Modelling Historical Reservoir Releases with Recurrent Deep Learning Methods

Mattéo Debiere

Master's thesis
2026:E78

Lund University

Faculty of Science
Centre for Mathematical Sciences
Computational Science

Centrum Scientiarum Mathematicarum

Modelling Historical Reservoir Releases with Recurrent Deep Learning Methods

Mattéo Debiere

Thesis supervisor: Zheng Duan
Co-supervisor: Shaokun He

Date: 12/06/2026

Abstract

Reliable representation of reservoir operations in large-scale hydrological models remains challenging because reservoir releases are shaped not only by highly variable hydrological conditions but also by complex human operating objectives. While simplified rule-based schemes are commonly used, they often struggle to reproduce observed release behaviour across diverse reservoirs. This thesis evaluates whether recurrent machine learning models can reproduce historical daily reservoir outflows across multiple reservoirs using recent daily inflow, storage, and cyclical day-of-year information as dynamic inputs. It further assesses whether conditioning these models on static reservoir attributes, namely main use, broad climate group, and storage capacity, improves their performance and generalisation.

Long short-term memory (LSTM) and gated recurrent unit (GRU) models were trained to simulate historical reservoir outflow using these dynamic inputs. Both standard and conditioned versions of each architecture were evaluated. The conditioned models used the static reservoir attributes to initialise the recurrent state. Two dataset versions were used: a main dataset based on ResOpsUS, and an extra dataset that additionally included Chinese reservoir records. For evaluation, 20 reservoirs were held out entirely to test generalisation to unseen reservoirs, while the remaining reservoirs were split temporally into training, validation, and in-sample test periods. Models were trained by minimising MAE on normalised outflow, and hyperparameters were selected using normalised validation MAE. Model performance was assessed using MAE, RMSE, Nash–Sutcliffe efficiency, and Kling–Gupta efficiency. Additional experiments tested the effect of lookback-window length, with seven-, three- and one-day windows being investigated, and assessed the models' robustness to Gaussian noise.

The results show that recurrent models are suitable for the task examined in this thesis. Conditioned LSTM models achieved the strongest validation and in-sample test performance, suggesting that static reservoir attributes help when predicting later periods for reservoirs represented during training. However, this improvement did not transfer consistently to fully held-out reservoirs, indicating that generalisation to unseen reservoirs remains challenging. The seven-day lookback window gave the best overall performance, while robustness testing showed that perturbations to storage caused the largest reduction in performance. Compared with the rule-based baseline, the recurrent models better reproduced the timing and variability of daily releases.

Overall, this thesis shows that recurrent machine learning models are a promising approach for reproducing historical reservoir outflows. Static conditioning can improve temporal generalisation, but further work is needed to improve transferability to fully held-out reservoirs.

Popular Scientific Abstract

Reservoirs store water for uses like drinking water, farming and generating electricity. For these uses, they decide when and how much water to release. Rules for these releases are usually not available publicly, so can be difficult to predict in large water models.

This thesis tests whether computer models can learn how reservoirs are operated from historical data. They are given recent information about how much water is entering the reservoir, how much water is already there, and the time of year. Some models are also additionally given simple information about the reservoirs, like its main purpose, the climate around it and its size.

We found that the computer models were better at this task than other methods that are traditionally used. The models that were given the additional information had the best results for reservoirs that they had already seen in training. However, predicting releases for completely new reservoirs was still difficult, and the simpler models without the additional information still did better on them.

Acknowledgements

Firstly, I would like to thank my supervisor, Zheng Duan, and my co-supervisor, Shaokun He, for their expertise and invaluable feedback throughout the development and writing of this thesis.

I would like to thank my parents, for their unconditional support in many ways, and for enabling me to take risks.

I am grateful to the friends I have made in Lund for their companionship and help, both inside and outside of class.

Finally, I would like to thank Martina for being there every single day, for her patience, and for her love.

To Kanga

Contents

1 Introduction

Reservoirs are man-made water storage systems created by retaining rivers or streams behind dams. They enable water to be stored and regulated for uses such as irrigation, water supply, hydropower generation, flood control, and navigation (Lehner et al., 2011; World Commission on Dams, 2000).

According to Wang et al. (2022), one of the most comprehensive global dam datasets is the World Register of Dams, which reports around 62 362 dams worldwide (International Commission on Large Dams, 2025). This shows the extent of reservoir operations and the importance of being able to study reservoir behaviour on a global scale. However, while inventories of dam locations and characteristics are increasingly available, detailed information on how reservoirs are actually operated often remains hidden.

This is particularly an issue in the case of hydrological models, where reservoir regulation is challenging to represent (Wada et al., 2017). Reservoir releases are not determined by hydrological conditions alone, but also by decisions taken by the reservoir operators. These decisions are affected by the reservoir objectives, seasonal priorities, anticipated inflows, downstream demand, and flood risk. Even under identical hydrological conditions, reservoirs that have different main uses may exhibit different release patterns. Often, individual reservoirs have multiple purposes, which further complicates the problem.

A large part of the reservoir operation literature has focused on optimisation, where operating rules and models are created to optimise reservoir operation (Giuliani et al., 2021). In contrast, the focus of this thesis is not on identifying optimal operation strategies, but on modelling historical reservoir behaviour from observed data. The aim is therefore to reproduce how reservoirs were actually operated, rather than how they should ideally be operated.

Traditionally, reservoir releases are guided by operating rules, often expressed as rule curves or functions that describe how storage and release decisions are expected to vary under different conditions (Hanasaki et al., 2006). However, these rules do not always fully reflect historical reservoir behaviour. In practice, reservoir operation may change over time with changes in demand, climate, objectives, or institutions. In addition, even when operating rules are known for a given reservoir, they may not capture the human factor, where operators might choose to deviate from the recommended releases.

For hydrological modelling, an additional difficulty lies in translating operating policies for single reservoirs into a form that can be applied consistently

across many systems. Even when operating rules are available for specific reservoirs, collecting them and applying them to the correct reservoirs is a very difficult task. This has contributed to the widespread use of simplified reservoir representations in large-scale models, despite the limited ability of such approaches to reproduce observed operations in detail (J. C. Steyaert & Condon, 2024).

Data-driven methods, such as long short-term memory (LSTM) networks and gated recurrent units (GRU), provide an alternative by predicting release behaviour directly from historical observations, without needing prior knowledge of the reservoir-specific operating rules. This is particularly useful for reservoir modelling over large scales, where detailed operating policies are often unavailable, simplified, or may not fully reflect how releases are performed in practice. Recurrent models are well suited to this task because release decisions may depend not only on the conditions of a single day, but also on the recent sequence of inflow and storage conditions. The addition of static reservoir attributes such as main use, climate group, and capacity can provide information that can help the model distinguish between reservoirs. This is important in large-scale reservoir modelling because similar inflow and storage conditions may lead to different release behaviour (Tran et al., 2025).

Main use refers to the primary purpose of the reservoir, such as flood control, irrigation, hydroelectricity and water supply. For example, irrigation and water-supply reservoirs are meant to store water during periods of oversupply and release it during periods of demand, while hydropower reservoirs are operated for electricity generation, which places additional constraints on the releases. The addition of main use information is therefore, in theory, critical for the model to be able to adapt to the large differences in operations.

This thesis also examines the effect of input sequence length, since the amount of recent history provided to the model may influence both performance and applicability.

The literature shows a need to test whether data-driven recurrent models can reproduce observed reservoir releases across diverse reservoirs, especially where operating rules are unavailable or incomplete. While Tran et al. (2025) applied a conditioned LSTM approach using ResOpsUS, a dataset of historical reservoir operations in the contiguous United States (J. Steyaert et al., 2021, 2022), it remains useful to evaluate whether this approach remains effective with more data from another region, alternative recurrent architectures, and different modelling assumptions. This thesis addresses these gaps by comparing standard and conditioned LSTM and GRU models on both the ResOpsUS dataset and an additional Chinese reservoir dataset.

It also assesses whether conditioning on static reservoir attributes, including main use, climate group, and capacity, improves generalisation, while testing the effect of lookback-window length and input perturbations.

The following research questions are addressed:

1. How well can recurrent machine learning models reproduce historical daily reservoir outflows, and does conditioning them on static reservoir attributes improve performance and generalisation to reservoirs not seen during training?
2. How robust are the trained recurrent models to uncertainty in the input variables and to the size of the lookback window?
3. How does the performance of the recurrent machine learning models compare with a traditional reservoir operation baseline?

This thesis continues as follows:

Section 2 reviews approaches to modelling historical reservoir operations, recurrent machine learning models, static conditioning, robustness testing, and the role of sequence length. Section 3 describes the datasets and the preparation steps applied before modelling. Section 4 describes the modelling, training, and evaluation methodology used in this thesis. Section 5 presents the results of the model selection, baseline comparison, held-out evaluation, lookback-window experiment, and robustness testing. Section 6 discusses the main findings, their implications, and the limitations of the study. Section 7 summarises the conclusions of the thesis and suggests possible directions for future work.

2 Literature Review

These sections review the literature relevant to modelling historical reservoir operation using recurrent machine learning models. We first discuss the main approaches that have been used to reconstruct reservoir operations, including rule-based schemes, optimisation methods, simplified large-scale reservoir models, and data-driven approaches. We then introduce the machine learning architectures used in this thesis, and how static reservoir attributes can be used as inputs to them. The section also discusses robustness to input perturbations, the role of sequence length, and the evaluation metrics commonly used in hydrological modelling.

2.1 Reservoir operation modelling approaches

Reservoir operation modelling approaches are typically divided into two types: predefined operating rules and data-driven policies. For hydrological models, simplified reservoir operation schemes are often used, because they can be applied across many reservoirs.

A common approach is to represent reservoir operation using predefined operating rules. In these, releases are expressed as a function of variables such as storage, inflow, demand, and time of year. Typical examples include rule curves and operating functions. A major advantage of this class of methods is that the resulting behaviour is easy to interpret and apply widely, which is especially relevant in the context of hydrological models.

A representative example is presented in Hanasaki et al. (2006), in which reservoir release is estimated using a generalised relationship between inflow, storage and water availability at the start of the season. This simplified scheme will be used as a benchmark for whether the machine learning models improve on a simple rule-based representation of reservoir behaviour. The Hanasaki non-irrigation reservoir operation scheme (HNIRS) is appropriate for this comparison because it was designed for global applications like the problem in this thesis.

Based on the literature reviewed, the earliest source to use machine learning techniques specifically for historical reservoir operation modelling reconstruction is Ehsani et al. (2016). This paper specifically mentions the application of machine learning models to reproduce reservoir release decisions from historical data, rather than to derive an optimal operating policy. Similarly to this thesis, they trained models to learn the relationship between recent inflow, storage, and seasonal information and the subsequent reservoir outflow. They also investigated whether the inclusion of static reservoir at-

tributes can improve performance across different reservoirs. However, they used a feed-forward artificial neural network for this task. While these models include information from previous timesteps, each past value is provided as a separate input variable. This means they do not process the temporal inputs as an ordered sequence. As a result, feed-forward architectures are theoretically less well suited to modelling temporal dependencies than recurrent models, such as those introduced in the following sections.

Another relevant data-driven approach is the Generic Data-driven Reservoir Operation Model (GDROM) proposed by Chen et al. (2022). Their aim was to create a model that is general and as effective as a machine learning model, while remaining interpretable. The approach first uses a hidden Markov-decision tree model to identify a small number of representative operation modules for each reservoir. This creates and trains tree models, each of which can be interpreted as a release rule. A classification and regression tree model is then used to identify which module should be applied at which time. This keeps much of the interpretability of decision-tree models, while reducing their complexity by grouping operating behaviour into a limited number of modules. Chen et al. (2022) found that only a few modules were needed to represent the operation of the reservoirs tested, and that the method performed better than traditional decision-tree models, particularly for storage prediction. This thesis is related to Chen et al. (2022) through its data-driven aim of learning reservoir operating behaviour from historical inflow, storage, and release records across multiple reservoirs. However, recurrent neural networks will be used to learn temporal release patterns directly, rather than separating operation into explicit decision-tree modules.

A recent publication in the field which has heavily inspired this thesis is Tran et al. (2025). In this publication, they developed the conditioned LSTM approach used for part of this thesis and applied it to the ResOpsUS and Global Reservoir and Dam (GRanD) datasets. Their model included main use, climate and maximum capacity along with the input time series. They found that the conditioned LSTM generally achieved better performance than both a non-conditioned LSTM and a policy-based approach. However, they also found that the conditioned model did not improve performance in every case, with the standard LSTM achieving better results for flood control reservoirs, for example. These findings provide a useful basis for the present thesis, which follows a similar conditioned LSTM framework but adds another dataset, additional model architectures, different preprocessing, and model inputs.

2.2 Machine learning models for time-series prediction

In this thesis, two machine learning model architectures were used. This section gives an overview of the relevant literature, and how these models operate.

Neural networks are a type of data-driven model, which learn relationships between inputs and outputs directly from data, without requiring explicit explanation of the physical or decision-making processes (Goodfellow et al., 2016). They consist of layers of interconnected neurons, where each neuron applies a nonlinear transformation to a weighted combination of its inputs. By adjusting these weights during training, usually through propagating errors in the output backwards through the network, neural networks can approximate complex, nonlinear relationships.

Recurrent Neural Networks (RNNs) are types of neural networks which can transmit information back into themselves. This is in contrast to other types of neural networks such as Multi-Layer Perceptrons (MLPs) which take a feed-forward approach, where information is always passed forward through the network with no recurrence. RNNs use cycles that transmit information back into themselves (Schmidt, 2019), allowing the network to take into account not only the current input but also the timestep before. This is the property which motivated their use in this thesis, as reservoir outflows do not only depend on the state on the current day, but also on the previous days.

Long short-term memory (LSTM) models are a type of RNN, which were introduced to handle the vanishing or exploding gradient problem (Hochreiter & Schmidhuber, 1997). The paper above introduces the concept of memory cells and gate units, to store information across various timesteps. They do this by introducing a memory cell, c_t, and a hidden state, h_t, which are controlled by gates. At each timestep t, the LSTM receives the current input x_t, the previous hidden state h_{t-1}, and the previous cell state c_{t-1}. The flow of information through the LSTM is controlled by several gates. The source above described two gates, input and output. The input gate controls how much new information is written to the cell state, and the output gate controls how much of the cell state is exposed as the hidden state. Forget gates were later added to the structure of LSTMs by Gers et al. (2000), with the purpose of controlling how much previous memory is retained. This allows the network to reset itself at appropriate times, removing useless information.

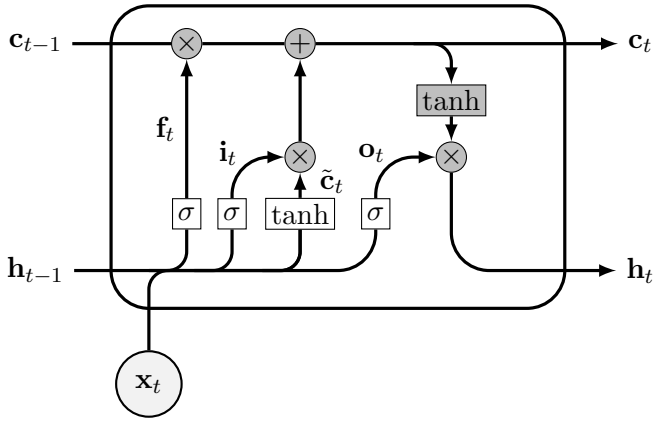


Figure 1: Schematic representation of a long short-term memory cell. The current input x_t, previous hidden state h_{t-1}, and previous cell state c_{t-1} are used to update the cell state c_t and hidden state h_t. Adapted from Love (2024).

The standard LSTM update can be written as:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \tag{1}$$
$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \tag{2}$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \tag{3}$$
$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c), \tag{4}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \tag{5}$$
$$h_t = o_t \odot \tanh(c_t), \tag{6}$$

where i_t, f_t, and o_t are the input, forget, and output gates respectively, $\tilde{c}_t$ is the candidate cell state, σ is the sigmoid activation function, and $\odot$ is an element-wise multiplication.

Figure 1 shows how the current input, previous hidden state, and previous cell state are combined within the cell. The upper pathway represents the cell state, which carries memory through time, while the gates below control how much information is forgotten, added, and exposed as the new hidden state.

Gated recurrent units (GRUs) are another gated RNN architecture, introduced by Cho et al. (2014). They were designed for a similar purpose as LSTMs, but with a simpler structure with fewer gates. Instead of having both a cell state and hidden state, the GRU only updates a hidden state. This is done using an update gate, which controls how much information from the previous hidden state is brought forward, while the reset gate controls how strongly past information is used. This allows the model to retain

useful information over time while discarding information that is no longer relevant.

