連続パラメータ最適化：　滑降シンプレックス法　(Downhill simplex method)

オリジナルの文献：Nelder, J.A., and Mead, R.: Computer Journal, Vol.7, p.308 (1965).

参考資料：ニューメリカルレシピ・イン・シー（技術評論社）

1　勾配を利用しない山登り法

ニューメリカルレシピ・イン・シー（技術評論社）pp.284 より

「これはごく素直に坂を滑り降り、関数についてはほとんどどんな仮定もしない。収束は非常に遅いが、場合によっては非常に頑健である。しかも、プログラムは簡潔で、全く自己充足的である。N次元の最小化プログラムが１００行以下で書けるのである！この方法は特に最小化が問題全体の中にたまたま含まれた一部分であるときに便利である。必要な記憶量は N^2 のオーダーであるが、導関数の計算は不要である。」

2　アルゴリズム

2.1　定義および基本操作

シンプレックス法における基本操作ではシンプレックス（simplex: 幾何学的な図形）の各頂点を構成する探索点 x_i の他、

1. 目的関数値 $f(x)$ が最大の点 x_h　← 関数最小化なので、最悪の点
2. 目的関数値 $f(x)$ が 2 番目に大きい点 x_s
3. 目的関数値 $f(x)$ が最小の点 x_l　← 関数最小化なので、最良点
4. $i = h$ なる点を除いた全ての x_i の重心 x_g

が必要。これらの探索点が増えた分だけ考慮に入れ、あとは以下に示す通常の操作を繰り返し適用する。

〔操作 1：反射〕重心 x_g を中心に x_h の対称位置にある x_r を生成：

$x_r = (1+\alpha)x_g - \alpha x_h,$　　ただし $\alpha > 0.$　α は反射係数

〔操作 2：拡大〕重心 x_g から x_r 方向に沿って x_r を γ 倍延長して x_e を生成：

$x_e = \gamma x_r + (1-\gamma)x_g,$　　ただし $\gamma > 1.$　γ は拡大係数

〔操作 3：縮小〕重心 x_g から x_h 方向に沿って β 倍の点 x_c を生成：

$x_c = \beta x_h + (1-\beta)x_g,$　　ただし $0 < \beta < 1.$　β は縮小係数

〔操作 4：収縮〕シンプレックス全体を x_l の方向へ半分に縮小する：

$x_i = 0.5(x_l + x_i),$　　ただし $i = 1, \cdots n+1.$　　（備考：点 x_l は移動しない）

$\alpha = 1, \beta = 0.5, \gamma = 2$ が経験的に良いと言われている

2.2 関数最小化のシンプレックス法操作手順

1. 初期シンプレックスを生成し，各探索点での関数評価を行う。
   シンプレックスを構成する探索点の数は，n 次元関数の場合 $n+1$ コ以上[1] とする。
2. 各探索点の評価値 $f(x)$ の大小関係から 最悪点 x_h, 2番目に悪い点 x_s, 最良点 x_l を探し，点 x_h を除いた探索点集合の重心 x_g を求める（x_g の関数評価 $f(x_g)$ は必要ない）。
3. 〔反射操作〕により x_r を求め、関数 $f(x_r)$ の評価を行う。
4. $f(x_r) < f(x_l)$ だった場合は〔拡大操作〕により x_e を求め関数 $f(x_e)$ の評価を行う：
   - $f(x_e) < f(x_r)$ の場合は x_h を x_e に置き換えて手順 2 へ戻る。
   - $f(x_e) \geq f(x_r)$ の場合は x_h を x_r に置き換えて手順 2 へ戻る。
5. $f(x_s) < f(x_r) < f(x_h)$ の場合は x_h を x_r に置き換えて〔縮小操作〕により x_c を求め関数 $f(x_c)$ の評価を行う：
   - $f(x_c) < f(x_h)$ であれば、x_h を x_c に置き換えて手順 2 へ戻る。
   - さもなければ 〔収縮操作〕を行い，新しく生成した全ての点の関数値 $f(x)$ を評価して手順 2 へ戻る。
6. $f(x_h) \leq f(x_r)$ の場合は〔縮小操作〕により x_c を求め関数 $f(x_c)$ の評価を行う：
   - $f(x_c) < f(x_h)$ であれば、x_h を x_c に置き換えて手順 2 へ戻る。
   - さもなければ 〔収縮操作〕を行い，新しく生成した全ての点の関数値 $f(x)$ を評価して手順 2 へ戻る。
7. 上記 4〜6 の条件以外の場合（つまり $f(x_l) \leq f(x_r) \leq f(x_s)$ のとき），x_h を x_r に置き換えて手順 2 へ戻る。

3 適用に際しての注意事項

3.1 シンプレックスの次元の縮退

探索途中において、シンプレックスが何らかの超平面に陥ってしまうと、そこから抜け出すことができなくなり、探索が停滞する（次元の縮退と呼ばれる）。これを回避するために、n 次元関数最適化においては $n+1$ コより多くて多様な探索点を用いる。多数の探索点を用いることは、次元の縮退を回避する効果だけでなく、多峰性関数において局所解から抜け出す効果も得られる。（しかし、シンプレックス端点の個数が多過ぎると収束が遅くなり探索効率が悪化する）

3.2 停止条件

さまざまな方法が考えられるが、文献で紹介したプログラムでは、停止条件を特徴付けるパラメータ δ を導入し、目的関数値 $f(x)$ が最大の点 $f(x_h)$ と目的関数値 $f(x)$ が最小の点 $f(x_l)$ を用いた以下の条件式

$$\delta > \frac{2|f(x_h) - f(x_l)|}{|f(x_h)| + |f(x_l)|}$$

が成り立つとき、収束したと判断して停止するようになっている。

このほか、関数評価を所定回数行ったら強制的に停止させるなどの方法も考えられる。

[1] シンプレックスの個数を多くすると多峰性関数における探索性能が改善されるが，収束は遅くなる。

3.3　パラメータ定義域に制約が存在する場合

このアルゴリズムでは、パラメータの定義域に関係なく、関数値の低い方向へパラメータを調節していってしまうので、シンプレックスが定義域からはみ出る場合には適切に修正操作を行わなければならない。