\_Error\_Check()

If IsEmpty(Sheets("Weather\_Input").Range("D4")) Then

MsgBox "Error: No starting date on the worksheet '*Weather\_Input*'. Please make sure you

↪ have imported weather data into the worksheet " & \_

",that all weather data starts on row 4, and that the dates are in Column D."

Weather\_Error\_Check = -99.9

Exit Function

ElseIf IsEmpty(Sheets("Weather\_Input").Range("E4")) Then

MsgBox "Error: Could not find precipitation values on the worksheet 'Weather\_Input'.

↪ Please make sure you have imported weather data into the worksheet " & \_

",that all weather data starts on row 4, and that the precipitation values are in Column E."

Weather\_Error\_Check = -99.9

Exit Function

ElseIf IsEmpty(Sheets("Weather\_Input").Range("F4")) Then

MsgBox "Error: Could not find minimum daily temperature values on the worksheet

↪ 'Weather\_Input'. Please make sure you have imported weather data into the

↪ worksheet " & \_

",that all weather data starts on row 4, and that the minimum temperature values are in

↪ Column F."

Weather\_Error\_Check = -99.9

Exit Function

ElseIf IsEmpty(Sheets("Weather\_Input").Range("G4")) Then

MsgBox "Error: Could not find maximum daily temperature values on the worksheet

↪ 'Weather\_Input'. Please make sure you have imported weather data into the

↪ worksheet " & \_

",that all weather data starts on row 4, and that the maximum daily temperature values

↪ are in Column G."

Weather\_Error\_Check = -99.9

Exit Function

End If

Weather\_Error.Check = 0

End Function

Public Sub PRJ\_WRITE()

*'#####*

*'Begin writing .prj files*

*'#*

*'#*

*'#####*

MsgBox ("Begin writing .prj files")

For count = 1 To seriesnumber

    i = FreeFile

    Dim strProjFile As String

    Dim strGen As String

    Dim areasym As String

```

Dim intCropFlag As Integer
Dim intWTMflag As Integer
Dim b As Object
areasym = Sheets("SoilData").Range("areasym").Value

strProjFile = folderpath & "\inputs\" & projectname & CStr(count) & ".prj"
MsgBox (strProjFile)
    Set fs = CreateObject("scripting.filesystemobject")
    Set b = fs.createTextFile(strProjFile, True)
    b.Close
Open strProjFile For Output As #CStr(i)

Print #CStr(i), "[Options]"
Print #CStr(i), "Project=2"

Print #CStr(i), "Yield=1"
Print #CStr(i), "PET=2"
Print #CStr(i), "CREAMS=0"
Print #CStr(i), "Runoff=0"
Print #CStr(i), "SoilTemp=1"
Print #CStr(i), "[General]"
Print #CStr(i), "Hydrology=" & folderpath & "\inputs\" & projectname &
    ↪ CStr(count) & ".GEN"
Print #CStr(i), "Runoff="
Print #CStr(i), "FieldRatio=0"
Print #CStr(i), "Nitrogen=" & folderpath & "\inputs\" & projectname & CStr(count)
    ↪ & ".DMN"

```

```

Print #CStr(i), "Salinity="
Print #CStr(i), "[Soils]"
Print #CStr(i), "SoilFile=" & folderpath & "\soils\" & projectname & CStr(count) &
    ↪ ".SIN"
Print #CStr(i), "SoilWater=" & folderpath & "\soils\" & projectname & CStr(count)
    ↪ & ".MIS"
Print #CStr(i), "VolDrained=" & folderpath & "\soils\" & projectname & CStr(count)
    ↪ & ".WDV"
Print #CStr(i), "[Weather]"
Print #CStr(i), "Rainfall=" & folderpath & "\weather\" & areasym & ".RAI"
Print #CStr(i), "Temperature=" & folderpath & "\weather\" & areasym & ".TEM"
Print #CStr(i), "PET="; folderpath & "\weather\" & areasym & ".PET"
Print #CStr(i), "[Measured]"
Print #CStr(i), "WaterTable="
Print #CStr(i), "Drainage="
Print #CStr(i), "[State]"
Print #CStr(i), "stateset=true"
Print #CStr(i), "[Analysis]"
Print #CStr(i), "Drainmode=1"

'#####
'get these parameters from "Soils" Sheet
'#####

Dim maxsurfacestorage As String
maxsurfacestorage = Sheets("Soils").Range("maxsurfacestorage")

Dim kirkdepthtoflowfordrains As String
kirkdepthtoflowfordrains = Sheets("Soils").Range("kirkdepth")

```

```

Print #CStr(i), "DStmax=" & maxsurfacestorage
Print #CStr(i), "DStorro=" & kirkdepthtoflowfordrains
Print #CStr(i), "DrainDepth=" & Sheets("Soils").Range("E11") & ", " &
    ↪ Sheets("Soils").Range("E12") & ", " & Sheets("Soils").Range("E13")

'#####
'these parameters are based upon crop flags on the sheet "New Flow File". These need
    ↪ to be changed
'#####
Print #CStr(i), "[Crops]"

    If Sheets("New Flow File").Range("intCropFlag") = 1 Then
        Print #CStr(i), "NumCrop=1"
        If Sheets("New Flow File").Range("intWTMflag").Value = 0 Then
            Print #CStr(i), "Crop1="; folderpath & "\crops\corn.cin"; ""
        Else
            Print #CStr(i), "Crop1="; folderpath & "\crops\cornWT.cin"; ""
        End If
    End If

    If Sheets("New Flow File").Range("intCropFlag").Value = 3 Then
        Print #CStr(i), "NumCrop=2"
        If Sheets("New Flow File").Range("intWTMflag").Value = 0 Then
            Print #CStr(i), "Crop1="; folderpath & "\crops\corn.cin"; ""
            Print #CStr(i), "Crop2="; folderpath & "\crops\soybeans.cin"; ""
        Else

```



```
        Print #CStr(i), "Crop1="; folderpath & "\crops\cornWT.cin"; ""
        Print #CStr(i), "Crop2="; folderpath & "\crops\soybeansWT.cin"; ""
    End If
End If

Close #CStr(i)
Next count

'#####
'end writing .prj file
'#####

End Sub
```

```
Public Sub RUN_dmhydro()
```

```
'#####
```

```
'#Run Drainmod dmhydro.exe
```

```
'#
```

```
'#####
```

```
For count = 1 To seriesnumber
```

```
retval = ShellAndWait(""" & folderpath & "\program\" & "dmhydro.exe" & """" & " " _  
& """" & folderpath & "\inputs\" & projectname & CStr(count) & ".prj" & """" , 0,  
    ↪ vbNormalFocus, PromptUser)
```

```
DoEvents
```

```
    If retval <> 0 Then
```

```
        MsgBox ("There was an error running dmhydro.exe for Soil Series number " &
```

```

        ↪ CStr(count) & ". Check soils data and review the dmhydro log found at" &
        ↪ folderpath & "\inputs\dmhydro.log")
    End If
Next count

End Sub

'#####
'Travis A. Grohman, July 2017.
'This function returns the day of the year
' (1–365, ignores leap years) given a month and a day.
'#####

Public Function dayofyear(month As String, day As Integer)

Dim i As Integer

If month = "January" Then i = 0
If month = "February" Then i = 31
If month = "March" Then i = 31 + 28
If month = "April" Then i = 31 + 28 + 31

