# ***Experiments* 1**

Chapter 1

**Experiments**

We conducted a series of experiments to measure the NKN’s predictive ability in several settings: time series, regression benchmarks, and texture images. We focused in particular on extrapolation, since this is a strong test of whether it has uncovered the underlying structure. Furthermore, we tested the NKN on Bayesian Optimization, where model structure and calibrated uncertainty can each enable more efficient exploration.

## **1.1. Activation function**

Analogously to ordinary neural nets, each layer may also include a nonlinear activation function, the pre-activations, are the result of a linear combination or product. However, must be selected with care in order to ensure closure of the kernels. Polynomials with positive coefficients, as well as the exponential function, fulfill this requirement.

|  |  |
| --- | --- |
|  | (1) |

Allowing units in NKN to represent complex-valued kernels as in Definition 1 and take the real component only at the end, can make the network’s representations significantly more compact. As complexvalued kernels also maintain closure under summation and multiplication (Yaglom, 2012), additional modifications are unnecessary. In practice, we can include in our primitive kernels.

* True labels: the original dataset without modification;
* Partially corrupted labels: independently with probability p, the label of each image is corrupted as a uniform random class;
* Random labels: all the labels are replaced with random ones;
* Shuffled pixels: a random permutation of the pixels is chosen and then the same permutation is applied to all the images in both training and test set;
* Random pixels: a different random permutation is applied to each image independently;
* Gaussian: A Gaussian distribution (with matching mean and variance to the original image dataset) is used to generate random pixels for each image.

## **. Regression Benchmarks**

Since the experiments just presented used random training/test splits, they can be thought of as measuring interpolation performance. We are also interested in measuring extrapolation performance, since this is a better measure of whether the model has captured the underlying structure. In results for Mauna and Solar are shown in Figure 1 and Figure 2 in the Appendix. All three models were able to capture the periodic and increasing patterns. However, the heuristic kernel failed to fit the data points well or capture the increasing amplitude, stemming from its lack of PER\*LIN structure.

1. The effective capacity of neural networks is sufficient for memorizing the entire data set;
2. Even optimization on random labels remains easy. In fact, training time increases only by a small constant factor compared with training on the true labels;
3. Randomizing labels is solely a data transformation, leaving all other properties of the learning problem unchanged.

## **1.3. Bayesian Optimization**

Bayesian optimization (Brochu et al., 2010; Snoek et al., 2012) is a technique for optimization expensive black-box functions which repeatedly queries the function, fits a surrogate function to past queries, and maximizes an acquisition function to choose the next query. It’s important to model both the predictive mean (to query points that are likely to perform well) and the predictive variance (to query points that have high uncertainty). Typically, the surrogate functions are estimated using a GP with a simple kernel, such as Matern.

|  |  |
| --- | --- |
|  | (2) |

But simple kernels lead to inefficient exploration due to the curse of dimensionality, leading various researchers to consider additive kernels (Kandasamy et al., 2015; Gardner et al., 2017; Wang et al., 2017). Since additivity is among the patterns the NKN can learn, we were interested in testing its performance on Bayesian optimization tasks with additive structure.

|  |  |
| --- | --- |
|  | (3) |

Let X denote the data matrix whose i-th row is. If X has rank n, then the system of equations has an infinite number of solutions regardless of the right-hand side. We can find a global minimum in the ERM problem (3) by simply solving this linear system. But do all global minima generalize equally well? Is there a way to determine when one global minimum will generalize whereas another will not? One popular way to understand quality of minima is the curvature of the loss function at the solution. But in the linear case, the curvature of all optimal solutions is the same.

