Reminder

Open a terminal inside VSCode: Terminal > New Terminal from the menu. Inside the terminal, type the following:

- `conda create -n simopt python=3.13` (skip if already created)
- `conda activate simopt`
- `pip install -U jupyterlab simoptlib`
- `python -m simopt`

Simulation Optimization Library

WSC 2025 Workshop, Seattle, WA

David J. Eckman, Texas A&M University
Shane G. Henderson, Cornell University
Sara Shashaani, North Carolina State University
Cen Wang, Texas A&M University

Supported by NSF grants OAC-2410948/49/50.

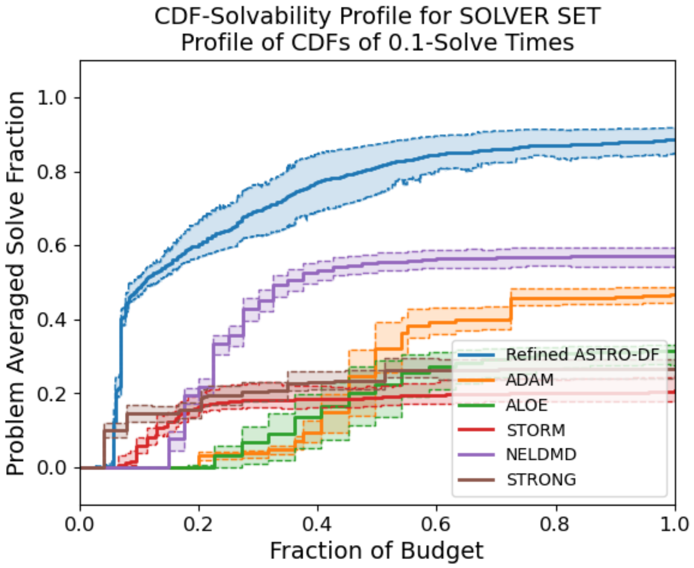
The Big Picture I

SimOpt is a library of simulation-optimization problems and solvers, together with experimental tools, intended

1. To benchmark solvers to aid with solver development
2. To focus attention on the finite-time performance of solvers
3. To identify important/difficult classes of problems
4. To aid with solver hyper-parameter tuning
5. For use mostly by simulation-optimization researchers and solver developers
6. To enable practitioners to apply solvers from the library to their own simulation-optimization problems

Users of SimOpt should be comfortable using Python tools and GitHub, though a GUI provides a lot of functionality

The Big Picture II



Goals

Our goal in this workshop is for you to learn how to:

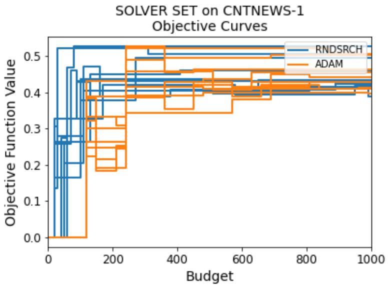
- Run SimOpt experiments
- Understand the plots
- Use the GUI
- See a data farming experiment in SimOpt
- Build and contribute your own models, problems and solvers

Comparing solvers

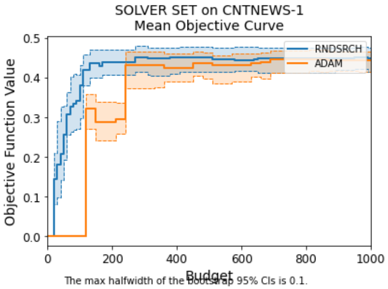
Newsvendor problem

- Per-unit order cost $c = 5$, selling price $s = 9$, salvage value $w = 1$
- Real-valued order quantities
- Demand $D \sim F(x) = 1 - (1 + x^{\alpha})^{-\beta}$ where $x \geq 0, \alpha = 2, \beta = 20$ (this is Burr Type XII)
- Maximize expected profit, where we use simulation to estimate expected profit rather than use explicit formulas
- Which of the following two solvers is better?
  1. ADAM
  2. Random search (exp(1)), with 10 reps per solution
- Budget of 1000 replications, start at $x_0 = 0$

Progress curves



(a) for 10 macroreplications



(b) mean and CI

Figure: Unnormalized progress curves.

