進捗報告

1　第 2 回までにしたこと

- Pyomo の検討
- Nelder-Mead 法と CMA-ES を用いた最適化

2　問題設定

ガスタービン一台，ボイラ一台，ターボ式冷凍機一台，蒸気吸収式冷凍機二台の 5 つの機器からなるエネルギープラントの 24 時刻運用問題であり，120 次元の入力変数 $\boldsymbol{x}$ が存在する．表 1 に $\boldsymbol{x}$ の定義域を示す．なお，変数の定義域は各機器が稼働時のものであり，停止時は 0 となる．

表 1: 変数説明

変数	変数の定義域	変数の意味
x_t	1.5〜5.0	ターボ式冷凍機の熱生成量
x_{s1}	4.5〜15.0	蒸気吸収式冷凍機 1 の熱生成量
x_{s2}	4.5〜15.0	蒸気吸収式冷凍機 2 の熱生成量
x_g	1103〜3679	ガスタービンのガス消費量
x_b	8.02〜803	ボイラのガス消費量

3　Pyomo の検討

数理計画法からのアプローチとしていくつかのソルバを検討した．その中で Python ベースのオープンソースの最適化モデリング言語として Pyomo[1][2] があるため，第一段階として使用することを検討した．Pyomo は多様な最適化機能を備えている．

3.1　MindtPy

MindtPy(Mixed-Integer Nonlinear Dicomposition Toolbox for Pyomo)[3] は Pyomo で開発されたソフトウェアフレームワークで，分解アルゴリズムを用いて MINLP を解くことができる．MINLP の分解アルゴリズムは，基本的に混合整数線形計画法 (MILP) や非線形計画法 (NLP) に依存している．MindtPy は Outer Approximation(OA) アルゴリズム [4] と Extended Cutting Plane(ECP) アルゴリズムが実装されている．

3.2　変数・制約条件の多さによるエラー

Pyomo で MindtPy を用いてベンチマーク問題の実装をし，実行した結果，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

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

このエラーは MindtPy の非線形計画問題 (NLP) 部分のソルバーとして使われている Interior Point Optimizer(Ipopt) で発生し，制約条件や変数が多すぎることを意味する．メモリーが多いサーバでの実行，変数の固定をしたが結果は変わらなかった．一方で，最適化する運用時刻数を小さくし，定義する変数や制約条件自体を減らすと実行できた．

4　Nelder-Mead 法と CMA-ES を用いた最適化

エネルギープラント運用計画のための最適化ベンチマーク問題 ver.1[5] を対象とする．なお，deap の cma の Strategy を使用した [6].

単目的最適化の場合 (Strategy) は元の目的関数を f, 機器の上下限制約，起動・停止状態などの満たすべき等式・不等式をまとめた制約違反関数を V, ペナルティ関数の重みを ρ とすると，Nelder-Mead 法及び CMA-ES 上での目的関数 F は (1) 式で表される．

$$F = f + \rho V \tag{1}$$

今回の実験では，制約違反関数 V が 1.0×10^{-10} より小さい解を実行可能解と定義する．

予備実験により，Nelder-Mead 法はそのまま適用しても F を十分に小さくできないことがわかったため，決定変数の一部である $\boldsymbol{x}$ を固定し，2 値で各機器の稼働状態を表す $\boldsymbol{y}$ のみの探索をした．$x_i (i = 0, 1, \ldots, 120)$ は $y_i = 0$ ，すなわち機器が停止状態のときは 0, そうでないときは各上下限値の平均値とした．そして，Nelder-Mead 法で見つかった最後の解を CMA-ES の初期平均ベクトルとし，初期個体群に加えた．

Nelder-Mead 法では各値が離散値である $\boldsymbol{y}$ を連続変数で探索できるように，最初の単体における頂点座標の各値を 0.99 以上 1.01 以下のランダムな実数と設定して探索し，目的関数内で各値が 1 より小さければ 0，そうでないとき 1，と処理した後に評価した．

CMA-ES では f が定めた閾値 t より小さくなったら F にペナルティを与えることで，ρ を小さくしても探索中に目的関数が小さくなりすぎて実行不可能な領域を探索することを防いだ．

5 実験

表 2 に Nelder-Mead 法で用いた実験パラメータを，表 3 に CMA-ES で用いた実験パラメータを示す．閾値 t は，Nelder-Mead 法により $\{f, V\} = \{3999381.510, 99.58317874\}$ が得られ，実行可能解の f がこの付近の値となると予測したことから決定した．

表 2: Nelder-Mead 法の実験パラメータ

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

表 3: CMA-ES の実験パラメータ

パラメータ	値
σ (初期標準偏差)	0.1
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}
t (閾値)	3999000

