Chapter 5: APPLICABILITY OF SURROGATE MODELS IN LARGE SCALE GW OPTIMIZATION

5.1 Introduction

Surrogate modeling is a widely used technique in simulation optimization of water resource problems to address the challenges posed by computationally expensive simulations. These models, called metamodels or emulators, are simplified approximations of complex, high-fidelity simulation models. Water resource systems, such as groundwater-surface water interactions, reservoir management, or urban drainage systems, often require extensive computational resources to evaluate performance metrics due to the complexity of underlying physical and hydrological processes. Surrogate models, such as Kriging, polynomial regression, radial basis functions, or machine learning models (e.g., neural networks and random forests), are trained using a limited number of simulation outputs to approximate the behavior of the complete simulation model. It significantly reduces computational costs and enables more efficient exploration of the decision space, allowing for extensive analysis. They are particularly valuable in iterative optimization approaches, where multiple model evaluations are needed, and in real-time decision-making contexts. Compared to traditional numerical methods, surrogate models come with several distinct advantages: (1) Universality: it is a versatile method capable of solving various types of problems across arbitrary domains; (2) Mesh-Free Nature: it eliminates the need for mesh generation, requiring only a reasonable distribution of sample points, simplifying the process significantly; (3) Efficiency: traditional simulation methods, become

computationally expensive as the number of time iterations increases, with predictions requiring sequential computation. In contrast, a trained machine learning model can rapidly predict solutions at any given time, making it a valuable surrogate model for water resource management; (4) Flexibility: traditional numerical methods are limited to providing solutions at grid points and require additional post-processing to approximate solutions at non-grid points, whereas surrogate models can directly predict solutions at any spatial-temporal location. The applicability of surrogates in groundwater optimization problems can be summarized in Figure 5.1

In this chapter, the applicability of surrogate models was tested via two tests. The first task identified the efficiency of different techniques to predict the head values based on different aquifer properties and simulation model data. The pumping rates and heads were considered transient. Therefore, training the GW surrogate models depended on the initial conditions of time-varying heads during the training period. Physics-Informed Neural Networks and machine learning methods such as XGBoost, Bayesian regression, and Artificial Neural Networks were employed. The best-performing model from this test was then integrated into a surrogate-assisted simulation-optimization framework. The surrogate models were compared using root mean square error (RMSE), R- squared (R^2), and Root Mean Squared Log Error (RMSLE) mean absolute error (MAE). In the second test, the impact of incorporating the surrogate model was quantified by analyzing the quality of the Pareto front in the optimization results.

This chapter also determines the applicability of PINNs in predicting GW heads in simple and complex domains. It was found that the efficiency of using machine learning to predict the head values was superior to PINNs. However, PINN performed equally well for simple domains, giving smooth output data.

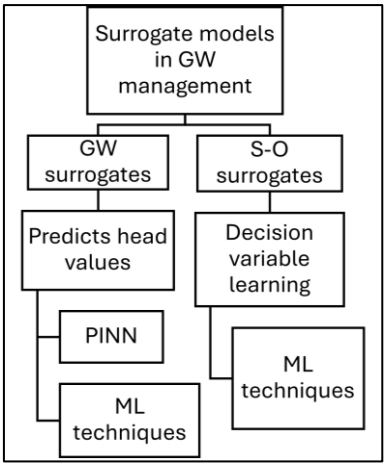

Figure 5.1 Types of surrogates used for groundwater optimization

5.2 Methods

The study was tested on the lower Ain basin, for which the input data was generated by running the groundwater flow model multiple times and recording the input parameters. For all the models, the input dataset was Space (X, Y), time (t), hydraulic conductivity (HK), total source/sink (S), and specific yield (W), while the output was the head value (h). A summary of the dataset is given in Table 5.1.