Important SimOpt objects

model	A simulation model has a replicate method that generates replications. Models can have many factors like c, s, w, α, β in continuous newsvendor
problem	Built from a model, with objective function, decision variables, constraints. Many problems can be generated from a single model
solver	solvers tackle one or more problems, take simulation replications sequentially, and report a “running estimated best solution”
generators	SimOpt makes careful use of common random numbers through a custom implementation of the generator MRG32k3a

Problems

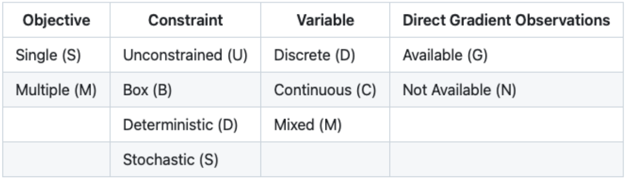
$$\begin{aligned} \min_x\ & f(x,w) = \mathbb{E} f(x,w,\xi) \\ \text{s/t}\ & g(x,w) = \mathbb{E} g(x,w,\xi) \leq 0 \\ & h(x,w) \leq 0 \\ & x \in \mathcal{D}(w). \end{aligned}$$

Problems have a problem-specific *budget* T measured in simulation replications. We choose it to be “just right” - not too big, not too small

Problems can have continuous variables, integer-ordered variables, or a mixture of these

Solvers

- We have many more solvers for continuous-variable problems than for integer-ordered-variable problems.
- Research opportunity: Add more problems and solvers.
- Some solvers use a random budget, e.g., R&S with statistical guarantees. Currently, we don't support random-budget solvers

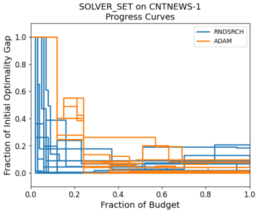


Objective	Constraint	Variable	Direct Gradient Observations
Single (S)	Unconstrained (U)	Discrete (D)	Available (G)
Multiple (M)	Box (B)	Continuous (C)	Not Available (N)
	Deterministic (D)	Mixed (M)	
	Stochastic (S)		

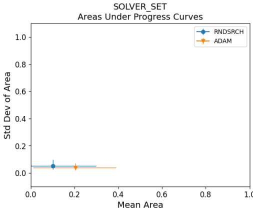
Your turn: Generate some useful plots

1. Click “Simulation Optimization Experiments” on the SimOpt GUI Main Menu.
2. On the New Experiment form, click the “Quick Add Problems/Solvers” tab on the right panel.
3. Select RNDSRCH and ADAM solvers and “CNTNEWS-1 Max Profit for Continuous Newsvendor” and click “Add Cross Design to Experiment”.
4. On the lower left panel (Current Experiments Workspace), click “Create Experiment”.
5. Click “Run All Remaining Steps” on upper left panel (Created Experiments).
6. Click “Open Plotting Window” and in the Experiment Plots form select the Experiment, simultaneously select the solvers and problems with a Ctrl button, and select Plot Type. Click “Plot” and view the saved plot by clicking “View/Edit” or “View All Created Plots”.

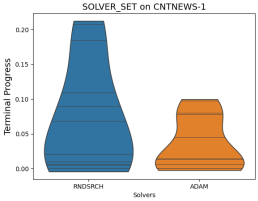
Example plots



(a) all progress curves



(b) area scatter plots



(c) terminal violin plots

Figure: Comparison of two solvers on the Newsvendor problem.

