Analysis of Error in a Model Predictive Irrigation Controller

By

Sam Ingersoll

SB, Mechanical Engineering and Electrical Engineering and Computer Science,
MIT, 2022

Submitted to the Department of Electrical Engineering and Computer Science in
Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science
at the
Massachusetts Institute of Technology
September 2023

Authored by: Sam Ingersoll
Department of Electrical Engineering and Computer Science
August 25, 2023

Supervised by: Amos G. Winter V.
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by: Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Analysis of Error in a Model Predictive Irrigation Controller

by

Sam Ingersoll

Submitted to the Department of Electrical Engineering and Computer Science
on August 25, 2023, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Significant portions of the world's agricultural land are vulnerable to desertification, leading to water shortages and changing climate conditions. Smart irrigation controllers could be part of the solution by helping farmers save water and adapt to changing climate without sacrificing yield. This thesis presents an analysis of sensitivity to crop model parameters in the MIT GEAR Lab's new POWEIr irrigation controller with the goal of making it cheaper and easier to deploy and therefore more accessible. The analysis shows that, of the four crop parameters, the controller is most sensitive to the crop coefficient (K_c), moderately sensitive to the maximum rooting depth (Z_r), less sensitive to depletion fraction (f_d), and almost completely independent of the the yield response factor (K_y). This result is potentially useful for designing calibration procedures for the deployment of the POWEIr Controller, especially where there may be limited ability to calibrate the controller.

Thesis Supervisor: Amos G. Winter V.
Title: Professor of Mechanical Engineering

Acknowledgments

I would like to thank Carolyn Sheline for all of her guidance and advice during this project, for answering all of my questions about the infinite agronomy parameters, and for letting me a be part of her research.

I would also like to thank Georgia Van de Zande, Fiona Grant, and the other members of the GEAR Lab for sharing their wisdom and work with me and for helping make this thesis possible.

I would like to thank Catherine "Cakky" Forrest for helping out when things went wrong.

Finally, I am very grateful to Amos Winter for all of his patience and measured guidance through the very end of this project and for giving me the opportunity to see so much that was new to me.

For Y., C., and C.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The United Nations Environmental Program reports that some 40% of global land area should be considered vulnerable to desertification with vulnerable regions spanning the globe. These areas are home to an estimated one sixth of the world's population [10]. Desertification leads to increasing temperatures and water scarcity, both of which can lead to impaired agriculture and food and water scarcity. At the same time, access to powerful computers in the form of mobile phones and to the internet is more prevalent than ever. In a study of 577 farming households in smallholder farming communities located in central Kenya, 98% of respondents owned a mobile phone [11]. Can we use these powerful, connected computers to help farmers around the world more strategically cultivate desertifying farmland while conserving dwindling water supplies?

1.2 Background

The MIT GEAR Lab has developed a smart irrigation system called the Predictive Optimal Water and Energy Irrigation (POWEIr) Controller which is designed to precisely track and control irrigation for efficient water use [16][15]. The purpose of the controller is to carefully control soil moisture to prevent over or under-irrigation

in order to save water without sacrificing yield. Currently, the POWEIr Controller is under test in Kenya, Jordan, and Morocco.

The POWEIr Controller is a model predictive controller (MPC) which controls irrigation by turning on and off the flow of water to drip irrigation emitters on the field in order to maximize crop yield while minimizing the amount of water used. A diagram of the controller system is shown in Figure 1-1. More specifically, the controller uses irrigation to control a quantity called root zone depletion (D_r) which is a measure of the amount of water in the soil that a plant can access. Figure 1-2 shows a schematic view from the AquaCrop manual illustrating D_r and the more general concept of the soil as a one dimensional water reservoir with inflows and outflows (see [14] for more details). In agronomic modeling, the soil is often modeled as a reservoir or "bucket" holding water. Using that idea, D_r is a measure of how empty the soil-bucket is. It has the units of mm and is the distance from the top of the soil to the water level. The "top" of the soil-bucket is the top of the soil and the "bottom" of the soil-bucket, also called total available water (or TAW, measured in mm) is the depth beyond which the roots can no longer get water from the soil. A D_r of zero corresponds to soil that is full of water (the distance from the top of the soil to the top of the water filling it is zero). On the other hand, when D_r is equal to TAW, there is no water left in the soil that the plant can access. D_r is the state variable that the POWEIr Controller controls through irrigation. The goal of the controller is to keep D_r below the point at which the plant is stressed from a lack of water, while at the same time using as little water for irrigation as possible. The exact cost function used in the controller is described in detail in [16]. The main objective of this thesis is to describe how error in different parameters used to model the crop and soil affect the POWEIr Controller's performance in terms of crop yield and water used. Put another way, how much yield is lost or extra water is used when the controller is not perfectly calibrated.

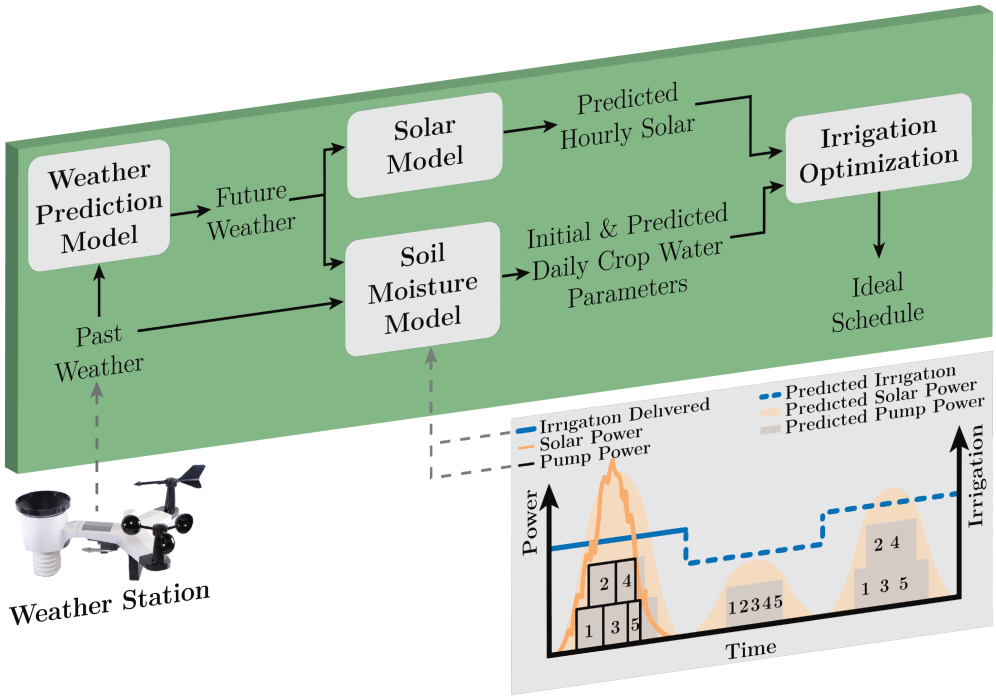


Figure 1-1: POWEIr Controller system architecture reprinted from [15]. Of particular interest in this thesis are the "soil moisture model" block and the "irrigation optimization" block.

1.2.1 Working Principles of the POWEIr Controller

The working of the POWEIr Controller is described in detail in [16] and [15], however, the working principles are summarized at a high level below.

The POWEIr Controller is a model predictive controller which is a type of optimization based controller. The MPC algorithm solves an optimization problem to minimize an objective function subject to dynamics (expressed as constraints) over a time horizon. The solution to the optimization problem is a schedule describing when and how much to irrigate. The "model" in model predictive control refers to the fact that the constraints of the optimization problem are often used to describe a dynamics model of the system being controlled. As such, the cost function and the dynamics model are essential in defining the controller.

The dynamics model in a model predictive controller is expressed in the constraints of the optimization problem and therefore has a significant impact the complexity of the optimization problem. As a result, it is important to use a dynamics model that

