INVESTIGATION OF AN ENERGY EFFICIENT PUMP SPEED CONTROL ALGORITHM FOR CONTROLLING SUMP LEVEL

by

Josh Dubey

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the Master of Science in Electrical and Computer Engineering

Fall 2018

© 2018 Josh Dubey All Rights Reserved

INVESTIGATION OF AN ENERGY EFFICIENT PUMP SPEED CONTROL ALGORITHM FOR CONTROLLING SUMP LEVEL

by

Josh Dubey

Approved:

Keith Goossen, Ph.D. Professor in charge of thesis on behalf of the Advisory Committee

Approved:

Kenneth E. Barner, Ph.D. Chair of the Department of Electrical and Computer Engineering

Approved:

Levi T. Thompson, Ph.D. Dean of the College of Engineering

Approved:

Douglas J. Doren, Ph.D. Interim Vice Provost for the Office of Graduate and Professional Education

ACKNOWLEDGMENTS

I wish to thank my parents, friends and my advisor for the continued support necessary to complete my thesis.

TABLE OF CONTENTS

LIST	OF TA	ABLES	v	
LIST	OF FI	GURES	vi	
AB21	KAU	1V1	.11	
Chapt	er			
1	INT	RODUCTION	1	
2	THE	CORY	4	
	2.1 2.2	Background on Pumping Variable Level Control	4 7	
3	EXPERIMENTAL SETUP			
	3.1	Description of Pilot Scale Design	9	
		 3.1.1 Control Hardware and Software	9 5 6	
	3.2 3.3	Proposed Algorithm	9 22	
4	RES	ULTS AND DISCUSSION2	3	
5	CON	ICLUSION	9	
REFE	RENG	CES	0	
Apper	ndix			
	MA	TLAB CODE	1	

LIST OF TABLES

Table 4.1	Experimental SE values and % Savings	24
Table 4.2	Experimental Process Parameter Variables	

LIST OF FIGURES

Figure 2.1	An arbitrary pump system with three system curves, S1, S2 and S3. Isoefficiency lines are plotted as well. Adapted from 'Variable speed wastewater pumping', 2013, White Paper, 7
Figure 2.2	Specific energy as a function of frequency for the system curves from Figure 1. Adapted from 'Variable speed wastewater pumping', 2013, White Paper, 7
Figure 2.3	Pump speed as a function of sump level for linear and concave up control relationships
Figure 3.1	Control hardware for the pump station shown. From left to right, the Micrologix PLC, the VFD, and the two power supplies for field devices. 11
Figure 3.2	The main program file of the pump station control logic in RSLogix Micro Starter
Figure 3.3	A section of the ladder logic for the nonlinear control algorithm. The exact algorithm is programmed in the block called PUMP SPEED ALGORITHM OUTPUT
Figure 3.4	A section of the ladder logic for PID control of the pump. This logic is executed mostly to return the sump level to a certain value following an experimental test run
Figure 3.5	The HMI for the pump station. A visualization of the process with real time values is shown on the left, while a historical data trend of process variables is shown graphically on the right
Figure 3.6	(a) Simulated variable speed pump curves for the experiment, with the system curve developed for Level=50.8 cm (20 in.) and (b) Pump efficiencies at variable speeds
Figure 3.7	The pilot scale pump station
Figure 3.8	Experimental concave up speed control curves. "A" is the curvature value. The sump tank level is shown here in centimeters

Figure 3.9	Sump inlet flow rate (liters/min) as a function of time for one experimental cycle. This exact flow regime is repeated for all cycles22
Figure 4.1	(a) Specific Energy and (b) Specific Energy Savings plotted for experimental data. The markers are averages of each of the three test cycles for each value of curvature. A second order fit of the savings points is also plotted to show the experimental correlation
Figure 4.2	(a) Average Level, (b) Speed and (c) Maximum Speed collected from data for each curvature value. Each marker represents the average of three test cycles for the corresponding curvature value
Figure 4.3	(a) Pump Speed, (b) Sump Level and (c) Inlet Flow Rate collected over the cycle time of 1800 seconds for each curvature value. Each line represents the average of three test cycles at that particular instance
Figure 4.4	Active Power plotted for linear and curvature values of 7.5 and 1527

ABSTRACT

This thesis explores a new nonlinear control algorithm for variable speed centrifugal pumps at water/wastewater pump stations that leads to specific energy savings over the conventional linear one. The algorithm is useful for facilities where pump speed is a linear function of liquid level in order to transport fluid and smooth inflow peaks. A nonlinearity in the form of a quadratic term is added to the conventional linear one in order to produce efficiency gains, with a single parameter, curvature, varied to optimize energy savings. Results obtained by implementing the new algorithm on a pilot-scale pump station show significant energy savings for fixed pump flow, with a parabolic correlation of specific energy savings versus curvature of the nonlinear quadratic determined. In addition, the cost of implementing this algorithm is minimal to none, so the work presented has significant industrial potential.

Chapter 1

INTRODUCTION

Growing awareness of issues such as sustainable energy use and energy efficiency have prompted interest in energy saving research in various industrial applications, including pumping systems. Particularly, *Goldstein and Smith* [1] reported that nearly 4% of United States electricity is consumed by wastewater treatment plants and almost 80% of the electricity in the wastewater treatment process is used by pumps. Based on the large proportion of energy used for pumping, and the vast number of wastewater plants across the country, just a small reduction in energy use can lead to large savings. Here, we report a new control algorithm for sump pumping systems, for example in wastewater treatment plants that results in up to 4 % savings compared to the currently used control algorithms.

In general, savings in pump energy can be achieved in two main ways. One is by designing more efficient pumps. Another, discussed and presented here, is improving pump performance with effective control strategies. The latter often times involves employing speed control of centrifugal pumps with frequency regulated by Variable Speed Drives (VFD). By regulating pump speed according to process parameters such as tank level, pump power can be reduced significantly compared to constant-speed control. However, the precise speed control algorithm can result in additional savings. Roughly, there are two algorithmic approaches. One is to attempt to maintain speed at the best-efficiency point of the pump. *Bakman, Geverkov and Vodovozov* [2] discussed a method for single and multi-pump predictive control to

1

maintain operation in the best-efficiency region. *Tang and Zhang* [3] considered a model predictive control approach to improve operational efficiency incorporating variables such as TOU tariff and water demand. Zhang, Zhen and *Kusiak* [4] developed a scheduling model to generate energy optimal operational schedules for wastewater pump systems. As discussed in [2-4], most of the pump control research has focused on predictive control strategies, with emphasis on scheduling variable speed centrifugal pump runs based on modelling of future inlet flow rates. While these methods have potential to bring about efficiency gains and better operation, they are often costly and require large investments in existing pump control systems. Therefore, a simpler more cost-effective control strategy that results in energy savings is highly desirable.