The GRU update can be written as:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z), \tag{7}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r), \tag{8}$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h), \tag{9}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t, \tag{10}$$

where z_t is the update gate, r_t is the reset gate, and $\tilde{h}_t$ is the candidate hidden state.

The GRU equations show that the architecture performs a similar gated update to the LSTM, but without maintaining a separate cell state. This simpler structure is illustrated in Figure 2. The reset gate controls how much of the previous hidden state is used when computing the candidate hidden state, while the update gate determines the balance between retaining past information and replacing it with new information.

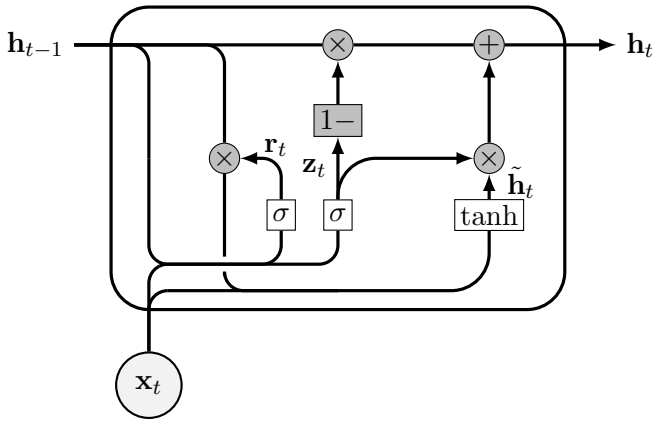


Figure 2: Schematic representation of a gated recurrent unit cell. The current input x_t and previous hidden state h_{t-1} are used to compute the reset gate r_t, update gate z_t, and candidate hidden state $\tilde{h}_t$. The update gate controls the balance between retaining the previous hidden state and using the candidate hidden state to produce h_t. Adapted from Love (2024).

Compared with LSTMs, GRUs therefore have fewer gates and fewer trainable parameters. This can make them faster to train and less prone to overfitting in some cases, while keeping the advantages of LSTMs. GRUs have already been applied to a similar problem by Zhang et al. (2019), who applied RNN, LSTM and GRU models to reservoir outflow prediction for the Xiluodu reservoir. In this thesis, GRUs are therefore included as a sim-

pler alternative to LSTMs, to see if they can achieve similar performance with smaller architectures.

2.3 Static attributes in recurrent models

The same time series inputs for different reservoirs can have different outputs because of differences in static attributes. With the same recent inflow and storage pattern, a flood-control reservoir may release water to maintain storage space for possible incoming floods, whereas an irrigation reservoir may retain that water for later demand. Climate is another relevant static attribute, since reservoir operation is strongly influenced by the climate the reservoir is in.

There are several ways to include static attributes in recurrent models. The simplest is to concatenate the attributes to the time-series data at every timestep, as considered by Kratzert et al. (2019). The recurrent layer in this approach receives the same static attributes repeatedly, throughout the length of the time-series inputs.

A more structured approach, as implemented in the Entity-Aware LSTM of Kratzert et al. (2019), is to modify the memory cell architecture of the LSTM directly. The standard time-varying input gate is replaced by a gate derived from the static attributes, while the remaining recurrent updates continue to depend on the time-series inputs.

Finally, another option is to use the static variables to initialise the recurrent model. This is the approach implemented in `cond_rnn` (Remy, 2020), and used in Tran et al. (2025), as well as this thesis. In this approach, the static attributes are first converted into the recurrent model's initial internal state, as shown in Figure 3. This is done using a simple trainable neural network. The recurrent model therefore begins each input sequence with information about the reservoir being simulated. The time-series inputs are then processed as usual, but their interpretation is influenced by the static attributes provided at the start.

The conditioning setup explained above allows the same time-series input sequence to be interpreted differently depending on the static attributes associated with it. This method is applied to both the LSTM and GRU models.

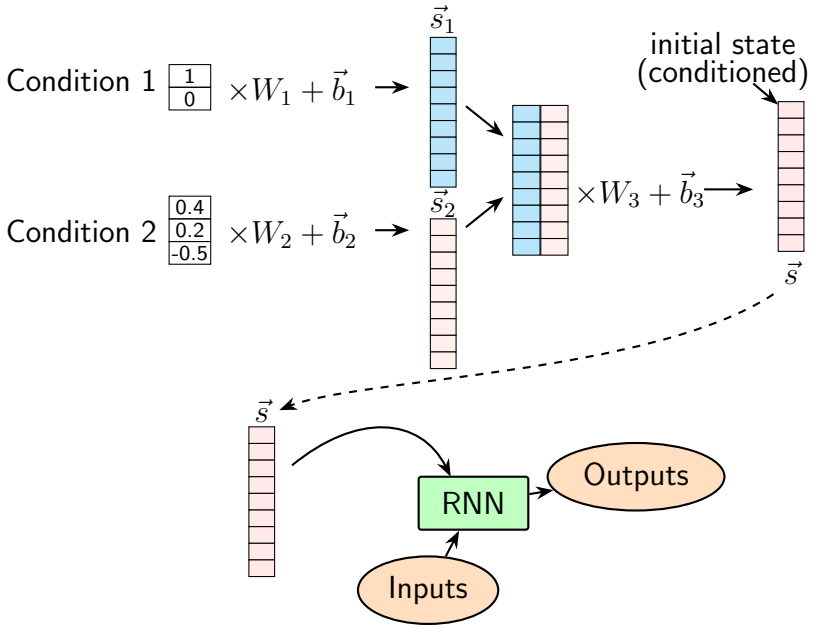


Figure 3: Schematic representation of the conditioned recurrent model. Static conditioning inputs are transformed into state vectors, concatenated, and mapped to the initial recurrent state $\vec{s}$. This initial state is then used by the recurrent model together with the time-series inputs to produce the output sequence. Adapted from Remy (2020).

2.4 Summary of modelling frameworks

Table 1 summarises the modelling frameworks discussed above in terms of their applications, advantages, limitations, and key references.

2.5 Robustness to input perturbations

Reservoir operation modelling is strongly affected by uncertainty in the input data. Observed inflow, storage and release records may contain measurement errors, and when the model is used within a hydrological model the uncertainties from other components can propagate through it. It is therefore useful to assess whether the trained models remain stable when there is additional noise in the inputs.

This was tested by adding Gaussian noise to the inputs of an already trained model and measuring the resulting change in predictive performance. Kratzert et al. (2019) used this approach for their Entity-Aware LSTM by adding Gaussian noise with increasing standard deviation to their static attributes. Their results showed a gradual decrease in performance as the noise level increased, suggesting that the model had learned the relationships between the inputs and reservoir operation, rather than relying only on exact attribute values.

In this thesis, Gaussian noise perturbation is used as a sensitivity test, rather than as a full variable importance test. This means that the results show how model performance changes when selected inputs have noise added, but they do not provide a direct measure of how important each variable is to the model. The results should also be interpreted with some limitations in mind, since some of the inputs may contain overlapping information, and large perturbations may move inputs outside realistic ranges. These are discussed further in subsection 6.8.

2.6 Sequence length

The sequence length experiment is carried out to evaluate how much recent information is useful for predicting reservoir releases. The term sequence length, or lookback window, refers to how many previous timesteps are provided as an input to the recurrent model. In this thesis, it defines how many days of inflow and storage are given to the model. This is an important decision because reservoir releases may not depend solely on the current day's conditions. Giving information about the recent past therefore, theoretically, gives critical information that can help the model performance.

Table 1: Summary of modelling frameworks relevant to hydrological time-series prediction and reservoir operation modelling.

Framework	Application	Advantages	Limitations	Key references
Rule-based reservoir schemes	Predefined release relationships using variables such as inflow, storage, demand, and seasonality.	Interpretable, widely applicable, and useful as a baseline.	May not capture unknown, changing, or reservoir-specific operating behaviour.	(Hanasaki et al., 2006)
Feed-forward neural networks	Data-driven release prediction from historical hydrological and seasonal inputs.	Can learn nonlinear relationships without explicit operating rules.	Past timesteps are supplied as separate inputs rather than processed as an ordered sequence.	(Ehsani et al., 2016)
Decision-tree and modular models	Identification of operating schemes from historical behaviour.	More interpretable than neural networks and able to represent different operating modes.	Separate behaviour into explicit modules rather than learning temporal patterns directly.	(Chen et al., 2022)
Standard recurrent models	Time-series prediction using recent inflow, storage, and seasonal information. Includes LSTM and GRU models.	Process ordered sequences, so previous timesteps can influence later predictions.	More computationally expensive and performance depends on the selected lookback window	(Cho et al., 2014; Fan et al., 2023; Hochreiter & Schmidhuber, 1997; Zhang et al., 2019)
Conditioned recurrent models	Recurrent models trained across multiple reservoirs with static reservoir attributes.	Allow dynamic inputs to be interpreted in the context of reservoir characteristics.	The benefit depends on whether the static attributes describe operational differences well.	(Kratzert et al., 2019; Remy, 2020; Tran et al., 2025)

Recurrent architectures such as the ones used in this thesis are specifically designed for time series data. Thanks to their architectures they can extract information from timesteps and choose which to use or disregard. Theoretically, longer input sequences are therefore ideal for these models. However, they can lead to problems such as increasing the training time, and overfitting where the model learns the individual samples rather than the underlying patterns, which leads to poor generalisation performance. Smaller lookback windows may also be easier to implement and more useful in practical applications, since they require a shorter continuous input record.

A similar sensitivity to input sequence length has been investigated by Fan et al. (2023), who treated the LSTM lookback window as a hyperparameter for reservoir release prediction. They tested lag lengths from 1 to 15 days using validation performance and found that for an example reservoir, increasing the length of the lookback window increased performance until the window was 7 days. After this point, they found that increasing the lookback window length led to lower performance. They therefore chose to use 7 days as the lookback window length for the rest of their analysis.

There is no single sequence length that is guaranteed to be optimal across all reservoirs. This is because different reservoirs use different operating rules that use various timescales, and reservoirs with different main uses may rely on different parts of the recent attributes. For these reasons, the sequence length experiment is used to evaluate how sensitive model performance is to the chosen input window, and whether shorter windows can still achieve good performance.

2.7 Evaluation metrics used in Hydrology

Multiple complementary metrics were used for model evaluation in this thesis. This is because no single metric can capture all aspects of predictive performance. Different metrics reflect different aspects of performance, such as average error magnitude, sensitivity to large errors, and ability to reproduce the pattern of the time series. In hydrological modelling, it is common to assess the model performance using error-based and efficiency-based metrics.

Error-based metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are common in machine learning applications, and measure differences between predicted and observed values. These metrics are useful because they are directly related to prediction error, but they do not fully describe whether the model reproduces the temporal dynamics of the observed time series.

For this reason, many hydrological studies often also use efficiency metrics such as the Nash–Sutcliffe Efficiency (NSE) and the Kling–Gupta Efficiency (KGE). NSE compares the squared prediction error of the model with the variability of the observed time series around its mean (Nash & Sutcliffe, 1970).

KGE was introduced as an alternative performance metric which separates performance into several components, correlation, variability, and bias (Gupta et al., 2009). This makes it useful for finding whether the performance of the model is due to reproduction of the magnitude of reservoir release, timing, or variability. In this thesis, KGE and NSE are therefore used alongside MAE and RMSE to evaluate both average prediction error and hydrological time-series behaviour. These metrics are further described in subsection 4.7.

3 Data

The data for this thesis came from three databases. We first use ResOpsUS, a dataset of historical reservoir operations in the contiguous United States (J. Steyaert et al., 2021, 2022). Second, an additional confidential dataset provides reservoir operation records for a separate set of dams in China. Finally, GRanD (Global Dam Watch, 2019; Lehner et al., 2011) is used as a lookup dataset providing static reservoir and dam attributes, which are later used as static attributes and metadata.

3.1 Dataset summary

Table 2 summarises the two cleaned dataset versions used in this thesis. The *Main* dataset corresponds to the ResOpsUS-only version, while the *Extra* dataset adds the Chinese reservoir records.

The split was designed to evaluate two forms of generalisation. The temporal in-sample split tests whether models can predict later periods for reservoirs represented during training, while the held-out reservoir split tests whether models can transfer to reservoirs not seen during training.

The main dataset contains 280 reservoirs before splitting, whereas the extra dataset version contains 323 reservoirs after adding the Chinese records. The training sets dominate the sample count because they cover the longest time period, ending in 2014. The validation set is restricted to 2015, and the in-sample test set covers 2016–2020. The held-out test set contains 20 randomly selected reservoirs in both dataset versions, but the number of samples differs because the selected reservoirs and their record lengths are different. The number of reservoirs also changes between the in-sample splits because not all reservoirs have observations available for every time period.

In the following tables, *in train*, *in val*, and *in test* refer to the temporal training, validation, and test splits for reservoirs included in the training set. *Out test* refers to the fully held-out reservoir test set, containing reservoirs excluded entirely from model training and selection.

Figure 4 shows the spatial distribution of the reservoirs used in the two dataset versions. The main dataset contains only the ResOpsUS reservoirs, therefore all its reservoirs are located in the continental United States. The extra dataset adds 43 Chinese reservoirs that are not present in the main version; these reservoirs are plotted separately to make the spatial extension of the dataset explicit. In both panels, red outlines on the points indicate reservoirs selected for the held-out test set, and the broad Köppen–Geiger climate groups are shown as an overlay on the background. This shows that,

although it was not strictly enforced, the randomly selected reservoirs are well spread out spatially.

Figure 5 compares the composition of the in-sample and held-out reservoirs for both dataset versions. In the main dataset, the in-sample and held-out sets are both dominated by flood-control and irrigation reservoirs. In contrast, the held-out set contains more navigation, recreation, and unknown-use samples, because the held-out set was required to include at least one reservoir from each main-use category, as discussed in subsection 4.2. The climate distribution is also broadly similar, but the held-out set has a larger arid share and a slightly smaller temperate share.

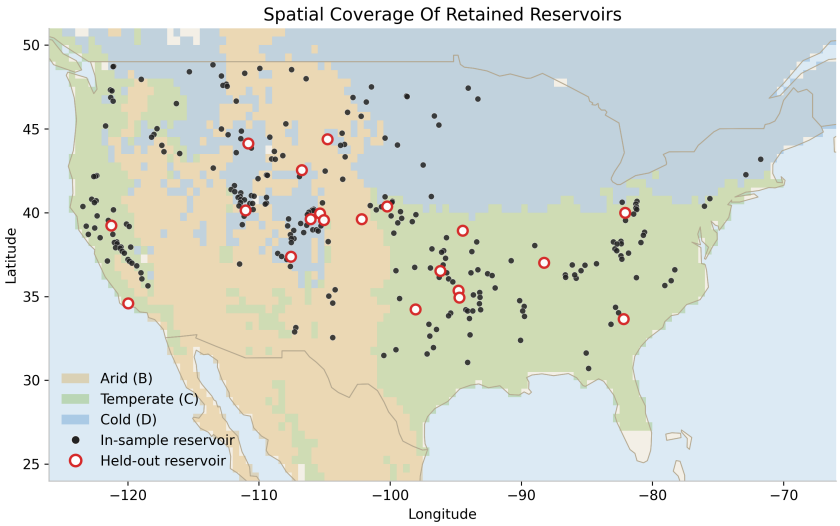
The extra dataset version changes the composition of the held-out set more noticeably. The in-sample portion remains dominated by flood-control and irrigation reservoirs, but to a lesser extent, while the held-out set contains a larger share of unknown-use, recreation, and water-supply samples. The climate composition also changes: the held-out set in the extra dataset version has a much larger cold-climate share and a smaller arid share than the in-sample data.

Table 2: Sample and reservoir counts for the data splits used in training and evaluation. The in-sample splits share reservoirs and are separated temporally, while `out_test` contains reservoirs held out entirely from training.