```

```

If month = "May" Then i = 31 + 28 + 31 + 30
If month = "June" Then i = 31 + 28 + 31 + 30 + 31
If month = "July" Then i = 31 + 28 + 31 + 30 + 31 + 30
If month = "August" Then i = 31 + 28 + 31 + 30 + 31 + 30 + 31
If month = "September" Then i = 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31
If month = "October" Then i = 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30
If month = "November" Then i = 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31
If month = "December" Then i = 31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 30
dayofyear = i + day

End Function

Public Function DOY_to_dayofmonth(dayofyear As Integer)
If dayofyear <= 31 Then
DOY_to_dayofmonth = dayofyear
Exit Function

ElseIf dayofyear <= (31 + 28) Then
DOY_to_dayofmonth = dayofyear - 31
Exit Function

ElseIf dayofyear <= (31 + 28 + 31) Then
DOY_to_dayofmonth = dayofyear - 31 - 28
Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30) Then
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31
Exit Function

```

```
ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31) Then
```

```
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31 - 30
```

```
Exit Function
```

```
ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30) Then
```

```
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31 - 30 - 31
```

```
Exit Function
```

```
ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31) Then
```

```
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31 - 30 - 31 - 30
```

```
Exit Function
```

```
ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31) Then
```

```
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31 - 30 - 31 - 30 - 31
```

```
Exit Function
```

```
ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30) Then
```

```
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31
```

```
Exit Function
```

```
ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31) Then
```

```
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31 - 30
```

```
Exit Function
```

```
ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 30) Then
```

```
DOY_to_dayofmonth = dayofyear - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31 - 30 - 31
```

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 30 + 31) Then

DOY\_to\_dayofmonth = dayofyear - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31 - 30 - 31 - 30

Exit Function

End If

End Function

Public Function DOY\_to\_month(dayofyear As Integer)

If dayofyear <= 31 Then

DOY\_to\_month = 1

Exit Function

ElseIf dayofyear <= (31 + 28) Then

DOY\_to\_month = 2

Exit Function

ElseIf dayofyear <= (31 + 28 + 31) Then

DOY\_to\_month = 3

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30) Then

DOY\_to\_month = 4

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31) Then

DOY\_to\_month = 5

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30) Then

DOY\_to\_month = 6

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31) Then

DOY\_to\_month = 7

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31) Then

DOY\_to\_month = 8

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30) Then

DOY\_to\_month = 9

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31) Then

DOY\_to\_month = 10

Exit Function

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 30) Then

DOY\_to\_month = 11

Exit Function

```

ElseIf dayofyear <= (31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 30 + 31) Then
DOY_to_month = 12
Exit Function
End If
End Function

Public Sub yieldguess()
Sheets("Start").Range("A20") = "days"
Sheets("Start").Range("B20") = "yield kg/hectare"
Sheets("start").Range("A21") = (365 - dayofyear("September", 10)) + dayofyear("May", 1)
Sheets("start").Range("B21") = 3600
Sheets("start").Range("A22") = dayofyear("June", 10) - dayofyear("May", 2)
Sheets("start").Range("B22") = 2500
Sheets("start").Range("A23") = dayofyear("July", 25) - dayofyear("June", 11)
Sheets("start").Range("b23") = 2500
Sheets("start").Range("A24") = dayofyear("September", 9) - dayofyear("July", 26)
Sheets("start").Range("b24") = 1400

End Sub

Public Sub CIN_write()
'On Error GoTo errormsg 'set errorhandle
Call getprojectname
Dim ncuts, i, n, count As Integer 'number of cuts the user specified, generic counter
Dim day As Integer
Dim potyield() As String 'Holds the potential yield for each crop in kg/ha

```



```

Dim filenumber As Long
Dim dayofcut, cut_dayofyear, dayofnextcut, nextcut_dayofyear As Integer
Dim nspace1, nspace2, nspace3, nspace4 As Integer 'hold the number of spaces needed for fixed
    ↪ width at various points
Dim monthofcut, monthofnextcut As String
Dim arr() As String
Dim nlayers, DOY, work_period_startmonth, work_period_startday, work_period_endmonth As
    ↪ Integer
Dim work_period_endday As Integer
Dim filepath, str As String 'path to the CIN files being created, generic string
Dim fs, a As Object
Dim plantdate, harvestdate, growingdays, windowbegin, windowend As Double

If FORM_crops.OPTION_pasture.Value = True Then
    GoTo pasture
ElseIf FORM_crops.OPTION_grassgenericcrop Then
    GoTo generic
ElseIf FORM_crops.OPTION_cotton = True Then
    GoTo cotton
ElseIf FORM_crops.OPTION_wheat = True Then
    GoTo wheat
ElseIf FORM_crops.OPTION_corn.Value = True Then
    GoTo corn
ElseIf FORM_crops.OPTION_soybean.Value = True Then
    GoTo soybean
ElseIf FORM_crops.Option_cornsoybean.Value = True Then

```

```

    GoTo cornsoybean
Else
MsgBox ("Error: Could not determine vegetation type. Please be sure to select a crop option and
    ↪ try again.")
Exit Sub
End If

'|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
'|| Begin case: Pasture Crop Selection
'|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||

pasture: MsgBox ("Creating Pasture crops")
ncuts = CInt(FORM_crops.COMBO_ncuts.Value)
If FORM_crops.CHECK_legumes = False Then
ncrops = ncuts 'the global number of crop yeild files is set to the number of cuts in the case of
    ↪ pasture

ReDim potyield(1 To ncrops)

'msgbox "Warning! yield calculation is under construction. Currently yeild is set as a constant
    ↪ (3500 kg/ha)
For i = 1 To ncrops
potyield(i) = 3500
Next i

ElseIf FORM_crops.CHECK_legumes = True Then
ncrops = ncuts + 1 'if there are legumes mixed in the pasture, a single crop file representing the
    ↪ legumes is added
ReDim potyield(ncrops)

```

```

Call CIN_365_Legume_Write(CInt(ncrops))

End If

Call GetFolderPath 'get folderpath and project name if not defined in memory already
Call getprojectname

For i = 1 To ncuts

filepath = folderpath & "\crops\" & projectname & CStr(i) & ".CIN"
Set fs = CreateObject("scripting.filesystemobject")
Set a = fs.createTextFile(filepath, True) 'create the text file *projectname*_*i*.CIN
a.Close 'close it because we want to print not write lines (dont want commas or " inserted)
filenumber = FreeFile 'make the print function's number alias of precipitation file the next
    ↪ avaliable (unused)file number

,,,,,,,,,,,,,,,,,,,,,
''Begin Writing .CIN files.
,,,,,,,,,,,,,,,,,,,,,

'A seperate CIN file ("crop") is created for each period between Cuts.

'## Note that Drainmod is a Fortran 77 program that requires fixed with input text files--
    ↪ mind spacing

Open filepath For Output As #filenumber

Print #filenumber, "*** First Possible and last possible dates for crop *** "
Print #filenumber, " 1 365"