Time, macroreplications, postreplications, bootstrapping

We measure time t through the fraction of the budget T that has been used, $t \in [0, 1]$

A macroreplication is a single run of a single solver on a single problem. The solver tracks an estimated best solution as a function of time $(X(t) : 0 \leq t \leq 1)$

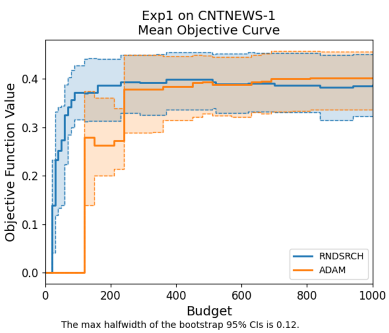
After the macroreplications are complete we use postreplications to estimate $(f(X(t)) : 0 \leq t \leq 1)$ for each macroreplication

Both macroreplications and postreplications are stochastic. We use bootstrapping to provide confidence intervals for $\mathbb{E}f(X(t))$ and related quantities at each $t \in [0, 1]$

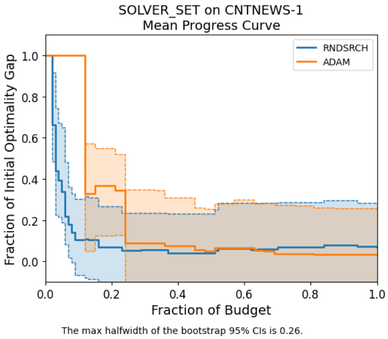
(Normalized) progress curves

- Recall that time is measured as a fraction of the budget T, so $t \in [0, 1]$.
- Rescale the objective function to
$$\frac{f(X(t)) - f(x^*)}{f(x_0) - f(x^*)}.$$
- $f(x^*) =$ optimal objective function value, part of problem. If unknown, estimated as best solution seen.

Normalized progress curves



(a) unnormalized



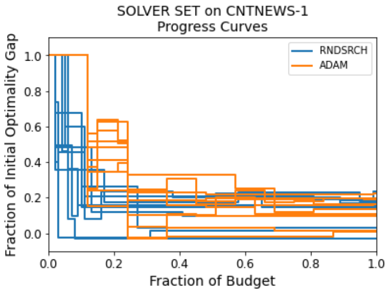
(b) normalized

Figure: Mean+CI progress curves.

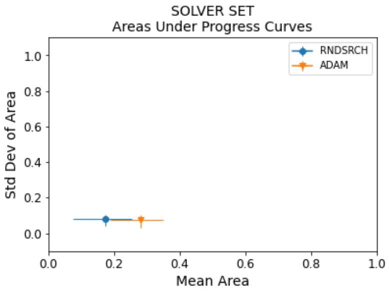
Multiple problems

Area under progress curves and scatter plots

- Summarize performance of many solvers on many problems
- Let A be the (random) area under a normalized progress curve from a single macroreplication. Plot (μ_A, σ_A) and their marginal CIs for each problem and solver



(a) progress curves



(b) scatter plots

Figure: Area under progress curves.

Area scatter plots

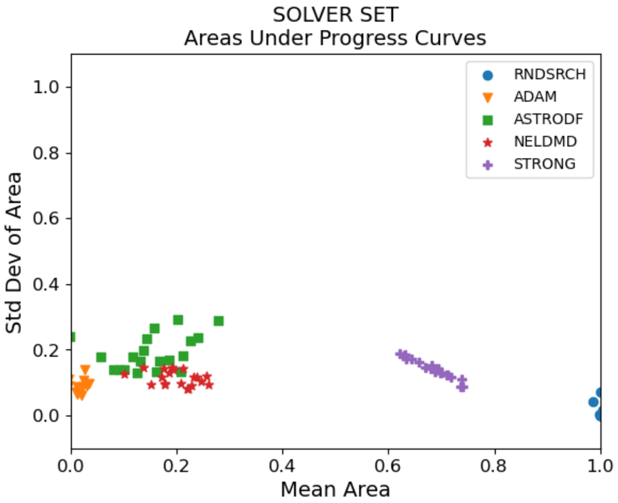


Figure: Comparing 5 solvers on 20 problems.

