Effective Programming Practices for Economists

Numerical Optimization

Using optimagic’s minimize and maximize

Janoś Gabler and Hans-Martin von Gaudecker

Simple usage of minimize and maximize

```python
import numpy as np
import optimagic as om

def sphere(x):
    return (x ** 2).sum()

start_params = np.ones(5)

res = om.minimize(
    fun=sphere,
    params=start_params,
    algorithm="scipy_lbfgsb",
)

res.params
```

```
array([0., 0., 0., 0., 0.])
```

1. Import optimagic
2. Define criterion function
3. Define start params
4. Call minimize

`maximize` works the same!

Same problem — different params

```
>>> params = {"a": 0, "b": 1, "c": pd.Series([2, 3, 4])}

>>> def dict_sphere(x):
...     return x["a"] ** 2 + x["b"] ** 2 + (x["c"] ** 2).sum()

>>> res = om.minimize(
...     fun=dict_sphere,
...     params=params,
...     algorithm="scipy_neldermead",
... )
>>> res.params
{
  'a': 0.,
  'b': 0.,
  'c': 0    0.
       1    0.
       2    0.
  dtype: float64
}
```

Bounds for parameters

```
>>> res = om.minimize(
...     fun=dict_sphere,
...     params=params,
...     algorithm="scipy_neldermead",
...     lower_bounds={"b": 0.5}
... )

>>> res.params
{
  'a': 0.,
  'b': 0.5,
  'c': 0    0.
       1    0.
       2    0.
  dtype: float64
}
```

- Extend previous example
- Only need to specify bounds for parameters that need them
- `upper_bounds` work analogously
- Can use `np.inf` and `-np.inf` to explicitly specify no bound

Inspecting results

```
>>> res.criterion

0.

>>> res.n_criterion_evaluations

805

>>> res.success

True

>>> res.message

'Optimization terminated successfully.'
```

- You already know res.params
- There are many other useful attributes
- Elements of results objects can also be plotted

Documentation of more features

- How to specify algorithms and their options
- How to use constraints
- How to do multistart optimization
- How to handle errors during optimization