```

*'assuming no controlled drainage*

Print #filenumber, "\*\*\* Weir Control \*\*\* "

Print #filenumber, " 1"

Print #filenumber, " 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0"

dayofcut = FORM\_crops.Controls("Box\_day\_cut" & CStr(i)).Value *'day of this cut (day of*  
 ➔ *month 1–31)*

monthofcut = FORM\_crops.Controls("combo\_month\_cut" & CStr(i)).Value *'month of this cut*  
 ➔ *(string "January" – "December")*

cut\_dayofyear = dayofyear(CStr(monthofcut), CInt(dayofcut)) *'reset the dayofcut variable to be*  
 ➔ *in day of year format by passing day/month to the function 'DAYOFYEAR'*

If i = ncuts Then *'if this is the final cut of the year, cut1 is set as the next cut*

monthofnextcut = FORM\_crops.Controls("COMBO\_month\_cut1").Value

dayofnextcut = FORM\_crops.Controls("Box\_day\_cut1").Value

nextcut\_dayofyear = dayofyear(CStr(monthofnextcut), CInt(dayofnextcut))

➔ *'nextcut\_dayofyear is the dayofnextcut variable in day of year format.*

growingdays = nextcut\_dayofyear + (365 – cut\_dayofyear)

Else *'if not the final cut, next cut is the cut i+1*

monthofnextcut = FORM\_crops.Controls("combo\_month\_cut" & CStr(i + 1)).Value

dayofnextcut = FORM\_crops.Controls("Box\_day\_cut" & CStr(i + 1)).Value

nextcut\_dayofyear = dayofyear(CStr(monthofnextcut), CInt(dayofnextcut))

→ *'nextcut\_dayofyear is the dayofnextcut variable in day of year format.*

growingdays = nextcut\_dayofyear – cut\_dayofyear

End If

monthofcut = Get\_month\_number(CStr(monthofcut))

monthofnextcut = Get\_month\_number(CStr(monthofnextcut)) *'convert from names to numbers*

→ *(e.g. april → 4, sept → 9, etc.)*

nspace1 = 2 – Len(monthofcut)

nspace2 = 2 – Len(dayofcut)

nspace3 = 2 – Len(monthofnextcut)

nspace4 = 2 – Len(dayofnextcut)

Print #filenumber, "\*\*\* Trafficability \*\*\* "

Print #filenumber, Space(nspace1) & monthofcut & Space(nspace2) & dayofcut &

→ Space(nspace3) & monthofnextcut & Space(nspace4) & dayofnextcut & " 720 0.01 9.99

→ 0.00"

Print #filenumber, Space(nspace1) & monthofcut & Space(nspace2) & dayofcut &

→ Space(nspace3) & monthofnextcut & Space(nspace4) & dayofnextcut & " 720 0.01 9.99

→ 0.00"

*'working periods 1 & 2 are equivilant. They are set as the lenth between cuts. The delay is set to*

→ *0 days, so there will be no delaying of work for a wet field simulated here.*

Print #filenumber, "\*\*\* Crop \*\*\* "

Print #filenumber, " 1 11231 30.00"

Print #filenumber, " 1 11231"

*'^ not defining a wet and dry period explicitly*

*'^30cm is the depth at which the pasture roots begin to feel wet stress*

*' This means if the water table is 30 cm below the surface or higher, the crops will start to react*

*↪ negativley.*

Print #filenumber, "\*\*\* Root Depths \*\*\* "

Print #filenumber, "12"

Print #filenumber, " 1 1 40.00 4 1 40.00 5 2 40.00 6 1 40.00 616 40.00 630 40.00 710 40.00 726

*↪ 40.00"*

Print #filenumber, " 822 40.0010 6 40.0010 7 40.001231 40.00"

*'constant avg root depth of 40cm throughout the year.*

Print #filenumber, "\*\*\* Yield Inputs \*\*\* "

*'The next values printed to the .CIN files are the beggining and ending dates of the crop.*

*'Since there is no planting date, the beggining and ending days of each 'crop' is set ad the day of*

*↪ a cut/harvest*

*'This following routine gets the date (Day of year 1–365 format) for this cut and the next cut.*

Print #filenumber, " 1" *'Crop window beggining day is one (dosen't matter because there is no*

*↪ 'planting' delay*

*'The Drainmod Fortran 77 code alots 3 spaces for the starting/ending (for example " 3" " 30"*



*'Begin Writing Portion of .TAG file that has crop information.*

,,,,,,,,,,,,,,,,,,,,,

*'number of crops stored in the 13th object of the TAGfile header line (line one)*

for element 12 of the array TAGheader (array starts at 0)

Call TAG\_read\_file(projectname)

Call TAG\_header\_read

TAGheader(12) = ncrops

Call TAG\_header\_write

Call TAG\_write\_file(projectname)

Erase TAGarray

,,,,,,,,,,,,,,

*'END Writing Portion of .TAG file that has crop information.*

,,,,,,,,,,,,,

```
For n = 1 To seriesnumber
```

,,,,,,,,,,,,,

*'Begin Writing Portion of .DMN file that has crop information.*

,,,,,,,,,,,,,,,,,,,,,

Call TAG\_read\_file(projectname)

Call TAG\_header\_read



```
arr = Split(TAGheader(n), ",")
```

```
nlayers = CInt(arr(0))
```

```
Erase arr
```

```
Erase TAGarray
```

```
Erase TAGheader
```

```
Call DMN_read_file(projectname, CInt(n))
```

```
DMNArray(4) = " 1" & Space(6 - Len(ncuts)) & ncuts '<> write number of 'crops' To
```

→ *DMN file (each period from cut to cut is considered a seperate 'crop')*

```
DMNArray(nlayers + 25) = "*Crops"
```

```
DMNArray(nlayers + 26) = "&====="
```

```
DMNArray(nlayers + 27) = "&C IC Leg Pday Lgrw PotYld HI RSR CrpN ShtN RotN Tl
```

→ *Fr Rt Up Rs Mn RtCf'*

```
For count = 1 To ncuts
```

```
dayofcut = FORM_crops.Controls("Box_day_cut" & CStr(count)).Value 'day of this
```

→ *cut (day of month 1-31)*

```
monthofcut = FORM_crops.Controls("combo_month_cut" & CStr(count)).Value
```

→ *'month of this cut (string "January" - "December")*

```
cut_dayofyear = dayofyear(CStr(monthofcut), CInt(dayofcut)) 'reset the dayofcut
```

→ *variable to be in day of year format by passing day/month to the function*

→ *'DAYOFYEAR'*

```
If count = ncuts Then 'if this is the final cut of the year, cut1 is set as the next
```

→ *cut*

```

monthofnextcut = FORM_crops.Controls("COMBO_month_cut1").Value
dayofnextcut = FORM_crops.Controls("Box_day_cut1").Value
nextcut_dayofyear = dayofyear(CStr(monthofnextcut), CInt(dayofnextcut))
    ↪ 'nextcut_dayofyear is the dayofnextcut variable in day of year format.
growingdays = nextcut_dayofyear + (365 - cut_dayofyear)