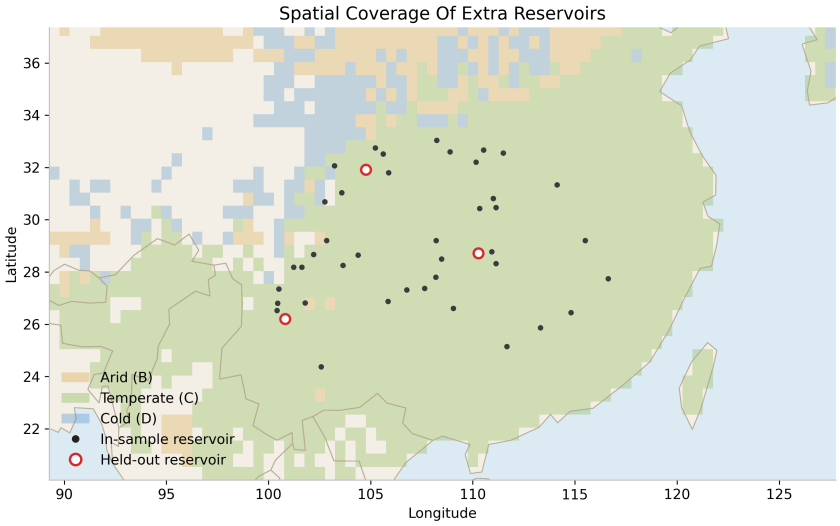
Dataset type	Split	Samples	Reservoirs	Start year	End year
Main	`in_train`	3,241,296	260	1900	2014
Main	`in_val`	81,153	240	2015	2015
Main	`in_test`	297,990	217	2016	2020
Main	`out_test`	308,633	20	1941	2020
Extra	`in_train`	3,333,671	263	1900	2014
Extra	`in_val`	81,814	243	2015	2015
Extra	`in_test`	337,597	253	2016	2020
Extra	`out_test`	217,050	20	1941	2020

3.2 Cleaning and preparation

Dams are assigned the same indices in both the ResOpsUS and GRanD datasets, making matching those two datasets trivial. For the additional dataset, reservoir records were harmonised before merging with the other sources. Reservoir names were matched to GRanD using available reservoir, dam, and alternative names, with a small number of manual matches where automatic matching was insufficient. Storage values in this dataset were converted from billions of cubic metres to millions of cubic metres to ensure consistency with the GRanD capacity units.



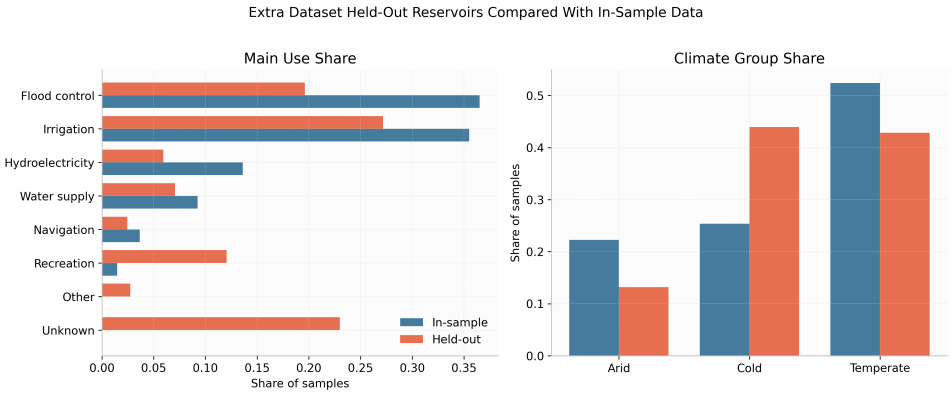
(a) Location of reservoirs in this thesis from ResOpsUS.



(b) Location of the additional reservoirs from the extra dataset.

Figure 4: Spatial coverage and broad Köppen–Geiger climate grouping of the reservoir datasets used in this thesis. The semi-transparent background overlay shows the broad climate group. Panel (a) shows the main cleaned ResOpsUS dataset. Panel (b) shows only the additional reservoirs introduced by the extra Chinese dataset. Held-out reservoirs are marked with red outlines in both panels.

(a) Composition of the held-out reservoirs compared with the in-sample reservoirs for the main ResOpsUS dataset.



(b) Composition of the held-out reservoirs compared with the in-sample reservoirs for the extra dataset.

Figure 5: Comparison between the held-out and in-sample reservoir sets used for model evaluation. Panel (a) shows the main dataset, while panel (b) shows the extra dataset containing both the main reservoirs and the additional Chinese reservoirs. The distributions are shown by primary reservoir use and climate group.

The Köppen-Geiger climate classification is a widely used global classification of climate zones. The locations of the dams were extracted from GRanD, and `kgcpy` version 1.1.8, which is based on the classification maps of Rubel et al. (2017), was used to translate these to Köppen-Geiger codes. This is then reduced to a broad, one-hot encoded climate group: Tropical, Arid, Temperate, Cold, Polar, or Other. The purpose of the addition of the climate group as a static attribute is because it provides information about the climatic setting of each reservoir. For example, cold-climate reservoirs may be influenced by snow accumulation and spring snowmelt, while arid reservoirs may be more affected by a high rate of evaporation and long periods with no inflow. These differences can affect how storage is maintained and the timing of releases. The reservoir main use from GRanD is also one-hot encoded.

Two cyclical inputs representing the position within the year were added to the dataset using sine and cosine transformations of the day of year:

$$\mathrm{doy}_{\sin} = \sin\left(\frac{2\pi t}{365}\right), \qquad \mathrm{doy}_{\cos} = \cos\left(\frac{2\pi t}{365}\right).$$

For each reservoir, the 80th percentile inflow was computed and used to scale the inflow and outflow. Reservoirs were retained only when both this inflow scaling factor and the GRanD storage capacity were available and positive. After this cleaning step, an effective storage capacity was selected for each reservoir. This was set to the GRanD capacity when it was consistent with the cleaned storage record, but when the observed storage range was clearly inconsistent with the reported capacity, the 99th percentile of the cleaned storage series was used instead. This was done to improve consistency between the reported static capacity and the storage time-series used for modelling. The storage series for each reservoir was then normalised by dividing by this effective capacity, so that storage was expressed as a fraction of the reservoir's usable maximum storage. As a result, a value close to 1 indicates storage near the effective maximum capacity, while values near 0 indicate relatively low storage.

Before being passed to the models, the static reservoir capacity input was normalised separately from the storage series. For each reservoir, the effective maximum capacity in million cubic metres was first transformed using

$$C_i' = \log(1 + C_i), \tag{11}$$

where C_i is the raw effective maximum capacity of reservoir i.

Although reservoirs with non-positive reported GRanD capacities were removed during preprocessing, $\log(C_i)$ can still produce large negative values when the effective capacity C_i is small. Therefore, one was added to the

capacity during the logarithmic transformation to reduce this sensitivity. This has little effect for large reservoirs, where $\log(1 + C_i) \approx \log(C_i)$.

The transformed capacity was then standardised using the mean and standard deviation estimated from the training metadata only,

$$\tilde{C}_i = \frac{C'_i - \mu_C}{\sigma_C}, \tag{12}$$

where μ_C and σ_C are the mean and standard deviation of $\log(1+C)$ over the training reservoirs. The same fitted scaling parameters were then applied to the validation and test reservoirs. This was done to reduce the large differences in scale between reservoirs caused by the raw storage capacity, while also avoiding information leakage from the validation or test sets.

Table 3 summarises the variables retained after cleaning and feature construction. The dynamic inputs are available for each day within the 7-day window, while the static variables describe fixed reservoir characteristics used by the conditional models. The target variable is the outflow on the following day.

Table 3: Variables used to construct the datasets. Dynamic variables are available for each day in the 7-day input window, while static variables are fixed for each reservoir.

Variable	Type	Source	Description
`inflow`	Dynamic input	ResOpsUS / extra dataset	Daily reservoir inflow, scaled by the reservoir-specific 80th percentile inflow before being passed to the model.
`storage`	Dynamic input	ResOpsUS / extra dataset	Daily reservoir storage, converted to million cubic metres where needed and then normalised by the effective reservoir capacity.
`outflow`	Target	ResOpsUS / extra dataset	Daily reservoir outflow. The prediction target is next-day outflow, scaled by the reservoir-specific 80th percentile inflow.
`doy_sin`, `doy_cos`	Dynamic input	Derived	Cyclical encoding of the day of year.
`CAP_MCM`	Static input	GRanD	Reservoir storage capacity in million cubic metres. When the GRanD capacity was inconsistent with the cleaned observed storage range, the 99th percentile of cleaned storage was used as an effective capacity.
`MAIN_USE`	Static categorical input	GRanD	Primary reservoir purpose, one-hot encoded.
`CLIMATE_GROUP`	Static categorical input	Derived from GRanD location	Broad climate group derived from the reservoir coordinates using the Köppen–Geiger classification, one-hot encoded.

4 Methods

This section details the methodology followed to obtain the results of this thesis. Firstly, the creation of samples from the datasets discussed in section 3 is presented in subsection 4.1, followed by the hold-out and splitting methodology in subsection 4.2. The rule-based baseline is described in subsection 4.3. The model inputs are then discussed in subsection 4.4, after which the model architectures and training strategy are presented in subsection 4.5. The model selection strategy and evaluation metrics are then described in subsection 4.6 and subsection 4.7 respectively. Finally, the lookback-window experiment and the sensitivity of the trained models to noise in selected inputs are described in subsection 4.8 and subsection 4.9.

4.1 Sample creation

The time series cleaned in section 3 are split into uninterrupted runs of length L. Each sample contained L consecutive input days and one target outflow value on the following day. Here, L is the lookback window for the models, meaning the number of previous timesteps available to the model to make one prediction. In the main model experiments, this was seven days. This was selected based on the results from Fan et al. (2023). The effect of this choice was later tested in the lookback-window experiment described in subsection 4.8, where models were trained and evaluated with seven-, three-, or one-day lookback windows. The days are reused such that, for longer continuous sequences, all possible valid runs are extracted, even if this results in overlap with previously used samples.

A storage spike filter is applied to the storage time series, to remove spikes in storage which might be created by measurement errors or documentation typos as identified by Li et al. (2024). This is done by identifying large outliers which are immediately followed by a reversal of the opposite sign and a similar magnitude. Days which are flagged are treated as unusable and removed from the dataset.

4.2 Dataset splitting

The training data is then split into in-sample and held-out reservoirs. This is done to keep a representative sample of reservoirs entirely separate from the data that the models are trained on. The held-out sample is then used as a measure of the generalisation performance of the final models on reservoirs not seen during training. Twenty reservoirs are randomly chosen to be fully held-out, and it is ensured that these include examples from each main use

category and climate group. When running with the extra dataset, it is also ensured that this held-out set includes reservoirs from both datasets.

The in-sample reservoirs are further divided into training, validation and testing, which are defined temporally. Samples from the beginning of the dataset through 2014 are used for training, samples from 2015 for validation, and samples from 2016 onward for testing. The training set was used to fit model parameters, the validation set was used for model selection and early stopping, and the test set provided an estimate of performance on unseen time periods. This also reduces temporal leakage when compared with a random split, as the latter would mix nearby days and provide an inflated result when compared with reality.

4.3 Rule-based baseline

The Hanasaki et al. (2006) simplified reservoir operation scheme, also presented in Han et al. (2020), was used as a baseline to compare the models to. This was originally formulated with separate formulas for the irrigation and non-irrigation main-use reservoirs. However, the irrigation formula requires downstream water demand data, which were not available in this thesis. For this reason, only the non-irrigation part was used, which was therefore applied to hydropower and flood control reservoirs. The Hanasaki et al. (2006) scheme was also originally developed using a monthly time increment. Its application to daily release prediction in this thesis should therefore be interpreted as a simplified adaptation.

The release is computed from the mean inflow, the current inflow, and the initial storage on the first day of the hydrological season, in this case the first of May. Following the notation in Han et al. (2020), the capacity-inflow coefficient c and the release coefficient k are defined as

$$c = \frac{S_{\max}}{Q_{\text{in,avg}}}, \qquad k = \frac{S_{\text{first}}}{0.85 S_{\max}}, \tag{13}$$

where $S_{\max}$ is the reservoir storage capacity, $Q_{\text{in,avg}}$ is the long-term average inflow, and S_{first} is the storage at the beginning of the hydrological season. The release is then given by

$$Q_{\text{out}} = \begin{cases} k Q_{\text{in,avg}}, & c \geq 0.5, \\ \left(\frac{c}{0.5}\right)^2 k Q_{\text{in,avg}} + \left[1 - \left(\frac{c}{0.5}\right)^2\right] Q_{\text{in}}, & c < 0.5. \end{cases} \tag{14}$$

Here, Q_{out} is the simulated release and Q_{in} is the inflow at the current timestep.

This baseline was used as a simple point of comparison, rather than as a complete representation of the state-of-the-art in rule-based reservoir operation modelling. Compared with the machine learning models implemented in this thesis, it uses substantially less input information and has only a small number of fixed parameters. The HNIRS baseline was therefore only used as a simple, interpretable rule-based benchmark to test whether the recurrent machine learning models could improve on a generic representation of reservoir operation.

4.4 Machine learning model inputs

There are two types of inputs used in this thesis, sequential and static. The non-conditional models use the sequential inputs only, while the conditional models use both sequential and static inputs.

The sequential inputs are given as a three-dimensional tensor with shape $(n, L, 4)$, where n is the number of samples and L is the lookback window length. The lookback window length was chosen as discussed in subsection 4.8. The four features in the final tensor dimension correspond to the sequential inputs, namely normalised inflow, normalised storage, and the sine and cosine representations of the day of year. The inflow and storage are the observed values, as reported by the dam operators, representing the water input into the reservoirs and the current water volume in the reservoir respectively.

The static inputs are the reservoir main-use, climate group and maximum capacity. The main-use and climate-group variables are one-hot encoded, while the maximum capacity is given as a normalised scalar value as discussed in subsection 3.2. These provide context to the machine learning model about attributes of the reservoir, with the aim of allowing a single model to be able to accurately model the outflows of many different types of reservoirs.

The non-conditional models were trained with the sequential inputs, with the target being the normalised outflow on day `t+1`.

For the conditional models, the sequential and static inputs are both given, and the target is also the normalised outflow on day `t+1`. In order to incorporate static reservoir attributes alongside time-series data, the approach taken by Tran et al. (2025) was used. This was implemented using the `ConditionalRecurrent` wrapper from the `cond_rnn` GitHub repository (Remy, 2020). The wrapper uses the static inputs to initialise the recurrent state of the network as discussed in subsection 2.3.

4.5 Model training

Both architectures consisted of a single recurrent layer followed by a dense output layer with one neuron, which produced the next-day normalised outflow prediction. Figure 6 shows one input sample passing through these architectures. Each sample contains L daily input vectors, where L is the lookback-window length. The recurrent layer is applied sequentially across these L timesteps, with the hidden state passed forward from one timestep to the next.

The number of recurrent units, denoted here by d_h, controls the length of the hidden state vector. This means the number of values that are stored in the model's hidden state at each timestep. After the final timestep, the hidden state is passed to a dense output layer with one neuron, which predicts the next-day normalised outflow. For the conditioned models only, the static reservoir attributes are first transformed and used to initialise the recurrent state of the model.

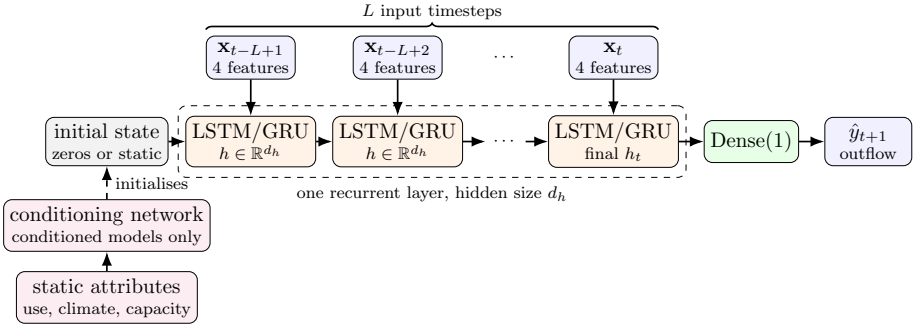


Figure 6: Architecture of the recurrent reservoir-release models. Each sample contains L daily input vectors, where each vector contains inflow, storage, and cyclical day-of-year variables. The LSTM or GRU layer is applied across these timesteps, passing the hidden state forward through the sequence. The hidden size d_h controls the width of the hidden state at each timestep. The final hidden state is passed to a dense layer to predict next-day normalised outflow.

All models were trained to minimise MAE on the normalised target variable. MAE is calculated as follows:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{15}$$

where y_i represents the observed values, $\hat{y}_i$ the predicted values, and n the number of observations. The Adam optimiser was used, with the learn-

ing rate treated as a model-selection hyperparameter. During training, the training split was shuffled and batched, while the validation split was batched without shuffling. The training pipeline used prefetching so that data loading and model computation could overlap where supported by the hardware.

