はじめに

「**問題を解決する能力**」は，そうたやすく身に付くものではない．しかし，ある種のタイプのものは，私が「**魔法の学問**」といっているものを理解するだけで誰でも身につけ実践することができる．なぜ「魔法の学問」なのか？なぜ簡単に実践できるのか?それを説明したい．

１）　理論が簡単である

皆さんは，例えば y=f(x)=3x+1 という 1 次式は，数学が得意でなくてもそれほど拒否反応はないでしょう．ここで，x が 0 から 2 の間にあるとします．このとき，高校数学で x の定義域は区間[0，2]（0≦x≦2）にあるという．このとき，y の最小値は x=0 で y=f(0)=3*0+1=1 になり，最大値は x=2 で y=f(2)=3*2+1=7 になる．そして，値域は[1，7]といいます．実は，これがこの学問の理論的背景です．すなわち，定義域という制約条件つきの関数の最大値と最小値を求めるだけです．あるいは，与えられた定義域から値域を求めることになります．

例えば，PC の量販店を考えてみよう．「メーカーから，月額 1 万円の報償金と，1 台に付き 3 万円の利益があるとする．在庫は残念ながら 2 台しかない．このとき，最大の利益はいくらか?この問題を解決したいとき，PC の在庫数を変数 x で表して，問題を数式で表せば上の数学の問題になる．すなわち，解決したい問題が数式で表すことさえできれば，その問題は「**数理計画法ソフト**」でたちどころに解決できるわけです．幾ら難しい理論を理解しても，必ずしも実生活で役に立ちません．しかし，数理計画法で扱える問題は，読者の想像を超え幅広く多彩です．これまで，それを解くソフトが未熟であったため，注目されてきませんでした．この分野でも，ようやく誰でも，簡単かつ廉価で強力な数理計画法ソフト LINGO(リンゴー)が利用可能になりました．

２）　領域の最大最小問題

皆さん，微積分といえばとっつきにくいでしょう．関数の最大最小問題は，一般には微分で求めることができます．しかし，関数が複雑であれば，とたんに落ちこぼれてしまいます．私もそうです．昭和 45 年の 9 月に学卒の採用試験が終わっていたのですが，工学部の大学院生と同じ微積分の採用試験問題で NEC を受けました．演習で微積分の解を求める訓練をしている工学部の院生の問題を解けるわけがありません．白紙なのに，なぜか合格通知をもらいましたが．

話は変わりますが，高校数学Ⅱに「領域の最大最小問題」というテーマがありますが，覚えていますか．例えば次のような問題です．

問）連立不等式 x≦5，x+ y≦10，x +2y≦16，x≧0，y≧0 の表す領域 D を図示し，点（x，y）がこの領域を動くとき 2x+3y の最大値と最小値を求めよ．

これは，点（x，y）の動ける領域を図示して，その中から z=2x+3y を最大・最小にする点（x，y）の値と z の値を求める問題である．数学の中では，絵に描いて理解できるので，分かりやすいテーマの一つです．もし，苦手意識があるのなら，「こんな問題が何の役に立つのか?」と疑問に思ったり，担当の数学の先生が嫌いであったりという，些細な原因からでないかと思います．これが，数理計画法の中核である「**線形計画法の理論の全て**」です．どうです，微積分よりずっと易しいでしょう?

1 章で紹介するように，変数 X と Y を廉価 PC（格安バーガー）と高級 PC(高級バーガー)の製造個数（販売個数）と考えるだけで，部品を組み立てる産業の同種の全ての問題が解決できる．すなわち，変数を読み替えるだけで，組み立て産業に共通で使えます．

３）　数理計画法ソフトの利用の簡便さ

上記の問題は，絵に描いて解くことができます．しかし，変数が 10 個も出てこれば，絵で表すことはできません．その場合，数理計画法ソフトを使いますが，使い方はいたって簡単です．どのような領域で最大(最小)にしたいかを次のように記述するだけです．

MAX=2x+3y

x≦5

x+ y≦10

x +2y≦16

x≧0

y≧0

どうです，簡単でしょう．私はこれまで統計ソフトを使って皆さんのデータ分析能力をかさ上げすることの普及に努力してきました．統計手法は，人類の英知が様々な手法を開発してきました．そのため，統計ソフトを用いた啓蒙書を書く場合，統計ソフトの操作法がかなりの部分を占め，統計理論の紹介とのバランスに悩みました．しかし，数理計画法では，ソフトの操作法の解説はほぼ不要です．上のテキスト形式の問題（モデルということにします）を作成し，実行するだけです．また分析結果も，最大値とそのときの X と Y の数値だけです．ですから，どんな難しい理論であっても，数理計画法のモデルと結果の解釈は誰でもできます．これも「魔法の学問」の大きな利点の一つです．本書に添付しているモデルのファイルをダブルクリックし，実行すれば，答えが出ます．皆さんが注力すべきは，その数値を現実にどう解釈すべきかを本書で理解することだけです．

4)　読者のすべきこと

こんな簡単な記述でなぜ複雑な現実のさまざまな問題解決に役立つのでしょうか?

それは，多くの天才が驚くほど柔軟な発想で現実問題を表す数式モデルを開発してきたからです．これらの問題を最初にモデル化することは才能がいります．しかし，一旦作られたモデルは，上の式の意味さえ分かれば誰でも容易に理解できます．そして，出てきた解の意味が正しく分かり，問題解決に当てはめればよいのです．すなわち，びっくりするような多彩なモデルがすでに開発され，すぐに利用できる点も「数理計画法が魔法の学問」と主張する点の一つです．

5) 解決できる問題のリスト

数理計画法で解決できる問題の多彩さに,読者はきっとびっくりするでしょう.何しろ,式に書ければ，その最大値や最小値あるいは解を求める問題が全て「魔法の学問」で解決できる対象になります．本書の 2 章以降では，その中から代表的なものを選びました．例えば，7 章のポートフォリオ分析はノーベル経済学賞をもらいました．普通そのような理論は難しくて理解が困難です.もうお分かりでしょうが,これが数理計画法で扱えるので,何が制約式で何が目的関数かさえ押えておけば，理解は容易です．

6) なぜ日本では「魔法の学問」でなかったのか?

さて，ここまで読んでこられてきっと疑問を持つ方もいるでしょう.「そのようなすばらしい魔法の学問がなぜ日本でそれほど普及していないのか?」一つには，高校数学で「領域の最大最小問題」を抽象的に教え，問題解決学の入門という位置づけがなかったこと．次に，大学教育の問題が大きいと考えます．多分 40 歳以上の工学部，経済学部，農学部などの出身の方は「数理計画法」を必修科目として習ってきたと思います．そして，多くの方がもう 2 度とお目にかかりたくないと思っているはずです．なぜなら，線形計画法の計算方法を教えることが授業の中心でした．実は，そのような教育はほんの一握りの数理計画法の研究家に任せればよいのです．多くの人は実際の問題を解決する能力を養う「21 世紀の一般教育として多彩なモデルを勉強すべきだ」と思っています．ここまでは，私の 10 年来の主張です．

私は 10 年以上前から,現実のモデルを中心にすえた新しい視点の入門書の原稿を 3 冊も書き溜めてきましたが，読者から受け入れられないのではないかという懸念があり，出版にこぎつけませんでした．

その明快な理由をやっと 2007 年 12 月末になって，明確に認識しました．

その理由は，皆さんが手に入れた問題解決能力を現実の大きな問題に適用しようとしたとき，これまでの数理計画法ソフトではモデルの作成に時間がかかりめげてしまうことです.実践できない実用の学問の習得ほどむなしいものはありません.しかし,数年前から,

大きな問題であってもモデルのサイズに無関係に「**モデルを汎用的にする**」事が容易になった．これで入門書でありながら，本書で取り上げた問題は，現実のどんな大きな問題でもすぐに対応できます．ですから，私は本書をこれまでと異なり自信を持って上梓することができました．

6）本書の画期的な点

本書は，心から問題解決能力を身に付けたいという読者を対象にしています．必要な知識は，1 章で紹介する１次や２次の関数の最大と最小の意味を理解すること，そして「領域の最大・最小問題」が理解できることの２点だけです．

2 章以降の各章は，重要な問題に絞って，最初に問題を我々が普段目にする自然な数式表記で記述しています．その後，「**汎用モデル**」を説明します．万が一この説明が完全に分からなくても，どのようなデータを Excel 上に作成すれば現実の問題を解くことができるかを最低限理解してください.それによって,もし役立つモデルであることが分かれば，頓智の塊である汎用モデルをクイズのように楽しく玩味できるでしょう．

8）　産業界へのお願い

日本の現場は「感，コツ，経験」に優れているため，現在の各種の計画はおそらく最適解の 90%以上は達成していると思います．これも全ての産業の隅々で数理計画法が使われていない理由でしょう．あと数%の改善が「魔法の学問」で簡単に行えるのにもったいない話です．原材料高の今こそ，考え直しませんか？

本書の主なみどころは，次のとおりである．

1 章：数理計画法の重要なポイントを分かりやすく説明します．

2 章：非線形連立不等式の解を求める

数理計画法は，現象が式で表されさえすれば，その解を求めることができる汎用の数学ソフトです．単に数式，連立方程式，非線形連立不等式，などの解を求めるのにも利用できることを紹介します．

3 章：プロダクト・ミックス(製品組み立て)問題

製品組み立て問題は，電気，自動車，ファースト・フードなどの部品を組み立て，最終製品を作る産業をモデルにしています．そして,「領域の最大問題」そのものです．ここでは，汎用モデルを作成するテクニックを紹介します．

４章：**配合問題**

配合問題は,材料を混合し最終製品を作る産業の基本モデルです.石油産業や化学産業，製鉄，酪農や養鶏業の配合飼料，金融商品の組み合わせなどに広く応用されている．このためモデルのサイズも大きくなるが，汎用モデルを使えば誰でも実践できる．また，これ

まで利用されてこなかった食材加工工場，病院食，原材料高に悩む中小企業の材料決定などは，規模が小さいモデルが多いと考えられるので，デモ版で十分分析できるでしょう．

５章：巡回セールスマン問題

巡回セールスマン(TSP)問題は，与えられた都市を一筆書きで，最短距離(最小時間でもかまわない)でまわる問題である．遊びの要素を備えているが，実は産業界で非常に応用範囲の広いものです．作業台と部品の保管場所の行き来を最小化する，製造ラインの取替え，各種回路の設計で回路を接続する距離の最短化，私は経済学部の同僚と東京近郊の不動産価格の補正のための研究に用いています．

6章：　ポートフォリオ分析

ノーベル経済学賞を取ったポートフォリオ分析の理論を，3 社の株式で説明し，その後金融機関でも使える汎用モデルを紹介する．

7章：　日程管理（PERT）

日程管理は，ブルーバックスでもロングセラーの「**計画の科**学」で紹介されている．しかし，インターネットでは「**ガント・チャート**」と呼ばれる低レベルで問題の多い工程管理ソフトが花盛りである．なぜ汎用の PERT プログラムが紹介されていないのか不思議である．PERT の汎用プログラムは，プロジェクトを幾つかの作業工程に分け，各作業工程で，先行する作業と後続する作業のペアのリストを Excel 上に与えるだけで計算できるので，ガント・チャート利用者にとって朗報であろう．学生さんは，学園祭の準備などに利用してみよう．

８章 判別分析のニューフェース（SVM）

回帰分析，判別分析，コンジョイント分析などの統計手法の汎用モデルがすでにありますが，本章では最近注目されている SVM（サポートベクターマシン）の汎用モデルを紹介する．

９章：評価の科学（DEA）

経営効率分析法または包絡分析法（Data Envelopment Analysis，DEA）は，企業の事業部，百貨店などの複数の店舗，自治体の複数の図書館，野球選手の評価などを，多入力多出力のデータを用いて行う手法である．私も始めて DEA の汎用モデルでデータを分析し，2008 年 2 月 13 日と 14 日の 2 日間で本章を書き，2 月 18 日に成蹊大学で開催された DEA の国際会議に提出できました．

付録：整数計画法のアルゴリズム

整数計画法のアルゴリズムである分枝限定法は，クイズを解くような感じで楽しめる方法です．

目次

1.　魔法の学問

1.1　魔法の学問を公開します

理論が絵でもって理解でき，利用に際して覚えることが少なく，それでいて広範な問題の意思決定や問題解決に使える技術（学問）をご存知ですか．私一人が，10 年以上前から「魔法の学問」といっているものです．

今まで出し惜しみしていたわけでなく，それを主張し皆さんに納得してもらえるものが一つだけ欠けていました．魔法を実現するソフトウエアが，ようやく期待にこたえてくれるようになったわけです．

この「魔法の学問」を本書で理解すれば，皆さんは広い視野をもつことができます．そして，Excel 上に必要なデータさえ準備すれば，それを「汎用モデル」で簡単に分析し解決できます．

(1)　なぜ広範な問題が解決できるのか

私が「魔法の学問」といっているのは，「数理計画法」という学問です．「何だ」といって，ここで読むのをやめないでください．この学問は，制約条件のある関数の最大値や最小値を求める学問です．ですから，問題解決したいものが，数式で定義さえできれば，それらが全て数理計画法の対象になります．（目的）関数 y= f（ｘ）=2ｘ+１が，定義域（制約条件）[2,3]で考えた場合，ｘ=2 で最小値 y=5，ｘ=3 で最大値 y=７をとるという単純このうえないことが数理計画法の対象です．

ですから，解決したい問題が関数で記述さえできれば，全てが数理計画法ソフトでやっと解決できるようになりました．これまで，数理計画法ソフトの機能が未熟で「どんな問題でも解決できますよ」と自信を持っていえなかったのです．

しかし，「関数で記述できれば解決できます」といわれても，皆さん困ってしまうでしょう．しかし心配は要りません．これまで，人類の天才，英才，秀才，そして鈍才（私のことです）が実にさまざまな問題を，すでに数理計画法の雛形モデルとして開発済みです．まずそれらを理解し，利用するだけでも十分成果が上がります．

(2)　なぜ理解が容易か?