Table 5.1 Summary statistics of GW prediction dataset

Property	Count	Mean	Std deviation	Min	Max
X coordinate (x)	55000	13435.46447	5875.730451	623.932991	25331.67944
Y coordinate (y)	55000	20173.55932	8360.924404	3847.414749	37357.15611
Time (t)	55000	1176.824893	899.7650192	0	2556.001953
Hydraulic conductivity (HK)	55000	483.3989136	658.3084167	5	3456
Specific yield (W)	55000	-5.90E-06	0.000283553	-0.008321	0.00359595
Sources/sinks (S)	55000	0.055267171	0.024684041	0.01	0.170000002
Head (h)	55000	220.8253217	32.60017606	-888	285.5195007

5.2.1 Physics-Informed Neural Networks

The PINN model addresses partial differential equations (PDEs) by integrating deep neural networks with the automatic differentiation (AD) algorithm. Unlike finite-difference (FD)-based numerical derivatives, such as those used in models like MODFLOW, the AD algorithm relies on the chain rule to compute derivatives. This approach enables the encoding of PDEs within a deep learning framework, facilitating the discovery of nonlinear relationships between input parameters (e.g., spatial and temporal variables like (x) and (t) and output quantities by minimizing a loss function. The working methodology of PINN is conceptually illustrated in Figure 5.2, where the model is trained on the input dataset of space, time, and sources/sinks. The predicted head dataset is incorporated with the loss function as the GW partial differential equation.

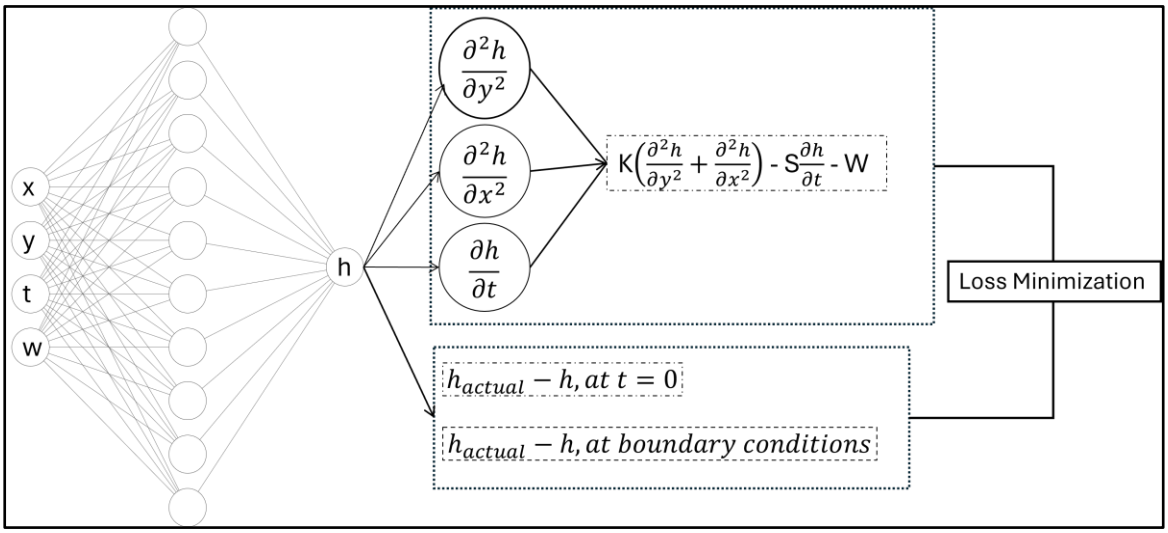

Figure 5.2 PINN architecture

In this chapter, the PINN model utilized 40,000 training points and 10,000 testing points to learn the underlying patterns and evaluate its performance effectively. The neural network architecture consisted of 7 hidden layers, each containing 40 neurons, optimized to capture the complexity of the problem. The ReLU (Rectified Linear Unit) activation function introduced nonlinearity, enabling the model to handle intricate relationships in the data.

The training process was conducted over two configurations: 2000 and 5000 epochs, allowing for sufficient iterations to minimize the loss and improve model accuracy.

5.2.2 Machine Learning techniques

Three machine learning approaches were used to predict the GW head values: XgBoost, bayesian regression, and simple ANN. The following section discusses the types of algorithms used.

