修士論文

ニューラルネットワークによる
ランドスケープの学習を用いた多目的最適化

2022 年 1 月 27 日提出

指導教員　　伊庭 斉志　教授

東京大学大学院情報理工学系研究科　電子情報学専攻

48-206432　須田 瑛斗

概要

最適化問題とは, ある目的関数について最小値または最大値を示す変数の組を求める問題である. 特に目的関数が 2 つ以上あるものを多目的最適化と呼ぶ. この論文ではニューラルネットワークを用いた最適化手法である Deep-Opt を多目的最適化へと拡張することで, Deep-Opt が多目的最適化にも有効であることを示した論文である. 実験には LZ と RE の二種類のベンチマーク問題を用いて, どちらでも高い支配率を持つパレート界を発見することに成功した. 特に RE における一部の問題では NSGAII, MOEA/D, MOEA/D-DRA という多目的最適化において非常に優秀な成績を誇る手法と比較して, 統計的に有意に HyperVolume が良くなるという結果が得られた.

目次

第 1 章　序論 1

第 2 章　関連研究 3
2.1　多目的最適化問題 . . . 3
2.2　Deep-Opt . . . 6
2.3　NSGAII . . . 9
2.4　NSGAIII . . . 12
2.5　MOEA/D . . . 16
2.6　MOEA/D-DRA . . . 19
2.7　ベンチマーク問題 . . . 20

第 3 章　提案手法 33
3.1　提案手法の概要 . . . 33
3.2　Surrogate Model . . . 35
3.3　パラメータレスの手法 . . . 35

第 4 章　実験設定 39
4.1　ベンチマーク問題 . . . 39
4.2　パラメータ設定 . . . 40
4.3　評価手法 . . . 42
4.4　性能比較 . . . 43

第 5 章　実験結果 45
5.1　LZ の実験結果 . . . 45
5.2　RE の実験結果 . . . 45
5.3　Surrogate Model の実験結果 . . . 52

第 6 章　考察 56
6.1　既存手法と比較した実験結果に関する考察 . . . 56
6.2　Surrogate Model についての考察 . . . 58

第 7 章　結論 62
7.1　まとめ 62
7.2　今後の展望 62

参考文献 65

発表文献 68

第 1 章

序論

最適化問題とは, ある目的関数について最小値または最大値を示す変数の組みを求める問題である. 特に, 目的関数が与えられておらず, 入力に対する出力のみが与えられるような場合の最適化問題をブラックボックス最適化 (BBO : Black Box Optimization) と呼ぶ. ブラックボックス最適化は実世界でも多数の応用が存在するが, 実世界の関数は悪条件であることが多く, 様々なアルゴリズムが研究されている.

目的関数が 1 つのものを単目的最適化と言うことに対して, 目的関数が 2 つ以上あるような問題を多目的最適化と言う. 多目的最適化はロケット発射台に設置される空力音響火炎デフレクタの設計 [1] や洋上風力発電所の設計 [2] に応用されている. 図 1.1 は二つの目的関数に対して最小化しようとする問題を想定した図であるが, 一般に F_1 および F_2 がともに最小になるような理想的な解は実行可能解に含まれていない. そこで, 多目的最適化の目標は二つの目的関数がトレードオフな関係になるような領域であるパレート界を探索することである.

単目的最適化問題と比べた多目的最適化問題の難しさは, 目的関数が複数あることによりできるだけ多くのトレードオフな関係にある解を探索することが求められる点である. すなわち, パレート最適界の広がりを確保しながら探索を進める必要がある. 一方, 多目的最適化問題の利点としては, パレート最適界が広がっていた場合にはパレート最適界上にある解から目的に応じた解を選択できるという点が挙げられる. 単目的最適化問題では一つの目的関数について最適化するため最終的な解は一つしか求まらず, その目的関数を最適化することによる他への副作用等は考慮されないが, 多目的最適化問題ではトレードオフな関係にある解が複数求まるため, 一つの目的関数に特化した解からうまくバランスが取れた解まで選択肢に幅が生まれる. これにより利用者の意図に応じた解の選択が可能である.

その多目的最適化問題においては NSGAII[3] や MOEA/D[4] のように様々なアルゴリズムが研究されており, この論文も新たな多目的最適化のアルゴリズムを提案する論文である. この論文では単目的最適化のアルゴリズムとして提案されていた Deep-Opt[5] を多目的最適化に拡張することでニューラルネットワークを用いた最適化が多目的最適化にも応用可能であることを示すことを目標としている. また, 本論文での提案手法ではニューラルネットワークにより設計変数から目的関数値へのランドスケープを学習して同時に最適化を行う. そこでランドスケープの学習を行うメリットとして同時にそのニューラルネットワークによるモデルを Surrogate Model として用いることで最適化を効率的に

行うことが可能であるかということを検証することも本論文の目標である.

この論文の構成としては, 2 章では関連研究である Deep-Opt[5] や多目的最適化に関する既存手法, それからベンチマーク問題について説明した後に 3 章で本論文で提案する手法について説明する. その後, 4 章では本論文で行った実験の設定について説明し, 5 章ではその結果を述べた後に, 6 章で 5 章で紹介する結果に対する考察を述べる. 最後に 7 章では本論文で行った実験から導かれる結論を述べて今後の課題について検討し本論文を締めくくる.

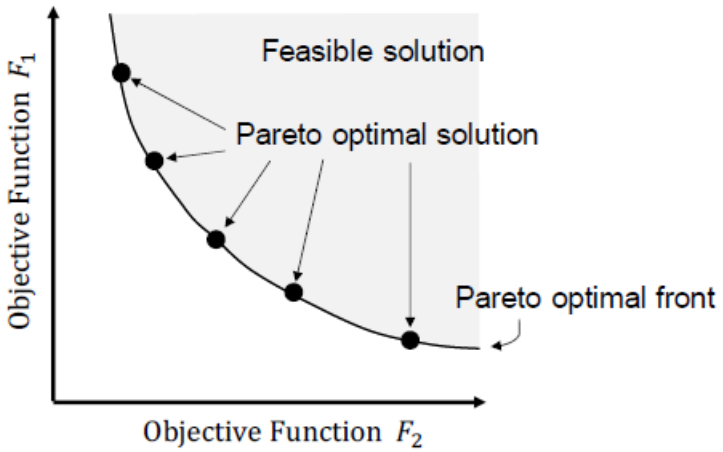


図 1.1: パレート界と実行可能解の様子 ([6] より引用)

第 2 章

関連研究

2.1 多目的最適化問題

2.1.1 多目的最適化の用語についての解説

多目的最適化問題とは目的関数が複数ある問題のことである. その概要は図 1.1 にある通りであるが, ここでは多目的最適化問題における各用語について説明していく. まず初めに多目的最適化問題には実行可能解領域と実行不可能解領域がある. これはその領域に実際に解が存在するかどうかという領域である. 一般的に多目的最適化問題では解の様子を観察するのに設計変数空間は用いずに目的関数空間を用いる. それは図 1.1 のように各軸の値が各目的関数の値になっているようなものである. この目的関数空間において最も良い解はどちらも最小値をとる原点であるがそのような目的関数値を満たすような設計変数は存在しない. このように目的関数空間においてその目的関数値を満たすような設計変数が存在する領域のことを実行可能領域と言い, 存在しない領域のことを実行不可能領域と言う.

次にパレート界について説明する. パレート界の説明の前に非優越解と優越解について説明する必要がある. 優越解とは他の解によって優越される解のことである. ここで優越とはすべての目的関数において他の解に劣っているか等しいことを言う. 一方, 非優越解とは他のどの解にも優越されない解のことである. パレート界とはこの非優越解の集合である. 非優越解の集合内では他の解に優越されることは無いが, 他の解と比べていずれかの目的関数は必ず劣りいずれかの目的関数では勝るトレードオフな関係となる. そのため, パレート界は図 1.1 のような曲線を描く. この論文内では問題にとって最適なものを最適パレート界と呼び単にパレート界と呼んでいる場合にはそれはアルゴリズムによって最適化された疑似最適パレート界のことである. また, 後に RE[7] というベンチマーク関数の説明をする際に疑似最適パレート界という言葉を使うがこれはアルゴリズムによって求められたパレート界と現状発見されている疑似最適パレート界を区別するために使い分けをしている. すなわち, RE を使う文脈で疑似最適パレート界という言葉が出てきた際には現在見つかっている中では最適なパレート界のことを指し, 単にパレート界と書いてある場合にはそれはアルゴリズムによって求められたパレート界のことである. このような使い分けをする理由は, RE は実問題をベンチマーク関数にした問題で

あるため最適なパレート界というものを求めることが不可能であるからである.

2.1.2 多目的最適化の応用例

ここでは多目的最適化の応用例について触れることで, 多目的最適化がどのように役に立つのかということを明らかにしていく.

[8] では MAZDA CX-5, MAZDA6 Wagon, MAZDA3 Hatchback を対象として車のデザインにおける最適化を同時に行っている. ここでは最適化される目的関数は車の総重量と厚さが共通している部品の数である. 総重量は小さい方が良く, 厚さが共通している部品の数は多い方が良い. 車の総重量はそのまま生産コストの削減に繋がる. MAZDA が'SKYACTIVE BODY'[9] というコンセプトを追い求めるにあたって MAZDA CX-5, MAZDA6 Wagon, MAZDA3 Hatchback のような普通車の部品を共通にすることを促進していた. そのためこれらの二つを目的函数としている. [10] では MAZDA が車の総重量の削減を単目的最適化問題として解いたが, 部品の共通化と車の総重量にはトレードオフの関係があるため単目的問題として解くのでは不十分である. そこで [8] では多目的最適化問題として解いている.

結果として, 図 2.1 に示されているように青の MAZDA によってデザインされた車よりも共通する部品の個数と総重量がともに勝る解が見つかった. 図中の赤の点が実行可能解であり, 赤の四角がその内パレート界となっている解であるが, パレート界を形成する解はいくつか存在し, この中から設計者の意図に沿ったものを選択することができる. それは, 共通する部品の数を減らしてでも総重量が少ないデザインを選択するのか, 総重量が多少増えても共通する部品の数が多いデザインを選択するのかということである. 総重量のみを目的関数として単目的最適化問題として解いた場合には, 図 2.1 のパレート界において共通する部品の数が最も少ない解しか見つけることができず, 利用者の意図を必ずしも反映したものが得られるわけではない. このように多目的最適化問題では, 単目的最適化問題では扱い切れないトレードオフな関係を扱うことができ, さらに用途に応じた解の選択を行うことができる.

もう一例として, ニューラルネットワークの構造探索に多目的最適化を適用した例を紹介する. [11] では Convolutional Neural Network(CNN) の構造を多目的に最適化している. ここでの目的関数は認識精度と推論するまでのレイテンシーである. CNN の認識精度を上げるためには構造をある程度複雑にする必要があるが, それは同時に推論にかかる時間の増加に繋がる. 豊富なコンピュータリソースが利用可能で認識精度のみを追い求める場合には多目的最適化は必要ないが, 低いレイテンシーが求められるウェブアプリケーションやコンピュータリソースが限られた環境で動作する必要があるモバイルアプリで画像認識を行いたい場合には, 認識精度を多少犠牲にしても推論のレイテンシーを減らす必要がある. その場合には認識精度と推論のレイテンシーを目的関数とした多目的最適化が有効である. ここでも許容できるレイテンシーに応じて解が選択できるという利点がある.

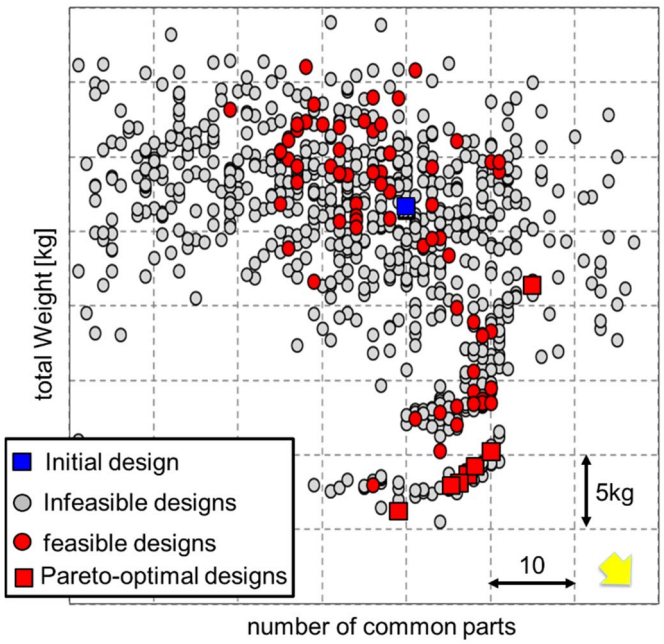


図 2.1: [8] において車のデザインを多目的にして最適化を行った結果 (縦軸が総重量で下に行くほど良く, 横軸が共通する部品の数であり右に行くほど良い. そのためパレート界は右下の領域に分布する.)

2.1.3 多目的最適化の現状

ここまで多目的最適化の利点を述べてきたが, 多目的最適化問題は単目的最適化問題に比べて目的関数が増えているため難しくなっている. それはパレート界の広がりを保ったまま最適化を行う必要があるため, 設計変数空間における最適化と目的関数空間におけるパレート界の広がりという二つの要素を考える必要があるからである. 目的関数がさらに増えたような問題は Many Objective Optimization と呼ばれ, 特に難しいとされている.

現状, 多目的最適化問題を解くようなアルゴリズムは多数研究されている. ただ, 2.7 節で後述するようにそれらの性能評価に用いられるベンチマーク問題のパレート界の形がアルゴリズムの性能に影響を及ぼすことが報告されており [7][12], アルゴリズムを平等に評価することができているとは言い難い状況である. また, ベンチマーク問題は必ずしも実問題の性質を持っているわけではないため実問題でどのアルゴリズムが有用であるかということも評価が難しい. RE[7] は実問題を基にして作られたベンチマーク問題であり, このベンチマーク問題の登場によって様々なアルゴリズムが実問題をベースにして容易に評価されることが可能になったと言える. [7] では様々なアルゴリズムが RE を用いて比較されているがどれも一長一短であり, ここから実問題にも応用可能なアルゴリズムの研究がさらに加速されることが期待されている.

2.2 Deep-Opt

2.2.1 Next-Ascent Stochastic Hillclimbing(NASH)

Deep-Opt の説明に入る前にその中で使われている探索アルゴリズムについて触れる必要がある. next-ascent stochastic hillclimbing(NASH) とは非常に単純な局所探索アルゴリズムである. その疑似コードは Algorithm1 にある. 各世代において, いくつか突然変異を起こすパラメータを選択して 0.75*p〜1.25*p の間の数で置き換えるということを行う. ただし, Algorithm1 における m は 0.02 程度の値である. 置き換えた後の解が基の解よりも適合度がよくなっていれば次の世代では置き換えた解を基にして突然変異を起こす.

Algorithm 1 Next-Ascent Stochastic Hillclimbing(NASH) for Maximization

```
1:  Initialize a random solution c
2:  Best_Evaluation = Evaluate(c)
3:  while termination condition is not met do
4:      c′ = c
5:      Number_Perturbations = Select an Integer from [1,m*|c|]
6:      for i = 1 .. Number_Perturbations do
7:          Select a parameter p from c′ randomly
8:          c′ = Modify p to a random value between [(0.75*p), (1.25*p)]
9:      end for
10:     Candidate_Evaluation = Evaluate(c′)
11:     if Candidate_Evaluation < Best_Evaluation then
12:         c = c′
13:         Best_Evaluation = Candidate_Evaluation
14:     else
15:         Discard c′
16:     end if
17: end while
```

2.2.2 Deep-Opt

Deep-Opt[5] はディープニューラルネットワーク (DNN) を用いて NASH の効率を上昇させた研究である. 疑似コードは Algorithm2 にあり概要図は図 2.2 にある. ディープニューラルネットワークを用いて BBO の解とその適合値のマッピングを学習し, back-drive によりそのネットワーク上で適合度が良いとされる箇所を見つけてその解周辺を NASH により探索している.

Algorithm 2 の 1 行目では最初に DNN がマッピングを学習するための解を生成するが, この方法は

いくつかある. 1 つ目は単に全くランダムに解を生成する方法である. 2 つ目は NASH を行いその際に評価された解を全て保存しておいてそれらを初期解とする方法である. 3 つ目はある程度は全くランダムに生成するが, 残りはその生成された解の周辺を探索することで生成し, それらを初期解とする方法である. 元の論文 [5] では 3 つ目の方法が実験をして良い結果が得られたと書かれているため, 本研究でも 3 つ目の初期化の方法を用いることにする. 疑似コード 5 行目の DNN の学習について, 論文内ではテストセットをランダムに生成して学習の度にテストを行い, テストセットにおける真の適合値と DNN が出力する適合値の間の相関係数が負になると一度 DNN の重みを初期化するということを行っている. また相関係数が下がったタイミングで学習を終了させる.

8 から 13 行目では後述する back-driving のために改良する基となる解を作成する. 今まで最も適合度が高かった解を突然変異させることで今世代で生成する個体数分だけ新たに個体を作り出す. 8 行目では次世代の子個体群を初期化し, 10 行目では突然変異を行い 11 行目では新たに生成された解を次世代の子個体群へと追加している. ただ, このように突然変異にて生み出された解の全てを back-driving にて改良するわけではなく, その一部を改良する. その動作は 13 行目にて行われている.

17 から 19 行目について, back-drive により DNN が高い適合値を示す場所を探し当てる. ここで back-drive とは, 一般に DNN の重みを学習する際に行われる誤差逆伝播と同じことを入力について行うことを指している. すなわち DNN の重みを固定した上で (6 行目), 誤差関数についてその誤差の値が小さくなる方向に最急降下法にて入力を更新していく. 今回出力値は 0.0 から 1.0 にスケーリングされているので最大値の探索の場合には 1.0 を目標値としてそれに近づくように入力を変化させている.

23,24 行目では back-drive により作られた解のうち最も良い解について NASH を行うことでその解周辺を探索している. NASH による探索を行う際に, 作り出された解を全て残しておく. それは次世代における DNN の学習のためである. ただし, その残された解の全てを次世代に受け継ぐわけではなく適合度の高い一部の解のみを次世代に受け渡す (26 から 28 行目). その理由として, 最初期はランダムな広範囲のマッピングを学習するが, 徐々に世代を経るにつれて適合度の高い解がある周辺のマッピングを DNN が学習することで back-drive による探索を効率的に行い, 全体における探索範囲を絞っていくためである. 最後に DNN を学習させるためのデータセットのサイズを一定にするために必要に応じて新たな解が追加されたデータセットを削減する. この削減は適合度は関係なく一番古いものから順番に削除していくという First-In-First-Out の方法を採っている.

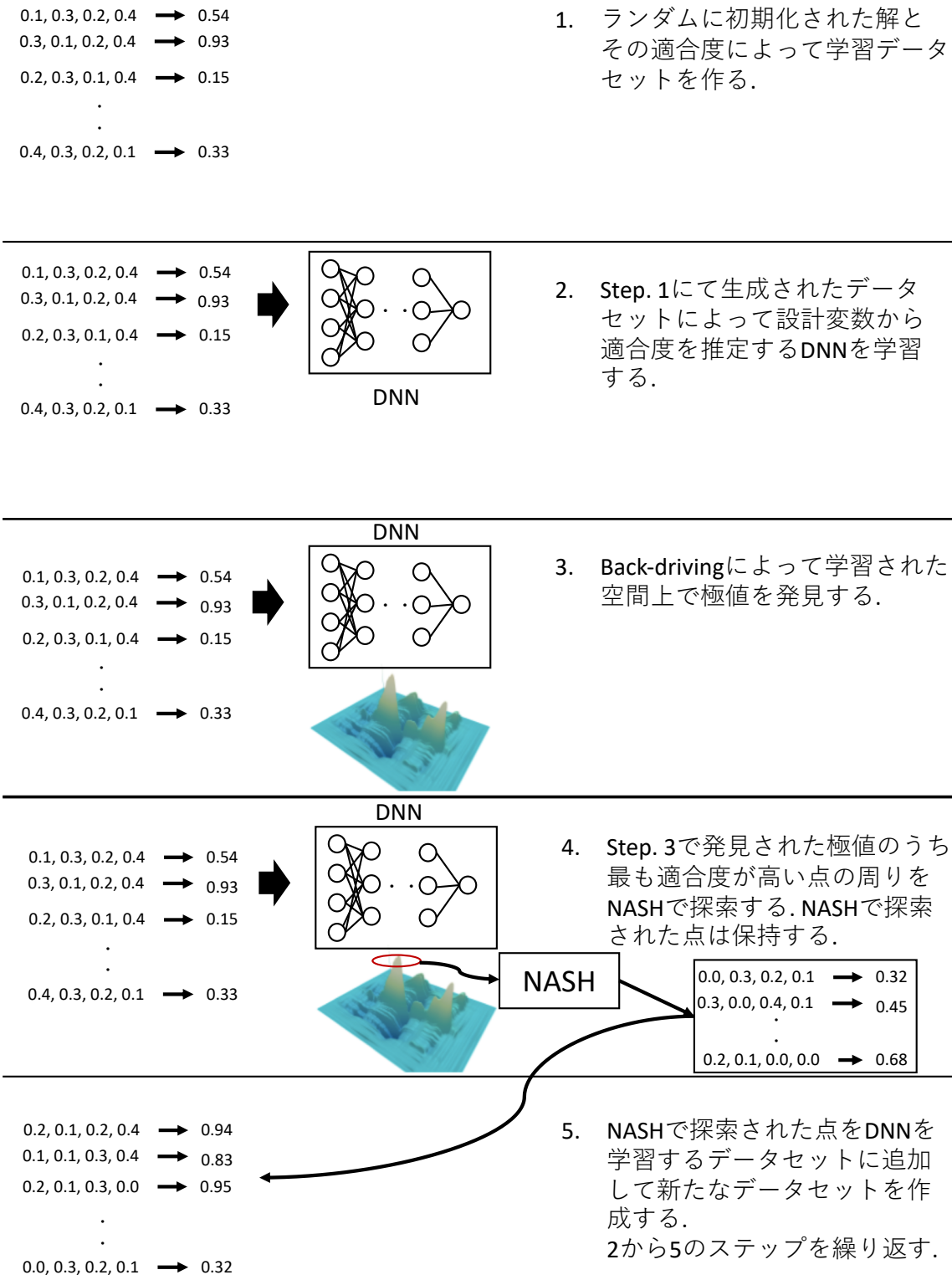


図 2.2: DeepOpt の概要図

Algorithm 2 Deep-Opt

```
Create a set of solutions S
Evaluate all solutions in S
while termination condition is not met do
    Scale all solutions in S to [0.0,1.0]
    Train a DNN to map S → Evaluation(S)
    Freeze the weights of DNN

    Initialize C to be empty
    while |C| < Number-To-Generate-From-Model do
        Create a new solution by perturbing the best solution
        Append the new solution to C
    end while
    Select a fraction of C → C′

    Clamp the network's output to 1.0
    for each d ∈ C′ do
        Initialize the DNN's input with d
        Change the input to d' by back-drive
        Replace d in C with d'
    end for
    Evaluate all solutions in C

    Initialize NASH with a solution which have the highest evaluation in C
    Run NASH and record all of the unique solutions (into set Y ) evaluated by the NASH run.

    Select a subset of the best solutions from Y → Y'
    S ← S + Y'
    prune S if necessary
end while
```

2.3 NSGAII

NSGAII[3] とは高速非優越ソートおよび混雑度ソートを用いて単目的最適化における遺伝的アルゴリズムを多目的最適化問題へと応用した手法である. NSGAII における親集団の選択法の概要図は図 2.3 にある. ここにある解群は今の世代の親集団とその親集団から生み出された子集団を合わせ

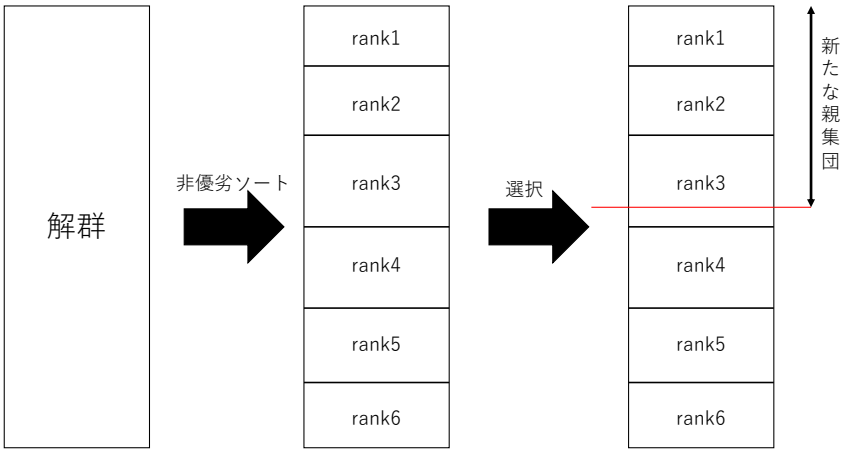


図 2.3: NSGAII における次世代の親集団の選択

たものである. その解群から次世代の親集団を選択するためにまず初めに高速非優越ソートを行う. 高速非優越ソートによって解をランク分けする. その後, 次世代の個体数分だけ rank1 から順番に次世代の親集団とするが, その際に当然 rank の中で分けなければならない場面が出てくる. 具体的に, 図 2.3 では rank3 の間に個体数を区切る必要がある. そこで, rank3 の個体間で優劣をつけるために行われる操作が混雑度ソートである. 混雑度ソートでは目的空間において同じランクで隣り合う解との距離を測ることでその解の混雑度を計算する. 混雑度の値が低いほど解は密集しているという指標になっていて, 解の順位は混雑度 n の値が高く疎なものが上となる. 以降では高速非優越ソートおよび混雑度ソートについてそれぞれ説明していく.

