Poster Abstract: A toolkit for spatial interpolation and sensor placement

S Deepak Narayanan*
Zeel B Patel*
deepak.narayanan@alumni.iitgn.ac.in
patel_zeel@iitgn.ac.in
IIT Gandhinagar

Apoorv Agnihotri
Nipun Batra
apoorv.agnihotri@alumni.iitgn.ac.in
nipun.batra@iitgn.ac.in
IIT Gandhinagar

ABSTRACT

Sensing is central to the SenSys and related communities. However, fine-grained spatial sensing remains a challenge despite recent advancements, owing to cost, maintenance, among other factors. Thus, estimating the sensed phenomenon at unmonitored locations and strategically installing sensors is of prime importance. In this work, we introduce Polire - an open-source tool that provides a suite of algorithms for spatial interpolation and near-field passive sensor placements. We replicate two existing papers on these two tasks to show the efficacy of Polire. We believe that Polire is an essential step towards lowering entry barriers towards sensing and scientific reproducibility.

CCS CONCEPTS

• **Computing methodologies → Modeling methodologies**.

ACM Reference Format:
S Deepak Narayanan, Zeel B Patel, Apoorv Agnihotri, and Nipun Batra. 2020. Poster Abstract: A toolkit for spatial interpolation and sensor placement. In *The 18th ACM International Conference on Embedded Networked Sensor Systems (SenSys), Nov 16-19, 2020, Virtual Event, Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Sensing is central to the SenSys and related communities to enable a host of applications. Prior work has sensed various environmental phenomena, including, but not limited to, water distribution systems , atmospheric phenomena and soil. Sensing these phenomena has few fundamental challenges, including, but not limited to, missing or faulty measurements, huge deployment cost, energy overhead and optimizing the quality of sensor deployment.

The installation and maintenance of sensors entail a huge cost. Thus, it is common to *estimate* environmental variables at unsensed locations using spatial interpolation techniques. Researchers use multiple spatial interpolation methods while comparing estimates and choose the one having the maximum predictive accuracy. Comparison among various interpolation methods can help benchmark them on widely used datasets and can be used for the baselines for future work. A major hindrance to tackling spatial interpolation problems is lack of good quality open-source implementations of existing spatial interpolation methods. Widely used proprietary tools like ESRI's ArcGIS are expensive and non-trivial to integrate into data collection, processing and prediction pipelines.

Interpolation loses its efficacy when the sensor deployment is sparse. "Intelligent" sensor placement is central to maximize reconstruction accuracy and to account for the non-trivial sensor deployment and maintenance cost. While prior work has looked at various sensor placement/deployment algorithms [2], there is a lack of open-source, generalized implementations. An open-source Python library using mixed-integer programming [3] is available; however, it requires extensive domain expertise.

To lower the entry barrier in this field of research, and to enable reproducibility; we propose an open-source spatial interpolation and sensor placement toolkit: Polire, implemented in Python. Polire is inspired by the impact of similar toolkits in other domains [1]. We implement widely used spatial interpolation algorithms and a few state-of-the-art sensor placement algorithms in Polire. Polire has been i) made available open-source (with inbuilt documentation using Python docstrings) to enable collaborations among users within and across communities; and ii) designed with an abstract and transparent API influenced by scikit-learn [4], a well-documented and consistent Python library.

We replicate two highly influential papers to demonstrate the efficacy of Polire. For interpolation, we replicate a paper by Wong et al. [5], which shows a comparison of several interpolation methods for air quality data estimation of Ozone and PM_{10}. Polire can replicate most results from this paper with just a few lines of code. For sensor placement, we replicate a state-of-the-art sensor placement paper [2]. We use the temperature and precipitation datasets used by Guestrin et al. [2]. We outperform the baselines and replicate the majority of the results successfully. Our code repository is hosted on GitHub here: https://github.com/sustainability-lab/polire/tree/SenSys20_Poster.

2 POLIRE

An important aim of Polire is to provide users with a consistent and easy-to-learn API. Our API is inspired by a popular machine learning package scikit-learn's API [4]. We implement our toolkit in Python, owing to its easy to learn nature, and the enormous number of libraries that can be used for scientific computing.

*Both authors contributed equally to this research.

SenSys'20, November 16-19, 2020, Virtual Event, Japan
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

For spatial interpolation we have implemented kriging, IDW (Inverse Distance Weighted), Spatial Averaging Interpolation, Trend Interpolation, Natural Neighbor Interpolation, Spline Interpolation and Random Interpolation. Additionally, any custom machine learning model from scikit-learn [4] can be used for the interpolation. For near-field passive sensor deployment we have implemented a state-of-the-art near optimal algorithm proposed by Guestrin et al. [2].