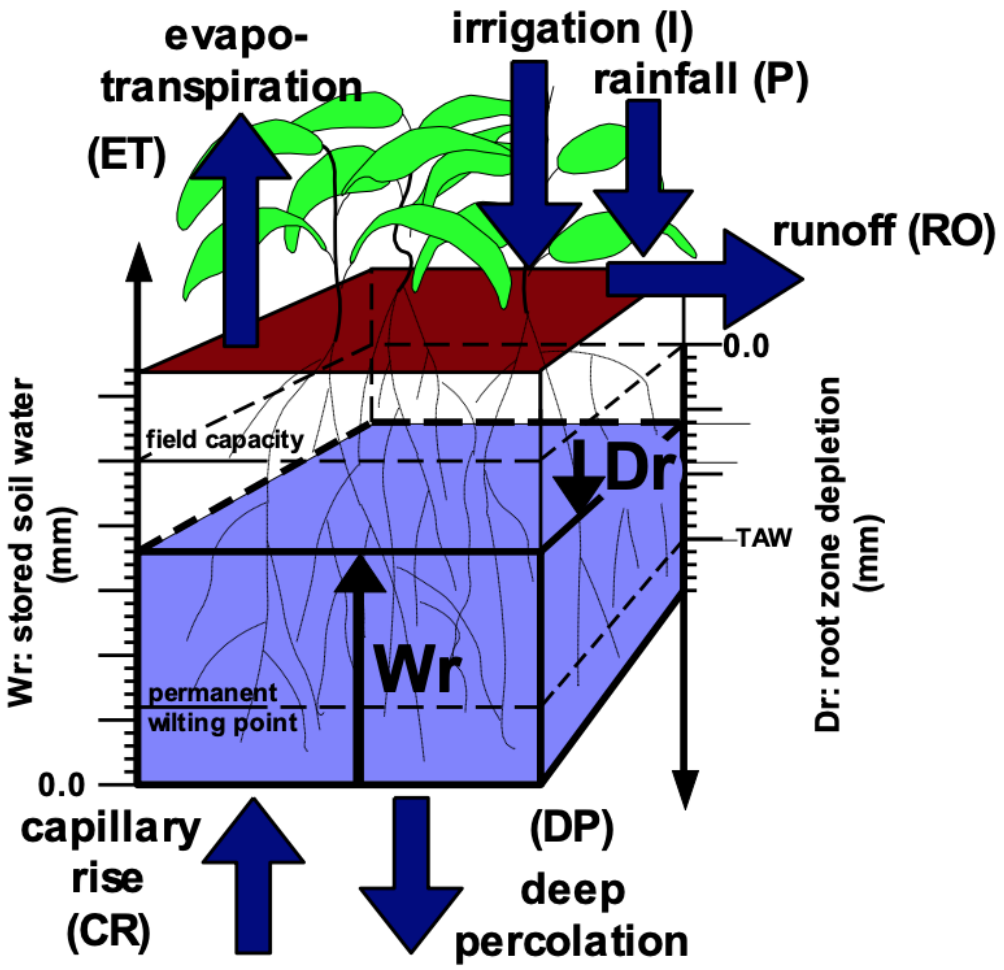


Figure 1-2: The soil represented as a 1D water reservoir with the important inflows and outflows indicated. Reprinted from [14]. Of particular notes is root zone depletion (D_r) which is the state variable that the POWEIr Controller is designed to control.

is rich enough to capture the complexity of the system while also simple enough to keep the optimization problem solvable. What makes a model "rich enough" and what makes an optimization problem "simple enough" must be determined using the system requirements. Chapter 3 will discuss these system requirements and controller performance further.

FAO56

The POWEIr Controller has an internal model describing the dynamics of the soil/crop system. The model used is called FAO56 and was first described in [2]. The integration of the model into the POWEIr Controller is described in more detail in [16] and [15], but the description of the general model, adapted from [15] is presented below.

The FAO56 model is formulated as a discrete time linear differential equation in which the soil water balance D_r is the independent variable.

$$D_{r,n} + (1 - k_{RO})Pr_n + \frac{1000 I_{del,n}}{A_s f_w} = D_{r,n-1} + K_{st} ET_{c,n}, \tag{1.1}$$

where $D_{r,n}$ and $D_{r,n-1}$ are the root zone depletion [mm] on day n and on the previous day, respectively. k_{RO} is the runoff coefficient corresponding to soil texture, I_{del} is the irrigation amount [m^3], A_s is the field's area [m^2], f_w is the soil wetted fraction, K_{st} is the water stress coefficient, and ET_c is the crop evapotranspiration [mm]. ET_c is given by $ET_c = K_c ET_0$, where K_c is the crop coefficient. K_{st} is calculated as $K_{st} = \frac{TAW - D_r}{TAW(1-f_d)}$ where f_d is depletion fraction calculated as $f_{d,n} = f_{d,const} + 0.04\,(5 - ET_{c,n})$. $f_{d,const}$ is a crop dependant constant defined in [2] and TAW is the total available water that the crop can extract from the soil [mm] which depends on the depth of the crop roots (Z_r) and soil texture [15].

D_r is constrained such that $0 \leq D_{r,n} \leq TAW$. If D_r is less than or equal to the readily available water ($RAW_n = d_n TAW$), then there is no water stress on the crop, $K_{st} = 1$. If the D_r is greater than RAW, then there is water stress and $0 < K_{st} < 1$. The water stress affects the amount of crop evapotranspiration as $ET_{adj} = K_{st} ET_c$. If there is water stress on the crop, then the reduction in evapotranspiration relates

to a reduction in yield,

$$1 - \frac{Y_a}{Y_{max}} = K_y \left(1 - \frac{ET_{c,adj}}{ET_c}\right), \tag{1.2}$$

where Y_{max} is the maximum yield [kg/m^2] calculated using the method described in [6] and K_y is the crop yield response factor [2] [15].

In the FAO56 model, three curves and one scalar parameter are used to describe the crop. K_c is the crop coefficient, it is a scalar valued parameter that changes with the growth of the plant. K_c describes how different a plant's evapotranspiration is from the reference crop (which is grass). K_c changes throughout the lifetime of the plant. K_y is the yield response factor which describes how the crop yield is affected by water stress, it also changes as the plant grows. Z_r is the rooting depth measured in meters, this parameter changes as the plant grows. f_d is the depletion fraction and represents the water level at which the plant becomes stressed due to lack of water. f_d is the depth of water at which the plant becomes water stressed divided by the maximum depth at which the plant can absorb water (TAW) and is therefore unitless and forms a sort of normalized ratio [2].

The FAO56 model is used to describe the evolution of the crop/soil system within the controller in part because it is simple and descriptive. A more detailed model is required to simulate the performance of the controller. The model we used for simulation is AquaCrop.

AquaCrop

The AquaCrop model was also published by FAO but is more than two decades more modern and significantly more complex than FAO56 [7]. Instead of a single (or a few) differential equations, AquaCrop models the relationship between many sub-systems. The functional relationships between the different model components are depicted in Figure 1-3. AquaCrop models the effects of cold and heat on pollination, the influence ground cover from foliage on evaporation, and much more. It is a very expressive model, but it is also very complex. Defining a custom crop in AquaCrop

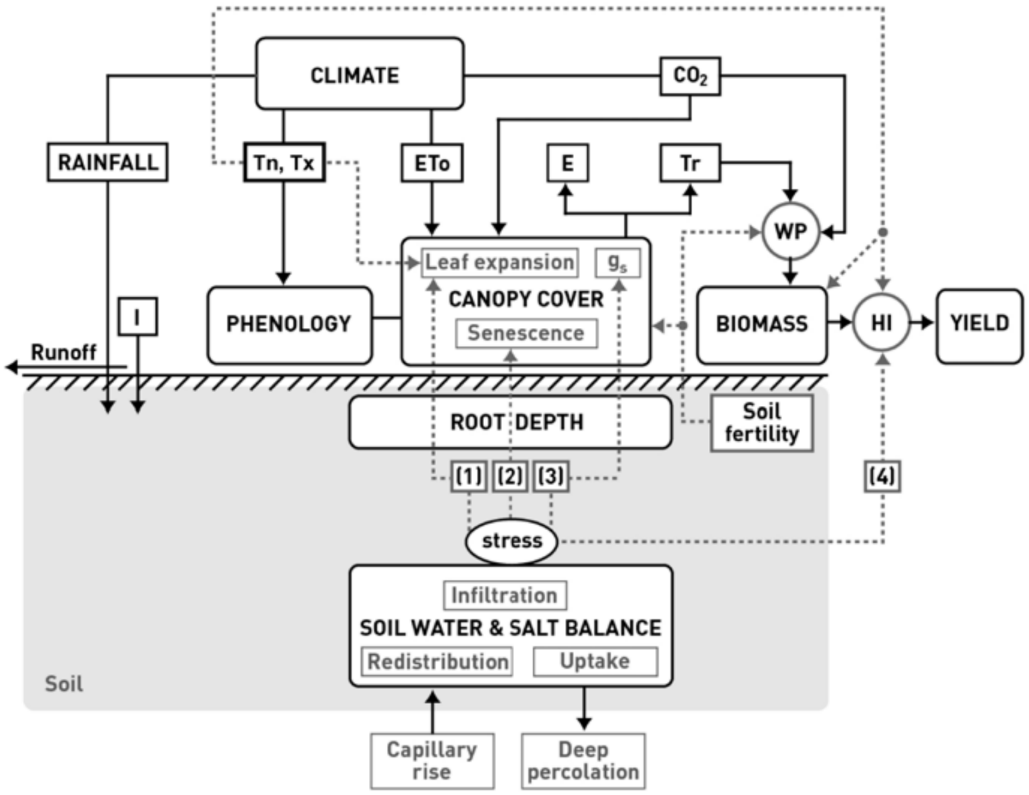


Figure 1-3: A block diagram showing the AquaCrop system architecture. Each block has its own state variables and is updated at every timestep. Reprinted from [7]

requires calibration of 59 parameters with 14 additional parameters listed as those that "...should not be changed without expert knowledge" [8]. The academic community has calibrated the model to many crops in many different environments such as in [5] and [3], so there are good default parameter values can be used, however, the complexity of the AquaCrop model, its many required libraries, and lack of a closed form mathematical representation makes it challenging to use for controller design. On the other hand, due to its richness, AquaCrop is a good fit for validating the performance of the POWEIr Controller.

Chapter 2

AquaCrop-Os Simulation Environment

This chapter discusses the first of the two contributions of this thesis: the integration of the POWEIr Controller and AquaCrop-OS (a Python port of the AquaCrop model) for efficient development and validation of the controller.

