USENIX Security '26 Artifact Appendix: Interpolation-Based Optimization for Enforcing ℓ_p-Norm Metric Differential Privacy in Continuous and Fine-Grained Domains

Chenxi Qiu
University of North Texas

A Artifact Appendix

A.1 Abstract

This artifact accompanies the paper "Interpolation-Based Optimization for Enforcing ℓ_p-Norm Metric Differential Privacy in Continuous and Fine-Grained Domains" and provides the complete MATLAB implementation for the proposed *anchor-interpolated privacy optimization (AIPO)* framework.

The package contains end-to-end driver scripts to reproduce the paper's main results (e.g., `main_2norm.m`), core algorithmic routines under `./functions/` (including optimization, log-convex interpolation, and violation evaluation), and implementations of representative baselines. It also includes processed OpenStreetMap road-network datasets (nodes/edges) for Rome, London, and New York City under `./datasets/`, plus cached intermediate files under `./intermediate/`. Running the provided scripts generates metrics such as utility loss, mDP violation ratio, and runtime, and writes structured outputs to subfolders under `./results/`. The artifact is released as a single ZIP archive on Zenodo (DOI: `10.5281/zenodo.17851733`).

A.2 Description & Requirements

The artifact is distributed as a single ZIP archive. After extracting, the top-level directory contains:

- Primary driver scripts to reproduce tables/figures:
  `main_2norm.m`,
  `main_1norm.m`,
  `main_granularity.m`,
  `main_ablation_privacybudget.m`
- `functions/`: implementation of anchor-based optimization, log-convex interpolation, evaluation (utility/violation), and baseline methods.
- `datasets/`: processed road-network datasets (`nodes.csv`, `edges.csv`) for each city benchmark.
- `intermediate/`: cached intermediate files (e.g., precomputed loss matrices) to reduce preprocessing overhead.
- `results/`: output directory populated by the scripts (cost/violation/time/PPR and related logs).

Configuration is centralized in `parameters.m` (e.g., number of repeats). Input data and/or download instructions are under `dataset/`. A paper copy (`paper.pdf`) is included for reference. To run the experiments, start MATLAB in the repository root, modify `parameters.m` if needed, and then execute `main_2norm.m` to reproduce the main results. To reproduce the supplementary (appendix) results, run `main_1norm_appendix.m`, `main_ablation_budget_appendix.m`, and `main_granularity_appendix.m`. This organization allows evaluators to quickly locate the README, source code, datasets, and run scripts without navigating extraneous files.

A.2.1 Security, privacy, and ethical concerns

Executing this artifact poses *low risk* to evaluator machine security and data privacy. The MATLAB scripts are intended to run locally in user space, require no administrator/root privileges, and do not ask evaluators to disable security mechanisms. The code writes outputs only to local folders (i.e., `./results/`) and is not designed to modify system files. The included datasets are road-network graphs (nodes/edges) and do not contain personal or sensitive information, and the artifact does not require uploading data or sending telemetry.

A.2.2 How to access

The artifact is archived on Zenodo. It can be downloaded from:

`https://doi.org/10.5281/zenodo.17851733`

(DOI: `10.5281/zenodo.17851733`). The record provides `AIPO_artifact.zip`, which contains the full MATLAB code, datasets, and scripts needed to reproduce the results.

A.2.3 Hardware dependencies

- *CPU.* A commodity x86_64 machine; a multi-core processor is recommended (4+ cores).

- *Memory. 32 GB RAM* is recommended to run `main_2norm.m`, `main_1norm.m`, `main_granularity.m`, `main_ablation_privacybudget.m` More memory can reduce runtime by avoiding swapping.
- *Accelerators.* Not required (no GPU/FPGA needed).
- *Network.* Not required for normal use.

A.2.4 Software dependencies

- *OS.* Windows 10/11, macOS, or Linux (any recent 64-bit distribution).
- *MATLAB.* MATLAB (R2022b or newer recommended).
- *Toolboxes.* The **Optimization Toolbox** is required for linear programming routines used by some baselines. The **Statistics and Machine Learning Toolbox** is used for random sampling (e.g., `randsample`).
- *Other dependencies.* No external solvers, GPUs, or compilers are required for the provided workflows. Network access is not needed for normal use.

A.2.5 Benchmarks

Included datasets. The artifact bundles all data needed to run the experiments entirely offline. The `datasets/` directory contains preprocessed road-network graphs for: **Rome**, **New York City**, and **London**.

