非線形計画法を使う

八巻　直一

1. プロローグ

突然ですが，寝台のお話しです．寝台は柔らかなものよりも硬いものの方が，健康にはよいそうです．例えば，インドの人々は往来の石畳に寝そべってお昼寝をされるからか，花粉症やアレルギーなどは皆無だそうです．我々も，大いに石の上で寝るべきでしょう．ですが，結構辛そうですし，3年程は頑張らないと効果はないそうです．(石の上にも3年！) 失礼．ともあれ，左様に努力しなければ本物の健康は得られないらしい，という訳ですね．

そこで本題ですが．ORの本来の目的は，経営や社会的政策などにおける意思決定に際して，科学的なアプローチにより最適な方策を探ることであります．間違っても，線形計画法やAIなどなどの手法の倉庫を指すのではありますまい．ありますまいが，科学的アプローチの帰結として，数理モデルが出現し，必然的に数理計画法が登場することになります．困ったことに，数理計画法は極めて数学的な理論体系の中にあり，一部の専門家以外にとって難攻不落のように見えます．その結果，やみくもに何か解らしいものを求める行動にでるか，さもなければ数理計画法などは信用せずに，使わないで逃げましょうということになり勝ちです．前者がいわゆる手法的ORというやつで，後者はOR不信症候群に他なりません．

数理計画法は，本来のORの活動を支える重要な武器たり得るに十分な実力を備えています．しかし，ORワーカが数理計画法を内部まで熟知するには，数学的に過ぎるのも事実です．敢えて挑戦して，数理計画法を自在に使いこなせるようになれば，その向こうには金棒を持つ鬼となった自分が居るでしょう．ですが，よい道具をマスターするのには，石の上にも3年かかります．(?) それに，多分一部の猛烈なワーカ以外は，柔らかい寝台がいいに決まっていますから，挑戦してもすぐに投げ出すに違いありません．我々の願いは，ORワーカの方々には本来の仕事であるところの，『意思決定』の材料作りに専念していただき，数理モデルの構築や数理計画法の操作などのごとき下働きは，専門家を有効に使っていただけるように，ということです．

さて，本稿ではORワーカの皆様に，数理計画法の中でとくに非線形計画法なるものの概要のご説明をいたします．同時に，非線形計画法が使えるかもしれない場面をご紹介しつつ，ソフトウェアや専門家をどう使い回せばよろしいか，下働きの経験者として述べたいと考えました．なお，線形計画法や離散的なモデル，多目的計画法あるいは相補性問題などについては他の記事に詳しいですから，そちらをお読みください．

2. 非線形計画法

ここでは，非線形最適化問題がどんな工夫で解かれるか(すなわち非線形計画法)を，お話しましょう．

さて，非線形最適化問題は，非線形というくらいですから，大層難しそうではありますし，旨くいきそうな気もしません．ですが世の中の森羅万象の内，原因と結果の関係で説明の付きそうな事象を考えれば，多くの場合，

$$\text{結果} = F(\text{原因})$$

と表したときの関数Fは非線形となります．例えば，買物では沢山買うと勉強してくれます．すると，普通ならば

$$\text{支払い金額} = \text{単価} \times \text{数量}$$

という具合いに，支払い金額は数量の一次関数(すなわち線形関数)のところ，数量が相当大きくなると支払い金額があまり増えないような関数となるでしょう．つまりは，支払い金額は数量の非線形関数ということに他なりません．世の中の事象においては，厳密に線形関数で説明出来る事例も有るには有るでしょう

やまき　なおかず　システム計画研究所
〒150 東京都渋谷区桜丘町2-9
Email: yamaki@isp.co.jp

が，そのような事例は極少ないに違いありません．目的関数をいくつかの要因の線形関数として説明し，かつ要因間の関係に生じる様々な制約条件を総て線形不等式で説明すれば，初めて線形計画法(LP)が使えるのですが，どこか一カ所でも非線形関数が有ると，非線形最適化問題となって，たちまちLPは無力となります．