2.1 Motivation

The POWEIr Controller is a complex engineered system with many variables and possible sources of error. Furthermore, it has been designed to control the irrigation of potentially large numbers of farms on which people's livelihoods depend. As such, the validation and tuning of this controller is essential. As an analogue, validation for safety critical controllers is done using either control theory: proving convergence, bounded behavior, or robustness, or by approximating these results using Monte Carlo simulations [12][20]. The convergence and stability of model predictive controllers has historically been challenging to prove using control theory because the control law is the result of an optimization problem and therefore there is no closed form formulation [19]. Monte Carlo simulations have been used successfully as an alternative [12] [20]. Therefore, Monte Carlo style testing is an attractive route for validating the POWEIr Controller, especially because historical weather data for simulation is plentiful.

Initial validation of the GEAR Lab controller was done during development using AquaCrop GUI, a state of the art agricultural simulation software [16]. In this paper, I sometimes refer to "regular" AquaCrop as AquaCrop GUI to better distinguish is from AquaCrop-OS, the Python version. AquaCrop GUI is a user interface only application with no API exposed, meaning that each simulation must be prepared and run manually by clicking buttons on a user interface. The inability to interact programmatically with AquaCrop GUI prohibits true Monte Carlo simulations which typically require many simulated sessions in order to show that a controller is likely to be stable [12] [20].

The work in this thesis began with the integration of AquaCrop-OS and the existing POWEIr Controller with the goal of validating the performance of the controller. For the rest of this chapter, the system consisting of AquaCrop-OS and the POWEIr Controller will be referred to as the validation system.

2.2 System Description and Results

The main engineering consideration at play in the new validation system was ease of use. The accuracy of Monte Carlo simulations is often related directly to the number of simulations run [13]. Therefore, the usefulness of the validation system comes first and foremost from its ability to make running large numbers of tests easy. Performance is also important, however, the run-time of AquaCrop and the POWEIr Controller will always dominate and as the validation system simply links them together, it should not be resource intensive.

One of the main priorities of the validation system is to deal with the many input and output data files used and generated during simulation runs. A block diagram for the validation system is shown in Figure 2-1. The solution implemented is to define the validation system in a Python script. Before the script is run, the user can define experiments in a JSON file which points to the paths where input files can be found and paths where output files should be written. The system copies the input files and the experiment definition file into the output file directory so that all of the data used

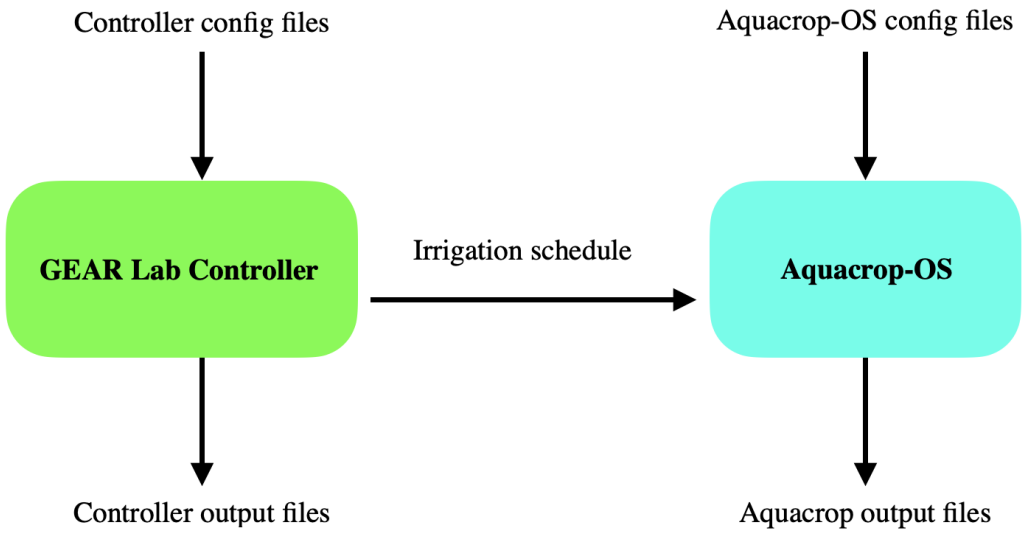


Figure 2-1: Block diagram showing the flow of information in the verification system. Each arrow corresponds to one or more data files generated during an experiment which must be saved.

in the experiment can be found. The data files are typically CSV files with 10-100 entries per day over the course of a growing season. This does not result in very large file sizes and keeping track of the data used in experiments is more valuable than saving the megabytes of disk space.

The results in Chapter 4 were all generated using the validation system. The simulations were run in batches of forty. Each batch was defined by providing a path to a directory containing the required parameter files and the program ran each valid file in the directory. Previously, only small experiments of around ten experiments had been run to validate the POWEIr Controller due to the impracticality of manually defining and starting large numbers of experiments. 128 experiments were run to create the final data for the document and many hundreds more were run along the way. With a batch of 80 experiments capable of completing in one night on a laptop and the ability to run the validation system on cloud infrastructure, the capability to test, tune, and validate the controller has been significantly increased.

Chapter 3

System Requirements, Model Complexity and Sensitivity

This chapter discusses model complexity and sensitivity in terms of calibrating a system and in terms of optimization problem complexity to provide context for the simulations in Chapter 4.

3.1 The Cost of Parameters

System parameters are what allow a general model of water moving through soil to describe water moving through an individual plot of sandy soil planted with onions in the Jordan Valley. Calibrating these parameters is essential for making a model useful for understanding physical behavior. However, as we will see in Chapter 4, accuracy in every system parameter is not equally important despite the fact that almost all of the model parameters take time and money to calibrate. Just as a model could be less sensitive to some parameters, some parameters could be cheaper and easier to calibrate than others. In this section, we will explore the challenges associated with calibrating different agricultural models.

The cost and effort of tuning a parameter in an agricultural model can vary significantly, so too can the model's sensitivity to this parameter. If a model is more sensitive to a given parameter, it will need to be calibrated more carefully, often

resulting in a higher cost. For instance, to measure evapotranspiration accurately, a lysimeter must be constructed. A lysimeter is a container of soil in which crops can be grown which measures water input and output from the contained soil through the growing season [1]. The result is a well calibrated evapotranspiration value for that specific crop in that specific soil. Although lysimeters can be constructed cheaply, running a lysimeter accurately is not trivial. However, if the model is very sensitive to the evapotranspiration parameter, a lysimeter may be a requirement for utilizing the model leading to increased cost and complexity of setting up the model.

An issue associated with high sensitivity to model parameters or high cost of calibrating parameters particular to agriculture is that agricultural parameters can vary significantly over a field. For instance, soil texture could vary greatly over a field near a sandy river bed. Agricultural parameters can also vary in time, for example, topsoil running off due to rain or flooding. In the worst case, these parameters can be thought of as only locally valid in time and space. Therefore, unless these parameters are measured at many points in the field and many times during the growing season, there will always be some appreciable error. In this case, high sensitivity to error in these parameters will result in poor aggregate performance even if the model can accurately predict the development of a specific plant in a specific row at a certain point in time. Additionally, high calibration cost can prohibit re-calibrating parameters which may change through time or space similarly resulting in degraded aggregate model accuracy.

As a result of the above, low model sensitivity to parameters is desirable. So too is a model formulation in which the parameters are easy and cheap to calibrate. Opposing these priorities are the requirement that the model is descriptive enough to usefully predict the behavior of the system.

3.2 Model Complexity from an Optimization Perspective

From a different perspective, in optimization based controllers the model is encoded in the constraints of the optimization problem and therefore model complexity has a great impact in controller performance. In this sense, complexity refers not just to the number of variables or how "complicated" the model is, but rather to the class of optimization problem that results from encoding the dynamics into the problem constraints. For example, powerful solvers exist for linear and quadratic programs [18]. The constraints in these problems are often limited to be linear, therefore limiting the system dynamics model to be linear. Good solvers exist for other specific flavors of mathematical program, however general nonlinear programs with arbitrary nonlinear constraints cannot necessarily be solved reliably or quickly.

Navigating the trade-off between the ability to express rich dynamics and constraints and reliably and quickly solving the optimization problem is at the crux of many realtime controls problems. In [4] the model predictive controller used to control the MIT Cheetah robot is run at 30 Hz. In the case of robots, the system (the physical robot) can be produced to tight tolerances making it possible to estimate model parameters like masses and inertias accurately from CAD. In these cases, the realtime requirements dominate and careful calibration and standardization of the system can be done in order to help achieve high performance. In our case, only one irrigation schedule must be generated each day corresponding to a control loop frequency of 1/86400 Hertz. Clearly, the POWEIr Controller is not subject to the same realtime performance constraints as the MIT Cheetah Controller. However, the POWEIr Controller is expected to control irrigation in farms with arbitrary soil composition, in arbitrary weather conditions, and with potentially different crop varieties while robot controllers are often only expected to control a single, well defined model of robot. Due to this requirement on the POWEIr Controller, ease of calibration and the capability of the model to generalize are more important when discussing controller performance than solve speed or computational resource requirements. To

this end, Chapter 4 seeks to understand the sensitivity of the POWEIr Controller to the four parameters used to define the crop within its internal model.

