Installation

We assume you have done the following: Installed miniconda and

```
git clone https://github.com/OpenSourceEconomics/scipy-estimagic.git
cd scipy-estimagic
conda env create -f environment.yml
conda activate scipy-estimagic
```

- If you haven't done so, please do so until the first practice session
- Details: https://github.com/OpenSourceEconomics/scipy-estimagic

Practical Numerical Optimization with Scipy, Estimagic and JAXopt

Scipy Conference 2022

Janos Gabler & Tim Mensinger

University of Bonn

About Us

- Website: janosg.com
- GitHub: janosg
- Started estimagic in 2019
- Just submitted PhD thesis, looking for jobs soon

- Website: tmensinger.com
- GitHub: timmens
- estimagic core contributor
- PhD student in Econ, University of Bonn

Sections

1. Introduction to `scipy.optimize`
2. Introduction to `estimagic`
3. Choosing algorithms
4. Advanced topics
5. Jax and Jaxopt

Structure of each topic

1. Summary of exercise you will solve
2. Some theory
3. Syntax in very simplified example
4. You solve a more difficult example in a notebook
5. Discuss one possible solution

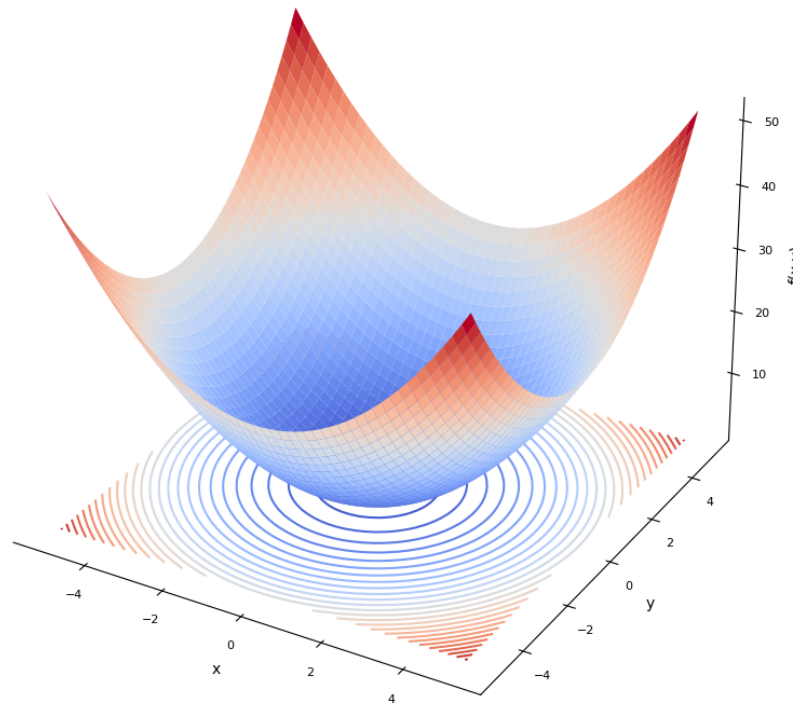
Introduction to scipy.optimize

Preview of practice session

- Translate a criterion function from math to code
- Use `scipy.optimize` to minimize the criterion function

