Enhancing Stormwater Management through Machine Learning-based Real-time Prediction and Control

---

A Dissertation
Presented to
the Faculty of the School of Engineering and Applied Science
University of Virginia

In Partial Fulfillment
of the requirements for the Degree
Doctor of Philosophy (Civil Engineering)
by

Benjamin D. Bowes
August 2021

Abstract

Many cities face high levels of flooding and pollution from stormwater runoff due to factors such as ongoing urbanization and aging stormwater management infrastructure. As climate change continues to alter precipitation, temperature, and sea levels, existing stormwater systems will be pushed beyond their designed capacity, further increasing flooding and pollution. This dissertation focuses on real-time prediction and control of stormwater related systems as a means to enhance community resilience to these issues. The research advanced the application of emerging deep machine learning techniques to water resources engineering using the coastal city of Norfolk, Virginia as a test-bed for these novel approaches. Norfolk faces recurrent flooding from storm events and ongoing sea level rise, while having to reduce polluted stormwater runoff entering the Chesapeake Bay. The first study uses supervised deep machine learning to create forecasts of groundwater table response to storm events, providing additional information for flood forecasting and stormwater management. The second and third studies explore deep reinforcement learning as a method for real-time control of stormwater systems. In the second study, reinforcement learning is used to create control strategies that mitigate flooding in a simple stormwater system scenario inspired by a watershed in Norfolk. The third study uses reinforcement learning for real-time stormwater system control with the competing objectives of mitigating flooding while also improving water quality by capturing sediment. This was done using a real-world simulation of Norfolk's Hague neighborhood instead of the simplified system from the second study. Key findings from this research are (i) deep machine learning can be used to create real-time hourly forecasts of the groundwater table response to storm events in a coastal city using forecast rainfall and tide conditions as input data with a mean root mean squared error of 0.05 m, (ii) reinforcement learning can learn real-time stormwater system control strategies that reduce flooding compared to conventional, uncontrolled stormwater systems by 32%, (iii) system-level stormwater real-time control with reinforcement learning can reduce flooding by 13% compared to local-scale control rules, and (iv) reinforcement learning can use real-time water quality observations to reduce sediment loads by an average of 52% with only a small increase in flooding (5%) compared to conventional, uncontrolled stormwater systems. While this dissertation has focused on coastal cities, the knowledge and methods developed could be applied to inland stormwater systems as well. These advancements contribute to a growing body of knowledge related to smart stormwater systems, which can aid communities through improved prediction and control of stormwater to reduce flooding and pollution.

Approval Sheet

This dissertation is submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Civil Engineering)

Benjamin Donald Bowes

This dissertation has been read and approved by the Examining Committee:

Jonathan L. Goodall, Adviser

Teresa B. Culver, Committee Chair

Peter A. Beling

Madhur Behl

Matthew Reidenbach

Accepted for the School of Engineering and Applied Science:

Jennifer L. West, Dean, School of Engineering and Applied Science

August 2021

To Caitlin, Arthur, and baby Bowes

Acknowledgements

This dissertation has been possible through the love and support of my family. Thank you to my wife, Caitlin, who encouraged me to leave a stable, boring job and start grad school. Five years later, your support has been constant and invaluable, even while you've worked full time and became a mom! Thank you to Arthur, our three year old son, who has made life, especially pandemic life, fun and exciting. With baby number two on the way, I know that we are ready for our next adventure as a family!

Thank you to my parents, Anne and Andy, for your love, support, and faith. You set me on a firm foundation with years of home school and outdoor adventures. Thank you for being amazing role models. Thanks to my sister, Katie, for adventures around C-Ville and always being ready to provide veterinary support for Olive and Violet. I would also like to thank my in-laws, Boomer (Richard, Mech/Aerospace Engineering '89) and Jolie, and brother-in-law Rowan. I know we can always count on your support, whether it's with babysitting or touring local breweries (or both)! Thank you to Andy, Anne, Richard, and Jolie for your brave and loyal service in our country's military.

I would also like to acknowledge my good friends and colleagues from the UVA Hydroinformatics group including Jeff Sadler, Mohamed Morsy, Bakinam Esawy, Gina O'Neil, Alex Chen, Youngdon Choi, Iman Maghami, Faria Tuz Zahura, Linnea Saby, Jake Nelson, Natalie Lerma, Ciara Horne, Chase Dong, Yidi Wang, Binata Roy, Victor Sobral, and Ruchir Shah. I especially thank Jeff Sadler and Mohamed Morsy, who were fantastic mentors in my early grad years. You showed me the meaning of hard work and dedication; I hope I've been able to reflect those high standards and pass them on to others. We've all become part of the Link Lab and I thank the many friends I've made there as well, including my RL coauthors Arash Tavakoli and Cheng Wang. I've been fortunate to be part of such a diverse group of people. Keep up the great work! Wahoowa!

My dissertation committee deserves special thanks time and guidance. Thanks to Dr. Teresa Culver for agreeing to chair the committee and for mentoring me as a TA for your Intro to Environmental Engineering class. Thank you Dr. Madhur Behl for your expertise with all things cyber-physical. Thank you Dr. Peter Beling for introducing me to reinforcement learning. Thank you to Dr. Matt Reidenbach for your expertise with coastal resilience. Thank you all for asking hard questions and seeking new knowledge!

I especially thank my advisor, Dr. Jon Goodall. From the time I took your Water Resources Engineering class as a continuing studies student while working full time, you've enabled me to see myself as an engineer, even when I didn't believe it myself. You've provided me countless opportunities to learn and grow as a student, researcher, and teacher. Thank you Jon for taking a chance on me as a PhD student and for leading a diverse, engaged, and friendly research group!

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Coastal communities are experiencing increased flooding and pollution due to sea level rise, increased precipitation, more frequent extreme weather events, and urbanization (Wuebbles et al., 2017; Sweet and Park, 2014; Moftakhari et al., 2015; Moftakhari et al., 2017; Alamdari et al., 2020). Recurrent nuisance flooding (i.e., low-level flooding caused by small storm events, high tides, etc.) disrupts transportation and other infrastructure systems and, over time, can lead to severe economic impacts and infrastructure damage (Moftakhari et al., 2015; Moftakhari et al., 2017). The pollutants in urban stormwater runoff, such as nutrients, sediment, and metals, can negatively impact aquatic ecosystems through toxicity and eutrophication (Brudler et al., 2019; Murphy et al., 2011). In order to increase the resilience of coastal communities and protect natural ecosystems to these challenges, this dissertation focuses on leveraging emerging machine learning techniques to forecast physical processes within an urban setting and improve the ability of stormwater systems to mitigate flooding and protect water quality.

Stormwater systems play a critical role in mitigating flooding and pollution impacts in urban areas and are conventionally designed based on historic conditions to operate passively using only the force of gravity. These systems are required because urbanization increases impervious area (e.g., roofs, parking lots), which limits stormwater infiltration and increases the speed and volume of water that becomes runoff. In turn, this often leads to greater flooding and increased pollutant loads discharged to receiving waters (Boyer and Kieser, 2012; Walsh et al., 2005). Conventional stormwater systems attempt to alleviate this by temporarily storing runoff in ponds, allowing infiltration, settling of sediment, and uptake of nutrients while slowing its flow to receiving waters. Traditionally, increased flooding and pollution would be mitigated by expanding stormwater system capacity. However, such capital improvement projects are expensive and disruptive, especially considering the estimated $8 billion funding gap for operation and maintenance of existing stormwater infrastructure in the U.S. (ASCE, 2021). Additionally, conventional stormwater systems are designed to operate under specific conditions and are not able to adapt to the wide range of possible storm events or future environmental regulations. These factors indicate that conventional stormwater systems will struggle to keep pace with changing land use and climate change (Berggren et al., 2012; Mynett and Vojinovic, 2009; Neumann et al., 2015).

Due to the growth and ubiquity of the Internet of Things (IoT), it is now possible to monitor and operate stormwater systems more efficiently and in real-time as cyber-physical systems. By retro-fitting parts of conventional stormwater systems with sensors and actuators (e.g. remotely controlled valves and pumps), they can be monitored and controlled in real-time based on current and forecast conditions (Kerkez et al., 2016). This has been shown to be a cost-effective way to increase the efficiency of existing stormwater infrastructure and is an emerging stormwater management tool (Jose Meneses et al., 2018). In current practice, stormwater real-time control (RTC) is based on control rules for single components of an infrastructure system (e.g., level control of a retention pond), which has been shown to reduce flooding and improve water quality locally (Marchese et al., 2018; OptiRTC and Geosyntec Consultants Inc., 2017). However, as the complexity of controlled stormwater systems increases, the task of creating rules and policies able to consider all system interactions and changing land use and climate conditions will be extremely difficult.

Efficient RTC within a complex system that mitigates flooding and protects the quality of receiving waters, is a challenging task that can benefit from system-level control (Wong and Kerkez, 2018). Recent research has explored system-level methods of optimizing stormwater RTC. However, these methods typically focus solely on water quantity, can be very computationally expensive, or require simplifying the non-linear dynamics of stormwater systems, thus not completely capturing the system behavior (Sadler et al., 2019). The hardware technology to enable system-level RTC based on water quantity is readily available and continuing improvements in real-time water quality sensors may soon allow more direct observation and control of not only water quantity but water quality (Wong and Kerkez, 2016). System-level control methods will be needed to make the best use of these data streams instead of attempting to engineer rules that cover all possible interactions between stormwater system components, pollutants, and environmental conditions.

Coastal cities face additional challenges in implementing system-level stormwater RTC beyond the multiple of objectives of flood mitigation and water quality protection. First, outfalls of coastal stormwater systems that drain to tidal waterbodies can be blocked during high tides. In some cases tidal water can flow back into inland parts of the stormwater system and cause flooding or reduce system capacity; this is expected to become more severe with sea level rise (Sadler et al., 2020a; Shen et al., 2019). If stormwater RTC does not take tidal conditions into account, too much water may be released from upstream areas while the outfall is fully or partially blocked, causing increased flooding. Second, coastal areas often have high groundwater tables that can infiltrate into stormwater systems, decreasing the capacity available for stormwater runoff (Karpf and Krebs, 2013; Flood and Cahoon, 2011). If groundwater exchange with the stormwater system is not considered and ponds are drawndown to prepare for a storm, there maybe a significant amount of inflow from groundwater that could alter the intended outcome of RTC strategies. Stormwater RTC relies on forecasts (predominantly rainfall forecasts) to make predictive decisions. Therefore, there is a need to incorporate current and forecast tidal conditions into stormwater RTC, as well as developing a methodology for forecasting the groundwater table, which, unlike rain and tide, is rarely forecast. Advances in machine learning provide emerging alternatives to improve our understanding and ability to model such processes in coastal cities.

Machine learning (ML), specifically deep ML, techniques are increasingly used in hydrology and water resources engineering to complement or replace physics-based models (Shen, 2018; Sadler et al., 2018; Fahimi et al., 2017; Maier et al., 2010; Maier and Dandy, 2000; Yang et al., 2017; Yaseen et al., 2015; Mullapudi et al., 2020). While physics-based models are valued for their interpretability and ability to simulate many kinds of scenarios (e.g., the future impacts of climate change), they have long-standing limitations such as scale dependency, simplified physics for computational efficiency, and challenging parameterization and calibration in heterogeneous urban settings (Shen, 2018). In contrast, ML is able to use observed data to learn the relationships between model inputs and outputs. The physical relationships and parameters of a system do not need to be explicitly defined, but they can be approximated through an iterative learning process (Solomatine and Ostfeld, 2008). This is especially beneficial in complex environments like coastal cities where many processes interact and are difficult to model with other techniques (Sadler et al., 2018). Despite the broad potential and proven capabilities of ML, open questions regarding its use in coastal urban hydrology and stormwater management remain. In this dissertation two types of ML are explored: supervised learning for time series forecasting of natural processes and reinforcement learning for real-time stormwater system control. These techniques are applied within the City of Norfolk, Virginia and are explained and evaluated in this dissertation.

The City of Norfolk, Virginia is a prime example of a coastal city with recurrent flooding and impaired water quality and serves as the study area for this dissertation. Norfolk has a high rate of relative sea level rise due to regional land subsidence (Eggleston and Pope, 2013) and climate change; its low elevation, flat topography, and regular hurricane season also contribute to increasingly frequent and severe recurrent flooding (Sweet and Park, 2014). Because of Norfolk's location on the Chesapeake Bay, pollutants in stormwater runoff (i.e., sediment and the nutrients nitrogen and phosphorous) must be managed to limit harmful impacts on the Bay, such as algal blooms and dead zones. By addressing these flooding and water quality concerns, Norfolk can act as a model for other communities facing similar conditions in the future.

This dissertation is composed of three studies that collectively advance understanding of the use of machine learning techniques for real-time control of stormwater systems, with a focus on urban coastal watersheds. The first study (Chapter 2) focuses on creating forecasts of groundwater table response to storm events. This study evaluates supervised ML approaches for time series forecasting using neural networks that can learn to approximate groundwater table fluctuations. Observed groundwater table, sea level, and rainfall data are used as input to train these models. The NN models are tested with forecast sea level and rainfall data to quantify their real-time forecasting accuracy.

The second study (Chapter 3) focuses on using reinforcement learning (RL) to develop stormwater system control strategies to mitigate flooding. This study evaluates RL's ability to learn control policies for a tidally influenced coastal stormwater system that improves system-level performance (flood mitigation and pond depth control), instead of setting simple control rules for individual components of stormwater systems. Observed and forecast rainfall and tide are used so that RL can learn proactive control strategies. RL's performance is bench-marked against (i) a passive, gravity-driven system, (ii) model predictive control (MPC), and (iii) rule-based control.

To further advance real-time control of stormwater systems with RL, the third and final study (Chapter 4) incorporates water quality, in addition to flood mitigation, in the control strategies learned with RL. This study uses a real-world simulation representing the stormwater system in Norfolk's flood-prone Hague neighborhood. The flood mitigation and water quality performance of RL is compared to (i) the passive, uncontrolled system currently in place, (ii) a predictive rule-based control strategy with extended detention time, and (iii) a reactive rule-based control strategy based on observed water quality measurements. The impact of groundwater exchange on the performance of the controlled ponds is evaluated to assess its importance in RTC of stormwater systems in coastal areas.

Chapter 2

Forecasting Groundwater Table in a Flood Prone Coastal City with Long Short-term Memory and Recurrent Neural Networks [1]

2.1 Introduction

Storm events in low relief coastal areas can quickly raise the groundwater table, which is often relatively shallow (Giambastiani et al., 2017; Taormina et al., 2012). During these events, infiltration and groundwater table response decrease the volume available for stormwater storage, therefore increasing runoff and, by extension, loads on stormwater systems (Rotzoll and Fletcher, 2012). Many coastal urban areas are also experiencing increased flooding due to land subsidence and climate change effects, such as sea level rise (Sweet and Park, 2014), increased precipitation, and more frequent extreme climactic events (Wuebbles et al., 2017). While there are several causes of flooding in coastal cities (Sadler et al., 2018), the groundwater table level is a largely unrepresented factor and forecasting its variations can provide valuable information to aid in planning and response to storm events. Furthermore, because the groundwater table will rise as sea level rises (Rotzoll and Fletcher, 2012; Bjerklie et al., 2012; Hoover et al., 2017; Masterson et al., 2016), stormwater storage capacity will continue to decrease and inundation from groundwater may occur. Damage from groundwater inundation, which occurs through different mechanisms than overland flooding, can have significant impacts on subsurface structures (Kreibich and Thieken, 2008; Abboud et al., 2018). Even if groundwater inundation does not regularly reach the land surface, increased duration of high groundwater table levels could have significant impacts on infrastructure (Hoover et al., 2017; Bloetscher et al., 2012; Flood and Cahoon, 2011; Sadler et al., 2017) making groundwater table forecasting an increasingly important part of effectively modeling and predicting coastal urban flooding.

In the field of groundwater hydrology, models based on the physical principles of groundwater flow have traditionally been some of the main tools for understanding

[1] This chapter has been published as Bowes, B.D., Sadler, J.M., Morsy, M.M., Behl, M., Goodall, J.L., 2019. Forecasting Groundwater Table in a Flood Prone Coastal City with Long Short-term Memory and Recurrent Neural Networks. Water 11, 1098. https://doi.org/10.3390/w11051098

the mechanics of these systems (Masterson et al., 2016; Chang et al., 2016b; Doble et al., 2017; Heywood and Pope, 2009; Masterson and Garabedian, 2007; Park and Parker, 2008; Pauw et al., 2014). Developing these models, however, requires extensive details about aquifer properties. In urban areas, this level of detail is hard to achieve at high resolutions because the subsurface is a complex mix of natural and anthropogenic structures such as varied geologic deposits, buried creeks or wetlands, roadbeds, building foundations, and sanitary and stormwater pipes. These factors should be considered when developing a physics-based model; if the necessary data are not available then assumptions and estimations must be substituted based on domain knowledge. Even if the data necessary to build a physics-based model are available, there is still the challenge of calibrating the model to adequately reflect reality.

Machine learning approaches are being increasingly used by hydrologists in order to mitigate the difficulties associated with physics-based models (Sadler et al., 2018; Fahimi et al., 2017; Govindaraju, 2000b; Govindaraju, 2000a; Maier and Dandy, 2000; Maier et al., 2010; Yang et al., 2017; Yaseen et al., 2015). The advantage of such data-driven modeling is that physical relationships and the physical parameters needed to describe the physical environment do not need to be explicitly defined; the machine learning algorithm approximates the relationship between model inputs and outputs through an iterative learning process (Solomatine and Ostfeld, 2008). Neural networks (NN) have been used to model and predict nonlinear time series data, such as the groundwater table, and have been found to perform as well as, and in some cases, better than physics-based models (Karandish and Šimůnek, 2016; Mohanty et al., 2013). Several studies have applied NN models on a daily or monthly time step to aquifers used for water supply in order to make forecasts appropriate for groundwater management (Chang et al., 2016a; Coulibaly et al., 2001; Daliakopoulos et al., 2005; Guzman et al., 2017; Nayak et al., 2006; Sahoo and Jha, 2013)[31–36]. Few studies, however, have used NNs for predicting the groundwater table in unconfined surficial coastal aquifers where flooding is a major concern and a finer time scale is needed to capture the impacts of storm events (Taormina et al., 2012).

Recurrent neural networks (RNNs) have been a popular choice for modeling groundwater time series data because they attempt to retain a memory of past network conditions. While RNNs have been successfully applied to groundwater modeling (Chang et al., 2016a; Coulibaly et al., 2001; Daliakopoulos et al., 2005; Guzman et al., 2017), it's been found that standard RNN architectures have difficulty capturing long term dependencies between variables (Bengio et al., 1994). This is due to two problems, (i) vanishing and (ii) exploding gradient, where weights in the network go to zero or become extremely large during model training. These two problems occur because the error signal can only be effectively backpropagated for a limited number of steps (Hochreiter and Schmidhuber, 1997).

One of the most successful approaches to avoiding the vanishing and exploding gradient problems has been the long short-term memory (LSTM) variant of standard RNNs (Hochreiter and Schmidhuber, 1997). LSTM is able to avoid these training problems by eliminating unnecessary information being passed to future model states, while retaining a memory of important past events. In the field of natural language processing, LSTM has become a popular choice of neural network because of its ability to retain context over long spans (Graves et al., 2013). LSTM has also been effective for financial time series prediction (Fischer and Krauss, 2018) and for short-term traffic and travel time predictions (Liu et al., 2017; Zhao et al., 2017) Despite the wide variety of applications, however, LSTM has only recently been used for hydrologic time series prediction (Hu et al., 2018; Liang et al., 2018). For example, LSTM was found to outperform two simpler RNN architectures for predicting streamflow (Tian et al., 2018). LSTM networks have also recently been used to model the groundwater table on a monthly time step in an inland agricultural area of China (Zhang et al., 2018b). This agriculture focused study provides valuable information on the advantages of LSTM for groundwater level prediction over a basic feed-forward neural network (FFNN), but only presents predictions for one time step ahead. In a real-time flood forecasting application, however, longer forecasts of the groundwater table at short time intervals would be needed (Taormina et al., 2012) and should include the use of forecast input data. LSTM models have yet to be evaluated for this type of application.

With the growing availability of large datasets and high performance computing, data-driven modeling techniques can now be evaluated for groundwater table forecasting. The objective of this study, therefore, is to compare RNN and LSTM neural networks for their ability to model and predict groundwater table changes in an unconfined coastal aquifer system with an emphasis on capturing groundwater table response to storm events. Based on prior research on this topic, it is expected that LSTM will outperform RNN for forecasting groundwater table levels. In this study, LSTM and RNN models were built for seven sites in Norfolk, Virginia USA, a flood prone coastal city. The models were trained and tested using observed groundwater table, sea level, and rainfall data as input. In addition to comparing RNN and LSTM neural networks, the effect of different training methods on model accuracy was evaluated by creating two unique datasets, one of the complete time series and one containing only periods identified as storms. The two types of datasets were bootstrapped and a statistical comparison of the two model types was made with t-tests to determine if differences in the results were significant. To ensure fair comparison, the hyperparameters of the RNN and LSTM networks were individually optimized with an advanced tuning technique instead of traditional ad-hoc methods. Once trained and evaluated, the RNN and LSTM models were tested with forecast sea level and rainfall input data to quantify the accuracy that could be expected in a real-time forecasting scenario.

2.2 Study Area

The City of Norfolk, Virginia is located on the southern portion of the Chesapeake Bay along the eastern coast of the United States (Figure 2.1, inset). The city covers 171 km2 of land with an average elevation of 3.2 m (above the North American Vertical Datum of 1988) and has 232 km of shoreline. Home to almost a quarter million people (USCB, 2018), Norfolk serves important economic and national security roles with one of the U.S.'s largest commercial ports, the world's largest naval base, and the North American Headquarters for the North Atlantic Treaty Organization (NATO). The larger Hampton Roads Region, of which Norfolk is a major part, has the second greatest risk from sea level rise in the U.S. and is surpassed only by New Orleans (Fears, 2012). This risk is partly due to coupled sea level rise and regional land subsidence from groundwater withdrawals from the deep Potomac Aquifer for water supply and glacial isostatic adjustment (Eggleston and Pope, 2013). Because of these and other factors, including low relief terrain and a regular hurricane season, the city and larger Hampton Roads region face increasingly frequent and severe recurrent flooding (Sweet and Park, 2014) which threatens its economic, military, and historic importance.

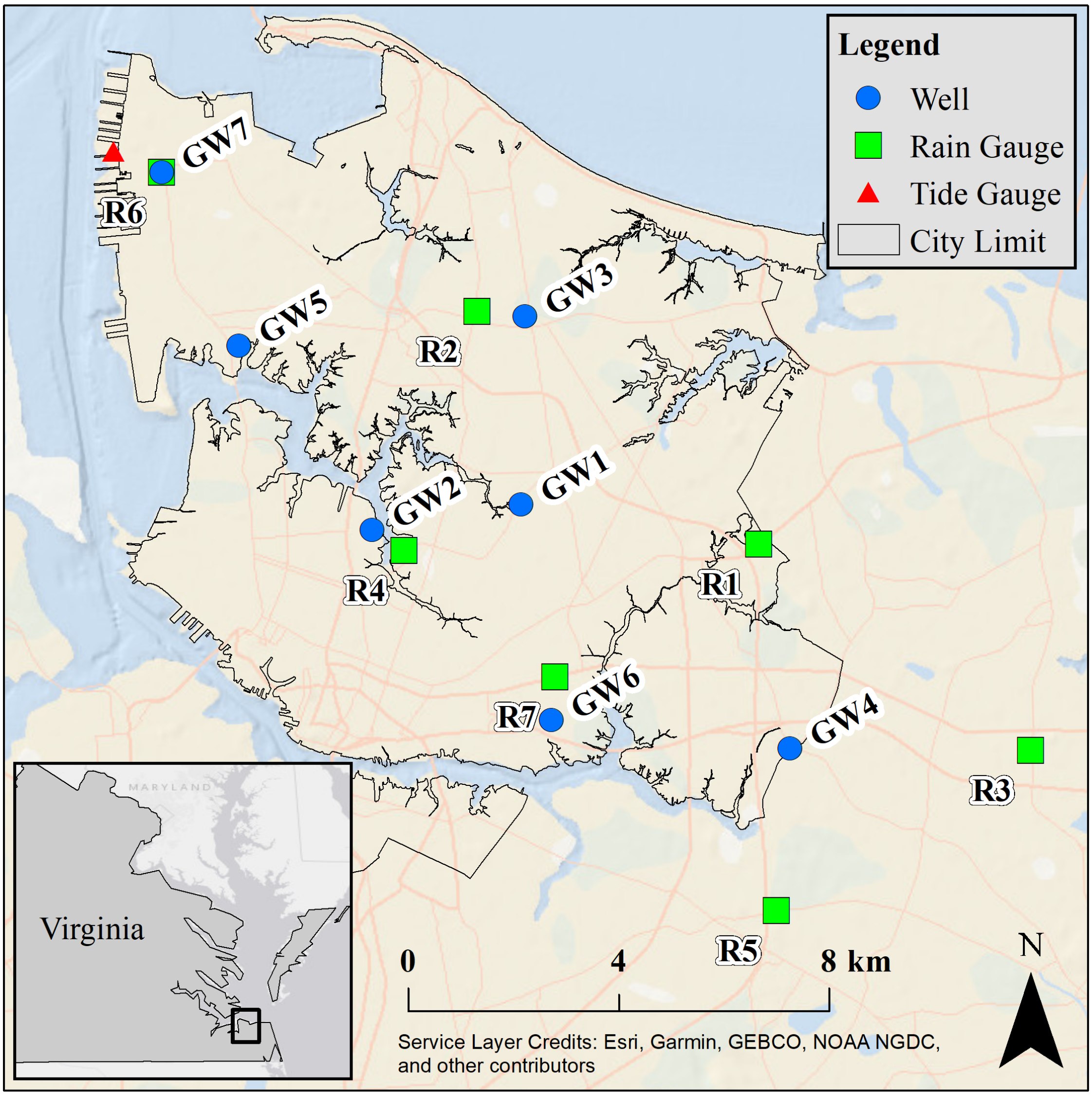

Figure 2.1: Location of gauges in Norfolk, Virginia.

2.3 Data

In order to predict groundwater table levels, the neural networks created in this study were trained and tested with the available groundwater table, rainfall, and sea level data as input. Input data was collected in two forms: observed and forecasted.

2.3.1 Observed Data

A unique dataset of groundwater table level observations for seven shallow monitoring wells in Norfolk was provided by the Hampton Roads Sanitation District (HRSD) (Figure 2.1, Table 2.1). Groundwater observations, in meters, are measured at a two minute time step and referenced to the North American Vertical Datum of 1988 (NAVD88). Observed rainfall data, in millimeters, also came from HRSD and was measured at a fifteen minute time step. Observed sea level data, in meters, was measured at a six minute time step, and is referenced to NAVD88. Sea level data came from the National Oceanic and Atmospheric Administration (NOAA) Sewells Point gauge (NOAA, 2018b). The mean, minimum, and maximum sea level at this station is 0.11 m, -0.98 m, and 1.88 m, respectively. All of the observed data are for the period between 1 January 2010 and 31 May 2018.

Table 2.1: Groundwater table monitoring well details.

Well ID	Land Surface Elevation (m)[a]	Well Depth (m)[b]	Distance to Tidal Water (m)	Impervious Area (%)[c]	Groundwater Table Level (m)[a,d]		
					Minimum	Maximum	Mean
GW1	2.21	4.27	36	27	-0.678	0.883	-0.102
GW2	1.24	4.08	32	23	-0.670	1.476	0.635
GW3	4.35	5.18	668	42	1.197	3.844	2.026
GW4	3.24	4.57	777	53	0.659	2.021	1.075
GW5	1.72	2.53	32	20	-0.167	1.5562	0.492
GW6	2.35	3.23	41	30	0.259	2.012	0.742
GW7	2.57	4.60	650	73	0.200	1.750	0.707

[a] Referenced to North American Vertical Datum of 1988 (NAVD88); [b] Below land surface;
[c] Percent of area classified as impervious within a 610m buffer around the well; [d] Statistics calculated from January 2010 to May 2018.

An examination of the observed data shows that each well has a different response to storm events (Figure 2.2). For instance, GW2 shows a large, rapid increase in the groundwater table from the first pulse of rainfall and GW4 shows more of a step response in the groundwater table to the three distinct pulses of rainfall. The groundwater level at GW6, however, shows a small, gradual increase in response to the storm event. While rainfall appears to be the main driver of groundwater table levels in all of these wells, sea level is also an important forcing factor which has a diminishing impact with increasing distance from the coast (Rotzoll and Fletcher, 2012; Freeze and Cherry, 1979).

2.3.2 Forecast Data

In order to simulate a real-time forecast scenario, archived forecast data were collected for three months: September, 2016, January, 2017, and May 2018. These months were selected because archived forecast data was available and had both dry periods and storm events. The storm events in the archived forecast data ranged from unnamed storms to Hurricane Hermine and Tropical Storm Julia, which has an estimated return period of 100–200 years, based on the 24 and 48 h rainfall (Smirnov et al., 2018). Forecast rainfall was generated by the High-Resolution Rapid Refresh (HRRR) model, a product of the National Center for Environmental Prediction (NCEP), which generates hourly forecasts of meteorological conditions, including total surface precipitation, for the coming 18 h with a resolution of 3 km2. These

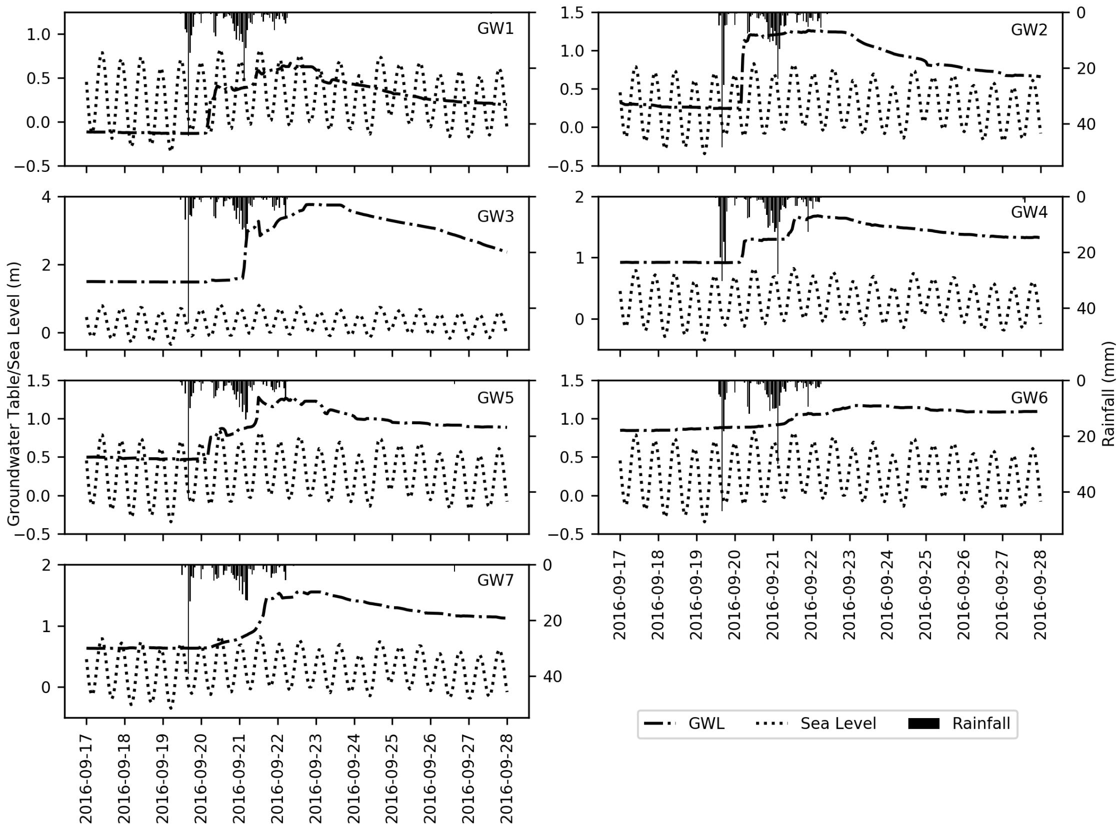

Figure 2.2: Hourly groundwater table level, sea level, and rainfall at individual wells for Tropical Storm Julia.

data are archived by the Center for High Performance Computing at the University of Utah (Blaylock et al., 2017) and was accessed from that database.

Forecast sea level data for the Sewells Point station was gathered from NOAA (NOAA, 2018d) for the same three months as the rainfall forecasts. These sea level data were downloaded at an hourly time step, and is referenced to NAVD88. The model used to generate sea level predictions at this station is based on the harmonic constituents of the observed tide cycle (NOAA, 2018c; NOAA, 2018a). While harmonic predictions can closely match the observed sea level under normal weather conditions, they do not include any storm surge effects.

2.4 Methods

This study was carried out through the workflow detailed in Figure 2.3. As such, this section is divided into three main subsections: Data preprocessing, neural network modeling, and results post-processing. Links to model code and data are given in the Supplemental Data section at the end of this article.

2.4.1 Input Data Preprocessing

Data preprocessing involves a number of steps for observed and forecast data (Figure 3). Raw groundwater table observations were filtered with a Hampel filter (MathWorks, 2015) to remove large erroneous values. This filter used the standard devi-

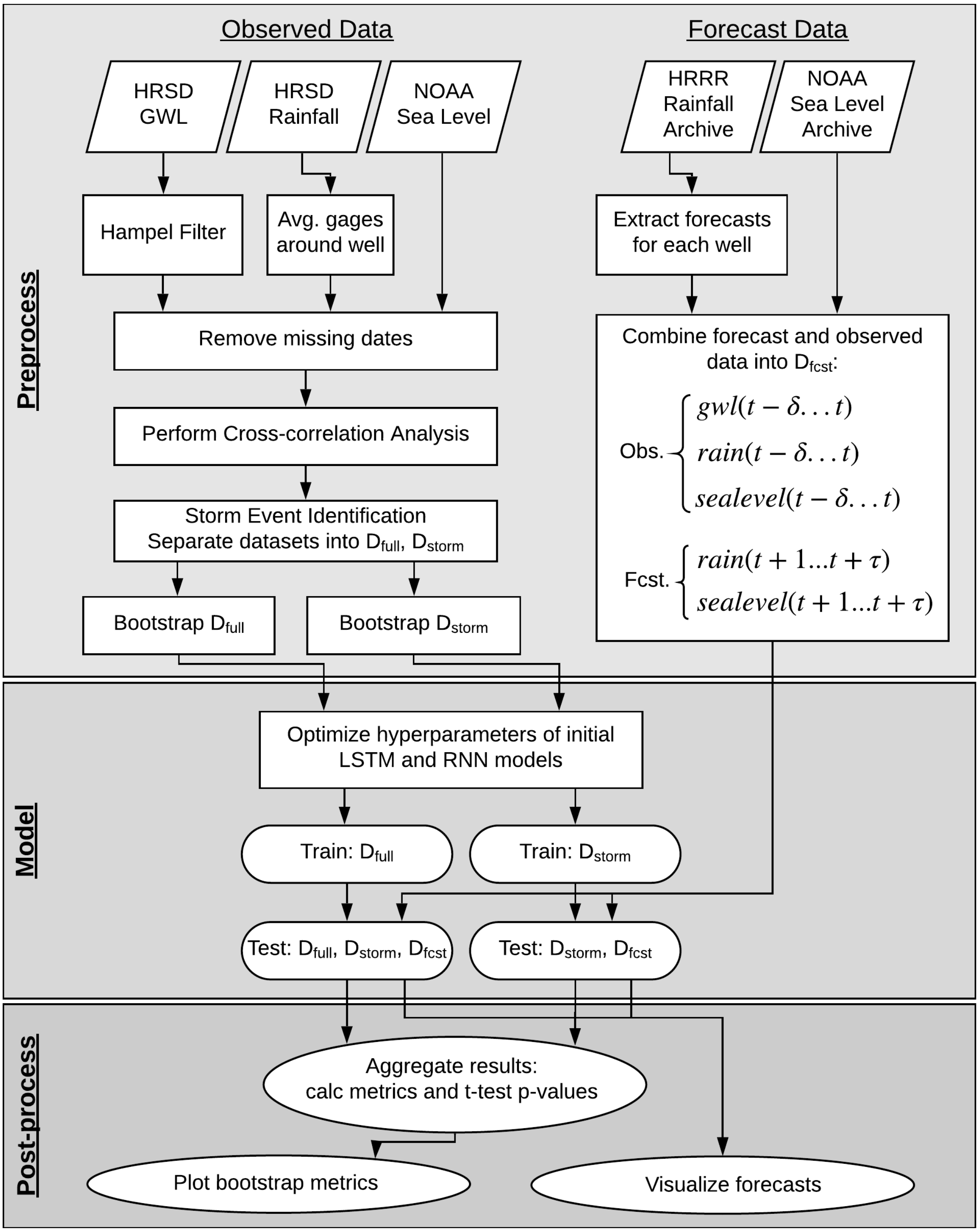

Figure 2.3: Study workflow detailing major steps in the data preprocessing, neural network modeling, and results post-processing.

ation of the observations within a single day (720 two minute observations) rolling window as a threshold; any observations greater than the threshold were replaced by the rolling median. All of the raw observed data were aggregated to an hourly time step to match the format of the forecast data. Groundwater table and sea levels were aggregated using the hourly mean value and rainfall is the cumulative hourly total. Because some wells did have several months of missing data, any time steps with any missing values were removed. For wells without an immediately adjacent rain gauge, the rainfall at the well was assumed to be the mean of the surrounding rain gauges (Table 2.2).

Table 2.2: Rain gauges associated with each well based on geographic proximity.

Well ID	Rain Gauge (s)
GW1	R1
	R2
	R4
	R7
GW2	R4
GW3	R2
GW4	R1
	R3
	R5
	R7
GW5	R2
	R6
GW6	R7
GW7	R6

To prepare the filtered and continuous data as model input, the time series of each variable (groundwater table, sea level, and rainfall) was shifted to include relevant past observations, based on an appropriate lag δ, and observations up to the forecast horizon τ (18 h in this study to correspond to the HRRR model forecast horizon). Lags for each well represent the delay between a rainfall or sea level observation and the corresponding response of the groundwater table and were identified by cross-correlation analysis (see Section 3.1.1). After shifting the time series of each variable, all data were normalized to values between 0–1 and combined into an input matrix or tensor and a label tensor. Each row in the input tensor contains three vectors: Groundwater table gwl_I, rainfall $rain$, and sea level sea. Each row in the label tensor is a vector of groundwater table values gwl_L to be predicted (Table 2.3).

Table 2.3: Input and label tensors for neural network modeling.

Inputs	Labels
$gwl_I = \{t - \delta ... t\}$	
$rain = \{t - \delta ... t + \tau\}$	$gwl_L = \{t + 1 ... t + \tau\}$
$sea = \{t - \delta ... t + \tau\}$	

Preprocessing of forecast data, which is retrieved at an hourly time step, consists of two steps (Figure 2.3). First, the time series of HRRR rainfall data, which is a gridded product over the continental United States, has to be extracted for the coordinates of each well. Second, the forecast data have to be inserted into the correct locations in the input tensor. Specifically, the observed rainfall and sea level data in columns (t+1) to (t+τ) has to be replaced with the corresponding forecast data. This creates a dataset D_{fcst} that includes both observed and forecast data as specified in Figure 2.3. The same normalization from 0–1 used for the observed data was applied to the forecast data.

2.4.2 Input Variable Cross-Correlation Analysis

Parsing the relationships between a rainfall or sea level observation and the corresponding groundwater table response is a crucial component of input data preprocessing. This response time is called the lag δ and can be separated into two components: δ_R between rainfall and groundwater table response and δ_S between sea level and groundwater table response. The appropriate δ_R and δ_S, in hours, for each well was approximated by a cross correlation analysis (Maier and Dandy, 2000). This process involves shifting one signal in relation to the other until a rainfall or sea level observation lines up with its corresponding groundwater table response. The highest cross correlation value (CCF) corresponds to the most influential δ_R or δ_S.

2.4.3 Storm Event Response Identification

In order to evaluate the performance of RNN and LSTM models for groundwater table forecasting during storm events, two training datasets were used (Figure 3). The first training set D_{full} represents the continuous time series data and includes both dry and wet days. The second training set D_{storm} consists only of time periods that were identified as storm events. D_{storm} was created through a filtering process using the gradient and peaks of the observed groundwater table values. For any storm event, the starting time of the event was based on locating the local maxima of the gradient of the groundwater table and looking backward in time to the first occurrence of zero gradient. A peak finding algorithm (SciPy, 2019a) was then used to locate the peak of the groundwater table that occurred after the corresponding starting time; peak values were used as the end point of the storm.

2.4.4 Bootstrapping Datasets

Bootstrapping was used to generate many datasets with characteristics similar to the observed datasets. While bootstrapping is generally done by selecting values at random and combining them into a new dataset, special techniques are needed to preserve the dependence in time series data. In order to bootstrap the D_{full} datasets in a manner appropriate for time series data, circular block bootstrapping with replacement was used (Rohilla Shalizi, 2018). The block size was based on the average storm length found when creating the storm datasets for each well. Because the D_{storm} datasets were already separated into blocks of different time periods, they were bootstrapped by randomly sampling the blocks with replacement. By creating one thousand bootstrap replicates of each dataset, a normal distribution of error can be approximated when the models are trained and tested. The first 70% of each bootstrapped dataset was taken as the training data and the remaining 30% was used as the test set.

2.4.5 Recurrent Neural Networks

RNNs (Elman, 1990) have been specifically designed to capture the structure that is often inherent in time series data. They do this by passing the output, or state, of the hidden layer neurons, which represent what has been learned at the previous time steps, as an additional input to the next time step (Figure 2.4A). RNN training was done with back-propagation through time (BPTT) (Werbos, 1990), or some variant, to adjust network weights based on the error gradient with respect to both the network weights and the previous hidden states. Because gradients can change exponentially during this process, they tend to either vanish or explode. In this study, a fully connected RNN (Chollet, 2015) was used and the output was calculated by stacking a fully connected layer on top of the RNN cell. The product of the output layer is the groundwater table level for the forecast horizon τ. The RNN calculations can be written as:

$$h_t = tanh(Wx_t + Uh_{t-1} + b) \tag{2.1}$$

$$y_t = Vh_t + b \tag{2.2}$$

where ht is the hidden state, yt is the output, and xt is the input vector. The input, hidden, and output weights are represented by W, U, and V, respectively, and b is the bias. The hyperbolic tangent activation function is noted as $tanh$.

2.4.6 Long Short-term Memory Neural Networks

LSTM neural networks are a type of RNN that were developed to overcome the vanishing and exploding gradient obstacles of traditional RNNs (Hochreiter and Schmidhuber, 1997). The LSTM architecture (Figure 4B) minimizes gradient problems by enforcing constant error flow between hidden cell states, without passing through an activation function. In addition to this constant error path, an LSTM cell contains three multiplicative units known as gates: The forget gate, the input gate, and the output gate. Because each gate acts as a neuron, it can learn what inputs and cell states are important for predicting the output through the process of passing inputs forward, back propagating the error, and adjusting the weights. The processes within the LSTM cell can be represented with the following equations:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{2.3}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{2.4}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{2.5}$$

$$C'_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{2.6}$$

$$C_t = f_t \circ C_{t-1} + i_t \circ C'_t \tag{2.7}$$

$$h_t = tanh(C_t) \circ o_t \tag{2.8}$$

$$y_t = Vh_t + b \tag{2.9}$$

where f_t, i_t, and o_t represent the forget, input, and output gates, respectively. The new cell state candidate values and updated cell state are represented by C'_t and C_t, respectively. Element-wise multiplication of vectors is represented by $\circ$ and the sigmoid activation function is noted as σ. While studies have experimented with different gate configurations, significant improvements over the standard configuration were not found (Greff et al., 2017). This study uses LSTM cells with three gates (Chollet, 2015). The network output was calculated by stacking a fully connected layer on top of the LSTM cell. The product of the output layer is the groundwater table level for forecast horizon τ.

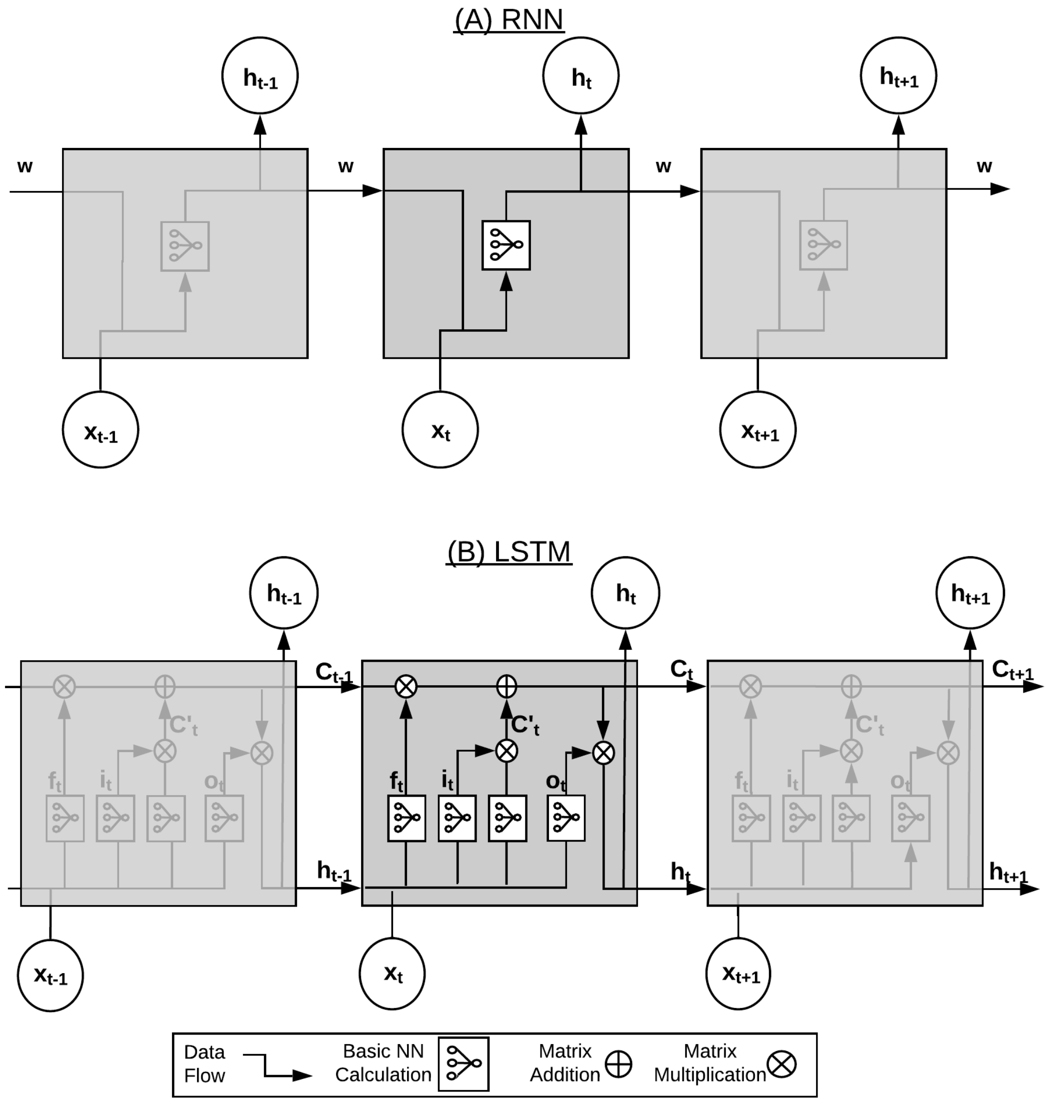

Figure 2.4: Recurrent neural network (RNN) (A) and long short-term memory (LSTM) (B) model architectures. Merging lines show concatenation and splitting lines represent copies of matrices being sent to different locations.

