Reproduction Package README

Overview

This reproducibility package contains the data inputs, simulation code, precomputed simulation outputs, and Jupyter notebooks needed to reproduce the paper's tables and figures. The analysis uses anonymized mediation case records from the Kenyan Judiciary's Court Annexed Mediation (CAM) system, including case arrivals, mediator assignments, case types, court stations, conclusion dates, mediation outcomes, mediator accreditations, and station preferences.

The package supports two replication workflows:

1. Precomputed-results workflow: use the supplied camera-ready JSON outputs to regenerate tables and figures without rerunning the simulations. This is the recommended workflow for reviewers because it avoids long simulation runtimes and does not require Slurm or Gurobi for plotting.
2. Simulation-regeneration workflow: rerun the simulation cells or scripts to regenerate the raw JSON outputs, then run the plotting cells to recreate the tables and figures. This workflow is computationally heavier, requires more storage, and may benefit from Gurobi and/or a Slurm-based cluster.

The main notebooks are:

```
reproduce_exp_one_click_QP.ipynb
reproduce_exp_one_click_LP_refinedC5.ipynb
sensitivity_analysis_of_VA_esitmation.ipynb
```

For the recommended precomputed-results workflow, reviewers should leave the top-level notebook variable `one_click_path_A = True` and select Run All. With this setting, simulation-regeneration cells are skipped automatically, while the cells that load precomputed outputs and generate plots still run.

Generated exhibits are saved under:

```
plotsReproduceRun/
```

Data Availability Statement

The analysis uses administrative data from the Kenyan Judiciary's Court Annexed Mediation system.

```
File name: vw_deidentified_case_w_appointer_20250729.csv
Access/download date: July 29, 2025
Citation: Kenyan Judiciary, Court Annexed Mediation Administrative Data, 2016-2025.

File name: vw_all_mediators_20250729.csv
Access/download date: July 29, 2025
Citation: Kenyan Judiciary, Court Annexed Mediation Administrative Data, 2016-2025.

File name: cases_raw_07012025.csv
Access/download date: July 1, 2025
Citation: Kenyan Judiciary, Court Annexed Mediation Administrative Data, 2016-2025.

File name: data_casetype_20250729.csv
Access/download date: July 29, 2025
Citation: Kenyan Judiciary, Court Annexed Mediation Administrative Data, 2016-2025.

File name: data_accreditationcategory_20250729.csv
Access/download date: July 29, 2025
Citation: Kenyan Judiciary, Court Annexed Mediation Administrative Data, 2016-2025.

File name: data_casetype_eligible_accreditation_categories_20250729.csv
Access/download date: July 29, 2025
Citation: Kenyan Judiciary, Court Annexed Mediation Administrative Data, 2016-2025.
```

The dataset contains anonymized records of 30,633 mediation cases from 2016-2025. It includes case arrivals, mediator assignments, case types across 13 categories, court stations across 87 locations, case conclusion dates, mediation outcomes, mediator accreditations, and station preferences for approximately 2,100 mediators. Personally identifying information about mediators and parties has been removed.

The data are not publicly available due to confidentiality agreements and privacy considerations. Researchers wishing to access the anonymized dataset for replication purposes should contact the Judiciary of Kenya through the Court Annexed Mediation Secretariat.

Statement Of Rights

Instructions For Replicators

1. Unzip The Package

Unzip `Reproduction Package.zip` to a local folder. This README refers to that folder as:

```
<PACKAGE_DIR>
```

For example, on Windows:

```
C:\Users\<you>\Downloads\Reproduction Package
```

Open a terminal in `<PACKAGE_DIR>`. All commands below assume the current working directory is the package root.

On Windows PowerShell:

```
cd "<PACKAGE_DIR>"
```

2. Install Requirements

Create or activate a Python environment, then install the requirements:

```
pip install -r requirements.txt
```

The notebooks include configuration variables:

```
one_click_path_A = True
run_slurm = False
use_gurobi = True
```

For the precomputed-results workflow, reviewers only need to keep `one_click_path_A = True` and run all cells after the camera-ready result folders are extracted. The simulation-regeneration cells check this variable and skip themselves when it is true; the plotting cells still run and generate the exhibits from the precomputed JSON files.