どうでしょう？想像するに，殆どの場合非線形問題になりそうですので，あんなにLPがよく使われるのが不思議なくらいです．線形モデルで十分説明が可能とのお墨付きを得ている一部の問題，例えば『原油を精製していろいろな製品をどのように作れば利益が最大になるか』，など以外では，しばしば非線形最適化問題になります．非線形最適化問題はLPの場合のように，みごとに旨く解ける解法が有りませんので，専門家は『線形近似』や『問題のすり替え』，あるいはLPを利用した手品のような解法などを様々工夫して，尤もらしい結果を出してしまいます．言い替えれば，何かの戦略を持って問題に当たらないと容易には解けない，という訳です．一方，非線形最適化問題をちゃんと解こうとする試みは，1960年代から本格的に研究され始めました．もっとも，非線形方程式の解法で誰でも知っているニュートン法は，かのニュートンの発明だそうですから驚きます．

大多数の最適化問題では，決定すべき変数に制約条件が課せられるでしょう．しかし，もし制約条件がなければ話は割と簡単です．目的関数を最小化する問題ならば，初期点から出発して目的関数を減少させる方向に移動を繰り返すと，最終的に最小点に到達できるでしょう．もっとも，何時でもそんなに話はうまくはいきません．目的関数が凸凹していると，仮に目的関数の谷底に到達しても，そこが必ずしも一番深い谷の底(すなわち最適点)とは限りません．一般の非線形最適化問題では，本質的にこの種の困難さは避けられないのです．ですから，初期点を出来るだけ最適点の近くに設定する努力が重要です．

目的関数の減少方向としてよく用いられるのが，微分を利用した最急降下方向と，2階微分を用いたニュートン方向です．これらの方向は好ましいのですが，目的関数の微分が必要です．しかし，微分が厄介な関数の場合には困ります．そこで，いろいろと工夫されています．微分を用いるかどうかによって数値解法を分類すると，以下のようになります．

微分を用いない方法 たとえば，シンプレックス法(LPのそれとは別もの)が有名です．その他，パターン探索法やランダム法などがよく使われています．微分を用いない方法は，微分しないことの代償としてやや収束が遅いという欠点があります．あなたが使用するソフトウェアがこの手のものであれば，手軽さを享受するかわり遅いことには我慢しましょう．

1階微分を用いる方法 最急降下法が有名です．ですが，これはあまりお勧めしません．遅いのです．そのうえ，収束しない現象がときどき起こります．1階微分を利用する方法では，準ニュートン法が最も優れているでしょう．あなたが使うソフトウェアが準ニュートン法を用いているならば，早さも安定性も期待していいでしょう．

2階微分を用いる方法 ニュートン法がこれです．ニュートン法は，うまくゆくときは速いです．しかし，安定性を保証するための工夫がないと，途中で計算不能になることがあります．

微分をユーザにさせないで自動化する工夫もあります．ひとつは差分で近似する方法で，昔から広く行われています．近年では，数式処理システムが普及してきましたから，数式処理によって微分する方法も提案されています．最近の数式処理ソフトウェアは，単に数式を扱うにとどまらず，アルゴリズムを表せたり図形化したり出来ますので，非線形計画法と数式処理をうまく組み合わせたプログラムが書けるでしょう．しかし，数式処理が非線形計画法のソフトウェアに予め組み入れられているケースを，著者は知りません．

最近の流行は，いわゆる自動微分という技術です．自動微分では，与えられた関数を解釈して，微分の計算をするプログラムを自動的に生成します．数式処理のように数式の形で微分するのではなく，プログラムを生成することによって，非常に効率のよい計算が実現されるのです．最近の非線形計画法のソフトウェアには，大抵自動微分が組み込まれているようです．

さて，多くの問題がそうであるところの，制約条件のある非線形最適化問題ではどうでしょうか．この場合には，制約条件のない場合に比べて飛躍的に難しくなります．以下，代表的な解法を，歴史順にご紹介し

ましょう.

