Statistical Learning for Unimpaired Flow Prediction in Ungauged Basins

By

Elaheh White
B.S. University of Kentucky 2015
M.S. University of California, Davis 2017

Dissertation

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Philosophy

in

Civil and Environmental Engineering

in the

Office of Graduate Studies

of the

University of California

Davis

Approved:

Jay R. Lund, Chair

Robert J. Hijmans

Jonathan D. Herman

Committee in Charge

2020

To Jim White,...
I hope you were right, and that we meet again.

Contents

List of Figures

List of Tables

Abstract

Statistical Learning for Unimpaired Flow Prediction in Ungauged Basins

All science is the search for unity in hidden likeness (Bronowski, 1988). There are two practical reasons to approximate processes that produce such hidden likeness: (1) *prediction* for interpolation or extrapolation to unknown (often future) situations; and (2) *inference* to understand how variables are connected or how change in one affects others. Statistical learning tools aid prediction and at times inference. In recent years, rapidly growing computing power, the advent of machine learning algorithms, and more user-friendly programming languages (e.g., R and Python) support applying statistical learning methods to broader societal problems.

This dissertation develops statistical learning models, generally simpler than mechanistic models, to predict unimpaired flows of California basins from available data. Unimpaired flow is the flow produced by the basin in its current state, but without human-created or operated water storage, diversion, or return flows (California Department of Water Resources, Bay-Delta Office, 2016). The models predict unimpaired flows for ungauged basins, an International Association of Hydrological Sciences "grand challenge" in hydrology. In Predicting Ungauged Basins (PUB), the models learn from information at gauged points on a river and extrapolate to ungauged locations.

Several issues arise in this prediction problem: (1) How we view hydrology and how we define observational units determine how data is pre-processed for statistical learning methods. So, one issue is in deciding the organization of the data (e.g., aggregate vs. incremental basins). Such data transformation or pre-processing is explored in Chapter 2. (2) Often, water resources problems are not concerned with accurately predicting the expectation (or mean) of a distribution but require better estimates of extreme values of the distribution (e.g., floods and droughts). Solving this problem involves defining asymmetric loss functions, which is presented in Chapter 3. (3) Hydrologic observations have inherent dependencies and correlation structure; gauge data are structured in time and space, and rivers form a network of flows that feed into one another (i.e., temporal, spatial, and hierarchical autocorrelation). These characteristics require careful construction of resampling techniques for model error

estimation, which is discussed in Chapter 4. (4) Non-stationarity due to climate change may require adjustments to statistical models, especially for long-term decision-making. Chapter 5 compares unimpaired flow predictions from a statistical model that uses climate variables representing future hydrology to projections from climate models.

These issues make Predicting Ungauged Basins (PUB) a non-trivial problem for statistical learning methods operating with no *a priori* knowledge of the system. Compared to physical or semi-physical models, statistical learning models learn from the data itself, with no assumptions on underlying processes. Their advantages lie in their fast and easy development, simplicity of use, lesser data requirements, good performance, and flexibility in model structure and parameter specifications. In the past two decades, more sophisticated statistical learning models have been applied to rainfall-runoff modeling. However, with these methods, there are issues such as the danger of overfitting, their lack of justification outside the range of underlying data sets, complexity in model structure, and limitations from the nature of the algorithms deployed.

Keywords: predicting ungauged basins (PUB); rainfall-runoff modeling; asymmetric loss functions; structured data; blocked resampling methods; climate change; water resources; hydrology; statistical learning.

Acknowledgments

This Dissertation owes its completion to committee members Dr. Jay R. Lund, Dr. Robert J. Hijmans, and Dr. Jon D. Herman. With special thanks to: Dr. Duncan Temple-Lang for making me a better R programmer; Dr. Carlos Puente for the many hours of conversation; Dr. Carole Hom for her help attaining both NSF grants; Dr. Graham Fogg for his mentorship; Marielle Pinheiro for holding my hand while programming; Emily Frankel for encouraging me to work when times were hard; Caroline McKusick, Duane Wright, BB Buchanan, Toby Smith, Sarah Grajdura, and Kristy Smith for enriching my life outside of academia through organizing against injustice; Nadya Alexander-Sanchez for offering great advice at very key moments; Belinda, Kelly, and Marty for their love and encouragement; and Brad White, my significant other, without whom this PhD would not have been completed.

All data processing and model development for this dissertation was done in `R version 3.6.0 (2019-04-26)`, a statistical programming language, (R Core Team, 2019) on platform: x86_64-w64-mingw32/x64 (64-bit), running under: Windows 10 x64 (build 18363). Programs were written in `RStudio`, an integrated development environment, (RStudio Team, 2016), and used the following packages: `astsa` (Stoffer, 2020), `dismo` (Hijmans, Phillips, Leathwick, & Elith, 2017), `doParallel` (Corporation & Weston, 2017), `fBasics` (Wuertz, Setz, & Chalabi, 2017), `foreach` (Microsoft & Weston, 2017), `geojsonio` (Chamberlain & Teucher, 2018), `ggplot2` (Wickham, 2016), `ggpmisc` (Aphalo, 2016), `Hmisc` (Harrell Jr, with contributions from Charles Dupont, & many others., 2020), `hydroGOF` (Mauricio Zambrano-Bigiarini, 2017), `keras` (Allaire & Chollet, 2019), `lattice` (Sarkar, 2008), `lmtest` (Zeileis & Hothorn, 2002), `magick` (Ooms, 2019), `purrr` (Henry & Wickham, 2020), `randomForest` (Liaw & Wiener, 2002a), `raster` (Hijmans, 2019), `RColorBrewer` (Neuwirth, 2014), `reshape2` (Wickham, 2007), `rgdal` (R. Bivand, Keitt, & Rowlingson, 2018), `rgeos` (R. Bivand & Rundel, 2018), `sp` (Pebesma & Bivand, 2005; R. S. Bivand, Pebesma, & Gomez-Rubio, 2013), `statmod` (Giner & Smyth, 2016), `xtable` (Dahl, Scott, Roosen, Magnusson, & Swinton, 2019), and `zoo` (Zeileis & Grothendieck, 2005).

Code written for this dissertation lives at:
https://github.com/whiteellie/predicting-ungauged-basins.

This material is based upon work partly supported by the NSF GRFP under Grant No. 1650042 and the Climate Change, Water, and Society NSF IGERT, to UC Davis DGE No. 1069333. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Acronyms & Abbreviations

AGG: Aggregate Basins (Chapter 2)
AF: Acre-feet
AF/m: Acre-feet/Month
BBG: Blocked By Group (Chapter 4)
BBMG: Blocked by Multiple Groups (Chapter 4)
bR^2: Bias-Corrected Coefficient of Determination
CDWR: California Department of Water Resources
CDEC: California Data Exchange Center
GLM: Generalized Linear Multivariate Regression
IID: Independent and Identically Distributed (Chapter 4)
INC: Incremental Basins (Chapter 2)
LINEXE: Linear-Exponential Error (Chapter 3)
LM: Linear Multivariate Regression
LOGCOSH: Log Cosine Hyperbolic Loss (Chapter 3)
LOGO: Leave One Group Out (Chapter 4)
LMGO: Leave Multiple Groups Out (Chapter 4)
MAE: Mean Absolute Error (Chapter 3)
MSE: Mean Squared Error (Chapter 3)
MSPE: Mean Squared Percentage Error (Chapter 3)
NN: Neural Networks
PUB: Predicting Ungauged Basins
RESUB: Resubstitution (Chapter 4)
RF: Random Forests
SQM: Square Miles
TAF: Thousand Acre-feet
UF: Unimpaired Flow
WLSE: Weighted Least Squares Error (Chapter 3)

Chapter 1

Introduction & Literature Review

> Life must be lived forwards, but it can only be understood backwards.
>
> Sören Kierkegaard, *"The Journals of Sören Kierkegaard"*, 1844

1.1 Introduction

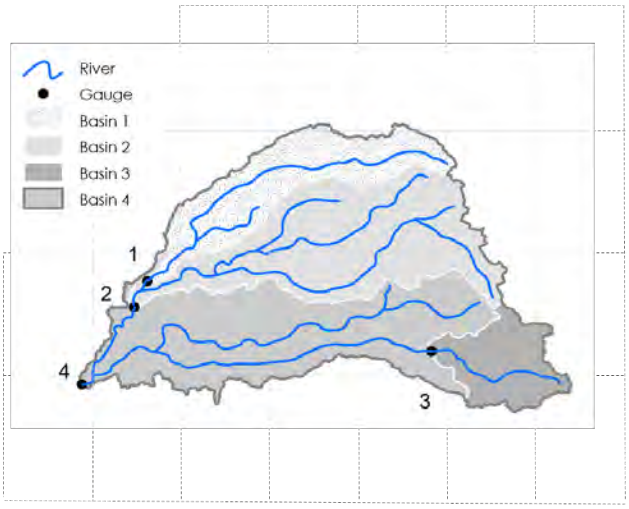
Our ability to extract insights from large diverse data sets has rapidly improved with growing computing power and sophisticated algorithms. The field of *statistical learning* has emerged as a framework that ranges from simple linear regression to complex algorithmic methods (James, Witten, Hastie, & Tibshirani, 2013). A main contribution of this field is the development of modeling techniques that allow for the semi-automatic creation of complex models, with many interacting predictor variables, which are not overfit, and predict well.

These developments allow for more accurate and flexible empirical models to manage complex systems. For example, in hydrology, runoff formation processes are highly variable, non-linear, and spatially heterogeneous, which are a challenge for predicting processes such as streamflow (Dooge, 1986). The International Association of Hydrological Sciences (IAHS) dubbed the 2003-2012 years the decade on Predictions in Ungauged Basins (PUB) (Sivapalan et al., 2003). The PUB initiative has aimed the scientific community, in a coordinated manner, towards achieving major advances in thydrologic predictions for ungauged basins (Figure 1.1).

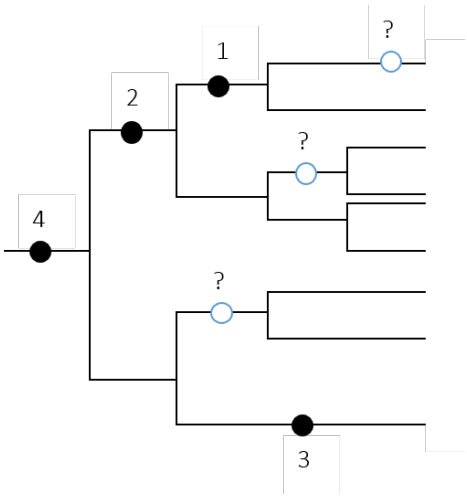
Predicting and forecasting hydrology at ungauged sites promotes better management of water and the environment (Sivapalan et al., 2003). Hydrologic estimation is important for managing river basins; integrating economic, social and environmental perspectives (Sivapalan, 2003); flood protection; water supply and drought management; solving water quality issues (Hrachowitz et al., 2013); and they can serve as inputs for other models.

1.2 Terms & Definitions

This dissertation investigates the relationships between the response variable, **unimpaired flow**, and various predictor variables, climate and basin characteristics. Unimpaired flow is the flow produced by the basin in its current state, but without human created or operated water storage, diversion, or return flows (California Department of Water Resources,



(a) PUB Problem



(b) PUB Schematic

Figure 1.1: The predicting ungauged basins (PUB) problem. This dissertation focuses on predicting unimpaired flows at ungauged locations from other gauges on the network. Predictor variables include climate and basin characteristics.

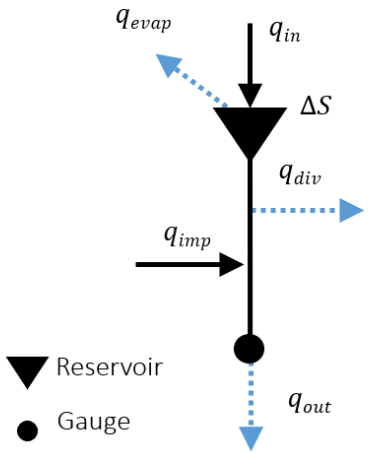


Figure 1.2: Calculating unimpaired flow. Unimpaired flow is calculated by adding back in diversions, subtracting imports, accounting for change in storage and evaporation caused by the reservoir.

Bay-Delta Office, 2016). Unimpaired flow is mostly used where dams have changed the natural flow regime and is calculated by a simple accounting of water in the system (Figure 1.2 and Equation 1.1),

$$q_{uf} = q_{out} - q_{imp} + q_{div} + \Delta S + q_{evap} \tag{1.1}$$

where q_{uf} is unimpaired flow, q_{out} is observed gauge data, q_{imp} is imported flows, q_{div} is diverted flows, ΔS is the change in storage, and q_{evap} is the evaporation out of the system.

In contrast, **natural flow** is the runoff produced by a basin in its pre-development state prior to human alterations (Poff et al., 1997). The differences between unimpaired flow

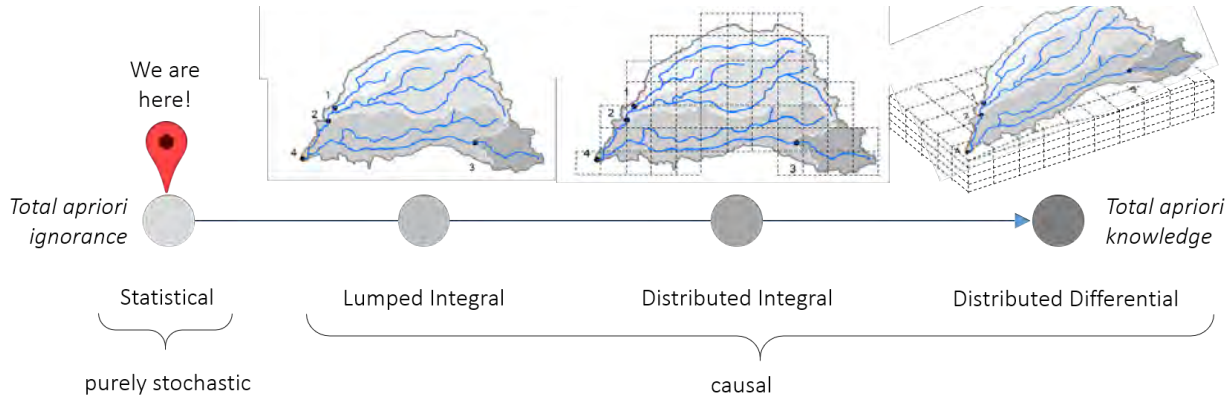


Figure 1.3: The different classes of hydrologic models. The hydrologic modeling field has been moving from total a priori ignorance to total a priori knowledge of the system. With the increase in computing power and the development of statistical learning methods, hydrologist can now re-visit predicting hydrologic conditions with purely stochastic methods.

and natural flow are usually driven by effects of levees, upland land use, wetlands, and groundwater. This study is only concerned with unimpaired flow; its models were built with unimpaired flow data from the California Data Exchange Center (CDEC) and predictor variables from various sources discussed in Appendix A.

1.3 Literature Review

1.3.1 Hydrologic Modeling

Hydrologic models for PUB can be classified as **mechanistic** (physical process-based, causal) or **empirical** (statistical, purely stochastic) (Guisan & Zimmermann, 2000) (Figure 1.3). Each approach strikes a different balance between generality, realism, cost, and precision for better understanding, predicting, and managing natural resources (Levins, 1966; Klemes, 1982). However, all modeling techniques assume that the past is a reasonable guide to the future, and that data from one basin is useful for understanding hydrologic responses at another basin (Sivapalan, 2003).

Hydrologists have used both mechanistic and empirical models to represent complex runoff processes; since the mid-19th century, with the employment of the **rational method**, empirical relationships have been used in rainfall-runoff modeling (Beven, 2011). Engineers developed the rational method in response to problems in which the design discharge was of major concern (i.e., urban sewer, land reclamation drainage systems, and reservoir spillway design) (Todini, 1988). This method, based on the concept of concentration time, calculates runoff by simply multiplying a runoff coefficient by rainfall intensity and the basin's drainage area. It is applicable only to small or mountainous catchments where the rainfall duration normally exceeds the basin's **concentration time**–the time needed for the entire basin area's precipitation to reach the basin's outlet as discharge.

To address more complexities in rainfall duration, basin size, and non-uniform characteristics, other methods emerged. In the 1930s, the **unit hydrograph** method was developed (Sherman, 1932). In the 1950s, mathematical techniques such as Z, Laplace or Fourier trans-

forms led to the derivation of response functions from the analysis of input and output data (Dooge, 1973). In the 1960s, grander approaches emerged to model physical processes of the hydrologic cycle. Models increased in complexity over time and often lacked realistic parameter estimates, leading researchers to other ambitious mechanistic modeling efforts (Todini, 1988). These models require considerable field input data collection and calibration to obtain basin-specific parameters (Singh & Frevert, 2005). Unfortunately, as mechanistic models increase in complexity, it is unclear if hydrologic predictions improve commensurately (Beven, 2011).

Our incomplete understanding of the process (Hrachowitz et al., 2013), poor understanding of where water goes when it rains, what flow paths it takes to the stream, and the age of the water that emerges in the channel (Sivapalan, 2003) make PUB a difficult problem to model. Moreover, spatio-temporal heterogeneity of climate and basin characteristics create uniqueness-of-place and time issues, and there is a lack of agreement on suitable regionalization techniques for this problem (Hrachowitz et al., 2013).

Without a unifying approach, and considering the increasing availability of environmental data, in the past two decades, more sophisticated statistical learning models have been applied to rainfall-runoff modeling. In juxtaposition with physical or semi-physical models, machine learning models learn from the data itself, with no (or few) assumptions about underlying processes. Appendix B defines terms and concepts used in statistical learning.

1.3.2 Statistical Learning

Artificial intelligence has gone through the ages of speculation (1940s), dawn, business, and bulldozer (Winston, 2010). In the bulldozer age, with seemingly unlimited computing capacity, machines process more abundant data much like a bulldozer processes soil. Recent advances in reinforcement learning, one-hot learning (where machines learn from the first example), learning in sparse spaces, and the integration of thinking, perception, and action (rather than viewing them separately) are moving us away from the bulldozer era (Winston, 2010). The application of these newer techniques to water resources problems is slow. Appendix C presents a brief history of statistical learning.

The taxonomy presented in Figure 1.4 can help guide users through a discovery process. Its goal is for the user to be able to identify a statistical model or method of interest without prior knowledge of its existence. Here, we have grouped statistical learning and data analysis methods into seven categories: supervised machine learning, regression family, time series analysis, geostatistics, multi-variate analysis, unsupervised machine learning, and other methods. **Supervised machine learning** methods are more generally used for predicting a variable in the past where no equation is needed to represent the model. In contrast, the **regression family** of methods are used when the purpose is more *inference* than *prediction*, and equations-or more specifically the coefficients of variables in the equations-are of interest. **Time series analysis** is most suited to prediction problems where the time component is of interest (e.g., problem of extrapolating to the future), as opposed to **geostatistics**, which is mainly concerned with the spatial component of the data. **Pattern recognition**, **multivariate analysis**, and **unsupervised machine learning** methods find natural groupings in the data. Other methods handle networks, text, patterns caused by latent factors, and relationships between variables. Lastly, in **descriptive methods**, measures of centrality (e.g., mean and median), measures of position (e.g., quantiles), measures of spread (e.g.,

range, standard deviation, and quantiles), and the distributions of variables are some ways to describe data.

Because of the nature of the data civil and environmental engineers come to contact with, this taxonomy was more refined for regression type problems. Other taxonomies have emphasized combinatorics and probability theory (i.e., the theoretical foundations of statistics) (Chiou, 2008; bioquest.org, 2011). We categorized statistical methods into two broad categories of prediction and inference while others immediately branch into more categories: estimation, exploration, prediction, decisions (hypothesis), uncertainties, and descriptive categories (bioquest.org, 2011). Some taxonomies list many methods without distinguishing between them (gogeometry.com, 2017; Brownlee, 2020). Also, unlike this taxonomy, it is popular to put data visualization into a separate category (Covington, Hill, & Bruff, 2012; Chiou, 2008). Altogether, in statistical learning, method or model selection is iterative and should follow the *generate-and-test* approach. So, any guide to model selection is only a heuristic, meaning as a general rule it will recommend appropriate methods, but also may fail or mislead.

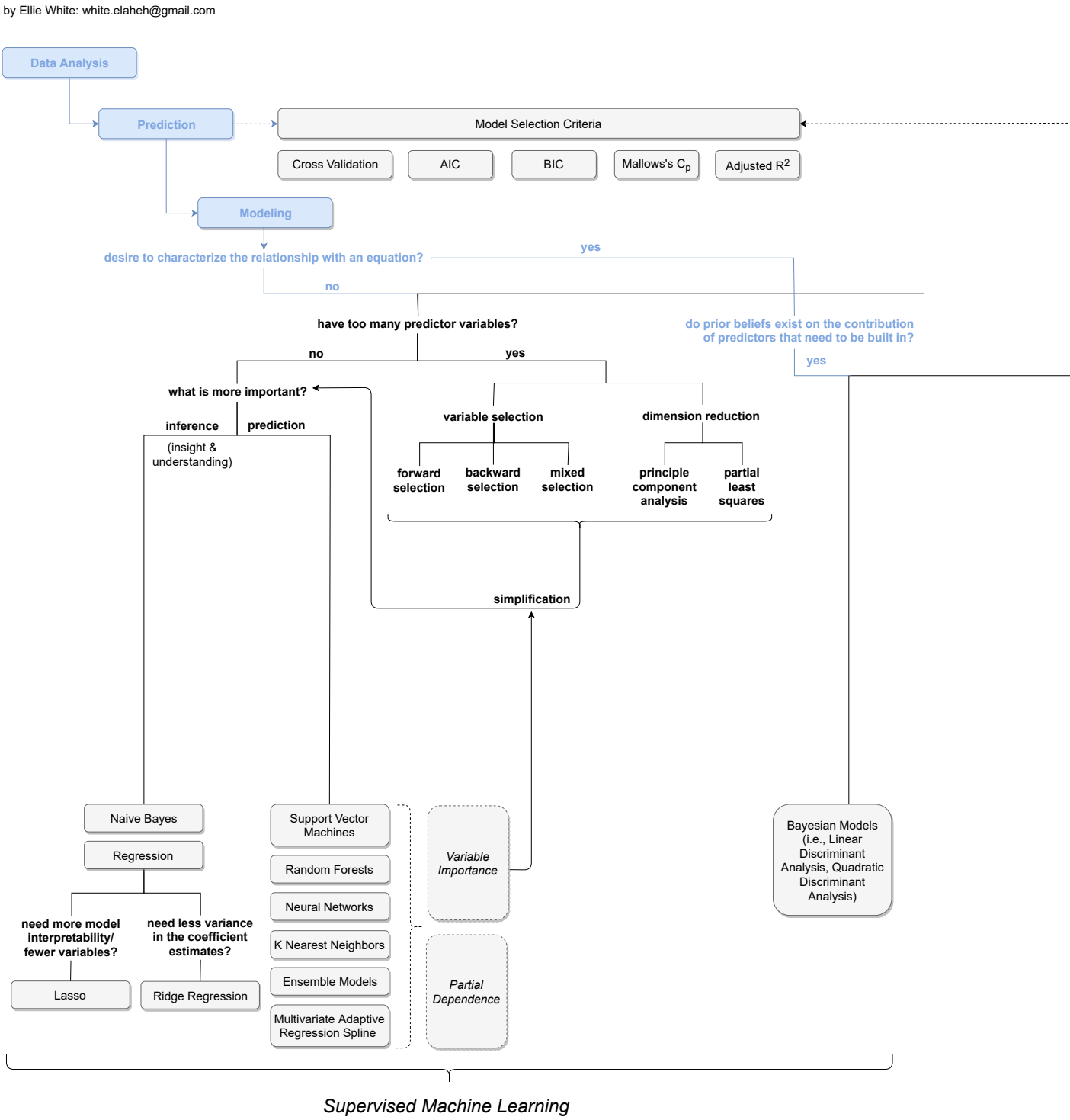
The hydroinformatics literature shows that the techniques presented in the heuristic guide are aiding civil engineers in fields like: (1) hydrology: e.g., rainfall-runoff modeling and model calibration; (2) hydraulics: e.g., water levels and flows in channels, reservoirs, and aquifers; (3) environmental water quality: e.g., temperature and chemical concentrations; (4) urban water supply: e.g., water demand and water distribution networks; and (5) general data cleaning and anomaly detection. The following sections discuss models suitable to the PUB problem.

1.3.3 Suitable Statistical Modeling for Hydrologic Data

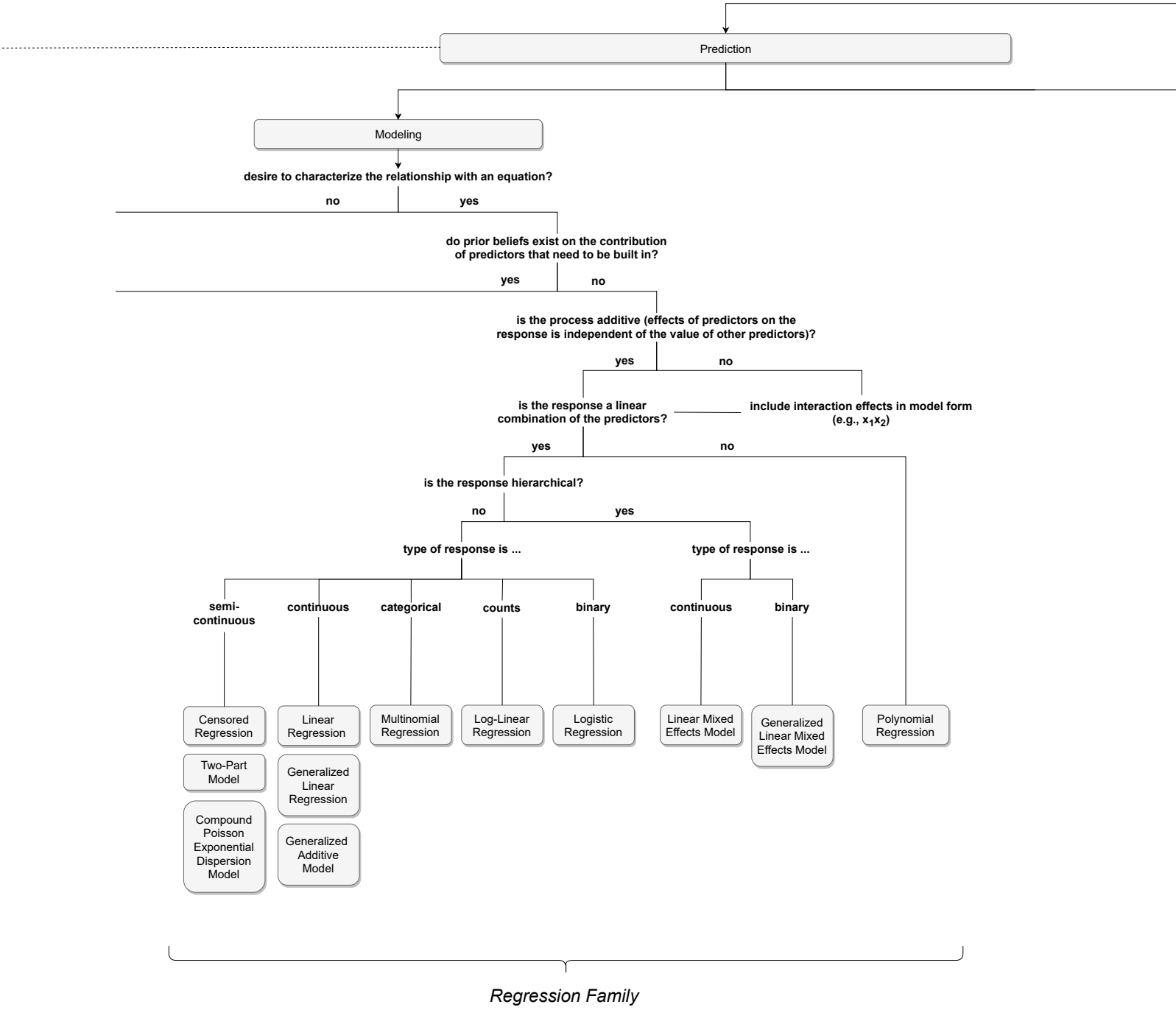
Precipitation feeding into a stream must satisfy soil moisture deficits along its flow path before it produces runoff. In other words, the soil needs to "fill" to a threshold before it can "spill" to become runoff (Spence & Woo, 2006). So, **threshold behavior** is frequently discussed as influencing local, hillslope and catchment scale runoff generation processes (Zehe & Sivapalan, 2008). This physical phenomenon may be why most successful machine learning studies in rainfall-runoff modeling use artificial neural networks (e.g., Minns & Hall, 1996; Dawson & Wilby, 1998; Tokar & Johnson, 1999; Hsu, Gupta, Gao, Sorooshian, & Imam, 2002; Hu, Wu, & Zhang, 2007; Abrahart, Heppenstall, & See, 2007; Govindaraju & Rao, 2013).

In **artificial neural networks**, at each node, the weighted sum of all inputs are passed through a non-linear activation function. Much like the neurons in our brains, there is a threshold that determines if the neuron will "fire." Recent state-of-the-art technology in neural networks show that Long Short-Term Memory (LSTM) networks offer unprecedented accuracy for prediction in ungauged basins (Kratzert et al., 2019). LSTMs are a special type of recurrent neural networks capable of learning long-term dependencies in sequence (e.g., time series) prediction problems.

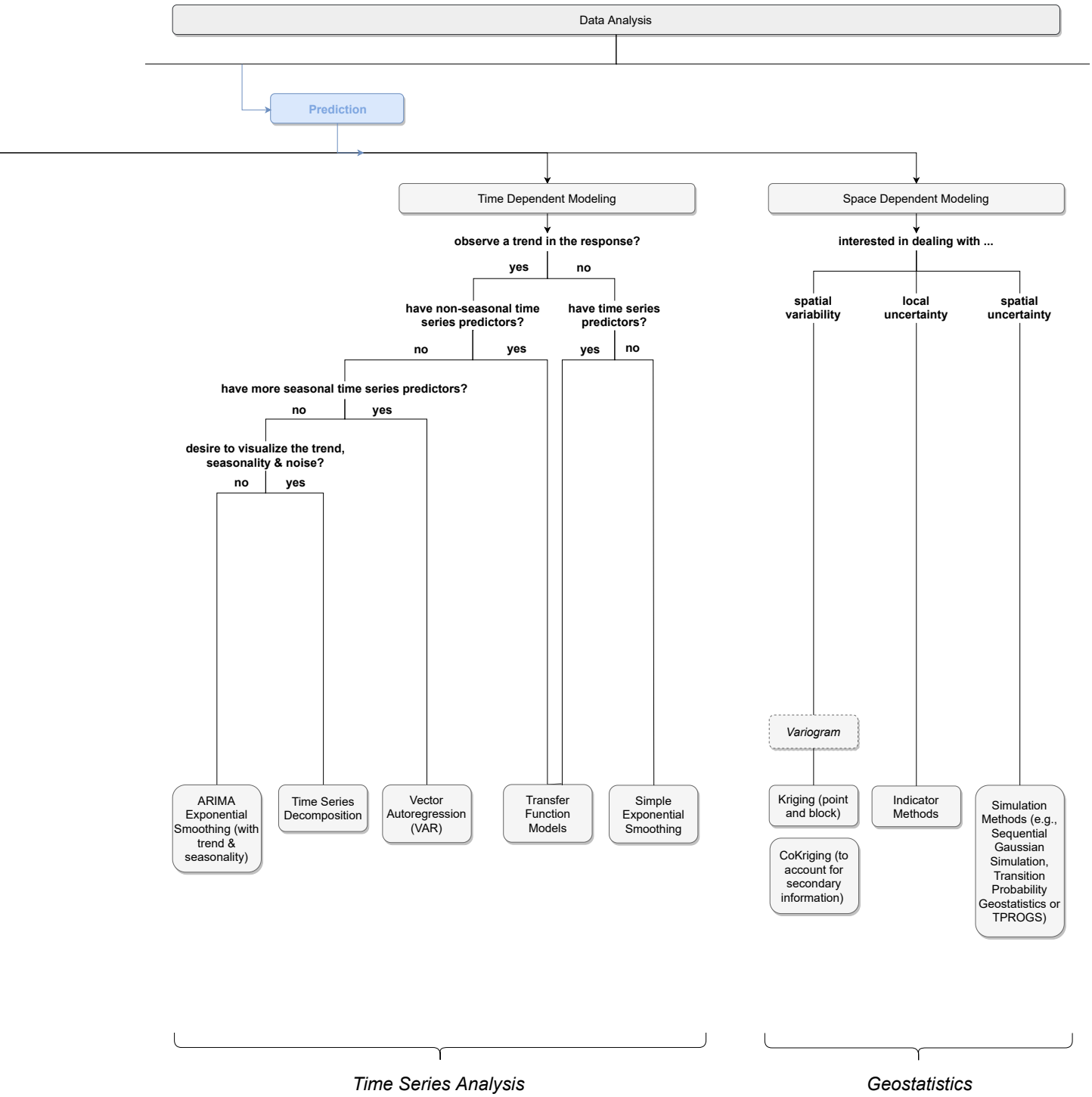
The same threshold effect can be replicated with **tree based algorithms** where models are built with a series of binary splits on the predictor variables (e.g., Iorgulescu & Beven, 2004; Galelli & Castelletti, 2013; Magnuson-Skeels, 2016; Worland, Farmer, & Kiang, 2018). Studies which have fairly small data sets suffer when forming the test/train or calibration/-validation split. Usually, in these studies data for one whole basin is not held out when



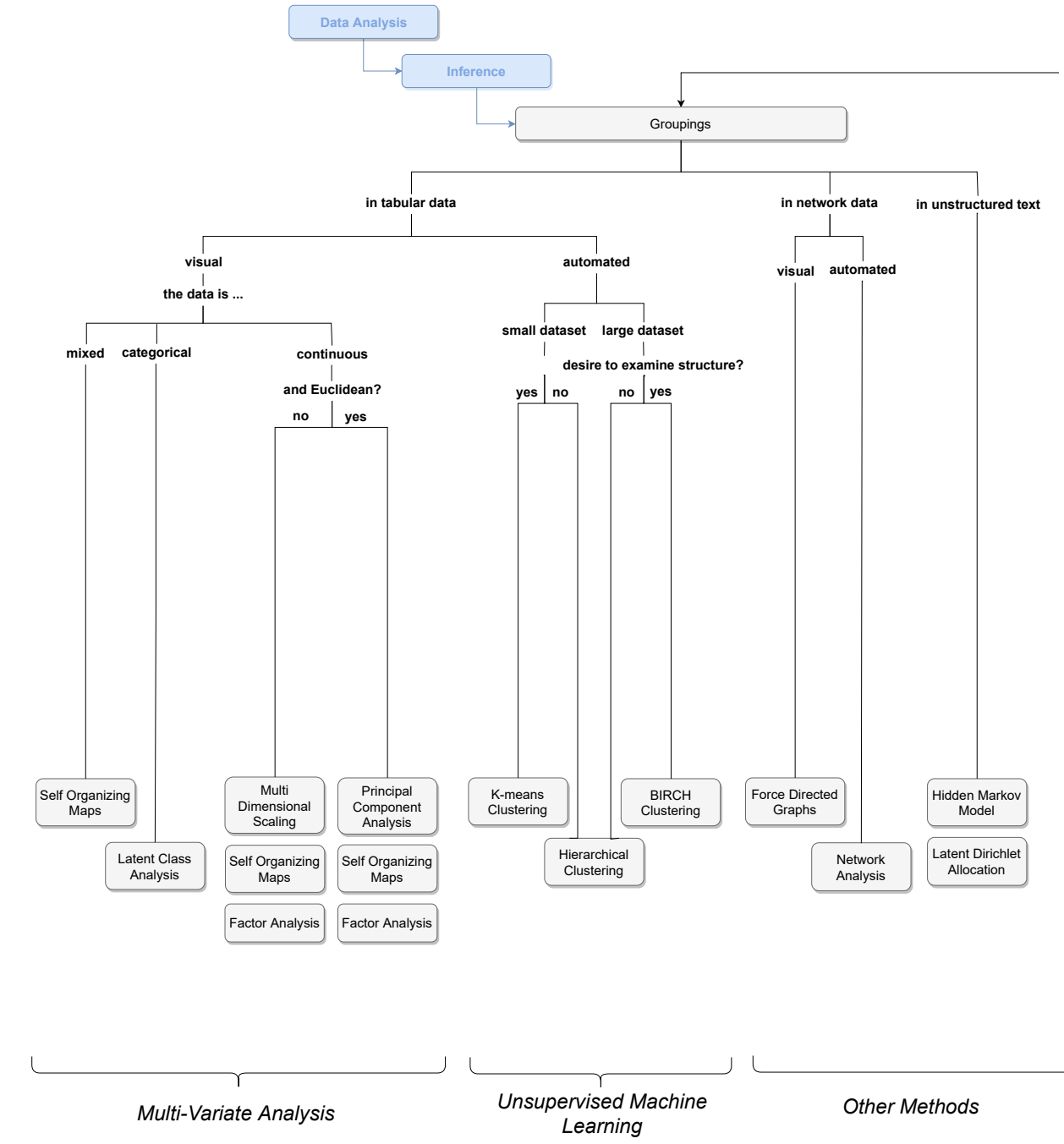
(a) Supervised machine learning methods.



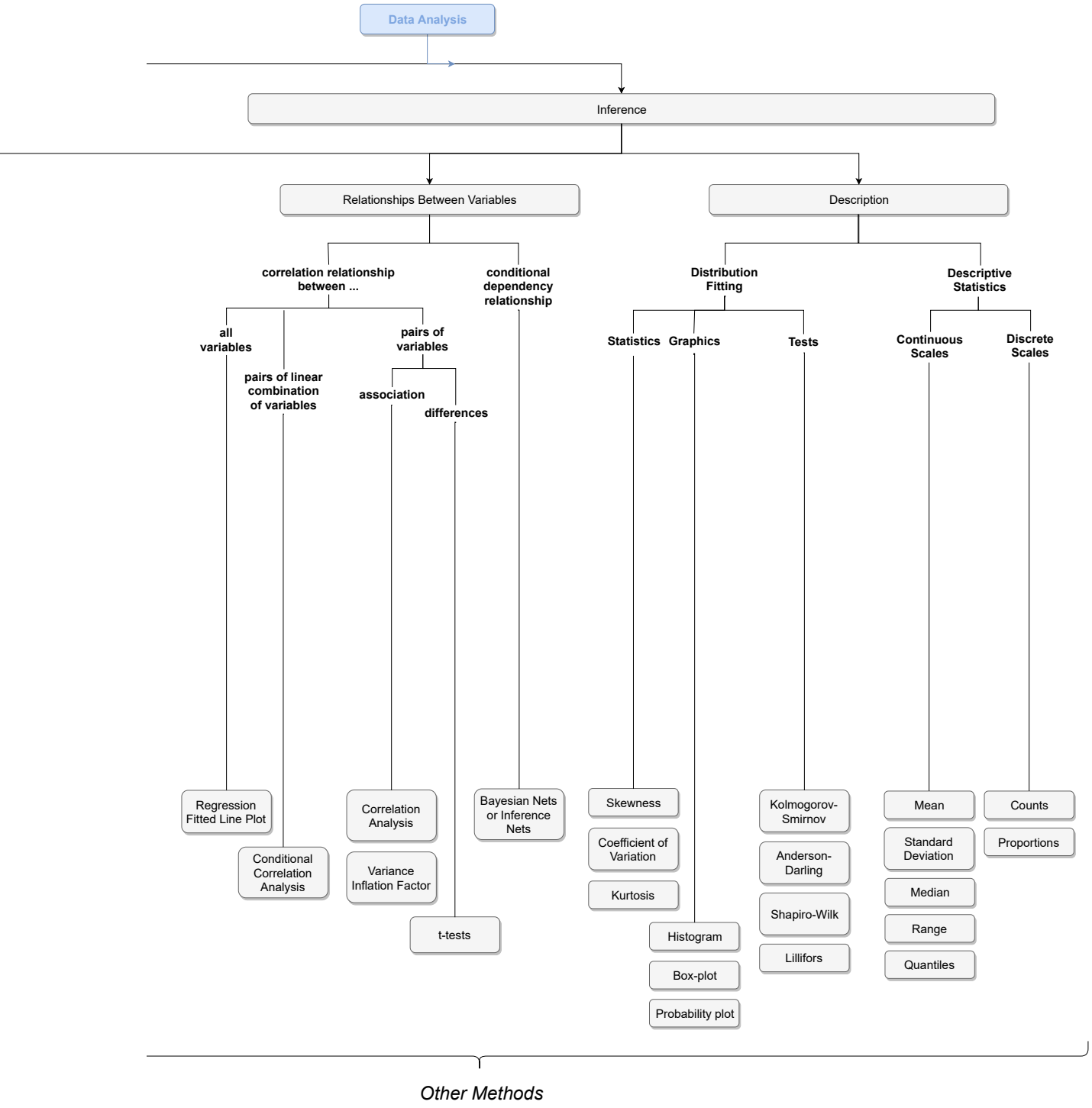
(b) Regression family of methods.



(c) Time series analysis and methods in geostatistics.



(d) Multi-variate analysis, unsupervised machine learning, and some other methods in inference.



(e) Other methods in inference.

Figure 1.4: Heuristic guide for data analysis. We can group statistical learning methods into seven main categories: (a) supervised machine learning, (b) regression family, (c) time series analysis, geostatistics, (d) multi-variate analysis, unsupervised machine learning, and (e) other methods. Blue text is repeated top-level information that is off screen.

training; in other words, the models can learn from a partial record of the basin of interest. Although this approach seems to be accepted for rainfall-runoff modeling in the current literature, it does not comply by the test set requirements in the PUB problem–where no data from the basin in the test set are available to the model. When data sets are large (e.g., studies done on the GAGESII data set, a massive USGS hydrologic data set), this problem is less pronounced. Some studies employ a random test/train split which is not appropriate when the dataset has internal correlations. We discuss this concept further in Chapter 4. These studies also employ a pre-modeling split on the dataset by classifying basins as "impaired" vs. "reference." This imposes a subjective top split in the data and homogenizes basins in the study; the reference or unimpaired basins are usually smaller headwater basins with low flows. As such, and as expected, these models fail to accurately predict flows when extrapolating to basins lower in the network, with higher flows, since the model was denied such information.

More recently, studies have turned to **support vector machines** (SVM) (Asefa, Kemblowski, McKee, & Khalil, 2006; Lin, Cheng, & Chau, 2006), which initially were only applied to classification problems and have now been modified to accommodate regression problems (e.g., applications in flood forecasting: Han, Chan, & Zhu, 2007; Yu, Liong, & Babovic, 2004; Bray & Han, 2004). Such studies show that advances are putting SVMs generally on par with artificial neural networks in terms of model performance. However, application of SVMs in time-series regression are still in their infancy; one study showed a peculiar behavior of SVMs where lighter rainfall would generate unrealistic hydrographs that would increase to an equilibrium point rather than having the characteristic skewed bell shape (Han et al., 2007). This contradicts the physical principle that less rainfall cannot generate more flow.

The difficulty in modeling lower flows is not unique to SVMs. Other modeling techniques (e.g., linear and generalized linear models) suffer from the same problem given that the response, unimpaired flow, is a **semi-continuous** variable. Semi-continuous data take non-negative values but have a substantial proportion of values at zero. The modeling of such "clumped-at-zero" or "zero-inflated" data is challenging (Min & Agresti, 2002). Several methods have been developed to address this issue:

- Censored regression models: A censored regression, or Tobit, model assumes that data comes from a single underlying Normal distribution, but that negative values are censored and stacked on zero (Tobin, 1958).

- Two-part models: As opposed to the Tobit model that allows the same underlying stochastic process to determine whether the response is zero or positive as well as the value of a positive response, two-part models allow the two components to have different parameters. Without assuming an underlying distribution, Duan, Manning, Morris, and Newhouse (1983) proposed a two-part model that uses two equations to separate the modeling into two stages. The first stage refers to whether the response outcome is positive (e.g., a binomial model). Conditional on its being positive, the second stage refers to its level (e.g., linear model).

- Compound Poisson exponential dispersion models: A model that uses a single distribution from the exponential dispersion family (i.e., Tweedie distribution) to analyze

semi-continuous data. Distributions in this family have a given range of shape parameters ($1 < \alpha < 2$) which define a point mass at zero and a skewed positive distribution for positive values.

As Min and Agresti (2002) explain, other modeling methods exist for the problem of inflated zeros or other inflated boundaries (e.g., ordinal threshold, finite mixture, Neyman type A models). Unfortunately, these methods may require groupings that necessitate information loss, may overestimate the number of components when there is a lack of model fit, or employ methods where the mathematical and inferential advantages associated with the family of distributions are not available and are simply difficult to fit. As such, we will not discuss them here.

This thesis develops Linear Multivariate Regression (LM) as a first pass model, followed by Generalized Linear Regression (GLM) with the Tweedie distribution, Random Forest (RF), and Neural Network (NN) models.

1.4 Limitations & Assumptions of Statistical Modeling

Many hydrologists are skeptical of statistical modeling. Klemes (1982) warns modelers of the general limitations of empirical modeling, some of which are discussed here.

In search of "better calculus", the modeler may be in danger of **overfitting**–regarding noise in the data as information (Klemes, 1982). Resampling methods, when correctly applied, can illuminate differences between training and testing set performances (Friedman, Hastie, & Tibshirani, 2001).

Furthermore, empirical models must be regarded as **interpolation formulas**, and so, lack justification outside the range of underlying data sets (Klemes, 1982). The models in this study were fitted with data on the California Sierra Nevada mountainous basins, and some coastal, and southern California basins (Figure A.1). These training data sets mostly span the same hydrologic region (i.e., the United States Geological Survey Region No. 18). As such, the model may perform poorly for basins outside this spatial range where other hydrologic processes may dominate. We can expect this from observing the spatial variability (characterized by the annual coefficient of variation) in precipitation across the United States (Figure 1.5; Dettinger et al., 2011).

In addition to concerns with spatial extrapolation, there is the issue of temporal extrapolation. Climate change brings **non-stationarity** in environmental variables like precipitation and temperature. Empirical models for flow should not be used to extrapolate beyond the limits of the variables the model observes or it will risk large errors. However, many advances in time series analysis can include non-stationarity in data; one can reduce the process to a stationary one (i.e., trend seasonality and noise can be decomposed) or consider these processes as stochastic.

Another downside is **complexity in model structure**, especially in ensemble statistical learning methods, sometimes referred to as black-box models. If inference, or model parameters, are of interest, complex models introduce challenges. Dimensionality reduction methods (e.g., principle component analysis, partial least squares) and regularization techniques in regression (e.g., ridge, lasso, and elastic net) can help reduce the number of model parameters, and systematically produce simpler models (Friedman et al., 2001).

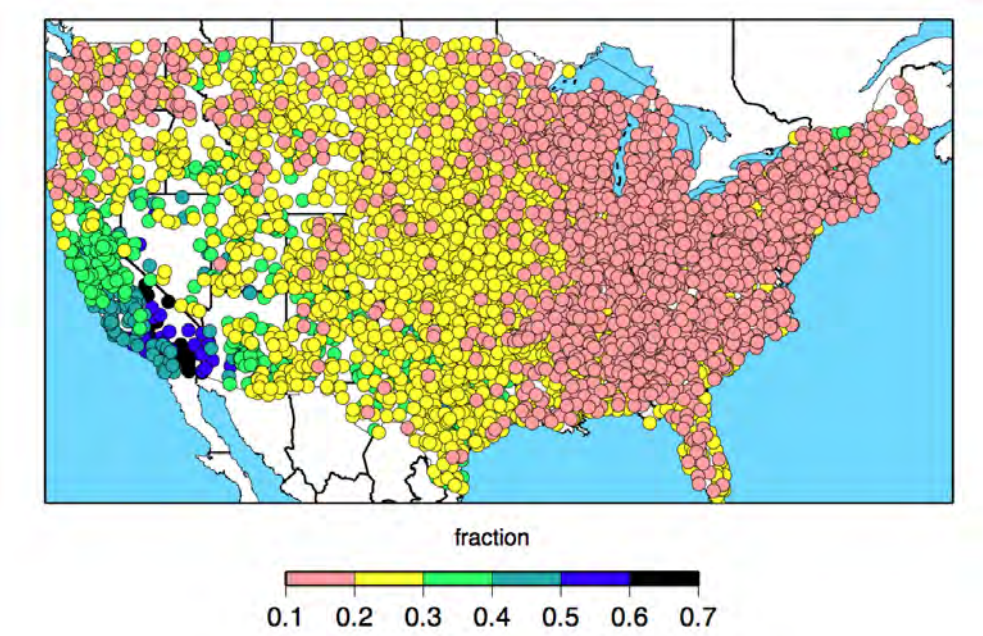


Figure 1.5: Annual coefficient of variation in total precipitation from 1951-2008 in the United States. Reprinted from Dettinger et al., 2011.

The essential **arbitrariness in the selection of the form** of an empirical model is another drawback (Klemes, 1982). Most studies report using one modeling method, which perhaps suggests that researchers are not employing more than one modeling method. Deploying different models can provide insights into the system by revealing the sensitivity of results to the algorithms employed. Therefore, application and comparison of different machine learning models to the PUB problem was considered in this study.