5.2.2.1 XGBoost Regressor

XGBoost, or Extreme Gradient Boosting, is a robust machine learning algorithm based on the gradient boosting framework. It is designed to enhance predictive accuracy by iteratively combining the outputs of multiple weak learners, typically decision trees, to form a strong, cohesive model. XGBoost is particularly effective in water resources because it can model complex, non-linear relationships between input variables such as precipitation, aquifer properties, and historical data. Given a dataset of Pareto front, denoted as PF, with m features and n examples, PF $= \{(x_i - y_i): i = 1 \ldots . n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R}\}$. Let Y_i represent the anticipated outcome of an ensemble tree model that is created using the equations provided below:

$$A_{.i} = \mathrm{B}(x_i) = \sum_{k=1}^{K} f_k(x_i), f_k \qquad (5.1)$$

To solve the given equation by minimizing the loss and regularization objective, we must determine the optimal set of functions, denoted as f_k, where k indicates the number of trees in the model. Boosting is a technique used in decision trees to train the model and minimize the objective function. It involves continuously adding a new function *f* to the model during training. Therefore, in the t-th iteration, a novel function (tree) is incorporated in the following manner:

$$\xi = \sum_{i=1}^{n} l(y_i A._i^{(t-1)} + f_t(x_i)) + \mu(f_t) \qquad (5.2)$$

$$\xi_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda} + \frac{\left(\sum_{i\in J} g_i\right)^2}{\sum_{i\in J} h_i + \lambda} - \frac{(\sum_{i\in L} g_i)^2}{\sum_{i\in L} h_i + \lambda}\right] - \zeta \qquad (5.3)$$

The algorithm employs regularization techniques, L1 and L2 penalty regularization, which help prevent overfitting, which is crucial when working with S-O datasets with high variability or noise. XGBoost was utilized with the following hyperparameter settings: the learning_rate was set to 0.01, which determines the step size at each iteration during the training process; the max_depth was specified as 5, controlling the maximum depth of the individual trees to prevent overfitting; and the n_estimators was set to 100, indicating the number of boosting rounds or trees in the model.

5.2.2.2 Support vector machine regressor

The Support Vector Machine Regressor (SVR) is a supervised machine learning algorithm widely used for regression tasks. SVR operates on the structural risk minimization principleation, which aims to find a function $f(x)$ that maps input features x to output values y, while ensuring that the predictions are as accurate as possible within a defined tolerance margin ϵ. Mathematically, SVR minimizes the following objective function:

$$\frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i^*) \qquad (5.4)$$

Subject to the constraints:

$$y_i - (\mathbf{w}^T \phi(x_i) + b) \leq \epsilon + \xi_i,\ (\mathbf{w}^T \phi(x_i) + b) - y_i \leq \epsilon + \xi_i^*,\ \xi_i, \xi_i^* \geq 0 \qquad (5.5)$$

where

w is the weight vector, $\phi(xi)$ represents a kernel-transformed feature space, b is the bias term, ξ_i and ξ_i^* are slack variables for handling data points outside the ϵ-margin, and C is a

regularization parameter that balances the trade-off between model complexity and prediction error. In groundwater head prediction, SVR maps input features into a high-dimensional space using kernel functions (e.g., radial basis function (RBF) or polynomial kernels) to capture nonlinear relationships. Focusing only on the support vectors—data points outside or on the margin—ensures computational efficiency and generalization to unseen data. This ability to model complex, nonlinear interactions makes SVR a valuable tool in groundwater hydrology, where spatial and temporal dependencies often exhibit nonlinear characteristics.

5.2.2.3 Artificial neural networks

ANNs are computational models inspired by the functioning of biological neural networks designed to approximate complex relationships between inputs and outputs. In this chapter, Multi-layer perceptron (MLP) architecture was used. In predicting groundwater head values, MLPs are highly effective due to their ability to learn and capture nonlinear patterns from data. MLP is a neural network composed of fully connected dense layers that map input data from one dimension to another. It is termed "multi-layer" because it includes an input layer, one or more hidden layers, and an output layer. The primary objective of an MLP is to capture and model complex relationships between inputs and outputs, making it a highly effective tool for diverse machine-learning applications. During forward propagation, the model computes a weighted sum of the inputs for each neuron and passes it through the activation function to generate an output, which is then propagated through all layers to produce the final prediction.

The model is trained by adjusting its weights and biases to minimize a loss function, such as the mean squared error between observed and predicted groundwater head values. This process, known as backpropagation, calculates the gradients of the loss function concerning the weights and updates them using optimization algorithms like gradient descent. Through

iterative training, the ANN learns to map input features, such as recharge rates and boundary conditions, to the corresponding groundwater head values.