実用的な最初の解法は，SUMT法でしょう．SUMT法は，最近わが国でも物議をかもした『カーマーカー特許』で有名な，線形計画法における『内点法』の元祖といえる方法で，Fiacco と McCormick(1967) によって開発されました．国産大型コンピュータに搭載された，おそらく最古の非線形計画法のパッケージはSUMT法であったと思われます．内点とは，不等式群で記述された制約条件を満たす点で，更にどの不等式についても境界上にはないような点をいいます．要するに，制約条件を満たす領域を池と考えると，池の中であって岸に接してない点のことです．SUMT法は等式で記述された制約も扱いますが，制約条件は全部不等式と思って，以下に原理をお話します．また，目的関数を最小化する問題とします．

SUMT 法の原理

1. ペナルティ関数というものをつくります．これは何かといいますと，目的関数の値が小さくなればそれにつれて関数値が小さくなり，池の中から池の岸に近づくと極端に値が大きくなるような性質をもった関数です．岸に近づいたときの罰金の大きさを決めるパラメータを，ペナルティパラメータといいます．
2. まず，何とかして内点を見つけます．ペナルティパラメータの初期値を設定します．
3. ペナルティ関数の制約条件のない最小点に移動します．
4. ペナルティパラメータを更新し，再びペナルティ関数の最小化を行います．
5. 3と4を繰り返すと，内点から出発すればペナルティ関数の性質から，制約条件から飛び出ることなく，最適点に到達することができます．

SUMT 法は当時としては画期的でしたが，収束の遅いことが難点でした．しかし，LP の内点法でその理論が再び脚光を浴びることになろうとは，誰が予測したでしょう．

さて解法は進歩し，乗数法といわれる解法が発展しました．等式で表された制約条件のもとで，目的関数の極値を求める問題に対して，ラグランジュ関数といわれる関数を構成すれば，その極値を求めることで解を得ることができるという事実は，古くから知られています．乗数法 (なかんずく拡張ラグランジュ乗数法) はこの原理を拡張して，不等式で表された制約条件も考慮できるように工夫された方法です．以下に原理をお話します．

拡張ラグランジュ乗数法の原理

1. 拡張ラグランジュ関数というものをつくります．これは何かといいますと，目的関数に制約条件を表す関数の乗数倍を加え，更に拡張項といわれる，制約条件を表す関数の2乗和から構成される項を加えた関数です．制約条件を表す関数にかけられる乗数をラグランジュ乗数とよびます．
2. まず，変数と，ラグランジュ乗数の適当な初期値を設定します．
3. 拡張ラグランジュ関数の制約のない最小点に移動します．
4. ラグランジュ乗数を更新し，再び拡張ラグランジュ関数の最小化を行います．
5. 3と4を繰り返すと，最適点に到達することができます．

1980 年代の前半に作られた非線形計画法のソフトウェアならば，概ね拡張ラグランジュ乗数法が組み込まれているでしょう．

その後更に研究が進み，1980 年代後半から 1990 年代になると，逐次二次計画法 (SQP 法) なる方法が普及しました．二次計画法というのは，目的関数が二次関数ですべての制約条件が線形であるような問題，すなわち二次計画問題の解法を指します．二次計画法は線形計画法に次いで普及しています．二次計画法には様々な解法があり，ソフトウェアも多数存在します．ここでは二次計画法には触れないでおきますが，二次計画問題はかなり易しく解けるとだけ認識しておいて下さい．非線形最適化問題においては，幾つかの仮定 (目的関数が凸凹でないとか，微分がどうのこうのとか) があれば，Karush-Kuhn-Tucker 条件 (KKT 条件) と呼ばれる条件を満たす点が最適点となります．ところで，ある変数値とラグランジュ乗数値に対して，目的関数と制約条件とその微分から二次計画問題をうまいこと構成すると，その二次計画問題の解が **0** ベクトルであるとき，この変数値とラグランジュ乗数値