2.4.7 Hyperparameter Tuning

Hyperparameter tuning has traditionally been done in an ad-hoc manner through manual trial and error or random search (Govindaraju, 2000b; Maier and Dandy, 2000; Maier et al., 2010). This type of tuning can be efficient, but is hard to reproduce or compare fairly (Bergstra et al., 2013a); with the increasing complexity of network architectures, more formal methods of hyperparameter optimization are also emerging. In this study, tuning was accomplished for each model type and for each well using a sequential model-based optimization (SMBO) search with the tree-structured Parzen estimator (TPE) algorithm, a Bayesian optimization approach (Bergstra et al., 2013b). Given the search history of parameter values and model loss, TPE suggests hyperparameter values for the next trial which are expected to improve the model loss (reduce root mean squared error (RMSE), in this case). As the number of trials increases, the search history grows and the hyperparameter values chosen become better. The Hyperas library (Pumperla, 2015) implements the SMBO/TPE technique and was used in this study to advance what has been done in previous research. For example, when comparing four types of neural networks, Zhang et al. 2018 simply stated that a trial and error procedure was used to select the best network architecture. When predicting groundwater levels, Zhang et al. 2018 presented results for a trial and error optimization of LSTM hyperparameters, but then state that the same hyperparameters were used for the much simpler architecture of FFNN models. By not optimizing the hyperparameters of the FFNN it is more difficult to draw comparisons with the LSTM. Optimizing the hyperparameters of both the LSTM and RNN models in this study allowed each model the best chance to perform well. The hyperparameters tuned for each model in this study were the number of neurons, the activation function, the optimization function, the learning rate, and the dropout rate (Table 2.4). The number of neurons influences the model's ability to fit a complex function. The dropout rate helps prevent overfitting by randomly dropping some connections between neurons during training (Srivastava et al., 2014). A minimum value of 10% ensures some dropout is used, as the natural tendency would be for models to not have any connections dropped during training. The combination of hyperparameters for each model type that resulted in the lowest RMSE, based on 100 trials, was used in the final models.

Table 2.4: Hyperparameter choices explored.

Hyperparameter	Type	Options Explored
Number of Neurons	Choice	10, 15, 20, 40, 50, 75
Activation Function	Choice	Rectified Linear Unit (relu), Hyperbolic tangent (tanh), Sigmoid
Optimization Function	Choice	Adam, Stochastic Gradient Descent (SGD), Root Mean Square Propagation (RMSProp)
Learning Rate	Choice	$10^{-3}, 10^{-2}, 10^{-1}$
Dropout Rate	Continuous	0.1-0.5

2.4.8 Model Training and Evaluation

All the models for this study were built with the Keras deep learning library for Python (Chollet, 2015) using the Tensorflow backend (Abadi et al., 2016a). Model training was carried out on the Rivanna HPC at the University of Virginia using either one NVIDIA Tesla K80 or P100 graphical processing unit (GPU), depending on which was available at the time of execution (Figure 3). RNN and LSTM models were trained for each well using each of the one thousand bootstrap datasets for both the D_{full} and the D_{storm} datasets (Figure 2.5). At each time step, models were fed input data and output a vector of forecast groundwater table levels, as shown in Table 3. During training, the models sought to minimize the cost function, which is the RMSE between predicted and observed values, by iteratively adjusting the network weights. After training, the D_{full} models were tested on the D_{full}, D_{storm}, and D_{fcst} test sets. Likewise, the D_{storm} models were tested on the D_{storm} and D_{fcst} test sets.

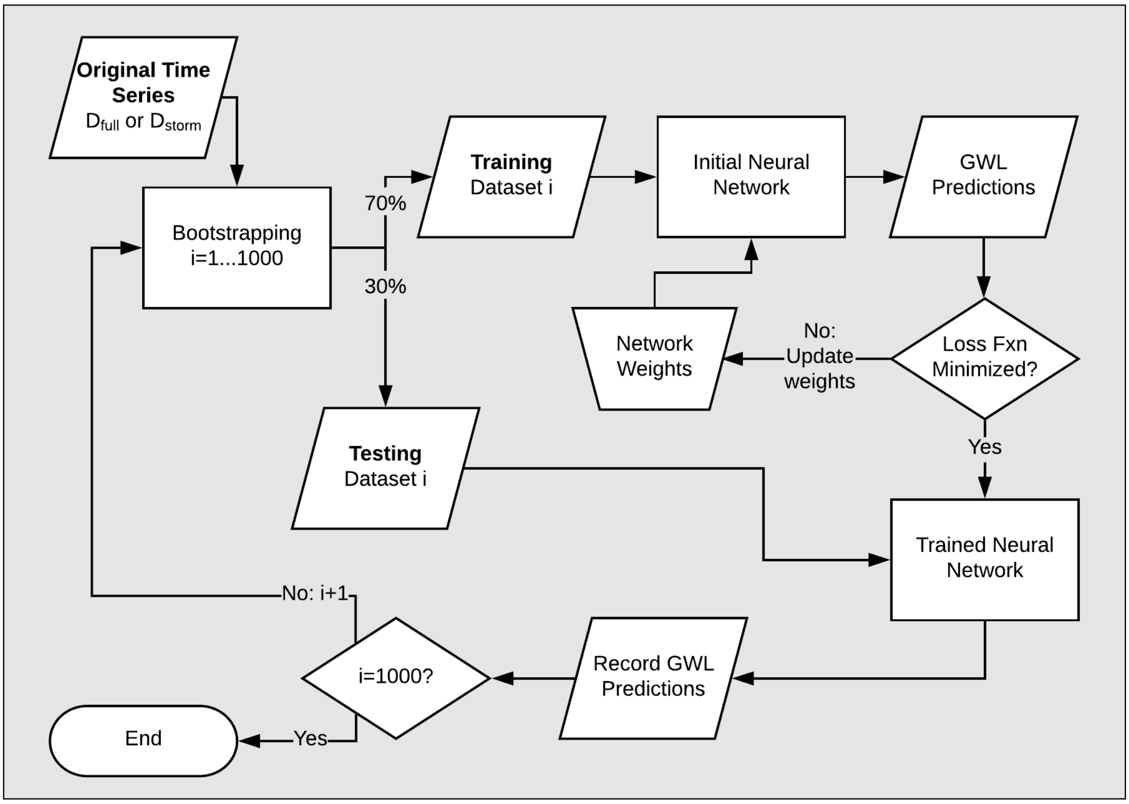

Figure 2.5: Model training and evaluation with bootstrapped datasets.

Besides being the training cost function, RMSE was also the main metric used for model evaluation. Additionally, the mean absolute error (MAE) was also calculated. Values approaching zero are preferred for both metrics. Both RMSE and MAE were calculated by comparing the predicted water table level (18 predictions at each time step) to the observed values for the corresponding time periods. To help prevent overfitting and increase the ability of models to generalize, early stopping was used in addition to dropout. Early stopping ends the training process once the cost function has failed to decrease by a threshold value after 5 epochs.

2.4.9 Results Post-Processing

Results post-processing consisted mainly of aggregating model predictions and RMSE values, performing t-tests for model comparison, and visualization (Figure 2.3). Before these actions, however, all predicted values were post-processed to cap predicted groundwater table levels at the land surface elevation for each well.

A number of hypotheses were formulated to test the effects of model type and training dataset on forecast accuracy (Table 2.5). For example, it was hypothesized that LSTM models would have a lower mean RMSE than RNN models when trained and tested with the D_{full} dataset (Table 2.5, Comparison ID A). The hypotheses were evaluated using t-tests to evaluate whether or not there was a statistically significant difference between the mean of the 1000 RMSEs between two models

(SciPy, 2019b). In order to reject a null hypothesis that the two models have identical average values, the p-value from the t-test would need to be significant (less than 0.01).

Table 2.5: t-test null hypotheses for model type and training data comparison.

Comparison ID	Null Hypothesis	Testing Data
A	RMSE(LSTM, D_{full}) = RMSE(RNN, D_{full})	D_{full}
B	RMSE(LSTM, D_{storm}) = RMSE(RNN, D_{storm})	
C	RMSE(RNN, D_{storm}) = RMSE(RNN, D_{full})	D_{storm}
D	RMSE(LSTM, D_{storm}) = RMSE(LSTM, D_{full})	
E	RMSE(LSTM, D_{full}) = RMSE(RNN, D_{full})	
F	RMSE(LSTM, D_{storm}) = RMSE(RNN, D_{storm})	D_{fcst}
G	RMSE(RNN, D_{storm}) = RMSE(RNN, D_{full})	
H	RMSE(LSTM, D_{storm}) = RMSE(LSTM, D_{full})	

2.5 Results

The results of this study are divided into two subsections. The first subsection, data preprocessing results, describes the findings of the cross correlation analysis, the storm event identification, and the hyperparameter tuning for each well and model type. The second subsection, model results, describes the model performance and the statistical evaluation of differences between models and training data types. This subsection concludes with a visualization of model predictions.

2.5.1 Data Preprocessing Results

Input Variable Cross-Correlation Analysis

Using cross correlation analysis, appropriate median lags δ for the entire period of record were found for each well (Table 2.6). Rainfall lags δ_R were generally expected to increase with a greater distance between the land surface and the groundwater table. It was found δ_R did increase with greater depth to the groundwater table when GW2 and GW3 were compared. At GW2, δ_R was 26 h and the mean groundwater table depth was 0.61 m (Table 2.1) while at GW3 δ_R was 59 h and the mean groundwater table depth was 2.32 m. At the other wells, however, this relationship did not hold. For example, GW1 had the same δ_R as GW2, but the mean groundwater table depth was very similar to that of GW3 (2.31 m). Other characteristics that influence infiltration rate, such as vertical hydraulic conductivity, porosity, impermeable surfaces, or the configuration of the stormwater system appear to have had a large effect on δ_R at these wells. In addition, sea level may also be influencing groundwater table levels at some or all of these wells.

The impact of sea level lags δ_S on the groundwater table was more difficult to determine than rainfall lags δ_R, indicating that sea level does not have as much impact on certain wells; there did not seem to be clear correlations for GW3, GW5, or GW6. It was expected that the impact of sea level would decrease with greater distance between a given well and the closest tidal waterbody influencing it. However, this did not seem to have a strong relationship. GW4, for example, was the farthest

Table 2.6: Rainfall δ_R and sea level δ_S lags found for each well.

Well ID	δ_R (h)	δ_S (h)
GW1	26	19
GW2	26	18
GW3	59	-
GW4	25	17
GW5	28	-
GW6	48	-
GW7	58	51

well from a tidal water body but had the shortest δ_SS, suggesting that tidal water may have a more direct route to this location. While a strong correlation between sea level and groundwater table was not found for three wells, it was deemed that sea level could still be an important input variable for models at those wells because of their proximity to tidal water bodies (Yoon et al., 2011; Moss and Marani, 2016). In order to keep the data preprocessing consistent, and because δ_S values could not be found for all wells and the δ_S values found were always shorter than δ_R values, δ_R was taken as the lag value for all input variables.

Storm Event Response Identification

The storm identification process produced a unique dataset and a different average storm duration and total number of events for each well (Table 2.7). Average storm duration, the average length in hours of the identified periods, was used as the block size for bootstrapping the D_{full} datasets. The storm events identified for each well also accounted for the majority of total rainfall, indicating that the method is capturing large rainfall events. Storm surge is also being captured at most wells as shown by the positive increase in mean sea level for the storm events compared to the D_{full} datasets (Table 2.7). Figure 2.6 shows an example of storms found with this process; large responses of the groundwater table are captured, but smaller responses are excluded.

Table 2.7: Storm characteristics for each well.

Well ID	Average Storm Duration (h)	Number of Events	% of Total Rain	% Increase in Mean Sea Level Over D_{full}
GW1	83	239	75	27
GW2	82	307	85	36
GW3	137	155	57	18
GW4	89	254	67	18
GW5	91	149	60	64
GW6	120	295	60	0
GW7	132	166	63	0

Hyperparameter Tuning

Tuned hyperparameters were generally consistent across wells and model types (Tables 2.8 and 2.9). Dropout rates ranged from just above the minimum of 0.1 to a

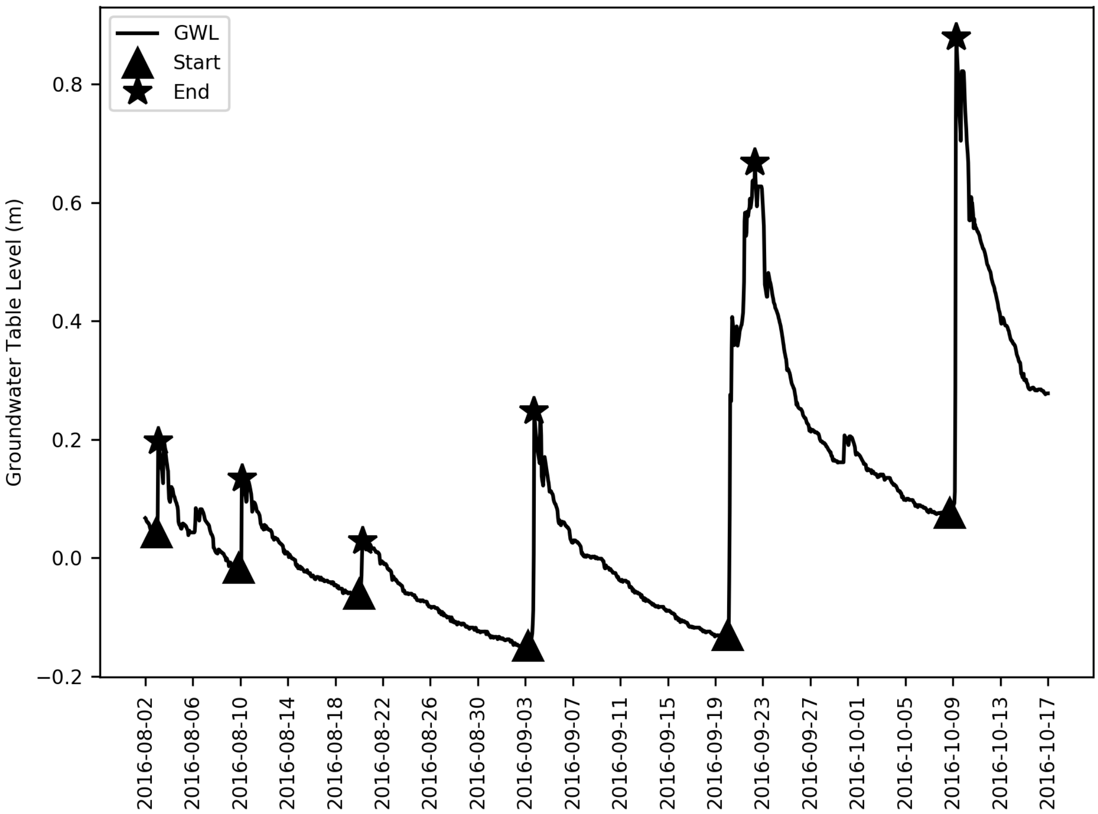

Figure 2.6: Detail of identified storm periods found for well GW1.

high of 0.355. The preferred activation function was the hyperbolic tangent, except for the GW5 RNN. In all cases the Adam optimization function performed the best with its recommended learning rate of 10^{-3} (Kingma and Ba, 2014). The largest number of neurons possible (75) was used in five of the seven RNN (Table 2.8) and LSTM (Table 2.9) models. The other models of each type used a mid-range number of neurons (40 or 50).

Table 2.8: Tuned hyperparameters for RNN models.

Well	Dropout Rate	Activation Function	Optimization Function	Learning Rate	Neurons
GW1	0.126	tanh	adam	10^{-3}	40
GW2	0.340	tanh	adam	10^{-3}	75
GW3	0.320	tanh	adam	10^{-3}	75
GW4	0.111	tanh	adam	10^{-3}	75
GW5	0.127	relu	adam	10^{-3}	75
GW6	0.145	tanh	adam	10^{-3}	75
GW7	0.104	tanh	adam	10^{-3}	40

Table 2.9: Tuned hyperparameters for LSTM models.

Well	Dropout Rate	Activation Function	Optimization Function	Learning Rate	Neurons
GW1	0.355	tanh	adam	10^{-3}	75
GW2	0.106	tanh	adam	10^{-3}	40
GW3	0.166	tanh	adam	10^{-3}	75
GW4	0.102	tanh	adam	10^{-3}	75
GW5	0.103	tanh	adam	10^{-3}	50
GW6	0.251	tanh	adam	10^{-3}	75
GW7	0.177	tanh	adam	10^{-3}	75

2.5.2 Model Results

Network and Training Data Type Comparison

The results in this subsection address hypotheses A–D (Table 5), which compare performance of the two model types trained using the two different datasets. All of these comparisons had significant p-values (<0.001). This shows that the null hypotheses that two models have identical average values was rejected and there are significant differences in RMSE for different model types and training datasets. The distributions of RMSE values for all bootstrap models in this subsection is available in Appendix A; corresponding MAE values are available in Appendix C.

When trained with either D_{full} or D_{storm}, LSTM models have lower mean RMSE values than RNN models (Figure 2.7A,B), as hypothesized (Table 5, A and B). LSTM models trained and tested with D_{full} had average RMSE values that were lower than RNN models by 49%, 38%, and 18% for the t+1, t+9, and t+18 predictions, respectively. LSTM's advantage over RNN decreased as the prediction horizon increased. Similarly, LSTM models trained and tested with D_{storm} had lower average RMSE values than RNN models by 50%, 55%, and 36% for the t+1, t+9, and t+18 predictions when tested on D_{storm}, respectively.

When tested with D_{storm}, the models trained with D_{storm} outperformed the models trained with D_{full} (Figure 7C,D), with the exception of the RNN for GW4. In this scenario, the models trained with D_{storm} had RMSE values that were lower than models trained with D_{full} by an average of 33%, 39%, and 42% for the RNN models and by an average of 40%, 58%, and 56% for the LSTM models for the t+1, t+9, and t+18 predictions, respectively. The improvement in performance when using D_{storm} as opposed to D_{full}, increased with the prediction horizon. While this was true for both model types, the performance improvement for LSTM was greater than for the RNN. In most cases the model error increased as the prediction horizon increased. This held for all of the LSTM models, but not with the RNN at GW4 and GW6 for certain datasets. For example, the RNN trained and tested on D_{storm} (Figure 7B,C) had a larger RMSE for the t+9 prediction than the t+18 prediction. This pattern is the same for the GW6 RNN (Figure 7A,C) and may have been caused by some combination of hyperparameters and/or some unknown error in the dataset. Causes of individual errors in these types of models, however, are very difficult to pinpoint (Maier and Dandy, 2000).

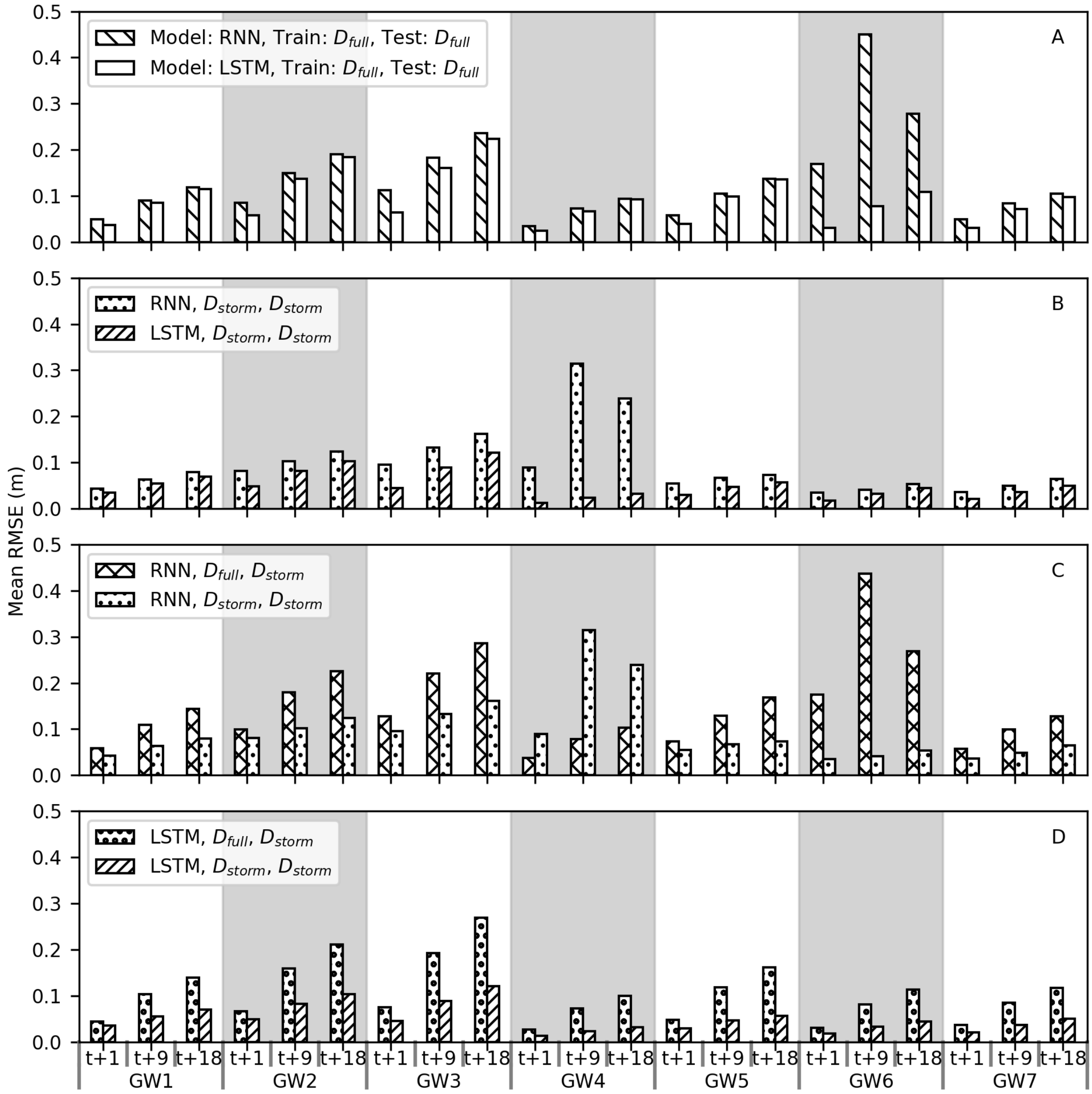

Figure 2.7: Mean root mean squared error (RMSE) values for each model type and training dataset treatment at each well and forecast period. Subplot letters correspond to the hypothesis being tested (Table 5) and are comparisons of (A) RNN and LSTM models trained and tested with D_{full} (B) RNN and LSTM models trained and tested with D_{storm} (C) RNN models trained with either D_{full} or D_{storm} and tested on D_{storm} (D) LSTM models trained with either D_{full} or D_{storm} and tested on D_{storm}.

Real-time Forecast Scenario

By training and testing models with observed data, comparisons can be made between model types and training datasets in terms of performance (as shown in Figure 2.8). The performance of these models, however, also needs to be evaluated in a real-time scenario that includes forecast conditions of rainfall and sea level level. The mean RMSE values from testing the models and data treatments with the D_{fcst} test set are shown in Figure 8 and correspond to hypotheses E–H (Table 5). The distributions of RMSE values for all bootstrap models in this subsection is available in Appendix B; corresponding MAE values are available in Appendix C.

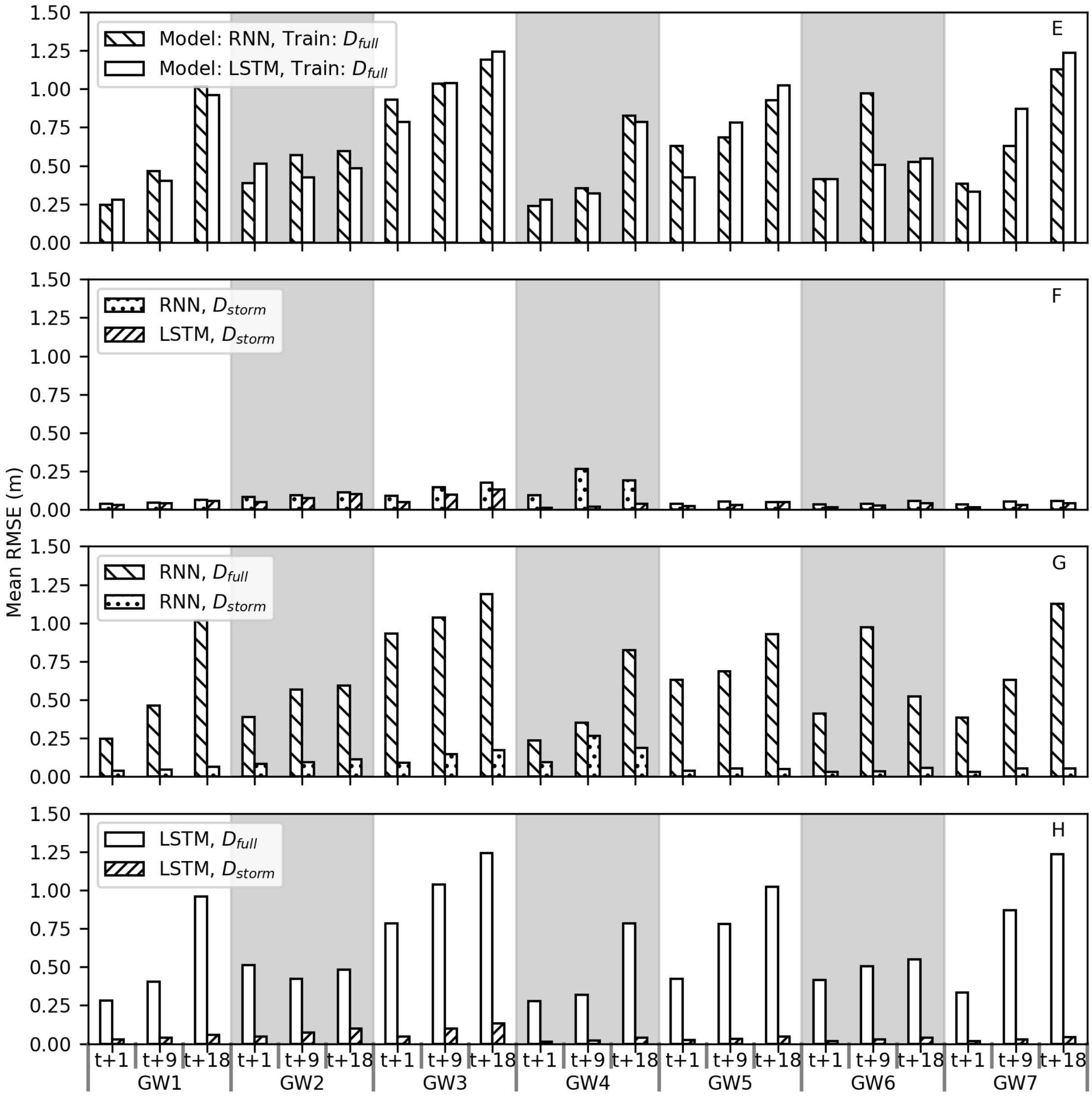

Figure 2.8: Mean RMSE values from the forecast test set D_{fcst} for each model type and training dataset treatment at each well and forecast period. Subplot letters correspond to the hypothesis being tested (Table 5) and are comparisons of (E) RNN and LSTM models trained with D_{full} (F) RNN and LSTM models trained with D_{storm} (G) RNN models trained with either D_{full} or D_{storm} (H) LSTM models trained with either D_{full} or D_{storm}.

In the real-time use simulation, models trained on D_{storm} (Figure 2.8F–H) performed much better than those trained with D_{full} (Figure 8E), which had RMSE values of up to nearly 1.25 m. In contrast to the difference training data type made, model architecture only made a small difference in performance (Figure 8E,F). All differences seen in Figure 8E were statistically significant at the 0.001 level, except GW3 at t+9 and GW6 at t+1 where the results were almost identical. The comparisons in Figure 8F–H all had significant p-values.

Visualizations from the real-time forecasting scenario (Figure 2.9) complement the aggregate metrics from bootstrap testing of models and training data treatments and demonstrate the response of predicted groundwater table levels to a storm when using D_{fcst} as input data. The forecasts at GW1 are shown in Figure 9 for Tropical

Storm Julia, which impacted Norfolk in late September of 2016. The initial rainfall from this storm on the 19th caused the groundwater table to spike early on the 20th. Subsequent rainfall on the 20th, 21st, and 22nd maintained the elevated groundwater table level. The LSTM model trained with D_{full} has greatly increasing error as the forecast horizon grows (Figure 9 t+1, t+9, t+18) and tends to be overly impacted by sea level fluctuations. In contrast, the predicted groundwater table level from the LSTM model trained with D_{storm} has much better agreement with the observed groundwater table levels, even as the forecast horizon increases.

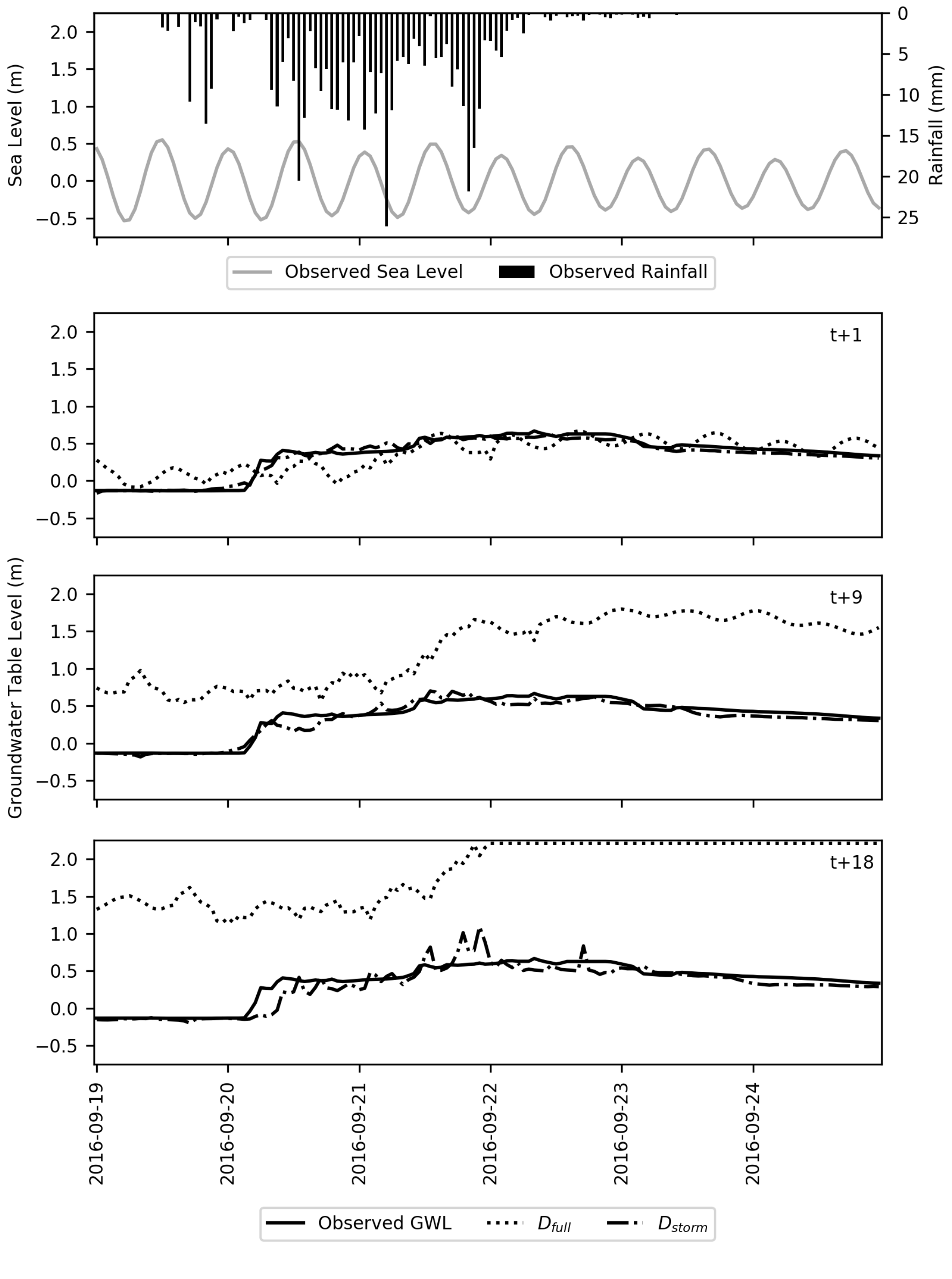

Figure 2.9: Comparison of groundwater table observations and forecasts at GW1 from LSTM models trained with the D_{full} and D_{storm} training sets.

2.6 Discussion

The results of hypothesis testing (Table 5) indicate that both model type and the training data influenced the accuracy of groundwater table forecasts. The LSTM architecture was better able to learn the relationships between groundwater table, rainfall, and sea level than the simpler RNN. Additionally, models trained with storm data D_{storm} outperformed models trained with the full dataset D_{full} when tested on either observed for forecast data. In the real-time scenario one reason for this difference in performance could be the structure of the test set D_{fcst}. These results indicate that the structure of the time series data in D_{storm} and D_{fcst} are more closely aligned, as opposed to the time series structure of D_{full} and D_{fcst}. The models trained on D_{full} also have to learn groundwater table response with many observations where no rainfall occurred. In contrast, models trained on D_{storm}, which have a higher proportion of observations with rainfall, may have a clearer pattern to learn.

In the real-time forecasting scenario, both RNN and LSTM models trained with D_{storm} demonstrated predictive skill, forecasting groundwater table levels with low RMSE values (Figure 8F). Models trained with D_{full} however performed much worse because of the noisier signal they had to learn (Figure 9) and are not suitable for use in a real-time forecasting scenario. Across all wells, averaged RMSE values for the RNN models were 0.06 m, 0.1 m, and 0.1 m for the t+1, t+9, and t+18 predictions, respectively. Averaged RMSE values for the LSTMs were slightly lower at 0.03 m, 0.05 m, and 0.07 m for the t+1, t+9, and t+18 predictions, respectively. While there is limited research on the use of LSTMs for forecasting groundwater table, these results are comparable with the work of J. Zhang et al. (2018), who reported RMSE values for one-step ahead prediction of monthly groundwater table at six sites ranging from 0.07 m to 0.18 m. The current work makes advances by showing that both LSTM and RNN can accurately forecast groundwater table response to storm events at an hourly time step, with forecast input data, and at longer prediction horizons all of which are necessary in a coastal urban environment.

Because the effect of sea level on the groundwater table is heavily dependent on well location and soil characteristics not included in this study, a sensitivity analysis was performed by removing sea level from the D_{full} and D_{storm} data sets and retraining and retesting the models. Of the wells that were not correlated with sea level, GW3 and GW6 performed better without sea level data. Using RNN models trained with D_{full}, there was an average decrease in RMSE of 12% for GW3 and 41% for GW6. The only exception to this is the GW6 RNN trained with D_{storm} which performed much worse without sea level. For LSTM models trained with D_{full} however, there was only a 3% decrease in RMSE for GW3 and a 2% decrease for GW6. The third well that was not correlated with sea level, GW5, was worse without sea level for the RNN trained with D_{full}; the average increase in RMSE was 17%. Removing sea level at this well had no change in RMSE for the LSTM models trained with D_{full}. This particular well is only 32 m from the coast so the influence of sea level seems reasonable. When models were trained with D_{storm} excluding sea level, across all well there was an average increase in RMSE of 8% for RNN models and no change for LSTM models. This demonstrates that sea level data is important for groundwater table prediction during storms for wells close to the coast and this is captured effectively by the D_{storm} datasets (Table 7). This analysis indicated that

RNN models were much more sensitive to the inputs used than LSTM models. As designed, the structure of LSTM models allowed them to filter out noisy data and have little to no change in RMSE if sea level data was removed, especially when using the best performing combination of LSTM and D_{storm} training data.

The results of this study illustrate the trade-off between model complexity and performance that has implications beyond creating forecasts. The increased complexity of LSTM models, in terms of gates that learn and the constant error pathway, allowed them to have more predictive skill than the RNN models for forecasting groundwater table response to storm events. Additionally, the structure of LSTM models allowed them to filter out noise from the sea level signal which RNN struggled to do. Most of the comparisons presented in the Results had significant p-values; because of the large sample size (1000) however, even a very small difference in RMSE values between two models was considered significant. For example, the differences between LSTM and RNN models trained with D_{storm} in the real-time forecasting scenario were statistically significant (Figure 8F). The average difference in the RNN and LSTM RMSE values, however, was only 0.03 m, 0.05 m, and 0.03 m for the t+1, t+9, and t+18 predictions, respectively. If these groundwater table forecasts were to be used as additional input to a rainfall-runoff model to predict flooding, it seems unlikely that the small differences between RNN and LSTM models would have a large impact, especially when compared to other factors like rainfall variability and storm surge timing.

The increased complexity of the LSTM models, while they had better performance than the RNN models, also increased their computational cost. The main difference in computational cost of the LSTM and RNN in this study was the length of training time. When trained on an HPC with either an NVIDIA Tesla K80 or P100 GPU or a smaller NVIDIA Quadro P2000 GPU on a desktop computer, wall-clock training time for LSTM models was approximately three times that of RNN models. Factors in training time include hyperparameters, such as the number of neurons in the hidden layer, which were relatively similar between model types. Once models are trained, groundwater table forecasts are obtained by a forward pass of input data through the network; this time was short and comparable for both models. For this groundwater table forecasting application training time was not a major concern, but if the application was time sensitive and the models were frequently retrained, RNNs could be an appropriate choice that does not sacrifice much in terms of accuracy.

Because forecast data were used as model input in the real-time scenario, it's important to note some of the uncertainties that dataset might introduce. HRRR rainfall data are a product of a numerical forecast model and as such is subject to the uncertainty of that model, which includes the transformation of radar reflectivity data into precipitation amounts (Krajewski and Smith, 2002). Additionally, the uncertainty of HRRR forecasts will increase the farther into the future they are. NOAA sea level forecasts, as previously mentioned, are based only on the harmonic constituents of the astronomical tide cycle. For rainfall-dominated storm events this type of forecast may be accurate enough as a model input, but any storm surge from hurricanes or nor'easters would not be included. This could result in under prediction of groundwater table levels. While archived storm surge predictions were not available for this study, in a real scenario predictions of storm surge could be incorporated into the model input.

The neural networks and data processing techniques presented in this paper are applicable to other coastal cities facing sea level rise and recurrent flooding. Because there is a lack of groundwater table data in most locations however, the direct transferability of the models created for Norfolk should be explored in other locations were observational data are not available. Even in Norfolk, questions still remain about how much data, both temporally and spatially, is needed to accurately forecast groundwater table levels using the methods presented in this study. In this study, at least eight years of data were available for each well, but other researchers have found acceptable results when training neural networks with more (Coulibaly et al., 2001; Daliakopoulos et al., 2005) and less (Taormina et al., 2012; Yoon et al., 2011) time series data. Based on our sensitivity analysis, rainfall is the most important input for the models. However, sea level data was from a single station; if there were more sea level gauges throughout the city it could provide a more accurate input for these models to learn from. The groundwater table monitoring network in Norfolk consists of only seven wells; while this network is a valuable source of data, it may not be dense enough to accurately represent the groundwater table across the complex urban landscape. The city is divided by many tidal rivers and stormwater conveyances and the effects these features have on the groundwater table maybe highly localized. Areas where groundwater table level is important to flooding are likely not well represented by a distant monitoring well. Research has been done with kriging to determine potential densities of groundwater monitoring (Ran et al., 2015) and rain gauge networks (Sadler et al., 2017). A similar approach may be valuable in Norfolk or comparable cities to determine the optimal density of monitoring networks when planning for and adapting to climate change and sea level rise.

2.7 Conclusions

The objective of this study was to compare two types of neural networks, RNN and LSTM, for their ability to predict groundwater table response to storm events in a coastal environment. The study area was the city of Norfolk, Virginia where time series data from 2010–2018 were collected from seven shallow groundwater table wells distributed throughout the city. Two sets of observed data, the full continuous time series D_{full} and a dataset of only time periods with storm events D_{storm}, were bootstrapped and used to train and test the models. An additional dataset D_{fcst} including forecasts of rainfall and sea level was used to evaluate model performance in a simulation of real-time model application. Statistical significance in model performance was evaluated with t-tests. Major conclusions from this study, in light of the hypotheses described in Table 4 are:

- Both model type and training data are important factors in creating skilled predictions of hourly groundwater table using observed data:
  - Using D_{full}, LSTM had a lower average RMSE than RNN (0.09 m versus 0.14 m, respectively)
  - Using D_{storm}, LSTM had a lower average RMSE than RNN (0.05 m versus 0.10 m, respectively)

- The best predictive skill was achieved using LSTM models trained with D_{storm} (average RMSE = 0.05 m) versus RNN models trained with D_{storm} (average RMSE = 0.10 m)
- LSTM has better performance than RNN but requires approximately 3 times more time to train
- In a real-time scenario using observed and forecasted input data, accurate forecasts of groundwater table were created with an 18 h horizon:
  - LSTM: Average RMSE values of 0.03, 0.05, and 0.07 m, for the t+1, t+9, and t+18h forecasts, respectively
  - RNN: Average RMSE values of 0.06, 0.10, and 0.10 m, for the t+1, t+9, and t+18h forecasts, respectively

Forecasts of groundwater table levels are not common; in many locations even direct measurements of the groundwater table are not widely available. As sea levels rise and storms become more extreme, however, forecasts of groundwater table will become an increasingly important part of flood modeling. In low-lying coastal areas, sea level rise, stormwater infiltration, and storm surge could cause groundwater inundation. Even if groundwater inundation does not occur, increased duration of high groundwater table levels could have significant impacts on infrastructure. Forecasts of groundwater table, an often overlooked part of coastal urban flooding, can provide valuable information on subsurface storage available for stormwater and help inform infrastructure management and planning.

2.8 Data, Model, and Code Availability

Supplementary Materials: Model code is available on Github at: https://github.com/UVAdMIST/Norfolk_Groundwater_Model. Data is available on Hydroshare at: http://www.hydroshare.org/resource/813dedd3568b4ef3897202988c14a522.

2.9 Acknowledgements

This work was funded as part of a National Science Foundation grant: Award 1735587 (CRISP—Critical, Resilient Interdependent Infrastructure Systems and Processes). The authors thank the Hampton Roads Sanitation District for access to their data and the Advanced Research Computing Services at the University of Virginia for HPC assistance.

Chapter 3

Flood Mitigation in Coastal Urban Catchments using Real-time Stormwater Infrastructure Control and Reinforcement Learning [1]

3.1 Introduction

As the frequency and intensity of storms increases due to changes in climate, the ability of existing stormwater infrastructure to mitigate urban flooding is being increasingly stressed (Wuebbles et al., 2017; Berggren et al., 2012; Neumann et al., 2015; Mounce et al., 2020; Mynett and Vojinovic, 2009). In coastal cities, gravity-driven stormwater systems are critical for flood management, but their functionality is also being reduced by sea level rise (Sadler et al., 2020). These stressors, combined with the flat, low elevation topography of many coastal cities, means that these communities are already experiencing increased flooding during high tide events (Sweet and Park, 2014; Moftakhari et al., 2015, 2017).

Advances in urban hydroinformatics (Mynett and Vojinovic, 2009), including smart stormwater systems (Kerkez et al., 2016), provide promising tools and approaches to improve stormwater system performance in coastal communities. In the smart stormwater system approach, existing stormwater systems are retro-fitted with real-time sensors and actuators (e.g., remotely controlled valves and pumps) to allow active monitoring and control. Active control is a cost-effective way to more efficiently use the existing capacity of stormwater infrastructure (Jose Meneses et al., 2018). In addition, active control can allow a system to function as a whole, which can be much more effective than piece-wise capital improvement of passive infrastructure systems (Wong and Kerkez, 2018).

