Supplementary Material for Large Linear Multi-output Gaussian Process Learning for Time Series

Anonymous Author(s)
Affiliation
Address
email

1 Implementation Details

LLGP was implemented in Python 3 from the Anaconda, which offered an Intel MKL-linked scipy [1]. The code made heavy use of other packages, namely climin [2], GPy [3], and paramz [4]. Code and benchmarks are available at <anonymous repository>.

Application of our approach to all replication studies was carried out on a large server in a multi-programming environment: CentOS 6.5 with 80 Intel(R) Xeon(R) CPU E7-4870 @ 2.40GHz. The representation evaluation benchmarks were done at once on a cluster of machines running CentOS 5.2-5.9 with Intel(R) Xeon(R) CPU E5430 @ 2.66GHz, where these jobs ran on a single thread per CPU.

References

[1] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. URL http://www.scipy.org/. [Online; accessed 2017-02-06].

[2] J. Bayer, C. Osendorfer, S. Diot-Girard, T. Rückstiess, and Sebastian Urban. climin - a pythonic framework for gradient-based function optimization. http://github.com/BRML/climin, 2016.

[3] GPy. GPy: A Gaussian process framework in python. http://github.com/SheffieldML/GPy, since 2012.

[4] Max Zwiessele. paramz. https://github.com/sods/paramz, 2017.

Submitted to 31st Conference on Neural Information Processing Systems (NIPS 2017). Do not distribute.