これらの問題を自分で再発見することは,よほどの天分に恵まれなければ困難でしょう．しかし，開発済みのモデルを理解することは容易です．なぜなら，定義域を表す制約条件と売り上げや利益や視聴率などの最大化したい目的関数を理解するだけです．また，分析結果の理解も簡単です．何しろ，どの値で最大あるいは最小値をとるかを理解するだけですから．

私はこれまで，統計ソフトを用いた実践的な教育の普及に注力してきました．しかし，統計は種々の統計手法が開発されていて，理解すべき統計量も実にさまざまで，出力結果も多彩です．また，「帰無仮説」，「母集団と標本」といった日常生活と異なった思考の飛躍が要求されます．

それに比べて，数理計画法は扱える問題は多岐にわたっていますが，モデルの表記法と解析結果の入出力の形式は一通りで，いたって簡単です．ではなぜ統計ほど皆さんになじみがなかったのでしょうか?

(3)　これまで数理計画法が広く受け入れられなかった理由

統計に比べなぜ数理計画法は，広く皆さんに受け入れられなかったのでしょうか?私は10年ほど前から，数理計画法の解説書を3冊ほど書き溜めてきましたが，ある理由で出版にまでいたりませんでした．しかし，2007年の12月末にようやくその理由が啓示的に分かりました．

実は，数理計画法を勉強しても，それを実社会で応用しようとした場合，これまで数理計画法ソフトの機能が弱かったため，適用が大変だったためです．ですから，数理計画法の実践は一部の専門家にとどまっていました．

しかし，数年前からようやく数理計画法ソフトの機能が格段に向上し，多くの人が利用できることを10年間判別分析の研究に没頭してきた私は見逃していました．

本書では，代表的な8種類の問題解決法を2章から9章で紹介します．重要なのは，それらを現実の問題にすぐに適用できる「汎用モデル」を提示していることです．

例えば，ノーベル経済学賞を取った「ポートフォリオ分析」を6章で紹介しています．ノーベル経済学賞といっても，数理計画法のモデルですので制約式と目的関数だけになり他のモデルと同じレベルで理解できます．

最初，3社の株式で読者は理解し，その後「汎用モデル」で金融機関が行っている規模の分析もできます．

私自身は，2007年度まで成蹊大学の2年次配当の半期科目で数理計画法の種々のモデルを紹介してきました．しかし，多くの学生は，多分社会に出てもそれらを実践することはすくないとあきらめていました，多少むなしく悲しい思いで教えていたわけです．

2008年度からの受講生は幸せです．「汎用モデル」を教えますので，多くの学生が社会で役立てることが期待できるからです．

(4)　高校数学で分る魔法の学問

数理計画法の理解には，1.2で紹介する「関数の最大値と最小値」を理解することが基本です．

このほか，数理計画法には大きく分けて次の 4 つの手法があり，そこでは次の点を理解することが重要だ.

・ 線形計画法（Linear Programming, LP）

制約条件と目的関数が 1 次式で表されるものを線形計画法（LP）という．数理計画法の入門で，応用範囲も広い．LP の理論的背景は，高校数学Ⅱの「領域の最大最小問題」が分かればそれで終わりです．ただ，3 章で説明する「減少費用」と「双対価格」という有用な情報も使いこなせればそれで十分です.

・ 2 次計画法（Quadratic Programming, QP）

制約条件が 1 次式で，目的関数が 2 次式で表されるものを 2 次計画法（QP）という.

LP と QP はそれほど難しくありません.しかし次の 2 つがこれまで数理計画法ソフトの鬼門であったわけです.

・ 整数計画法（Integer Programming, IP）

LP や QP の変数は実数ですが，整数計画法（IP）では変数が 0/1 の 2 値の整数値，あるいは非負の整数値（x>=0）に制限されたものをいいます．大規模な整数計画法モデルは，組み合わせの爆発のため計算時間がかかります．今後とも改良が続けられますが，ようやく多くの分野に適用できるように成りました.

・ 非線形計画法（Non Linear Programming, NLP）

制約条件と目的関数が 1 次式で表されないものを非線形計画法（NLP）といいます．これは 2 章で紹介しますが,「最大値/最小値」のほか,「局所最適解と大域的最適解（いわゆる最大値/最小値)」を理解する必要があります.これまで数理計画法のソフトは,大規模で,複雑な非線形の「局所最適解」を求めることが困難でした．また得られた解が，大域的最適解かどうかの判定も困難でした．しかし，漸くこれらの難題も解決されました.

1.2 関数の最大と最小値

数理計画法の理解は，ここで紹介することの理解が重要だ.

（1）定義域と値域

最初に 1 次式 y=ax+b の最大値・最小値問題を考えてみよう．例えば，y=2x+1 を考える．x の動く範囲（定義域あるいは数理計画法では実行可能領域という）を実数全体（－∞<x<∞）とすれば，y の取る範囲（値域という）も実数全体（ - ∞<y<∞）になる.

しかし，現実問題の多くは資源の制約，資金制約，労働制約などの各種の制約の範囲内で，利益や売り上げや視聴率の最大化や，材料費の最小化を計りたいわけである.

（2）y=2x+1 の最大値と最小値

1 次式（y=2x+1）をグラフで表わすと直線になる．例えば定義域を 1≦x≦3 にすると，**図 1･1** のような線分になる．この時，関数 y=f(x)の値は，3 から 7 の間にある．すなわち，最大値は x=3 で y=7 になる．最小値は x=1 で y=３になる．すなわち，定義域が有限で，目的関数が（2x+1）のように x の線形の式で表わされる場合，最大と最小が必ず定義域の端に現れる．

この本のテーマである数理計画法では，同じ用語を次のように読み替えている．x は変数あるいは決定変数(decision variable)，定義域のことを「実行可能領域」，そして関数 y のことを「目的関数」といっている．

そして，数理計画法の教科書では一般に次のように記述する．制約式は全て AND 条件であり，制約式の表す共通集合が実行可能領域になる．

MAX=2x+1

X≧1

X≦3

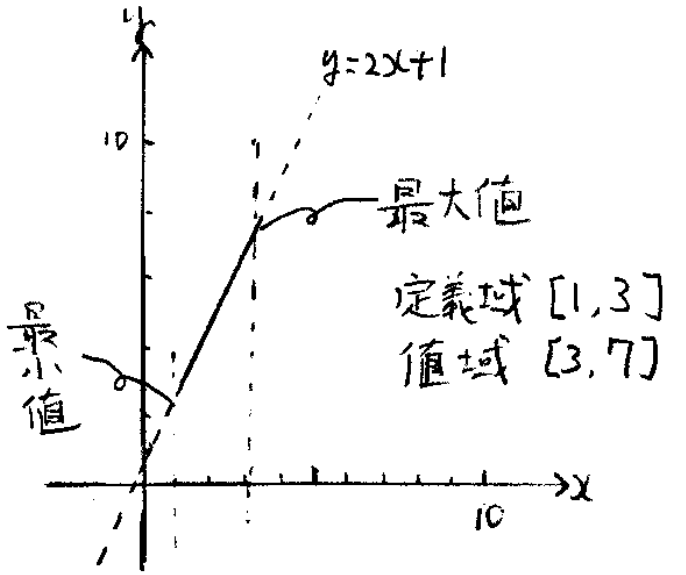

図 1･1　最大と最小は端に表われる

本書で紹介する「魔法の学問」の決めてである LINGO では，加減乗除とべき乗は「＋，-，*，/，^」を用い，目的関数や制約式の終わりは「；」で区切り，不等号の「≦」は「<（または<=)」で「≧」は「>（または>=)」で表すことにする．LINGO の入力モデルは，次のようになる．

```
MAX=2*x+1;
X>1 ;
```

X<3 ：

以上のモデルを**図 1.2** の LINGO の「モデル作成画面」に入力する．そして，メニューから[LINGO]→[SOLVE]を選ぶか，アイコン  をクリックするだけで，**図 1.3** の LINGO の「Solution Report 画面」と「Solver Status 画面」が表示される．すなわち，「モデル作成画面」でモデルを作成する．ただし本書では全てのモデルをファイルにしてあるのでそれをダブルクリックすれば，モデル画面が表示される．そして，実行ボタンを押せば，解を表示する画面と，解の状態を示す画面が現れる．いたって簡単で，読者は拍子抜けするであろう．多分，Windows のソフトの中でも利用法が最も簡単なソフトの一つであろう．そこで本書では，LINGO に関する記述は最小限にとどめ，問題解決学を集中して学ぶことにしたい．

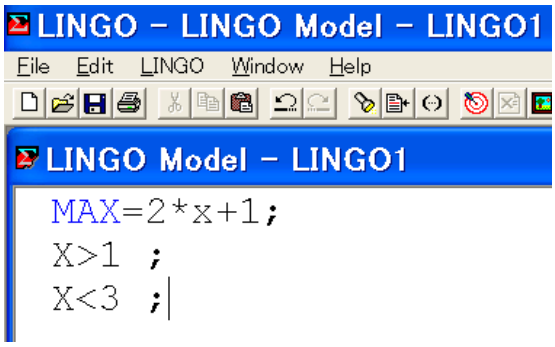

図 1.2　LINGO のモデル作成画面

さて解の画面に，X の値が 3 で目的関数（Objective value）の値が 7 と表示されている．この他，X の減少費用（Reduced Cost）と，目的関数（Row の 1）と制約式(Row の 2 は X≧1，Row の 3 は X≦3)のスラックあるいはサープラス（Slack or Surplus)と双対価格(Dual Price)が出力されている．これが単に関数の最大値や最小値を求めるだけでなく，数理計画法の重要な情報である．これは，3 章で説明する．Solver Status 画面では，解の状態や計算時間を示す情報が表示されている．

（3）2 次式

関数 $y=f(x)=ax^2+bx+c$ のように，x を入力すると x^2 が最高次の項として出力されるものを 2 次式という．2 次式は，物理学では放物線を表わす数式モデルになっている．