Wastewater treatment plants as well as other applications usually have an input sump for collecting inlet flows. Since the inlet flows are highly variable, it is essential to have an effective control strategy that will adjust outflow. For sump stations that employ centrifugal outflow pumps on variable speed drives, two types of control strategies exist. Constant sump level control, a form of closed loop Proportional-Integral (PI) control where pump frequency is adjusted to maintain the tank level at a desired set point, is one method. Variable level control, a soft control strategy where pump frequency is a linear proportional function of sump level, is the second method, and has the advantage of smoothing large inflow peaks compared to constant level control. In the latter strategy, pump speed increases as inlet flow increases and level rises, with no actual level set point and error variable, until a new equilibrium level is achieved. All reported algorithms for speed-level control show pump speed increasing linearly with level. In this paper, a nonlinear, quadratic term is added to the speed

2

versus level function and is explored experimentally to show lower energy use compared to the linear-only function. Particularly, using a quadratic negativecurvature function, a 4 % reduction in energy is found for a particular curvature. While the exact curvature for a particular sump pumping system that minimizes energy may depend upon the particular pumping system, we demonstrate here that adding the quadratic into the algorithm can result in significant energy savings, and potentially reduce overall United States electricity consumption on the order of a tenth of a percent.

Chapter 2

THEORY

2.1 Background on Pumping

Over the last 10-15 years, there has been a large increase in the number of municipalities adapting variable frequency drives (VFDs) to their pump stations. Advantages of variable speed operation include the potential of decreased energy use, more flexibility and the ability to soft start pump motors to extend lifetime. While not all pump stations are necessarily fit for variable speed operation, particularly those who are solely lift stations with high static head, many accrue considerable benefit from installing adopting these drives. Energy savings for centrifugal pumps on variable drives are a direct result of the pump affinity laws. Equations (1) and (2) shows the relationships between pump speed N_1 and N_2 , discharge flow Q_1 and Q_2 , and pump power consumption P_1 and P_2 .

$$\frac{Q_1}{Q_2} = \left(\frac{N_1}{N_2}\right) \tag{1.1}$$

$$\frac{P_1}{P_2} = \left(\frac{N_1}{N_2}\right)^3 \tag{1.2}$$

Based on the affinity laws, just a small reduction in pump speed can result in a much larger reduction in pump power consumption. However, for municipalities determining whether to incorporate variable speed operation, simply looking at power consumption does not show the complete picture regarding energy savings and potential payback. The most useful variable, and one that will be referred to in this paper, is specific energy SE written as:

$$SE = \frac{E}{V} \tag{1.3}$$

where E is the unit energy and V is the unit volume. Specific energy takes into account the amount of discharge flow for a unit of energy consumed, and therefore serves as the preferred statistic for comparing energy savings for different control algorithms on a given pumping system. Figure 2.1 from Xylem Corporation [5] shows an example of three different system curves operated by a single pump. Since the static and dynamic head components are different for each system, the specific energy curves shown in Figure 2.2 are dramatically different.



Figure 2.1 An arbitrary pump system with three system curves, S1, S2 and S3. Isoefficiency lines are plotted as well. Adapted from 'Variable speed wastewater pumping', 2013, White Paper, 7.



Figure 2.2 Specific energy as a function of frequency for the system curves from Figure 1. Adapted from 'Variable speed wastewater pumping', 2013, White Paper, 7.

In retrospect, S3 brings the most benefit to incorporating variable speed operation, since the system curve roughly follows the path of the isoefficiency lines. This particular aspect of the system is important in developing pump control algorithms that bring about efficiency gains. Often times, there is an optimum speed that minimizes specific energy, and operating as close as physically possible to the point should be the focus of a particular algorithm.

2.2 Variable Level Control

For pump stations employing the technique known as variable level control, pump speed is a function of sump level where the speed increases proportionally with level. The pump speed will fluctuate with tank level until a temporary equilibrium point is reached where the inlet flow matches the outlet flow from the sump. Thus far, only algorithms where speed varies linearly with level have been in use, however exploring adding a nonlinearity can potentially result in specific energy reductions for pump systems. Especially for systems like S3 in Figure 2.2, it is possible that developing an algorithm that results in a higher average sump level, and lower average pump speed for a given operational period can bring about efficiency gains.

Compared to an algorithm where pump speed is a linear functional relationship of sump level, a curved, concave up relationship one may result in a reduction in specific energy due to operation at a higher average sump level and lower average pump speed. Figure 3 shows how a potential concave up control function compares to a linear one where pump speed is a function of level.



Figure 2.3 Pump speed as a function of sump level for linear and concave up control relationships.

Experimentally, the curvature of the quadratic term can be altered to find an optimum one. If the curvature reaches a certain value, it is possible the pump speed will actually increase to an excessively high value for a prolonged period due to the lower initial reaction to increasing sump level from an inlet flow spike. The exact optimum curvature can be determined for a particular pump system by experimentation. It should also be noted that if such a control method were to result in savings, the exact algorithm could be programmed directly into an existing variable level control program with minimal implementation cost due to its simplicity.

Chapter 3

EXPERIMENTAL SETUP

Testing of the proposed control algorithm was done experimentally on a pilot scale pump station. Data was collected and analyzed to determine if there were specific energy reductions. The exact methods used for developing the nonlinear control algorithms are highlighted in this section.

3.1 Description of Pilot Scale Design

A pilot scale pump station is designed for experimenting with different algorithms. A Xylem 0.75HP centrifugal pump is used to draw room temperature water from a 58 centimeter (20 inch) sump tank that resides 0.61 meters (2 feet) below the elevation of the pump. The pump frequency is controlled by a 1HP TECO-Westinghouse Variable Speed Drive (VFD), which receives both run/stop and frequency command via a 0-10V analog signal from an Allen Bradley Micrologix Programmable Logic Controller (PLC). Water is pumped over a total distance of 10 feet from the sump to a 189 liter (50 gallon) tank that is located 9.5 feet above ground level, which comprises the static head in the system.

