Effective Programming Practices for Economists

Numerical Optimization

Visualizing optimizer histories

Janoś Gabler and Hans-Martin von Gaudecker

Steps for choosing an algorithm

1. Theory (algorithms video)
2. Experimentation (here)
3. Refine until convergence

Motivation

- You rarely have a guarantee that an optimizer will work
  - Assumptions of convergence proofs might not hold in practice
  - You might get stuck in local optima
  - Floating point calculations are never exact
- But you can compare the performance of optimizers
  - Which one finds the lowest/highest function value?
  - Which one leads to the quickest decrease/increase in function values?
- The `criterion_plot` makes this very easy!

Criterion plot

We assume you have done an optimization and the result is called `res`

An error occurred on this slide. Check the terminal for more information.

An error occurred on this slide. Check the terminal for more information.

An error occurred on this slide. Check the terminal for more information.

Criterion plot for multiple optimizations

```python
def sphere(x):
    return x @ x

results = {}
for algo in ["scipy_neldermead", "nlopt_neldermead", "fides"]:
    results[algo] = om.minimize(
        sphere,
        np.arange(10),
        algorithm=algo,
    )

om.criterion_plot(results, max_evaluations=200)
```