```

Else *'if not the final cut, next cut is the cut i+1*

```

monthofnextcut = FORM_crops.Controls("combo_month_cut" & CStr(count
    ↪ + 1)).Value
dayofnextcut = FORM_crops.Controls("Box_day_cut" & CStr(count +
    ↪ 1)).Value
nextcut_dayofyear = dayofyear(CStr(monthofnextcut), CInt(dayofnextcut))
    ↪ 'nextcut_dayofyear is the dayofnextcut variable in day of year format.
growingdays = nextcut_dayofyear - cut_dayofyear

```

End If

*\*\*\*needs work*

```

DMNArray(nlayers + 27 + count) = "C" & CStr(count) & Space(5 - (2 *
    ↪ Len(count))) & CStr(count) & " .F" & Space(5 - Len(cut_dayofyear)) &
    ↪ cut_dayofyear & Space(5 - Len(growingdays)) & growingdays & _
    Space(8 - Len(Format(CDbl(potyield(count)), "###00.0"))) &
    ↪ Format(CDbl(potyield(count)), "###00.0") & " 0.30 1.00 0.00
    ↪ 3.00 1.50 0 1 1 1 1 1 1.00"

```

```
DMNArray(nlayers + 28 + count) = ""
```

```
DMNArray(nlayers + 29 + count) = ""
```

```
DMNArray(nlayers + 30 + count) = ""
```

```
DMNArray(nlayers + 31 + count) = ""
```

```
DMNArray(nlayers + 32 + count) = ""
```

```
Next count
```

```
If ncuts < ncrops Then 'signals legumes have been chosen
```

```
DMNArray(nlayers + 27 + ncrops) = "C" & CStr(ncrops) & Space(5 - (2 *  
    ↪ Len(ncrops))) & CStr(ncrops) & ".T"
```

```
End If
```

```
Call DMN_write_file(projectname, CInt(n))
```

```
Erase DMNArray
```

```
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
```

```
'END Writing Portion of .DMN file that has crop information.
```

```
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
```

```
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
```

```
' Begin Writing Portion of .GEN file that has crop information.
```

```
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
```

```
Call GEN_read_file(projectname, CInt(n))
```

```

GENArray(23) = "*** Trafficability *** " & _
" "

GENArray(24) = " 1 1 1 1 820 0.01 9.99 0.00"
GENArray(25) = "12311231 820 0.01 9.99 0.00"
GENArray(26) = "*** Crop *** " & _
" "

GENArray(27) = " 0.300" '(wilting point is 0.300 cm3/cm3)
GENArray(28) = " 1 11231 30.00 " ' wet and dry periods are the whole year.
GENArray(29) = " 1 11231" ' Crops start experiencing stress
GENArray(30) = "11" 'when water table drops below 20 cm
GENArray(31) = " 1 1 40.00 416 40.00 5 4 40.00 517 40.00" & _
" 6 1 40.00 620 40.00 718 40.00 820 40.00"
GENArray(32) = " 924 40.00 925 4.001231 40.00"
Call GEN_write_file(projectname, CInt(n))
Erase GENArray

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
'END Writing Portion of .GEN file that has crop information.
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

'Begin Writing Portion of .PRJ file that has crop information.
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

Call PRJ_read_file(projectname, CInt(n))
PRJarray(35) = "[Crops]"
PRJarray(36) = "NumCrop=" & ncrops

```





soybean: MsgBox "Soybean Sub under Construction."

'get user-defined plant and harvest dates.

'All row crops (corn, soybean, corn\soybean, wheat, and cotton)

parameters are taken from examples included in the drainmod 6 distribution.

'dates are simply altered based upon the user's planting and

'harvest dates

```
plantdate = FORM_crops.BOX_rowplantdate.Value
```

```
harvestdate = FORM_crops.BOX_rowharvestdate.Value
```

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

```
'Begin writing .CIN file
```

,,,,,,,,,,,,,,,,,,,,,

,,,,,,,,,,,,,

' End writing .CIN file

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

Exit Sub

'

'

' || Corn/Soybean Option

'

'

cornsoybean: MsgBox "corn/soybean sub under construction"

```

'get user-defined plant and harvest dates.
'All row crops (corn, soybean, corn\soybean, wheat, and cotton)
'parameters are taken from exaples incuded in the drainmod 6 distribution.
'dates are simply altered based upon the user's planting and
'harvest dates
plantdate = FORM_crops.BOX_rowplantdate.Value
harvestdate = FORM_crops.BOX_rowharvestdate.Value
  If harvestdate > plantdate Then
    growingdays = harvestdate - plantdate
  Else
    growingdays = 365 - plantdate + harvestdate
  End If

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
'Begin writing .CIN file
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

filepath = folderpath & "\crops\" & projectname & CStr(i) & ".CIN"
Set fs = CreateObject("scripting.filesystemobject")
Set a = fs.createTextFile(filepath, True) 'create the text file *projectname*_*i*.CIN
a.Close 'close it because we want to print not write lines (dont want commas or " inserted)
filenumber = FreeFile 'set the numerical alias as the next avaiable (unused) file number

'* Reminder: Drainmod is a Fortran 77 program that requires fixed with input
  'text files-- mind your spacing when editing!"
Open filepath For Output As #filenumber
Print #filenumber, "*** First Possible and last possible" & _

```





```

    If harvestdate > plantdate Then
        growingdays = harvestdate - plantdate
    Else
        growingdays = 365 - plantdate + harvestdate
    End If
windowbegin = plantdate - 8
    If windowbegin <= 0 Then
        windowbegin = 365 + windowbegin
    End If
windowend = plantdate + 164
    If windowend > 365 Then
        windowend = windowend - 365
    End If

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
'Begin writing .CIN file
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

filepath = folderpath & "\crops\" & projectname & CStr(i) & ".CIN"
Set fs = CreateObject("scripting.filesystemobject")
Set a = fs.createTextFile(filepath, True) 'create the text file *projectname*_i*.CIN
a.Close 'close it because we want to print not write lines (dont want commas or " inserted)
filenumber = FreeFile 'set the numerical alias as the next available (unused) file number

'* Reminder: Drainmod is a Fortran 77 program that requires fixed with input
   'text files— mind your spacing when editing!"
Open filepath For Output As #filenumber
Print #filenumber, "*** First Possible and last possible" & _
" dates for crop *** "

```

```

Print #filename, Space(3 - Len(windowbegin)) & windowbegin & _
    Space(4 - Len(windowend)) & windowend
Print #filename, "*** Weir Control *** " & _
" "
Print #filename, " 1"
Print #filename, " 1 0 1 0 1 0 1 0 1 0 1 0 1 0 " & _
"1 0 1 0 1 0 1 0"
Print #filename, "*** Trafficability *** " & _
" "

    'calculate first workperiod dates
DOY = plantdate - 15
    If DOY <= 0 Then
        DOY = DOY + 365
    End If

work_period_startmonth = DOY_to_month(CInt(DOY))
work_period_startday = DOY_to_dayofmonth(CInt(DOY))

DOY = plantdate + 123
    If DOY > 365 Then
        DOY = DOY - 365
    End If
work_period_endmonth = DOY_to_month(CInt(DOY))
work_period_endday = DOY_to_dayofmonth(CInt(DOY))

Print #filename, Space(2 - Len(work_period_startmonth)) & work_period_startmonth & _
Space(2 - Len(work_period_startday)) & work_period_startday & _
Space(2 - Len(work_period_endmonth)) & work_period_endmonth & _