表 4 に実験で得た最終的な目的関数値 f, 制約違反関数 V および既知解を示す．結果として既知解 3 より目的関数値が小さい実行可能解を得ることができた．

次に，図 1 に最適化をした目的関数 F の推移を，図 2 に元の目的関数 f の推移を，図 3 に制約違反関数 V の推移を示す．縦軸はそれぞれ F, f, V, 横軸は Nelder-Mead 法の試行数と CMA-ES の世代数である． ペナルティ関数の係数 ρ が 2 手法間で異なり，Nelder-Mead

表 4: 実験結果

解法	目的関数値	制約違反関数
既知解 1	3999631.278	2.44×10^{-10}
既知解 2	3999635.845	6.43×10^{-12}
既知解 3	4052185.662	3.93×10^{-14}
実験	4033408.653	9.78×10^{-11}

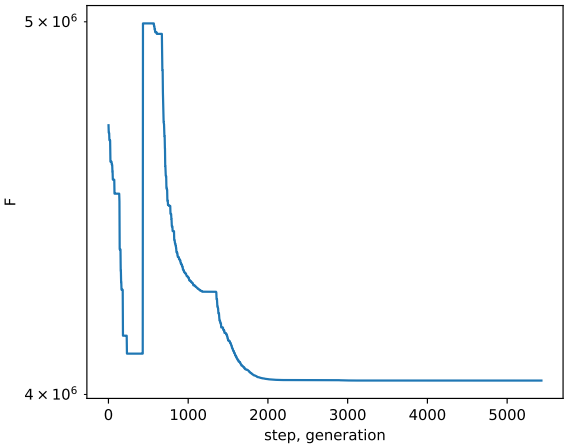

図 1: 目的関数値 F の推移

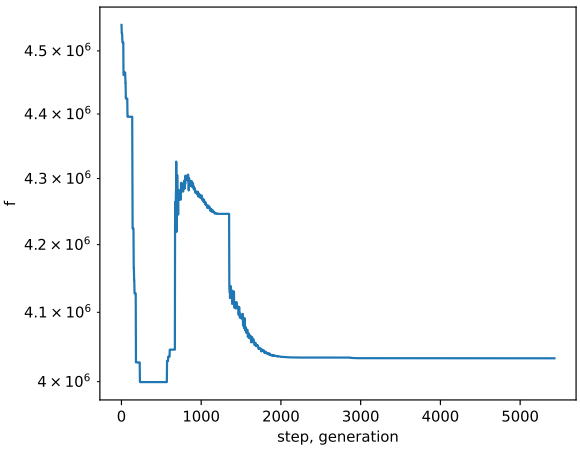

図 2: 目的関数値 f の推移

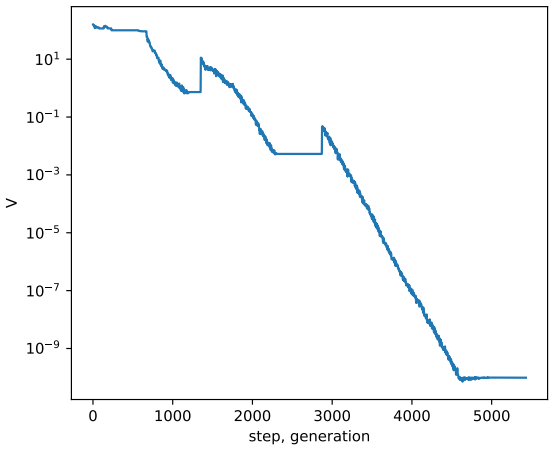

図 3: 制約違反関数 V の推移

法は 430 試行であったため，その前後で F が急に増加している．f は Nelder-Mead 法で 400 万ほどまで小さくできたが，その後の CMA-ES で一度 430 万程度まで増加した．また，そのときの V は減少傾向にある．これは ρ が大きく， V を小さくすることを優先したためだと考えられる．Nelder-Mead 法では， V は最初から低く，あまり減少していない．これは $\boldsymbol{x}$ を適切な値に固定したためだと考えた．

図 4 に 結果として得られた $\boldsymbol{x}$ の変化を機器ごとに示す．

6 今後の展望

数理計画法からのアプローチで，Pyomo 以外のソルバを調査し，検討したい．また，今回 Nelder-Mead 法と CMA-ES による電力プラントの最適化について検討し，既知解に近い実行可能解を得ることができたが，最良の既知解を改善することはできなかった．今後の課題として，ペナルティ関数の係数を最適化の途中で調整することや，各手法の実験パラメータの調整があげられる．

参考文献

[1] Michael L. Bynum, Gabriel A. Hackebeil, William E. Hart, Carl D. Laird, Bethany L. Nicholson, John D. Siirola, Jean-Paul Watson, and David L. Woodruff. *Pyomo–optimization modeling in python*, Vol. 67. Springer Science & Business Media, third edition, 2021.