がKKT条件を満たす，という誠に都合の良い事実があります．また，構成された二次計画問題の解が0ベクトルでないときは，得られたベクトルの指し示す方向に移動すれば，最適点に接近することが出来るのです．SQP法は，このような好ましい性質を持つ二次計画問題を，逐次解くことによって最適点に到達するように設計された解法です．

SQP法の原理

1. ラグランジュ関数というものをつくります．これは何かといいますと，目的関数に制約条件を表す関数の乗数倍を加えた関数です．制約条件を表す関数にかけられる乗数をラグランジュ乗数とよびます．

2. まず，変数と，ラグランジュ乗数の適当な初期値を設定します．

3. その点の値とラグランジュ乗数の値を用いて，二次計画問題を作り適当な二次計画法を用いて解きます．

4. 二次計画問題の解が0ベクトルならば停止します．さもなければ，解ベクトルの方向に移動します．

5. 3と4を繰り返すと，最適点に到達することができます．

SQP法は，現在のところ最も優れた解法といえます．したがって，最近のソフトウェアではSQP法が主力となっていると思われます．しかし，SQP法にも欠点があります．その中で一番困った問題は，もとの問題が解を持つ場合でも，途中に現れる二次計画問題が解を持たないことが起こることです．ソフトウェアのなかには，そのような事態に対処しているものもありますが，みなさんがお使いのソフトウェアでそのような事態が発生したら，初期値を変えて見られるのが良いでしょう．

今のところ，定番となって認知された解法の代表はこんなものですが，研究は進歩し続けていますので，もっとよい解法が出現する日もそう遠くないに相違ありません．例えば，線形計画法で非常な成功を納めた主双対内点法などを，非線形問題に適用する試みなどは有力でしょう．

3. 非線形計画法のソフトウェア

さて，ここでは非線形計画法のソフトウェアのお話しをしましょう．まずは，問題の規模からお話します．非線形計画法では，LPのように数十万変数，数十万制約などという途方もない問題は扱う事が出来ません．大規模問題といっても，せいぜい数千変数といったところです．数千変数といえども，扱う事の出来るソフトウェアはほとんど無いでしょうし，第一に問題を定義してソフトウェアに与えることも大仕事です．何故ならば，線形最適化問題の場合は関数はすべて線形ですから，関数は係数を与えるだけで定義できます．従って，データとしては数値のみを用意すれば足りるのです．一方，非線形最適化問題では目的関数や制約条件は非線形関数ですから，関数を定義するのに，数値だけでなく$\sin, \log, x^2$ ，などなどといった形も必要です．このように，非線形計画法のソフトウェアでは，解法そのものの限界もさることながら問題の入力という壁が存在します．そんな訳で，問題の規模について極めて制約が強いというのが実態です．

非線形計画法のソフトウェアを分類するとき，用意されている解法で分けるよりも，構造あるいは入手方法で分ける方がよいと思われます．なんとなれば，解法にはそんなにバラエティーはなく，チューニングによるそれぞれの自慢は有りましょうが，性能に決定的な差異は見られないと考えて差し支えないでしょう．(この辺のところは制作者の方々には叱られるかな？)

構造によって分ければ，以下のようになります．

問題リンク型 ユーザが，目的関数や制約条件を副プログラムとしてコーディングし，メインである解法プログラムとリンクして実行するタイプです．古い時代のソフトウェアの多くは，このタイプです．このタイプを用いるユーザは，指定された言語を知っていなければなりませんが，一方でプログラムが可能ならばどんな複雑な問題にも対応できます．

ライブラリ型 問題リンク型と同様に，ユーザが目的関数や制約条件の副プログラムを作りますが，このタイプでは解法プログラム自身がライブラリ形式になっています．したがって，全体のプログラムはユーザに任されています．ライブラリ型は，ユーザのシステムの中に最適化機能が埋めこまれるような場合に便利でしょう．