例えば，a =1，b=-2，c=0 とした次の 2 次式を考えてみよう．

$$y=x^2-2x$$

これを次のように変形すれば，**図 1･4** のような x＝1 で y＝－1 が最小値になる下に凸の放物線になる．

$y=x(x-2)=(x-1)^2-1$

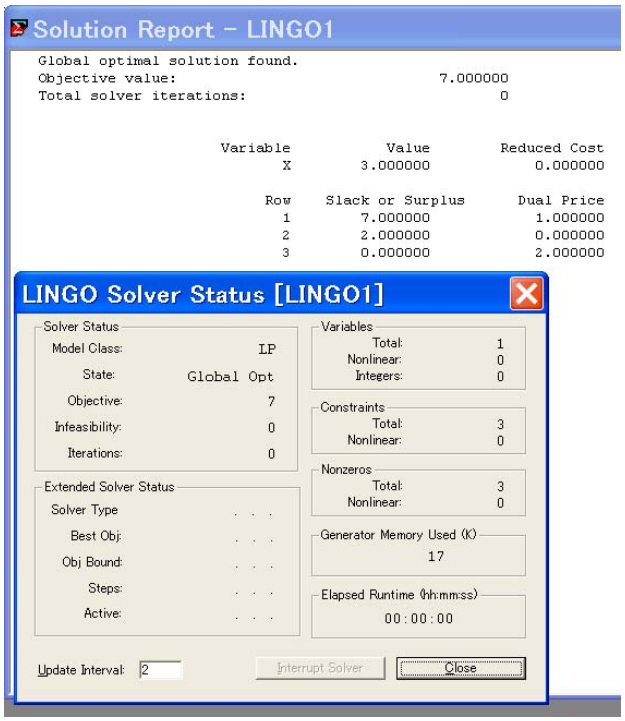

図 1.3　LINGO の Solution Report 画面と Solver Status 画面

すなわち，定義域は$-\infty<x<\infty$であるけれど，値域は$-1\leqq y$となる．すなわち，x=1 で y=−1 が最小値になる．上に凸の放物線であれば，最小値に変わって，最大値が存在する．

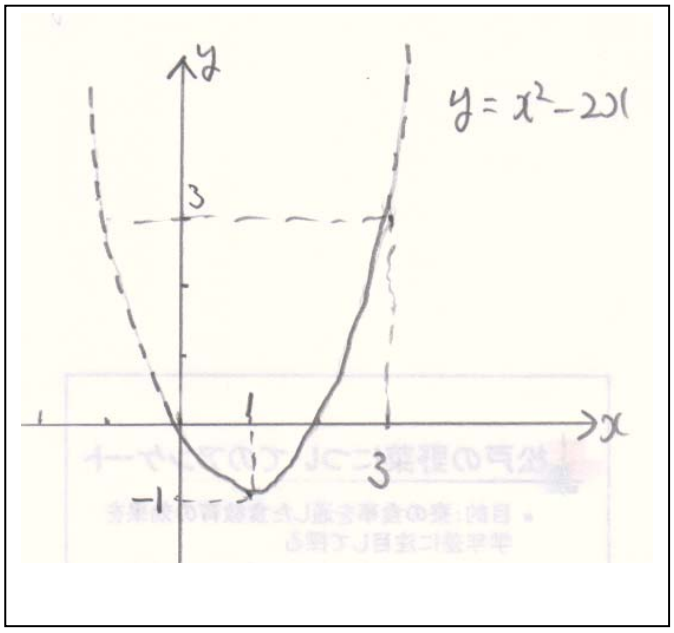

図 1･4　2 次式のグラフ

目的関数が 1 次式の場合と異なり，最小値(最大値)は必ずしも定義式の端で現れない点が異なっている．

1 次式や 2 次式の最大値や最小値は，グラフで表す事で簡単に分る．しかし，もっと複雑な関数は図でもって理解できない．そこで，最大値あるいは最小値を求める一般的な方法が微分である．微分は，数理計画法の理解に直接関係無いが，最大や最小を扱う一般的な方法なので簡単に紹介する．上の 2 次式を x で微分することを，dy/dx あるいは y'のように表わす．

$$y'=2x-2$$

y'=0 となるのは，x=1 である．すなわち，y'(1)=2-2=0，y(1)=1-2=-1 である．この点(1，－1)で接線の傾きは 0 すなわち水平になるので,最大値あるいは最小値をとることが分る．最大値になるか最小値になるかは，y'をもう一度微分した 2 階微分 $y''=2$ の正負で決めることができる．正であれば最小値になり，負であれば最大値になる．この場合は 2 で正なので，最小値になる．

x に 0 を入れると y'(0)=－2 になるが，これは点(0,0)でこの 2 次式の接線の傾きが－2 であることを表す．すなわち，点(0,0)でこの 2 次式の接線は y=-2x になる．

さて，定義域を 0≦x≦3 とすれば 2 次式は**図 1・4** のように実線に制限され，端点（3，3）で最大値，端点でない（1，-1）で最小値になる．すなわち，値域は[-1,3]になる．

さて，これを LINGO で表わすと次のようになる．「＾」はべき乗をあらわす．

```
MIN=x^2-2*x；
X>0；
X<3；
```

このように，目的関数が 2 次式のものを，2 次計画法（QP）という．これが，ノーベル経済学賞をとった「ポートフォリオ分析」のモデルになっている．また，回帰分析の最小二乗法のモデルにもなる．

（4）n 次方程式

入力変数 x に対し，出力に最高次の項 x^n が表われるものを n 次の多項式という．ここで n=3，すなわち 3 次の多項式を考えてみよう．

$$y=f(x)=ax^3+bx^2+cx+d$$

暗黙の了解であるが，一般的に a，b，c，d ぐらい迄は，方程式では定数を表わす．ｉ，j，は添字を表わし，x，y，z は変数を表わしている．n 次多項式のように…の省略が現れるような場合は，係数を a，b，c のように使い分けしないで，a や b に添字をつけて次のように表わすことが多い．

$$y=f(x)=a_0x^n+a_1x^{n-1}+\cdots+a_{n-1}x^1+a_n$$

さて，a =1，b =0，c =-1，d =0 である次の 3 次多項式(3 次関数)を考える．

$y=x^3-x$

$=x(x+1)(x-1)$

これを微分すると，次のようになる．

$y'=3x^2-1$

これが 0 になるのは $x=\pm 1/\sqrt{3}$ であり，この 2 点で接線は水平になる．$x<-1/\sqrt{3}$ を満たす任意の点 $x=-2$ では，$y'=3(-2)^2-1=11>0$ となるので，この区間での接線の傾きは正になる．そこで，y' を「+」と表記し，y は増加傾向を示す記号の「↗」で表わす．$-1/\sqrt{3}<x<1/\sqrt{3}$ を満たす任意の点 $x=0$ では，$y'=-1<0$ になるので，y' は「-」と表わし，y は減少傾向を表わす「↘」で表わす．$1/\sqrt{3}<x$ である任意の点 $x=2$ では $y'=11>0$ になるので，y' は「＋」と表し，y は増加傾向を表す記号「↗」で表す．以上をまとめて，**表 1・1** のような増減表を作る．

表 1・1　3 次関数の増減表

x	…	$-1/\sqrt{3}$	…	$1/\sqrt{3}$	…
y'	＋	0	－	0	＋
y	↗	極大	↘	極小	↗

この 3 次式は，**図 1・5** のように表わされる．定義域は $-\infty<x<\infty$ で，値域も $-\infty<y<\infty$ であるが，$x=-1/\sqrt{3}$ で山の頂上，$x=1/\sqrt{3}$ で谷底になる．山の頂上のように，その点の周りをみわたしても，それより大きな値がない場合を「極大値」という．ただし，例えば $x=2$ で $y=6$ になるので，$x=-1/\sqrt{3}$ は最大値ではない．$x=1/\sqrt{3}$ のように周りにその点より小さい値がない場合，すなわち谷底の状態を「極小値」という．ただし，例えば $x=-2$ で $y=-6$ になるので，この点は最小値ではない．これらを併せて極値という．

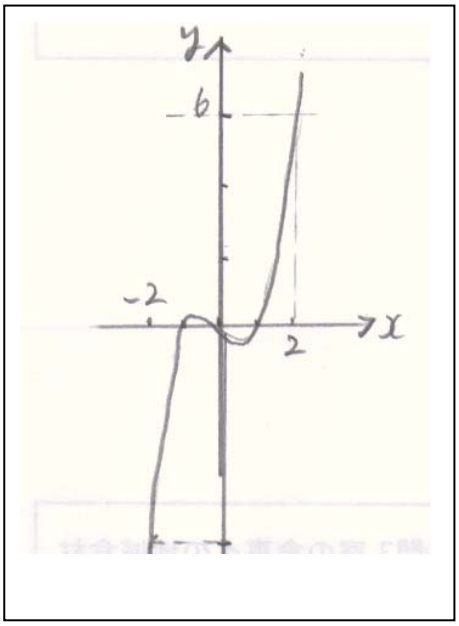

図 1・5　3 次多項式

図 1・5　3 次関数

すなわち，この 3 次式には最大や最小値はないが(あるいは∞と-∞)，極大値（-1/√3,2*√3/9）と極小値（1/√3,-2*√3/9）が存在する．しかし，定義域を 0≦x≦2 とすれば，x=1/√3 は極小値であり最小値でもある．x=2 で最大値 6 が求められる．

これを LINGO でモデル化すると次のようになる．「＾」はべき乗を表す．

```
MIN=x^3-x;
X>0;
X<2;
```

目的関数や制約式が 1 次式で表わされないものを，一般に非線形計画法（NLP）といっている．簡単な非線形問題では極大値や極小値は容易に求まるが，それが最大値か最小値かの判断が難しい点が非線形計画法のこれまでの悩ましい問題であった．

（5）まとめ

極大値と極小値は，非線形最適化と呼ばれる分野で重要な概念である．

最小値は，値域で一番小さい値．

最大値は，値域で一番大きい値．

極小値は，その値の周りにそれより小さな値が無い場合．

極大値は，その値の周りにそれより大きな値が無い場合．

1.3　領域の最大最小問題

（1）数理計画法は高校数学のテーマである

高校の「数学 II」は，文科系の受験生も一応 2 年生で履修することになっているらしい．ただし，かなりの高校生が数学 I しか学習していないようだ．その中で，領域の最大値・最小値という次のような問題がある．（戸田宏編，啓林館）

問）連立不等式 x≦5，x+ y≦10，x +2y≦16，x≧0，y≧0 の表す領域 D を図示し，点（x，y）がこの領域を動くとき 2x+3y の最大値と最小値を求めよ

読者も，高校生にたちもどってチャレンジして欲しい．実は，この問題が本書のテーマである数理計画法の代表である LP 理論そのものである．数学は，一般的にいって，役に立たない，難しい，といった間違ったイメージがある．これにも一理ある．数学者は，できるだけ一般化や抽象化を好む人種である．このため，どんな分野に応用できるかの説明を欠いていたことに大きな問題があろう．

「数理計画法」は“Mathematical Programming”の訳で，現実の問題を数式でモデル化

し（Mathematical)，計画を立てる学問（Programming）である．

領域の最大・最小問題は，数理計画法という現実に役立ち，学問としても面白く重要な問題を，高校数学では味も素っ気も無い「領域の最大最小問題」としているだけだ．

考えてもみてほしい，この問題は連立方程式だけ，すなわち四則演算が理解できればそれで十分理解できる内容である．それだけで，人間社会のかなり多くの問題を理解し計画を立てる新しい力を与えてくれる．

（2）教科書の解答

教科書の解答は，次のようなものである．

求める領域 D は，原点 O(0,0)，点 A(5,0)，B(5,5)，C(4,6)，D(0,8)を頂点とする五角形 OABCD の周および内部である．つまり，境界を含む**図 1･6** の斜線部分である．