2.3.1 高速非優越ソート

多目的最適化問題において個体の優劣の付け方というのはいくつか存在するが [13], NSGAII は高速非優越ソートによって高速に個体の優劣を付けることができる. 高速非優越ソートとは以下の手順で行われる.

STEP1 それぞれの個体について自分が支配されている個体の数を数えると同時に自分が支配している個体の数を数える

STEP2 どの個体にも支配されていない個体を rank1 として個体群から除く

STEP3 rank を一つ上げて (rank i+1) ひとつ前の rank i の解によって支配されている個体について, 支配されている個体数を一つ減らす

STEP4　今ある個体群のどの個体にも支配されていない個体を rank i+1 として個体群から除く
STEP5　STEP3,4 をすべての個体が取り除かれるまで繰り返す

STEP1 では図 2.4 のように, 自分が支配している解の個数と自分が支配されている解の個数を同時に数えることで, ランクを付けるにあたって何度も支配している解を数える必要が無くなり計算量を削減している. このように数えた後に STEP2 では支配されている解が 0 個のものを rank1 として解群から取り除く. このとき取り除かれた rank1 の解によって支配されている解についてそれぞれ支配されている個数を 1 ずつ減らす. 続いて同様に支配されている個数が 0 個のものを rank2 として取り除き, rank2 の解によって支配されている解についてそれぞれ支配されている個数を 1 ずつ減らす. この一連の操作を全ての解がランク付けされるまで行う.

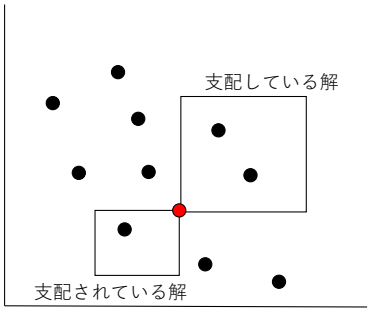


図 2.4: NSGAII において支配解をカウントする際の様子

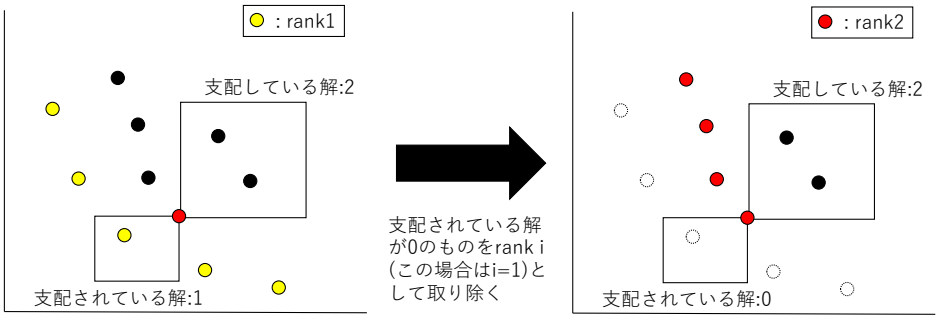


図 2.5: 高速非優越ソートの概要図

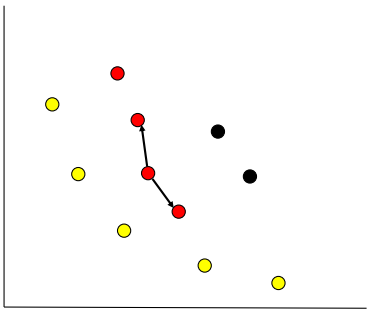


図 2.6: 混雑度ソートの概要図

2.3.2 混雑度ソート

次に混雑度ソートについて説明する. 混雑度ソートとは高速非優越ソートによってランク付けされた解について, 同じランク内の解のなかで優劣を付けるためのものである. 混雑度ソートでは文字通り解の混雑度を計算する. この混雑度が低いものほど解は密集していて解の順位としては低いものとなる. 具体的な混雑度の計算方法は, 図 2.6 にあるように隣り合う解との距離を計算してその和が混雑度となる. この際に目的関数によってソートして隣り合う解を発見する.

2.4 NSGAIII

この節では NSGAIII[14][15] について説明する. NSGAIII とは目的関数が多くなった際により良いパレート界を求めることができるように NSGAII を改良した手法である. 前節で説明した通り NSGAII では大きく分けて個体のランク付・混雑度ソート・交叉および突然変異から成るが NSGAII と NSGAIII で異なる部分は混雑度ソートの部分である. すなわち, 同ランクにおける順位付けの方法を NSGAIII では NSGAII から発展させている. NSGAIII では referene points を用いてパレート界の広がりを保ちながら次世代の親集団を選択している. Reference points を用いた次世代の親集団を選択は主に 4 つの手順からなる. それは, Reference points の設定・正規化・Association・Niche-Preservation である.

2.4.1 Reference points の設定

先述の通り, 同ランクの解の間での順位付けに NSGAII では混雑度ソートを用いていた一方, NSGAIII では reference points を用いてパレート界の探索を行っている. reference points の数 H は p を分割数, M を目的関数の数として式 (2.1) によって計算される. 例えば図 2.7 では目的関数の数が 3 個で分割数 p が 4 の場合の reference points の様子を図示している.

$$H = \begin{pmatrix} M + p - 1 \\ p \end{pmatrix} \tag{2.1}$$

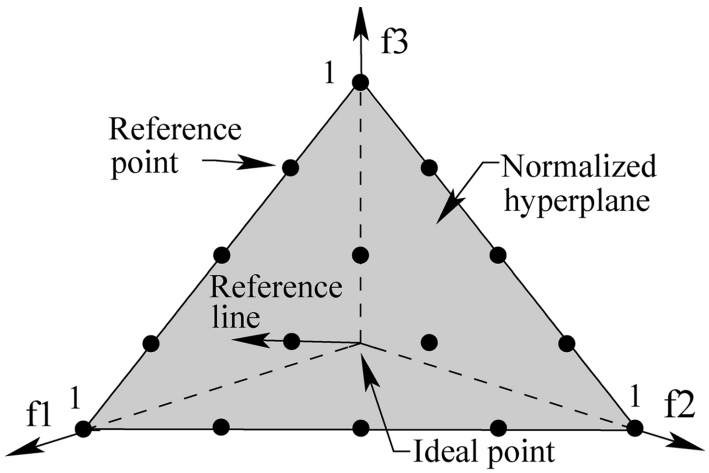


図 2.7: NSGA3 における reference points の様子 ([14] より引用)

2.4.2 正規化

正規化するための最初のステップとして理想点を見つける必要がある. それは各目的関数における最良値を各成分に持つ点である (z_j^{min}=min$_{x \in S_t}$ $f_j(x)$, ただし S_t は t 世代における解集合). 次に $f_j^{'}$(x) = f_j(x)-z_j^{min} という式で $f_j^{'}$(x) を計算する. そして i 番目の目的関数軸に近い重みと f_i(x) を用いて計算されるスカラー関数を最小化するような x を求めてそれを極点 z_i^{max} とする. その極点を用いて M 次元の超平面を作成し, その超平面の i 番目の軸に対する切片を a_i とする (図 2.8 を参照). この a_i を用いて式 (2.2) を用いて正規化された関数値が計算される.

$$f_i^n(x) = \frac{f_i^{'}(x)}{a_i} \tag{2.2}$$

2.4.1 節で設定した reference points は図 2.7 からもわかる通り正規化された超平面上に存在する. もしもその他の方法で reference points を設定した場合には正規化された超平面上には存在しないため式 (2.2) を用いて正規化された超平面上に reference points を設置する.

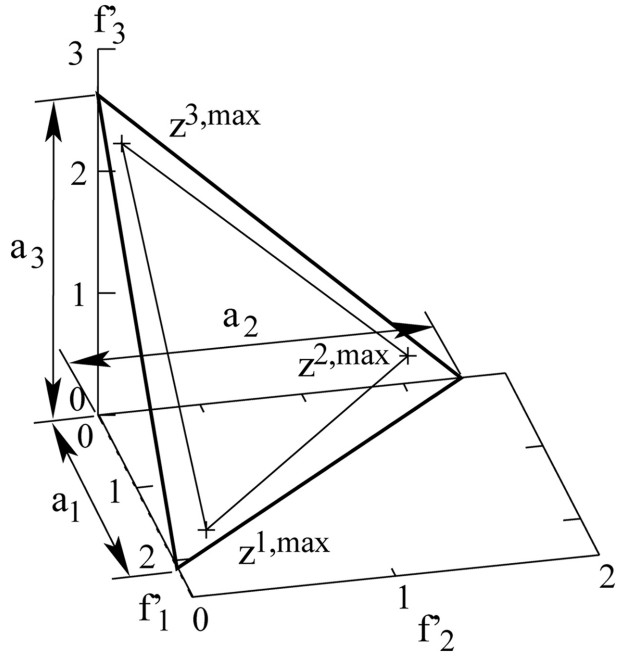


図 2.8: NSGAIII において正規化された超平面を作成する様子 (ここでは目的関数の数は 3 個としている)([14] より引用)

2.4.3 Association

各解を正規化した後にその解を reference points に紐づけをする必要がある. Association とはその作業のことである. 正規化された超平面上にある reference points と原点を結んで reference line を定義する. 各解と reference line との垂直距離を計算して, 最も距離が短い線が繋がっている reference points にその解は紐づけられる. π(s) を紐づけられた reference points, d(s) を s と π(s) の距離である. ただし, s $\in$ S_t で S_t は世代 t における解集合である. この紐づけの様子は図 2.9 にある.

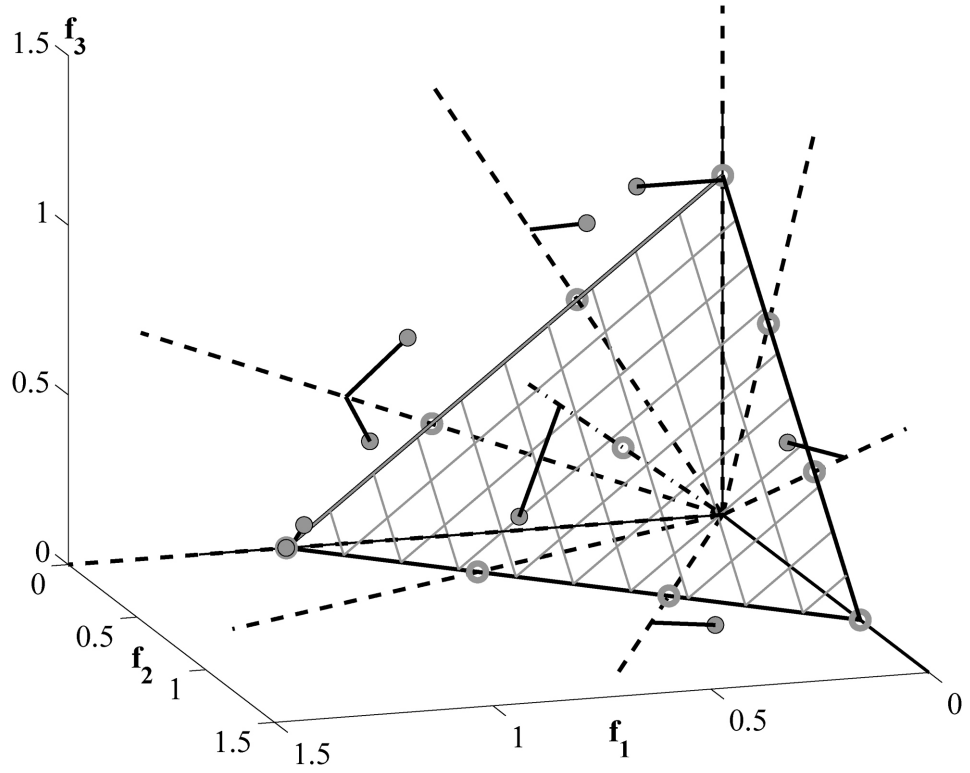


図 2.9: NSGA3 の association において各解を reference points に紐づけする様子 (ここでは目的関数の数は 3 個としている)([14] より引用)

2.4.4 Niche Preservation

NSGAIII では NSGAII と同様に図 2.3 のように次世代の親の選択の際に同ランク間で優劣を付ける必要がある. Niche Preservation とは同ランク間で優劣をつけ次世代の親集団へと追加される個体の選択を行う作業である. 次世代の親集団へと追加される個体の選択を行う必要があるランクを F_l と置く (図 2.3 では F_l= rank 3).

まず初めに, 各 reference points において F_l 未満のランクの解でその reference points に紐づけられている解の個数を数える. j 番目の reference point に紐づけられている F_l 未満のランクの解の個数を ρ_j と置く. J_{min} = { j | argmin$_j$ ρ_j }として J_{min} が複数の要素からなる場合にはそのうちの一つをランダムで選択する. そうして選択されたインデックスを $\bar{j}$ と置く.

$\rho_{\bar{j}}$ が 0 の場合に二つの状況が存在する. 一つは F_l の解で $\bar{j}$ の reference point に紐づけられている解が存在する場合である. この場合にはその解のうち最も reference line との垂直距離が近い解を次世代の親集団へと追加し $\rho_{\bar{j}}$ の値を一つ増やす. もう一つの状況として $\bar{j}$ の reference point に F_l の解が一つも紐づけされていない場合であるが, この場合には今世代において以降の処理でこの reference

point については考えないこととする.

次に $\rho_{\bar{j}}$ が 1 以上の場合, F_l のうち $\bar{j}$ に紐づけされている個体が存在する場合には親集団へと追加する個体の選び方はいくつか存在する. それは $\rho_{\bar{j}}$ が 0 の場合と同様に F_l のうち $\bar{j}$ の reference point に紐づけされている個体の中で最も reference line に近い個体を選択する方法と F_l のうち $\bar{j}$ の reference point に紐づけされている個体の中からランダムに選択する方法である. これらの方法で選択された個体を親集団へと追加した後に $\rho_{\bar{j}}$ の値を一つ増やす.

これらのステップを親集団の個体数が既定の値に達するまで行い親集団の選択を行う.

2.5 MOEA/D

MOEA/D とは多目的最適化問題を多数の単目的最適化問題へと分割して最適化する手法である. 単目的最適化問題へと分割する際には各解に割り当てられた重みベクトルによって計算される集約関数で定義される. 集約関数にはいくつか種類があり, [4] では Weighted Sum (式 (2.3)), Tchebycheff (式 (2.4)), BI (式 (2.5)), PBI (式 (2.6)) が提案されている. MOEA/D の流れは下記の通りである.

STEP1　初期化のステップ

- STEP1.1　外部個体群を初期化
- STEP1.2　一様な重みベクトル $\lambda^1, ..., \lambda^N$ を生成して各重みベクトルに対して T 個の近傍 B(i) = $\{i_1, ... , i_T\}$を定義する.
- STEP1.3　各重みベクトルに対して初期個体 $x^1, ..., x^N$ を生み出し評価する.
- STEP1.4　参照点を z=($z_1, ..., z_m$) と設定 (z_i = min ($F_i(x_1), F_i(x_2), ... , F_i(x_N)$))

STEP2　次世代の解を生成 (i = 1, ..., N)

- STEP2.1　B(i) から二つの個体 (j,k) を選択して x_j, x_k を交叉させ, その後突然変異を施し子個体 y を生成する.
- STEP2.2　必要に応じて参照点を更新する.
- STEP2.3　子個体 y と B(i) 内の個体を集約関数を用いて比較して優越していたら更新.
- STEP2.4　外部個体群に含まれる各個体に y が優越されなかったら外部個体群に y を追加し, 一方 y によって優越される個体が外部個体群に存在したらその個体を外部個体群から削除する.

MOEA/D の概要図は図 2.10 にある. ここでは各解に重みベクトルが割り当てられてその改良の方向が決められていることが図示されている. 図中の () 内の数字は重みベクトルの値で, 矢印がその重みベクトルによって定まる解が更新される方向を示している. このように改良の方向を定めることによって最適化によって得られるパレート界が広がるというメリットがある. 具体的には, (1,0) のような重みベクトルが存在することで最適パレート界において最も目的関数値が小さいところまで探索を広げることが可能である. また同時に集約関数によって多目的最適化問題が単目的最適化へと分割しているため (部分問題への分割), 外部個体群に追加する際には多目的の比較を行わなければならないものの近傍 B 内の解との比較が容易という利点もある.

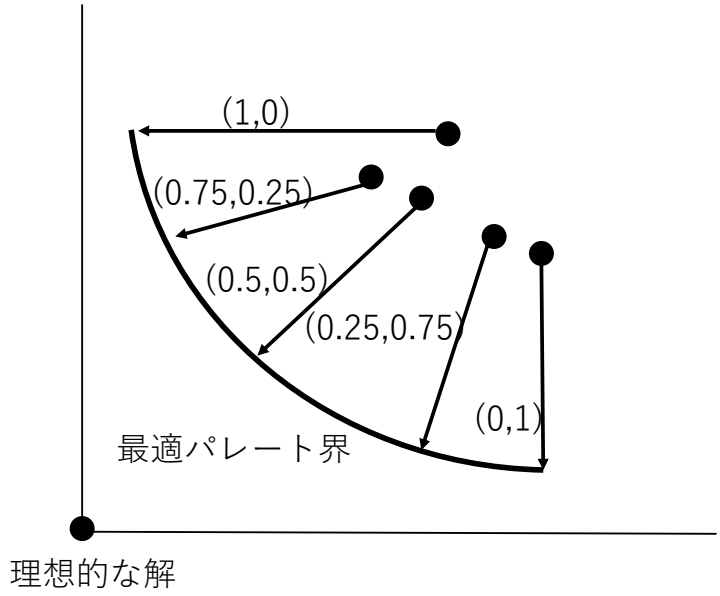


図 2.10: MOEA/D の概要図

集約関数についてそれぞれ詳しく説明していく. まず初めに最も単純な集約関数として Weighted Sum (式 (2.3)) がある. これは重みベクトルと目的関数値の内積を取るだけである. 非常に単純という利点がある一方で, 目的関数値のスケールが目的関数ごとに全く異なる場合には集約関数がうまく解が広がらないという不利点がある. 例えば, 一つ目の目的関数値は 0~10 の値を取る一方で二つ目の目的関数値が 100~1000 の値を取る場合にはたとえ重みベクトルが (0.9, 0.1) であっても二つ目の目的関数値を下げる方が有益と判断されてしまうことがある.

$$\text{minimize} \qquad g = \sum_{i=1}^{N} \lambda_i F_i(x) \tag{2.3}$$

次に Tchebycheff(式 (2.4)) である. Tchebycheff は目的関数値と参照点の差を小さくするような集約関数である. 各パレート最適解には式 (2.4) が最小となるような重みベクトルが存在し, また式 (2.4) の最適解はパレート最適解となる. この関数を使うことの不利点としては連続の MOP について集約関数が滑らかではないことが挙げられる. そのため, 集約関数を微分する必要があるようなアルゴリズムで用いることは難しい.

$$\text{minimize} \qquad g = \max_{1 \le i \le N} \{\lambda_i |F_i(x) - z^*{}_i|\} \tag{2.4}$$

3 つ目の集約関数として紹介するのは Boundary Intersection (BI) (式 (2.5)) である. BI では境界の最上部と直線の交点を求めることを目標としている. この直線が均等に配置されれば交点の集合が最適パレート界になることが期待される. BI の概要図は図 2.11 にある. 式 (2.5) にある d や F(x) は例え

ば二次元を考えた際の幾何学的な意味は図 2.11 にある通りである. $z^* - F(x) = d\lambda$ という制約条件は F(x) が常に λ によって定められる方向を持ち z^* に向かう直線状にあることを表している. ここでこの集約関数の目標は d を小さくしてなるべく F(x) を上方向に上げることである.

BI の欠点として等式の制約条件を処理しなければならない. これを改良したものが Penalty-based Boundary Intersection (PBI) (式 (2.6)) である. 式 (2.6) における θ はペナルティを表すハイパーパラメータである. 図 2.12 には PBI の概要図がある. 図 2.12 にある通り y を F(x) の L に対する射影とすると d_1 は y から参照点 z^* までの距離を表し, d2 は F(x) と y との距離である. PBI の目標は BI と同様で d_1 の最小化であるが d_2 を罰則項として足している. Tchebycheff と比較した際に PBI の利点は二点あげられる. 一つ目は, 目的関数が 3 つ以上の際に Tchebycheff と同じ重みベクトルを用いる場合で尚且つ重みベクトルの数が多くない場合には, Tchebycheff に比べて遥かに均等に分割されるはずである [16]. 二つ目は, x が y を支配している状況を考えて, Tchebycheff を用いる際にはその両者に集約関数が等しくなることがあるが (g^{te}(x) = g^{te}(y)), PBI ではそのようになることは稀である.

PBI の欠点としてはペナルティ項のためのパラメータ θ を適切な値に設定する必要があることである. θ の値によってアルゴリズムの性能が大きく変化することが知られている.

$$\begin{aligned} &\text{minimize} \quad g = d \\ &\text{subject to} \quad z^* - F(x) = d\lambda \end{aligned} \tag{2.5}$$

$$\begin{aligned} &\text{minimize} \quad g = d_1 + \theta d_2 \\ &\text{subject to} \quad d_1 = \frac{||(z^* - F(x))^T \lambda||}{||\lambda||} \\ &\qquad\qquad\quad d_2 = ||F(x) - (z^* - d_1\lambda)|| \end{aligned} \tag{2.6}$$

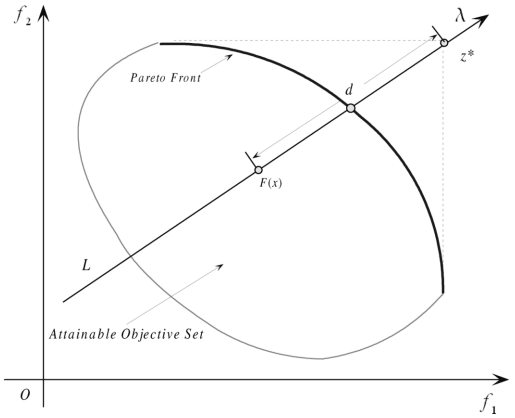


図 2.11: BI の概要図 ([4] より引用)

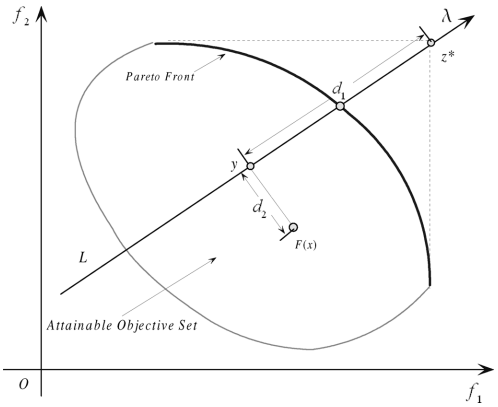


図 2.12: PBI の概要図 ([4] より引用)

2.6 MOEA/D-DRA

MOEA/D Dynamical Resource Allocation(MOEA/D-DRA)[17] は前述した MOEA/D[4] を発展させたものである. MOEA/D では次世代の解を生み出すのに近傍内から選択して生み出し, また比較も近傍内からのみ行っていたが, MOEA/D-DRA では近傍内からだけでなく確率に応じてすべての個体を比較・更新の対象としている. こうすることで集団の多様性を向上させることに成功している. また, MOEA/D では多目的の問題を単目的へと分割した全ての部分問題を平等に扱い探索を行っているが, 部分問題ごとに異なる計算量を持っている可能性があるため MOEA/D-DRA ではそれぞれの部分問題に異なる計算量を割り当てている. MOEA/D-DRA は MOEA/D の持つパラメータに加えて π_i というパラメータを持っている. これは各部分問題についてのパラメータで, i 番目の部分問題にどれだけの計算用を割り当てるかというものを決定するパラメータとなっている. MOEA/D-DRA は以下のステップで行われる.