Solvability profiles

Main purpose is to explore time-dependent performance of multiple solvers on multiple problems. First consider *one* problem and *one* solver

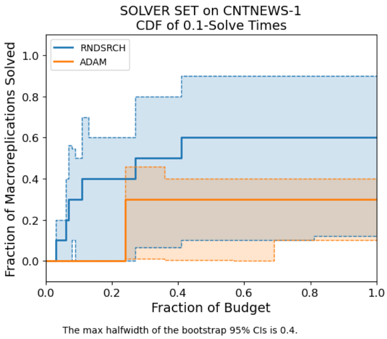
- Look at progress curves a different way. How long to reduce initial optimality gap to 10% of initial value?

$$\tau = \text{budget } t \text{ when first get within } 10\%$$

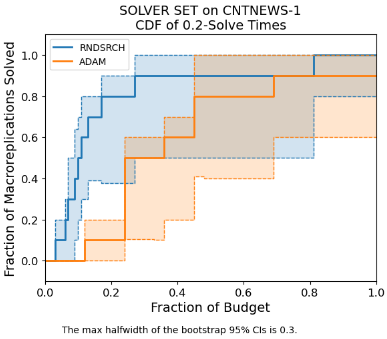
- τ is *random* and can $= \infty$ with positive probability
- We call τ the 10% solve time
- Could look at cdf and/or quantiles of τ ...

Solve time

One problem, Multiple solvers



(a) $\alpha = 0.1$



(b) $\alpha = 0.2$

Figure: α-solve time CDF.

Solvability profiles
Multiple problems

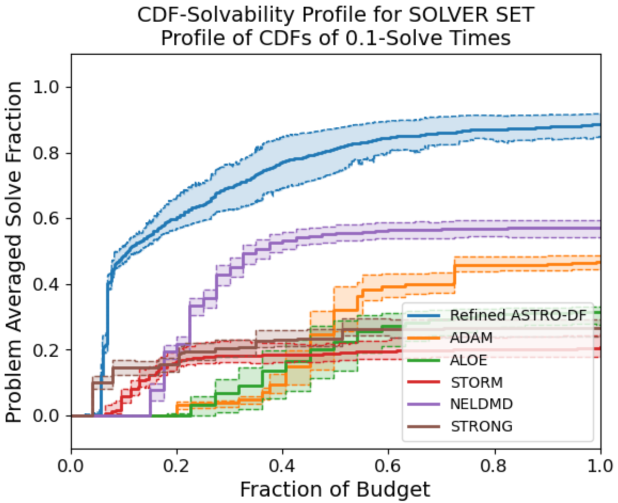
Fix a solver

- Let τ^p be the solve time for problem p
- CDF solvability profile: As a function of t,

  $$\rho(t) = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \Pr(\tau^p \leq t)$$

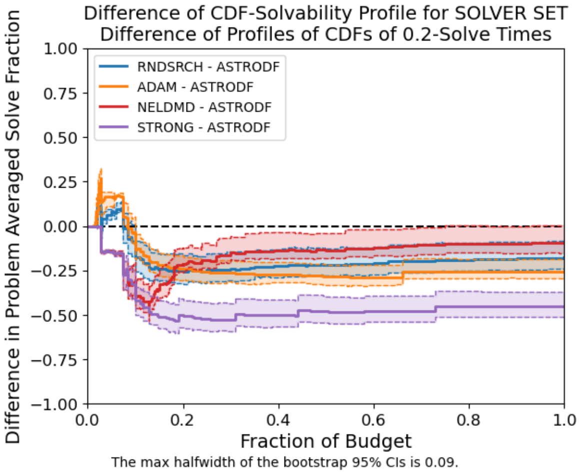
  Average, over problems, of probability solve by time t
- Quantile solvability profiles gives the fraction of problems that are likely (a quantile) solved by time t, as function of t. (Skipping those, today.)