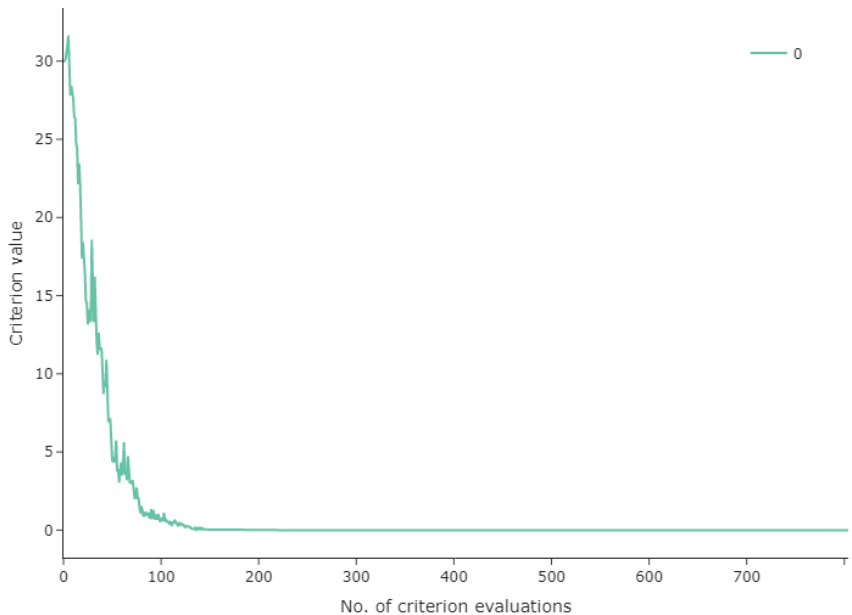
Example problem

- Criterion $f(a, b) = a^2 + b^2$
- Parameters a, b
- Want: $a^*, b^* = \operatorname{argmin} f(a, b)$
- Possible extensions:
  - Constraints
  - Bounds
- Optimum at $a^* = 0, b^* = 0, f(a^*, b^*) = 0$



Criterion plot

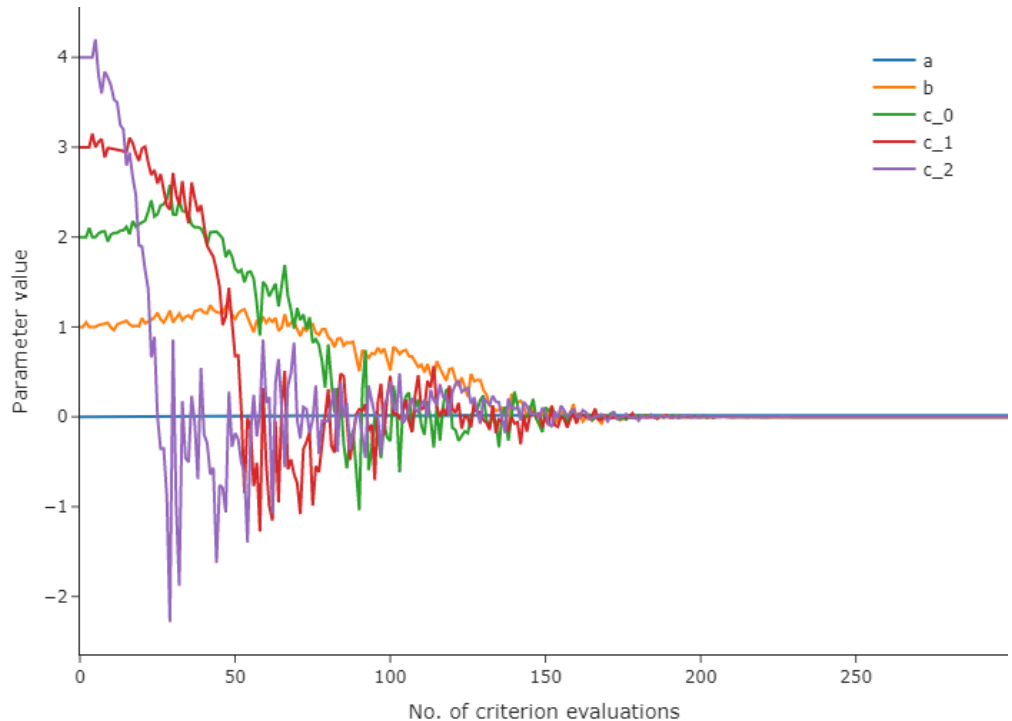
```
em.criterion_plot(res)
```



- First argument can be:
  - `OptimizeResult`
  - path to log file
  - list or dict thereof
- Dictionary keys are used for legend