STEP1　初期化

STEP1.1　一様な重みベクトル $\lambda^1, ..., \lambda^N$ を生成して, すべての重みベクトル間のユークリッド距離を計算し各重みベクトルに対して T 個の近傍 B(i) = {i_1, ... , i_T}を定義する.

STEP1.2　各重みベクトルについて初期個体をランダムに生成する.

STEP1.3　参照点を z=(z_1, ..., z_m) と設定 (z_i = min ($F_i(x_1)$, $F_i(x_2)$, ... , $F_i(x_N)$))

STEP1.4　すべての i に対して π_i=1 とする (i = 1, ... , N). 同時に世代数を管理する変数 gen について gen=0 とする.

STEP2　選択

STEP2.1　更新する個体群 I を初期化した後に各目的関数において最も良かった値をもつ個体を I に追加

STEP2.2　π_i を用いて 10-トーナメント選択を行い I に追加する. この選択は $\frac{N}{5}$- m 回行う.

STEP3　更新 (i∈ I)

STEP3.1　[0,1] の一様分布から乱数 rand を生成しその値によって更新範囲を式 (2.7) のように決定する.

$$\mathrm{P} = \begin{cases} B(i) & (\text{if} \quad \text{rand} < \delta) \\ \{1, ..., N\} & (\text{otherwise}) \end{cases} \tag{2.7}$$

STEP3.2　r_1=i として r_2, r_3 をランダムに P から選択する. r_1, r_2, r_3 に相当する個体 x^{r_1}, x^{r_2}, x^{r_3} を用いて差分進化によって次世代の解 $\bar{y}$ を生成する. 差分進化によって生み出された解に突然変異を施すことで最終的な次世代の解 y を生み出す.

STEP3.3　y の各要素が設計変数の範囲から外れていた場合はその外れている要素を設計変数の範囲内からランダムに数値を選択し置き換える.

STEP3.4　j=1, ..., m について z_j<F_j(y) であれば z_j=F_j(y) とする.

STEP3.5　c=0 として, n_r を予め設定してあるパラメータとして以下の操作を行う.

(1) c=n_r または P が空ならば終了して STEP4 へ. そうでないならば P から一つインデックス j を選択する.

(2) $g(y|\lambda^j, z) \leq g(x^j|\lambda^j, z)$ ならば x^j=y および c=c+1 とする.

(3) j を P から取り除き (1) へと戻る.

STEP4 終了

終了条件を満たしていたら x^i (i=1, ..., N) および $F(x^i)$ (i=1, ..., N) を出力する. そうでないならば STEP5 へと進む.

STEP5 パラメータ更新

gen=gen+1 として, gen が 50 で割り切れるならば Δ_i を式 (2.8) を用いて計算し式 (2.9) によって π_i を更新する. ただし, F_i^{new} は i 番目の集約関数の現在の値で F_i^{new} は 50 世代前の値である (i=1, ... , N). Δ_i が 0.001 を下回った場合, π_i を小さくする.

$$\Delta_i = \frac{F_i^{old} - F_i^{new}}{F_i^{old}} \tag{2.8}$$

$$\pi_i = \begin{cases} 1 & (\text{if} \quad \Delta_i > 0.001) \\ (0.95 + 0.05\dfrac{\Delta_i}{0.001})\pi_i & (\text{otherwise}) \end{cases} \tag{2.9}$$

STEP2 における 10-トーナメント選択とは, 10 個の個体をランダムに選び最も π_i が大きい個体を選択しそのインデックスを I に追加する. STEP3.2 における差分進化による解の生成は MOEA/D-DE[18] で行われている差分進化と同じである. 差分進化における $\bar{y}$ の k 番目成分 $\bar{y}_k$ は式 (2.10) によって計算される. ただし F と CR はハイパーパラメータである. さらに $\bar{y}$ から突然変異を起こして y を生み出す際の計算は式 (2.11) によって行われる. ここでも k 番目の成分についてそれぞれ計算を行う. ただし, p_m は突然変異率で a_k, b_k は k 番目の設計変数の下限と上限である. 突然変異の計算で用いられる σ_k は式 (2.12) によって計算される. ただし, rand は [0,1] の一様分布からサンプリングされた値で η も新たなハイパーパラメータである.

$$\bar{y}_k = \begin{cases} x_k^{r_1} + F(x_k^{r_2} - x_k^{r_3}) & \text{with probability CR} \\ x_k^{r_1} & \text{with probability } 1 - \text{CR} \end{cases} \tag{2.10}$$

$$y_k = \begin{cases} \bar{y}_k + \sigma_k(b_k - a_k) & \text{with probability } p_m \\ \bar{y}_k & \text{with probability } 1 - p_m \end{cases} \tag{2.11}$$

$$\sigma_k = \begin{cases} (2 \times rand)^{\frac{1}{1+\eta}} - 1 & \text{if} \quad rand < 0.5 \\ 1 - (2 - 2 \times rand)^{\frac{1}{1+\eta}} & \text{otherwise} \end{cases} \tag{2.12}$$

2.7 ベンチマーク問題

ここでは二つのベンチマーク問題について説明する. 一つは LZ[18] でもう一つは RE[7] である.

2.7.1 LZ

LZ は設計変数空間において最適パレート界を複雑になるように設計された多目的最適化問題におけるベンチマーク問題である. LZ には F1 から F9 の 9 つの問題があり, F6 のみ目的関数の数は 3 個でその他は 2 個である.

- **F1**
  - 二つの目的関数 f_1, f_2 はそれぞれ式 (2.13), 式 (2.14) である. ただしそれらの式に登場する J_1 および J_2 は式 (2.15) にある通りである. また, n は設計変数の次元数である.
  - 設計変数の範囲は $[0,1]^n$ である.
  - この問題における最適パレート界の設計変数は式 (2.16) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - x_i^{0.5(1.0+\frac{3(j-2)}{n-2})}\right)^2 \tag{2.13}$$

$$f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - x_i^{0.5(1.0+\frac{3(j-2)}{n-2})}\right)^2 \tag{2.14}$$

$$\begin{cases} J_1 = \{j | j \equiv 0 \pmod 2 \quad \text{and} \quad 2 \le j \le n\} \\ J_2 = \{j | j \equiv 1 \pmod 2 \quad \text{and} \quad 2 \le j \le n\} \end{cases} \tag{2.15}$$

$$x_j = x_1^{(1.0+\frac{3(j-2)}{n-2})} \quad j = 2, 3, ..., n \tag{2.16}$$

- **F2**
  - 二つの目的関数 f_1, f_2 はそれぞれ式 (2.17), 式 (2.18) である. ただしそれらの式に登場する J_1 および J_2 は F1 と同様で式 (2.15) にある通りである. また, n は設計変数の次元数である.
  - 設計変数の範囲は $[0,1]\times[-1,1]^{n-1}$ である. すなわち x_1 の範囲は [0,1] でその他の変数の範囲は [-1,1] である.
  - この問題における最適パレート界の設計変数は式 (2.19) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.17}$$

$$f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.18}$$

$$x_j = \sin(6\pi x_1 + \frac{j\pi}{n}) \quad j = 2, 3, ..., n \tag{2.19}$$

- **F3**
  - 二つの目的関数 f_1, f_2 はそれぞれ式 (2.20), 式 (2.21) である. ただしそれらの式に登場する J_1 および J_2 は F1 と同様で式 (2.15) にある通りである. また, n は設計変数の次元数である.

– 設計変数の範囲は [0,1]×[-1,1]$^{n-1}$ である. すなわち x_1 の範囲は [0,1] でその他の変数の範囲は [-1,1] である.
– この問題における最適パレート界の設計変数は式 (2.22) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 0.8x_1\cos(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.20}$$

$$f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.21}$$

$$x_j = \begin{cases} 0.8x_1\cos(6\pi x_1 + \frac{j\pi}{n}) & j \in J_1 \\ 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{n}) & j \in J_2 \end{cases} \tag{2.22}$$

• **F4**

– 二つの目的関数 f_1, f_2 はそれぞれ式 (2.23), 式 (2.24) である. ただしそれらの式に登場する J_1 および J_2 は F1 と同様で式 (2.15) にある通りである. また, n は設計変数の次元数である.
– 設計変数の範囲は [0,1]×[-1,1]$^{n-1}$ である. すなわち x_1 の範囲は [0,1] でその他の変数の範囲は [-1,1] である.
– この問題における最適パレート界の設計変数は式 (2.25) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 0.8x_1\cos(\frac{6\pi x_1 + \frac{j\pi}{n}}{3})\right)^2 \tag{2.23}$$

$$f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.24}$$

$$x_j = \begin{cases} 0.8x_1\cos(\frac{6\pi x_1 + \frac{j\pi}{n}}{3}) & j \in J_1 \\ 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{n}) & j \in J_2 \end{cases} \tag{2.25}$$

• **F5**

– 二つの目的関数 f_1, f_2 はそれぞれ式 (2.26), 式 (2.27) である. ただしそれらの式に登場する J_1 および J_2 は F1 と同様で式 (2.15) にある通りである. また, n は設計変数の次元数である.
– 設計変数の範囲は [0,1]×[-1,1]$^{n-1}$ である. すなわち x_1 の範囲は [0,1] でその他の変数の範囲は [-1,1] である.
– この問題における最適パレート界の設計変数は式 (2.28) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \left(0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1\right)\cos(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.26}$$

$$f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \left(0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1\right)\sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.27}$$

$$x_j = \begin{cases} (0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1)\cos(6\pi x_1 + \frac{j\pi}{n}) & j \in J_1 \\ (0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1)\sin(6\pi x_1 + \frac{j\pi}{n}) & j \in J_2 \end{cases} \tag{2.28}$$

- **F6**
  - 3 つの目的関数 f_1, f_2, f_3 はそれぞれ式 (2.29), 式 (2.30), 式 (2.31) である. ただしそれらの式に登場する J_1, J_2 および J_3 は式 (2.32) にある通りである. また, n は設計変数の次元数である.
  - 設計変数の範囲は $[0,1]^2\times[-2,2]^{n-2}$ である. すなわち x_1 および x_2 の範囲は [0,1] でその他の変数の範囲は [-2,2] である.
  - この問題における最適パレート界の設計変数は式 (2.33) にある通りである.

$$f_1 = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.29}$$

$$f_2 = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.30}$$

$$f_3 = \sin(0.5x_1\pi) + \frac{2}{|J_3|}\sum_{j\in J_3}\left(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.31}$$

$$\begin{cases} J_1 = \{j | j \equiv 1 \pmod 3 \quad \text{and} \quad 3 \le j \le n\} \\ J_2 = \{j | j \equiv 2 \pmod 3 \quad \text{and} \quad 3 \le j \le n\} \\ J_3 = \{j | j \equiv 0 \pmod 3 \quad \text{and} \quad 3 \le j \le n\} \end{cases} \tag{2.32}$$

$$x_j = 2x_2\sin(2\pi x_1 + \frac{j\pi}{n}) \quad j = 3, 4, ..., n \tag{2.33}$$

- **F7**
  - 二つの目的関数 f_1, f_2 はそれぞれ式 (2.34), 式 (2.35) である. ただしそれらの式に登場する J_1 および J_2 は F1 と同様で式 (2.15) にある通りであり, y_j は式 (2.36) にある通りである. また, n は設計変数の次元数である.
  - 設計変数の範囲は $[0,1]^n$ である.
  - この問題における最適パレート界の設計変数は式 (2.37) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(4y_j^2 - \cos(8y_j\pi) + 1.0\right) \tag{2.34}$$

$$f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(4y_j^2 - \cos(8y_j\pi) + 1.0\right) \tag{2.35}$$

$$y_j = x_j - x_1^{0.5(1.0+\frac{3(j-2)}{n-2})} \quad j = 2, 3, ..., n \tag{2.36}$$

$$x_j = x_1^{0.5(1.0+\frac{3(j-2)}{n-2})} \quad j = 2, 3, ..., n \tag{2.37}$$

- **F8**

– 二つの目的関数 f_1, f_2 はそれぞれ式 (2.38), 式 (2.39) である. ただしそれらの式に登場する J_1 および J_2 は F1 と同様で式 (2.15) にある通りであり, y_j は式 (2.40) にある通りである. また, n は設計変数の次元数である.
– 設計変数の範囲は $[0,1]^n$ である.
– この問題における最適パレート界の設計変数は式 (2.41) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\left(4\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_1}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 2\right) \tag{2.38}$$

$$f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\left(4\sum_{j\in J_2} y_j^2 - 2\prod_{j\in J_2}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 2\right) \tag{2.39}$$

$$y_j = x_j - x_1^{0.5(1.0+\frac{3(j-2)}{n-2})} \quad j = 2, 3, ..., n \tag{2.40}$$

$$x_j = x_1^{0.5(1.0+\frac{3(j-2)}{n-2})} \quad j = 2, 3, ..., n \tag{2.41}$$

• **F9**
– 二つの目的関数 f_1, f_2 はそれぞれ式 (2.42), 式 (2.43) である. ただしそれらの式に登場する J_1 および J_2 は F1 と同様で式 (2.15) にある通りである. また, n は設計変数の次元数である.
– 設計変数の範囲は $[0,1]\times[-1,1]^{n-1}$ である. すなわち x_1 の範囲は [0,1] でその他の変数の範囲は [-1,1] である.
– この問題における最適パレート界の設計変数は式 (2.44) にある通りである.

$$f_1 = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.42}$$

$$f_2 = 1 - x_1^2 + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \tag{2.43}$$

$$x_j = \sin(6\pi x_1 + \frac{j\pi}{n}) \quad j = 2, 3, ..., n \tag{2.44}$$

LZ の最適パレート界における x_1,x_2 および x_3 の様子は図 2.13 にある. LZ 以前のベンチマーク問題である ZDT[19] や DTLZ[20] に比べるとこのパレート界は非常に複雑になっている. ZDT における最適パレート界の設計変数は式 (2.45) にある通りで, DTLZ における最適パレート界の設計変数は式 (2.46) にある通りである. 両式からもわかる通り ZDT および DTLZ は実問題にはあり得ないほど設計変数における最適パレート界が単純になっていた. 一方で, LZ では F1 から F9 の式および図 2.13 からもわかる通り ZDT や DTLZ よりも目に見えて複雑なベンチマーク問題になっていることがわかる. また, F6 を除き F1 から F8 までは目的関数空間において全て同じパレート界の形をしていてそれは図 2.14 である. 目的関数の数が二つの場合では F9 のみ形が異なり, それは図 2.15 のようになっている. 図 2.14 において一部が離散的なようになっているがこれはグラフを描画する上で離散的なようになってしまっただけであり, 実際の最適パレート界は連続である.

図 2.16 には F6 の最適パレート界を表示している. これは三角形を球のように膨らませたような形になっている. ここでは図示の都合上離散的な面となっているが実際の最適パレート界は連続した面になっている.

$$\begin{aligned} 0 \le x_1 \le 1 \\ x_2 = x_3 = \ldots = x_n = 0 \end{aligned} \tag{2.45}$$

$$\begin{aligned} 0 \le x_1, x_2 \le 1 \\ x_3 = x_4 = \ldots = x_n = 0 \end{aligned} \tag{2.46}$$

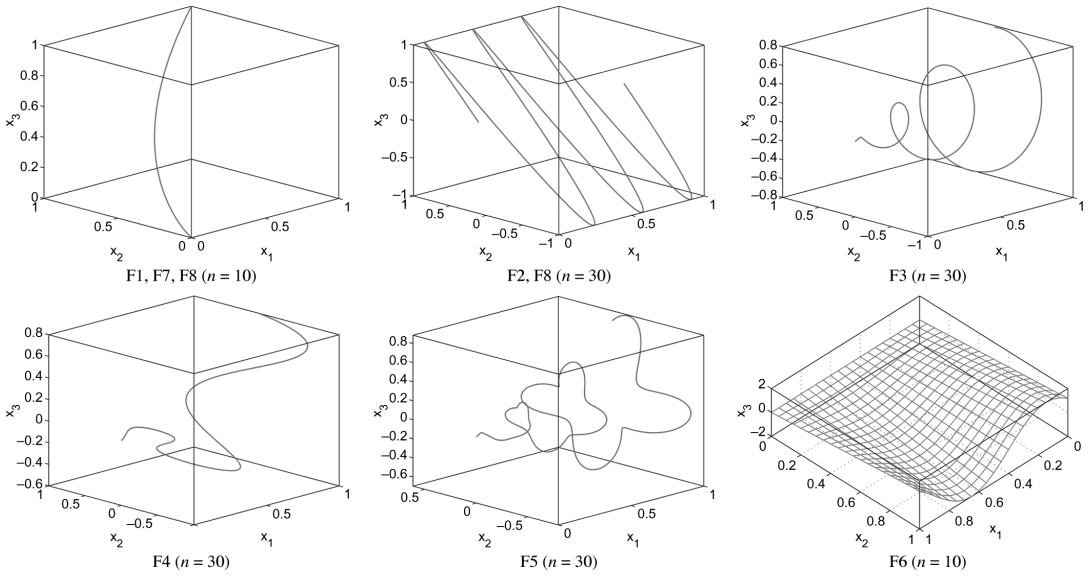


図 2.13: LZ の最適パレート界における x_1,x_2 および x_3 の様子

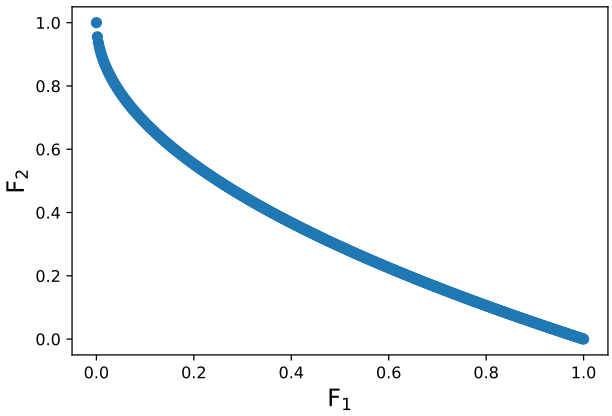


図 2.14: F1 から F8 のパレート界の様子 (F6 を除く)

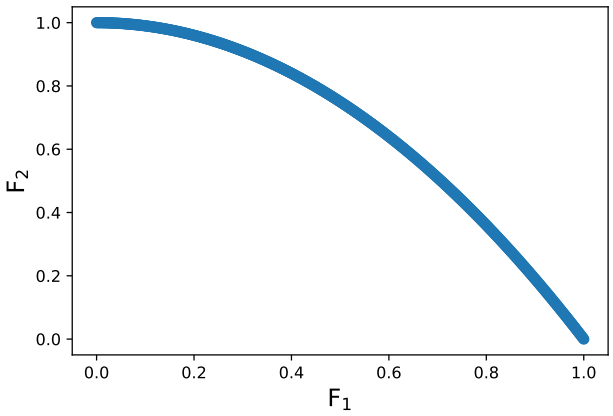


図 2.15: F9 のパレート界の様子

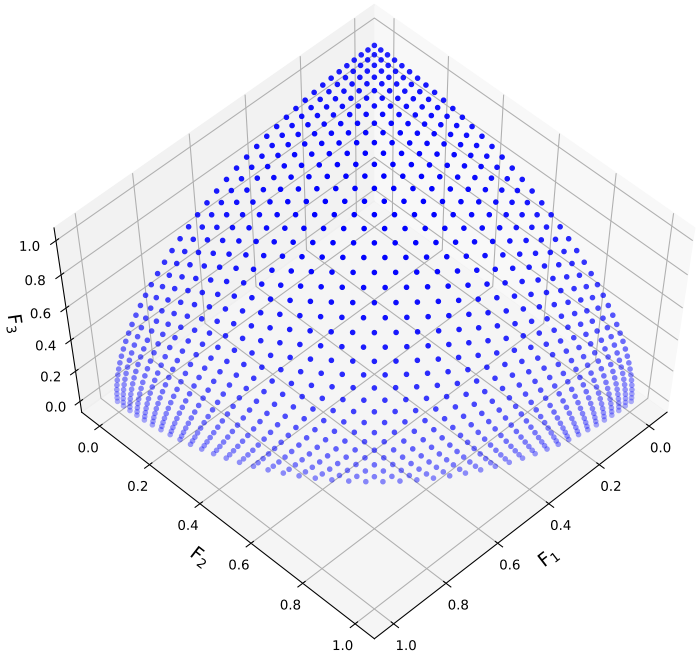


図 2.16: F6 のパレート界の様子

2.7.2 RE

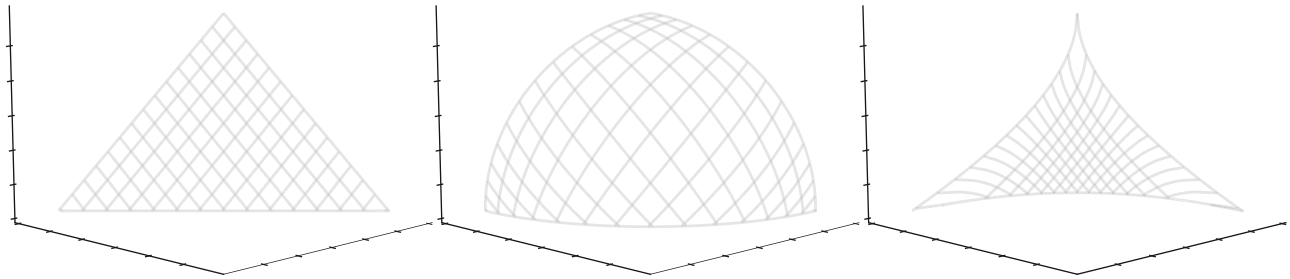
RE は実世界の問題を基にして作られたベンチマーク問題である. ZDT[19], DTLZ[20], LZ[18] のベンチマーク問題は 3 つの大きな利点がある. 一つ目はどのプログラミング言語でも実装が容易であり, ベンチマーク問題を使用するのに大きな労力を必要としないことである. 二つ目は, 最適パレート界の様子が既知であることである. 三つ目は, 設計変数および目的関数の次元数を容易に変化させることができることである. 一つ目の利点について, ベンチマーク問題として利用するためには容易に利用ができてアルゴリズム間での性能を比較する必要があるため不可欠なことである. 二つ目について, 最適パレート界の様子が既知であることの利点は, アルゴリズムがどれほどその最適パレート界に近づけたかという指標によって性能を測ることができることである. これによって様々な視点からアルゴリズムを評価することができる. 三つ目について, 設計変数および目的関数の数を自由に設定できることによりその数に応じた振る舞いを観察することが可能である. 人手で作成されたベンチマーク問題はこれらの 3 つの利点を持っている一方で当然欠点も持っている. それは, 実世界の問題には無い特性を持っていることである [12][21]. 例えば, [12] および [7] 内で言及されていることとして MOEA/D[4] や NSGAIII[14][15] が DTLZ[20] や WFG[22] の目的関数が 3 以上の問題においても高い成績が出るのはそれらの問題で最適パレート界が三角形の形をしていることと関連がある. そのため, アルゴリズムをそのような最適パレート界を持つベンチマーク問題で比較する際には過大評価もしくは過小評価に繋がる可能性がある.