Key to the effectiveness of active systems is the use of real-time control (RTC) (Kerkez et al., 2016; Mounce et al., 2020; Schwanenberg et al., 2015). Real-time control uses streaming sensor data (i.e., current rainfall and retention pond depths) to approximate the current system states. The system state can then be used to

[1] This chapter has been published as Bowes, B.D., Tavakoli, A., Wang, C., Heydarian, A., Behl, M., Beling, P.A., Goodall, J.L., 2020. Flood mitigation in coastal urban catchments using real-time stormwater infrastructure control and reinforcement learning. J. Hydroinformatics 23, 529–547. https://doi.org/10.2166/hydro.2020.080

inform changes to control assets (e.g., valves, pumps) that adapt the behavior of the system to current or forecast conditions. The decisions on when and what structures to control, and how to change them in order for a system to meet certain objectives (e.g., minimize flooding, maintain certain flow conditions), is based on a control policy. In a smart stormwater system, control policies map system states, such as water levels in a pipe network, to actions that need to be taken in order to meet management objectives (Sadler et al., 2019). In current practice, control policies are often predefined simple heuristics, such as opening a valve when a storage pond reaches a certain depth (level control or feedback control), and may be based on the experience of a human operator (García et al., 2015; Abou Rjeily et al., 2018). This heuristic approach may be effective in simple systems (i.e., a system with only a few controlled assets); however, it requires that control actions are predefined for all scenarios and becomes increasingly difficult as the number of assets grows and/or more external factors start influencing the system.

Heuristic control can be improved by incorporating some aspects of feedback control (i.e., system observations) and feed-forward control (i.e., forecasts and predicted system states) (Schwanenberg et al., 2015), termed rule-based control (RBC) in this research. RBC can be implemented in stormwater systems based on watershed characteristics and forecast rainfall data in order to meet flooding, water quality, and/or channel protection objectives (Marchese et al., 2018). By continuously monitoring retention pond depths and rainfall forecasts, inflow from storm events can be estimated using simple rainfall-runoff models and used to inform control decisions. For example, if a storm event has been forecast, the available volume in a pond can be then adjusted based on the estimated inflow from the storm event. This adjustment is made by actuating a valve at the pond's outlet that can be opened to drain water until the necessary storage volume in the pond is reached. RBC is intuitive to end-users and can be effective for controlling individual stormwater system components (OptiRTC and Geosyntec Consultants Inc., 2017). However, as system complexity increases (e.g., releases of water need to be coordinated if multiple ponds drain to the same downstream location in order to prevent flooding), RBC becomes more difficult to implement.

Two approaches for finding control policies for RTC beyond simple heuristics and rule-based control include model predictive control (MPC) and reinforcement learning (RL). MPC uses a process model to evaluate various control strategies for a specified control horizon (Camacho and Bordons, 2007; García et al., 2015). It has been successfully applied to large scale water systems for multi-objective optimization (Schwanenberg et al., 2015) and for stormwater system control (Sadler et al., 2019, 2020). Sadler et al. (2019) examined the effectiveness of MPC for a tidally influenced stormwater system and successfully found control policies that reduced flooding compared to the passive system. This particular formulation of MPC used a physics-based stormwater system model optimized with a genetic algorithm. Using high performance and cloud-based computing, the authors were able to speed up this computationally expensive MPC formulation. However, as a limitation of this approach, the authors indicate that for large real-world stormwater systems such MPC techniques could be prohibitively slow for RTC. When scaled to part of a real-world stormwater system, Sadler et al. (2020) were able to run MPC for three simulated actuators and determined that RTC could help reduce flooding for future sea level rise scenarios.

RL is a category of machine learning that aims to learn from trial-and-error experience by interacting with an environment (Sutton and Barto, 2018). An RL agent does not have known answers to learn from, but instead tries to maximize the amount of reward it can receive from its environment by taking certain actions. One of the differences between RL and MPC is that RL can learn control policies offline, which moves the computational burden to prior to taking any actions. The use of RL in water resources engineering has been compared with Dynamic Programming (DP) for multi-objective reservoir management (Lee and Labadie, 2007; Castelletti et al., 2014; Castelletti et al., 2013; Pianosi et al., 2013; Delipetrev et al., 2017). These studies used tabular RL methods, which can be more computationally efficient than DP but are constrained to simple systems (i.e., a small number of potential system states and actions), and demonstrated that a physics-based model could act as the environment for RL agents to train on. Recent advances in deep learning have allowed RL to use neural networks as function approximators to overcome the limitations of tabular RL. For instance, Mullapudi and Kerkez (2018) demonstrated control of a stormwater system with a Deep Q-Network (DQN) (Mnih et al., 2015). By throttling valves, they were able to make control decisions for a discrete action space (a limitation of DQN). Their work also highlighted how rewards can be shaped for real-time stormwater control through deep RL and illustrated how RTC with RL can increase a stormwater system's effective capacity. However, their DQN agent was only reactive to the current conditions of the stormwater system; effective RTC strategies should be based on both the current conditions and forecasts. Additionally, discretizing control actions may not always provide the most efficient policy. Further research is needed, therefore, to determine if RL control can be further refined using a continuous action space that allows any valve position to be used and create forecast-based predictive control policies.

In this paper, RL is used to create control policies for RTC of a tidally influenced stormwater system. In addition to presenting the first work exploring RL for RTC of tidally influenced stormwater systems, we advance on prior work in smart stormwater by exploring the Deep Deterministic Policy Gradient (DDPG) RL algorithm to control valves over a continuous action space, allowing the agent to learn more refined control policies. Additionally, forecasts of rainfall and tide are included as part of the state information received by the RL agent, allowing the agent to learn proactive control strategies. The RL agent's performance is compared to (i) a passive, gravity-driven system (ii) MPC (as implemented by Sadler et al. (2019)), and (iii) RBC. Through this comparison, we illustrate the applicability of RL for RTC of a simulated coastal urban stormwater system.

3.2 Methodology

In this section, the simulated stormwater system is introduced, a hypothetical scenario for categorizing the impact of flood events is detailed, and each RTC method is described. The stormwater system is similar to that used by Sadler et al. (2019) and is the same for all scenarios (RL, Passive, MPC, RBC), except that retention ponds in the passive system have weirs at a fixed elevation to maintain a depth of water in the ponds while the ponds for the RTC scenarios have a controllable valve at the bottom of the pond side. An overview of the methodology used to compare the performance of an RL controlled stormwater system with the other scenarios is provided in Table 3.1. Open-source code for these scenarios is available on github (Bowes, 2020b). SWMM simulation files and data are available as a resource on HydroShare (Bowes, 2020a).

Table 3.1: Overview of methodology for stormwater system control scenarios

Method	Valve	Training/Optimization	Control Policy	Testing
RBC system	Controllable valve at 0m	N/A	Fixed for simulation duration	Test on 2010-2019 data
MPC system	Controllable valve at 0m.	Online optimization with genetic algorithm	Adjusted valve positions for specified time horizon based on modeled scenarios and objective functions	Test on first week of August, 2019 data (due to computational cost)
RL system	Controllable valve at 0m.	Offline training on August, 2019 data for 197,000 control steps	Learned policy relating states to actions	Test on 2010-2019 data
Passive system	Fixed weir at 0.61m.	N/A	N/A	Test on 2010-2019 data

3.2.1 Stormwater Simulation

Stormwater system simulations are carried out using the U.S. Environmental Protection Agency's Stormwater Management Model (SWMM), version 5. The hypothetical stormwater system used in this study is a simple model inspired by conditions within an urban catchment located in Norfolk, Virginia, USA. It consists of two subcatchments, two storage units (retention ponds), and pipes going to the system outfall that discharges to a tidally influenced waterbody (Fig. 3.1). In the passive scenario, each pond has a weir at a fixed elevation and cannot be completely emptied. Infiltration and evaporation are excluded in this simple system as the focus is on flood control. Ponds in the RTC scenarios have been retro-fitted by removing the weirs and adding controllable valves (orifices) at the bottom of the pond side. This allows the full pond volume to act as active storage that can be adapted for different storm events. Input to this system is rainfall, which falls onto a subcatchment and is transformed into runoff that flows into a storage unit. Flow out of the storage units can be regulated by the valves; both ponds drain directly into a single node before flowing through two pipe segments to the outfall. The tail water condition at the outfall is influenced by the tide level. At high tide, the tail water condition can cause sea level to partially block the outfall, backing up stormwater drainage. The physical parameters of the SWMM model can be found in Table 3.2.

In this SWMM model, flooding can be caused by (i) rainfall, (ii) high tide, or (iii) a combination of rainfall and tide. Flooding caused by these factors can be in the form of the ponds over-topping or the downstream node J1, which can be thought of as a roadway storm drain, surcharging. Ponds can over-top if the subcatchment runoff volume for a storm event exceeds current pond capacity and inflow is greater than outflow. Flooding at the downstream node can occur if flow from the ponds is not regulated and coordinated by adjusting the two valves. Node J1 can also flood if tidal conditions at the outfall are preventing the normal flow of water from the system or causing backflow if the tide is especially high.

The pyswmm (McDonnell et al., 2020) Python wrapper for SWMM is used for step-by-step running of simulations as needed for the RTC methods. Each control scenario can be updated once every 15 minutes (an adjustable control time step).

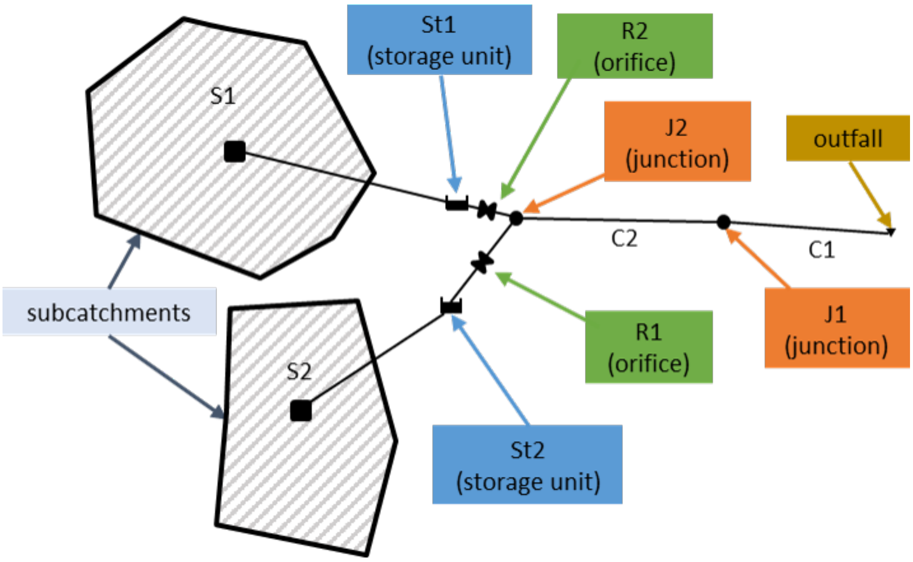

Figure 3.1: SWMM simulation schema

Table 3.2: Properties of SWMM model

Subcatchment Properties				
Name	Area (hectares)	Width (km)	Slope (%)	Impervious (%)
S1	32	0.46	0.5	30
S2	20	0.61	0.5	45

Storage Unit Properties				
Name	Surface Area (m^2) (constant)	Initial Depth (m)	Max Depth (m)	Bottom Elev (m)
St1	6039	0.61	1.41	0.91
St2	4645	0.61	1.41	0.91

Pipe Properties			
Name	Length (m)	Diameter (m)	Roughness
C1, C2	122	0.3	0.01

Node Properties		
Name	Max Depth (m)	Bottom Elev (m)
J2	1.5	0.91
J1	0.6	0.30
Outfall	NA	0

Orifice Properties		
Name	Height (m)	Discharge Coefficient
R1, R2	0.61	0.65

3.2.2 SWMM Input Data

Input data for the SWMM simulation comes from observed data for stations in Norfolk, Virginia (Figure 3.2). Rainfall data is collected at a fifteen minute timestep by the Hampton Roads Sanitation District (HRSD); rainfall data for subcatchment S1 is from gauge Rain1 and data for subcatchment S2 is from gauge Rain2. Rainfall data is cleaned through a number of checks. First, any values over the 1000-year, 15-minute rainfall for Norfolk (59.18mm) are assumed to be erroneous and removed. Next, missing data in each rainfall time series are replaced; if both rain gauges are missing values at the same time stamp, both get zero. Otherwise, if one station is missing a value but the other is not, the missing value is replaced with the known value from the other station. Observed tide level data comes from the National Oceanic and Atmospheric Administration (NOAA) Sewells Point gauge and is measured at six minute intervals. For use as a SWMM boundary condition, tidal data are resampled to an hourly interval and referenced to the North American Vertical Datum of 1988 (NAVD88). All of the observed data are for the period between 1 January, 2010 and 6 November, 2019 and are divided into individual months to make simulation run times tractable.

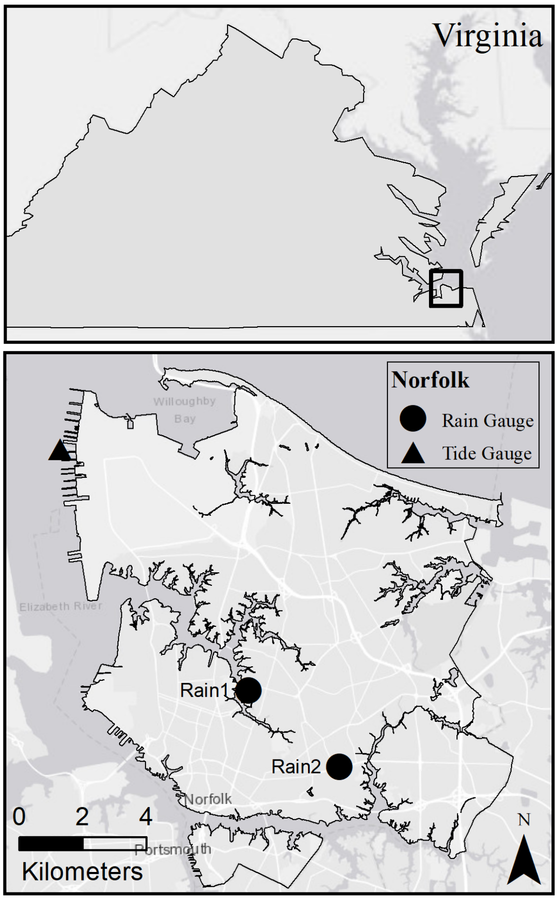

Figure 3.2: Gauge locations in Norfolk, VA USA.

Forecasts were created from the observed data for use in the various control methods, therefore assuming perfect knowledge of future events. A single forecast in this case is an array of values representing the rainfall or tide measurement over the next n time steps. For example, a 24 hour forecast of 15 minute rainfall would contain 96 values. In future work, noise could be added to these forecasts to explore how RL (or any other RTC method) handles uncertainty, but this is beyond the scope of this research (for future directions, see van Andel, et al., (2008; 2014), Hartono and Hashimoto, (2007)).

3.2.3 Flood Event Classification

In order to quantify flooding impact, a hypothetical scenario is developed from physical data for Norfolk. In this scenario, the subcatchments considered are residential neighborhoods where any flooding of the ponds will impact roadway intersections. The downstream node J1 is considered a storm drain at a roadway intersection; flooding at this node will make the intersection impassible if the depth of flood water is above a certain threshold. For this hypothetical scenario, 0.2m of roadway flooding slows traffic considerably, the threshold for safe passenger vehicle passage is 0.3m (Pregnolato et al., 2017), and 0.4m is the limit for safe emergency vehicle passage. The relationship between flood volume and depth was developed from digital elevation data from Norfolk as described in Appendix E. The number, volume, and duration of these flood events is used as an additional metric for quantifying flooding along with the total volume.

3.2.4 Implementing RL in Stormwater Systems

In reinforcement learning, an agent learns to optimize its behavior by interacting with its environment (Sutton and Barto, 2018). The environment is usually modeled as a Markov Decision Process (MDP): $< \mathcal{S}, \mathcal{A}, P, r, \gamma >$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s'|s,a)$ is the stochastic probability of transitioning to a new state s' after taking action a at the current state s, $r(s,a,s')$ is the reward function, and $\gamma \in [0,1]$ is the discount factor that weighs the importance of short term and long term reward. The RL agent's goal is to find an optimal policy that maximizes the expected discounted return

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{3.1}$$

where $r_t = r(s_t, a_t, s_{t+1})$.

In this paper, the states $\mathcal{S}$ are defined as the current depths and rate of flooding (if any) at the ponds and downstream nodes (*St1, St2, J1*), the current valve positions (*R1, R2*), the sum of the 24hr rainfall forecast (F) for each subcatchment, and the mean value of the 24hr tide forecast. These values are gathered from the SWMM simulation at each control time step. The actions $\mathcal{A}$ that the RL agent can take at any step is to close or open any valve to any degree. Finally, the reward r the RL agent receives in this system is based on how well the agent prevents flooding and maintains certain target pond water depths. It is defined as

$$r = \begin{cases} -\Sigma(flooding) & F > \delta \\ -J1_{flooding} - (|St1_{depth} - target| + |St2_{depth} - target|) & F = 0 \end{cases} \tag{3.2}$$

where $flooding$ is the flooding rate at each particular node ($St1, St2, J1$) and δ represents a forecast rainfall threshold (>0 in this case); $target$ is the target water depth for the storage ponds ($St1$ and $St2$). In this relatively simple stormwater system, the target depth is 0.61m for both ponds. In a real system, different ponds would most likely have different target depths; this can be taken into account in the RL implementation by having different target depths for each pond in the reward function.

As an example, consider a case where the agent is in a specific state s in $\mathcal{S}$ (e.g., the water depth in a specific pond is 1.0m), and takes an action a in $\mathcal{A}$ (e.g., completely opens the valve) with a probability given by the policy $\pi(a|s)$. The agent will then transition to a new state s' with a probability of $P^a_{s,s'} = P(s'|s,a)$ (e.g., the water depth in a specific pond is 0.75m) and receives a reward $r(s,a,s')$. The value of this action depends on the reward that the agent receives and the discounted value of all the future rewards if the agent follows the policy afterwards. Using a discount factor γ, the value of a future reward of x after n steps is $x\gamma^{n-1}$. The expected discounted return when starting in state s, then taking action a, and following π is called the Q-value function:

$$\begin{aligned} Q^{\pi}(s,a) &= E[G_t|s,a] \\ &= r(s,a,s') + \gamma \sum_{s' \in \mathcal{S}} P^a_{s,s'} \sum_{a' \in \mathcal{A}} \pi(a'|s') Q^{\pi}(s',a'), \end{aligned} \tag{3.3}$$

where the second equation is known as the Bellman equation (Sutton and Barto, 2018).

By having the optimal Q-values, one can find the optimal policy by finding the specific actions in each state that give the maximum Q-value. However, due to the curse of dimensionality this tabular type of Q-learning is limited to problems with relatively small state and action spaces. Recent advances in deep learning have been applied to RL to overcome this problem by using deep neural networks to approximate value functions instead of storing them in tables (Mnih et al., 2015).

In order to have an RL agent that can set the valves to any position over a continuous action space , the Deep Deterministic Policy Gradients (DDPG) (Lillicrap et al., 2015) actor-critic algorithm is used. DDPG uses deep neural networks to approximate a policy and the difference between policies, the gradient, is used to update the agent. In this case the agent consists of two parts: an actor which represents the policy, and a critic which estimates the q-value of actions taken by the actor. During the training process, the actor is fed information on the state of the stormwater system and outputs the actions to be taken. These actions, along with the state information, are used as input to the critic. The actions and q-value estimates output from the critic are used to update the agent.

The keras-rl (Plappert, 2016), openai gym (Brockman et al., 2016), and Tensorflow (Abadi et al., 2016b) python packages are used to implement the DDPG algorithm for this research. Each part of the DDPG agent, the actor and the critic, is composed of a deep feed-forward neural network (Table 3.3). The hyperparameters of each neural network are determined by trial and error (Maier et al., 2010). Through experimentation, it can be found that training the RL agent on the August, 2019 dataset and looping through the SWMM simulation approximately 100 times, provided enough experience of a wide range of rainfall and tidal events for the agent to learn from. This month has a total of 256.54mm of rainfall over 7 events. The average tide level is 0.16m with a maximum value of 1.01m from Tropical Storm Erin late in the month. A visualization of this data is given in Figure 3.4. RL training and testing are carried out on a standard PC with 8 cores, 16GB RAM, and an NVIDIA Quadro P2000 Graphical Processing Unit (GPU).

Table 3.3: DDPG RL agent architecture and hyperparameter settings

NN Layer	Actor		Critic	
	Neurons	Activation	Neurons	Activation
Input	Current state s	N/A	Current state s and action a	N/A
Hidden 1	16	RELU	32	RELU
Hidden 2	16	RELU	32	RELU
Hidden 3	8	RELU	32	RELU
Output	2 [R1, R2]	Sigmoid	1 [q-value]	Linear

3.2.5 MPC Settings

The swmm_mpc software developed by Sadler et al. (2019) is used to implement MPC for comparison with RL; readers are referred to this paper for full details on the MPC implementation. Briefly, swmm_mpc uses SWMM as a process model and an evolutionary algorithm to search for a control policy. At each time step in a SWMM simulation, swmm_mpc runs many variations of the SWMM simulation in order to determine which control actions minimize an objective function for a specified time horizon. In this case, the objective function is based on the amount of flooding and deviations from target water level depths as

$$\text{MPC objective function} = \alpha(\boldsymbol{a} \cdot \boldsymbol{v}(\boldsymbol{u}, \boldsymbol{x})) + \beta(\boldsymbol{b} \cdot \boldsymbol{d}(\boldsymbol{u}, \boldsymbol{x})) \tag{3.4}$$

$\boldsymbol{v}$ and $\boldsymbol{d}$ are 1-dimensional vectors of flood volumes at each node and deviations from target depths, respectively. The 2-dimensional vectors $\boldsymbol{u}$ and $\boldsymbol{x}$ represent the control policies for all controls and the system states, respectively. The user defined parameters and their definitions are given in Table 3.4. The scalar multipliers α and β are overall weights for the cost of flooding and the cost of water level deviations. These will be adjusted in order to optimize the MPC control.

Table 3.4: MPC cost function parameters

Parameter (description)	Value
α (overall flood weight)	Scalar
a (individual node flood weight [St1, St2, J1, J2])	[1, 1, 1, 1]
β (overall deviation weight)	Scalar
b (individual deviation weight [St1, St2, J1, J2])	[1, 1, 0, 0]
Target depths (m)	[0.61, 0.61, NA, NA]

Because of the computational expense of running swmm_mpc, where many variants of the SWMM model have to be executed at each simulated time step to find the best control actions, a high performance computer (HPC) was used to run the software. The HPC computational resources consisted of 28 cores with a CPU speed of 2.4 GHz, an Intel Xeon processor, and 128 GB RAM.

3.2.6 Rule-based Control

Rule-based control (RBC) was implemented based on documented industry standard methods (OptiRTC and Geosyntec Consultants Inc., 2017; Marchese et al., 2018;

Wright and Marchese, 2017). In practice, this type of control uses forecasts of rainfall and watershed characteristics to inform the control of valves on stormwater assets (wet/dry ponds, bioswales) in order to meet flood control, water quantity, and/or quality objectives.

Because the current research is done on a simulated system, the expected flood volume from a forecast of rainfall, if any, is used to control the level of water in an individual pond. For example, if a forecast storm event is expected to cause 1000m^3 of flooding, the pond's outlet valve would open before the storm in order to drain out a corresponding volume of water plus a 20% safety factor. After the pond's depth is drawn down by the appropriate level, the valve can be closed to retain the incoming stormwater, which helps improve water quality. In this way, storm runoff should not flood the pond and will be retained to prevent flooding downstream. After a storm event, water can be held in the pond for a specified settling period (24hrs in this case) and then slowly released (over 24hrs) to bring the pond back to its standard operating range. Outside of storm events, rules can also be in-place to maintain the pond level within the operating range or maintain certain flow conditions. The exact control rules and their hierarchy, as implemented in this research, are detailed in Figure 3.3.

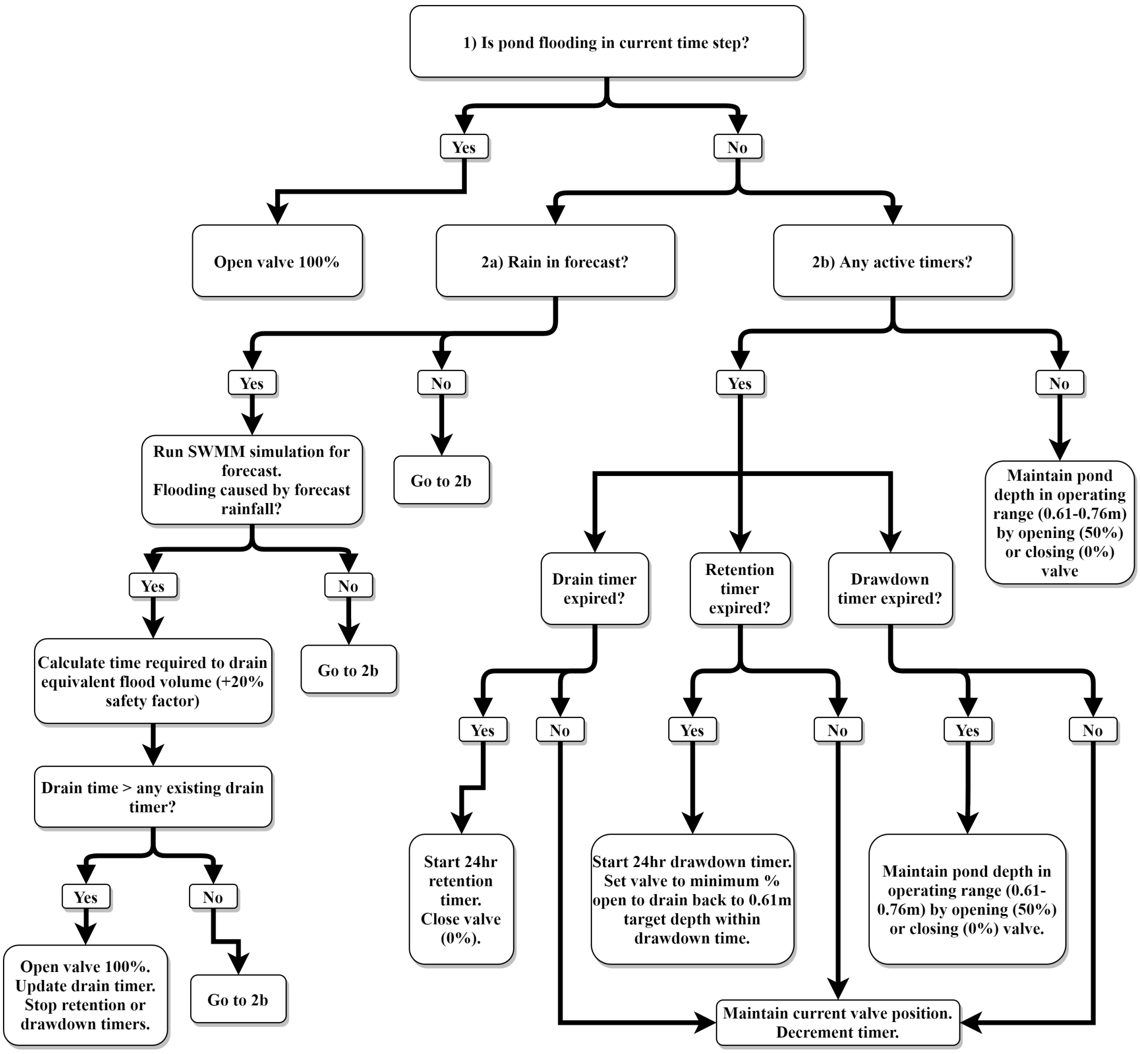

Figure 3.3: Rule-based control hierarchy and settings.

3.3 Results

3.3.1 Comparison of RL and Passive System

A comparison of the RL agent's policy against the passive system shows that the agent can learn to effectively control valves to maximize its reward. As indicated in Figure 3.4, the training data shows valves are opened when rainfall is in the forecast, allowing additional storage space in the retention ponds. After a storm is over, valve positions are adjusted again in order to maintain a pond depth close to the target of 0.61m. Following this policy allows the RL agent to reduce total flood volume for this month by approximately 70% (5936 vs. 19957m^3) compared to the passive system. For example, in the first storm of August, 2019, both valves (R1 and R2) are opened to drain water in response to the rainfall forecast (a detailed figure with this comparison is available in Appendix D, Fig. 1). However, due to the difference in rainfall on the subcatchments, R2 closes earlier than R1 in order to maintain the target depth. Directly after storm events, the RL agent tries to balance returning the ponds to the target depth and preventing flooding downstream at J1. Because the RL agent also needs to maintain the target pond depths compared to the passive system, there is an increase in the number of minor events at the downstream node, despite these being minor events.

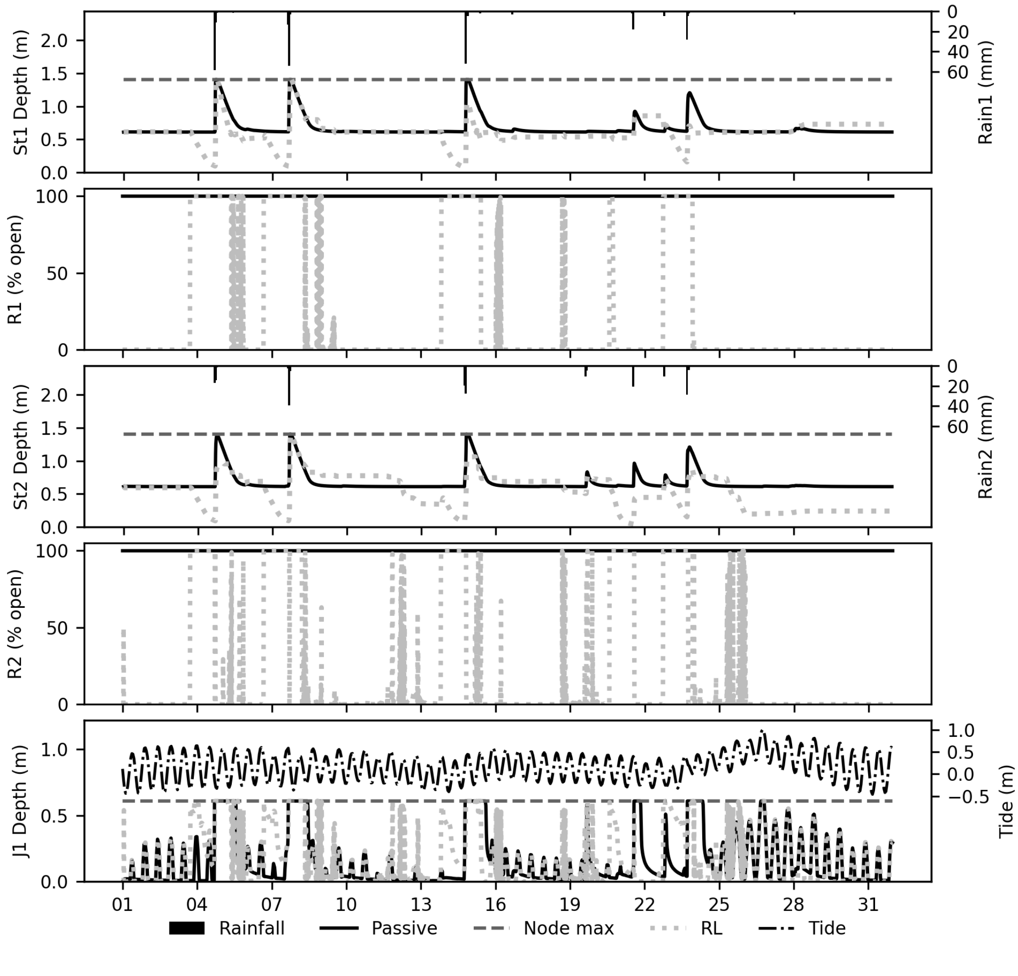

Figure 3.4: Comparison of RL controlled and passive system performance on August, 2019 training data.

Applying the policy learned on the August, 2019 training data to the test sets shows that this RL agent has learned a policy that works well in many other conditions (Appendix D, Fig. 2). Compared to the passive system, total flooding was reduced by RL in 85 of the 120 months of data (71%). In particular, this policy works well on test sets with similar (e.g., 08-2018) or larger (e.g., 09-2016) amounts of flooding than the August, 2019 training set (the mean total flood volume for these months was 4278m^3). In a few cases (such as 04-2019 or 09-2017), the RL policy increases the amount of flooding. These are months with little or no flooding (mean of 606m^3) and the agent has learned to respond to rainfall events in a manner that is not ideal for these months with less flood risk. The agent's performance on these months can be improved by increasing the threshold value for rainfall forecasts used in the conditional reward. For example, if the conditional reward threshold is increased from 0 to 1.3mm of rain, the agent's performance on months with low flooding is improved (Appendix D, Fig. 2, "RL: 1.3"). However, this is at the expense of performance on the larger storms. Overall, the agent trained with the 1.3mm threshold had lower flooding than the agent trained with the 0.0mm threshold in 28 months (23% of the data), but increased flooding in the remaining 92 months (77%).

3.3.2 Comparison of RL and MPC

In order to make computational expense tractable, the MPC setup from Sadler et al. (2019) was only run using data from the first week in August, 2018. However, due to the nature of this MPC formulation (online optimization using a genetic algorithm), computational times are still high. Finding an MPC policy for this week of data took almost 50 hours (2 days, 1 hour, and 48 minutes); computational time for each 15 minute simulated control step is 3.9 minutes. This is tractable for a simple system, but is partly a function of the simulation length and would increase with system complexity (Sadler et al., (2019)).

To investigate MPC's performance for this specific dataset, several combinations of objective function weights were tried in order to prevent flooding and maintain target storage pond depths (Table 3.5). The best performing of these combinations is having the flood weight set to 1 and the deviation weight set to 10 (MPC4). This result was unexpected given that an even weighting seems like it would provide the best balance of flood mitigation and pond depth maintenance. Further, this MPC formulation was the only one in which both ponds were not kept empty for the dry periods in the simulation. A visualization of the policies carried out by MPC and RL shows that, while MPC modulated orifice R2 and kept pond St2 close to the target depth, orifice R1 was slightly open and static for much of the time (Fig. 3.5). This allowed pond St1 to essentially empty during dry periods, which is an undesirable behavior. In contrast, RL was better at maintaining the target depth before the first storm and between the storm events.

3.3.3 Comparison of RL and RBC

Rule-based control results were generated for the same monthly datasets used in the RL training and testing. RBC was able to reduce flooding by 57% for the month of August, 2019 compared to the passive system (8540 vs. 19957m^3). This method

Table 3.5: MPC trials and performance comparison with the passive and RL systems for the first week in August, 2019.

Model	Alpha (overall flood weight)	Beta (overall deviation weight)	Control Horizon (hrs)	Total Flood Volume (m^3)	Accumulated Deviation (m)
Passive	N/A	N/A	N/A	13586	126.3
MPC1	1000.0	0.5	1.0	2767	620.2
MPC2	0.75	0.25	1.0	2582	614.5
MPC3	1.0	1.0	1.0	2714	591.5
MPC4	1.0	10.0	0.5	3929	439.2
RL	N/A	N/A	N/A	4058	234.5

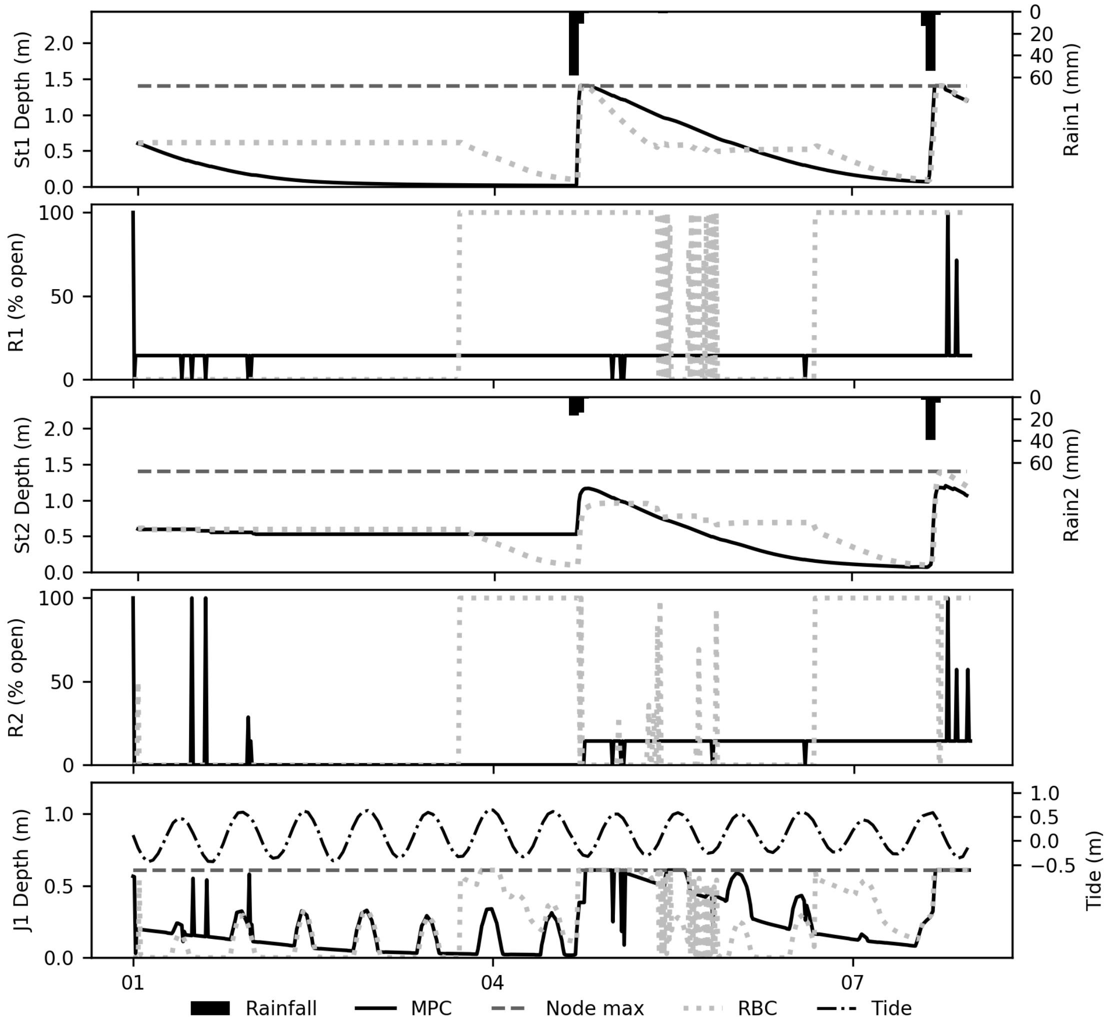

Figure 3.5: RL and MPC control policies and states for the first week of August, 2019.

of control is also able to extend the retention time of stormwater in the ponds and maintain target pond depths during dry periods or small rainfall events. Because this RBC is based on rainfall forecasts (with perfect knowledge of future events), it is able to drawdown ponds prior to a storm event based on the expected flood volume (Fig. 3.6).

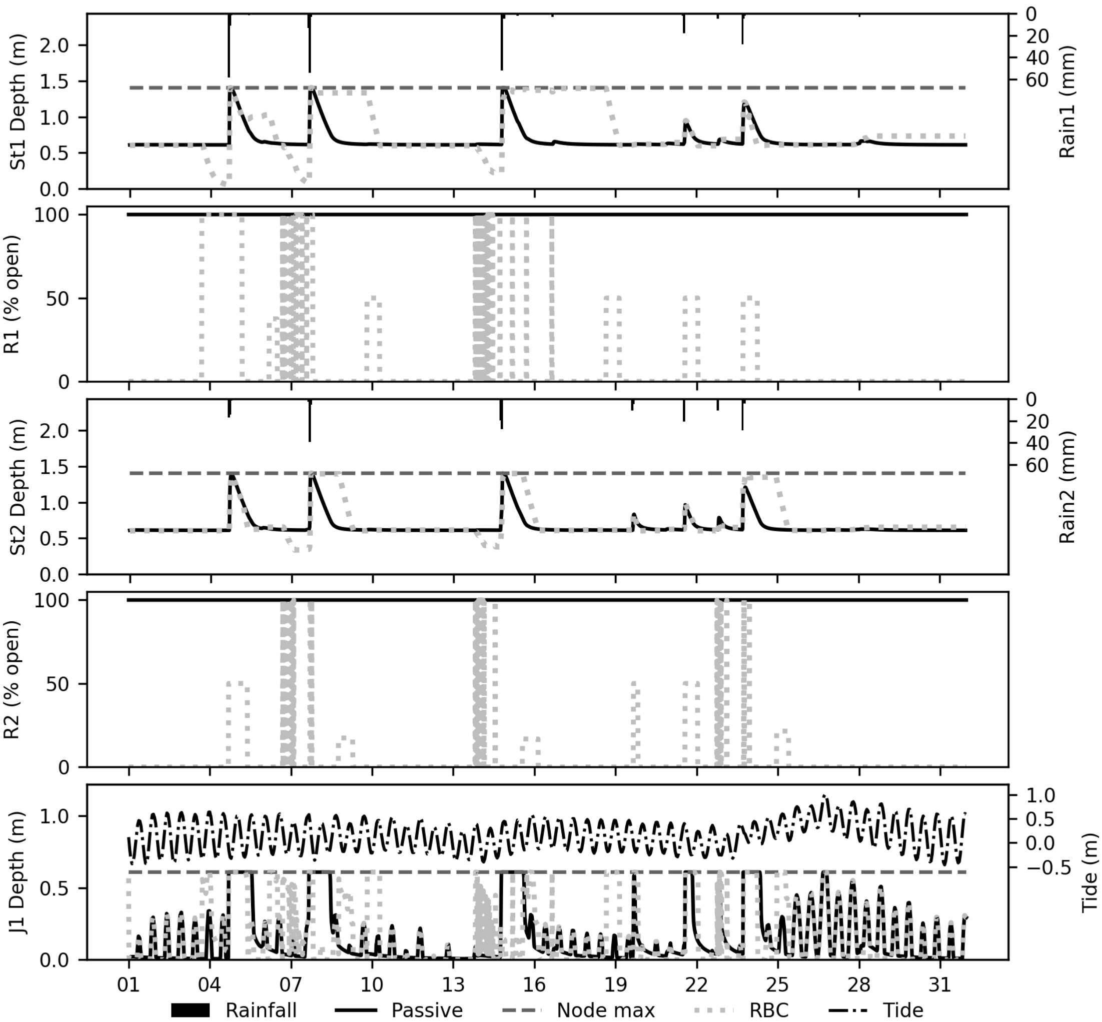

Figure 3.6: Comparison of RBC and passive system performance on August, 2019.

The RL agent's performance on the August, 2019 training data is similar to RBC but has an advantage in that the entire system state is used to inform control decisions, as opposed to using only the depth in the individual ponds (Fig. 3.7). Because of this increased system knowledge and flexibility in its valve control settings, the RL agent was able to reduce flooding by 30% over RBC (5936 vs. 8540m^3). The RBC logic is based on individual pond depths; conditions at other parts of the system (e.g., flooding downstream or tidal influence on the outfall) are only considered indirectly if they impact pond depth.

Over all the months of data, RBC reduced flooding over the passive system in 45 months (38%) (Fig. 3.8). However, these were months with large total flood volumes (mean total flood volume of 8559m^3). Similar to RL, RBC performs less well on the months with little or no flooding of the passive system (e.g., 02-2019, 03-2014, 03-2019) and a few months with more flooding (e.g., 09-2018, 10-2012). In comparison with RL, RBC had more or equal flooding in 101 months (84% of data, with a mean increase in flooding of 60%) and reduced flooding in the remaining 19 months (16% of data, with a mean decrease in flooding of 25%).

Examining the month of 07-2010 shows a situation where RBC outperforms RL

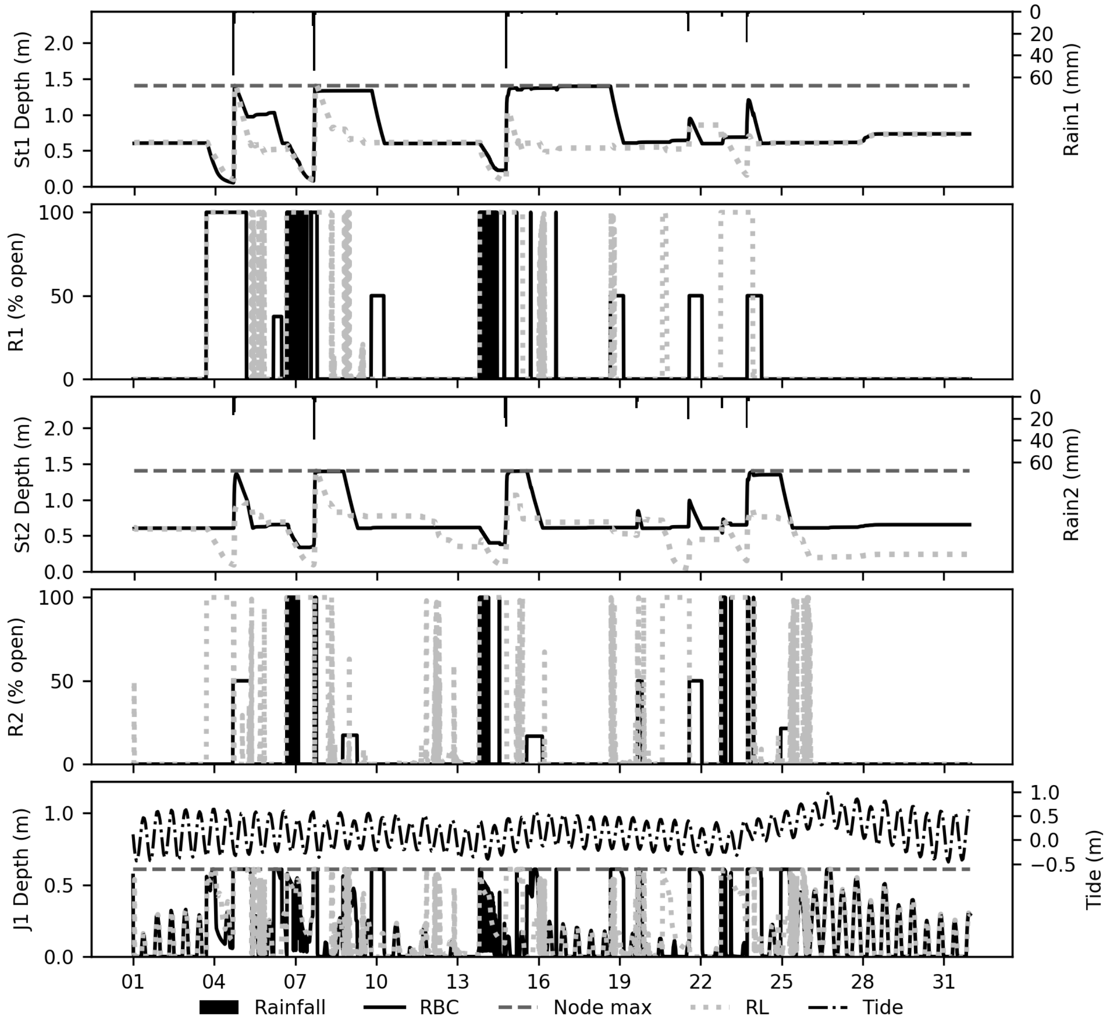

Figure 3.7: Comparison of RL and RBC system performance on August, 2019.