```

```

Space(2 - Len(work_period_endday)) & work_period_endday & _
" 820 3.9 1.2 2.0"
Print #filenumber, "12311231 820 3.9 1.2 2.0"
Print #filenumber, "*** Crop *** " & _
" "
Print #filenumber, " 930 2 2 30.00"
Print #filenumber, " 930 2 2"
Print #filenumber, "*** Root Depths *** " & _
" "
Print #filenumber, " 8"
Print #filenumber, " 1 1 42.00 1 8 30.00 128 15.00 3 2 3.0010 1 " & _
"3.001031 14.0011 9 30.001231 42.00"
Print #filenumber, "*** Yield Inputs *** " & _
" "
Print #filenumber, " 1"
Print #filenumber, " 278 126 .87000 1.00000 1.70000 15.00000"
Print #filenumber, "32 7 .00000 .00000 .00000 .00000 .00000 1.00000"
Print #filenumber, "100.0000 1.2200100.0000 .7100 273 130 1"
Print #filenumber, " 0 29 .19 30 49 .13 50 64 .19 65 79 .26 80 95 .25 96114 .08115120 .01"
Print #filenumber, " .00 .00 .00 .00 .00 .00 .05
    ↪ .051.001.001.001.001.752.102.101.301.301.301.301.30"
Print #filenumber, "1.201.00 .50 .00 .00 .00 .00 .00 .00 .00 .00"
Print #filenumber, "*** Salinity Modifications *** "
Print #filenumber, "Threshold Slope "
Print #filenumber, " 4800.000000 8.880000E-03"
Print #filenumber, "*** Irrigation Water Salinity *** "
Print #filenumber, " 0"
Print #filenumber, ""

```

```

Close #filenumber
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
' End writing .CIN file
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

Unload FORM_crops

Exit Sub

?|,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
?|,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
?| Cotton Option
?|,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
?|,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

cotton: MsgBox "Cotton Sub under construction"

'get user-defined plant and harvest dates.
'All row crops (corn, soybean, corn\soybean, wheat, and cotton)
'parameters are taken from exaples incuded in the drainmod 6 distribution.
'dates are simply altered based upon the user's planting and
'harvest dates

plantdate = FORM_crops.BOX_rowplantdate.Value
harvestdate = FORM_crops.BOX_rowharvestdate.Value

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
'Begin writing .CIN file
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
' End writing .CIN file

```





Erase TAGheader

Call DMN\_read\_file(projectname, CInt(n))

DMNArray(4) = " 1" & Space(6 - Len(ncrops)) & ncrops *'n rotations and n crops*

DMNArray(nlayers + 25) = "\*Crops"

DMNArray(nlayers + 26) = "&=====

DMNArray(nlayers + 27) = "&C IC Leg Pday Lgrw PotYld HI RSR CrpN ShtN RotN Tl

↪ Fr Rt Up Rs Mn RtCf"

Do Until count = ncrops

count = count + 1

potyield = CDbI(FORM\_crops.BOX\_RSR.Value) \* maxrootdepth \* 88

*'potential yeild is the maximum height(inches) \* 200 lb/acre.*

*' this translates into maximum root depth (cm) \* root to shoot ratio \* 88 kg/hectare*

DMNArray(nlayers + 27 + count) = "C" & CStr(count) & Space(5 - (2 \*

↪ Len(count))) & \_

CStr(count) & " .F" & Space(4) & "1" & Space(2) & "365" & \_

Space(8 - Len(Format(potyield, "###00.0")))) & \_

Format(potyield, "###00.0") & " 0.01 " & Space(4 -

↪ Len(FORM\_crops.BOX\_RSR.Value)) & \_

FORM\_crops.BOX\_RSR.Value & " 2.20 2.20 2.20 0 1 1 1 1 1.00"

DMNArray(nlayers + 28 + count) = "\*Tillage"

DMNArray(nlayers + 29 + count) = "&=====

DMNArray(nlayers + 30 + count) = "\*Fertilization" *'the fertilizer application*





GENarray(30) = "12" '12 root depths (1 a month)

Dim phase As Integer

If monthmax = 1 Then phase = 2

If monthmax = 2 Then phase = 1

If monthmax = 3 Then phase = 12

If monthmax = 4 Then phase = 11

If monthmax = 5 Then phase = 10

If monthmax = 6 Then phase = 9

If monthmax = 7 Then phase = 8

If monthmax = 8 Then phase = 7

If monthmax = 9 Then phase = 6

If monthmax = 10 Then phase = 5

If monthmax = 11 Then phase = 4

If monthmax = 12 Then phase = 3

Dim janrootdpth, febrootdpth, marrootdpth, aprrootdpth, mayrootdpth, junrootdpth As

↪ Double

Dim julrootdpth, augrootdpth, seprootdpth, octrootdpth, novrootdpth, decrootdpth As

↪ Double

janrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) \* Sin((phase + 1) \* 2 \*

↪ WorksheetFunction.Pi / 12))

febrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) \* Sin((phase + 2) \* 2 \*

↪ WorksheetFunction.Pi / 12))

marrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) \* Sin((phase + 3) \* 2 \*

↪ WorksheetFunction.Pi / 12))

```

aprrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 4) * 2 *
    ↪ WorksheetFunction.Pi / 12))
mayrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 5) * 2 *
    ↪ WorksheetFunction.Pi / 12))
junrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 6) * 2 *
    ↪ WorksheetFunction.Pi / 12))
julrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 7) * 2 *
    ↪ WorksheetFunction.Pi / 12))
augrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 8) * 2 *
    ↪ WorksheetFunction.Pi / 12))
seprootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 9) * 2 *
    ↪ WorksheetFunction.Pi / 12))
octrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 10) * 2 *
    ↪ WorksheetFunction.Pi / 12))
novrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 11) * 2 *
    ↪ WorksheetFunction.Pi / 12))
decrootdpth = avgrootdepth + ((maxrootdepth - avgrootdepth) * Sin((phase + 12) * 2 *
    ↪ WorksheetFunction.Pi / 12))

GENarray(31) = " 1 1" & Space(6 - Len(janrootdpth)) & janrootdpth & _
" 2 1" & Space(6 - Len(febrootdpth)) & febrootdpth & _
" 3 1" & Space(6 - Len(marrootdpth)) & marrootdpth & _
" 4 1" & Space(6 - Len(aprrootdpth)) & aprrootdpth & _
" 5 1" & Space(6 - Len(mayrootdpth)) & mayrootdpth & _
" 6 1" & Space(6 - Len(junrootdpth)) & junrootdpth & _
" 7 1" & Space(6 - Len(julrootdpth)) & julrootdpth & _
" 8 1" & Space(6 - Len(augrootdpth)) & augrootdpth