5.2.3 ANN- SO Model Development

This work produced 10^4 random data points using a calibrated groundwater simulation model, with zone-wise discharge as the input, leakage out (L with penalty), and total discharge (Q with penalty) as the output. The data was generated using PyGWMO for MOPSO, NSGA-II, and MOEA/D algorithms, and random data points were chosen from the model to cover the whole domain. The total data is divided into three subsets: training (70%), validation (15%), and testing (15%). An MLP with a single hidden layer and enough hidden layer size (neurons) is capable of any input-output mapping. However, the ANN training time also increases with the size of the training dataset and layer size (neurons). This makes combining an extensive training data set and layer size infeasible. The model was trained in MATLAB 2021, a neural net fitting toolbox. Different layer sizes (20, 10, 25) were tested, and the best ANN model was selected based on R-square and Mean Absolute Percentage Error (MAPE) of Leakage out because the total discharge predictions were accurate due to linear relationships between the input and outputs. Along with these matrices, the values of Maximum Absolute Error (MXAE) and Root Mean Squared Error (RMSE) are shown in Table 5.2. The ANN-SO model generated different pareto sets for various models and performed a pareto comparison.

Table 5.2 ANN optimization results for different combinations of layer size for all three domains. The highlighted row indicates the best parameters for the ANN model.

Model 1

	Leakage out				**Total discharge**			
Layer size	MXAE	RMSE	R square	MAPE (%)	MXAE	RMSE	R square	MAPE (%)
20	11172.1	1856.1	0.984	1.010	1496.2	323.5	1.000	0.105
10	12523.0	1970.5	0.982	1.140	1622.5	351.1	1.000	0.114

25	12713.2	1872.8	0.984	1.035	1735.5	312.7	1.000	0.095

Model 2

	Leakage out				**Total discharge**			
Layer size	MXAE	RMSE	R square	MAPE (%)	MXAE	RMSE	R square	MAPE (%)
20	92676.2	5677.8	0.974	25.963	91579.4	5513.9	0.980	1.601
10	93669.8	5482.7	0.976	**27.137**	91244.6	5219.5	0.982	1.715
25	94107.8	6822.7	0.963	28.047	94727.1	6663.2	0.971	2.038

Model 3

	Leakage out				**Total discharge**			
Layer size	MXAE	RMSE	R square	MAPE (%)	MXAE	RMSE	R square	MAPE (%)
20	4033.8	456.6	0.999	0.456	832.3	220.0	1.000	0.070
10	3676.8	584.6	0.998	0.630	897.0	241.8	1.000	0.080
25	3017.9	435.3	0.999	0.433	940.4	225.4	1.000	0.072

5.3 Results

The prediction of head values and the impact of introducing surrogates on the S-O model is discussed in this section.

5.3.1 Head prediction

ANN performs better than machine learning algorithms for predicting head values with an RMSE of 4.63. At the same time, the PINN struggles to capture the variability of aquifer parameters and gives scattered variations of head values. In the machine learning techniques, the SVM regressor performs the worst with an RMSE of 25.09. However, it is still better performing than PINNs. The PINN model was retrained with 5000 epochs to determine the accuracy. At 5000 epochs, the model can capture some of the variability in the center. However, the results were consistently poor. It could be attributed to the high variability in aquifer properties like hydraulic conductivity, which varies from 0.008 m/d to 3450 m/d due to graveled riverbeds. Different scaling features, such as taking logarithmic

scales of the values, were also tried but did not improve the results. In terms of training time, the time required by XgBoost was the least, while ANN came close.

The ANN with MLP architecture outperforms every other algorithm, as it almost perfectly captures the input and output relation. In Figure 5.3, RMSE is greatly reduced after 100 epochs and uniform in training and validation datasets. For PINN, the training RMSE was lower than the validation dataset. The summary of the RMSE for all the algorithms is in Table 5.3

Table 5.3 RMSE table for different algorithms

	PINN	XgBoost	Bayesian	ANN
Training time (hours)	8.5	4.1	5.5	4.76
RMSE (training)	37.21	18.91	25.09	4.63
RMSE (validation)	32.12	21.87	25.30	4.98