3.1.1 Control Hardware and Software

The Micrologix PLC is used for controlling all operations of the pilot scale pump station. The particular version used, the Micrologix 1400, has both analog and digital inputs and outputs, enabling both continuous and discrete on-off control. This particular series can receive 120 Volt AC digital signals, and has in excess of 16 dry contact outputs that can send relay signals to field devices and motors, including the run-stop command for the VFD. The power supplied to the controller is only sufficient for powering the processer, so all field devices must be powered externally. For this reason, a 24V DC external power supply is used for powering all analog and digital transmitters and control valves. The VFD receives a digital input as a run/stop command and a 0-10V analog signal for the motor frequency command. It also sends a 0-10V analog signal back to the PLC for a frequency feedback. The analog devices used in the pump station include the VFD, a motorized control valve for modulating flow to the sump tank, two flow sensors for measuring the input flow rate to the sump, as well as a level transmitter, which records sump level and sends a voltage signal to the PLC. Several digital devices are used, including two motorized ball valves, a solenoid valve, and a momentary push button that is programmed to change the operating mode of the station when pushed.



Figure 3.1 Control hardware for the pump station shown. From left to right, the Micrologix PLC, the VFD, and the two power supplies for field devices.

The PLC is programmed with software called RSLogix Micro, which is specifically designed for Allen Bradley's line of Micrologix controllers. The communication between the PC and PLC for programming purposes is set over Ethernet IP to enable fast movement. In addition, a serial communication protocol, Modbus RTU, is used to read and send process data to a historian and process interface in the PC from memory addresses in the PLC. This protocol is setup over an RS-232 point-point connection.

The control logic is programmed and executed using ladder diagrams, where inputs and outputs are bridged together by actions taken by the CPU of the controller. This is how most municipal wastewater treatment plants are controlled, where multiple PLCs execute control logic across the entire process. In Figure 4, the main page of the ladder program is shown:



Figure 3.2 The main program file of the pump station control logic in RSLogix Micro Starter.

The ladder logic is executed from top to bottom at a particular scan rate set by the user. The main routine (shown in Figure 3.2) is executed first, with subroutines executed thereafter. For this setup, subroutines are created specifically for controlling the pump speed based on sump level based on the experimental algorithms, for controlling flow rate to the sump, and for controlling the pump via a typical Proportional-Integral-Derivative (PID) control loop for experimental purposes. Figure 5 shows part of the logic used to execute the experimental, nonlinear control algorithm.



Figure 3.3 A section of the ladder logic for the nonlinear control algorithm. The exact algorithm is programmed in the block called PUMP SPEED ALGORITHM OUTPUT.

Figure 3.4 shows part of the PID logic that is mostly used for returning the

tank level to a certain value following an experimental test of the nonlinear algorithm.



Figure 3.4 A section of the ladder logic for PID control of the pump. This logic is executed mostly to return the sump level to a certain value following an experimental test run.

The version of programming software works by using memory addresses in the controller to assign values and execute logic. The user must assign the type of variable, such as an integer, a bit or a floating-point value. For example, an address shown in Figure 3.4, N7:0, is the value of the sump level stored as an integer value. The analog values have 12-bit resolution, so N7:0 can range anywhere from 0-4096. For the pump frequency signal coming to the PLC, the value of frequency between 0 and 60 Hz, is scaled to a value between 0-4096.

3.1.2 Human Machine Interface

For any process controlled using a device like a PLC, proper operation requires an interface to visualize the process. For the pilot-scale pump station, an HMI software called Winlog Lite is programmed to provide a live visual representation of the process and to read and record process variables such as level, pump speed, and sump inlet flow for data analysis. The HMI not only provides an eye into the process, but also provides capabilities to the plant personnel to issue control commands such as energizing valves and motors. Figure 7 shows the interface for the pump station:



Figure 3.5 The HMI for the pump station. A visualization of the process with real time values is shown on the left, while a historical data trend of process variables is shown graphically on the right

The process variable data trends shown in Figure 3.5 are a result of a Modbus RTU communication between the HMI software and the controller. Commands are issued from the software to read particular analog points that correspond to the correct process variables and those values are recorded and displayed in the HMI continuously. The HMI in this case is known as the Modbus master, while the PLC is the Modbus slave, only capable of receiving commands from the HMI.

3.1.3 System Characteristics

Fittings in the pipe system include a full-bore manual ball valve, a swing check valve, a globe valve on the discharge side used for introducing frictional resistance into the system, as well as several 90-degree elbows. An integral flow meter is also located on the discharge side, which records the total flow produced by the pump for a given experimental cycle.

A simulated pump/system curve was created using AFT Fathom, a steady-state fluid mechanics software used for modelling incompressible flow. The curve is developed with the pump manufacture's data and interpolation at variable speeds, with the system curve created for a sump tank level of 58 centimeters (20 inches). Based on the efficiency curves, it is found that all operating points from 30-60 Hz lie within \pm 10% of the Best Efficiency Point (BEP) at a given speed. This figure is shown below:



Figure 3.6 (a) Simulated variable speed pump curves for the experiment, with the system curve developed for Level=50.8 cm (20 in.) and (b) Pump efficiencies at variable speeds

Inlet flow to the sump comes from the 189 liter (50 gallon) tank, as well as a 95 liter (25 gallon) tank that allows for simulating peak flow rates. Flow rate is modulated with two on-off valves, as well as a proportional analog control valve that also receives a 0-10V analog signal from the PLC. A picture of the pilot scale station is shown below in Figure 3.7.



Figure 3.7 The pilot-scale pump station

The PLC programming software was used to control all process parameters, including pump control algorithms and sump inlet flow rate. Process data was collected via serial communication and sent to a Human Machine Interface (HMI) for viewing and analysis. Pump power consumption was recorded through a plug load logger at an interval of one second and is also collected via data collection software. At the end of every flow cycle, the sump tank level is returned to a predetermined value, and the next cycle commences.

3.2 Proposed Algorithm

The nonlinear speed versus level control algorithm experimented with is a quadratic function that is concave up and varies in curvature only. Figure 3.8 shows the algorithm where speed is a function of level with varied curvature (A value) that is tested experimentally.



Figure 3.8 Experimental concave up speed control curves. "A" is the curvature value. The sump tank level is shown here in centimeters

The linear functional relationship (A=0) is shown first, with increasing A values listed below consecutively. For this experimental setup, since the sump tank maximum height is 50.8 centimeters (20 inches) and the maximum pump speed is 60Hz, the following linear speed versus level control function is created:

$$LS = 1.2 * L$$
 (3.1)

where LS is equal to linear speed and L is the sump tank level in centimeters.