```

```

GENarray(32) = " 9 1" & Space(6 - Len(seprootdpth)) & seprootdpth & _
"10 1" & Space(6 - Len(octrootdpth)) & octrootdpth & _
"11 1" & Space(6 - Len(novrootdpth)) & novrootdpth & _
"12 1" & Space(6 - Len(decrootdpth)) & decrootdpth
Call GEN_write_file(projectname, CInt(n))
Erase GENarray

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
'END Writing Portion of .GEN file that has crop information.
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
'Begin writing .CIN file
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

filepath = folderpath & "\crops\" & projectname & CStr(i) & ".CIN"
Set fs = CreateObject("scripting.filesystemobject")
Set a = fs.createTextFile(filepath, True) 'create the text file *projectname*_i*.CIN
a.Close 'close it because we want to print not write lines (dont want commas or " inserted)
filenumber = FreeFile 'set the numerical alias as the next available (unused) file number

'* Reminder: Drainmod is a Fortran 77 program that requires fixed with input
  'text files-- mind your spacing when editing!'

Open filepath For Output As #filenumber
Print #filenumber, "*** First Possible and last possible dates for crop *** "
Print #filenumber, " 1 365"
Print #filenumber, "*** Weir Control *** "
Print #filenumber, " 1"
Print #filenumber, " 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0"

```

```

Print #filename, "*** Trafficability *** "
Print #filename, " 1 1 1 1 720 0.01 9.99 0.00"
Print #filename, " 1 1 1 1 720 0.01 9.99 0.00"
Print #filename, "*** Crop *** "
Print #filename, " 1 11231 30.00"
Print #filename, " 1 11231"
Print #filename, "*** Root Depths *** "
Print #filename, "12"
Print #filename, " 1 1" & Space(6 - Len(janrootdpth)) & janrootdpth & _
    " 2 1" & Space(6 - Len(febrotdpth)) & febrotdpth & _
    " 3 1" & Space(6 - Len(marrootdpth)) & marrootdpth & _
    " 4 1" & Space(6 - Len(aprrootdpth)) & aprrootdpth & _
    " 5 1" & Space(6 - Len(mayrootdpth)) & mayrootdpth & _
    " 6 1" & Space(6 - Len(junrootdpth)) & junrootdpth & _
    " 7 1" & Space(6 - Len(julrootdpth)) & julrootdpth & _
    " 8 1" & Space(6 - Len(augrootdpth)) & augrootdpth
Print #filename, " 9 1" & Space(6 - Len(seprootdpth)) & seprootdpth & _
    "10 1" & Space(6 - Len(octrootdpth)) & octrootdpth & _
    "11 1" & Space(6 - Len(novrootdpth)) & novrootdpth & _
    "12 1" & Space(6 - Len(decrootdpth)) & decrootdpth
Print #filename, "*** Yield Inputs *** "
Print #filename, " 1"
Print #filename, " 1 365 0.0000 0.0000 0.0000 0.0000"
Print #filename, "26 7 11.16000 -1.17000 .05800 -.00050 100.00000 1.50000"
Print #filename, "100.0000 1.2200102.0000 0.7500 1 365 1"
Print #filename, " 0 290.20 30 490.22 50 690.32 70 890.19 901090.081101290.021301300.00"
Print #filename,
    ↪ "0.000.000.000.000.000.500.501.001.001.001.001.752.002.001.301.301.301.301.20"

```

```
Print #filename, "1.000.500.000.000.000.00"
```

Print #filename, "\*\*\* Salinity Modifications \*\*\* "

Print #filename, "Threshold Slope "

Print #filename, " 0.000000E+00 0.000000E+00"

Print #filename, "\*\*\* Irrigation Water Salinity \*\*\* "

Print #filename, " 0"

Close #filename

,,,,,,,,,,,,,

' End writing .CIN file

,,,,,,,,,,,,,

,,,,,,,,,,,,,

'Begin Writing Portion of .PRJ file that has crop information.

,,,,,,,,,,,,,

Call PRJ\_read\_file(projectname, CInt(n))

PRJarray(35) = "[Crops]"

PRJarray(36) = "NumCrop=" & ncrops

For count = 37 To (ncrops + 36)

PRJarray(count) = "Crop" & CStr(count - 36) & "=" & folderpath & \_  
"\crops\" & projectname & CStr(count - 36) & ".CIN"

Next count

PRJarray(ncrops + 37) = "[CropAlias]"

For count = (ncrops + 38) To (ncrops + ncrops + 37)

PRJarray(count) = "Crop" & CStr(count - 37 - ncrops) & "=" & \_  
folderpath & "\crops\" & projectname & CStr(count - 37 - ncrops) & ".CIN"

Next count



```

    For i = 1 To i = 100
' On Error GoTo errormsg_croptnotrun
        If DMNArray(i) = "*Fertilization" Then
            DMNArray(i + 1) = "&=====
            DMNArray(i + 2) = "&F AppDay FerType AddFer AddInh AppMeth IncDep"
            startrow = i + 3
            Exit For
        End If
    Next i

i = 7 'intialize, data starts on row 7
count = 0 'intialize the GENarray element counter
Do Until IsEmpty(Sheets("Fertilizer").Range("B" & CStr(i)))
count = count + 1
'entry 1 is blank, 2 is ammonium, 3 is anhydrous ammonium, 4 is nitrate,and 5 is manure
    If Sheets("Fertilizer").DropDowns("Drop Down " & CStr(count)).Value = 1 Then
        MsgBox "Please enter a fertalizer type for application #" & CStr(count)
        Exit Sub
    ElseIf Sheets("Fertilizer").DropDowns("Drop Down " & CStr(count)).Value = 2 Then
        nfert = nfert + 1
        DMNArray(startrow + count) = Space(2 - Len(nfert)) & CStr(nfert) & _
        space(8-len(sheets("Fertilizer").range("&cstr(i)))) & "& " " & _
        "2" & Space(7 - Len(Sheets("Fertilizer").Range("F" & CStr(i)))) & _
        Sheets("Fertilizer").Range("F" & CStr(i)) & " 1 "

    ElseIf Sheets("Fertilizer").DropDowns("Drop Down " & CStr(count)).Value = 3 Then
        nfert = nfert + 1

```



```

ElseIf Sheets("Fertilizer").DropDowns("Drop Down " & CStr(count)).Value = 4 Then
    nfert = nfert + 1
ElseIf Sheets("Fertilizer").DropDowns("Drop Down " & CStr(count)).Value = 5 Then
    nmanure = nmanure + 1
End If
i = i + 3 'data entered every three rows, itterate by 3s
Loop

i = 7 'intialize, data starts on row 7
count = 0 'intialize the GENarray element counter
Do Until IsEmpty(Sheets("Fertilizer").Range("B" & CStr(i)))
count = count + 1
    ElseIf Sheets("Fertilizer").DropDowns("Drop Down " & CStr(count)).Value = 5 Then
        nmanure = nmanure + 1
    End If
i = i + 3 'data entered every three rows, itterate by 3s
Loop

Next n

'errmsg_croptnotrun:
'MsgBox "There was an error writing you fertilzer applicaiton data." & _
""Please make sure you have run the crop routine for this project " & _
""prior to the fertilzer routine."

End Sub