We demonstrate a simple use of the interpolation and sensor placement APIs. All data are passed in the NumPy array format. To demonstrate how to use our API, we show two minimal working examples in Listing 1 and Listing 2.

```python
from Polire.interpolate import Kriging
intpr = Kriging()
intpr.fit(X, y)
prediction = intpr.predict(X_unsensed)
grid_prediction = intpr.predict_grid()
# Access parameters of algorithm
intpr.variogram_model
```

Listing 1: A simple usage of Polire.interpolate

```python
from Polire.placement import NottDuns
model = NottDuns(N=10)
model.fit(X, y)
index, loc = model.place(X_unplaced, N=1)
# Access parameters of placement algorithm
param_dict = model.get_all_params()
```

Listing 2: A simple usage of Polire.placement

3 EXPERIMENTS AND EVALUATION

To demonstrate the efficacy of Polire for spatial interpolation, we replicate an impactful paper by Wong et al. [5]. authors interpolate PM_{10} and Ozone (O_3) over 82 different counties in the United States. The dataset is hourly sampled over the year 1990. We had access to the data from 35 counties out of 82. We interpolate at the block group level similar to the authors. Our results for various interpolation techniques compared to the original work are described in Table 1. The unit of measurement is parts per billion (ppb). Replicated graphs and other results are available in our GitHub repository.

Implementation	IDW	Kriging	Spt. Avg.	Near. Nbr.
Wong et al [5]	53.3	49.4	49.5	49.2
Polire (Cal.)	51.1	51.2	48.8	48.5
Polire (S.Cal)	58.4	55.2	55.6	55.1

Table 1: Comparison of spatial interpolation algorithms' mean estimate in California for O_3. Cal. refers to California and S.Cal refers to Southern California. Our interpolation for the whole of California roughly corresponds to the values obtained by the authors. Near. Nbr. is Nearest neighbor and Spt. Avg. is Spatial Averaging. All values are in ppb.

We now discuss our replication of sensor placement by Guestrin et al. [2]. The paper uses the Berkeley lab temperature dataset and precipitation dataset to show the efficacy of the greedy near-optimal sensor placement algorithm. The algorithm selects sensors that maximize a criterion. MI (Mutual Information) and Entropy are the two criteria used by the authors. Our results of RMSE error at k placements shown in Figure 11 and Figure 12 are consistent with the original paper [2]. We also believe that ours is the first publicly available implementation of non-stationary kernels for sensor placement. More results are available in our GitHub repository.

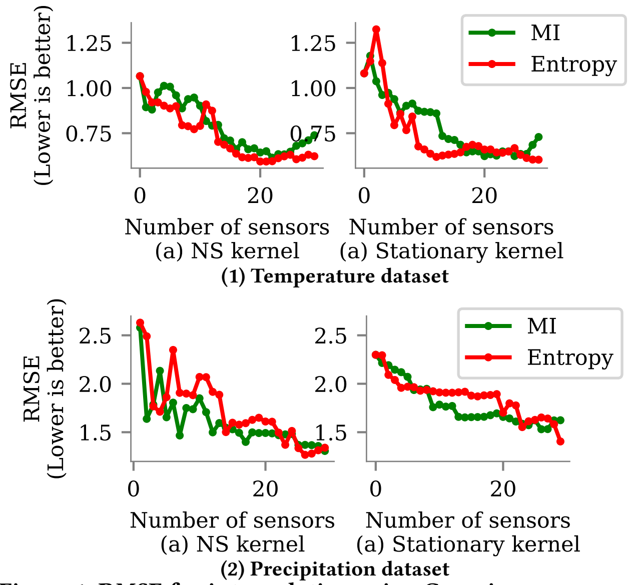

Figure 1: RMSE for interpolation using Gaussian process regression at k placements, $k \in \{1, 2, 3, ..., 30\}$. NS kernel is non-stationary kernel.

4 DISCUSSION AND FUTURE WORK

While Polire code is vectorized, we did not take advantage of hardware accelerators like GPUs. As part of future work, we plan to do an exploratory user study to understand our package's efficacy. We will try to address the difficulties or problems that arise while using our toolkit. We plan to do this by conducting a set of well-designed experiments. We would also like to create a reproducibility challenge, similar to those conducted in the machine learning conferences such as ICLR and NeurIPS. We believe reproducibility challenges are a great way to promote more openness and clarity in research.

REFERENCES

[1] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, and Mani Srivastava. 2014. NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring. In *Fifth International Conference on Future Energy Systems*. ACM Press, Cambridge, UK, 265–276. https://doi.org/10.1145/2602044.2602051 arXiv:1404.3878

[2] Carlos Guestrin, Andreas Krause, and Ajit Paul Singh. 2005. Near-Optimal Sensor Placements in Gaussian Processes. In *Proceedings of the 22nd International Conference on Machine Learning* (Bonn, Germany) *(ICML '05)*. Association for Computing Machinery, New York, NY, USA, 265–272. https://doi.org/10.1145/1102351.1102385

[3] Katherine A. Klise, Bethany L. Nicholson, and Carl Damon Laird. 2017. Sensor Placement Optimization using Chama. *Number SAND2017-11472. Albuquerque, NM: Sandia National Laboratories* (10 2017). https://doi.org/10.2172/1405271

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[5] David W Wong, Lester Yuan, and Susan A Perlin. 2004. Comparison of spatial interpolation methods for the estimation of air quality data. *Journal of Exposure Science & Environmental Epidemiology* 14, 5 (Sept. 2004), 404–415. https://doi.org/10.1038/sj.jea.7500338