The curves are created by specifying the desired endpoints, and then adding them to the linear control function. The defining characteristic of each quadratic, which is the curvature, is defined as the A value, which is listed as experimental constants in Equation (3.2):

$$A = [0\ 2.5\ 7.5\ 10\ 12.5\ 15] \tag{3.2}$$

$$B = \frac{A}{\left(\frac{L1}{2} - \frac{L2}{2}\right)^2} \tag{3.3}$$

where B is equal to a constant, and L_1 and L_2 are the points where the curve will intersect the x axis, and where the speed curve will intersect the linear speed versus level function.

The equation of the curve C is based on the values of A and B and is shown in Equation (3.4):

$$C = -A + B^{*} \left(L - \left(\frac{L_{1} + L_{2}}{2} \right) \right)^{2}$$
(3.4)

To get the actual speed versus level control curve, C is added to the linear speed LS, to come up with the actual speed curve S, shown in Equation (7).

$$S = 1.2 * L - A + B * \left(L - \left(\frac{L_1 + L_2}{2} \right) \right)^2$$
(3.5)

.

The values of L1 and L2 can be chosen based on the physical constraints of any given setup. For the purpose of this experiment, they are chosen as 27.9 cm (11 inches) and 48.3 cm (19 inches) respectively. For A values greater than 5, there exist points where the derivative of the curve is negative and the speed actually decreases with increasing level. These points are rejected, and the last known value with a nonnegative derivative is held indefinitely. Since the level endpoint L1 is 27.9 cm, if the level falls below this value, the calculated pump speed will be held indefinitely with no further decrease in speed. No pump speed values less than 27.5Hz are considered in the experiment; in wastewater plants, a minimum speed is generally established to protect against clogging from solid materials in the sump.

The algorithm itself is created and then implemented using PLC ladder logic programming, where the liquid level measurement is the input and the pump speed is the output. The design of the control algorithm consists of specifying the level endpoints, L1 and L2, and choosing an A value that maximizes the system efficiency based on the characteristics of the individual pumping system. Essentially, the value of A can be tuned in order to maximize system efficiency. Since there is no actual level set-point, there is no error input to the algorithm and therefore no additional design parameters are considered.

3.3 Test Flow Regime

The experiments consist of a test flow regime over a period of 1800 seconds, where the inlet flow to the sump is controlled to meet a dynamic flow set point. This flow regime is repeated for each test cycle. Figure 6 shows flow data collected during one of the cycles to show what the flow regime looks like over an 1800 second period. Noise looking features in the data are a result of oscillatory behavior of the analog control valve.



Figure 3.9 Sump inlet flow rate (liters/min) as a function of time for one experimental cycle. This exact flow regime is repeated for all cycles.

Chapter 4

RESULTS AND DISCUSSION

Each pump run of a single curvature consisted of three cycles, where the tank level was reset to a predetermined value between cycles. Figure 4.1 shows the specific energy use of each particular curvature, denoted by "A" value, with A=0 representing the linear only relationship.



Figure 4.1 (a) Specific Energy and (b) Specific Energy Savings plotted for experimental data. The markers are averages of each of the three test cycles for each value of curvature. A second order fit of the savings points is also plotted to show the experimental correlation

Davamatar	A Value							
rarameter	0	2.5	5.0	7.5	10.0	12.5	15.0	
Specific	157.1	153.5	152.0	150.8	152.0	153.8	157.7	
Energy	± 0.5	±0.4	±0.3	±0.7	±0.3	± 0.1	±0.3	
(Wh/m^3)								
% Savings	-	2.32 ±0.2	3.33 ±0.2	4.08 ±0.4	3.37 ±0.1	2.12 ±0.1	-0.35 ±0.2	

Table 4.1: Experimental SE Values and % Savings

The specific energy roughly follows a second order parabolic relationship with a minimum at A=7.5. Using an A value of 15 actually leads to an increase in specific energy compared to linear control, indicating that there is an optimum curvature to the algorithm somewhere between A=0 and 15.

In addition, process parameters including average sump level, average pump speed, as well as maximum pump speed are collected and plotted for each A value.



Figure 4.2 (a) Average Level, (b) Speed and (c) Maximum Speed collected from data for each curvature value. Each marker represents the average of three test cycles for the corresponding curvature value.

As shown, while average speed is lower for A=15 than A=0, the maximum speed is substantially higher. The amount of time spent operating at speeds in excess of 50 Hz is highlighted in Figure 4.3.



Figure 4.3 (a) Pump Speed, (b) Sump Level and (c) Inlet Flow Rate collected over the cycle time of 1800 seconds for each curvature value. Each line represents the average of three test cycles at that particular instance.

It is apparent from the speed versus time results that as the A value increases, the absolute maximum recorded speed increases. However, for A values greater than 7.5, the duration of operation at high speeds increases substantially. For A=15, when flow rate reaches one of its four peaks, the speed increases to above 50Hz for a total of 178 seconds on average, substantially more than the 23 seconds for A=7.5. Based on the affinity laws, since pump power increases cubically with speed, the energy use increases drastically during periods of high flow due to the pump operating at really high speeds for an extended duration. At the minimum specific energy value of A=7.5, while the absolute maximum speed (51.9 Hz) was higher than that for A=0 (49.9 Hz), the duration of operation at the maximum speed was comparatively small, and the reduced speed during times of lower inlet flow resulted in significant specific energy savings compared to the linear control. The total operational time at speeds of greater than 45Hz was 248 seconds for A=0, 196 seconds for A=7.5, and 313 seconds for A=15. The same trends can be seen in Figure 11, where active power consumption is monitored over time.