Contents and format. Each dataset provides a road graph within the target region (node/edge lists; `nodes.csv` and `edges.csv`). The provided driver scripts (e.g., `main_2norm.m`) load these files automatically.

Compared methods included. Implementations of the paper's compared mechanisms are included in the artifact alongside AIPO (e.g., exponential-mechanism and LP-based baselines), exposed through the run scripts and the `functions/` directory. No external repositories or additional datasets/models are required.

No external downloads. All datasets are included with the package; internet access is not needed to reproduce results.

Data use and privacy. The datasets are public/aggregated research data and contain no personally identifiable information.

A.3 Set-up

A.3.1 Installation

Step 1: Download the artifact. Download the ZIP archive from the Zenodo DOI https://doi.org/10.5281/zenodo.17851733 and extract it to a local folder.

Step 2: Install dependencies. Install MATLAB (R2022b or newer recommended) with the following toolboxes: (i) **Optimization Toolbox** (required for LP-based baselines), and (ii) **Statistics and Machine Learning Toolbox** (used for `randsample`).

A.3.2 Basic Test

This test verifies that (i) MATLAB can locate and execute the artifact code, (ii) datasets can be loaded, (iii) the core optimization/interpolation pipeline runs end-to-end, and (iv) result files are generated.

Prerequisites. MATLAB is launched in the artifact root directory and the code is on the path: `addpath(genpath(pwd));`

Command. Run the MATLAB file `main_2norm.m`. No input parameters are required (the script uses the default settings shipped with the artifact).

Expected successful output. The test is successful if:

- MATLAB finishes without errors, and
- the `results/2norm/` directory is updated with output files produced by the run.

Troubleshooting. If MATLAB reports missing functions (e.g., `randsample` or LP solvers), verify that the required toolboxes are installed and that `addpath(genpath(pwd))` was executed in the artifact root directory.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): *AIPO empirically satisfies* (ε, d_p)*-mDP on the evaluated city benchmarks, while discretization-based baselines (e.g., LP and COPT) may exhibit nonzero empirical violations.* Supported by (E1) / Table 1.

(C2): *Under the same privacy budgets, AIPO improves utility loss compared to predefined-noise mechanisms (Laplace, EM, TEM) and hybrid methods (COPT, RMP).* Supported by (E2) / Table 2.

(C3): *AIPO is more time-efficient than optimization-based baselines (LP, COPT).* Supported by (E3) / Table 3.

A.4.2 Experiments

All three experiments (E1)–(E3) are executed together by running the MATLAB script `main_2norm.m`. With the default setting (`NT_TEST`=1), each dataset run typically takes about 5–6 hours, so running all three datasets takes about 15–18

hours (runtime may vary by machine). Increasing `NT_TEST` scales the runtime approximately linearly.

Preparation: Extract the artifact ZIP, start MATLAB in the artifact root, and run:

```
addpath(genpath(pwd));
```

To set the number of repeats, set value of "`NT_TEST`" in `parameters.m`.

Execution: Run:

```
main_2norm.m
```

(E1): *(Supports C1) mDP violation evaluation.* MATLAB prints a table titled: "`[city] road map - Violation ratio`", as Fig. 1 shows:

```
================================================
rome road map - Violation ratio
================================================
Method        e=0.2   e=0.4   e=0.6   e=0.8   e=1.0   e=1.2   e=1.4   e=1.6
------------------------------------------------------------------------------
Pre-defined Noise Distribution:
EM          0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000
Laplace     0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000
TEM         0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000

Hybrid Method:
RMP         0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000
COPT        1.978±0.000  8.097±0.000  6.294±0.000  5.998±0.000  5.723±0.000  5.476±0.000  5.165±0.000  4.929±0.000

LP          2.076±0.000  2.108±0.000  1.817±0.000  1.977±0.000  1.652±0.000  1.362±0.000  1.017±0.000  0.834±0.000
AIPO-R      9.285±0.000  7.582±0.000  5.012±0.000  3.314±0.000  2.099±0.000  1.250±0.000  0.898±0.000  0.573±0.000
AIPO*       0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000  0.000±0.000
```

Figure 1: Printed table for mDP violation ratio (Rome roadmap).

The results are also saved to:

```
results/violation/city/violation_method.mat
```

Verify that AIPO and predefined-noise mechanisms (EM, Laplace, TEM) to achieve 0% empirical violations, while relaxed/approximate baselines (COPT, LP, AIPO-R) may yield nonzero violation ratios, matching the results in Table 1.

