Adaptive Learn-then-Test: Statistically Valid and Efficient Hyperparameter Selection

Code of "Adaptive Learn-then-Test: Statistically Valid and Efficient Hyperparameter Selection".

Reproducing the experiments

We provide 2 different experimental set-ups `{policy_selection, resource_allocation}`, one in each folder.

- For the `policy_selection` models were trained using the TD3-BC code. We already provide data collected by testing the trained models in the `logs\data\collected_data_aggregate.pkl` file. Alternatively, models can be trained using the generate_models.py script and then tested. To reproduce figure 1, `test.py` script and then `plot_fig_1.py`, similar to evaluate aLTT with different error tolerance levels run `test_delta.py` followed by `plot_fig_2.py`. Finally, the betting strategies can be compared by running `test_betting.py` and then `plot_fig_3.py`.
- For run the `resource_allocation` experiment we use the existing code from Nokia Wireless Suite. Testing data is pre-collected and stored in the `logs\data\collected_data_aggregate.pkl` file. To reproduce the figure on the energy-delay product and queue size trade-off, run the `test_high_priority_class.py` script followed by `plot_energy_delay_product.py`. Similarly, for the energy efficiency vs delay plot, run `test_single_class.py` followed by `plot_delay.py`.

The `collected_data_aggregate.pkl` file for the `resource_allocation` can be downloaded here

Dependencies

To run the code, the following packages are necessary:

- `dr4l` for offline RL in `policy_selection` .
- `gym` for RL algorithms `policy_selection` and `resource_allocation` .
- `matplotlib` to plot the figures.
- `numpy` for array numerical operations.
- `pickle` to store and load results.