For generating plots from the camera-ready result files, Slurm and Gurobi are not required because the notebooks only read precomputed JSON files. Gurobi is relevant only if raw simulation outputs are regenerated with Gurobi enabled.

Workflow A: Generate Exhibits From Precomputed Results

This is the recommended workflow for replication review. Essentially, the authors of the paper ran simulations for the following scenarios:

Scenario 1 Scenario 2 Real data with no learning Real data with learning from scratch Real data with learning with initial beliefs computed from data

We ran simulations with both a Quadratic Program and a Linear Program. For every scenario, 128 random instantiations of mediator VAs, case arrivals, etc. are simulated and the results are finally averaged. QP_camera_ready.zip and LP_camera_ready.zip contain the intermediate raw outputs from these simulations. Every case arrival, mediator VA, assignment to mediator, etc. is recorded in detail in these intermediate files. Workflow A simply uses these precomputed simulation files and computes the final tables and plots.

The Quadratic Program based tables and plots are all generated using the reproduce_exp_one_click_QP.ipynb notebook. The inputs and outputs in this case are:

Output	Cell	Input Folders	Date Generated	Saved Outputs
Figure 1	7	Generated directly in notebook		`plotsReproduceRun/Figure`

Table 2	11	`QP_camera_ready/Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.01`
		`QP_camera_ready/Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.05`
		`QP_camera_ready/Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.1`
		`QP_camera_ready/Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.5`
			6/9/2026 12:41:54 AM
			6/9/2026 12:54:55 AM
			6/9/2026 1:07:54 AM
			6/9/2026 1:20:54 AM
				`plotsReproduceRun/Scenar:`
				`plotsReproduceRun/Scenar:`
				`plotsReproduceRun/Scenar:`

Output	Cell	Input Folders	Date Generated	Saved Outputs
Figure 3 top/middle/bottom	11	Same as Table 2	Same as Table 2	`plotsReproduceRun/Scenar:` `plotsReproduceRun/Scenar:` `plotsReproduceRun/Scenar:`
Figure 4 left, Scenario 1	11	Same as Table 2	Same as Table 2	`plotsReproduceRun/Scenar:`
Table 3	16	`QP_camera_ready/Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.01` `QP_camera_ready/Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.05` `QP_camera_ready/Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.1` `QP_camera_ready/Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.5`	6/9/2026 1:37:57 AM 6/9/2026 1:50:58 AM 6/9/2026 2:03:59 AM 6/9/2026 2:16:58 AM	`plotsReproduceRun/Scenar:` `plotsReproduceRun/Scenar:` `plotsReproduceRun/Scenar:`
Figure 4 right, Scenario 2	16	Same as Table 3	Same as Table 3	`plotsReproduceRun/Scenar:`
Table 4	20	`QP_camera_ready/FullDataGroundReproduce_No_C5_Cap3_lambda_0.01` `QP_camera_ready/FullDataGroundReproduce_No_C5_Cap3_lambda_0.02` `QP_camera_ready/FullDataGroundReproduce_No_C5_Cap3_lambda_0.03` `QP_camera_ready/FullDataGroundReproduce_No_C5_Cap3_lambda_0.04` `QP_camera_ready/FullDataGroundReproduce_No_C5_Cap3_lambda_0.05` `QP_camera_ready/FullDataGroundReproduce_No_C5_Cap3_lambda_0.1` `QP_camera_ready/FullDataGroundReproduce_No_C5_Cap3_lambda_0.5`	6/9/2026 2:42:17 AM 6/9/2026 4:18:30 AM 6/9/2026 5:55:42 AM 6/9/2026 7:37:42 AM 6/9/2026 9:00:00 AM 6/9/2026 10:27:14 AM 6/9/2026 12:04:23 PM	`plotsReproduceRun/FullDa†`
Table 5, initial belief `N(0, sigma^2)`	23	`QP_camera_ready/FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.01` `QP_camera_ready/FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.05` `QP_camera_ready/FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.1` `QP_camera_ready/FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.5`	6/9/2026 3:42:09 AM 6/9/2026 5:23:45 AM 6/9/2026 7:04:19 AM 6/9/2026 8:44:18 AM	`plotsReproduceRun/FullDa†` `5 (Initial Belief set to`
Table 5, initial belief from data	28	`QP_camera_ready/FullDataLearningBootstrap_No_C5_Cap3_lambda_0.01` `QP_camera_ready/FullDataLearningBootstrap_No_C5_Cap3_lambda_0.05` `QP_camera_ready/FullDataLearningBootstrap_No_C5_Cap3_lambda_0.1` `QP_camera_ready/FullDataLearningBootstrap_No_C5_Cap3_lambda_0.5`	6/9/2026 2:39:42 PM 6/9/2026 4:46:08 PM 6/9/2026 6:51:09 PM 6/9/2026 8:56:59 PM	`plotsReproduceRun/FullDa†` `(Initial belief calculat`