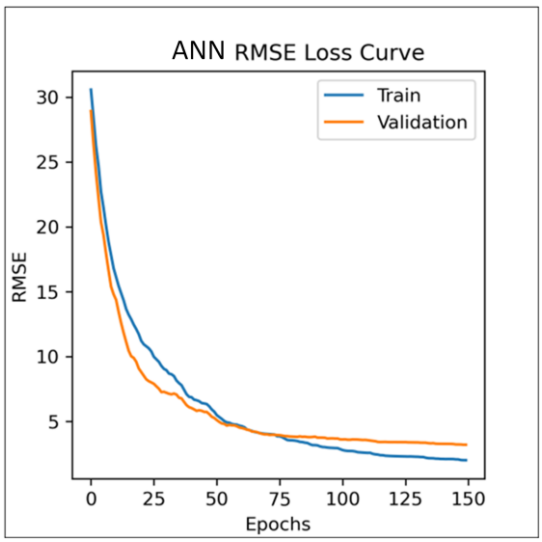

Figure 5.3 RMSE loss over epochs for the ANN model

In the first row of Figure 5.4, corresponding to the PINN model at 2000 epochs, the predicted head values exhibit smoother transitions in the spatial domain but lack fine detail in regions with high variability. This is evident in the uniform gradients and muted representation of abrupt changes, particularly in areas with steep hydraulic gradients. By

contrast, the second row, representing the PINN model at 5000 epochs, shows improved spatial resolution and better alignment with the MODFLOW results, as seen in the increased differentiation in high and low head zones. However, major discrepancies persist in delineating certain boundaries, where the PINN outputs appear more segmented than the smoother transitions in MODFLOW (row 3).

The color gradients in both PINN predictions signify head values ranging from approximately 180 m (blue) to 260 m (red), reflecting spatial variations in groundwater levels. While the PINN outputs converge toward MODFLOW results with increasing epochs, the MODFLOW predictions remain more uniform and continuous, particularly in areas of low head variability. The differences in the shapes and gradients of the output maps suggest that further training of the PINN model or additional refinement of its parameters may enhance its predictive capabilities, particularly in capturing the finer spatial heterogeneity observed in the MODFLOW results.