Chapter 4

Model Parameter Sensitivity Analysis

This chapter describes a set of simulated experiments in which the four model parameters in the POWEIr Controller's internal crop model are varied to imitate different degrees of miscalibration. The performance of the purposefully miscalibrated controller is evaluated by simulation using AquaCrop-OS. The architecture of the setup is described in Chapter 2. The resulting performance of the controller is analyzed and conclusions are drawn about the effects and relative importance of error in the calibration of each parameter.

4.1 Description of the Experiment

In the FAO56 model, a crop is described by four parameters. These parameters are: the yield response factor (K_y), the maximum effective rooting depth (Z_r), the crop coefficient K_c, and the soil water depletion fraction (f_d). The first three of the four parameters vary with time and growth stage of the plant, however, these curves have canonical shapes that are well described [17]. To vary these parameters, we scale the respective canonical curve shape by a scalar, see Figure 4-1. The fourth parameter: f_d, is a scalar and so we simply vary it linearly across experiments.

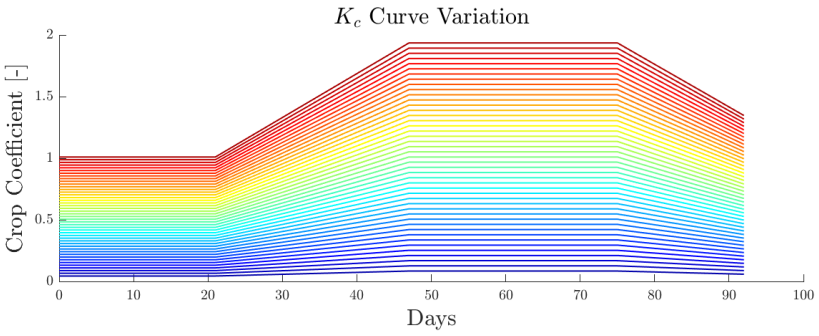
The GEAR Lab controller uses the FAO56 model to predict the behavior of the crop/soil system and therefore the parameters of this model are important to the performance of the controller. However, it is not obvious how much error in the

calibration of each of these parameters influences controller performance in terms of the final yield or irrigation used.

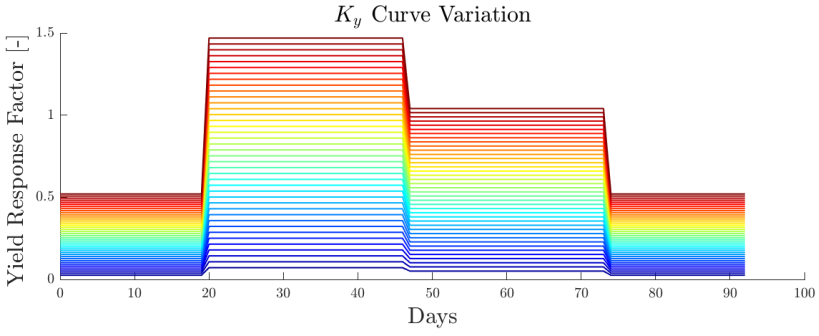
Additionally, yield and irrigation can both be assigned a monetary value corresponding to the value of the crops and the cost of water and power for irrigation. The calibration of model parameters can also be assigned an (approximate) monetary value based on hours of work, materials required, or cost of consultations with agronomists. As such, the relative costs and savings of calibrating parameters can be weighed against the potential loss of yield and over/under irrigation of the crop. This comparison is not explored in detail in this thesis, but the groundwork is laid for a relatively objective means by which to decide which model parameters need be calibrated and to what degree.

As discussed in Chapter 3, by simulating the controller's performance over many trajectories, it is possible to gain an understanding of its performance. It may be possible to understand the global behavior of the model across the whole parameter space by running simulations on a four dimensional grid in which each dimension represents a single parameter varied across its full range. This would be informative and perhaps useful in understanding some behaviors of the controller, however, it would require a huge number of simulations to be run and the goal of the analysis in this thesis is to understand the performance of the controller in expected parameter ranges based on calibrated values from the literature. As FAO56 is a mature model with significant academic work calibrating the model in different settings, we expect that the initial guess used for parameters in many cases will be the default values according to the literature and that parameter tuning will lie within the range of crops listed in the FAO56 model literature [17]. Accordingly, we vary each parameter individually through the range of values reported in the FAO56 model literature. These values can be found in the following tables.

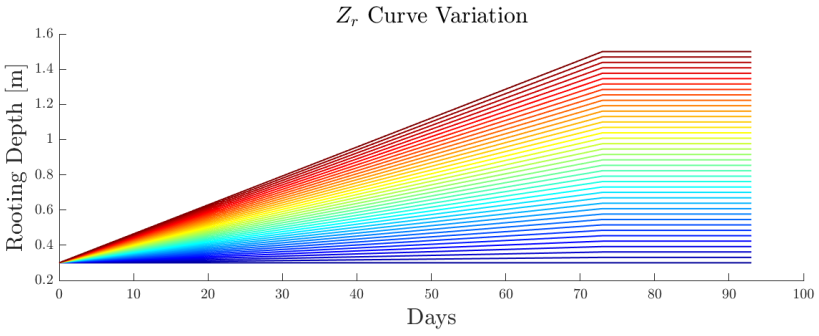
Figure 4-1 shows the crop input parameters for the FAO56 model varied across the ranges for vegetables described in Tables 4.1, 4.2, and 4.3. The resulting performance of the controller for these different parameters is discussed in the next section.



(a)



(b)



(c)

Figure 4-1: The full range of three out of the four FAO56 crop parameters. The three parameters are curves where the default shape from the literature is scaled by a constant to create the variants [17]. The fourth parameter (f_d, not shown) is a scalar value and is varied linearly.

Z_r **and** f_d **Values for Common Crops**		
Crop	Z_r **value** [m]	f_d **value** [-]
Alfalfa	1.0 - 2.0	0.55
Beans	0.5 - 0.7	0.45
Cabbage	0.5 - 0.8	0.45
Maize	1.0 - 1.7	0.55
Onion	0.3 - 0.6	0.3
Peas	0.6 - 1.0	0.35
Pepper	0.5 - 1.0	0.3
Potato	0.4 - 0.6	0.35
Sorghum	1.0 - 2.0	0.55
Soybean	0.6 - 1.3	0.5
Spring wheat	1.0 - 1.5	0.55
Sunflower	0.8 - 1.5	0.46
Tomato	0.7 - 1.5	0.4
Watermelon	0.9 - 1.5	0.4
Winter wheat	1.5 - 1.8	0.55

Table 4.1: Maximum Z_r and f_d values for common crops adapted from [2].

K_c **Values for Common Crops**			
Crop	**Initial** K_c **value** [-]	**Mid** K_c **value** [-]	**Final** K_c **value** [-]
Alfalfa	0.4	0.95	0.90
Beans	0.5	1.05	0.90
Cabbage	-	1.05	0.95
Maize	-	1.20	0.35 - 0.60
Onion	-	1.05	0.75
Peas	0.5	1.15	1.10
Pepper	-	1.05	0.7 - 0.9
Potato	-	1.15	0.75
Sorghum	-	1.05	0.55
Soybean	-	1.15	0.50
Spring wheat	-	1.15	0.25 - 0.4
Sunflower	-	1.0 - 1.15	0.35
Tomato	-	1.15	0.70 - 0.90
Watermelon	0.4	1.0	0.75
Winter wheat	0.70	1.15	0.25 - 0.4

Table 4.2: Time averaged K_c values for three growth phases of common crops adapted from [2].

K_y **Values for Common Crops**	
Crop	K_y **value** [-]
Alfalfa	1.1
Beans	1.15
Cabbage	0.95
Maize	1.25
Onion	1.1
Peas	1.15
Pepper	1.1
Potato	1.1
Sorghum	0.9
Soybean	0.85
Spring wheat	1.15
Sunflower	0.95
Tomato	1.05
Watermelon	1.1
Winter wheat	1.05

Table 4.3: Time averaged K_y values for common crops adapted from [17].

4.2 Discussion of the Results

Figure 4-2 illustrates the effect that varying each parameter has on yield and irrigation with the other parameters held constant. The resulting sensitivities are discussed for each parameter in the following subsections. Additionally, simulated trajectories are shown for extreme parameter values to illustrate how varying each parameter affects the irrigation schedule, root zone depletion (D_r), and readily available water (RAW) during a full growing season. These plots allow for a qualitative look at the different "behaviors" that extreme values for each parameter engender from the controller.

4.2.1 K_c

K_c is the crop coefficient which describes how different a crop's evapotranspiration is from the reference crop: grass. In the model, K_c is simply multiplied with the reference evapotranspiration value for grass.