Early stopping was applied during training by monitoring validation MAE after each epoch. For each model, training was terminated when validation loss failed to improve for the specified patience period, and the weights from the epoch with the lowest validation loss were restored. This follows the approach detailed in Prechelt (2002), with the aim of stopping the model when the neural network has had time to learn the problem, but before significant overfitting takes place. Early stopping can also improve training efficiency by avoiding unnecessary epochs once validation performance no longer improves.

Model predictions were first produced in the same normalised units as the training target. This is beneficial for neural networks because it places the model inputs on a comparable scale, leading to more stable optimisation and faster training (Lecun et al., 2002). The reservoir-specific scaling quantities used are stored, so that the metrics computed can later be denormalised to physical units, for interpretation and integration with hydrological models. MAE and RMSE were calculated over all samples in the evaluated split after denormalisation. KGE, its components, and NSE were calculated separately for each reservoir using the time-ordered predictions, and the median value across reservoirs was reported. This reservoir-wise aggregation was used so that reservoirs with many valid samples would not dominate the hydrological efficiency metrics. The model-selection script also saved the configuration, training history, trained model, aggregate metrics, and per-reservoir metrics for each run, allowing the same trained models to be reused for later robustness and feature-importance experiments.

4.6 Model selection

The validation set was used to compare candidate model architectures and hyperparameter settings through a model selection procedure. Each configuration was first trained once in an initial screening stage, and the completed runs were ranked according to their performance on the validation set, with the validation MAE as a metric. The five best performing models and hyperparameter combinations were then rerun with two additional random seeds in a finalist stage. This was done to make sure the results obtained were not due to random initialisation. The in-sample test set was used only for these finalist checks, while the held-out reservoir test set was excluded from

the model selection procedure and reserved for final evaluation.

The model classes evaluated were the standard LSTM, conditioned LSTM, standard GRU, and conditioned GRU. For each model class, the following hyperparameter grid was evaluated:

$$\texttt{units} \in \{64, 128\}, \qquad \texttt{dropout} \in \{0.00, 0.05\},$$
$$\texttt{learning rate} \in \{10^{-3}, 3 \times 10^{-4}\}$$

all trained with the screening seed 42.

Each model was trained for up to 300 epochs using the Adam optimiser. Training used early stopping with a patience of 30 epochs and a minimum improvement threshold of 10^{-4}. This resulted in 32 screening configurations.

The `units` hyperparameter controls the number of neurons in the recurrent hidden layer. A larger value gives the LSTM or GRU more capacity to store information from the previous input days, but also increases the number of trainable parameters and therefore the risk of overfitting.

The `dropout` hyperparameter applies regularisation within the recurrent layer by randomly removing a small fraction of the neuron connections during training.

The learning rate controls the step size used by the Adam optimiser. A higher value can make training faster but may be less stable and overshoot good solutions. A lower learning rate usually makes training slower but can theoretically reach a more precise solution.

The batch size controls how many samples are used for each gradient update. A batch size of 1024 was used throughout to make training efficient on the available hardware while keeping the optimisation stable for the large sample set.

4.7 Evaluation metrics

KGE, its components, and NSE were calculated separately for each reservoir, and the median value across reservoirs was reported. This reservoir-wise aggregation was used so that reservoirs with longer valid records did not dominate the hydrological efficiency metrics.

4.7.1 Mean Absolute Error (MAE)

MAE, defined in Equation 15, measures the average magnitude of the errors between predicted and observed values without considering their direction.

In the evaluation stage, MAE was calculated after denormalising predictions and observations to physical units.

MAE provides the error in the same units as the variable being measured, which helps with interpretability. MAE is also relatively insensitive to outliers compared to squared-error metrics, since it does not disproportionately penalise large errors. This can also be a disadvantage, as MAE treats all errors equally, meaning that large deviations are not emphasised. Models trained on MAE therefore tend to prefer overall performance more, with less sensitivity to outliers when compared to RMSE and other metrics.

4.7.2 Root Mean Squared Error (RMSE)

The Root Mean Squared Error (RMSE) quantifies the square root of the average squared differences between predicted and observed values:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{16}$$

RMSE penalises larger errors more strongly due to the square term, making it more sensitive to outliers. Good performance with this metric therefore means the outliers are being accurately predicted. This sensitivity can be useful in reservoir-release modelling, since large errors may correspond to poorly reproduced high-release events or operational extremes. However, RMSE can therefore be strongly influenced by a small number of large errors, potentially giving a misleading impression of model performance. RMSE is therefore a useful complementary metric to MAE.

4.7.3 Nash–Sutcliffe Efficiency (NSE)

According to Nash and Sutcliffe (1970), NSE is defined as:

$$\text{NSE} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{17}$$

where y_i represents the observed values, $\hat{y}_i$ the predicted values, $\bar{y}$ the mean of the observed values, and n the number of observations. For each evaluated split, NSE was first calculated independently for each reservoir, and the median reservoir NSE was then used as the reported summary value.

An NSE value of 1 indicates a perfect match between predictions and observations, while 0 means that the model could have performed equally well

using the mean value as the prediction. Negative NSE values indicate that the model performs worse than this.

Because NSE is based on squared errors, it is sensitive to large errors and outliers. This means that a small number of badly predicted events can strongly reduce the NSE value. For this reason, NSE was interpreted together with MAE, RMSE, and KGE rather than used as the only performance measure. For the same reason, we also calculated the NSE per reservoir and reported the median value.

4.7.4 Kling–Gupta Efficiency (KGE)

Kling-Gupta Efficiency (KGE) (Gupta et al., 2009) is used as a goodness-of-fit indicator, and for comparison with other publications as it is commonly used in hydrological sciences. In this thesis, KGE and its components were calculated using the `KGE` objective function from the `hydroeval` Python package (Hallouin, 2021).

It is defined as:

$$\mathrm{KGE} = 1 - \sqrt{(r-1)^2 + (\beta-1)^2 + (\alpha-1)^2} \tag{18}$$

where r is the Pearson correlation coefficient between observed and simulated outflow, $\beta = \frac{\mu_{\hat{y}}}{\mu_y}$ is the bias ratio, and $\alpha = \frac{\sigma_{\hat{y}}}{\sigma_y}$ is the variability ratio. In these expressions, μ_y and $\mu_{\hat{y}}$ are the means of the observed and predicted outflows respectively, while σ_y and $\sigma_{\hat{y}}$ are their standard deviations. KGE values closer to 1 indicate good performance, with 1 being a perfect model. For each evaluated split, KGE and its components were calculated independently for each reservoir, and their median reservoir values were then used as the reported summary value.

KGE complements NSE by separating performance into interpretable components, making it easier to identify whether poor performance is due to timing issues, too much or too little variability, or a consistent bias where the model is systematically predicting too high or too low releases. However, similar KGE values can arise from different combinations of component errors, meaning that models with different behaviours may appear equally good (Knoben et al., 2025). Therefore, KGE was interpreted together with the other metrics to reduce this effect.

4.8 Lookback window testing

As discussed in subsection 2.6, the number of days of inputs given to the model is variable and can be adjusted to suit the needs of the problem. To

evaluate the sensitivity of the models to this choice, the dataset creation procedure was repeated using three different lookback windows:

$$L \in \{1, 3, 7\}.$$

For each value of L, the sequential input tensor was trimmed so that each sample contained L consecutive input days and one target outflow value on the following day. The target was therefore kept unchanged as the next-day normalised outflow, while only the number of previous input days available to the model was varied. The static inputs used by the conditioned models, namely main use, climate group, and maximum capacity, were not changed between lookback-window experiments.

The same temporal and reservoir-based splits described in subsection 4.2 were used for all lookback-window settings. This ensured that differences in performance were due to the input sequence length rather than to a different train-validation-test split.

For the lookback window length experiment, models were trained across the same grid as the full model selection experiment. For each lookback-window length, recurrent models were trained across the tested architecture and hyperparameter combinations. They were trained using the same training objective, optimiser, and evaluation metrics described in subsection 4.5 and subsection 4.6. The only difference was in the early-stopping patience, which was reduced to fifteen epochs to improve computational efficiency. This setting was applied consistently across all lookback-window lengths, so the comparison between sequence lengths remained internally consistent. The resulting models were then compared using validation and test-set MAE, RMSE, KGE, KGE components, and NSE.

4.9 Robustness testing implementation

Robustness to input uncertainty was evaluated by adding Gaussian noise to the inputs of already trained models and measuring the resulting change in performance. The approach used was adapted from the robustness test in Kratzert et al. (2019). The trained model weights were kept fixed during this test. This is so that the experiment measures the sensitivity of the fitted models to input perturbations, rather than their ability to adapt through retraining. The validation set was used for this task. The robustness testing was performed for the top performing hyperparameter configurations from the model selection, for each of the model architectures.

For an input variable x, the perturbed input was defined as

$$\tilde{x} = x + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2), \tag{19}$$

where σ is the standard deviation of the added noise. The tested noise levels were

$$\sigma \in \{0.1, 0.2, \ldots, 1.0\}.$$

Since the perturbed variables were already normalised before being passed to the models, the noise is added in an equal way regardless of the reservoirs.

For the dynamic inputs, noise was applied to normalised inflow and normalised storage, both separately and together. With the conditional models, noise was also applied to the normalised capacity. Additionally, all three inputs were perturbed at the same time, to provide an overall measure of input robustness.

Noise was not added to the seasonality variables or to the one-hot encoded static variables, such as main use and climate group. This was because Gaussian noise would break the encoding of these variables in a way which does not make sense. The robustness test should therefore be interpreted as a sensitivity analysis for the inputs to which noise was added.

For each noise level, the perturbation was repeated 50 times, and model performance was recalculated. The change in performance in MAE, RMSE, KGE, KGE components, and NSE, relative to the baseline with no additional noise was then used as a measure of sensitivity. The repeated perturbation results were then aggregated by input group and noise level. For each combination, the mean and standard deviation of the performance change across repeats were calculated. This gives information about both the average reduction in model performance and the variability caused by different noise initialisation.

5 Results

5.1 Model selection

The 32 candidate models were ranked using normalised validation MAE, while the in-sample test set was used only for the finalist runs. The full distribution of screening and finalist model-selection runs is shown in Figure A5, with the selected models highlighted. Below, in Table 4, the top five finalist runs are presented for the main and extra dataset model-selection runs.

Model selection first tested the performance of the trained models on the in-sample validation and test sets. These correspond to the task of predicting releases in additional time periods for the reservoirs already present in the training set. For these sets, the conditioned models dominated the validation ranking. In both the main and extra datasets, the best configuration was a conditioned LSTM with 128 units, no dropout, a learning rate of 10^{-3}.

The model-selection results show that conditioning recurrent models with static reservoir attributes was useful for predicting later time periods for reservoirs already represented during training. Across both the main and extra datasets, the strongest configurations were dominated by conditioned recurrent models, with conditioned LSTMs occupying nearly all the highest-ranked positions. This pattern suggests that the additional static inputs improved the recurrent model performance when predicting the outflow.

The consistency of the best configuration is an interesting result. In both dataset versions, the conditioned LSTM with 128 recurrent units, no dropout, and a learning rate of 10^{-3} ranked first during screening and remained the best model after the finalist reruns. For the main dataset, this model obtained the lowest finalist validation MAE and the highest median validation KGE, and it also performed best on the in-sample test period. The same configuration was again strongest for the extra dataset, where it achieved the best validation and in-sample test performance among the finalist models. The agreement between the two dataset versions indicates that the selected configuration was not only favoured by one particular dataset composition, although it is important to keep in mind that the two datasets have significant overlap.

The finalist runs also support the stability of the screening model-selection ranking. The top conditioned LSTM remained the best performing model when retrained with additional random seeds, suggesting that its performance was not due to favourable initialisation. Within the tested hyperparameter range, larger recurrent layers were generally preferred, while adding

dropout did not improve the best-performing models. The learning rate of 10^{-3} was also favoured for the leading configuration, although the lower learning rate of 3×10^{-4} remained competitive for some 128-unit conditioned LSTM runs. Overall, the model-selection experiment points to the conditioned LSTM with 128 units, no dropout, and a learning rate of 10^{-3} as the strongest configuration for in-sample temporal prediction. This result supports the usefulness of static conditioning during model selection, while the separate held-out reservoir evaluation is still needed to assess whether this advantage transfers to reservoirs not seen during training.

To further inspect the selected models, reservoir-wise `in_test` KGE values were grouped by climate group and main reservoir use in Figure 7. Across climate groups, arid and cold reservoirs had the highest typical performance in both dataset versions, with group medians above the overall medians. Temperate reservoirs showed lower and more variable performance, especially in the extra dataset. All climate groups still contained some poorly predicted reservoirs, as shown by the long lower tails and off-scale markers.

The main-use comparison shows clearer differences between reservoir categories. Irrigation and water-supply reservoirs had consistently high group medians in both dataset versions. The performance on flood-control reservoirs was slightly lower, with lower medians. Hydroelectric reservoirs performed well in the main dataset but showed weaker and more variable performance in the extra dataset. Navigation, recreation, and unknown-use groups should be interpreted more cautiously because these categories contain fewer reservoirs. Overall, the grouped KGE distributions show that the high overall `in_test` performance hides differences between reservoir types, especially between irrigation and water-supply reservoirs and flood-control or temperate reservoirs.

5.2 Rule-based baseline

The Hanasaki non-irrigation rule-based baseline was applied and evaluated on hydroelectricity and flood control main use reservoirs. It showed mixed performance across both dataset versions, with its relative performance depending strongly on the metric considered (Table 5). Median KGE and NSE values were negative for all splits, indicating that the rule-based baseline did not reproduce reservoir-level release dynamics well. However, the KGE component breakdown gives more detail on the source of this poor performance. For KGE, the ideal value of each component is one: $r = 1$ indicates perfect temporal correlation, $\alpha = 1$ indicates the correct variability, and $\beta = 1$ indicates the correct mean bias. The results show that the main source of error is not primarily the bias term, since β is generally closer to one than the other

Table 4: Grouped model-selection results for the main and extra model selection runs. The screening section shows the seed 42 screening result for each selected configuration. The finalist section summarises the mean of the seed 43 and 44 reruns of those same configurations. KGE and NSE cells are shown as mean/median across reservoirs; for finalist rows, each value is first calculated per seed and then averaged across finalist seeds.