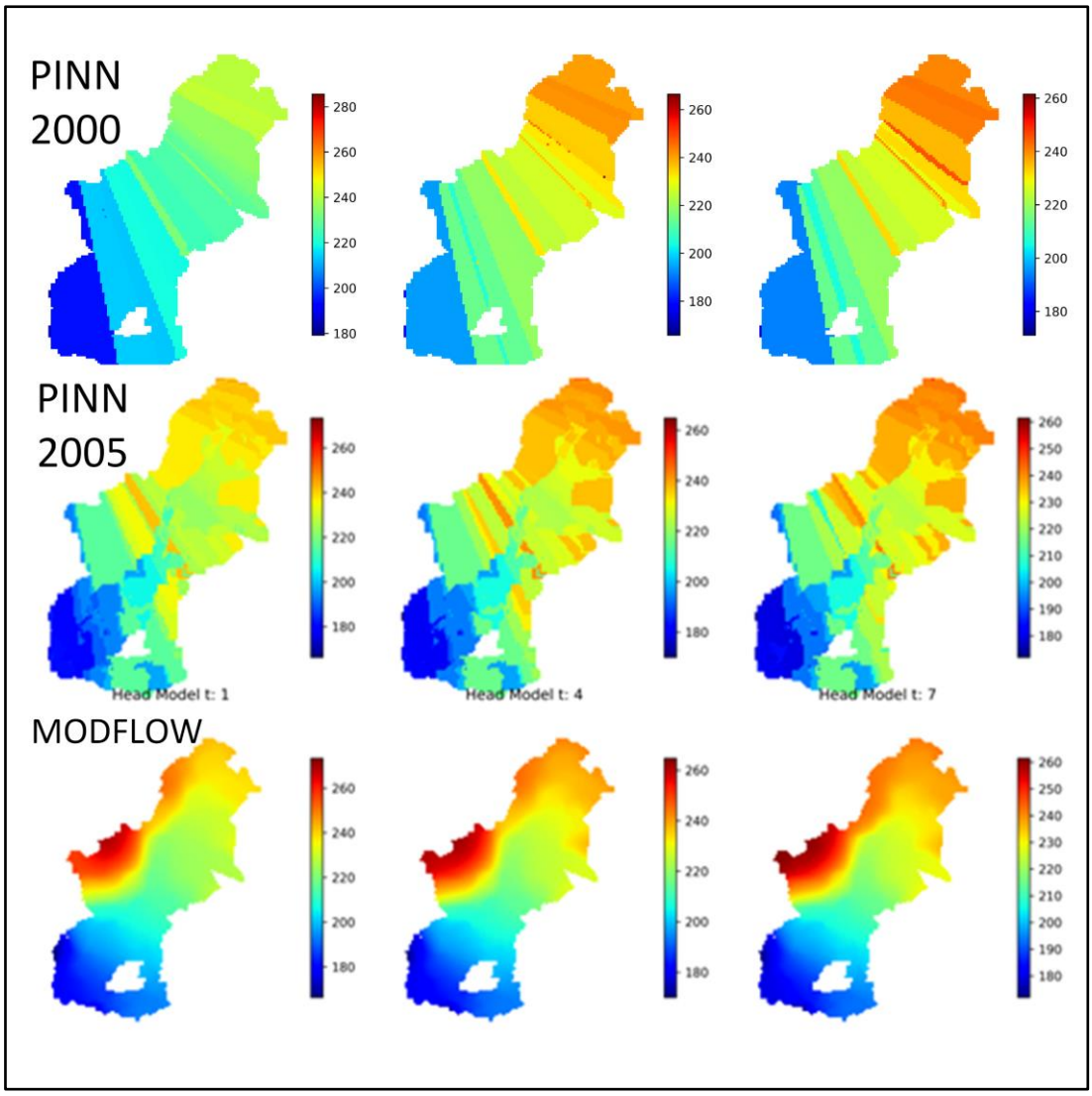

Figure 5.4 Predicted head values by PINN with 2000 and 5000 epochs at three-time steps. MODFLOW results are presented in the third row.

The head values obtained from the XgBoost could capture the head and topology variation. However, the Ain River, acting as a source/sink, was not captured properly. The SVM had similar results. However, the ANN model was more efficient in terms of capturing the presence of streams and small head variations in the model as shown in Figure 5.5.