```

```
Public Sub testerrr()  
MsgBox Sheets("Fertilizer").DropDowns("Drop Down 2")  
End Sub  
  
Public Sub noaction()  
  
End Sub  
  
Public Function Get_month_number(month As String)  
  
If month = "January" Then Get_month_number = 1  
If month = "February" Then Get_month_number = 2  
If month = "March" Then Get_month_number = 3  
If month = "April" Then Get_month_number = 4  
If month = "May" Then Get_month_number = 5  
If month = "June" Then Get_month_number = 6  
If month = "July" Then Get_month_number = 7  
If month = "August" Then Get_month_number = 8  
If month = "September" Then Get_month_number = 9  
If month = "October" Then Get_month_number = 10  
If month = "November" Then Get_month_number = 11  
If month = "December" Then Get_month_number = 12  
  
End Function
```

'<><><><><>' *This sub writes the TAN header line from the TAGheader array back to the*  
 ↪ *first line of TAGarray*

Public Sub TAG\_header\_write()

Dim count As Integer

Dim ITEM As Variant

count = 0 *'initialize generic counter*

TAGarray(0) = "" *'Erase first line of TAGarray since we are appending to TAGarray(1)*

↪ *in the loop*

For Each ITEM In TAGheader

If count = 0 Then

TAGarray(0) = TAGheader(count) *'No comma delimiter before the*

↪ *first item*

Else

TAGarray(0) = TAGarray(0) & "," & TAGheader(count)

End If

count = count + 1

Next

End Sub

'<><><>' *This sub splits the first element of the TAGarray (TAGarray(0)) by comma*

↪ *delimiters and places each element in the*

'<><>' *dynamic array TAGheader(). The purpose is to write specific elements to the*

↪ *TAGheader in any order throughout the program.*

```
Public Sub TAG_header_read()
```

```
    TAGheader = Split(TAGarray(0), ",") 'split the first element in TAGarray ( TAGarray(0)
```

```
    ↪ ) by comma delimiters
```

```
                                'and put split elements in the dynamic array
```

```
                                ↪ TAGheader
```

```
End Sub
```

```
'This form is called there is an error retrieving lat/long from a PRISM weather csv file
```

```
    ↪ (error_truefalse=true)
```

```
'OR when the user selects continue on the sheet 'Weather Input' without a lat/long being
```

```
    ↪ entered (error_truefalse=false)
```

```
Public Sub error_or_manual(error_truefalse As Boolean)
```

```
'check if this is being called by user or by error.
```

```
' Initialize the userform 'FORM_latlongerror' and display appropriate Form label
```

```
Load FORM_latlongerror
```

```
FORM_latlongerror.LABEL_error.Visible = False
```

```
FORM_latlongerror.LABEL_no_error.Visible = False
```

```
If error_truefalse Then
```

```
    FORM_latlongerror.LABEL_error.Visible = True
```

```
Else
```

```
    FORM_latlongerror.LABEL_no_error.Visible = True
```

```
End If
```

```

FORM_latlongerror.Show
End Sub

Public Sub BUTTON_reset_weather_inputs()

'double check they user hit 'clear' on purpose

Dim clear_or_abort As String

    clear_or_abort = MsgBox("Are you sure you want to clear the data on this sheet?",
        ↪ vbYesNo)

    If clear_or_abort = vbNo Then
        End
    End If

weather_unit_warning = False

'clear page, rewrite headers
Call turnoff_AUTOS

Worksheets("Weather_Input").Cells.ClearContents

'write non-unit dependent headers

Sheets("Weather_Input").Range("j2") = "Latitude (decimal)"
Sheets("Weather_Input").Range("j3") = "Longitude (decimal)"
Sheets("weather_input").Range("D3") = "Date"

Call turnon_AUTOS

    Call weather_input_reset_headers 'call this sub to write unit-dependent headers.

```

End Sub

Public Sub weather\_input\_reset\_headers()

Dim clear\_or\_abort As String

*'warn the user if they are trying to change the units after importing*

If weather\_unit\_warning = True Then

clear\_or\_abort = MsgBox("Warning! You are attempting to change the units of the weather

→ data after importing. Correct units are automatically set during import." & \_

"This action will lead to erroneous data if continue without manually converting values. Do

→ you want to continue with this action?", vbYesNo + vbExclamation +

→ vbDefaultButton2)

If clear\_or\_abort = vbNo Then

*'reset the units based upon PRISM headers*

If Sheets("WEATHER.INPUT").Range("E3").Value = "ppt (inches)" Then

→ Sheets("Weather.Input").Shapes("option\_inches").ControlFormat.Value = 1

If Sheets("WEATHER.INPUT").Range("E3").Value = "ppt (mm)" Then

→ Sheets("Weather.Input").Shapes("option\_mm").ControlFormat.Value = 1

If Sheets("WEATHER.INPUT").Range("F3").Value = "tmin (degrees F)" Then

→ Sheets("Weather.Input").Shapes("option\_F").ControlFormat.Value = 1

```

    If Sheets("WEATHER.INPUT").Range("F3").Value = "tmin (degrees C)" Then
        ↪ Sheets("Weather_Input").Shapes("option_C").ControlFormat.Value = 1
    'exit sub
    Exit Sub
End If
End If

Call turnoff_AUTOS

'The following headers display units based upon the option button selected.

If Sheets("Weather_Input").Shapes("option_inches").ControlFormat.Value = 1 Then
    Sheets("weather_input").Range("E3") = "Precip. (inches)"
ElseIf Sheets("Weather_Input").Shapes("option_cm").ControlFormat.Value = 1 Then
    Sheets("weather_input").Range("E3") = "Precip. (cm)"
ElseIf Sheets("Weather_Input").Shapes("option_mm").ControlFormat.Value = 1 Then
    Sheets("weather_input").Range("E3") = "Precip. (mm)"
End If

If Sheets("Weather_Input").Shapes("option_f").ControlFormat.Value = 1 Then
    Sheets("weather_input").Range("F3") = "Min.Temp.(F)"
    Sheets("weather_input").Range("G3") = "Max.Temp.(F)"
ElseIf Sheets("Weather_Input").Shapes("option_c").ControlFormat.Value = 1 Then
    Sheets("weather_input").Range("F3") = "Min.Temp.(C)"
    Sheets("weather_input").Range("G3") = "Max.Temp.(C)"
End If

If Sheets("Weather_Input").Shapes("option_dailyprecip").ControlFormat.Value = 1 Then

```

```

        Sheets("weather_input").Range("C3") = ""
    ElseIf Sheets("Weather_Input").Shapes("option_hourlyprecip").ControlFormat.Value = 1
        ↪ Then
            Sheets("weather_input").Range("C3") = "Hour(2-24)"
    End If
Call turnon_AUTOS

End Sub

Public Sub testfindjulian()
    Dim jd As String
    Call GetFolderPath
    jd = find_line_julian("02", "28", 1981)
    MsgBox "jd: " & jd
End Sub

Public Function find_line_julian(month As String, day As String, year As Integer)
    Dim str As String
    If (year Mod 4) = 0 Then
        str = "Julian_leap.txt"
    Else