Dataset	Rank	Configuration	Stage	Val. MAE_{norm}	Val. $RMSE_{norm}$	Val. KGE_{med}	Val. NSE_{med}	`in_test` MAE_{norm}	`in_test` $RMSE_{norm}$	`in_test` KGE_{med}	`in_test` NSE_{med}
Screening results											
Main	1	LSTM-cond 128, d=0, lr=10^{-3}	Screen	0.86	23.61	0.73	0.67	–	–	–	–
Main	2	LSTM-cond 64, d=0, lr=10^{-3}	Screen	0.86	23.88	0.71	0.66	–	–	–	–
Main	3	LSTM-cond 128, d=0, lr=3×10^{-4}	Screen	0.93	24.34	0.64	0.53	–	–	–	–
Main	4	GRU-cond 128, d=0.05, lr=10^{-3}	Screen	0.99	24.50	0.58	0.51	–	–	–	–
Main	5	LSTM-cond 128, d=0.05, lr=10^{-3}	Screen	1.06	25.33	0.50	0.42	–	–	–	–
Extra	1	LSTM-cond 128, d=0, lr=10^{-3}	Screen	0.88	23.96	0.68	0.62	–	–	–	–
Extra	2	LSTM-cond 64, d=0, lr=10^{-3}	Screen	0.90	23.74	0.61	0.54	–	–	–	–
Extra	3	LSTM-cond 128, d=0, lr=3×10^{-4}	Screen	0.98	24.75	0.54	0.46	–	–	–	–
Extra	4	LSTM-cond 128, d=0.05, lr=10^{-3}	Screen	1.03	26.41	0.67	0.60	–	–	–	–
Extra	5	LSTM-cond 128, d=0.05, lr=3×10^{-4}	Screen	1.14	26.06	0.50	0.44	–	–	–	–
Finalist rerun summaries											
Main	1	LSTM-cond 128, d=0, lr=10^{-3}	Finalist	0.85	22.92	0.73	0.68	0.69	15.64	0.82	0.79
Main	2	LSTM-cond 64, d=0, lr=10^{-3}	Finalist	0.89	23.74	0.65	0.57	0.71	15.71	0.76	0.72
Main	3	LSTM-cond 128, d=0, lr=3×10^{-4}	Finalist	0.93	24.39	0.61	0.53	0.78	16.80	0.74	0.70
Main	4	GRU-cond 128, d=0.05, lr=10^{-3}	Finalist	1.57	28.51	0.43	0.24	1.05	20.70	0.53	0.47
Main	5	LSTM-cond 128, d=0.05, lr=10^{-3}	Finalist	1.01	25.57	0.54	0.47	0.82	17.59	0.66	0.64
Extra	1	LSTM-cond 128, d=0, lr=10^{-3}	Finalist	0.86	23.67	0.73	0.68	0.66	14.70	0.77	0.74
Extra	2	LSTM-cond 64, d=0, lr=10^{-3}	Finalist	1.03	24.38	0.50	0.41	0.82	16.08	0.60	0.54
Extra	3	LSTM-cond 128, d=0, lr=3×10^{-4}	Finalist	0.93	24.25	0.64	0.57	0.73	15.49	0.68	0.64
Extra	4	LSTM-cond 128, d=0.05, lr=10^{-3}	Finalist	1.05	26.38	0.67	0.61	0.76	17.21	0.72	0.67
Extra	5	LSTM-cond 128, d=0.05, lr=3×10^{-4}	Finalist	1.11	26.15	0.51	0.42	0.83	17.59	0.59	0.57

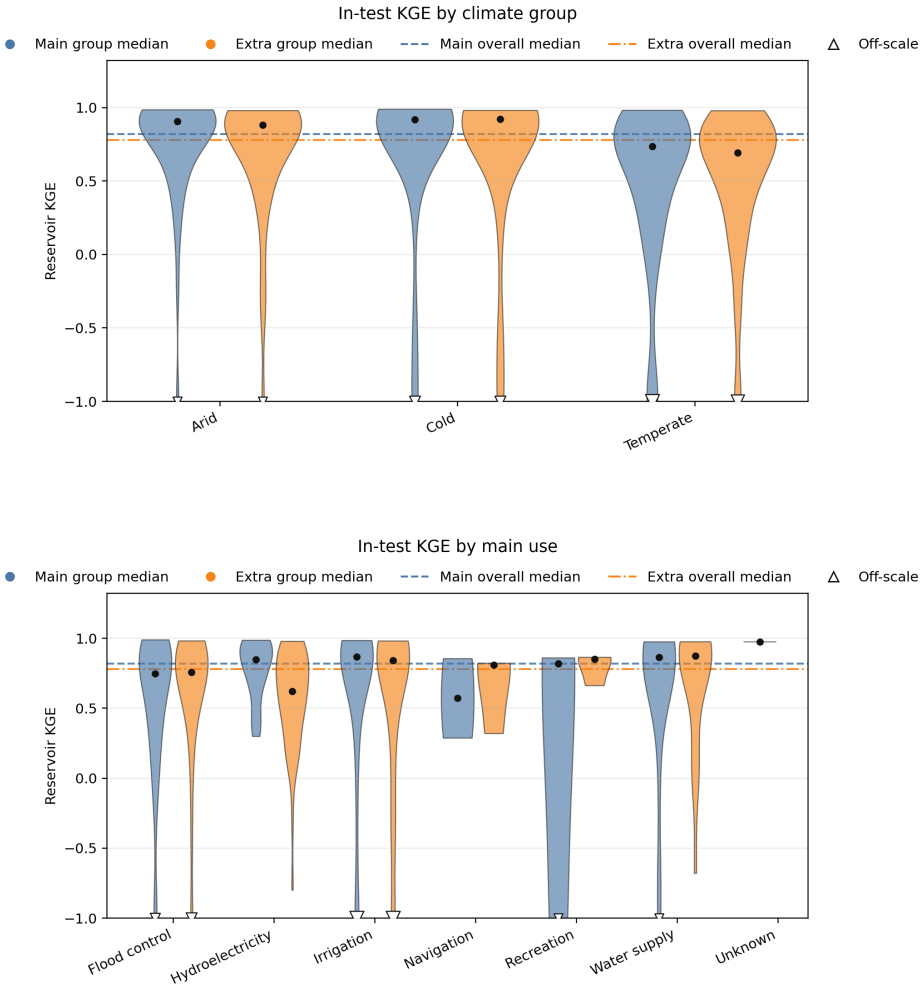


Figure 7: Reservoir-wise `in_test` KGE distributions for the main and extra datasets, grouped by climate group and main reservoir use. Filled points show group medians, while the dashed and dash-dotted horizontal lines show the overall median KGE values for the main and extra datasets respectively. Triangles indicate reservoirs with KGE values outside the plotted range.

components, especially for the `in_val` and `in_test` splits. Instead, the very low α values show that the baseline captures only a small fraction of the observed variability in releases. This suggests that the baseline predictions are too smooth and do not reproduce the magnitude of observed release fluctuations. The low r values also show that the timing of the predicted variations is poor, meaning that peaks and low-flow periods are often not placed correctly. This contrasts with the relatively good MAE and RMSE values obtained by the baseline, which further suggests that the baseline can approximate average release magnitudes reasonably well, but struggles with the magnitude and timing of fluctuations.

The similar values across the splits suggest that the rule-based baseline performance does not reduce when applied to new reservoirs. This is expected because the baseline is not fitted to the training reservoirs in the same way as the machine-learning models. Performance was similar between the main and extra datasets, although the extra dataset gave slightly improved median KGE and NSE on the `in_test` and `out_test` splits. The large RMSE values in the `in_val` and `in_test` splits, particularly compared with the lower `out_test` RMSE values, suggest that a small number of high-error samples strongly affected those metrics. Overall, the baseline provides a useful reference point, but its consistently negative reservoir-level efficiency scores show that a fixed non-irrigation baseline is too simple to capture the observed release behaviour across the reservoirs.

Table 5: Hanasaki non-irrigation rule-based baseline on the main and extra datasets. MAE and RMSE are sample-mean errors in normalised units. KGE, its components, and NSE are calculated per reservoir and shown as medians.

Run	Split	MAE_{norm}	RMSE_{norm}	KGE	r	α	β	NSE
Main	`in_val`	0.633	1.611	-0.400	0.177	0.049	0.708	-0.181
Main	`in_test`	0.654	2.954	-0.347	0.113	0.091	0.749	-0.085
Main	`out_test`	0.921	2.701	-0.403	0.068	0.078	0.369	-0.036
Extra	`in_val`	0.637	1.658	-0.398	0.229	0.049	0.707	-0.182
Extra	`in_test`	0.558	2.620	-0.260	0.219	0.157	0.782	-0.033
Extra	`out_test`	0.598	1.249	-0.311	0.288	0.078	0.669	-0.046

5.3 Held-out evaluation

The best screening-stage run from each architecture was also evaluated on the fully held-out `out_test` reservoirs. These were previously unseen by the model; therefore, this evaluates how well the models' performance generalises to other reservoirs that were not present in the training set. Figure 8 shows the reservoir-wise KGE distributions for the main and extra datasets,

while the corresponding summary metrics are shown in Table 6. Across both datasets, the highest median KGE values on the held-out test reservoirs were obtained by the unconditioned GRU and LSTM models. In the main dataset, the GRU and LSTM both reached a median `out_test` KGE of 0.543, while the conditioned LSTM reached 0.433 and the conditioned GRU only 0.080. A similar pattern was seen in the extra dataset, where the GRU obtained the highest median KGE of 0.604, followed by the LSTM with 0.569. The conditioned models again performed worse on the held-out reservoirs, with median KGE values of 0.382 for the conditioned GRU and 0.408 for the conditioned LSTM.

This is surprising, as it is a reversal of the trend seen during the model selection, where the conditioned models obtained the best results. However, this advantage did not carry over to the fully held-out reservoirs. This suggests that the conditioning information helped the model fit the in-sample reservoirs better, even on unseen time samples, but did not necessarily improve transfer to reservoirs outside the training set.

The error-based metrics show a similar but less consistent pattern. For the main dataset, the unconditioned GRU and LSTM both had a lower `out_test` MAE than the conditioned LSTM. The conditioned GRU performed particularly poorly, with a very large `out_test` MAE and RMSE, suggesting that this error came from one or a few held-out reservoirs which had very bad performance. For the extra dataset, GRU again had the lowest `out_test` MAE, while the conditioned LSTM and conditioned GRU were worse according to MAE. However, the conditioned models had lower RMSE than the unconditioned ones in the extra dataset, which suggests they performed better on the reservoirs which had large errors.

The reservoir-level behaviour behind these summary metrics is illustrated by two representative held-out simulations in Figure A6 and Figure A7. The first figure shows a high-performing held-out reservoir for which the LSTM reproduces the observed release pattern well. The second figure shows the lowest-KGE held-out reservoir, where the LSTM severely overestimates the observed variability and mean releases.

Overall, the held-out evaluation indicates that good validation performance on the in-sample reservoirs does not always translate to the best performance on unseen reservoirs. The conditioned LSTM remained the best model during validation, but the unconditioned GRU and LSTM generalised better to the held-out reservoirs in terms of median KGE.

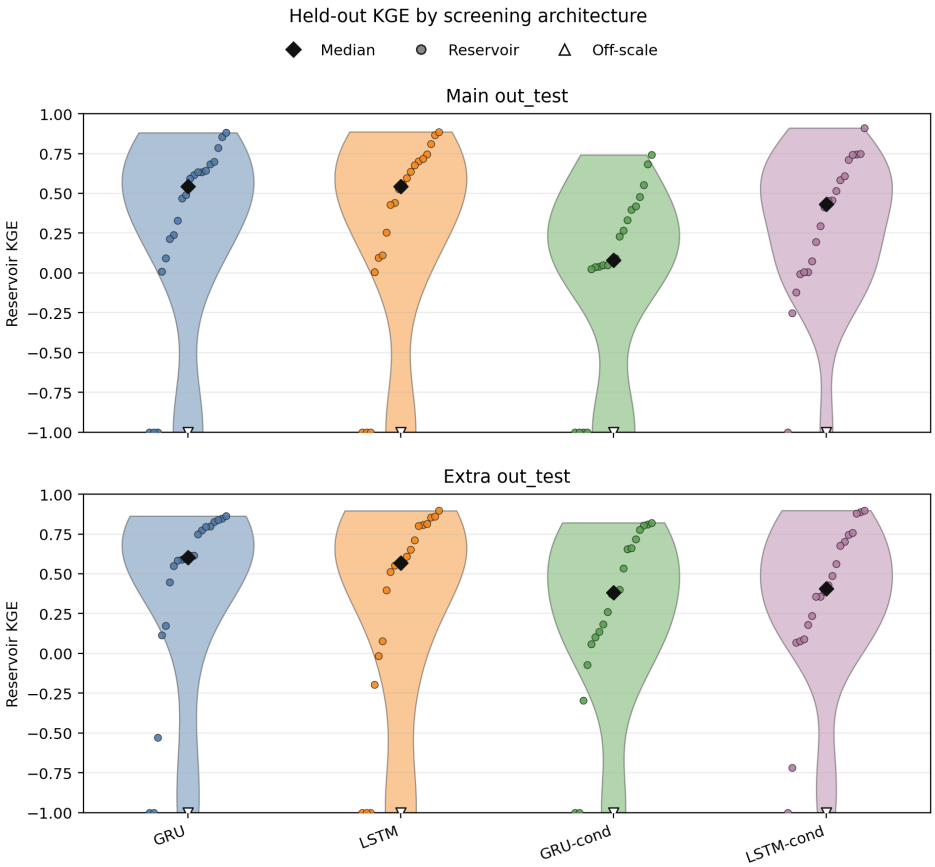


Figure 8: Reservoir-wise `out_test` KGE distributions for the best screening-stage GRU, LSTM, conditioned GRU, and conditioned LSTM runs. Results are shown separately for the main and extra datasets. Diamonds show median reservoir KGE, circles show individual reservoirs, and triangles indicate reservoirs with KGE values outside the plotted range.

5.4 Lookback window

The lookback window experiment results in Table 7 show that using seven input days gives the best overall performance in both datasets. In the main dataset, reducing the length of the window from seven to three days increased the validation normalised MAE from 0.872 to 1.000, which corresponds to a reduction in performance of 14.7%. When only one input day was used, the performance is much weaker, with the validation normalised MAE increasing by 122.4% when compared with the seven-day model. A very similar pattern is seen in KGE and NSE, however, in those metrics, the difference between 7 and the other two lookback lengths is larger when compared to the normalised MAE and RMSE.

The extra model selection run confirms the same trend, although the difference between the seven- and three-day windows is smaller. The normalised validation MAE increased from 0.908 with seven days to 0.963 with three days, corresponding to a 6.1% increase. However, the one-day model again performed worse, with a normalised validation MAE 114.3% higher than the seven-day model. Again, the difference in the MAE and RMSE performance for the 7- and 3-day lookback length is smaller than the difference in the KGE and NSE metrics.

The in-sample test results generally support the validation results. In both model selection runs, the seven-day models obtain the lowest test MAE and RMSE, as well as the highest median KGE and NSE. The three-day models remain usable but are consistently worse than the seven-day models, while the one-day models show a large loss of performance across all metrics. This suggests that one day of inflow, storage and seasonal information is insufficient for the recurrent models to represent reservoir operation behaviour effectively.

Overall, these results indicate that the models benefit from having access to several previous days of information. Based on these results, the seven-day lookback window was retained for the main modelling experiments.

5.5 Gaussian noise robustness

In the figures below, the results of the Gaussian robustness runs can be seen on the best performing screening stage run for each model architecture. The results for the unconditioned models and conditioned models for the main dataset are shown in Figure 9 and Figure 10 respectively. The results on the extra dataset can be found in Appendix A. As expected, in every case, KGE values decrease with increasing amounts of Gaussian noise.

Table 6: Held-out-reservoir evaluation of the best screening-stage run from each architecture family. Screening rows were selected by validation MAE_{norm} within each family. The `out_test` columns report evaluation on reservoirs excluded from training. KGE_{med} and NSE_{med} are reservoir-wise medians.

Dataset	Architecture	Configuration	Validation				`out_test`			
			MAE_{norm}	RMSE_{norm}	KGE_{med}	NSE_{med}	MAE_{norm}	RMSE_{norm}	KGE_{med}	NSE_{med}
Main	GRU	GRU 128, d=0, lr=10^{-3}	1.695	32.591	0.562	0.415	0.515	3.095	0.543	0.388
Main	LSTM	LSTM 128, d=0, lr=10^{-3}	1.641	31.899	0.575	0.376	0.515	4.384	0.543	0.326
Main	GRU-cond	GRU-cond 128, d=0.05, lr=10^{-3}	0.989	24.501	0.578	0.508	36.986	120.889	0.080	-0.357
Main	LSTM-cond	LSTM-cond 128, d=0, lr=10^{-3}	0.858	23.613	0.732	0.671	0.642	2.157	0.433	0.163
Extra	GRU	GRU 128, d=0, lr=10^{-3}	1.665	31.561	0.480	0.297	0.411	4.153	0.604	0.315
Extra	LSTM	LSTM 128, d=0, lr=10^{-3}	1.706	32.550	0.560	0.383	0.415	5.167	0.569	0.207
Extra	GRU-cond	GRU-cond 64, d=0.05, lr=10^{-3}	1.314	27.871	0.380	0.243	0.807	2.802	0.382	0.273
Extra	LSTM-cond	LSTM-cond 128, d=0, lr=10^{-3}	0.882	23.958	0.682	0.621	0.642	2.429	0.408	0.215

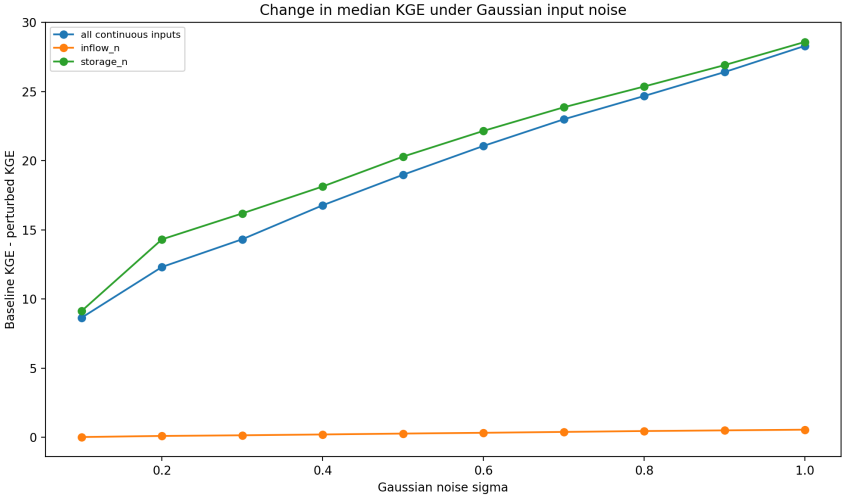
Table 7: Lookback-window ablation results for the main and extra model selection runs. Each row reports the best completed run for that lookback length, selected by validation MAE_{norm}. KGE_{med} and NSE_{med} are reservoir-wise medians. The validation MAE change is relative to the seven-day lookback within the same sweep.