Lastly, some limitations are caused by the **nature of the algorithms** deployed. For example, regression-based random forest models make predictions by averaging predictions made by multiple regression trees. Therefore, the ensemble model limits its predictions to the range seen in the training data; predictions do not extrapolate to ranges not seen in the training data. In fact, averaging dampens the density function when we compare observed to predicted data. This is especially problematic where the extreme tails of the distribution (i.e., floods and droughts) are of interest. Another example is that of the SVMs mentioned before that seem to perform poorly with low rainfall data.

1.5 Conclusion

Generally, in statistical learning, applications lag behind advances in theory; the application of statistical learning theory to water resource problems is still in the bulldozer era. So, most models are computationally expensive. In the past two decades, in hydrology, statistical learning methods have been applied to modeling rainfall-runoff processes, predicting streamflow temperatures, sediment and nutrient loadings, forecasting the groundwater heads in an aquifer, or water demand, among many other problems.

This chapter's main contribution is a heuristic guide to empirical model selection. Like a flowchart, it guides in selecting methods tailored to general purposes and limitations of

various empirical modeling approaches. This guide should help in selecting from range of methods available for a problem at hand and give some comparative insights on these diverse methods. As a heuristic it works in most cases, but it is not comprehensive or applicable to all problems.

In some cases, a wide range of empirical models can be employed, suggesting that no one single modeling method is best across all locations, timescales, and problems. Also, despite their limitations discussed in this chapter, these methods are much easier, faster, and less expensive to apply and study than mechanistic models. They are well suited to dynamic, non-linear, and sometimes noisy data, especially when underlying physical processes are complex or not fully understood. In addition, the purpose of modeling is often to inform decision makers with adequate timing. For example, models need to be run during and just before flood events. Real-time applications require rapid computation, which statistical methods are well suited to. The merits of statistical learning techniques, as a subset of empirical models, motivate their study in this dissertation.

1.6 Thesis Structure

This dissertation follows steps outlined in Figure 1.6. Chapter 2 compares two different data transformations that reflect our view of hydrology: (1) each basin is a separate function that transforms its inputs (precipitation and snow) into runoff (or unimpaired stream flow), or (2) basins are interconnected, and overlapping where one flows into another. Chapter 3 explores the effect of the loss functions on model estimates. Loss functions reflect the modeling objective. Chapter 4 compares different resampling methods for test error approximation. Chapter 5 estimates future hydrology with climate changed data. Chapter 6 discusses model improvement strategies.

Because model development is an iterative process, one can argue a different order to the chapters in this dissertation. However, we recommend reading them in the order they were presented here as it closely mimics the order of decisions made in modeling: pre-processing data, defining an modeling objective, validating/testing the model, and using the model in different, maybe unintended, ways.

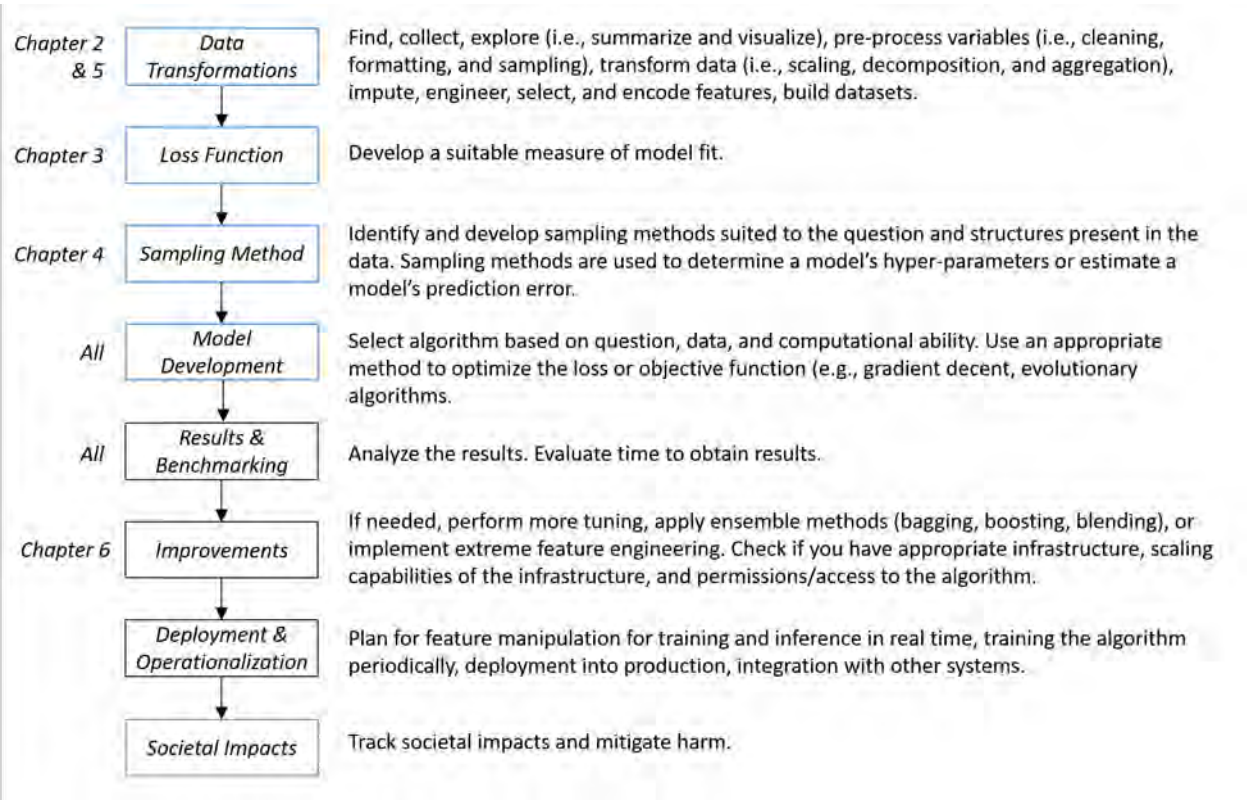


Figure 1.6: Statistical learning steps and thesis structure. Adapted from Brownlee, 2014; Ingle, 2017. Each chapter of this dissertation discusses a unique step.

Chapter 2

Data Transformations: Two Views on Hydrologic Processes

> Science is what we know, and philosophy is what we don't know.
>
> Bertrand Russell, *"Unpopular Essays"*, 1950

Summary

There are three distinct model types in hydrologic modeling: process-based, statistical, and theoretical. This dissertation develops statistical models (i.e., linear multivariate regression (LM), generalized linear regression (GLM), random forest (RF), and Neural Network (NN) models) with a typical least squares loss function. Loss functions are discussed in Chapter 3. The models are trained to predict calculated monthly unimpaired flows in 67 California basins, and their results are compared to a process-based model.

In hydrology, there are two ways of defining basins: "aggregate" and "incremental." Aggregate basins are basins where all the land that contributes to runoff at an outlet or a gauge is included within the basin boundary, and incremental basins are non-overlapping segments of the basin between two gauges. Most studies in the literature use the aggregate method, since it requires no pre-processing of the gauge data. This chapter compares these two views on hydrology and shows that the NN model with the incremental basin approach performed the best in the NSE criterion. The best overall error (Bias-Corrected Coefficient of Determination, bR^2=0.92, Nash-Sutcliffe Efficiency, NSE=0.97) reflects the model's ability to represent monthly variations in flow.

The test set error from "leave one group out" (LOGO) cross-validation shows that model quality in predicting unimpaired flow is variable in space. LOGO cross validation and other resampling strategies are discussed in Chapter 4. A comparison of different models concludes that the incremental basin approach to hydrologic modeling provides increasing benefits as the outlet of interest moves further downstream in the gauge network.

2.1 Introduction

Unimpaired flows can be presented in two fundamentally different ways: (1) **aggregate**: we can imagine each basin as a separate function that transforms its inputs (precipitation and snow) into runoff (or unimpaired flow). Flows for these basins are simply the observed gauge values (Figure 2.1a); or (2) **incremental**: we can imagine the basins as interconnected and overlapping. One stream flows into another, like in a network, and so, some basins overlap. Here, "incremental" basins are segments of basins that do not overlap. Flows for these basins are the amount not observed by gauges above the outlet of interest (Figure 2.1b). Therefore, when modeling with incremental flows, network information is preserved.

In both methods, regardless of how we draw the boundaries, basin characteristics are lumped. Therefore, this concept is similar to the concept of **Hydrologic Response Units**– the basic computational units assumed to be homogeneous in hydrologic response. Much like HRUs, incremental basins are smaller than the aggregate basin and can be dubbed "sub-basin" size.

This chapter compares these two types of data pre-processing: aggregate and incremental basins. Each data transformation reflects a way of viewing hydrologic basins and processes as independent or connected in a network. Throughout the dissertation, "hierarchies" determine the relative location of the gauges in the network. For example, hierarchies of 1 are gauges that do not have any gauges above them, hierarchies of 2 have one gauge above them, and so on. In this dissertation, hierarchies are different from the *Strahler stream order*; here, the gauges determine the hierarchy within the network, whereas all branches of a stream can have a Strahler stream order number regardless of whether they are gauged.

2.2 Methods

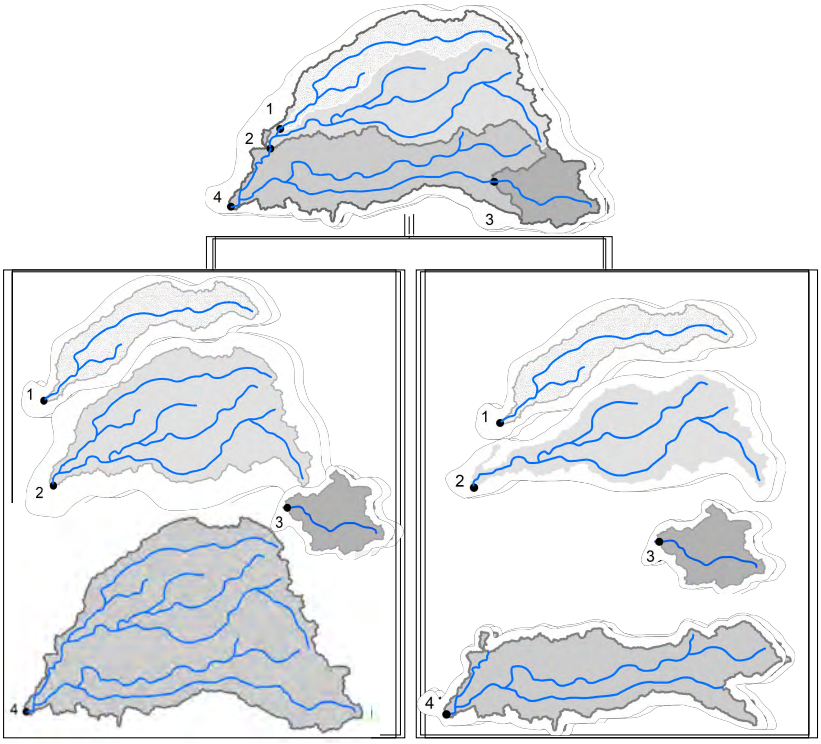
2.2.1 Model Types and Loss Functions

The choice of a suitable model relies on striking a balance between three desirable model properties: generality, reality, and precision (Guisan & Zimmermann, 2000). Generally, only any two out of these three properties can be improved simultaneously, while the third property must be sacrificed (Levins, 1966). This trade-off leads to at least two distinct models: **process-based**, **statistical**. Model selection should not solely rely on performance or fit statistics (precision); some models better reflect physical foundations in hydrology (reality) or are useful for a wider range of basins (generality). This dissertation develops different statistical models and compares the results to a process-based model (Table 2.1).

Linear Multivariate Regression Models

In 1805, Adrien Marie Legendre introduced the least squares method of estimating parameters as an appendix to his book on the paths of comets. A few years later, Carl Freidrich Gauss also published the method (Stigler, 1981). The method became widespread with its application to linear regression and curve fitting.

Linear Multivariate Regression models (LM) are customarily made of systematic and random error components, where the errors are usually assumed to have a Normal distribution



(a) Aggregate Basins (b) Incremental Basins

Figure 2.1: A basin's hydrologic response can be interpreted in two fundamentally different ways: (a) aggregate basins, where each basin's response is a function of all the land above the outlet that drains to it, or (b) incremental basins, where each piece of land below an outlet incrementally alters the observed flows from gauges above it.

(Equation 2.1).

$$\begin{aligned} &Y \sim N(\mu, \sigma^2)\text{: random} \\ &\mu = X\beta\text{: systematic} \end{aligned} \tag{2.1}$$

Given the model, the fitted values can be estimated by Equation 2.2.

$$Y_i^{sim} = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_i X_{ki} \tag{2.2}$$

The unknown parameters in Equation 2.2 are: β_0 (the overall mean) and β_k (the regression coefficients). To find the best fit, much like simple linear regression, we need to estimate the unknown parameters by minimizing a loss function, customarily the residual

Table 2.1: Model types and their parameters.

Model Type	R package	Parameters defined in model formulation	Parameters selected through cross validation
LM	stats	not applicable	not applicable
GLM	stats statmod	family=Tweedie link.power=0 maxit=1000	var.power=1.1
RF	randomForest	ntree=500 sampsize=length(training set) nodesize=5	mtry=20
NN	keras tensorflow	batch_size=25 validation_split=0.2	epochs=100

sum of squares (RSS) (Equation 2.3).

$$\begin{aligned} RSS &= \sum_{i=1}^{n} e_i^2 \\ &= \sum_{i=1}^{n} (Y_i^{obs} - y_i^{sim})^2 \\ &= \sum_{i=1}^{n} (Y_i^{obs} - \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_i X_{ki}) \end{aligned} \tag{2.3}$$

The `lm()` function in R constructs LMs. They are easy to understand and interpret, which makes them a great first cut at predictive modeling. However, they oversimplify reality (hydrologic processes are not linear) and lack precision or predictive ability (as demonstrated by poor goodness-of-fit measures). Another major flaw is that a linear predictor can give predictions that are physically impossible (e.g., negative flows). In PUB modeling, the variance cannot be considered constant since there is a boundary on the response. These shortcomings can be overcome with generalized linear models.

Generalized Linear Regression Models

In 1972, Nelder and Wedderburn introduced Generalized Linear Regression models (GLM). This work allowed for a unified fitting procedure, despite the type of error distribution, based on likelihood (Nelder & Wedderburn, 1972). Therefore, unlike LMs, GLMs can accommodate non-Normal distributions of error. However, except for Normal distributions, most other distributions do not have a closed-form solution.

In GLMs, the linear model is related to the response variable via a **link function**. This function allows the magnitude of the variance of each measurement to be a function of its predicted value. Therefore, a GLMs components are (Equation 2.4):

$$\begin{aligned} &Y \sim P(\mu, \phi)\text{: random} \\ &g(\mu) = X\beta\text{: systematic} \end{aligned} \tag{2.4}$$

Where P is the distribution of random errors, and $g(\mu)$ is the link function. P and g can be specified by the user.

The `glm()` function in R constructs GLMs. The GLMs developed here are characterized by the **Tweedie distribution**, since the outcome (i.e., unimpaired flow) is continuous, non-negative, skewed, and unbalanced with exact zeros. Tweedie distributions are a special case of exponential dispersion models where the variance function is a power function (Equation 2.5), and the link, or the function used to explain how the expectation of the outcome is related to the linear predictor can be specified in terms of Box-Cox transformations (Jorgensen, 1997).

$$var(Y) = V(\mu)\phi = \mu^{\alpha}\phi \tag{2.5}$$

The power, alpha, can be set by the user, or determined through cross-validation. Special cases include Normal (α=0), Poisson (α=1), Gamma (α=2), and inverse-Gaussian (α=3) GLMs . Here, we set the power α to be 1.1-found through cross-validation. The link g can be specified as log or identity. Here, we used the log link.

Therefore, the above model assumes that $y_i \sim Tweedie_{\alpha}(\mu_i, \phi)$ where

$$var(Y_i) = \mu_i^{1.1}\phi$$

and

$$log(\mu_i) = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_i X_{ki}$$

The regression coefficients, β_j, were estimated by maximum likelihood. The dispersion parameter, ϕ, was estimated using the RSS otherwise called the Pearson estimator.

Both LMs and GLMs are parametric models. For prediction purposes, non-parametric methods have potential to out-perform parametric methods, since their form is shaped by the data and not fixed a priori (James et al., 2013). Therefore, next, we consider a non-parametric modeling method, random forests.

Tree Building Algorithms

Classification and Regression Trees (CARTs) involve stratifying or segmenting the predictor space, into a several regions, using a series of if-then statements. At each internal node in the tree, a test is made to one of the inputs. Depending on the outcome of the test (or split rule), the algorithm goes to either the left or the right sub-branch of the tree. Eventually the algorithm arrives at a terminal branch, which is the prediction. The prediction for a given observation is the mean or the mode of the training observations in the region to which it belongs (Breiman, Friedman, Stone, & Olshen, 1984).

In essence, each tree is a series of split rules. The split rule is found using a **greedy** top-down search for recursively splitting of the data into binary partitions. It is greedy, because, the split rule at each internal node is selected to maximize the homogeneity of its child nodes, without consideration of nodes further down the tree, yielding only locally optimal trees (Grubinger, Zeileis, Pfeiffer, et al., 2011). For regression trees, the mean of all the observation points that fall within a branch is considered the prediction of that branch in the tree. The best tree is one which has the minimum test set error rate usually calculated by the RSS.

Since trees have a finite number of terminal nodes; CARTs are pruned based on a complexity parameter, α. The predictions of these methods are discrete, and therefore, not particularly suited to modeling a continuous variable. In addition, CARTs suffer from high variance; trees grown on different subsets of the training set will produce different predictions. This phenomenon is a major drawback of CARTs. Methods such as *bagging* (Breiman, 1996), *random forests* (Breiman, 2001), *boosting* (Friedman, 2001) and *bumping* (Grubinger, Kobel, & Pfeiffer, 2010) attempt to improve the prediction accuracy of trees with the idea that combining and averaging trees reduces variance.

A **Random Forest** (RF) consists of an assemblage of unpruned CART models and is essentially a weighted neighborhood scheme (Equation 2.6). Each CART model in a RF is different because it is grown using: (1) a new training set: in each bootstrapped training set, about one-third of the instances are left out; and (2) random feature selection: each time a split in a tree is considered, a random sample of predictors is chosen as split candidates from the full set of predictors.

$$\hat{y} = \frac{1}{m} \sum_{j=1}^{m} \sum_{i=1}^{n} W_j(x_i, x') \, y_i = \sum_{i=1}^{n} \left(\frac{1}{m} \sum_{j=1}^{m} W_j(x_i, x') \right) \, y_i \tag{2.6}$$

Using a random selection of features to split each node de-correlates the trees. Suppose there is one very strong predictor in the dataset, along with other moderately strong predictors. Then, in the collection of trees, most or all trees will use this strong predictor in the top split. Consequently, all trees will look quite similar. So, predictions from the trees will be highly correlated. However, forcing each split to consider only a subset of the predictors makes the resulting trees less variable and more reliable (James et al., 2013). This strategy introduces some randomness that improves the accuracy of the predictions of the trees as a whole and yields error rates that are robust with respect to noise (Breiman, 2001).

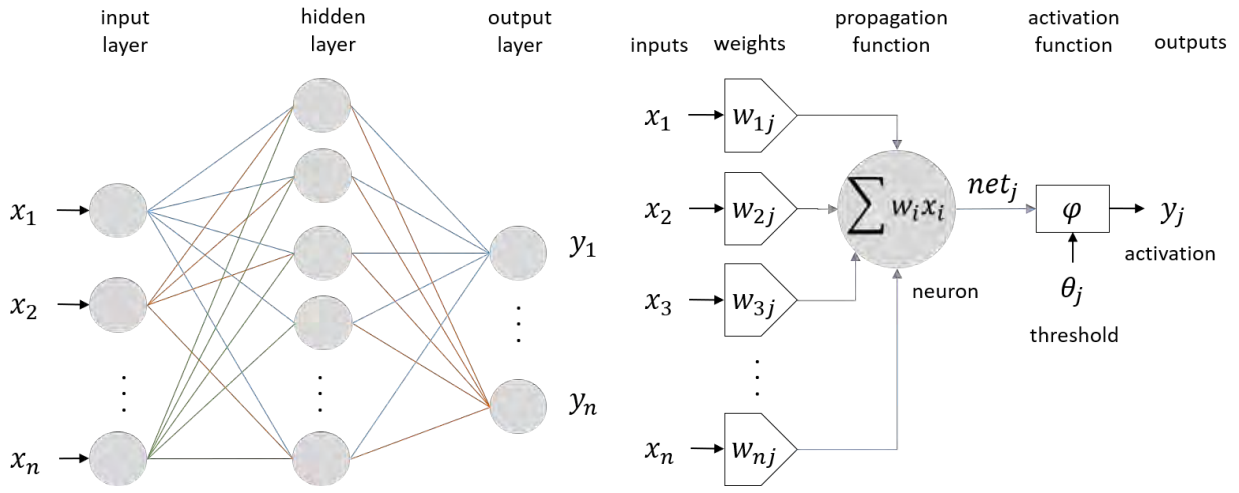
The `randomForest()` function in the `randomForest` library (Liaw & Wiener, 2002b) constructs RF models. This function takes in the following tuning parameters:

`mtry` or number of split features: In RFs, internal estimates monitor error, strength, and correlation, which are used to show the response to increasing the number of features used in the splitting. Here, this parameter was set to 20 out of the full 25 predictor variables available (found through cross-validation).

`ntree` or number of trees to grow: The generalization error of a forest of trees depends on the strength of the individual trees in the forest and the correlation between them (Breiman, 2001). This error converges to a limit as the number of trees in the forest increases. Here, the number of trees was kept at the default 500.

`sampsize` or sample size: In RFs, trees are built on a bootstrap sample of the training data, a sample equal in size to the original dataset, but selected with replacement. Therefore, some observations are not selected, and others are selected more than once. Here, the sample size was kept at the default value, the length of the training set.

`maxnodes` or maximum terminal nodes: Using the maximum number of terminal nodes, the user can "prune" the trees back to a smaller version. The default value was used, which is a function of `nodesize` or the allowed minimum number of observations in each node. The default value for nodesize is 5.



(a) Two Layer Dense Network (b) Operations of a Neuron

Figure 2.2: The anatomy of a neural network. (a) From left to right, the first layer is the layer in which inputs are entered, then an internal layer (called a hidden layer), and a final output layer. (b) At each node simple calculations are carried out: first, a weighted sum of all the inputs is calculated. An optional bias term can be added here. Then, an activation function maps the total value onto a number between 0 and 1. If the value is above the threshold value the neuron is considered active. Once the process is repeated for all the hidden layers, the last values obtained determine the output.

Like LMs, RFs also typically use the RSS loss function to find the optimal split value. Loss functions are examined in Chapter 3.

Neural Networks

In 1951, Marvin Minsky and graduate student Dean Edmonds built the first neural network (NN) machine. This machine was a randomly connected network of capacitors that have a finite amount of memory and time to keep or remember that memory. The memory holds the probability that a signal will come in one input and another signal will come out of the output. This machine, modeled after the Hebbian theory of learning in the human brain, was one of the first pioneering attempts at artificial intelligence. Shortly after, in 1957, Frank Rosenblatt invents the perceptron, the first **neural network** for computers.

Figure 2.2 shows the components of a neural network: the layers, nodes, activation function, and output layer transformations. **Deep learning** refers to neural networks with a higher number of hidden layers than the typical two layer fully connected network that is depicted in Figure 2.2a.

In training the model, a backward propagation of errors or **backpropagation** is used to establish the weights in Figure 2.2b. This method calculates the gradient of the error function with respect to the neural network's weights starting from the final layer and propagating backwards through the network. Since the error depends on the weighted sum and the weighted sum depends on the weight, the chain rule is used to estimate the error function's

partial derivative with respect to the weights (Equation 2.7).

$$\frac{\partial E}{\partial w_{ij}^k} = \frac{\partial E}{\partial {a^k}_j} * \frac{\partial {a^k}_j}{\partial w_{ij}^k} \tag{2.7}$$

where E is the loss function, w_{ij}^k is the weight for node j in layer k for incoming node i, a_j^k is product sum plus bias (activation) for node j in layer k.

Google's `TensorFlow`, and its accompanying application programming interface (API) `Keras`, allow for an easy application of neural networks with the following flexible parameters:

`activation`: The activation function determines whether a neuron will be activated. Here, we use the default Rectified Linear Unit (ReLU) function that is an activation function defined as the positive part of its argument also known as a ramp function.

`layer_dense(units=1)`: The units in the final layer define the number of outputs per observation. Here, we want one prediction per observation. Otherwise, we would have to average the predictions.

`epochs`: In batch training, the number of epochs is the number of times all training vectors are used to update the weights. The number of epochs determines the length of the training time. Here, we used 100 epochs since the weights typically stabilized before this number.

`batch_size`: Batch size determines the number of training examples in one forward or backward pass. The smaller the batch the less accurate the estimate of the gradient. However, minibatch methods reduce memory space needs and increase training efficiency as compared to using the entire sample. Here, we used 25 observations per training.

`validation_split`: The validation split helps automate the evaluation of the model's performance. Here, it was set to 0.2. However, since the data set was manually divided into testing and training sets, it was not necessary and the results from this test are not reported.

`optimizer`: Set to "rmsprop", the optimizer searches for the optimal node weights. The RMSprop optimizer is similar to the gradient descent algorithm with **momentum**. Momentum restricts the oscillation in one direction. Therefore, by increasing the learning rate, the algorithm takes larger steps in one direction to converge faster.

`loss`: Loss defines the error function and enables modelers to write custom loss functions. Here, we used keras::loss_mean_squared_error.

2.2.2 Test Set Error Approximation

Blocking cross-validation is used to approximate the test set error. All data for the basin to be modeled is left out of the training data and becomes the test set (i.e., leave one group out cross validation). Therefore, the training data is the data from all the other basins. This process was repeated for all basins in the study, and so, for model evaluation, one LM, GLM, RF, and NN model exists for each basin. Chapter 4 examines resampling methods. In all other chapters, the errors reported are test set errors.

With the developed model's predictions and the observations in the test set, we can calculate the desired model goodness-of-fit: Bias-Corrected Coefficient of Determination (bR^2) and Nash-Sutcliffe Efficiency factor (NSE) (Equations 2.8, 2.9, and 2.10). See appendix

D for more model measures-of-fit.

$$R^2 = \left(\frac{\sum_{i=1}^{n} \left(Y_i^{obs} - \overline{Y^{obs}} \right) \left(Y_i^{sim} - \overline{Y^{sim}} \right)}{\sqrt{\sum_{i=1}^{n} \left(Y_i^{obs} - \overline{Y^{obs}} \right)^2} \sqrt{\sum_{i=1}^{n} \left(Y_i^{sim} - \overline{Y^{sim}} \right)^2}} \right)^2 \qquad R^2 \in [0, 1] \tag{2.8}$$

R^2 is insensitive to additive and proportional differences between model simulation and observations. One can simply show that for a non-zero value of β_0 and β_1, if the predictions follow a linear form, $Y^{sim} = \beta_0 + \beta_1 Y^{obs}$, the R^2 equals one (Legates & McCabe Jr, 1999). Therefore, for a proper model assessment, it is recommended that the slope of the predicted vs. observed graph be reported or systematically included as in Equation 2.9.

$$bR^2 = \begin{cases} |b| \, R^2 & \text{for b} \leq 1 \\ |b|^{-1} \, R^2 & \text{for b} > 1 \end{cases} \qquad bR^2 \in [0, 1] \tag{2.9}$$

By weighting R^2, under and over predictions are quantified together with the model dynamics, which results in a more comprehensive reflection of model results.

Another commonly used model goodness-of-fit is the Nash-Sutcliffe Efficiency factor (Equation D.11).

$$NSE = 1 - \frac{\sum_{i=1}^{n} \left(Y_i^{sim} - Y_i^{obs} \right)^2}{\sum_{i=1}^{n} \left(Y_i^{obs} - \overline{Y^{obs}} \right)^2} = 1 - \frac{RSS}{Var(Y^{obs})} \qquad NSE \in (-\infty, 1] \tag{2.10}$$

A Nash-Sutcliffe efficiency factor of less than zero indicates that the mean value of the observed time series would have been a better predictor than the model. Like the bR^2, the largest disadvantage of the Nash-Sutcliffe efficiency factor is the differences between observed and predicted values are calculated as squared values. As a result, larger values in a time series are strongly overestimated whereas lower values are neglected (Legates & McCabe Jr, 1999). For the quantification of runoff predictions, this leads to an overestimation of the model performance during peak flows and underestimation during low flow conditions (Krause, Boyle, & Bäse, 2005).

2.2.3 Post-Processing

In post-processing, all model predictions are modified to be comparable to original gauge flows; all incremental basins are back-transformed into aggregate forms. In other words, the steps used in pre-processing are reversed to fairly compare goodness-of-fit across all models.

2.3 Results

2.3.1 Model Evaluation

Figure 2.3 shows predicted versus observed unimpaired flows, in the test set, for each model type, in order of increasing NSE. A perfect model would follow the $y = x$ line. The

regression line shows a tendency for the LM and RF aggregate and incremental models to underpredict and for the GLM aggregate and incremental models to slightly overpredict unimpaired flows. Underpredicting flows are generally bad in times of floods and overpredicting flows are generally bad in times of droughts, each misleading managers in damaging ways. The NN out-performs other models and is somewhat insensitive to the input data pre-processing.

Figure 2.4 shows how each model scores as to the bR^2 and NSE. In the LM and GLM the incremental modeling method performs better than the aggregate. In the RF and NN, their performances are very similar.

In the NSE, the NN incremental model has the best performance, so, we abandon further comparative analysis across model types and examine the spatial distribution of this model's performance.

2.3.2 Spatial Distribution of Errors

Figures 2.6 and 2.7 show the bR^2 values for the 67 basins in this study. As expected, the model's ability to predict unimpaired flow varies across California. The model performs better at larger basins lower in the network (i.e., have a higher hierarchy). This could be due to: (1) basins with higher hierarchies generally have larger flows and less variability (A.5), and the model is trained with a squared error loss that penalizes large errors more harshly; or (2) there is substantial value in having a gauge or flow information upstream. In other words, the decline in error is due to modeling with incremental basins.

Figures 2.8 and 2.9 show that when there is no flow information upstream (i.e., hierarchy=1) there is not much difference in the performance of incremental and aggregate models. However, when we increase information from upstream gauges (i.e., hierarchy=2,3,4, and 5) the NN incremental model can perform much better than the NN aggregate model. Even though the data set is smaller in the basins lower in the network, we can conclude that, the model is performing better at these basins due to information upstream and not just due to its higher flows and the loss function.

2.3.3 Benchmarking

Next, we compared the test set NSE of the NN models with that of the Basin Characterization Model (BCM), for basins that overlap the two studies (Figure 2.10). The BCM is a process-based model that calculates the hydrologic inputs and outputs of a specific landscape area; using climate data inputs a simple accounting solves the water balance for each cell. Model calculations include potential evapotranspiration, snow, excess water moving through the soil profile, actual evapotranspiration, and climatic water deficit—the difference between potential and actual evapotranspiration. Post-processing calculations are made to estimate baseflow, streamflow, and potential recharge to the groundwater system for watersheds (Flint & Flint, 2014).

The comparison in Figure 2.10 shows that the NN in both aggregate and incremental methods out-perform the BCM in most basins.

A model with higher predictive accuracy can produce more reliable information about the mechanism producing the underlying data, and weak predictive accuracy can lead to questionable conclusions (Breiman et al., 2001). Therefore, given a "good" model, interpretation methods can give more reliable insights into what would otherwise be a black-box

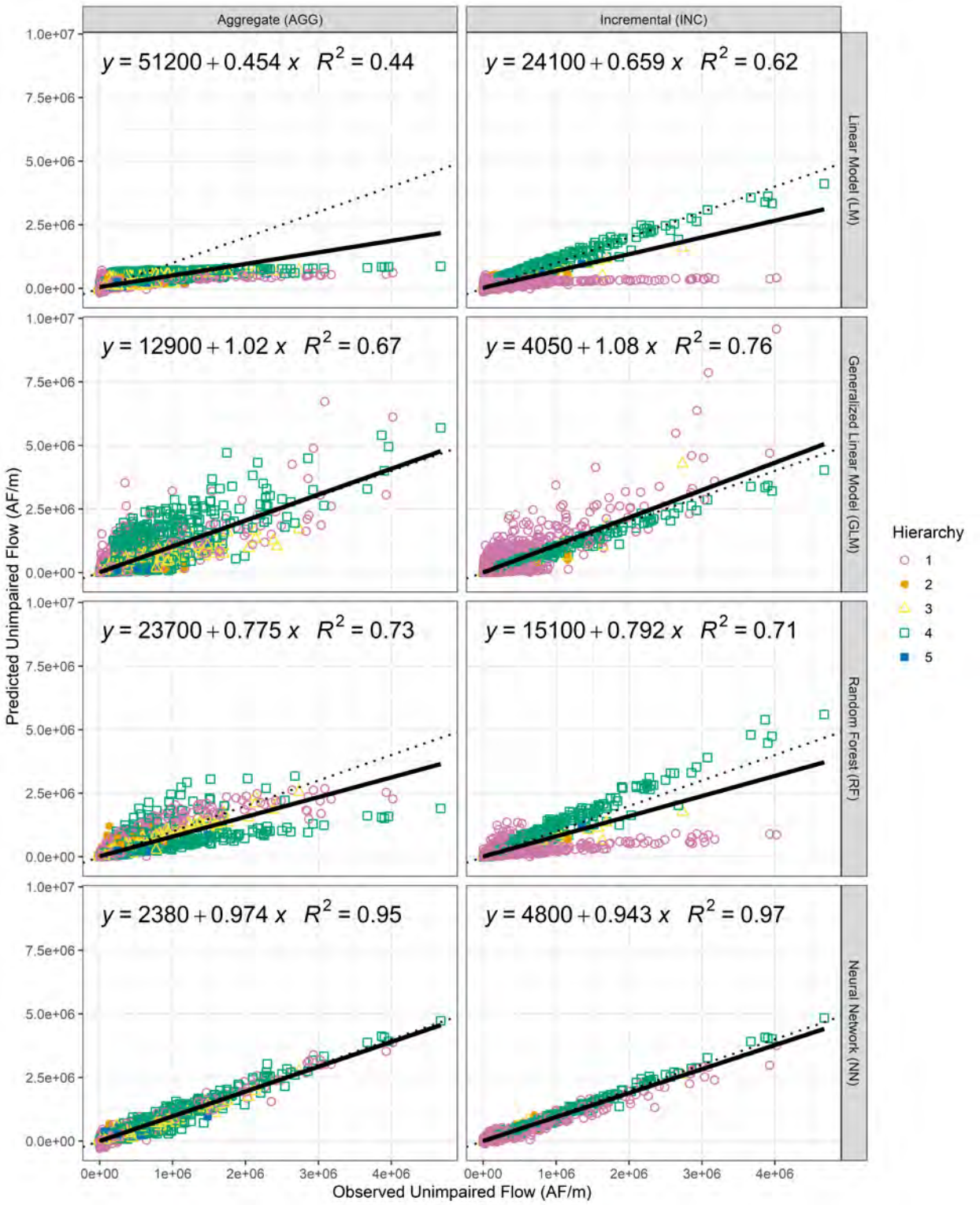
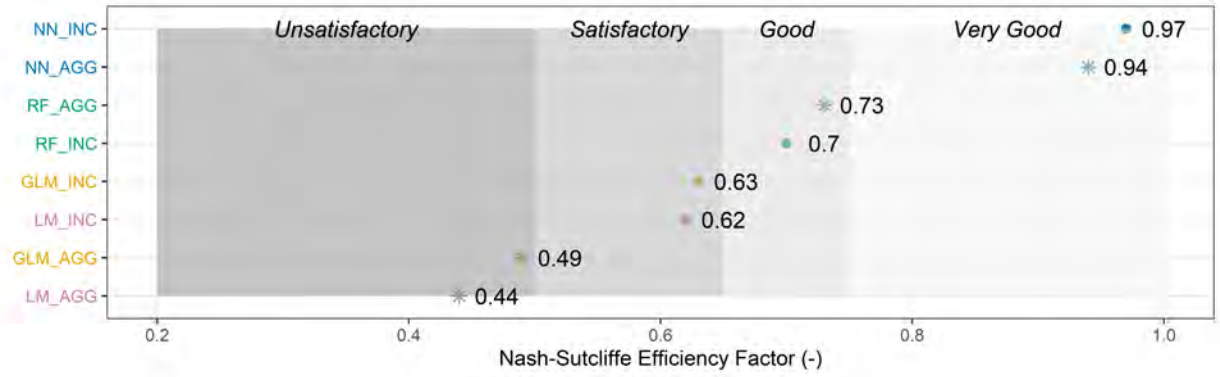


Figure 2.3: Predicted vs. observed results for models trained on aggregate and incremental data. Incremental (or networked) data leads to better model performance for most model types. The LMs generally underpredicts unimpaired flows and are a bad fit for lower order basins. The GLMs slightly overpredict unimpaired flows, but have a better fit. The RF generally underpredicts unimpaired flows and as a non-parametric model, has a better fit compared to the previous parametric models mentioned. The NN out-performs all models with an excellent fit for all flows and basin hierarchies.

(a) bR^2



(b) NSE

Figure 2.4: The goodness-of-fit of models trained on the two types of data (i.e., aggregate and incremental) as measured by the Coefficient-of-Determination (bR^2) and Nash-Sutcliffe Efficiency (NSE). The NN aggregate and incremental model provides the best model performance in the bR^2 and NSE respectively.

model. With the model measures of fit in mind, the following sections discuss two popular interpretability methods: variable importance and partial dependence for the NN aggregate model.

2.3.4 Variable Importance

It is often of interest to know how much the predictors in a fitted model influence the predictions.

In LMs and GLMs, the absolute value of the *t-statistic* is commonly used as a measure of variable importance (VI) although a score is assigned to each term in the model, rather than to each feature.

RFs and gradient boosted decision trees have a natural way of quantifying the importance or relative influence of each feature. In these models the data is sampled at each node. This sampling creates a leftover out-of-bag (OOB) data used to construct validation-set errors for each tree. To calculate VI, each predictor is randomly shuffled in the OOB data and the error is computed again. If a variable is important, then the validation error will increase when it is perturbed in the OOB data. The difference in the two errors is recorded and

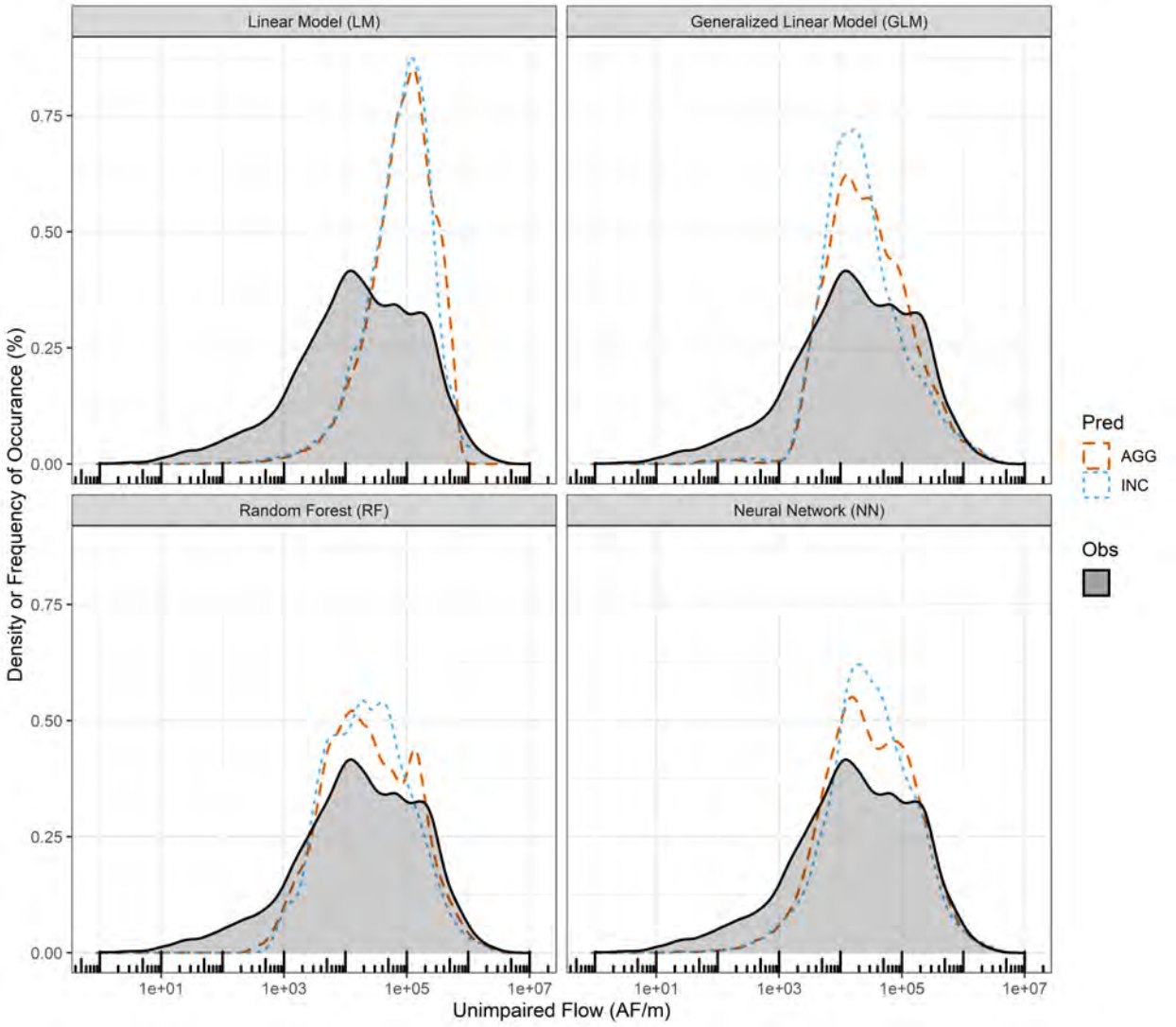
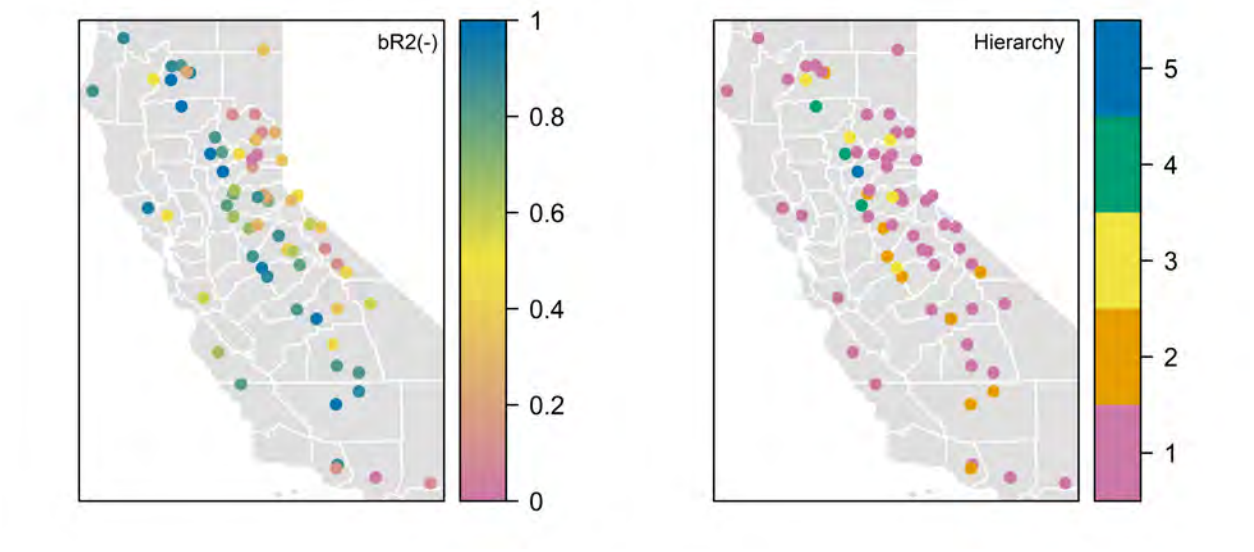
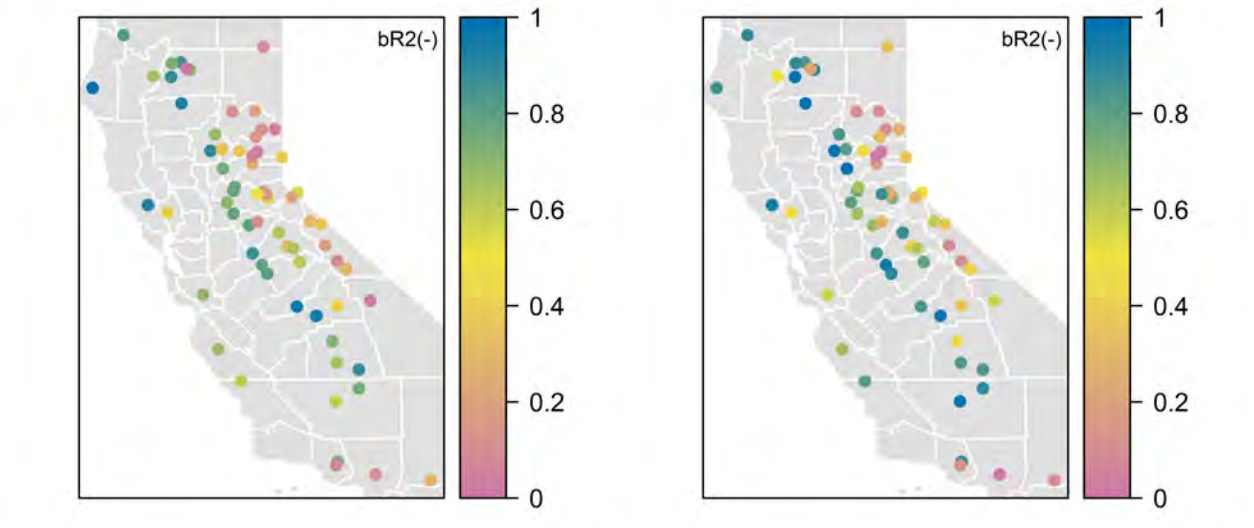


Figure 2.5: Residual error densities for aggregate and incremental models by model type. All models have a tendency to predict to the mean due to the MSE loss function. GLM, RF, and NN more accurately predict the frequency of "floods" (i.e. the right side tail). All models, to varying extents, fail to predict the frequency of "droughts" (i.e. the left side tail). LM is the least sensitive to the data transformation. In other models, the aggregate method better reflects the observed probability distribution compared to the incremental.



(a) NN Incremental (b) Basin Hierarchies

Figure 2.6: The spatial distribution of errors in the NN incremental model. (a) The bR^2 fit is not random and its higher values follow a line down the middle of California. This pattern coincides with higher basin hierarchies. (b) The basins are not evenly distributed between the hierarchies; the lower the hierarchy number the more basins in this study. Altogether, the lower the basin is in the network, the better the model performs.



(a) NN Aggregate (b) NN Incremental

Figure 2.7: The aggregate and incremental basins perform very similarly when there is no information upstream (i.e., hierarchy=1). However, when we introduce information upstream (i.e., hierarchy=2,3,4, and 5, which follows a line down the middle of California), incremental basins can perform much better than the aggregate.

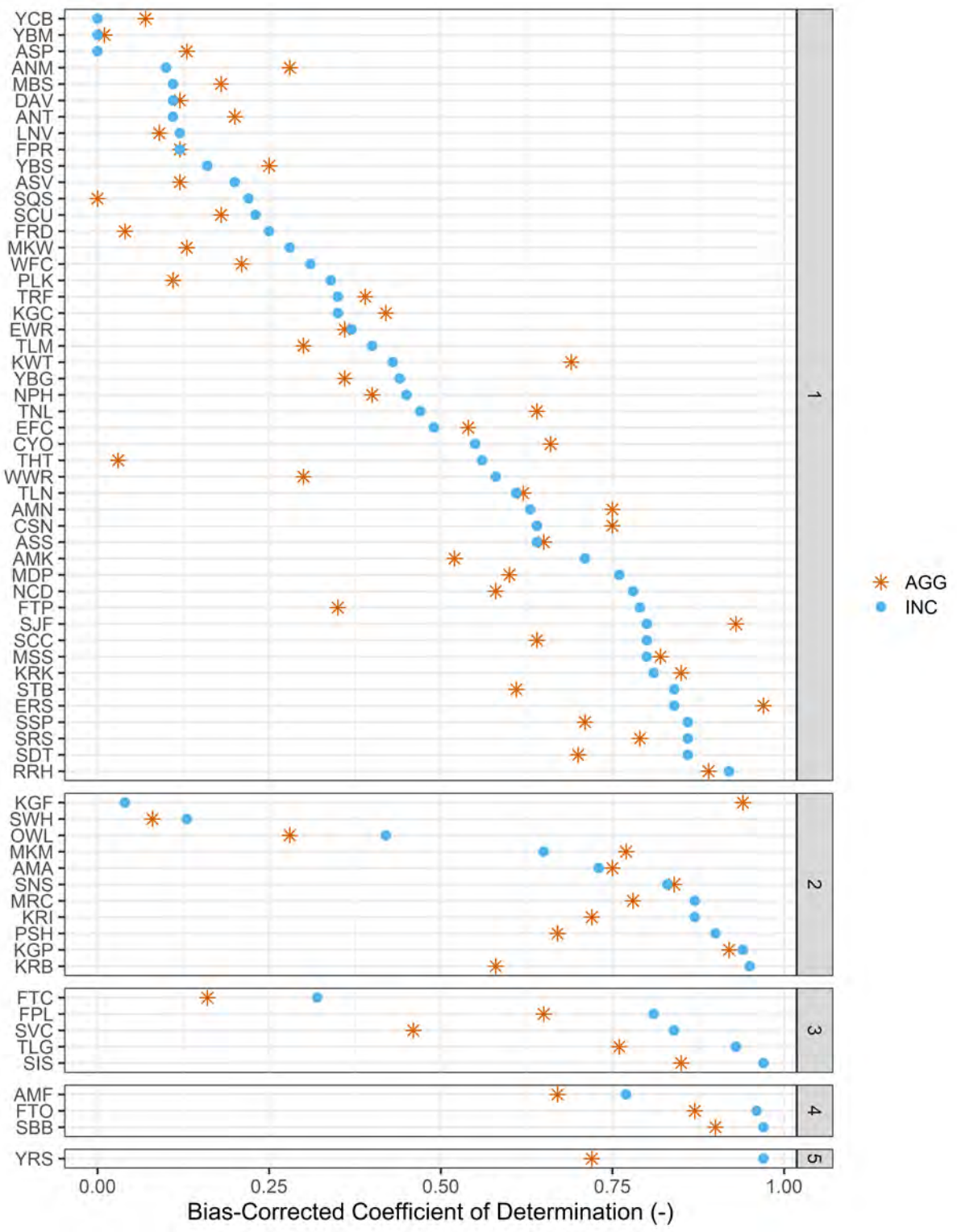


Figure 2.8: Basin bR^2 performance grouped by hierarchies for the NN model. The lower a basin is in the network, the better the incremental models perform compared to the aggregate. For example, the YRS basin (lowest in the network) has a $bR^2 = 0.97$ in the incremental model but a 0.72 in the aggregate.