For low K_c values, the controller's internal model predicts no evapotranspiration, so no irrigation is used. The "real" crop in the simulation has nonzero evapotranspiration and therefore the soil dries out, the crop is water stressed, and the resulting

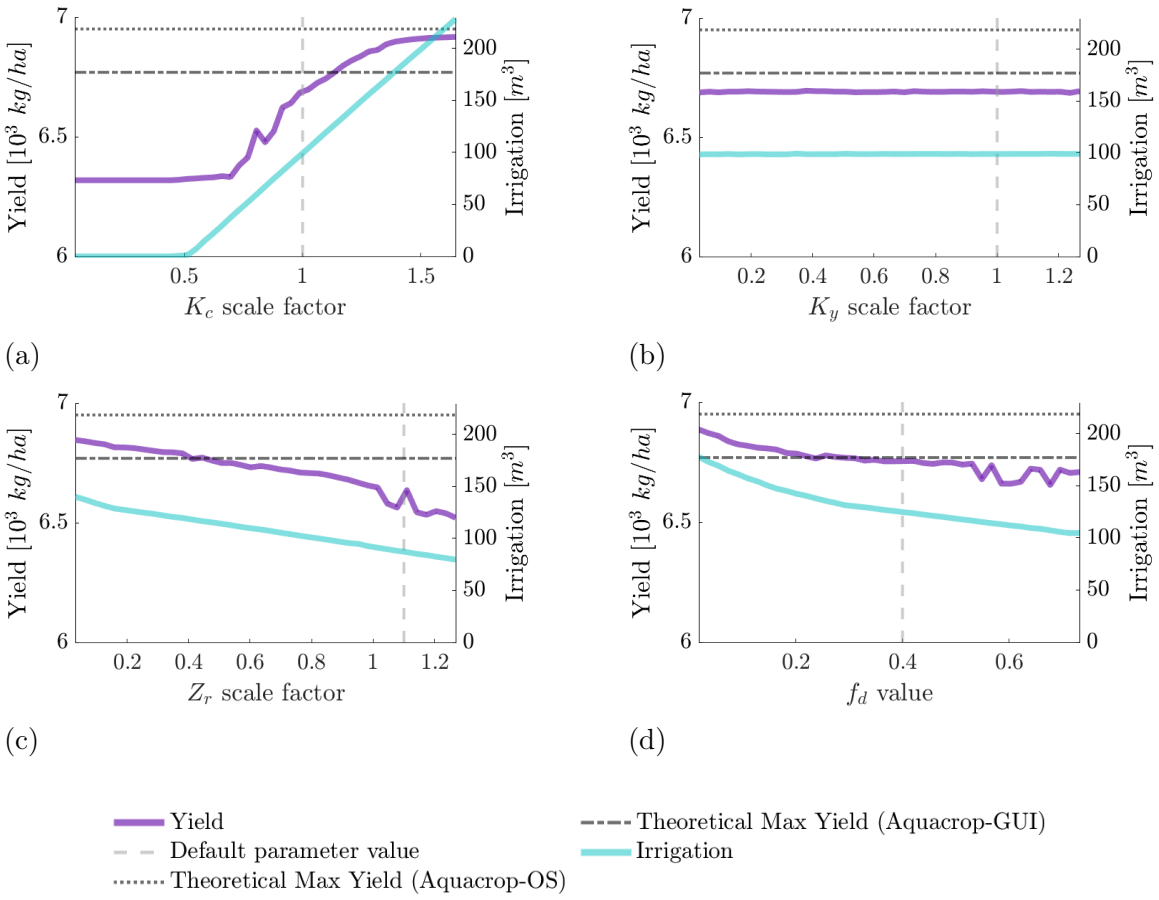


Figure 4-2: This figure shows the change in yield and irrigation associated with miscalibration or error in each crop parameter in the POWEIr Controller's internal model. The yield numbers were generated from an AquaCrop-OS simulation using ground truth weather data and system parameters. The gray dashed vertical lines show the default values of each parameter according to the the literature (see Tables 4.1, 4.2, and 4.3). The horizontal lines show the theoretical maximum yield from AquaCrop GUI and AquaCrop-OS which are not in perfect agreement (see Section 5.1).

yield is low.

From Figure 4-2a, it can be seen that as K_c increases, so does irrigation, and then slightly later, so does yield. For some time, the increase is proportional and monotonic for both variables until the yield nears the theoretical maximum yield at which point more irrigation does not lead to more yield. The yield and irrigation amount are sensitive to K_c values.

Figure 4-3 shows simulations for a growing season using three different K_c values. The values are the maximum, minimum, and middle values used in Figure 4-2a. In Figure 4-2a, the resulting yield values can be divided roughly into three regions, the first region (for low K_c values) is a constant low yield. The middle region is an approximately linear increase in yield with respect to K_c. The final region is the region where the yield curve approaches the maximum yield and levels off. The three plots in 4-3 correspond to these three regions. The minimum K_c plot (Figure 4-3a corresponds to a simulation from the first constant yield region of 4-2a. The "middle" K_c plot (Figure 4-3b) corresponds to a simulation run from the approximately linear region of 4-2a. Finally, the maximum K_c plot (Figure 4-3c), corresponds to a simulation using values from the final approximately asymptotic region where the curve nears the maximum yield in Figure 4-2a.

The plots in Figure 4-3 illustrate the controller going from heavily underestimating the evapotranspiration of the crop in Figure 4-3a and not irrigating at all as a result to heavily overestimating the the evapotranspiration of the crop in Figure 4-3c and over-irrigating as a result.

4.2.2 K_y

K_y is the yield response factor, a crop (and growth stage specific) sensitivity to water stress. It describes how water stress (lack of water) affects the resulting yield and can vary significantly between plants and throughout the growth of a plant.

Figure 4-2b shows the effects of varying K_y on yield and irrigation. From this figure, we can conclude that K_y has little effect on yield or irrigation under the conditions simulated. The full range of parameters results in a 0.0811% change in