Figure 4.4 Active Power plotted for linear and curvature values of 7.5 and 15

Daramatar	A Value							
Farameter	0	2.5	5.0	7.5	10.0	12.5	15.0	
Avg. Taply	32.73±	$33.73 \pm$	35.30	36.81	38.31	$39.98 \pm$	41.24	
Avg. 1 ank	0.03	0.09	± 0.04	± 0.13	± 0.05	0.03	± 0.05	
Level (cm)								
Avg. In. Flow	24.19	24.18	24.05	23.95	23.77	23.65	23.48	
Rate	± 0.07	±0.02	±0.03	±0.01	±0.01	±0.05	± 0.04	
(lpm)								
Avg. Pump	38.36	38.12	37.78	37.26	36.41	36.79	36.83	
Speed	± 0.06	± 0.09	±0.04	± 0.07	±0.06	± 0.07	± 0.09	
(Hz)								
Max. Pump	49.85	10.60	50.20	51.00	52.80	57 57	60.00	
Speed	± 0.07	± 0.00	± 0.20	± 0.08	± 0.07	+0.08	± 0.00	
(Hz)		± 0.10	± 0.03	± 0.08	±0.07	±0.08	± 0.10	
Total Pumped	0.8121	0 807 +	0.801	0.794	0.787	0778 +	0 7603	
Flow	±	0.007 ± 0.002	±	±	±	0.0006	$\perp 0.7093$	
(m^3/cycle)	0.0006	0.005	0.001	0.001	0.001		± 0.002	

 Table 4.2:
 Experimental Process Parameter Values

While active power consumption for A=15 remains less than that of A=7.5 and A=0 for more than 80% of the cycle duration, the power reaches values in excess of 500 W during the inflow peaks, which actually consumes enough to end up with more energy drawn during the whole cycle compared to the A=0 algorithm. For A=7.5, the peak power actually is only slightly more than that of A=0 and integrated over the whole 1800 seconds and divided by total flow produced, leads to a substantially lower specific energy compared to the A=0, on the order of 4% less.

Chapter 5

CONCLUSION

This paper has examined an energy efficient control algorithm for pump stations tasked with controlled sump level, and has presented experimental results showing specific energy reduction in excess of 4% compared with conventional, linear variable level control. While the exact amount of savings will depend on specific pump station parameters, the experimental data shows that there is savings from varying pump speed nonlinearly with level, and that there is an optimum concave up curve that produces the most reduction in specific energy.

REFERENCES

- Goldstein, R., and Smith, W. (2002). "Water & sustainability (Vol. 4): U.S. electricity consumption for water supply & treatment—The next half century." Technical Rep., Electric Power Research Institute (EPRI), Palo Alto, CA.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
- [2] I. Bakman, L. Gevorkov and V. Vodovozov, "Predictive control of a variablespeed multi-pump motor drive," 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE), Istanbul, 2014, pp. 1409-1414.K. Elissa, "Title of paper if known," unpublished.
- [3] Y. Tang and S. Zhang, "A Model Predictive Control Approach to Operational Efficiency of Intake Pump Stations," 2010 International Conference on Electrical and Control Engineering,
- [4] Zhang, Zijun & Zeng, Yaohui & Kusiak, A. (2012). Minimizing pump energy in a wastewater processing plant. Energy. 47. 505–514. 10.1016/j.energy.2012.08.048.
- [5] FLYGT, a xylem brand (2013). Variable speed wastewater pumping. White Paper, Retrieved from http://www.wioa.org.au/operator_resources/documents/XylemVariableSpeedP umping.pdf

Appendix

MATLAB CODE

Matlab code for:

- Pump Speed Algorithm Curves
- Power Data
- Speed Data
- Level Data

Pump_Algorithm_Curves.m

function[]=Pump_Algorithm_Curves(A)
% Plots the experimental speed versus level curves for different values of curvature (A
value)

% experimental constants 1=[0:0.25:20] % tank level range (in.) s1=3*1; % linear speed vs level x1=11; % low level endpoint x2=19; % high level endpoint % for loop to iterate multiple A values for k =1:length(A) $B=A(k)./(x1/2-x2/2)^{2}$ $s3=-A(k)+B.*(1-((x1+x2)/2)).^2;$ s4 = s1 + s3s5=gradient(s4)s4(gradient(s4)<0)=NaN s4((s4)>60)=60 s4(1<11)=NaN %s4(l>19)=NaN s1(l<11)=NaN plot(1,s4)hold on plot(l,s1) find(s5>0)end xlim([11 20]) legend('A=0','A=2.5','A=5','A=7.5','A=10','A=12.5','A=15') xlabel('Tank Level(In)') ylabel('Pump Speed(Hz)') title('Pump Speed vs. Level Control Algorithm') set(gca, 'FontSize', [14]) set(gca, 'Box', ['off']) set(gca, 'FontName', ['Times New Roman'])

end

POWER_Data.m

% Experiment_Pumping Power Data for Pilot Scale Station % Pump Power Data is taken for variable curvature(A value)Speed vs. Level

%% Excel Data % Each curavture (A value) has 3 experimental cycles:

% A=0

dataset=xlsread('A=0(average).xlsx','A=0(2)','C2:C1801');%A=0 dataset2=xlsread('A=0(average).xlsx','A=0(3)','C2:C1801');%A=0 dataset3=xlsread('A=5,A=7.5(average).xlsx','Pump_Station_17','C10857:C12656');% A=0

% A=5

dataset4=xlsread('A=5(new average).xlsx','Pump_Station_19','C3:C1802');%A=5 dataset5=xlsread('A=5(new average).xlsx','Pump_Station_19','C1813:C3612');%A=5 dataset6=xlsread('A=5(new average).xlsx','Pump_Station_19','H3:H1802');%A=5

% A=7.5

dataset7=xlsread('A=7.5_new_average.xlsx','Pump_Station_21','C3:C1802');%A=7.5 dataset8=xlsread('A=7.5_new_average.xlsx','Pump_Station_21','J3:J1802');%A=7.5 dataset9=xlsread('A=7.5_new_average.xlsx','Pump_Station_21','P3:P1802');%A=7.5

% A=10

dataset10=xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','C3:C1802');%A=1 0

dataset11=xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','C1813:C3612');% A=10

dataset12=xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','C3623:C5422');% A=10

% A=2.5

dataset13=xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','C5433:C7232');% A=2.5

dataset14=xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','C7243:C9042');% A=2.5 dataset15=xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','C9053:C10852');% A=2.5

% A=12.5

dataset16=xlsread('A=12.5.xlsx','Pump_Station_22','C3:C1802');%A=12.5 dataset17=xlsread('A=12.5.xlsx','Pump_Station_22','C1813:C3612');%A=12.5 dataset18=xlsread('A=12.5.xlsx','Pump_Station_22','I3:I1802');%A=12.5

% A=15

dataset19=xlsread('A=15.xlsx','Pump_Station_25','C3:C1802');%A=15 dataset20=xlsread('A=15.xlsx','Pump_Station_25','C1813:C3612');%A=15 dataset21=xlsread('A=15.xlsx','Pump_Station_25','C3623:C5422');%A=15 %%