RE では容易に実装及び使用ができるようにしながら実世界の問題をベンチマーク問題へと落とし込むことでそれらの問題点を解消している. 実際に RE は著者の github[23] には C, Python, Matlab, Java の 4 つの言語で実装されている. RE に含まれる問題の詳細は表 2.1 にある. RE はすべて基となっている問題があるため表にはその基となる問題も共に表記している. 設計変数のタイプは連続する実数値の最適化 (Continuous), 整数値を取る最適化問題 (Integer), もしくは整数または離散的な値を取る変数と連続的な値を取る変数の両方がある最適化問題 (Mixed) の 3 種類がある.

近似最適パレート界のタイプは目的関数が 3 個以上の場合には多くは Unknown となっていて, 2 個の場合は凸面になっているものと凹面と凸面が組み合わさっているもの (Mixed) がある. また, 関数によっては近似最適パレート界が連続ではない場合も存在する. 図 2.18 から 2.22 には RE21 から RE25 の近似最適パレート界の様子が図示されている. ここまで最適パレート界ではなく近似最適パレート界と書いているが, 図示されているものは必ずしも最適パレート界ではないことに注意する必要がある. 実世界の問題では最適パレート界を求めることは不可能に近いため図示されているものは近似最適パレート界である. これらの図を見てもわかる通り, LZ の最適パレート界 (図 2.14, 2.15) と比較するとその形は単純な凸面や線形な形ではないのに加えて, F_1 と F_2 のオーダーが大きく異なるという特徴をもつ問題が存在することが挙げられる. 例えば, RE21 では F_1 が $[1.25\times10^{3}, 3.0\times10^{3}]$ の間の値を取ることに対して F_2 が $[2.8\times10^{-4}, 4.0\times10^{-2}]$ の間の値を取る. また, 関数ごとに全く異なる値を取る. 近似最適パレート界の形も RE21 のように凸面の形をしているものもあれば凹面, 凸面および線形が組み合わさった形をしているものもある. RE23, RE25 は図 2.20, 2.22 からも見てわかる通り, 連続ではなくて離散的なパレート界である. このように RE は LZ とは異なり, 問題ごとに全く異なる特徴をもつ.

目的関数が 3 個以上の場合に RE36 以外のパレート界の様子が Unknown となっている理由は実世界の問題は複雑なパレート界を持つため図示してその様子を掴むことが難しいからである. RE31 から RE37 の疑似最適パレート界の様子は図 2.23 から図 2.29 にそれぞれある. これらの図を見ても疑似最適パレート界の特徴を掴むことは難しいことがわかる. ただ, 図 2.17 にあるような三角形の形をしたパレート界ではないことや一部のパレート界で連続していないことがわかる. また, 各目的関数のオーダーも二次元の際と同様で全く異なることが見て取れる. 具体的には, RE31 において F_1 は [0, 500] に分布していることに対して, F_2 および F_3 はそれぞれ [0.2×10^6, 0.2×10^8], [0.0, 2.0×10^7] の間に疑似最適パレート界は分布していることがわかる. また, 疑似最適パレート界の形も LZ の F6 とは異なる. F6 の最適パレート界は図 2.16 にように三角形の形をしているが RE ではどの関数も疑似最適パレート界が三角形の形をしていない. このことから一部のアルゴリズムが過大もしくは過小に評価されることがなくなることにつながる.

RE では設計変数の次元数が小さいものが多い. その理由は設計変数の次元数が大きい問題はシミュレーションによって解の適合度を得る問題が多く定式化することが難しいからである. RE においても RE34, RE37, RE41 そして RE91 はシミュレーションからサンプル点を抽出して関数近似を行っている. したがって, RE におけるこれらの問題は実際の問題とは全く同じ問題ではないということに注意する必要がある. また, これらの問題は実問題が基となってはいるが実際の問題とは異なる可能性があるため人工的に作られた問題とみなすこともできてしまう.



(a) Triangular-linear (b) Triangular-concave (c) Triangular-convex

図 2.17: 三角形の形をした最適パレート界

RE における問題名	基となる問題名	目的関数の数	設計変数の次元数	設計変数のタイプ	近似最適パレート界のタイプ
RE21	Four bar truss design[24]	2	4	Continuous	Convex
RE22	Reinforced Concrete beam design[25]	2	3	Mixed	Mixed
RE23	Pressure vessel design[26]	2	4	Mixed	Mixed, Disconnected
RE24	Hatch cover design[25]	2	2	Continuous	Convex
RE25	Coil compression spring design[27]	2	3	Mixed	Mixed, Disconnected
RE31	Two bar truss design[28]	3	3	Continuous	Unknown
RE32	Welded beam design[29]	3	4	Continuous	Unknown
RE33	Disk brake design[29]	3	4	Continuous	Unknown
RE34	Vehicle crashworthiness design[30]	3	5	Continuous	Unknown
RE35	Speed reducer design[31]	3	7	Mixed	Unknown
RE36	Gear train design[32]	3	4	Integer	Concave, Disconnected
RE37	Rocket injector design[33]	3	4	Continuous	Unknown
RE41	Car slide impact design[15]	4	7	Continuous	Unknown
RE42	Conceptual marine design[34]	4	6	Continuous	Unknown
RE61	Water resource planning[35]	6	3	Continuous	Unknown
RE91	Car cab design[14]	9	7	Continuous	Unknown

表 2.1: RE に含まれる問題の詳細

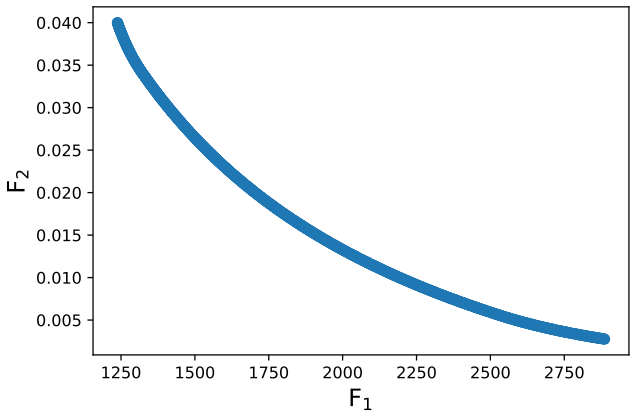


図 2.18: RE21 のパレート界の様子

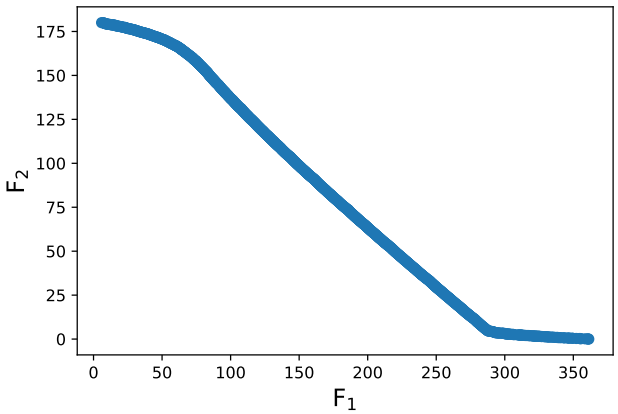


図 2.19: RE22 のパレート界の様子

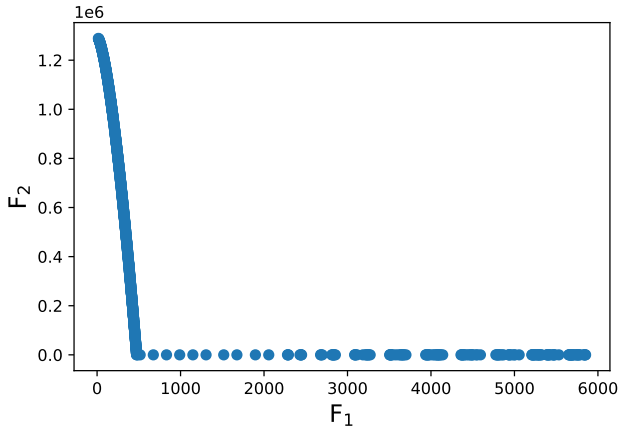


図 2.20: RE23 のパレート界の様子

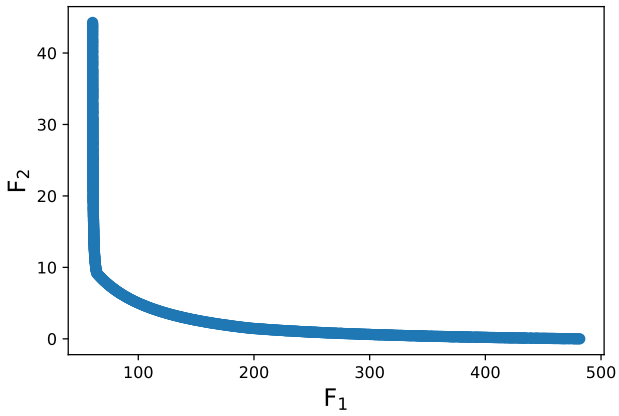


図 2.21: RE24 のパレート界の様子

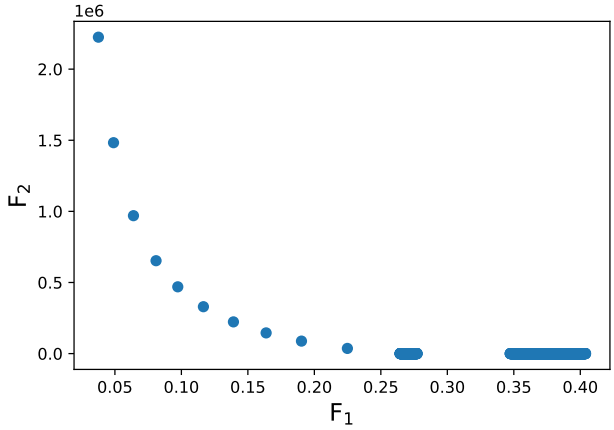


図 2.22: RE25 のパレート界の様子

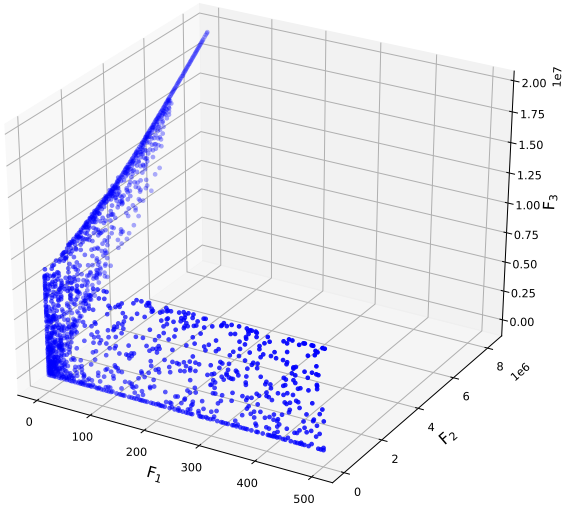


図 2.23: RE31 のパレート界の様子

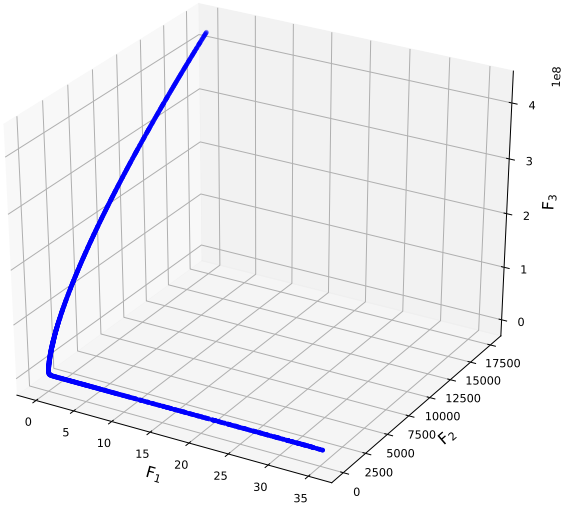


図 2.24: RE32 のパレート界の様子

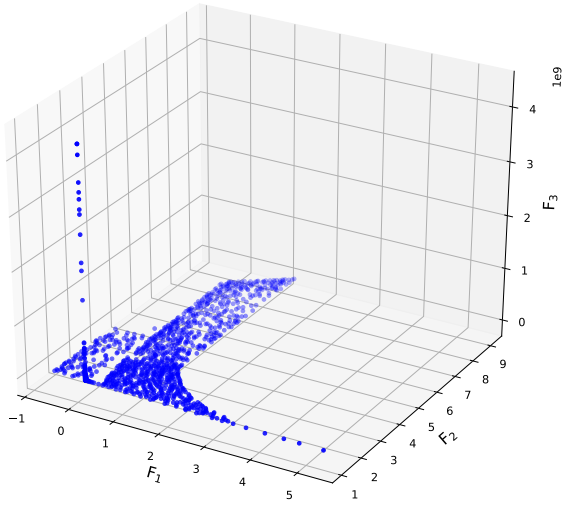


図 2.25: RE33 のパレート界の様子

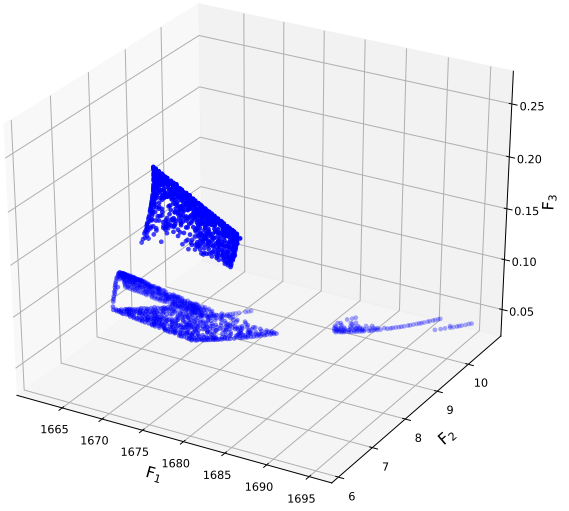


図 2.26: RE34 のパレート界の様子

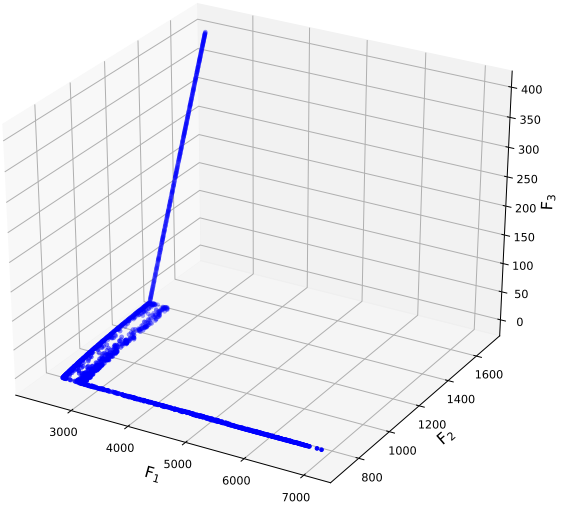


図 2.27: RE35 のパレート界の様子

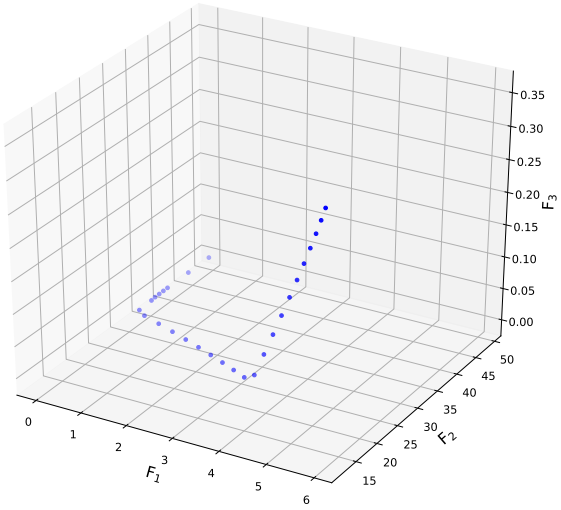


図 2.28: RE36 のパレート界の様子

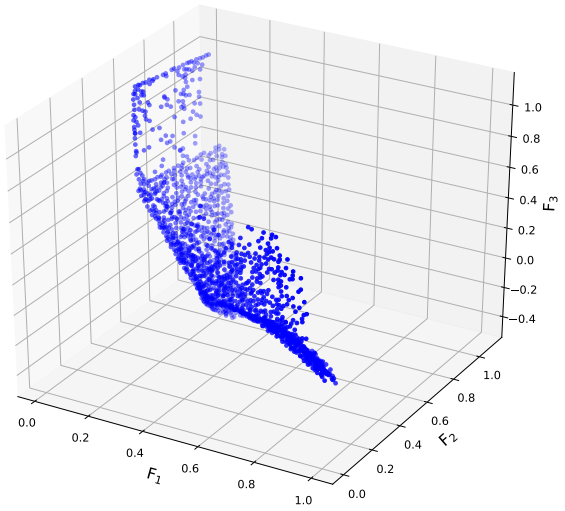


図 2.29: RE37 のパレート界の様子

第 3 章

提案手法

3.1 提案手法の概要

提案手法では, Deep-Opt を多目的最適化に拡張する. 多目的への応用とは Deep-Opt[5] では単目的最適化を行っていたことに対して本論文では多目的最適化に取り組むということである. Deep-Opt と同様に図 3.1 にあるようにニューラルネットワークを設計変数から多目的の目的関数値を推定するように設計し, back-driving を用いて探索し最適化する. すなわち, ニューラルネットワークは多目的最適化問題のランドスケープを学習するように設計されている. その疑似コードは Algorithm3 にある.

基本的な考え方は MOEA/D と同じで, 初期解をランダムに生み出した後に各解に重みベクトルを割り当てる. その後, 初期解を突然変異させることで DNN を学習するためのデータセットを作成する (4 行目). データセットの大きさは一世代の個体数の n 倍である. 本論文では n は 10 で固定している. 5 行目では最終的に出力するパレート界を 4 行目で作り出したデータセットの非優越解で初期化する.

その後, 終了条件を満たすまで探索を行う. 提案手法でも元の Deep-Opt と同様に解を [0.0, 1.0] にスケーリングする. その後, 先ほど作り出したデータセットを用いて DNN を学習する. この際, DNN が学習するのは設計変数から目的関数値のマッピングである. 学習の終了条件はそれぞれの目的関数において DNN の出力と実際の値の相関係数が一つでも下がったらである. また, その相関係数が一つでも 0 を下回った場合には DNN の重みを一度初期化してもう一度最初から学習する. これらは基となる Deep-Opt を参考にしている. 学習が終了したら back-driving を行うために DNN の重みを固定する (9 行目). DNN の出力を 0.0 に固定して目的関数の値が 0.0 に近づくように入力を更新していく. ただしこのときの入力は S に含まれる解で, S は各重みベクトルについてその重みベクトルが割り当てられている解の中で最も適合度の高い解の集合である. すなわち, S の要素数は重みベクトルの数に等しい. back-driving で解を改善したら突然変異を行って新たな解を作成する. このように作成された解と基となる解を重みベクトルを用いて適合度を比較して新たに作成した方が良ければ基となる解は S から取り除いて新たに作成した解を S に追加する. ここで新たに生み出した解が現在のパレート界にあるどの解にも優越されなければ Pareto_Front に追加し, Pareto_Front 内の解で新たに作成された解に優越される解が存在する場合にはその解は Pareto_Front から取り除く.

一方, 良くならなければ新たに作成された解は基の解に置き換わらずに DNN を学習するデータセッ

トに追加されるのみである. データセットに追加する理由としては, データセットには過去数世代に生み出された解が入っており, 一番古くに生み出された解よりは今世代の解の方がより現在探索している空間に近いと考えられるからである. そうすることで DNN が現在探索している空間周辺をより精密に学習することでより高精度な解の生成が行える. 最後に, データセットは一定の大きさになるように必要に応じて最も古いものから順番に削除していく.

そうして作成されたパレート界を終了条件を満たした後に出力する.

MOEA/D と同様に重みベクトルを割り当てる理由としては, その重みベクトルによってパレート界が広がることが期待されるからである. 図 3.1 にあるようにニューラルネットワークは多数の目的関数値を同時に推定する. 損失関数には最良点と出力の最小二乗誤差を採用するため back-driving を行うと最良点に向かって最適化が行われるため最良点付近の解が多く見つかりパレート界があまり広がらないことが想定される (図 3.2). そこで重みベクトルを用いて突然変異の方向性をある程度決定することでパレート界が広がることが期待される (図 3.3).

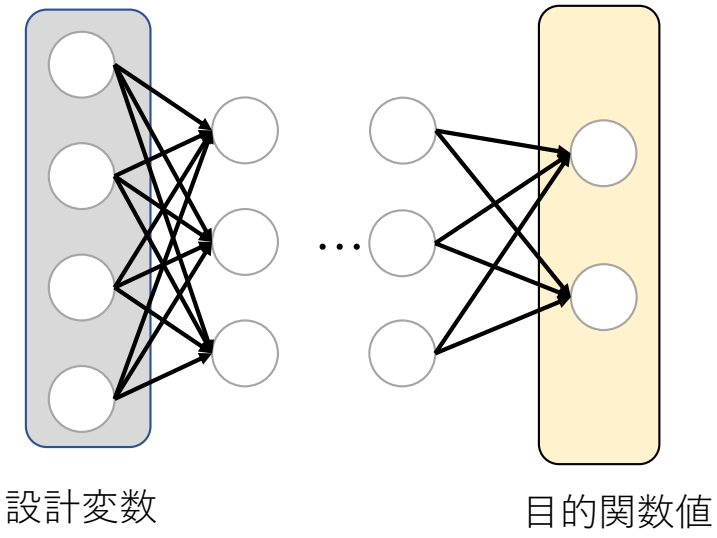


図 3.1: 提案手法におけるニューラルネットワークの様子

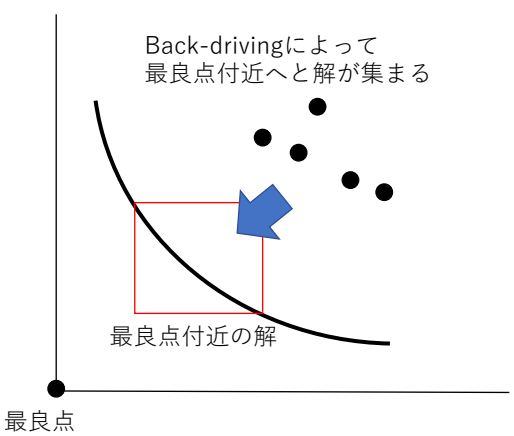


図 3.2: back-driving による最適化の様子

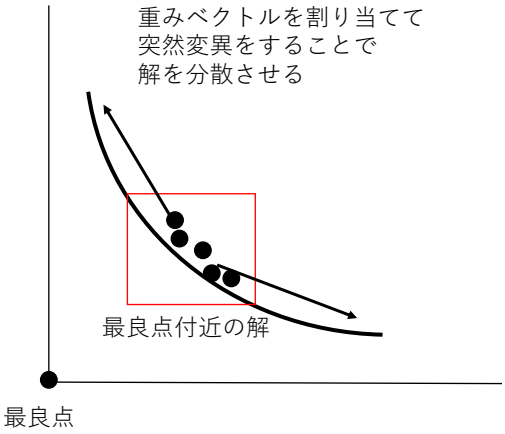


図 3.3: 重みベクトルを用いる利点

3.2 Surrogate Model

提案手法では DNN を設計変数から目的関数値を推測するためのものとして学習させる. そのため, この DNN を Surrogate Model として用いることも可能である. 具体的には, Algorithm3 の 15 行目において突然変異する際に, 突然変異で生み出された解を DNN に通すことでその解の目的関数値の推定値を得ることができる. 本論文での Surrogate Model の使い方としては, Algorithm3 の 15 行目において突然変異にて新しい個体を 100 個体生み出し, その個体すべてで目的関数値の推定値を得る. そうした後に, その推定値を用いてルーレット選択を行うことで今世代の解を決定するという流れになっている. ここで最大値を取らない理由としては, DNN の推定精度が高い水準になるためには学習点が少ないと考えられるためルーレット選択にすることで多少の誤差は許容して適合度が低くなるような解を選ばなくするということに主眼を置いているからである.