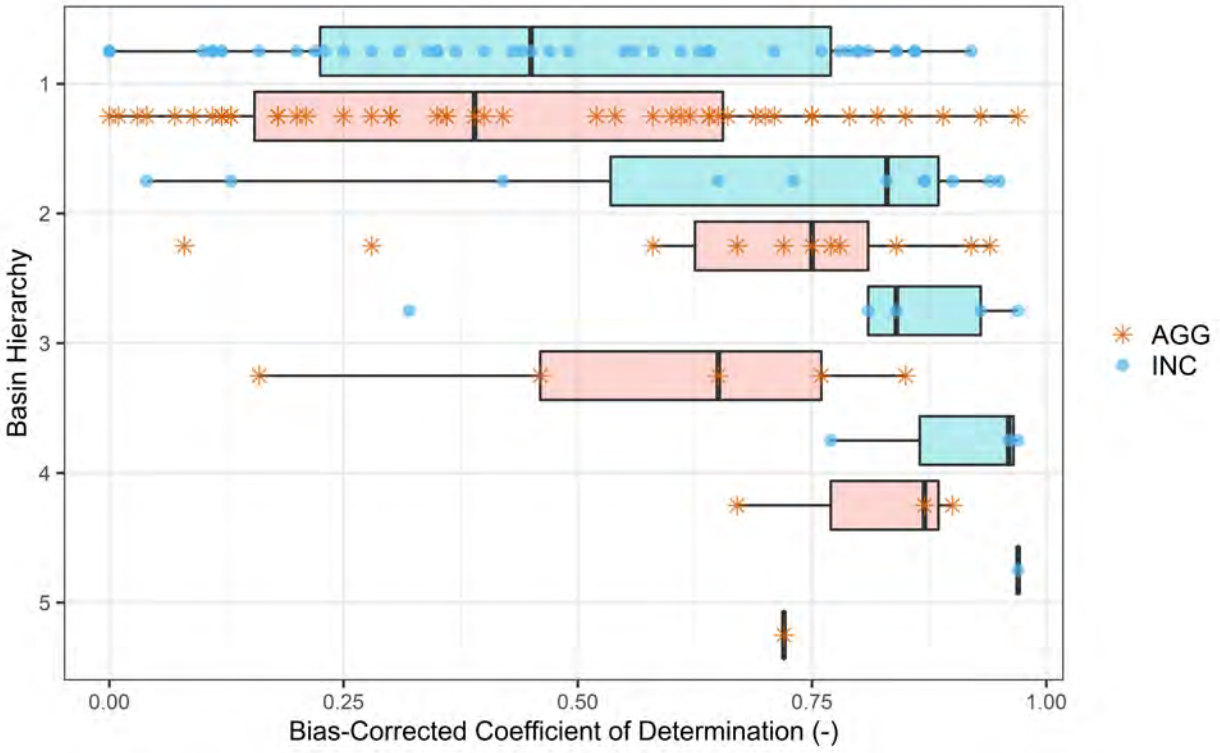


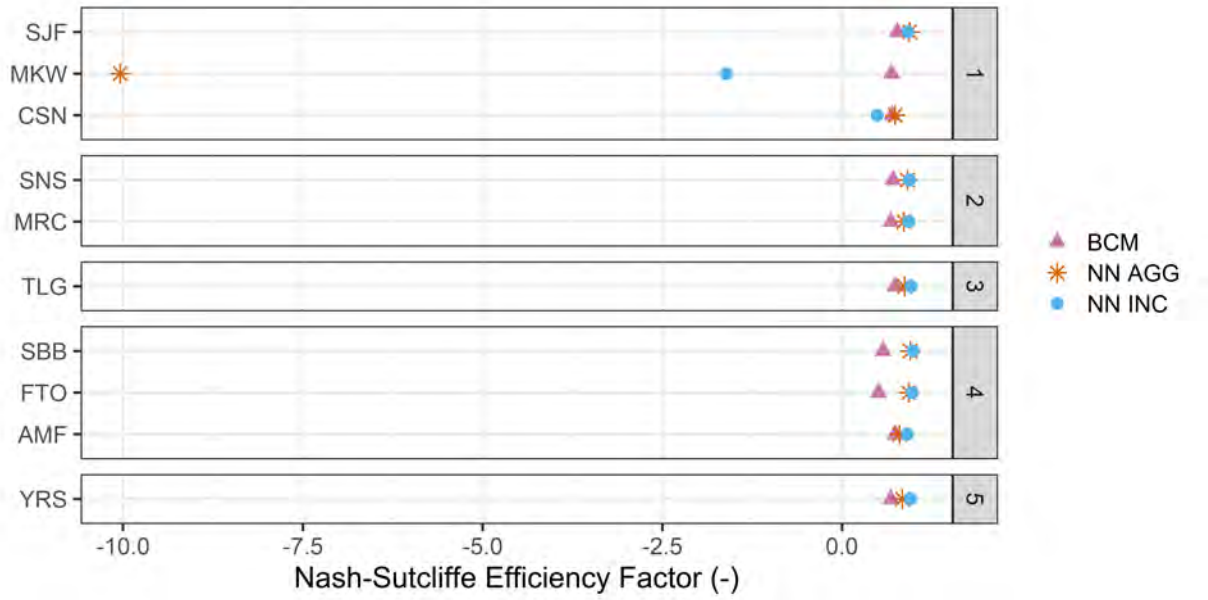
Figure 2.9: Basin bR^2 performance for the NN model. The incremental and aggregate basins perform very similarly when there is no information upstream (i.e., hierarchy=1). However, when we introduce information upstream (i.e., hierarchy=2,3,4, and 5) the incremental basins can perform much better than the aggregate.

averaged across all trees in the forest.

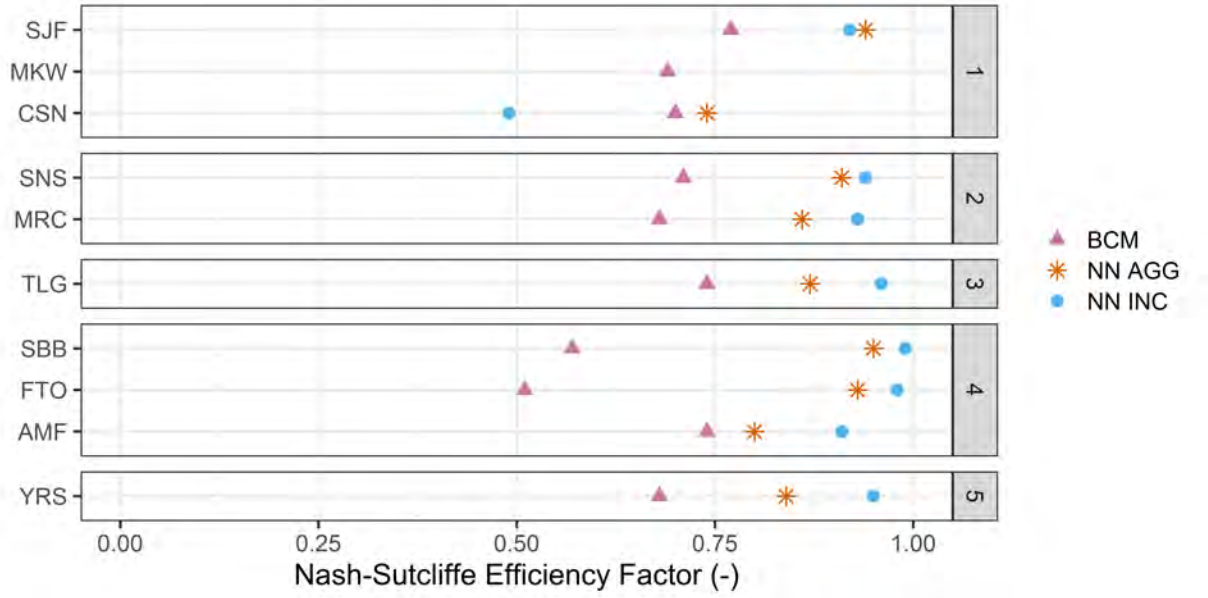
In NNs, two popular methods for constructing VI scores are the **Garson algorithm** (Garson, 1991), later modified by Goh (1995), and the **Olden algorithm** (Olden, Joy, & Death, 2004). In both algorithms, the basis of the importance scores is the network's connection weights.

If an algorithm does not have a natural way of calculating VI, or if the object is to compare VI across different model types, model-agnostic approaches are used. **Model-agnostic interpretability** separates interpretation from the model. One such method is **permutation**, a method popularized by Breiman (2001). The underlying principle in permutation methods is: if the values of an important feature is shuffled in the training data, the training performance would degrade, since permuting the values of a feature effectively destroys any relationship between that feature and the response variable.

The `vip` library constructs VI scores and plots for many types of supervised learning algorithms using model-specific and model-agnostic approaches. Here, we used the `permute` method and a user defined prediction function to manually block training and testing data by basins (i.e., the BBG method described in Chapter 4). The number of perturbations (nsim) was kept at the default, 1, to avoid long processing times. However, higher `nsim` values and averaging results can reduce the error introduced by the randomness in the permutation procedure.



(a) Full x-axis range.



(b) Truncated x-axis range.

Figure 2.10: Model goodness-of-fit comparisons with the Basin Characterization Model. The NN in both aggregate and incremental methods out-perform the BCM in eight of ten basins. The models in this study are especially weak in predicting MKW unimpaired flows, possibly, due to its very short length of record (only two years or 24 observations) compared to other basins (typically 32 years or 393 observations).

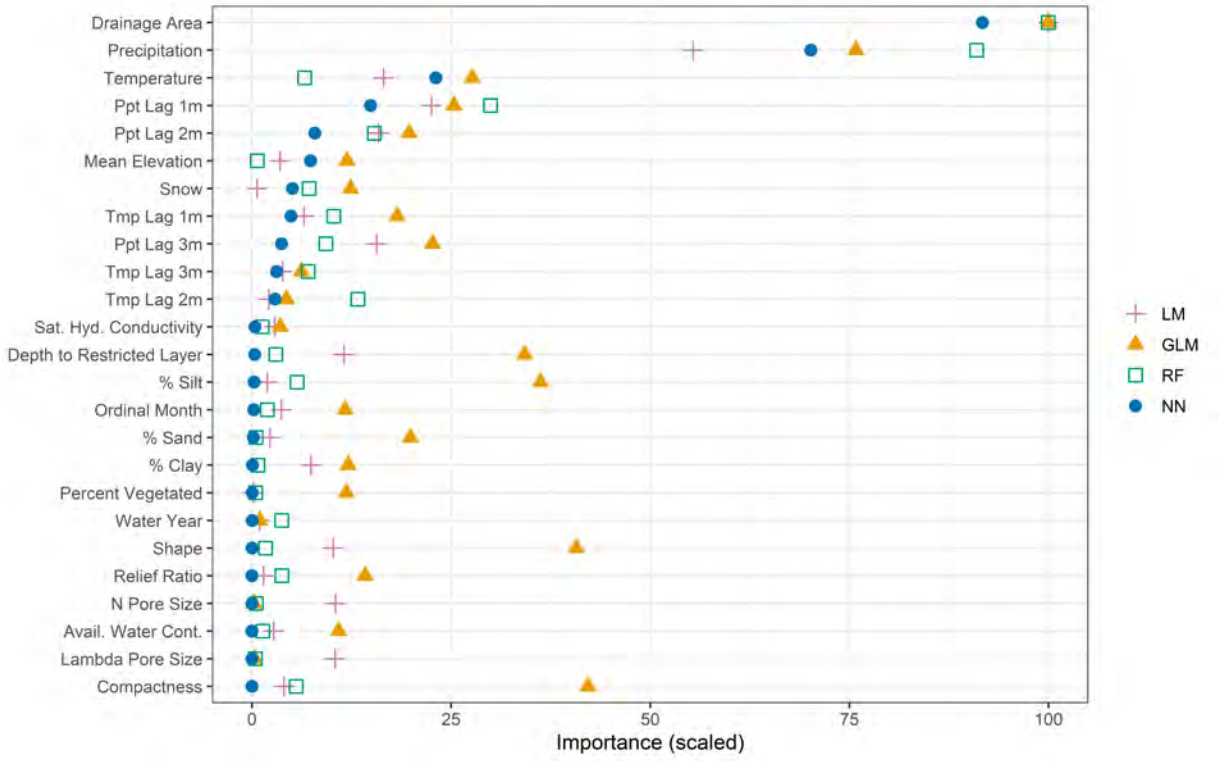


Figure 2.11: Scaled mean variable importance. A basin's drainage area and precipitation are the most important variables in predicting monthly unimpaired flow regardless of model type.

Figure 2.11 shows that a basin's drainage area and precipitation are the most important variables in predicting monthly unimpaired flow regardless of model type. Next are lagged precipitation and temperature variables, the snow variable manually calculated from precipitation and temperature, and the mean elevation of the basin. Other variables do not have significant effects on the model performance especially in the NNs.

As the addition of variables brings new information to models like the GLM, and improves their performance, the NN prefers extra layers in its network. The trend in the development of NNs has been to develop deeper networks, rather than wider ones, to improve model performance. The advantage of multiple layers is that they can learn features at various levels of *abstraction* (Antognini, 2016). For example, if you train a deep convolutional neural network to predict daily unimpaired flow, you may find that the first layer will train itself to recognize very basic things like the effects of drainage area and precipitation on the target. The next layer will train itself to recognize classifications and higher-order interactions like the different hydrologic regimes that may govern smaller headwater basins as opposed to larger ones, or that antecedent conditions can change the target. The flexibility in the depth of a NN means NNs can learn information from fewer variables.

Here, the presence of all predictor variables in the model can be justified with knowledge of hydrological processes. However, if this was not the case, **Stein's paradox** can describe an unavoidable issue. In estimation theory, Stein's paradox is a phenomenon that appears when three or more parameters are estimated simultaneously. In this case, a combined estimator

exists that is more accurate on average (i.e., has a lower MSE) than any method that handles the parameters separately (Efron & Morris, 1977). It implies that we can produce a better estimation of a parameter by simultaneously using three unrelated measurements. This occurs because the cost of a bad estimate in one component of the vector is compensated by a better estimate in another component. Since we do not know which parameter's estimate is improving, the best way to approach a given estimation problem is to make sure all predictor variables have been chosen with sound scientific reasoning.

2.3.5 Partial Dependence

Another model-agnostic interpretability method for quantifying feature importance is the **partial dependence plot** (PDP) and **individual conditional expectation** (ICE) curves. When constructing a PDP for a particular feature, the values of every observation in that feature is replaced with one unique observation. Then, the predicted values are averaged. This is repeated for each unique observation and the observation and predicted value pairs are plotted to become a PDP.

PDPs are valuable for understanding the relationships uncovered by complex models. However, they can be misleading in the presence of substantial interaction effects (Goldstein, Kapelner, Bleich, & Pitkin, 2015), which can be addressed with individual conditional expectation (ICE) curves. ICE plots display one line per observation that shows how the observation's prediction changes when a feature changes. The values for a line can be computed by replacing the feature's value with values from a grid, keeping all other feature the same, and making predictions with the model for these newly created instances. The result is a set of points for an instance with the feature values from the grid and their respective predictions. Consequently, the PDP for a predictor of interest can be obtained by averaging the corresponding ICE curves across all observations (Greenwell, Boehmke, & McCarthy, 2018).

Figure 2.12 shows the ICE curves for a few predictors in the YRS basin (the basin lowest in the network) as an example. Increasing drainage area, precipitation, and snow increases the unimpaired flow prediction. Interestingly, the same happens for temperature. Elevation and other variables with low VI scores (like % sand) have flat ICE curves showing their little influence over the predictions at any level (high or low percentages).

2.4 Conclusion

Incremental basin modeling provides an easy way to include network information in statistical models and the results show its value for modeling hydrology with parametric models, especially those with few parameters like LM and GLM. As the results showed, the LM and GLM prefer the incremental modeling approach, whereas the RF and the NN are somewhat insensitive to it.

On this data set, and according to the performance ratings provided by Moriasi et al. (2007), the GLM and RF provide a "good" prediction for unimpaired flows, and the NN provides a "very good" one (Table 2.2). The RF might perform well due to the nature of non-parametric methods where the model form is determined by the data and prediction becomes easier. The NN performed the best possibly due to its learning by abstraction and its ability to mimic threshold behaviors in hydrology. The results from the NN models show

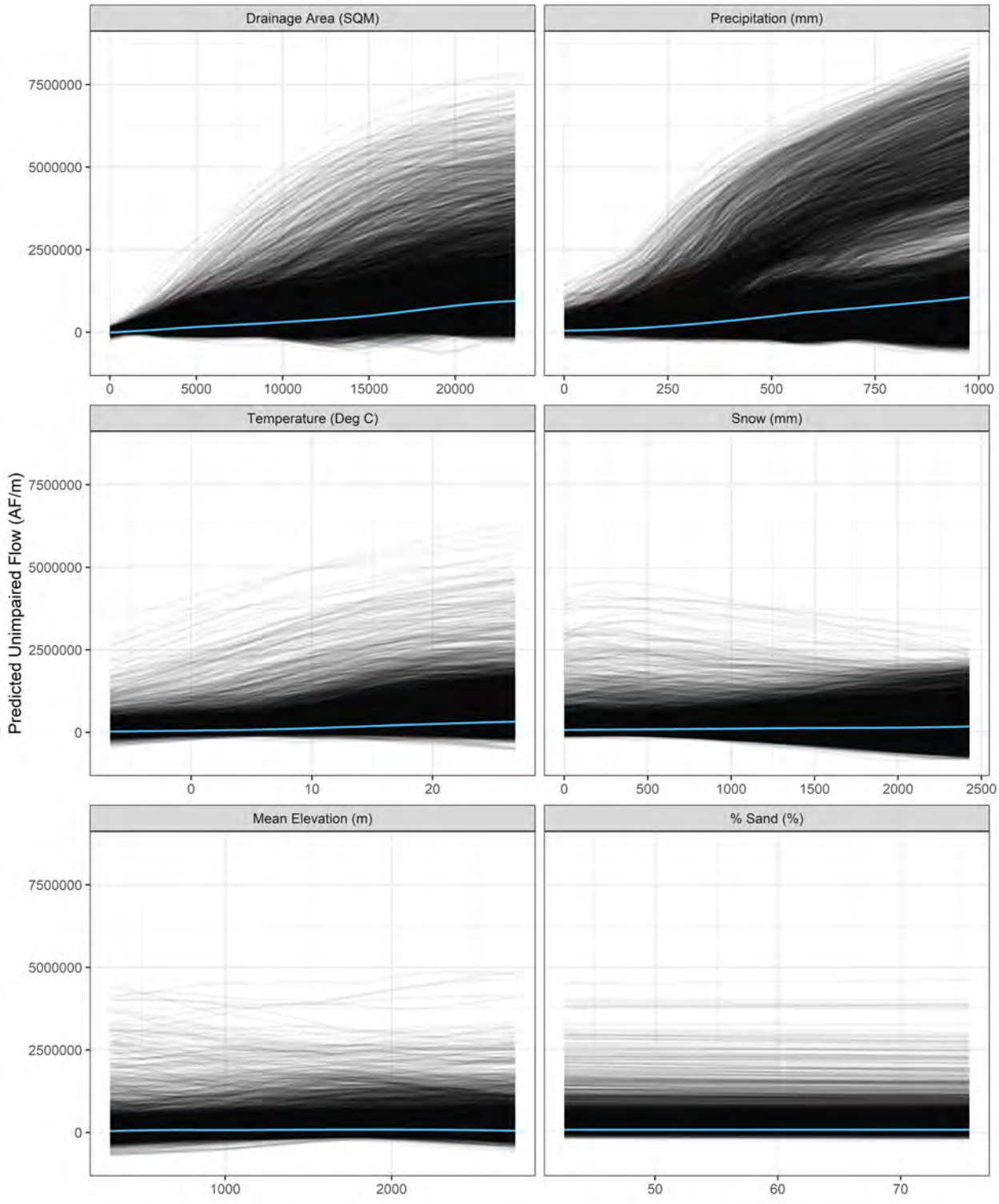


Figure 2.12: Individual conditional expectation (ICE) curves and their averages (partial dependence) for the YRS basin. The flatness of the ICE curves show the influence of the predictor variable in predicting unimpaired flow. For example, increasing the precipitation that falls on the basin increases the amount of unimpaired flows predicted by the model. However, increasing the % sand for each observation does not have an effect on unimpaired flow predictions at any level (low or high percentages).

why these methods are so popular in hydroinformatics.

Table 2.2: Model performance ratings. Criteria are given by Moriasi et al., 2007 (Appendix D).

Model	Aggregate	Incremental
LM	Unsatisfactory, NSE=0.44	Satisfactory, NSE=0.62
GLM	Unsatisfactory, NSE=0.49	Satisfactory, NSE=0.63
RF	Good, NSE=0.73	Good, NSE=0.70
NN	Very Good, NSE=0.94	Very Good, NSE=0.97

In another experiment, the models were trained on their cumulative flows and cumulative rainfall. Given that snow-melt driven hydrology dominates the Sierra-Nevada basins, and processing the data to its cumulative forms would have given the model a "memory" effect, we repeated the experiment in this chapter with cumulative values. However, surprisingly, none of the models provided satisfactory results, and so, we have omitted those results from this chapter.

The next chapter explores one flaw pointed out here: the squared loss function forcing better predictions at flood levels at the expense of drought level data. Models are trained and compared using different asymmetric loss functions to penalize the under predicting of floods and overpredicting of droughts at a higher cost (i.e., forcing the model to reach the peaks and valleys of the hydrograph).

Chapter 3

New Loss Functions: Comparing Asymmetric and Symmetric losses in Evaluating Error

> Maybe all one can do is hope to end up with the right regrets.
>
> Arthur Miller, *"The Ride Down Mount Morgan"*, 1991

Summary

In statistical learning, the loss function is a translation of an informal philosophical modeling objective into the formal language of mathematics (Hennig & Kutlukaya, 2007). Some measures of fit have already been established and are common in hydrologic modeling (e.g., Mean Squared Error [MSE], Nash-Sutcliffe Efficiency [NSE]). However, these loss functions are all symmetric. The MSE is the loss function of choice in most modeling efforts, because it is mathematically easier to implement (differentiable) and is the default loss in many functions imported from libraries.

Symmetric functions produce the same loss when underpredicting and overpredicting of the same absolute error. However, an asymmetric loss function applies a different penalty to the different directions of loss. This feature allows an asymmetric loss function to force the model to overpredict unimpaired flows in times of floods and underpredict them in droughts rather than the less desirable opposite. This approach leads water managers to more conservative decisions, since the models predict more extreme floods and droughts.

This chapter uses six loss functions. Four are symmetric: Mean Squared Error (MSE), Log Hyperbolic Cosine (LOGCOSH), Mean Absolute Error (MAE), Mean Squared Percentage Error (MSPE); and two are asymmetric: Mean Weighted Least Squares Error (WLSE) and Linear Exponential Error (LINEXE). We present the results obtained by a NN model with a LOGO cross-validation resampling scheme. A visual comparison of the model fits shows that the asymmetric functions, with appropriate nuisance parameter estimates, can force better fits at the peaks and valleys of a hydrograph.

3.1 Introduction

3.1.1 Loss Functions in Statistical Learning

Typical loss functions in statistical learning are the ℓ_1-norm and ℓ_2-norm (Equations 3.1 and 3.2). The ℓ_2-norm is the familiar objective function in simple least-squares regression, a convex function, emphasizing points distant from the bulk of the data.

$$\ell_1(y_i, \hat{f}(x_i)) = ||y_i - \hat{f}(x_i)||_1 = \left|y_i - \hat{f}(x_i)\right| \tag{3.1}$$

$$\ell_2(y_i, \hat{f}(x_i)) = ||y_i - \hat{f}(x_i)||_2^2 = \left(y_i - \hat{f}(x_i)\right)^2 \tag{3.2}$$

Risk, or cost, is defined as the expectation of the loss function. For example, the risk of overpredicting the severity of a drought can be defined as *how much* it was overpredicted on average. This distance can be defined as the absolute value of the difference or the difference squared as in Equations 3.3 and 3.4, the empirical risks associated with the ℓ_1-norm and ℓ_2-norm. The expectation of the ℓ_2-norm will produce a model that regresses to the mean, and the ℓ_1-norm regresses to the median. That is, the ℓ_2-norm is more sensitive to outliers than the ℓ_1-norm. Using either norm implies that the modeler is more concerned with a conservative measure of centrality rather than getting predictions closer to the extremes of the distribution. Asymmetric loss functions discussed later can address this issue.

$$L_1(y_i, \hat{f}(x_i)) = E\left[\ell_1(y_i, \hat{f}(x_i))\right] = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{f}(x_i)\right| \tag{3.3}$$

$$L_2(y_i, \hat{f}(x_i)) = E\left[\ell_2(y_i, \hat{f}(x_i))\right] = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 \tag{3.4}$$

As an aside, **regret** is the difference between the consequences of a sub-optimal decision and the optimal decision. Often, in reinforcement learning, the objective is to minimize total regret, which is equivalent to maximizing the highest accumulated reward (Sutton & Barto, 2018). For example, maybe overpredicting the severity of a drought this year will lead to better management of resources and fewer regrets in later years. To avoid developing a mathematical representation for regret, we will proceed with the much simpler **risk-minimization framework** (Equation 3.5).

$$\hat{f}(x_i) = \underset{\tilde{f}}{\operatorname{argmin}}\ E\left[L(y, \tilde{f}(x))\right] \tag{3.5}$$

3.1.2 Loss Functions in Hydrologic Modeling

In practice, the loss function for a chosen statistical learning method is the translation of a informal philosophical modeling objective into the formal language of mathematics (Hennig & Kutlukaya, 2007). So, the choice of a loss function in estimation is somewhat subjective and depends on the specific application of the model or the decisions being made when used.

Mechanistic models in hydrology simulate conditions based on available input parameters, modeled processes, and calibration to specific locations. **Measures of fit**-the similarity of the simulations to the observations-help in assessing model performance. Visual similarity is recommended as the most fundamental approach to assessing model performance (i.e., the plot of observed and simulated time series), and calculated measures of fit are recommended next as an objective assessment (Krause et al., 2005). In addition to model performance estimation, these measures can help guide better fits of simulations to observations in model calibration or "nuisance" parameter estimation.

Some measures of fit have already been established and are common in hydrologic modeling: the Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Square Error (RMSE), normalized RMSE (nRMSE), RMSE standard deviation ratio (RSR), Relative Standard Deviation (RSD), Relative Mean (RMU), Percent Bias (PBIAS), Coefficient of Determination (R^2), Nash-Sutcliffe Efficiency (NSE), Index of Agreement (d), Modified NSE, Modified d, Relative NSE, Relative d, King-Gupta Efficiency (KGE), and Volumetric Efficiency (VE). Appendix D presents their equations, strengths, and weaknesses.

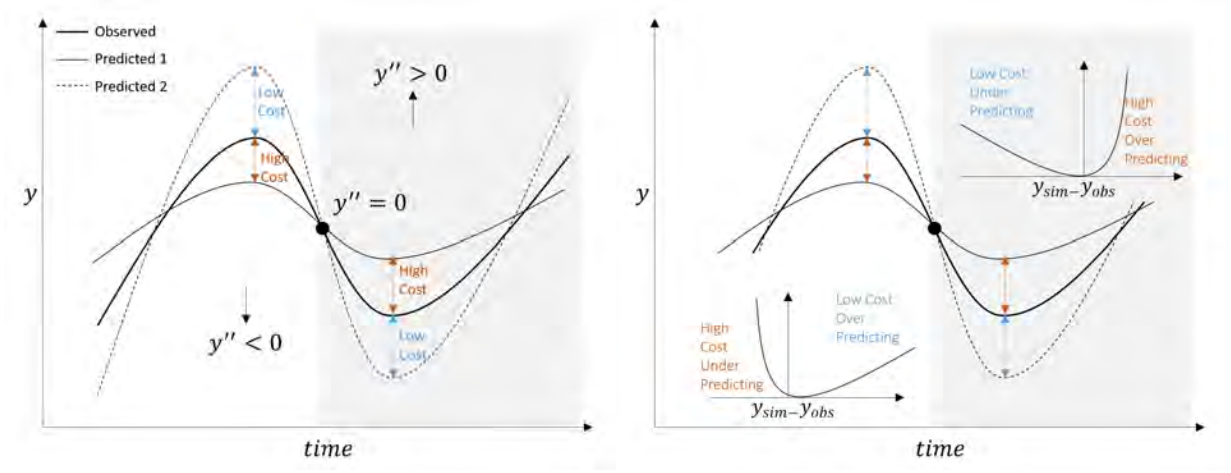
The following is a discussion on the characteristics of the loss function in its application to hydrologic prediction:

(1) **Symmetric** vs. **Asymmetric**: In symmetric functions, underpredicting produces the same loss as overpredicting of the same absolute error. However, a conservative loss function applies a different penalty to the different directions of loss (overpredicting vs. underpredicting). So, an asymmetric loss function can force the model to over predict the unimpaired flows in times of floods and under predict them in droughts rather than the less desirable opposite. This approach requires the labeling of all instances of the data as either a peak (flood), or valley (drought) point. Therefore, we need a labeling mechanism (i.e., a classification model) before fitting the predictive regression model.

Great care should be taken not to introduce "data leakage" or the inclusion of information from the response variable into the training of the predictive model; the classification model will have to either be trained on the predictor variables only, or use a portion of the data that is set aside for the rest of the study.

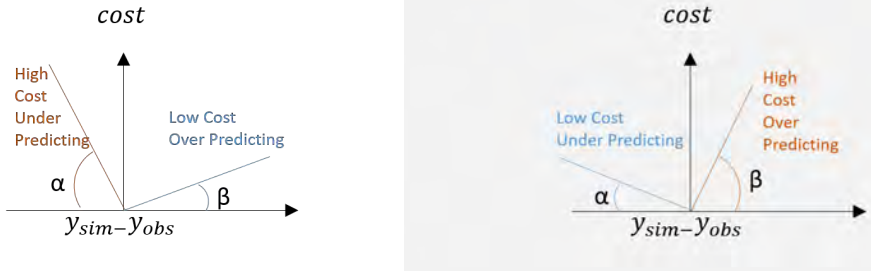
Once all observations are classified, we define different loss functions for each peak and valley section. Such loss functions can be defined as linear exponential (LINEX) loss if smoothness is desired (Equation 3.6 and Figure 3.1). However, current subgradient-based and derivative-free methods of optimization in convex programming can easily handle non-differentiability at the origin of the loss function. Many asymmetric loss functions in machine learning have a simple **kink** in them. They are otherwise entirely differentiable (Equation 3.7 and Figure 3.2).

$$LINEX(y_i, \hat{f}(x_i)) = e^{\phi(y_i - \hat{f}(x_i))} - \phi(y_i - \hat{f}(x_i)) - 1, \quad \phi \in \mathbb{R} \tag{3.6}$$



(a) Desired asymmetricity in peaks and valleys (b) Asymmetric loss defined with LINEXE

Figure 3.1: Asymmetric loss functions define different losses to overpredicting and underpredicting a value. A positive ϕ defines the LINEXE needed for drought cases and a negative ϕ for flood cases.



(a) Floods (b) Droughts

Figure 3.2: An asymmetric weighted absolute value loss function can define different slopes to overpredicting and underpredicting a value.

$$Hinge(y_i, \hat{f}(x_i)) = \alpha * min(0, \hat{f}(x_i) - y_i) + \beta * max(0, \hat{f}(x_i) - y_i) \tag{3.7}$$

The squared error loss penalizes larger errors more than smaller error; the function is steeper in the tails than in the middle. To preserve this feature, we can combine the concepts above and define a weighted ℓ_2-norm (Equation 3.8).

$$Weighted\ Squared\ Error(y_i, \hat{f}(x_i)) = \alpha * \left[min\left(0, (\hat{f}(x_i) - y_i)\right)\right]^2 + \beta * \left[max\left(0, (\hat{f}(x_i) - y_i)\right)\right]^2 \tag{3.8}$$

(2) **Relative Errors** vs. **Absolute Errors**: In hydrology, manual and automatic attempts aimed at minimizing absolute errors often lead to fitting the higher portions of the hydrograph (peak flows) at the expense of the lower portions (baseflow) (Krause et al., 2005). Relative errors are generally more important than absolute errors unless the goal is to estimate water supply. For example, depending on the modeling purpose, a 100 TAF error in 1,800 TAF (monthly annual average of the Sacramento River) could be a less extreme error than in 300 TAF (monthly annual average of the Trinity River). Relative error loss functions or a simple log transformation of the data can help in this regard. We have used one relative error loss function, the Mean Squared Percentage Error (MSPE), for illustrative purposes.

(3) **Continuous** vs. **Stepwise**: Although, most outcomes may follow a discontinuous step function (e.g., a neuron firing or not), many decisions in water resources (e.g., releases from a reservoir) are continuous. Continuity and differentiability make the math more convenient. One major development in neural networks was doing away with the concept of thresholds in the step function (representing the collective influence of all the inputs) and replacing it with a smoother *sigmoid* function. As with neural networks, many optimization algorithms require continuity and differentiability (e.g., gradient decent). However, advances in these methods now allow for piece-wise differentiability in the loss function. To make matters simple, we use continuous functions.

(4) **Homogeneous** vs. **Heterogeneous (i.e., weighted based on geographic region)**: The cost of incorrectly managing a densely populated urban basin may be very different than a desert or a headwater basin; the importance of having accurate flow estimates is not completely homogeneous especially across a big and diverse region like California. However, to avoid making those judgments, we use a single loss function across all regions.

3.2 Methods

Table 3.1 shows the loss functions used in the NN model.

Table 3.1: Loss functions used in NN model.

Type*	Abb.	Name	Function
S	MSE	Mean Squared Error	keras::loss_mean_squared_error()
S	LOGCOSH	Log Hyperbolic Cosine	keras::loss_logcosh()
S	MAE	Mean Absolute Error	keras::loss_mean_absolute_error()
S	MSPE	Mean Squared Percentage Error	custom_metric (Appendix E)
A	WLSE	Mean Weighted Least Squares Error	custom_metric (Appendix E)
A	LINEXE	Linear Exponential Error	custom_metric (Appendix E)

* S: Symmetric; A: Asymmetric

In asymmetric loss functions, a simple classification model is needed to label points as flood ("FLOOD"==1) and drought ("FLOOD"==0). For each basin, the mean precipitation across the full record was designated a hard threshold; if the precipitation of a given month fell below this value, that observation was designated a "drought" and if above, a "flood".

Given this designation, we can apply different losses to the prediction error at different locations in the hydrograph.

Another consideration had to be made for defining loss functions in `keras`. Losses in `keras` can accept only two arguments: y_true and y_pred, which are the target tensor and model output tensor, respectively. However, if we desire the loss to depend on other tensors-as is the case with asymmetric losses-we are required to use **function closures**. Here, the loss function takes in whatever arguments we desire and returns the function that only depends on y_true and y_pred. Hence, the name *wrappers*. Code snippets in Appendix E for the WLSE and LINEXE wrappers show how this is accomplished.

Also, since we now have labeled observations (floods or droughts), we need this designation to line up with each y_true and y_pred correctly. Therefore, we can no longer use **minibatch** training methods that scramble the data without significantly changing the scrambling algorithm to accommodate labels. The size of the minibatch is determined by the validation split (e.g, 0.2) and only aids in speeding up the model training. To still make accurate predictions without minibatch, we simply increased the training epochs from 100 to 1000. Note that in this case, `shuffle`=FALSE.

3.3 Results

3.3.1 Model Evaluation

Figure 3.3 shows the visual fit of the time series. As you scroll through the basins, you can see the results from the asymmetric loss functions stretched in the y direction in contrast with the symmetric functions. Therefore, the asymmetry introduced is having the intended effect at the peaks and valleys of the hydrograph.

Figure 3.4 shows the predicted vs. observed data for models built with different loss functions. There is very little difference observed between the MSE, LOGCOSH and MAE methods. The LOGCOSH or log(cosh(x)) is approximately $x^2/2$ for small x and $abs(x) - log(2)$ for large x. Therefore, the LOGCOSH works much like the MSE, but will be less affected by occasional wildly incorrect predictions and in this regard is like the MAE. Therefore, unsurprisingly the slope, β_1, in the fitted linear equation for LOGCOSH falls in between that of the MSE and MAE ($0.909 < \beta_1 = 0.919 < 0.959$).

In addition, Figure 3.4 shows that the MSPE greatly underpredicts observations. Relative loss functions put a heavier penalty on negative errors than on positive errors. In other words, equal errors above the actual value result in a greater absolute percentage error than those below the actual value (Makridakis, 1993). So, MSPE produces predictions that are biased low.

In general, Figure 3.4 shows the asymmetric loss functions generally underpredict the largest floods but overpredict lower flood values as shown in the higher values of the y-intercept, β_0, in the fitted linear equation (β_0=29, 31 TAF $>$ 10, 7, 8, 4 TAF).

Figure 3.5 shows the MSE perform the best in the bR^2. This is expected since both the MSE and bR^2 metrics are calculating similar squared errors. As mentioned before, the LOGCOSH in its mathematical formulation is similar to the MSE for small errors and the MAE for large errors. Therefore, unsurprisingly, it performs similarly to the MSE and MAE in the bR^2. The two asymmetric losses perform similar to each other, which reassures us that the form of the asymmetric function has less impact on model goodness-of-fit than it

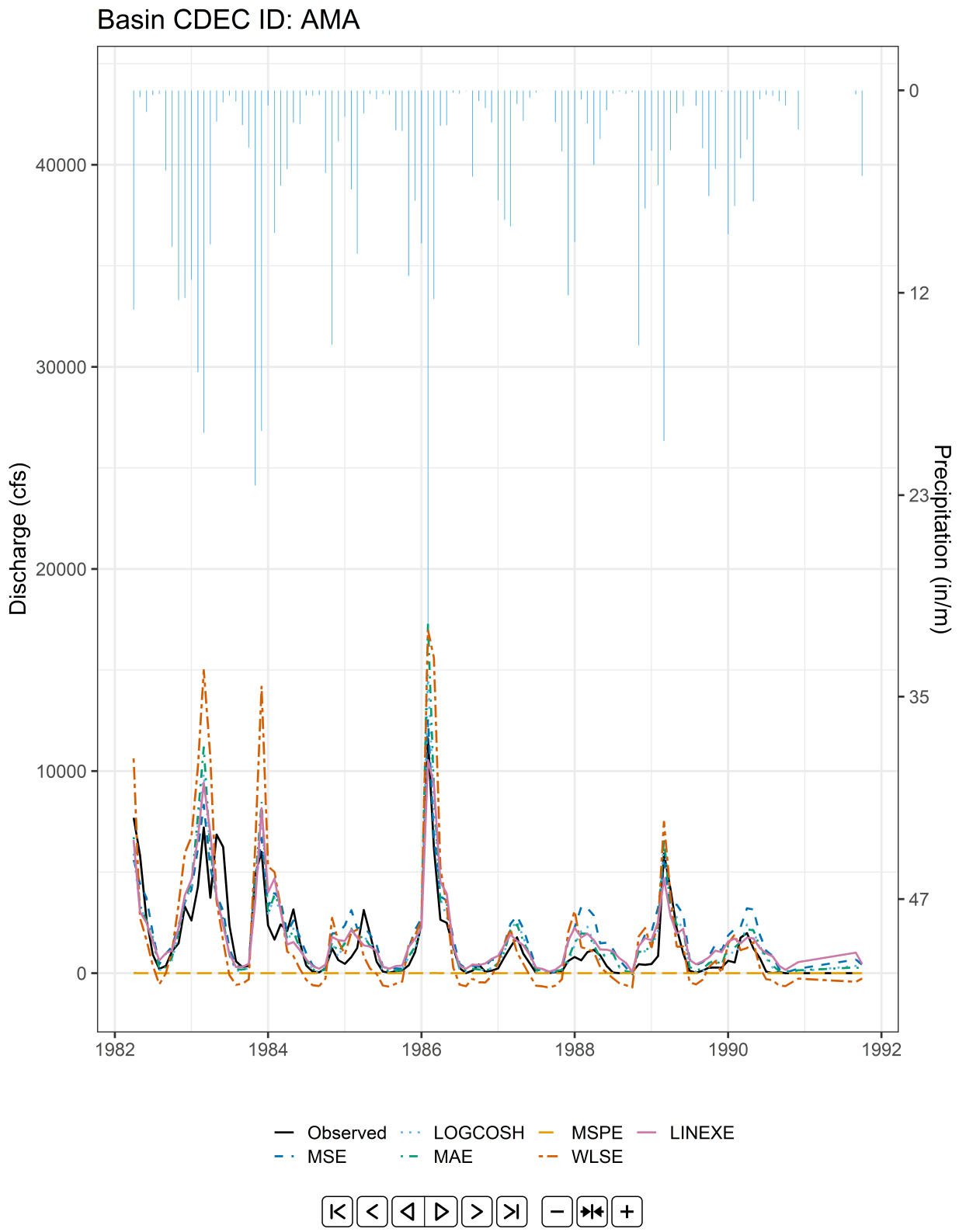


Figure 3.3: Visual fit. Generally, the asymmetric loss functions (i.e., LINEXE and WLSE) try to fit the peaks more so than the other loss functions.

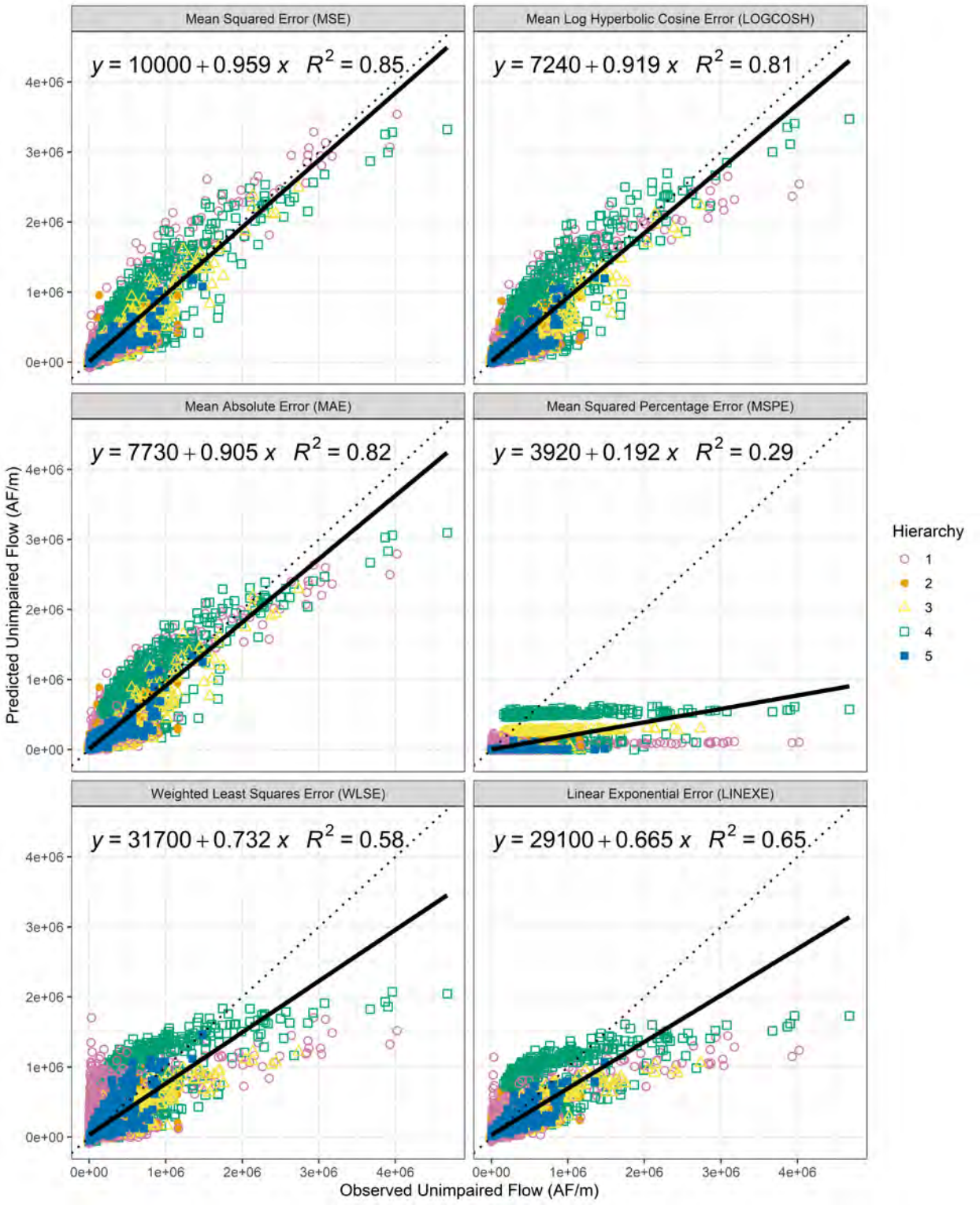


Figure 3.4: Predicted vs. observed plot for different loss functions.There is very little difference observed between the MSE, LOGCOSH and MAE methods. The MSPE greatly underpredicts observations. The WLSE and LINEXE loss functions generally underpredict the largest floods but overpredict lower flood values.

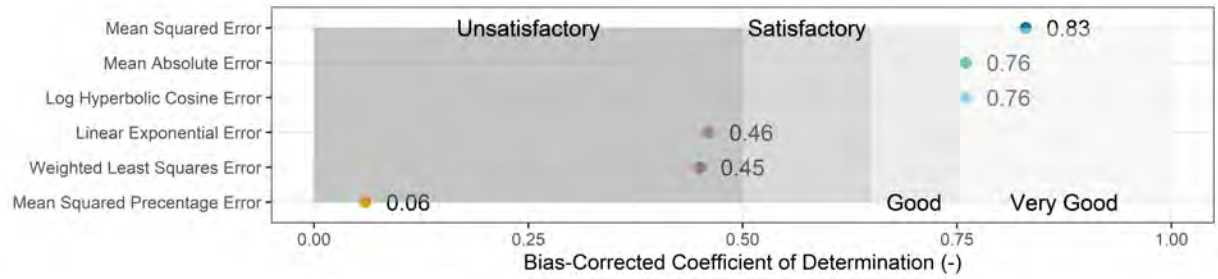


Figure 3.5: bR^2 performance for different loss functions. Since both MSE and bR^2 metrics are calculating similar squared errors, the MSE performs the best in the bR^2. Since the LOGCOSH approximates $x^2/2$ for small x and to $abs(x) - log(2)$ for large x, it performs similarly to the MSE and MAE measures. The two asymmetric losses perform similar to each other, which reassures us that the form of the asymmetric function has less impact on model goodness-of-fit than it being asymmetric. The MSPE performs poorly because of its inherent bias towards lower predictions.

being asymmetric. The MSPE performs poorly because of its inherent bias towards lower predictions.

Figure 3.6 shows the MSPE biased towards smaller predictions as there is a spike in the density (in orange) at the lower values compared to the observations (in black). The asymmetric losses predict larger floods more often (LINEXE > WLSE > MSE), which was their intended use. The MSE density (in dark blue) shows three peaks like the observations, except the floods get more pronounced. This shows the effects of having a squared error loss. The MAE and LOGCOSH perform very similarly predicting more droughts than floods. In contrast, the WLSE predicts more floods than droughts and the LINEXE predicts the most amount of floods. Therefore, MAE and LOGCOSH losses are suitable for models where conservative *drought* management is concerned. The WLSE and LINEXE (with its flexible parameters) are suitable for models where conservative *flood* management is paramount.