%% Mean Active Power (W)

pdmean=(dataset+dataset2+dataset3)/3; pdemean2=(dataset4+dataset5+dataset6)/3; pdemean3=(dataset7+dataset8+dataset9)/3; pdemean4=(dataset10+dataset11+dataset12)/3; pdemean5=(dataset13+dataset14+dataset15)/3; pdemean6=(dataset16+dataset17+dataset18)/3; pdemean7=(dataset19+dataset20+dataset21)/3;

x=[1:1800] % Array spanning length of dataset for plotting purposes

% For example sake, I overwrote values of power for times where the pump was turned off pdmean(pdmean<200)=200 pdemean3(pdemean3<190)=190 %%

%% Converting to Specific Energy % Trapz takes integral under power data to get energy [Wh] % Divide by total value of pumped flow to get specific energy [Wh/gallon] % Convert to metric units for publication in Canada [Wh/m^3]

me=264.17 %gpm to m3/m

%A=0 energy = trapz(dataset)/(3600*214.71/me) energy2 = trapz(dataset2)/(3600*214.5/me) energy3 = trapz(dataset3)/(3600*214.38/me)

%A=5

energy4 = trapz(dataset4)/(3600*211.23/me)energy5 = trapz(dataset5)/(3600*211.97/me)energy6 = trapz(dataset6)/(3600*211.55/me)%A=7.5 energy7 = trapz(dataset7)/(3600*209.22/me)energy8 = trapz(dataset8)/(3600*209.91/me)energy9 = trapz(dataset9)/(3600*209.74/me)%A=10 energy10 = trapz(dataset10)/(3600*207.52/me)energy11 = trapz(dataset11)/(3600*208/me)energy12 = trapz(dataset12)/(3600*208.22/me)%A=2.5 energy13 = trapz(dataset13)/(3600*213.59/me)energy14 = trapz(dataset14)/(3600*213.55/me)energy15 = trapz(dataset15)/(3600*212.38/me)%A=12.5 energy16 = trapz(dataset16)/(3600*205.38/me)energy17 = trapz(dataset17)/(3600*205.5/me)energy18 = trapz(dataset18)/(3600*205.69/me)%A=15 energy19 = trapz(dataset19)/(3600*203.32/me)energy20 = trapz(dataset20)/(3600*203.74/me)energy21 = trapz(dataset21)/(3600*202.59/me)

%Specific Energy Arrays

SE=[energy energy2 energy3] %A=0 SE2=[energy4 energy5 energy6] %A=5 SE3=[energy7 energy8 energy9] %A=7.5 SE4=[energy10 energy11 energy12] %A=10 SE5=[energy13 energy14 energy15] %A=2.5 SE6=[energy16 energy17 energy18] %A=12.5 SE7=[energy19 energy20 energy21] %A=15

%% Average, %Savings, and STD for different values of Curvature

% Average of Specific Energy

Avg=[mean(SE) mean(SE5) mean(SE2) mean(SE3) mean(SE2) mean(SE6) mean(SE7)] % %Savings of Specific Energy savings=[0 (mean(SE)-mean(SE5))/((mean(SE)+mean(SE5))/2)*100 (mean(SE)mean(SE2))/((mean(SE)+mean(SE2))/2)*100 (mean(SE)mean(SE3))/((mean(SE)+mean(SE3))/2)*100 (mean(SE)- mean(SE4))/((mean(SE)+mean(SE4))/2)*100 (mean(SE)mean(SE6))/((mean(SE)+mean(SE6))/2)*100 (mean(SE)mean(SE7))/((mean(SE)+mean(SE7))/2)*100] % STD of Specific Energy STD2=[0 std((SE)) std((SE5)) std((SE2)) std((SE3)) std((SE2)) std((SE6)) std((SE7))]

A=[0 2.5 5 7.5 10 12.5 15]

% Subplot 1 is average specific energy for each value of curvature subplot(2,1,1) plot(A,Avg,'o') hold on ylabel('Specific Energy (Wh/m^3)') set(gca, 'FontSize', [14]) set(gca, 'Box', ['off']) set(gca, 'Box', ['off']) set(gca, 'FontName', ['Times New Roman']) set(gca, 'XTick', []) set(gca, 'XTick', []) set(gca, 'YGrid', 'on') set(gca, 'LineWidth', [2.000])

% Subplot 2 is the %savings of each curvature compared to the linear one(A=0) % Added a polyfit curve just to illustrate how the data is follow a trend subplot(2,1,2)plot(A,savings,'o') P=polyfit(A, savings, 2) x1 = linspace(0, 15) $y_1 = polyval(P, x_1)$ hold on plot(x1,y1)xlabel("'A" Value','fontsize',18) set(gca, 'Box', ['off']) set(gca, 'YGrid', 'on') set(gca, 'FontName', ['Times New Roman']) set(gca, 'XTick', [0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15]) set(gca, 'LineWidth', [2.000]) text(11,3.5,'\leftarrow 2nd Order Fit', 'FontSize',8, 'FontWeight', 'bold') ylabel('Wh/m^3 Savings(%)','fontsize',14) %%