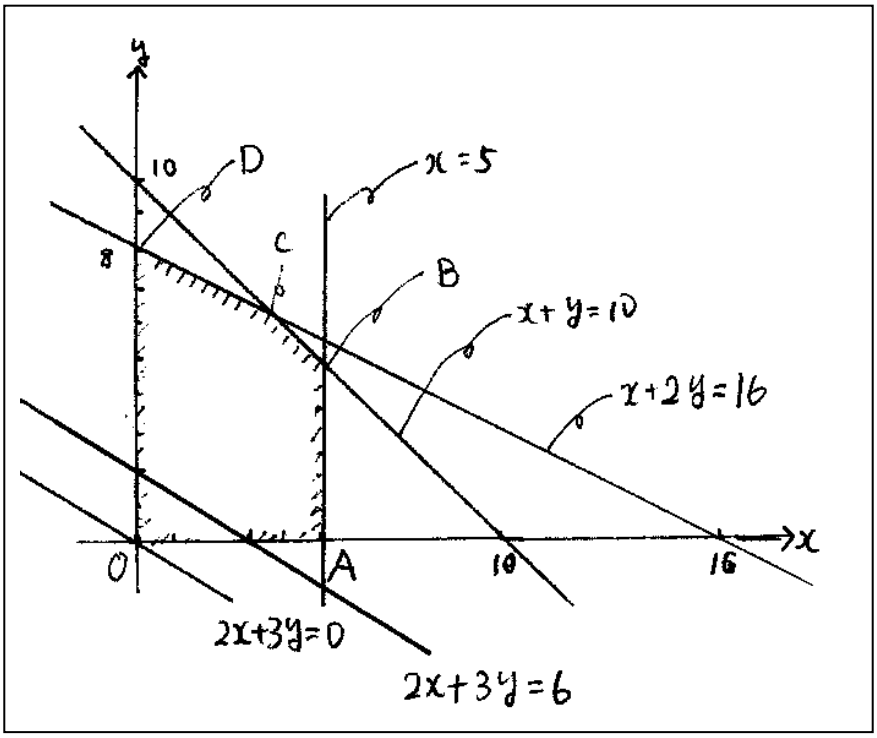

図 1･6　領域の最大・最小問題の図による解

いま，2x+3y=k とすると，y=－2x/3+k/3 だから，これは，傾きが-2/3 で y の切片が k/3 の直線を表している．

この直線が領域 D と共有点をもって動くとき，k の値,つまり y 切片が最大になるのは，直線が点 C(4,6) を通るときで，このとき，k＝2*4+3*6=26 となる．

したがって，2x+3y の最大値は 26 である．この他，5 角形のすべての頂点の値を目的関数に代入し，最大値を求める「総当り法」もある．

この問題が，本当に現実問題で役に立つのだろうか？

問）この問題を LINGO モデルに変えてみよう．

答）　MAX=2*x+3*y;

　　X<5；

　　x+y<10；

　　x+2*y<16；　　ただし，x≧0，y≧0

数理計画法では x≧0，y≧0 のように決定変数の非負条件は前提としているので，モデルに入れる必要はない．

1.5　現実への適用

(1) PC の生産計画

さて，1.3 で紹介した問題は現実にどのように利用されるのであろうか．

まず不等式を表す変数 x と y について考えてみる．変数(variable)は，値が変わりうるものを数学では x，y，z などのアルファベット記号を使って表す．

例えば，最近では中学生や高校生でも PC の組み立てをホビーにする若者もいる．話を簡単にするために,シャーシとハードディスクの 2 つの部品から作られている PC を自宅で組み立て，販売するものとしよう．

廉価 PC(S 台生産)は，1 つのシャーシと 1 つのハードディスクから作られる．高級 PC(D 台生産)は，1 つのシャーシと 2 つのハードディスクから作られる．

今資金繰りの関係から，シャーシは 10 個，ハードディスクは 15 個しか購入できない．そして，廉価 PC(S)は 10 万円で，高級 PC(D)は 15 万円で販売するものとする．このとき売上を最大化したいと思うのは人情である．そこで問題になるのは，廉価 PC と高級 PC をそれぞれ何台作ればよいだろうということが経営上問題になってくる．

(2)LP モデルの定式化

このように，決めてやりたい生産台数を記号 S と D を用いた変数で表すことにする．数理計画法では，決定変数（Decision Variable）といっている．すなわち，廉価 PC を S 台と高級 PC を D 台作ることにする．決定変数 S と D を用いることで，具体的に 5 台とか 10 台という台数が決まらなくても,売上額を求める式が分かる.すなわち,売上額は(10S+15D)万円になる．今この売上額を最大にしたい．

数理計画法では，この 1 次式(10S+15D)を目的関数といい，それを最大化するので次のように表す．

　MAX= 10*S+15*D;

しかし，この売上高は無限にできない．それは，PC を生産するための部品によって制約を受けるからである．今の場合，シャーシが 10 個，ハードディスクが 15 個という部品の

手持在庫によって生産計画が制約されるわけだ.

廉価 PC を S 台作るには，シャーシは S 個必要である．高級 PC を D 台作るには，シャーシは D 個必要になる．そして，結局(S+D)個のシャーシが必要になる．今シャーシの在庫は 10 個しかないので，(S+D)は 10 個以下で無ければならない．これを表すのが，次の不等式である．左辺はシャーシの「使用台数」であり，右辺はシャーシの「在庫数」である．

S+D≦10

すなわち不等式は，生産活動においては，部品や資金や労働可能な資源の制約を表すのに用いられる．この意味で，数理計画法では制約式と呼んでいる．

同様にして，ハードディスクの制約式は，次のようになる．

S+2D≦15

この外，生産台数は負にならないので，次のような非負の制約条件が数理計画法ソフトの内部で自動的に付加される．

S≧0,　D≧0

以上を LINGO の記述方法でまとめれば，次のように表される．

```
MAX=10*S+15*D;
S+D<10;
S+2*D<15;          ただし，S≧0,　D≧0
```

(3)　ゆとり教育について

S と D を x と y に読み替えれば，x≧0, y≧0, x+y≦10, x+2y≦15 を満たす領域で，10x+15y を最大にする値を求めよという 1･3 章の(1)の領域の最大最小と同じ問題になる．というよりも，(1) の領域の最大問題は数理計画法でプロダクト・ミックスすなわち部品制約から最終製品の生産個数を決める「製品混合」と呼ばれる問題を，抽象化したに過ぎない．

このように領域の最大最小問題として味も素っ気も無い形で教えられれば，興味を覚える学生も少なくなるのは当然であろう．

ゆとり教育とか創造性ある能力を養うためには，多くの人に興味がある，現実の応用例にまで踏み込んで，分かりやすく教えることが重要でなかろうか．

今日の数学教育の受難は，近日出男・曽野綾子の有名作家夫妻の「難しい数学なんて人生で役に立たない」に代表される数学無用論に端を発しているようだ．芸術家のように一芸に秀でた人には数学は一生必要ないかもしれない．しかし，井上靖の「敦煌」にさえ王女が城壁から放物線を描いて身を投げる感動的なシーンが出てくる．それが何だといわれればそれ以上反論できないが，2 次式で表される放物線を知っていればその場面が臨場感をもって体験できる．

また，一般の我々は，製造に携わらない人でも，組み立て加工に代表される製造業の重要な一面を数理計画法で簡単に理解できることは，人間社会の一員である限り不用とは言えまい．小説は直接的な利益を追求しない最たるものである．その小説家の考えが，日本の教育に与えた影響はあまりにも大きい．

「最近の学生は，理数系の学問についてこられない」というのは，評論家の言である．そのようにしたのは，我々大人の世代の責任である．しかし好都合なことに，この難しい理数系の学問が，使いやすいソフトでもって実際の問題を解いて解決できることが可能になった．統計から始まって，数理計画法や数学を「問題解決の２１世紀の一般教養にしたい」というのが私の人生の最後の使命であると思っている．

（4）柔らか頭をもとう

上の問題に関しては，現実と照らし合わせると色々と疑問が出てくる．

例えば,「企業にとっては売上高よりも利益が重要でないか？」ということである．バブル以前は，オールドエコノミーに属する企業の多くは，業界内のシェアすなわち売上を重視したものである．これに対し，バブル後不況になり，同一業種内で勝ち組み負け組みが明らかになるにつれ利益が重視されるようになった．この場合，廉価 PC と高級 PC の利益が 1 万円と 1.5 万円とすれば，目的関数を単に(S+1.5D)と変更すればよい．制約式は，変更する必要は無い．

最近の学生は，製造業よりもサービス業になじみがあるようだ．その理由はいくつかある．製造業は額に汗してダサイのに比べ，サービス業はなんとなくファッショナブルであるという世間知らずの思い込みがある．またアルバイトでなじみがある，などである．しかしデフレ経済化における一部の勝ち組みと考えられたマクドナルドも，2003 年後半には経営悪化の責任を取り伝説の経営者の藤田田氏が退任する事になった．外食産業全般は，社会経験の浅い学生は，意外と労働がきつく給与水準が低いことを知らない．

そこで，製造業の問題だと興味が沸かないのであれば，この問題を学生に人気のあるハンバーガーショップに置き換えて考えてみよう．これは最近組み立て産業化したチェーン・レストランに普遍化できる．すなわち，チェーン・レストランは工場で食材を加工し，店で組み立て加工しているわけだ．

例えば，標準バーガー（S）は 1 枚のチーズと 1 枚の肉パテから作られる．高級バーガー（D）は 1 枚のチーズと 2 枚の肉パテから作られている．チーズの在庫は 10（単位千枚）で肉パテは 15 の在庫があり，標準バーガーは 10 円の利益，高級バーガーは 15 円の利益がある．この問題は決定変数の S と D を PC からバーガーに読み替えただけで，同じ領域の最大最小問題であることが容易に分る．

すなわち，数理計画法は決定変数の意味を単に読み替える事で，色々な分野で利用できる．これは入力形式が一種類で単純なのに，色々な分野に応用できる魔法の秘密の一つである．また，部品を組み合わせて最終製品を作る問題であれば，全てこのモデルを雛形にして修正すればよい．雛形モデルは，これまでの人類の天才や秀才が開発してきた．後で紹介するノーベル経済学賞を受賞したポートフォリオ分析のモデルは，自分で考えだすことは難しいが，理解し利用する事は簡単だ．

2　非線形モデル

京大の理学部で「線形代数」に頭を悩ませていたころ，1 年下の矢野環君（現，同志社大学教授）は「Non Linear Algebra」の原著を読んでいてびっくりした．線形でも手一杯なのに，後輩は非線形の世界に遊んでいて負けたと思った．

しかし，社会人になって統計の世界の非線形回帰分析や数理計画法の非線形最適化などを自然な形で受け入れられた．世の中の現象は，大概は非線形現象である．その解を求める場合，ソフトウエアの能力が低い場合は線形で近似しなければ困難なことが多いという単純なことに気づかなかっただけである．そして，統計ソフトの機能が改善され，非線形回帰モデルが線形回帰モデルと等しく解けるようになって，私も線形と非線形の間を自由に浮遊できるようになった．

一方，数理計画法の世界では，なかなか非線形最適化（Non Linear Programming, NLP）が線形計画法のように使いやすいものではなかった．ようやく LINGO の８版以降でそれが可能になった．例えば，NLP の大家らの著である『GINO による非線形最適化（共立出版)』をみれば，GINO で扱えた非線形の世界がいかに幼稚であったかが分かる．

読者も，LP や NLP の違いを気にすることなく，これらの間を自由に浮遊しましょう．

注：GINO は LINDO Systems Inc.の 3 番目の製品であるが，LINGO の開発で生産停止した．

2.1　非線形最適化と大域的最適解

数理計画法を非線形最適化モデルから紹介することは，これまでの常識では考えられないことである．通常は，LP，QP，IP，NLP のようにアルゴリズムの容易さの順で紹介するのが一般的であろう．しかし，多くの読者は数理計画法のアルゴリズムに関心があるのではなく，実際の問題を解くことに興味がある．この場合は，これらの解法の特長さえ抑えておけば，解法をことさら意識する必要はないのである．

一般に，世の中の最適化したい現象は非線形な連立方程式で記述できる．これまでソフトウエアやコンピュータの能力が強力でなかったため,計算が容易な線形モデルに近似(置き換えて）して扱う必要があった．現象をうまく線形モデルで近似できれば，現在でも線形モデルとして扱うことに越したことはない．

しかし，非線形でしか記述できない問題もある．この場合は，気楽に NLP として扱えばよい．NLP で気をつけることは，得られた解が「局所的な最適解(極値)」であって,「大域的な最適解（最大値/最小値)」でないことを理解しておくだけである．

2.2　局所最適解と大域的最適解

LP モデルで解が見つかった場合，1 章で紹介したとおりそれは間違いなく最善の解であり，大域的最適解（global optimum）である．これは，非線形モデルの場合には当てはまらない．非線形モデルは，局所最適解と呼ばれる解が幾つかあり，それは近くにそれ以上よい実行可能解がないということ示しているにすぎない．

大域的最適解は，実行可能解の中で一番良い解である．これまでの非線形モデルでは，単に局所解が見つかるだけで，大域的最適解が必ずしも見つかるわけではないことに注意すべきである．