(see appendix D, Fig. 3). For this month, the difference in flooding between RL and RBC is relatively small (17844 vs. 16311m^3, respectively) and both reduced flooding compared to the passive system (21997m^3), but RBC better maintained the target pond depths. This example illustrates the difficulty in shaping rewards for RL; because the conditional reward function is based on the rainfall forecast, very small amounts of rainfall will cause the agent to only be rewarded for preventing flooding. As mentioned in the RL/Passive system results, the rainfall threshold used in the reward function can influence this behavior. However, using a threshold value of >0mm let the agent keep pond St1 higher than the 0.61m target depth. Because RBC maintained the target pond depths better than RL in this case, RBC did not have as much water to drain out of the ponds before the large storm event at the end of the month.

Examining the months where the passive system had lower total flooding than RBC helps illustrate its limitations. For these 75 months, the mean total flooding was 80m^3. For example, in March, 2014 RBC increased total flooding of the system by nearly 5.5 times over the passive system (2396 vs. 439m^{3}MG) (see Appendix D, Fig. 4).

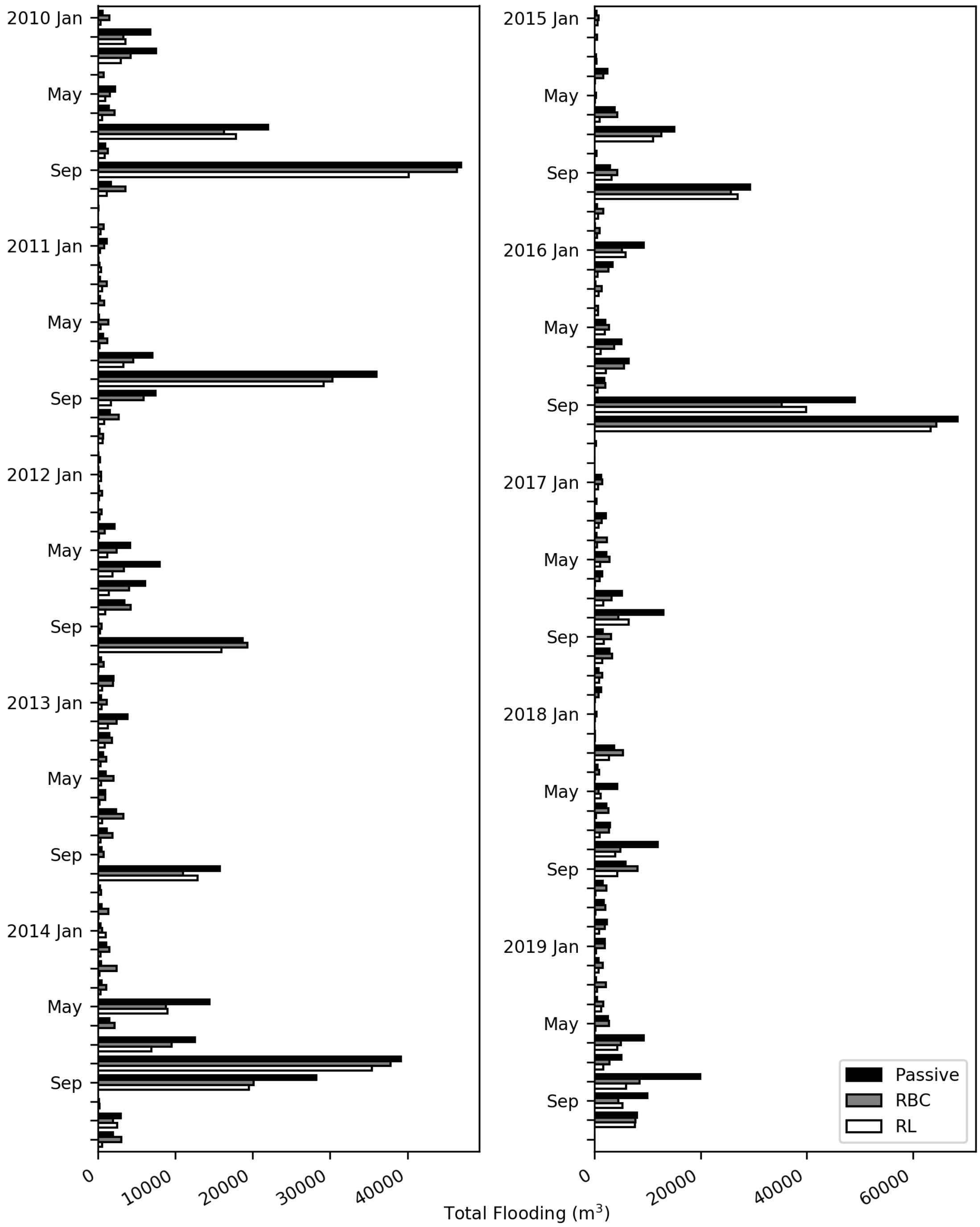

Figure 3.8: Comparison of RL, RBC, and passive system performance on all months of data.

3.3.4 Flood Event Classification Results

Based on the flood event analysis at the two storage ponds, RL had the fewest flood events (St1: 22, St2: 26), followed by the passive system (St1: 41, St2: 42). RBC had the greatest number of pond flooding events (St1: 56, St2: 59). In terms of maximum flood volume for a single event, RL had the lowest, followed by RBC, and the passive system at pond St1 (27558, 28883, 30094m^3, respectively) and pond St2 (26119, 27331, 27596m^3). The mean single event flood volume showed a similar pattern at pond St1 (719, 719, 1060m^3) and pond St2 (871, 871, 1136m^3) for the RL, RBC, and passive systems, respectively. Flood event duration at the two ponds was similar for the three scenarios, with RL having the lowest mean duration (St1: 0.45hr, St2: 0.47hr) followed by RBC (St1: 0.51hr, St2: 0.53hr) and the passive system (St1: 0.73hr, St2: 0.66hr).

Results of flood event classification for downstream node J1 (a hypothetical roadway storm drain inlet) are shown in Figure 3.9. RL had the lowest number of flood events classified at the 0.2m and 0.3m thresholds but the highest for the 0.4m threshold (Figure 3.9, A). RBC had more flood events for the 0.2m and 0.3m thresholds than RL or the passive system, but the lowest for the 0.4m threshold. This was expected, as these control rules manage the two ponds individually and not as a unified system considering downstream conditions. Flood volume for the 0.2m and 0.3m thresholds was similar across the 3 systems (Figure 3.9, C). At the 0.4m threshold however, the two RTC methods (RL and RBC) had lower minimum, maximum, and mean flood volumes than the passive system.

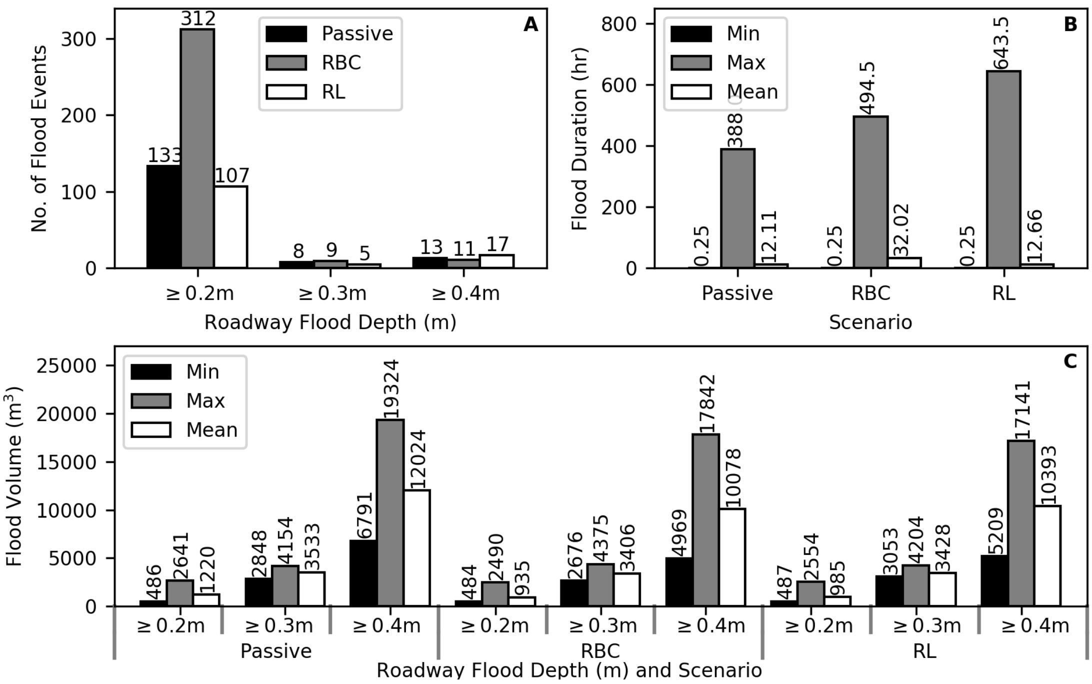

Figure 3.9: Number (A), duration (B), and volume (C) of flood events at downstream node J1. Flood volumes at node J1 were categorized as causing $\geq$ 0.2, 0.3, or 0.4m of water depth on the roadway.

Flood event duration for node J1 was lowest for the passive system, followed by RBC, and RL (Figure 3.9, B). This result makes sense when viewed in context with the flood volumes coming from the upstream ponds. RL prevented flooding at the ponds by routing more water downstream to node J1. RL had a similar mean flood duration to the passive systems, but a 60% higher maximum, indicating that RL allowed more low volume, but long duration flood events at J1 in order to reduce flooding at the two storage ponds. This behavior is influenced by the reward function; if the reward function was based on whether or not a node was flooding instead of the rate of flooding, the agent may have learned a different trade-off for managing flooding between the three nodes. Due to the lack of system coordination, RBC had longer duration flood events at J1 than the passive system, but at lower volumes. In the passive system, the short duration, but high volume of flood events at node J1 shows that, without control, this system is flashy, which is a challenge in many urban systems.

3.4 Discussion

One important aspect of any RTC application is the computational cost, both in terms of when the computation needs to happen and the time needed to compute a policy. With the RL algorithm used here, the agent learns off-line on a training datasets. Once learned, the RL policy can be quickly applied. This formulation of MPC, in contrast, uses a genetic algorithm to perform on-line optimization (i.e., the best control actions are not known until the time that they need to be implemented). In this research, RL was able to learn a policy in approximately 34 minutes using one month of data on a standard desktop computer. Once developed, testing the RL policy simply requires running a SWMM simulation, passing the system state at each control time step through the agent, and implementing the resulting control actions (this takes 9 seconds for the 08-2019 training data). Running MPC with the genetic algorithm and physics-based model as implemented in Sadler et al. (2019) required access to a high performance computer and took almost 50 hours for one week of simulation. Because optimization is on-line, additional testing of MPC on other datasets would take a similar amount of time and would increase as the complexity of the system increases (Sadler et al., 2019). In practice, MPC may only need to run using the available forecast data (e.g., 18, 24, 36 hours), not an entire week, reducing the computational burden (Sadler et al., 2020). Additionally, other formulations of MPC could use a different process model than SWMM (for instance using a state-space model learned from observed data) which could dramatically reduce MPC's computational cost (Li et al., 2013; Balchen et al., 1992; Corbin et al., 2013; Cigler et al., 2013; Behl et al., 2014). The training time for RL is also dependent on system complexity, but needs further research to determine the feasibility and limitations for larger, more complex stormwater systems.

The RBC logic used in this research, like RL, can be considered an off-line policy. Instead of an RL agent learning the control policy by interacting with the system, a human operator must understand the system well enough to formulate the rules. The growing adoption of Internet of Things (IoT) sensors for monitoring water levels provides the data needed to create control rules. In practice, the amount of time required to create these rules, and their quality, is dependent on factors like the availability of data on the physical watershed characteristics and the complexity of the system to be controlled. For the relatively simple system used in this research, and real single ponds (Marchese et al., 2018; OptiRTC and Geosyntec Consultants Inc., 2017), developing control rules is feasible. Ensuring coordinated and effective system-level control, however, will become increasingly difficult as complexity increases. As an example, the rule for valve position when trying to maintain the target depth was originally to completely open the valve. Through simulation it was found that this often caused increased flooding at the downstream node. Adjusting that rule to only open the valve 50% when maintaining the target depth helped eliminate downstream flooding but is system specific and most likely not optimal. Adding a depth sensor in the downstream pipe as an additional factor in the control rules would be possible in this case; with enough time and IoT sensors, it may be possible to create control rules considering system wide performance. However, there could be many such factors in a real urban stormwater system and accounting for each one and their interactions under different flow conditions will quickly become unmanageable. RL has an advantage here because the relationships between components of the system do not have to be known or stated explicitly, but can be learned. The disadvantage of RL, however, is that it is much less transparent than RCB in terms of how and why certain control decisions are made.

While this paper has explored RTC with RL, MPC, and RBC there are other methods from the field of control theory that could be applied to stormwater systems. Wong and Kerkez (2018) provide an elegant example by using a linear quadratic regulator to manage storage pond depths in urban headwater catchments. This uses a state-space model as a linear representation of a watershed and performs control with a feedback controller. Another key contribution of this work is the ability to optimize the location of control structures and show that the entire system does not have to be controlled to achieve system-wide benefits. The state-space representation used by Wong and Kerkez (2018) or the discrete time dynamic system shown in Schwanenberg et al. (2015) could be used in RL or MPC as a replacement to the more computationally expensive SWMM model to speed up control of larger systems, but the full dynamics of the system represented in physics-based models may be lost.

Groundwater could contribute a significant amount to retention ponds that are being actively controlled, especially in coastal cities with high groundwater tables like Norfolk, Virginia, that respond quickly to storm events (Bowes et al., 2019). For a retention pond in Norfolk, we've estimated that groundwater would contribute approximately 0.16m or 11% of the pond's volume per hour if the pond is completely emptied (See Appendix E for details on these approximations). Considering the storm event of August 4-5$^{\text{th}}$, 2019, the RL agent lowers the depth of water in the simulated ponds by almost 0.61m over a 24 hour period (Figure 1). Over that time, groundwater would have contributed an additional 0.71m or 50% of the pond's total volume. This is not currently reflected in the SWMM simulations, but has important implications in practice. While the additional inflow would most likely not change the general policy learned by the RL agent (i.e., lowering depths before a storm and maintaining depths otherwise), a larger valve may be needed to drain the ponds more quickly or the agent may need a longer forecast in order to drain the ponds prior to a storm event. Additionally, evaporation should be included in these simulations before being applied to real world systems.

When implementing any of the RTC methods presented in this research, the method's interpretability will influence its adoption and use by decision makers. While RBC is easy to understand and highly transparent, MPC is less so, and RL is the least transparent. The control policies created by RL, while effective, can cause the system to make decisions that are non-intuitive to a human operator. Therefore, fully automating smart stormwater systems with RL may not be advisable at this time until more testing and safety controls can be put in place. However, RL could assist human operators in determining control policies and support decision making, for example as part of a recommendation system (Solomatine and Ostfeld, 2008). RL-based policies should continue to be trained with new data as it becomes available to increase confidence that the RL policies will produce desirable outcomes.

This study shows, however, that even with a single month's worth of training data, RL shows great potential for determining effective control policies.

3.5 Conclusions and Future Work

This research has explored the application of an RL agent for real-time stormwater system control where both rainfall and tidal level can impact flooding and retention pond depths in the system. In contrast to previous work, this paper used a continuous action space to create more refined control policies, by implementing the DDPG RL algorithm. A conditional reward structure based on the rainfall forecast and inclusion of forecasts in the system state allowed the RL agent to learn proactive control strategies. The performance of RL was compared to a passive system as well as two other RTC methods: MPC and RBC.

Results of this research show that both RL and RBC can improve stormwater system performance compared to the passive system. Using a control policy developed from a single month of rainfall and tide data, RL reduced total flood volume by 32% over the passive system for the 2010-2019 data. RBC, while only controlling ponds individually, still reduced total flood volume by 13% compared to the passive system. Additionally, this research showed that RL was able to learn to balance flooding throughout the system to maximize the conditional reward and meet the control objectives of mitigating flooding and maintaining target pond water levels. When implemented using the SWMM physics-based model, as described in Sadler et al. (2019; 2020), MPC was too computationally expensive to run for more than a small portion of the datasets. In this research, RL provided an 88x speedup in the creation of control policies compared to MPC.

Although the simple stormwater system, which is inspired by conditions in the coastal city of Norfolk, Virginia, demonstrates that RL can outperform other methods, more complex systems will face different computational burdens that could be a barrier to using such methods in real-time. This needs to be explored through future research testing RL on real-world systems. In addition, an alternative implementation of MPC using a state-space model, instead of the SWMM model used here, could dramatically reduce computational cost for this control method. Lastly, the feasibility of using, and potentially combining, any of the real-time control methods for decision support to enhance stormwater system performance should be investigated.

In order to move this work toward implementation within real-world systems, it may be valuable to explore more complex reward functions than the one used in this study. For example, it may be better to base the reward on different variables beyond flood volume and pond water depth. It may be the case that costs due to valve operation and drainage of ponds in specific cases are higher than a small amount of flooding that does not have societal impact. Additionally, the flow rates and velocities in the system may have additional restrictions to consider in the reward function (e.g., maintaining certain flow conditions for water quality or stream biota health or preventing flow velocities that can cause soil erosion). More complex reward functions can be explored in future work to account for more complex situations, move towards the control of real systems, and integrate specific characteristics of valves and ponds in real-world systems. Finally, this paper trained an RL agent on a single month of data. In future research the sensitivity of the algorithm to the amount and diversity of the data during training should be investigated. This will help construct a trade-off analysis between the amount of data needed, the training time required, and the accuracy of the predictive models needed for the training procedure.

3.6 Acknowledgments

This work was funded as part of two National Science Foundation grants: Award #1735587 (CRISP-Critical, Resilient Interdependent Infrastructure Systems and Processes) and Award #1737432 (SCC-IRG Track 1: Overcoming Social and Technical Barriers for the Broad Adoption of Smart Stormwater Systems). We acknowledge HRSD for continued access to their high quality data and the City of Norfolk for information regarding their stormwater retention ponds.

Chapter 4

Reinforcement Learning-based Real-time Control of Coastal Urban Stormwater Systems to Mitigate Flooding and Improve Water Quality[1]

4.1 Introduction

Communities rely on stormwater systems to mitigate flooding and treat polluted runoff from urban areas. However, as urbanization increases and climate change continues to alter precipitation, temperature, and sea levels, communities will be faced with increased stormwater runoff causing greater flooding and water pollution (Sweet and Park, 2014; Moftakhari et al., 2015; Moftakhari et al., 2017; Alamdari et al., 2020). Conventional stormwater systems are designed based on historic data assuming stationarity of future conditions. They are largely static systems, unable to dynamically adapt to unanticipated conditions. Increasing the resilience of stormwater systems to these unanticipated and changing land use and climate conditions will require new approaches to dynamically control both flood mitigation and pollutant treatment.

The adoption of smart cities approaches is allowing stormwater managers to begin to monitor and control individual components of conventional stormwater systems, which are gravity-driven and behave statically, in real-time (Kerkez et al., 2016). While the use of real-time control (RTC) is fairly established in combined sewer systems (Troutman et al., 2020; Kroll et al., 2018; Montestruque and Lemmon, 2015), recent research has shown that retro-fitting conventional stormwater components (e.g., a retention pond) for RTC can allow more efficient local operation, mitigating flooding from storms (Sadler et al., 2020a; Bowes et al., 2020) and preventing erosive, high velocity flows (Wong and Kerkez, 2018). RTC can also provide more efficient treatment of pollutants such as sediment and nutrients, primarily through increased detention time (Marchese et al., 2018; Shishegar et al., 2019). For instance, RTC of a retention pond increased removal of total suspended solids (TSS)

[1]This chapter is in preparation for submission to a peer reviewed journal.

and nitrate (NO_3) by roughly 40%, compared to passive pond operation (OptiRTC and Geosyntec Consultants Inc., 2017).

In practice, stormwater RTC is generally performed using local rule-based control (RBC), which is almost exclusively based on volumetric data (e.g., depth, current and forecast rainfall) (OptiRTC and Geosyntec Consultants Inc., 2017; Muschalla et al., 2014; Gaborit et al., 2013). For instance, a rule may open a valve when the water level in a storage pond reaches a certain height or proactively drain water from a pond based on a rainfall forecast to create additional storage capacity before a large storm. In most studies using RBC, water quality is not considered or is inferred through hydraulic retention time, rather than directly observed or used in control rules. However, pollutant characteristics are highly variable between sites and storms and there is a need for more generalizable RTC methods for enhancing pollutant treatment. Toward this end, the benefits of using real-time water quality observations in control rules has recently been explored in simulation. For example, using the concentration of TSS to trigger a valve controlling outflow from a storage pond can improve TSS capture in the pond compared to the passive system and other volumetric control rules (Sharior et al., 2019). Given the effectiveness of RTC-enabled individual infrastructure components to adapt to different storm events, system-level RTC has the potential to more holistically enhance flood and pollution mitigation through coordinated control of multiple components (Mullapudi et al., 2017).

As the complexity of controlled stormwater systems increases, the task of creating rules to (i) mitigate flooding, (ii) protect the quality of receiving waters, or (iii) balance both flooding and water quality, becomes nontrivial. Instead of attempting to engineer rules that cover all possible interactions between stormwater system components, pollutants, and environmental conditions, recent research has explored system-level methods of optimizing stormwater RTC. For instance, in a coastal urban stormwater system, model predictive control has been shown to reduce total system flooding, even under sea level rise conditions (Sadler et al., 2020b). In terms of water quality, Mullapudi et al. (2017) demonstrated that rules controlling flow from ponds to a treatment wetland increased the efficiency of nitrate removal by 46%. While these studies illustrate the vast potential of stormwater RTC, work remains for system-level optimization of both water quantity and quality. Shishegar et al. (2021) developed a system-level RTC method that controls outflow from retention basins as a linear optimization problem while controlling water quality with rules to extend detention time (i.e., hold water after a storm for a set amount of time). While this work presents a significant step for system-level RTC, direct observation and system control based on real-time water quality measurements was not included. Continuing improvements in real-time water quality sensors, however, are now allowing control to move beyond simple rules and heuristics to more direct observation and control of not only water quantity but also many water quality parameters (Wong and Kerkez, 2016; Chen and Han, 2018) .

Recent advances in machine learning provide an alternate approach to system-level stormwater RTC where control policies can be learned, instead of assumed. Deep Reinforcement Learning (referred to as RL here) is a type of machine learning which aims to learn from trial-and-error experience through interaction with an environment (Sutton and Barto, 2018). In RL, an agent (i.e., algorithm) does not have known answers to learn from, but instead is rewarded based on how well its control actions meet specified stormwater system goals (e.g., flood mitigation, improved water quality). The reward signal is used to guide the agent's learning towards actions that maximize the return from the reward function. This approach to learning allows RL increased flexibility to optimize control actions and has the potential to continually adapt system controls to evolving environmental conditions (e.g., increased runoff from urbanization or climate change).

Initial research with RL for stormwater system control shows promise in terms of flood mitigation and peak flow reduction (Bowes et al., 2020; Mullapudi et al., 2020; Wang et al., 2020), while being robust to uncertainty in sensed and forecast environmental data (Saliba et al., 2020). In Bowes et al. (2020) a system-level RL agent outperformed local RBC in reducing total flood volume of a conceptual coastal stormwater system. However, that research did not consider any water quality observations or impacts of the RTC methods. Given real-time water quality observations, RL may be able to learn to balance competing water quantity and quality goals throughout a stormwater system. No previous research has been done with RL RTC for the combined goals of flood mitigation and water quality protection. Therefore, this paper aims to illustrate RL's ability to learn system-level control policies considering these two objectives.

4.2 Methods

This research compares RL and RBC for their ability to both mitigate flooding and improve water quality compared to conventional static stormwater infrastructure. A simulation of Norfolk, Virginia's stormwater system including water quantity and quality processes is used as the controlled system. Two methods of local-scale, rule-based control are implemented: (i) predictive RBC with a fixed detention time and (ii) RBC based on water quality observations. RL is implemented for system-level control that incorporates measures of water quality and flood mitigation. After comparing the performance of these methods, their robustness to changes in system behavior is evaluated by simulating groundwater exchange with the controlled ponds.

4.2.1 Study Area

The City of Norfolk, Virginia, specifically its Hague neighborhood, is used as the study area for this research. Norfolk is situated near the mouth of the Chesapeake Bay on the eastern coast of the U.S. (Fig. 4.1). The city has a high rate of relative sea level rise partly due to regional land subsidence (Eggleston and Pope, 2013) and its low elevation, flat topography, and regular hurricane season contribute to increasingly frequent and severe recurrent flooding (Sweet and Park, 2014). The Hague neighborhood is a historic part of Norfolk and is adjacent to many city government buildings and the region's main hospital; the Hague also experiences some of the most frequent flooding in the city (Sadler et al., 2018; Sadler et al., 2020b). Additionally, Norfolk has a high groundwater table that responds quickly to storm events (Bowes et al., 2019) and could contribute significant amounts of water to retention ponds that are being actively controlled (Bowes et al., 2020). The quality of stormwater runoff from the city contributes to the health of the Chesapeake Bay, which has a long history of impairments such as hypoxia caused by eutrophication (Chesapeake Bay Foundation, 2018; Murphy et al., 2011). Pollutants carried by the city's stormwater (such as TSS, nitrogen, and phosphorous) are regulated to meet the Total Maximum Daily Loads (TMDLs) set for the Bay.

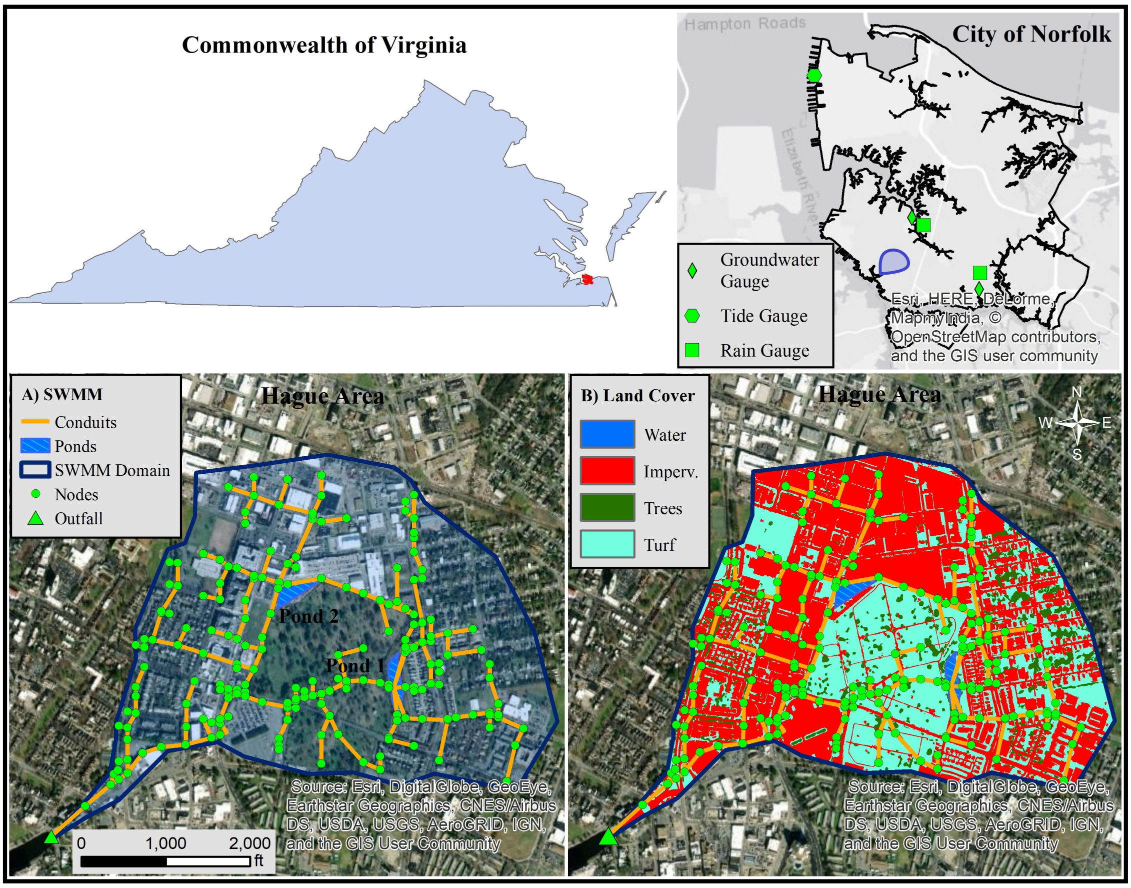

Figure 4.1: Study area - Hague area of Norfolk, Virginia USA with (A) the SWMM model and (B) land cover data.

4.2.2 SWMM Model

The Hague's recurrent flooding prompted Norfolk to build a simulation of the existing conventional stormwater system using the U.S. Environmental Protection Agency's (EPA) Stormwater Management Model (SWMM) (Fig. 4.1, A). This SWMM model was calibrated to match observed flooding in the Hague from Hurricane Matthew, which caused wide-spread flooding in October, 2016. The Hague SWMM model was updated by Sadler et al. (2020) to simulate real-time control infrastructure (i.e., an additional retention pond and a valve, pump, and inflatable dam). In the current study, the SWMM simulation from Sadler et al. (2020) is driven by long-term observed rainfall with a tidal boundary condition and has been enhanced to include groundwater and water quality processes. SWMM input files with full configuration details can be found in the open source code repository (see Section 4.6).

Input Data

Observed rainfall, tide, and groundwater data were collected from gauges in Norfolk for the period between 1 January, 2010 and 6 November, 2019 (Fig. 4.1). Fifteen minute rainfall data came from two stations near the Hague that are operated by the Hampton Roads Sanitation District (HRSD). Rainfall data is processed by first removing any values over the 1000-year 15-minute value for Norfolk (2.33in); these large values represented less than 0.01% of the rainfall datasets. Any missing values from one rain gauge are filled with the value from the other gauge if available; there were no periods were both rain gauges were missing data. Finally, the mean of the two rain gauges is taken to create a single time series for the SWMM model. Observed 6-minute tide data came from the Sewells Point gauge operated by the National Oceanic and Atmospheric Administration (NOAA). Tide data are referenced to the North American Vertical Datum of 1988 (NAVD88) and were resampled to an hourly interval for use as a SWMM boundary at the stormwater system outfall.

Forecasts for use in the RTC control methods were created from the observed data. A single forecast is an array of values representing the rainfall or tide measurement over the next n time steps. In this work, a 24 hour forecast of 15 minute rainfall contains n=96 values. Because the focus of this work is on comparison of the RTC scenarios, the forecasts were assumed to represent perfect knowledge.

Groundwater Exchange Simulation

Groundwater data was collected from two shallow monitoring wells operated by HRSD and referenced to NAVD88. Outliers from these data were removed with a Hampel filter (as in Bowes et al. (2019)) to remove large erroneous values and replace them with the median of a one-day rolling window. Groundwater observations are then aggregated to an hourly time step. A single time series for the Hague area was interpolated using inverse distance weighting between Pond 1, the two groundwater monitoring wells, and the tidal level at the stormwater system outfall (assumed to be equal to the groundwater table level at the land/water interface). The groundwater table is higher than the water level in Pond 1 93.7% and lower than Pond 2 73.8% of the 2010-2019 dataset; the groundwater table level is only below the bottom of the ponds 0.09% of the 2010-2019 dataset.

The Hague SWMM model provided by the City of Norfolk did not originally simulate groundwater processes and was not configured to easily allow simulation of groundwater exchange with the controlled ponds using SWMM's aquifer components. To address this, a conceptual model of the unconfined aquifer surrounding the existing Hague pond (Pond 1) was developed. Groundwater exchange was calculated externally from the SWMM simulation using the Dupuit equation and added (or subtracted, in the case of infiltration) to the pond as an inflow using pyswmm functionality (McDonnell et al., 2020). The Dupuit equation is commonly used to calculate exchange between a water body and an unconfined aquifer (Pells and N. Pells, 2016) and is written as

$$Q = \frac{K}{2L}(h_1^2 - h_2^2) \cdot A \tag{4.1}$$

where Q is the seepage rate into or out of the pond, K is the saturated hydraulic conductivity of soil surrounding the pond, h_1 and h_2 are the heights above a fixed datum for the pond water level and groundwater table level, respectively. L is the horizontal distance between h_1 and h_2, and A is the surface area over which seepage can occur (a function of pond water level).

Saturated hydraulic conductivity of the soil surrounding the existing pond (Pond 1) was estimated from the National Resource Conservation Service (NRCS) Web Soil Survey as 1.96ft/day. This soil is classified as a fine sandy loam with 61% sand, 22% clay, and 17% silt. Values for h_1 were based on SWMM's simulation of pond water level and h_2 was the observed groundwater table level. The sensitivity of groundwater exchange (Q) to the distance between measured water levels (L), was tested for $L = 25$, 10, 5, and 1ft using the SWMM model. A single value of L was chosen and used to demonstrate the impact of groundwater exchange on flooding and water quality with the control methods.

Water Quality Simulation

Water quality processes, specifically for TSS, were modelled using SWMM's buildup, washoff, and treatment equations (Rossman and Huber, 2016). TSS was chosen for this study to allow comparison with previous RTC literature, and because it is straight-forward to simulate (through gravitational settling) and known to carry other sorbed pollutants (Guan et al., 2018). Pollutant buildup within each subcatchment is modelled as a power function

$$B = min(C_1, C_2 \cdot t^{C_3}) \tag{4.2}$$

where B is the buildup of TSS (mass per unit area), C_1 is the maximum buildup possible, C_2 is the buildup rate (buildup per day), t is the antecedent dry period, and C_3 is a dimensionless buildup time exponent. Washoff of accumulated TSS from subcatchments is modelled with an exponential function

$$W = E_1 \cdot q^{E_2} \cdot B \tag{4.3}$$

where W is the washoff rate (mass per area per hr), E_1 is the washoff coefficient (per unit of rain), q is the runoff rate (per hr), E_2 is the washoff exponent, and B is the amount of built-up pollutant remaining. Treatment of TSS occurs in the retention ponds and is modelled as a first order decay based on a generalized settling velocity (similar to Sharior, et al. (2019)) with resuspension as a factor of depth and inflow velocity (inspired by Troutman et al., (2020))

$$C = \begin{cases} TSS \cdot exp(-v_s/DEPTH \cdot DT/3600)) & FLOW \leq \tau \\ TSS & FLOW > \tau \\ TSS \cdot (1 - exp(-v_s/DEPTH \cdot DT/3600)) & FLOW > \tau,\ DEPTH \leq \delta \end{cases} \tag{4.4}$$

where C is the TSS concentration (mg/L) in the pond after treatment, TSS is the inflow concentration, v_s is the generalized settling velocity (ft/hr), $DEPTH$ is the pond water depth (ft), DT is the SWMM routing time step (seconds), $FLOW$ is the inflow rate (cfs), τ is a flow threshold to distinguish when settling occurs, and δ is a depth threshold to distinguish when resuspension occurs (one quarter of the maximum pond depth in this implementation). Resuspension is included because RTC creates the potential for low water depths in retention ponds; if a pond is drawndown before high storm inflows, sediment may be resuspended and carried downstream.

Each land-use category within the SWMM model domain (Fig. 4.1, B) is given individual characteristics for the buildup and washoff processes. The SWMM pollutant processes were calibrated based on the annual loading and treatment of TSS in

Pond 1 (the existing pond) because no observed water quality data were available. TSS loading was estimated using the loading rates provided in Norfolk's Virginia Stormwater Management Permit (Norfolk, 2018). The treatment efficiency of the passive retention ponds was assumed to be 60% as specified in the Chesapeake Bay Program Established BMP Efficiencies (Virginia Department of Environmental Quality, 2015, Table V.C.1). The load into Pond 1 was calibrated using the buildup coefficient C_2 so that the mean annual load over 2010-2019 was within 2% of the estimated value. The treatment was calibrated using the flow threshold (τ) and the settling velocity (v_s) so that the mean annual reduction was within 5% of the estimated value for the passive simulation. While calibrating this SWMM model to observed values would be desirable, the scope of this paper is on comparison of the RTC methods and not exact quantification of TSS. Final values for the buildup, washoff, and treatment equations are specified in the SWMM input file.

4.2.3 Real-time Control Scenarios

Real-time control of the Hague stormwater system was simulated with three strategies and compared to the passive system. The three control strategies are (i) predictive RBC with a fixed detention time, (ii) TSS concentration-based RBC, and (iii) RL approaches that includes simulated real-time measurement of TSS concentration in the system state and/or reward function. In the passive system scenario, weirs control flow out of the retention ponds and maintain a permanent pool of approximately half capacity. In the RTC scenarios, the passive weirs are replaced with valves. The valve on Pond 1 is at the same elevation of the passive weir (due to pipe configuration constraints). The valve on Pond 2 is at the bottom of the pond side, which allows Pond 2 to be fully emptied or filled. Both RBC scenarios represent local (i.e., individual) control of the retention ponds, while RL can coordinate its control actions based on system-level information. The pyswmm Python package is used to implement all RTC scenarios.

Detention Rule-based Control

In this scenario, RBC is based on industry standard methods that use rainfall forecasts for predictive control of stored water to mitigate flooding, while controlling water quality with a fixed detention time (OptiRTC and Geosyntec Consultants Inc., 2017; Marchese et al., 2018; Wright and Marchese, 2017). The general process of this RBC (RBC-DTN) is shown in Figure 4.2 and detailed in Bowes et al. (2020). Briefly, if a forecast storm is expected to flood the pond, the valve will open to drain an equivalent volume of water (plus a safety factor). When the pond is drawndown sufficiently, the valve will close to retain the incoming runoff for a fixed time (24hr in this case). At the end of the retention period, the valve opens to the minimum setting to bring the water level back to the target operating depth within a fixed time (24hr). Outside of storm events, the valve operates in order to maintain a target depth in the pond.

TSS Rule-based Control

The TSS RBC (RBC-TSS) scenario was inspired by Sharior et al. (2019). Instead of using a fixed detention time, this RBC is innovative because it uses the real-

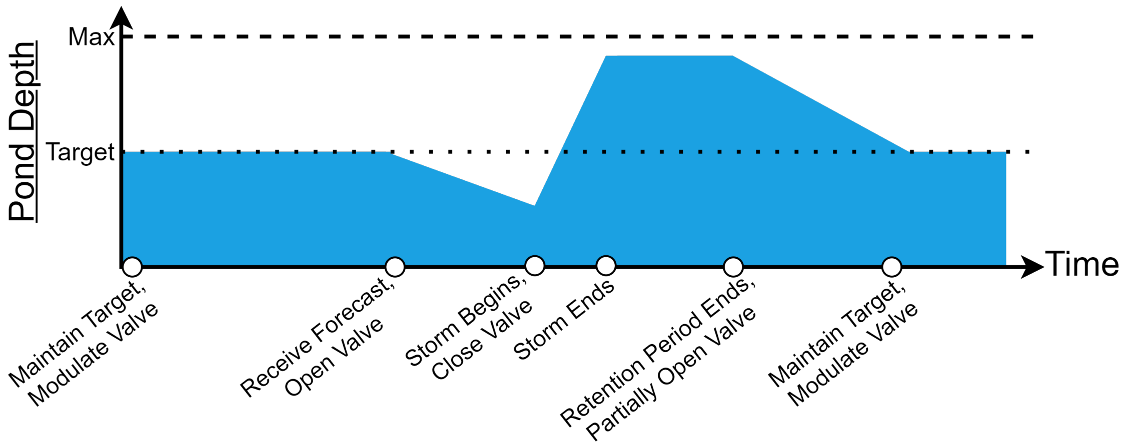

Figure 4.2: General schema of the Detention Rule-based Control (RBC-DTN) scenario. Forecasts allow predictive control of the pond water level to mitigate flooding while a fixed detention time after storm events helps improve water quality.

time concentration of TSS in a retention pond to trigger valve operation (Fig. 4.3). For example, when the TSS concentration is above a threshold, the valve can be closed to retain stormwater and allow treatment by settling. Otherwise, the valve is open and acts as a weir to maintain a permanent pool of water. In this study, the TSS threshold was set to 1 mg/L because observed data from the ponds were not available for a more realistic threshold; in Sharior et al. (2019), the threshold is 15 mg/L based on regulatory constraints for their study area. A contingency rule limits flooding of the pond by opening the valve if a threshold depth is reached.

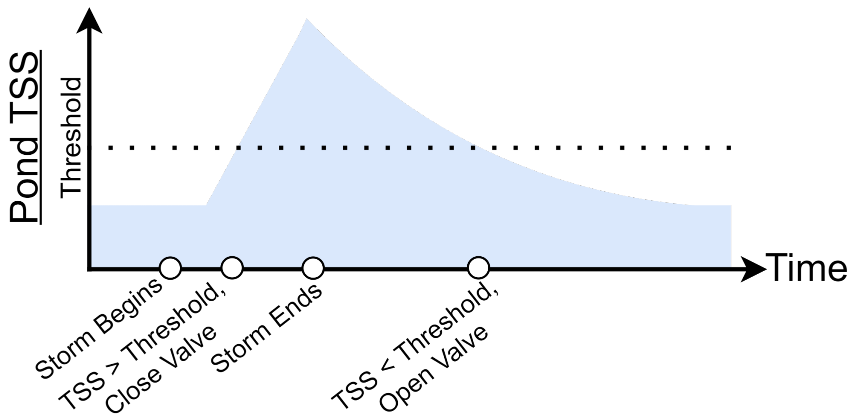

Figure 4.3: General schema of the TSS Rule-based Control (RBC-TSS) scenario. Detention is based on observed TSS concentration, not a fixed length of time, making it adaptive to individual storm events.

Reinforcement Learning

Reinforcement learning can be visualized as an agent that interacts with an environment (Fig. 4.4). The RL agent learns through sequential interactions with the environment. At each step in the learning process, the RL agent receives information about the state (s) of the environment and can take actions (a). The next state (s'), therefore, depends on the agent's actions and the agent is rewarded (positively or negatively) based on how well its actions meet user-specified objectives in a reward function (r). The agent's ultimate goal is to find a policy ($\pi(a|s)$) that maximizes the expected return

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{4.5}$$

where $r_t = r(s_t, a_t, s_{t+1})$ and $\gamma \in [0, 1]$ is a discount factor weighting the importance of short-term and long-term reward.

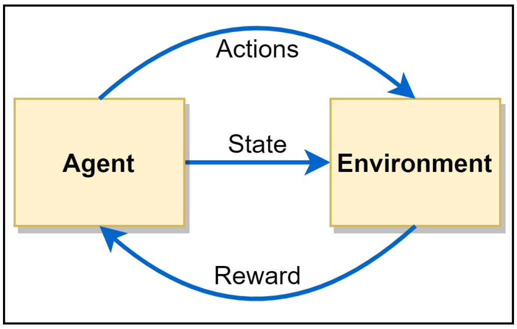

Figure 4.4: Reinforcement learning paradigm.

In this case the environment is the SWMM model described in section 4.2.2 and provides state information at a 15-minute simulation time step. The state space (S) is defined as: the current depths (ft) and outflow (cfs) of the two retention ponds, the concentration of TSS (mg/L) in pond outflow, the current valve positions, the sum of the 24 hr rainfall forecast (in), and the mean value of the 24 hr tide forecast (ft). The action space (A) of the agent is to open or close either valve to any degree. The reward (r) is based on how well the agent meets user-specified objectives such as flood and pollutant reduction.

The deep reinforcement learning algorithm used in this research, Deep Deterministic Policy Gradients (DDPG), is an actor-critic RL agent using deep neural networks as function approximators (Lillicrap et al., 2015). DDPG allows controls (i.e., valve positions) over a continuous action state and has been used in previous research to learn control policies that mitigate flooding (Bowes et al., 2020; Saliba et al., 2020; Wang et al., 2020). The actor in DDPG is a deep feed-forward neural network that learns a policy ($\pi(a|s)$); the critic is a deep feed-forward neural network that approximates the value of being in a specific state and taking specific actions called the Q-value

$$Q^{\pi}(s, a) = r(s, a, s') + \gamma \sum_{s' \in S} P^{a}_{s,s'} \sum_{a' \in A} \pi(a'|s') Q^{\pi}(s', a') \tag{4.6}$$

where $P^{a}_{s,s'}$ is the probability of transitioning between two states. This equation is known as the Bellman equation and is a key component of RL (Sutton and Barto, 2018). By approximating the Q-value, the critic can reduce the variance of policy gradients from the actor, which helps speed the learning process. During training, the actor receives the state of the stormwater system and outputs the actions to be taken based on its learned policy. The critic then receives the actions and states and outputs an estimated Q-value. The actions and Q-value estimates output from the critic are used to update the agent. An in-depth description of the DDPG algorithm can be found in Lillicrap et al. (2015).