SPEED_data.m

```
sdataset=xlsread('A=0(average).xlsx','A=0(2)','E2:E1801');%A=0
sdataset2=xlsread('A=0(average).xlsx','A=0(3)','E2:E1801');%A=0
sdataset3=xlsread('A=5,A=7.5(average).xlsx','Pump Station 17','C10857:C12656');%
A=0
sdataset4=xlsread('A=5(new average).xlsx','Pump Station 19','E3:E1802');%A=5
sdataset5=xlsread('A=5(new average).xlsx','Pump Station 19','E1813:E3612');%A=5
sdataset6=xlsread('A=5(new average).xlsx','Pump Station 19','J3:J1802');%A=5
sdataset7=xlsread('A=7.5 new average.xlsx','Pump Station 21','L3:L1802');%A=7.5
sdataset8=xlsread('A=7.5 new average.xlsx','Pump Station 21','E3:E1802');%A=7.5
sdataset9=xlsread('A=7.5 new average.xlsx','Pump Station 21','R3:R1802');%A=7.5
sdataset10=xlsread('A=10,A=2.5(average).xlsx','Pump Station 18','E3:E1802');%A=1
0
sdataset11=xlsread('A=10,A=2.5(average).xlsx','Pump Station 18','E1813:E3612');%
A = 10
sdataset12=xlsread('A=10,A=2.5(average).xlsx','Pump Station 18','E3623:E5422');%
A=10
sdataset13=xlsread('A=10,A=2.5(average).xlsx','Pump Station 18','E5433:E7232');%
A=2.5
sdataset14=xlsread('A=10,A=2.5(average).xlsx','Pump Station 18','E7243:E9042');%
A=2.5
sdataset15=xlsread('A=10,A=2.5(average).xlsx','Pump Station 18','E9053:E10852');
%A=2.5
sdataset16=xlsread('A=12.5.xlsx','Pump Station 22','E3:E1802');%A=12.5
sdataset17=xlsread('A=12.5.xlsx','Pump Station 22','E1813:E3612');%A=12.5
sdataset18=xlsread('A=12.5.xlsx','Pump Station 22','K3:K1802');%A=12.5
sdataset19=xlsread('A=15.xlsx','Pump Station 25','E3:E1802');%A=12.5
sdataset20=xlsread('A=15.xlsx','Pump Station 25','E1813:E3612');%A=12.5
sdataset21=xlsread('A=15.xlsx','Pump Station 25','E3623:E5422');%A=12.5
sdmean=(sdataset+sdataset2)/2;
sdemean2=(sdataset4+sdataset5+sdataset6)/3;
sdemean3=(sdataset7+sdataset8+sdataset9)/3;
sdemean4=(sdataset10+sdataset11+sdataset12)/3;
sdemean5=(sdataset13+sdataset14+sdataset15)/3:
sdemean6=(sdataset16+sdataset17+sdataset18)/3;
```

```
sdemean7=(sdataset19+sdataset20+sdataset21)/3;
```

```
sdmean(sdmean<30)=NaN
x=[1:1800]
%subplot(2,1,1)
%plot(x,dataset,x,dataset2,x,dataset3)
%legend('Linear(A=0)','Conc. Up(A=5)','Conc. Up(A=-6.5)')
xlabel('Time(s)')
ylabel('Pump Speed(Hz)')
%title('Pump Energy Data')
set(gca, 'FontSize', [10])
set(gca, 'Box', ['off'])
%text(10,500,'Spec. E =0.5979 wH','FontSize',8)
%text(10,475,'Spec. E=0.5832 wH','FontSize',8)
%text(10,450,'Spec. E=0.5853 wH','FontSize',8)
%subplot(2,1,2)
%A=0
speed = mean(sdataset)
speed2 = mean(sdataset2)
speed3 = mean(sdataset3)
%A=5
speed4 = mean(sdataset4)
speed5 = mean(sdataset5)
speed6 = mean(sdataset6)
%A=7.5
speed7 = mean(sdataset7)
speed8 = mean(sdataset8)
speed9 = mean(sdataset9)
%A=10
speed10 = mean(sdataset10)
speed11 = mean(sdataset11)
speed12 = mean(sdataset12)
%A=2.5
speed13 = mean(sdataset13)
speed14 = mean(sdataset14)
speed15 = mean(sdataset15)
%A=12.5
speed16 = mean(sdataset16)
speed17 = mean(sdataset17)
speed18 = mean(sdataset18)
%A=15
speed19 = mean(sdataset19)
speed20 = mean(sdataset20)
```

speed21 = mean(sdataset21)

```
Sp=[speed speed2] %A=0
Sp2=[speed4 speed5 speed6] %A=5
Sp3=[speed7 speed8 speed9] %A=7.5
Sp4=[speed10 speed11 speed12] %A=10
Sp5=[speed13 speed14 speed15] %A=2.5
Sp6=[speed16 speed17 speed18] %A=12.5
Sp7=[speed19 speed20 speed21] %A=15
STD=[std(Sp) std(Sp5) std(Sp2) std(Sp3) std(Sp4) std(Sp6) std(Sp7)]
Avgs=[mean(Sp) mean(Sp5) mean(Sp2) mean(Sp3) mean(Sp4) mean(Sp6)
mean(Sp7)]
%max=[max(dataset) max(dataset15) max(dataset4) max(dataset7) max(dataset10)
max(dataset16) max(dataset19)]
Max=[max(sdmean),max(sdemean5),max(sdemean2),max(sdemean4)]
,max(sdemean6),60]
%savings=[0 (mean(SE)-mean(SE5))/((mean(SE)+mean(SE5))/2)*100 (mean(SE)-
mean(SE2))/((mean(SE)+mean(SE2))/2)*100 (mean(SE)-
mean(SE3))/((mean(SE)+mean(SE3))/2)*100 (mean(SE)-
mean(SE4))/((mean(SE)+mean(SE4))/2)*100 (mean(SE)-
mean(SE6))/((mean(SE)+mean(SE6))/2)*100 (mean(SE)-
mean(SE7))/((mean(SE)+mean(SE7))/2)*100]
A=[0 2.5 5 7.5 10 12.5 15]
J=find(sdemean3>=45)
i=length(J)
H=find(sdmean>=45)
i2=length(H)
%P=polyfit(A, savings, 2)
%x1 = linspace(0, 15)
%y1=polyval(P,x1)
subplot(3,1,1)
plot(A,AvgL,'+')
%plot(x,sdmean,x,sdemean2,x,sdemean3,x,sdemean4,x,sdemean5,x,sdemean6,x,sdem
ean7)
%hold on
%plot(x1,y1)
%legend('SE Savings','2nd Degree Fit')
vlabel('Avg. Level(cm)','fontsize',12)
%title('Specific Energy Savings')
%set(gca, 'FontSize', [15])
set(gca, 'FontWeight', ['bold'])
```

```
set(gca, 'Box', ['off'])
```

set(gca, 'FontName', ['Times New Roman']) %set(gca, 'XTick', [0 300 600 900 1200 1500 1800]) set(gca, 'YGrid', 'on') set(gca, 'XTick', []) set(gcf, 'Color', [1 1 1])

subplot(3,1,2) plot(A,Avgs,'+') %xlabel(''A'' Value') ylabel('Avg. Speed(Hz)','fontsize',12) set(gca, 'FontWeight', ['bold']) %set(gca, 'FontSize', [15]) set(gca, 'Box', ['off']) set(gca, 'Box', ['off']) set(gca, 'Yortick', []) ylim([35 40]) set(gca, 'YGrid', 'on')

subplot(3,1,3) plot(A,Max,'+') xlabel(''A'' Value','fontsize',18) ylabel('Max. Speed(Hz)','fontsize',12) set(gca, 'FontWeight', ['bold']) %set(gca, 'FontSize', [18]) set(gca, 'Box', ['off']) set(gca, 'FontName', ['Times New Roman']) set(gca, 'XTick', [0 2.5 5.0 7.5 10.0 12.5 15.0]) set(gca, 'YGrid', 'on')

LEVEL_data.m

%% Sump Tank Level Data taken for different values of Curvature (A Value)

%% Sump Tank Level Data pulled from Excel
% 3 datasets for each value of curvature (Each pump cycle is 1800 seconds)
% 2.54 converts inches to centimeters

% A=0

ldataset=2.54*xlsread('A=0(average).xlsx','A=0(2)','D2:D1801');%A=0 ldataset2=2.54*xlsread('A=0(average).xlsx','A=0(3)','D2:D1801');%A=0 % ldataset3 excel data is corrupted, just ignored it.