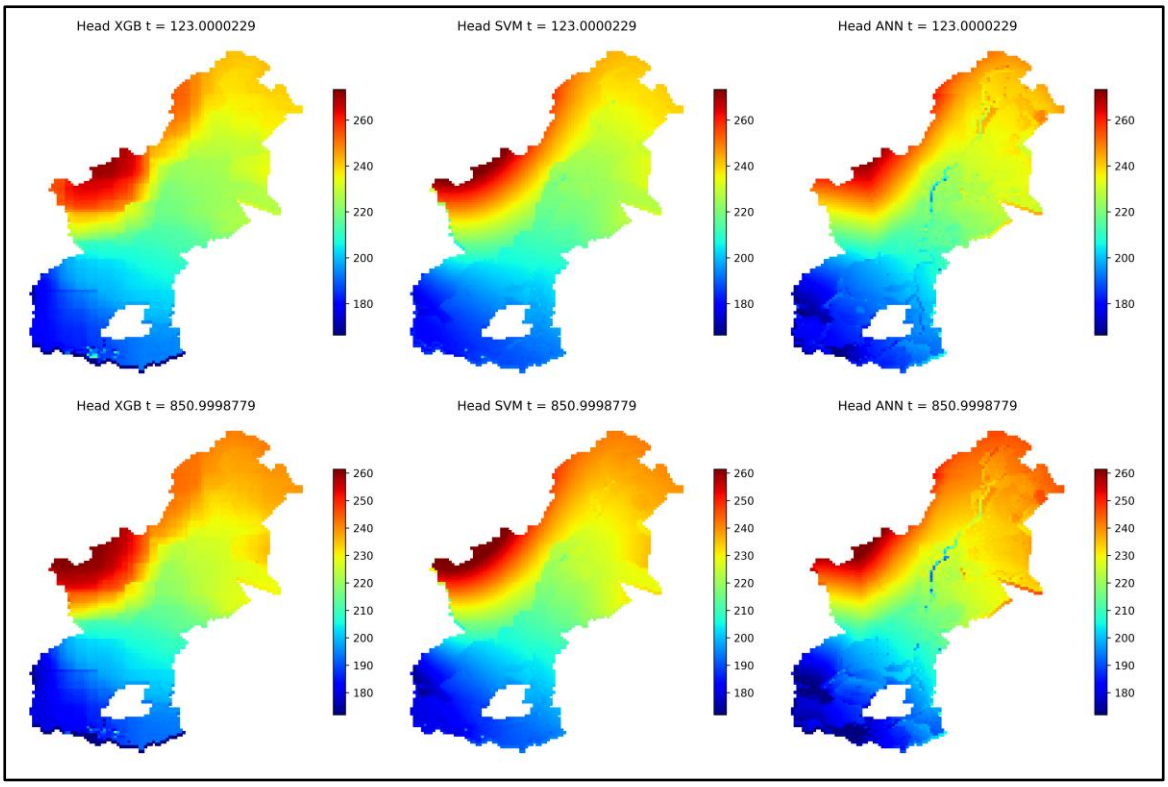

Figure 5.5 Predicted head values by SVM, XgBoost, and ANN

5.3.2 ANN-SO model

After introducing ANN-SO, MOPSO significantly improved the solutions' convergence, diversity, and uniformity (Figure 5.6). MOEA/D showed improved diversity and uniformity, whereas convergence was still like MOEA/D without ANN. Although NSGA-II showed no major improvement in the convergence, the uniformity of Pareto solution distribution increased. The diversity and uniformity of the NSGA-II and MOPSO are close and comparable. Thus, in terms of convergence, the solution points of MOPSO dominate every other solution, and the solution points of NSGA-II are dominated by most of the different points. MOEAD solutions lie between the NSGA-II and MOPSO fronts. Thus, the efficacy of the ANN model is compared using standard statistical measures.

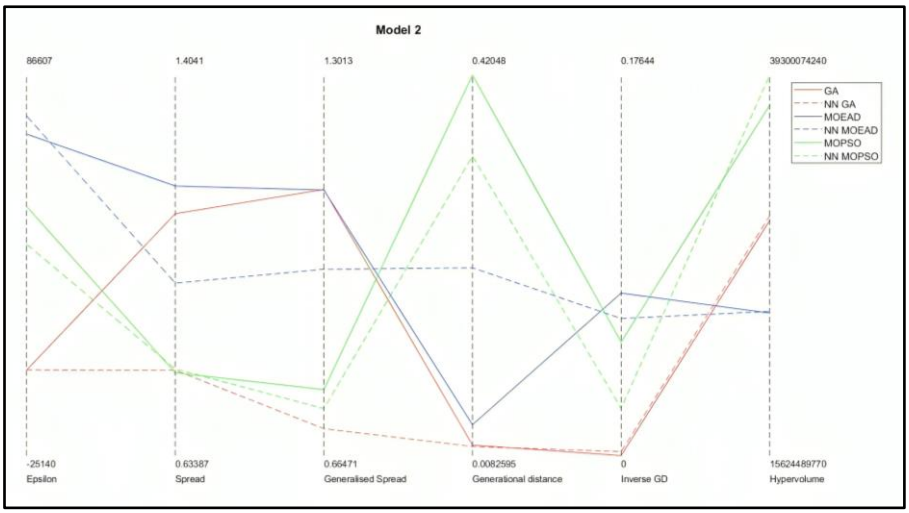

Figure 5.6 Pareto performance metrics after introducing ANN-SO

The analysis shows that surrogate models effectively reduce simulation time and generate a pareto front closer to or better than regular simulation models as much as a 50% increase in hypervolume was observed when the ANN was utilized as the simulator in S-O models as shown in Figure 5.7.