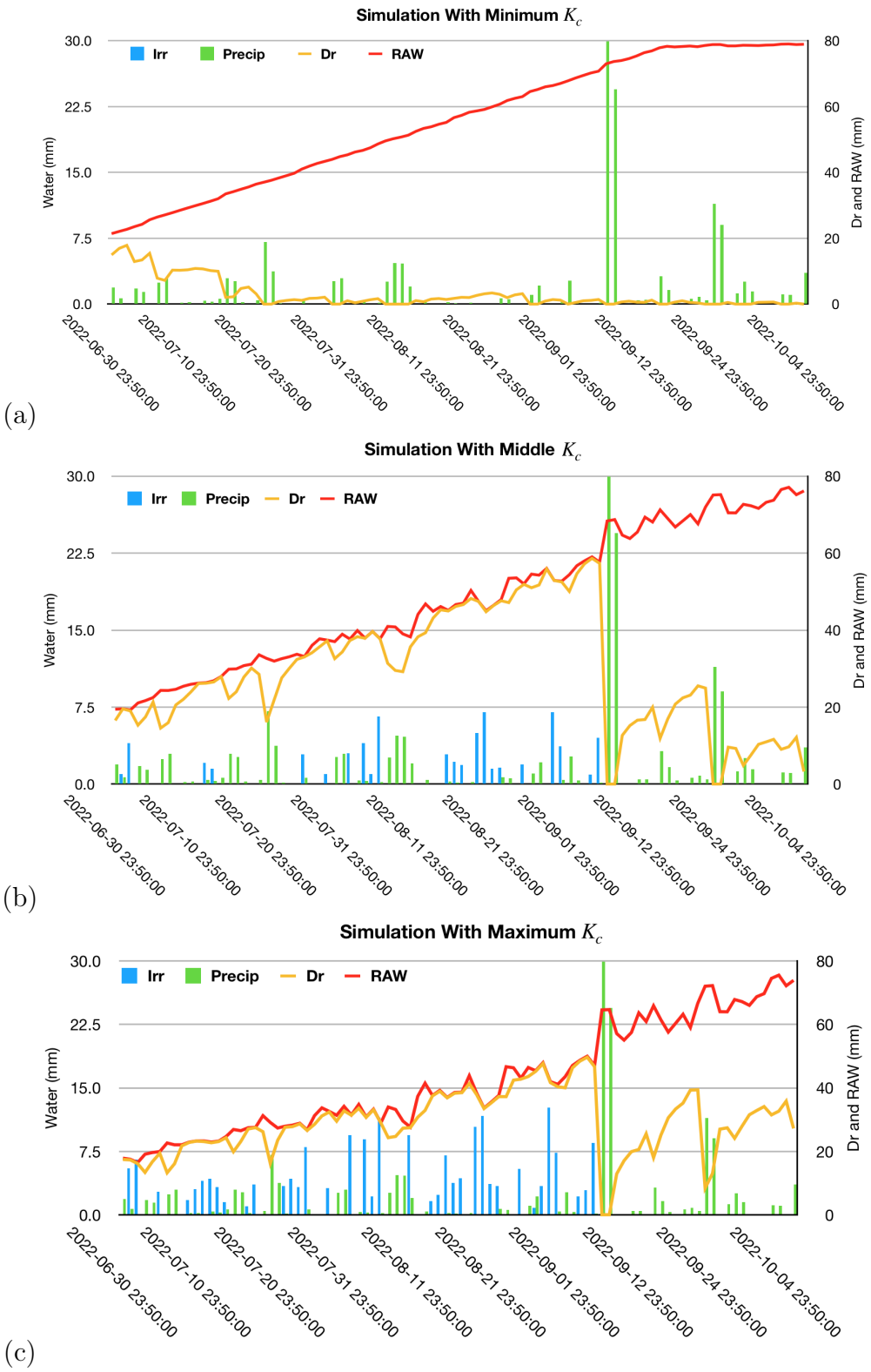
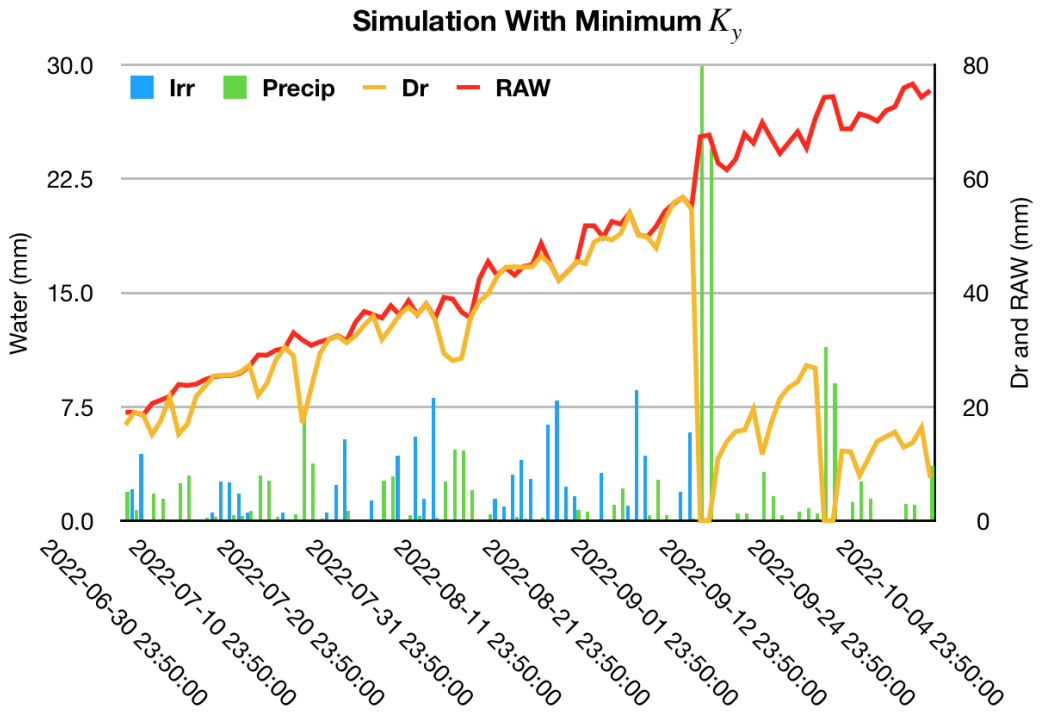
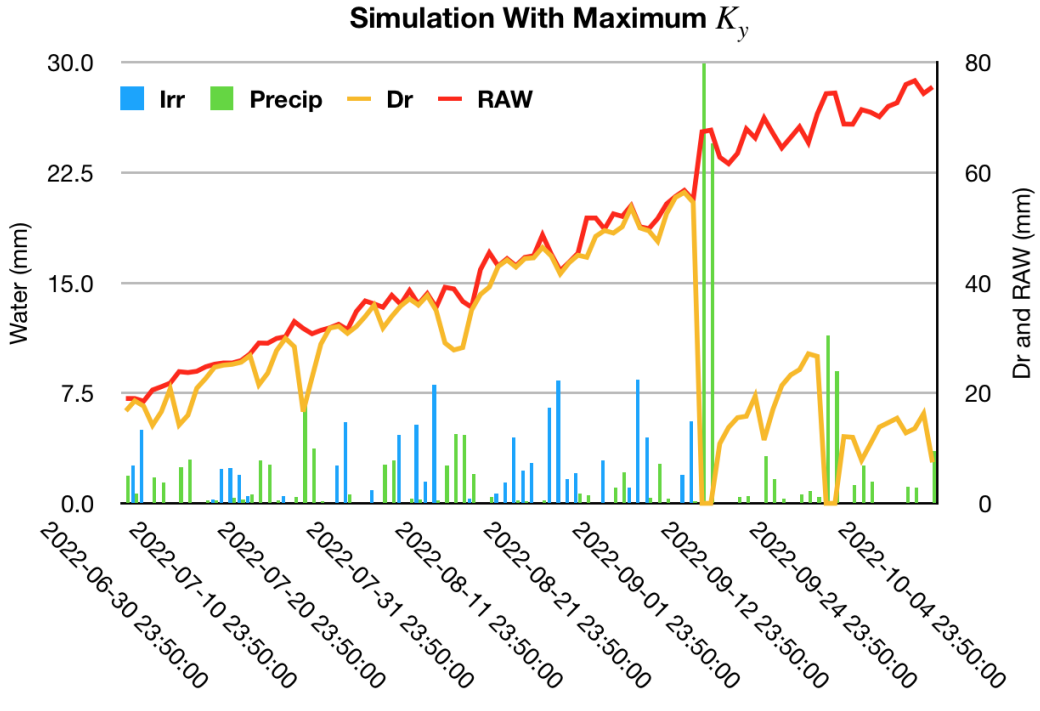


Figure 4-3: Simulated results for a growing season using minimum, middle, and maximum K_c values. Irrigation and precipitation are on the left axis, D_r and RAW are on the right axis.



(a)



(b)

Figure 4-4: Simulation results for a growing season using maximum and minimum K_y values. Irrigation and precipitation are on the left axis, D_r and RAW are on the right axis.

yield and a 0.51% change in irrigation volume. Figure 4-4 shows two simulated growing seasons, one with maximum K_y and one with minimum K_y. The resulting trajectories are very similar, reinforcing the relative independence of the controller's behavior from this parameter under these conditions. It is not surprising that K_y does not significantly influence yield or irrigation under the simulated conditions because the POWEIr Controller's cost function prioritizes the prevention of water stress (and therefore prevention of yield loss). K_y governs the severity of water stress once it has occurred, but the controller doesn't let water stress occur and therefore the effects of K_y are not visible in the simulation. This result also illustrates how in some cases it may be possible to design a controller to be nearly independent of error in a given parameter (which may be challenging to calibrate) making the controller easier and cheaper to set up as well as more reliable.

By significantly increasing the price of power or water, or by significantly undersizing the solar array or battery powering the irrigation pump, it would possible to put the controller in a position where yield must be sacrificed due to the exorbitant cost of or inability to irrigate the crops. If this were the case, K_y would have an effect on yield acting against the cost associated with water and power and acting alongside the term representing the price of the crops which the controller seeks to maximize. It is undoubtedly an important parameter in the model, however the terms associated with maximizing the dollar value of the yield perform a similar task in the cost function and often take priority. In future work, it may be worth investigating under what conditions K_y becomes relevant to controller performance and what effect it has in those cases.

4.2.3 Z_r

Z_r is the maximum effective rooting depth of the crop and is a depth measured in meters. It governs the depth of the water "bucket" that the soil can store to use the analogy from section 1.2. When Z_r is bigger, the plant has access to more soil and therefore a larger reservoir of water.

When simulating with calibration error resulting in a lower Z_r relative to the true

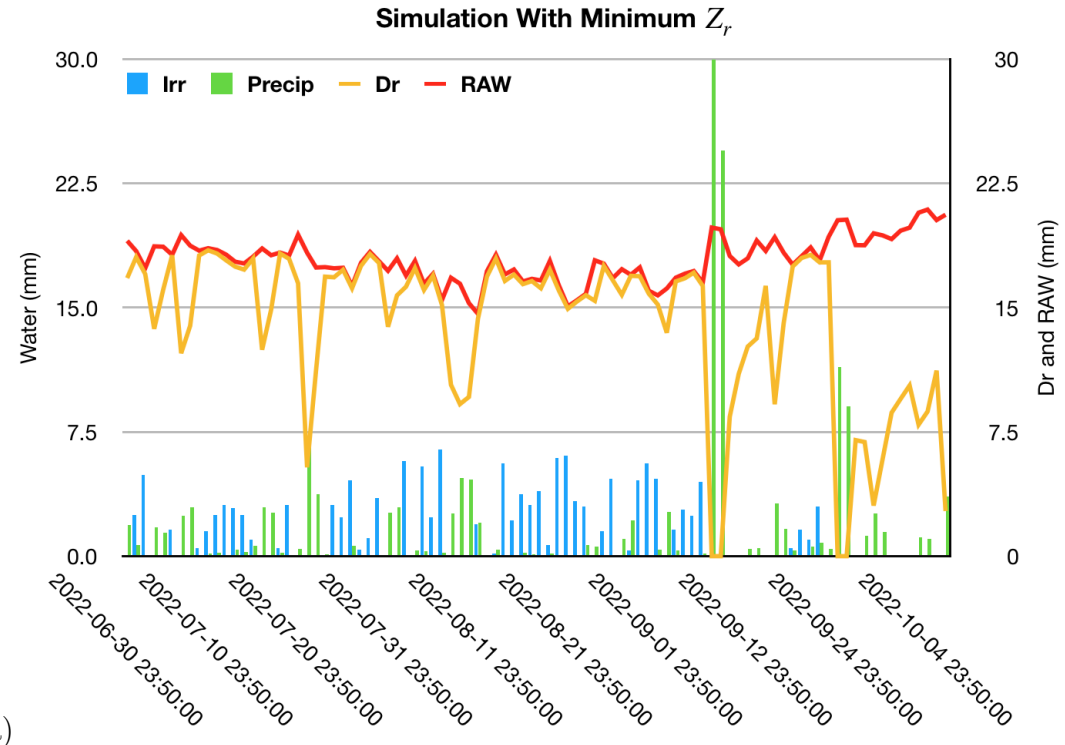
value, irrigation is high and yield remains near the theoretical maximum reported by AquaCrop. This is because in the controller's internal model, the soil "bucket" is small and dries out quickly, therefore it over-irrigates. Yield and irrigation both decrease with higher effective rooting depths relative to the true value. This means that plants with deeper roots need less irrigation. This was not necessarily an intuitive result at first and is worth further discussion. The plots in Figure 4-2c illustrate this trend.

