第2章　多目的最適化

多目的最適化とは，複数の目的について，同時に最適化することである．しかしながら，同時に最適になるような目的は多くない．例えば，計算機の最適化を考える．計算機の価格と計算速度を目的とし，計算速度の向上と価格の低コスト化を目指すものとする．この場合，計算速度向上の目的を満たすためには，高性能な演算装置を搭載する必要があり，コストの観点では悪化してしまう恐れがある．このように片方の目的を満たすことで，もう一方の目的を満たさなくなるような，トレードオフ関係を持つものが多い．こうした互いにトレードオフの関係を持つ複数の目的を，同時に最適化することを多目的最適化という．

2.1　多目的最適化問題

多目的最適化問題とは，与えられた設計変数と制約条件の元，複数の目的関数を最小（最大）化する問題のことを言い，式 (2.1.1) によって定義される．ここで，$\boldsymbol{f}$ は目的関数，m は目的関数の数，g_j は制約条件を示す．多目的最適化問題では，互いにトレードオフの関係を持つ目的関数を最適化するため，単一な最適解が得られるわけではなく，パレート最適と呼ばれる概念が必要となる．したがって，多目的最適化問題では，複数個のパレート最適解を得ることが目標となる．この得られた複数個のパレート最適解から，ユーザー嗜好の解を選択することとなる．次節でパレート最適解について詳しく述べる．

$$\begin{cases} minimize\ (maximize) & \boldsymbol{f}(\boldsymbol{x}) & = & ((f_1(\boldsymbol{x}), \cdots, f_m(\boldsymbol{x}))^T, \\ subject\ to & & & \forall j, g_j(\boldsymbol{x}) \leq 0. \end{cases} \tag{2.1.1}$$

2.2 パレート最適解

パレート最適解とは,「解の優越性に基づいた，最適性の高い解」のことである．パレート最適解は，以下で表される解の優越関係により定義される解のことである．なお，以下の式では全ての目的関数に関して，最小化を仮定した優越関係である．

$$\forall m[f_m(x_1) \leq f_m(x_2)] \text{ かつ } \exists m[f_m(x_1) < f_m(x_2)] \tag{2.2.1}$$

式 (2.2.1) を満たす時，x_1 は x_2 に優越すると言う．特に，$\forall m[f_m(x_1) < f_m(x_2)]$ である時，x_1 は x_2 に強い意味で優越するという．これらの優越関係を用いて，パレート最適解を以下のように定義する．

(1)x^* に強い意味で優越する x が存在しない時，x^* を (弱) パレート最適解と呼ぶ
(2)x^* に優越する x が存在しない時，x^* を (強) パレート最適解と呼ぶ

目的関数が 2 つの時の，パレート最適解の例を図 2.2.1 に示す．

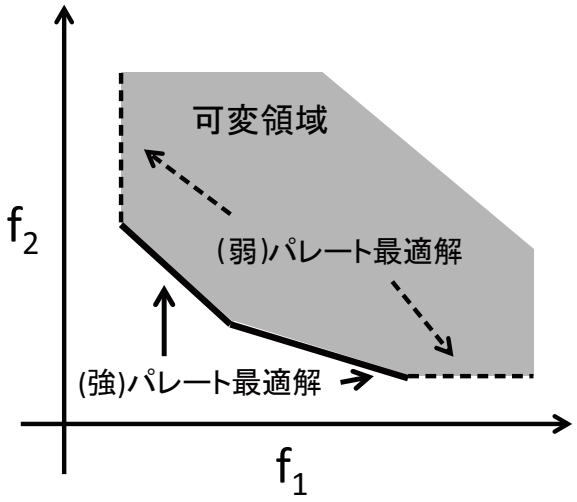

図 2.2.1: パレート最適解の例

2.3　進化的多目的最適化アルゴリズム

多目的最適化問題を解く場合，パレート最適解集合が求められることが望ましい．そこで，多点探索かつ関数の勾配情報を必要としない進化計算を用いてパレート最適解集合を求める方法が提案されている．進化計算とは，自然界における生物の進化を模倣した計算手法の総称であり，特に多目的最適化に焦点を当てた手法は進化的多目的最適化アルゴリズム（Evolutionary Multiobjective Optimization Algorithms, EMOAs）と呼称される．EMOA は多点探索を行うことのできる確率的最適化手法であり，多目的最適化問題に対して有効性が示されており，近年注目を集めている．本研究では，EMOA の中でも遺伝的アルゴリズムを用いて計算を行う．