# **2 *Experiments***

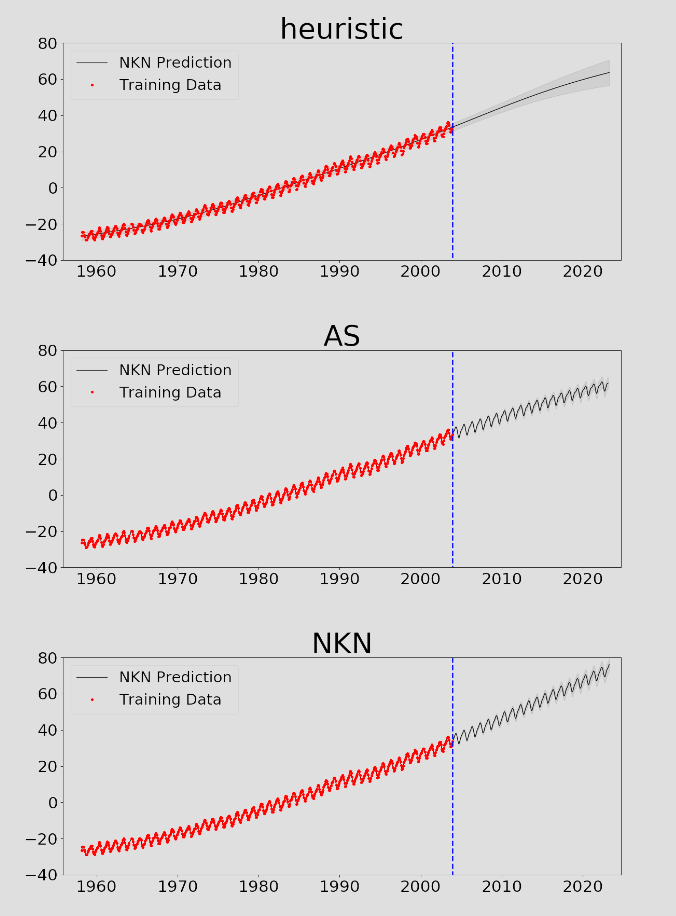


Figure 1 – Extrapolation results of NKN on the Mauna datasets. Heuristic kernel represents linear combination of RBF, PER, LIN, and Constant kernels. AS represents Automatic Statistician (Duvenaud et al., 2013). The red circles are the training points, and the curve after the blue dashed line is the extrapolation result. Shaded areas represent 1 standard deviation.

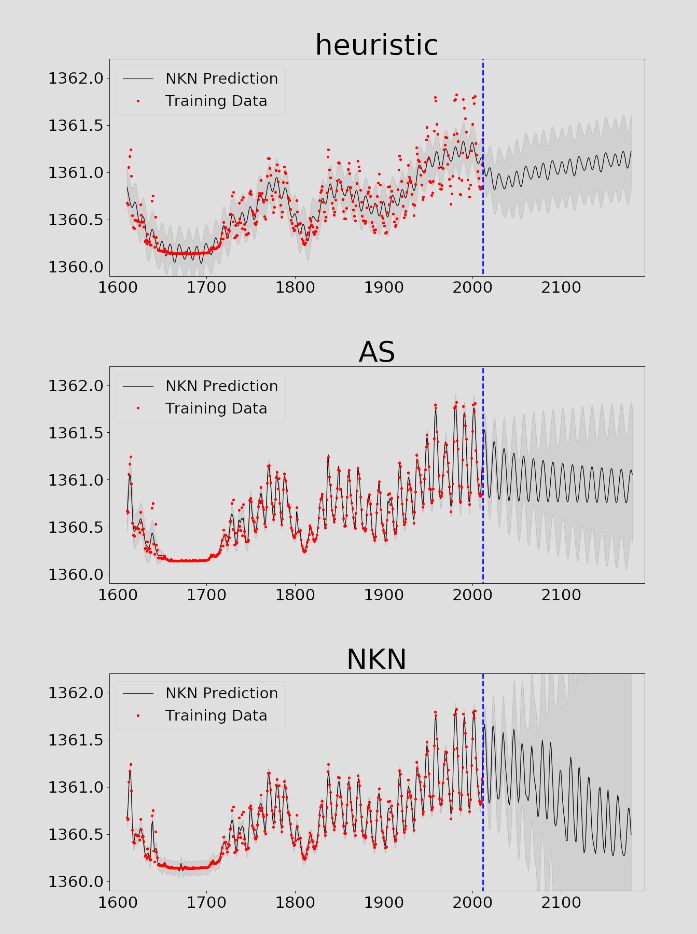


Figure 2 – Extrapolation results of NKN on the Solar datasets. Heuristic kernel represents linear combination of RBF, PER, LIN, and Constant kernels. AS represents Automatic Statistician (Duvenaud et al., 2013). The red circles are the training points, and the curve after the blue dashed line is the extrapolation result. Shaded areas represent 1 standard deviation.

In light of our randomization experiments, we discuss how our findings pose a challenge for several traditional approaches for reasoning about generalization. Rademacher complexity and VC-dimension.

Rademacher complexity is commonly used and flexible complexity measure of a hypothesis class. The empirical Rademacher complexity of a function class on a dataset {z1,…,zn}is defined as

|  |  |
| --- | --- |
|  | (4) |

where σ1, ..., σn ∈ {±1} are i.i.d. uniform random variables.

The resulting test RMSE and log-likelihood are shown in Table 1.

**Table 1**

Test RMSE and log-likelihood for the PCA-split regression benchmarks. N denotes the number of data points.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TEST RMSE | | | | | TEST LOG-LIKELIHOOD | | |
| Dataset | N | GP-RBF | GP-SM | GP\_NKN | GP-SM | GP-SM | GP-NKM |
| Boston | 506 | 6.390 | 8.600 | 4.658 | -4.063 | -4.404 | -3.557 |
| Concrete | 1031 | 8.531 | 7.591 | 6.242 | -3.246 | -3.285 | -3.112 |
| Energy | 768 | 0.974 | 0.447 | 0.459 | -1.297 | -0.564 | -0.649 |
| Kin8nm | 8192 | 0.093 | 0.065 | 0.086 | 0.998 | 1.322 | 1.057 |
| Naval | 11934 | 0.000 | 0.000 | 0.000 | 7.222 | 9.037 | 6.442 |
| Pow. Plant | 9568 | 4.768 | 3.931 | 3.742 | -3.076 | -2.877 | -2.763 |
| Wine | 1599 | 0.701 | 0.660 | 0.650 | -1.076 | -1.002 | -0.972 |
| Yacht | 308 | 1.190 | 1.736 | 0.528 | -2.896 | -2.768 | -0.694 |