In this research, three RL agents are trained and tested. The reward functions used by these agents have a conditional format that aims to simplify what the agent has to learn under different conditions (Bowes et al., 2020). Agent 1 is rewarded for reducing total flooding throughout the stormwater system and maintaining target pond depths

$$r = \begin{cases} -\Sigma Flooding[system,\ Pond1 * 1000,\ Pond2] & F \geq \delta \\ -(|Pond1_{depth} - \tau| + |Pond2_{depth} - \tau|) & F < \delta \end{cases} \tag{4.7}$$

where $Flooding[system]$ is the incremental system flood volume, $Flooding[Pond1]$ is the flooding rate at Pond 1, and $Flooding[Pond2]$ is a binary reward (0 or 1000) for having Pond 2 depth above a level that causes upstream flooding (5.75ft). F is the sum of rainfall in a 24hr forecast, δ is the rainfall threshold (0.5in in this research), and τ is the target depth (6.0ft and 3.56ft for Ponds 1 and 2, respectively). Agent 2 is rewarded for reducing total flooding throughout the stormwater system, maintaining target pond depths, and minimizing the export of TSS from the ponds

$$r = \begin{cases} -\Sigma Flooding[system,\ Pond1 * 1000,\ Pond2] & \\ \qquad +TSS[Valve1,\ Valve2] & F \geq \delta \\ -(|Pond1_{depth} - \tau| + |Pond2_{depth} - \tau| & \\ \qquad +TSS[Valve1,\ Valve2] + Flooding[system/35000]) & F < \delta \end{cases} \tag{4.8}$$

where $TSS[Valve1, Valve2]$ is the incremental TSS load of the controlled valves. Agent 3 aims to balance Agents 1 and 2 by initializing the trained neural network weights and memory from Agent 1 and training for 50,000 additional time steps using the reward for Agent 2. This can be considered as pre-training for Agent 3, a common practice in deep machine learning to provide appropriate initial conditions and reduce computational time (for examples in hydrology see Read et al., 2019 or Jia et al., 2019).

The RL agents are trained on one month of data (August, 2019), which has the fifth highest monthly total rainfall (10.1in) of the dataset distributed across 7 storm events. The mean tide level in this month is 0.52ft, with a maximum value of 3.31ft from Tropical Storm Erin late in the month. A visualization of the training data is given in Figure 4.5. RL Agent 1 is trained for 100,000 steps of the training data with a discount factor (weighting of current and future rewards) of 0.5. RL Agents 2 and 3 are trained for 150,000 steps, when the pre-training from Agent 1 is considered for Agent 3, with a discount factor of 0.99. RL agents are tested on the remaining data (2010-2019). Each RL agent has the same neural neural network architecture; these and the shared RL hyperparemeters are documented in the open source code repository linked in section 4.6. The DDPG algorithm is implemented with the keras-rl (Plappert, 2016), openai gym (Brockman et al., 2016), and Tensorflow (Abadi et al., 2016b) python packages.

RTC Comparisons

The RTC scenarios are evaluated in three main comparisons as shown in Table 4.1. First, a baseline for flood mitigation and TSS reduction is first established by comparing the passive system and RL Agent 1. This comparison focuses on differences between the passive system and a system-level control strategy that aims to mitigate flooding, but does not consider water quality. Second, trade-offs between the RBC methods, which focus on flood and TSS mitigation at the pond scale, are compared to the passive system. Third, system-level control trade-offs with RL Agents 2 and 3, which considered both flooding and TSS in their training, are compared to the passive system and RL Agent 1. These three comparisons are made without simulating groundwater exchange to keep the focus on control actions and reduce computational expense. The impact of groundwater exchange is then examined on a subset of the data to evaluate its potential impact on RTC of the stormwater system.

Table 4.1: Comparisons of stormwater control scenarios.

Comparison	Control Method	
Baseline	Passive	RL Agent 1
Local	RBC-DTN	RBC-TSS
System	RL Agent 2	RL Agent 3

4.3 Results

4.3.1 Baseline Flood and TSS Control

Figure 4.5 illustrates how the passive system and RL Agent 1 respond to the storm events in August, 2019. Operation of Pond 1 is similar between these two methods because the controllable valve is at the same elevation as the fixed weir; water is released as soon as depth increases from a storm event. However, RL Agent 1 learned to close the valve when high tide levels caused backflow into the pond to prevent water level fluctuations (e.g., Aug. 26-27). RL Agent 1 learned to lower Pond 2's depth, which is fully controllable, before certain storm events (e.g., the Aug. 5 storm) while remaining close to the target depth during dry periods.

The system-level control policy learned by RL Agent 1 allowed it to reduce the total volume of flooding by 4.0% (19.1MG) compared to the passive system (Fig. 4.6, A). While RL Agent 1's training did not include any water quality information, it's policy does provide improved TSS capture at both ponds (i.e., lower loads at the valves). Compared to the passive system, RL Agent 1 reduced TSS by 15.1% (36,235lbs) and 14.8% (31,027lbs) a Valves 1 and 2, respectively (Fig. 4.6, B).

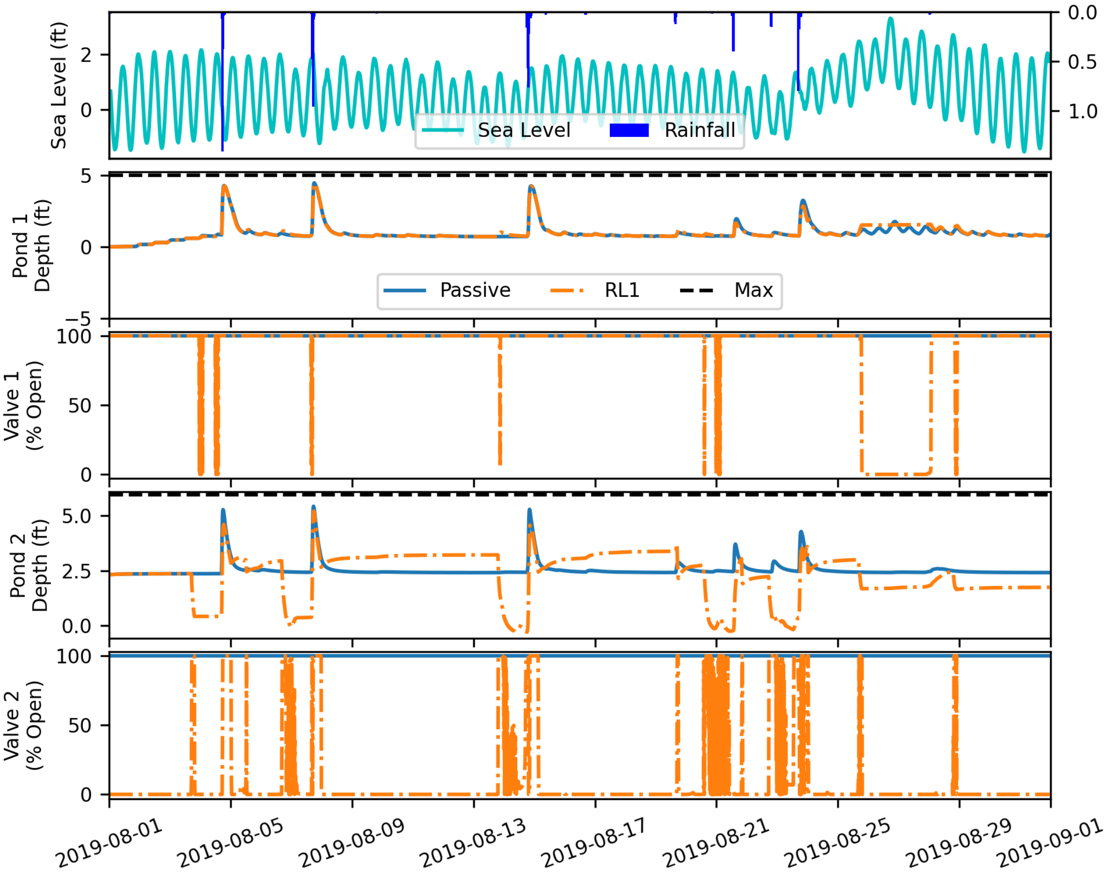

Figure 4.5: Comparison of passive and RL Agent 1 system operation for August, 2019.

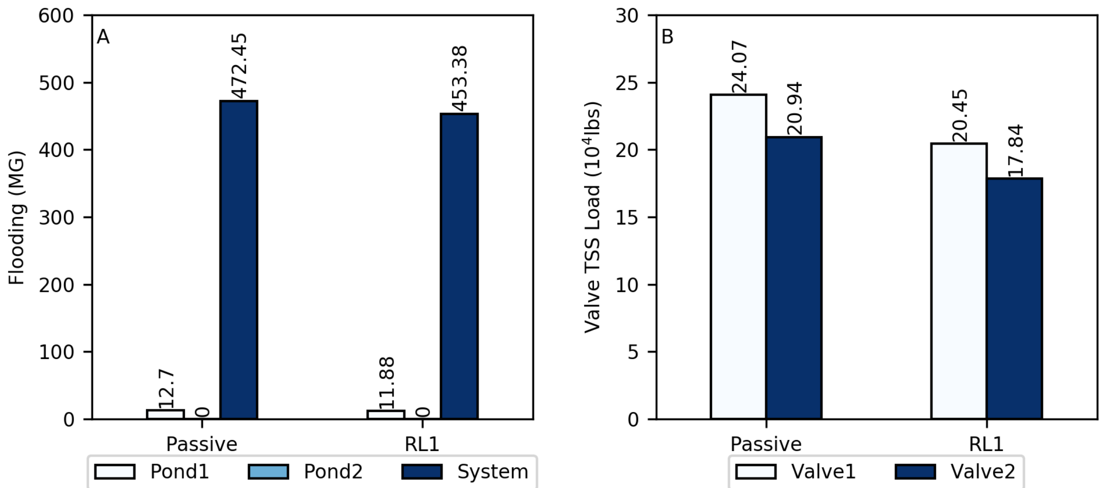

Figure 4.6: Total flood volumes (A) and TSS loads (B) for the passive and RL Agent 1 baseline scenarios, 2010-2019.

4.3.2 Local Control with RBC

An example of the RBC methodologies compared to the passive system is shown in Figure 4.7. Both RBC methods operate the ponds individually (i.e., rules are not coordinated between the ponds) to mitigate flooding of the pond by releasing water or to improve water quality by retaining runoff after a storm event. RBC-DTN has a fixed detention time, while RBC-TSS adapts detention time based on the concentration of TSS in the pond. For example, after the Aug. 4 storm RBC-TSS retains stormwater slightly longer than the fixed 24hr in RBC-DTN, but releases water sooner after the Aug. 8 storm. Because of the limited buildup time for TSS after the first storm, runoff did not have to be retained as long after the second storm. The opposite is seen in the Aug. 15 storm, where RBC-TSS holds water longer than RBC-DTN to provide more treatment after a long buildup period.

The two rule-based control methods both provide reductions in TSS export from the controlled ponds compared to the passive system. However, this is at the expense of increased flooding because operation of the two valves is not coordinated and does not consider flooding in other parts of the stormwater system (Fig. 4.8). Compared to the passive system, RBC-TSS increased total system flood volume by 12.0% (56.8MG), while decreasing TSS by 95.5% (229,770lb) and 32.8% (68,600lb) at Valves 1 and 2, respectively. RBC-DTN increased flooding by 9.0% (42.6MG) and decreased TSS for Valves 1 and 2 by 49.2% (118,410lb) and 4.5% (9,320lb) compared to the passive system. RBC for Pond 2 does not treat TSS as efficiently as Pond 1 because water needs to be released if the Pond 2 depth exceeds 5.75ft; this is necessary to alleviate upstream flooding due to this SWMM model's specific pipe configuration.

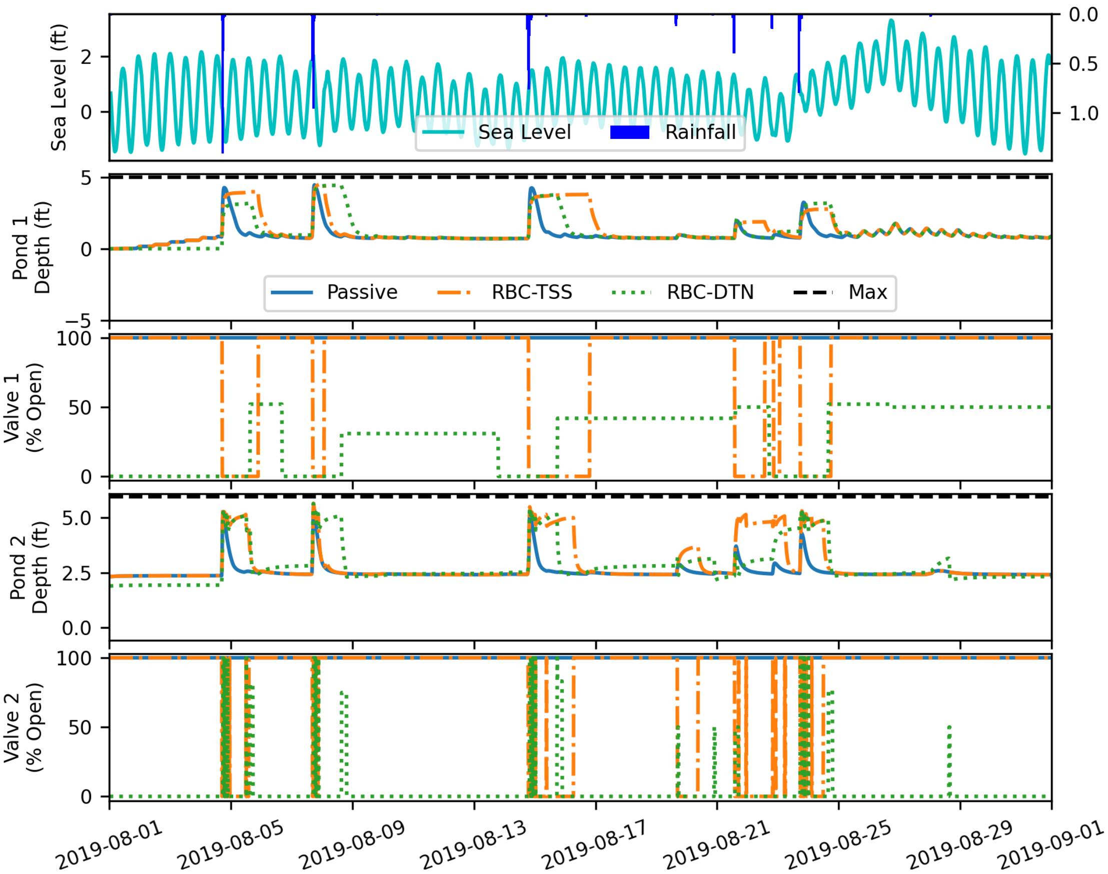

Figure 4.7: Comparison of local RTC methods (RBC-TSS, RBC-DTN) and passive system operation for August, 2019.

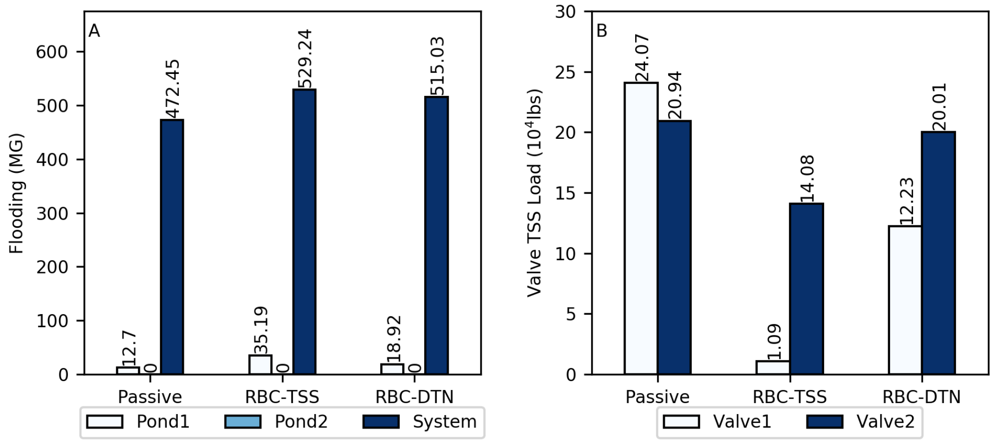

Figure 4.8: Total flood volumes (A) and TSS loads (B) for local RTC methods (RBC-TSS, RBC-DTN) and passive system operation, 2010-2019.

4.3.3 System-level Control with RL

Both RL Agent 2 and RL Agent 3 learned policies with multiple objectives of flood mitigation, TSS reduction, and target pond depths. When tested on the training data (Fig. 4.9), the Agents generally kept valve 1 open to maintain the target depth and closed valve 1 during storms to capture TSS. The agents have similar policies for valve 2 that favor holding water above the target depth to treat TSS while draining the pond before storm events to prevent flooding.

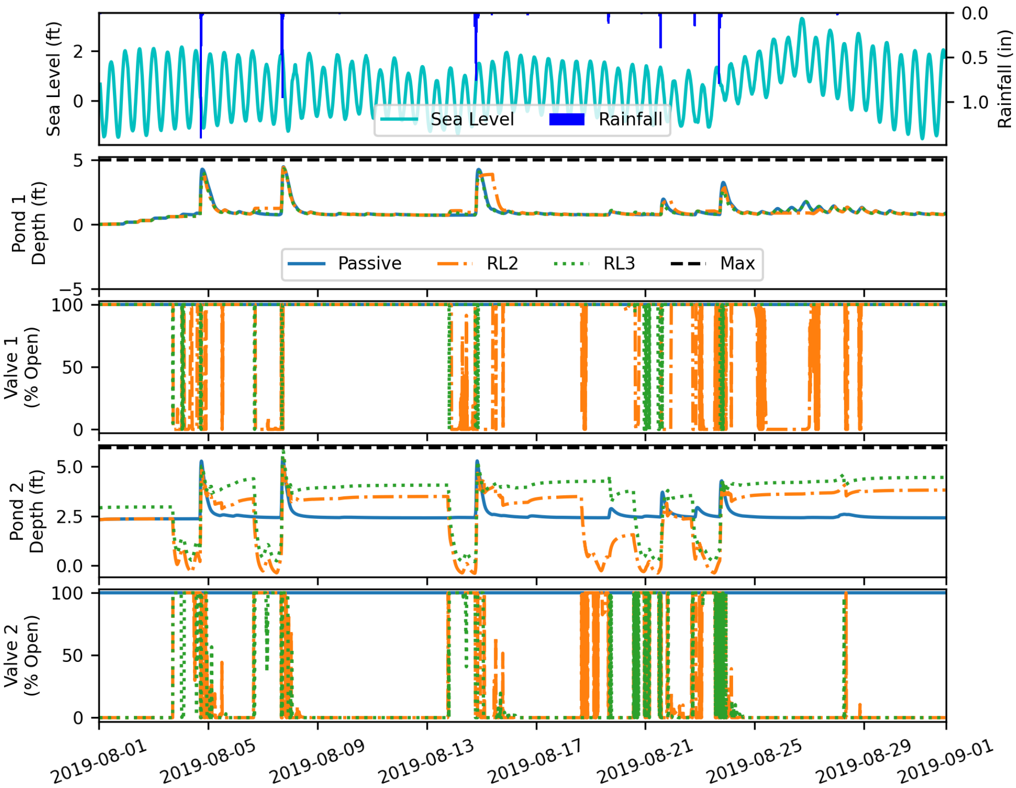

Figure 4.9: Comparison of RL Agent 2, RL Agent 3, and passive system operation for August, 2019.

On the test dataset (2010-2019), however, RL Agent 2 had 11.3% (56.2MG) more total system flooding and 74.6% (47.4MG) more flooding at Pond 1 than RL Agent 3 (Fig. 4.10). Both RL Agents 2 and 3 increased system-wide flooding compared to the passive system by 16.8% (79.3MG) and 4.9% (23.1MG), respectively. In terms of TSS reduction, both RL Agents provide improvements compared to the passive system. RL Agent 2 reduced TSS by 95.1% (228875.5lb) and 81.3% (170164lb) at valves 1 and 2, while Agent 3 reduced TSS by 39.5% (95083.5lb) and 65.0% (136027.5lb).

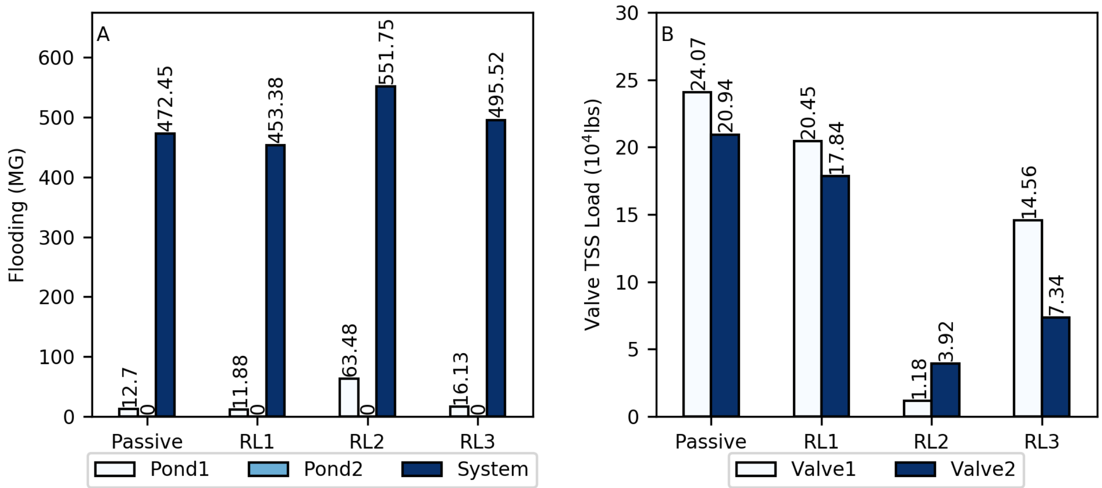

Figure 4.10: Total flood volumes (A) and TSS loads (B) for RL Agent 2 and RL Agent 3, 2010-2019.

4.3.4 Multi-objective Comparison of RTC Methods

A comparison of performance trade-offs for each stormwater control method is shown in Figure 4.11. In terms of flood volume, only RL Agent 1 reduced flooding compared to the passive system at both the system-level and at Pond 1. RL Agent 3 outperformed the local-scale RBC methods and RL Agent 2. Pond 2 did not flood in any of the scenarios because of the configuration of this SWMM model; several nodes upstream of Pond 2 have lower maximum depths and flood with any rainfall when the pond is above a certain level.

All RTC methods reduced TSS loads at both valves compared to the passive system. TSS load reduction at valve 1 was greatest for RBC-TSS and RL Agent 2; RBC-TSS used water quality observations to inform control, while RL Agent 2 learned a control policy from scratch that included penalties for high TSS loads. At valve 2, the local-scale RBC methods had fixed rules to release water when Pond 2's depth reached the threshold for upstream flooding. This limited their ability to capture the first flush of TSS during large storm events. The system-level RL agents outperformed the passive system and had similar trends in performance for both valves. RL Agent 1 did not consider TSS in its policy and had the smallest reduction, followed by RL Agent 3, and RL Agent 2 which had the greatest reductions in TSS.

In terms of maintaining the target depth at Pond 2, RBC-TSS was most similar to the passive system because the valve was at the same height as the target depth

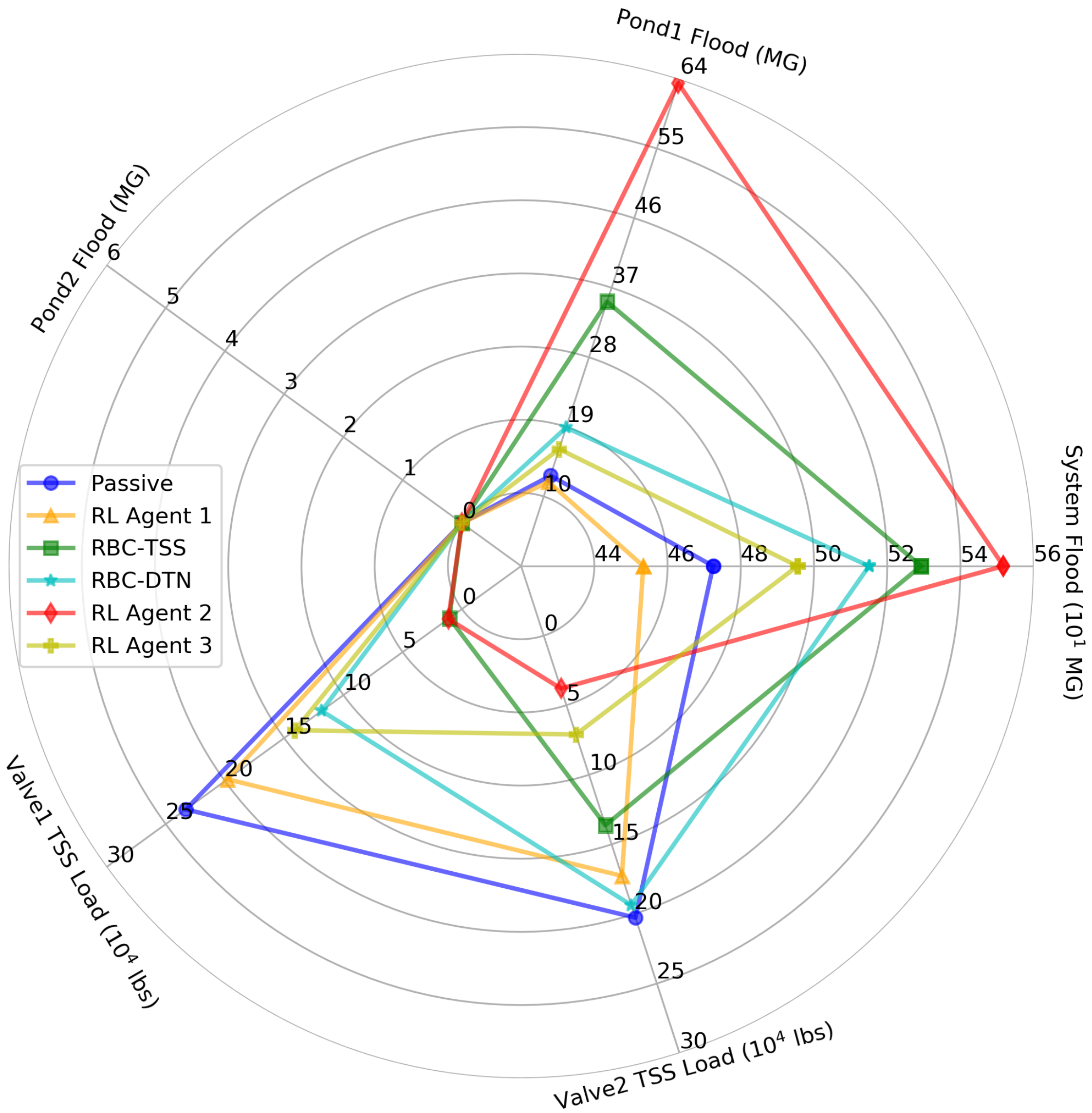

Figure 4.11: Comparison of flood volume and TSS load trade-offs for each control method, 2010-2019.

(Fig. 4.12). However, RBC-TSS was able to close the valve to treat TSS and therefore had a greater percentage of time above the target compared to the passive system. RBC-DTN and the RL Agents could fully drain or fill Pond 2 and had a greater percentage of time at lower depths. This helped prevent the pond from flooding, but long periods of time at low depths are undesirable in reality. The target depth comparison also illustrates differences in policy learned by RL Agent 2 and 3. Across the entire test set, RL Agent 2 had a tendency to keep Pond 2 at very low water levels. In contrast, RL Agent 3's policy kept the water level at or above the target depth approximately 90% of the time, indicating that it learned a policy to only drain the ponds when needed (a benefit of pretraining RL Agent 3 from RL Agent 1).

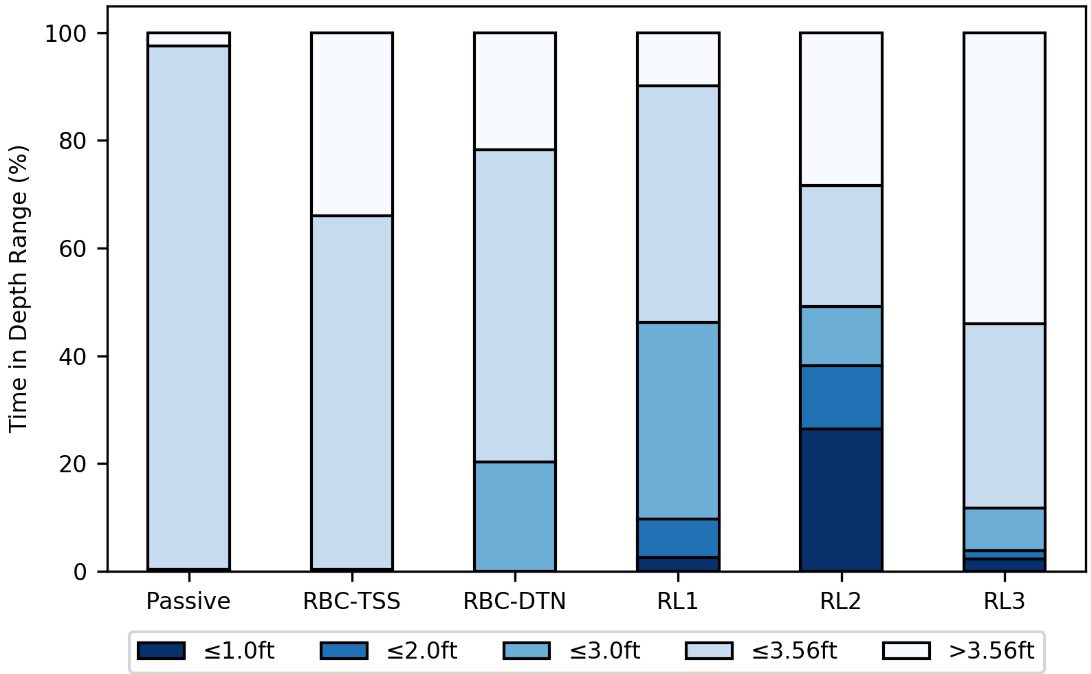

Figure 4.12: Comparison of time below or above the Pond 2 target depth (3.56ft) for each control method, 2010-2019.

4.3.5 Impact of Groundwater Exchange on RTC Methods

The impact of groundwater exchange with the controlled ponds was evaluated for the month of September, 2016 (an example time series visualization and statistics of valve operation by the RTC methods is available in Appendix G, Figs. 1 and 2). This month had two hurricanes and one tropical storm, which caused the groundwater table level to reach a height of 3.54ft (compared to the mean of 1.99ft). Because groundwater exchange also allows increased infiltration, these simulations have less flooding than without groundwater exchange. With less water flowing through the valves in the simulations with groundwater exchange, the TSS load is also reduced. To account for these changes, the total flooding and TSS loads of each RTC method are compared to the corresponding passive system results (with or without groundwater exchange) as shown in Figure 4.13.

All RTC methods have a smaller change in total flood volume compared to the passive system when groundwater exchange is included (with the exception of RBC-DTN, which had a larger percent change and reduced flooding, instead of increasing it) (Fig. 4.13, A). All RTC methods were still effective at reducing TSS loads for valves 1 and 2 (Fig. 4.13, B and C, respectively). Of the RBC methods, RBC-DTN had a smaller decrease in Valve 1 TSS load with groundwater exchange than without. RL Agent 2 was the only RL method to perform worse for TSS reduction when groundwater exchange was added to the simulation. This may indicate overfitting to the training data (which did not include groundwater exchange), limiting RL Agent 2's ability to control new pond behaviors.

In comparing the sensitivity of pond-aquifer flow to the Dupuit fitting parameter L, it was found that $L = 25ft$ and $L = 10ft$ had no noticeable impact on pond operation. When $L = 5ft$, Pond 1 tends to gain a small amount of water and Pond

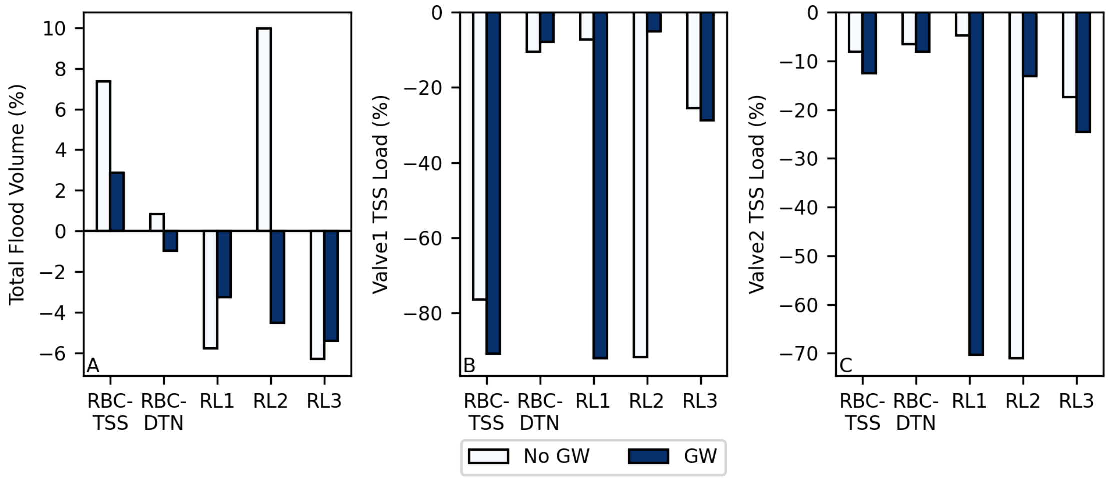

Figure 4.13: Comparison of percent difference from the passive system for each RTC method's total flood volume (A) and TSS loads (B and C) for simulations with and without groundwater (GW) exchange for September, 2016.

2 loses a small amount of water (Fig. 4.14, A and B, respectively). These trends are further amplified for $L = 1ft$; as L decreases, flow between the ponds and aquifer increases. Because $L = 1ft$ had the largest impact on pond level, it was chosen for use in the RTC simulation with groundwater exchange.

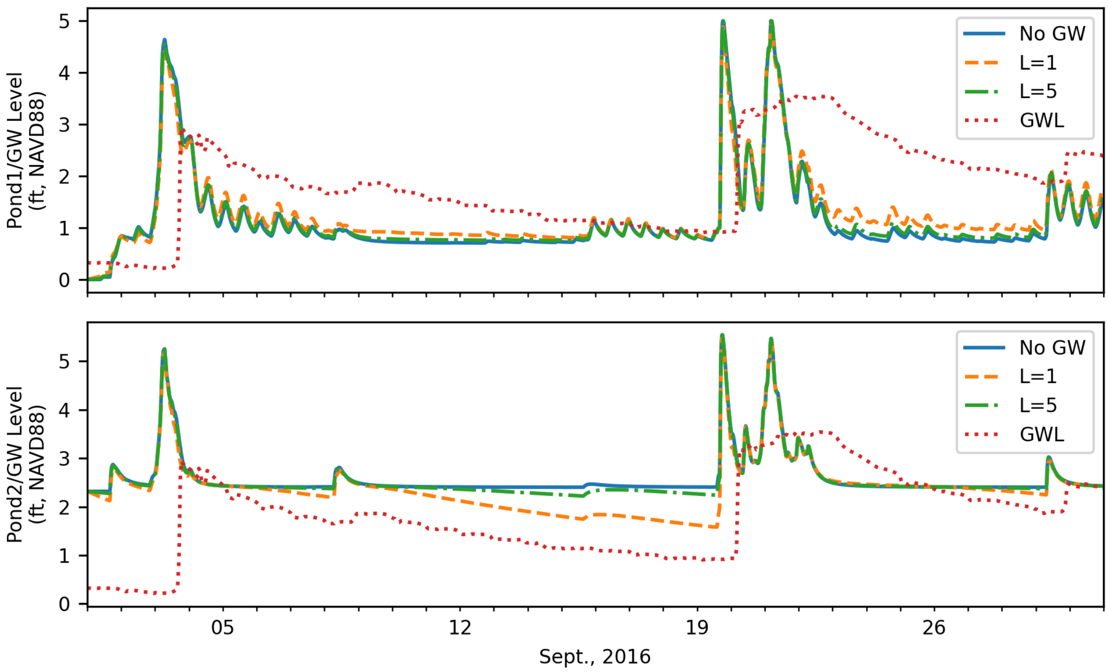

Figure 4.14: Comparison of passive pond operation for simulations without groundwater exchange (No GW) and with $L = 5ft$ or $L = 1ft$ in the Dupuit equation, September, 2016.

4.4 Discussion

4.4.1 Towards System-level Control

As the complexity of an environment and control objectives increases, it becomes much harder for a single RL agent to learn an effective control policy. This can be seen in the performance of RL Agents 1 and 2. Agent 1 had fewer goals and a simpler reward function that allowed it to learn an effective policy. In contrast, Agent 2 had a more complicated reward function and more goals. While it learned an effective policy for minimizing TSS, that was at the expense of both increasing system flooding and allowing Pond 2 to remain at undesirably low depths for long periods of time. As demonstrated by Agent 3, pretraining from an agent that performs well on simpler, but related, goals is one way to approach this challenge. Other methods such as Multi Agent RL (MARL), Multi-Objective RL (MORL), and boosting/ensemble methods may also be beneficial. In MARL, each pond could be controlled by an individual agent tuned to that pond's specific goals, while also operating cooperatively towards system-level goals (Su et al., 2020; Baldazo et al., 2019). In MORL, sets of policies are learned to approximate a Pareto frontier (Parisi et al., 2016); this is especially valuable for comparing trade-offs among agents. Similar multi-objective optimization is well studied for reservoir operation and could provide an alternative to MORL (Quinn et al., 2019). Boosting and other ensemble methods attempt to combine agent policies or neural network outputs to increase performance (Wiering and Hasselt, 2008; Wang and Jin, 2018). In the context of RL for stormwater systems, this maybe beneficial for combining agents that are trained for different purposes (e.g., an agent for extreme events, an agent for average events, an agent for dry periods).

Of the RTC methods implemented here, both RBC-DTN and the RL agents use current observations and forecasts to inform control decisions ahead of storm events. Perfect forecast data were used in this research to keep the focus on the control methodology. However, forecasts can contain a significant amount of uncertainty in reality. As an example specific to coastal systems, tide forecasts are based on the astronomical tide cycle which does not account for storm tides. In practice, RBC implementations have handled forecast uncertainty by using a probability threshold (e.g., take an action if the rainfall forecast probability is greater than 50%), as well as other fail-safes (OptiRTC and Geosyntec Consultants Inc., 2017). Stormwater RTC research using linear optimization and water quality control rules found that errors in rainfall prediction (i.e., an unforeseen storm event) could cause flooding of stormwater ponds, but that the system-level control could quickly adapt and recover based on observations of current conditions (Shishegar et al., 2021). Recent work with RL (specifically the DDPG algorithm used in this study) for stormwater RTC has indicated that this algorithm is robust to uncertainty in both sensed and forecast data (Saliba et al., 2020). In the current research, the RL agents were robust to altered pond behavior when groundwater exchange was simulated (groundwater exchange was not included in the RL training process). However, as stormwater RTC continues to move towards system-level control to accommodate the increasing density of controlled infrastructure components, changing environmental conditions, and more stringent environmental regulations, understanding the impact of sensed and forecast data uncertainty on RTC methods will be essential.

RL is known to suffer from issues including reward gaming, where the agent learns to exploit its environment in unintended ways to gain reward (Amodei et al., 2016). In the context of stormwater RTC, reward gaming was observed in early attempts at training RL agents related to simulation processes within the SWMM model. For example, flood water in the Hague SWMM model does not pond and reenter the stormwater system as it would in reality, but is simply recorded as flooding and lost from the simulation. One consequence of this model process is that any TSS within flood water is also lost from the system. If rewards are poorly shaped (i.e., TSS much more heavily weighted than flooding), the RL agent can learn policies that induce flooding because the rewards gained by the corresponding TSS reduction outweigh penalties for flooding. This highlights the need for domain specific knowledge when crafting reward functions and careful consideration of simplifications within simulated environments.

4.4.2 Trade-offs of Local-scale RBC

Both RBC methods used in this research performed RTC at the local-scale (i.e., operating each pond individually) and reduced TSS loads, but at the expense of increased system-level flooding. RBC-DTN showed similar TSS reductions for Pond 1 (49%) as previous studies in other locations (approximately 40% reported by Marchese et al., (2018)). However, as water quality sensor technology becomes less expensive and more robust, control based on water quality observations, such as the RBC-TSS implemented here, may provide a more adaptive solution. RBC-TSS reduced TSS by 96% for Pond 1 compared to the passive system, similar to the value found by Sharior et al., (2019) for a different site. The RBC methods did not perform as well for Pond 2 in this study due to the configuration of the SWMM model. Specifically, when water reached 5.75ft (which is less than the maximum depth), the contingency rules to prevent upstream flooding would open valve 2. Without this rule, the RBC methods greatly increased upstream flooding, but it also releases stormwater with high concentrations of TSS during large storm events.

The results of RBC demonstrate that fixed rules, like those used in RBC-DTN, may not provide the most efficient treatment because pollutants are highly variable between sites and storms (Wong and Kerkez, 2016). One solution could be the combination of the two RBC methods used here (e.g., the predictive drawdown capability of RBC-DTN coupled with adaptive detention time based on observed water quality as in RBC-TSS), but this is still limited as a local-control scheme. While adapting rules based on water quality may be fairly straight-forward for a single pollutant at a single site, controlling a stormwater system for multiple pollutants with different treatment processes (e.g., nitrogen species) will require system-level control (Mullapudi et al., 2017).

4.4.3 Groundwater Exchange Limitations and Impact

Due to the specific configuration of the studied SWMM model, groundwater exchange was calculated externally from the SWMM model and added (or subtracted in the case of infiltration) to the ponds' inflow at each control time step. While this process is based on in-situ soil properties for Pond 1 in Norfolk's Hague region, the Dupuit equation (which is intended for systems at a steady state) may not provide the most accurate representation of groundwater exchange. Under real-time control, ponds can be rapidly drained and refilled before and during a storm event. The Boussinesq equation for transient unconfined aquifer flow would provide a more realistic representation and is commonly implemented as a simpler alternative to Richards equation (see for example, Litwin et al., 2020). Coupling such a model with the SWMM model used here would allow for more precision, but as an initial demonstration of groundwater impact on ponds controlled in real-time, the Dupuit equation was quick to implement and run.

In the simulated RTC scenarios set up in this research, groundwater exchange with controlled ponds decreased flooding through infiltration; TSS loads were also reduced because less water was exiting the ponds through the valves. While groundwater interactions with the retention ponds in Norfolk have not been studied specifically, it has been demonstrated that increased groundwater table levels due to sea level rise could contribute to retention ponds in coastal areas, decreasing their ability to appropriately manage consecutive storm events (Davtalab et al., 2020). Because Norfolk has a high groundwater table and is already experiencing impacts from a high rate of relative sea level rise, considering the robustness of stormwater RTC methods to this will be increasingly important.

4.5 Conclusions

In this research, real-time control (RTC) methods were applied to a coastal stormwater infrastructure system and evaluated on their ability to mitigate flooding and improve water quality by capturing TSS in controlled retention ponds. The RTC methods used include local control with rules (RBC) and system-level control with deep reinforcement learning (RL). The impact of groundwater exchange on the performance of the controlled ponds was evaluated as a condition that may be important in coastal areas. This research contributes to the growing field of stormwater RTC by being the first to evaluate the ability of RL to learn system-level control policies considering both water quantity and water quality goals, as well as being the first to consider the impact of groundwater on the performance on controlled ponds in a coastal city.

Two methods of RBC were used (i) RBC-DTN, which is based on industry standard stormwater RTC and predictively manages ponds to prevent flooding while retaining runoff for a fixed detention time to improve water quality and (ii) RBC-TSS, which uses observations of water quality to inform valve operation in order to improve TSS capture. Both RBC methods are transparent and provide water quality benefits compared to the passive system. RBC-TSS provided more adaptive operation and demonstrates the potential for water quality observations to be incorporated with RTC as sensor technology improves. However, the local operation of both RBC methods caused increased total system flooding.

Three RL Agents were trained and tested for their ability to learn effective system-level control policies. The goal of RL Agent 1 was to mitigate flooding and maintain target pond depths; it reduced flooding compared to the passive system, but did not consider water quality in its control policy. RL Agents 2 and 3 attempted to learn policies for more objectives: mitigate flooding, maintain target pond depths, and reduce TSS loads at the controlled valves. RL Agent 2 learned a policy from scratch, while RL Agent 3 was pretrained by using the neural network weights and memory from RL Agent 1, but was trained to consider water quality as well using the reward function from RL Agent 2. Both RL Agent 2 and 3 provided water quality benefits but increased flooding compared to the passive system. RL Agent 2 decreased TSS loads by an average of 88%, but increased system-wide flooding by 17%. RL Agent 2's pretraining was effective at reducing training time and allowed it to learn a policy that reduced TSS by an average of 52%, with only a 5% increase in total flood volume, compared to the passive system.

Given the growing adoption of rule-based stormwater RTC and the ability of RL to learn system-level control policies, future work could investigate control of more complex stormwater systems and integrations of RL and RBC. More complex stormwater systems could include retention ponds in series, pollutants that are treated through chemical and biological processes (e.g., nitrogen)/multiple pollutants, and different controllable assets such as pumps. Integration of RL and RBC could include using RL to better parameterize variables within an existing control rule (see Likmeta, et al., 2020, for an example in autonomous vehicles), as well as adding or removing rules from a set of rules. These avenues for future research could allow stormwater RTC providers to increase the complexity of controlled networks, improving flood mitigation and water quality, while maintaining the operational transparency needed for critical stormwater infrastructure systems.

4.6 Data, Model, and Code Availability

The data, models, and code used in this study are available on GitHub at https://github.com/UVAdMIST/swmm_wq_rl.

4.7 Acknowledgments

This work was funded as part of two National Science Foundation grants: Award #1735587 (CRISP-Critical, Resilient Interdependent Infrastructure Systems and Processes) and Award #1737432 (SCC-IRG Track 1: Overcoming Social and Technical Barriers for the Broad Adoption of Smart Stormwater Systems). We acknowledge HRSD for continued access to their high quality data and the City of Norfolk for information regarding their stormwater retention ponds.

Chapter 5

Conclusion

This dissertation advanced understanding of the use of emerging machine learning techniques to forecast hydrologic processes within an urban setting and improve the ability of stormwater systems to mitigate flooding and protect water quality. Data from the City of Norfolk, Virginia, were used to demonstrate the applicability of these techniques to a flood-prone coastal urban environment. The key contributions from this research are (i) a method for creating real-time hourly forecasts of the groundwater table using deep machine learning, (ii) an evaluation of reinforcement learning to create real-time stormwater system control strategies for reducing flooding, and (iii) a novel method for using real-time water quality, rainfall, and tide observations in a reinforcement learning framework to balance flood mitigation and water quality goals with real-time stormwater control.