2.4　遺伝的アルゴリズム

遺伝的アルゴリズ（Genetic Algorithm, GA）とは,「世代を経るごとに，環境に適さない個体は淘汰され，環境に適した個体同士が生存繁殖することで，より最適な個体となる」という自然界のプロせずから着想を得た最適化アルゴリズムのことである．実際のアルゴリズムでは，個体集合に遺伝的操作（交叉・突然変異等）を加えることにより，確率的に近似最適解を求めている．図 2.4.1 にアルゴリズムのフローチャートを示す．

初期集団の生成

「初期集団の生成」では，設定された数 (集団サイズ) の個体 (解) をランダムに生成する．ここで生成された個体を初期集団と呼ぶ．

個体の評価

「個体の評価」は，個体集団の中で，どの個体が優れているかを評価する操作である．個体を評価するために，評価値を算出し，適合度を求める．適合度は，環境にどれだけ適応しているかを判断する値であり，個体の生き残りやすさを定量的に定めている．また，制約条件に反しているかどうかについてもこのステップで評価する．

親個体の選択

「親個体の選択」では,「個体の評価」で算出された適合度を基に，子個体を生成する

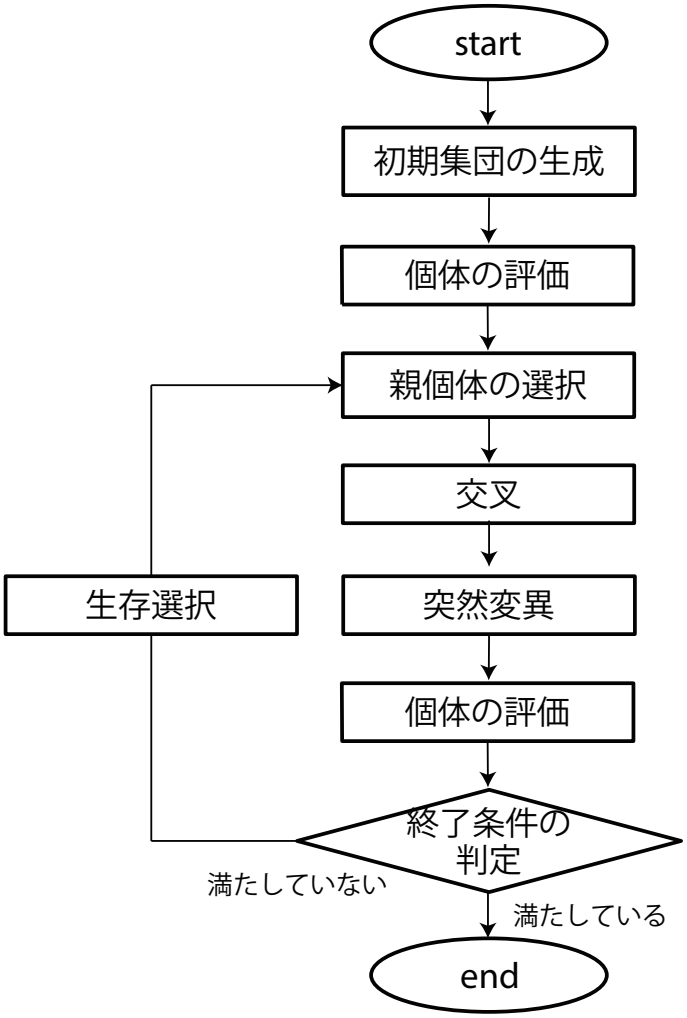

図 2.4.1: 遺伝的アルゴリズム（GA）のフローチャート

親個体を選択する操作である．適応度の高い個体ほど親個体に選択されやすく，適応度が低い個体ほど，親個体として選択されにくくなる．

交叉

「交叉」では,「親個体の選択」で選ばれた親個体ペアを用いて，子個体を生成する操作である．この操作は，自然解での生殖を模倣したものであり，子個体は親個体の特徴を受け継ぐ．両親個体の特徴を受け継ぐことで，次世代の個体の適応度を上げることを目的としている．