Simulated growing seasons were run using the maximum and minimum Z_r values from Figure 4-2c. These schedules are shown in Figure 4-5. What can be seen is that in Figure 4-5a, more small irrigation events are needed to keep the plant from water stress. In Figure 4-5b, fewer, larger irrigation events are required. The explanation for this trend is that Z_r controls the size of the soil water reservoir that the plant has access to. The flows in and out of the soil water reservoir happen at the top (evaporation) and bottom (deep percolation) and for the most part do not depend on Z_r. Therefore, if the plant has a larger Z_r value, it has access to a larger volume of soil with the same inflows and outflows as a smaller plant which can therefore store more water over a longer period of time.

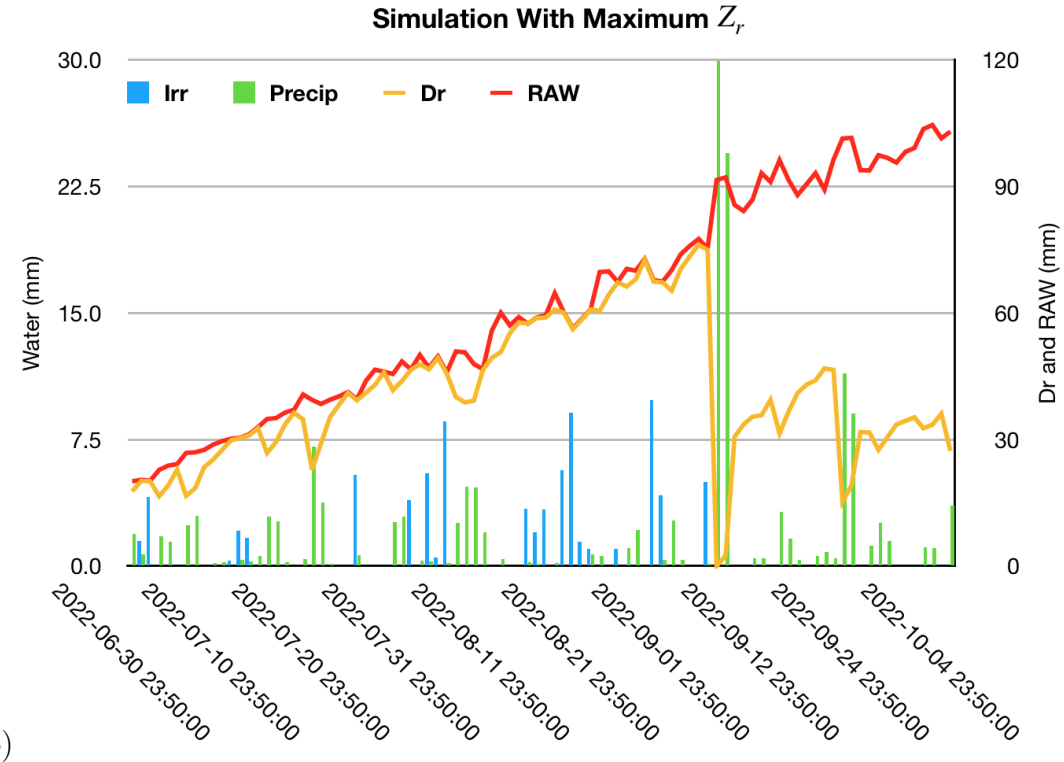
4.2.4 f_d

Depletion fraction (f_d) is another crop parameter related to the soil water balance. Depletion fraction has an approximately linear effect on yield and irrigation. Similarly to Z_r, as f_d is increased, irrigation and yield decrease. However, despite similarity in the trends, the model's sensitivity to f_d is noticeably smaller than its sensitivity to Z_r. The relatively similar scaling of yield and irrigation between f_d and Z_r is intuitive because f_d factors into the calculation in a similar way to Z_r, affecting the size of the soil water reservoir accessible to a plant.

Figure 4-6 shows trajectories generated from the smallest and largest values of depletion fraction used to generate Figure 4-2d. The general trends observed in Figure 4-6 are similar to that of the model behavior under varied Z_r, however the model is less sensitive to f_d.

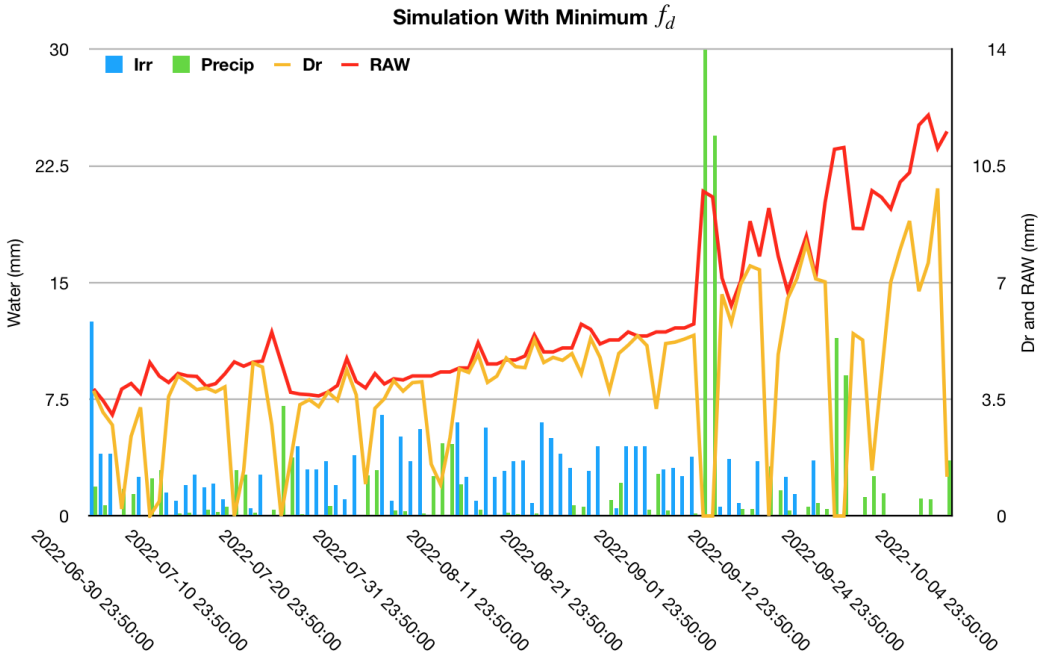


(a)

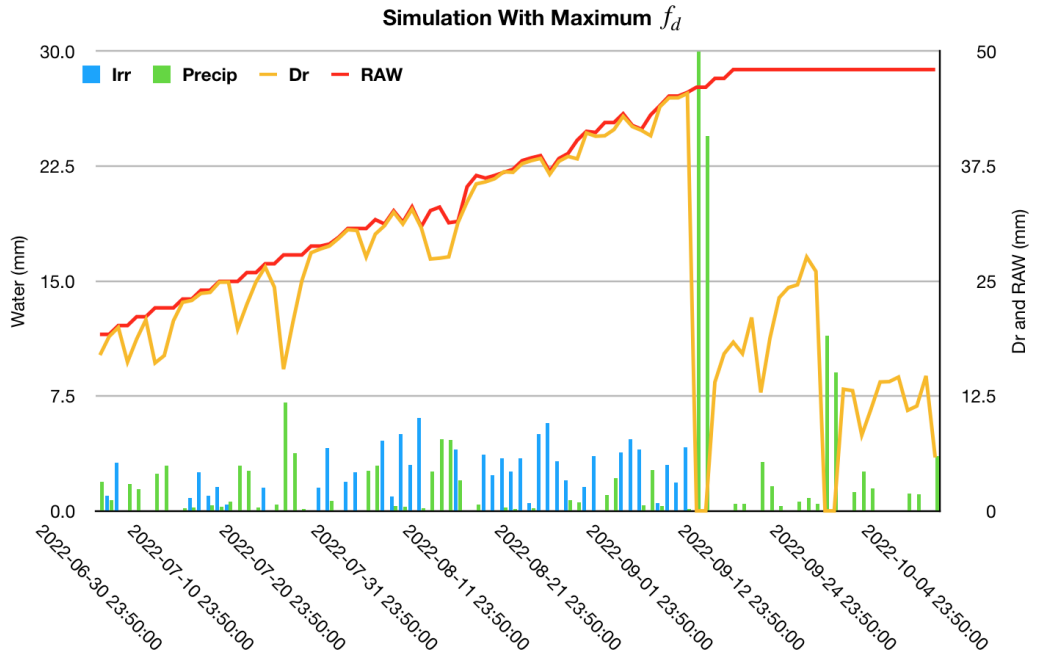


(b)

Figure 4-5: Simulation results for a growing season using maximum and minimum Z_r values. Irrigation and precipitation are on the left axis, D_r and RAW are on the right axis.



(a)



(b)

Figure 4-6: Simulation results for a growing season using maximum and minimum depletion fraction (f_d) values. Irrigation and precipitation are on the left axis, D_r and RAW are on the right axis.