3.3 パラメータレスの手法

3.1 節で述べた提案手法において algotirhm3 の 7 行目で解をスケーリングする必要がある. その際のスケーリングは式 (3.1) によって計算されるがこのスケーリングには x_{max} および x_{min} が必要となる. LZ であればすべての目的関数のオーダーが同じであるので x_{max} および x_{min} はすべての要素で同じ値にすればよく, また最適パレート界がわかっているため設定する値はある程度設定しやすいが, RE では目的関数ごとに疑似最適パレート界の範囲が異なるため適切に設定する必要がある. 例えば, RE21 について F_1 のスケーリングの範囲を [0,10000] として F_2 のスケーリングの範囲を [0,1] とすると, 本論文では weighted sum を用いているため F_2 の値が多少上がっても F_1 の値を下げた方がスカラー関数の値が小さくなるためパレート界が広がらないという問題点がある. 具体例は図 3.4 および 3.5 にある. これら

Algorithm 3 Deep-Opt for Multi-Objecive

```
Create a set of solutions S randomly
Evaluate all solutions in S
Assign a weight vector to each solution in S
Make dataset to train DNN
Pareto_Front = pareto front of dataset made above
while termination condition is not met do
    Scale all solutions to [0.0,1.0]
    Train a DNN to map solutions → Evaluation(solutions) with dataset
    Freeze the weights of DNN

    Clamp the network's outputs to [0.0]*number_of_objectives
    for each s ∈ S do
        Initialize the DNN's input with s
        Change the input to s' by back-drive
        Mutate s' to s"
        if evaluate(s", weight_vector_for_s) is better than evaluate(s, weight_vector_for_s) then
            replace s in S with s"
        end if
        Update Pareto_Front if necessary
        Add s" to dataset to train DNN
    end for
    prune dataset if necessary
end while
Return Pareto_Front
```

はどちらも全世代の約 1/3 が経過した際に形成されているパレート界の様子であるが, 赤の点がその世代において DNN を学習するために用いられる解であり, オレンジの点が非優越解, 青の点が疑似最適パレート界である. これらの図はどちらも同じ世代およびスケーリング以外の条件は同じ条件で比較しているのにも関わらずスケーリングのための x_{max} および x_{min} を適切に設定できないと先述した通り F_1 を小さくする方が F_2 を小さくするよりもスカラー関数値は小さくなるため F_1 が小さくなるところに解が集中し結果的にパレート界が一部のみとなり広がらないという問題点が発生する (図 3.4). 一方, 図 3.5 では正しくスケーリングが行われているため全体に解が広がっていることがわかる. このことから特に目的関数ごとにオーダーが大きく異なる RE ではスケーリングのための x_{max} および x_{min} を適切に設定することが精度の高いパレート界の発見に役立つことがわかる. ただ, このパラメータはある程度アルゴリズムを動かしてパレート界の範囲を見る必要があるので何度も設定しなおす必要がある.

そこでこのパラメータの設定を無くすためにアルゴリズムの中で動的にパレート界を見て x_{max} およ

び x_{min} を変化させるという手法を提案する. その方法は単純に非優越解における i 番目の目的関数の最小値と最大値を x_{max} および x_{min} の i 番目の成分としている. 当然この値の間に入らない解も存在するがその値もちゃんとスケーリングされた上で大小関係も変わらないため問題はない. ここで主眼に置いているのは現在探索しているパレート界の中を正しくスケーリングして評価することである. これを全ての解における最大値および最小値とすると性能の低い外れ値の解によってパレート界付近の解が正しくスケーリングされず評価されないことが発生し図 3.4 のようになるためこのように非優越解の最大値および最小値をスケーリングの幅としている.

$$f_{scaled} = \frac{f - x_{min}}{x_{max} - x_{min}} \tag{3.1}$$

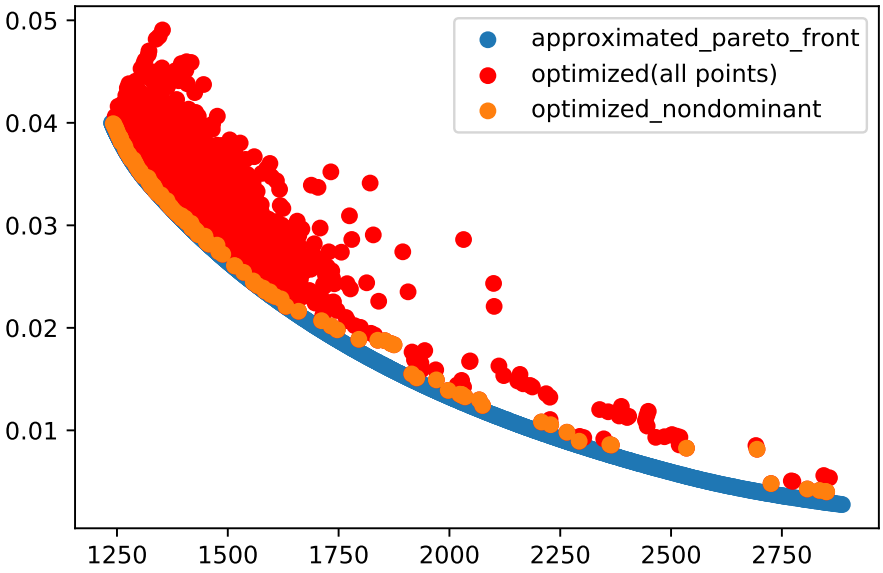


図 3.4: RE21 においてスケーリングの値を適切に設定できなかった結果 (F_1 の値の範囲を [0,10000], F_2 の値の範囲を [0,1] とした)

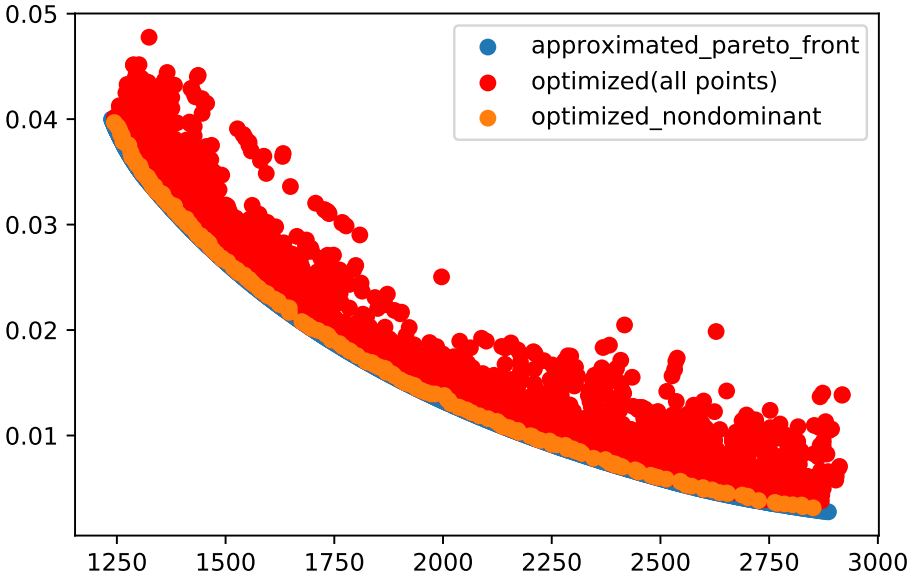


図 3.5: RE21 においてスケーリングの値を適切に設定した結果 (F_1 の値の範囲を [0,3000], F_2 の値の範囲を [0,0.05] とした)

第 4 章

実験設定

4.1 ベンチマーク問題

提案手法の性能を評価するために 2 種類のベンチマーク問題を用いた. 一つ目は LZ[18] であり, 設計変数においてパレート界が複雑になるように設定された関数である. LZ には 9 つの関数が存在してその目的関数および設計変数の次元数は表 4.1 の通りである. この実験では jMetalPy[36] による実装を用いた. 一つ目の問題として LZ を用いた理由としては, 関連研究の章でも説明した通り LZ がそれまでのベンチマーク問題に比べて設計変数空間において複雑なパレート界を持っているからである. ベンチマーク問題は最適パレート界がわかっているため, 後述する RE を用いる際と比べて様々な指標で評価することができる. 評価手法は後に詳細に説明するが, 評価手法の中には最適パレート界との距離を測るような手法もあるため最適パレート界がわかっている必要がある. そのため, ベンチマーク問題を用いることでアルゴリズムを様々な側面から評価することができる.

二つ目の関数は RE[7] である. この関数は実問題を関数に落とし込んだベンチマーク問題となっている. それぞれの問題における目的関数および設計変数の次元数は表 4.2 の通りである. RE は比較的新しいベンチマーク問題であり, まだ多く使われているわけではない. その RE を今回用いた理由としてはこのベンチマーク問題が最も実問題に近いベンチマーク問題であると考えられるからである. 関連研究の章でも説明した通りそれまでのベンチマーク問題は人手で設計されているため実問題の特徴を必ずしも捉えられているとは言えないが RE は実問題をそのままベンチマーク問題にしているためこのアルゴリズムを実問題に適用した際の挙動を確かめることができると考えられる.

関数	目的関数の数	設計変数の次元数
F1	2	10
F2	2	30
F3	2	30
F4	2	30
F5	2	30
F6	3	10
F7	2	10
F8	2	10
F9	2	30

表 4.1: LZ における目的関数および設計変数も次元数

関数	目的関数の数	設計変数の次元数
RE21	2	4
RE22	2	3
RE23	2	4
RE24	2	2
RE25	2	3
RE31	3	3
RE32	3	4
RE33	3	4
RE34	3	5
RE35	3	7
RE36	3	4
RE37	3	4

表 4.2: RE における目的関数および設計変数も次元数

4.2 パラメータ設定

提案手法におけるパラメータは一世代の個体数・突然変異率・DNN の学習に用いるデータセットの数である. 突然変異率は 1/(設計変数の数) としていて, 一度の突然変異で変化する遺伝子の数の期待値が一つとなるようにしている. DNN の学習に用いるデータセットの数は提案手法の章でも書いた通り本論文では一世代の個体数の 10 倍としている.

一世代の個体数について説明する前に提案手法における重みベクトルの生成方法を説明する. 提案手

目的関数の数	H	N(一世代の個体数)
2	29	30
	49	50
	99	100
	199	200
	299	300
3	6	28
	12	91
	18	190
	23	300
	33	595
	43	990

表 4.3: 本実験における H の値

法における重みベクトルは H をハイパーパラメータとして{i/H | i=0,1,...H-1,H}から合計が 1 となるように重複を許して目的関数の数と同じ数だけ並べることで作り出す. すなわち, 重みベクトルの i 番目の成分を λ_i と置くと式 (4.1) のようになる. また, 重みベクトルは網羅的に生成するためその総数は式 (4.2) の N のようになる. ただし, ここで m は目的関数の数である.

$$\sum_i \lambda_i = 1, \lambda_i \in \{i/H | i = 0, 1, ...H - 1, H\} \tag{4.1}$$

$$N = {}_{H+m-1}C_{m-1} \tag{4.2}$$

したがって, 一世代の個体数は N となり H によって決定される数となる. H は性能に大きく影響を及ぼすパラメータであるためいくつか変えて実験を行っている. 具体的には表 4.3 の通りである. 目的関数が 2 つの際には N=H+1 となるので N は自由に設定できるが目的関数が 3 つ以上になると N は自由に設定できないため, N が 30, 100, 200, 300, 600, 1000 に最も近くなるような H を採用した. また, MOEA/D において目的関数値と重みベクトルを用いた適合度の計算法はいくつかあるが今回は単純に重みベクトルと目的関数値の内積を用いた (Weighted Sum).

また, 今回の提案手法ではニューラルネットワークを用いているためその詳細についてもここで述べる. まずニューラルネットワークの構造は中間層が 3 層あり, 入力層および出力層を合わせたら 5 層のニューラルネットワークを用いた. また, オプティマイザーには Adam を用いて学習率は 0.001, weight decay は 0.98 とした. 損失関数には最良点と出力の最小二乗誤差を採用した. ここで最良点とはスケーリング後の [0.0]*(目的関数の数) である. 予めスケーリング幅を決めている場合には最良点はある程度分かった上で最適化をしていることとなるが, スケーリング幅をアルゴリズム中で調整する場合には最良点は MOEA/D で用いられる参照点と同じく z=(z_1, ..., z_m)(ただし, z_i = min ($F_i(x_1)$, $F_i(x_2)$, ... , $F_i(x_N)$)) を目指して最適化することとなる.

4.3 評価手法

評価手法は LZ では IGD(Inverted Generational Distance) と HV(HyperVolume) を用いて, RE では HV のみを用いた. IGD とは理想的なパレート界における各解からアルゴリズムによって求められた解群のうち最も近い点の距離の平均である. すなわちアルゴリズムによって求められた解がどれほど理想的なパレート界に近いのかということを測る指標である. IGD は式 (4.3) で計算することができる. ただし, A はアルゴリズムによって求められた解群であり, R は理想的なパレート界である. また本実験では p=2 としているので $d(r,a)^p$ は r と a のユークリッド距離である. IGD は小さければ小さいほど最適パレート界に近いため良いということとなる.

$$IGD = \left(\frac{1}{|R|} \sum_{r \in R} \min_{a \in A} d(r,a)^p \right)^{1/p} \tag{4.3}$$

HV は参照点を一つ設定した際にアルゴリズムによって求められた解とその参照点によって作り出される超平面の大きさである. 例えば目的関数の次元数が 2 のときを考えてみると図 4.1 のようになる. ここで $p^{(i)}$(i=1,2,3) はアルゴリズムによって求められた解であり, r は参照点である. HV は黒色の部分の面積となり, 大きければ大きいほど求めた解群がカバーできる面積が大きくなるため良い. 目的関数の数が 3 以上になっても同様に考えることができる. また, HV は参照点があれば計算ができるため理想的なパレート界が必要ない評価指標である. RE で IGD を用いない理由としては, RE で現在報告されているパレート界は存在するが実問題において理想的なパレート界というのはわからないため IGD を使うことが適切ではないと考えられるからである. 一方, LZ は人手で設計された問題であるため理想的なパレート界が既知であるから HV と IGD の両方を評価指標として用いている.

ただし, HV の結果については HV そのままの値を用いずに式 (4.4) によって計算される値を用いている.

$$\text{value\_in\_table} = 1 - \text{HV}/(\text{maximize\_of\_HV}) \tag{4.4}$$

分母にある HV の最大値はパレート最適解による HV である. すなわち, この表の値はパレート最適解と比較してどれだけの割合の領域をカバーできていないかという値となる. そのため, この値は小さい方がより広い領域をカバーできているということとなるため, 小さい方が性能が良いということとなる. HV の値をそのまま用いない理由としては, 特に RE で問題ごとに HV の値が大きく変わるためどれだけの性能なのかということをわかりやすくするためであり, 同時に HV の最大値で割った値を 1 から引く理由としては, アルゴリズムごとの差をより顕著にするためである. 例えば, HV の最大値で割った値をそのまま採用すると結果の表の上では同じ値であるのにも関わらずその両者で有意差が生じる可能性があるからである.

また, 実験結果の表に記載されている値は 20 回アルゴリズムを実行した際の平均値と標準偏差である.

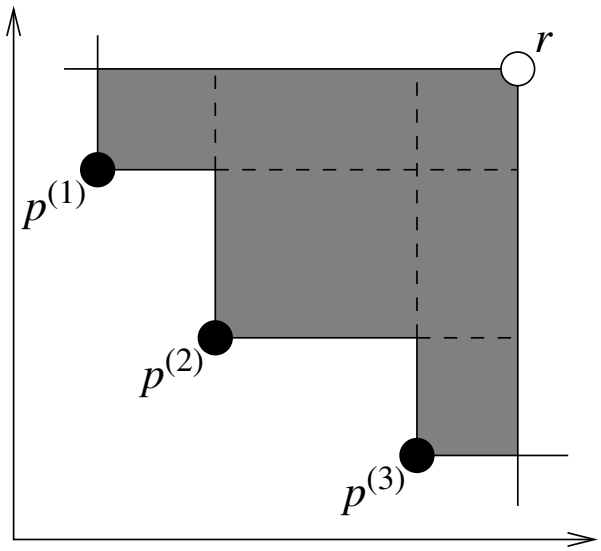


図 4.1: 二次元における HV の様子 ([37] より引用)

4.4 性能比較

今回の実験で比較する条件は表 4.4 にある通りである. 変える条件としては 3.2 節で説明した Surrogate Model の有無と 3.3 節で説明したパラメータの有無である. これらをそれぞれ変更させて比較実験を行うため 4 通りの条件で比較していく. 以後表中には提案手法 (条件 1) のように記載するがその条件番号と条件の内容は表 4.4 と対応している.

その他に提案手法と比較する既存手法は NSGAII[3], MOEA/D[4], MOEA/D-DRA[17] である. これらの手法を比較する際には jMetalPy[36] の実装を用いた. アルゴリズム間の評価を平等に行うために本研究のような最適化手法の研究では解の評価回数を等しくして比較を行うことが多い. すなわち, ブラックボックスとなっている関数に何回解を通してその評価値を受け取ったかという回数を等しくする. ここでは解の評価回数は 50,000 回にして行った. そのため, 解が 50,000 回評価されたらその時点でアルゴリズムを終了して最終的なパレート界を出力するということになっている. 次章で結果を表示する表において出てくる値は特に注釈がない場合には解を 50,000 回評価した段階で得られたパレート界についての値となっている.

条件	スケーリング幅に関するパラメータ	Surrogate Model
条件 1	スケーリング幅をパラメータとしてあらかじめ設定	無し
条件 2	スケーリング幅をパラメータとしてあらかじめ設定	有り
条件 3	スケーリング幅は自動調整 (パラメータレス)	無し
条件 4	スケーリング幅は自動調整 (パラメータレス)	有り

表 4.4: 提案手法の比較条件

第 5 章

実験結果

5.1　LZ の実験結果

4 章で述べた通り LZ の評価には HV と IGD の両者を用いる. HV についての結果は表 5.1 にあり, IGD についての結果は表 5.2 にある. IGD の結果の表には IGD の値をそのまま記載しているが, HV の結果の表に記載している値は 4.3 節でも述べた通り式 (4.4) で計算される値である. IGD も HV の表中の値も小さい方が性能が良い. 表 5.1, 5.2 共に有意水準 0.01 で全ての手法の中で最もよかったものを太字にしている. ただし, それぞれの表にある提案手法 (条件 i) とあるものは表 4.4 にある条件と対応している. ここでは既存手法と提案手法のうち Surrogate Model を用いないものとを比較した. LZ における HV の結果では全ての関数において提案手法は他手法と比較して統計的に有意に悪くなっていた. 一方, IGD では F8 のみ提案手法が他手法に比べて良くなっているか悪くなってはいなかったがその他の関数では他手法に比べて統計的に有意に悪くなっていた. ただ, HV の結果を見るとわかる通り全ての関数において提案手法でも理想的なパレート界と比較して 95% 以上の領域をカバーすることができている. そのため, 他手法に比べて悪くなっているとはいえ提案手法はパレート界を充分求めることができているといえる.

5.2　RE の実験結果

RE の実験結果は表 5.3 にある. 4.3 節で説明した通り, ここでも表に記載している値は式 (4.4) によって計算される値である. また, 記載されている値は 20 回アルゴリズムを実行した際の平均値と標準偏差である. ここでも同様に有意水準を 0.01 としてすべての手法の中で最も良くなっていたものを太字にしている. LZ では全ての関数で HV は他手法に比べて提案手法は統計的に有意に悪くなっていたが, RE では一部の関数で統計的に有意に良くなっていることがわかる. また, ほかの関数でも他手法よりも悪くなっているとはいえ, 理想的なパレート界と比較して 95% 以上の領域をカバーすることができているため提案手法は充分パレート界を発見することができているということがわかる.

特に RE24, RE33 に注目するとこの関数での表の値は負の値となっている. これは現在報告されているパレート界よりも HV が大きいパレート界が発見されたことを意味している. 他の手法では全ての関

数で正の値を取っているため報告されているパレート最適界よりも HV が大きいパレート界は見つからなかった. そのため問題によっては提案手法が他のアルゴリズムでは発見できない新たな解を発見できる可能性を RE24 および RE33 の結果は示唆している.

さらに, RE24 および RE33 において従来の疑似最適パレート界よりを優越する解を図示した. それは図 5.1 および 5.3 にある. これらの図では従来の疑似最適パレート界は青色の点で示されていて, 提案手法によって見つかった従来の疑似最適パレート界よりを優越する解は赤色で図示されている. ただし, これらの図は独立して行った 20 回の結果を合わせたものであり, ある 1 回の試行について図示したものではない. 20 回の結果を図示したものをここに載せた理由としては毎回同じ解が得られるわけではなく多少の揺らぎが存在するため, 全体でどれほど良い解が見つかったかを図示するためである. また. それらの解に割り当てられた重みベクトルの要素についてヒストグラムを作成した. それが図 5.2, 5.4, 5.5 である. RE24 についてのヒストグラムは図 5.2 であるが, ここでは重みベクトルの第一成分 (F_1 をどれほど重要視するかという値) について数えている. 重みベクトルは正規化されており, RE24 において重みベクトルの要素数は 2 つであることから第一成分がわかれば第二成分も自動的に定まるため, 重みベクトルの第一成分についてヒストグラムを生成すれば十分である. これらの図からわかることとして RE24 のなかでもすべての領域で満遍なく良い解が見つかっているというわけではなく, 一部の領域に偏っている. それは, F_1>200 の領域と 9<F_2<30 の領域である. この領域外でも従来の疑似最適パレート界を優越している解は存在するが, この二つの領域に比べると少ない. そのためこのアルゴリズムはこれらの領域で既存手法よりも力を発揮できることがわかった. また, 図 5.2 からは重みベクトルの第一成分の値が 0.6 を境に数が大きく異なっている. このことからさらに提案手法では 9<F_2<30 の領域の方が F_1>200 の領域よりも従来の疑似最適パレート界を優越している点が多いことがわかる. それは F_1 を重要視する重みが多いことからより F_1 の値が小さくなるような点が多いからである. 実際に全体の約 90% が 9<F_2<30 の領域に存在し, 200<F_1 の領域には全体の 8.6% が存在した.

続いて RE33 について, 図 5.3 が従来の疑似最適パレート界を優越する解を図示したものであるが, これも RE24 の際と同様に 独立して行った 20 回の結果を合わせたものである. わかりやすくするためにここでは優越している点が存在しているところに限って図示しているが (0 < F_3 < 30 の領域をズームしている), RE33 のパレート界全体で考えるとこれはごく一部である. それは図 2.25 を見ると F_3 の値が大きく違うことからわかる. 図 5.3 からは F_2 が最小になる部分に従来の疑似最適パレート界を優越している点が多いことがわかる. また, 図 5.4 では重みベクトルの第 1 成分を横軸に, 第 2 成分を縦軸にした重みベクトルについてのヒストグラムである. 重みベクトルは正規化されているため第 1 成分および第 2 成分が分かれば第 3 成分も自動的に決まるが, ここでは分かりやすくするために第 3 成分についてのヒストグラムも図 5.5 に載せている. 重みベクトルのノルムは 1.0 であるから図 5.4 における右上の部分に全く要素は存在しない. これらの図から第 3 成分は小さい値が多く, 第 1 成分および第 2 成分は満遍なく分布していることが分かった. ただし, 第 1 成分が 0.8 を超えるような重みベクトルはほとんど存在しないが, 第 2 成分が 0.8 を超えるような重みベクトルは多少存在していることが確認できる.