突然変異

「突然変異」では，ある確率 (突然変異率) で個体にランダムな変化を与える操作である．この操作により，局所解に陥ってしまうリスクを回避する．

終了条件の判定

「終了条件の判定」では，このアルゴリズムをどれだけ繰り返し行うかの条件を判定する．例として，ある世代数に達したときに終了させる条件や，ある計算時間に達した場合終了するといった条件がある．

生存選択

「生存選択」では，次世代に生存する個体を選択する.「個体の評価」で得られた各個体の適合度を基に，次世代で親個体の選択に用いる集合を決定する．

2.5 NSGA-II

NSGA-II[?] は，2002 年に Deb らによって提案された RCGA の一つである．本研究では，NSGA-II を用いて数値実験を行っている．

NSGA-II は，NSGA [?] の改良版として提案されたアルゴリズムであり，解の優越性に基づいて個体を評価することが特徴である．また，生存選択の際に用いられるエリート主義の導入や，解の多様性を維持するためのニッチング手法に，混雑距離 (Crowding Distance) を用いていることも特徴の一つである．NSGA-II の計算の流れを以下に示す．さらに，NSGA-II の進化のイメージを図 2.5.1 に示す．また，NSGA-II で用いられる交叉手法 SBX（Simulated Binary crossover）と突然変異手法 polynomial mutation，混雑距離については後述する．

(1)$gen = 0$ として，初期集団 P_{gen} を生成する (集団サイズを N とする)

(2) 個体を評価し，優越関係から親個体を選択する

(3) 交叉と突然変異を用いて個体を更新し，集団 Q_{gen} を生成する

(4)$R_{gen} = P_{gen} + Q_{gen}$ を優越関係を用いて個体を評価し，次の世代集合 P_{gen+1} を生成する

(4.1) 優越関係を用いて非劣解集合を順に取り出していき，それぞれ $F_1, F_2, \cdots$ という小集合を作る

(4.2)$|F_1| + |F_2| + \cdots + |F_n| = N$ となった場合，$P_{gen+1} = F_1 + F_2 + \cdots + F_n$ とし，(5) に進み，それ以外の場合 (4.3) に進む

(4.3)$|F_1| + |F_2| + \cdots + |F_n| \geq N$ となった場合，混雑距離を用いて，F_n 内の個体に優先度をつける

(4.4)F_n 内の優先度の高い個体から順に，$N - (|F_1| + |F_2| + \cdots + |F_{n-1}|)$ 個取り出した個体集合を新たに $F\prime_n$ とし，$P_{gen+1} = F_1 + F_2 + \cdots + F_{n-1} + F\prime_n$ とし，(5) に進む