Params plot

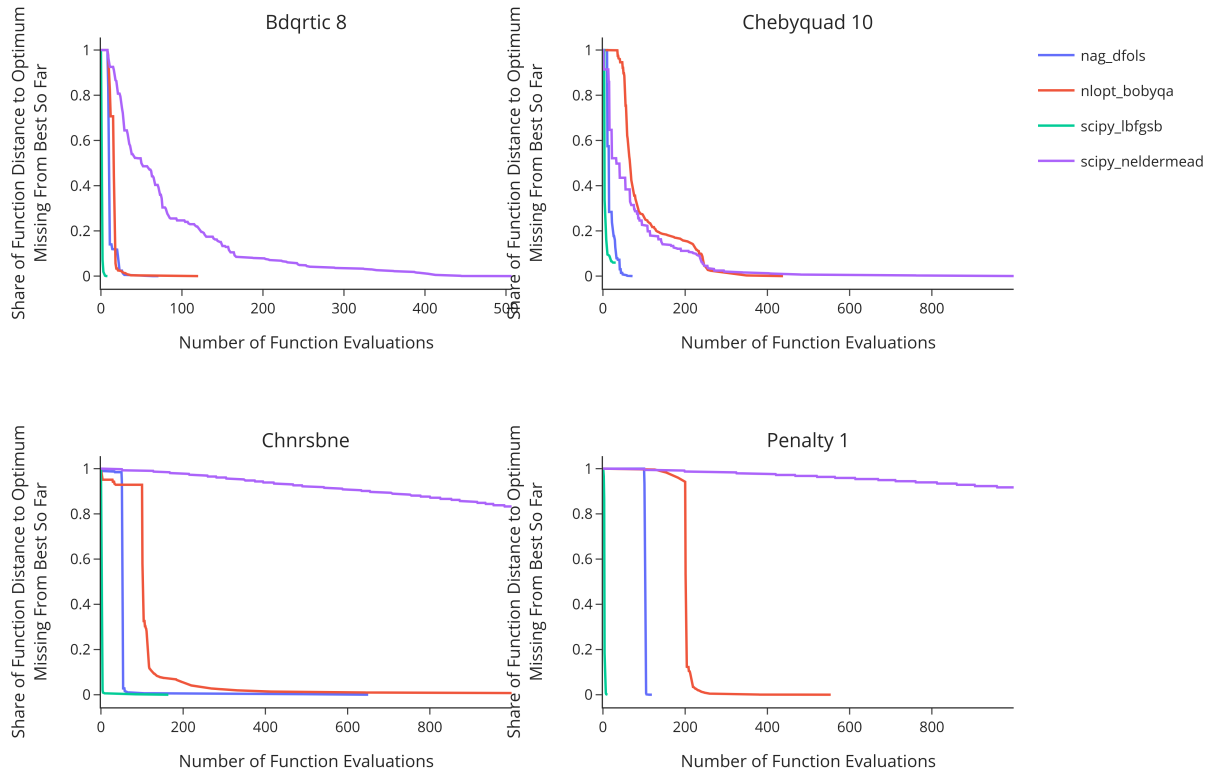
```
# reminder: params looks like this
params = {
    "a": 0,
    "b": 1,
    "c": pd.Series([2, 3, 4])
}

em.params_plot(
    res,
    max_evaluations=300,
)
```



- Similar options as `criterion_plot`

Convergence plots

```
subset = [
    "chebyquad_10",
    "chnrsbne",
    "penalty_1",
    "bdqrtic_8",
]
em.convergence_plot(
    problems,
    results,
    problem_subset=subset,
)
```



Logging and Dashboard

```
res = em.minimize(
    criterion=sphere,
    params=np.arange(5),
    algorithm="scipy_lbfgsb",
    logging="my_log.db",
 )
```

```
from estimagic import OptimizeLogReader

reader = OptimizeLogReader("my_log.db")
 reader.read_history().keys()
dict_keys(['params', 'criterion', 'runtime'])

reader.read_iteration(1)["params"]
array([0., 0.817, 1.635, 2.452, 3.27 ])
```

- Persistent log in sqlite database
- No data loss ever
- Can be read during optimization
- Provides data for dashboard
- No SQL knowledge needed