Each of these folders will contain JSON files with an algorithm mnemonic and instance id (ranging from 0-127). Each contains detailed simulation records for one algorithm and one random instantiation. The mnemonic algorithm identifiers are:

- `best_med_upperlimit`: upper-bound benchmark.
- `greedy_grnd` or `greedy_learning`: greedy variants.
- `TH`: Thompson sampling.
- `myp_sample`: SMaRT sample-from-belief variant.
- `myp_avg`: SMaRT mean-of-belief variant.
- `myp_grnd`: SMaRT ground-truth value-added variant.

Similarly, the Linear Program based tables and plots are all generated using the reproduce_exp_one_click_LP_refinedC5.ipynb notebook. The inputs and outputs in this case are:

Output	Cell	Input Folders	Date Generated	Saved Outputs

Table 6	5	`LP_camera_ready/Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.01`
		`LP_camera_ready/Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.05`
		`LP_camera_ready/Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.1`
		`LP_camera_ready/Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.5`
			6/18/2026 2:13:33 PM
			6/18/2026 2:28:25 PM
			6/18/2026 2:41:51 PM
			6/18/2026 2:55:04 PM
				`plotsReproduceRun/Scenario1/Table`
				`plotsReproduceRun/Scenario1/Table`
				`plotsReproduceRun/Scenario1/Table`

Table 7	7	`LP_camera_ready/Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.01`
		`LP_camera_ready/Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.05`
		`LP_camera_ready/Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.1`
		`LP_camera_ready/Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.5`
			6/18/2026 3:42:35 PM
			6/18/2026 3:56:37 PM
			6/18/2026 4:11:12 PM
			6/18/2026 4:24:32 PM
				`plotsReproduceRun/Scenario2/Table`
				`plotsReproduceRun/Scenario2/Table`
				`plotsReproduceRun/Scenario2/Table`

Table 8	9	`LP_camera_ready/FullDataGroundLP_modified_C5_Cap3_lambda_0.01`
		`LP_camera_ready/FullDataGroundLP_modified_C5_Cap3_lambda_0.02`
		`LP_camera_ready/FullDataGroundLP_modified_C5_Cap3_lambda_0.03`
		`LP_camera_ready/FullDataGroundLP_modified_C5_Cap3_lambda_0.04`
		`LP_camera_ready/FullDataGroundLP_modified_C5_Cap3_lambda_0.05`
		`LP_camera_ready/FullDataGroundLP_modified_C5_Cap3_lambda_0.1`
		`LP_camera_ready/FullDataGroundLP_modified_C5_Cap3_lambda_0.5`
			6/16/2026 2:00:06 AM
			6/16/2026 5:10:14 AM
			6/16/2026 8:14:47 AM
			6/16/2026 11:34:10 AM
			6/16/2026 2:57:00 PM
			6/16/2026 6:03:37 PM
			6/16/2026 9:29:27 PM
				`plotsReproduceRun/FullDataGroundRe`

Table 9, initial belief `N(0, sigma^2)`	11	`LP_camera_ready/FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.01`
		`LP_camera_ready/FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.05`
		`LP_camera_ready/FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.1`
		`LP_camera_ready/FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.5`
			6/16/2026 4:24:25 PM
			6/16/2026 8:16:52 PM
			6/17/2026 12:01:19 AM
			6/17/2026 3:48:25 AM
				`plotsReproduceRun/FullDataLearning`
				`9 (Initial Belief set to N(0,sigma`