(5)$gen = gen + 1$

(6)(2)〜(5) を終了条件を満たすまで繰り返す

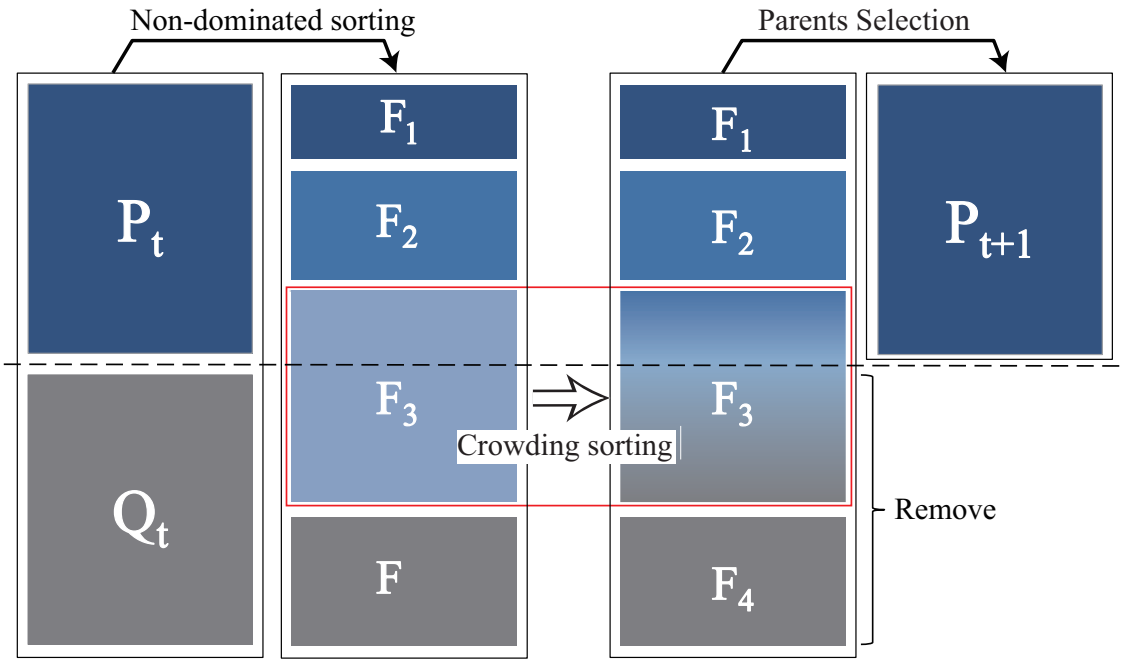

図 2.5.1: NSGA-II における進化のイメージ図

SBX

SBX(Simulated Binary crossover)[?] は，1995 年に Deb らによって提案された交叉法の一つであり，NSGA-II 等多くの GA に採用されている．SBX を用いた子個体生成の手順を示す．

はじめに，一様乱数 u を $0 \leq u \leq 1$ の範囲で生成する．その後，以下で示される確率分布関数と乱数 u を用いて，β_q を算出する．

$$\beta_q = \begin{cases} (2u)^{\frac{1}{\eta_c+1}} & \text{if } \ u < 0.5; \\ (\frac{1}{2(1-u)})^{\frac{1}{\eta_c+1}} & \text{otherwise.} \end{cases} \tag{2.5.1}$$

ここで，式 (2.5.1) 中の η_c は固有パラメータであり，生成される子個体の分散を制御することができる．η_c が大きいほど，子個体は親個体付近に生成されやすく，η_c が小さいほど，子個体は親個体から遠ざかったところに生成されやすい．

次に，算出した β_q を用いて，子個体を生成する．ここで子個体を $x^{(1,t+1)}$，$x^{(2,t+1)}$，親個体を $x^{(1,t)}$，$x^{(2,t)}$ とすると，子個体は次の式で算出される．

$$x^{(1,t+1)} = 0.5[(1+\beta_q)x^{(1,t)} + (1-\beta_q)x^{(2,t)}] \tag{2.5.2a}$$

$$x^{(2,t+1)} = 0.5[(1-\beta_q)x^{(1,t)} + (1+\beta_q)x^{(2,t)}] \tag{2.5.2b}$$

Polynomial mutation

polynomial mutation[?] は，1996 年に Deb らによって提案された突然変異法の一つであり，NSGA-II 等多くの GA に用いられている．polynomial mutation を用いた個体生成の手順を示す．

はじめに，式 (2.5.3) に従って親個体 x^t から δ を算出する．x^t_{Upper}，x^t_{Lower} は親個体 x^t の定義域の最大値，最小値を表している．

$$\delta = \frac{min(x^t_{Upper} - x^t, x^t - x^t_{Lower})}{x^t_{Upper} - x^t_{Lower}} \tag{2.5.3}$$

次に，一様乱数 u を $0 \leq u \leq 1$ の範囲で生成し，以下で示される確率分布関数と乱数 u を用いて，δ_q を算出する．

$$\delta_q = \begin{cases} [2u + (1-2u) * (1-\delta)^{\eta_m+1}]^{\frac{1}{\eta_m+1}} - 1 & \text{if } \ u < 0.5; \\ 1 - [2(1-u) + 2(u-0.5) * (1-\delta)^{\eta_m+1}]^{\frac{1}{\eta_m+1}} & \text{otherwise.} \end{cases} \tag{2.5.4}$$

ここで，式 (2.5.4) 中の η_m は固有パラメータであり，生成される子個体の分散を制御することができる．η_m が大きいほど，子個体は親個体付近に生成されやすく，η_c が小さいほど，子個体は親個体から遠ざかったところに生成されやすい．次に，算出した β_q を用いて，子個体を生成する．子個体を x^{t+1} とすると，子個体は式 (2.5.5) で算出される．

$$x^{t+1} = x^t + \delta_q(x^t_{Upper} - x^t_{Lower}) \tag{2.5.5}$$

混雑距離 (Crowding Distance)