非線形の特徴は，複数の局所最適解があることだ．y=f(x)=x*sin(Π*x)というモデルを考えてみよう．X の定義域を[0，6]とすれば，**図 2・1** で表される．

最小値を探す場合，初歩的な数理計画法ソフトは初期値の状態によって X の局所解として 0，1.564，3.529，5.518 のいずれかを見つける．すなわち，初期値に一番近い局所解を見つけ，それ以上の改善ができなかった．この場合，5.518 が大域的最適解(最小値)になる．

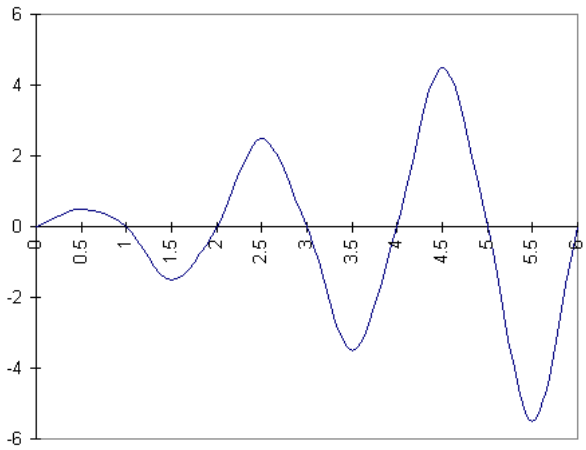

図 2・1　合成関数 Y=f(x)=x*sin(Π*x)

このグラフを一連の丘と考えよう．あなたは，真っ暗な中，最も標高が低いところを探している．探索を X=3 から始めた場合，左へ進む一歩一歩は上り坂である．右へ進むと下り坂となる．したがって，低い点を探すために右へ進む．あなたは，この坂が下っている限り右方向へ進む．X＝3.529 に到達すると，小さな平坦な場所に出る（傾斜が 0 の場所）．そして，右に行き続けると上り坂になる．左へ戻ると下ってきた坂をまた上る．現在すぐそばに見つかる範囲で一番低いところ－局所的に最も低い地点－にいる．それははたして一番標高の低いところなのだろうか.暗闇の中では判断がつかない.これまでのソフトは，このような状態であった．

これまでの非線形ソルバーは,初期値に一番近い局所最適解を探す.そこで初期値を色々変えることで，局所最適解を探索することができる．初期値を変えて何回か解いてみて，より良い解をみつけ大域的最適解であってほしいと祈るだけであった．しかし，LINGO の８版から大域的探索オプションで，大域的最適解を探すことが可能になった.

2.3　LINGO で解いてみよう

このモデルを LINGO のスカラー形式（自然表記）で記述すると次のようになる．INIT:と ENDINIT は，初期値（ x=0.1 ）を定義する INIT 節である.

```
MIN=x*@sin(3.1415*x);
x<=6;
INIT:
x=0.1;
ENDINIT
```

わざわざ初期値を x＝0.1 にして解いても,以前の LINGO と異なり初期値に最も近い局所解でない**図 2.2** の解が出力される．この様な簡単なモデルの場合，大域的最適解のオプションを指定しなくても，大域的最適解が出力される．しかし，最初の行の表示に見るように「Local optimal solution found」ということで大域的最適解と保障していないので，読者はこれが大域的最適解かどうか分からない．LINGO メニューで[OPTION]->[Global Solver]->[Use Global Option]を指定すれば，大域的探索が行われ「Global optimal solution found」というメーセージが出力され大域的最適解であることが保障される．この保障ができるようになったのは，2000 年以降のことである.

```
 Local optimal solution found.
 Objective value:                              -5.509345
 Extended solver steps:                                5
 Total solver iterations:                             34
                      Variable           Value        Reduced Cost
                             X        5.518503            0.000000
                           Row    Slack or Surplus      Dual Price
                             1       -5.509345           -1.000000
                             2        0.4814968           0.000000
```

図 2.2　LINGO のモデルと解

2.4　2 次計画法

図 2.3 は，下に凸な 2 次関数である．目的関数が 2 次関数で，制約条件が線形制約の場合を 2 次計画法（QP）という．この場合，LP と同じく局所最適解は大域最適解であり，最小値を求めることができる.非線形関数の凸性が,すなわち 3 次関数か 2 次関数以上かで,大域最適解が 1 つあるか複数の局所最適解があるかということを決定する.

QP の例として，後でポートフォリオ分析を紹介する.

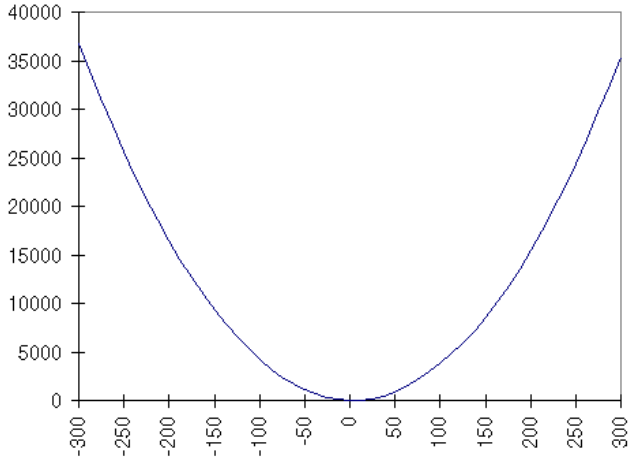

図 2.3　完全な凸関数:　0.4*(A1-3)2 +0.5

幾何学的に，関数上もしくはそれよりも上の領域に含まれる 2 つの点を結ぶ直線が全体的にその関数上もしくはそれよりも上の領域に含まれる場合，凸であると定義される．上記のグラフにみられるように，制約のない凸関数は，唯一の最小値をもち，初期値に関係なく大域的最適解が得られる.しかし,関数が凸状でない場合,複数の局所解があるため,求まった解は大域的最適解ではない．複数の変数を持つ関数の凸性の決定は容易でない．数学では，2 次微分の行列全てが正定値，あるいは正の固有値を持つ場合凸になる．そして，2 次微分の全てが非負なら凹である．凸と凹は逆向きの関係にある.

2.5　箱の設計

(1) 問題の概要

ある電機会社で，種々の部門の要求にあう製品の筐体（キャビネット）を，最小費用で設計したい.技術部門は,機器の熱を分散させるために少なくても 1516 立方インチの体積をもち，888 平方インチの表面が必要と考えている．営業部門は，筐体の床面積が 656 平方インチ以下だと売れ行きが良いと考えている．最後に，デザイナーは美的な要求から，縦横比が 0.618±0.1（0.518 から 0.718 の間）であることを望んでいる．筐体製造に用いる金属板の費用は$0.05/平方インチであり，前と後ろのパネルに要求される人件費は

$0.10／平方インチである.

(2) LINGO のモデルと大域的最適化オプション

これを LINGO でモデル化(2BOX1.lng にこのモデルが格納されている)すると次のようになる. 筐体の横幅を W, 奥行きを L,高さを H で表している.

```
MIN=0.1*(L*W+L*H)+0.2*H*W;
L*W+L*H+W*H>444;
W*L<656;
L*W*H>1516;
W/H<0.718;
W/H>0.518;
END
```

これを解くと, 実行可能解がない (No feasible solution found.) という次のような出力が出される.LP の場合,制約式がきつすぎて実行可能領域が存在しないことを表すので,制約式を緩める必要がある. NLP の場合, 実行可能解があっても, 間違った方向に探索した結果, このメッセージが出る場合がある.

```
No feasible solution found.
Extended solver steps:                     0
Total solver iterations:                  27
```

なぜこのようになったのだろうか?デフォルトの初期値の設定で L,W,H が 0 に収束し,解がもとまらなかったためである. そこで, 大域的最適化オプションを指定すると図 2.4 の大域的最適解が求まる. このような単純なことが最近になってようやく可能になったわけである.

```
Global optimal solution found.
Objective value:                              51.01855
Extended solver steps:                              10
Total solver iterations:                           712
                     Variable           Value        Reduced Cost
                            L        22.90530            0.000000
                            W        6.893564            0.000000
                            H        9.601064            0.000000
                          Row    Slack or Surplus      Dual Price
                            1        51.01855           -1.000000
```

2	0.000000	-0.4606858E-01
3	498.1008	0.000000
4	0.000000	-0.1344068E-01
5	0.000000	2.329120
6	0.2000000	0.000000

図 2.4　箱の設計の出力 1

(3)　初期値を設定する

大域的最適化オプションを使わないで，初期値を変えて対応してみよう．L,W,H の初期値が，1 や 1.5 ではやはり実行可能解がない．2 にすると**図 2.5** の解がもとまる．解は同じであるが，Global オプションを指定していないので表示は局所解（Local optimal solution found.）である．この場合，他にもっとよい解がある可能性は否定できないので，読者は宙ぶらりんの状態に置かれる．

```
MIN=0.1*(L*W+L*H)+0.6*H*W;
L*W+L*H+W*H>444;
W*L<656;
L*W*H>1516;
W/H<0.718;
W/H>0.518;
INIT:
L=2;
W=2;
H=2;
ENDINIT
END
```

```
Local optimal solution found.
Objective value:                              51.01855
Extended solver steps:                           10
Total solver iterations:                        712
                     Variable           Value        Reduced Cost
                            L        22.90530            0.000000
                            W        6.893564            0.000000
                            H        9.601064            0.000000
```

Row	Slack or Surplus	Dual Price
1	51.01855	-1.000000
2	0.000000	-0.4606858E-01
3	498.1008	0.000000
4	0.000000	-0.1344068E-01
5	0.000000	2.329120
6	0.2000000	0.000000

図2.5　初期値によるアプローチ

(4)　モデルを変える

このモデル化の問題は，デザイナー要求の縦横比（W/H）をそのままモデルに取り込んでいる点である．これによってHが0の方に収束していくと，0で割ることでプログラムが停止する．以前のLINGOでは「0で割る状態」になり，エラーと表示していたが，LINGOの10版では「実行可能解なし」と表示している．次のように，デザイナー要求は簡単に線形不等式に変形でき，大域的最適化オプションを用いなくても解は**図2.5**と同じものが求まる．すなわち，モデル作成に際してできるだけ非線形的な要素は避けるように努力すべきだ．

```
MIN=0.1*(L*W+L*H)+0.6*H*W;
L*W+L*H+W*H>444;
W*L<656;
L*W*H>1516;
W<0.718*H;
W>0.518*H;
END
```

(5)　非線形連立不等式の解

最適化ソルバーは，目的関数を指定しなければ，非線形連立不等式の解を一つ求めてくれる．目的関数の前に「！」を入れることで，目的関数はコメントになる．

```
!MIN=0.1*(L*W+L*H)+0.6*H*W;
```

これを解くと，次の**図2.6**の解がもとまる．すなわち，LINGOは非線形の連立方程式の解を求めてくれる強力なソフトウエアである．国立循環器病センターの先生が，グラフ表示装置に出た関数グラフの最大/最小の点を表示するのに機能の劣ったGINOを用いられていたのが思い出される．

```
  Feasible solution found.
  Extended solver steps:                                0
```

Total solver iterations:	360

Variable	Value
L	517.9766
W	1.265663
H	2.312443

Row	Slack or Surplus
1	1412.302
2	0.4162400
3	-0.9120313E-06
4	0.1706728
5	0.2932719E-01

図2.6　非線形連立不等式の解

(6)　丸め解

さて，話を元に戻そう．実際に生産するため，整数解を求めたい．これまでは実数解を四捨五入し,L=23,W=7,H=10のような制約をモデルに追加することで求まる丸め解でお茶を濁すことが多かった．解は**図2.7**の通りである．

Global optimal solution found.	
Objective value:	53.10000
Total solver iterations:	0

Variable	Value	Reduced Cost
L	23.00000	0.000000
W	7.000000	0.000000
H	10.00000	0.000000

Row	Slack or Surplus	Dual Price
1	53.10000	-1.000000
2	17.00000	0.000000
3	495.0000	0.000000
4	94.00000	0.000000
5	0.1800000E-01	0.000000
6	0.1820000	0.000000
7	0.000000	-1.700000
8	0.000000	-4.300000