Unsurprisingly, model residuals do not have a normal distribution (Figure 3.7). The quantile-quantile plot is created by plotting two sets of quantiles against one another: one a sample (e.g., model residuals) and one the theoretical Normal distribution. Here, the points fall along a line in the middle of the graph, but curve off in the extremities. This behavior implies that model residuals have more extreme values than would be expected if they truly came from a Normal distribution.

In general, these densities show the amount of control the modeler has on the probability distribution of the predicted values when picking a loss function. This flexibility is especially useful in risk-based decision making where the modeling aim is to accurately predict the probability distribution particularly at its tails where high cost consequences may occur. A more direct way of approaching the problem of predicting accurate densities is evaluating the goodness of the density estimate by calculating the generalization log-loss (or log-likelihood out-of-bag) and using conditional density estimation and **probabilistic supervised learning** methods. Models like Mixture Density Networks (MDN) not only predict the expected value of a target, but also the underlying probability distribution (Gressmann, Király, Mateen, & Oberhauser, 2018; Bishop, 1994).

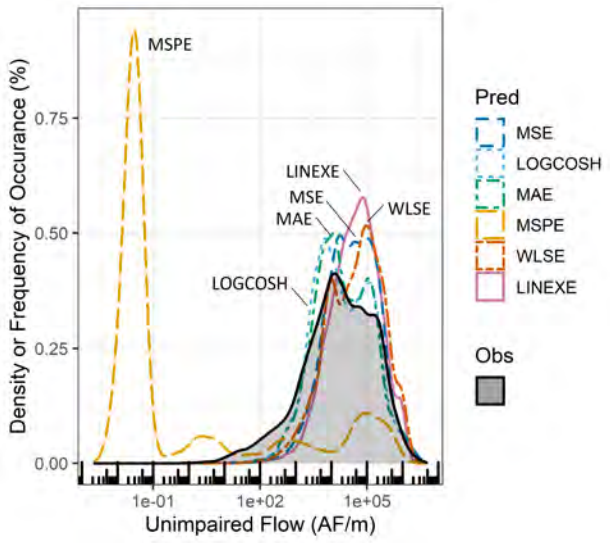


Figure 3.6: Probability densities of predictions and observations. The MSPE is biased towards smaller predictions as there is a spike in the density (in orange) at the lower values compared to the observations (in black). The asymmetric losses predict larger floods more often (LINEXE > WLSE > MSE), which was their intended use. The MSE density (in dark blue) shows three peaks like the observations, except the floods get more pronounced. This shows the effects of having a squared error loss. The MAE and LOGCOSH perform very similarly predicting more droughts than floods. In contrast, the WLSE predicts more floods than droughts and the LINEXE predicts the most amount of floods.

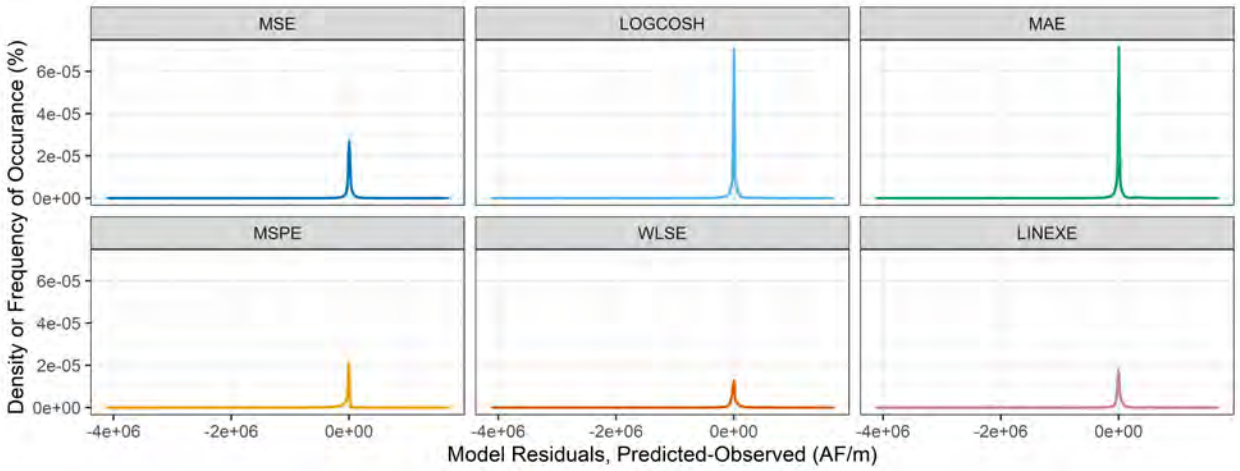
3.3.2 Spatial Distribution of Error

Figure 3.8 shows spatial performance for each loss function. Except for the MSPE which suffers from a major bias, other methods perform similarly. These methods favor the northern basins and watersheds lower in the network. The LINEXE and WLSE perform "worse" in headwater basins than the MSE because the asymmetry is pushing the model to underpredict at low flows. This effect is more pronounced in the WLSE than the LINEXE possibly due to the nuisance parameters. As explained before, the MSE, LOGCOSH and MAE perform similarly.

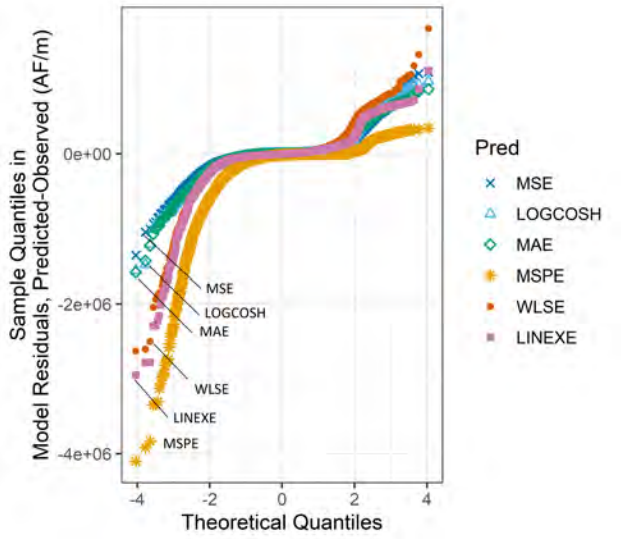
3.4 Conclusion

This chapter followed a risk minimization framework in developing models with different loss functions. We *are putting the horse before the cart*; the loss function is developed before performing the learning, not just as an evaluation step after. In squared error loss functions (i.e., MSE), the peaks or high leverage points get fitted at the expense of the low flows.

The proposed WLSE and LINEXE asymmetric losses are able to force a fit to the tails of the distribution (the peaks and valleys of the hydrograph). These results are shown in the shape of the predicted time series compared to the observations. Asymmetric losses are useful when fitting the peaks and valleys of a hydrograph is important. Their downside is the difficulty in implementing and developing the code and finding appropriate nuisance parameter estimates.



(a) Residual error densities.



(b) Model residuals compared to a Normal distribution, Q-Q plot.

Figure 3.7: Probability densities of model residual error: (a) Model residuals are skewed left in all models, but is less extreme in WLSE and LINEXE; there is more of a tendency to underpredict floods in models, but less so in WLSE and LINEXE. (b) Model residuals do not have a normal distribution. The points fall along a line in the middle of the graph, but curve off in the extremities. This behavior implies that model residuals have more extreme values than would be expected if they truly came from a Normal distribution. WLSE shows the most extreme behavior at higher levels (i.e., overpredicting floods).

As for symmetric functions, the LOGCOSH performs similarly to the MAE and MSE as is shown in the predicted vs. observed graphs and the bR^2 measure. The LOGCOSH and MAE are useful when errors in the estimates for larger values do not need to be penalized more by squaring like in the MSE. The MSPE is biased towards lower predictions and is not suited to problems where the data is skewed positive. However, it can be useful in problems

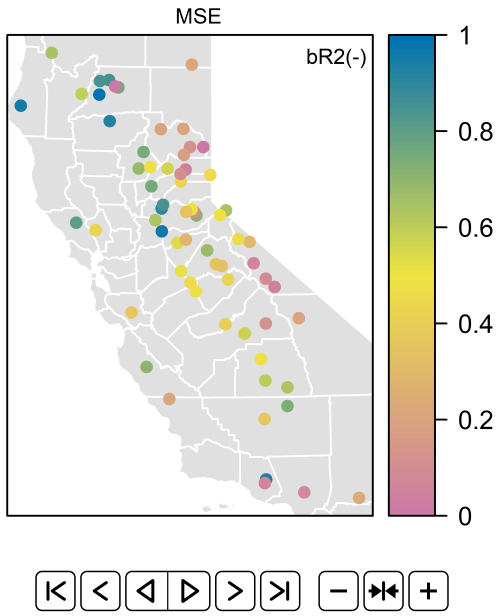


Figure 3.8: Spatial distribution of bR^2 for different loss functions. Except for the MSPE which suffers from a major bias, other methods perform similarly. These methods favor the northern basins and watersheds lower in the network.

where relative errors are more of interest.

In general, the differences show the amount of control the modeler has on the predictions and their probability distribution when picking a loss function. Many reasonable calibration loss functions exist, with somewhat different rationales that affect model estimates differently. The flexibility in picking a loss function is especially useful in risk-based decision making where the modeling aim is to accurately predict the probability distribution particularly at its tails where high cost consequences may occur. Water managers can then use this somewhat unbiased probability distribution in decision analysis. Asymmetric loss functions proved useful in this regard.

Chapter 4

Resample Like Sample: Blocked Resampling for Data Dependent in Time, Space, and Unique Structure

> If two things are similar, the thought of one will tend to trigger the thought of the other.
>
> Aristotle, *"Laws of Association"*, 300 B.C.

Summary

Having chosen and fitted an estimator based on observations, we need to answer further questions about the accuracy of the estimator or the likely quality of inferences. Most resampling techniques used in water resources modeling employ a simple nonparametric random splitting of data into several folds to estimate model error (i.e., k-fold cross-validation). In each iteration, one fold is held out as a test set and others are designated as a training set. While suitable for independent and identically distributed random variables, such random resampling schemes can ignore structures in dependent data, and underestimate model error. The central assumption in resampling is that training and evaluation data are independent. If not, error estimates will be too optimistic, and model selection will favor more complex models (Roberts et al., 2017). Despite this shortcoming, random resampling is widely used in hydroinformatics.

For the PUB problem, where observations are correlated in time, space, or unique structure (e.g., by hydrologic basin networks), more accurate estimates of model error can come from blocked resampling. In blocked methods, correlated observations move in and out of the training set together. Therefore, these methods give us a collection of *approximately* independent and identically distributed random vectors. The difficulty with blocking methods lies in specifying block sizes and structures. Blocking potentially reduces the range of parameters seen by the model or may exclude a particular meaningful combination of predictor variables in the training data set. Too small of a block size and the resampling strategy more closely mimics the randomized method and increases risk of underestimating model error.

Large block sizes force too much model extrapolation and risk overestimating model errors.

This chapter compares the following random and blocked resampling methods. Cross-validation strategies include resubstitution (i.e., training set is test set), random k-fold (or Monte Carlo), leave one group out (LOGO), and leave multiple groups out (LMGO), and bootstrapping strategies include random or independent and identically distributed (IID), blocked by group (BBG), and blocked by multiple groups (BBMG). Of the four model types (i.e., LM, GLM, RF, and NN discussed in Chapter 2), the LM performs the poorest and is also least sensitive to the resampling scheme (bR^2 ranges from 0.23 to 0.26). The RF is most sensitive to the resampling scheme; larger block sizes more accurately captures performance (bR^2 is 0.51 for LMGO, largest block size, and 0.94 for resubstitution, no blocking for the same model). Surprisingly, in the NN, the more *appropriate* the block size the better it performs (bR^2 is 0.75 for randomized 2-fold, which has a large fold size but not blocked systematically, and 0.92 for LOGO, which blocks by basin that is the natural grouping scheme). In the NN, even the LMGO method out-performs the randomized k-fold methods proving that in the NN intelligent blocking (i.e. by basins) proves more useful even though the block size is large (i.e., multiple basins in a block rather than just one).

As expected, when using bootstrapped resampling, models built with the IID method appear to perform better on average and are more reliable (have smaller spread). However, IID resampling underestimates model error and overestimates its reliability, so, true model error is closer to those of the BBG and BBMG methods. Generally, regardless of model type, larger block sizes produce more uncertain model results (i.e., more spread in BBMG and BBG than in IID estimates). These results illustrate the sensitivity of each model's estimated uncertainty to resampling methods, and its importance in designing a resampling strategy. Overall, random resampling is not recommended for studies with correlated data sets.

4.1 Introduction

Having chosen and fitted an estimator $\hat{\theta_n}$ based on observations, X_n, we need to evaluate the accuracy of the estimator $\hat{\theta_n}$ and the quality of inferences made based on $\hat{\theta_n}$ and model parameters. Bootstrap and other resampling methods are general methods for finding estimators of parameters like $\text{MSE}(\hat{\theta_n})$. In predictive modeling, the estimator needs to be accurate at unmeasured locations: either ungauged locations, or at future times where observations do not yet exist. Therefore, the predictive accuracy on the **training set**, the data the model is trained on, is of little consequence. The **test set** error, the error of a set of data not seen by the model, is a better measure of model accuracy.

Test set error can be easily calculated if such a data set exists, or, it can be estimated by holding out a subset of the training data. The holding-out is done by resampling strategies, to create an otherwise non-existent test set. Two popular resampling methods are: cross-validation and bootstrapping. In **cross-validation**, the data set is split into non-overlapping testing and training data sets where each observational unit gets a chance at being in the test set once. In **bootstrapping**, sampling is done with replacement where each observational unit has an equal chance at being selected and being selected more than once. In this case, the probability that the observation $X_0 = x_i$ appears in the training sample is $1-(1-1/n)^n \approx 0.632$ (Efron & Tibshirani, 1997). Therefore, in bootstrapping, approximately $1/3^{\text{rd}}$ of the

data set will end up not being selected and are **out-of-bag**.

Resampling helps avoid problems caused by an unknown population; if the resampling method is chosen appropriately then, the *resample*, together with the sample, is expected to reflect the original relation between the population and the sample (Lahiri, 2013). Appropriate resampling is one where the same dependence structures seen in the sample appear in the resample. The goal is to approximate the data generating mechanism as well as possible or "resample like the sample."

Geographic data often have internal correlation and dependence structures (Legendre, 1993): (1) **temporal autocorrelation**: nature responds to changes gradually. For example, today's precipitation is correlated with yesterday's precipitation; (2) **spatial autocorrelation**: nearby things tend to be more related than those far away. For example, two points close together on a topographic map are more likely to have similar elevations; and (3) **hierarchical structures**: the network of streams flowing into one another (or more formally, the stream order) provides a hierarchical structure. That is, basin topology provides a spatial structure more complicated than mere proximity of river gauges. For example, two points on a river may be close in proximity but depending on which side of the watershed divide they fall on they can be fed by two different basins, in different hierarchies in the network, with different governing hydrologic processes and so, have different measured flows (Figure 4.1).

A dependence structure points to a pseudoreplication problem (Figure 4.2). For an observation, x^d, at a distance, Δd, from another observation, $x^{d+\Delta d}$, where x^d and $x^{d+\Delta d}$ are autocorrelated, the distance Δd can be defined in time, space, or hierarchy. In random resampling, either autocorrelated value is free to lie in the bag of samples given to the model or be left out-of-bag. Therefore, the model can easily predict one, given that the other is likely in the bag. However, in blocking resampling the two observations are connected and will both end up in the bag or out-of-bag. Here, the model is forced to predict a phenomenon from other observations.

Most studies in water resources ignore dependence structures in the data when devising a resampling strategy. When test data are randomly selected from the entire temporal, spatial, and hierarchical domain, training and testing data from nearby locations will be dependent due to autocorrelation. Therefore, if the objective is to project outside the spatial structure of the training data (e.g., to an ungauged basin), error estimates from random resampling, will be overly optimistic. This chapter will compare a model's error estimates given multiple random and blocked resampling strategies.

Moreover, cross-validation has been the traditional method used for the predictive accuracy problem and provides a nearly unbiased estimate of the future error rate. However, the low bias of cross-validation is often paid for by high variability. Efron and Tibshirani (1997) showed that suitably defined bootstrap procedures can substantially reduce the variability of error rate predictions. Bootstrapping and its blocked variants are now the preferred method for resampling in statistics, while cross-validation still remains popular in water resources. This chapter applied both methods to PUB models.

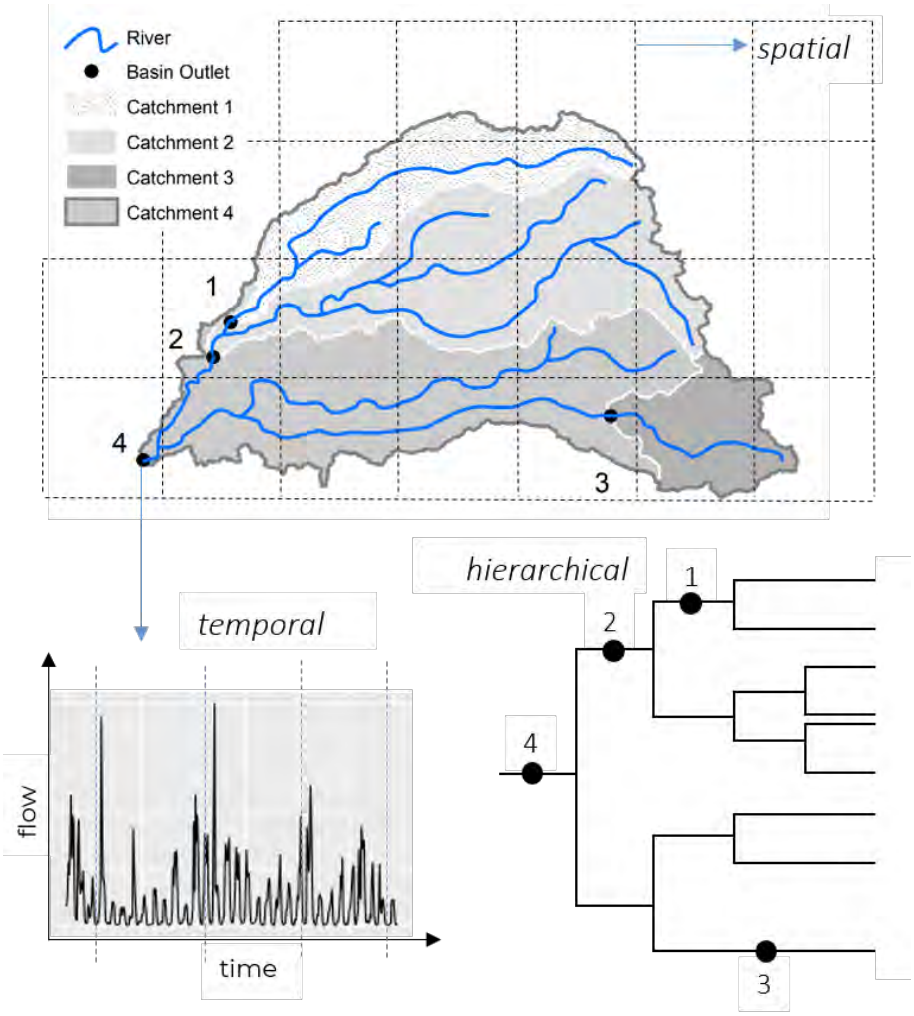


Figure 4.1: Dependence structures in streamflow data are: temporal autocorrelation, spatial autocorrelation, and hierarchical structures. Appropriate resampling is one where the same dependence structures seen in the sample appear in the resample.

4.2 Methods

To find the test set error of the estimator we: (1) simulated n landscapes of the data by resampling the original data set using a chosen resampling method. This separates the data into training and testing sets; (2) for each simulation, fed the training data into the desired machine learning algorithm (i.e., LM, GLM, RF, and NN); and (3) calculated the desired model measure of fit for each simulation (Figures 4.3 and 4.4).

Separating the data into training and testing sets (step 1), can be done by one of the following methods:

Cross-Validation

- Resubstitution: The test set is the training set. Here, the model is evaluated against the same data it has already seen. We expect the model to perform the best here with the distribution of residuals closest to zero.

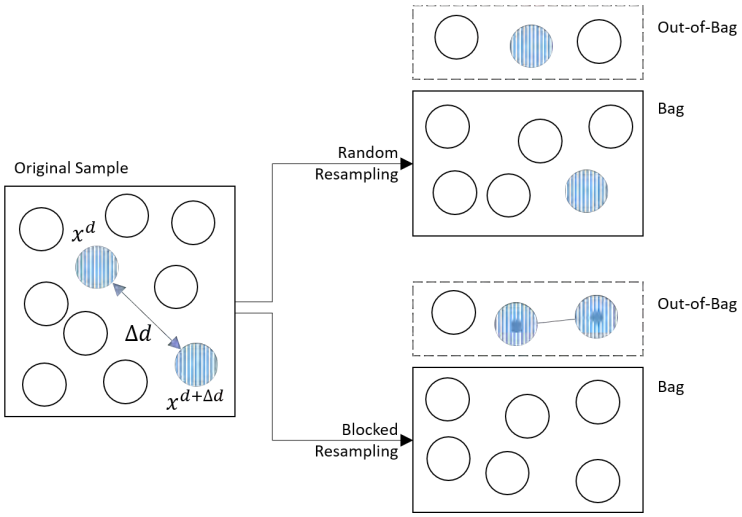


Figure 4.2: Autocorrelation as a pseudoreplication problem. The two striped marbles are autocorrelated. A model that uses random resampling will be able to easily predict one striped marble since it has seen the other. When blocking, the observations move in and out of the bag together.

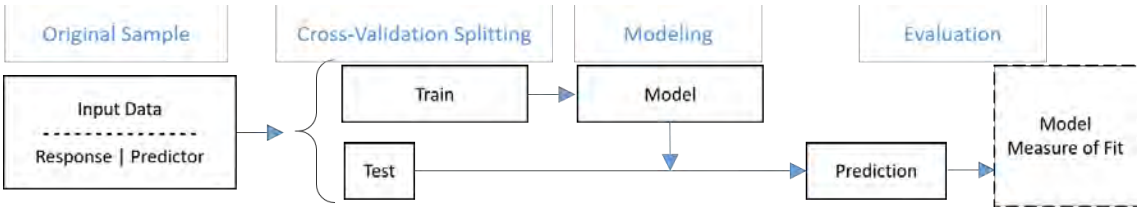
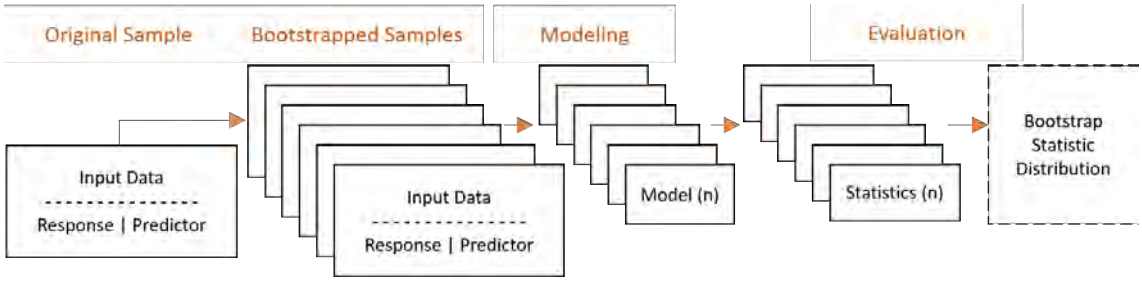


Figure 4.3: Research design for cross-validation resampling to estimate model error.



s

Figure 4.4: Research design for bootstrapped resampling to estimate the distribution of the bootstrap statistic.

- Randomized k-fold or Monte-Carlo:

  2-fold or validation set: The test set is a random 1/2 of the full data set. The training set is the other 1/2. This method is run twice, once with the first half as a test set and next with the second half as the test set.

  5-fold: The data is split into five folds. In each iteration each fold is considered the test set and the other four folds the training set.

  10-fold: The data is split into ten folds. In each iteration each fold is considered

the test set and the other nine folds the training set.

- Leave One Out (LOO): In each iteration, one instance of the data is held out, and the rest of the data set is the training set. This method was too computationally intense to perform on our data set.

- Leave One Group Out (LOGO): In each iteration, one basin's data is held out as a whole and the rest of the basins become the training set. The process is repeated for each basin.

- Leave Multiple Groups Out (LMGO): In each iteration, $1/5^{\text{th}}$ of the basins are held out and the other basins become the training set. The process is repeated for each fold.

- Leave Hierarchies Out (LHO): Blocking is designed across basins that cluster on a river branch. Because of the limited size of the dataset we have not considered this strategy.

Bootstrapping

- Randomized or IID: It is the most popular form of bootstrapping where a new data set is built from randomly resampling the original sample with substitution. The length of the data set is the original length of the data set.

- Blocked By Group (BBG): the data set is blocked by unique basins. The basins are randomly resampled with substitution. Since the basins may have differing record lengths, the length of the data set may not match the original data set. However, the data set will have the same number of basins as in the original data set.

- Blocked By Multiple Groups (BBMG): The data set is blocked by multiple basins. The grouped basins are randomly resampled with substitution. As the group sizes become larger the blocking size becomes larger.

- Blocked By Hierarchy (BBH): Blocking is designed across basins that cluster on a river branch. The grouped basins are randomly resampled with substitution. Because of the limited size of the data set we have not considered this strategy.

When working with gauge records, random resampling can be used to fill in a sparsely incomplete gauge record. Blocked resampling in time is most appropriate for predicting, extrapolating or forecasting flows. In PUB, the most appropriate resampling method is those which block observations in groups since, the goal is to approximate the data generating mechanism as well as possible. In LOGO cross-validation or BBG bootstrapping the grouping structure is simple; we assume that data are correlated within a basin, but independent between basins. The structure of the block bootstrap is easily obtained (where the block just corresponds to the group), and only the groups are resampled, while the observations within the groups are left unchanged (Cameron, Gelbach, & Miller, 2008).

4.3 Results

4.3.1 Model Evaluation

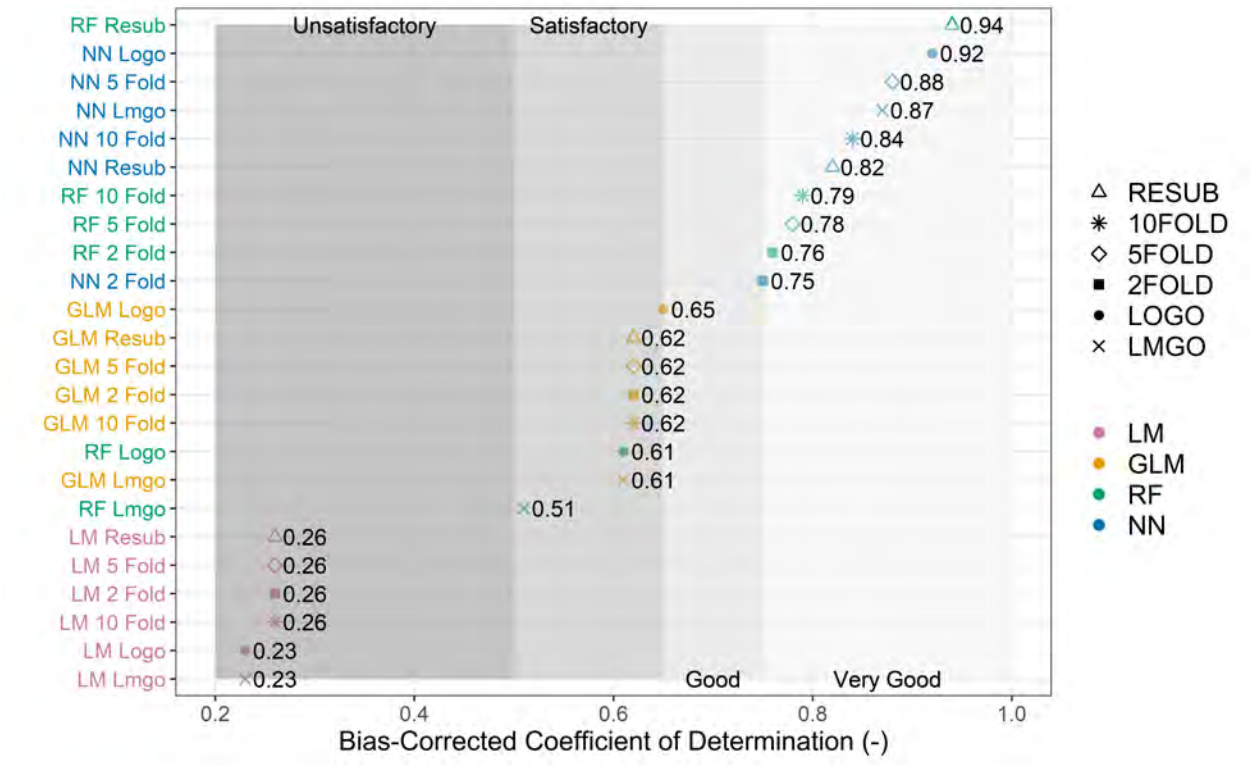
Figure 4.5 shows that generally, in the LM, GLM, and RF, the test set error is lower for smaller block/fold sizes. The LOGO and BBG methods provide the most accurate estimates, since they replicate the natural grouping by basin of the sample in the resample. Therefore, in PUB modeling with an RF, a ten fold cross-validation strategy can underestimate model error (bR^2=0.79 in random ten fold vs. 0.61 in LOGO). The same is true in bootstrapping; IID bootstrapping underestimates model error (bR^2=0.79 vs. 0.50 in BBG). Surprisingly, the NN performs better given a more accurate cross-validation technique (LOGO bR^2=0.92). In bootstrapping, the IID suffers the same fate as in other models and gives artificially low estimates compared to the BBG and BBMG methods. Overall, the performance in each method is clustered together with the NN and RF performing very similarly with the different cross-validations strategies.

Figure 4.6 shows model goodness-of-fit given by cross-validation and bootstrapping. Figure 4.6a shows LM and GLM models are not as sensitive to cross-validation strategies as NN and especially RF models. Figure 4.6b shows bR^2 values obtained where each dot is a simulation, with 100 simulations depicted in each line. The IID method generally underestimates model error in all four model types. In the BBG method, RF shows the biggest spread (standard deviation=0.19, N=100) and NN the lowest spread (standard deviation=0.07, N=100). In general, the bootstrap methods show the spread (or reliability) of an estimate more than cross-validation, which is why bootstrapping methods have become more popular in statistics.

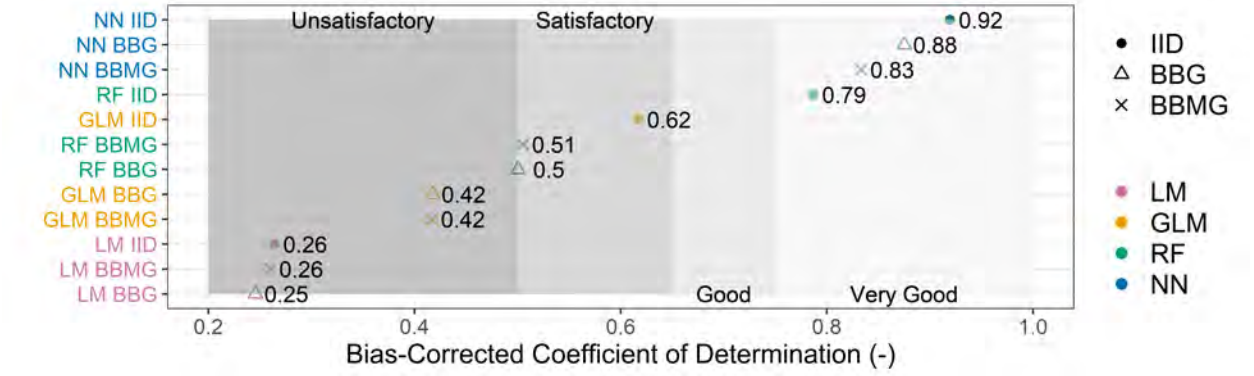
Figure 4.7 shows the observed vs. predicted values for the NN model. It confirms the goodness-of-fit results; in LOGO cross-validation the points more closely fall along the line of best fit, $y = x$, with a slight tendency to underpredict (slope of the fitted line β=0.97). The lowest performance was with the LMGO method proving that grouping basins together can hide meaningful information from the model. In this method, the 67 basins are randomly grouped into 5 folds (block size=13 or 14 basins). Unsurprisingly, the performance degrades compared to the BBG method (block size=1 basin), where less information is held out from the model.

Figure 4.8 shows that the goodness-of-fit obtained by bootstrapping decreases when switching to a blocking method, and the reliability in such measures decreases (standard deviation of bR^2=0.10 in BBMG vs. 0.03 in IID). Increasing block size decreases mean goodness-of-fit slightly (bR^2=0.88 in BBG vs. 0.83 in BBMG) and decreases the reliability of this estimate (standard deviation of bR^2=0.07 in BBG vs. 0.10 in BBMG). This is evident in the increase in spread and slight funnel shape (i.e., heteroscedasticity) of the plots as block sizes increase; withholding more information with bigger block sizes proves especially detrimental for accurately predicting lower flows. The data in the bootstrapping strategies somewhat confusingly have more variability and higher R^2 as compared to the cross-validation strategies. The higher R^2 is due to more data points being closer to the 1:1 line that are being plotted on top of one another.

Figure 4.9 shows the density of predicted vs. observed unimpaired flows for cross-validation resampling. All models tend to predict to the mean due to MSE being the loss function of choice. GLM, RF, and NN more accurately predict the frequency of "floods" (i.e.

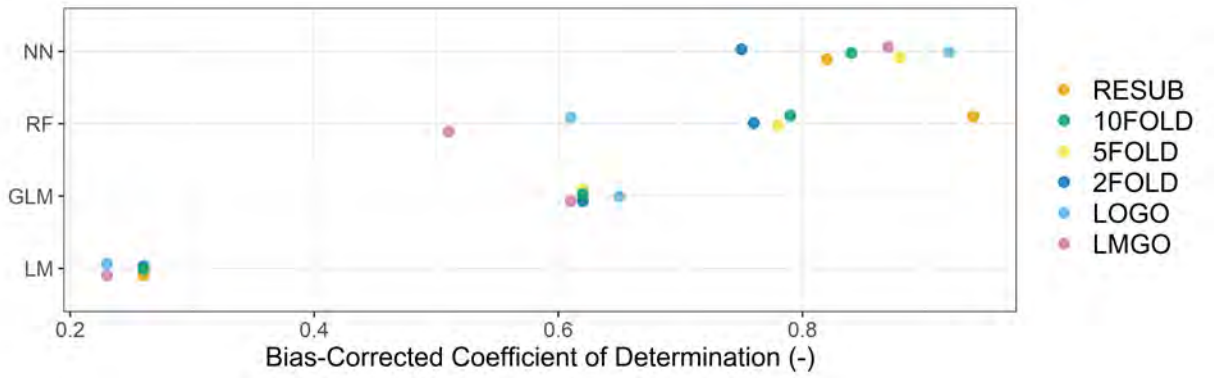


(a) The cross-validation test set error.

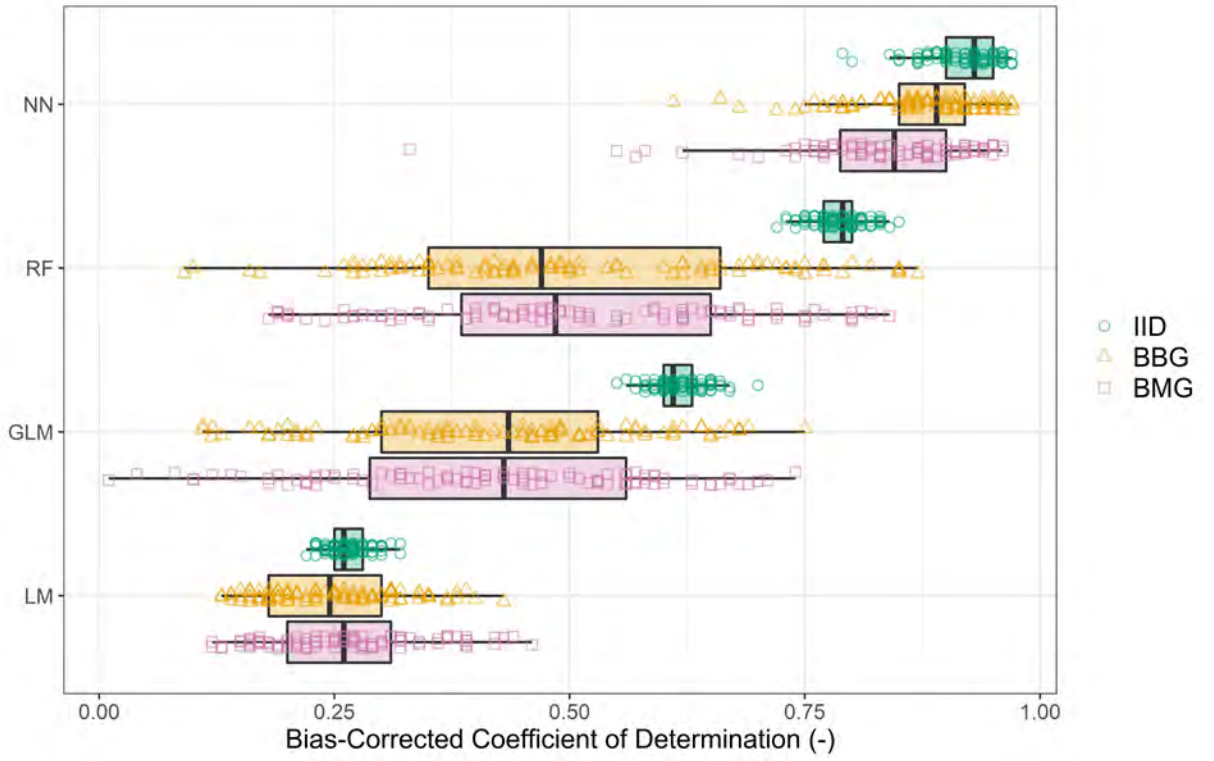


(b) The average bootstrapping test set error.

Figure 4.5: Model goodness-of-fit and average goodness-of-fit given by cross-validation and bootstrapping strategies. Generally, in the LM, GLM, and RF models, the test set error is lower the smaller the block size or fold size. However, the NN prefers the more appropriate LOGO cross-validation strategy.



(a) The cross-validation test set error.



(b) The bootstrapping test set error.

Figure 4.6: Model goodness-of-fit given by cross-validation and bootstrapping (100 simulations per model per bootstrapping strategy). (a) LM and GLM models are not as sensitive to cross-validation strategies as NN and especially RF models. (b) The spread of the bR^2 values obtained proving the value of having repeated experiments when estimating a model measure of fit.

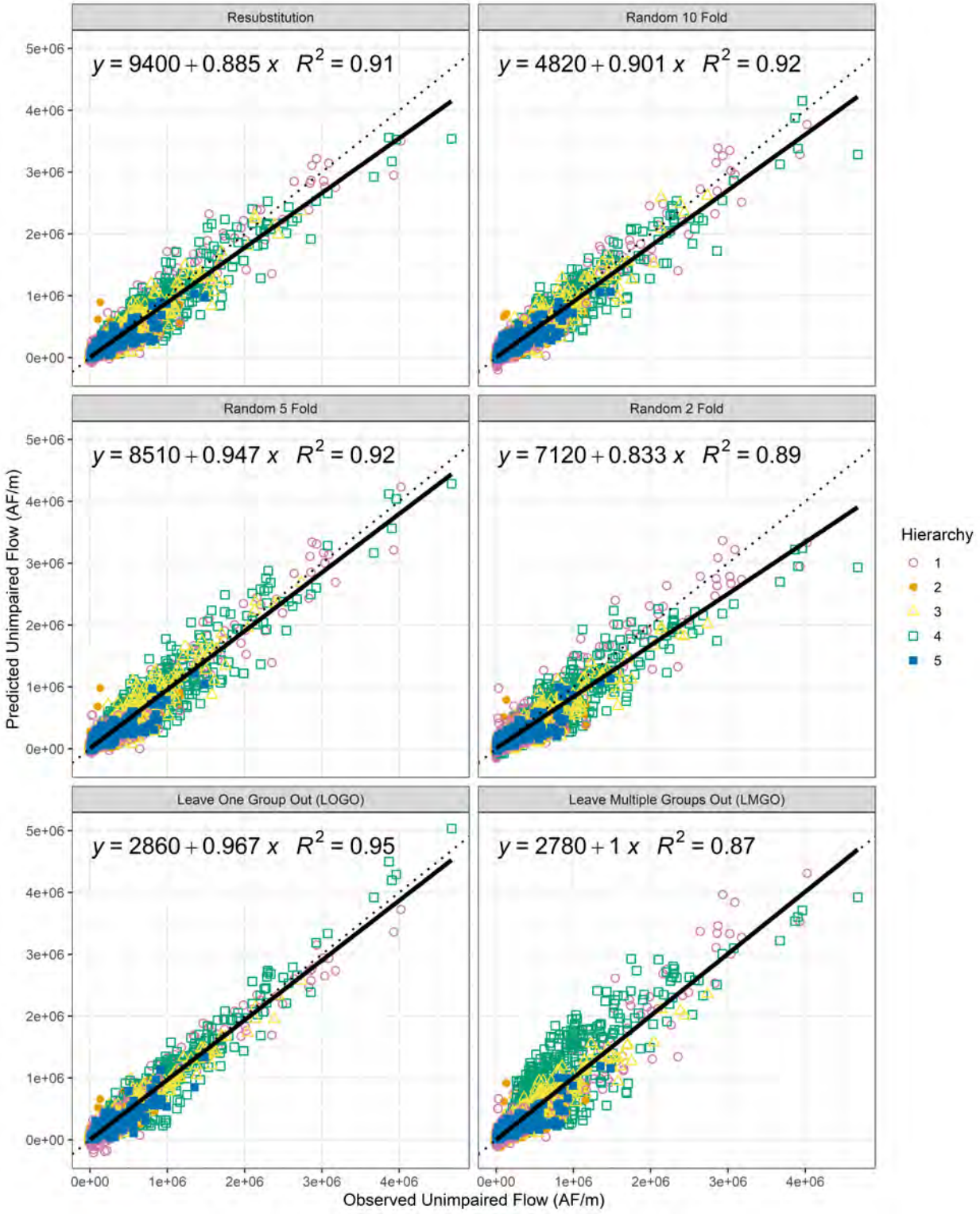


Figure 4.7: NN observed vs. predicted for different cross-validation strategies. NN's performance is not as sensitive to the cross-validation strategy and prefers the more appropriate LOGO method.

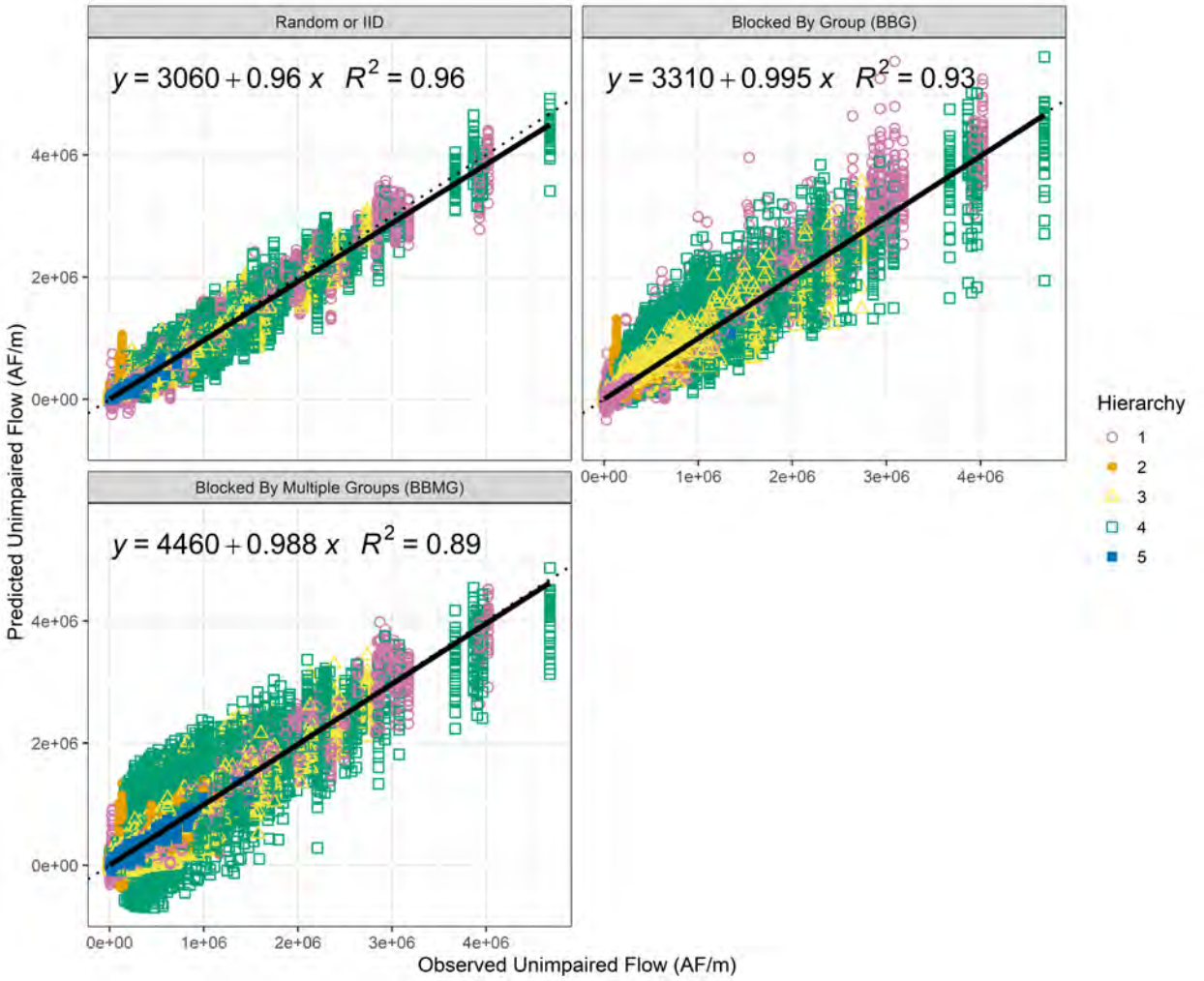


Figure 4.8: NN observed vs. predicted for different bootstrapping strategies (No. of simulations=100). As block sizes increase, the spread increases especially for low flows; withholding more information with bigger block sizes proves especially detrimental for accurately predicting lower flows.

the right side tail). All models, except for the RESUB, 10 FOLD and 5 FOLD in the RF, fail to predict the frequency of "droughts" (i.e. the left side tail). As block sizes increase in the RF, the harder it becomes for the RF to predict droughts, much like other models. LM shows very little sensitivity to the cross-validation resampling method. GLM has a slight increase in low flow densities due to the Tweedie distribution (a semi-continuous function) defining variance in the y values. Also, blocking strategies (i.e., LOGO and LMGO) have a slight improvement in predicting floods. In the RF, blocking strategies (i.e., LOGO and LMGO) more closely follow each other than the non-blocked methods, with the RESUB method being closest to observed values. The LOGO and LMGO methods predict more floods compared to other non-blocked methods. In the NN, all methods except for the random two fold method follow each other, indicating that the 1/2 fold size, significantly affects the model. The fact that large folds or blocking itself produces higher flood densities may be because in a model built with less data, the bigger flood values have more leverage in producing a flood sensitive model.

Figure 4.10 shows the density of predicted vs. observed unimpaired flows for bootstrapped resampling. The observed value densities also are depicted and slightly differ from one another because in bootstrapping the data set gets resampled. The densities here, much like with the cross-validation methods, show a regression to the mean due to MSE being the loss function of choice. In the LM, bootstrapping strategies are virtually indistinguishable. In the other models, the blocking strategies (i.e., BBG and BBMG) follow each other more closely than the IID. Just as in cross-validation, blocking produces higher flood densities.

4.3.2 Spatial Distribution of Error

Figures 4.11 and 4.12 show goodness-of-fit results spatially. In all models there is a ridge of basins (lower Sierra Nevada basins) that generally have better fit. These basins have larger flows and the models trying to predict these values are more accurate at the expense of poorly predicting lower flows seen predominantly in headwater basins. In all model types, generally, performance declines as fold/block sizes increase.

4.4 Conclusion

This chapter presented various blocking resampling techniques where the observations in a block are bonded together. The idea behind blocked resampling is simple: *birds of a feather flock together*, or more accurately birds of a feather *should* flock together (Figure 4.13). That is, if two observations are autocorrelated they should be both included in the bag, or training set, or both be out-of-bag, or in the test set.

These blocking methods show how much random resampling underestimates model error. Models evaluated with random methods have artificially low errors due to pseudoreplication from autocorrelation. This is not to say that, in hydrology, random resampling is never useful; a random test-train split is most appropriate for predicting flow for a sparsely incomplete gauge record. Blocked resampling in time is most appropriate for predicting or extrapolating streamflow in time for that location. One should not expect to use these resampling strategies and get the same predictive accuracy in a purely ungauged basin problem, where blocks are supposed to be designed across geographic space (or more accurately hierarchical structure). Results show that generally model performance estimates decline as block sizes increase.