(E2): *(Supports C2) Utility-loss comparison.* MATLAB prints a table titled: "`[city] road map - Utility loss`", as Fig. 2 shows:

```
================================================
rome road map - Utility loss
================================================
Method        e=0.2   e=0.4   e=0.6   e=0.8   e=1.0   e=1.2   e=1.4   e=1.6
------------------------------------------------------------------------------
Pre-defined Noise Distribution:
EM          8.54±0.00   8.51±0.00   8.50±0.00   8.49±0.00   8.49±0.00   8.49±0.00   8.49±0.00   8.48±0.00
Laplace     8.54±0.00   8.51±0.00   8.50±0.00   8.49±0.00   8.49±0.00   8.49±0.00   8.49±0.00   8.48±0.00
TEM         8.59±0.00   8.59±0.00   8.59±0.00   8.59±0.00   8.59±0.00   8.59±0.00   8.59±0.00   8.59±0.00

Hybrid Method:
RMP         5.86±0.00   5.19±0.00   5.00±0.00   4.92±0.00   4.88±0.00   4.85±0.00   4.83±0.00   4.82±0.00
COPT        7.71±0.00   8.51±0.00   8.48±0.00   8.47±0.00   8.46±0.00   8.44±0.00   8.41±0.00   8.37±0.00

LP          4.89±0.00   3.18±0.00   2.56±0.00   2.26±0.00   2.11±0.00   2.03±0.00   1.99±0.00   1.97±0.00
AIPO-R      3.93±0.00   2.99±0.00   2.62±0.00   2.46±0.00   2.36±0.00   2.32±0.00   2.29±0.00   2.27±0.00
LB          2.32±0.00   1.78±0.00   1.74±0.00   1.73±0.00   1.73±0.00   1.73±0.00   1.73±0.00   1.73±0.00
AIPO*       5.64±0.00   4.56±0.00   3.90±0.00   3.47±0.00   3.18±0.00   2.96±0.00   2.82±0.00   2.71±0.00
```

Figure 2: Printed table for utility loss (Rome roadmap).

The results are also saved to:

```
results/cost/city/loss_method.mat
```

Verify that the relative ordering among methods matches Table 2: AIPO achieves lower utility loss than predefined-noise mechanisms (Laplace, EM, TEM) and hybrid methods (COPT, RMP), while incurring higher utility loss than the lower bound (LB) and relaxed/discretization-based methods (AIPO-R, LP).

(E3): *(Supports C3) Computation time evaluation.* MATLAB prints a table titled: "`[city] road map - Computation time (s)`", as Fig. 3 shows:

```
================================================
rome road map - Computation time (s)
================================================
Method        e=0.2   e=0.4   e=0.6   e=0.8   e=1.0   e=1.2   e=1.4   e=1.6
------------------------------------------------------------------------------
COPT        145.972±0.000  152.042±0.000  143.868±0.000  143.131±0.000  143.390±0.000  141.733±0.000  143.319±0.000  142.019±0.000
LP          237.313±0.000   46.383±0.000  262.018±0.000  118.086±0.000   89.109±0.000   65.737±0.000   82.725±0.000  120.175±0.000
AIPO*        15.454±0.000   15.711±0.000   14.776±0.000   13.347±0.000   12.795±0.000   12.496±0.000   12.339±0.000   12.511±0.000
```

Figure 3: Printed table for computation time (Rome roadmap).

The results are also saved to:

```
results/time/city/time_method.mat
```

Verify that AIPO is faster than LP and COPT as in Table 3 (absolute runtimes may vary by machine, but the ordering should be consistent).

A.5 Notes on Reusability

Scaling experiments up/down. The provided run scripts execute predefined parameter sweeps. To *scale down* for quick testing, reduce the number of privacy budgets (e.g., fewer ε values), restrict to a single dataset (e.g., Rome), or disable expensive LP-based baselines. To *scale up*, extend the parameter lists or enable additional baselines.

Running on a new road network. To evaluate on a new region, add a new folder under `datasets/` with `nodes.csv` and `edges.csv` in the same format as the included benchmarks.

Changing the metric or utility model. The code separates dataset loading, metric construction, and utility/loss computation. Users can replace or modify the distance computation (e.g., switch from Euclidean distance to Manhattan distance, or plug in a custom metric) and update the loss-matrix generation accordingly. Cached inputs under `intermediate/` can be deleted to force regeneration when changing these components.

Extending baselines. Baseline mechanisms are organized under `functions/` (including a `benchmarks/` subdirectory). New mechanisms can be added by implementing a perturbation-generator function with the same input/output interface as existing baselines and registering it in the run scripts.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2026/.