Chapter 2 of this dissertation demonstrated the importance of data quality on the predictive accuracy of groundwater table forecasts created with deep neural networks. Using a unique dataset of groundwater table observations from Norfolk, the LSTM type of neural network was trained on data from storm events to predict groundwater table level with an average RMSE of 0.05m. This machine learning approach can be used instead of more traditional physics-based models, which are often time and cost prohibitive to calibrate and run at the city scale. Chapter 2 contributed to urban hydrology and stormwater management through the first demonstration of LSTM for hourly groundwater table level forecasts in a coastal city.

Chapter 3 of this dissertation explored the application of deep reinforcement learning (RL), an emerging machine learning technique, to learn control policies for real-time stormwater system control. RL and other control methods were applied to a stormwater system simulation inspired by conditions in Norfolk where both rainfall and tidal level can impact flooding and retention pond depths. This chapter showed RL is able to learn system-level control policies that mitigated flood volume by 32% compared to the conventional passive system while also maintaining normal pond operation during dry periods. Compared to industry-standard control rules that operate retention ponds individually, RL was able to reduce flooding by 19% by coordinating the control of the retention ponds. Chapter 3 contributed to the growing field of stormwater RTC through the first demonstration of the applicability of RL to coastal stormwater systems for flood mitigation.

In Chapter 4 of this dissertation, the impact of stormwater system RTC at varying scales on flooding and water quality was evaluated. Using a simulation of

Norfolk's Hague neighborhood, system-level control with deep reinforcement learning was compared to two methods of local rule-based control: (i) industry standard rules that predictively managed ponds to prevent flooding while retaining runoff for a fixed detention time to improve water quality and (ii) rules based on observations of water quality to inform valve operation in order to improve water quality. The local-scale control rules improved water quality, but increased system-wide flooding. In comparison, RL learned system-level control policies that improved water quality with less flooding than the local control rules. Chapter 4 contributed to stormwater RTC through the first evaluation of RL's ability to learn system-level control policies considering both water quantity and water quality goals, as well as being the first to consider the impact of groundwater on the performance of real-time controlled ponds in a coastal city.

Collectively, the studies in this dissertation provide methodologies and demonstrations of leveraging machine learning techniques to better predict and control flooding and water quality in coastal cities. Each chapter has a corresponding open-source code repository to facilitate reproducibility and provide building blocks for future work. The methods and insights from this dissertation contribute to a growing body of knowledge about smart stormwater systems. Together, this research has the potential to help increase the resilience of coastal communities and protect natural ecosystems from increased flooding and pollution from urbanization and climate change.

Appendices

Appendix A: Groundwater Table Forecast Histograms

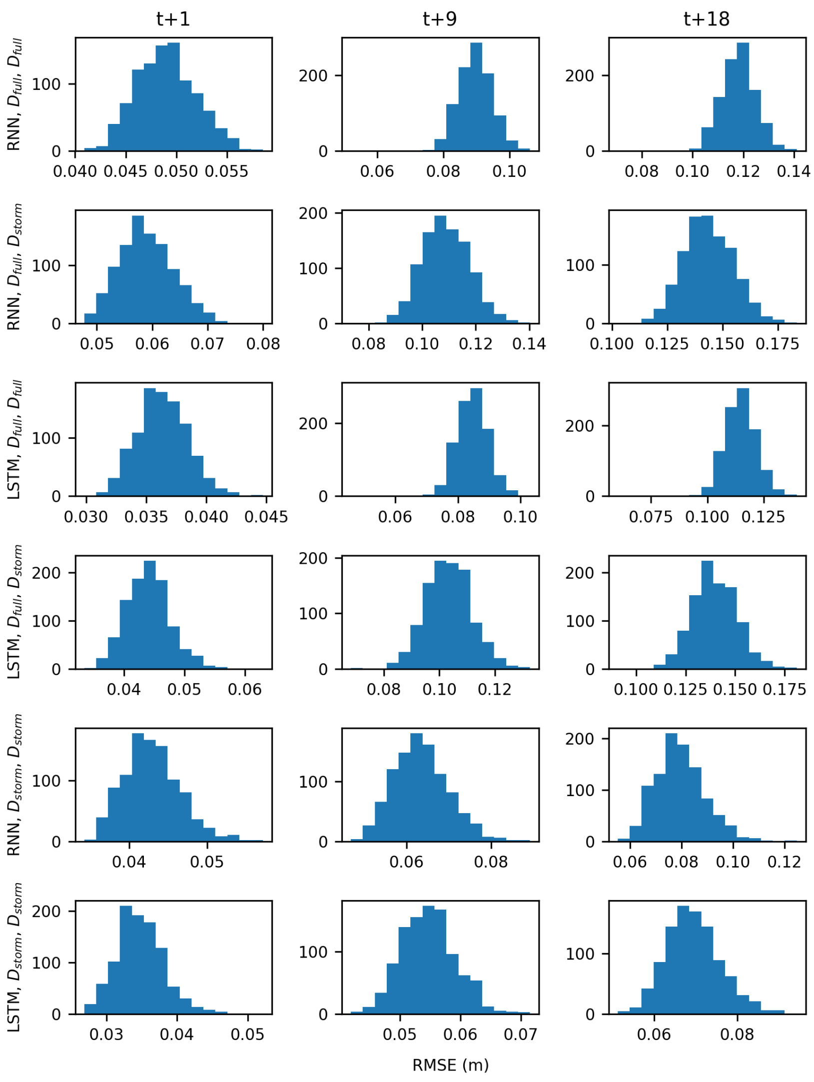

Figure 1: RMSE distributions for GW1 using observed data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

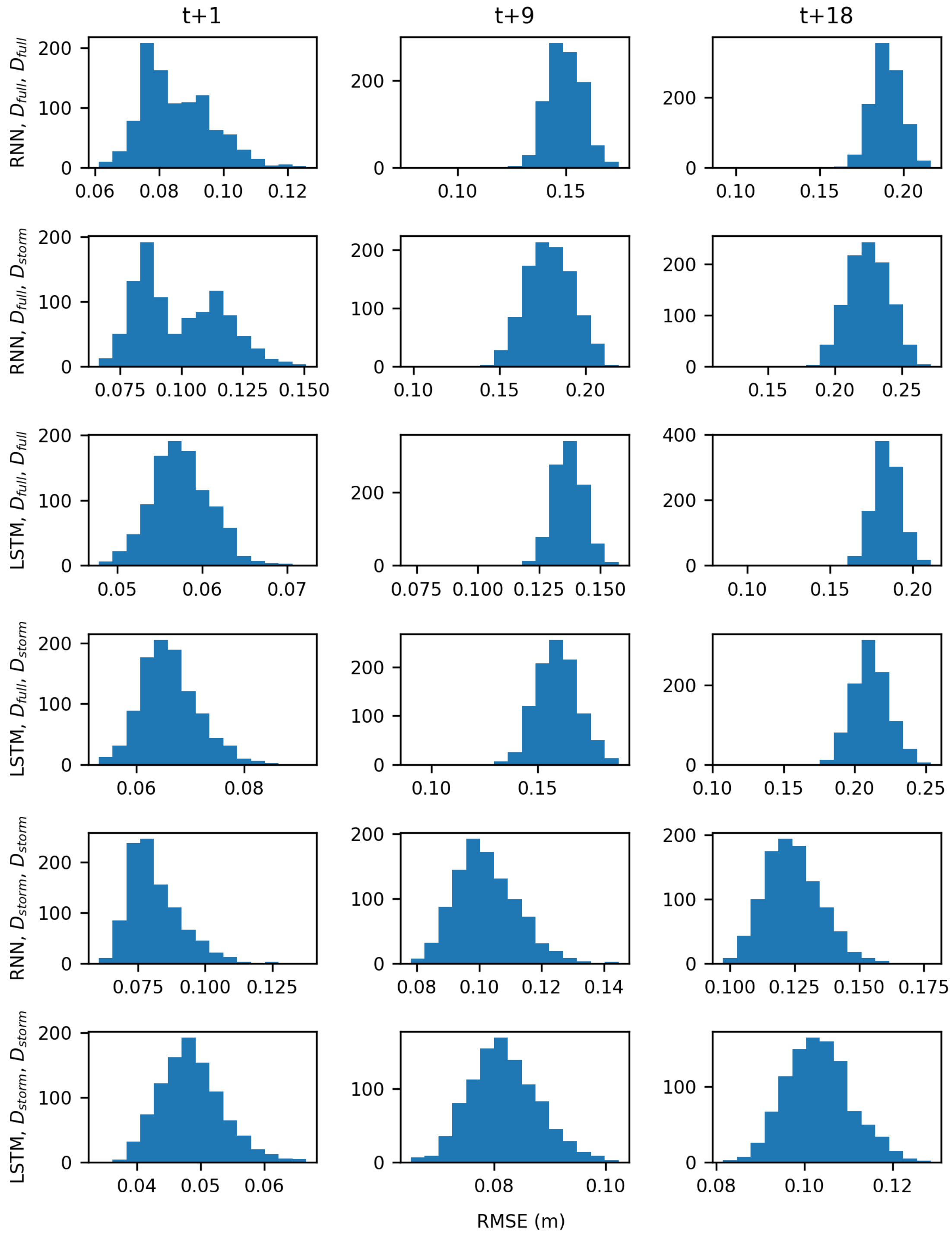

Figure 2: RMSE distributions for GW2 using observed data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

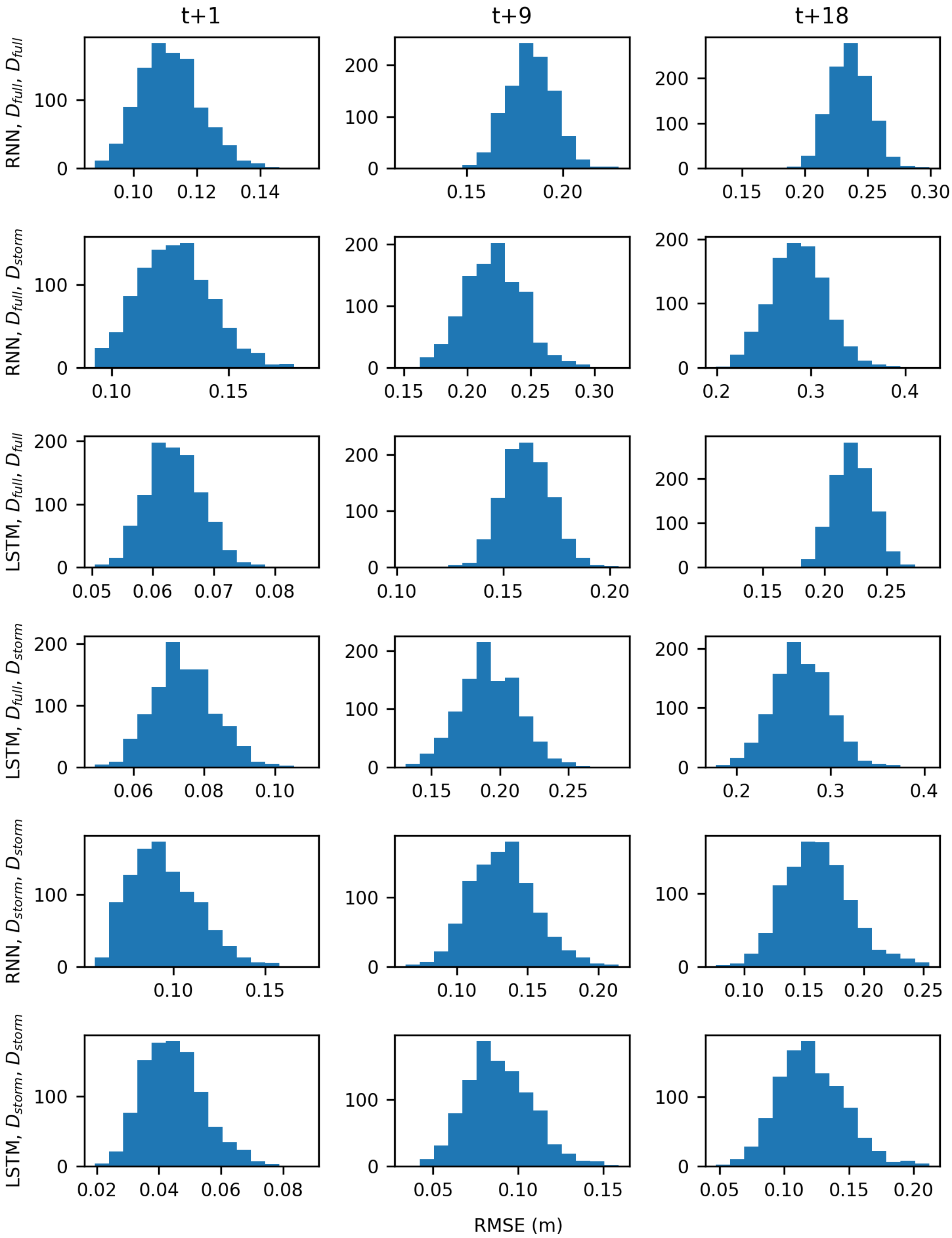

Figure 3: RMSE distributions for GW3 using observed data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

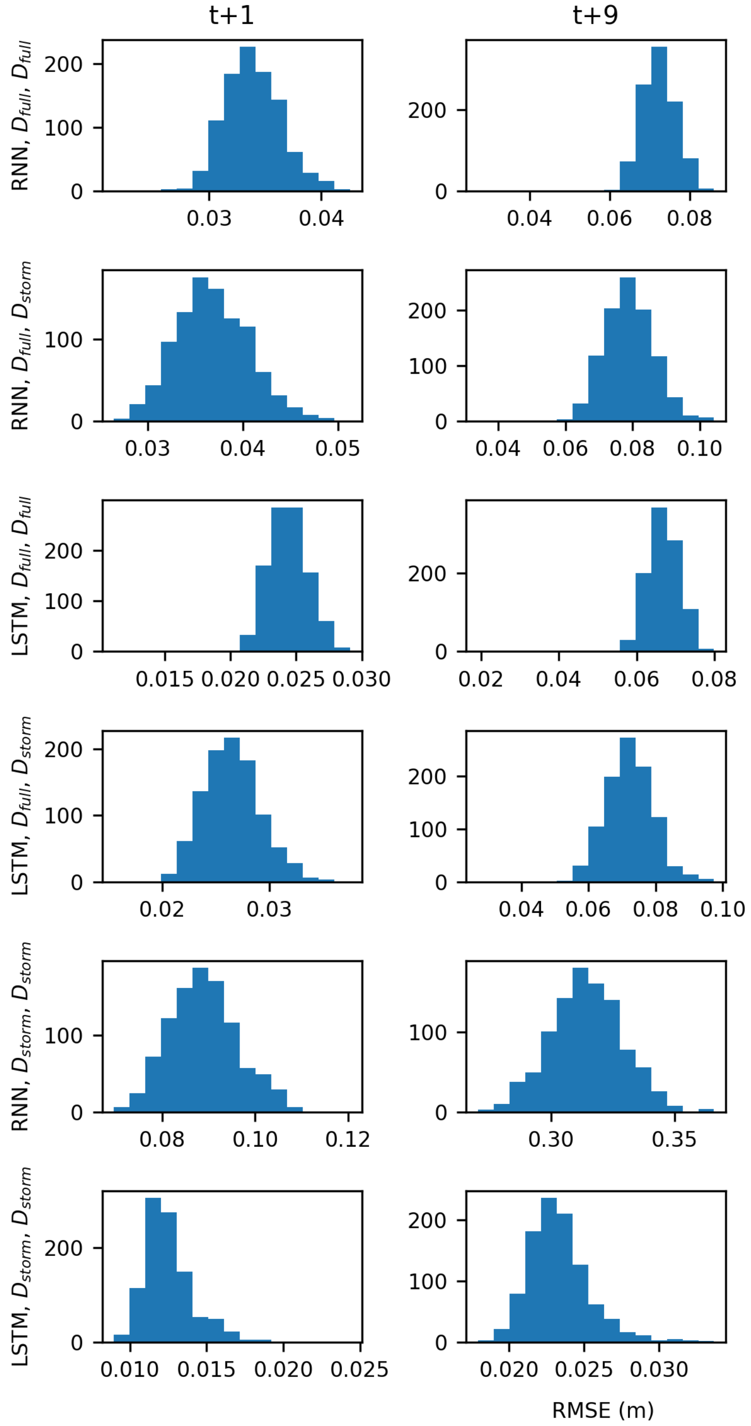

Figure 4: RMSE distributions for GW4 using observed data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

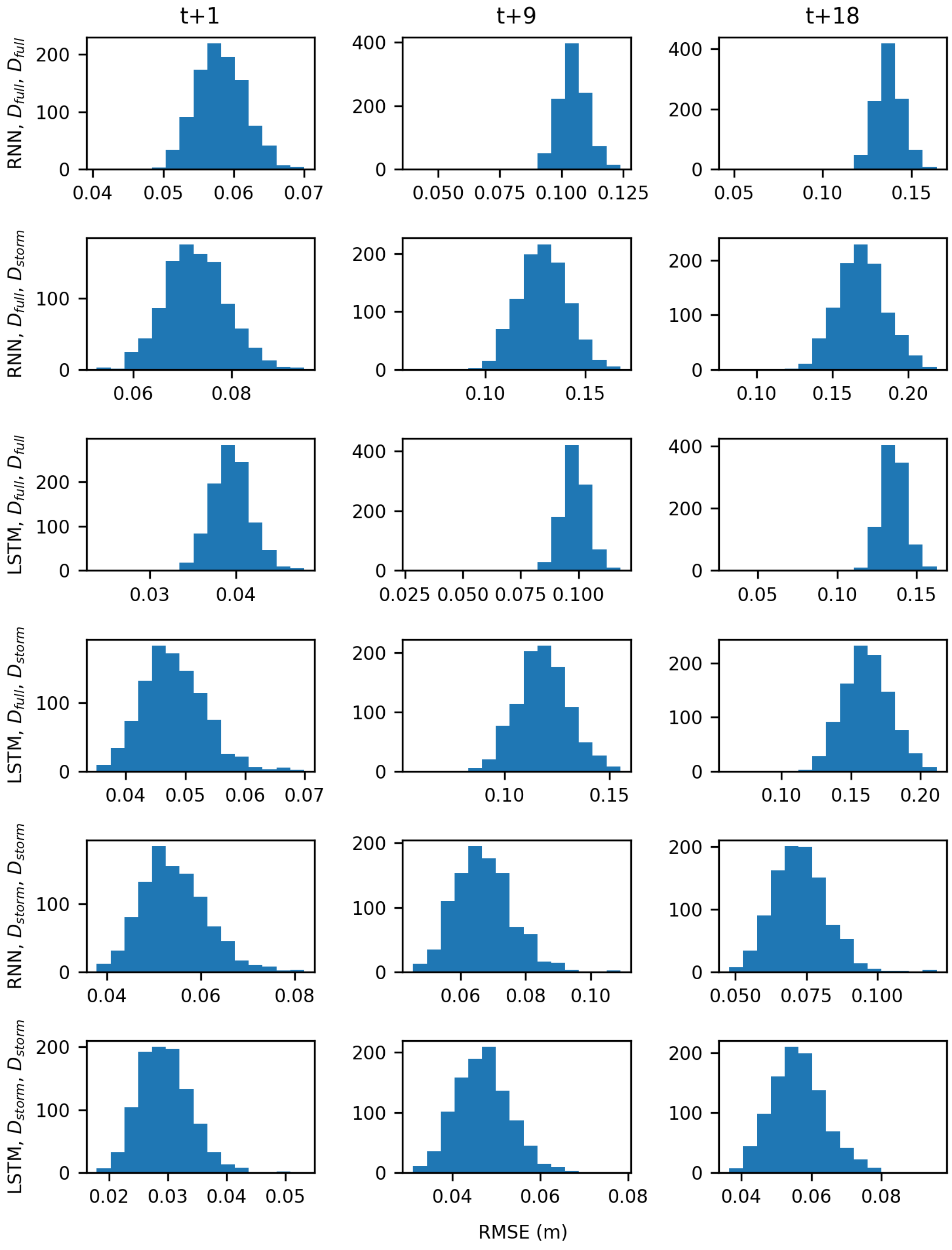

Figure 5: RMSE distributions for GW5 using observed data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

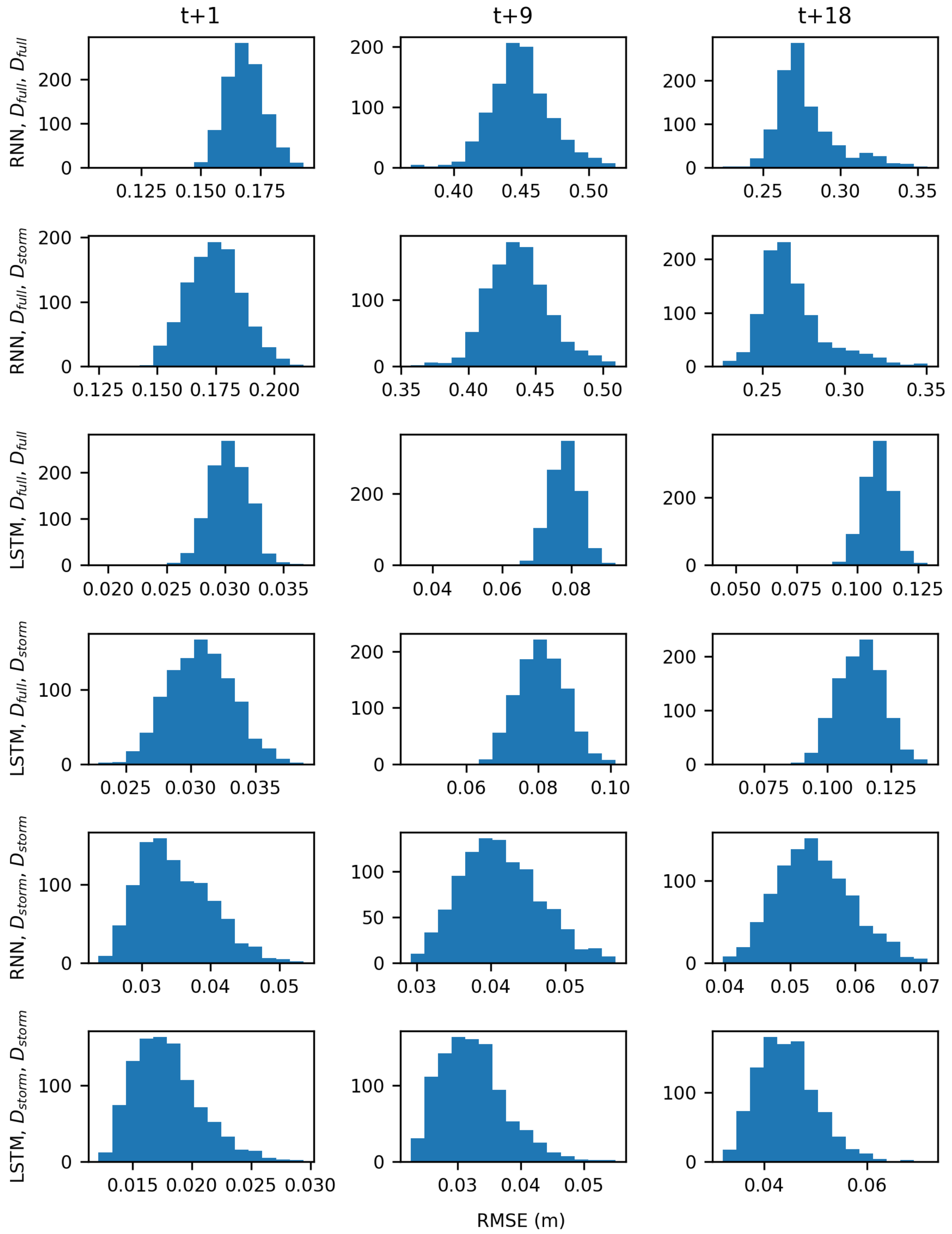

Figure 6: RMSE distributions for GW6 using observed data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

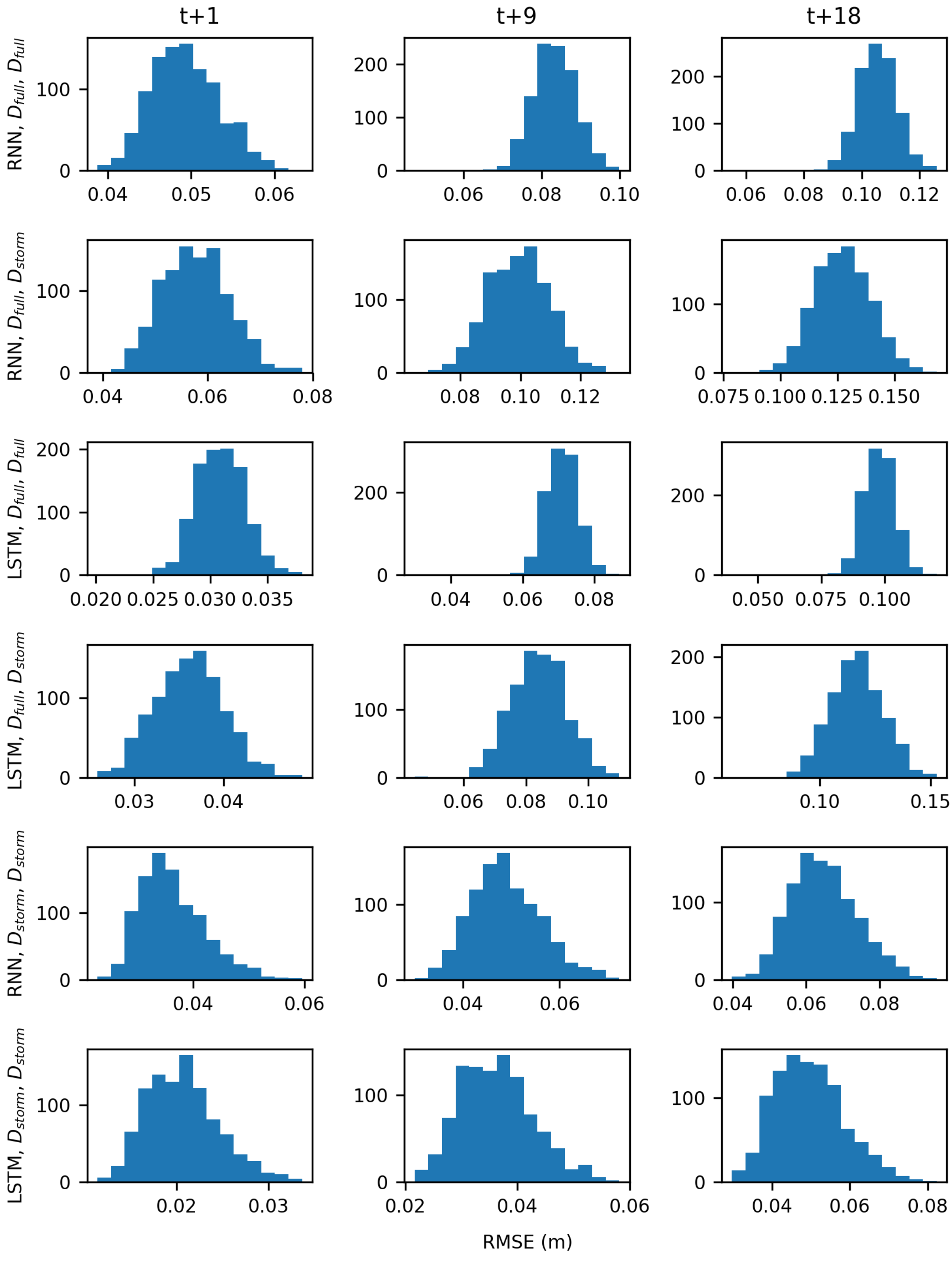

Figure 7: RMSE distributions for GW7 using observed data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

Appendix B: Groundwater Table Forecast Histograms, D_{fcst}

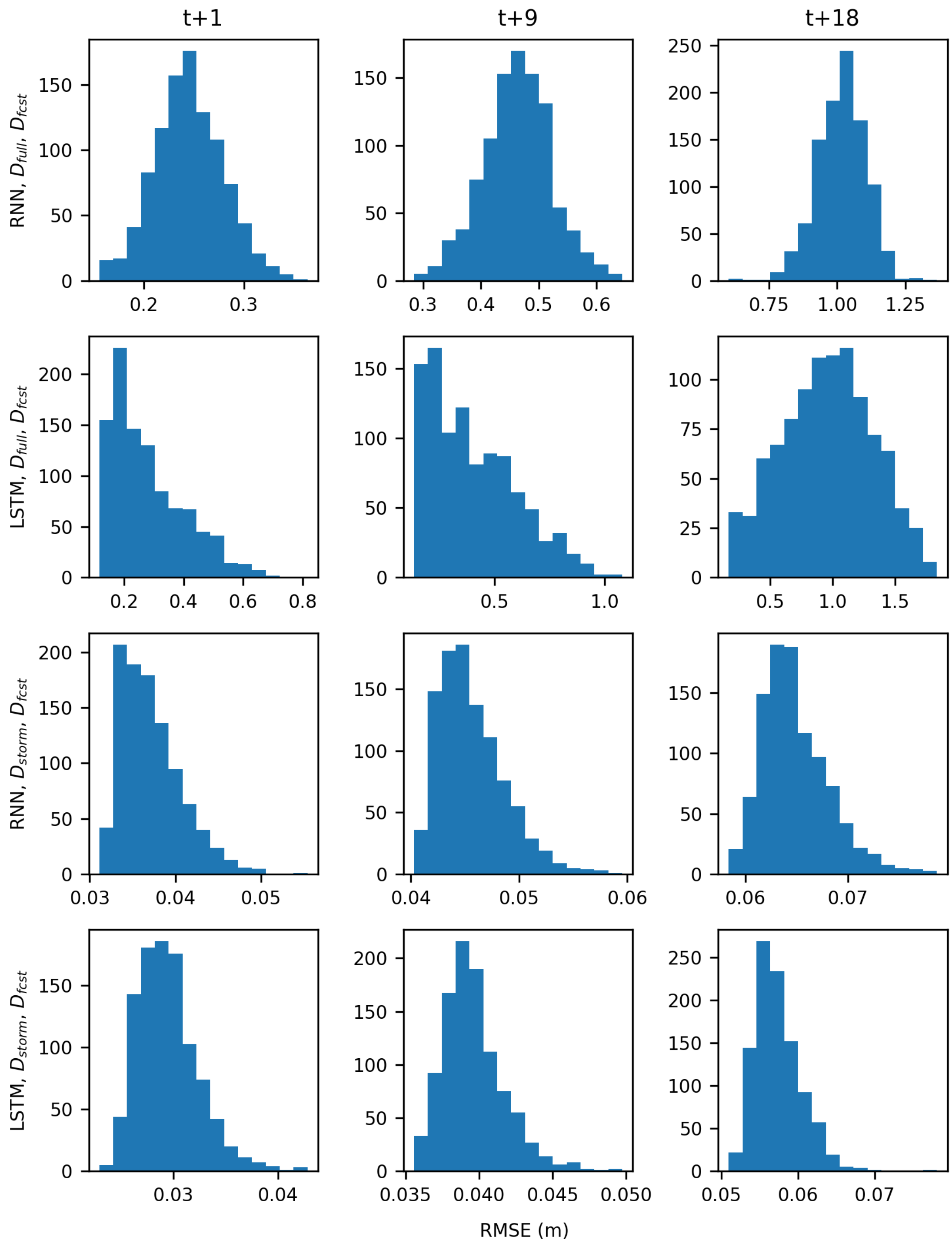

Figure 1: RMSE distributions for GW1 using forecast input data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

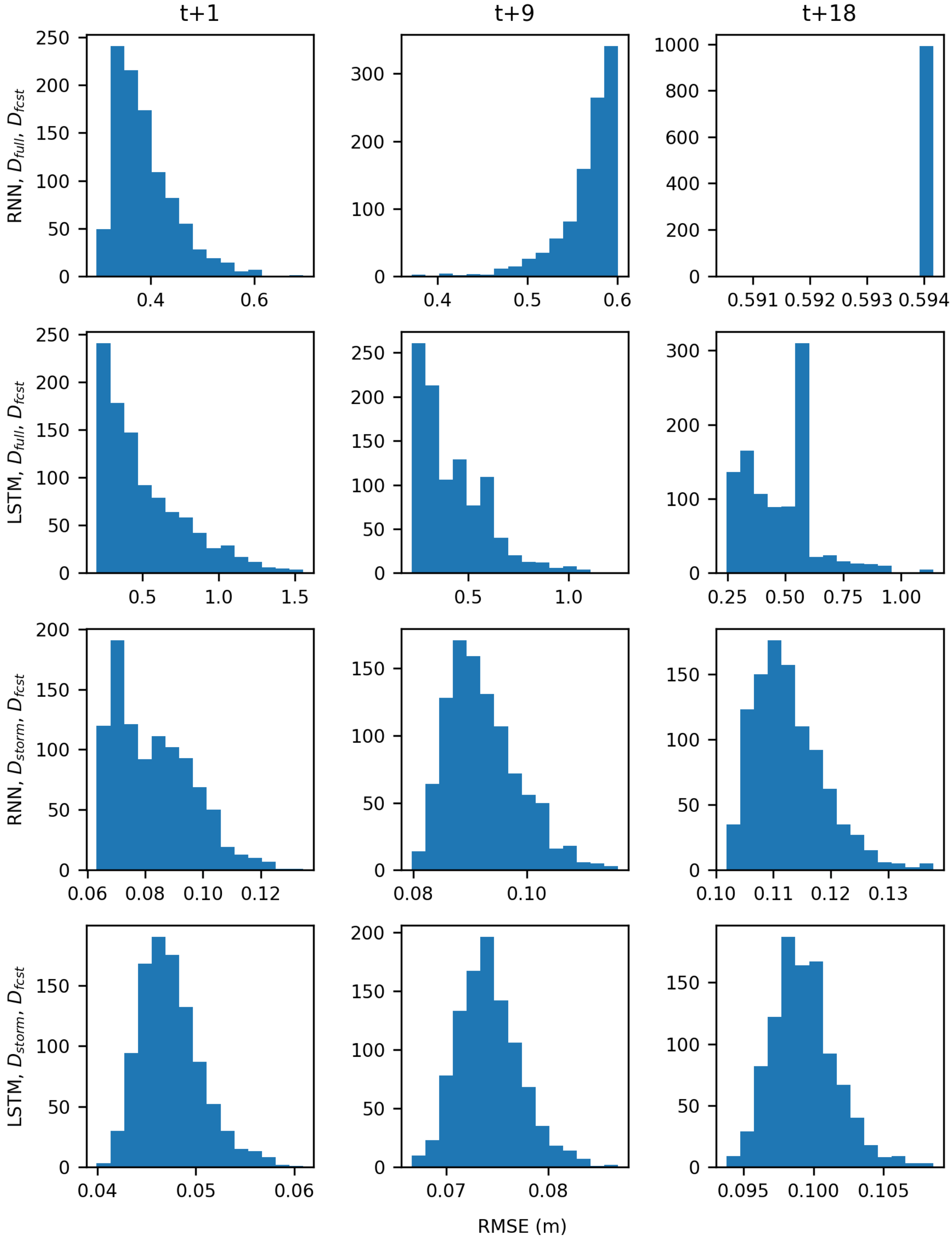

Figure 2: RMSE distributions for GW2 using forecast input data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

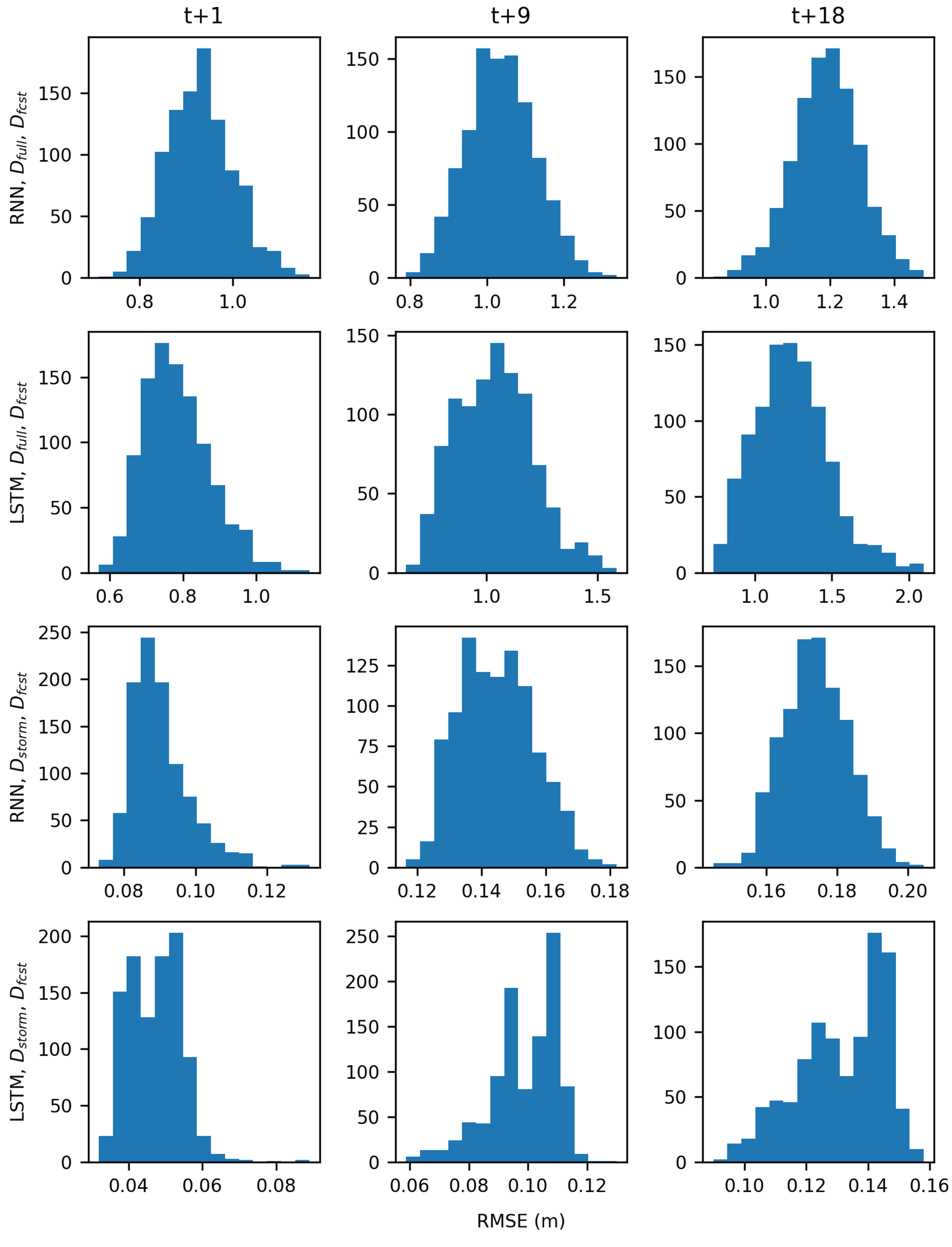

Figure 3: RMSE distributions for GW3 using forecast input data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

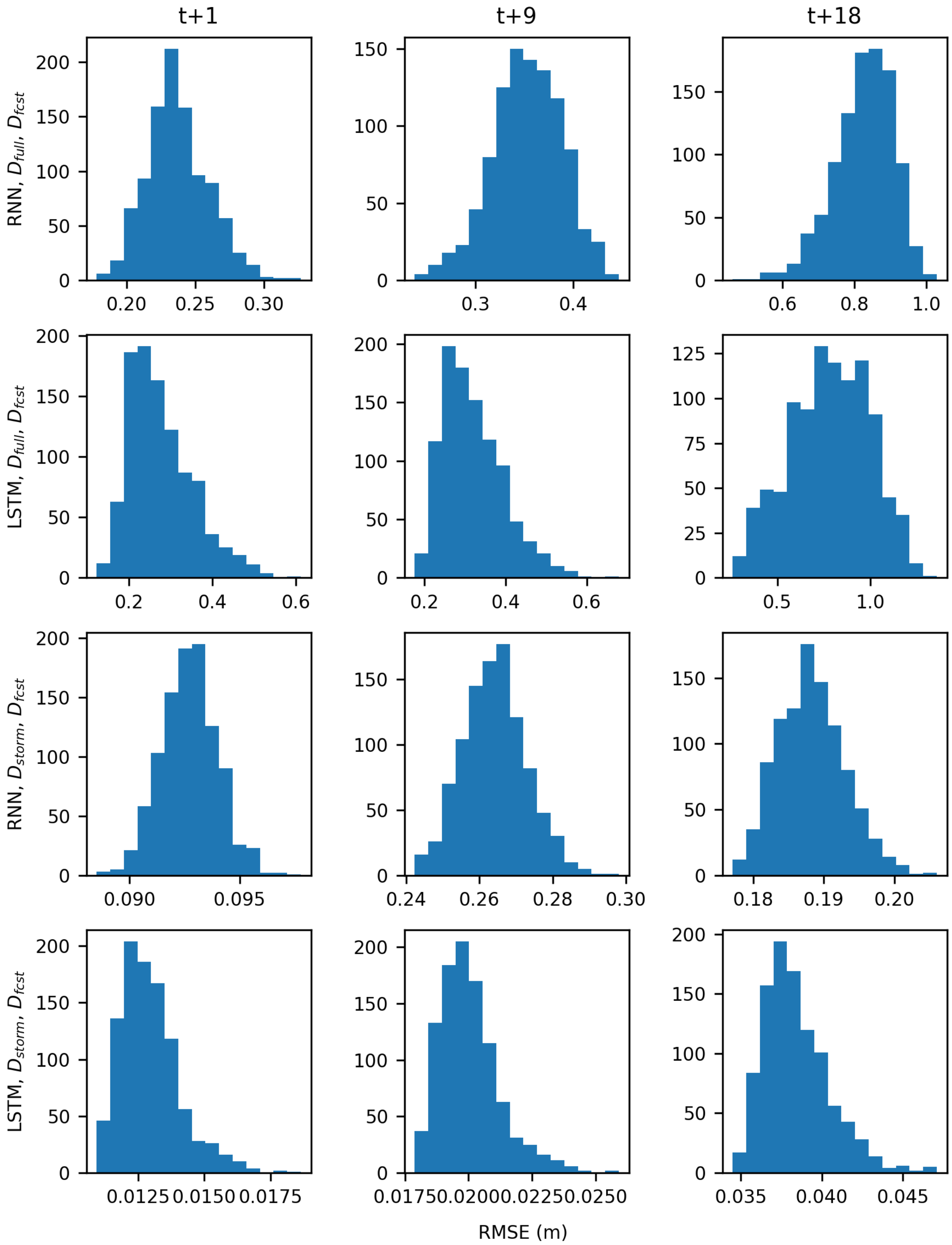

Figure 4: RMSE distributions for GW4 using forecast input data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

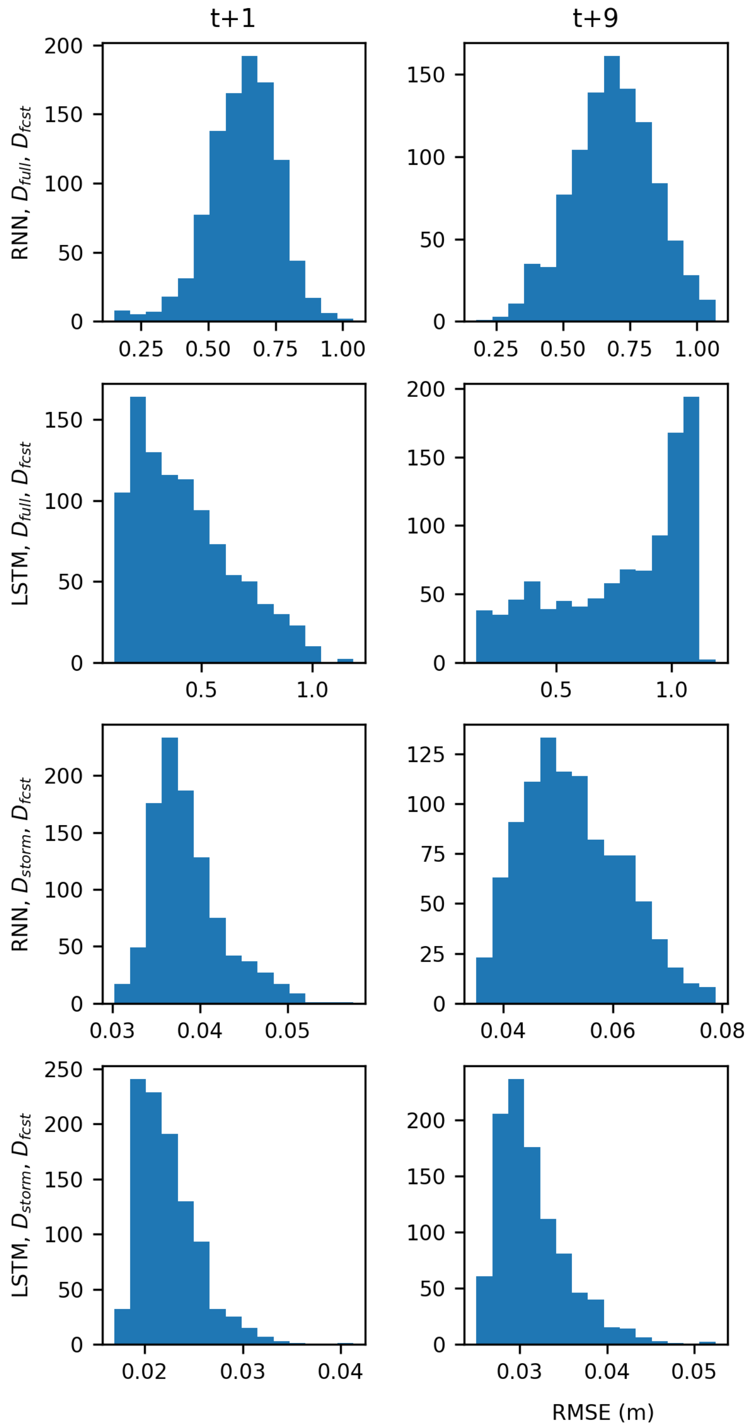

Figure 5: RMSE distributions for GW5 using forecast input data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