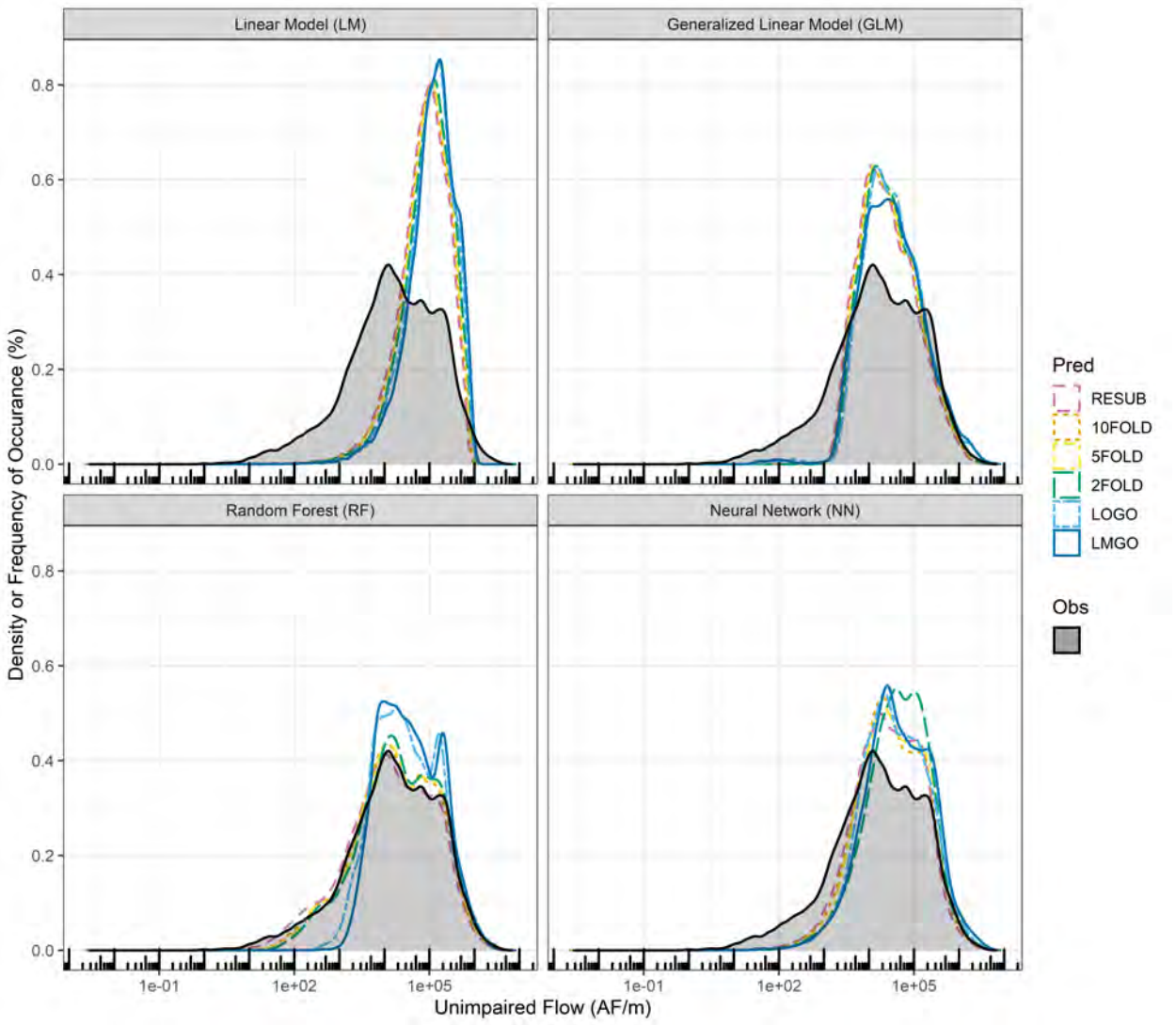


Figure 4.9: Cross-validation prediction density on a log transformed x axis. All models tend to predict to the mean due to MSE being the loss function of choice. GLM, RF, and NN more accurately predict the frequency of "floods" (i.e. the right side tail). LM shows very little sensitivity to the cross-validation resampling method. GLM has a slight increase in low flow densities due to the Tweedie distribution (a semi-continuous function) defining variance in the y values. Also, blocking strategies (i.e., LOGO and LMGO) have a slight improvement in predicting floods. In the RF, blocking strategies (i.e., LOGO and LMGO) more closely follow each other than the non-blocked methods, and these methods (i.e., RESUB and K-FOLD) are closest to observed value densities. The LOGO and LMGO methods predict more floods compared to other non-blocked methods.

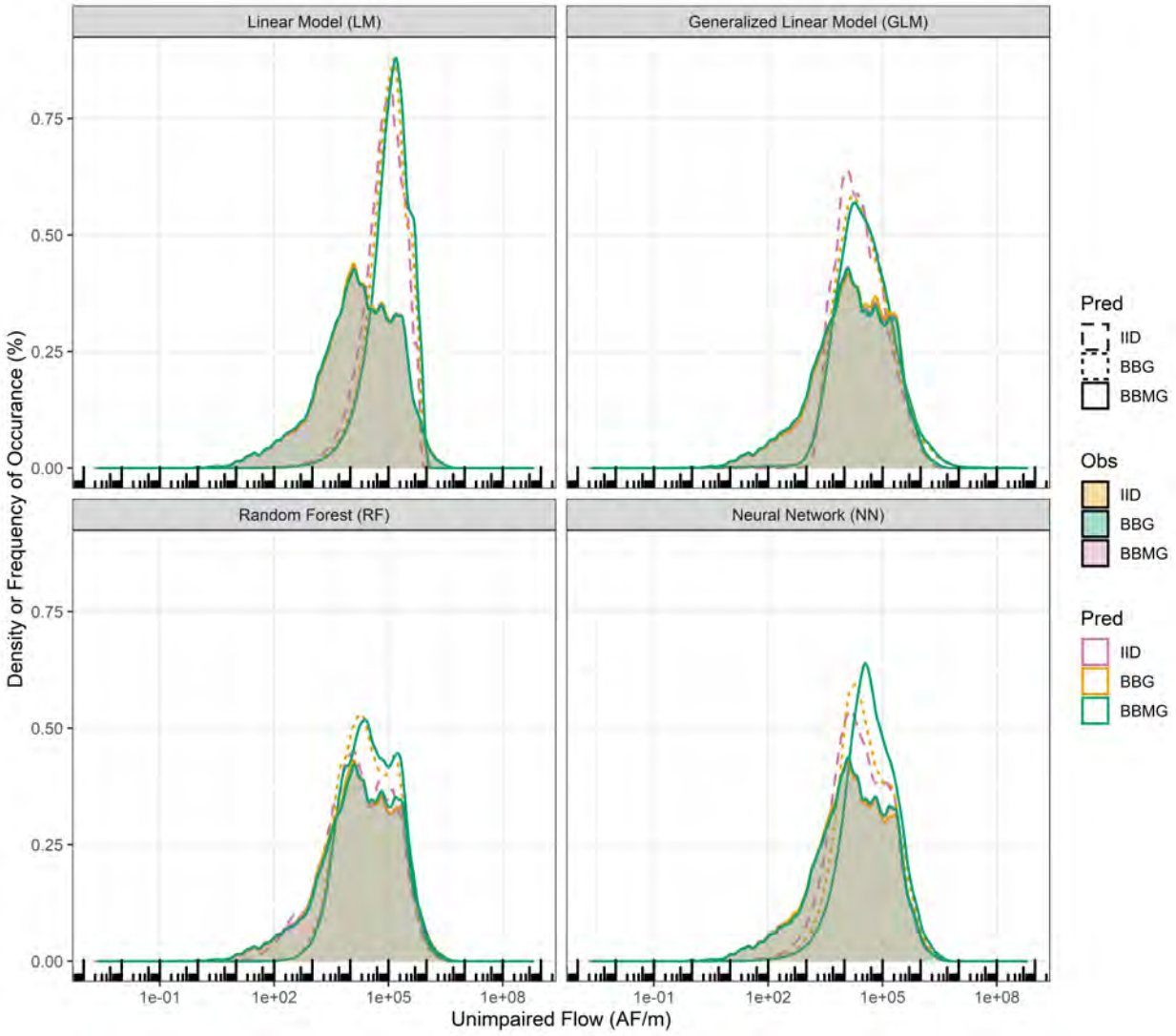
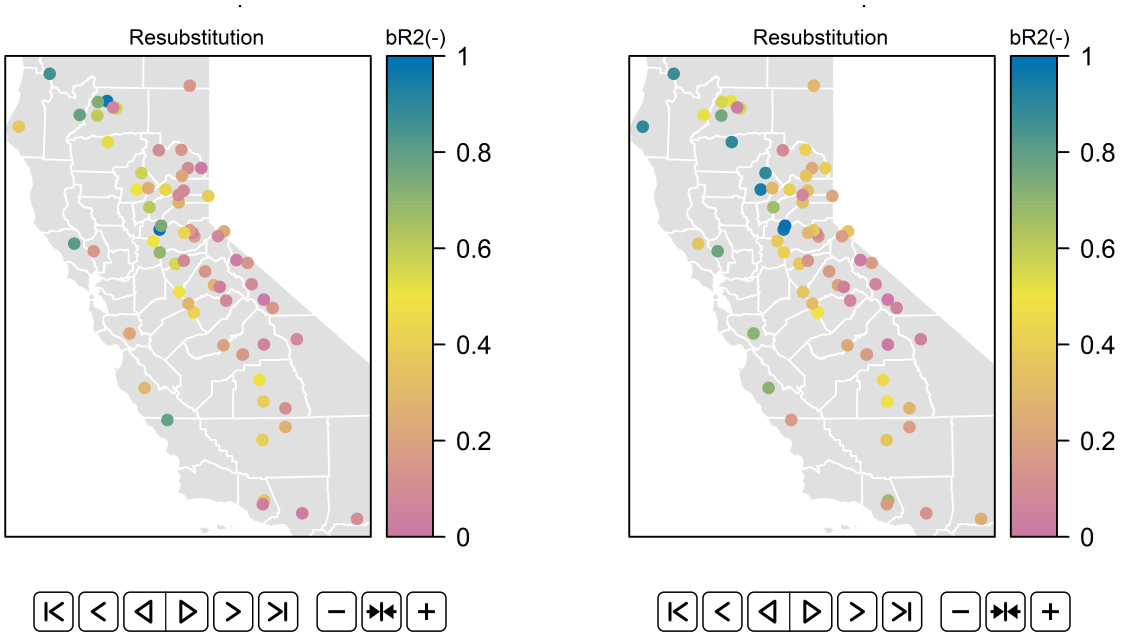
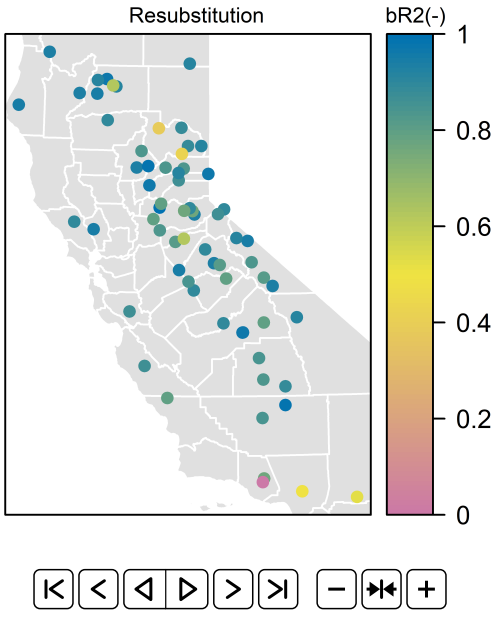


Figure 4.10: Bootstrapping prediction density on a log transformed x axis. In the LM, bootstrapping strategies are virtually indistinguishable. In the other models, the blocking strategies (i.e., BBG and BBMG) follow each other more closely than the IID, a non-blocking resampling method. Just as in cross-validation, blocking produces higher flood densities. In the RF, the IID method is closest to the observed value densities.

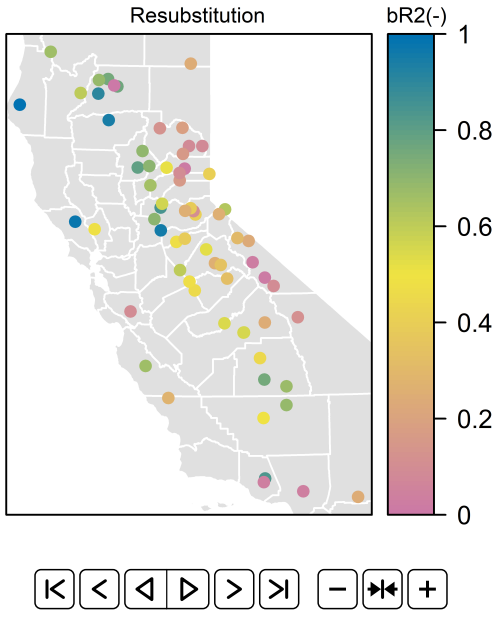


(a) LM

(b) GLM

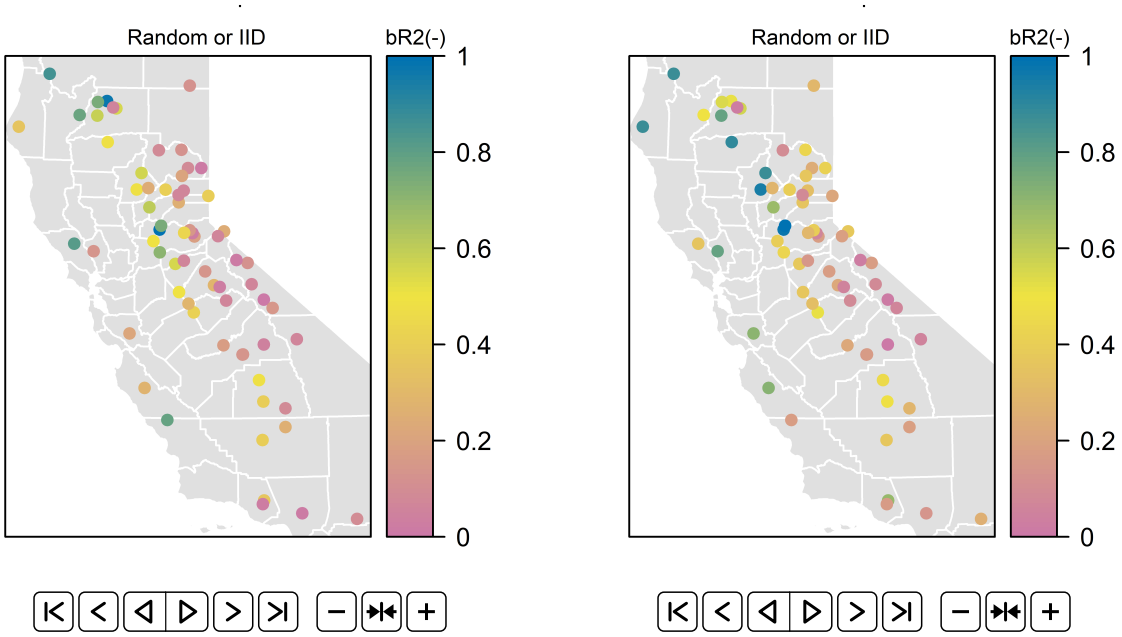


(c) RF



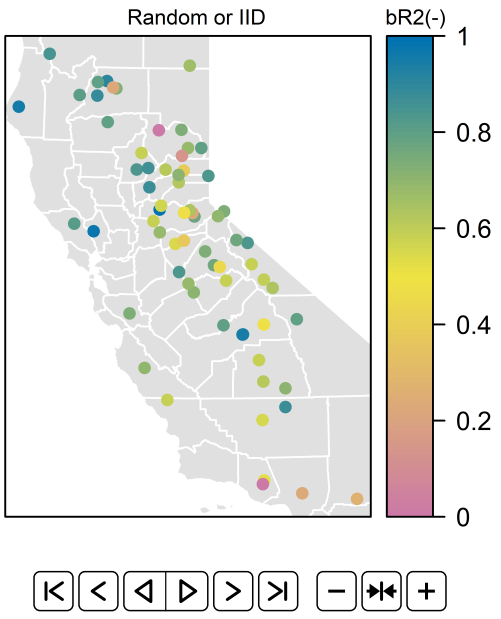
(d) NN

Figure 4.11: The spatial distribution of bR^2 performance for cross-validation strategies. In all model types, generally, performance declines as fold sizes increase.

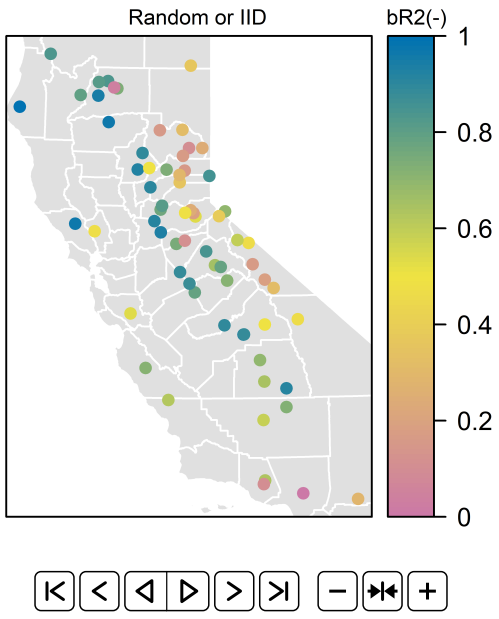


(a) LM

(b) GLM



(c) RF



(d) NN

Figure 4.12: The spatial distribution of bR^2 performance for bootstrapping strategies. In all model types, generally, performance declines as block sizes increase.

Figure 4.13: The idea behind blocked resampling. Birds of a feather *should* flock together.

Some modeling methods are more sensitive to the resampling scheme than others. The LM performs poorly and is least sensitive. The RF is the most sensitive quite possibly because it uses bootstrapping to construct a tree. The NN, performs well with all resampling strategies and surprisingly, performs better given the LOGO cross-validation strategy. The bootstrapping results also show how much variance there could be around a particular error estimate and the importance of the field moving away from cross-validation that gives one estimate of the error to bootstrapping that can give an estimate of the reliability of the error as well. Larger fold sizes and blocked strategies seem to favor predicting more floods, which could be because fewer data available to the model gives more leverage to observations with a higher value (especially when using the MSE loss function).

Chapter 5

Climate Change and Future Unimpaired Flow

> For all of its uncertainty we cannot flee the future.
>
> Barbara Jordan, *"Democratic National Convention Keynote Address"*, 1976

Summary

Future water resources conditions are usually estimated using projected climate variables (i.e., precipitation and temperature) from global climate models (GCMs) and a hydrologic model that routes precipitation to runoff. Pierce et al. (2018) identify a subset of four of the 32 GCMs that have done particularly well reproducing California's historical climate and are recommended when using more GCM data is prohibitive. This chapter uses these models each with two Representative Concentration Pathways (4.5 and 8.5) to study future hydrology. Pierce et al. (2018) bias-corrected and downscaled the GCM data using the Localized Constructed Analogues (LOCA) statistical method and the Variable Infiltration Capacity (VIC) model. The downscaled runoff rasters were then aggregated to the CDEC basin boundaries as a simple routing technique for this dissertation. Each downscaled climate change model data is put through the NN model built on past hydrology and the model predictions are compared to the runoff projections from the VIC+LOCA+Aggregation model.

There is fairly good agreement in the statistical (NN) model's unimpaired flow predictions and the mechanistic and statistical (VIC+LOCA+Aggregation) model's runoff (R^2= [0.65-0.72]). However, the NN model predicts more low flows than the VIC+LOCA+Aggregation models; when we compare more smoothed data (with a moving average window), we can see a bias emerge (e.g., β_1= 0.95 to 1.84 for CanESM2 RCP 4.5). This can also be seen in the time series comparisons; with a larger moving average window, the NN model's predictions are consistently higher than the runoff projections. The climate changed experiment is much like the problem of ungauged basins where the "true" test set is one which has no observations. However, we can argue that the runoff projections from the VIC+LOCA+Aggregation models are slightly more reliable since the processes of finding the amount of recharge and runoff for each pixel ($\sim$ 100 km) is grounded in hydrology (VIC model).

5.1 Introduction

Future water resources conditions are usually estimated using projected climate variables (i.e., precipitation and temperature) from global climate models (GCMs) and a hydrologic model that routes precipitation to runoff. A more robust climate assessment relies on multiple scenarios of future climate from current GCMs available. Two common **Representative Concentration Pathways** (RCPs) provide information on possible scenarios or development trajectories for the main forcing agents of climate change. RCPs encapsulate particular sets containing emission of aerosols, concentration of greenhouse gasses, and land-use trajectories. For example, RCP 4.5 is a "medium" stabilization emissions scenario that models a future where societies attempt to reduce greenhouse gas emissions, while RCP 8.5 is a very high baseline or "business-as-usual" emission scenario (Van Vuuren et al., 2011).

The precipitation and temperature data used here originally came from the Climate Model Intercomparison Project version 5 (CMIP5; Taylor, Stouffer, & Meehl, 2012), which includes 32 coarse-resolution ($\sim$ 100 km) GCMs. Runoff rasters were produced with the Variable Infiltration Capacity (VIC) model. The temperature, precipitation, and runoff were bias corrected and downscaled using the Localized Constructed Analogues (LOCA) statistical method to better capture key features in California's climate (Pierce et al., 2018). These data are available at www.caladapt.org and can be downloaded with a simple `Rcurl` script and the CalAdapt API.

Differences in climate change projections from different climate models arises from: (1) model uncertainty: differing representations of various processes, (2) scenario uncertainty: unknown rates and changes in climate forcings (e.g., rate and concentration of CO_2 and other greenhouse gases), and (3) climate uncertainty: unknown internal variability of the climate such as El Niño (Hawkins & Sutton, 2011). Pierce et al. (2018) identifies a subset of four, out of 32, GCMs that have done particularly well reproducing California's historical climate and are recommended when using more GCM data is prohibitive. In this chapter, the following models, each with two RCPs (4.5 and 8.5), are used to explore future hydrology:

- CanESM2 (CCCma, BC, Canada): an "average" model in terms of changes in precipitation and temperature.
- CNRMCM5 (CNRM and CERFACS, Toulouse, France): a "cool/wet" model.
- HadGEM2ES (Met Office Hadley Centre, UK): a "warm/dry" model.
- MIROC5 (JAMSTEC, AORI, and NIES, Japan): a model most unlike the other three.

5.2 Climate Change Data

Figures 5.1, 5.2, and 5.3 compare the **relative percent difference** (RPD) between historical and future projections in precipitation, temperature, and runoff in the different climate+VIC models. To calculate RPD, first, the annual values are averaged across time in both the projected data (2015-2099) and the observed data (1914-2014). Then, the RPD is calculated using Equation 5.1. Its value always lies between -200% and 200%. It is positive when the mean projected data exceeds the mean observed and negative when the

Figure 5.1: Relative percent difference in precipitation for each climate model. MIROC5 RCP 8.5 and CanESM2 RCP 4.5 show the most amount of dryness across California. In other models, southern California becomes wetter (e.g, CNRMCM5 RCP 4.5 and HadGEM2ES 4.5). This projected wetness can be viewed as a southward shift in the cooler/wetter climates of today. In all models, except for CanESM2, the RCP 8.5 scenario is dryer than its counterpart RCP 4.5.

mean observed exceeds the mean projected. A regular percent difference was not calculated because dividing by zeros in the mean observed values creates large relative errors.

$$RPD(x,y) = 2\frac{x-y}{|x|+|y|} * 100\% \qquad \text{RPD} \in [\text{-}200\% \text{ , } 200\%] \tag{5.1}$$

Figure 5.4 compares the mean RPD between historical and future precipitation, temperature, and runoff for different GCMs. These values were calculated by: taking a 30 year subset of the annual values for the projected data (2070-2099) and the observed (1976-2005); finding the RDP for each year (with 1-to-1 matching: 2070 with 1976, 2071 with 1977, etc.); cropping these rasters to the California boundary; averaging the RPDs for each pixel across time; and averaging across space to arrive at one mean RDP for each GCM and RCP combination. Figure 5.4(a) shows that models with RCP 8.5, on average, project hotter climates. The CanESM models and the CNRMCM5 RCP 8.5, on average, project wetter climates. Using these models can give us a wide range of possible futures (dry/wet, and warm/less warm) for California. Figure 5.4(b) shows that runoff in all models generally increases linearly with precipitation and at a higher rate, confirming the positive relationship used when constructing runoff.

Figure 5.5 shows the mean annual precipitation, temperature, and runoff over time. These values were calculated by averaging the parameter value across California and Nevada for each year. With uncertainty in models, scenarios, and climate there are many varied projection paths these variables can take. MIROC 5 RCP 4.5 and CNRMCM5 RCP 4.5

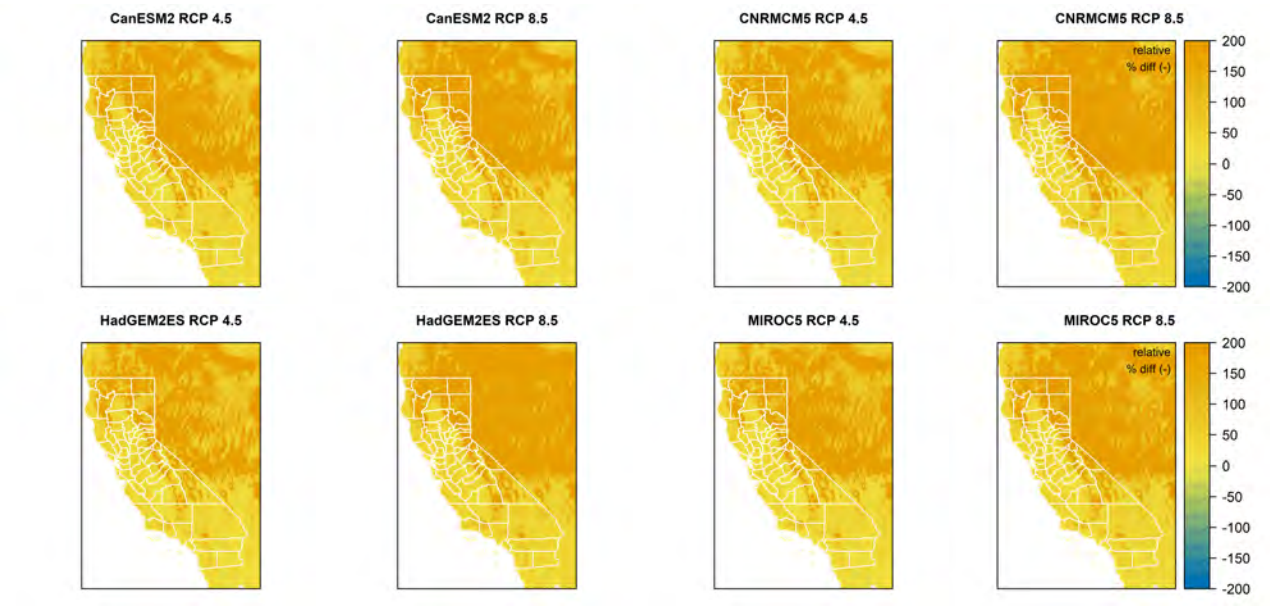


Figure 5.2: Relative percent difference between historical and future air temperature for each climate model. In all models, California becomes hotter in all areas of the state; the relative percent difference is never negative. The Sierra-Nevada range and the northern part of California are projected to experience the brunt of the warming and the central valley and southern California are projected warm to a lesser extent. In all models, the RCP 8.5 scenario is slightly hotter than its counterpart RCP 4.5. Overall, there is more agreement between the models on air temperature changes than there is with precipitation.

project a wetter climate (confirming mapped values in Figure 5.3). Annual temperature values slightly increase over time in all models with RCP 4.5 values generally lower than RCP 8.5. Runoff shows more of a change compared to precipitation, with more frequent peaks in the projections compared to the observed.

To smooth out Figure 5.5, Figures 5.6 and 5.7 show the rolling 10 year mean and standard deviations in annual precipitation, temperature, and runoff. These values were calculated by: first, taking a 10 year rolling window starting from 2015 and looking forward in time; finding the average or the standard deviation of the annual parameter values for each raster pixel in time with the last 10 years (2090-2099) discarded from the analysis; finding the spatial mean of the 10 year rolling mean or standard deviation for each parameter. Figure 5.6 shows mean precipitation increasing in CNRMCM5 RCP 8.5. Mean temperature increases in most models and is more pronounced in RCP 8.5. Mean runoff follows mean precipitation trends. Figure 5.7 shows standard deviations in precipitation increasing in CNRMCM5 RCP 8.5 and HadGEM2ES RCP 8.5. Standard deviation in temperature increase in MIROC5 RCP 4.5 and CNRMCM5 RCP 4.5. Standard deviations in runoff follows the trends seen in precipitation. In all but CNRMCM5 RCP 8.5 (cool/wet model), the differences in precipitation and runoff across models are evident from the beginning of the time period and the mean and standard deviations remain stationary.

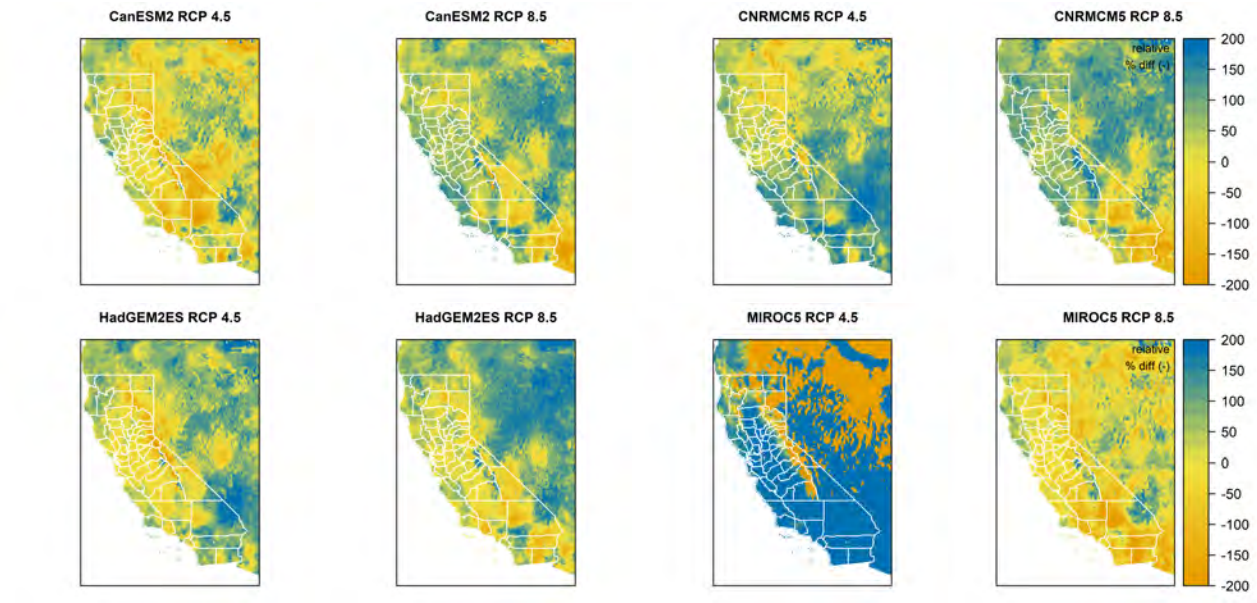


Figure 5.3: Relative percent difference between historical and future runoff for each climate+VIC model. Much like the changes in precipitation, southern California becomes wetter in CNRMCM5 RCP 4.5 and HadGEM2ES 4.5, and in all models, except for CanESM2, the RCP 8.5 scenario is dryer than its counterpart RCP 4.5. MIROC5 RCP 4.5 is most unlike the other models in that it projects a wetter environment for the majority of California (i.e., the central valley and southern California). However, its RCP 8.5 projects a much drier state.

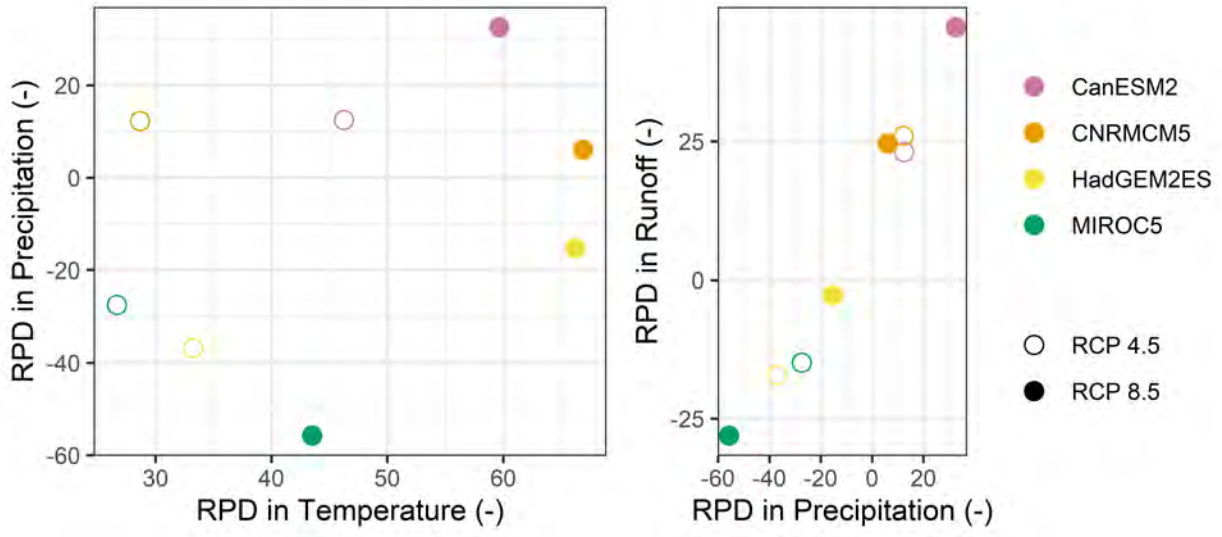


Figure 5.4: Relative percent difference between historical and future precipitation, temperature, and runoff for different GCMs. (a) Models with RCP 8.5, on average, project hotter climates. The CanESM models and the CNRMCM5 RCP 8.5, on average, project wetter climates. (b) Runoff in all models generally increases linearly with precipitation and at a higher rate.

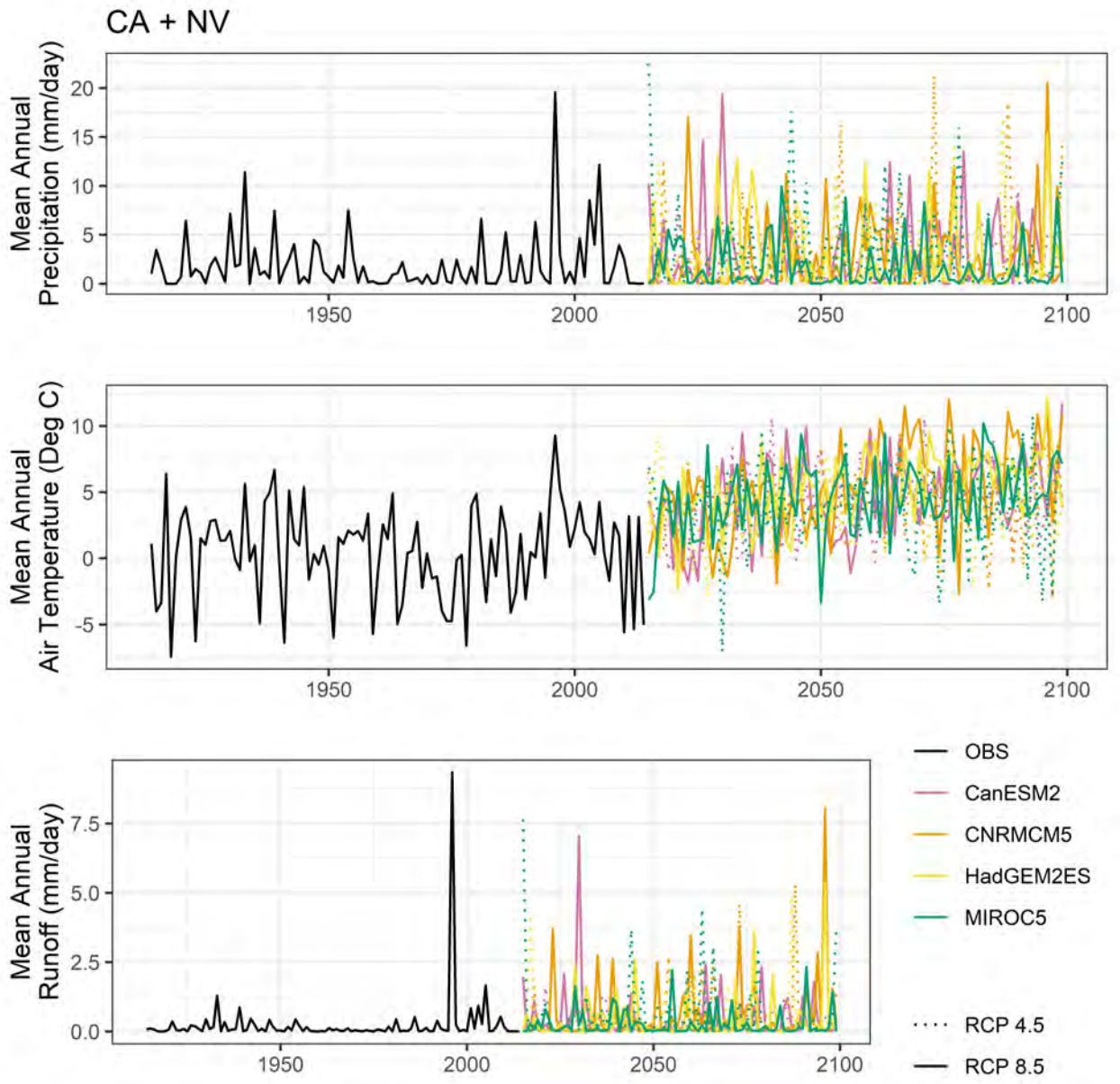


Figure 5.5: Time series of projections in precipitation, temperature, and runoff. MIROC 5 RCP 4.5 and CNRMCM5 RCP 4.5 project a wetter climate. Annual temperature values slightly increase over time in all models with RCP 4.5 values generally lower than RCP 8.5. Runoff seems to show more of a change compared to precipitation, with more frequent higher values in the projections compared to the observed.

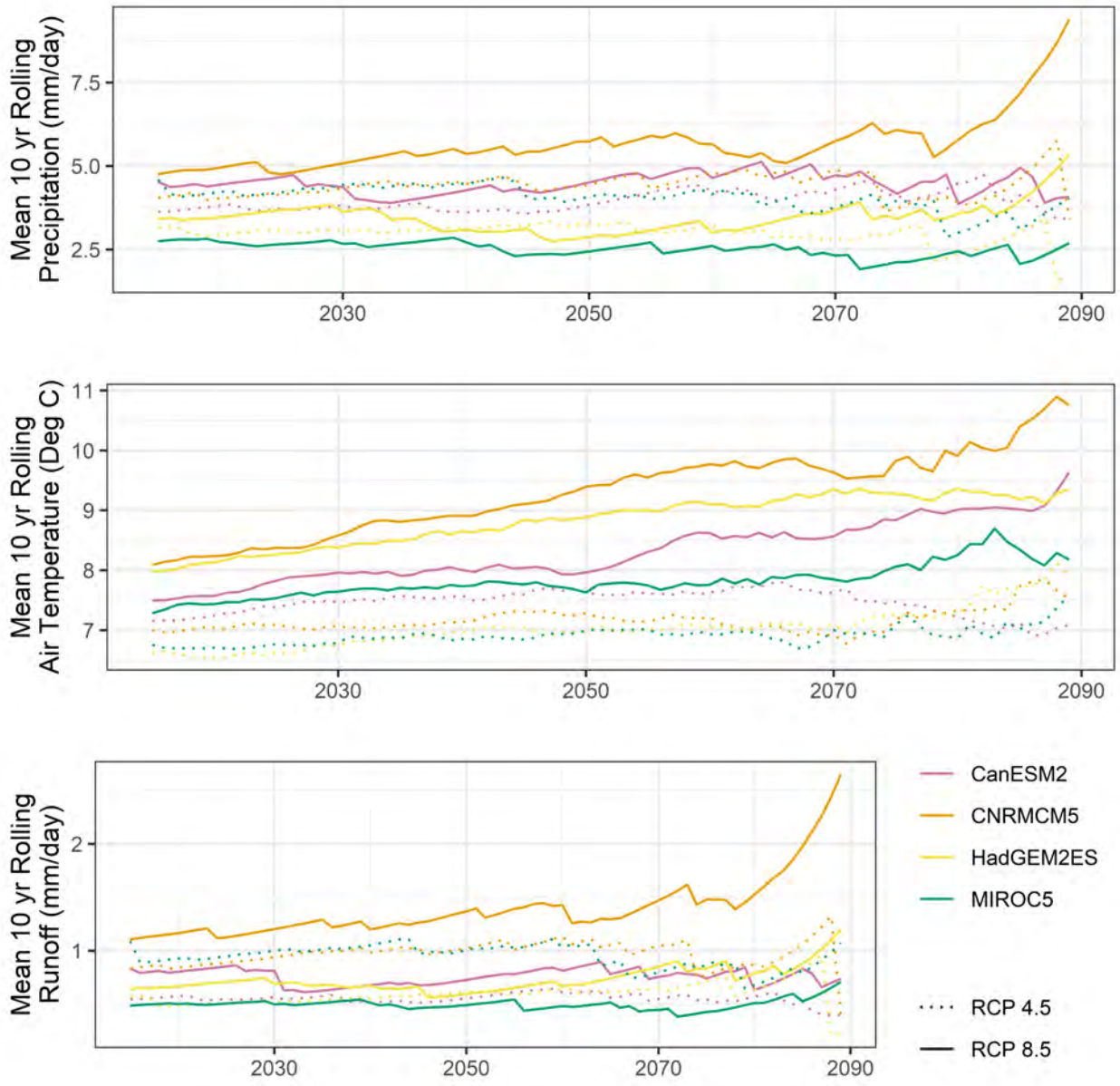


Figure 5.6: Time series of 10 year rolling mean in precipitation, temperature, and runoff. Most differences in precipitation between models is in the mean precipitation rather than in trends, except in CNRMCM5 RCP8.5 (cool/wet model). Mean temperature increases in most models and is more pronounced in RCP 8.5. Mean runoff follows the trends seen in mean precipitation.

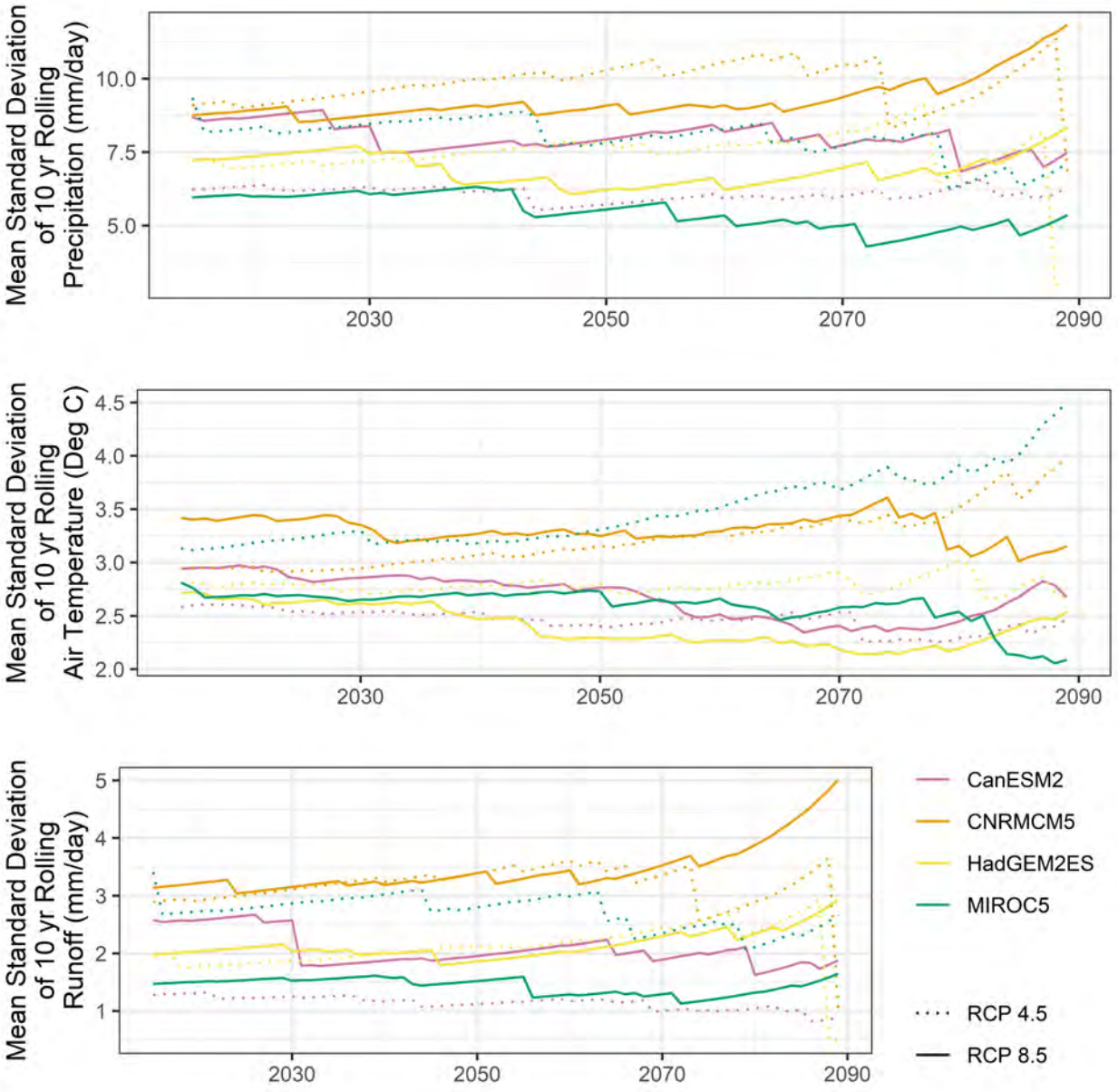


Figure 5.7: Time series of 10 year rolling standard deviation in precipitation, temperature, and runoff. Standard deviations in precipitation increase in CNRMCM5 RCP 8.5 (cool/wet model) and HadGEM2ES RCP 8.5 (the warm/dry model). Standard deviation in temperature increase in MIROC5 RCP 4.5 and CNRMCM5 RCP 4.5. In RCP 4.5, the standard deviations decrease at the end of the century. Again, standard deviations in runoff follows trends seen in precipitation.

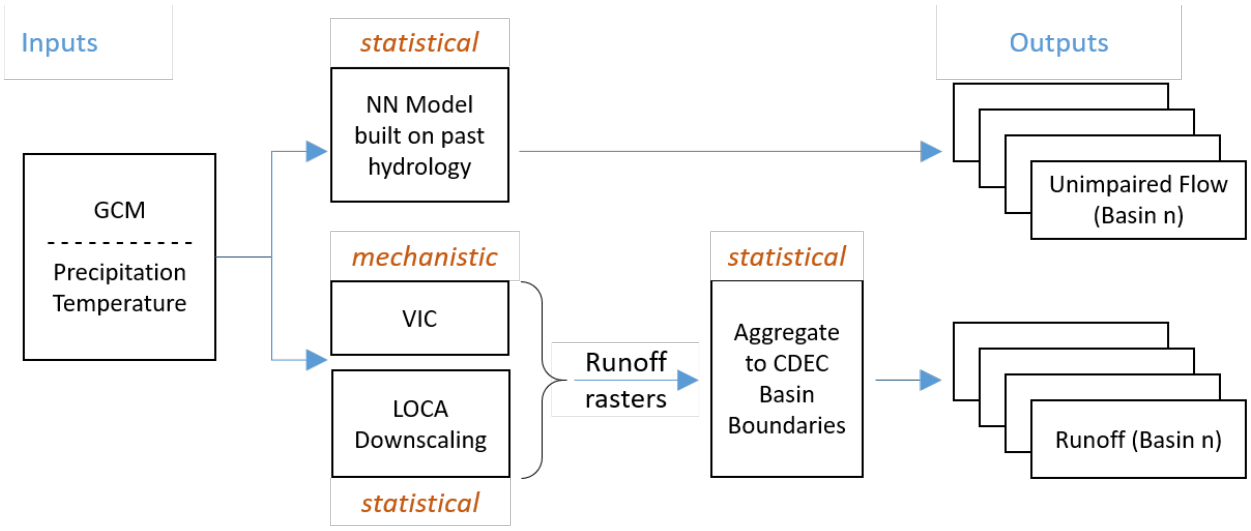


Figure 5.8: Studying future hydrology research design. The predictions from the NN model (purely statistical) were then compared to the downscaled and routed climate change model (mechanistic and statistical) projections.

5.3 Methods

The precipitation and temperature data used here originally came from the Climate Model Intercomparison Project version 5 (CMIP5; Taylor et al., 2012), which includes 32 coarse-resolution ($\sim$ 100 km) GCMs. Pierce et al. (2018) produced runoff rasters with the Variable Infiltration Capacity (VIC) model and bias corrected and downscaled VIC's hydrologic parameters (e.g., precipitation, temperature, and runoff) using the Localized Constructed Analogues (LOCA) statistical method to better capture key features in California's climate (Pierce et al., 2018).