混雑距離とは，目的関数空間内での個体間の距離を計算するものである．NSGA-II では，混雑距離を算出し，生き残る個体の優先度を決めることで，個体が偏って生成されてしまうのを防ぐ，ニッチング法に用いられている．個体 d_i の混雑距離は式 (2.5.6), (2.5.7) で算出される．なお，目的関数の数を m，S を非劣解集合，$|S|$ を非劣解の数，f_k を第 k 目的関数とする．また，i 番目の個体の混雑距離のイメージを図 2.5.2 に示す．

$$d_1 = d_{|S|} = \infty \tag{2.5.6}$$

$$d_i = \sum_{k=1}^{m} \sum_{i=2}^{|S|-1} |f_k(i+1) - f_k(i-1)| \tag{2.5.7}$$

図 2.5.2 のように隣り合う個体を用いて，i 番目の個体の周りに骨格を取る．その後，骨格の片側の 2 辺を足し合わせた値が，i 番目の個体の混雑距離となる．

混雑距離を求めることで，各個体の疎密度が算出される．したがって，混雑距離によってソートを行うことで，疎な解が生存しやすくなるため，解の多様性を維持する機構として用いられている．

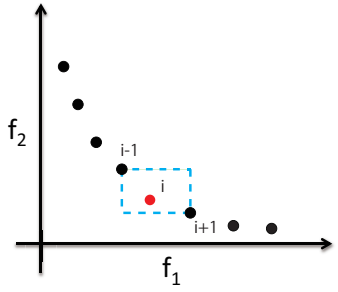

図 2.5.2: 混雑距離のイメージ図

2.6 MOEA/D

MOEA/D (Multi-objective Evolutionary Algorithm Based on Decomposition)[?] は 2007 年に Zhang らによって提案された RCGA の一つである．本研究においても，数値実験に使用している．

MOEA/D では探索空間上に重みベクトルを均一の間隔で分布させ，それらの重みベクトルとスカラー適応度関数を用いて，多目的最適化問題をベクトル毎の単一目的最適化の部分問題へと帰着させる．その後，各重みベクトルに対し，ユークリッド距離の近いベクトルを近傍とし，近傍内において遺伝的操作（交叉や突然変異）を行う．また，MOEA/D では最適化に用いる個体群とは別に，計算中に得られた非劣解群をアーカイブとして保持する機構を持つ．

MOEA/D の計算の流れを以下に示す．

Step 1 : 初期化

1.1 : 重みベクトル生成

探索に用いる N 個の重みベクトル $\boldsymbol{\lambda}$ を式 (2.6.1) により生成する．

$$\left.\begin{array}{rcl} \boldsymbol{\lambda} & = & (\lambda_1, \lambda_2, \cdots, \lambda_k) \\ \sum_{i=1}^{k} \lambda_i & = & 1 \\ N & = & {}_{H+k-1}C_{k-1} \\ where \quad \lambda_i & \in & \left\{ \dfrac{0}{H}, \dfrac{1}{H}, \cdots, \dfrac{H}{H} \right\} \end{array}\right\} \tag{2.6.1}$$

ここで，k は目的関数の数，H は分割数を表す．

1.2 : 近傍ベクトル生成

1.1 で作成した各ベクトル間のユークリッド距離を計算し，各ベクトル $\boldsymbol{\lambda}^i$ に最も近い T 個のベクトル $\boldsymbol{\lambda}^{i_1}, \cdots, \boldsymbol{\lambda}^{i_T}$ を λ^i の近傍とする．

1.3 : 初期個体群生成

ベクトルの総数 N 個と等しい数の初期個体をランダムに生成し，1 つのベクトルに対し 1 つの個体を対応させる．

1.4 : 参照点更新

参照点 $\boldsymbol{z}^*$ の更新を式 (2.6.3) によって行う．

$$\left.\begin{aligned} \boldsymbol{z}^* &= (z_1^*, z_2^*, \cdots, z_k^*) \\ z_j^* &= \max\{f_j(\boldsymbol{x}) | \boldsymbol{x} \in \boldsymbol{X}\} \end{aligned}\right\} \tag{2.6.2}$$