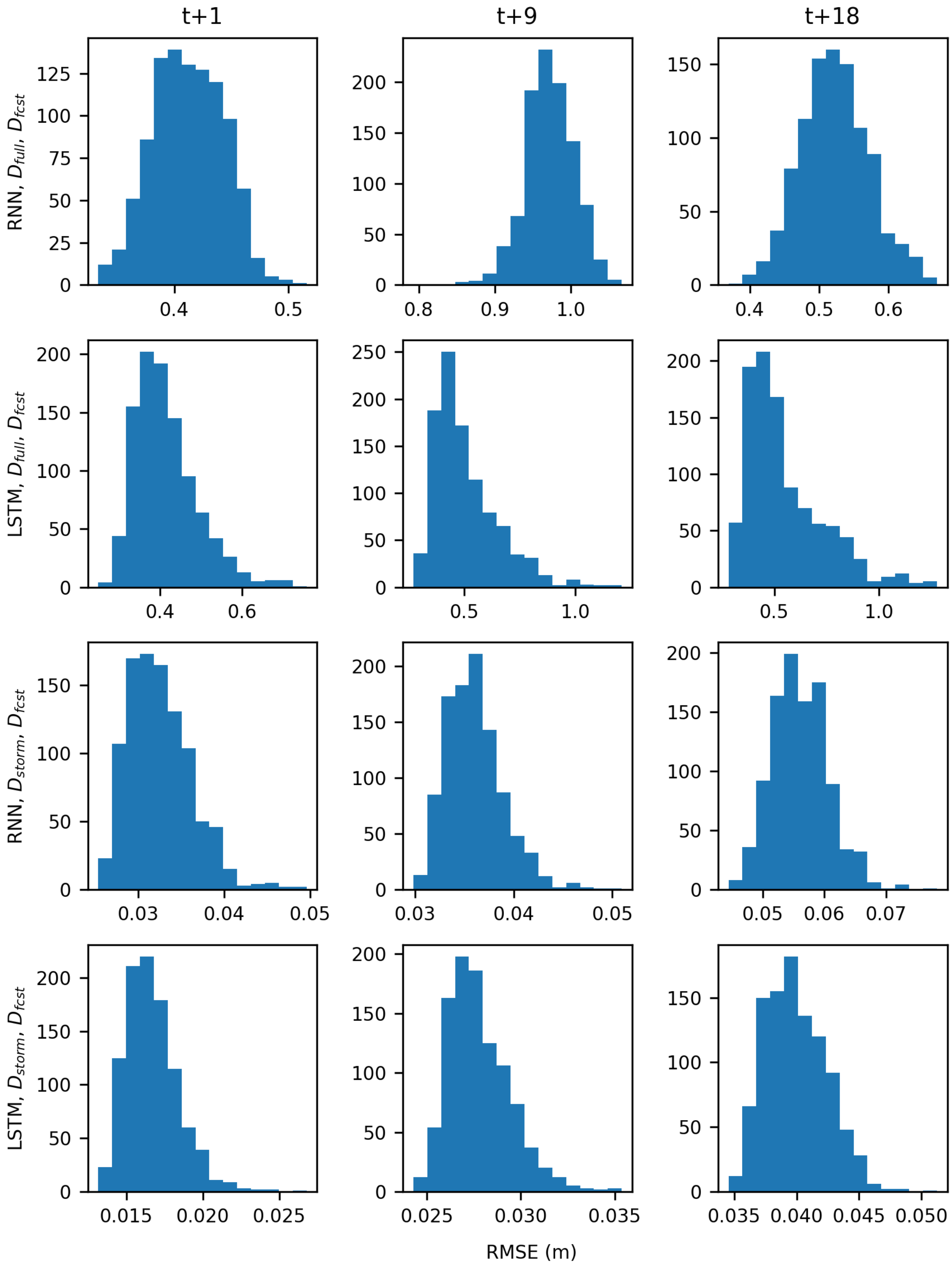

Figure 6: RMSE distributions for GW6 using forecast input data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

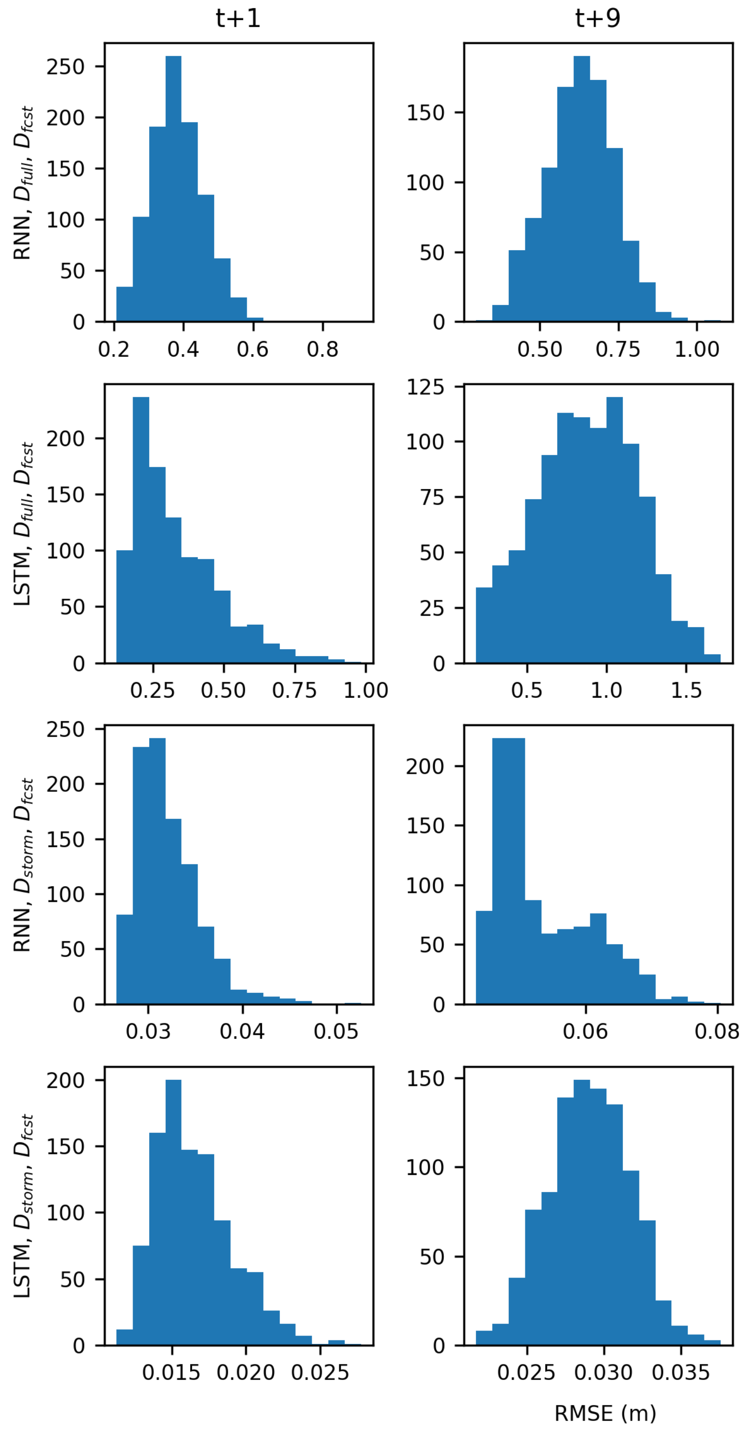

Figure 7: RMSE distributions for GW7 using forecast input data. Columns represent the forecast horizons t+1, t+9, and t+18. Rows are specified as model type, training data, and testing data.

Appendix C: Groundwater Table Forecast Mean Absolute Error (MAE) Data

Model Type	Training Data	Testing Data	Forecast Period	GW1	GW2	GW3	GW4	GW5	GW6	GW7
RNN	D_{full}	D_{full}	t+1	0.031	0.060	0.072	0.019	0.035	0.116	0.029
			t+9	0.049	0.089	0.099	0.036	0.054	0.410	0.044
			t+18	0.069	0.118	0.127	0.052	0.075	0.236	0.060
RNN	D_{full}	D_{storm}	t+1	0.038	0.072	0.080	0.022	0.047	0.121	0.034
			t+9	0.064	0.114	0.119	0.042	0.074	0.397	0.054
			t+18	0.092	0.151	0.157	0.060	0.102	0.228	0.076
LSTM	D_{full}	D_{full}	t+1	0.020	0.029	0.021	0.008	0.016	0.013	0.014
			t+9	0.040	0.067	0.053	0.027	0.039	0.032	0.028
			t+18	0.061	0.102	0.087	0.046	0.063	0.052	0.045
LSTM	D_{full}	D_{storm}	t+1	0.025	0.033	0.026	0.010	0.020	0.013	0.016
			t+9	0.056	0.083	0.070	0.032	0.053	0.034	0.036
			t+18	0.084	0.128	0.116	0.054	0.084	0.057	0.058
RNN	D_{storm}	D_{storm}	t+1	0.030	0.060	0.069	0.069	0.039	0.026	0.026
			t+9	0.041	0.068	0.080	0.288	0.045	0.028	0.033
			t+18	0.051	0.085	0.095	0.208	0.048	0.036	0.043
LSTM	D_{storm}	D_{storm}	t+1	0.024	0.031	0.023	0.008	0.017	0.012	0.013
			t+9	0.036	0.049	0.037	0.015	0.027	0.019	0.021
			t+18	0.045	0.066	0.052	0.023	0.033	0.027	0.029

Table 1: Mean MAE values for each model type and training dataset treatment at each well and forecast period when tested on observed data.

Model Type	Training Data	Testing Data	Forecast Period	GW1	GW2	GW3	GW4	GW5	GW6	GW7
RNN	D_{full}	D_{fcst}	t+1	0.211	0.308	0.881	0.206	0.613	0.369	0.356
			t+9	0.439	0.513	1.001	0.333	0.668	0.960	0.608
			t+18	0.998	0.537	1.131	0.800	0.913	0.493	1.113
LSTM	D_{full}	D_{fcst}	t+1	0.235	0.454	0.716	0.199	0.394	0.346	0.295
			t+9	0.374	0.362	0.976	0.285	0.759	0.440	0.853
			t+18	0.939	0.421	1.178	0.764	1.011	0.488	1.222
RNN	D_{storm}	D_{fcst}	t+1	0.027	0.064	0.064	0.068	0.027	0.026	0.023
			t+9	0.032	0.060	0.096	0.241	0.037	0.026	0.036
			t+18	0.038	0.073	0.106	0.160	0.034	0.037	0.034
LSTM	D_{storm}	D_{fcst}	t+1	0.022	0.029	0.027	0.007	0.014	0.012	0.012
			t+9	0.028	0.044	0.038	0.012	0.019	0.019	0.017
			t+18	0.037	0.059	0.055	0.022	0.025	0.027	0.025

Table 2: Mean MAE values for each model type and training dataset treatment at each well and forecast period when tested on forecast data D_{fcst}.

Appendix D: Additional RTC Figures

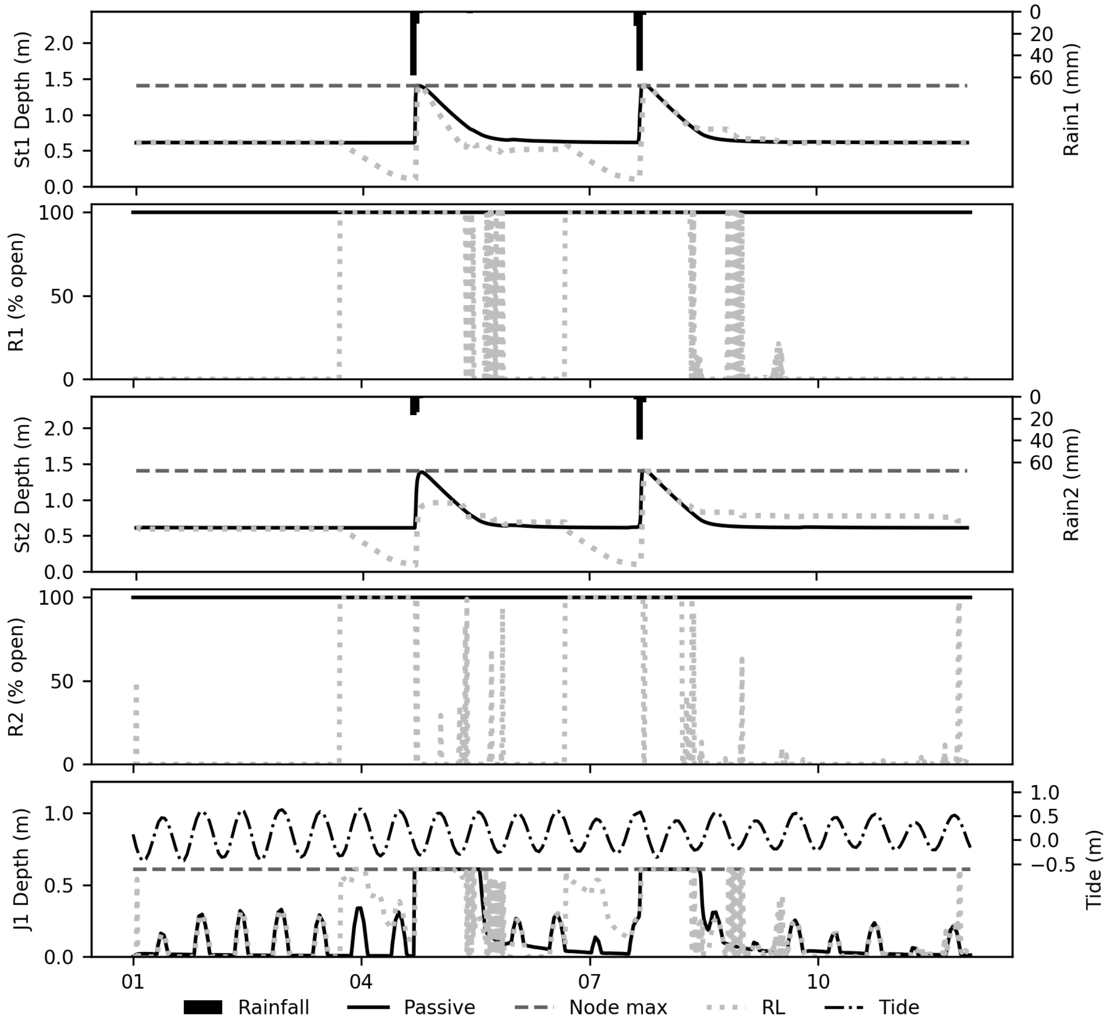

Figure 1: Detailed comparison of RL controlled and passive system performance for the first two storms of August, 2019.

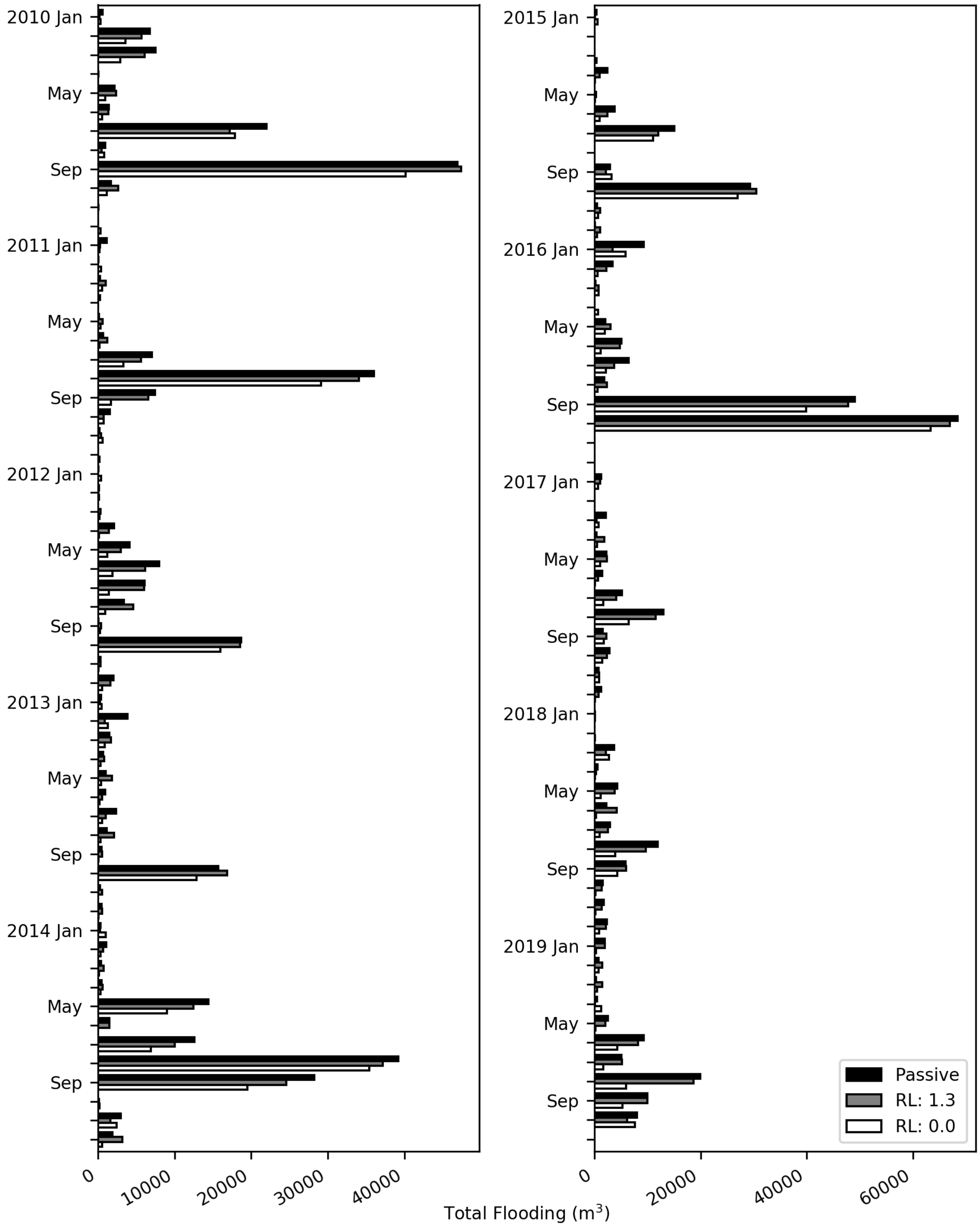

Figure 2: Total flood volumes in RL controlled and passive systems. Two values of rainfall threshold for the RL reward function (0 and 1.3mm) are shown to illustrate the impact of this parameter on the RL agent's performance.

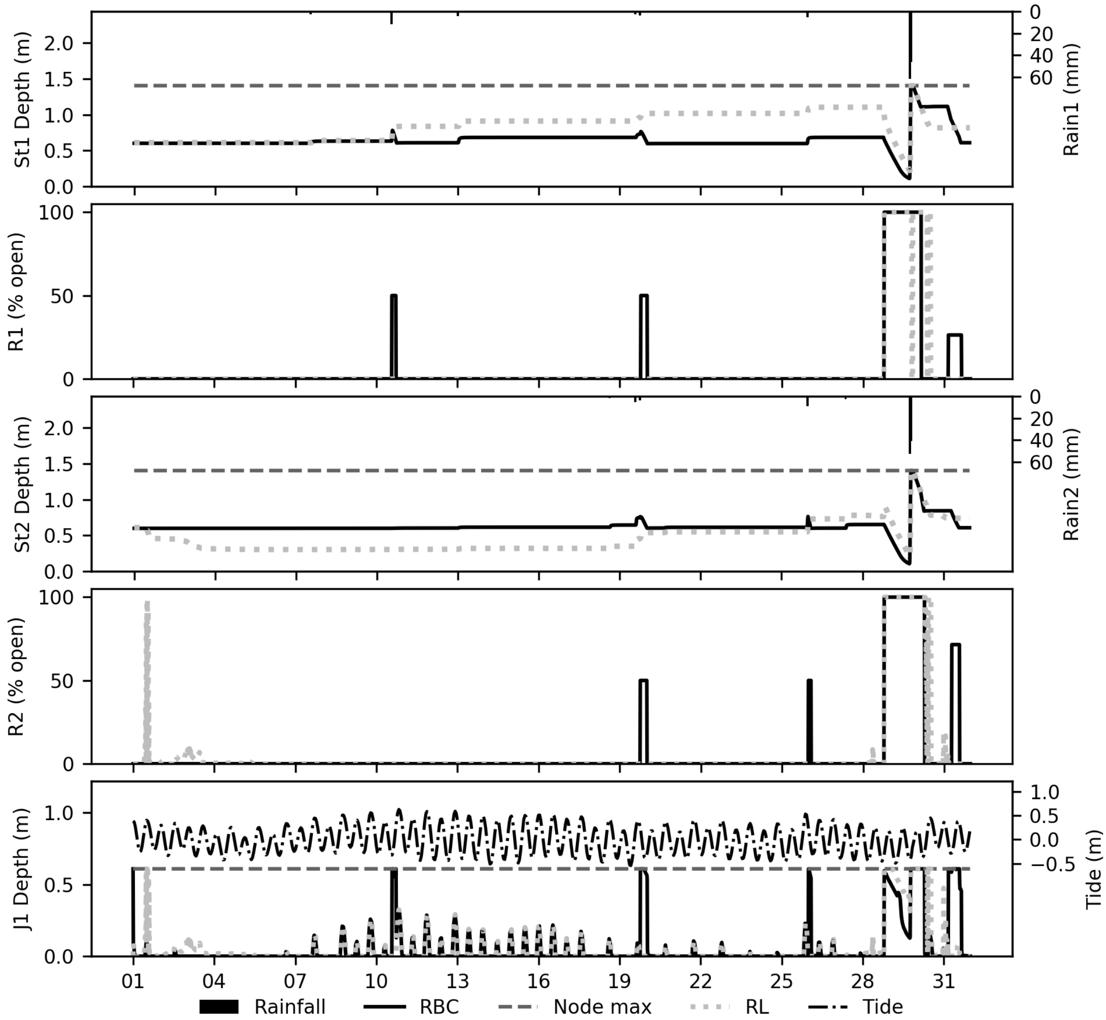

Figure 3: Comparison of RL and RBC system performance for the month of July, 2010.

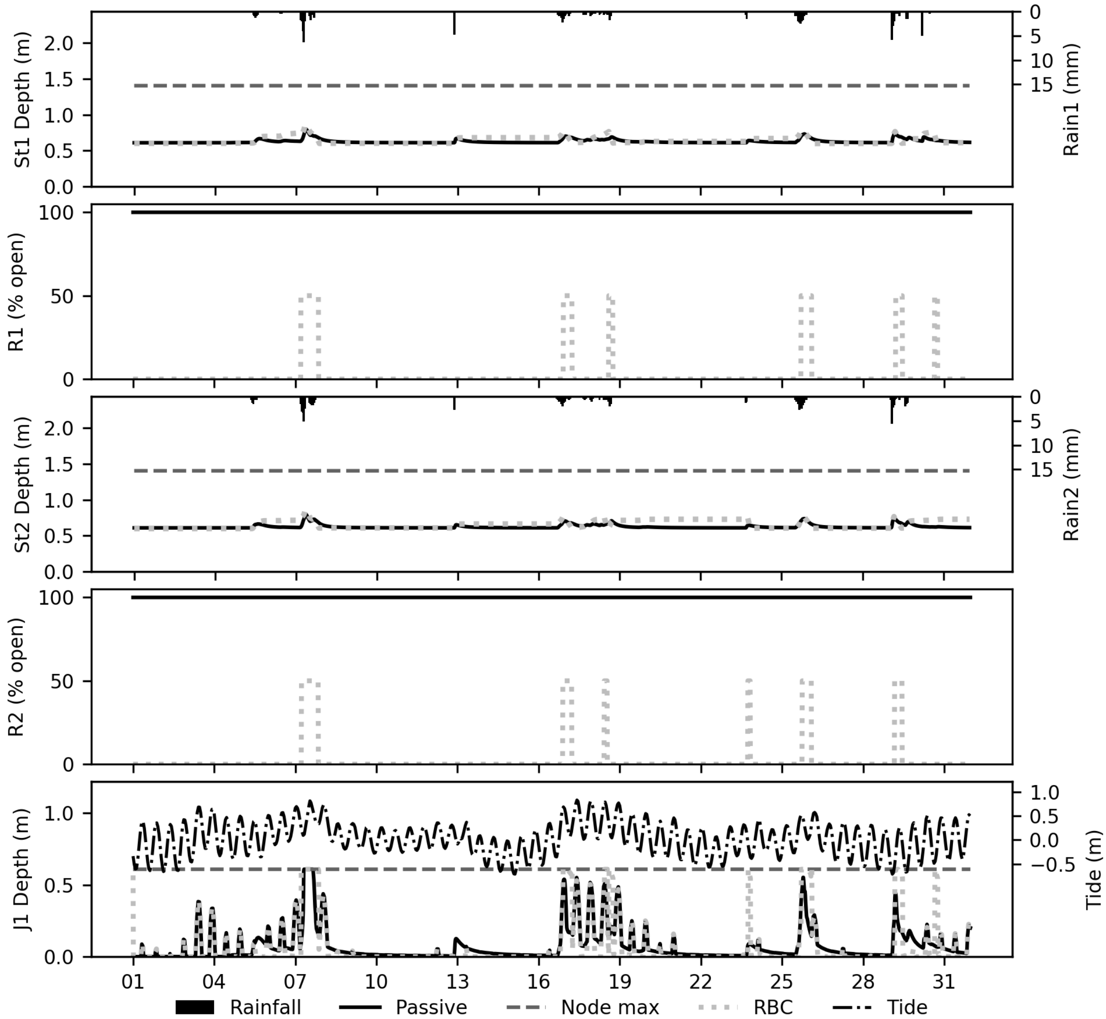

Figure 4: Comparison of RBC and passive system performance for the month of March, 2014.

Appendix E: Classification of Flood Events for Roadway Intersections in Norfolk, Virginia

Using a 1m LiDAR-derived digital elevation model (DEM) and flood event information (Sadler et al., 2018) provided by the City of Norfolk, Virginia, three intersections were identified as potential locations impacted by stormwater system flooding (Fig. 1). As these intersections are in depressions in the land surface, the Whitebox GIS sink tool was used to find the area of each depression. The difference between the minimum and maximum elevation in a sink was assumed to be its depth and the volume was approximated as the depth times the area. It should be noted that this is a rough calculation of the depth adequate for the purposes of this study, but is most likely an over estimate. The sink depths and volumes were plotted and a linear relationship was assumed (Fig. 2). Using the equation for this line, volumes for 0.2, 0.3, and 0.4m depths of roadway flooding were calculated.

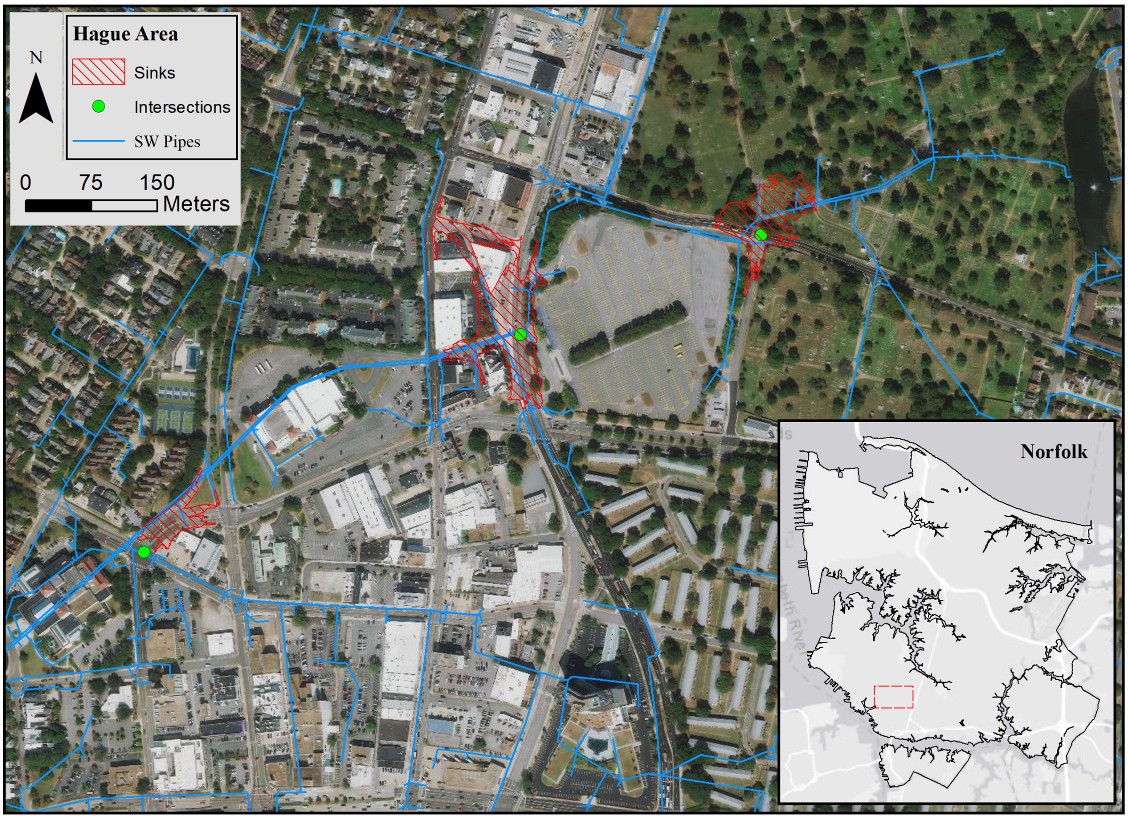

Figure 1: Locations of intersections in the Hague area of Norfolk, Virginia where sinks where identified and elevation data were extracted.

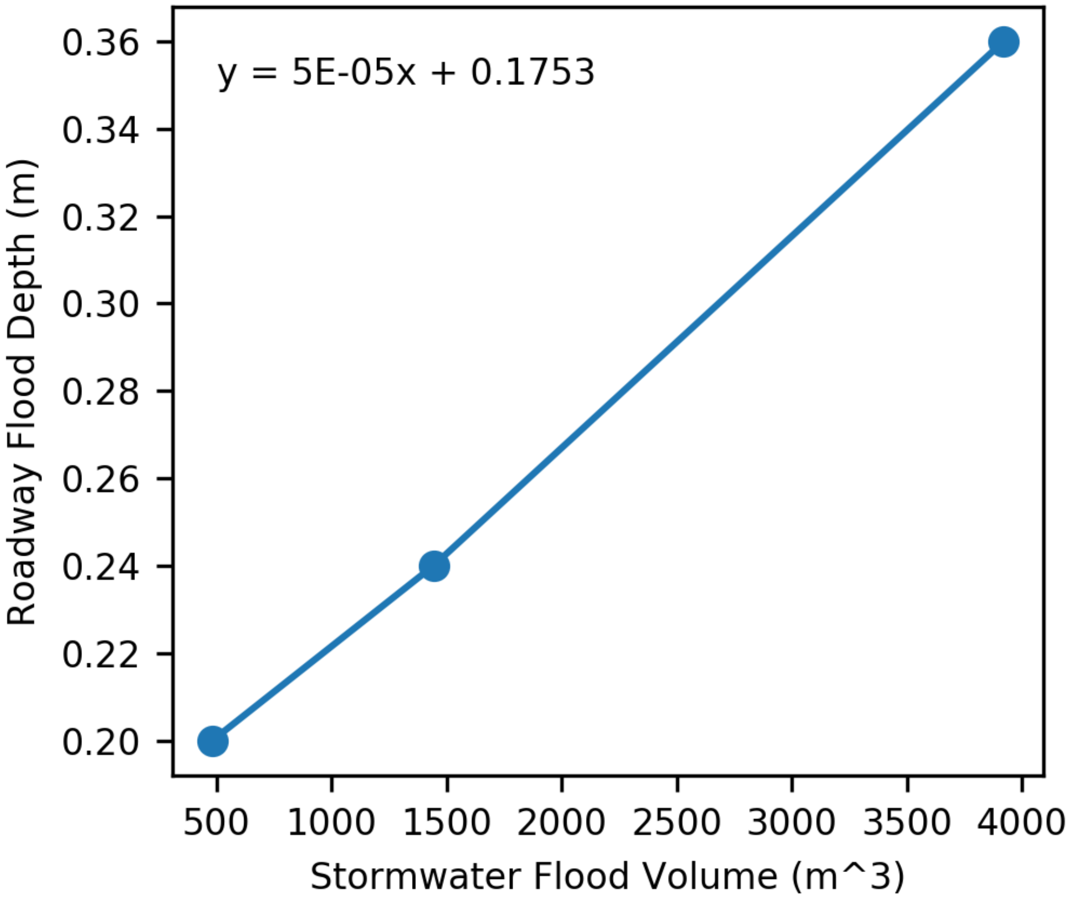

Figure 2: Relationship between roadway intersection flood depth and the corresponding stormwater flood volume.

Appendix F: Calculation of Groundwater Contribution to Pond Volume

A depth-volume relationship for the Elmwood Cemetery stormwater retention pond was estimated from design plans provided by the City of Norfolk. The pond's volume was estimated to be 11612.09m^3 and the surface area over which seepage can occur (pond sides and bottom) was 28340.30m^2. The normal pond water surface elevation given in these plans was assumed to also correspond to the groundwater table at the pond. Hydraulic conductivity of the soil surrounding the pond was accessed from the NRCS Web Soil Survey and is approximately evenly split between two soil complexes, resulting in a mean hydraulic conductivity of 0.60m/day. The seepage rate Q of groundwater into the pond was calculated as:

$$Q = KIA \tag{1}$$

where K is the hydraulic conductivity, I is the gradient between the pond surface and groundwater table elevation, and A is the surface area over which seepage can occur. This relationship is shown in Figure 1.

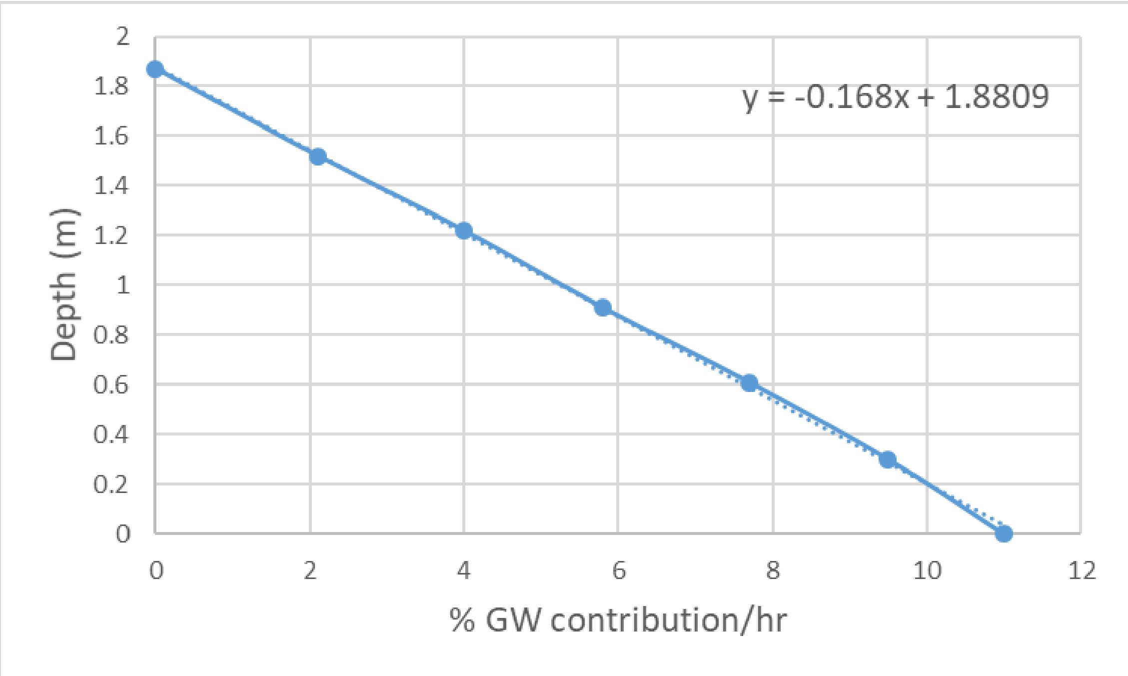

Figure 1: Relationship between Elmwood Pond depth and assumed groundwater (GW) contribution.

Appendix G: Groundwater Exchange Impact on RTC Ponds

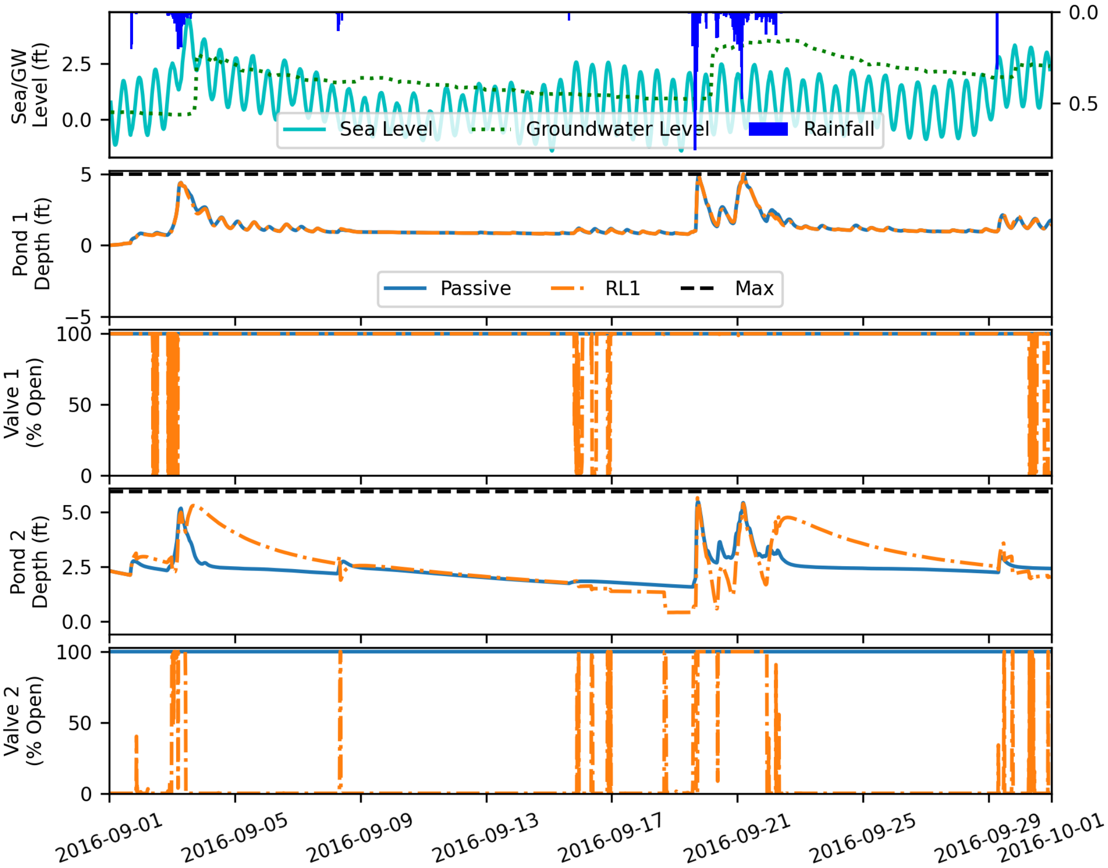

Figure 1: Comparison of RL Agent 1 and Passive system operation for September, 2016 with groundwater exchange at the controlled ponds.

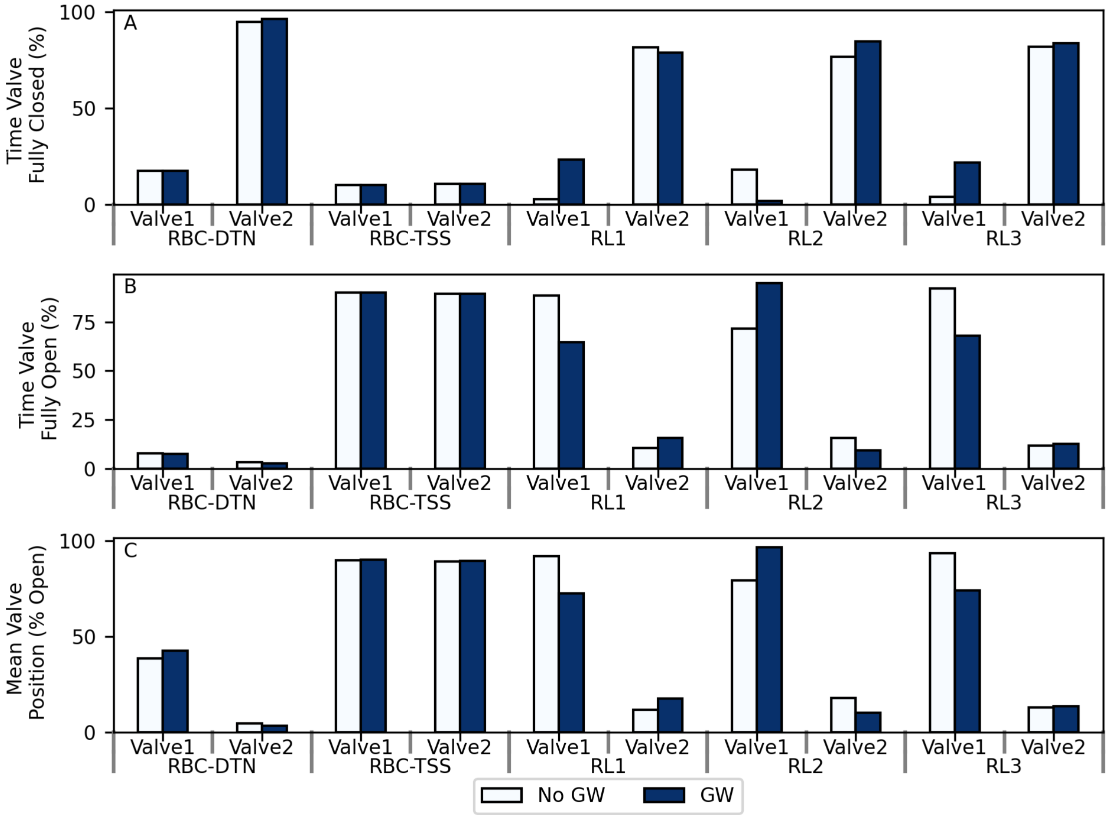

Figure 2: Comparison of control policies (% of time a valve is fully closed (A), fully open (B), and the mean valve position (C)) for simulations with and without groundwater (GW) exchange for September, 2016.

Bibliography

Abadi, Martín et al. (2016a). "TensorFlow: A System for Large-Scale Machine Learning". In: *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation*. Savannah, GA: USENIX, pp. 265–283. ISBN: 978-1-931971-33-1. URL: https://tensorflow.org..

Abadi, Martín et al. (2016b). "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems". In: *arXiv preprint arXiv:160304467*. URL: https://arxiv.org/pdf/1603.04467.pdf.

Abboud, J M, M C Ryan, and G D Osborn (2018). "Groundwater flooding in a river-connected alluvial aquifer". In: *Journal of Flood Risk Management* 11.4. DOI: 10.1111/jfr3.12334. URL: http://doi.wiley.com/10.1111/jfr3.12334.

Abou Rjeily, Yves et al. (2018). "Model Predictive Control for optimising the operation of Urban Drainage Systems". In: *Journal of Hydrology* 566, pp. 558–565. DOI: 10.1016/j.jhydrol.2018.09.044. URL: https://linkinghub.elsevier.com/retrieve/pii/S0022169418307388.

Alamdari, Nasrin et al. (Jan. 2020). "Evaluating the Impact of Climate Change on Water Quality and Quantity in an Urban Watershed Using an Ensemble Approach". In: *Estuaries and Coasts* 43.1, pp. 56–72. ISSN: 15592731. DOI: 10.1007/s12237-019-00649-4. URL: https://doi.org/10.1007/s12237-019-00649-4.

Amodei, Dario et al. (2016). "Concrete Problems in AI Safety". In: *CoRR*. URL: http://arxiv.org/abs/1606.06565.

Andel, Schalk Jan van et al. (2008). "Ensemble precipitation and water-level forecasts for anticipatory water-system control". In: *Journal of Hydrometeorology* 9.4, pp. 776–788. ISSN: 1525755X. DOI: 10.1175/2008JHM971.1. URL: http://journals.ametsoc.org/jhm/article-pdf/9/4/776/4164156/2008jhm971_1.pdf.

Andel, Schalk Jan van et al. (2014). "Framework for Anticipatory Water Management: Testing for Flood Control in the Rijnland Storage Basin". In: *Journal of Water Resources Planning and Management* 140.4, pp. 533–542. ISSN: 0733-9496. DOI: 10.1061/(ASCE)WR.1943-5452.0000254. URL: http://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000254.

ASCE (2021). *Stormwater - 2021 Infrastructure Report Card*. Tech. rep. ASCE. URL: https://infrastructurereportcard.org/wp-content/uploads/2020/12/Stormwater-2021.pdf.

Balchen, Jens G, Dag Ljungquist, and Stig Strand (1992). "State—space predictive control". In: *Chemical Engineering Science* 47.4, pp. 787–807.

Baldazo, David, Juan Parras, and Santiago Zazo (2019). "Decentralized Multi-Agent Deep Reinforcement Learning in Swarms of Drones for Flood Monitoring". In: *27th European Signal Processing Conference (EUSIPCO)*. ISBN: 9789082797039. URL: https://www.eurasip.org/Proceedings/Eusipco/eusipco2019/Proceedings/papers/1570533953.pdf.

Behl, Madhur, Truong X Nghiem, and Rahul Mangharam (2014). "Model-iq: Uncertainty propagation from sensing to modeling and control in buildings". In: *2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, pp. 13–24.

Bengio, Y., P. Simard, and P. Frasconi (Mar. 1994). "Learning long-term dependencies with gradient descent is difficult". In: *IEEE Transactions on Neural Networks* 5.2, pp. 157–166. ISSN: 10459227. DOI: 10.1109/72.279181. URL: http://ieeexplore.ieee.org/document/279181/.