関数	提案手法 (条件 1)	提案手法 (条件 3)	NSGAII	MOEA/D	MOEA/D-DRA
F1	$8.18\times10^{-5}\pm4.47\times10^{-5}$	$8.72\times10^{-5}\pm2.66\times10^{-5}$	$8.98\times10^{-5}\pm2.09\times10^{-5}$	$\mathbf{3.12\times10^{-5}\pm1.34\times10^{-5}}$	$1.42\times10^{-4}\pm1.05\times10^{-4}$
F2	$2.22\times10^{-2}\pm4.51\times10^{-3}$	$2.29\times10^{-2}\pm5.86\times10^{-3}$	$1.52\times10^{-2}\pm8.04\times10^{-3}$	$\mathbf{3.59\times10^{-3}\pm3.05\times10^{-3}}$	$\mathbf{7.30\times10^{-3}\pm8.35\times10^{-3}}$
F3	$1.02\times10^{-2}\pm6.96\times10^{-4}$	$1.12\times10^{-2}\pm1.05\times10^{-3}$	$9.51\times10^{-3}\pm4.99\times10^{-3}$	$6.92\times10^{-3}\pm8.48\times10^{-4}$	$\mathbf{3.94\times10^{-3}\pm6.27\times10^{-3}}$
F4	$1.57\times10^{-2}\pm3.93\times10^{-3}$	$1.75\times10^{-2}\pm3.10\times10^{-3}$	$1.47\times10^{-2}\pm1.44\times10^{-3}$	$4.16\times10^{-3}\pm3.35\times10^{-3}$	$\mathbf{9.59\times10^{-4}\pm3.32\times10^{-4}}$
F5	$1.02\times10^{-2}\pm1.34\times10^{-3}$	$1.30\times10^{-2}\pm5.21\times10^{-3}$	$\mathbf{5.39\times10^{-3}\pm3.08\times10^{-3}}$	$\mathbf{3.93\times10^{-3}\pm4.73\times10^{-3}}$	$\mathbf{3.05\times10^{-3}\pm2.54\times10^{-3}}$
F6	$1.55\times10^{-3}\pm4.59\times10^{-3}$	$1.43\times10^{-3}\pm6.03\times10^{-3}$	$1.28\times10^{-3}\pm7.40\times10^{-4}$	$2.76\times10^{-4}\pm9.07\times10^{-5}$	$\mathbf{1.88\times10^{-4}\pm4.34\times10^{-5}}$
F7	$1.71\times10^{-2}\pm3.84\times10^{-3}$	$2.27\times10^{-2}\pm7.14\times10^{-3}$	$3.69\times10^{-2}\pm1.29\times10^{-3}$	$\mathbf{1.13\times10^{-2}\pm7.01\times10^{-3}}$	$2.17\times10^{-2}\pm8.46\times10^{-3}$
F8	$3.56\times10^{-2}\pm6.00\times10^{-3}$	$3.39\times10^{-2}\pm6.26\times10^{-3}$	$3.63\times10^{-2}\pm1.23\times10^{-2}$	$\mathbf{1.43\times10^{-2}\pm1.04\times10^{-2}}$	$2.19\times10^{-2}\pm1.32\times10^{-2}$
F9	$3.73\times10^{-2}\pm2.60\times10^{-3}$	$3.93\times10^{-2}\pm2.86\times10^{-3}$	$4.16\times10^{-2}\pm1.76\times10^{-3}$	$\mathbf{2.96\times10^{-3}\pm1.43\times10^{-3}}$	$\mathbf{5.48\times10^{-3}\pm4.65\times10^{-3}}$

表 5.1: LZ における HV の結果

関数	提案手法 (条件 1)	提案手法 (条件 3)	NSGAII	MOEA/D	MOEA/D-DRA
F1	$3.06\times10^{-3}\pm1.60\times10^{-4}$	$3.69\times10^{-3}\pm3.88\times10^{-4}$	$4.24\times10^{-3}\pm4.44\times10^{-4}$	$\mathbf{1.32\times10^{-3}\pm2.66\times10^{-5}}$	$1.56\times10^{-3}\pm1.26\times10^{-4}$
F2	$7.60\times10^{-2}\pm1.50\times10^{-2}$	$8.45\times10^{-2}\pm3.54\times10^{-2}$	$1.02\times10^{-1}\pm1.80\times10^{-2}$	$\mathbf{3.84\times10^{-2}\pm1.36\times10^{-2}}$	$4.56\times10^{-2}\pm2.49\times10^{-2}$
F3	$4.11\times10^{-2}\pm4.82\times10^{-3}$	$4.75\times10^{-2}\pm7.51\times10^{-3}$	$5.13\times10^{-2}\pm1.27\times10^{-2}$	$\mathbf{2.81\times10^{-2}\pm2.18\times10^{-2}}$	$\mathbf{2.07\times10^{-2}\pm1.37\times10^{-2}}$
F4	$5.16\times10^{-2}\pm1.56\times10^{-3}$	$5.85\times10^{-2}\pm1.05\times10^{-2}$	$4.93\times10^{-2}\pm5.15\times10^{-3}$	$1.85\times10^{-2}\pm6.74\times10^{-3}$	$\mathbf{7.77\times10^{-3}\pm1.46\times10^{-3}}$
F5	$3.74\times10^{-2}\pm5.98\times10^{-3}$	$5.73\times10^{-2}\pm3.05\times10^{-2}$	$3.89\times10^{-2}\pm8.29\times10^{-3}$	$\mathbf{2.22\times10^{-2}\pm6.65\times10^{-3}}$	$\mathbf{2.03\times10^{-2}\pm9.37\times10^{-3}}$
F6	$6.46\times10^{-2}\pm8.05\times10^{-3}$	$5.81\times10^{-2}\pm7.14\times10^{-3}$	$1.11\times10^{-1}\pm1.54\times10^{-2}$	$\mathbf{4.53\times10^{-2}\pm4.05\times10^{-3}}$	$\mathbf{4.48\times10^{-2}\pm3.38\times10^{-3}}$
F7	$6.93\times10^{-2}\pm9.07\times10^{-3}$	$8.32\times10^{-2}\pm2.10\times10^{-2}$	$1.50\times10^{-1}\pm7.79\times10^{-2}$	$\mathbf{2.42\times10^{-2}\pm2.22\times10^{-2}}$	$4.75\times10^{-2}\pm2.48\times10^{-2}$
F8	$\mathbf{1.79\times10^{-1}\pm2.05\times10^{-2}}$	$\mathbf{1.75\times10^{-1}\pm2.60\times10^{-2}}$	$\mathbf{1.53\times10^{-1}\pm3.04\times10^{-2}}$	$\mathbf{1.75\times10^{-1}\pm3.34\times10^{-2}}$	$2.09\times10^{-1}\pm8.11\times10^{-2}$
F9	$7.94\times10^{-2}\pm6.36\times10^{-3}$	$8.33\times10^{-2}\pm9.76\times10^{-3}$	$1.09\times10^{-1}\pm2.88\times10^{-2}$	$\mathbf{4.14\times10^{-3}\pm2.54\times10^{-2}}$	$\mathbf{5.15\times10^{-2}\pm3.16\times10^{-2}}$

表 5.2: LZ における IGD の結果

関数	提案手法 (条件 1)	提案手法 (条件 3)	NSGAII	MOEA/D	MOEA/D-DRA
RE21	$\mathbf{8.79\times10^{-4}\pm1.30\times10^{-4}}$	$\mathbf{7.84\times10^{-4}\pm1.18\times10^{-5}}$	$2.97\times10^{-3}\pm1.06\times10^{-4}$	$6.03\times10^{-1}\pm1.06\times10^{-2}$	$5.90\times10^{-1}\pm1.04\times10^{-2}$
RE22	$7.89\times10^{-3}\pm1.01\times10^{-3}$	$9.10\times10^{-3}\pm1.84\times10^{-3}$	$4.08\times10^{-3}\pm3.08\times10^{-4}$	$\mathbf{2.74\times10^{-3}\pm8.21\times10^{-5}}$	$\mathbf{2.71\times10^{-3}\pm7.43\times10^{-4}}$
RE23	$\mathbf{1.86\times10^{-4}\pm2.77\times10^{-5}}$	$2.17\times10^{-4}\pm2.90\times10^{-5}$	$2.41\times10^{-4}\pm1.26\times10^{-5}$	$1.98\times10^{-2}\pm2.51\times10^{-4}$	$2.01\times10^{-2}\pm4.38\times10^{-4}$
RE24	$\mathbf{-1.54\times10^{-5}\pm8.54\times10^{-6}}$	$\mathbf{-9.67\times10^{-6}\pm7.49\times10^{-6}}$	$6.72\times10^{-4}\pm5.34\times10^{-5}$	$5.52\times10^{-3}\pm3.53\times10^{-5}$	$5.50\times10^{-3}\pm3.22\times10^{-5}$
RE25	$1.16\times10^{-9}\pm3.98\times10^{-10}$	$5.53\times10^{-10}\pm1.83\times10^{-10}$	$\mathbf{3.01\times10^{-10}\pm1.05\times10^{-11}}$	$5.63\times10^{-1}\pm3.00\times10^{-12}$	$5.63\times10^{-1}\pm7.89\times10^{-12}$
RE31	$3.37\times10^{-5}\pm1.46\times10^{-5}$	$3.82\times10^{-5}\pm2.05\times10^{-5}$	$\mathbf{4.01\times10^{-6}\pm2.61\times10^{-6}}$	$2.42\times10^{-4}\pm9.79\times10^{-6}$	$2.43\times10^{-4}\pm9.34\times10^{-6}$
RE32	$\mathbf{3.69\times10^{-5}\pm7.97\times10^{-6}}$	$4.89\times10^{-5}\pm1.49\times10^{-5}$	$4.58\times10^{-5}\pm1.05\times10^{-5}$	$3.58\times10^{-2}\pm7.51\times10^{-4}$	$3.54\times10^{-2}\pm6.05\times10^{-4}$
RE33	$\mathbf{-1.04\times10^{-3}\pm4.41\times10^{-5}}$	$-9.69\times10^{-4}\pm8.45\times10^{-5}$	$9.53\times10^{-4}\pm6.61\times10^{-4}$	$2.90\times10^{-4}\pm4.14\times10^{-4}$	$6.76\times10^{-5}\pm9.17\times10^{-4}$
RE34	$1.05\times10^{-2}\pm1.30\times10^{-3}$	$1.16\times10^{-2}\pm1.65\times10^{-3}$	$\mathbf{4.21\times10^{-3}\pm5.69\times10^{-4}}$	$6.07\times10^{-2}\pm8.39\times10^{-3}$	$6.23\times10^{-2}\pm9.10\times10^{-3}$
RE35	$1.74\times10^{-3}\pm2.26\times10^{-4}$	$2.27\times10^{-3}\pm5.12\times10^{-4}$	$1.42\times10^{-3}\pm4.22\times10^{-4}$	$\mathbf{1.04\times10^{-3}\pm1.70\times10^{-4}}$	$\mathbf{1.07\times10^{-3}\pm2.03\times10^{-3}}$
RE36	$3.54\times10^{-3}\pm2.20\times10^{-3}$	$3.60\times10^{-3}\pm1.47\times10^{-3}$	$\mathbf{1.08\times10^{-4}\pm7.81\times10^{-5}}$	$5.18\times10^{-4}\pm4.20\times10^{-4}$	$3.25\times10^{-4}\pm1.21\times10^{-4}$
RE37	$2.77\times10^{-2}\pm4.68\times10^{-3}$	$2.83\times10^{-2}\pm3.43\times10^{-3}$	$1.08\times10^{-2}\pm1.11\times10^{-3}$	$\mathbf{6.33\times10^{-3}\pm3.96\times10^{-4}}$	$\mathbf{6.16\times10^{-3}\pm3.18\times10^{-4}}$

表 5.3: RE における HV の結果

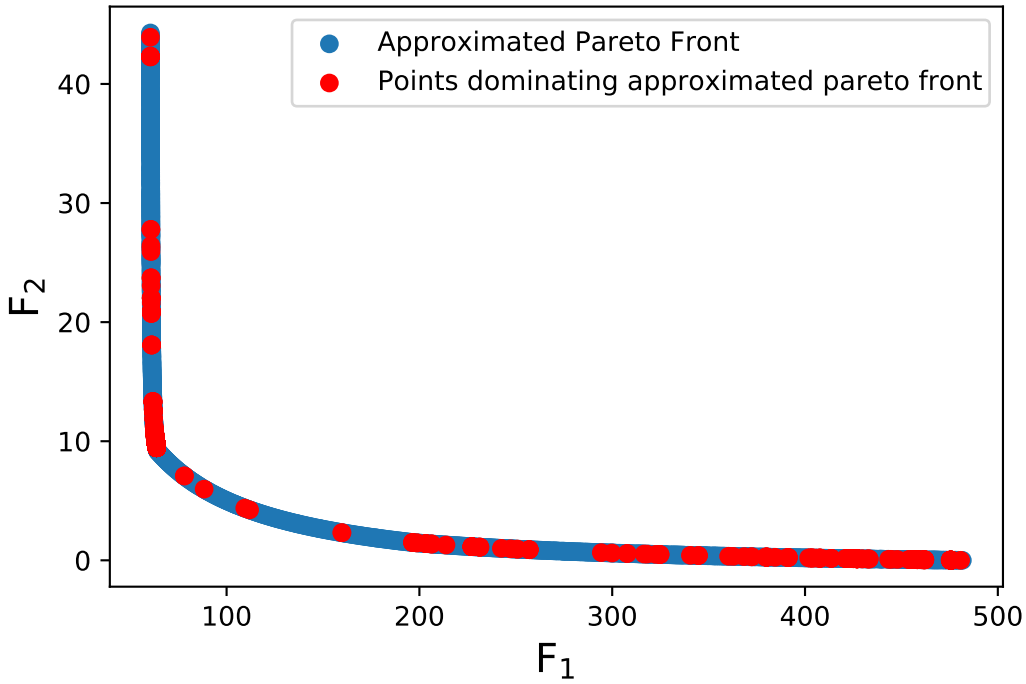


図 5.1: RE24 において疑似最適パレート界を優越した点

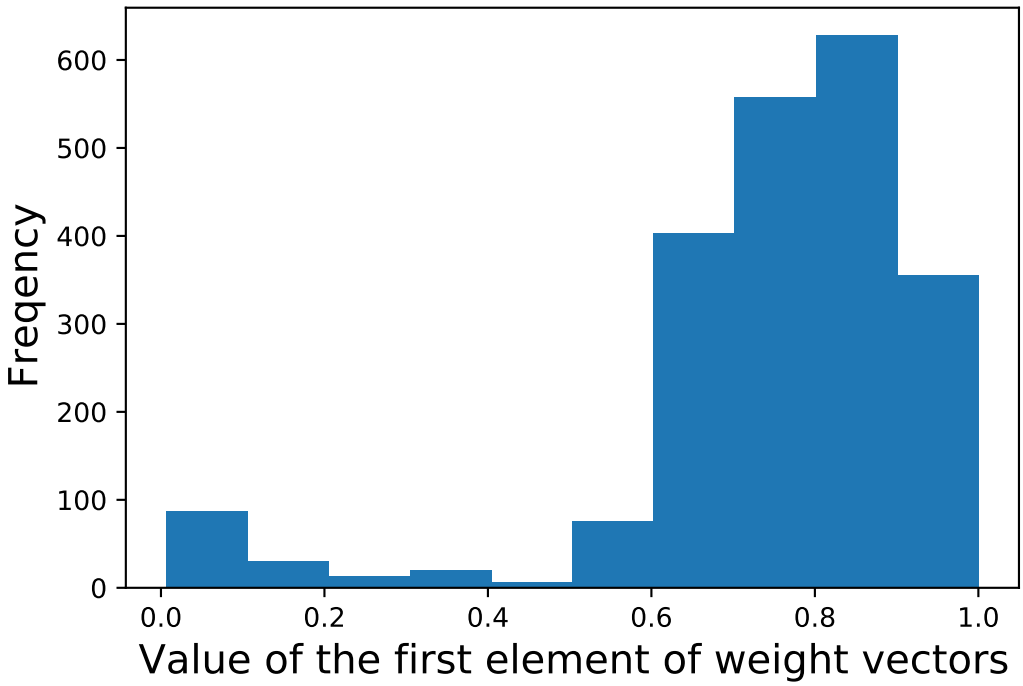


図 5.2: RE24 において疑似最適パレート界を優越した点の重みベクトルを数えたヒストグラム (グラフにしているのは重みベクトルの第 1 成分のみである. その理由は第 1 成分と第 2 成分の合計は 1 であるから第 1 成分が決まれば第 2 成分は自動的に定まるからである.)

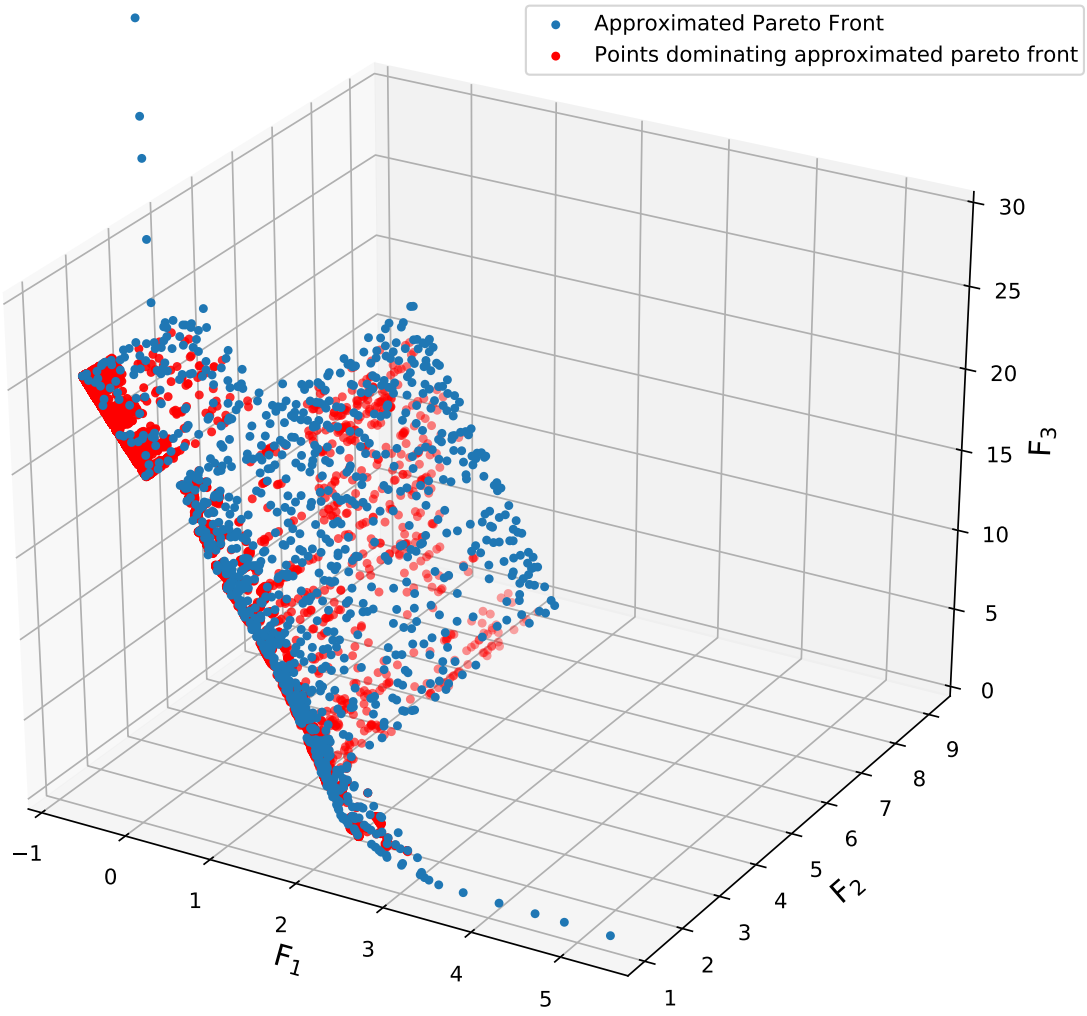


図 5.3: RE33 において疑似最適パレート界を優越した点

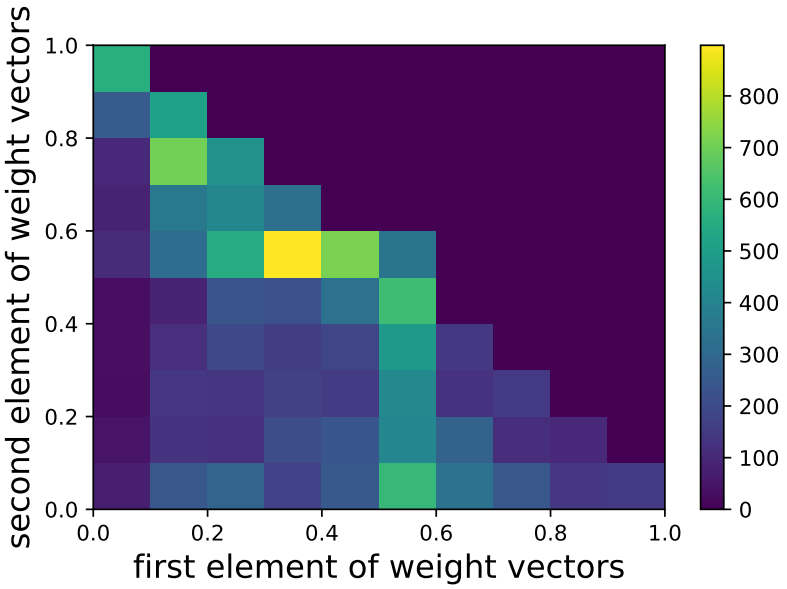


図 5.4: RE33 において疑似最適パレート界を優越した点の重みベクトルを数えたヒストグラム (グラフにしているのは重みベクトルの第 1 成分および第 2 成分のみである. その理由は全成分の合計は 1 であるから第 1 成分と第 2 成分が決まれば第 3 成分は自動的に定まるからである.)

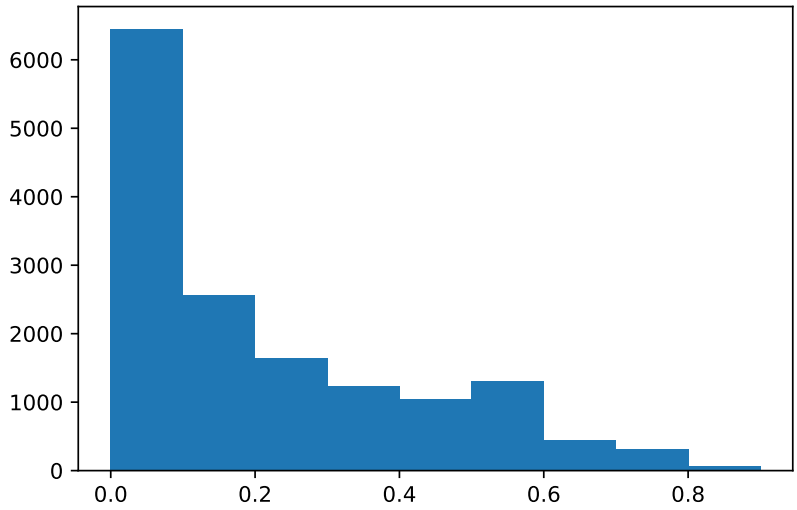


図 5.5: RE33 において疑似最適パレート界を優越した点の重みベクトルのうち第 3 成分について数えたヒストグラム

5.3 Surrogate Model の実験結果

Surrogate Model についての実験結果は表 5.4, 5.5, 5.8, 5.6, 5.7, 5.9 にある. ここでは提案手法の条件間での結果のみを比較している. その理由は, Surrogate Model を用いることの効果を比較することが目標であるからである. また, 実験設定の章でも説明したが, 最適化のアルゴリズム評価では評価関数に解を通す回数を等しくして比較をする. この実験では主に 50,000 回の評価で最終的な性能としているが, ここでは一部で 10,000 回評価関数に通した際の性能の比較も行っている. LZ について実験を行った結果は表 5.4, 5.5, 5.6, 5.7 で, それぞれ評価回数が 50,000 回の HV について, 評価回数が 50,000 回の IGD について, 評価回数が 10,000 回の HV について, 評価回数が 10,000 回の IGD についてである. RE についての実験結果は表 5.8, 5.9 であり, RE は HV でしか比較を行わないため表 5.8 が RE の HV を 50,000 回の評価で比較した際の表で表 5.9 が RE の HV を 10,000 回の評価で比較した際の表である. 今までの表と同じく 50,000 回での比較だけでなくて 10,000 回での比較も行った理由としては Surrogate Model を用いることで最終的な成績が良くなることがなくても悪い解の評価をすることが無くなって早い段階で良いパレート界を発見することができると考えたからである.