Chapter 5

Conclusion

5.1 Limitations of the Work

This work utilized results from AquaCrop-OS, but initial simulations of controller performance from [15] were done in AquaCrop GUI. The results for the same cases run in AquaCrop-OS and AquaCrop GUI should ideally be the same, but they were not. AquaCrop-OS consistently predicted higher theoretical maximum yields. This can be seen in Figure 4-2 where the theoretical maximum yields from AquaCrop-OS and AquaCrop GUI are shown as horizontal dashed lines. The analysis in this thesis focuses exclusively on the relative change in yield as parameters are changed with very little focus on absolute yield or irrigation amounts – although the absolute amounts are expected to still be close to the true amount (the reported yield values between AquaCrop-OS and AquaCrop GUI differed by only 2.6%). Another aspect of this model disagreement is that, for the default values of each parameters, AquaCrop-OS showed some lost yield while AquaCrop GUI reported almost zero lost yield. The cost function inside the POWEIr Controller is designed to keep yield high as a top priority, and in some of the cases run, the irrigation values are very large (Figure 4-2). For irrigation values this large, it is unexpected that yield should be lost. Therefore, it is likely that the AquaCrop-OS model is not perfectly calibrated and I expect the AquaCrop GUI results to be more accurate. Through future work, the models can be made to agree better.

5.2 Recommendations Based on the Sensitivity Results

As described in Section 4.1, the ranges used in the parameter variation experiment represent the ranges of each parameter in common non-tree crops. As such, we compare the sensitivity of the controller to these different parameters with an understanding that this is the full range of each parameter expected in practice. With this in mind, the K_c parameter has the most significant effect on yield and irrigation with irrigation in particular going from zero to its maximum value during the variation of this parameter.

The change in yield over the range of K_c values tested was 1.8 times greater than the second most sensitive parameter (which is rooting depth, Z_r). K_c was also the most sensitive parameter for irrigation, with Z_r again being the second most sensitive. The range of irrigation values for the tested parameters was 3.8 times greater for K_c than that of the second highest parameter Z_r. On the other hand, K_y had the smallest overall effect on yield. The change in yield over its full range was 33 times smaller than the second least sensitive parameter (which was f_d). For irrigation, K_y also had the smallest overall effect with f_d again being second smallest. K_y's effect on irrigation was 146 times smaller than the second least sensitive parameter f_d. Z_r and f_d fall squarely between the other two parameters in terms of sensitivity. The total variation across the full range of Z_r and f_d variation was 0.3251 ton/ha and 0.1772 tons/ha for yield and 60 mm and 73 mm for irrigation.

The gray vertical dashed lines in the four subfigures of Figure 4-2 communicate the location of the default values of the respective parameters for Tomato (the crop simulated here). These default values show a loss in yield which is not recapitulated in the AquaCrop GUI model. Based on the amount of water used in some of the high irrigation cases run here which still show lost yield, it is unlikely that this aspect of the AquaCrop-OS simulation is fully calibrated, however, the overall scaling of the system with the different model parameters is clear. It may be worth understanding the distribution of default values for each crop to understand how conservatively the

default values should be set to avoid loss of yield. The single values provided in Tables 4.1, 4.2, and 4.3 do not allow us to completely understand the distribution of parameters for the same crop which is important in determining risk. Some values are reported as ranges which is helpful, however, the likelihood of existing in some part of the range is also an important factor.

The parameter sensitivity results indicate an intuitive order of priority for model calibration. First calibrate K_c (discussed further in the future work section), then consider calibrating Z_r and f_d – likely calibrating the cheaper or more convenient parameter first. For Z_r and f_d the effect on yield and irrigation is approximately linear, especially around the default values. Therefore, it can be understood that if instead of calibrating these parameters, a default value is used, the expected resulting under (or over) irrigation and corresponding gained/lost yield will be approximately linear in the parameter error. In future work, calibrated crop parameter values for each crop on many farms could be aggregated and used to create some probabilistic bounds on the expected yield and irrigation error associated with using default values from the literature. In rare situations where water use has a high dollar cost, K_y may become important and could be worth calibrating for this reason. However, in many situations, it need not be calibrated at all.

This work illustrates the POWEIr Controller's sensitivity to its internal model's crop parameters which may be useful for determining the procedure for calibrating the POWEIr Controller at new farms. It may be especially useful when resources are limited and not all of the parameters can be calibrated, hopefully contributing to lower cost and wider accessibility of the controller.

5.3 Future Work

As discussed in Section 3, model parameters can make model calibration challenging making simple models attractive even if the system is not computationally limited. Section 4 showed that controllers can sometimes help reduce that complexity. However, calibration will always be an essential part of applying a model or deploying a

controller.

The main result of this thesis is that the POWEIr controller is most sensitive to K_c. K_c is unfortunately also a historically expensive parameter to calibrate because calibrating it requires measuring the water balance which is typically done with a lysimeter (typically a very large bucket of soil with various sensors). Cheaper lysimeters have been developed including in [9], however, these still cost $1,700 each. Cheaper still, homemade lysimeters can be fashioned using five gallon buckets. Work done to understand the error vs. cost trade-offs associated with these progressively more accessible methods of K_c calibration paired with the sensitivity analysis in this thesis could provide information about a "sweet spot" in lysimeter accuracy and cost.

Due to the importance of model calibration, future work related to better understanding model error and sensitivity (for example, sensitivity to soil parameters) could be very useful as well as work related to cheaper and easier ways of calibrating model parameters. In the case of cheap and accessible irrigation controllers, it is possible that satellite image analysis or other "high-tech" methods of crop or soil parameter estimation could be productive to peruse. However, equally useful would be methods that farmers can do with little to no extra technology. Methods that use the tools and materials they have at their disposal. After all, a smartphone is a powerful measuring and information processing instrument. Precision agriculture is a field replete with innovations and with a slight change in perspective, it may be possible to make these innovations widely accessible.

Bibliography

[1] Water resources data system. chapter 3: Evapotranspiration measurements. Website. http://library.wrds.uwyo.edu/wrp/87-06/ch-03.html.

[2] R. G. Allen, L. S. Pereira, D. Raes, and M. Smith. Crop evapotranspiration: Guidlines for computing crop water requirements. *FAO Irrigation and Drainage Paper No. 56*, 1998.

[3] Terán-Chaves CA, García-Prats A, and Polo-Murcia SM. Calibration and validation of the fao aquacrop water productivity model for perennial ryegrass (lolium perenne l.). *Water*, 2022.

[4] Jared Di Carlo, Patrick Wensing, Benjamin M. Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. *IEEE International Conference on Intelligent Robots and Systems*, 2018.

[5] Willem Coudron, Pieter De Frenne, Kris Verheyen, Anne Gobin, Charlotte Boeckaert, Tim De Cuypere, Peter Lootens, Sabien Pollet, and Tom De Swaef. Usefulness of cultivar-level calibration of aquacrop for vegetables depends on the crop and data availability. *Frontiers in Plant Science*, 2023.

[6] J Doorenbos and A H Kassam. Yield response to water. *FAO Irrigation and Drainage Paper No. 33*, 1979.

[7] P. Steduto et al. Concepts and applications of aquacrop: The fao crop water productivity model. *Crop Modeling and Decision Support*, 2009.

[8] FAO, https://github.com/aquacropos/aquacrop/. *AquaCrop OSPy Notebook 1.*

[9] Daniel K. Fisher. Simple and inexpensive lysimeters for monitoring reference and crop-et. *USDA Agricultural Research Service Application and Production Technology Research Unit*, 2004.

[10] D. Hillel and C. Rosenzweig. *Encyclopedia of Soils in the Environment*. Elsevier, 2005.

[11] N. T. Krell, S. A. Giroux, Z. Guido, S. E. Lopus C. Hannah, K. K. Caylor, and T. P. Evans. Smallholder farmers' use of mobile phone services in central kenya. *Climate and Development*, 2021.

[12] Jingqi Li, Yaosong Long, Mao Su, Lei Liu, Bo Wang, and Zhongtao Cheng. Fault-tolerant guidance of rocket vertical landing phase based on mpc framework. *International Journal of Aerospace Engineering*, 2022.

[13] William Oberle. Monte carlo simulations: Number of iterations and accuracy. *US Army Research Laboratory*, 2015.

[14] Dirk Raes, Pasquale Steduto, Theodore C. Hsiao, and Elias Fereres. Chapter 3: Calculation procedures. *AquaCrop Version 7.1 Reference manual*, 2023.

[15] Carolyn Sheline, Fiona Grant, Simone Gelmini, and Amos G. Winter. Designing a predictive optimal water and energy irrigation (poweir) controller for pv-powered drip irrigation systems in resource-constrained contexts. *In Progress*, 2023.

[16] Carolyn Sheline, Fiona Grant, Simone Gelmini, and Amos G. Winter. Designing a predictive optimal water and energy irrigation (poweir) controller for pv-powered drip irrigation systems in resource-constrained contexts. *In Progress*, 2023.

[17] Pasquale Steduto, Theodore C. Hsiao, , Elias Fereres, and Dirk Raes. Crop yield response to water. *FAO Irrigation and Drainage Paper 66*, 2012.

[18] Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. Osqp: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 2020.

[19] Russ Tedrake. *Underactuated Robotics*. 2023.

[20] S. Vidano, C. Novara, M. Pagone, and J. Grzymisch. The lisa dfacs: Model predictive control design for the test mass release phase. *Acta Astronautica*, 2022.