Berggren, Karolina et al. (2012). "Hydraulic Impacts on Urban Drainage Systems due to Changes in Rainfall Caused by Climatic Change". In: *Journal of Hydrologic Engineering* 17.1, pp. 92–98. DOI: 10.1061/(ASCE)HE.1943-5584.0000406. URL: https://ascelibrary.org/doi/pdf/10.1061/%28ASCE%29HE.1943-5584.0000406.

Bergstra, J, D Yamins, and D D Cox (2013a). "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures". In: *Proceedings of the 30th International Conference on Machine Learning*. Vol. 28. Atlanta, Georgia: Journal of Machine Learning Research. URL: http://proceedings.mlr.press/v28/bergstra13.pdf.

— (2013b). "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures". In: *Proceedings of the 30th International Conference on Machine Learning*. Vol. 28. Atlanta, Georgia: Journal of Machine Learning Research. URL: http://proceedings.mlr.press/v28/bergstra13.pdf.

Bjerklie, David M et al. (2012). *Preliminary investigation of the effects of sea-level rise on groundwater levels in New Haven, Connecticut*. Tech. rep. United States Geological Survey.

Blaylock, Brian K, John D Horel, and Samuel T Liston (2017). "Cloud archiving and data mining of High-Resolution Rapid Refresh forecast model output". In: *Computers and Geosciences* 109, pp. 43–50. DOI: 10.1016/j.cageo.2017.08.005. URL: http://ac.els-cdn.com/S0098300417305083/1-s2.0-S0098300417305083-main.pdf?_tid=c013cc20-842c-11e7-a65e-00000aacb35e&acdnat=1503071519_d79fc68989226986236ae7c38c35dae0.

Bloetscher, Frederick et al. (2012). "Identification of Physical Transportation Infrastructure Vulnerable to Sea Level Rise". In: *Journal of Sustainable Development* 5.12, pp. 40–51. DOI: 10.5539/jsd.v5n12p40. URL: http://dx.doi.org/10.5539/jsd.v5n12p40.

Bowes, Benjamin D. (2020a). *Monthly SWMM Input Files for Real-time Control Simulation*. URL: http://www.hydroshare.org/resource/e2d21c9224ab4aefaf1a5b6394b270b1.

— (2020b). *UVAdMIST/swmm_rl: Repo for SWMM RL code*. URL: https://github.com/UVAdMIST/swmm_rl.

Bowes, Benjamin D. et al. (Oct. 2020). "Flood mitigation in coastal urban catchments using real-time stormwater infrastructure control and reinforcement learning". In: *Journal of Hydroinformatics*. ISSN: 1464-7141. DOI: 10.2166/hydro.2020.080. URL: https://iwaponline.com/jh/article/doi/10.2166/hydro.2020.080/77759/Flood-mitigation-in-coastal-urban-catchments-using.

Bowes, Benjamin D et al. (2019). "Forecasting Groundwater Table in a Flood Prone Coastal City with Long Short-term Memory and Recurrent Neural Networks". In: *Water* 11.5, p. 1098. ISSN: 2073-4441. DOI: 10.3390/w11051098. URL: https://www.mdpi.com/2073-4441/11/5/1098.

Boyer, K Brian and Mark S Kieser (2012). "URBAN STORMWATER MANGEMENT-AN MS4 SUCCESS STORY FOR WESTERN MICHIGAN UNIVERSITY". In: *Journal of Green Building* 7.1. URL: http://meridian.allenpress.com/jgb/article-pdf/7/1/28/1770769/jgb_7_1_28.pdf.

Brockman, Greg et al. (2016). *OpenAI Gym*. URL: http://arxiv.org/abs/1606.01540.

Brudler, Sarah et al. (2019). "Pollution levels of stormwater discharges and resulting environmental impacts". In: *Science of the Total Environment* 663, pp. 754–763. ISSN: 18791026. DOI: 10.1016/j.scitotenv.2019.01.388. URL: https://doi.org/10.1016/j.scitotenv.2019.01.388.

Camacho, E. F. and C. (Carlos) Bordons (2007). *Model predictive control.* London; New York: Springer. ISBN: 1852336943.
Castelletti, A, F Pianosi, and M Restelli (2013). "A multiobjective reinforcement learning approach to water resources systems operation: Pareto frontier approximation in a single run". In: *Water Resources Research* 49, pp. 3476–3486. DOI: 10.1002/wrcr.20295. URL: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1002/wrcr.20295.
Castelletti, Andrea et al. (2014). "Planning the Optimal Operation of a Multioutlet Water Reservoir with Water Quality and Quantity Targets". In: *Journal of Water Resources Planning and Management* 140.4, pp. 496–510. DOI: 10.1061/(ASCE)WR.1943-5452. URL: https://ascelibrary.org/doi/pdf/10.1061/%28ASCE%29WR.1943-5452.0000348.
Chang, Fi-John et al. (2016a). "Prediction of monthly regional groundwater levels through hybrid soft-computing techniques". In: *Journal of Hydrology* 541, pp. 965–976. DOI: 10.1016/j.jhydrol.2016.08.006. URL: http://ac.els-cdn.com/S0022169416304851/1-s2.0-S0022169416304851-main.pdf?_tid=c841e0cc-7873-11e7-9563-00000aacb362&acdnat=1501782615_a91b4f52d5bb10cdb9cbcd7936561a79.
Chang, Sun Woo et al. (2016b). "Impacts of Climate Change and Urbanization on Groundwater Resources in a Barrier Island". In: *Journal of Environmental Engineering* 142.Ipcc, pp. 1–12. ISSN: 0733-9372. DOI: 10.1061/(ASCE)EE.1943-7870.0001123..
Chen, Yiheng and Dawei Han (May 2018). "Water quality monitoring in smart city: A pilot project". In: *Automation in Construction* 89, pp. 307–316. ISSN: 09265805. DOI: 10.1016/j.autcon.2018.02.008.
Chesapeake Bay Foundation (2018). *State of the Bay.* Tech. rep. Chesapeake Bay Foundation. URL: https://www.cbf.org/document-library/cbf-reports/2018-state-of-the-bay-report.pdf.
Chollet, F. (2015). "Keras". In: *https://keras.io.*
Cigler, Jivri et al. (2013). "Beyond theory: the challenge of implementing model predictive control in buildings". In: *Proceedings of 11th Rehva world congress, Clima.* Vol. 250.
Corbin, Charles D, Gregor P Henze, and Peter May-Ostendorp (2013). "A model predictive control optimization environment for real-time commercial building application". In: *Journal of Building Performance Simulation* 6.3, pp. 159–174.
Coulibaly, Paulin et al. (Apr. 2001). "Artificial neural network modeling of water table depth fluctuations". In: *Water Resources Research* 37.4, pp. 885–896. ISSN: 00431397. DOI: 10.1029/2000WR900368. URL: http://doi.wiley.com/10.1029/2000WR900368.
Daliakopoulos, Ioannis N, Paulin Coulibaly, and Ioannis K Tsanis (2005). "Groundwater level forecasting using artificial neural networks". In: *Journal of Hydrology* 309.1-4, pp. 229–240. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2004.12.001. URL: http://ac.els-cdn.com/S0022169404005840/1-s2.0-S0022169404005840-main.pdf?_tid=1d90981a-4fa8-11e7-80d6-00000aacb35d&acdnat=1497297093_582727d5a7d2579956e468b79fade854.
Davtalab, Rahman et al. (Apr. 2020). "Sea Level Rise Effect on Groundwater Rise and Stormwater Retention Pond Reliability". In: *Water* 12.4, p. 1129. ISSN: 2073-4441. DOI: 10.3390/w12041129. URL: https://www.mdpi.com/2073-4441/12/4/1129.
Delipetrev, Blagoj, Andreja Jonoski, and Dimitri P Solomatine (2017). "A novel nested stochastic dynamic programming (nSDP) and nested reinforcement learning (nRL) algorithm for multipurpose reservoir optimization". In: *Journal of Hydroinformatics* 19.1, pp. 47–61. DOI: 10.2166/hydro.2016.243. URL: https://iwaponline.com/jh/article-pdf/19/1/47/390803/jh0190047.pdf.
Doble, Rebecca C. et al. (Dec. 2017). "Emulation of recharge and evapotranspiration processes in shallow groundwater systems". In: *Journal of Hydrology* 555, pp. 894–908. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2017.10.065. URL: https://linkinghub.elsevier.com/retrieve/pii/S0022169417307424.

Eggleston, Jack and Jason Pope (2013). *Land Subsidence and Relative Sea-Level Rise in the Southern Chesapeake Bay Region*. Tech. rep. Reston, Virginia: U.S. Geological Survey. URL: https://pubs.usgs.gov/circ/1392/pdf/circ1392.pdf.

Elman, Jeffrey L. (Mar. 1990). "Finding Structure in Time". In: *Cognitive Science* 14.2, pp. 179–211. DOI: 10.1207/s15516709cog1402{_}1. URL: http://doi.wiley.com/10.1207/s15516709cog1402_1.

Fahimi, Farzad, Zaher Mundher Yaseen, and Ahmed El-Shafie (2017). "Application of soft computing based hybrid models in hydrological variables modeling: a comprehensive review". In: *THEORETICAL AND APPLIED CLIMATOLOGY* 128.3-4, pp. 875–903. DOI: 10.1007/s00704-016-1735-8. URL: https://link.springer.com/content/pdf/10.1007%2Fs00704-016-1735-8.pdf.

Fears, Darryl (June 2012). *Built on sinking ground, Norfolk tries to hold back tide amid sea-level rise*. URL: https://www.washingtonpost.com/national/health-science/built-on-sinking-ground-norfolk-tries-to-hold-back-tide-amid-sea-level-rise/2012/06/17/gJQADUsxjV_story.html?noredirect=on&utm_term=.fc9be59c217a.

Fischer, Thomas and Christopher Krauss (Oct. 2018). "Deep learning with long short-term memory networks for financial market predictions". In: *European Journal of Operational Research* 270.2, pp. 654–669. ISSN: 03772217. DOI: 10.1016/j.ejor.2017.11.054. URL: https://linkinghub.elsevier.com/retrieve/pii/S0377221717310652.

Flood, Jefferson F and Lawrence B Cahoon (2011). "Risks to Coastal Wastewater Collection Systems from Sea-Level Rise and Climate Change". In: *Journal of Coastal Research* 274.4, pp. 652–660. ISSN: 0749-0208. DOI: 10.2112/JCOASTRES-D-10-00129.1. URL: http://www.bioone.org/doi/abs/10.2112/JCOASTRES-D-10-00129.1.

Freeze, R A and J A Cherry (1979). *Groundwater*. Englewood Cliffs, New Jersey: Prentice Hall, Inc., p. 604. ISBN: 0133653129.

Gaborit, E. et al. (Aug. 2013). "Improving the performance of stormwater detention basins by real-time control using rainfall forecasts". In: *Urban Water Journal* 10.4, pp. 230–246. ISSN: 1573062X. DOI: 10.1080/1573062X.2012.726229.

García, L et al. (2015). "Modeling and real-time control of urban drainage systems: A review". In: *Advances in Water Resources* 85, pp. 120–132. ISSN: 03091708. DOI: 10.1016/j.advwatres.2015.08.007. URL: https://linkinghub.elsevier.com/retrieve/pii/S0309170815001931.

Giambastiani, Beatrice M S et al. (2017). "¡b¿Coastal aquifer response to extreme storm events in Emilia-Romagna, Italy¡/b¿". In: *Hydrological Processes* July 2016, pp. 1613–1621. ISSN: 08856087. DOI: 10.1002/hyp.11130. URL: http://doi.wiley.com/10.1002/hyp.11130.

Govindaraju, Rao S. (2000a). "ARTIFICIAL NEURAL NETWORKS IN HYDROLOGY. I: PRELIMINARY CONCEPTS By the ASCE Task Committee on Application of Artificial Neural Networks in Hydrology". In: *JOURNAL OF HYDROLOGIC ENGINEERING* 5.2.

— (Apr. 2000b). "Artificial Neural Networks in Hydrology. II: Hydrologic Applications." In: *Journal of Hydrologic Engineering* 5.2, p. 124. ISSN: 10840699. URL: http://proxy01.its.virginia.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=eih&AN=6786456&site=ehost-live&scope=site.

Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton (May 2013). "Speech recognition with deep recurrent neural networks". In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. Vancouver: IEEE, pp. 6645–6649. ISBN: 978-1-4799-0356-6. DOI: 10.1109/ICASSP.2013.6638947. URL: http://ieeexplore.ieee.org/document/6638947/.

Greff, Klaus et al. (Oct. 2017). "LSTM: A Search Space Odyssey". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.10, pp. 2222–2232. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2016.2582924. URL: http://ieeexplore.ieee.org/document/7508408/.

Guan, Mingfu et al. (Jan. 2018). "Numerical modelling of hydro-morphological processes dominated by fine suspended sediment in a stormwater pond". In: *Journal of Hydrology* 556, pp. 87–99. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2017.11.006.

Guzman, Sandra M., Joel O. Paz, and Mary Love M. Tagert (2017). "The Use of NARX Neural Networks to Forecast Daily Groundwater Levels". In: *Water Resources Management* 31, pp. 1591–1603. DOI: 10.1007/s11269-017-1598-5. URL: http://download.springer.com/static/pdf/169/art%253A10.1007%252Fs11269-017-1598-5.pdf?originUrl=http%3A%2F%2Flink.springer.com%2Farticle%2F10.1007%2Fs11269-017-1598-5&token2=exp=1496936548~acl=%2Fstatic%2Fpdf%2F169%2Fart%25253A10.1007%25252Fs11269-017-159.

Hartono, Pitoyo and Shuji Hashimoto (Jan. 2007). "Learning from imperfect data". In: *Applied Soft Computing Journal* 7.1, pp. 353–363. ISSN: 15684946. DOI: 10.1016/j.asoc.2005.07.005.

Heywood, Charles E and Jason P Pope (2009). "Simulation of groundwater flow in the Coastal Plain aquifer system of Virginia". In: *Scientific Investigations Report*, p. 115. ISSN: 10449612. URL: https://www.lib.uwo.ca/cgi-bin/ezpauthn.cgi?url=http://search.proquest.com/docview/753848134?accountid=15115%5Cnhttp://sfx.scholarsportal.info/western?url_ver=Z39.88-2004&rft_val_fmt=info:ofi/fmt:kev:mtx:journal&genre=article&sid=ProQ:ProQ:georefmodule&at.

Hochreiter, Sepp and Urgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735. URL: https://www.mitpressjournals.org/doi/pdf/10.1162/neco.1997.9.8.1735.

Hoover, Daniel J, Kingsley O Odigie, and Patrick Barnard (2017). "Sea-level rise and coastal groundwater inundation and shoaling at select sites in California, USA". In: *Journal of Hydrology: Regional Studies* 11, pp. 234–249. DOI: 10.1016/J.EJRH.2015.12.055. URL: https://www.sciencedirect.com/science/article/pii/S2214581815002050?via%3Dihub.

Hu, Caihong et al. (2018). "Deep Learning with a Long Short-Term Memory Networks Approach for Rainfall-Runoff Simulation". In: *Water* 10.11, p. 1543. DOI: 10.3390/w10111543. URL: http://www.mdpi.com/2073-4441/10/11/1543.

Jia, Xiaowei et al. (2019). "Physics guided RNNs for modeling dynamical systems: A case study in simulating lake temperature profiles". In: *SIAM International Conference on Data Mining, SDM 2019*. Society for Industrial and Applied Mathematics Publications, pp. 558–566. ISBN: 9781611975673. DOI: 10.1137/1.9781611975673.63. URL: https://epubs.siam.org/page/terms.

Jose Meneses, Elbys et al. (2018). "Coordinating Rule-Based and System-Wide Model Predictive Control Strategies to Reduce Storage Expansion of Combined Urban Drainage Systems: The Case Study of Lundtofte, Denmark". In: *Water* 10.76. DOI: 10.3390/w10010076. URL: www.mikebydhi.com.

Karandish, Fatemeh and Jiří Šimůnek (2016). "A comparison of numerical and machine-learning modeling of soil water content with limited input data". In: *Journal of Hydrology* 543, pp. 892–909. DOI: 10.1016/J.JHYDROL.2016.11.007. URL: https://www.sciencedirect.com/science/article/pii/S0022169416307132?_rdoc=1&_fmt=high&_origin=gateway&_docanchor=&md5=b8429449ccfc9c30159a5f9aeaa92ffb.

Karpf, Christian and Peter Krebs (2013). "Modelling of groundwater infiltration into sewer systems". In: *Urban Water Journal* 10.4, pp. 221–229. ISSN: 1573-062X. DOI: 10.1080/1573062X.2012.724077. URL: http://www.tandfonline.com/doi/abs/10.1080/1573062X.2012.724077.

Kerkez, Branko et al. (2016). "Smarter Stormwater Systems". In: *Environmental Science and Technology* 50, 72677273. DOI: 10.1021/acs.est.5b05870. URL: https://pubs.acs.org/doi/10.1021/acs.est.5b05870..

Kingma, Diederik P and Jimmy Ba (Dec. 2014). "Adam: A Method for Stochastic Optimization". In: URL: https://arxiv.org/pdf/1412.6980.pdf%20http://arxiv.org/abs/1412.6980.

Krajewski, W.F. and J.A. Smith (Aug. 2002). "Radar hydrology: rainfall estimation". In: *Advances in Water Resources* 25.8-12, pp. 1387–1394. DOI: 10.1016/S0309-1708(02)00062-3. URL: https://www.sciencedirect.com/science/article/pii/S0309170802000623?via%3Dihub.

Kreibich, Heidi and Annegret H Thieken (2008). "Assessment of damage caused by high groundwater inundation". In: *Water Resources Research* 44, p. 9409. DOI: 10.1029/2007WR006621. URL: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2007WR006621.

Kroll, Stefan et al. (Nov. 2018). "A Methodology for the Design of RTC Strategies for Combined Sewer Networks". In: *Water* 10.11, p. 1675. ISSN: 2073-4441. DOI: 10.3390/w10111675. URL: http://www.mdpi.com/2073-4441/10/11/1675.

Lee, Jin-Hee and John W Labadie (2007). "Stochastic optimization of multireservoir systems via reinforcement learning". In: *Water Resources Research* 43.11. DOI: 10.1029/2006WR005627. URL: http://doi.wiley.com/10.1029/2006WR005627.

Li, Pengfei, Zheng O'Neill, and James Braun (2013). "Development of control-oriented models for model predictive control in buildings". In: *ASHRAE Trans.* Vol. 119. 2. Boulder, Colorado: Intelligent Buildings Operations Workshop. URL: https://www.ibpsa.us/sites/default/files/publications/Modeling%20for%20Design%20and%20Operations%20II_PengfeiLi_ZhengONeill.pdf.

Liang, Chen et al. (2018). "Dongting Lake Water Level Forecast and Its Relationship with the Three Gorges Dam Based on a Long Short-Term Memory Network". In: *Water* 10.10, p. 1389. DOI: 10.3390/w10101389. URL: http://www.mdpi.com/2073-4441/10/10/1389.

Likmeta, Amarildo et al. (2020). "Combining reinforcement learning with rule-based controllers for transparent and general decision-making in autonomous driving". In: *Robotics and Autonomous Systems* 131, p. 103568. ISSN: 09218890. DOI: 10.1016/j.robot.2020.103568.

Lillicrap, Timothy P et al. (Sept. 2015). "Continuous control with deep reinforcement learning". In: *International Conference on Learning Representations*, p. 14. URL: https://goo.gl/J4PIAz%20http://arxiv.org/abs/1509.02971.

Litwin, David et al. (Feb. 2020). "GroundwaterDupuitPercolator: A Landlab component for groundwater flow". In: *Journal of Open Source Software* 5.46, p. 1935. ISSN: 2475-9066. DOI: 10.21105/joss.01935. URL: https://joss.theoj.org/papers/10.21105/joss.01935.

Liu, Qiang et al. (Dec. 2017). "Regulation of drainage canals on the groundwater level in a typical coastal wetlands". In: *Journal of Hydrology* 555, pp. 463–478. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2017.10.035. URL: https://linkinghub.elsevier.com/retrieve/pii/S0022169417307102.

Maier, Holger R and Graeme C Dandy (2000). "Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications". In: *Environmental Modelling & Software* 15, pp. 101–124. DOI: 10.1016/S1364-8152(99)00007-9. URL: www.elsevier.com/locate/envsoft.

Maier, Holger R et al. (2010). "Methods used for the development of neural networks for the prediction of water resource variables in river systems: Current status and future directions". In: *Environmental Modelling and Software* 25, pp. 891–909. DOI:

10.1016/j.envsoft.2010.02.003. URL: https://ac.els-cdn.com/S1364815210000411/1-s2.0-S1364815210000411-main.pdf?_tid=c2eee266-ca06-11e7-9b87-00000aacb35d&acdnat=1510751785_781bbce8f7141902f2d187f99fd9081e.

Marchese, Dayton et al. (2018). "Quantitative Comparison of Active and Passive Stormwater Infrastructure: Case Study in Beckley, West Virginia". In: *Proceedings of the Water Environment Federation* 2018.9, pp. 4298–4311. ISSN: 1938-6478. DOI: 10.2175/193864718825139005. URL: https://www.researchgate.net/publication/329593005%20https://accesswater.org/publications/-300096/quantitative-comparison-of-active-and-passive-stormwater-infrastructure--case-study-in-beckley--west-virginia.

Masterson, John P. et al. (2016). *Assessment of Groundwater Availability in the Northern Atlantic Coastal Plain Aquifer System From Long Island, New York, to North Carolina*. Tech. rep. Professional Paper 1829. Reston, Virginia: USGS, p. 76. DOI: 10.3133/pp1829. URL: http://dx.doi.org/10.3133/pp1829.

Masterson, John P and Stephen P Garabedian (2007). "Effects of sea-level rise on ground water flow in a coastal aquifer system". In: *Ground Water* 45.2, pp. 209–217. ISSN: 0017467X. DOI: 10.1111/j.1745-6584.2006.00279.x.

MathWorks (2015). *Outlier removal using Hampel identifier*. URL: https://www.mathworks.com/help/signal/ref/hampel.html.

McDonnell, Bryant et al. (Aug. 2020). "PySWMM: The Python Interface to Stormwater Management Model (SWMM)". In: *Journal of Open Source Software* 5.52, p. 2292. ISSN: 2475-9066. DOI: 10.21105/joss.02292. URL: https://joss.theoj.org/papers/10.21105/joss.02292.

Mnih, Volodymyr et al. (2015). "Human-level control through deep reinforcement learning". In: *Nature* 518, pp. 529–543. DOI: 10.1038/nature14236. URL: https://www.nature.com/articles/nature14236.pdf.

Moftakhari, Hamed R. et al. (Nov. 2015). "Increased nuisance flooding along the coasts of the United States due to sea level rise: Past and future". In: *Geophysical Research Letters* 42.22, pp. 9846–9852. ISSN: 00948276. DOI: 10.1002/2015GL066072. URL: http://doi.wiley.com/10.1002/2015GL066072.

Moftakhari, Hamed R et al. (Feb. 2017). "Cumulative hazard: The case of nuisance flooding". In: *Earth's Future* 5.2, pp. 214–223. ISSN: 2328-4277. DOI: 10.1002/2016EF000494. URL: http://tidesandcurrents.noaa.gov/%20https://onlinelibrary.wiley.com/doi/abs/10.1002/2016EF000494.

Mohanty, S. et al. (July 2013). "Comparative evaluation of numerical model and artificial neural network for simulating groundwater flow in Kathajodi–Surua Inter-basin of Odisha, India". In: *Journal of Hydrology* 495, pp. 38–51. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2013.04.041. URL: http://linkinghub.elsevier.com/retrieve/pii/S0022169413003466.

Montestruque, Luis and M D Lemmon (2015). "Globally Coordinated Distributed Storm Water Management System". In: *Proceedings of the 1st ACM International Workshop on Cyber-Physical Systems for Smart Water Networks - CySWater'15*. New York, New York, USA: ACM Press, pp. 1–6. ISBN: 9781450334853. DOI: 10.1145/2738935.2738948. URL: http://dx.doi.org/10.1145/2738935.2738948.%20http://dl.acm.org/citation.cfm?doid=2738935.2738948.

Moss, Alaurah and Marco Marani (2016). "Coastal Water Table Mapping: Incorporating Groundwater Data into Flood Inundation Forecasts". PhD thesis. Duke University. URL: https://dukespace.lib.duke.edu/dspace/bitstream/handle/10161/11844/Moss_Masters_Project_Update.pdf?sequence=3.

Mounce, S R et al. (2020). "Optimisation of a fuzzy logic-based local real-time control system for mitigation of sewer flooding using genetic algorithms". In: *Journal of Hydroinformatics* 22.2, pp. 281–295. ISSN: 1464-7141. DOI: 10.2166/hydro.2019.058.

Mullapudi, Abhiram and Branko Kerkez (2018). "Autonomous Control of Urban Storm Water Networks Using Reinforcement Learning". In: *EPiC Series in Engineering* 3, pp. 1465–1469.

Mullapudi, Abhiram, Brandon P. Wong, and Branko Kerkez (Jan. 2017). "Emerging investigators series: building a theory for smart stormwater systems". In: *Environmental Science: Water Research & Technology* 3.1, pp. 66–77. ISSN: 2053-1400. DOI: 10.1039/C6EW00211K. URL: https://github.com/kLabUM/control-sim-es-wrt.%20http://xlink.rsc.org/?DOI=C6EW00211K.

Mullapudi, Abhiram et al. (2020). "Deep Reinforcement Learning for the Real Time Control of Stormwater Systems". In: *Advances in Water Resources*, p. 103600. ISSN: 03091708. DOI: 10.1016/j.advwatres.2020.103600. URL: https://linkinghub.elsevier.com/retrieve/pii/S0309170820302499.

Murphy, Rebecca R., W. Michael Kemp, and William P. Ball (Nov. 2011). "Long-Term Trends in Chesapeake Bay Seasonal Hypoxia, Stratification, and Nutrient Loading". In: *Estuaries and Coasts* 34.6, pp. 1293–1309. ISSN: 15592723. DOI: 10.1007/s12237-011-9413-7. URL: https://link.springer.com/article/10.1007/s12237-011-9413-7.

Muschalla, Dirk et al. (Apr. 2014). "Ecohydraulic-driven real-time control of stormwater basins". In: *Journal of Hydrology* 511, pp. 82–91. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2014.01.002.

Mynett, Arthur E and Zoran Vojinovic (2009). "Hydroinformatics in multi-colours-part red: urban flood and disaster management". In: *Journal of Hydroinformatics* 11.3.4, pp. 166–180. DOI: 10.2166/hydro.2009.027. URL: https://iwaponline.com/jh/article-pdf/11/3-4/166/386365/166.pdf.

Nayak, Purna C, Y R Satyaji Rao, and K P Sudheer (2006). "Groundwater Level Forecasting in a Shallow Aquifer Using Artificial Neural Network Approach". In: *Water Resources Management* 20, pp. 77–90. DOI: 10.1007/s11269-006-4007-z. URL: http://download.springer.com/static/pdf/454/art%253A10.1007%252Fs11269-006-4007-z.pdf?originUrl=http%3A%2F%2Flink.springer.com%2Farticle%2F10.1007%2Fs11269-006-4007-z&token2=exp=1497298557~acl=%2Fstatic%2Fpdf%2F454%2Fart%25253A10.1007%25252Fs11269-006-400.

Neumann, James E et al. (July 2015). "Climate change risks to US infrastructure: impacts on roads, bridges, coastal development, and urban drainage". In: *Climatic Change* 131.1, pp. 97–109. ISSN: 0165-0009. DOI: 10.1007/s10584-013-1037-4. URL: http://link.springer.com/10.1007/s10584-013-1037-4.

NOAA (2018a). *Harmonic Analysis*. URL: https://tidesandcurrents.noaa.gov/harmonic.html.

— (2018b). *Sewells Point - Station Home Page - NOAA Tides & Currents*. URL: https://tidesandcurrents.noaa.gov/stationhome.html?id=8638610.

— (2018c). *Tide Predictions - Help - NOAA Tides and Currents*. URL: https://tidesandcurrents.noaa.gov/PageHelp.html.

— (2018d). *Tide Predictions - NOAA Tides and Currents*. URL: https://tidesandcurrents.noaa.gov/noaatidepredictions.html?id=8638610.

Norfolk, Virginia of (2018). *CHESAPEAKE BAY TMDL ACTION PLAN VSMP MS4 Permit No. VA0088650*. Tech. rep. Norfolk. URL: https://www.norfolk.gov/DocumentCenter/View/38025/Final-Report---Chesapeake-Bay-TMDL-Action-Plan---06_28_2018_FINAL?bidId=.

OptiRTC and Geosyntec Consultants Inc. (2017). *Water Quality Summary Report National Fish and Wildlife Foundation Smart, Integrated Stormwater Management Systems Anacostia River Watershed Water Quality Study*. Tech. rep. URL: www.optirtc.com.

Parisi, Simone, Matteo Pirotta, and Marcello Restelli (Oct. 2016). "Multi-objective reinforcement learning through continuous pareto manifold approximation". In: *Journal of Artificial Intelligence Research* 57, pp. 187–227. ISSN: 10769757. DOI: 10.1613/jair.4961. URL: https://jair.org/index.php/jair/article/view/11026.

Park, Eungyu and J C Parker (2008). "A simple model for water table fluctuations in response to precipitation". In: *Journal of Hydrology* 356, pp. 344–349. DOI: 10.1016/j.jhydrol.2008.04.022. URL: http://ac.els-cdn.com/S0022169408002035/1-s2.0-S0022169408002035-main.pdf?_tid=f497e298-3b0b-11e7-8e57-00000aacb35e&acdnat=1495030999_82aa0e144484e9fecc5256a6c978c43d.

Pauw, P S et al. (2014). "Regional scale impact of tidal forcing on groundwater flow in unconfined coastal aquifers". In: *JOURNAL OF HYDROLOGY* 517, pp. 269–283. DOI: 10.1016/j.jhydrol.2014.05.042. URL: http://ac.els-cdn.com/S0022169414004065/1-s2.0-S0022169414004065-main.pdf?_tid=d196c642-3af7-11e7-999d-00000aab0f02&acdnat=1495022350_66b85505fc361b3fb27b5f4fc6b53912.

Pells, Steven E. and Philip J. N. Pells (Jan. 2016). "Application of Dupuit's Equation in SWMM to Simulate Baseflow". In: *Journal of Hydrologic Engineering* 21.1, p. 06015009. ISSN: 1084-0699. DOI: 10.1061/(asce)he.1943-5584.0001245. URL: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29HE.1943-5584.0001245.

Pianosi, F, A Castelletti, and M Restelli (2013). "Tree-based fitted Q-iteration for multi-objective Markov decision processes in water resource management". In: *Journal of Hydroinformatics* 15.2, pp. 258–270. DOI: 10.2166/hydro.2013.169. URL: https://iwaponline.com/jh/article-pdf/15/2/258/386917/258.pdf.

Plappert, Matthias (2016). *keras-rl*. URL: https://github.com/keras-rl/keras-rl.

Pregnolato, Maria et al. (2017). "The impact of flooding on road transport: A depth-disruption function". In: *Transportation Research Part D: Transport and Environment* 55, pp. 67–81. ISSN: 13619209. DOI: 10.1016/j.trd.2017.06.020.

Pumperla, Max (2015). *Hyperas*. URL: http://maxpumperla.com/hyperas/.

Quinn, J. D. et al. (July 2019). "What Is Controlling Our Control Rules? Opening the Black Box of Multireservoir Operating Policies Using Time-Varying Sensitivity Analysis". In: *Water Resources Research* 55.7, pp. 5962–5984. ISSN: 0043-1397. DOI: 10.1029/2018WR024177. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2018WR024177.

Ran, Youhua et al. (2015). "Optimal selection of groundwater-level monitoring sites in the Zhangye Basin, Northwest China". In: *JOURNAL OF HYDROLOGY* 525, pp. 209–215. DOI: 10.1016/j.jhydrol.2015.03.059. URL: http://dx.doi.org/10.1016/j.jhydrol.2015.03.059.

Read, Jordan S. et al. (Nov. 2019). "Process-Guided Deep Learning Predictions of Lake Water Temperature". In: *Water Resources Research* 55.11, pp. 9173–9190. ISSN: 19447973. DOI: 10.1029/2019WR024922. URL: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2019WR024922%20https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019WR024922%20https://agupubs.onlinelibrary.wiley.com/doi/10.1029/2019WR024922.

Rohilla Shalizi, Cosma (2018). "Bootstrapping Time Series". In: *Advanced Data Analysis from an Elementary Point of View*. Chap. 25, pp. 587–590. URL: http://www.stat.cmu.edu/~cshalizi/ADAfaEPoV/.

Rossman, Lewis A and Wayne C Huber (2016). *Storm Water Management Model Reference Manual Volume III – Water Quality*. Tech. rep. Cincinnati: USEPA.

Rotzoll, Kolja and Charles H Fletcher (2012). "Assessment of groundwater inundation as a consequence of sea-level rise". In: *Nature Climate Change* 3, pp. 477–481. DOI: 10.1038/NCLIMATE1725. URL: https://www.nature.com/articles/nclimate1725.pdf.

Sadler, J.M. et al. (Apr. 2018). "Modeling urban coastal flood severity from crowd-sourced flood reports using Poisson regression and Random Forest". In: *Journal of Hydrology* 559, pp. 43–55. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2018.01.044. URL: http://linkinghub.elsevier.com/retrieve/pii/S0022169418300519.

Sadler, J.M. et al. (2020a). "Exploring real-time control of stormwater systems for mitigating flood risk due to sea level rise". In: *Journal of Hydrology* 583. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2020.124571.

Sadler, Jeffrey M. et al. (Oct. 2019). "Leveraging open source software and parallel computing for model predictive control of urban drainage systems using EPA-SWMM5". In: *Environmental Modelling & Software* 120, p. 104484. DOI: 10.1016/j.envsoft.2019.07.009. URL: https://linkinghub.elsevier.com/retrieve/pii/S1364815218312325.

Sadler, Jeffrey M. et al. (2020b). "Exploring real-time control of stormwater systems for mitigating flood risk due to sea level rise". In: *Journal of Hydrology* 583. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2020.124571.

Sadler, Jeffrey M, Jonathan L Goodall, and Mohamed M Morsy (2017). "Effect of Rain Gauge Proximity on Rainfall Estimation for Problematic Urban Coastal Watersheds in Virginia Beach, Virginia". In: *Journal of Hydrologic Engineering* 22.9. DOI: 10.1061/(ASCE)HE.1943-5584.0001563. URL: http://www.wunderground.com/.

Sahoo, Sasmita and Madan K Jha (2013). "Groundwater-level prediction using multiple linear regression and artificial neural network techniques: a comparative assessment". In: *Hydrogeology Journal* 21, pp. 1865–1887. DOI: 10.1007/s10040-013-1029-5. URL: http://download.springer.com/static/pdf/269/art%253A10.1007%252Fs10040-013-1029-5.pdf?originUrl=http%3A%2F%2Flink.springer.com%2Farticle%2F10.1007%2Fs10040-013-1029-5&token2=exp=1497298600~acl=%2Fstatic%2Fpdf%2F269%2Fart%25253A10.1007%25252Fs10040-013-102.

Saliba, Sami M et al. (Nov. 2020). "Deep Reinforcement Learning with Uncertain Data for Real-Time Stormwater System Control and Flood Mitigation". In: *Water* 12.11, p. 3222. ISSN: 2073-4441. DOI: 10.3390/w12113222. URL: www.mdpi.com/journal/water%20https://www.mdpi.com/2073-4441/12/11/3222.

Schwanenberg, D, B P J Becker, and M Xu (2015). "The open real-time control (RTC)-Tools software framework for modeling RTC in water resources sytems". In: *Journal of Hydroinformatics* 17.1, pp. 130–148. DOI: 10.2166/hydro.2014.046. URL: https://iwaponline.com/jh/article-pdf/17/1/130/387947/jh0170130.pdf.

SciPy (2019a). *scipy.signal.find_peaks — SciPy v1.2.1 Reference Guide*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html.

— (2019b). *scipy.stats.ttest_ind — SciPy v1.2.1 Reference Guide*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_ind.html.

Sharior, Sazzad, Walter McDonald, and Anthony J Parolari (2019). "Improved reliability of stormwater detention basin performance through water quality data-informed real-time control". In: *Journal of Hydrology* 573, pp. 422–431. DOI: 10.1016/J.JHYDROL.2019.03.012. URL: https://www.sciencedirect.com/science/article/pii/S0022169419302598.

Shen, Chaopeng (2018). "A trans-disciplinary review of deep learning research and its relevance for water resources scientists". In: *Water Resources Research*, pp. 1–81. ISSN: 00431397. DOI: 10.1029/2018WR022643. URL: http://doi.wiley.com/10.1029/2018WR022643.

Shen, Yawen et al. (2019). "Flood risk assessment and increased resilience for coastal urban watersheds under the combined impact of storm tide and heavy rainfall". In: *Journal of Hydrology* 579, p. 124159. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2019.124159.

Shishegar, Shadab, Sophie Duchesne, and Geneviève Pelletier (2019). "An integrated optimization and rule-based approach for predictive real time control of urban stormwater management systems". In: *Journal of Hydrology* 577, p. 124000. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2019.124000.

Shishegar, Shadab et al. (Jan. 2021). "A smart predictive framework for system-level stormwater management optimization". In: *Journal of Environmental Management* 278, p. 111505. ISSN: 10958630. DOI: 10.1016/j.jenvman.2020.111505.

Smirnov, Dmitry et al. (2018). *Analysis of Historical and Future Heavy Precipitation.* Tech. rep. Virginia Beach, Virginia: City of Virginia Beach Department of Public Works. URL: https://www.vbgov.com/government/departments/public-works/comp-sea-level-rise/Documents/anaylsis-hist-and-future-hvy-precip-4-2-18.pdf.

Solomatine, Dimitri P. and Avi Ostfeld (Jan. 2008). "Data-driven modelling: some past experiences and new approaches". In: *Journal of Hydroinformatics* 10.1, pp. 3–22. ISSN: 1464-7141. DOI: 10.2166/hydro.2008.015. URL: https://iwaponline.com/jh/article/10/1/3-22/2913.

Srivastava, Nitish et al. (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15, pp. 1929–1958. URL: http://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf?utm_content=buffer79b43&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer.

Su, Jianyu, Stephen C. Adams, and Peter A. Beling (2020). "Value-Decomposition Multi-Agent Actor-Critics". In: *CoRR* abs/2007.1. URL: https://arxiv.org/abs/2007.12306.

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement Learning: An Introduction.* 2nd. Cambridge, Massachusetts: The MIT Press, p. 526. ISBN: 9780262039246. URL: https://drive.google.com/file/d/1opPSz5AZ_kVa1uWOdOiveNiBFiEOHjkG/view.

Sweet, William V. and Joseph Park (Dec. 2014). "From the extreme to the mean: Acceleration and tipping points of coastal inundation from sea level rise". In: *Earth's Future* 2.12, pp. 579–600. ISSN: 23284277. DOI: 10.1002/2014EF000272. URL: http://doi.wiley.com/10.1002/2014EF000272.

Taormina, Riccardo, Kwok-Wing Chau, and Rajandrea Sethi (2012). "Artificial neural network simulation of hourly groundwater levels in a coastal aquifer system of the Venice lagoon". In: *Engineering Applications of Artificial Intelligence* 25, pp. 1670–1676. DOI: 10.1016/j.engappai.2012.02.009. URL: http://ac.els-cdn.com/S0952197612000462/1-s2.0-S0952197612000462-main.pdf?_tid=7f7d8fa2-663b-11e7-885f-00000aab0f6b&acdnat=1499779318_e4e8d3caddbf42e19b039b1f57f5025b.

Tian, Ye et al. (2018). "Integration of a Parsimonious Hydrological Model with Recurrent Neural Networks for Improved Streamflow Forecasting". In: *Water* 10.11, p. 1655. DOI: 10.3390/w10111655. URL: http://www.mdpi.com/2073-4441/10/11/1655.

Troutman, Sara C., Nancy G. Love, and Branko Kerkez (May 2020). "Balancing water quality and flows in combined sewer systems using real-time control". In: *Environmental Science: Water Research and Technology* 6.5, pp. 1357–1369. ISSN: 20531419. DOI: 10.1039/c9ew00882a. URL: https://pubs.rsc.org/en/content/articlehtml/2020/ew/c9ew00882a%20https://pubs.rsc.org/en/content/articlelanding/2020/ew/c9ew00882a.

USCB (2018). *U.S. Census Bureau QuickFacts: Norfolk city, Virginia.* URL: https://www.census.gov/quickfacts/fact/table/norfolkcityvirginia/PST045217.

Virginia Department of Environmental Quality (2015). *Chesapeake Bay TMDL Action Plan Guidance.*

Walsh, Christopher J. et al. (July 2005). "The urban stream syndrome: Current knowledge and the search for a cure". In: *Journal of the North American Benthological Society.* Vol. 24. 3. North American Benthological Society, pp. 706–723. DOI: 10.1899/04-028.1. URL: https://www.journals.uchicago.edu/doi/abs/10.1899/04-028.1.

Wang, Cheng et al. (2020). "Smart Stormwater Control Systems: A Reinforcement Learning Approach". In: *ISCRAM 2020 Conference Proceedings - 17th International Con-*

ference on Information Systems for Crisis Response and Management. Ed. by Amanda Lee Hughes, Fiona McNeill, and Christopher Zobel. Blacksburg, VA, pp. 2–13. URL: https://github.com/OpenWaterAnalytics/pyswmm.

Wang, Yu and Hongxia Jin (June 2018). "A Boosting-based Deep Neural Networks Algorithm for Reinforcement Learning". In: *2018 Annual American Control Conference (ACC)*. IEEE, pp. 1065–1071. ISBN: 978-1-5386-5428-6. DOI: 10.23919/ACC.2018.8431647. URL: https://cpb-us-e1.wpmucdn.com/campuspress-test.yale.edu/dist/7/677/files/2018/03/paper_RL_2017_CDC-uotjpy.pdf%20https://ieeexplore.ieee.org/document/8431647/.

Werbos, P.J. (1990). "Backpropagation through time: what it does and how to do it". In: *Proceedings of the IEEE* 78.10, pp. 1550–1560. DOI: 10.1109/5.58337. URL: http://ieeexplore.ieee.org/document/58337/.

Wiering, Marco A. and Hado van Hasselt (Aug. 2008). "Ensemble algorithms in reinforcement learning". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 38.4, pp. 930–936. ISSN: 10834419. DOI: 10.1109/TSMCB.2008.920231. URL: https://ieeexplore.ieee.org/document/4509588.

Wong, B P and B Kerkez (2018). "Real-Time Control of Urban Headwater Catchments Through Linear Feedback: Performance, Analysis, and Site Selection". In: *Water Resources Research* 54.10, pp. 7309–7330. DOI: 10.1029/2018WR022657. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2018WR022657.

Wong, Brandon P. and Branko Kerkez (Nov. 2016). "Adaptive measurements of urban runoff quality". In: *Water Resources Research* 52.11, pp. 8986–9000. ISSN: 00431397. DOI: 10.1002/2015WR018013. URL: http://doi.wiley.com/10.1002/2015WR018013.

Wright, Jeffrey and Dayton Marchese (2017). "Briefing: Continuous monitoring and adaptive control: the 'smart' storm water management solution". In: *Proceedings of the Institution of Civil Engineers - Smart Infrastructure and Construction* 170.4, pp. 86–89. ISSN: 2397-8759. DOI: 10.1680/jsmic.17.00017. URL: https://doi.org/10.1680/jsmic.17.00017%20https://www.icevirtuallibrary.com/doi/10.1680/jsmic.17.00017.

Wuebbles, D.J. et al. (2017). *Climate Science Special Report: Fourth National Climate Assessment, Volume I*. Tech. rep. Washington, DC: U.S. Global Change Research Program (USGCRP), p. 470. DOI: 10.7930/J0J964J6.. URL: https://science2017.globalchange.gov/downloads/CSSR2017_FullReport.pdf.

Yang, Tiantian et al. (2017). "Developing reservoir monthly inflow forecasts using artificial intelligence and climate phenomenon information". In: *Water Resources Research* 53.4, pp. 2786–2812. DOI: 10.1002/2017WR020482. URL: http://doi.wiley.com/10.1002/2017WR020482.

Yaseen, Zaher Mundher et al. (2015). "Artificial intelligence based models for stream-flow forecasting: 2000–2015". In: *Journal of Hydrology* 530, pp. 829–844. DOI: 10.1016/J.JHYDROL.2015.10.038. URL: https://www.sciencedirect.com/science/article/pii/S0022169415008069.

Yoon, Heesung et al. (2011). "A comparative study of artificial neural networks and support vector machines for predicting groundwater levels in a coastal aquifer". In: *Journal of Hydrology* 396, pp. 128–138. DOI: 10.1016/j.jhydrol.2010.11.002. URL: http://ac.els-cdn.com/S0022169410006761/1-s2.0-S0022169410006761-main.pdf?_tid=645d2bda-4c62-11e7-a876-00000aacb35d&acdnat=1496937293_42a6e50fce90322be6a1f07fe55a298e.

Zhang, Duo, Geir Lindholm, and Harsha Ratnaweera (2018a). "Use long short-term memory to enhance Internet of Things for combined sewer overflow monitoring". In: *Journal of Hydrology* 556, pp. 409–418. DOI: 10.1016/j.jhydrol.2017.11.018. URL: https://ac.els-cdn.com/S0022169417307722/1-s2.0-S0022169417307722-main.pdf?_tid=c3f0094b-

062c-47b6-b02d-a2a7b3a6fdb3&acdnat=1523474052_43f0d59b99dea932c194a64877c89ee1.
Zhang, Jianfeng et al. (2018b). "Developing a Long Short-Term Memory (LSTM) based model for predicting water table depth in agricultural areas". In: *Journal of Hydrology* 561, pp. 918–929. DOI: 10.1016/j.jhydrol.2018.04.065. URL: http://linkinghub.elsevier.com/retrieve/pii/S0022169418303184.
Zhao, Zheng et al. (Mar. 2017). "LSTM network: a deep learning approach for short-term traffic forecast". In: *IET Intelligent Transport Systems* 11.2, pp. 68–75. ISSN: 1751-956X. DOI: 10.1049/iet-its.2016.0208. URL: https://digital-library.theiet.org/content/journals/10.1049/iet-its.2016.0208.