9	0.000000	-3.700000

図 2.7　丸め解

(7)　整数解を求める

しかし，@GIN(General Integer)で変数を 0 以上の一般整数変数に指定できる．0/1 の 2 値の場合は@BIN（Binary Integer）である．これによって**図 2.8** のように丸め解より 1.7 だけ良い解が求められる．この違いが，利益に大きく利いてくる場合は，丸め解はやめ整数解を求めるべきであろう．

もっと複雑な非線形モデルで，整数解を得ることができるようになったのはごく最近のことである．

```
MIN=0.1*(L*W+L*H)+0.2*H*W;
L*W+L*H+W*H>444;
W*L<656;
L*W*H>1516;
W/H<0.718;
W/H>0.518;
@gin(H);
@gin(W);
@gin(L);
END
```

```
Global optimal solution found.
Objective value:                              51.40000
Extended solver steps:                             10
Total solver iterations:                         1881
                     Variable           Value        Reduced Cost
                            L        22.00000          -0.4103441
                            W        7.000000           0.2275875
                            H        10.00000           0.000000
                          Row    Slack or Surplus      Dual Price
                            1        51.40000          -1.000000
                            2        0.000000          -0.1241379
                            3        502.0000           0.000000
```

4	24.00000	0.000000
5	0.1800000E-01	0.000000
6	0.1820000	0.000000

図 2.8　丸め解より良い整数解

2.6　方程式の球解

次は,『パソコン楽々数学(講談社ブルーバックス)』で紹介したローンの計算式である. P はローンの総額である. a は例えば月次返済金額, i は月金利, n は返済月数を表す. P はこの式のように P= f (a,i,n)のように他の変数 a,i, n の関数で表されるので, これらの値を代入すれば簡単にローン総額が Speakeasy や Excel で計算できる. 同様に a も計算できる. しかし, i や n は i=f(a,P,n)のような明示的な関数で簡単に表せない. このような暗示的な場合でも, LINGO は解を求めることができる.

```
MIN=N;
P=a*(1-(1+i)^(-n))/i;
P=5000;
a<=20;
i=0.04/12;
@GIN(n);
END
```

ローンを 5000 万円借りたい. 年金利 4%の月次均等払いで, 20 万円以下の返済を考えた際, 何ヶ月返済になるであろうか. この方程式は, 金利 i と支払い月数 n は他の変数による明示的な関数でないため, 多くのソフトウエアで解くことは難しい.

解は次の通りである. すなわち,539 ヶ月となり,約 45 年間の返済になる. 一般の 30 年返済に従うとすれば, ボーナス返済の併用が必要になる.

```
  Local optimal solution found.
Objective value:                              539.0000
Extended solver steps:                             4
Total solver iterations:                         447
                     Variable           Value        Reduced Cost
                            N        539.0000            1.000000
                            P        5000.000            0.000000
                            A        19.99633            0.000000
```

I	0.3333333E-06	0.000000
Row	Slack or Surplus	Dual Price
1	539.0000	-1.000000
6	0.000000	0.000000
3	0.000000	0.000000
4	0.7667659E-06	0.000000
5	0.000000	0.000000

図 2.14　ローンの返済年数の計算

読者も，難解な関数であっても，それを LINGO で定式化することは容易である．後は，解を求めるため[SOLVE]ボタンを押すだけである．

3　組み立て産業への応用

3.1　プロダクト・ミックス

製造業は,大きく分けると部品を組み立てる「組み立て産業」と石油会社などの「装置(パイプライン)産業」に分かれる．組み立て産業は，自動車，電気，機械産業に限らず，工場で製造された食材を店頭で調理するファースト・フード産業が考えられる．この種の数理計画法モデルは，プロダクト・ミックスという雛型モデルになる．

ここで取り上げるモデルをみて，読者は自動車産業や電気産業に応用できるとは考えないであろう．組み立て産業では，トヨタの看板方式や，Dell のように工場の横に部品倉庫を造り，部品メーカーの負担で在庫を切らさないようにさせる方が製造費用に大きな効果があるためである．

実際問題として，プロダクト・ミックスは製造装置とからめて生産計画を作成するという雛形モデルを応用しなければ，現実に利用されないであろう．しかし，本書では紙面の都合で紹介しないが，Linus 教授の解説書に紹介されている．

本書でプロダクト・ミックスを紹介するのは，１）領域の最大問題そのものである，２）装置産業の雛形モデルである配合問題と双対な関係（4 章で説明）にある．配合問題は鉄鋼業や石油産業を中心に今日でも数理計画法の大ユーザーである，３）減少費用や双対価格といった LP の重要な説明に適していること，などである．

3.2　PC の製造

(1)　あなたは，パソコンの生産者

組み立て問題（プロダクト・ミックス）は，部品を組み合わせて最終製品を作ることに利用できる．すなわち，電気製品や自動車などのお堅い産業に利用されてきた．しかし，外食産業は食材を工場で生産し店頭で組み合わせるので，同じタイプのモデルがそのまま利用できる．

今，あなたはガレージ産業のオーナーとしよう．自宅で，副社長の奥さんと 2 種類の PC を作っている．標準 PC は 1 個の標準シャーシと 1 個のハードディスクから作られ，1 台 30（千円）の利益がある．高級 PC は 1 個の高級シャーシと 2 個のハードディスクから作られ 50（千円）の利益がある．このようなべらぼうな利益が現実的でないと考える読者は，販売価格と読みかえればよい．これらの部品は，資金繰りの関係で，在庫が 60，50，120 個しかない．経営上の問題は，現在の在庫部品数の制約のもとで，標準 PC と高級 PC を何台ずつ作れば，今月の利益が最大化されるかである．これらの決定変数を S と D で表す．

(2) 人生の分かれ目

このような小さな問題を，教科書的な問題という．このような問題に対して，読者の反応は，きっと二通りに分かれるだろう．「全く現実と違ったばかげた問題だ」と考えるか，「現実の問題は，単に部品や製品の種類を増やすだけで，この問題を応用できる」と考えるかである．さらには，この単純な雛形モデルから，プロダクト・ミックス問題や数理計画法全般に興味を持つか否かが人生の分かれ道になる．

3.3 LINGOによるモデル

(1) モデルと解

このモデルをLINGOで定式化するためExcelのアドイン・ソフトであるWhat'sBest!で提唱されたモデルつくりのABC分析を行う．問題は標準PCをS台，高級PCをD台生産することである（Adjustableステップ）．これで，決定変数SとDが決まる．次に，決定変数を用いて利益（30S+50D）を最大化すればよいことが分かる（Best ステップ）．そして，その後で決定変数を用い部品制約を考えればよい（Constraintsステップ）．雛形モデルがなく，自分でモデルを新しく作る場合，このABC分析の手順でモデル作成を心がければよい．モデルは次の通りである．

```
[_1]MAX=30*S + 50*D;
[_2]S<60;
[_3]D<50;
[_4]S+2*D<120;
```

実行すると，図3.1の出力が得られる．

```
Global optimal solution found.
Objective value:                              3300.000
Total solver iterations:                             1
                     Variable           Value        Reduced Cost
                            S        60.00000            0.000000
                            D        30.00000            0.000000
                          Row    Slack or Surplus      Dual Price
                           _1        3300.000            1.000000
                           _2        0.000000            5.000000
                           _3        20.00000            0.000000
                           _4        0.000000            25.00000
```

図 3.1　解の出力 1

(2)　解の解釈

解の見方は簡単だ. 制約条件[_2], [_3], [_4]を満たす実行可能領域の範囲内で, S=60, D=50 すなわち標準 PC を 60 台, 高級 PC を 50 台作れば利益を最大化でき, 3300 の最適解を得る. このモデルは LP モデルであり, Global オプションを指定しなくても唯一の大域的最適解が得られる.

製品がたくさんある場合, 製品の中には製造しないものも出てくる. 減少費用 (Reduced Cost) は, 製造してはいけない製品を無理やり 1 単位製造した場合, 目的関数が悪くなる量をあらわす. 今回のように, 決定変数の値が正の場合, 減少費用はすべて 0 になる. 選択と集中でなく, 製品のラインナップを増やすなどの事情で支払う無駄な費用を表している. 要するに, 作れば作るほど, 他の製品で儲ける機会を犠牲にして利益の低下を招く.

Rowの下に, 目的関数を[_1]と考え, [_2], [_3], [_4]の3つの制約式に関する情報が表示される. 「Slack or Surplus」の下の数値は, 制約式にSとDの値を代入し, 右辺の定数項から使用量を引いた値である. この値は, (部品在庫 - 使用量)を表すので, 未使用の在庫数になる. 標準シャーシとハードディスクは在庫を使い切ったので0である. 高級シャーシは20個の余裕がある. すなわち, 次からは高級シャーシの在庫を減らすべきであろう.

京都の老舗のように予定分を売り切ってその日店じまいということも考えられる. しかし,製品を売った代金で標準シャーシとハードディスクを追加購入することが考えられる. 双対価格 (Dual Price) は, 標準シャーシを1個追加できれば5だけ利益が改善され, ハードディスクを1個追加できれば25だけ利益が改善されることを示す. この情報を参考にして, ハードディスクを発注したほうが利益貢献することが分かる.

(3)　LP解の状態

今まで, 最適解がある場合だけを考えてきた. 他にどんな場合があるだろうか. **図 3.2** が, 解の状態を示している. すなわち, 最初は実行可能解がある場合とない場合に分かれる. 実行可能解がない場合は, **図 3.3** に示すように, 制約条件に共通集合すなわち実行可能領域がない場合である. 実際の問題では, 考える条件を厳しくしすぎたり, 符号を間違ったりしている場合が多い.

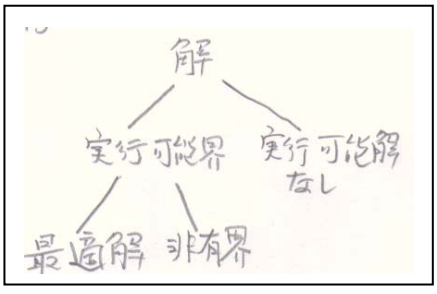

図 3．2　　解の分類

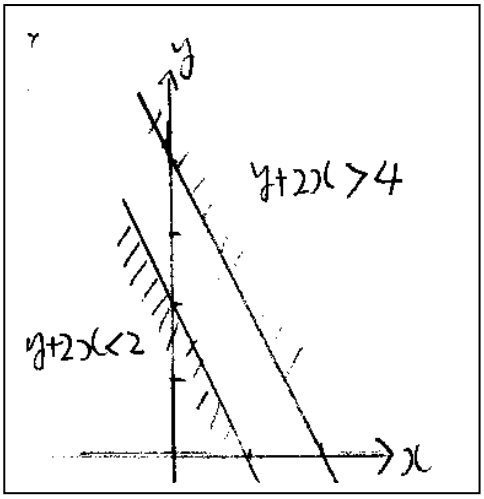

図 3.3　実行可能解なし

実行可能解がある場合でも，制約条件に押さえがない場合は，解が無限に良くなる．これを非有界という．図 3.4 は，非有界の場合である．資源制約がなく実行可能領域が無限に広がっている場合であり，何か重要な制約を見落としている可能性が高い．

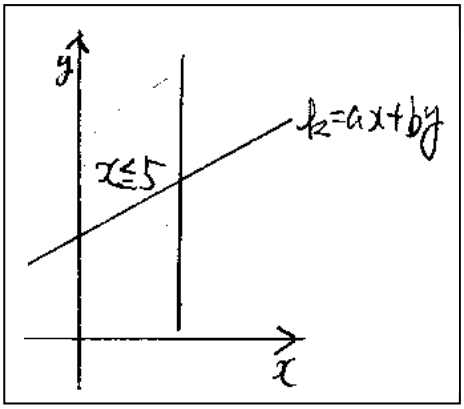

図 3．4　非有界の制約条件

（４）減少費用を考える

今高級シャーシの在庫が 60 個あり，高級 PC の利益が 100 になったとしよう．修正モデルは次のようになる．

```
MAX=30*S + 100*D;
S<60;
D<60;
S+2*D<120;
```

これでめでたく，図 3.5 のように S=0 になり，減少費用が 20 になる．すなわち，今の状況では高級 PC を 60 台生産することが最善であるが,標準 PC を何らかの理由で 1 台生産する