ここで，$f_j(\boldsymbol{x})$ は個体 $\boldsymbol{x}$ の j 番目の目的関数，$\boldsymbol{X}$ は設計変数空間内での実行可能領域を表す．

Step 2 : 遺伝的操作

2.1 : 親個体選択

ベクトル $\boldsymbol{\lambda}^i$ の近傍ないからランダムに 2 つの個体を選択する．

2.2 : 子個体生成

2.1 で選択された親個体から交叉（SBX）と突然変異（polynomial mutation）を行い子個体を生成する．

2.3 : 参照点更新

式 (2.6.3) を用いて参照点を更新する．

2.4 : 近傍解更新

2.2 で生成された子個体と $\boldsymbol{\lambda}^i$ 近傍内のベクトルに対応する個体をスカラー適応度関数を用いて比較し更新する．スカラー適応度関数は主に Tchebycheff Approach と Weighted Sum Approach がある．本研究で用いる Tchecycheff Approach を以下に示す．

$$\left.\begin{array}{rccl} minimize & g^{te}(\boldsymbol{x}|\boldsymbol{\lambda},\boldsymbol{z}^*) & = & \max\limits_{1\leq i\leq m}\{\lambda_i|f_i(\boldsymbol{x})-z_i^*|\} \\ sub.to & x\in X & & \\ where & z^* & = & (z_1^*,\cdots,z_k^*)^T \end{array}\right\} \tag{2.6.3}$$

2.5 : アーカイブ更新

非劣解となった子個体をアーカイブに追加し，アーカイブ内の劣解を削除する．

Step 3 : 終了判定

上述のアルゴリズムにおいて，Step2 の操作を終了判定を満たすまで繰り返すことで解の探索を行う．

2.7 解の評価指標について

本項では，本研究で用いた多目的最適化における解の評価指標について述べる．

2.7.1 GD

Generational Distance (GD)[**?**] とは解の収束性を評価する際に用いる評価指標である．GD は，得られた非劣解集合の各点から真のパレートフロントまでの最短距離の平均値を算出する指標である．値が小さいほど収束性が良いことを示し，0 が最小値となる．GD は式 (2.7.1) で定式化される．

$$\begin{array}{rcl} GD(\boldsymbol{S},\boldsymbol{P}) & = & \dfrac{\left(\sum_{i=1}^{|\boldsymbol{S}|} d_i^2\right)^{1/2}}{|\boldsymbol{S}|} \\ d_i & = & \min_{\vec{p}\in\boldsymbol{P}} ||F(\vec{s_i})-F(\vec{p})|| \end{array} \tag{2.7.1}$$

$\boldsymbol{S}$ は得られた非劣解集合，$\boldsymbol{P}$ は理論上のパレートフロントの点集合を示す．また，d_i は $\boldsymbol{S}$ の各点から $\boldsymbol{P}$ の最短距離となる点への距離を示し，$|\boldsymbol{S}|$ は S の点数を示す．

2.7.2 IGD

Inverted Generational Distance (IGD)[?] とは解の収束性と解の多様性を同時に評価できる総合評価指標である．IGD は，真のパレートフロントの各点から得られた非劣解集合までの最短距離の平均値を算出する指標である．値が小さいほど収束性と多様性が良いことを示し，0 が最小値となる．IGD は式 (2.7.2) で定式化される．

$$\begin{aligned} IGD(\boldsymbol{P}, \boldsymbol{S}) &= \frac{\left(\sum_{i=1}^{|\boldsymbol{P}|} d_i^2\right)^{1/2}}{|\boldsymbol{P}|} \\ d_i &= \min_{\vec{s} \in \boldsymbol{S}} ||F(\vec{p_i}) - F(\vec{s})|| \end{aligned} \tag{2.7.2}$$

$\boldsymbol{S}$ は得られた非劣解集合，$\boldsymbol{P}$ は理論上のパレートフロントの点集合を示す．また，d_i は $\boldsymbol{P}$ の各点から $\boldsymbol{S}$ の最短距離となる点への距離を示し，$|\boldsymbol{P}|$ は P の点数を示す．