Each downscaled climate change model data is put through the NN model built on past hydrology (model with aggregate data, MSE loss, and LOGO cross-validation resampling), and the model predictions are compared to the climate model projections. Figure 5.8 shows this process. The downscaled climate model projections come in raster (i.e., not routed) format. A simple aggregation to basin boundaries finds the mean precipitation (and its lagged values), mean temperature (and its lagged values), mean snow, and total runoff over the basin. One problem with this simple routing technique is that some end-of-month storms may end up as streamflow in the next month; some precipitation values are getting counted in the month that the flows it generates is not. Since the NN model is on a monthly time-step, we will ignore this small accounting difficulty. The other option was to use the Variable Infiltration Capacity (VIC) routed streamflows that do account for basin lags, however, this data set was only developed for 11 basins, hand selected for the CALSIM II model's major reservoir inflow locations. To keep all our 67 diverse basins in the study, we will use the aforementioned aggregation method as a good approximation for routing.

In the following sections, the VIC+LOCA+Aggregation output will be simply referred to as runoff projections.

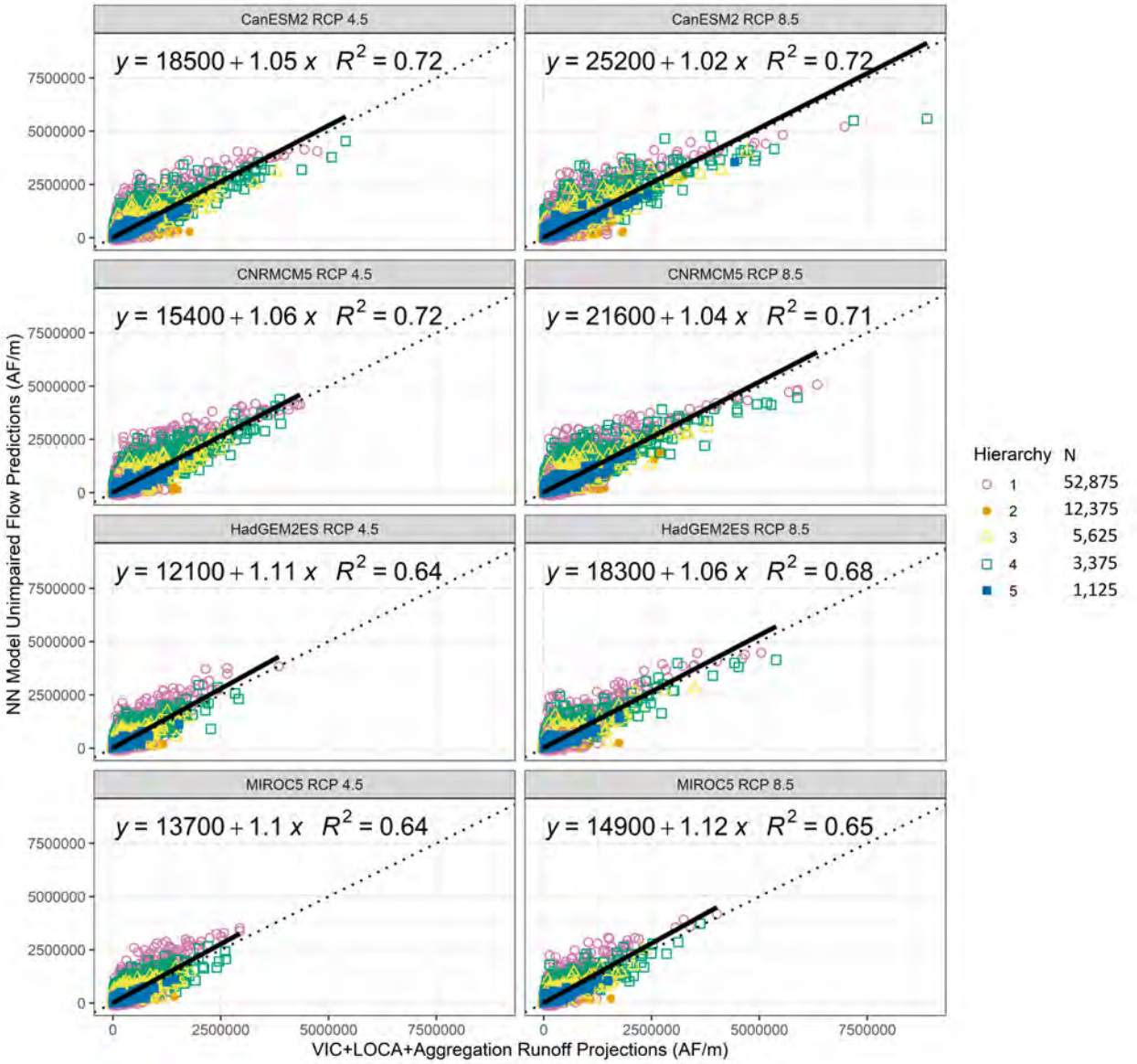


Figure 5.9: NN model predictions compared to runoff projections (monthly data for each basin). There is fairly good agreement between the purely statistical (the NN) and the mechanistic+statistical models (VIC+LOCA+Aggregation) with the highest R^2 being 0.72 for the CanESM2 and CNRMCM5 RCP 4.5 models.

5.4 Results

Figure 5.9 shows the NN model predictions compared to runoff projections for monthly data for each basin. There is fairly good agreement between the purely statistical NN model and the mechanistic and statistical climate models (VIC+LOCA+Aggregation) with the highest R^2 of 0.72 belonging to the CanESM2 and CNRMCM5 models. In all models the NN is slightly biased to predict higher values compared to the climate change models as is evident in the slope of the best fit line ($\beta_1 > 1$).

Figure 5.10 shows the probability densities for the monthly NN unimpaired flow pre-

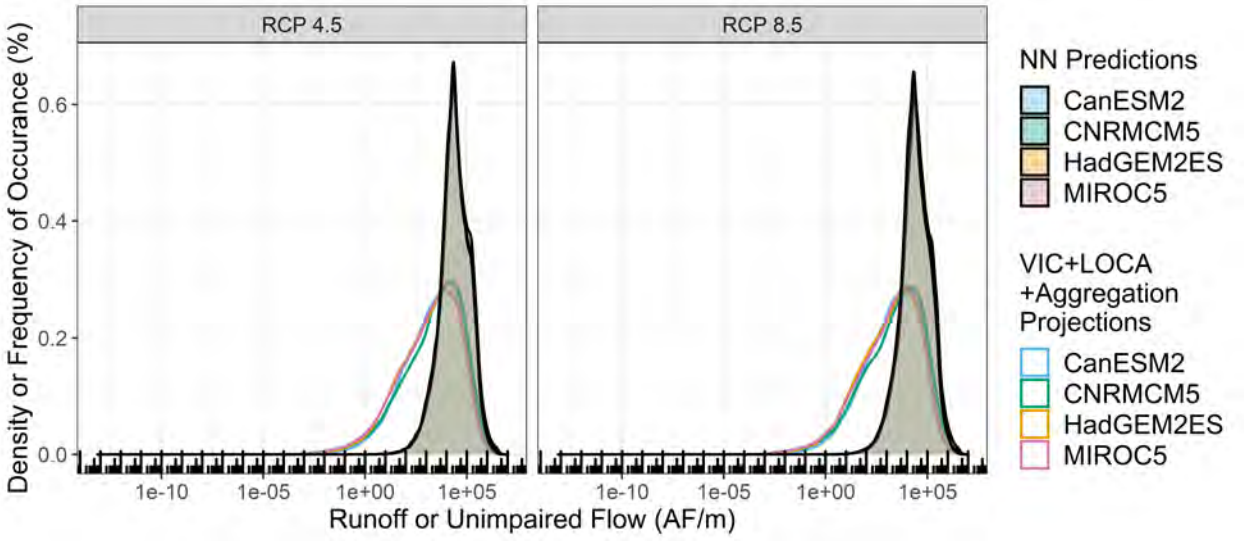


Figure 5.10: Predicted unimpaired flow density compared to projected runoff densities. The NN model does not capture low flows like the VIC+LOCA+Aggregation models. There is very little difference in the densities across the various climate data in either the NN model predictions or runoff projections. Also, there is very little difference observed across the two RCPs.

dictions and the runoff projections. Similar to results in previous chapters, the NN model predicts fewer low flows compared to the runoff from VIC+LOCA+Aggregation models. There is very little difference in the densities across the various climate data in either the NN model prediction or runoff projections themselves. Also, there is very little difference observed across the two RCPs. This shows that our simple routing precipitation to runoff eliminates the differences across models and should be replaced with a better model (e.g., VIC).

Figures 5.11, 5.12, and 5.13 compare the monthly, moving 1 year average, and moving 10 year average NN model predictions with runoff projections. Instead of showing each basin separately, this plot shows the mean flows across the 67 basins. The moving window allows us to view larger trends. Unsurprisingly, as the window grows larger, the values are "smoothed out", and the agreement (in terms of R^2) increases. For example, in the CNRMCM5 RCP 4.5 R^2 increases from 0.77, to 0.90, and 0.96 when comparing monthly, moving 1 year average, and moving 10 year average values. However, with the increase in R^2 the NN model biases increase as well. For example, the slope of the best fit line in CNRMCM5 RCP 4.5 increases from 0.95 to 1.44 and 1.74. This is due to the NN model's inability to capture a higher density at the lower flows, which was observed in Figure 5.10.

Figures 5.14, 5.15, and 5.16 confirm the NN model's bias observed in the previous plots. Here, the monthly, moving 1 year average, and moving 10 year average NN model predictions and runoff projections are plotted through time. As the window grows larger the NN model predictions increase as compared to the runoff projections. This is again due to the NN model's inability to capture a higher density at the lower flows, which was observed in Figure 5.10.

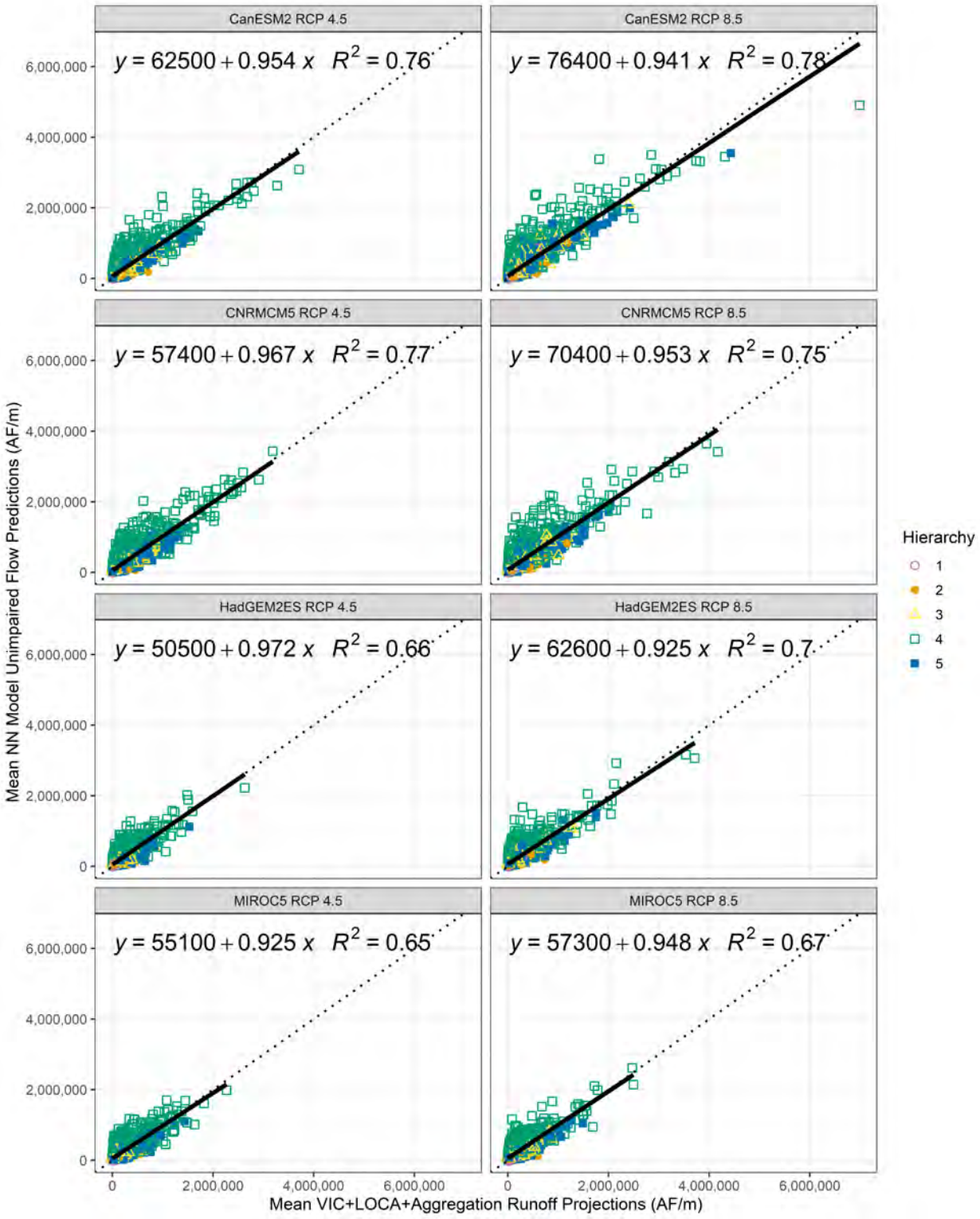


Figure 5.11: Mean California unimpaired flow NN model predictions vs. runoff projections (monthly data). There is fairly good agreement between the two types of models on the average California flows. Compared to unaggregated monthly data model agreement slightly increases as the data gets aggregated across California.

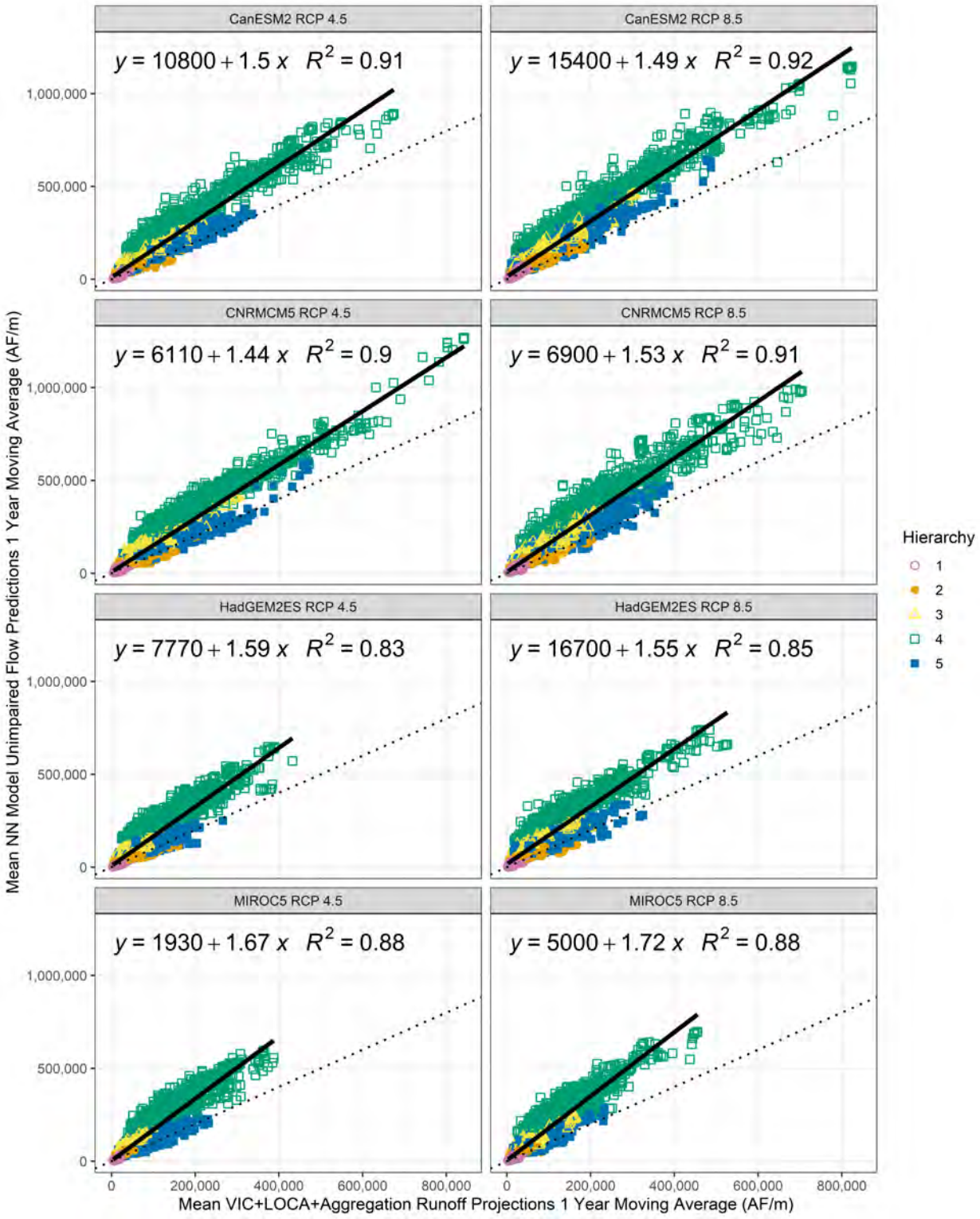


Figure 5.12: Mean California unimpaired flow NN model predictions vs. runoff projections (annual moving average data). There is fairly good agreement between the two types of models on the average California flows. Compared to monthly data, here, model agreement increases slightly.

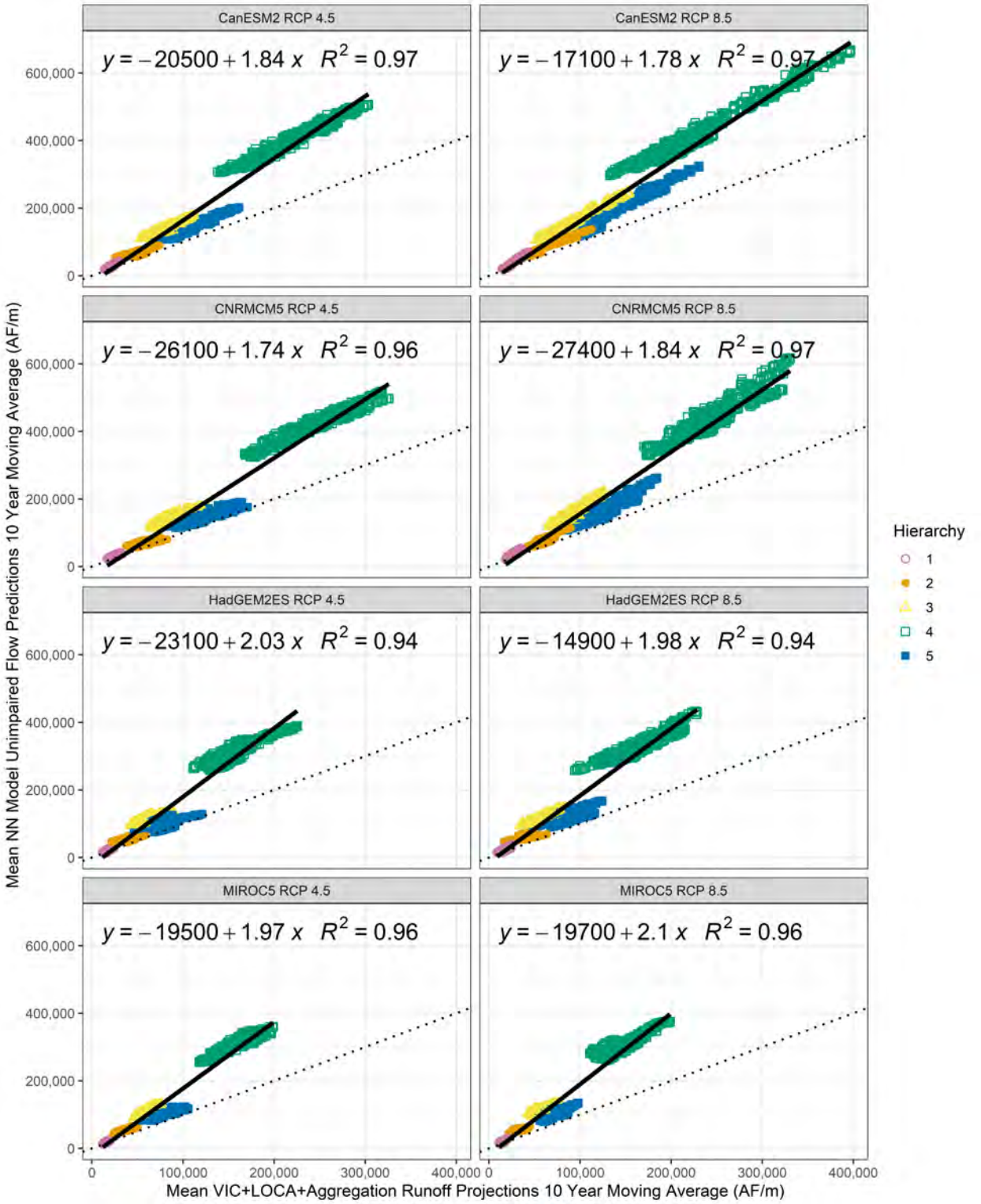


Figure 5.13: Mean California unimpaired flow NN model predictions vs. runoff projections (10 year moving average data). There is fairly good agreement between the two types of models on the average California flows. Compared to 1 year moving average data, here, agreement increases slightly but so does biases (slope of the best fit line). Averaging also produces a smoothing effect, which further distinguishes hierarchies by lumping them in groups.

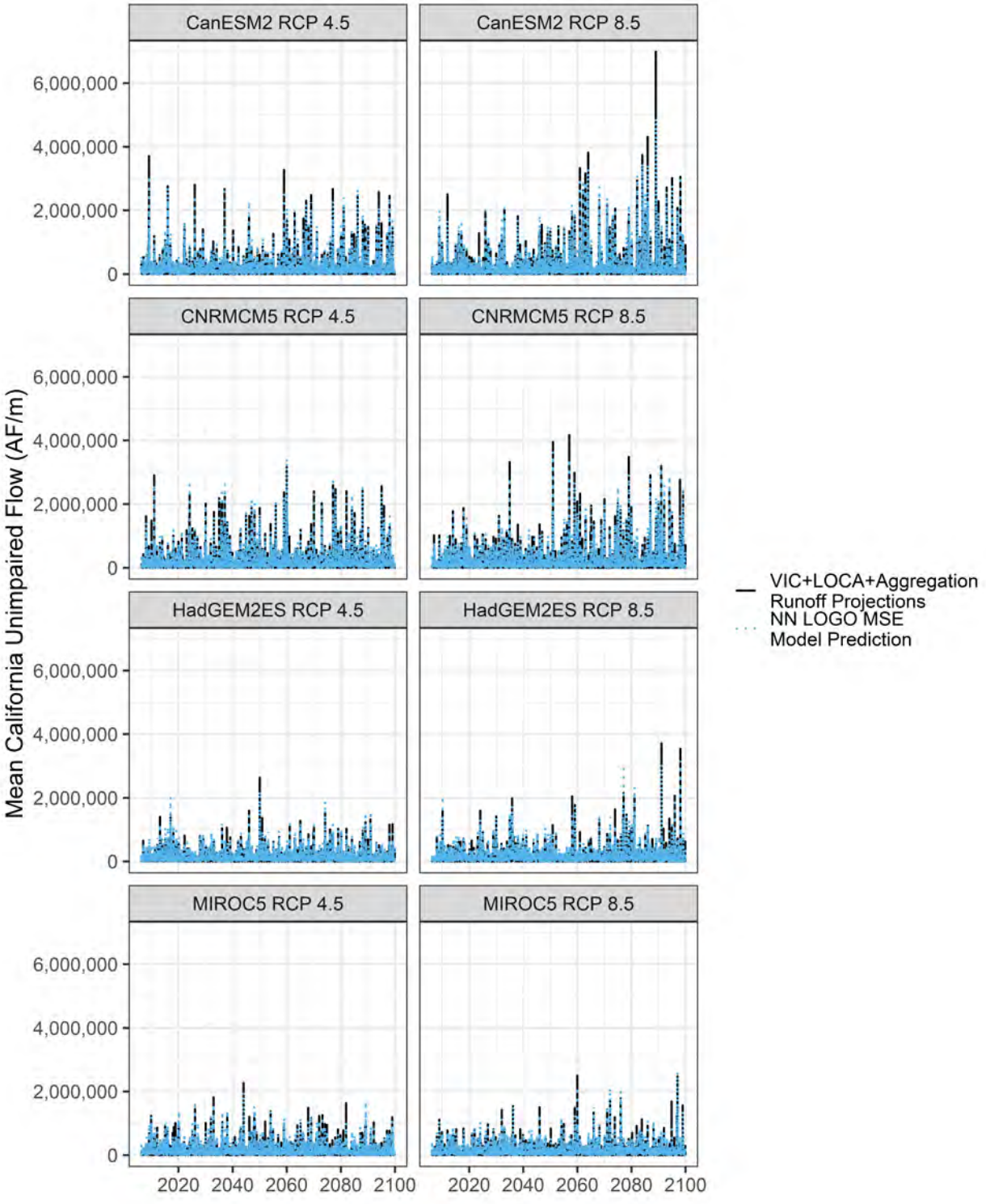


Figure 5.14: Mean California NN model unimpaired flow and runoff projections comparisons in time (monthly data). The NN model captures the high flow events at the right time but is slightly under predicting them compared to the climate model projections.

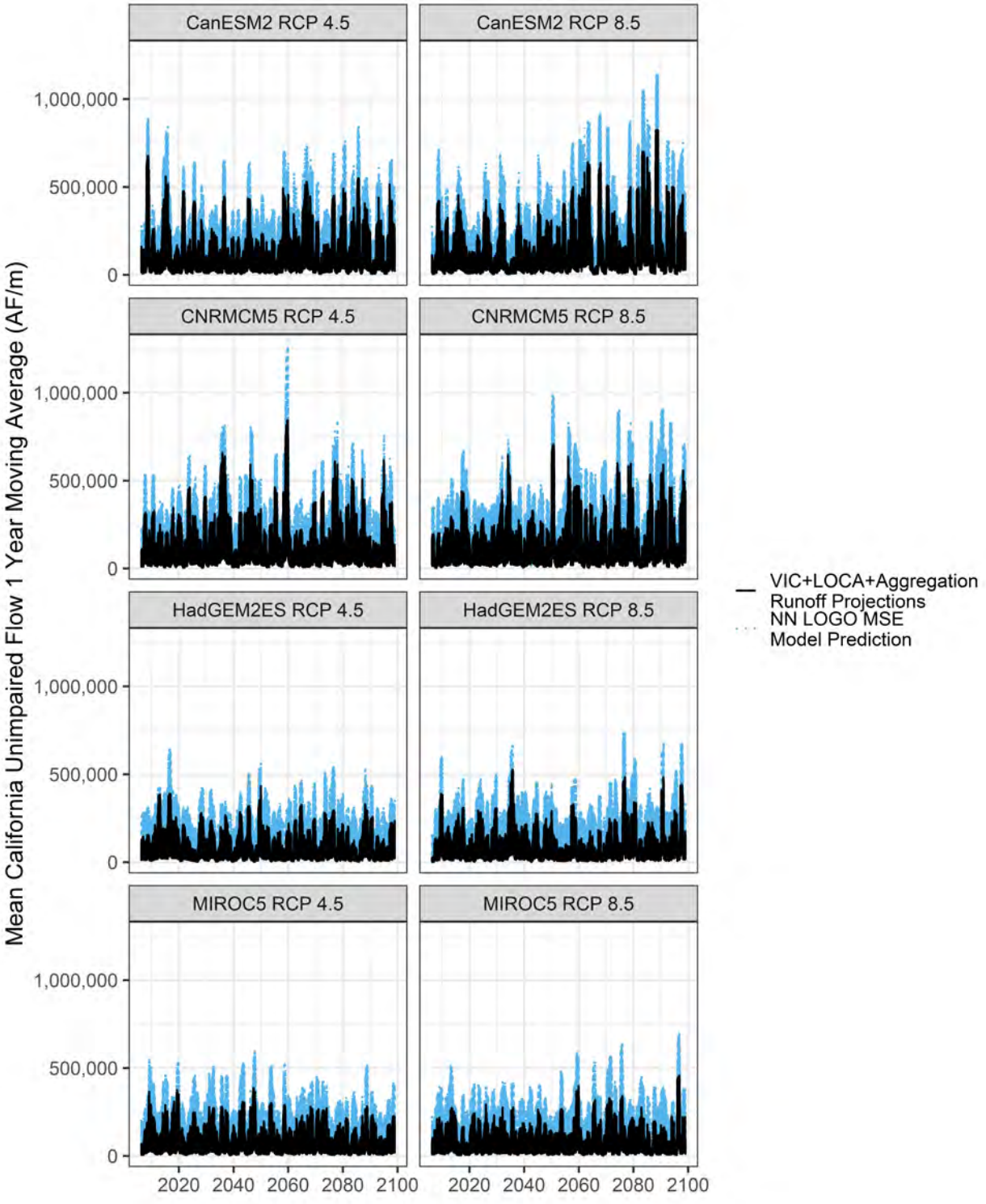


Figure 5.15: Mean California NN model unimpaired flow and runoff projections comparisons in time (1 year moving average data). The NN model's predictions over-take the climate model projections.

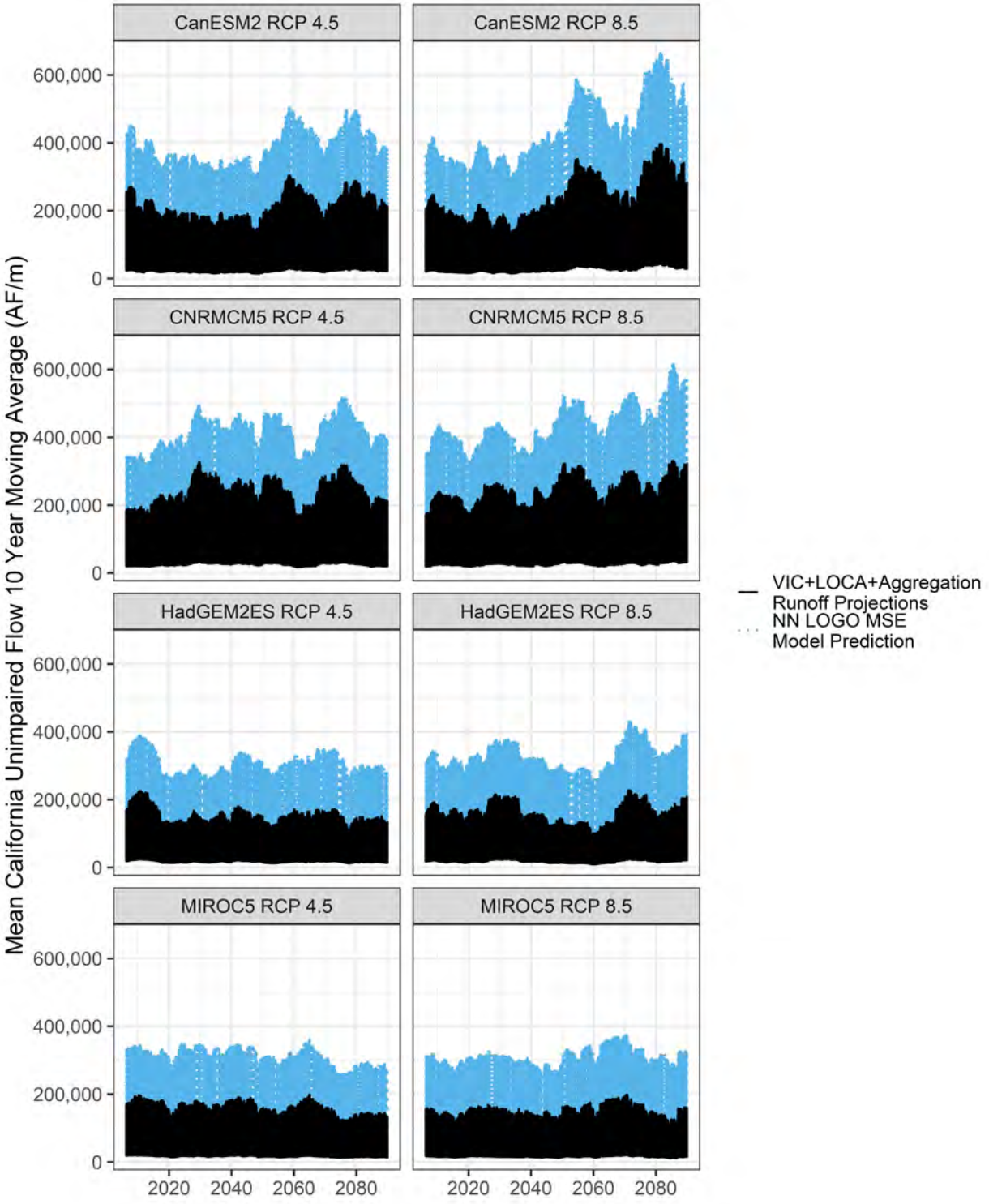


Figure 5.16: Mean California NN model unimpaired flow and runoff projections comparisons in time (10 year moving average data). The NN model's predictions over-take the climate model projections and a larger moving window means larger differences.

5.5 Conclusion

This chapter used four global climate models with two RCPs each to estimate future hydrology. The LOCA downscaled climate variable from these models give a wide range of possible futures for California. Some models like the MIROC5 RCP 4.5 and CNRMCM5 RCP 4.5 project a wetter California while most other models project a drier climate in terms of precipitation. Most models agree that in both RCPs, temperatures will rise and are projected to increase 2-4 °C for RCP 4.5 and 4-7 °C for RCP 8.5 (Pierce et al., 2018). Runoff projections show more floods compared to historical hydrology but are stationary in their mean and standard deviations in all but one model (CNRMCM5, the cool/wet model).

Statistical models operate from no a priori knowledge of hydrologic processes, and have to be used with caution when extrapolating beyond the time range they were trained on. There is fairly good agreement in the statistical (NN) model's unimpaired flow predictions and the mechanistic+statistical (VIC+LOCA+Aggregation) model's routed runoff (R^2= [0.64-0.72]). However, the NN model does not capture low flows like the VIC+LOCA+Aggregation models and overestimates their values so much so that when we compare more smoothed data (with a moving average window) we can see a bias emerge (β_1= [0.95-1.84] for CanESM2 RCP 4.5 for example). This can also be seen in the time series comparisons, where with a larger moving average window the NN model's predictions are systematically higher than the climate model projections.

Both runoff projections and NN model predictions are untestable since the "reality" we need to test against will be available either too late or never-an unavoidable feature of all hydrologic simulation models discussed by Klemeš (1986). Therefore, the climate changed experiment is much like the problem of ungauged basins where the "true" test set is one which has no observations. We can argue that the runoff projections are slightly more reliable since the processes of finding the amount of recharge and runoff for each pixel ($\sim$ 100 km) is grounded in hydrology (VIC model). However, the downscaling to a finer resolution was a statistical process (i.e., LOCA downscaling), and the routing scheme used here was a simple statistical aggregation to basin boundaries. Therefore, even in the runoff projections many approximations are used to arrive at water balance estimates. The advantages of using statistical learning models still remain; they are easier to use, apply, and operationalize. The next chapter will explain some improvement strategies for the NN model.

Chapter 6

Overall Conclusions and Future Directions

This dissertation develops statistical learning models, generally simpler than mechanistic models, to predict unimpaired flows of California basins from available data. Several issues arise in this prediction problem:

(1) How we view hydrology, and how we define an observational unit, determines how data is pre-processed for statistical learning methods. So, one issue is in deciding the organizational form of the data (i.e., aggregate vs. incremental basins). Chapter 2 showed that "incremental basin" modeling provides an easy way to include network information in statistical models, and the results show its value for modeling hydrology with parametric models, especially those with few parameters like LM and GLM. The LM and GLM benefit more from the incremental modeling approach, whereas the RF and the NN are somewhat insensitive to it.

(2) Often, water resources problems are not concerned with accurately predicting the expectation (or the mean) of a distribution but require better estimates of extreme values of the distribution (i.e., floods and droughts). Solving this problem involves defining asymmetric loss functions presented in Chapter 3. In symmetric loss functions such as the squared error loss functions (i.e., Mean Squared Error or MSE), the peaks or high leverage points get fitted at the expense of the low flows. The proposed Weighted Least Squared Error (WLSE) and Linear-Exponential Error (LINEXE) asymmetric losses are able to force a fit to the tails of the distribution (the peaks and valleys of the hydrograph). The symmetric Log-Cosine Hyperbolic Error (LOGCOSH) performs similarly to the Mean Absolute Error (MAE) and MSE. The Mean Squared Percentage Error (MSPE) is chronically biased towards lower predictions and is not suited to problems where the data are skewed positive. In general, the differences show the amount of control the modeler has on the predictions and their probability distribution when picking a loss function.

(3) Dependencies and correlation structures are inherent in hydrologic observations; gauge data are structured in time and space, and rivers form a network of flows that feed into one another (i.e., temporal, spatial, and hierarchical autocorrelation). These characteristics require careful construction of resampling techniques for model error estimation. In Chapter 4, blocking methods show how much random resampling methods underestimate model error. Models evaluated with random methods (e.g., Random 5-fold) have artificially low errors

due to pseudoreplication from autocorrelation. This is not to say that, in hydrology, random resampling is never useful; a random test-train split is most appropriate for predicting flow for a sparsely incomplete gauge record. Blocked resampling in time is most appropriate for predicting or extrapolating streamflow in time for that location. One should not expect to use these resampling strategies and get the same predictive accuracy in a purely ungauged basin problem, where blocks are supposed to be designed across geographic space or more accurately hierarchical structure (e.g., LOGO and BBG). Results show that generally model performance estimates decline as block sizes increase. However, these estimates are more accurate than random resampling methods since they better approximate the data generating mechanism. This chapter highlights the importance of the field in moving away from 5-fold or 10-fold random cross-validation to blocked cross-validation and eventually to blocked bootstrapping methods.

(4) Non-stationarity due to climate change may require adjustments to statistical models, especially if they are meant for long-term decision-making. Chapter 5 compares unimpaired flow predictions from a statistical model (NN) that uses climate variables representing future hydrology to projections from routed climate models simply aggregated to basins in the study (VIC+LOCA+Aggregation). There is fairly good agreement in the statistical (NN) model's unimpaired flow predictions and the mechanistic+statistical (VIC+LOCA+Aggregation) models runoff projections (R^2= [0.64-0.72]). However, the NN model predicts more low flows than the VIC+LOCA+Aggregation models; when we compare more smoothed data (with a moving average window), we can see a bias emerge (e.g., β_1= 0.95 to 1.84 for CanESM2 RCP 4.5). This can also be seen in the time series comparisons, where with a larger moving average window the NN model's predictions are systematically higher than the VIC+LOCA+Aggregation model projections.

6.1 Model Improvement Strategies

Figure 6.1 shows the residuals for the NN model built on observed data in the previous chapters (Appendix A). Model residuals are positive for smaller hierarchies and negative for larger ones. This was evident by the fact that the model overpredicts lower flows (that tend to occur at smaller hierarchies) and underpredicts higher flows (that tend to occur at larger hierarchies). Also, residuals tend to increase with increasing flow (Figure 6.2), but standardized residuals decrease with flow (Figure 6.3).

Overall, the NN model residuals may appear to be spaced at random. However, with the Box-Pierce and Ljung-Box tests for autocorrelation we can see that in most basins they are not (Figure 6.4). These tests, also known as the **portmanteau** tests, are used for examining the null hypothesis of independence in a given time series. P-values close to zero are evidence against independence and imply that the model can be improved to eliminate non-random patterns in the residuals.

Model improvement can come from taking advantage of the knowledge that flow data is sequential and therefore using models that focuses on the **time series** component of the data. For example, some models use a lagged response variable to construct the model as in the Auto Regressive Integrated Moving Average (ARIMA) models. These models are uni-variate and are solely trained on the response variable; in order to accommodate predictor (exogenous) variables, we can use a Seasonal ARIMA (SARIMA). In ungauged

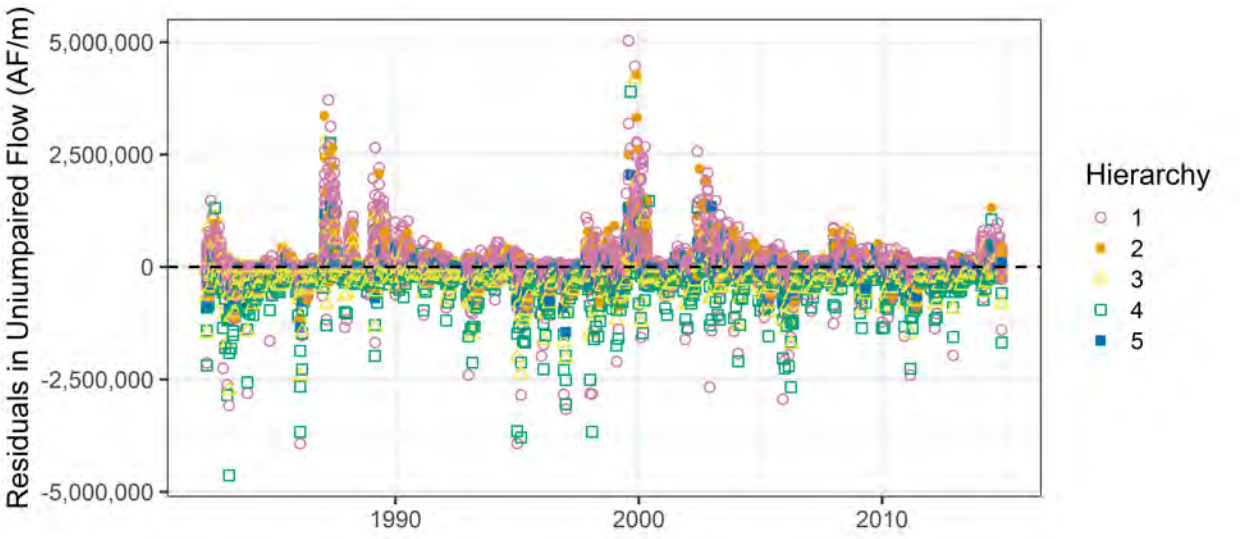


Figure 6.1: NN model residuals over time. Residuals may appear random as a whole, however, they tend to be positive for smaller hierarchies and negative for larger ones.

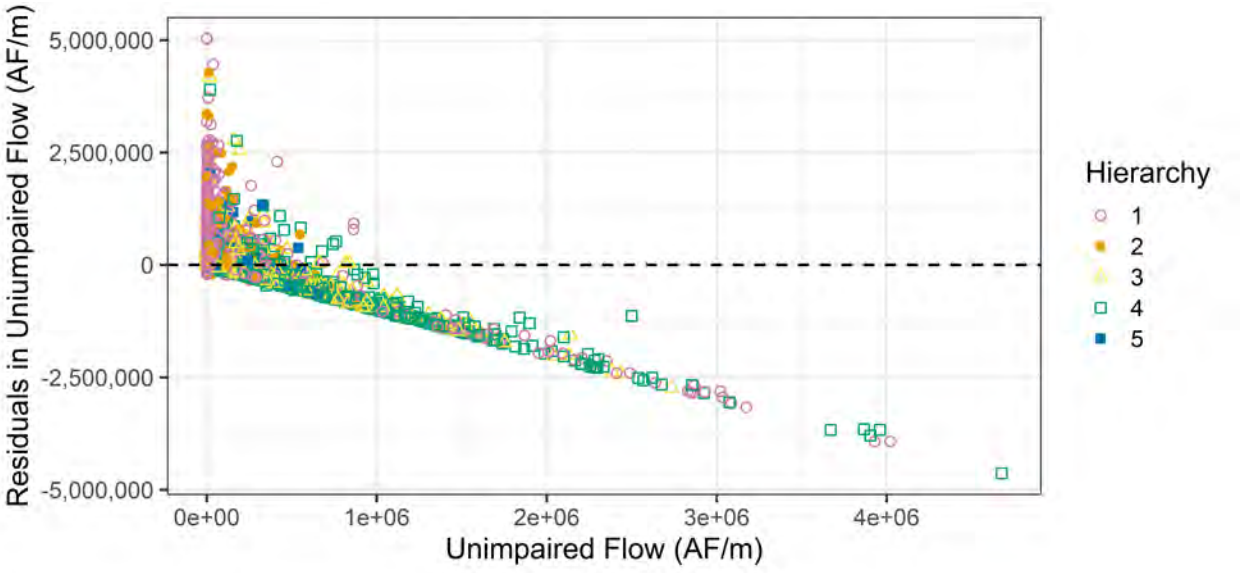


Figure 6.2: NN model residuals vs. unimpaired flow (CDEC). Model residual increase with an increasing response variable. Most of these flows occur at the larger hierarchy (4) as these basins are lower in the network and are expected to have larger flows.

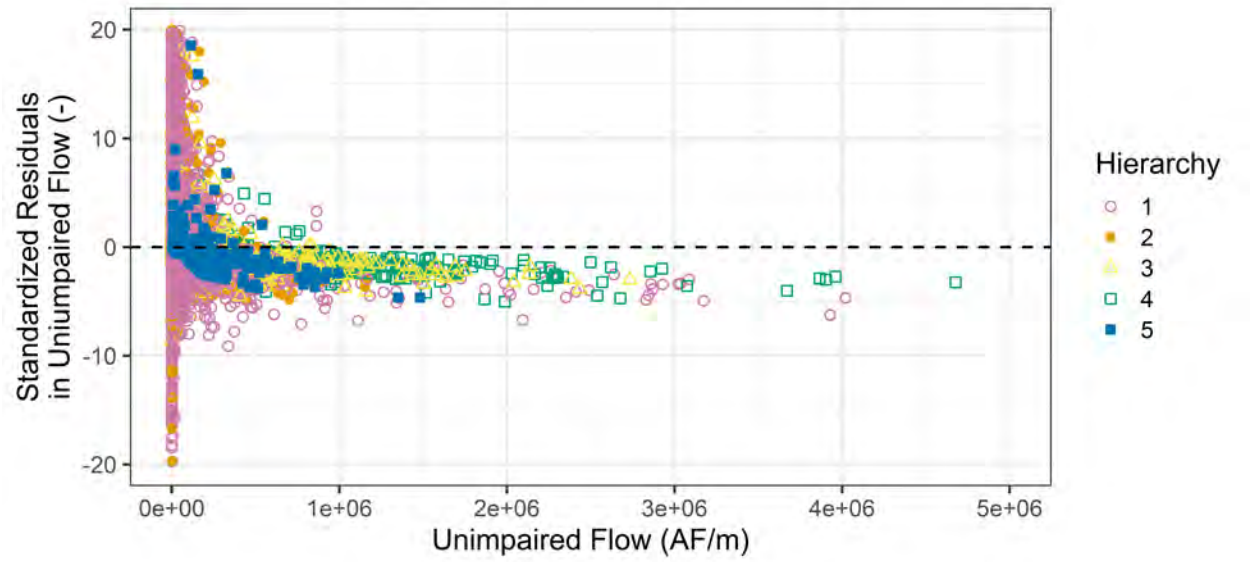


Figure 6.3: Standardized NN model residuals vs. unimpaired flow (CDEC). The ratio of model residuals to mean annual observed unimpaired flow decrease with an increasing response variable. Most relative errors occur at smaller hierarchies (1) as these basins are higher in the network and are expected to have smaller flows.

scenarios, to avoid information leaks we must eliminate training data that the model has seen, creating a problem with small data sets. To avoid this issue altogether, models like Recurrent Neural Networks (RNNs) have the capability of connecting previous information to the present in their architecture. Therefore, unlike ARIMAs, there is no need to include the response variable in the training. A special type of RNNs, Long Short Term Memory networks (LSTMs) are capable of learning long-term dependencies with fewer memory needs and have shown to be effective in time series modeling for PUB (Kratzert et al., 2019).

Also, semi-supervised statistical learning models aid in learning in **non-stationary** environments. A special case is known as **covariate shift**, in which the distributions of inputs (queries) change but the conditional distribution of outputs (answers) is unchanged. For example, with climate change, the distribution of precipitation flattens (more extreme weather) and the distribution of temperature shifts (hotter climate), but the runoff produced by a particular combination of temperature and precipitation is unchanged. In this case, covariate shift adaptation weights the importance of each training observation based on the probability it will be queried later. So, the climate change queries (input-only data) are included in the learning process along with the observations (input-output data), and the importance of each training observation is considered with an importance-weighted loss function (Sugiyama & Kawanabe, 2012). In some instances, even without query data, covariate shift is intentionally implemented in the modeling to improve generalization ability.