LZ では評価回数 10,000 回の HV および IGD において全てで条件 2 が条件 1 よりも悪くなっていることは無かった. F7 の IGD では条件 1 のみが太字になっているがこれは条件 3 と比較した際に統計的に有意に良くなっていたのが条件 1 であったため条件 1 を太字にしているが条件 1 と 2 の間で有意差は無かった. LZ において条件 1 と条件 2 の間または条件 3 および 4 の間では全ての関数で有意差は得られなかった.

RE では 10,000 回および 50,000 回評価の RE36 のみ Surrogate Model が有る方がそれが無いときに比べて有意に性能が悪くなっていた. 一方, Surrogate Model の方が良くなっていたものとしては, RE24 と RE33 における条件 3 と条件 4 の間で条件 4 の方が有意によくなっていた. それ以外の関数では条件 1 と条件 2 の間または条件 3 と条件 4 の間で有意差は出なかった. RE では Surrogate モデルの有無の差よりもスケーリング幅についてのパラメータの有無の差の方が性能に効いている. RE25 のみパラメータを人手で設定するよりも自動でスケーリング幅を設定した方が良い結果となっている.

関数	提案手法 (条件 1)	提案手法 (条件 2)	提案手法 (条件 3)	提案手法 (条件 4)
F1	$8.18\times10^{-5}\pm4.47\times10^{-5}$	$9.52\times10^{-5}\pm7.38\times10^{-5}$	$8.72\times10^{-5}\pm2.66\times10^{-5}$	$9.53\times10^{-5}\pm3.99\times10^{-5}$
F2	$2.22\times10^{-2}\pm4.51\times10^{-3}$	$2.10\times10^{-2}\pm1.39\times10^{-3}$	$2.29\times10^{-2}\pm5.86\times10^{-3}$	$2.28\times10^{-2}\pm4.69\times10^{-3}$
F3	$1.02\times10^{-2}\pm6.96\times10^{-4}$	$1.05\times10^{-2}\pm1.27\times10^{-3}$	$1.12\times10^{-2}\pm1.05\times10^{-3}$	$1.05\times10^{-2}\pm2.53\times10^{-3}$
F4	$1.57\times10^{-2}\pm3.93\times10^{-3}$	$1.66\times10^{-3}\pm3.44\times10^{-3}$	$1.75\times10^{-2}\pm3.10\times10^{-3}$	$1.84\times10^{-2}\pm3.78\times10^{-3}$
F5	$1.02\times10^{-2}\pm1.34\times10^{-3}$	$9.68\times10^{-3}\pm2.28\times10^{-3}$	$1.30\times10^{-2}\pm5.21\times10^{-3}$	$1.12\times10^{-3}\pm2.42\times10^{-3}$
F6	$1.55\times10^{-3}\pm4.59\times10^{-3}$	$1.73\times10^{-3}\pm4.86\times10^{-3}$	$1.43\times10^{-3}\pm6.03\times10^{-3}$	$1.58\times10^{-3}\pm6.48\times10^{-3}$
F7	$\mathbf{1.71\times10^{-2}\pm3.84\times10^{-3}}$	$1.73\times10^{-2}\pm6.40\times10^{-3}$	$2.27\times10^{-2}\pm7.14\times10^{-3}$	$2.10\times10^{-2}\pm5.11\times10^{-3}$
F8	$3.56\times10^{-2}\pm6.00\times10^{-3}$	$3.36\times10^{-2}\pm5.73\times10^{-3}$	$3.39\times10^{-2}\pm6.26\times10^{-3}$	$3.21\times10^{-2}\pm5.93\times10^{-3}$
F9	$3.73\times10^{-2}\pm2.60\times10^{-3}$	$3.81\times10^{-2}\pm2.13\times10^{-3}$	$3.93\times10^{-2}\pm2.86\times10^{-3}$	$3.86\times10^{-2}\pm2.38\times10^{-3}$

表 5.4: LZ における HV の surrogate モデルありとの比較結果 (評価回数:50000 回)

関数	提案手法 (条件 1)	提案手法 (条件 2)	提案手法 (条件 3)	提案手法 (条件 4)
F1	$3.06\times10^{-3}\pm1.60\times10^{-4}$	$3.11\times10^{-3}\pm1.58\times10^{-4}$	$3.69\times10^{-3}\pm3.88\times10^{-4}$	$3.67\times10^{-3}\pm53.72\times10^{-4}$
F2	$7.60\times10^{-2}\pm1.50\times10^{-2}$	$7.21\times10^{-2}\pm6.65\times10^{-3}$	$8.45\times10^{-2}\pm3.54\times10^{-2}$	$8.01\times10^{-2}\pm1.69\times10^{-2}$
F3	$\mathbf{4.11\times10^{-2}\pm4.82\times10^{-3}}$	$4.17\times10^{-2}\pm6.29\times10^{-2}$	$4.75\times10^{-2}\pm7.51\times10^{-3}$	$4.42\times10^{-2}\pm5.29\times10^{-3}$
F4	$5.16\times10^{-2}\pm1.56\times10^{-3}$	$5.51\times10^{-2}\pm1.19\times10^{-2}$	$5.85\times10^{-2}\pm1.05\times10^{-2}$	$5.53\times10^{-2}\pm1.16\times10^{-3}$
F5	$3.74\times10^{-2}\pm5.98\times10^{-3}$	$3.79\times10^{-2}\pm7.87\times10^{-3}$	$5.73\times10^{-2}\pm3.05\times10^{-2}$	$4.92\times10^{-2}\pm1.91\times10^{-3}$
F6	$6.46\times10^{-2}\pm8.05\times10^{-3}$	$6.71\times10^{-2}\pm1.13\times10^{-2}$	$\mathbf{5.81\times10^{-2}\pm7.14\times10^{-3}}$	$6.11\times10^{-2}\pm9.64\times10^{-3}$
F7	$6.93\times10^{-2}\pm9.07\times10^{-3}$	$6.98\times10^{-2}\pm1.73\times10^{-2}$	$8.32\times10^{-2}\pm2.10\times10^{-2}$	$8.16\times10^{-2}\pm1.84\times10^{-2}$
F8	$1.79\times10^{-1}\pm2.05\times10^{-2}$	$1.72\times10^{-1}\pm2.16\times10^{-2}$	$1.75\times10^{-1}\pm2.60\times10^{-2}$	$1.69\times10^{-1}\pm1.60\times10^{-2}$
F9	$7.94\times10^{-2}\pm6.36\times10^{-3}$	$8.09\times10^{-2}\pm7.80\times10^{-3}$	$8.33\times10^{-2}\pm9.76\times10^{-3}$	$8.46\times10^{-2}\pm6.90\times10^{-3}$

表 5.5: LZ における IGD の surrogate モデルありとの比較結果 (評価回数:50000 回)

関数	提案手法 (条件 1)	提案手法 (条件 2)	提案手法 (条件 3)	提案手法 (条件 4)
F1	$1.10\times10^{-2}\pm3.30\times10^{-3}$	$1.28\times10^{-2}\pm4.22\times10^{-3}$	$1.22\times10^{-2}\pm4.86\times10^{-3}$	$1.10\times10^{-2}\pm3.27\times10^{-3}$
F2	$\mathbf{5.79\times10^{-2}\pm1.23\times10^{-2}}$	$\mathbf{6.19\times10^{-2}\pm1.18\times10^{-2}}$	$8.20\times10^{-2}\pm2.67\times10^{-2}$	$7.63\times10^{-2}\pm1.88\times10^{-3}$
F3	$8.46\times10^{-2}\pm1.98\times10^{-2}$	$\mathbf{7.75\times10^{-2}\pm1.28\times10^{-2}}$	$9.84\times10^{-2}\pm1.77\times10^{-2}$	$9.40\times10^{-2}\pm2.51\times10^{-2}$
F4	$\mathbf{8.30\times10^{-2}\pm1.37\times10^{-2}}$	$\mathbf{8.90\times10^{-2}\pm1.62\times10^{-3}}$	$9.98\times10^{-2}\pm2.35\times10^{-2}$	$1.07\times10^{-1}\pm1.74\times10^{-2}$
F5	$8.30\times10^{-2}\pm2.23\times10^{-2}$	$8.24\times10^{-2}\pm1.53\times10^{-2}$	$9.82\times10^{-2}\pm2.49\times10^{-2}$	$9.60\times10^{-2}\pm2.46\times10^{-2}$
F6	$1.27\times10^{-2}\pm3.65\times10^{-3}$	$1.37\times10^{-2}\pm3.62\times10^{-3}$	$1.18\times10^{-2}\pm2.47\times10^{-3}$	$1.37\times10^{-2}\pm4.90\times10^{-3}$
F7	$7.24\times10^{-2}\pm1.50\times10^{-2}$	$7.24\times10^{-2}\pm1.42\times10^{-2}$	$7.60\times10^{-2}\pm1.09\times10^{-2}$	$7.39\times10^{-2}\pm1.18\times10^{-2}$
F8	$7.86\times10^{-2}\pm1.39\times10^{-2}$	$7.67\times10^{-2}\pm1.07\times10^{-3}$	$7.61\times10^{-2}\pm1.31\times10^{-2}$	$7.63\times10^{-2}\pm1.33\times10^{-3}$
F9	$1.35\times10^{-1}\pm1.68\times10^{-2}$	$1.26\times10^{-1}\pm1.79\times10^{-2}$	$1.49\times10^{-1}\pm2.93\times10^{-2}$	$1.62\times10^{-1}\pm3.12\times10^{-2}$

表 5.6: LZ における HV の Surrogate モデルありとの比較結果 (評価回数:10000 回)

関数	提案手法 (条件 1)	提案手法 (条件 2)	提案手法 (条件 3)	提案手法 (条件 4)
F1	$4.32\times10^{-2}\pm1.44\times10^{-2}$	$4.87\times10^{-2}\pm2.14\times10^{-2}$	$4.21\times10^{-2}\pm1.67\times10^{-2}$	$3.67\times10^{-2}\pm9.48\times10^{-3}$
F2	$\mathbf{3.67\times10^{-1}\pm9.35\times10^{-2}}$	$\mathbf{3.64\times10^{-1}\pm7.08\times10^{-2}}$	$5.34\times10^{-1}\pm1.73\times10^{-1}$	$5.50\times10^{-1}\pm9.76\times10^{-2}$
F3	$5.98\times10^{-1}\pm2.09\times10^{-1}$	$5.69\times10^{-1}\pm1.60\times10^{-1}$	$6.69\times10^{-1}\pm1.43\times10^{-1}$	$7.19\times10^{-1}\pm1.99\times10^{-1}$
F4	$4.33\times10^{-1}\pm1.01\times10^{-1}$	$5.15\times10^{-2}\pm1.03\times10^{-2}$	$6.90\times10^{-1}\pm2.53\times10^{-1}$	$7.24\times10^{-1}\pm2.05\times10^{-1}$
F5	$\mathbf{6.02\times10^{-1}\pm1.34\times10^{-1}}$	$\mathbf{5.92\times10^{-2}\pm1.45\times10^{-2}}$	$7.72\times10^{-1}\pm2.48\times10^{-1}$	$7.72\times10^{-1}\pm2.14\times10^{-1}$
F6	$1.87\times10^{-1}\pm2.50\times10^{-2}$	$1.88\times10^{-1}\pm2.82\times10^{-2}$	$1.78\times10^{-1}\pm1.75\times10^{-2}$	$1.87\times10^{-1}\pm1.93\times10^{-2}$
F7	$\mathbf{4.23\times10^{-1}\pm7.02\times10^{-1}}$	$4.44\times10^{-1}\pm7.58\times10^{-2}$	$4.98\times10^{-1}\pm8.54\times10^{-2}$	$4.77\times10^{-1}\pm7.24\times10^{-2}$
F8	$5.44\times10^{-1}\pm8.11\times10^{-2}$	$5.81\times10^{-1}\pm6.22\times10^{-2}$	$\mathbf{5.12\times10^{-1}\pm6.76\times10^{-2}}$	$5.61\times10^{-1}\pm9.01\times10^{-2}$
F9	$8.15\times10^{-1}\pm1.30\times10^{-1}$	$7.47\times10^{-1}\pm1.82\times10^{-1}$	$1.00\pm2.05\times10^{-1}$	$9.84\times10^{-1}\pm1.90\times10^{-1}$

表 5.7: LZ における IGD の surrogate モデルありとの比較結果 (評価回数:10000 回)

関数	提案手法 (条件 1)	提案手法 (条件 2)	提案手法 (条件 3)	提案手法 (条件 4)
RE21	$8.79\times10^{-4}\pm1.30\times10^{-4}$	$8.79\times10^{-4}\pm1.32\times10^{-4}$	$7.84\times10^{-4}\pm1.18\times10^{-5}$	$8.45\times10^{-4}\pm1.58\times10^{-4}$
RE22	$7.89\times10^{-3}\pm1.01\times10^{-3}$	$8.52\times10^{-3}\pm1.55\times10^{-3}$	$9.10\times10^{-3}\pm1.84\times10^{-3}$	$8.82\times10^{-3}\pm1.37\times10^{-3}$
RE23	$\mathbf{1.86\times10^{-4}\pm2.77\times10^{-5}}$	$1.93\times10^{-4}\pm2.98\times10^{-5}$	$2.17\times10^{-4}\pm2.90\times10^{-5}$	$2.60\times10^{-4}\pm1.50\times10^{-4}$
RE24	$-1.54\times10^{-5}\pm8.54\times10^{-6}$	$-1.58\times10^{-5}\pm8.05\times10^{-6}$	$-9.67\times10^{-6}\pm7.49\times10^{-6}$	$\mathbf{-1.61\times10^{-5}\pm6.98\times10^{-5}}$
RE25	$1.16\times10^{-9}\pm3.98\times10^{-10}$	$1.03\times10^{-9}\pm2.43\times10^{-10}$	$\mathbf{5.53\times10^{-10}\pm1.83\times10^{-10}}$	$\mathbf{6.79\times10^{-10}\pm3.39\times10^{-10}}$
RE31	$\mathbf{3.37\times10^{-5}\pm1.46\times10^{-5}}$	$\mathbf{3.25\times10^{-5}\pm1.39\times10^{-5}}$	$3.82\times10^{-5}\pm2.05\times10^{-5}$	$4.83\times10^{-5}\pm1.83\times10^{-5}$
RE32	$3.69\times10^{-5}\pm7.97\times10^{-6}$	$\mathbf{3.01\times10^{-5}\pm6.87\times10^{-6}}$	$4.89\times10^{-5}\pm1.49\times10^{-5}$	$6.43\times10^{-5}\pm2.49\times10^{-5}$
RE33	$\mathbf{-1.04\times10^{-3}\pm4.41\times10^{-5}}$	$\mathbf{-1.06\times10^{-3}\pm4.64\times10^{-5}}$	$-9.69\times10^{-4}\pm8.45\times10^{-5}$	$\mathbf{-1.06\times10^{-3}\pm8.25\times10^{-5}}$
RE34	$1.05\times10^{-2}\pm1.30\times10^{-3}$	$1.17\times10^{-2}\pm2.05\times10^{-3}$	$1.16\times10^{-2}\pm1.65\times10^{-3}$	$1.06\times10^{-2}\pm1.37\times10^{-3}$
RE35	$\mathbf{1.74\times10^{-3}\pm2.26\times10^{-4}}$	$\mathbf{1.78\times10^{-3}\pm2.47\times10^{-4}}$	$2.27\times10^{-3}\pm5.12\times10^{-4}$	$2.33\times10^{-3}\pm9.13\times10^{-4}$
RE36	$\mathbf{3.54\times10^{-3}\pm2.20\times10^{-3}}$	$7.03\times10^{-3}\pm4.26\times10^{-3}$	$\mathbf{3.60\times10^{-3}\pm1.47\times10^{-3}}$	$6.14\times10^{-3}\pm3.73\times10^{-3}$
RE37	$2.77\times10^{-2}\pm4.68\times10^{-3}$	$2.87\times10^{-2}\pm4.31\times10^{-3}$	$2.83\times10^{-2}\pm3.43\times10^{-3}$	$2.83\times10^{-2}\pm3.35\times10^{-3}$

表 5.8: RE における HV の surrogate モデルありとの比較結果 (評価回数:50000 回)

関数	提案手法 (条件 1)	提案手法 (条件 2)	提案手法 (条件 3)	提案手法 (条件 4)
RE21	$4.38\times10^{-2}\pm7.41\times10^{-3}$	$3.96\times10^{-2}\pm5.59\times10^{-3}$	$4.32\times10^{-2}\pm7.83\times10^{-3}$	$4.15\times10^{-2}\pm5.35\times10^{-3}$
RE22	$3.77\times10^{-2}\pm4.80\times10^{-3}$	$3.76\times10^{-2}\pm5.07\times10^{-3}$	$3.88\times10^{-2}\pm4.10\times10^{-3}$	$3.72\times10^{-4}\pm3.71\times10^{-3}$
RE23	$6.54\times10^{-2}\pm3.42\times10^{-2}$	$6.28\times10^{-2}\pm3.72\times10^{-2}$	$5.85\times10^{-2}\pm3.80\times10^{-2}$	$5.08\times10^{-2}\pm2.67\times10^{-2}$
RE24	$2.35\times10^{-3}\pm4.14\times10^{-4}$	$\mathbf{2.29\times10^{-3}\pm3.53\times10^{-4}}$	$2.99\times10^{-3}\pm9.69\times10^{-4}$	$2.87\times10^{-3}\pm6.81\times10^{-4}$
RE25	$6.84\times10^{-2}\pm3.39\times10^{-2}$	$6.41\times10^{-2}\pm2.73\times10^{-2}$	$8.20\times10^{-2}\pm4.93\times10^{-2}$	$8.50\times10^{-2}\pm4.09\times10^{-2}$
RE31	$\mathbf{3.94\times10^{-3}\pm2.34\times10^{-3}}$	$\mathbf{3.71\times10^{-3}\pm2.50\times10^{-3}}$	$8.61\times10^{-3}\pm5.42\times10^{-3}$	$7.81\times10^{-3}\pm6.62\times10^{-3}$
RE32	$\mathbf{1.64\times10^{-3}\pm3.25\times10^{-4}}$	$\mathbf{1.45\times10^{-3}\pm5.12\times10^{-4}}$	$9.05\times10^{-3}\pm6.58\times10^{-3}$	$9.82\times10^{-3}\pm6.21\times10^{-3}$
RE33	$\mathbf{4.15\times10^{-3}\pm1.63\times10^{-3}}$	$5.42\times10^{-3}\pm9.13\times10^{-4}$	$5.10\times10^{-3}\pm1.54\times10^{-3}$	$6.25\times10^{-3}\pm2.56\times10^{-3}$
RE34	$1.99\times10^{-1}\pm2.11\times10^{-2}$	$2.01\times10^{-1}\pm3.35\times10^{-2}$	$2.09\times10^{-1}\pm2.10\times10^{-2}$	$2.04\times10^{-1}\pm2.29\times10^{-2}$
RE35	$3.41\times10^{-2}\pm7.79\times10^{-3}$	$3.92\times10^{-2}\pm7.62\times10^{-3}$	$3.82\times10^{-2}\pm5.66\times10^{-3}$	$3.79\times10^{-2}\pm6.53\times10^{-3}$
RE36	$\mathbf{1.66\times10^{-1}\pm3.44\times10^{-2}}$	$2.28\times10^{-1}\pm5.79\times10^{-2}$	$\mathbf{1.77\times10^{-1}\pm3.32\times10^{-2}}$	$\mathbf{1.83\times10^{-1}\pm4.22\times10^{-2}}$
RE37	$1.31\times10^{-1}\pm1.69\times10^{-2}$	$1.33\times10^{-1}\pm1.58\times10^{-2}$	$1.35\times10^{-1}\pm1.84\times10^{-2}$	$1.32\times10^{-1}\pm2.15\times10^{-2}$

表 5.9: RE における HV の surrogate モデルありとの比較結果 (評価回数:10000 回)

第 6 章

考察

6.1 既存手法と比較した実験結果に関する考察

LZ において提案手法と既存手法を比較した結果, すべての関数で既存手法が提案手法に比べて統計的に有意に性能が良かった. 特にすべての問題で MOEA/D または MOEA/D-DRA が既存手法に勝っていた. 一方, RE では MOEA/D または MOEA/D-DRA が NSGAII に比べて力を発揮できなかった問題で提案手法が良い結果を出していた. それは, RE21, RE23, RE24, RE25, RE32 である. MOEA/D には, 最適化したい問題が複雑なパレート界を持つ場合重みベクトルが割り当てられない方向には最適化ができずパレート界が広がらないという大きな問題点が存在する [38]. また, 親個体は近傍から選択するため局所解に陥るとそこから抜け出すことが難しいという問題点も存在する [38]. RE は実問題を基にして作られたベンチマーク問題であるから設計変数空間においても目的関数空間においてもパレート界は複雑な形をしていると考えられる. MOEA/D と提案手法で同じ数の重みベクトルを使っているため, ある方向に重みベクトルが割り当てられないということが発生した場合には提案手法でも二つの MOEA/D の手法でも性能が悪くなるはずである. そのため, ここで両者の性能を分けているものは局所解へ陥ってしまい最適化がそれ以上進まなくなってしまうことであると考えられる. MOEA/D では親個体を近傍からの選択するため多くの解が局所解へと陥ってしまうとそこから抜け出すことは難しい. それは同じような解からは同じような解しか生まれないからである. 一方, 提案手法では back-driving によって極値の方向に一気に更新するため局所解に陥ってしまってもそこから抜け出すことができると考えられる. その様子は RE25 を最適化した結果を見るとよくわかる. MOEA/D 系の既存手法は RE25 ではどちらも局所解に陥ってしまった. 図 6.1 はその様子を図示したものになっているが, オレンジ色の最適化した解は一部に留まってしまい, F_1 が小さくなるところには全く解が到達していない. 図 6.2 では提案手法を用いて RE25 の最適化を行った際の最終的な出力結果であるが局所解に陥らずに F_1 が小さくなる点まで到達できていることがわかる. 図 6.1 のようになってしまうと近傍の解がとても近い部分にしか存在しないためそこからさらに F_1 を小さくするような解は生まれにくい. そのため局所解からの脱却は難しくなる. 一方, 提案手法ではそもそも局所解に陥ることがなく最適化ができている. これは解の生成方法の違いからくるものであると考えられ, back-driving では理想点に近づくように解を時には大きく変更することがあるため局所解に陥ることがなく最適化ができていると考えられる. 同じような現象は RE21, RE23

でも見られた.