と利益が 20 減少することを示す．あるいは，高級 PC の利益が 100 になったのに合わせ，標準 PC の利益も 20 だけ改善して 50 より大きくなれば，ふたたび S と D は正の値をとり，減少費用は 0 になる．これが減少費用の意味である．

```
Global optimal solution found.
Objective value:                              6000.000
Total solver iterations:                             0
                     Variable           Value        Reduced Cost
                            S        0.000000            20.00000
                            D        60.00000            0.000000
                          Row    Slack or Surplus      Dual Price
                            1        6000.000            1.000000
                            2        60.00000            0.000000
                            3        0.000000            0.000000
                            4        0.000000            50.00000
```

図 3.5　修正モデルの解

(5)　双対問題

今，生産現場で工場を持たずに生産を委託するファブレス産業が注目を集めている．生産者側から見れば，標準 PC1 台あたり 30，高級 PC は 50 の利益を生み，総計 3300 の利益を得られれば，自社ブランドにこだわらずファブレスの工場になってもいい．一方，委託側がこの希望を取り入れるにしても，部品の単価が妥当かどうか検討したい．そこで，標準シャーシ，高級シャーシ，HD の単価を SS，DS，HD として，現在の部品購入費用を最小化したい．一方，標準 PC は 1 個の標準シャーシと 1 個の HD から作られるので，原価費用は SS+HD になる．この値が，解釈に少し無理があるが，利益の下限制約を満たすものとする．最小化問題で，これを上限制約にすれば自明な 0 が解になるので，これを防ぐためである．このようにして，次の双対モデルが作られる

```
MIN=60*SS+50*DS+120*HD;
SS+HD>30;
DS+2*HD>50;
```

このモデルを解くと，次の図 3.6 の解が得られる．標準シャーシの単価は 5，HD の単価は 25 と値踏みされる.これが高いかどうかでこの企業に委託するかどうかの目安にすればよい．高級シャーシは在庫過剰で，価値を認めないことに解釈する．

しかし，重要なことは，双対モデルの最適解は元のモデルとまったく同じ値の 3300 であ

る．そして，決定変数の値はもとのモデルの双対価格に，減少費用は「Slack or Surplus」に，「Slack or Surplus」は減少費用に，「Dual Price」は減少費用に等値である．

この性質は，計算機能力の劣っていた時代には，双対モデルに変えることで，計算時間を早くできるという利点もある．すなわち，n 制約式で m 個の決定変数をもつモデル（n>m）の場合，双対モデルでは m 制約式で n 個の決定変数のモデルになる．LP では，制約式の数を少なくすれば計算速度が早くなることが分かっている．現在では，SVM（サポートヴェクターマシン）や DEA で双対モデルを用いて定式化されている．

```
Global optimal solution found.
Objective value:                              3300.000
Total solver iterations:                             3
                     Variable           Value        Reduced Cost
                           SS        5.000000            0.000000
                           DS        0.000000            20.00000
                           HD        25.00000            0.000000
                          Row    Slack or Surplus      Dual Price
                            1        3300.000           -1.000000
                            2        0.000000           -60.00000
                            3        0.000000           -30.00000
```

図 3.6　双対モデルの解

3.4　汎用モデル

（１）集合表記モデル

次は，プロダクト・ミックス問題を集合表記でモデル化する．モデルから利益や使用部品数や在庫数といったデータを分離するため，データの構造を集合(Set)節で定義し，具体的な値を DATA 節で与える．それを使ってモデルは配列で記述できる．

SETS 節では，この問題に現れる対象（オブジェクト）を考える．2 個の最終製品と 3 種類の部品が主役である．これらの集合名を SEIHIN と PARTS とする．最終製品の属性には各製品の利益(PROFIT)とどれだけ製造したいかの生産個数(PRODUCT)がある．これらはいずれも 2 個の最終製品のもつ属性であり，要素数が 2 の 1 次元配列になる．一方部品集合は 3 個あり，その在庫数（ZAIKO）が属性を表す要素数 3 の 1 次元配列になる．

これまでも Speakeasy（パソコン楽々数学にデモ版のソフトを添付）のように配列演算を行う数学ソフトはたくさんあった．屋上屋を重ねるように，読者は「集合概念を持ち込

むこと」に戸惑いを感じるだろう．しかし，同じオブジェクトに属する配列を同じ集合として繰り返し処理できるところが味噌である．また本書では触れないが，条件設定で集合に属する一部の要素に対してだけ処理することも簡単にできる．

DATA 節では，集合節で定義された配列の値を「割り当て文」で定義している．PROFIT は，廉価 PC の利益が 30(千円)，高級 PC が 50(千円)であることを表す．ZAIKO では，標準シャーシ,高級シャーシ,ディスク装置が 60 個,50 個,120 個あることを示す.USE は PARTS と SEIHIN からできる 2 次元の派生集合である．BUHIN は，3 行 2 列の配列になる．行が 3 個の部品，列が 2 個の製品に対応する 3 行 2 列の行列で，製品 1 台作る際の使用部品数を表す.

```
SETS:
SEIHIN:PROFIT,PRODUCT;
PARTS:ZAIKO;
USE(PARTS,SEIHIN):BUHIN;
ENDSETS
DATA:
PROFIT=30 50;
ZAIKO=60 50 120;
BUHIN=1 0
      0 1
      1 2;
ENDDATA
MAX=@SUM(SEIHIN(i): PROFIT(i)*PRODUCT(i));
@FOR(PARTS(j):@SUM(SEIHIN(i):BUHIN(j,i)*PRODUCT(i))<=ZAIKO(j));
```

その後，利益を最大化にする目的関数がくる．“@SUM(SEIHIN(i):”は，集合 SEIHIN の要素数だけ各製品の利益合計を計算している．通常の数式であれば次のとおりになる．

$$\Sigma_{I=1,2}\mathrm{PROFIT}(i)*\mathrm{PRODUCT}(i)$$

すなわち，次の式を表している．

総利益＝（標準 PC の利益）×（標準 PC の生産台数）
＋（高級 PC の利益）×（高級 PC の生産台数）

次の“@FOR(PARTS(j):”も繰り返しを表す．部品の数（j=1..., 3）だけ次の制約式を考える．

$$\Sigma_{I=1,2}\ \mathrm{BUHIN}(j,i)*\mathrm{PRODUCT}(i)\ <=\mathrm{ZAIKO}(j)$$

すなわち，次の 3 個の制約式を表す.

$\Sigma_{I=1,2}$ BUHIN(1,i)*PRODUCT(i) <=ZAIKO(1)

$\Sigma_{I=1,2}$ BUHIN(2,i)*PRODUCT(i) <=ZAIKO(2)

$\Sigma_{I=1,2}$ BUHIN(3,i)*PRODUCT(i) <=ZAIKO(3)

この繰り返しが容易に行え，同じ集合に属する配列は，一つの形式で行える点が重要だ.

(2)　データを Excel から入力する

上のモデルのように，モデルの構造を表すデータをDATA節で定義していては，その都度モデルを修正する必要がある．オブジェクト・リンク機能をうまく使い，データをExcelから入力することができる．すなわち，“PROFIT=@OLE( );”でもって，図3・7のExcelでセル「C4:D4」に入っている利益を［挿入］→［定義］でセル名を「PROFIT」にしておけば，この値を入力できる．同様にZAIKOとBUHINのセル名を与え，入力する．一般に「@OLE(　　)の ()」の中にExcelのパスを記述すれば，複数のExcelを開いておいても良い．ただし，一つのExcelだけを開いた場合，その記述を省略できて便利である．「＠OLE( )=PRODUCT;」は，計算結果をExcel上のセル名PRODUCTに出力する．結局，これによって，問題のサイズに影響されない「汎用のプロダクト・ミックスモデル」が完成した．読者は，部品組み立て産業にいれば，あるいはファースト・フードの店長であっても良いが，Excel上にデータを記述し，このプログラム（3製品混合3.1g4）をよびだし実行ボタンを押すだけで最適解が得られる．

すなわち汎用モデルは，モデルのサイズの大きさや変更に影響されないコンパクトなモデルである．データだけをExcel上で変更すればよい．

後は，その結果の意味することを提案するだけだ.

```
DATA:
PROFIT=@OLE( );
ZAIKO=@OLE( );
BUHIN=@OLE( );
＠OLE( )=PRODUCT;
ENDDATA
```

	A	B	C	D	E	F	G
1		新村コンピュータ㈱					
2	製品		標準PC	高級PC			
3	製造個数		0	0		利益	¥0
4	利益／台		¥30	¥50			
5			部品制約				
6	部品		要求数				
7			標準PC	高級PC	全使用数		在庫数
8	標準シャーシ		1	0	0	<=	60
9	高級シャーシ		0	1	0	<=	50
10	ディスク装置		1	2	0	<=	120
11							

図 3・7　新村コンピュータ

4　配合計画（Blending）

4.1　物を混ぜ合わせる配合計画とは？

配合計画は，原材料などの物をまぜあわせて，求められた品質基準を持つ最終製品を一番安く作る計画問題だ．製品混合計画と並んで，LP の代表的な計画問題である．また，単なる知的な興味をそそるが，配合計画と製品混合計画は，双対問題になる．

しかし，ブレンディングには悪いイメージがある．平成 6 年の米不足で，日本米とタイ米のブレンドというように，ブレンドには悪いイメージもあるようだ．本章執筆中の 2003 年(平成 15 年)には，肉骨粉による SARS に加え，鳥インフルエンザなど食の安全が大問題になっている．これらは，企業道徳を捨て，一時的な利益優先のため，結果として中長期的に見れば関係者は大きな損失をこうむることになった．

物と物をブレンディングすることは，産業活動で重要な役割を果たしている．例えば，

- 鉄鋼業では原鉱石やコークスなどから最高品質の各種鋼材を一番安く作っている.この場合，原材料費を安くすることが LP の使命である．一方，品質を高めることは製造技術の問題である.
- 石油産業では種類の異なる原油から各種石油製品を作っている.ある石油精製企業では，IBM の汎用機で IBM の数理計画法ソフト MPSX を利用していたが，PC で稼動する What’sBest！に置き換えることで，大きな経費削減と柔軟な運用が実現でき感謝されたことがある.
- また，セメントなどの業界でも配合計画は広く利用されている.

これらの産業は，古くからの数理計画法のユーザーである．そして，製品混合計画を利用する組み立て型の産業に対して，連続なパイプラインでイメージされる装置型産業の代表である.

われわれの日常生活と関係のあるものとしては，食べ物に関係した利用であろう.

- 種々の飼料を混ぜ合わせ一定の栄養価を持つ配合飼料を作ることが考えられる.日本では，酪農や養鶏業で広く利用されている．日本に初めて LINDO 製品を紹介したところ，宮崎の養鶏業者が購入したことが思い出される.また全農では，酪農家の配合飼料などで配合計画を利用しているようだ.

それでは，我々人間の食事の調理に使えるだろうか？例えば，病院の患者にだす食事は栄養計算を行い必要な栄養素を含んだ食事を作る必要がある．そこで旬の素材の中から栄養価が高くお値ごろな食材を選ぶことである．分かってはいたが，管理栄養士の卵の娘に言うと，食材は調理法によって栄養素が変わるので難しいのではと一蹴された．実際，数理

計画法の開発者のダンティック先生は，家庭料理への応用を試みたが，奥方から一蹴されたという話をどこかで聞いたことがある．なぜ問題になるかといえば，

・経済的に問題が小さすぎて，手間をかけて数理計画法で解決しても効果的でない，いわゆる「奥さんの感ピュータ」のほうが優れているわけだ．このように，適用分野の適・不適を考える癖をつけることも「問題解決能力」にとって重要だ．

・人間は，牛や鶏と異なり，単に栄養素を満たすだけでなく美味しさや目新しさがより重要になる.同じようなものを続けて出すと患者から当然クレームがくることは想像できる．

4.2　ある製鉄会社の配合問題

ここでは 20 年ほど前に，ある鉄鋼会社から実際に相談を受けた現実の問題を考える．

この会社は，大手鉄鋼会社の 2 次協力会社のようだ．それまで，最終製品の決められた制約条件に合うよう原材料を経験と勘に頼って配合していた．そして，数理計画法の存在を知り，それを用いれば使用原材料費を最小化できることを知り，電話と FAX で相談にみえた．

その時感じたことは，日本の鉄鋼会社は数理計画法の大ユーザーなのに，子会社や孫会社までその技術がいきわたっていないことの驚きである．

この鉄鋼メーカーでは，従来 LP を使わないで勘に頼って生産計画をたてていた．

そのひとつが，**表 4・1** に示す 11 個の原材料を用いて，ある製品を作る場合である．

この原材料を用いて最小の費用で，最終製品に含まれる銅をはじめとする 6 種類の成分量が，表の上下限値の範囲に入るような原材料の配合比率を決めたい．下限と上限が逆のほうがよいが，もらった資料のとおりにしてある．

銅とかシリコンが出てきて，私に関係ないと思っているそこのお嬢さん，この表は栄養成分表とそっくりでしょう．原材料は食材，含有成分は食材に含まれる栄養素に置き換えて考えれば，献立の問題になります．

実際に利用しなくても，なぜ利用できないかを考えることは，自分の職業使命を改めて見つめなおすことになると思いますが.....