% A=5

ldataset4=2.54*xlsread('A=5(new average).xlsx','Pump_Station_19','D3:D1802');%A=5 ldataset5=2.54*xlsread('A=5(new average).xlsx','Pump_Station_19','D1813:D3612');%A=5 ldataset6=2.54*xlsread('A=5(new average).xlsx','Pump_Station_19','I3:I1802');%A=5

% A=7.5

ldataset7=2.54*xlsread('A=7.5_new_average.xlsx','Pump_Station_21','K3:K1802');% A=7.5

ldataset8=2.54*xlsread('A=7.5_new_average.xlsx','Pump_Station_21','D3:D1802');% A=7.5

ldataset9=2.54*xlsread('A=7.5_new_average.xlsx','Pump_Station_21','Q3:Q1802');% A=7.5

% A=10

ldataset10=2.54*xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','D3:D1802'); %A=10

ldataset11=2.54*xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','D1813:D361 2');%A=10

ldataset12=2.54*xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','D3623:D542 2');%A=10

% A=2.5

ldataset13=2.54*xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','D5433:D723 2');%A=2.5 ldataset14=2.54*xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','D7243:D904 2');%A=2.5 ldataset15=2.54*xlsread('A=10,A=2.5(average).xlsx','Pump_Station_18','D9053:D108 52');%A=2.5

% A=12.5

ldataset16=2.54*xlsread('A=12.5.xlsx','Pump_Station_22','D3:D1802');%A=12.5 ldataset17=2.54*xlsread('A=12.5.xlsx','Pump_Station_22','D1813:D3612');%A=12.5 ldataset18=2.54*xlsread('A=12.5.xlsx','Pump_Station_22','J3:J1802');%A=12.5

% A=15

ldataset19=2.54*xlsread('A=15.xlsx','Pump_Station_25','D3:D1802');%A=15 ldataset20=2.54*xlsread('A=15.xlsx','Pump_Station_25','D1813:D3612');%A=15 ldataset21=2.54*xlsread('A=15.xlsx','Pump_Station_25','D3623:D5422');%A=15 %%

%% Mean Level Dataset

```
Isdmean=(Idataset+Idataset2)/2;
Isdemean2=(Idataset4+Idataset5+Idataset6)/3;
Isdemean3=(Idataset7+Idataset8+Idataset9)/3;
Isdemean4=(Idataset10+Idataset11+Idataset12)/3;
Isdemean5=(Idataset13+Idataset14+Idataset15)/3;
Isdemean6=(Idataset16+Idataset17+Idataset18)/3;
Isdemean7=(Idataset19+Idataset20+Idataset21)/3;
%%
```

%plot(x,lsdmean,x,lsdemean2,x,lsdemean3,x,lsdemean4,x,lsdemean5,x,lsdemean6,x,l sdemean7)

```
%x=[1:1800]
%subplot(2,1,1)
%plot(x,dataset,x,dataset2,x,dataset3)
%legend('Linear(A=0)','Conc. Up(A=5)','Conc. Up(A=-6.5)')
xlabel('Time(s)')
ylabel('Power(W)')
title('Pump Energy Data')
set(gca, 'FontSize', [14])
set(gca, 'FontSize', [14])
set(gca, 'Box', ['off'])
%text(10,500,'Spec. E =0.5979 wH','FontSize',8)
%text(10,475,'Spec. E=0.5853 wH','FontSize',8)
%text(10,450,'Spec. E=0.5853 wH','FontSize',8)
%subplot(2,1,2)
```

%A=0 %% Average Level during each Pumping Cycle for given Curvature Value % A=0 level = mean(ldataset) level2 = mean(ldataset2)

% A=5

level4 = mean(ldataset4) level5 = mean(ldataset5) level6 = mean(ldataset6)

% A=7.5

level7 = mean(ldataset7) level8 = mean(ldataset8) level9 = mean(ldataset9)

% A=10

level10 = mean(ldataset10)
level11 = mean(ldataset11)
level12 = mean(ldataset12)

% A=2.5

level13 = mean(ldataset13) level14 = mean(ldataset14) level15 = mean(ldataset15)

% A=12.5

level16 = mean(ldataset16) level17 = mean(ldataset17) level18 = mean(ldataset18)

% A=15

level19 = mean(ldataset19) level20 = mean(ldataset20) level21 = mean(ldataset21)

SEI=[level level2] %A=0 SEI2=[level4 level5 level6] %A=5 SEI3=[level7 level8 level9] %A=7.5 SEI4=[level10 level11 level12] %A=10 SEI5=[level13 level14 level15] %A=2.5 SEI6=[level16 level17 level18] %A=12.5 SEI7=[level19 level20 level21] %A=15 %% %% Average Level and STD STD=[std(SEl) std(SEl5) std(SEl2) std(SEl3) std(SEl4) std(SEl6) std(SEl7)] AvgL=[mean(SEl) mean(SEl5) mean(SEl2) mean(SEl3) mean(SEl4) mean(SEl6) mean(SEl7)] A=[0 2.5 5 7.5 10 12.5 15]

%Plot of Average Sump Level for value of Curvature plot(A,AvgL,'+') xlabel(''A'' Value') ylabel('Avg. Tank Level(cm)') set(gca, 'FontSize', [14]) set(gca, 'Box', ['off']) set(gca, 'Box', ['off']) set(gca, 'FontName', ['Times New Roman']) set(gca, 'XTick', [0 2.5 5.0 7.5 10.0 12.5 15.0]) %%