```



```

str = "Julian_noleap.txt"
End If
Dim stringfilename As String
Dim errorflag As Boolean
errorflag = True

    stringfilename = folderpath & "\program\" & str
    Dim filenumber, i As Long
    filenumber = FreeFile 'set number as the next free number available to set as a numbered
        ↪ file location.
    Dim str1, str2 As String
    Open stringfilename For Input As filenumber
    i = 0 'initialize i at zero
Do While Not EOF(filenumber) 'search until the end of the file
    Line Input #filenumber, str1
    str2 = Left(str1, 5)
    i = i + 1

    If str2 = CStr(month) & "," & CStr(day) Then
        errorflag = False
        MsgBox Right(str1, 5)
        find_line_julian = CStr(Format(year + CLng(Right(str1, 5)), 0#))

        Exit Do

    End If
Loop

```

```

    If errorflag = True Then
        MsgBox ("unable to convert to julian date. Check your date selection.")
    End If

    Close filenumber

End Function

Public Sub PEST_write_DMN_TPL(projectname As String)
    Dim filepath As String
    Dim arr() As String
    Dim layernumber As String
    Dim str1, str2 As String
    Dim filenumber, i, n, count, length, nlayers As Integer
    Dim fs, a, b, c As Object

    For i = 1 To series_n
        Call TAG_read_file(projectname)
        arr() = Split(TAGarray(series_n), ",")
        nlayers = arr(0)
        Erase TAGarray

        Call DMN_read_file(projectname)
        filepath = folderpath & "\PEST\" & "DMN" & projectname & CStr(i) & ".tpl"
    
```

```

End Sub

'This routine runs a pest calibration based upon the entries placed in worksheet
    ↪ "Drainmod-PEST"
Public Sub PESTRUN()
Dim i, ntot, nhydro, nnit As Long
Dim nseries, maxlayer, yesno As Integer
Dim arr() As String
Dim req_obs As Double

i = 3
ntot = 0
nnit = 0
nhydro = 0

If IsEmpty(Sheets("Drainmod-PEST").Range("C3")) Then GoTo errormsg_nodata
Do Until IsEmpty(Sheets("Drainmod-PEST").Range("c" & CStr(i)))
    ntot = ntot + 1
    If Not IsEmpty(Sheets("Drainmod-PEST").Range("D" & CStr(i))) Then nhydro = nhydro + 1
    If Not IsEmpty(Sheets("Drainmod-PEST").Range("E" & CStr(i))) Then nnit = nnit + 1
    i = i + 1
Loop

'The following block finds the maximum number of layers by reading it from the TAG file
Call getprojectname
Call TAG_read_file(projectname)
arr = Split(TAGarray(0), ",")
nseries = CInt(arr(1))

```

```

i = 0
Do Until i = nseries
i = i + 1
arr = Split(TAGarray(i), ",")
If CInt(arr(0)) > maxlayer Then maxlayer = CInt(arr(0))
Loop

'This block determines 1)if there are enough data points for a calibration, and what to
    ↳ hydrology parameters to
'calibrate for if the number of discharge observation points is less than the number required for
    ↳ maximum calibration.
'Max calibration is matching point conductivities, lateral conductivities,max storage, and kirkham
    ↳ depth. number of obs.
'points required for this is [(max number of layers *2)+2]

If nhydro = 0 Then GoTo nitrateonly 'if no hydrology, we know the user wants to do a nitrate
    ↳ only calibration

'if there are hydrology and nitrate, but not enough hydrology, ask to just do nitrate or not
If maxlayer > nhydro And nnit > 0 Then
yesno = MsgBox("Not enough discharge measurements for hydrology calibration (You need at
    ↳ least " & CStr(maxlayer) & _
" for this simulation). Would you like to continue with" & _
" nitrate calibration only?", vbYesNo + vbQuestion, "Insuff. Hydrology Observation Points")
    If yesno = vbYes Then
        GoTo nitrateonly
    Else
        Exit Sub

```

```

End If

'finally, if there are only hydrology and not enough, we know they need more calibration points.
    ↪ Tell them so and exit sub.

ElseIf maxlayer > nhydro And nnit = 0 Then
MsgBox ("Not enough discharge measurements for hydrology calibration (You need at least " &
    ↪ CStr(maxlayer) & _
    " for this simulation).")
Exit Sub
End If

'Send the number of
req_obs = (2 * CDb1(maxlayer)) + 2
If nhydro >= req_obs Then
GoTo maxcal
Else
req_obs = (2 * CDb1(maxlayer))

End If

max cal:

GoTo nitrateonly

koklat:

GoTo nitrateonly

koonly:

```

GoTo nitrateonly

nitrateonly:

MsgBox "max number of layers is: " & CStr(maxlayer)

MsgBox "nnitrate" & CStr(nnit) & "ntot" & CStr(ntot) & "nhydro: " & CStr(nhydro)

Exit Sub

errmsg\_nodata: MsgBox "Error: No data found. Please make sure measurments have been

→ entered and all data starts on row 3."

Exit Sub

errmsg: MsgBox "Unexpected error creating calibration. Please check the data you have

→ entered and try again"

Exit Sub

End Sub

'This sub writes GEN template files for the PEST calibration.

Public Sub PEST\_write\_GEN\_TPL(projectname As String)

Dim filepath As String

```

Dim arr() As String
Dim layernumber As String
Dim str1, str2 As String 'generic strings
Dim filenumber, i, n, count, length, nlayers As Integer
Dim fs, a, b, c As Object

'<>Begin Writing .GEN template file<>
For i = 1 To series_n
Call TAG_read_file(projectname)
arr() = Split(TAGarray(series_n), ",")
nlayers = arr(0)
Erase TAGarray

Call GEN_read_file(projectname, CInt(i)) 'read .GEN file lines into GENarray
filepath = folderpath & "\PEST\" & "GEN" & projectname & CStr(i) & ".tpl" 'GEN template
    ↳ files named GEN"projectname""series_n".tpl

'They are written to
    ↳ the "PEST"
    ↳ folder.

Set fs = CreateObject("scripting.filesystemobject")
Set a = fs.createTextFile(filepath, True) 'create a GEN PEST template file for each series
a.Close

filenumber = FreeFile() 'filenumber alias is set as next available integer
Open filepath For Output As #filenumber

```

```

Print #filenumber, "ptf $" 'line 1 of PEST template file. '$' is the marker denoting a parameters
    ↪ PEST will calibrate

For n = 1 To 12 'Not modifying the next 12 lines, loop to save space.
    Print #filenumber, GENArray(n)
Next n
str1 = Left(GENArray(13), 45) & "$DC" & CStr(i) & "$" & Right(GENArray(13), 30) 'Flag the
    ↪ Drainage Coefficient
str2 = "" 'as a calibrated parameter for each series.
Print #filenumber, str1
    For n = 14 To 20
        Print #filenumber, GENArray(n)
    Next n
    For n = 1 To nlayers
        count = (10 * (n - 1)) + 5
        str1 = Left(GENArray(21), count)
        length = 5
        str2 = str2 & Right(str1, length) & "$L" & i & n & "$"
    Next n
Print #filenumber, str2
For n = 22 To 100
    Print #filenumber, GENArray(n)
Next n
Erase GENArray
Close filenumber
Next i

```



End Sub