As we saw, some models predict **physically impossible flows**: negative flows, or more flow than there is in precipitation volumes. These unrealistic predictions can be eliminated by: (1) Constrained optimization where a high penalty is imposed when predicted flows are outside a prescribed range of flows (Ren, Stewart, Song, Kuleshov, & Ermon, 2018). The penalty can make the loss function be non-differentiable and therefore need more creative fit-

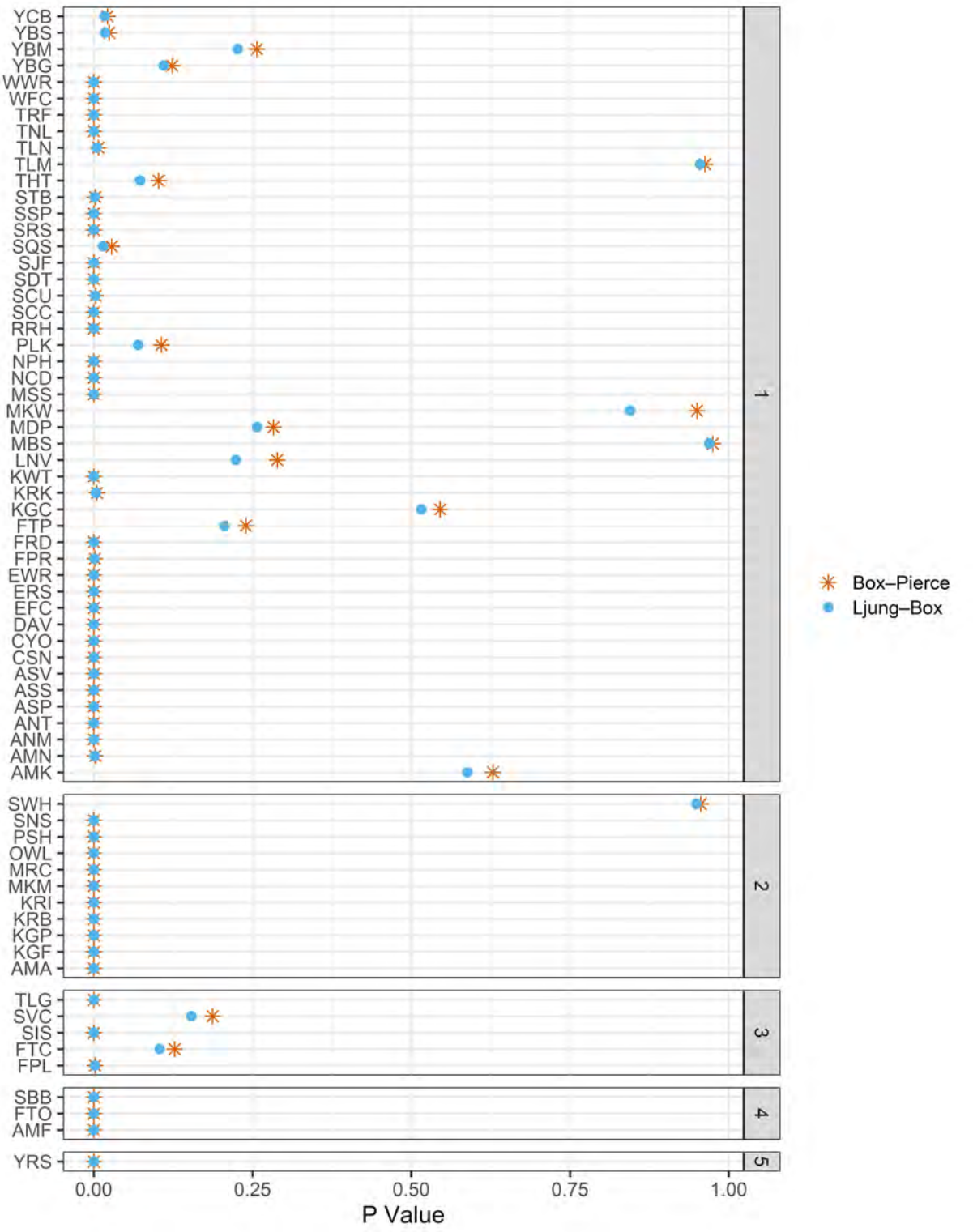


Figure 6.4: Box-Pierce and Ljung–Box tests for auto-correlation in model residuals (predicted - CDEC unimpaired flows). The p-values close to zero indicate that residuals are auto-correlated. Most basins suffer from 0 p-values indicating that the model can be improved to eliminate non-random patterns in its residuals.

ting techniques, or the penalty can be added to the loss function as a regularization method. This added term gives a differentiable measure of how close the model is to satisfying the constraint. (2) Adding a constrained layer to the neural architecture design to enforce constraints. The constrained layer increases the pre-activated value of a neuron if a constraint is met and decreases it otherwise. Because constraint violation is a binary variable (0 for violated, 1 for not) and is non-differentiable, a smooth surrogate must be used (Li & Srikumar, 2019). (3) Data augmentation, i.e. the addition of supplementary data sets that follow constraints, incentivizes networks to be more mindful of constraints. The infusion of domain knowledge with a simulation model also helps with data sets of limited size and the related issue of poor generalization performance.

Model improvement may also come from a more **granular data set**; daily rather than monthly data can increase model performance, since the variable importance plots showed that precipitation rates, which vary through time, contain the most amount of information needed for accurate predictions.

6.2 Final Thoughts

In general, rainfall-runoff models can be used inside hydrology, as exploratory research tools, or outside hydrology, for planning, design, or operational decisions. The models developed here are intended for use outside hydrology where hydrologic information is relevant and useful. I started studying the PUB problem during my undergraduate years, because streamflow estimates were needed as a pre-requisite to the original task of estimating nutrient loadings. The model was ultimately intended to inform the Kentucky Division of Water in establishing total maximum daily loads for Nitrogen and Phosphorous. This model was simple, transparent, and helped evaluate different nutrient management options. Like this application, streamflow dis-aggregation and water budget outputs can be used to aid in setting minimum in-stream flow regulations, establishing water rights, modeling river stage-discharge relationships or water levels in a channel, simulating sediment and nutrient loadings, cleaning up anomalies in the data or filling in missing records, and managing natural disasters.

Appendix A

Model Data

This appendix introduces the data used in the statistical learning model.

Study Area & Response Variable

This study used the monthly unimpaired flows dataset developed and maintained by the California Data Exchange Center (CDEC). The data spans 67 California basins (Figures A.1 and A.2, and Table A.1) from 1982 to 2014. It can be downloaded with a simple web-scraping script available on GitHub. It has approximately 19,000 monthly streamflow observations in acre-feet (AF) and as a continuous variable can be used for regression type studies (Figure A.3).

Table A.1: CDEC unimpaired flow gauges included in this study.

No.	ID	Name	Hierarchy	Longitude	Latitude	Area (km^2)
1	AMA	AMERICAN MF NR AUBURN	2	-2128686.67	2042808.70	2517.49
2	AMF	AMERICAN R AT FOLSOM	4	-2144854.69	2025238.28	4845.51
3	AMK	AMERICAN SF NR KYBURZ	1	-2071111.72	2014979.17	494.37
4	AMN	AMERICAN NF AT N FORK DAM	1	-2124144.54	2049024.85	886.93
5	ANM	SANTA ANA R NR MENTONE	1	-1916225.50	1440652.95	543.45
6	ANT	ANTELOPE LAKE	1	-2053955.06	2174556.63	183.94
7	ASP	ARROYO SECO (PASADENA)	1	-2009350.30	1475311.58	43.05
8	ASS	ARROYO SECO NR SOLEDAD	1	-2227364.03	1768676.13	626.27
9	ASV	AMERICAN-SF SILVER CREEK	1	-2072796.09	2021792.99	71.59
10	CSN	COSUMNES R AT MICHIGAN BAR	1	-2138635.20	2002305.87	1384.43
11	CYO	COYOTE CR NR MADRONE	1	-2228858.29	1872199.32	507.07
12	DAV	LAKE DAVIS (DWR)	1	-2051002.47	2139358.74	121.48
13	EFC	EAST FK CARSON RIVER NR GARDNERVILLE	1	-2016973.86	2010608.38	928.85
14	ERS	EEL RIVER AT SCOTIA	1	-2327255.02	2288682.16	8069.24
15	EWR	EAST WALKER RIVER NR BRIDGEPORT	1	-1990272.14	1943888.95	944.67

No.	ID	Name	Hierarchy	Longitude	Latitude	Area (km^2)
16	FPL	FEATHER NF AT PULGA	3	-2134185.57	2151395.21	5062.47
17	FPR	FEATHER NF NEAR PRATTVILLE	1	-2093742.59	2183928.16	1299.27
18	FRD	FRENCHMAN DAM	1	-2027714.74	2133264.26	226.81
19	FTC	FEATHER MF NR CLIO	3	-2065191.81	2128154.52	1772.44
20	FTO	FEATHER RIVER AT OROVILLE	4	-2150136.78	2124168.74	9374.46
21	FTP	FEATHER SF AT PONDEROSA	1	-2129324.12	2121485.04	278.67
22	KGC	KINGS NF NR CLIFF CAMP	1	-1999299.27	1792108.25	6.42
23	KGF	KINGS R PINE FLAT DAM	2	-2041219.40	1783693.80	4001.10
24	KGP	KINGS PRE PROJECT PIEDRA	2	-2040307.84	1783696.42	4000.12
25	KRB	KERN R BAKERSFIELD	2	-2045237.12	1623254.90	6229.30
26	KRI	KERN R BLW ISABELLA	2	-1999491.62	1635957.72	4975.50
27	KRK	KERN R NEAR KERNVILLE	1	-1990852.72	1669117.93	2197.81
28	KWT	KAWEAH R TERMINUS DM	1	-2024002.66	1731041.34	1427.86
29	LNV	LONG VALLEY TO TINEMAHA	1	-1978782.77	1871636.59	283.25
30	MBS	MONO BASIN	1	-1993147.05	1904294.06	215.36
31	MDP	MERCED AT POHONO BRIDGE	1	-2045021.97	1887054.05	835.76
32	MKM	MOKELUMNE MOKELUMNE HILL	2	-2116905.46	1974783.27	1414.03
33	MKW	MOKELUMNE AT WEST POINT	1	-2100680.10	1976858.85	186.68
34	MRC	MERCED R NR MERCED FALLS	2	-2106863.79	1880441.07	2748.96
35	MSS	MCCLOUD RIVER ABOVE SHASTA LAKE	1	-2161448.62	2294358.06	1575.08
36	NCD	NACIMIENTO R BLW DAM	1	-2202248.56	1701662.95	858.59
37	NPH	NAPA R NR ST HELENA	1	-2253820.95	2034051.71	211.92
38	OWL	OWENS RIVER-LONG VLY	2	-1967405.66	1852775.42	994.11
39	PLK	SF PIT RIVER NEAR LIKELY	1	-2010166.22	2284708.33	673.47
40	PSH	PIT RIVER AT SHASTA LAKE	2	-2148617.08	2277359.96	13240.32
41	RRH	RUSSIAN R NR HEALDSBERG	1	-2284103.06	2056180.11	2058.80
42	SBB	SACRAMENTO RIVER ABV BEND BRIDGE	4	-2179184.13	2221503.29	23474.25
43	SCC	SUCCESS DAM	1	-2026405.06	1691144.77	1011.73
44	SCU	SILVER CR AT UNION VALLEY	1	-2076668.35	2028174.76	216.89
45	SDT	SACRAMENTO R AT DELTA	1	-2177823.22	2296916.82	1086.65
46	SIS	SACTO INFLOW SHASTA	3	-2184967.47	2273101.54	17098.24
47	SJF	SAN JOAQUIN RIVER BELOW FRIANT	1	-2070323.68	1808777.82	4350.67
48	SNS	STANISLAUS R GOODWIN	2	-2123287.23	1923015.75	2546.40
49	SQS	SQUAW CR INFLOW-SHASTA LAKE	1	-2153666.60	2280408.47	165.81
50	SRS	SALMON R AT SOMES BAR	1	-2248832.40	2368685.00	1945.97
51	SSP	SESPE CREEK NR FILLMORE	1	-2070163.63	1515286.95	662.87
52	STB	STANISLAUS MF BLW BEARDS	1	-2068257.97	1948099.97	812.74

No.	ID	Name	Hierarchy	Longitude	Latitude	Area (km^2)
53	SVC	SILVER CR BLW CAMINO DAM	3	-2087127.00	2026143.25	442.74
54	SWH	SESPE CK AT WHEELER	2	-2074131.12	1509395.89	693.90
55	THT	TINEMAHA TO HAIWEE	1	-1939473.99	1786499.22	98.68
56	TLG	TUOLUMNE R-LA GRANGE DAM	3	-2112073.47	1898492.85	3988.05
57	TLM	TUOLUMNE CHERRY CREEK	1	-2059288.72	1919713.74	302.09
58	TLN	TUOLUMNE NR HETCH HETCHY	1	-2050052.23	1913899.28	1175.09
59	TNL	TRINITY R AT LEWISTON	1	-2215692.33	2282053.74	1862.19
60	TRF	TRUCKEE RIVER AT FARAD	1	-2028165.07	2080706.71	2430.66
61	WFC	WEST FORK CARSON RIVER AT WOODFORDS	1	-2029890.03	2004884.38	170.41
62	WWR	WEST WALKER RIVER NR COLEVILLE	1	-2008642.77	1954269.52	166.77
63	YBG	YUBA NF BELOW GOODYEARS BAR	1	-2099908.30	2110853.00	645.47
64	YBM	YUBA MF NR JACKSON MDWS	1	-2068784.00	2100936.65	97.69
65	YBS	YUBA SF BLW SPAULDING	1	-2082870.09	2082407.80	319.76
66	YCB	YUBA CANYON CK/BOWMAN LK	1	-2079638.13	2095580.35	73.80
67	YRS	YUBA RIVER NEAR SMARTVILLE	5	-2136165.99	2087011.27	2871.21

Some unimpaired flow basins in the CDEC dataset were removed from this study: (1) YBJ. YBJ (INFL JACKSON MDWS & BOWMAN) and YBM (YUBA MF NR JACKSON MDWS) basin outlets are close to one another but have slightly different flow data. Upon further inspection we learned that YBJ=YBM+USGS Gauge after a diversion. Therefore, YBM is the truly unimpaired flow gauge. (2) SFJ and OTR. These gauges were in the dataset but were stripped of their data. We are leaving them out until CDEC updates their database including these basins. (3) KLO. KLO is downstream of SFJ. Since SFJ was pulled from the dataset for revision, we removed KLO, believing its calculated values will also be effected. (4) BHN, FTM, SFR, and SJM. These stations were discontinued and as such their time series do not overlap with the rest of the basins. That gives us a total of 67 (75 total - 8 omitted) unimpaired flow basins. Figure A.2 shows how these basins are connected.

Predictor Variables

Predictor attributes were calculated for each observation point (Table A.3). A total of 24 predictor variables were selected based on the knowledge of basin characteristics and processes that influence a watershed's response to precipitation: evaporation (temperature); snowfall (cumulative sum of precipitation below 2°C); storage in soil (with soil and land cover parameters); antecedent conditions (with lagged precipitation and temperature parameters); and groundwater processes (with depth to restricted layer).

The climate data were derived from the Parameter elevation Regression on Independent

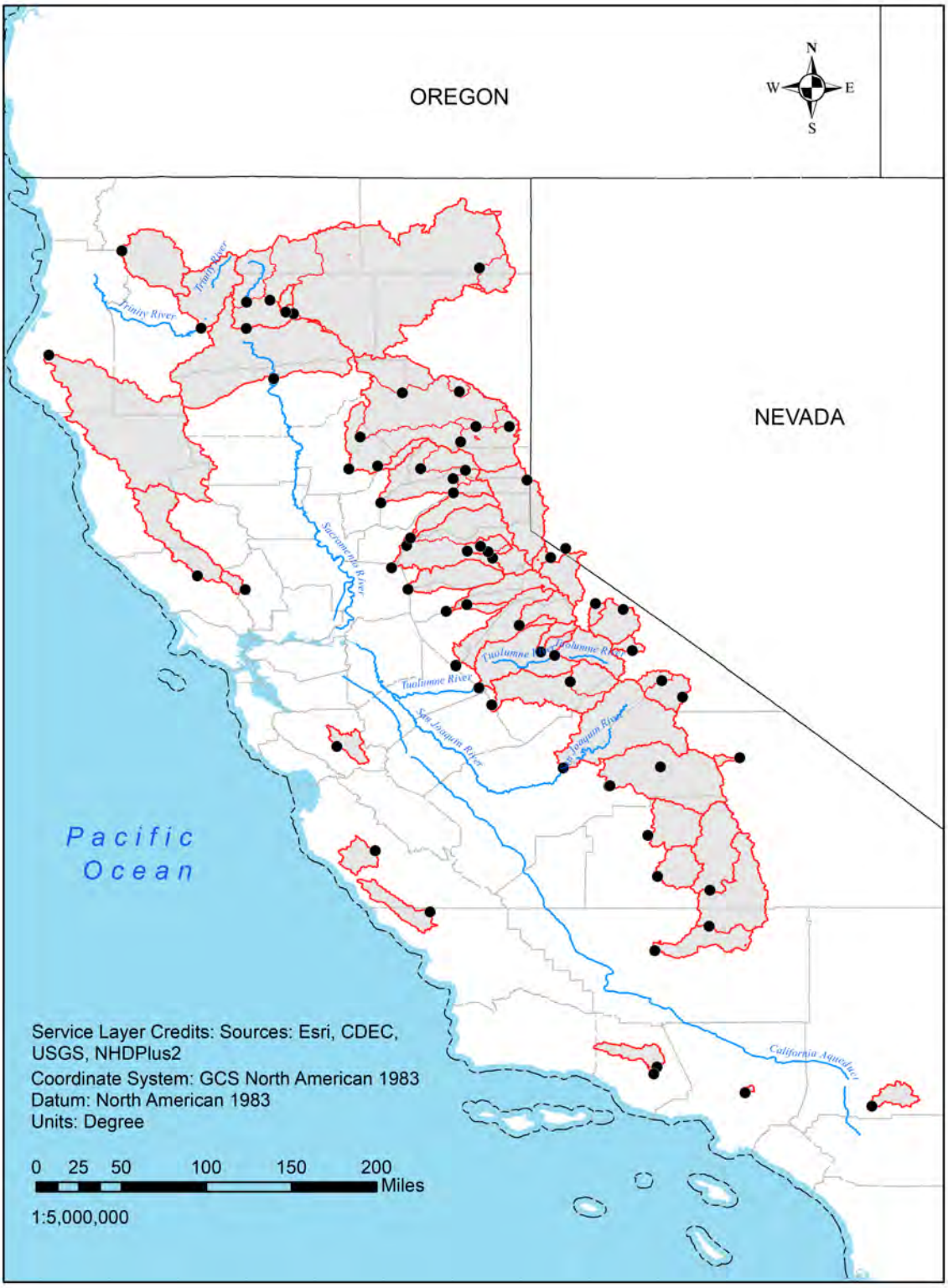


Figure A.1: The 67 California basins under study are the CDEC unimpaired flow basins.

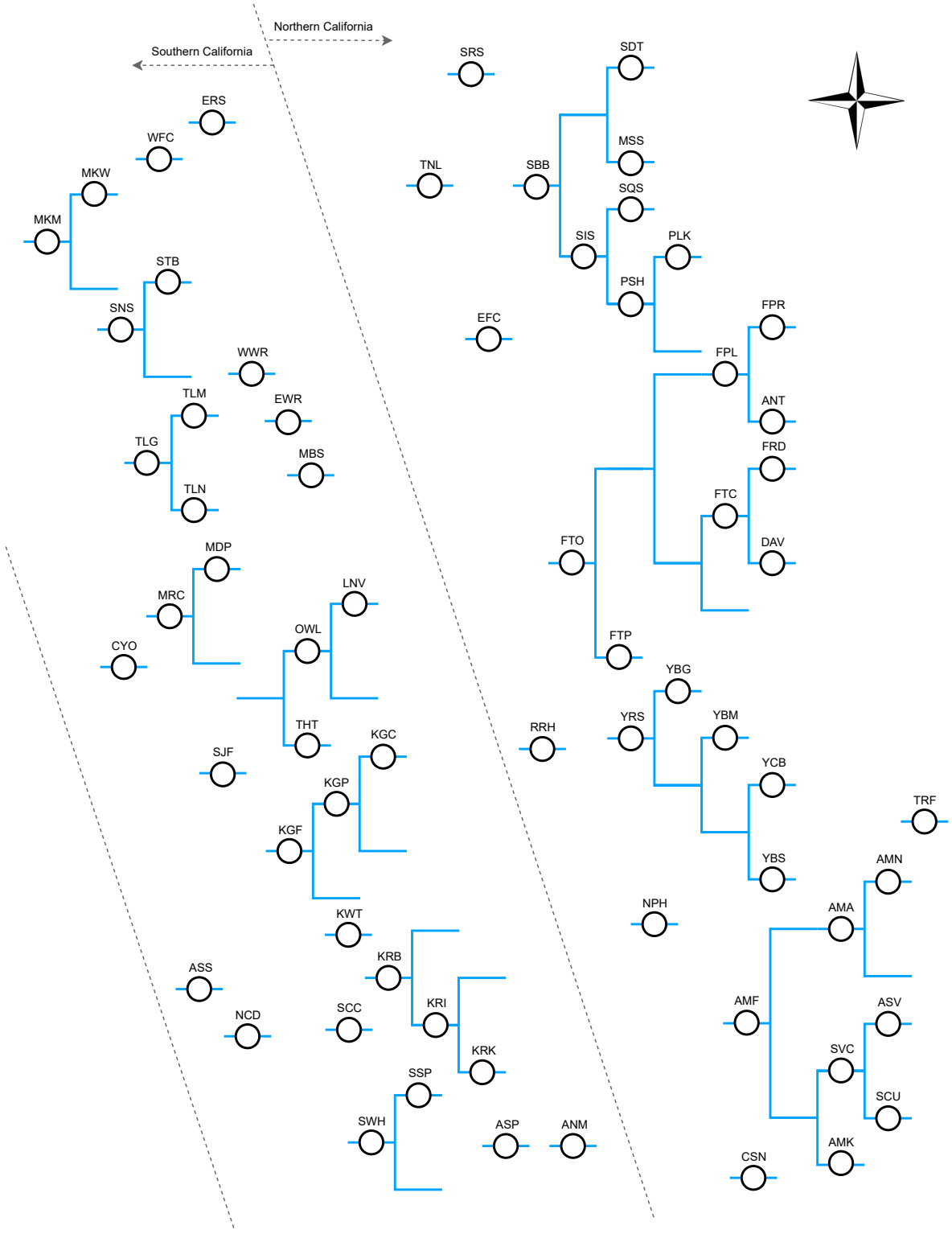
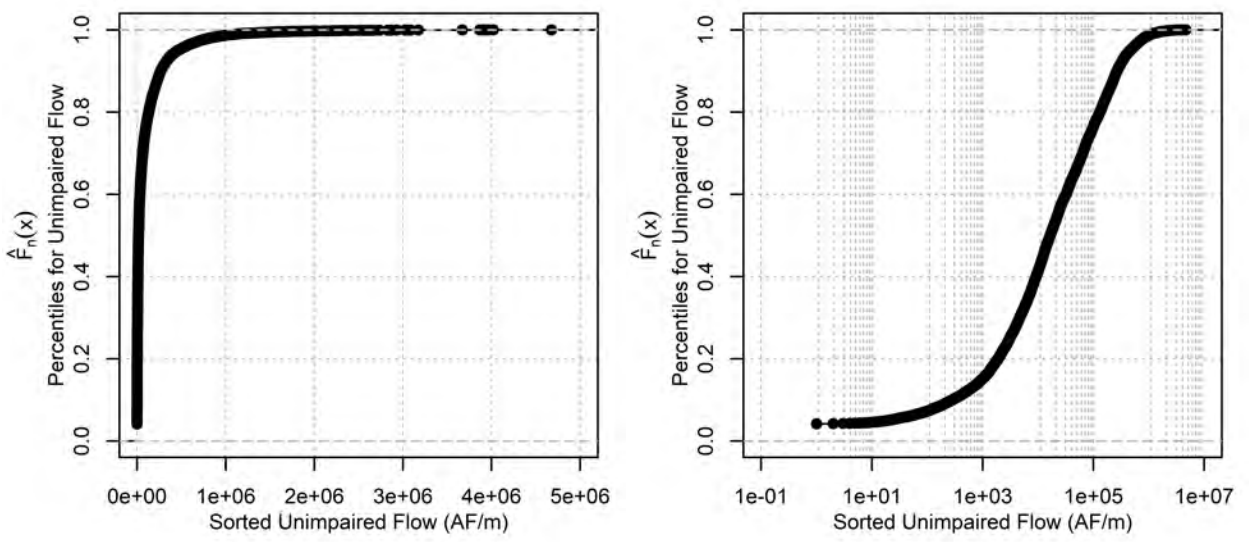


Figure A.2: Network schematic.



(a) The cumulative distribution function.

(b) The cumulative distribution function in log space.

Figure A.3: Distributions of the response variable. Approximately 19,000 unimpaired flows in acre-feet/month (AF/m).

Slopes Model (PRISM) dataset, which contains gridded rasters for the continental United States at $4km$ resolution from 1891 to 2014. The **temperature** variable and its lagged forms are the basin averaged PRISM *tmean* variable, which in turn was calculated by the mean of the monthly minimum temperatures and the monthly maximum temperatures. The **precipitation** variable and its lagged forms are the basin averaged PRISM *ppt* variable, which is a measure of total precipitation (rain and snow).

Low flows in some Sierra Nevada basins exhibit a "memory" effect in which they depend on the current and previous year's snowpack (Godsey, Kirchner, & Tague, 2014). Since we did not want to include 24 lagged precipitation parameters in the model, we developed a snow variable. The **snow** variable was the cumulative sum of precipitation, starting in October of each water year, for temperatures under 2°C.

Basin shape can affect the peak discharge; peak discharge for a circular basin arrives sooner than for an elongated basin of the same area. Because of how the tributary network in a circular basin is organized, the flows in a circular basin enter the main stem at roughly the same time, so more runoff is delivered to the outlet together, sooner. In an elongated basin, because of the mismatch in timing, peak runoff is more attenuated, except for some slow-moving streams. The **shape** parameter, calculated by basin length divided by basin width, and the **compactness** parameter, calculated by basin area divided by (basin perimeter)^2, account for this phenomenon. Although, this phenomenon is more pronounced in runoff on a smaller time step, we included these parameters in the final model to see their importance.

Basin hypsometric information was derived from the Shuttle Radar Topography Mission (SRTM) $90m$ model, which is a gridded raster of static elevation at a $3arc-second$ resolution. The vertical error of the model is reported to be less than $16m$. The **mean basin elevation** and **basin relief ratio** parameters (Pike & Wilson, 1971) were calculated from this dataset.

Basin relief ratio is calculated by the difference in maximum and minimum elevations divided by basin length.

Soil properties were derived from the POLARIS dataset, a Soil Survey Geographic Database (SSURGO) processed dataset at a $3arc - second$ resolution. Percent **clay**, **silt**, and **sand**, **saturated hydraulic conductivity**, **lambda** and **n** pore size, **available water content**, and **depth to restricted layer** information was averaged for each basin.

Table A.3: Summary of the variables used in the implementation of the model.

Type	Variable	Description	Source
Response	Unimpaired Flow	monthly estimated unimpaired flows, in AF	CDEC (Beaudette, 2016)
Time	Ordinal Month	numerical distance till October	
	Water Year	numeric year starting from the October of previous Gregorian year	
Climate	Temperature, Lag 1, 2 and 3 Months	temperature and lagged monthly temperature, in °C	PRISM (Hart & Bell, 2015)
	Precipitation, Lag 1, 2 and 3 Months	precipitation and lagged monthly precipitation, in mm	
	Snow	cumulative precipitation of the same water year for temperatures bellow 2 °C, in mm	
Hypsometric	Relief Ratio	(max(elev) - min(elev))/ basin length in, m/m	SRTM90 (Jarvis, Reuter, Nelson, Guevara, et al., 2008)
	Mean Elevation	mean basin elevation, in m	
Basin Boundaries	Area	basin drainage area, in $miles^2$	NHD2PLUS (McKay et al., 2012)
	Shape	basin length/basin width, in m/m	
	Compactness	basin area/(basin perimeter)^2, in m^2/m^2	
Soil	% Clay	percent clay in surface layer, in %	POLARIS (Chaney et al., 2016)
	% Silt	percent silt in surface layer, in %	
	% Sand	percent sand in surface layer, in %	
	Sat. Hydraulic Conductivity	hydrologic conductivity of surface layer, in cm/hr	
	Lambda	pore size distribution index (brooks-corey)	
	N	measure of the pore size distribution (van genuchten)	
	Available water content	available water content, in m^3/m^3	

Type	Variable	Description	Source
Land Cover	Vegetated	Percent of area in the basin vegetated in %	CALVEG (Forest Service, USDA, Pacific Southwest Region, 2006)
Ground Water	Depth to Restricted Layer	depth to aquitard, in *cm*	POLARIS (Chaney et al., 2016)

Other Descriptive Variables

Some variables are included in the dataset, but not in the modeling; these variables define the location of the gauges, and consist of the following: Longitude and Latitude (definite location), Hierarchy or the number of gauges that exist above (relative location in the network), river basin, county, and gauge operator. Hierarchies are different from the Strahler stream order; the gauges determine the hierarchy within the network whereas all branches can have a stream order number regardless of whether they are gauged or not. These descriptive variables are only used for plotting purposes (Table A.5 and Figures A.4, A.5, and A.6).

Table A.5: Descriptive statistics for unimpaired flow grouped by hierarchy.

Hierarchy	Total length of record	Standard deviation (σ)	Mean (μ)	Coefficient of Variation ($\frac{\sigma}{\mu}$)
1	12,225	162,341	47,491	3.41
2	3,548	131,218	91,677	1.43
3	1,175	320,759	242,211	1.32
4	1,179	518,580	424,348	1.22
5	393	225,426	186,534	1.21

Correlations

A simple examination of the partial correlations of predictor variables with flow shows that most of the information content lies within drainage area, precipitation, and some measures of infiltration (i.e., lambda pore size, n pore size, and saturated hydraulic conductivity). The correlated variables were not removed from the model (Figure A.7). For a more complete correlation plot see Figure A.8.

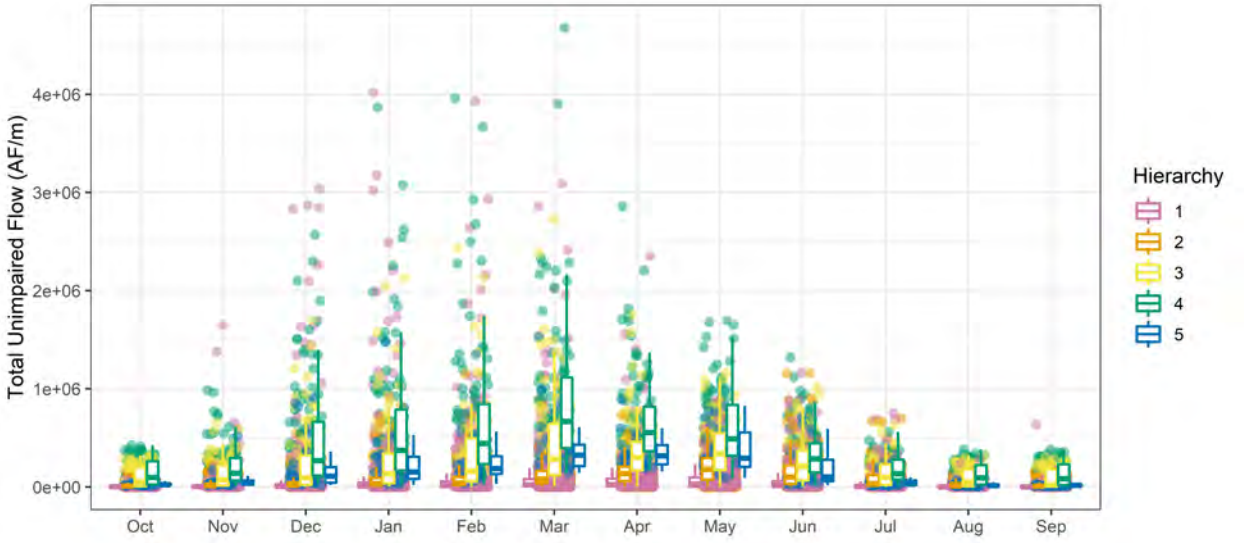


Figure A.4: The cyclical behavior of total monthly unimpaired flows. The flows start to rise in October, the start of the "water year." The boxplots also show that given a higher hierarchy (i.e., being lower in the network of gauges) the monthly distribution of flows becomes larger. The only exception to this is basin hierarchy number 5, and that is because this data set only had one basin in that hierarchy. Had there been more basins, its distribution would be wider showing that the lower you are in the network, the higher the flows and the bigger the distribution of flows.

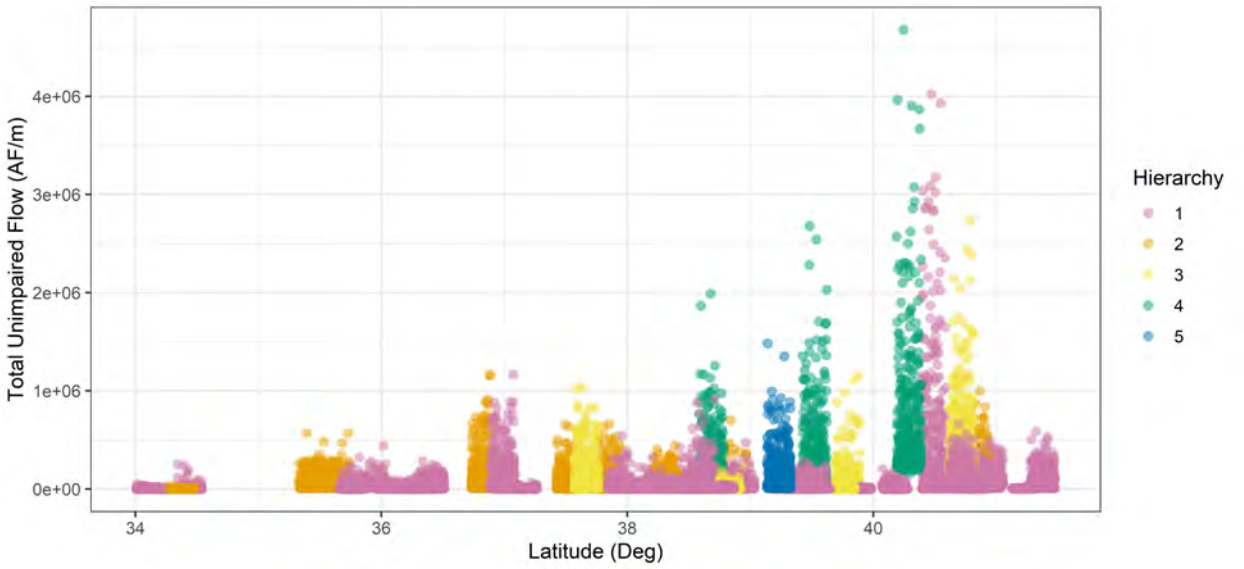


Figure A.5: Total monthly unimpaired flow vs. latitude. Total monthly unimpaired flow increases at higher latitudes in California. Note that each of the basins were at a unique latitude, for illustration purposes the latitude variable was jittered.

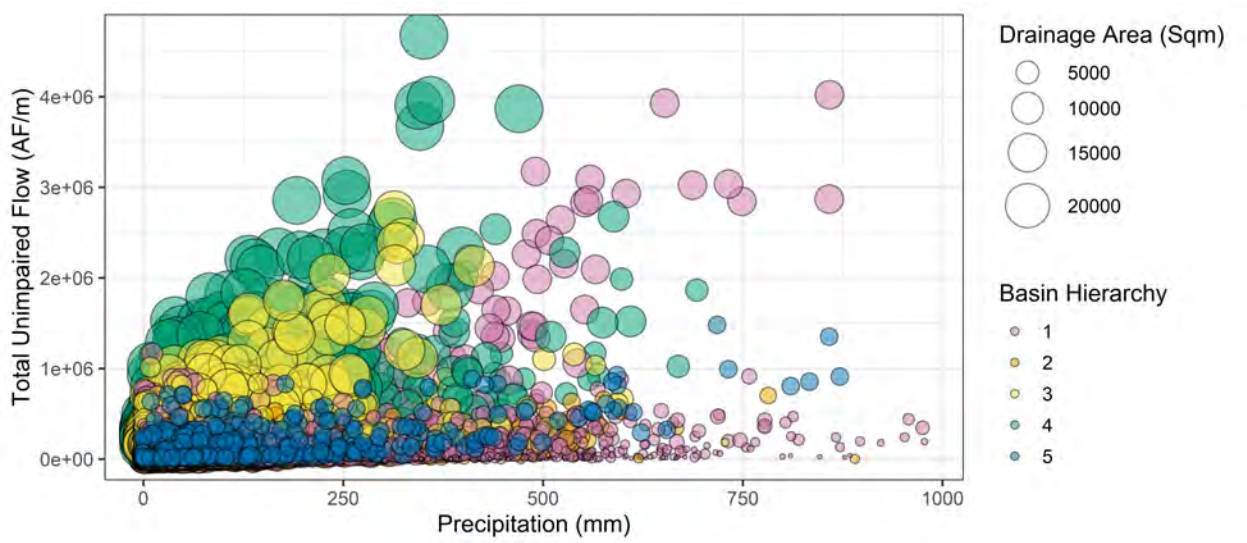
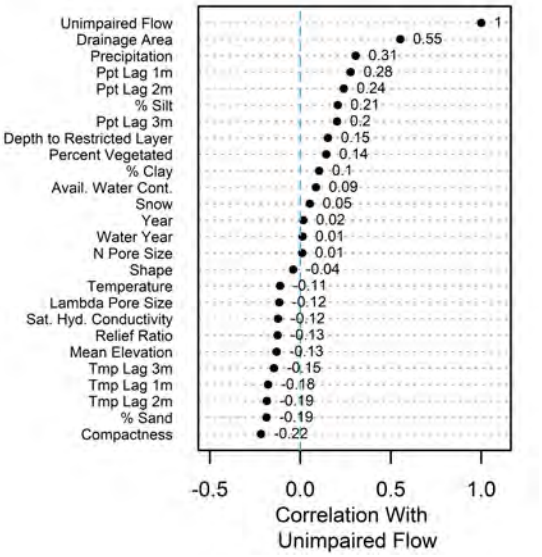
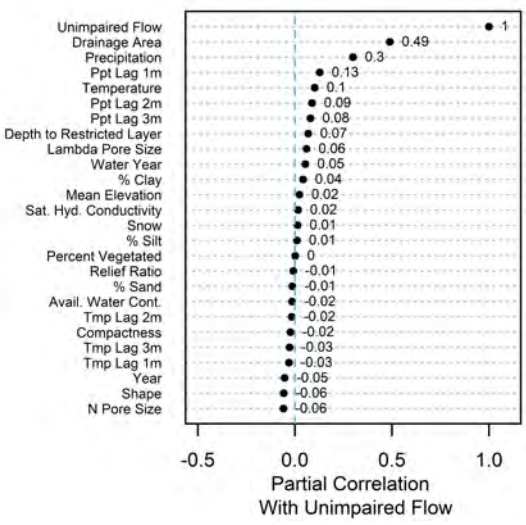


Figure A.6: Total monthly unimpaired flow vs. precipitation. The total monthly unimpaired flow increases with increasing precipitation. This is also drainage area dependent, as the smaller drainage areas that happen to have high amounts of precipitation still produce low flows. Basin hierarchies also show that the larger basins are lower in the network. The only exception being hierarchy number 5, and that is because this data set only had one basin in that hierarchy.



(a) Pearson's Correlation



(b) Partial Correlations

Figure A.7: Correlation of predictor variables with monthly flow volumes. Drainage area and precipitation correlate the most with flow.

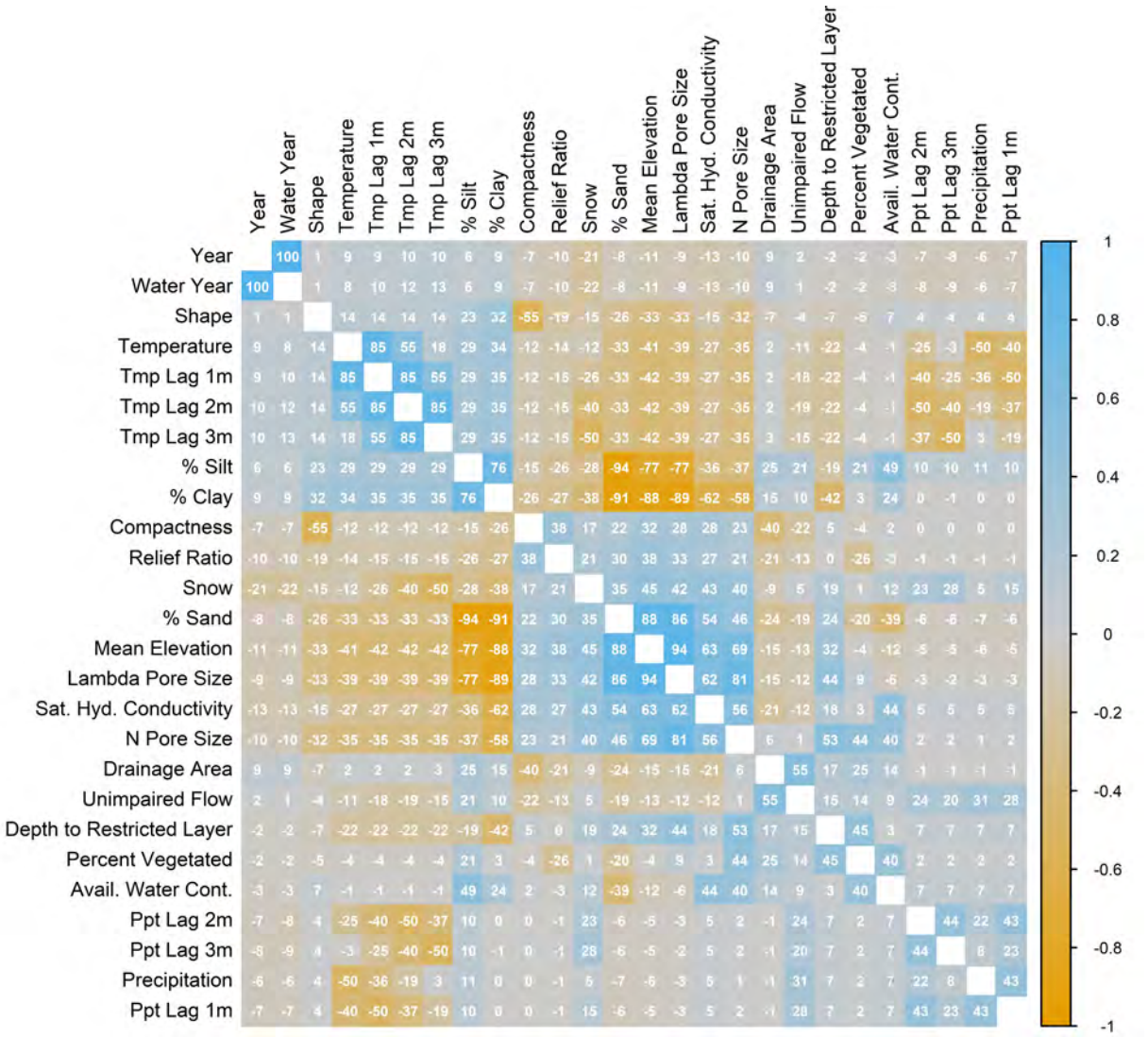


Figure A.8: Correlation plot. Patterns can arise in correlations especially when some variables are calculated from or are directly related to others. For example, the percentage of sand silt and clay in a basin adds to one. Therefore, these variables are negatively correlated. Also, lag variables calculated from precipitation and temperature will tend to correlate with one another. However, the snow variable that was calculated from precipitation does not significantly correlate with precipitation.

Appendix B

Terms & Concepts in Machine Learning

This appendix introduces common terms and concepts used in statistical learning and in this paper.

Terminology

Variables: Predictors, independent variables (sometimes just variables), or features all are the inputs into a model that we believe in some way will inform us about another variable we are interested in. The response, output, or dependent variable, is the output of the model we are interested in.

Training and Test sets: Data sets used for training the model and testing the model's predictive capabilities, respectively.

Bias Variance Trade-off: Bias and variance make up part of the expected test set squared error (See Equation B.1).

$$\begin{aligned} &E[(y-\hat{f})^2] = Var[\hat{f}] + (Bias[\hat{f}])^2 + Var(\epsilon) \\ &Var[\hat{f}] = E[\hat{f}^2] - (E[\hat{f}])^2 \\ &Bias[\hat{f}] = E[\hat{f} - E[y]] \\ &Var[\epsilon] = \sigma^2 \end{aligned} \tag{B.1}$$

where y is the observed response variable, x is the observed predictor variable and $y = f(x) + \epsilon$, $\hat{f}(x)$ is the modeled or predicted response variable, and ϵ is the irreducible error in the response variable.

That is, variance and bias make up the reducible error in the response variable. It is reducible because we can modify it by changing the training data (e.g., adding more data), which effects the variance component, or changing the model type (e.g., going from linear to nonlinear), which effects the bias component of the bias variance trade-off.

Resampling: These methods create synthetic or "extra" data from the original data set. This data set, different from the whole sample, is sometimes needed for nuisance parameter estimation (usually with cross-validation) or model error estimation (usually achieved with the bootstrap). We discuss the importance of resampling methods in Chapter 4.

Loss or Objective Function: The expectation of the loss function, $L(y_i, \hat{y}_i)$ is the function that is minimized (or maximized) in a statistical learning algorithm representing

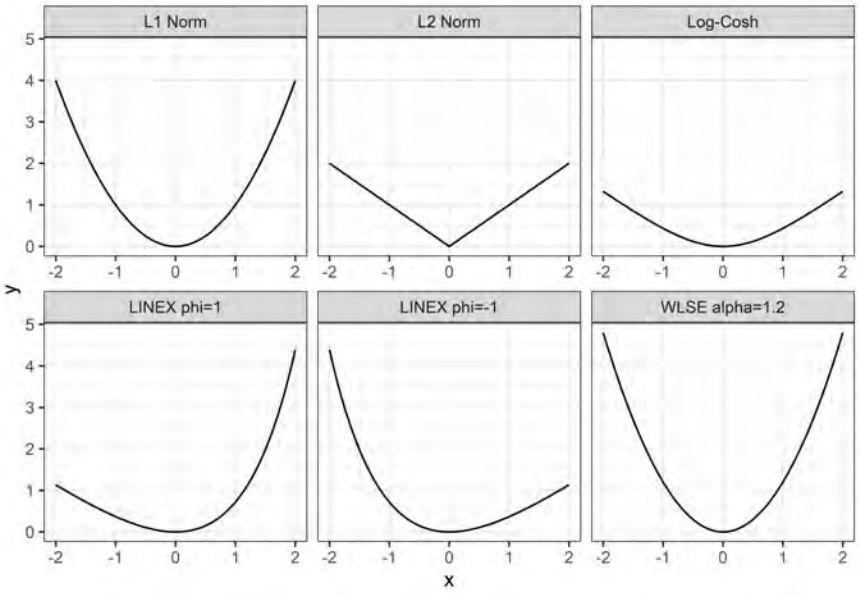


Figure B.1: Typical convex loss functions in statistical learning.

linear or non-linear penalties for mis-estimation. Figure B.1 depicts typical loss functions used in machine learning. A loss function is a statement of priorities; what we want the model to get right and how much we care about the error relatively. For example, what is the cost of getting low flows predicted incorrectly (additional drought damage cost)? What is the cost for predicting high flows incorrectly (additional flood damage cost)? To some extent the choice of a loss function is subjective. We examine loss functions in chapter 3.

Convex Optimization Problems: Optimization problems that are convex if the objective function and constraints have optimal edge solutions; if a solution is found to the minimization or maximization, it is guaranteed to be a global solution.

Gradient-Based Optimization Methods: These methods find local minima or maxima of an objective function by searching along the gradient of the objective function. For example, in a minimization problem using the steepest gradient search methods, the decent direction and step size is found in each iteration. Gradient-based methods require a differentiable loss function. However, variations such as subgradient methods allow for minimization of convex problems with kinks in the loss function.

Derivative-Free Optimization Methods: These methods do not require gradient calculations and are well suited to problems where a loss function is not non-convex or irregular. For example, evolutionary algorithms find local minima or maxima by evaluating the loss function on a population of solutions, and allowing them to evolve in each iteration.

Learning Scenarios

Supervised vs. Unsupervised: In supervised settings, we have a variable of interest, y, that we believe follows a functional form: $y = f(x) + \epsilon$, where $f(x)$ provides systematic information about y, and ϵ is the error term. In modeling we try to approximate this functional form (i.e., $\hat{f}(x)$) with the observations (i.e., $\hat{y}$). We also can try to estimate y from the data itself, without assuming a functional form (See next section on Parametric vs.

Non-Parametric).

In unsupervised learning, we do not have a variable of interest, y, to model. Instead, we have observations of many variables that we can still study for their natural groupings, patterns, or relationships between variables. For example, when classifying streams, the model can learn from various stream and basin characteristics without a particular variable being of interest.

Prediction vs. Inference: The two major goals of statistical analysis are prediction or inference. In prediction, we are interested in getting the simulated value to closely resemble observed values (e.g., can we accurately predict the value of a house). That is, we are concerned with accuracy.

However, in inference, we are interested in the relationship of predictor variables to the response variable (e.g., how much extra will a house be worth with a scenic view). That is, we are concerned with model interpretability, which implies that a simpler (fewer variables) model is preferred even at some cost to prediction accuracy (James et al., 2013).

Parametric vs. Non-Parametric: Parametric models assume a functional form. For example, from Ohm's law ($V = IR$), we assume that given an unknown resistor, voltage and current have a linear relationship ($y = \beta_1 x + \epsilon$), where y is the voltmeter readings and x is the ammeter readings. By assuming this functional, form errors in observations can be due to the measurement device (the voltmeter or ammeter) or human error. Now, we can estimate the parameters of the model from the observations. In this case, we estimate resistance, R, by fitting $\hat{y} = \hat{\beta}_1 x$. We have reduced the problem of finding $\hat{f}(x)$ to finding $\hat{\beta}_1$.