Examples


CDF-Solvability Profile for SOLVER SET
Profile of CDFs of 0.1-Solve Times
Problem Averaged Solve Fraction
Fraction of Budget
0.0
0.2
0.4
0.6
0.8
1.0
Refined ASTRO-DF
ADAM
ALOE
STORM
NELDMD
STRONG

Difference profiles

If you are focused on one solver in particular, compute differences in solvability profiles relative to that solver



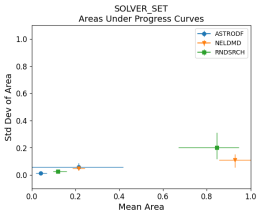
Your turn

1. On New Experiment form, click the "Quick Add Problems/Solvers" tab on the right panel.
2. Select the following problems
   - IRONORECONT-1 - Max Revenue for Continuous Iron Ore, and
   - SAN-1 - Min Mean Longest Path for Stochastic Activity Network

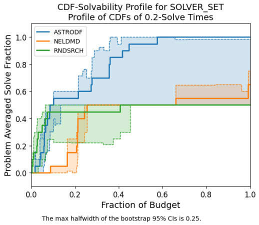
   and solvers
   - ASTRODF - ASTRO-DF,
   - RNDSRCH - Random Search, and
   - NELDMD - Nelder-Mead.

   and click "Add Cross Design to Experiment".
3. On the lower left panel, click "Create Experiment".
4. Click "Run All Remaining Steps" on upper left panel.
5. Click "Open Plotting Window" and in the Experiment Plots form select the Experiment, simultaneously select the solvers and problems with a Ctrl button, and select Plot Type. Click "Plot" and view the saved plot by clicking "View/Edit" or "View All Created Plots".

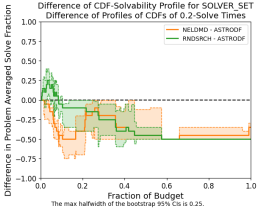
Two problems, Three solvers



(a) Area scatter plots



(b) Solvability profiles



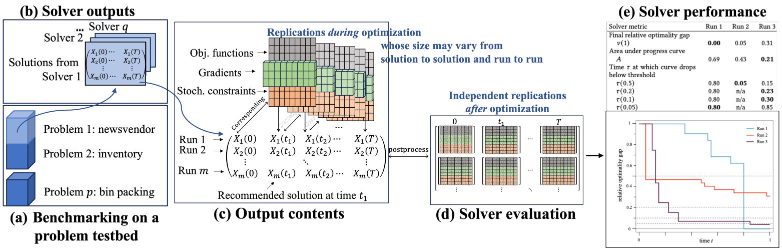
(c) Difference profiles

Figure: Comparing 3 solvers on 2 problems.

- Log file of this experiment

Running a data farming experiment

Review of SimOpt



Solver metric	Run 1	Run 2	Run 3
Final relative optimality gap			
$\nu(1)$	**0.00**	0.05	0.31
Area under progress curve			
A	0.69	0.43	**0.21**
Time τ at which curve drops below threshold			
$\tau(0.5)$	0.80	**0.05**	0.15
$\tau(0.2)$	0.80	n/a	**0.23**
$\tau(0.1)$	0.80	n/a	**0.30**
$\tau(0.05)$	**0.80**	n/a	0.85

Data Farming and SimOpt

Recall

$$\min_{x \in \mathcal{D}(w)} f(x, w) = \mathbb{E} f(x, w, \xi)$$
$$\text{s/t } g(x, w) = \mathbb{E} g(x, w, \xi) \leq 0.$$

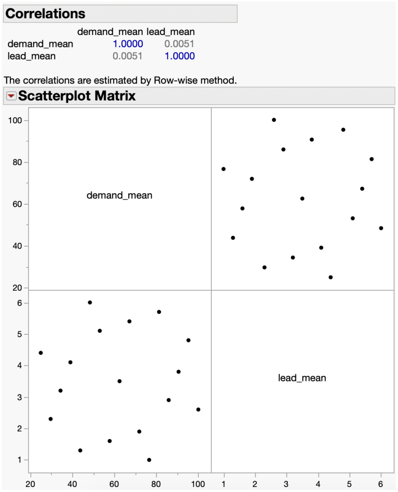
Data farming: draw inferences about a stochastic system by "growing" a dataset from an experimental design.

