進捗報告

1 今週やったこと

- Pyomo のコードのデバッグ
- ネルダーミード法 (x 固定) から CMA-ES

2 Pyomo のコードのデバッグ

先週から引き続き，Pyomo で書いたベンチマーク問題のコードのデバッグをした．変数の大半を固定してローカル及びサーバーで実行したが以前から出ているエラーが発生した．Listing 1 にそのエラーを示す.

Listing 1: 発生したエラー

```
1 ValueError: MindtPy unable to handle
    relaxed NLP termination condition of
     other. Solver message: Too few
    degrees of freedom (rethrown)!
```

ただし，このとき使用したソルバーは MindtPy である．次に，定義する変数や制約条件自体を減らす目的で，運用計画を考える時刻数 I を小さくした．その結果, $I \leq 2$ にするとコードを実行できた．このとき，変数の固定の有無は関係がなく，ローカル及びサーバーも実行結果は同じであった．よって，Listing 1 のエラーは変数や制約条件を定義した段階でその数が多いと発生すると考えられる.

3 ネルダーミード法 (x 固定) から CMA-ES

ベンチマーク問題に対して，各機器の熱出力及び消費ガス量 x を機器の稼働状態に応じて固定しネルダーミード法を適用し，その最終ステップの解を CMA-ES の初期平均ベクトルとして最適化をした．x の固定は，機器が稼働状態のときは各上・下限値の平均値，非稼働状態のときは 0 とした．また，制約違反はペナルティ法を用いて各手法の目的関数 F に組み込んだ．表 1 にネルダーミード法の実験パラメータを，表 2 に CMA-ES の実験パラメータを示す． CMA-ES において制約違反をある程度許容し，目的関数を柔軟に最小化するためにペナルティ関数の係数 ρ を小さめに設定した．その設定により目的関数が下がりすぎることが予備実験でわかったので，目的関数が定めた閾値より小さくなったらペナルティを与えることで対処した.

表 1: ネルダーミード法の実験パラメータ

パラメータ	値
変化とみなす閾値	1.0×10^{-5}
変化しない場合終了するステップ数	200
最大操作数	1000
ρ (ペナルティ関数の係数)	1.0×10^{3}

表 2: CMA-ES の実験パラメータ

パラメータ	値
σ (初期標準偏差)	0.05
世代数	5000
入力変数の次元	120
一世代の個体数 λ	1200
ρ (ペナルティ関数の係数)	1.0×10^{4}

表 3 に実験で得た最終的な目的関数値と制約違反を示す．図 1，図 2 に実験で得た目的関数値の推移及び制約違反の推移を示す.

表 3: 実験結果

目的関数値	制約違反
4033408.653	$9.778991847 \times 10^{-11}$

結果として，実行可能解を得ることができた．ネルダーミードの探索は高速であり，およその実行可能解を探せるのではないかと考えた．また，CMA-ES においてペナルティ関数の係数を小さくし，目的関数が閾値より小さくなったらペナルティを与える方法は有効ではないかと考えた.

4 今後の展望

一部固定ネルダーミード法と CMA-ES を組み合わせたパラメータの調整及び，数理計画ソルバーの検討をしたい.

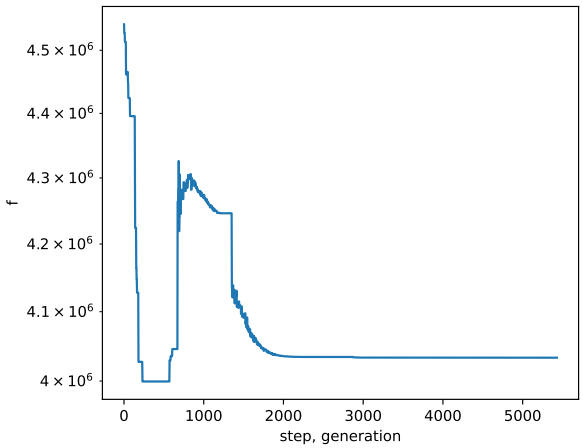

図 1: 目的関数値の推移

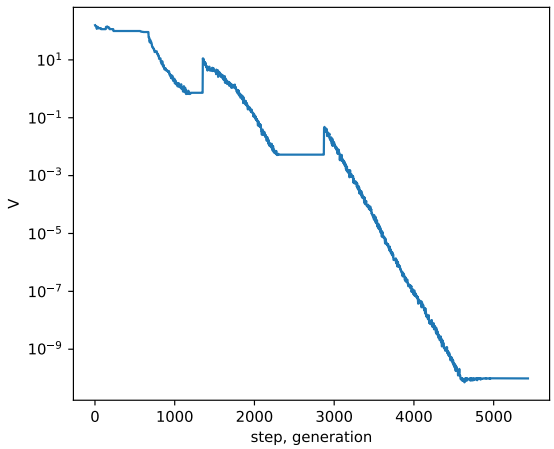

図 2: 制約違反値の推移