プログラム生成型 ユーザが問題やパラメータを設定すると，システムがそれを読んで最適化プログラムのソースコードを生成します．このタイプは，いわゆるプリプロセッサ方式であり，問題記述言語(モデリング言語)を持っているのが特徴でしょう．とくに大規模問題では，前述のようにモデリング言語が必須となりますので，これからはこのタイプのソフトウェアが多くなると思われます．

入手方法で分類すると，以下のようになります．

パッケージ型 いわゆる市販ソフトウェアです．ただし，非線形計画法のパッケージは，日本では驚くほど少数しか市販されていません．そりゃそうでしょう．だって，あんまり売れるとは思えませんから．パソコンショップで問い合わせても，まず情報は得られないでしょう．こんな時にこそ，専門家に聞きましょう．

メーカラインアップ型 コンピュータメーカは，ソフトウェアラインアップと称して，自社ソフトウェアの他にサードパーティーのソフトウェアを登録しています．この中に非線形計画法が含まれている例があります．あなたのコンピュータのメーカに問い合わせてみましょう．

本のおまけ型 非線形計画法を扱った一部の書物には，無償または有償でソフトウェアがついているものがあります．ただし，本の中に印刷されたソースコードを打ち込んで使うのは，信頼性の面でお薦めできません．この種のものは，本屋さんで聞けば分かります．

無償型 無償ソフトウェアには，ただ同然の使用料を払うもの，使用許諾だけが必要なもの，あるいは全くフリーのものがあります．研究目的ならば，ネットワーク上に流通しているものには，無償型の優れたものがあります．しかし，実用の場面で無償ソフトウェアを用いるには，いろいろと考慮する事項があるでしょう．もし，顧客に引き渡すものならば，著作権の問題をすっきりしなければなりません．改造が自由かどうかもチェックする必要があります．

あなたに適したソフトウェアは，どんなタイプでしょうか？上の分類をご参考になさって，それこそ最適な選択をして下さい．そうそう，自作という道もあります．腕に自信のある方は挑戦しても良いかもしれません．

非線形計画法のソフトウェアの入手を計画したならば，次の質問を用意して然るべき筋に問い合わせることをお薦めします．

- どんな解法が用意されていますか？
- どんな構造ですか？
- どの位の規模の問題まで対応していますか？
- 自動微分 (正式には高速計算微分) 機能はありますか？

どんな環境で動くか？とか，価格は？とか，その他通常のソフトウェアの購入の際の注意事項も，もちろんお忘れなく．

4. 非線形最適化問題の事例

この辺で，非線形最適化問題の事例を幾つかご紹介しましょう．

まず，今野先生の記事に出て来る，マーコビッツモデルといわれる投資選択モデルは，実は前節にちょっと出て来ました二次計画問題です．二次計画問題は，非線形最適化問題の仲間ではありますが，特別な解法が発達していて，一般の非線形最適化問題に比べると格段に取り扱い易い問題なのです．

工学的なお話しでは，現象を測定して原因を推定する問題(いわゆる逆問題)は，多くの場合非線形最適化問題となります．逆問題では，原因となるパラメータを変数とし，パラメータの関数として現象の理論値を定義します．このとき目的関数を，理論値と測定値の差の二乗とすると，パラメータの非線形関数を最小化する問題となります．工学的な諸々の条件は制約条件として表現されますが，しばしば非線形関数となります．このように，逆問題は非線形最適化問題として定義されることになります．

工学的な問題は，OR ではあまり扱いませんので，変わった分野の事例をご紹介しましょう．『確率的計画法』(南石晃明：現代数学社，1995)を引用させていただきますと，農業の分野を中心としたいろんな例が発見できます．

1. 野菜の出荷計画

野菜を出荷する計画を立てるのも，大変に数理的であるようです．ここではある地方から，関東，

近畿，および九州地域へ，どのような出荷配分にするかを計画します．

目的関数は，収益の期待値を収益の分散で割ったものです．これは非線形関数なので，非線形最適化問題となります．制約条件は，それぞれの地域への最低出荷量を確保することですので，それぞれの地域への出荷量の線形制約となります．