However, in non-parametric models, we do not assume a functional form and try to get the model as close to the observations as possible without being too "rough." For example, Kriging interpolators are known as exact interpolators where the predictions at every observation are the observations precisely. This approach highly depends on the observations, and so, it suffers from high variance in the bias-variance trade-off; is the sample data were to change, even slightly, the model shifts (high variance) because it needs to pass through these observations. Smoothing techniques, such as thin plate splines, relax this constraint, and depending on the degrees of freedom or flexibility we allow, the prediction can be close to or far from the observations. This approach is data intensive and usually performs better for prediction than for inference, because, after all, it is trading the parameters that aid inference for model flexibility.

Regression vs. Classification: Variables can be classified as quantitative or qualitative. Quantitative variables have numerical values and a quantitative response variable is used in what we refer to as regression models. In contrast, qualitative variables have class values, categories or ordinal levels and a categorical response variable is used in classification models. The predictors may take either form and are generally less important (James et al., 2013).

Appendix C

Brief History of Statistical Learning

This appendix explains how some of the ideas organized in chapter 1's heuristic guide developed over time.

In 1763, Thomas Bayes' *An Essay towards solving a Problem in the Doctrine of Chances* was published posthumously. In it, Bayes explained that "given the number of times in which an unknown event has happened and failed, the chance that the probability of its happening in a single trial lies somewhere between any two degrees of probability that can be named" (Bayes & Price, 1763). This work later underpins **Bayes' Theorem**.

In 1805, Adrien Marie Legendre introduced the least squares method of estimating parameters as an appendix to his book on the paths of comets. Carl Freidrich Gauss also published the method a few years later but claimed he had been using it since 1795 (Stigler, 1981). Regardless of the original inventor, the method is refined with its application in **linear regression** and curve fitting.

In 1812, Pierre-Simon Laplace, expanding on the work of Bayes, introduced methods of finding probabilities of compound events when the probabilities of their simple components are known, and he defined what is now known as **Bayes' Theorem** (O'Connor & Robertson, 2000).

In 1913, Andrey Markov founded a new branch of probability theory by applying mathematics to poetry. Later called **Markov chains**, the method went beyond coin-flipping (where each event is independent of all others) to chains of linked events (where what happens next depends on the current state of the system) (Hayes et al., 2013).

In 1936, Ronald Fisher introduced a method to find a linear combination of features that separates (or discriminates between) two or more classes of events. Fisher's discriminant is later slightly modified to add the assumptions of normally distributed classes or equal class covariances, and became the more famous **linear discriminant analysis (LDA)** (Härdle & Simar, 2007).

In the 1958, David Cox developed **logistic regression** for situations where it is not reasonable to assume that the independent variables are normally distributed as in LDA (Cox, 1958).

In 1951, Marvin Minsky and graduate student Dean Edmonds built the first **neural network machine**. This machine was a randomly connected network of capacitors that have a finite amount of memory and time to keep or remember that memory. The memory holds the probability that a signal will come in one input and another signal will come out of

the output. This machine, modeled after the Hebbian theory of learning in the human brain, was one of the first pioneering attempts at artificial intelligence (Crevier, 1993). Shortly after, in 1957, Frank Rosenblatt invents the perceptron, the first **neural network** for computers (Rosenblatt, 1957).

In 1967, the Thomas Cover and Peter Hart invent the **nearest neighbor** algorithm, which kickstarted basic pattern recognition (Cover & Hart, 1967). The algorithm was used to map a route for the *traveling salesmen problem*, starting at a random city, but ensuring a visit to all cities during the shortest tour (Marr, 2016).

In 1972, Nelder and Wedderburn introduced **generalized linear models**. Linear models are customarily made of systematic and random error components, with the errors usually assumed to have normal distribution. This work allowed for a unified fitting procedure, despite the type of error distribution, based on likelihood (Nelder & Wedderburn, 1972).

In 1980, Kunihiko Fukushima developed the neocognitron, a type of **artificial neural network** (Fukushima & Miyake, 1982). This work later inspired the development of **convolutional neural networks**.

In 1981, Gerald Dejong introduced **explanation-based learning**, where a computer algorithm analyzes data, creates a general rule it can follow, and discards unimportant data (Marr, 2016). The new knowledge structure is not constructed by noticing the similarities and differences among a large number of inputs, nor is it constructed from a more general one already existing within the system. The system can learn from just one example and adapt its learning. The knowledge structure can be expanded later but is already a viable new schema capable of adding future processing (DeJong, 1981).

In 1982, John Hopfield developed Hopfield networks, a type of **recurrent neural network** that can serve as content-addressable memory systems (Hopfield, 1982). Based on aspects of neurobiology, the content-addressable memory can yield an entire memory from any subpart of sufficient size. The recurrent aspect of RNNs make it a breakthrough for processes driven by lagged parameters. For example, in hydrology, runoff processes are affected by time-lagged precipitation; depending on the size of the watershed, precipitation at the headwaters may take days to reach the outlet, or, snowfall in the winter will take months to melt and turn into baseflow. In 1997, Sepp Hochreiter and Jorgen Schmidhuber invent **long short-term memory (LSTM) recurrent neural networks**. This method greatly improved the efficiency of neural networks (i.e., more successful runs, at a higher learning rate) and it solved complex (i.e., artificial long-time-lag) tasks that have never been solved by previous recurrent network algorithms (Hochreiter & Schmidhuber, 1997).

In 1984, Brieman, Friedman, Olshen, and Stone introduced **classification and regression trees (CART)** (Breiman, Friedman, Olshen, & Stone, 1993), a method of recursively partitioning the feature space. In 1995, Tin Kam Ho fixes the issue of high variance in the CART with his proposed **random forest** algorithm (Ho, 1995).

In 1986, Hastie and Tibshirani developed the **generalized additive model**, a non-parametric extension to the generalized linear models where the linear predictor is replaced by an additive predictor (Hastie & Tibshirani, 1990). This means the model is fit on multiple predictors and the fit on each predictor is updated by holding the others fixed (i.e., fit to a partial residual).

In 1995, Corinna Cortes and Vladimir Vapnik published their work on **support vector machines**. Originally applied to only two-group classification problems, this procedure

constructs a linear decision surface in high dimensions with corresponding "support vectors" at a margin, M, from the decision surface. The purpose of the method is to maximize the margin, M (Cortes & Vapnik, 1995).

Until the 1990's, statistical learning was a purely theoretical analysis of the problem of function estimation from a given collection of data (Vapnik, 1999). Since then, with the commercialization of software programs, these methods can be applied to "real-world" data and therefore used in fields outside of statistics and computer science. Work on these methods has also shifted from knowledge-driven approaches to a data-driven approaches; we are letting the computer analyze large amounts of data and "learn" from the results. As Winston (2010) puts it, "the computer is learning much like a bulldozer processing gravel."

In 2006, Geoffrey Hinton developed *deep learning* techniques that let computers "see" and distinguish text in images (using the famous MNIST database of hand-written digits). These methods make inference easier in densely connected belief nets that have many hidden layers and scale poorly to increases in the number of parameters (Hinton, Osindero, & Teh, 2006). **Deep convolutional networks** have brought about breakthroughs in processing images, video, speech, and audio (Marr, 2016).

In 2010, the Microsoft **Kinect** was launched. The devise could track 20 human features at a rate of 30 times per second (Marr, 2016), allowing people to interact with the computer (or more pointedly, the console) via movements and gestures. Microsoft's vision was to incorporate motion into gaming, eliminating the need for controllers you would have to charge or could accidentally fling into your TV (Cranz, 2018).

In 2012, **Google Brain** started. Led by Andrew Ng and Jeff Dean, its deep neural network can learn to discover and categorize objects. Despite the fact that the network had never been told what a cat was, nor was it given even a single image labeled as a cat, it "discovered" what a cat looked like from unlabeled YouTube images (Dean & Ng, 2015).

In 2014, Facebook developed **DeepFace**, a software algorithm that is able to recognize that two images show the same face (i.e., facial verification). It employs a nine-layer neural net with over 120 million connection weights and was trained on four million images uploaded by Facebook users (Simonite, 2014). This algorithm raised some privacy concerns and their recent Cambridge Analytica scandal did not help Facebook with the heightened scrutiny either.

In 2014, Google researchers presented their work on **Sibyl**. This proprietary platform started off by recommending YouTube videos to users. Now it can predict spam and a user's ad preferences. In general, its goal is to predict how Google users will behave in the future, based on what they did in the past (Woodie, 2014).

In 2015, Amazon launched its own machine learning platform, **SageMaker**. This platform was designed to help developers and data scientists from the data acquisition step to full model deployment (Amazon Web Services, 2018).

In 2015, Microsoft created the **Distributed Machine Learning Toolkit**, which makes machine learning tasks on big data highly scalable, efficient, and flexible. The toolkit employs a special sampling technique to create and distribute training data throughout the cluster (Rolle, 2015).

In 2015, over 3,000 AI and Robotics researchers, endorsed by Stephen Hawking, Elon Musk, and Steve Wozniak (among many others), signed an open letter calling for a ban on offensive autonomous weapons beyond meaningful human control. The letter warns us that

"Artificial Intelligence technology has reached a point where the deployment of such systems is–practically if not legally–feasible within years (Hawking, Musk, Wozniak, et al., 2015)."

In August of 2018, artificial intelligence bots beat five human players at the video game Dota 2. OpenAI, an independent research institute cofounded by Elon Musk developed the bots and used reinforcement learning to train for the match. In contrast to chess or go, it is especially difficult to train machines to play videogames, because the action takes place on a much larger board, where not all opponent's moves are visible, and it requires players to make decisions quickly.

Appendix D

Model Measures of Fit

This appendix introduces typical model measures-of-fit developed in hydrologic modeling listed in Table D.1.

Table D.1: Summary of the variables used in the implementation of loss functions.

MOF	Name	Type	Ideal Value	Range
MAE	Mean Absolute Error	absolute measure	0	$[0, \infty)$
MSE	Mean Squared Error	absolute measure	0	$[0, \infty)$
RMSE	Root Mean Squared Error	absolute measure	0	$[0, \infty)$
nRMSE	Normalized RMSE	absolute measure	0	$[0, \infty)$
RSR	RMSE standard deviation ratio	absolute measure	0	$[0, \infty)$
RSD	Relative Standard Deviation	supporting measure	1	$(-\infty, \infty)$
RMU	Relative Mean	supporting measure	1	$(-\infty, \infty)$
PBIAS	Percent Bias	supporting measure	0	$(-100\%, 100\%)$
R^2	Coefficient of Determination	measure of linearity in simulated vs. predicted	1	$[0, 1]$
bR^2	Weighted R^2	bias corrected R^2	1	$[0, 1]$
NSE	Nash-Sutcliffe Efficiency	square difference measure of fit	1	$(-\infty, 1]$
d	Index of Agreement	square difference measure of fit	1	$[0, 1]$
mNSE	Modified NSE	sensitivity to peaks can be modified	1	$(-\infty, 1]$
md	Modified d	sensitivity to peaks can be modified	1	$[0, 1]$
rNSE	Relative NSE	sensitivity to peaks eliminated	1	$(-\infty, 1]$
rd	Relative d	sensitivity to peaks eliminated	1	$[0, 1]$
KGE	Kling-Gupta Efficiency	relative importance of error component made explicit	1	$(-\infty, 1]$
VE	Volumetric Efficiency	volumes made important no matter if it is in a peak or recession	1	$(-\infty, 1]$

See Equations D.1 to D.8 where Y_i^{obs} are the observed unimpaired flows, and Y_i^{sim} are the predicted or simulated unimpaired flows, and n is the number of observations.

$$MAE = \frac{\sum_{i=1}^{n}\left|Y_i^{sim} - Y_i^{obs}\right|}{n} \tag{D.1}$$

$$MSE = \frac{\sum_{i=1}^{n}\left(Y_i^{sim} - Y_i^{obs}\right)^2}{n} \tag{D.2}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\left(Y_i^{sim} - Y_i^{obs}\right)^2}{n}} \tag{D.3}$$

$$nRMSE = \frac{RMSE}{MU_{obs}} = \frac{\sqrt{\sum_{i=1}^{n}\left(Y_i^{sim} - Y_i^{obs}\right)^2}}{\overline{Y^{obs}}} \tag{D.4}$$

$$RSR = \frac{RMSE}{\sigma_{obs}} = \frac{\sqrt{\sum_{i=1}^{n}\left(Y_i^{sim} - Y_i^{obs}\right)^2}}{\sqrt{\sum_{i=1}^{n}\left(Y_i^{obs} - \overline{Y^{obs}}\right)^2}} \tag{D.5}$$

The MAE, MSE, RMSE, nRMSE, RSR, are absolute measures of error.

$$RSD = \frac{\sigma_{sim}}{\sigma_{obs}} = \frac{\sqrt{\sum_{i=1}^{n}\left(Y_i^{sim} - \overline{Y^{sim}}\right)^2}}{\sqrt{\sum_{i=1}^{n}\left(Y_i^{obs} - \overline{Y^{obs}}\right)^2}} \tag{D.6}$$

$$RMU = \frac{\overline{Y^{sim}}}{\overline{Y_i^{obs}}} = \frac{\sum_{i=1}^{n} Y^{sim}}{\sum_{i=1}^{n} Y^{obs}} \tag{D.7}$$

$$PBIAS = \frac{\sum_{i=1}^{n}\left(Y_i^{obs} - Y_i^{sim}\right) * 100}{n\sum_{i=1}^{n}\left(Y_i^{obs}\right)} \tag{D.8}$$

The RSD, RMU, and PBIAS are additional supporting measures of error.

$$R^2 = \left(\frac{\sum_{i=1}^{n}\left(Y_i^{obs} - \overline{Y^{obs}}\right)\left(Y_i^{sim} - \overline{Y^{sim}}\right)}{\sqrt{\sum_{i=1}^{n}\left(Y_i^{obs} - \overline{Y^{obs}}\right)^2}\sqrt{\sum_{i=1}^{n}\left(Y_i^{sim} - \overline{Y^{sim}}\right)^2}}\right)^2 \tag{D.9}$$

R^2 is insensitive to additive and proportional difference between model simulation and observations. One can simply show that for a non-zero value of β_0 and β_1, if the predictions follow a linear form, $Y^{sim} = \beta_0 + \beta_1 Y^{obs}$, the R^2 equals one (Legates & McCabe Jr, 1999). Therefore, for a proper model assessment, it is recommended that the slope of the predicted vs. observed graph be reported or systematically included as in Equation D.10.

$$bR^2 = \begin{cases} |b|\, R^2 & \text{for b} \leq 1 \\ |b|^{-1}\, R^2 & \text{for b} > 1 \end{cases} \tag{D.10}$$

By weighting R^2 under or over predictions are quantified together with the dynamics which results in a more comprehensive reflection of model results.

$$NSE = \; 1 - \frac{\sum_{i=1}^{n} \left(Y_i^{sim} - Y_i^{obs}\right)^2}{\sum_{i=1}^{n} \left(Y_i^{obs} - \overline{Y^{obs}}\right)^2} \tag{D.11}$$

A Nash-Sutcliffe Efficiency factor of lower than zero indicates that the mean value of the observed time series would have been a better predictor than the model. The largest disadvantage of the Nash-Sutcliffe Efficiency factor is the fact that the differences between the observed and predicted values are calculated as squared values. As a result, larger values in a time series are strongly overestimated whereas lower values are neglected (Legates & McCabe Jr, 1999). For the quantification of runoff predictions this leads to an overestimation of the model performance during peak flows and an underestimation during low flow conditions (Krause et al., 2005).

To reduce the problem of the squared differences and the resulting sensitivity to extreme values the Nash-Sutcliffe Efficiency factor is often calculated with logarithmic values of Y_i^{sim} and Y_i^{obs}. Through the logarithmic transformation of the runoff values the peaks are flattened and the low flows are kept more or less at the same level. As a result the influence of the low flow values is increased in comparison to the flood peaks resulting in an increase in sensitivity of $ln(NSE)$ to systematic model over or under prediction (Krause et al., 2005).

$$d = \; 1 - \frac{\sum_{i=1}^{n} \left(Y_i^{sim} - Y_i^{obs}\right)^2}{\sum_{i=1}^{n} \left(\left|Y_i^{sim} - \overline{Y^{obs}}\right| + \left|Y_i^{obs} - \overline{Y^{obs}}\right|\right)^2} \tag{D.12}$$

$$mNSE = \; 1 - \frac{\sum_{i=1}^{n} \left|Y_i^{sim} - Y_i^{obs}\right|^j}{\sum_{i=1}^{n} \left|Y_i^{obs} - \overline{Y^{obs}}\right|^j}, \quad j \in \mathbb{N} \tag{D.13}$$

$$md = \; 1 - \frac{\sum_{i=1}^{n} \left|Y_i^{sim} - Y_i^{obs}\right|^j}{\sum_{i=1}^{n} \left(\left|Y_i^{sim} - \overline{Y^{obs}}\right| + \left|Y_i^{obs} - \overline{Y^{obs}}\right|\right)^j}, \quad j \in \mathbb{N} \tag{D.14}$$

For j=1, the overestimation of the flood peaks in regular NSE is reduced significantly resulting in a better overall evaluation. j=3 is best for flood modeling.

$$rNSE = 1 - \frac{\sum_{i=1}^{n}\left(\frac{Y_i^{sim} - Y_i^{obs}}{Y_i^{obs}}\right)^2}{\sum_{i=1}^{n}\left(\frac{Y_i^{obs} - \overline{Y^{obs}}}{\overline{Y^{obs}}}\right)^2} \tag{D.15}$$

$$rd = 1 - \frac{\sum_{i=1}^{n}\left(\frac{Y_i^{sim} - Y_i^{obs}}{Y_i^{obs}}\right)^2}{\sum_{i=1}^{n}\left(\frac{\left|Y_i^{sim} - \overline{Y^{obs}}\right| + \left|Y_i^{obs} - \overline{Y^{obs}}\right|}{\overline{Y_i^{obs}}}\right)^2} \tag{D.16}$$

As a result, an over or under prediction of higher values (i.e., peaks) has, in general, a greater influence than those of lower values. Therefore, we can use relative values in the regular NSE equations. These equations will not be sensitive to peaks at all.

$$\begin{aligned} &KGE = 1 - \sqrt{(r-1)^2 + (\beta-1)^2 + (\gamma-1)^2}, \\ &r = Pearson's\ r, \\ &\beta = \frac{\overline{Y^{sim}}}{\overline{Y^{obs}}}, \\ &\gamma = \frac{C_v^{sim}}{C_v^{obs}} = \frac{\frac{\sigma_{sim}}{\overline{Y^{sim}}}}{\frac{\sigma_{obs}}{\overline{Y^{obs}}}} \end{aligned} \tag{D.17}$$

The Kling Gupta Efficiency (KGE) factor facilitates the analysis of the relative importance of its different components: r, correlation and timing; β: magnitude and bias; and γ: variability).

$$VE = 1 - \frac{\sum_{i=1}^{n}\left|Y_i^{sim} - Y_i^{obs}\right|}{\sum_{i=1}^{n}\left(Y_i^{obs}\right)} \tag{D.18}$$

To solve the problems presented with reporting bias in hydrologic models, the Volumetric Efficiency (VE) can be used. It is easy to calculate, and of treats every unit volume of water the same as any other unit volume, whether it be delivered during slow recession or during peak flow (Criss & Winston, 2008).

In conclusion, the optimal benchmark will differ for different applications, which is why so many benchmarks have been proposed in hydrology. It is especially critical when the

model measure of fit it to be used as a loss function in a machine learning algorithm. These discretionary choices tend to disappear when complex modeling is concerned. Therefore, the criteria for decisions should be made explicit and known before modeling begins.

Moriasi et al. (2007) provides a table of general performance ratings for recommended statistics for a monthly time step, useful for the modeling done in this dissertation D.3.

Table D.3: General performance ratings for recommended statistics for a monthly time step. Reprinted from Moriasi et al., 2007.

Performance Rating	RSR	NSE	PBIAS
Very good	$[0.0, 0.5]$	$(0.75, 1.00]$	$(-\infty, \pm 10)$
Good	$(0.5, 0.6]$	$(0.65, 0.75]$	$[\pm 10, \pm 15)$
Satisfactory	$(0.6, 0.7]$	$(0.50, 0.65]$	$[\pm 15, \pm 25)$
Unsatisfactory	$(0.7, \infty)$	$(-\infty, 0.50]$	$[\pm 25, \infty)$

Appendix E

Code for Implementing Custom Loss Functions

This appendix shows R code used for implementing custom loss functions in Chapter 3. MSPE loss is defined as follows:

```
mspe <- function(y_true, y_pred){
  K <- backend()
  # added a 1 to y_true in the denominator, because dividing by 0 is a
      problem. You can also use a relu function, or use a clip where the 0
      values get truncated. However, here, in the cases where y_true=0, the
      mspe function turns into a simple mse.
  mod_loss <- K$mean(K$pow((K$flatten(y_pred)-K$flatten(y_true))/K$flatten(
      y_true+1), 2))
  return(mod_loss)
}
```

WLSE loss is defined as follows:

```
wlse <- function(y_true, y_pred, flood_vect, alphad, betad, alphaf, betaf){
  alpha_vect <- ifelse(flood_vect==0, alphad, alphaf)
  beta_vect <- ifelse(flood_vect==0, betad, betaf)
  K <- backend()
  alpha_vect_cte <- K$constant(alpha_vect, dtype='float32')
  beta_vect_cte <- K$constant(beta_vect, dtype='float32')
  alpha_loss <- K$transpose(alpha_vect_cte)*K$cast(K$pow(K$minimum(0, K$
      flatten(y_pred)-K$flatten(y_true)), 2), dtype='float32')
  beta_loss <- K$transpose(beta_vect_cte)*K$cast(K$pow(K$maximum(0, K$flatten
      (y_pred)-K$flatten(y_true)), 2), dtype='float32')
  mod_loss <- K$sum(alpha_loss + beta_loss)*10^-6
  return(mod_loss)
}
```

WLSE wrapper is defined as follows:

```
wlse_wrapper_stochastic <- custom_metric("wlse", function(y_true, y_pred) {
    wlse(y_true, y_pred, flood_vect=trainsetpvs[, "FLOOD"], alphad=0.00001,
```

```
    betad=0.00005, alphaf=0.00005, betaf=0.00001)})
```

LINEXE loss is defined as follows:

```
linexe <- function(y_true, y_pred, flood_vect, phid, phif){
  phi_vect <- ifelse(flood_vect==0, phid, phif)
  K <- backend()
  phi_vect_cte <- K$constant(phi_vect, dtype='float32')
  exp_loss <- K$exp(K$transpose(phi_vect_cte)*K$cast(K$flatten(10^-6*(y_true-
      y_pred)), dtype='float32'))
  lin_loss <- K$transpose(phi_vect_cte)*K$cast(K$flatten(10^-6*(y_true-y_pred
      )), dtype='float32') + 1
  mod_loss <- 10^6*(K$mean(exp_loss-lin_loss))
  return(mod_loss)
}
```

LINEXE wrapper is defined as follows:

```
linexe_wrapper_stochastic <- custom_metric("linexe", function(y_true, y_pred)
    {linexe(y_true, y_pred, flood_vect=trainsetpvs[, "FLOOD"], phid=1.0,
    phif=-1.5})
```

The NN model is specified as follows:

```
nnmodel <- keras_model_sequential() %>% # can use: "relu", "sigmoid", "softmax"
     layer_dense(units=64, activation="relu", input_shape=dim(trainsetpvs)
         [[2]]) %>%
     layer_dense(units=64, activation="relu") %>%
     layer_dense(units=1) # number of outputs, here we just want one
         prediction
```

To compile the model and define loss functions we have:

```
nnmodel %>%
  compile(optimizer="rmsprop", loss=[chosen from loss functions defined
      above], metrics=c("mae"))
```

Fitting is done with the following lines of code for symmetric losses:

```
nnmodel %>%
  fit(trainsetpvs, trainsetrv, epochs=100, batch_size=25, verbose=1,
      validation_split=0.2)
```

Fitting is done with the following lines of code for asymmetric losses:

```
nnmodel %>%
  compile(optimizer="rmsprop", loss=[chosen from wrappers defined above],
      metrics=c("mae"))
nnmodel %>%
```

```
    fit(trainsetpvs, trainsetrv, epochs=1000, batch_size=nrow(trainsetpvs),
        shuffle=FALSE, verbose=1)
```

Lastly, predictions come from the following lines of code:

```
predictions <- nnmodel %>% predict(testsetpvs)
# since the output layer was specified to be unit=1, no need to average the
    responses
predictions <- predictions[ , 1]
```

References

Abrahart, R. J., Heppenstall, A. J., & See, L. M. (2007). Timing error correction procedure applied to neural network rainfall–runoff modelling. *Hydrological sciences journal*, *52*(3), 414–431.

Allaire, J., & Chollet, F. (2019). keras: R interface to 'keras' [Computer software manual]. Retrieved from https://keras.rstudio.com (R package version 2.2.4.1.9001)

Amazon Web Services. (2018). Amazon sagemaker: Build, train, and deploy machine learning models at scale. *Amazon*. Retrieved from https://aws.amazon.com/sagemaker/features/

Antognini, O. J. (2016). *Why are neural networks becoming deeper, but not wider?* Cross Validated. Retrieved from https://stats.stackexchange.com/q/223637 (URL:https://stats.stackexchange.com/q/223637 (version: 2019-02-06))

Aphalo, P. J. (2016). *Learn r ...as you learnt your mother tongue*. Leanpub. Retrieved from https://leanpub.com/learnr

Asefa, T., Kemblowski, M., McKee, M., & Khalil, A. (2006). Multi-time scale stream flow predictions: the support vector machines approach. *Journal of Hydrology*, *318*(1), 7–16.

Bayes, M., & Price, M. (1763). An essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfrs. *Philosophical Transactions (1683-1775)*, 370–418.

Beaudette, M. D. (2016). Package 'sharpshootr'.

Beven, K. J. (2011). *Rainfall-runoff modelling: the primer*. John Wiley & Sons.

bioquest.org. (2011). *Numbers count! statistics concept map*. Website. Retrieved from https://bioquest.org/numberscount/statistics-concept-map/

Bishop, C. M. (1994). Mixture density networks.

Bivand, R., Keitt, T., & Rowlingson, B. (2018). rgdal: Bindings for the 'geospatial' data abstraction library [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=rgdal (R package version 1.3-6)

Bivand, R., & Rundel, C. (2018). rgeos: Interface to geometry engine - open source ('geos') [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=rgeos (R package version 0.4-2)

Bivand, R. S., Pebesma, E., & Gomez-Rubio, V. (2013). *Applied spatial data analysis with R, second edition*. Springer, NY. Retrieved from http://www.asdar-book.org/

Bray, M., & Han, D. (2004). Identification of support vector machines for runoff modelling. *Journal of Hydroinformatics*, *6*(4), 265–280.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123–140.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Breiman, L., Friedman, J., Olshen, R. A., & Stone, C. J. (1993). Classification and regression trees. wadsworth, 1984. *Google Scholar*.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Breiman, L., et al. (2001). Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, *16*(3), 199–231.

Bronowski, J. (1988). The nature of scientific reasoning. *Occasions for Writing*, 443–45.

Brownlee, J. (2014). 4-steps to get started in machine learning: The top-down strategy for beginners to start and practice. *ML Mastery*. Retrieved from https://machinelearningmastery.com/4-steps-to-get-started-in-machine-learning/

Brownlee, J. (2020). *Machine learning mastery mindmap.* Retrieved from https://machinelearningmastery.com/

California Department of Water Resources, Bay-Delta Office. (2016). Estimates of natural and unimpaired flows for the central valley of california: Water years 1922-2014.

Cameron, A. C., Gelbach, J. B., & Miller, D. L. (2008). Bootstrap-based improvements for inference with clustered errors. *The Review of Economics and Statistics*, *90*(3), 414–427.

Chamberlain, S., & Teucher, A. (2018). geojsonio: Convert data from and to 'geojson' or 'topojson' [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=geojsonio (R package version 0.6.0)

Chaney, N. W., Wood, E. F., McBratney, A. B., Hempel, J. W., Nauman, T. W., Brungard, C. W., & Odgers, N. P. (2016). Polaris: A 30-meter probabilistic soil series map of the contiguous united states. *Geoderma*, *274*, 54–67.

Chiou, C.-C. (2008, 11). The effect of concept mapping on students' learning achievements and interests. *Innovations in Education & Teaching International*, *45*. doi: 10.1080/14703290802377240

Corporation, M., & Weston, S. (2017). doparallel: Foreach parallel adaptor for the 'parallel' package [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=doParallel (R package version 1.0.11)

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, *13*(1), 21–27.

Covington, M., Hill, K., & Bruff, D. (2012). *Math 216: Statistics for engineering vanderbilt university.* Retrieved from https://derekbruff.org/blogs/math216/

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, *20*(2), 215–232.

Cranz, A. (2018). Microsoft kinect refuses to die. *Gizmodo*. Retrieved from https://gizmodo.com/microsoft-kinect-refuses-to-die-1825847023

Crevier, D. (1993). *Ai: the tumultuous history of the search for artificial intelligence*. Basic Books, Inc.

Criss, R. E., & Winston, W. E. (2008). Do nash values have value? discussion and alternate proposals. *Hydrological Processes: An International Journal*, *22*(14), 2723–2725.

Dahl, D. B., Scott, D., Roosen, C., Magnusson, A., & Swinton, J. (2019). xtable: Export tables to latex or html [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=xtable (R package version 1.8-4)

Dawson, C. W., & Wilby, R. (1998). An artificial neural network approach to rainfall-runoff modelling. *Hydrological Sciences Journal*, *43*(1), 47–66.

Dean, J., & Ng, A. (2015). Using large-scale brain simulations for machine learning and ai. *Official Google Blog*, *26*. Retrieved from https://www.blog.google/technology/ai/using-large-scale-brain-simulations-for/

DeJong, G. (1981). Generalizations based on explanations. *Urbana*, *51*(61,801).

Dettinger, M. D., Ralph, F. M., Das, T., Neiman, P. J., Cayan, D. R., et al. (2011). Atmospheric rivers, floods and the water resources of california. *Water*, *3*(2), 445–478.

Dooge, J. C. (1973). *Linear theory of hydrologic systems* (No. 1468). Agricultural Research Service, US Department of Agriculture.

Dooge, J. C. (1986). Looking for hydrologic laws. *Water Resources Research*, *22*(9S).

Duan, N., Manning, W. G., Morris, C. N., & Newhouse, J. P. (1983). A comparison of alternative models for the demand for medical care. *Journal of business & economic statistics*, *1*(2), 115–126.

Efron, B., & Morris, C. (1977). Stein's paradox in statistics. *Scientific American*, *236*(5), 119–127.

Efron, B., & Tibshirani, R. (1997). Improvements on cross-validation: the 632+ bootstrap method. *Journal of the American Statistical Association*, *92*(438), 548–560.

Flint, L., & Flint, A. (2014). California basin characterization model: a dataset of historical and future hydrologic response to climate change. *US Geological Survey Data Release doi*, *10*, F76T0JPB.

Forest Service, USDA, Pacific Southwest Region. (2006). Existing vegetation–vegetation classification and mapping for region 5.

Friedman, J. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1). Springer series in statistics Springer, Berlin.

Fukushima, K., & Miyake, S. (1982). Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets* (pp. 267–285). Springer.

Galelli, S., & Castelletti, A. (2013). Tree-based iterative input variable selection for hydrological modeling. *Water Resources Research*, *49*(7), 4295–4310.

Garson, D. G. (1991). Interpreting neural network connection weights.

Giner, G., & Smyth, G. K. (2016). statmod: probability calculations for the inverse gaussian distribution. *R Journal*, *8*(1), 339-351.

Godsey, S. E., Kirchner, J. W., & Tague, C. L. (2014). Effects of changes in winter snowpacks on summer low flows: case studies in the sierra nevada, california, usa. *Hydrological Processes*, *28*(19), 5048–5064.

gogeometry.com. (2017). *Artificial intelligence (ai): Approaches mind map.* Retrieved from http://gogeometry.com

Goh, A. T. (1995). Back-propagation neural networks for modeling complex systems. *Artificial Intelligence in Engineering*, *9*(3), 143–151.

Goldstein, A., Kapelner, A., Bleich, J., & Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, *24*(1), 44–65.

Govindaraju, R. S., & Rao, A. R. (2013). *Artificial neural networks in hydrology* (Vol. 36). Springer Science & Business Media.

Greenwell, B. M., Boehmke, B. C., & McCarthy, A. J. (2018). A simple and effective model-based variable importance measure. *arXiv preprint arXiv:1805.04755*.

Gressmann, F., Király, F. J., Mateen, B., & Oberhauser, H. (2018). Probabilistic supervised learning. *arXiv preprint arXiv:1801.00753*.

Grubinger, T., Kobel, C., & Pfeiffer, K.-P. (2010). Regression tree construction by bootstrap: Model search for drg-systems applied to austrian health-data. *BMC Medical Informatics and Decision Making*, *10*(1), 1.

Grubinger, T., Zeileis, A., Pfeiffer, K.-P., et al. (2011). *evtree: Evolutionary learning of globally optimal classification and regression trees in r*. Department of Economics (Inst. für Wirtschaftstheorie und Wirtschaftsgeschichte).

Guisan, A., & Zimmermann, N. E. (2000). Predictive habitat distribution models in ecology. *Ecological modelling*, *135*(2), 147–186.

Han, D., Chan, L., & Zhu, N. (2007). Flood forecasting using support vector machines. *Journal of hydroinformatics*, *9*(4), 267–276.

Härdle, W., & Simar, L. (2007). *Applied multivariate statistical analysis* (Vol. 22007). Springer.

Harrell Jr, F. E., with contributions from Charles Dupont, & many others. (2020). Hmisc: Harrell miscellaneous [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=Hmisc (R package version 4.3-1)

Hart, E. M., & Bell, K. (2015). prism: Download data from the oregon prism project [Computer software manual]. Retrieved from http://github.com/ropensci/prism (R package version 0.0.6) doi: 10.5281/zenodo.33663

Hastie, T., & Tibshirani, R. (1990). *Generalized additive models* (Vol. 43). CRC Press.

Hawking, S., Musk, E., Wozniak, S., et al. (2015). *Autonomous weapons: an open letter from ai & robotics researchers. future of life institute.*

Hawkins, E., & Sutton, R. (2011). The potential to narrow uncertainty in projections of regional precipitation change. *Climate Dynamics*, *37*(1-2), 407–418.

Hayes, B., et al. (2013). First links in the markov chain. *American Scientist*, *101*(2), 252.

Hennig, C., & Kutlukaya, M. (2007). Some thoughts about the design of loss functions. *REVSTAT–Statistical Journal*, *5*(1), 19–39.

Henry, L., & Wickham, H. (2020). purrr: Functional programming tools [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=purrr (R package version 0.3.4)

Hijmans, R. J. (2019). raster: Geographic data analysis and modeling [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=raster (R package version 2.8-19)

Hijmans, R. J., Phillips, S., Leathwick, J., & Elith, J. (2017). dismo: Species distribution modeling [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=dismo (R package version 1.1-4)

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, *18*(7), 1527–1554.

Ho, T. K. (1995). Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on* (Vol. 1, pp. 278–282).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, *79*(8), 2554–

2558.

Hrachowitz, M., Savenije, H., Blöschl, G., McDonnell, J., Sivapalan, M., Pomeroy, J., ... others (2013). A decade of predictions in ungauged basins (pub)—a review. *Hydrological sciences journal*, *58*(6), 1198–1255.

Hsu, K.-l., Gupta, H. V., Gao, X., Sorooshian, S., & Imam, B. (2002). Self-organizing linear output map (solo): An artificial neural network suitable for hydrologic modeling and analysis. *Water Resources Research*, *38*(12).

Hu, T., Wu, F., & Zhang, X. (2007). Rainfall–runoff modeling using principal component analysis and neural network. *Hydrology Research*, *38*(3), 235–248.

Ingle, K. (2017). Machine learning–mind map cheatsheet. *Medium*. Retrieved from https://medium.com/@karan.ingle/machine-learning-mind-map-cheatsheet-cb200b2246fe

Iorgulescu, I., & Beven, K. J. (2004). Nonparametric direct mapping of rainfall-runoff relationships: An alternative approach to data analysis and modeling? *Water Resources Research*, *40*(8).

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 6). Springer.

Jarvis, A., Reuter, H. I., Nelson, A., Guevara, E., et al. (2008). Hole-filled srtm for the globe version 4. *available from the CGIAR-CSI SRTM 90m Database*. Retrieved from http://srtm.csi.cgiar.org

Jorgensen, B. (1997). *The theory of dispersion models*. CRC Press.

Klemes, V. (1982). Empirical and causal models in hydrology.

Klemeš, V. (1986). Operational testing of hydrological simulation models. *Hydrological Sciences Journal*, *31*(1), 13–24.

Kratzert, F., Klotz, D., Herrnegger, M., Sampson, A. K., Hochreiter, S., & Nearing, G. S. (2019). Toward improved predictions in ungauged basins: Exploiting the power of machine learning. *Water Resources Research*, *55*(12), 11344–11354.

Krause, P., Boyle, D., & Bäse, F. (2005). Comparison of different efficiency criteria for hydrological model assessment. *Advances in geosciences*, *5*, 89–97.

Lahiri, S. N. (2013). *Resampling methods for dependent data*. Springer Science & Business Media.

Legates, D. R., & McCabe Jr, G. J. (1999). Evaluating the use of goodness-of-fit measures in hydrologic and hydroclimatic model validation. *Water resources research*, *35*(1), 233–241.

Legendre, P. (1993). Spatial autocorrelation: trouble or new paradigm? *Ecology*, *74*(6), 1659–1673.

Levins, R. (1966). The strategy of model building in population biology. *American scientist*, *54*(4), 421–431.

Li, T., & Srikumar, V. (2019). Augmenting neural networks with first-order logic. *arXiv preprint arXiv:1906.06298*.

Liaw, A., & Wiener, M. (2002a). Classification and regression by randomforest. *R News*, *2*(3), 18-22. Retrieved from https://CRAN.R-project.org/doc/Rnews/

Liaw, A., & Wiener, M. (2002b). Classification and regression by randomforest. *R news*, *2*(3), 18–22.

Lin, J.-Y., Cheng, C.-T., & Chau, K.-W. (2006). Using support vector machines for long-term discharge prediction. *Hydrological Sciences Journal*, *51*(4), 599–612.

Magnuson-Skeels, B. (2016). *Using machine learning to statistically predict natural flow.* MS Thesis.

Makridakis, S. (1993). Accuracy measures: theoretical and practical concerns. *International Journal of Forecasting*, *9*(4), 527–529.

Marr, B. (2016). A short history of machine learning every manager should read. *Forbes*. Retrieved from http://tinyurl.com/gslvr6k

Mauricio Zambrano-Bigiarini. (2017). hydrogof: Goodness-of-fit functions for comparison of simulated and observed hydrological time series [Computer software manual]. Retrieved from http://hzambran.github.io/hydroGOF/ (R package version 0.3-10) doi: 10.5281/zenodo.840087

McKay, L., Bondelid, T., Dewald, T., Johnston, J., Moore, R., & Rea, A. (2012). Nhdplus version 2: user guide. *National Operational Hydrologic Remote Sensing Center, Washington, DC*.

Microsoft, & Weston, S. (2017). foreach: Provides foreach looping construct for r [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=foreach (R package version 1.4.4)

Min, Y., & Agresti, A. (2002). Modeling nonnegative data with clumping at zero: a survey. *Journal of the Iranian Statistical Society*, *1*(1), 7–33.

Minns, A., & Hall, M. (1996). Artificial neural networks as rainfall-runoff models. *Hydrological sciences journal*, *41*(3), 399–417.

Moriasi, D., Arnold, J., Van Liew, M., Bingner, R., Harmel, R., & Veith, T. (2007). Model evaluation guidelines for systematic quantification of accuracy in watershed simulations. *Transactions of the ASABE*, *50*(3), 885-900. Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-34447500396&partnerID=40&md5=50b5724614f28257edef46d43db96018 (cited By 2311)

Nelder, J. A., & Wedderburn, R. W. M. (1972). *Generalized linear models*. Wiley Online Library.

Neuwirth, E. (2014). Rcolorbrewer: Colorbrewer palettes [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=RColorBrewer (R package version 1.1-2)

O'Connor, J., & Robertson, E. (2000). Biography of pierre-simon laplace and article on orbits and gravitation. *Published by School of Mathematics and Statistics, University of St Andrews, Scotland.*. Retrieved from http://www-history.mcs.standrews.ac.uk/history/Mathematicians/Laplace.html

Olden, J. D., Joy, M. K., & Death, R. G. (2004). An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. *Ecological modelling*, *178*(3-4), 389–397.

Ooms, J. (2019). magick: Advanced graphics and image-processing in r [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=magick (R package version 2.1)

Pebesma, E. J., & Bivand, R. S. (2005, November). Classes and methods for spatial data in R. *R News*, *5*(2), 9–13. Retrieved from https://CRAN.R-project.org/doc/Rnews/

Pierce, D. W., Kalansky, J. F., Cayan, D. R., et al. (2018). *Climate, drought, and sea level rise scenarios for california's fourth climate change assessment* (Tech. Rep.). Technical Report CCCA4-CEC-2018-006, California Energy Commission.

Pike, R. J., & Wilson, S. E. (1971). Elevation-relief ratio, hypsometric integral, and geomorphic area-altitude analysis. *Geological Society of America Bulletin*, *82*(4), 1079–1084.

Poff, N. L., Allan, J. D., Bain, M. B., Karr, J. R., Prestegaard, K. L., Richter, B. D., ... Stromberg, J. C. (1997). The natural flow regime. *BioScience*, *47*(11), 769–784.

R Core Team. (2019). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria. Retrieved from https://www.R-project.org/

Ren, H., Stewart, R., Song, J., Kuleshov, V., & Ermon, S. (2018). Learning with weak supervision from physics and data-driven constraints. *AI Magazine*, *39*(1), 27–38.

Roberts, D. R., Bahn, V., Ciuti, S., Boyce, M. S., Elith, J., Guillera-Arroita, G., ... others (2017). Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*.

Rolle, O. (2015). Googles tensorflow and microsofts dmtk goes open source. *PosiDev Blog*. Retrieved from http://posidev.com/blog/2015/11/14/googles-tensorflow-and-microsofts-dmtk-goes-open-source/

Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton project para.* Cornell Aeronautical Laboratory.

RStudio Team. (2016). Rstudio: Integrated development environment for r [Computer software manual]. Boston, MA. Retrieved from http://www.rstudio.com/

Sarkar, D. (2008). *Lattice: Multivariate data visualization with r.* New York: Springer. Retrieved from http://lmdvr.r-forge.r-project.org (ISBN 978-0-387-75968-5)

Sherman, L. K. (1932). Streamflow from rainfall by the unit-graph method. *Eng. News Record*, *108*, 501–505.

Simonite, T. (2014). Software that matches faces almost as well as you do. *Technology Review*, *117*(3), 19–19.

Singh, V. P., & Frevert, D. K. (2005). *Watershed models.* CRC Press.

Sivapalan, M. (2003). Prediction in ungauged basins: a grand challenge for theoretical hydrology. *Hydrological Processes*, *17*(15), 3163–3170.

Sivapalan, M., Takeuchi, K., Franks, S., Gupta, V., Karambiri, H., Lakshmi, V., ... others (2003). Iahs decade on predictions in ungauged basins (pub), 2003–2012: Shaping an exciting future for the hydrological sciences. *Hydrological sciences journal*, *48*(6), 857–880.

Spence, C., & Woo, M.-k. (2006). Hydrology of subarctic canadian shield: heterogeneous headwater basins. *Journal of Hydrology*, *317*(1-2), 138–154.

Stigler, S. M. (1981). Gauss and the invention of least squares. *The Annals of Statistics*, 465–474.

Stoffer, D. (2020). astsa: Applied statistical time series analysis [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=astsa (R package version 1.10)

Sugiyama, M., & Kawanabe, M. (2012). *Machine learning in non-stationary environments: Introduction to covariate shift adaptation.* MIT press.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction.* MIT press.

Taylor, K. E., Stouffer, R. J., & Meehl, G. A. (2012). An overview of cmip5 and the experiment design. *Bulletin of the American Meteorological Society*, *93*(4), 485–498.

Tobin, J. (1958). Estimation of relationships for limited dependent variables. *Econometrica: journal of the Econometric Society*, 24–36.

Todini, E. (1988). Rainfall-runoff modeling past, present and future. *Journal of Hydrology*, *100*(1), 341–352.

Tokar, A. S., & Johnson, P. A. (1999). Rainfall-runoff modeling using artificial neural networks. *Journal of Hydrologic Engineering*, *4*(3), 232–239.

Van Vuuren, D. P., Edmonds, J., Kainuma, M., Riahi, K., Thomson, A., Hibbard, K., ... others (2011). The representative concentration pathways: an overview. *Climatic change*, *109*(1-2), 5.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE transactions on neural networks*, *10*(5), 988–999.

Wickham, H. (2007). Reshaping data with the reshape package. *Journal of Statistical Software*, *21*(12), 1–20. Retrieved from http://www.jstatsoft.org/v21/i12/

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. Retrieved from https://ggplot2.tidyverse.org

Winston, P. (2010). *6.034 artificial intelligence, fall 2010. massachusetts institute of technology: MIT opencourseware.* Retrieved from https://ocw.mit.edu.License: CreativeCommonsBY-NC-SA

Woodie, A. (2014). Inside sibyl, google's massively parallel machine learning platform. *Datanami*. Retrieved from https://www.datanami.com/2014/07/17/inside-sibyl-googles-massively-parallel-machine-learning-platform/

Worland, S. C., Farmer, W. H., & Kiang, J. E. (2018). Improving predictions of hydrological low-flow indices in ungaged basins using machine learning. *Environmental Modelling & Software*, *101*, 169–182.

Wuertz, D., Setz, T., & Chalabi, Y. (2017). fbasics: Rmetrics - markets and basic statistics [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=fBasics (R package version 3042.89)

Yu, X., Liong, S.-Y., & Babovic, V. (2004). Ec-svm approach for real-time hydrologic forecasting. *Journal of Hydroinformatics*, *6*(3), 209–223.

Zehe, E., & Sivapalan, M. (2008). Threshold behavior in hydrological systems and geo-ecosystems: manifestations, controls and implications for predictability. *Hydrology & Earth System Sciences Discussions*, *5*(6).

Zeileis, A., & Grothendieck, G. (2005). zoo: S3 infrastructure for regular and irregular time series. *Journal of Statistical Software*, *14*(6), 1–27. doi: 10.18637/jss.v014.i06

Zeileis, A., & Hothorn, T. (2002). Diagnostic checking in regression relationships. *R News*, *2*(3), 7–10. Retrieved from https://CRAN.R-project.org/doc/Rnews/

Elaheh White
September 2020
Civil and Environmental Engineering

Statistical Learning for Unimpaired Flow Prediction in Ungauged Basins

Abstract

All science is the search for unity in hidden likeness (Bronowski, 1988). There are two practical reasons to approximate processes that produce such hidden likeness: (1) *prediction* for interpolation or extrapolation to unknown (often future) situations; and (2) *inference* to understand how variables are connected or how change in one affects others. Statistical learning tools aid prediction and at times inference. In recent years, rapidly growing computing power, the advent of machine learning algorithms, and more user-friendly programming languages (e.g., R and Python) support applying statistical learning methods to broader societal problems.

This dissertation develops statistical learning models, generally simpler than mechanistic models, to predict unimpaired flows of California basins from available data. Unimpaired flow is the flow produced by the basin in its current state, but without human-created or operated water storage, diversion, or return flows (California Department of Water Resources, Bay-Delta Office, 2016). The models predict unimpaired flows for ungauged basins, an International Association of Hydrological Sciences "grand challenge" in hydrology. In Predicting Ungauged Basins (PUB), the models learn from information at gauged points on a river and extrapolate to ungauged locations.

Several issues arise in this prediction problem: (1) How we view hydrology and how we define observational units determine how data is pre-processed for statistical learning methods. So, one issue is in deciding the organization of the data (e.g., aggregate vs. incremental basins). Such data transformation or pre-processing is explored in Chapter 2. (2) Often, water resources problems are not concerned with accurately predicting the expectation (or mean) of a distribution but require better estimates of extreme values of the distribution (e.g., floods and droughts). Solving this problem involves defining asymmetric loss functions, which is presented in Chapter 3. (3) Hydrologic observations have inherent dependencies and correlation structure; gauge data are structured in time and space, and rivers form a network of flows that feed into one another (i.e., temporal, spatial, and hierarchical autocorrelation). These characteristics require careful construction of resampling techniques for model error estimation, which is discussed in Chapter 4. (4) Non-stationarity due to climate change may require adjustments to statistical models, especially for long-term decision-making. Chapter 5 compares unimpaired flow predictions from a statistical model that uses climate variables representing future hydrology to projections from climate models.

These issues make Predicting Ungauged Basins (PUB) a non-trivial problem for statistical learning methods operating with no *a priori* knowledge of the system. Compared to physical or semi-physical models, statistical learning models learn from the data itself, with no assumptions on underlying processes. Their advantages lie in their fast and easy development, simplicity of use, lesser data requirements, good performance, and flexibility in model structure and parameter specifications. In the past two decades, more sophisticated statistical learning models have been applied to rainfall-runoff modeling. However, with these

methods, there are issues such as the danger of overfitting, their lack of justification outside the range of underlying data sets, complexity in model structure, and limitations from the nature of the algorithms deployed.