- A data farming experiment in SimOpt has two steps:
  1. Create large designs over problem and/or solver *factors*.
  2. Run experiments on the resulting problem-solver pairs.
- Analyze the resulting plots and data with statistical software to:
  - use case 1: data farm a problem given a solver to
    - study solver robustness/sensitivity to problem variants.
  - use case 2: data farm a solver given a problem (set) to
    - tune solver hyperparameters for a (class of) problem(s).
- SimOpt has a rudimentary version of this data-farming capability.

Demonstration: Which solver is more robust?

1. From Quick-Add tab select ASTRODF and NELDMD and click "Add Cross Design to Experiment".
2. Create a *nearly orthogonal Latin Hypercube Sampling* (NOLHS) design of SSCONT-1 in "Add Problem" tab on left panel by selecting the following factors (via checkbox "Include in Design") with given details:
   - demand_mean: min=25.0, max=100.0, decimals=1
   - lead_mean: min=1.0, max=6.0, decimals=1
3. Click "Generate Problem Design"*, and after the table of design points appear, click "Add this Problem Design to Experiment".
4. Click "Change Default Experiment Options" and change the number of replications to 3; the "Create Experiment" and "Run All Remaining Steps". (Approx $\sim 1 - 2$ min)
5. Repeat previous steps to run the experiment and make plots.

Space-filling design of (s, S)-inventory problem instances



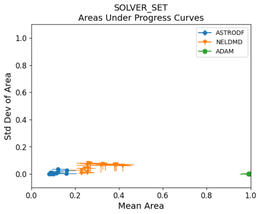
Correlations

	demand_mean	lead_mean
demand_mean	1.0000	0.0051
lead_mean	0.0051	1.0000

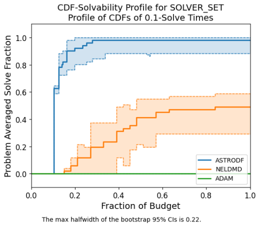
The correlations are estimated by Row-wise method.

Scatterplot Matrix

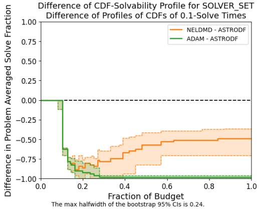
Diagnosing robustness for various inventory instances



(a) Area scatter plots



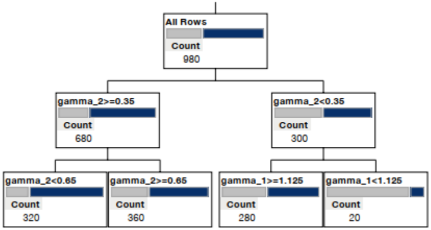
(b) Solvability profiles



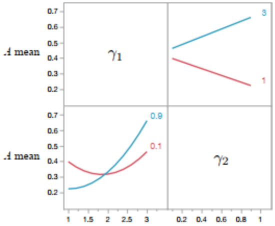
(c) Difference profiles

Figure: Solvers' performance on various inventory problem instances. (These plots include the additional solver, ADAM, which we didn't include with your design to reduce the running time.)

Choosing solver hyperparameters with "farmed" data...



(a) Partial partition tree for a generalist, showing the top three layers of the 10-split partition tree for the 0.1-solvability, $y(0.10)$, for both problems using the raw datafile. The light and dark bars in each leaf indicate the proportions of non-solved and solved runs, respectively.

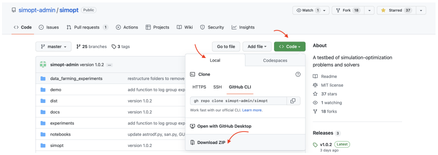


(b) Interaction profiles for a specialist, based on stepwise regression metamodel for A mean for the SSCONT problem. There is a quadratic effect for γ_1, a linear effect for γ_2, and a strong interaction.