Table 9, initial belief from data	13	`LP_camera_ready/FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.01`
		`LP_camera_ready/FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.05`
		`LP_camera_ready/FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.1`
		`LP_camera_ready/FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.5`
			6/17/2026 1:30:31 AM
			6/17/2026 8:36:13 AM
			6/17/2026 3:54:10 PM
			6/17/2026 11:01:07 PM
				`plotsReproduceRun/FullDataLearning`
				`(Initial belief calculated from da`

Place the separately supplied camera-ready archives in `<PACKAGE_DIR>`:

```
QP_camera_ready.zip
LP_camera_ready.zip
```

Extract the two top-level result archives:

```
Expand-Archive -LiteralPath QP_camera_ready.zip -DestinationPath QP_camera_ready -Force
Expand-Archive -LiteralPath LP_camera_ready.zip -DestinationPath LP_camera_ready -Force
```

Each extracted folder contains nested zip files. Extract those nested zip files in place:

```
$zips = Get-ChildItem -LiteralPath QP_camera_ready -Recurse -Filter *.zip -File
foreach ($zip in $zips) {
    $dest = Join-Path $zip.DirectoryName $zip.BaseName
    Expand-Archive -LiteralPath $zip.FullName -DestinationPath $dest -Force
}

$zips = Get-ChildItem -LiteralPath LP_camera_ready -Recurse -Filter *.zip -File
foreach ($zip in $zips) {
    $dest = Join-Path $zip.DirectoryName $zip.BaseName
    Expand-Archive -LiteralPath $zip.FullName -DestinationPath $dest -Force
}
```

After extraction, the notebooks read precomputed JSON files from:

```
<PACKAGE_DIR>\QP_camera_ready\...
<PACKAGE_DIR>\LP_camera_ready\...
```

Size of the directories

QP

Folder	Size GB	Last Updated
`Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.01`	0.106	2026-06-09 00:41:54
`Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.05`	0.106	2026-06-09 00:54:55
`Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.1`	0.107	2026-06-09 01:07:54
`Scenario1ReproduceQP_NOC5_1_Cap3_lambda_0.5`	0.107	2026-06-09 01:20:55
`Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.01`	0.107	2026-06-09 01:37:58
`Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.05`	0.107	2026-06-09 01:50:58
`Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.1`	0.107	2026-06-09 02:03:59
`Scenario2ReproduceQP_NOC5_1_Cap3_lambda_0.5`	0.107	2026-06-09 02:16:58
`FullDataGroundReproduce_No_C5_Cap3_lambda_0.01`	11.032	2026-06-09 03:58:22
`FullDataGroundReproduce_No_C5_Cap3_lambda_0.02`	11.032	2026-06-09 05:34:23
`FullDataGroundReproduce_No_C5_Cap3_lambda_0.03`	11.033	2026-06-09 07:16:41
`FullDataGroundReproduce_No_C5_Cap3_lambda_0.04`	11.032	2026-06-09 08:50:33
`FullDataGroundReproduce_No_C5_Cap3_lambda_0.05`	11.033	2026-06-09 10:16:45
`FullDataGroundReproduce_No_C5_Cap3_lambda_0.1`	11.032	2026-06-09 11:54:32
`FullDataGroundReproduce_No_C5_Cap3_lambda_0.5`	11.032	2026-06-09 13:30:46
`FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.01`	11.479	2026-06-09 04:24:26
`FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.05`	11.531	2026-06-09 06:05:40
`FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.1`	11.541	2026-06-09 07:45:57
`FullDataLearningFromScratchReproduce_NOC5_Cap3_lambda_0.5`	11.553	2026-06-09 09:26:25
`FullDataLearningBootstrap_No_C5_Cap3_lambda_0.01`	12.697	2026-06-09 15:29:22
`FullDataLearningBootstrap_No_C5_Cap3_lambda_0.05`	12.699	2026-06-09 17:34:01