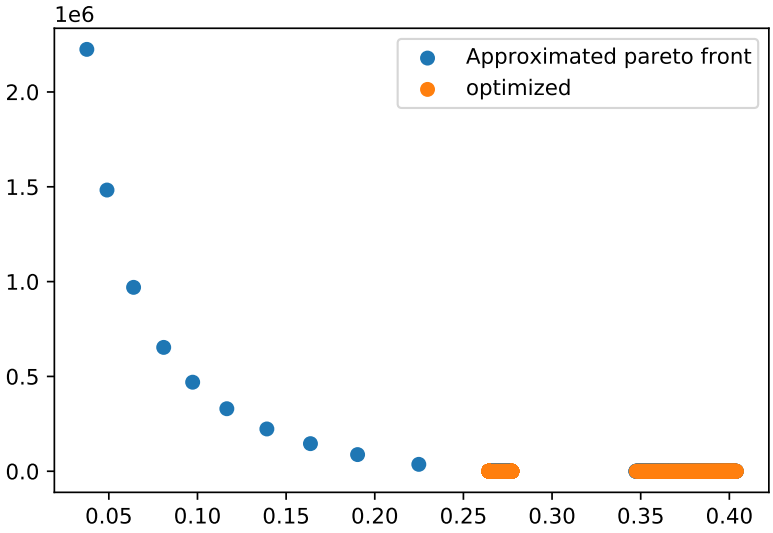


図 6.1: MOEA/D-DRA を用いた RE25 の最適化においての局所解に陥ってしまった様子

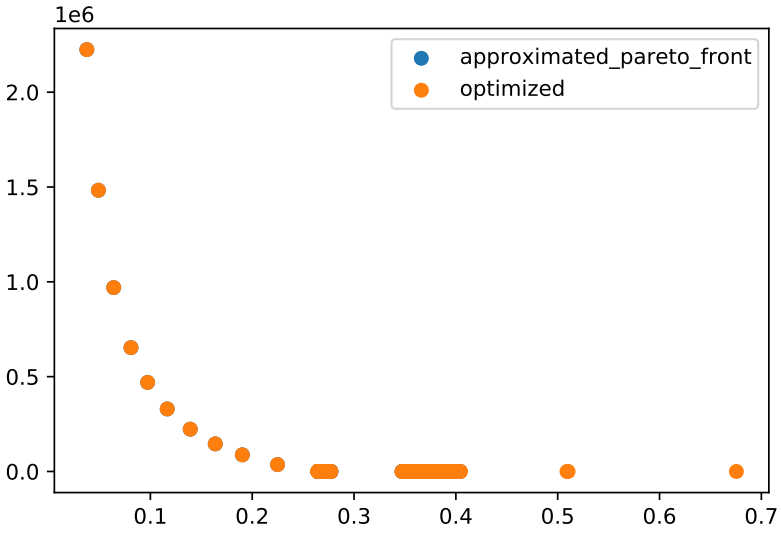


図 6.2: 提案手法を用いた RE25 の最適化においての局所解に陥らずに最適化ができた様子

RE24 では図 5.2 からもわかる通り従来の疑似最適パレート界を優越する解のうち F_1 を重要視する重みベクトルを持つ解が多かった. これは, RE24 において F_1 がとても単純な関数であるためニューラルネットワークでの学習がしやすく F_1 の最適化が促進されたためであると考えられる. RE24 における F_1 は式 (6.1) にある式で計算されるが, 学習点が足りないニューラルネットワークにおいても十分学習が可能なものである.

$$F_1 = x_1 + 120 \times x_2 \tag{6.1}$$

そこで, back-driving による解の生成が他の関数よりも高い性能を出すことができるようになったと考えられる. また, 従来の疑似最適パレート界を優越する解は (F_1,F_2)=(60,9) 辺りに集中していたが, ここは RE24 の疑似最適パレート界において曲線と直線の境目になっている. MOEA/D などの従来手法は人手で作成されたベンチマーク問題において性能が評価されることが多いため, LZ のようにきれいな形をしたパレート界を扱うことはとても得意である. しかし今回の曲線と直線の境目のような部分を最適化することは直線部分や曲線部分を最適化するときほど高い精度でできなかったと考えられ, 提案手法では既存手法では上手く最適化できなかった境目も最適化できたためこの部分に解が集中したと考えられる. 既存手法に比べて提案手法がそのようなパレート界を上手く扱うことができた理由に, 提案手法では次世代の解を生み出すために目的関数空間上の情報を考慮しないためということが考えられる. NSGAII や MOEA/D では混雑度ソートや目的関数空間上における近傍からの選択ということで次世代の親を選択し交叉を行っていたが, 提案手法では back-driving による解の改良が次世代の解を生み出す操作の主となっているため, 設計変数空間上での情報のみを扱うことで解を生み出している. それによって既存手法では最適化が進まなかった部分で提案手法では最適化ができたと考えられる.

RE33 において従来の疑似最適パレート界を優越した解に割り当てられていた重みベクトルを見てみると, 図 5.5 からは F_3 を重要視する重みベクトルの数がとても少なく, 0.0 から 0.1 の間の値を持つ重みベクトルが最も多いことが分かった. RE33 のパレート界全体と比較すると優越している解が分布している領域は, 図 5.3 を見ると F_3 の値がとても小さい領域になっているため, 重みベクトルのうち F_3 を重要視するものが少ないのは整合性が取れないように思われる. しかし, 提案手法によって最適化されたパレート界を見てみると, F_3 が大きい領域には広がっておらずほとんどが F_3 の値が小さい領域に分布していた. そのため F_3 を小さくすることよりも他の目的関数値を小さくする方が効果的であったためこのような重みベクトルの分布になったと考えられる. さらに図 5.4 からは F_2 を重要視する重みベクトルが多いことがわかる. 優越している解が従来のパレート界をどの関数で最もよく上回っていたかということを調べたところ, 新たに見つかった優越している解と従来の疑似最適パレート界で優越されている解の差の平均値は F_1 から順番に 8.42×10^{-3}, 6.92×10^{-1}, 8.37×10^{-2} となっていて, F_2 を最も改良していることがわかった. そのため, 重みベクトルの分布と良く改良された目的関数の間で整合性が取れていることがわかる.

6.2 Surrogate Model についての考察

5.3 節でも述べているが LZ において条件 1 と条件 2 の間または条件 3 と条件 4 の間で有意差が出ているのは, 評価回数が 10,000 回の F3, F4 と評価回数が 50,000 回の F4 で条件 2 が条件 1 を有意に上

回っているのみ, でその他の有意差は条件 1 と条件 3 のようにパラメータの有無による有意差になっていた. 多くの関数で Surrogate Model の有無による有意差が出なかった理由としてはニューラルネットワークの学習が不十分であることが挙げられる. 提案手法ではニューラルネットワークを用いて解をニューラルネットワーク上の極値の方向に更新すれば良いので詳細なランドスケープの学習は必ずしも必要が無い. 一方, ニューラルネットワークを Surrogate Model として用いる場合にはここではルーレット選択を用いているため多少の誤差は問題ないとはいえ, 突然変異した解の優劣を付ける必要がある. 突然変異は一つの設計変数のみで起こるように期待されて突然変異率を設定しているため, 設計変数が多数ある問題において突然変異によって一気に非常に悪い方向に進むということは期待されにくい. そこで解の優劣をうまく付けることができずにルーレット選択によって結局ランダムに突然変異された解を選択することと同じ状況に陥ってしまい Surrogate Model による解の選択はうまくいかなかったと考えられる.

RE において Surrogate Model を用いることで性能が上がったのは RE24 および RE33 における条件 3 と条件 4 間であり, Surrogate Model を用いると性能が下がってしまう結果が RE36 に見られた. これはニューラルネットワークによる解の評価値の推定により評価値の悪い解が選択されやすくなってしまっているということである. そこで RE36 において Surrogate Model の有無でパレート界の状態を比較した図が図 6.3 である. この図は 3 次元の図になっているため少々見辛いが F_2 の値が Surrogate Model 有りの方が悪くなっている様子が確認できる. RE36 における F_2 はすべての設計変数の最大値となっている. このように設計変数の最大値を目的関数とする問題は他には存在せずその他の関数では設計変数同士の多項式である. そこでニューラルネットワークでは少ない学習点からは Max 関数を上手く学習することができず, 設計変数の値が大きくても他の F_1 または F_3 が小さくなるようなものが選択されやすくなるように学習されてしまい, Surrogate Model を用いた方が悪くなるような結果となったと考えられる.

RE24 において Surrogate Model を用いることで性能が上がった理由は, RE24 において他の関数に比べて提案手法の性能が上がった理由と同じく RE24 の F_1 が単純な関数であるため学習点が少ないニューラルネットワークでも十分に学習可能であることが挙げられる. これによって少なくとも F_1 において悪くなってしまう解を排除することに成功し, 条件 3 に比べて条件 4 の性能が良かったと考えられる. 条件 1 と条件 2 の間では有意差が出なかったことに対して条件 3 と条件 4 の間では有意差が出た理由としては, パラメータレスにすることでパレート界を広げることに時間がかかってしまうことに対して F_1 の特徴を掴んだ Surrogate Model の利用により適合度が低い解を評価する回数が減りパレート界の広がりが促進されたことが考えられる.

最後にスケーリング幅を決定するパラメータについて考察する. スケーリング幅に関するパラメータを予め決める場合, back-driving による解の改良に用いる参照点も予め決まるため, スケーリング幅を人手で設定する場合には最初から理想点がわかった状態での最適化となる. 一方, スケーリング幅を自動で調整する場合には参照点も世代を経るにつれて更新されていくので前情報なしの最適化となる. 特に RE では目的関数の値が軸ごとに大きく異なるため予め情報がある最適化と無い最適化では難易度が変わると考えられる. これが多くの関数においてスケーリング幅の自動調整する方 (条件 3 と条件 4) が予め与える方 (条件 1 と条件 2) に比べて有意に性能が悪くなってしまっていた理由である. ただ , 予めパ

ラメータとして与えるスケーリング幅は何度かアルゴリズムを動かしてパレート界を見ながら調整する必要があるため適切なパラメータを設定するまでに時間がかかる. RE25 では条件 3 および 4 が条件 1 および 2 に比較して有意に性能が良くなっていたが, これはパラメータを適切に設定できなかった結果, 自動調整した方が良いスケーリング幅になったためと考えられる. また, LZ において評価回数 10,000 回では参照点がまだ定まっていないため最適化が充分に進んでおらず条件 3 および 4 での結果が条件 1 および 2 の結果よりも悪くなる関数がいくつか見られた. スケーリング幅の自動調整の問題点として特に探索初期ではスケーリング幅が世代ごとに大きく変わるため, 前世代で学習したニューラルネットワークの重みが意味を為さなくなってしまう恐れがある. なぜなら, スケーリング幅が異なると実際の目的関数値は変わらなくてもニューラルネットワークの出力に対する正解データの値が変わってしまうからである. ただ, 探索初期では探索領域も世代ごとに大きく異なる可能性もあると考えられるため, スケーリング幅の問題を抜きにしても毎世代ニューラルネットワークの重みが初期化されているためスケーリング幅の変更により前世代のニューラルネットワークの重みが使用できなくなってしまうことはさほど問題にはならない.

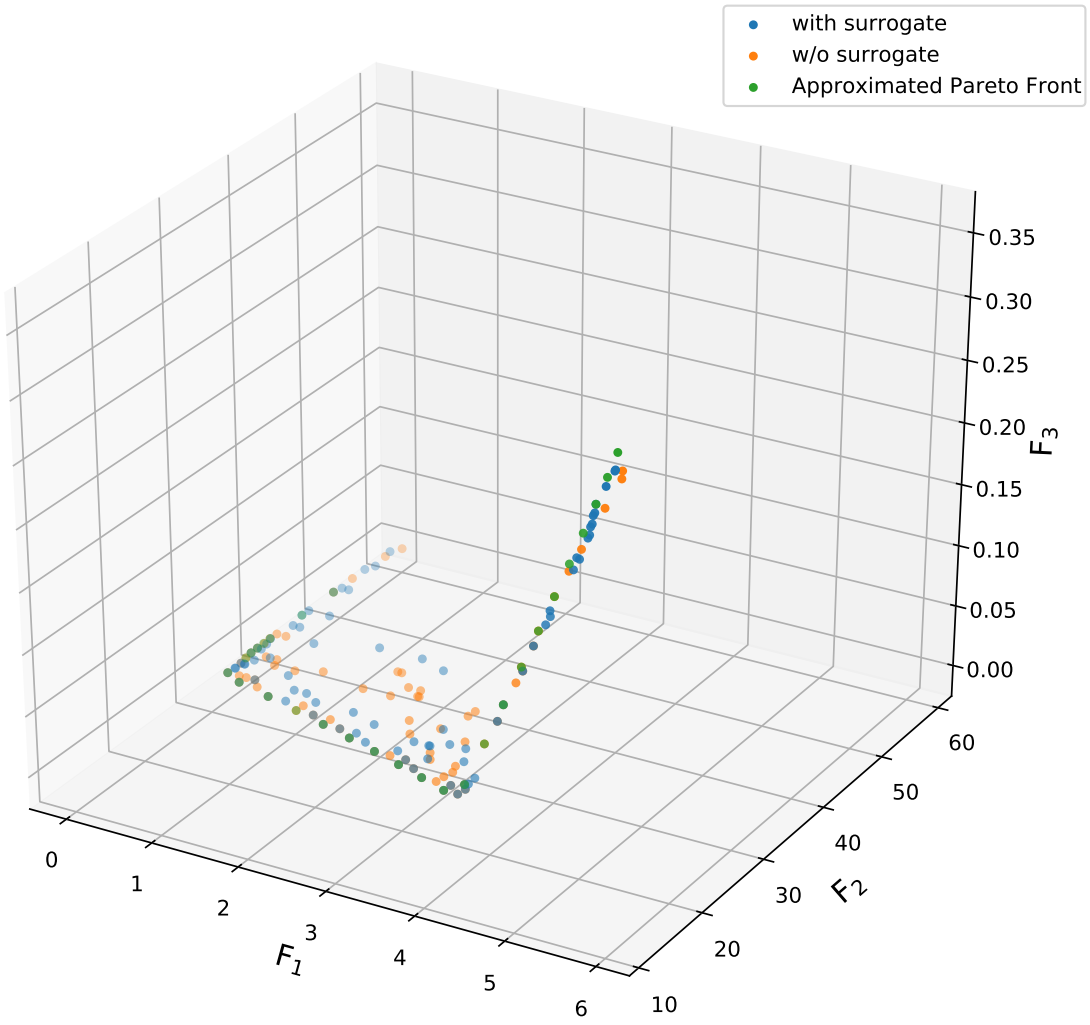


図 6.3: RE36 において Surrogate Model の有無でパレート界を比較した結果

第 7 章

結論

7.1 まとめ

本研究ではニューラルネットワークを用いて設計変数から目的関数値へと至るランドスケープを学習しながら単目的最適化問題を最適化する手法を多目的最適化問題に応用して多目的の問題を最適化する手法を提案した. 解の改良はニューラルネットワークの重みを固定して入力を変化させることで最適化を行う back-driving という手法を多目的最適化へと応用した. 結果として, LZ では NSGAII, MOEA/D, MOEA/D-DRA と比較して良くなっているような問題は無かったが, 実問題をベンチマーク関数に落とし込んだ RE ではいくつかの問題で他手法を統計的に有意に上回る結果が得られた. 特に RE24 と RE33 では今まで報告されているパレート界よりも高い HV を持つパレート界を発見することができていた. ただ, 既存手法に比べて性能が悪くなっていたとはいえ HV の支配率は高い値を示していたため, 全く最適化ができていなかった訳ではない. また, RE の一部の問題では既存手法が局所解に陥ってしまったり一部のパレート界を充分に最適化できなかった様子が見られたが, 提案手法ではそのような問題についても局所解に陥ることが無く探索を進めることができていた. このように, 実問題を基にして作成されたベンチマーク問題で既存手法よりも良い成績を出すことができたため, 本論文の提案手法は人手で作成されたベンチマーク問題に留まらず実問題でも高い性能を発揮できることが期待される.

同時にニューラルネットワークを Surrogate Model として用いて最適化を行った場合と用いなかった場合で有意差が得られるような結果はほとんど無かった. これは, 提案手法においてランドスケープを正確に学習するためにはニューラルネットワークにおける学習点が少ないためであると考えられた. また, 本論文では提案手法においてパラメータを削減する手法も提案していた. この手法でも十分に最適化は進んでいたと言え, 一部でパラメータ設定が適切に行えていなかったと思われる問題ではパラメータレスの手法の方が勝っていた.

7.2 今後の展望

今後の課題としては, モデル化しながら最適化をするメリットとして別の問題を最適化した際の知識を他の問題に適用することでより素早い探索を可能にするということが考えられる. 学習されたニュー

ラルネットワークには最適化した問題の知識が溜まっているためその知識を他の問題へと転用することが可能であると考えられる. ここで言う似たような問題とはドメインは同じであるが設計変数の数が増えるような問題や目的関数が増えるような問題を指している. ニューラルネットワークを用いる利点として設計変数や目的関数が増減しても転移学習が容易に行えることが挙げられる. 今回の論文ではこのようなニューラルネットワークの利点を充分に活かすことができなかったため, 今後の課題としてはニューラルネットワークへと蓄積された知識の転用ということが考えられる.

謝辞

本研究を進めるにあたり, 多くの方々からお力添えを賜りました. 学部 4 年次から 3 年間指導してくださった指導教員である伊庭斉志教授には本研究の実施の機会をいただき, 最後までご指導をいただきました. 土肥助教を始めとする伊庭研究室の皆様や OB の方々とも何気ない会話の中で研究のヒントを得ることができました.

また, 進化計算学会の皆様には新型コロナ感染症にて以前とは日常が変化してしまった現状においてもオンラインにて進化計算シンポジウムを開催してくださり, 研究について議論をする場を設けてくださったことに感謝致します. 他の大学の方々とも議論することでより研究を深めることができました.

皆様に深く感謝致します.

参考文献

[1] Tomoaki Tatsukawa, Taku Nonomura, Akira Oyama, and Kozo Fujii. Multi-objective aeroacoustic design exploration of launch-pad flame deflector using large-eddy simulation. *Journal of Spacecraft and Rockets*, 53(4):751–758, 2016.

[2] S Rodrigues, Pavol Bauer, and Peter AN Bosman. Multi-objective optimization of wind farm layouts–complexity, constraint handling and scalability. *Renewable and sustainable energy reviews*, 65:587–609, 2016.

[3] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[4] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, 11(6):712–731, 2007.

[5] Shumeet Baluja. Deep learning for explicitly modeling optimization landscapes. *arXiv preprint arXiv:1703.07394*, 2017.

[6] 池上 貴志. 用語解説（第 87 回テーマ：多目的最適化問題）／電力・エネルギー部門誌 2018 年 6 月号目次. **電気学会論文誌 B（電力・エネルギー部門誌）**, 138(6):NL6_6–NL6_6, 2018.

[7] Ryoji Tanabe and Hisao Ishibuchi. An easy-to-use real-world multi-objective optimization problem suite. *Applied Soft Computing*, 89:106078, 2020.

[8] Akira Oyama, Takehisa Kohira, Hiromasa Kemmotsu, Tomoaki Tatsukawa, and Takeshi Watanabe. Simultaneous structure design optimization of multiple car models using the k computer. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–4, 2017.

[9] `https://www.mazda.com/en/innovation/technology/skyactiv/skyactiv-body/`.

[10] T. Kohira. Applications and challenges of multidisciplinary approximate optimization in car body development. *Workshop on Multiobjective Design Exploration for Real-World Design Optimization Problems*, 2012.

[11] Bin Wang, Yanan Sun, Bing Xue, and Mengjie Zhang. Evolving deep neural networks by multi-objective particle swarm optimization for image classification. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '19, page 490–498, New York, NY, USA, 2019. Association for Computing Machinery.

[12] Hisao Ishibuchi, Yu Setoguchi, Hiroyuki Masuda, and Yusuke Nojima. Performance of decomposition-

based many-objective algorithms strongly depends on pareto front shapes. *IEEE Transactions on Evolutionary Computation*, 21(2):169–190, 2017.

[13] Abdullah Konak, David W Coit, and Alice E Smith. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability engineering & system safety*, 91(9):992–1007, 2006.

[14] Kalyanmoy Deb and Himanshu Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, 2014.

[15] Himanshu Jain and Kalyanmoy Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18(4):602–622, 2014.

[16] Indraneel Das and John E Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM journal on optimization*, 8(3):631–657, 1998.

[17] Qingfu Zhang, Wudong Liu, and Hui Li. The performance of a new version of moea/d on cec09 unconstrained mop test instances. In *2009 IEEE congress on evolutionary computation*, pages 203–208. IEEE, 2009.

[18] Hui Li and Qingfu Zhang. Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii. *IEEE transactions on evolutionary computation*, 13(2):284–302, 2008.

[19] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, 8(2):173–195, 2000.

[20] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, volume 1, pages 825–830. IEEE, 2002.

[21] Saúl Zapotecas-Martínez, Carlos A. Coello Coello, Hernán E. Aguirre, and Kiyoshi Tanaka. A review of features and limitations of existing scalable multiobjective test suites. *IEEE Transactions on Evolutionary Computation*, 23(1):130–142, 2019.

[22] S. Huband, P. Hingston, L. Barone, and L. While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006.

[23] `https://ryojitanabe.github.io/reproblems/`.

[24] FY Cheng and XS Li. Generalized center method for multiobjective engineering optimization. *Engineering Optimization*, 31(5):641–661, 1999.

[25] Hossain M Amir and Takashi Hasegawa. Nonlinear mixed-discrete structural optimization. *Journal of Structural Engineering*, 115(3):626–646, 1989.

[26] BK Kannan and Steven N Kramer. An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. 1994.

[27] Jouni Lampinen and Ivan Zelinka. Mixed integer-discrete-continuous optimization by differential evolution. In *Proceedings of the 5th international conference on soft computing*, pages 71–76, 1999.

[28] CA Coello Coello and Gregorio Toscano Pulido. Multiobjective structural optimization using a microgenetic algorithm. *Structural and Multidisciplinary Optimization*, 30(5):388–403, 2005.

[29] Tapabrata Ray and KM Liew. A swarm metaphor for multiobjective design optimization. *Engineering optimization*, 34(2):141–153, 2002.

[30] Xingtao Liao, Qing Li, Xujing Yang, Weigang Zhang, and Wei Li. Multiobjective optimization for crash safety design of vehicles using stepwise regression model. *Structural and multidisciplinary optimization*, 35(6):561–569, 2008.

[31] Azarm Farhang-Mehr and Shapour Azarm. Entropy-based multi-objective genetic algorithm for design optimization. *Structural and Multidisciplinary Optimization*, 24(5):351–361, 2002.

[32] Kalyanmoy Deb and Aravind Srinivasan. Innovization: Innovating design principles through optimization. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1629–1636, 2006.

[33] Rajkumar Vaidyanathan, Kevin Tucker, Nilay Papila, and Wei Shyy. Cfd-based design optimization for single element rocket injector. In *41st Aerospace Sciences Meeting and Exhibit*, page 296, 2003.

[34] Michael G Parsons and Randall L Scott. Formulation of multicriterion design optimization problems for solution with scalar numerical optimization methods. *Journal of Ship Research*, 48(01):61–76, 2004.

[35] Tapabrata Ray, Kang Tai, and Kin Chye Seow. Multiobjective design optimization by an evolutionary algorithm. *Engineering Optimization*, 33(4):399–424, 2001.

[36] Antonio Benítez-Hidalgo, Antonio J Nebro, José García-Nieto, Izaskun Oregi, and Javier Del Ser. jmetalpy: A python framework for multi-objective optimization with metaheuristics. *Swarm and Evolutionary Computation*, 51:100598, 2019.

[37] C.M. Fonseca, L. Paquete, and M. Lopez-Ibanez. An improved dimension-sweep algorithm for the hypervolume indicator. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1157–1163, 2006.

[38] Qian Xu, Zhanqi Xu, and Tao Ma. A survey of multiobjective evolutionary algorithms based on decomposition: Variants, challenges and future directions. *IEEE Access*, 8:41588–41614, 2020.

発表文献

[1] Eito Suda, Hitoshi Iba “Neural architecture search and weight adjustment by means of Ant Colony Optimization”, 5th International Conference on Intelligent Informatics and BioMedical Sciences, 184-191, 2020

[2] Eito Suda, Hitoshi Iba “Neural architecture search and weight adjustment by means of Ant Colony Optimization”, Journal of Information and Communications Engineering (JICE), 6 (2): 384-390, December 31, 2020

[3] 須田 瑛斗, 伊庭 斉志「ニューラルネットワークを用いた多目的最適化に関する研究」, 進化計算シンポジウム 2021, 183-190, 2021