Dataset	Days	Best configuration	Validation					`in_test`			
			MAE_{norm}	RMSE_{norm}	KGE_{med}	NSE_{med}	MAE change	MAE_{norm}	RMSE_{norm}	KGE_{med}	NSE_{med}
Main	7	LSTM-cond 128, d=0, lr=10^{-3}	0.872	24.048	0.689	0.584	+0.0%	0.700	15.694	0.776	0.750
Main	3	LSTM-cond 64, d=0, lr=10^{-3}	1.000	24.369	0.369	0.158	+14.7%	0.890	16.095	0.517	0.463
Main	1	GRU-cond 64, d=0.05, lr=3×10^{-4}	1.940	34.344	0.166	0.008	+122.4%	1.532	26.524	0.302	0.164
Extra	7	LSTM-cond 64, d=0, lr=10^{-3}	0.908	23.965	0.644	0.562	+0.0%	0.706	14.850	0.701	0.661
Extra	3	LSTM-cond 128, d=0, lr=3×10^{-4}	0.963	24.632	0.460	0.386	+6.1%	0.821	15.671	0.556	0.488
Extra	1	GRU-cond 64, d=0, lr=3×10^{-4}	1.946	34.200	0.143	0.049	+114.3%	1.389	24.915	0.315	0.213

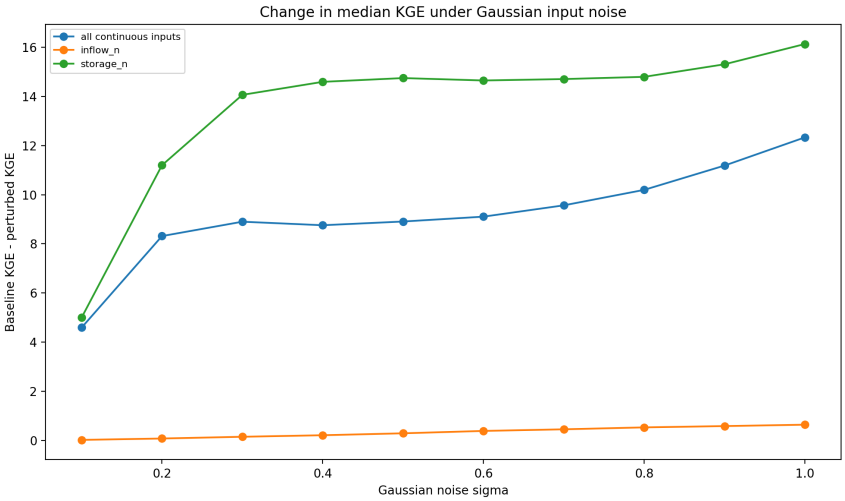
At the smallest perturbation level, 0.1σ, the effect of input noise differed between architectures and dataset versions. The unconditioned LSTM and conditioned GRU showed the smallest reductions in median KGE, while the conditioned LSTM showed a larger decrease even at 0.1σ. In the extra dataset, the difference between conditioned and unconditioned models was clearer. The unconditioned GRU and LSTM showed smaller KGE decreases at 0.1σ, while the conditioned GRU was more sensitive and the conditioned LSTM was again the most sensitive.

Storage created the largest drops in performance over the four plots, with inflow and capacity creating much smaller changes. The aggregate perturbations that included storage also caused large performance decreases, although, interestingly, the decrease was sometimes smaller than when noise was applied only to storage. This means adding noise to storage and to the other inputs simultaneously does not degrade performance as much as adding noise to storage on its own. This is not because the noise gets split between the inputs, as the combined measure is created by adding the same level of noise to each input.

The same overall behaviour was observed for the extra dataset, with the full robustness curves shown in Appendix A.

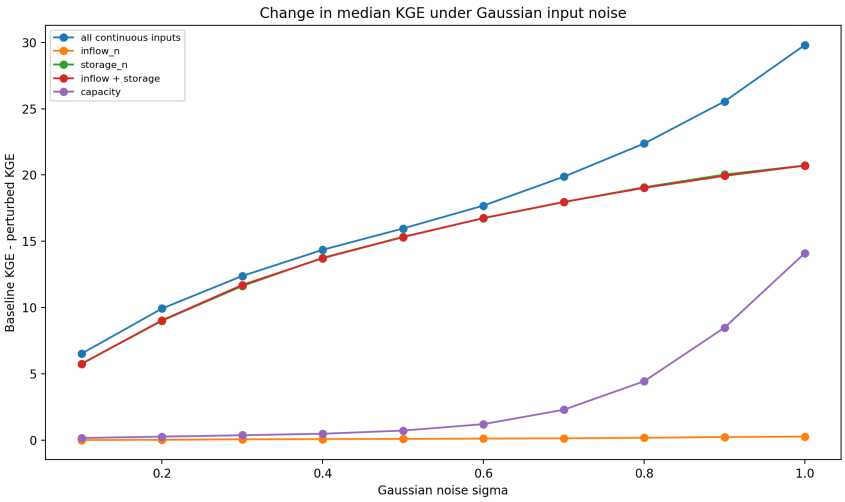


GRU

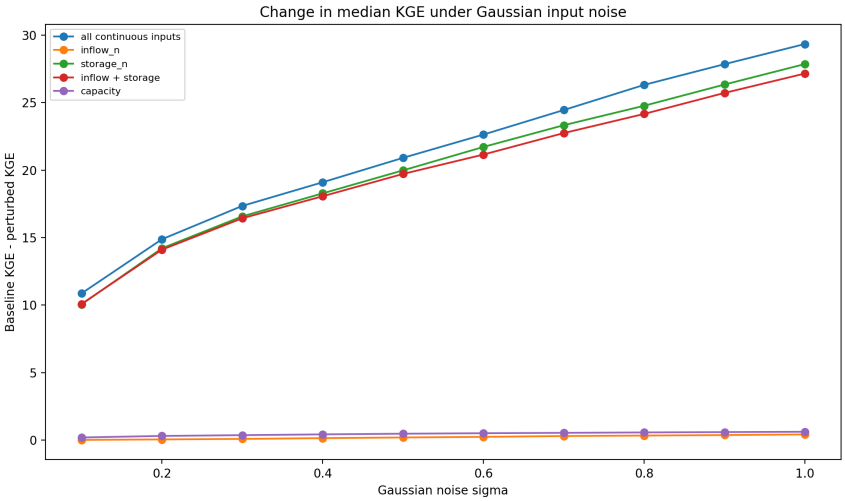


LSTM

Figure 9: KGE robustness curves for the best unconditioned GRU and LSTM screening-stage runs from the main dataset.



GRU-cond



LSTM-cond

Figure 10: KGE robustness curves for the best conditioned GRU and LSTM screening-stage runs from the main dataset.

6 Discussion

6.1 Recurrent models for reproducing historical reservoir outflows

The first part of research question one asked whether recurrent machine learning models are suitable for the task of modelling historical daily reservoir outflows. The results in this thesis show that LSTM and GRU models did learn useful relationships using the inputs provided to predict the next-day outflow. The strong validation and in-sample test results also show that the recurrent models generalise well to unseen time periods for reservoirs that are already present in the training set.

These results suggest that recurrent models are appropriate for this problem because reservoir releases are not only based on the conditions of a single day. Reservoir releases are likely to depend on multiple days of recent changes in inflow and storage. The recurrent structure of LSTMs and GRUs is therefore useful because it allows the model to use information from a sequence of previous days rather than treating each day independently.

This agrees with previous studies showing that recurrent architectures are suitable for reservoir-release prediction. Zhang et al. (2019) compared RNN, LSTM and GRU models for the Xiluodu reservoir and found that all three recurrent architectures could predict reservoir outflow effectively. Fan et al. (2023) also reported strong LSTM performance for reservoir releases in the Upper Colorado River Basin, with NSE values above 0.69 for the reservoirs considered. Compared with these studies, this thesis tests a more general task, since a single model is trained across many reservoirs and is also evaluated on held-out reservoirs as well as time periods. Therefore, the strong in-sample results are consistent with previous studies that use recurrent models, while the weaker held-out results show that generalisation to held-out reservoirs remains more difficult than temporal prediction for already seen reservoirs.

The model selection results suggest that model capacity affects performance within the hyperparameters tested. Large recurrent layers seem to have helped with modelling complex release behaviour, especially for reservoirs already present in the training data. At the same time, the large recurrent layers could be part of the reduction in performance on the held-out set, as the additional capacity could have allowed overfitting to happen. The result should therefore be interpreted as evidence that 128 units was preferable within the tested range, rather than as evidence for an optimal architecture more generally. However, the finding that model capacity affects performance is consistent with previous recurrent reservoir-operation

work. In Zhang et al. (2019), model settings such as the number of hidden nodes and training iterations were found to have the largest influence on model prediction.

The best performing model using no dropout may be due to the model selection using the validation set. Since this set did not contain any reservoirs unseen during training, the model selection favoured temporal generalisation. It is therefore possible that dropout could have been useful if the selection strategy focused more directly on unseen reservoir performance.

Overall, the results support the use of recurrent machine learning models for reproducing historical reservoir outflows, especially when the reservoirs being evaluated are represented in the training data. The models were able to reproduce both the magnitude and dynamics of daily releases better than would be expected from a purely static or single-day relationship. However, the held-out reservoir results show that good performance on unseen time periods does not automatically transfer to reservoirs that were not seen during training.

6.2 Effect of static conditioning

The conditioning of recurrent models with static reservoir attributes is a central part of this thesis. In the second part of research question one, this thesis asked whether this conditioning improved model performance and generalisation to unseen reservoirs.

The improved performance of the conditioned models on the in-sample test set suggests that main use, climate group, and capacity contain useful information about reservoir operating behaviour. This also suggests that the conditioning method used in this thesis allowed the models to use this information to some extent.

The conditioning, however, seemed to primarily improve the results on reservoirs already represented in the training set. On the fully held-out reservoirs, conditioned models performed worse than non-conditioned models. This indicates that, in this experiment, static conditioning improved temporal generalisation but reduced the models' ability to generalise to entirely unseen reservoirs.

A possible explanation for this behaviour is that the conditioned models may have relied more strongly on reservoir-specific patterns in the training set. The static attributes may also have been too coarse to describe the full range of operating rules across unseen reservoirs. For example, main use does not fully capture the multi-purpose nature of reservoir operation,

while climate group is only a broad descriptor of hydroclimatic setting. As a result, the conditioning variables may have helped the models distinguish between reservoirs already seen during training, but may not have provided enough information for transfer to entirely unseen reservoirs. By contrast, the non-conditioned models may have been forced to rely more on general relationships between the dynamic input variables and reservoir behaviour, which transferred better to the `out_test` reservoirs.

This result differs from Tran et al. (2025), who found that the conditioned LSTM improved generalisation to out-of-sample reservoirs. In their study, the conditioned LSTM achieved higher median KGE than the standard LSTM for several reservoir-use groups, including hydroelectricity, water supply, irrigation and recreation reservoirs, and they interpreted this as evidence that reservoir attributes improved transfer to unseen reservoirs. They also reported that LSTM performance dropped more strongly than conditioned LSTM performance when moving from in-sample to out-sample reservoirs.

There are several possible reasons for the difference between their results and those found in this thesis. The datasets, preprocessing and splitting are not identical between the two. Tran et al. (2025) used 143 in-sample reservoirs and 20 out-sample reservoirs, whereas this thesis used a larger number of reservoirs from ResOpsUS and, in the extra version, also included additional Chinese reservoirs. Differences in which reservoirs are held out can strongly affect the result, especially when only 20 reservoirs are used for the fully unseen test set. Second, the preprocessing and model-selection procedure differ, including the use of different cleaned storage capacities and additional architectures. This may have selected models that were particularly strong for in-sample temporal generalisation, but not necessarily for spatial transfer to new reservoirs. Regarding architectures, the models selected in this thesis contained many more hidden units than the ones selected in Tran et al. (2025).

Overall, conditioning did improve model performance, and generalisation to new time periods for seen reservoirs, but not the performance on unseen reservoirs. This suggests that the benefit of static conditioning is sensitive to the dataset, the reservoir split, and the architectures used.

6.3 Robustness to input noise

Reservoir operation datasets contain several sources of input uncertainty, such as measurement errors and errors due to derived measurements. It is therefore important to test how sensitive the trained models are to perturbations in the inputs, which is the focus of the first part of research question two.

This sensitivity can be interpreted as a measure of how errors in the inputs propagate through the model and affect the predictions, instead of a measure of feature importance. The fact that perturbing storage produced the strongest degradation in performance suggests that accurate storage information is especially important for historical outflow predictions. This result is also consistent hydrologically, since storage represents the current state of the reservoir and directly affects release decisions.

This also agrees with Fan et al. (2023), who found in their model evaluation that storage and inflow were more influential than precipitation and temperature for LSTM-based reservoir-release prediction. In fact, they also found from a shapley additive explanations analysis that inflow is the dominant factor for small reservoirs, while storage is dominant for larger ones. Since the robustness experiment conducted in this thesis is not separated by reservoir size, it cannot determine whether this effect also appears here.

One possible explanation for the combined perturbations involving storage sometimes performing better than storage-only perturbations is that it changed the model response in a different way. When several inputs were uncertain at the same time, the model might make more conservative predictions. This interpretation is speculative, but the result indicates that the model response to input uncertainty was not simply additive across variables.

The stronger response of the conditioned models to small perturbations suggests that static conditioning did not make the recurrent models more robust to input uncertainty. In particular, the conditioned LSTM appeared more sensitive than the other architectures, even at small perturbations. This suggests that the conditioned models, and especially the conditioned LSTM, relied more strongly on precise input values. This is important for operational use, because even small storage errors could reduce the performance of some architectures substantially.

The robustness experiment only tests the inputs to which Gaussian noise was added, and the results may also be influenced by correlations between variables. However, the consistently larger performance degradation under storage perturbations provides evidence that storage contains information that is particularly important for the fitted models.

The interpretation of the perturbation results is also complicated by the physical link between some variables. Inflow and storage are physically connected through the water balance equation of the reservoir, so information lost by perturbing one input may still be available through the other. Therefore, the individual perturbation results should be interpreted as the model response to noise in each input group separately, rather than as a measure

of each variable's full importance.

6.4 Effect of lookback-window length

The length of the input sequence given to the models was varied in an experiment in order to answer the second part of research question two. The results suggest that when given seven days, the models can learn to interpret the temporal dynamics of the reservoir outflow better, while three days is sufficient for low average absolute error. A lookback window of one day is clearly insufficient and strongly reduced performance across all metrics. This is to be expected, as one day provides no sequence information, which completely removes the value of a recurrent model. This result is consistent with Fan et al. (2023), who treated the LSTM lookback length as a hyperparameter and found that performance improved up to a seven-day window.

There is also a practical trade-off between performance and ease of use. Longer lookback windows require longer uninterrupted sequences of input data, which may be more difficult to obtain in operational applications or in datasets with missing values. Although the seven-day window gave the best overall performance, the three-day window still produced competitive MAE and RMSE values. Therefore, in settings where data continuity is limited or ease of application is important, a three-day lookback may be a reasonable compromise.

6.5 Comparison with the rule-based baseline

Research question three asked how the recurrent models compared with the Hanasaki non-irrigation rule-based baseline. The results show that the baseline had competitive MAE and RMSE, but very poor median KGE and NSE when compared to the machine learning models. This suggests that it is better at predicting average release magnitudes for extreme examples, but struggles with the dynamics of these releases. This is also supported by the low α values in the KGE components, which indicate that the baseline underestimates the variability of the observed releases. The underestimated variability may be partly due to the rule-based baseline being designed for monthly simulations, but applied here at a daily timescale.

It is also important to consider that model choice when implementing a hydrological model is not based only on predictive performance. Hydrological modellers might prefer the rule-based baseline because its behaviour is transparent and directly linked to established operating practice. Unlike machine

learning models, which have much lower interpretability, the baseline uses a simple and explainable relationship between hydrological inputs and release decisions. This can make it easier to justify decisions and diagnose unexpected behaviour.