Folder	Size GB	Last Updated
`FullDataLearningBootstrap_No_C5_Cap3_lambda_0.1`	12.701	2026-06-09 19:39:21
`FullDataLearningBootstrap_No_C5_Cap3_lambda_0.5`	12.699	2026-06-09 21:44:57
QP total	**174.983**	

LP

Folder	Size GB	Last Updated
`Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.01`	0.040	2026-06-18 14:13:34
`Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.05`	0.041	2026-06-18 14:28:26
`Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.1`	0.041	2026-06-18 14:41:51
`Scenario1ReproduceLP_modified_C5_Cap3_lambda_0.5`	0.041	2026-06-18 14:55:05
`Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.01`	0.040	2026-06-18 15:42:35
`Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.05`	0.041	2026-06-18 15:56:37
`Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.1`	0.041	2026-06-18 16:11:13
`Scenario2ReproduceLP_modified_C5_Cap3_lambda_0.5`	0.041	2026-06-18 16:24:33
`FullDataGroundLP_modified_C5_Cap3_lambda_0.01`	2.617	2026-06-16 04:56:02
`FullDataGroundLP_modified_C5_Cap3_lambda_0.02`	2.617	2026-06-16 08:12:55
`FullDataGroundLP_modified_C5_Cap3_lambda_0.03`	2.617	2026-06-16 11:19:18
`FullDataGroundLP_modified_C5_Cap3_lambda_0.04`	2.617	2026-06-16 14:52:31
`FullDataGroundLP_modified_C5_Cap3_lambda_0.05`	2.617	2026-06-16 18:02:00
`FullDataGroundLP_modified_C5_Cap3_lambda_0.1`	2.617	2026-06-16 21:15:15
`FullDataGroundLP_modified_C5_Cap3_lambda_0.5`	2.617	2026-06-17 00:34:19
`FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.01`	5.734	2026-06-16 18:15:42
`FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.05`	5.797	2026-06-16 22:02:17
`FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.1`	5.804	2026-06-17 01:50:50
`FullDataLearningScratchLP_modified_C5_Cap3_lambda_0.5`	5.805	2026-06-17 05:33:07
`FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.01`	6.379	2026-06-17 04:52:11
`FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.05`	6.378	2026-06-17 12:13:43
`FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.1`	6.377	2026-06-17 19:26:45
`FullDataLearningBootstrapLP_modified_C5_Cap3_lambda_0.5`	6.376	2026-06-18 02:29:35
LP total	**67.293**	

Grand Total

Category	Size GB
QP total	174.983
LP total	67.293
Combined	242.276

Some extracted folders may contain an extra same-name nested folder. The plotting helper `makeplots_reproduction_package.py` handles this layout.

Next, run the following notebooks:

List Of Exhibits

Cell indices below are zero-based notebook cell indices from the `.ipynb` files. Jupyter execution counters such as `In [12]` may differ depending on execution history, so use the source code snippets, folder paths, table names, and figure names as the definitive identifiers.

QP Notebook

Notebook:

```
reproduce_exp_one_click_QP.ipynb
```

First run the top import/setup cells, including the cell that imports:

```
from makeplots_reproduction_package import retrieve_data_and_make_plots
```

Then run the plotting cells below.

The QP exhibit table in Workflow A lists the plotting cell indices and required input folders.

The corresponding raw-data simulation cells are:

- Cell 11 for Table 2, Figure 3 top/middle/bottom, and Figure 4 left.
- Cell 16 for Table 3 and Figure 4 right.
- Cell 20 for Table 4.
- Cell 23 for Table 5, `N(0, sigma^2)` initial belief.
- Cell 28 for Table 5, data-based initial belief.

These simulation cells are not needed if the camera-ready folders were extracted successfully.

LP Notebook

Notebook:

```
reproduce_exp_one_click_LP_refinedC5.ipynb
```

First run the top import/setup cells, including the cell that imports:

```
from makeplots_reproduction_package import retrieve_data_and_make_plots
```

Then run the plotting cells below.

The LP exhibit table in Workflow A lists the plotting cell indices and required input folders.

The corresponding raw-data simulation cells are:

- Cell 5 for Table 6.
- Cell 7 for Table 7.
- Cell 9 for Table 8.
- Cell 11 for Table 9, `N(0, sigma^2)` initial belief.
- Cell 13 for Table 9, data-based initial belief.