[2] William E Hart, Jean-Paul Watson, and David L Woodruff. Pyomo: modeling and solving mathematical programs in python. *Mathematical Programming Computation*, Vol. 3, No. 3, pp. 219–260, 2011.

[3] David E. Bernal, Qi Chen, Felicity Gong, and Ignacio E. Grossmann. Mixed-integer nonlinear decomposition toolbox for pyomo (mindtpy). In Mario R. Eden, Marianthi G. Ierapetritou, and Gavin P. Towler, editors, *13th International Symposium on Process Systems Engineering (PSE 2018)*, Vol. 44 of *Computer Aided Chemical Engineering*, pp. 895–900. Elsevier, 2018.

[4] Marco A. Duran and Ignacio E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math. Program.*, Vol. 36, No. 3, p. 307–339, October 1986.

[5] 電気学会情報知能システムの新展開とその産業応用調査専門委員会. 産業応用のための最適化ベンチマーク問題集. 電気学会技術報告 第 1287 号, 3.2 節, 2013.

[6] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, Vol. 13, pp. 2171–2175, jul 2012.

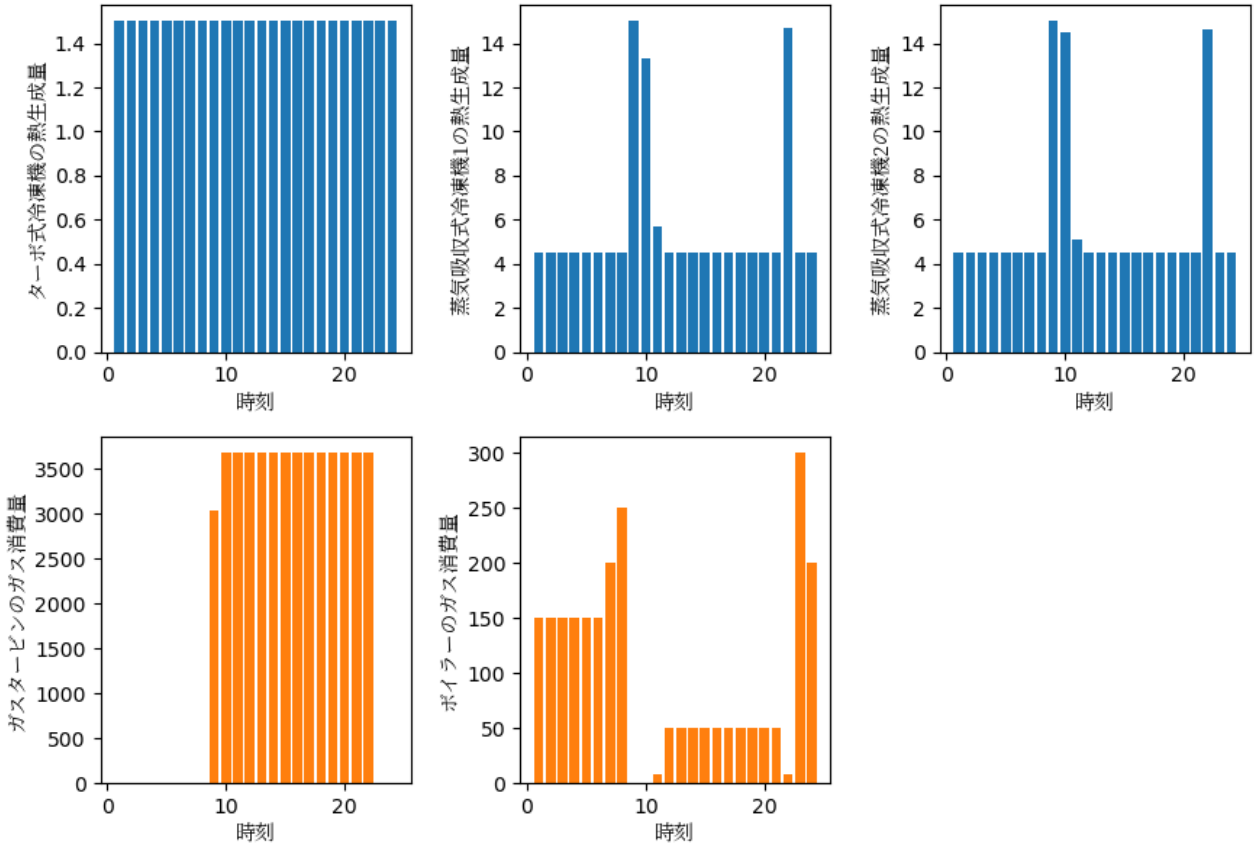

図 4: $\boldsymbol{x}$ の機器ごとの変化