2. 人気品種コシヒカリの栽培計画

コシヒカリを栽培するのも，極めて数理的なのです．ある農家では4月に植えて9月に収穫するものから，約1週間毎に田植えをずらして，5月田植えの9月末収穫の幾つかの製品系列があります．収穫が遅くなると収量が減ります．作業可能時間は天候に影響されます．天気の悪い日は，稲が湿っているので作業が出来ません．また，耕作面積が広いと作業時間はかかりますが，狭いと収量が少なくなります．この農家では借地をしていますので，借地面積を諸々のリスクのもとに収益を最大とするように設定したい．

さて，どのような計画を立てるべきか？

南石博士の著書では，巧妙なしかけで不確実性のある問題を非線形最適化問題に表し，かつ更に巧妙なトリックで，場合によってはLPで対処できるように工夫されています．そのようなトリックはさておき，ここに例示したような野菜やコシヒカリの栽培の局面にも，非線形最適化問題が現れることの面白さはどうでしょう？

さて，大規模問題の例を少しご紹介しましょう．工学的問題に現れる大規模問題の典型的事例は，最適設計でしょう．最適設計とは，例えば建造物を設計する際に，必要な強度を確保した上で柱の太さを最小にする問題などを指します．数多くの柱の太さを変数としたとき，強度の計算は変数の非線形関数となります．したがって，非線形制約条件をともなう最適化問題となります．構造計算では，しばしば感度といわれるものが問題にされます．感度とは，例えばどこかの材料の強度が全体の強度に及ぼす影響，といったものです．構造計算のソフトウェアの水準の高いものには，感度解析機能がついています．感度をうまく使うと，最適設計のための非線形最適化問題を解くプログラムが作れるでしょう．これからは，最適設計の機能を組み込んだソフトウェアが多くなるに違いありません．ついでにいえば，感度を計算するには複雑なモデル式を微分する必要がありますが，自動微分を用いれば快適に感度解析プログラムを生成することが出来ます．大規模問題では，自動微分は必須の道具といえましょう．

石油製品の生産計画は通常はLPの問題ですが，脱硫に関する制約条件に非線形関数が現れることがあります．すると，大規模なLPの中にちょっとだけ非線形関数の混じった問題となります．でも，れっきとした非線形最適化問題ですので，最早LPは使えません．

数式を使わず，かつ数学的表現を用いずに，具体的な問題を説明するのは大変難しいものです．この節のご説明は，ちょっと具体性に欠けるように思いますが，お許し下さい．しかし，みなさんが現場でどんな問題に直面しているかの一端は，少しばかり感じられたのではないでしょうか？

5. エピローグ

さて，非線形計画法のお話しは，これでおしまいです．筆者のささやかな経験では，『これはどうやら非線形だぞ』と，殺気を感じられた実務家からよく質問を受けます．その多くは，問題の定義をよく考え直さないと解けそうもないモデルでありました．したがって，モデリングが非常に重要な作業であると考えます．ソフトウェアを入手しただけでは，この点は何も解決しません．

非線形計画法のソフトウェアを使用するときの最大の難点は，モデリングについてソフトウェアは何もサポートしないことでありましょう．モデリングに対するサポート機能の重要性については，中山先生も指摘されています．しかし，現在のところどんなユーザにも有用なサポート機能を作るのは，ほとんど絶望的であるような気がします．

モデリング技術とそのサポートソフトウェアについては，今後大きな課題であると考えられます．という訳で，今のソフトウェアにはモデリングについてサポートは期待薄です．ですので，専門家を利用するならば，殺気を感じたらモデリング段階から利用するのが良い方法だと思います．

えっ？私に早速ご質問ですって？

では，よい人をご紹介しますので，耳を貸して下さい．ムニャムニャ．