These simulation cells are not needed if the camera-ready folders were extracted successfully.

Sensitivity Analysis Notebook

Notebook:

```
sensitivity_analysis_of_VA_esitmation.ipynb
```

This notebook generates the sensitivity-analysis plot, Figure 5. Run the notebook cells sequentially from top to bottom. The final plotting cell is cell 47.

Cell 47 displays the sensitivity-analysis figure inline and saves it to:

```
<PACKAGE_DIR>\plotsReproduceRun\Figure 5.png
```

To regenerate tables and figures from precomputed results, leave `one_click_path_A = True` and run all cells in each notebook. This is the simplest reviewer workflow: the simulation/script cells are skipped automatically, and the plotting cells load the precomputed data and write the exhibits.

If running cells manually instead, run only:

1. The import/setup cells at the top of each notebook.
2. The `retrieve_data_and_make_plots(...)` plotting cells listed in the exhibit tables above.

Generated plot and table images are saved relative to `<PACKAGE_DIR>` under:

```
<PACKAGE_DIR>\plotsReproduceRun\<result_save_dir>\
```

If `plotsReproduceRun/` does not exist, it is created automatically.

Workflow B: Regenerate Raw Simulation Outputs

The package also includes simulation cells that can regenerate raw JSON outputs. These cells are much slower than plotting from the camera-ready outputs.

Running all cells of the 3 notebooks (reproduce_exp_one_click_LP_refinedC5.ipynb,sensitivity_analysis_of_VA_esitmation.ipynb,reproduce_exp_one_click_QP.ipynb) with:

```
one_click_path_A = False
run_slurm = False
use_gurobi = True
```

will initiate workflow B

The notebooks call `run_simulation(...)` sequentially on the local machine. If `use_gurobi = False`, PuLP's default CBC solver is used instead. CBC is open source but can be slower than Gurobi. Please note using CBC instead of Gurobi can lead to small differences in outputs being generated.

When the authors ran this, they used:

```
one_click_path_A = False
run_slurm = True
```

The notebooks call the included `.sh` scripts. These scripts were written for a Slurm cluster and may require local edits for a different HPC environment.

The core simulation entry point is `run_simulation_toy_problem.py`. One local example is:

```
python run_simulation_toy_problem.py --run_id 0 --test_case_file_path scenario2.py --use_gurobi --antoineVA
```

Important arguments include:

- `run_id`: instance id for parallel processing; also used as a random seed.
- `lp_lmda`: slack penalty coefficient in the LP.
- `use_full_data`: run the simulation on the full dataset.
- `use_gurobi`: use Gurobi instead of PuLP CBC.
- `start_date`, `end_date`: simulation date range, in `YYYY-MM-DD` format.
- `same_duration`: use pre-sampled fixed case duration conditional on outcome.
- `antoineVA`: use Antoine's value-added calculation method.
- `test_case_file_path`: path to a test case script.
- `output_folder_path`: output directory for raw JSON simulation files.

Each simulation-regeneration cell is followed by a plotting cell that calls `retrieve_data_and_make_plots(...)`. That function reads the generated result folder and creates the corresponding tables and figures.

Runtime Expectations

The approximate runtime for each simulation (to generate one json file) is around 5 hours depending on the machine being used. We ran 128 simulations in parallel in slurm HPC which resulted in 5 hours to complete simulations for each setting. However, running them in sequence (when parallel compute in unavailable) could take upto 5*128 = 640 hours. More importantly, the CBC/PuLP solvers won't generate identical results as Gurobi does as these solvers dont converge on the same solution exactly. We merely included CBC/PuLP as a fallback so that one can run this code to get similar results without Gurobi.

Software Requirements

Install Python dependencies from:

```
requirements.txt
```

The listed Python packages include:

```
gurobipy
joblib
linearmodels
matplotlib
numpy
pandas
properscoring
pulp
scipy
statsmodels
tqdm
```

Notes:

- Gurobi is proprietary and requires a license. Free academic licenses may be available from Gurobi.
- If `use_gurobi` is not enabled, the simulations use PuLP CBC by default. This can lead to small deviations in results.
- Slurm is optional and only relevant if `run_slurm = True`.
- The precomputed-results workflow does not require Slurm or Gurobi for plotting.