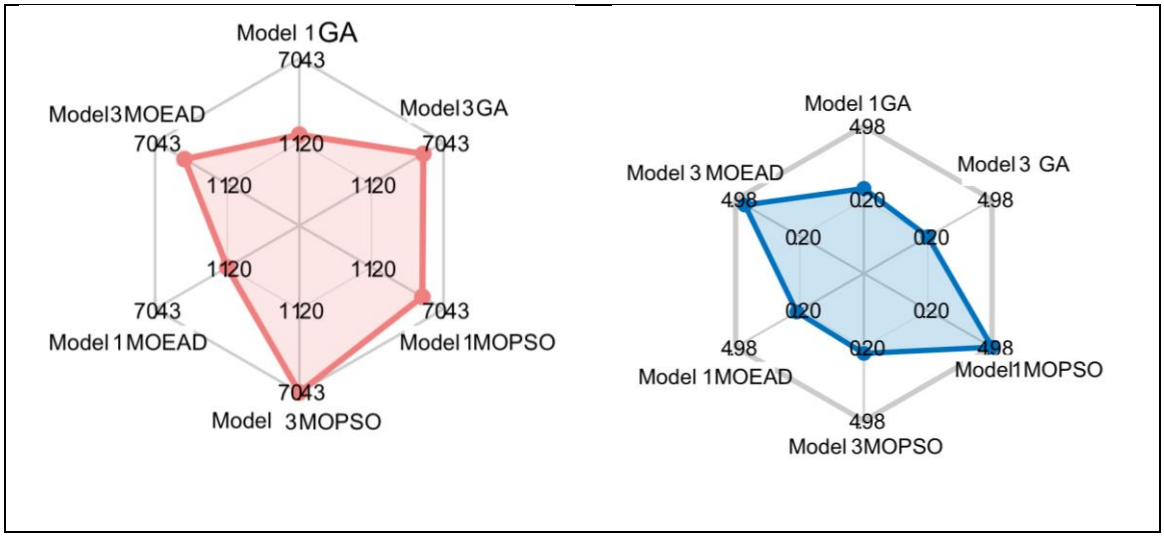

Figure 5.7 Percentage (a) increase in hypervolume (b) decrease in IGD after introduction of ANN

The MOPSO algorithm performs better, as can be seen visually in Figure 5.8. Only MOPSO and MOEA/D solutions are spread out with greater convergence compared to NSGA-II.

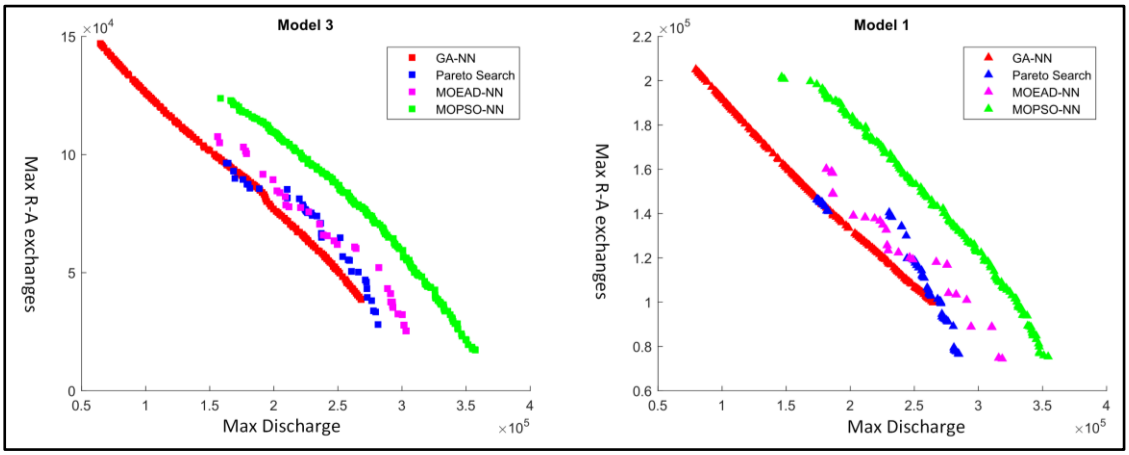

Figure 5.8 Pareto fronts for ANN-SO

5.4 Summary

This chapter tests the applicability of machine learning models and PINN to predict 3D GW heads in the transient heterogeneous aquifer in the LARB. The heads obtained from the ML models are compared with the standard MODFLOW to find the relative accuracy. Further, an S-O model was developed in PyGWMO for two scenarios: one incorporating an ANN model to speed up the optimization process and the other excluding it. The pareto front obtained was compared based on convergence and diversity to determine the improvements after the ANN-SO model. In the context of ANN-SO, MOPSO demonstrated significant advancements in solution diversity and convergence, while the NSGA-II solutions did not exhibit substantial improvements in convergence. This observation suggests that NSGA-II can derive a preliminary solution set independent of the ANN model but cannot be used as the standalone dataset for surrogate model development. In the case of PINN, the study demands further training and calibration of the input data sets. PINN could not capture the intricate aquifer properties that varied for complex domains. The overall conclusion from this chapter is that after the introduction of ANN-SO, there is a general trend of increase in hypervolume values, whereas NSGA-II shows consistent performance.