Writing your own models, problems, and solvers

Reminder

1. Download zip: `https://github.com/simopt-admin/simopt`
2. Unzip folder simopt-master, open in VS Code: File>Open Folder
3. Locate these files:



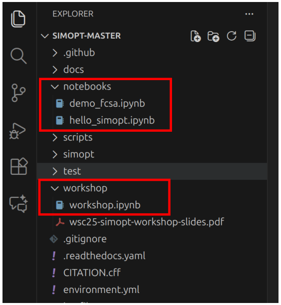
simopt-admin version 1.0.2

data_farming_experiments	restructure folders to remove
demo	add function to log group exp
dist	version 1.0.2
docs	version 1.0.2
experiments	add function to log group exp
notebooks	update astrodf.py, san.py, GU
simopt	version 1.0.2

```
gh repo clone simopt-admin/simopt
```

Work fast with our official CLI. Learn more.

A testbed of simulation-optimization problems and solvers



```
SIMOPT-MASTER
  > .github
  > docs
  v notebooks
      demo_fcsa.ipynb
      hello_simopt.ipynb
  > scripts
  > simopt
  > test
  v workshop
      workshop.ipynb
      wsc25-simopt-workshop-slides.pdf
  .gitignore
  .readthedocs.yaml
  CITATION.cff
  environment.yml
```

SimOpt for educators

- Explain discrete-event simulation
- Random number and random variate generation: Burr Type XII input distribution
- Testing common random numbers: switch on / off
- Simulation optimization on a continuous newsvendor and comparison with analytical solution: explain the evaluation per budget expended
- Compare sensitivity to solver parameters (data farming)
- Ask for changes in the algorithm (e.g., step size in ADAM: adding Polyak step, adding backtracking)
- Data driven example: write a problem for logistic regression with regularization using a dataset (new feature)

Wrapping up

Latest Release: SimOpt v1.2, November 2025

- Plotting for stochastically constrained problems, new FCSA solver, SAN-2 problem; see `demo_fcsa.ipynb`.
- Input-model hooks for data-driven simulation; see `demo_data_driven.ipynb`.
- Refactor `Solver`, `Problem`, and `Model` classes; simplify developer experience.
- Rust-based MRG32k3a backend (enable via `MRG32K3A_BACKEND=rust`); planned as the default in a future release.
- Rewrite the NOLHS generator for data farming.

Contributing to SimOpt

- Fork SimOpt on GitHub

- Make changes & fix potential issues (only for your files)
  - `ruff format`
  - `ruff check`
  - `pyrefly check`
- Commit, push & open a pull request (details at `https://docs.github.com/en/pull-requests`)
- We will review and merge your contributions

Future Plans for SimOpt

- Web-based GUI
- Increasing problem/solver diversity (DASSO, etc.)
- Trace-driven simulation (feedback welcome)
- Improve code quality, better developer experience
- Rust extension for faster simulation (V1.2.2)
- Random problem instances
- Metamodeling for multi-fidelity with data farming
- Multi-objective capabilities
- Calibration, empirical risk minimization, distributionally robust optimization
- Automatic differentiation

Thanks for attending today!

Please stay involved! Some ideas:

1. Email Cen Wang to join the GitHub. `cenwang@tamu.edu`
2. Use SimOpt to test/improve your own solver, and share your solver code with us.
3. Use SimOpt to tackle your own problem, and share your problem code with us.
4. Tackle some of the items on the SimOpt to-do list (see the README on GitHub and please work on your own fork). https://github.com/simopt-admin/simopt/issues
5. Suggest improvements, bug fixes, ideas for new problems, ideas for new solvers (send us email).

References

- http://simopt.org
- Diagnostic tools for evaluating and comparing simulation-optimization algorithms
  David J. Eckman, Shane G. Henderson, and Sara Shashaani
  INFORMS Journal on Computing **35** (2) 350–367, 2023.
- SimOpt: A testbed for simulation-optimization experiments
  David J. Eckman, Shane G. Henderson, and Sara Shashaani
  INFORMS Journal on Computing **35** (2) 495–508, 2023.
- Data farming the parameters of simulation-optimization solvers
  Sara Shashaani, David J. Eckman, and Susan M. Sanchez
  ACM TOMACS **34** (4), 2024.
- Sanchez, S. M. and P. J. Sanchez. 2025. Data Farming for Simulation Analysis.
  `https://github.com/simulationdoe/datafarmingbook`