Technology Environment

The package is intended to run in a Python/Jupyter environment from the package root. The notebooks and scripts were developed for Windows paths and Slurm-based HPC execution, but the plotting workflow should run locally after the camera-ready result folders are extracted.

Recommended local setup:

```
Operating system: Windows, macOS, or Linux
Python: Python 3 environment compatible with requirements.txt
Interface: Jupyter Notebook or JupyterLab
Solver for plotting workflow: not required
Solver for simulation workflow: Gurobi optional; PuLP CBC fallback available
HPC scheduler: Slurm optional; only needed for cluster execution
```

If running on a new machine, confirm that `config.py` points to the local `param_files/` folder.

Code Description

Core files:

- `Court_case.py`: defines the court case class.
- `Simulation.py`: implements the main assignment algorithms. The `simulate` method implements Thompson sampling and greedy algorithms; `SimulateMiopicFuture` implements the main proposed algorithm.
- `SlakedLPwithLoad.py`: defines the `slakedLP` class and implements the QP-PRIMAL-LinearApprox linear program using PuLP, with optional Gurobi support.
- `SlakedLP.py`, `SlakedQPwithLoad.py`, `slacked_LP_product.py`: related optimization formulations used directly or indirectly by the simulation code.
- `Simulation_product.py`: supporting simulation code used by imported modules.
- `VA_Antoine.py`: implements Antoine's value-added computation for mediators.
- `run_simulation_toy_problem.py`: simulation entry point used by the notebooks for local regeneration.
- `simulation_utils.py`: utilities for loading data and calculating statistics before simulation.
- `makeplots_reproduction_package.py`: reads simulation JSON outputs and generates paper tables and figures.
- `config.py`: stores local configuration, including the `param_files/` path.

The simulation output files use mnemonic algorithm identifiers. Important mappings include:

- `best_med_upperlimit`: upper-bound benchmark.
- `greedy_grnd` or `greedy_learning`: greedy variants.
- `TH`: Thompson sampling.
- `myp_sample`: SMaRT sample-from-belief variant.
- `myp_avg`: SMaRT mean-of-belief variant.
- `myp_grnd`: SMaRT ground-truth value-added variant.

Folder Structure

For full notebook support, including simulation-regeneration cells, keep this structure:

```
Reproduction Package/
  README.md
  requirements.txt
  reproduce_exp_one_click_QP.ipynb
  reproduce_exp_one_click_LP_refinedC5.ipynb
  sensitivity_analysis_of_VA_esitmation.ipynb
  config.py
  Court_case.py
  makeplots_reproduction_package.py
  run_simulation_toy_problem.py
  Simulation.py
  Simulation_product.py
  simulation_utils.py
  SlakedLP.py
  SlakedLPwithLoad.py
  SlakedQPwithLoad.py
  slacked_LP_product.py
  VA_Antoine.py
  cases_raw_07012025.csv
  QP_camera_ready.zip
  LP_camera_ready.zip
  param_files/
    Casetypes_parameters_lognormal.xlsx
```

```
    data_accreditationcategory_20250729.csv
    data_casetype_20250729.csv
    data_casetype_eligible_accreditation_categories_20250729.csv
    vw_all_mediators_20250729.csv
    vw_deidentified_case_w_appointer_20250729.csv
  test_cases/
    __init__.py
    Scenario1.py
    scenario2.py
```

After extracting the camera-ready archives, this layout should also include:

```
  QP_camera_ready/
  LP_camera_ready/
```