Model performance depends heavily on the metric used. As discussed in subsection 4.7, each metric captures different aspects of performance. This explains why the rule-based baseline performs well in terms of MAE and RMSE, but not in terms of KGE and NSE. Overall, machine learning models reproduce dynamic release behaviour better, even when the baseline appears competitive in MAE and RMSE.

6.6 Influence of the extra dataset

The conclusions for the main and extra datasets were generally similar. For both datasets, in their training, validation and in-sample test sets, conditioned LSTM was the top performing model. On the held-out test set, however, the best performance for both datasets is achieved by the unconditioned GRU or LSTM models, while the conditioned models perform worse.

This consistency between the two datasets is encouraging, because it suggests the findings in this thesis are not specific to the dataset used. This suggests that the method described is not only applicable to the continental United States and may be successfully applied to other geographical locations. These results should, however, be interpreted with caution because there is substantial overlap between the datasets, since the extra dataset also includes a majority of ResOpsUS reservoirs. In addition, the reservoirs in the extra dataset may not contribute equally across all data splits, due to differences in record length.

6.7 Influence of adding GRU models

The addition of GRU models provided a useful comparison with the LSTM-based architectures used in the main model selection. Overall, the GRU models did not outperform the LSTM models in most parts of the analysis. The best model-selection results for both dataset versions were still obtained by conditioned LSTM models. This suggests that the additional gating structure and cell state of the LSTM were useful for reproducing historical reservoir outflows in the in-sample setting.

However, the GRU models remained competitive in several evaluations, despite using a simpler recurrent structure. This agrees with Zhang et al. (2019), who found that RNN, LSTM and GRU architectures could all be

used effectively for reservoir outflow prediction. The relatively strong performance of the GRU models is important because GRUs have fewer gates and fewer trainable parameters than LSTMs, which is an advantage due to the lower model complexity and computational cost. The results therefore suggest that GRUs should not be dismissed only because they were not usually the best-performing architecture, but can be used as an alternative when computational efficiency is needed. This could be especially relevant for large-scale hydrological modelling, where models may need to be trained or applied across many reservoirs.

6.8 Limitations and future directions

6.8.1 Data quality, reservoir attributes and held out set representativeness

A first limitation concerns the dataset. The reservoir operation datasets can contain measurement errors, inconsistencies between the inputs or inconsistencies in reporting between the reservoirs. For example, reporting practices may vary between reservoir operators, regions, or the two data sources, meaning that variables such as storage, inflow and outflow may not always be recorded or processed in a fully consistent way. Since the models are trained directly on these observed time series, such issues may affect both the learned relationships and the evaluation of model performance. Errors or inconsistencies in variables such as storage, inflow and outflow can also lead to situations where the reservoir mass balance equation is not respected. As a result, part of the model error may be caused by limitations in the underlying data rather than only limitations of the modelling approach.

Related to this, another limitation is that the storage capacity used for the static conditioning and for normalisation had to be adjusted in some cases. This was done when the storage time series clearly differed from the storage capacity reservoir attribute reported in the GRanD dataset. The correction was applied to reduce clear errors, but remains an approximation that is important to mention.

A further data-related limitation is that the main use reservoir attribute does not reflect the multiple objectives that are present in many reservoirs. This simplification means that reservoirs with multiple purposes are represented by a single category. This could therefore limit the model's ability to distinguish between reservoirs with the same primary use but different secondary objectives. Similarly, the climate group is broad and may not describe the local hydrological regime or management context in enough detail.

Another limitation is that the static attributes are treated as constant through time, even though they may change in practice. Attributes such as main use, storage capacity, and even the broader management context of a reservoir can change over long historical periods. This is especially relevant because the dataset is split temporally. If the static attributes describe the recent state of the reservoir better than its historical state, then they may be more representative of the validation and in-sample test periods than of the older training period. This could partly contribute to the stronger performance of conditioned models on the in-sample test set, since the conditioning information could describe the reservoirs as they appear in it better.

The held-out set also only contains twenty reservoirs. Although it was ensured that at least one of each main use and climate group was included, this sample might not be enough to fully capture the range of scenarios found in the datasets. As a result, performance on the held-out set should be interpreted with some caution, as it may not fully reflect how well the model generalises across the broader range of reservoir scenarios.

6.8.2 Methodological limitations

A second group of limitations concerns the methodology. Firstly, the models only predict the next-day outflow, rather than a full sequence of releases. This means that the models are trained on short-term predictions, but do not directly test whether the learned behaviour remains realistic over longer periods. In addition, release decisions in the previous days affect future storage conditions and, therefore, future releases. As a result, good next-day performance does not necessarily imply that the model would perform equally well when used recursively over several days, weeks, or seasons.

The reservoir mass balance equation is also not enforced. This can sometimes lead to outflow predictions that are not physically consistent with the available storage conditions. Since the models predict outflow directly and independently of an explicit storage update equation, they may occasionally produce releases that would imply negative storage or storage above capacity.

The rule-based model is also a relatively simple baseline and did not achieve very good results. It was useful as a benchmark to compare to, but using a more state-of-the-art model to compare to would lead to more interesting results, and a fairer comparison. However, this would have increased the implementation difficulty, possibly requiring additional information.

Another methodological limitation is that only one-, three- and seven-day lookback windows were investigated. These values were chosen to provide an

initial assessment of the effect of sequence length while keeping a manageable number of experiments. However, testing a wider range of lookback windows could have provided a more complete understanding of exactly how much historical information is needed.

Regarding the Gaussian noise robustness experiment, the noise was used as a simple approximation of input uncertainty. In practical settings, however, the distribution of noise is likely to be more complex. Measurement errors may not be normally distributed and might include systematic biases. Large Gaussian perturbations can also produce physically unrealistic values, such as negative normalised inflow or storage. Performance losses at high noise levels may therefore be partly due to the model receiving out-of-distribution inputs, rather than only sensitivity to realistic input uncertainty. For these reasons, the robustness experiment should not be interpreted as a real-world experiment of how the model behaves with measurement noise, but as a simplified sensitivity test.

6.8.3 Evaluation metrics and reservoir influence

A third limitation concerns the evaluation strategy. The use of multiple metrics was necessary to have a complete picture of the performance of the model; however, it is important to keep in mind their limitations. MAE and RMSE are useful for measuring average prediction errors, but they do not give any indication as to which element of prediction the errors come from. In addition, these metrics can be strongly influenced by a small number of large errors. KGE and NSE provide a more hydrologically relevant assessment of time series performance, but they also have limitations. KGE breaks down the errors into several components, which is useful for interpretation, but this can lead to equifinality as discussed in subsection 4.7.4, where different types of errors can lead to similar KGE values. Additionally, KGE and NSE are dimensionless scores rather than errors in physical units, which makes them harder to interpret. For these reasons, the results should be interpreted using all metrics together rather than relying on an individual one.

One limitation of the sample creation procedure is that all valid lookback windows were retained, including overlapping windows from the same reservoir time series. As a result, many samples differ by only one day in their input sequence and target day. These samples are therefore not fully independent and may be highly correlated.

This overlap does not introduce leakage into the different sets, since the temporal splitting strategy ensures the sets are almost fully independent, and the `out_test` reservoirs are fully excluded. However, the use of overlap-

ping windows does affect the effective independence of the training samples. In particular, reservoirs with longer observational records contribute a larger number of highly similar samples and may therefore have a very large influence on the learned model.

6.8.4 Future directions

These limitations also point to several directions for future research. First, future work could add other static reservoir attributes from GRanD, as additional inputs for the static conditioning. These could be parameters such as degree of regulation, which is the ratio between the storage capacity and the total annual flow. This gives information about how much the reservoir regulates the flow of water through it. Another useful parameter could be the area of upstream catchment draining into the reservoir. The GRanD dataset also includes information on secondary reservoir uses, in addition to the main use. These could be included as additional static inputs to reduce the simplification of representing each reservoir by only a single primary purpose.

Future work could also take into account the temporal stability of static attributes used for conditioning. Where available, future studies could use time-varying reservoir metadata, or compare model performance across different historical periods to test whether conditioned models perform better when the static attributes are more representative of the period being predicted.

Second, the outputs could be physically constrained to reduce unrealistic predictions. In this thesis, the machine learning models were only trained to minimise prediction error on observed outflow. In order for them to take into account physical constraints as well, the loss function could include a mass-balance-aware penalty term, where predicted releases are discouraged from producing storage changes that are inconsistent with the observed inflow, previous storage, and reservoir capacity. This would encourage the model to not only learn the outflow behaviour, but also the conservation of physical properties in the system.

However, as described by Tran et al. (2025), applying strict physical constraints is not always straightforward because the observed data itself may sometimes violate these conditions. They found that some reservoirs in the ResOpsUS dataset had days when releases exceeded the sum of inflow and storage. They suggested that this may be due to unrecorded operational behaviour, such as additional releases through leeways. Therefore, future work should carefully consider how physical constraints are imposed, since enforcing a strict mass-balance condition could remove behaviours that are

present in the observational dataset but not fully represented by the inflow, storage, and release variables.

Finally, the in-sample validation and test sets could be modified to contain unseen reservoirs. This would allow the models to train for performance not only on new time periods for seen reservoirs, but also on generalising to unseen reservoirs. However, it is important that the held-out set remains entirely separate, so that it can still be used as a final independent test of model performance. One possible approach would be to hold out additional reservoirs for the validation and test sets exclusively, while keeping an additional external held-out set that is not used during model selection. This would make it possible to tune the model using validation reservoirs that are unseen during training, without using the final held-out reservoirs during either training or hyperparameter selection. Such a splitting strategy would provide a stronger assessment of spatial generalisation and reduce the risk that the model is only learning reservoir-specific behaviour from the training set.

7 Conclusion

Overall, the results suggest that recurrent models are promising for reproducing historical daily reservoir outflows. Their strongest performance was obtained on the task of predicting outflow in later time periods for reservoirs that were present in the training set. Generalising to completely unseen reservoirs remains the main challenge.

The results show that LSTM and GRU models can learn useful relationships between recent inflow, storage, seasonality, and next-day outflow. The strongest performance on the validation and in-sample test sets was obtained by conditioned LSTM models, showing that static reservoir attributes can improve predictions on later time periods for reservoirs represented during training. However, this improvement did not transfer consistently to the held-out reservoirs, where the unconditioned recurrent models generally performed better. This means that static conditioning helped temporal generalisation more clearly than spatial generalisation to reservoirs not seen during training.

Model performance decreased when Gaussian noise was added to the inputs, with storage perturbations causing the largest performance losses. This indicates that the trained models rely strongly on storage information when predicting reservoir releases. Among the tested sequence lengths, the seven-day lookback window gave the best overall performance, while the one-day window was clearly insufficient.

The Hanasaki et al. (2006) non-irrigation rule-based baseline achieved competitive MAE and RMSE when compared with the machine learning models in some cases, but performed poorly in terms of KGE and NSE. This suggests that the baseline could approximate average release magnitudes but was much weaker at reproducing the timing and variability of daily outflows. Overall, the recurrent machine learning models predicted dynamic reservoir release behaviour better than the baseline.

GRU models were also evaluated as a simpler alternative to LSTMs, as they contain fewer trainable parameters. It was found that they were generally outperformed by LSTMs but remained a useful option when computational efficiency or model simplicity is prioritised.

The findings were broadly consistent across the main and extra datasets, supporting the robustness of the main conclusions.

The results should however be interpreted with some limitations in mind, including possible measurement errors and inconsistencies in the datasets, the use of simplified static attributes, the small held-out set of twenty reser-

voirs, the absence of mass-balance constraints, and the simplified nature of the Gaussian-noise robustness test.

Future work should focus on improving generalisation to unseen reservoirs. This could include using additional static attributes, such as degree of regulation, catchment area, and secondary reservoir uses, introducing physically constrained loss functions based on the mass-balance equation, using held-out reservoirs during validation, and comparing the models against stronger reservoir operation baselines.

8 Data availability

The reservoir operation data used in this thesis were obtained from three main sources. The ResOpsUS dataset, containing historical reservoir operation records for the contiguous United States, is publicly available as described by J. Steyaert et al. (2021) and J. Steyaert et al. (2022). The additional Chinese reservoir operation records used in the extra dataset were provided as a confidential dataset and are therefore not publicly available.

Static reservoir and dam attributes were obtained from the Global Reservoir and Dam Database (GRanD), available through Global Dam Watch and described by Lehner et al. (2011) and Global Dam Watch (2019). Broad climate groups were derived from the reservoir coordinates using the Köppen–Geiger climate classification through `kgcpy`, based on the maps of Rubel et al. (2017).

The source code developed for this thesis is publicly available on GitHub at: https://github.com/MatteoDebiere/reservoir-outflow-modelling

References

Chen, Y., Li, D., Zhao, Q., & Cai, X. (2022). Developing a generic data-driven reservoir operation model. *Advances in Water Resources*, *167*, 104274. https://doi.org/10.1016/j.advwatres.2022.104274

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In A. Moschitti, B. Pang & W. Daelemans (Eds.), *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1724–1734). Association for Computational Linguistics. https://doi.org/10.3115/v1/D14-1179

Ehsani, N., Fekete, B. M., Vörösmarty, C. J., & Tessler, Z. D. (2016). A neural network based general reservoir operation scheme. *Stochastic Environmental Research and Risk Assessment*, *30*(4), 1151–1166. https://doi.org/10.1007/s00477-015-1147-9

Fan, M., Zhang, L., Liu, S., Yang, T., & Lu, D. (2023). Investigation of hydrometeorological influences on reservoir releases using explainable machine learning methods. *Frontiers in Water*, *5*. https://doi.org/10.3389/frwa.2023.1112970

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. *Neural Computation*, *12*(10), 2451–2471. https://doi.org/10.1162/089976600300015015

Giuliani, M., Lamontagne, J., Reed, P., & Castelletti, A. (2021). A state-of-the-art review of optimal reservoir control for managing conflicting demands in a changing world. *Water Resources Research*, *57*. https://doi.org/10.1029/2021WR029927

Global Dam Watch. (2019). *Global reservoir and dam database (GRanD)* (Version 1.3) [Dataset access via the Global Dam Watch directory]. Retrieved March 1, 2026, from https://www.globaldamwatch.org/directory

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning* [http://www.deeplearningbook.org]. MIT Press.

Gupta, H. V., Kling, H., Yilmaz, K. K., & Martinez, G. F. (2009). Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling. *Journal of Hydrology*, *377*(1), 80–91. https://doi.org/10.1016/j.jhydrol.2009.08.003

Hallouin, T. (2021, April). *Hydroeval: An evaluator for streamflow time series in Python* (Version v0.1.0). Zenodo. https://doi.org/10.5281/zenodo.4709652

Han, Z., Long, D., Huang, Q., Li, X., Zhao, F., & Wang, J. (2020). Improving reservoir outflow estimation for ungauged basins using satellite observations and a hydrological model. *Water Resources Research*, *56*(9), e2020WR027590. https://doi.org/10.1029/2020WR027590

Hanasaki, N., Kanae, S., & Oki, T. (2006). A reservoir operation scheme for global river routing models. *Journal of Hydrology*, *327*(1), 22–41. https://doi.org/10.1016/j.jhydrol.2005.11.011

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

International Commission on Large Dams. (2025, September). *World register of dams / registre mondial des barrages* [Database, September 2025 version]. ICOLD/CIGB.

Knoben, W. J. M., Raman, A., Gründemann, G. J., Kumar, M., Pietroniro, A., Shen, C., Song, Y., Thébault, C., van Werkhoven, K., Wood, A. W., & Clark, M. P. (2025). Technical note: How many models do we need to simulate hydrologic processes across large geographical domains? *Hydrology and Earth System Sciences*, *29*(11), 2361–2375. https://doi.org/10.5194/hess-29-2361-2025

Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., Hochreiter, S., & Nearing, G. (2019). Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets. *Hydrology and Earth System Sciences*, *23*(12), 5089–5110. https://doi.org/10.5194/hess-23-5089-2019

Lecun, Y., Bottou, L., Orr, G., & Müller, K.-R. (2002). Efficient backprop. In *Neural networks: Tricks of the trade* (pp. 9–48). Springer.