If `run_slurm = True`, also keep the required QP and LP `.sh` scripts:

```
run_seq_toy_example1.sh
run_seq_toy_example2.sh
run_seq_FullDataGround.sh
run_seq_FullDataLearningFromScratch.sh
run_seq_FullDataLearningBootstrap.sh
run_batch_parallel_array_automated_toy_example1.sh
run_batch_parallel_array_automated_toy_example2.sh
run_batch_parallel_FullDataGround.sh
run_batch_parallel_FullDataLearningFromScratch.sh
run_batch_parallel_FullDataLearningBootstrapCor.sh
run_seq_toy_example1LP.sh
run_seq_toy_example2LP.sh
run_seq_FullDataGroundLP.sh
run_seq_FullDataLearningFromScratchLP.sh
run_seq_FullDataLearningBootstrapLP.sh
run_batch_parallel_array_automated_toy_example1LP.sh
run_batch_parallel_array_automated_toy_example2LP.sh
run_batch_parallel_FullDataGroundLP.sh
run_batch_parallel_FullDataLearningFromScratchLP.sh
run_batch_parallel_FullDataLearningBootstrapLP.sh
```

If `one_click_path_A = False` and `run_slurm = False`, the shell scripts are not required because the notebooks call `run_simulation(...)` directly.

Input Files

All of the following files are necessary and integral to the experiment (located in param_files directory). The directory needs to contain all of them.

vw_deidentified_case_w_appointer_20250729.csv: Complete list of historical cases with details of each. Most important details are: Start and end date, mediator assigned to the case, outcome, case type and court station. data_casetype_20250729.csv: List of all case types and a case-type-ID used internally in our code. data_accreditationcategory_20250729.csv: List of all accreditations and accreditation-ID used internally in our code. data_casetype_eligible_accreditation_categories_20250729.csv: Mapping of accreditations to case types. So, which accreditations allow mediators to work in each case type. vw_all_mediators_20250729.csv: Complete list of present and past mediators, their accreditations, which court stations they can work in, and their current status (active/inactive/mentorship). Casetypes_parameters_lognormal.xlsx: Parameters of log-normal distributions fitted to case resolution time conditional on case outcome and case type. Derived from vw_deidentified_case_w_appointer_20250729.csv.

The files located in "test_cases" folder (Scenario1.py, Scenario2.py) are both necessary. These are synthetic test configurations and are used to generate Tables 2, 3, 6, and 7; and Figures 3, 4. In the paper, in section 4.2 (Stylized Scenario Analysis), we describe how we use stylized scenarios to demosntrate the behaviour of our algorithm under certain settings. Scenario1.py, Scenario2.py encode these two scenarios (descrived in Table 1). These files are used to run simulations and to generate the intermediate data files containing 'Scenario1' or 'Scenario2' in their names, which are in turn used to generate Tables 2, 3, 6, and 7; and Figures 3, 4 across reproduce_exp_one_click_QP.ipynb, reproduce_exp_one_click_LP_refinedC5.ipynb

Optional Files

The following folders are not required to rerun the three target notebooks and can be omitted from a clean replication package:

```
.git/
.agents/
.codex/
__pycache__/
RCTSimulation/
plotsReproduceRun/
```

`plotsReproduceRun/` is regenerated automatically when plotting cells run.

The following notebooks are not required for the three target notebook workflows:

```
analysis_mediator_distribution.ipynb
cluster_analysis.ipynb
debug.ipynb
double_check.ipynb
global_sd.ipynb
make_plots_with_clustering.ipynb
p_val_histogram_stugg.ipynb
plots_for_paper.ipynb
reproduction_package_one_clic.ipynb
small_scale_sim.ipynb
timeQPvsLP.ipynb
```

The following data files are not required for the three target notebook workflows:

```
avg_incoming_Rate_cases.csv
mediator_active_case_dates.csv
mediator_cs_casetype_24042024.csv
mediator_matrix.csv
```

Do not delete `cases_raw_07012025.csv`.

The following Python and Stata files are not imported by the three target notebooks or by `run_simulation_toy_problem.py` for the current notebook workflows:

```
02_Hazard_AS.do
analyze_and_gen_results.py
clervointLP.py
clervointLPDifferentFormulation.py
clervointQP.py
code_deviationsimulation_for_each_case.py
crps.py
hazard_AS.py
make_plots.py
run_simulation.py
run_simulation_product.py
run_simulation_toy_problem_loadLP.py
run_simulation_toy_problem_testing.py
simulation_for_each_case_joint_distribution.py
simulation_for_each_case_joint_distribution_old_algorithm.py
simulation_lp_withload.py
Sla.py
```