Lehner, B., Liermann, C. R., Revenga, C., Vörösmarty, C., Fekete, B., Crouzet, P., Döll, P., Endejan, M., Frenken, K., Magome, J., Nilsson, C., Robertson, J. C., Rödel, R., Sindorf, N., & Wisser, D. (2011). High-resolution mapping of the world's reservoirs and dams for sustainable river-flow management. *Frontiers in Ecology and the Environment*, *9*(9), 494–502. https://doi.org/10.1890/100125

Li, D., Chen, Y., Lyu, L., & Cai, X. (2024). Uncovering historical reservoir operation rules and patterns: Insights from 452 large reservoirs in the contiguous United States. *Water Resources Research*, *60*(8), e2023WR036686. https://doi.org/10.1029/2023WR036686

Love, F. (2024). Nntikz: Tikz diagrams for deep learning and neural networks. https://github.com/fraserlove/nntikz

Nash, J., & Sutcliffe, J. (1970). River flow forecasting through conceptual models part I — a discussion of principles. *Journal of Hydrology*, *10*(3), 282–290. https://doi.org/10.1016/0022-1694(70)90255-6

Prechelt, L. (2002). Early stopping-but when? In *Neural networks: Tricks of the trade* (pp. 55–69). Springer.

Remy, P. (2020). Conditional RNN for Keras. https://github.com/philipperemy/cond_rnn

Rubel, F., Brugger, K., Haslinger, K., & Auer, I. (2017). The climate of the European Alps: Shift of very high resolution Köppen-Geiger climate

zones 1800–2100. *Meteorologische Zeitschrift*, *26*(2), 115–125. https://doi.org/10.1127/metz/2016/0816

Schmidt, R. M. (2019). Recurrent neural networks (RNNs): A gentle introduction and overview. https://arxiv.org/abs/1912.05911

Steyaert, J. C., & Condon, L. E. (2024). Synthesis of historical reservoir operations from 1980 to 2020 for the evaluation of reservoir representation in large-scale hydrologic models. *Hydrology and Earth System Sciences*, *28*(4), 1071–1088. https://doi.org/10.5194/hess-28-1071-2024

Steyaert, J., Condon, L., Turner, S., & Voisin, N. (2021). *ResOpsUS* (Version 2). Zenodo. https://doi.org/10.5281/zenodo.6612040

Steyaert, J., Condon, L., Turner, S., & Voisin, N. (2022). ResOpsUS, a dataset of historical reservoir operations in the contiguous United States. *Scientific Data*, *9*(1), 34. https://doi.org/10.1038/s41597-022-01134-7

Tran, H., Zhou, T., Tan, Z., Fang, Y., & Leung, L. R. (2025). Improving the prediction of daily reservoir releases over the CONUS using conditioned LSTM. *Journal of Hydrology*, *661*, 133750. https://doi.org/10.1016/j.jhydrol.2025.133750

Wada, Y., Bierkens, M. F. P., de Roo, A., Dirmeyer, P. A., Famiglietti, J. S., Hanasaki, N., Konar, M., Liu, J., Müller Schmied, H., Oki, T., Pokhrel, Y., Sivapalan, M., Troy, T. J., van Dijk, A. I. J. M., van Emmerik, T., Van Huijgevoort, M. H. J., Van Lanen, H. A. J., Vörösmarty, C. J., Wanders, N., & Wheater, H. (2017). Human–water interface in hydrological modelling: Current status and future directions. *Hydrology and Earth System Sciences*, *21*(8), 4169–4193. https://doi.org/10.5194/hess-21-4169-2017

Wang, J., Walter, B. A., Yao, F., Song, C., Ding, M., Maroof, A. S., Zhu, J., Fan, C., McAlister, J. M., Sikder, S., Sheng, Y., Allen, G. H., Crétaux, J.-F., & Wada, Y. (2022). Geodar: Georeferenced global dams and reservoirs dataset for bridging attributes and geolocations. *Earth System Science Data*, *14*(4), 1869–1899. https://doi.org/10.5194/essd-14-1869-2022

World Commission on Dams. (2000). *Dams and development: A new framework for decision-making*. Earthscan Publications Ltd.

Zhang, D., Peng, Q., Lin, J., Wang, D., Liu, X., & Zhuang, J. (2019). Simulating reservoir operation using a recurrent neural network algorithm. *Water*, *11*(4). https://doi.org/10.3390/w11040865

A Appendix

This appendix presents the remaining Gaussian noise robustness curves for the recurrent models trained on the extra dataset, a visual summary of the model selection, and representative held-out reservoir simulations. The first four figures complement the robustness results discussed in subsection 5.5, the fifth visualises the model-selection results discussed in subsection 4.6, and the final two show the time-series behaviour behind the held-out evaluation.

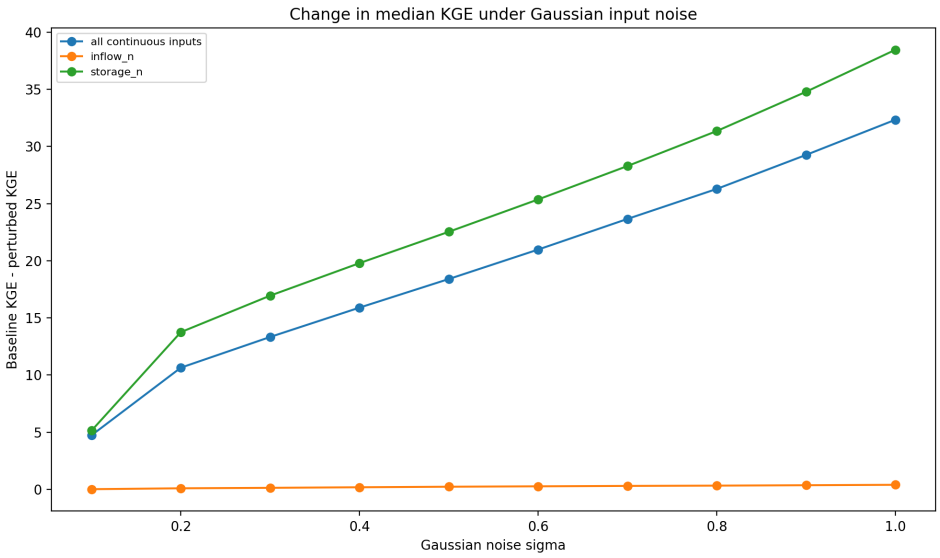


Figure A1: KGE robustness curve for the GRU model from the extra dataset.

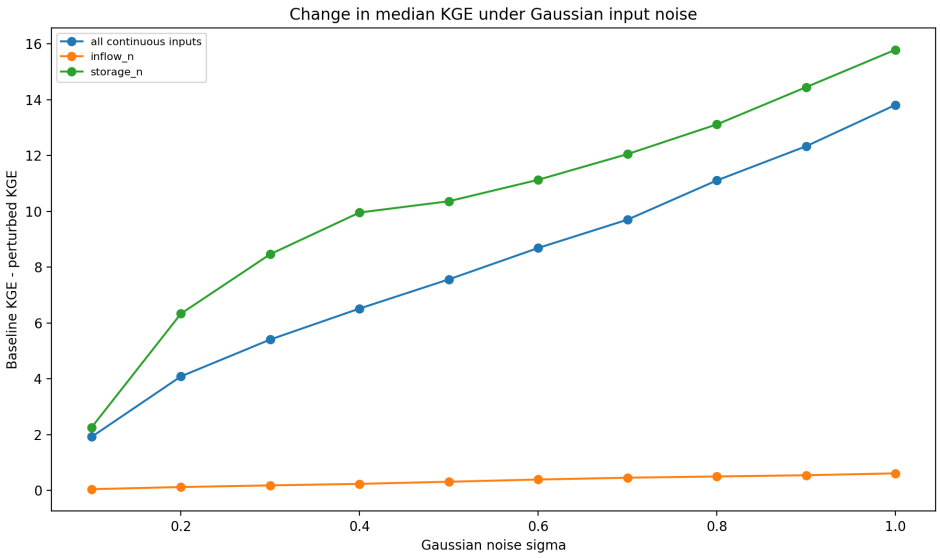


Figure A2: KGE robustness curve for the LSTM model from the extra dataset.

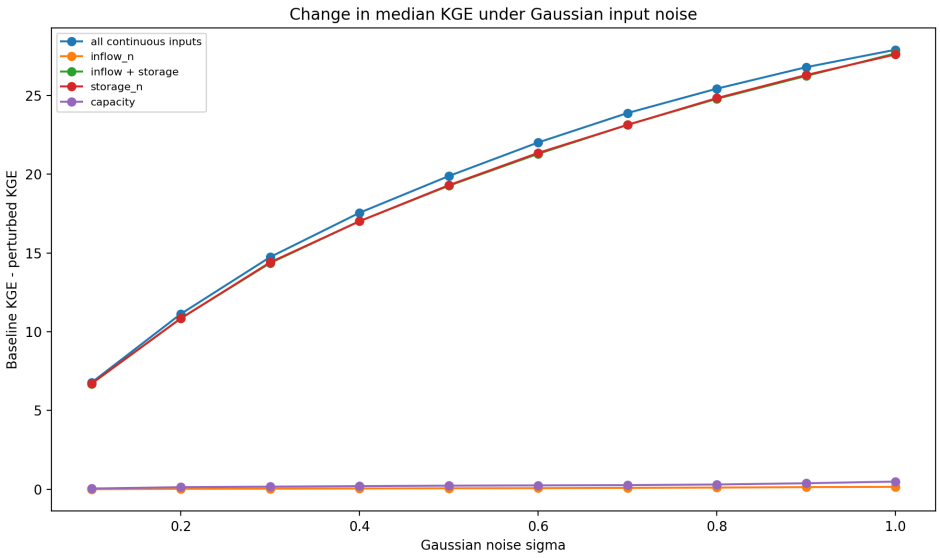


Figure A3: KGE robustness curve for the conditioned GRU model from the extra dataset.

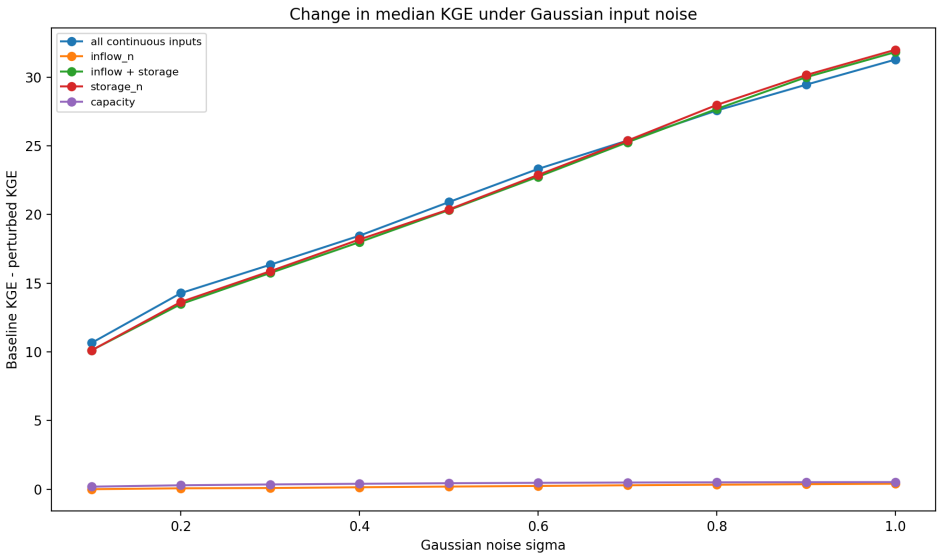


Figure A4: KGE robustness curve for the conditioned LSTM model from the extra dataset.

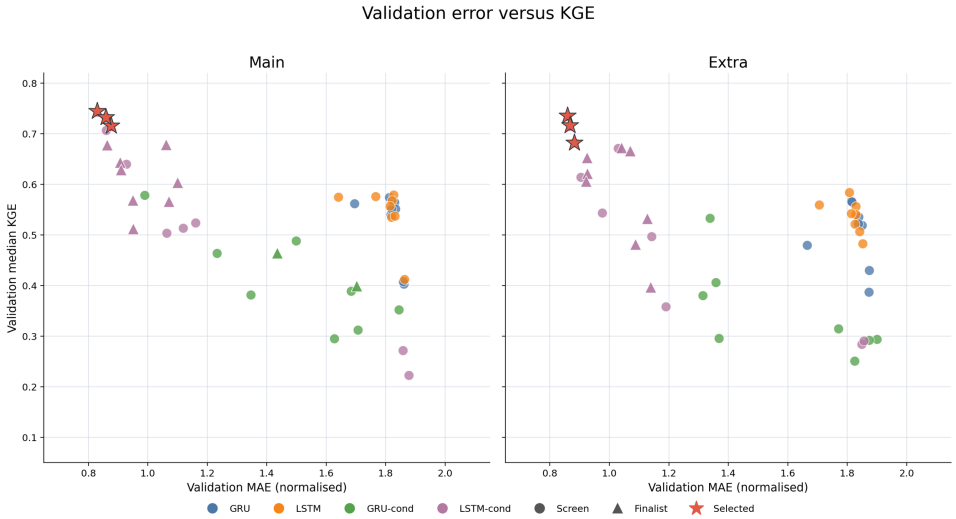


Figure A5: Validation median KGE plotted against normalised validation MAE for all model-selection runs in the main and extra datasets. Colours indicate the model architecture, marker shapes distinguish screening and finalist runs, and stars indicate the selected final configurations. Lower MAE and higher KGE indicate better validation performance.

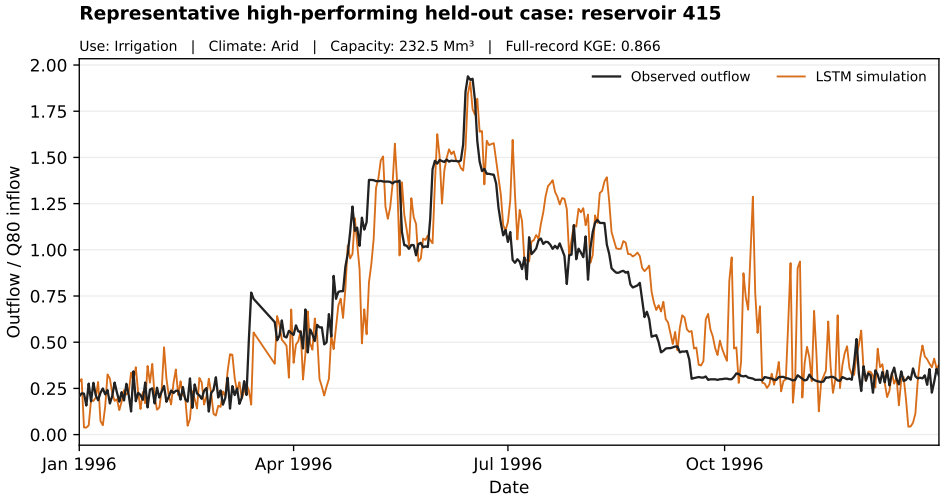


Figure A6: Observed and LSTM-simulated normalised outflow for a high-performing held-out reservoir in the main dataset. A year representative of the overall KGE metric is used for visualisation.

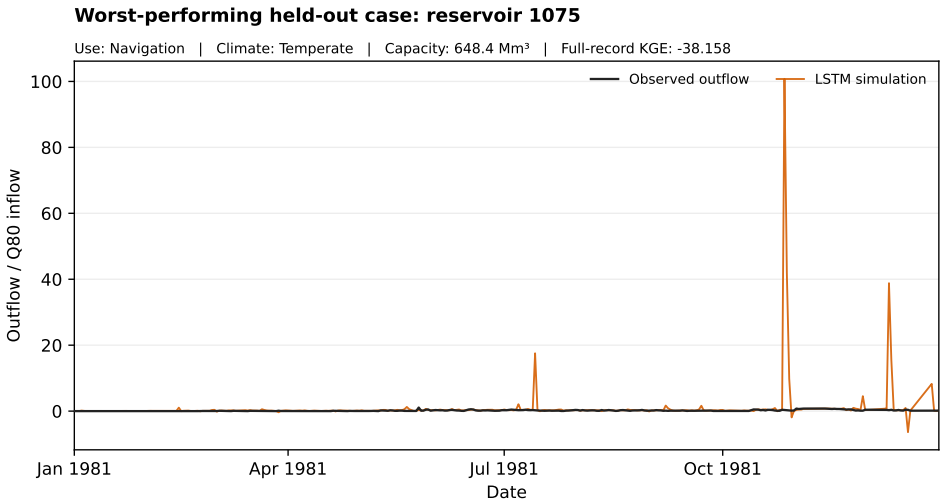


Figure A7: Observed and LSTM-simulated normalised outflow for the lowest-KGE held-out reservoir in the main dataset. A year representative of the overall KGE metric is used for visualisation.

Master's Theses in Mathematical Sciences 2026:E78
ISSN 1404-6342

LUNFBV-3020-2026

Computational Science
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lu.se/