表 4・1　　11 個の原材料を用いて，ある製品を作る

	含有成分	Ｃｕ	Ｓｉ	Ｆｅ	Ｚｎ	Ｍｎ	Ｍｇ
	下限	1.8	10.8	0.88	1.6	0	0.34
	上限	2.2	11.2	0.9	1.8	0.3	0.35

原材料	単価（千円）						
X１	275	1.4	3.3	0.7	1.5	0.2	0.8
X２	275	2.5	8	0.8	4.5	0.2	0.3
X３	285	2.5	7.7	0.9	0.9	0.18	0.19
X４	285	2.5	9.5	0.9	0.9	0.18	0.09
X５	185	2.5	9.3	0.95	0.93	0.18	0.09
X６	235	2.3	8.4	0.8	3	0.21	1.4
X７	235	2.5	9	0.9	0	0	0
X８	260	0.2	0.2	0.5	0	0.5	0
X９	290	98	0	0	0	0	0
X10	340	0	97	0.5	0	0	0
X11	255	4	0.5	0.5	0.1	0.5	0.5

これを，LP でどう解決すればよいだろう．

4.4　配合計画を LP でモデル化する

数理計画法モデルの作成は，次の 3 段階（ABC ステップ）で行えばよい．

（1）第 1 段階：決定変数を決める（What'sBest では Adjustable　Cell と読んでいる）

まず，第 1 段階で決定変数(修正可能変数)を決めよう．原材料 1 から原材料 11 までの最終製品における配合比率を，X1 から X11 の 11 個の決定変数で示す．

（2）第 2 段階：決定変数で目的関数を決める（Best セルを決める）

第 2 段階では，決定変数で目的関数を決める．X1 の単価が 275 円で，X11 の単価が 255 円だ．最終製品の原材料費は次の式になる．

```
275*X1+275*X2+...+255*X11
```

この値を，以下の制約条件を満たす範囲内で，最小にしたい．

（3）第 3 段階：決定変数で制約式を決める（Constraints を決める）

第 3 段階では，決定変数で制約式を決める．

これらの配合比率の間には，次の関係がある．

```
X1+X2+X3+...+X11=1;
```

あるいは，百分率で考えて次のようにしても良い．

```
X1+X2+X3+...+X11=100;
```

さらに実際に 150 トン作りたいのであれば次のようになる．

X1+X2+X3+...+X11=150;

これが，第 3 段階で考える，最初の等式制約になる．

次に考える制約式は，最終製品に含まれる銅が，1.8（単位不明なので仮に Kg とする）から 2.2Kg の間になければいけないという制約だ．原材料 X1 には 1.4 の銅が含まれており，X2 には 2.5 の銅が含まれている．実際に用いる比率は決まっていないが，これらの原材料を表す決定変数を用いれば，最終製品に含まれる銅の重量は次の式になる．

1.4*X1+2.5*X2+...+4.0*X11

そして，次の 2 つの不等式制約が銅に関する制約になる．

1.4*X1+2.5*X2+...+4.0*X11>=1.8;

1.4*X1+2.5*X2+...+4.0*X11<=2.2;

もちろん，このままでも良いが，次のように新しい決定変数 CU を導入しても良い．

CU=1.4*X1+2.5*X2+...+4.0*X11　なので

1.4*X1+2.5*X2+...+4.0*X11 - CU=0;

この場合，上の制約条件は次のようになる．

CU>=1.8;

CU<=2.2;

このような定式化によってモデルはすっきりした．しかし，決定変数が 1 個と等式制約が 1 個増え計算時間が増えることになる．

同じようにして，シリコンからマグネシウムまでの制約式を表そう．

最後にこの会社では，親会社との関係で政策上，原材料 X3 を 35%使用する必要がある．これは，親会社から供給されていて，使わなければいけない原材料である．次の等式制約で表される．

X3=0.35;

以上が制約条件だ．なにも，難しいことはないだろう．

(4) 非負条件

おっと，大切なことを忘れていた．つぎの決定変数の非負条件だ．

X1>=0:

X2>=0;

・

・

・

X11>=0;

数理計画法では，決定変数の非負条件は暗黙の了解事項である．なぜなら，決定変数は，資源や経済活動を表し，これらが負になることはないからだ．

ただし，株の売りと買い，輸入代金と輸出代金などを一つの変数として考えれば，正にも負にもなる．このような変数を自由変数という．

数理計画法のソフトには，決定変数は非負しか扱えないものがある．この場合は，変数 X を自由変数にするには，新しい非負の決定変数 Y1 と Y2 でもって，次のような等式制約におきかえればよい．ただし，LINGO では，X を「Free」コマンドで指定すれば良いので置き換えは不要である．

X=Y1 - Y2

ただし，Y1，X2 は非負

X が正の時は Y1>0 で Y2=0，

X が負の時は Y1=0 で X2>0 になる．

問）なぜ，上のようにすれば自由変数になるのだろうか?

4.4 LINGO でのモデル化

さて，上で述べたことを LINGO でモデル化すると，次のようになる．モデル化といっても，ほぼ式の通り，自然な表記で表されるので，説明の必要もないほどだ．

```
MIN=275*X1+275*X2+285*X3+285*X4+185*X5+235*X6+235*X7+260*X8+290*X9+340*X10+255*X
11;
[CU2] 1.4*X1+2.5*X2+2.5*X3+2.5*X4+2.5*X5+2.3*X6+2.5*X7+0.2*X8+98*X9+4*X11-CU=0;
[SI3] 3.3*X1+8*X2+7.7*X3+9.5*X4+9.3*X5+8.4*X6+9*X7+0.2*X8+97*X10+0.5*X11-SI=0;
[FE4]
0.7*X1+0.8*X2+0.9*X3+0.9*X4+0.95*X5+0.8*X6+0.9*X7+0.5*X8+0.5*X10+0.5*X11-FE=0;
[ZN5] 1.5*X1+4.5*X2+0.9*X3+0.9*X4+0.93*X5+3*X6+0.1*X11-ZN=0;
[MN6] 0.2*X1+0.2*X2+0.18*X3+0.18*X4+0.18*X5+0.21*X6+0.5*X10+0.5*X11-MN=0;
[MG7] 0.8*X1+0.3*X2+0.19*X3+0.09*X4+0.09*X5+1.4*X6+0.5*X11-MG=0;
[CU8] CU>1.8;
[CU9] CU<2.2;
[SI10] SI>10.8;
[SI11] SI<11.2;
[FE12] FE>0.88;
[FE13] FE<0.9;
```

[ZN14] ZN>1.6;

[ZN15] ZN<1.8;

[MN16] MN>0;

[MN17] MN<0.3;

[MG18] MG>0.34;

[MG19] MG<0.35;

[OTH1] X1+X2+X3+X4+X5+X6+X7+X8+X9+X10+X11=1;

[OTH2] X3=0.35;

END

目的関数と制約式の順は不同であるが,最初に目的関数を記述する方が良いだろう.MIN=の後に，目的関数を入力する．目的関数や制約式は，複数の行にまたがっていても良い．

この後の行に，制約式を入力すればよい．制約式名は[CU1]のような英数字でも良いし，省いても良い．省けば，目的関数を 1 番目として，真の制約条件を 2 番から数えたものが自動的にとられる．

決定変数の非負条件は指定する必要がない．

モデルを作成し終った後に，Solve コマンドで計算が開始する．この他，作成したモデルに対し，モデルの作成と編集，外部媒体への格納と検索，診断，実行，解の表示と出力，ヘルプ情報の提供などの機能があるが，多くの場合不要である．

ずいぶん簡単だろう．

読者は，添付の「4 配合 1.lg4」をダブルクリックし，実行ボタンを押せば解が求まる．

4.5　さて実行してみると

（1）エラーの発生

[Solve]コマンドでモデルを解くと，**図 4・1 のエラーメッセージの**出力がえられる．実行可能解がないことを示している.「実行可能解なし」とは，3 章で紹介したように制約式が厳しすぎて，それらを同時に満たす値がないことを示す．例えば，「x>=10 で x <=3」を同時に満たす領域がないことを表している．しかし，上のモデルのように 20 個も制約式があるとどこに問題があるか分からない．

［OK］を押すと**図 4・2** の LINGO の出力になる．計算結果の途中状態が示されている．1 行目は実行可能解がないことを示す（No feasible solution found）．そして，2 行目で 18 ステップの繰り返し計算で停止したことを示す.これは後で紹介する単体法で 18 回計算した後で停止したことを表す．これぐらいのモデルでも，手計算すると時間がかかる．これ

以降の解は，計算を打ち切った時点の状態を表示している．

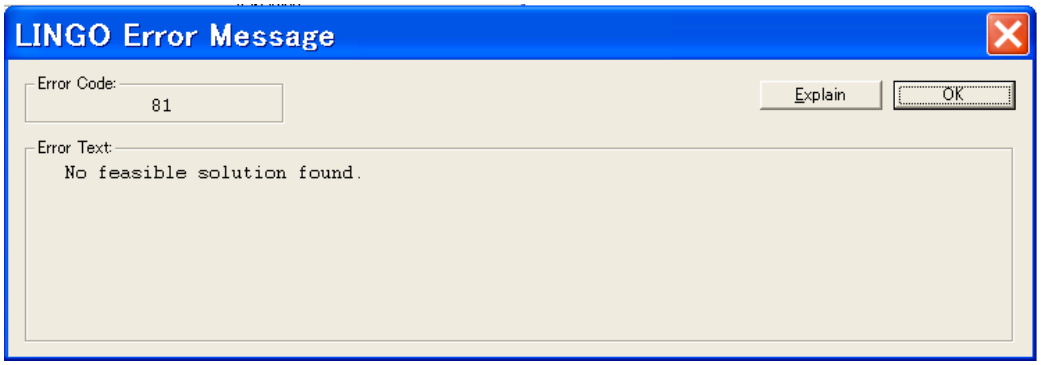

図 4・1　エラーの発生

```
No feasible solution found.

Total solver iterations:                           18

                     Variable           Value        Reduced Cost

                           X1        0.1771697          -0.2264977E-04

                           X2        0.1447350           0.1525879E-04

                           X3        0.3500000           0.000000

                           X4         0.000000           0.1065168E+12

                           X5        0.2423275          -0.2441406E-03

                           X6        0.4752451E-01       0.4882812E-03

                           X7         0.000000           0.1868839E+12

                           X8         0.000000           0.9561443E+11

                           X9         0.000000           0.4045347E+14

                          X10        0.3824339E-01       0.000000

                          X11         0.000000           0.1593421E+13

                           CU         2.200000           0.000000

                           SI         10.80000           0.000000

                           FE        0.8421591           0.000000

                           ZN         1.600000           0.000000

                           MN        0.2001017           0.000000

                           MG        0.3400000           0.000000

                          Row    Slack or Surplus      Dual Price

                            1       0.3784086E-01       -1.000000

                          CU2         0.000000           0.4020227E+12
```

SI3	0.000000	-0.1571270E+09
FE4	0.000000	-0.2080000E+13
ZN5	0.000000	-0.8494174E+11
MN6	0.000000	0.000000
MG7	0.000000	-0.4267747E+11
CU8	0.4000000	0.000000
CU9	0.000000	0.4020227E+12
SI10	0.000000	-0.1571270E+09
SI11	0.4000000	0.000000
FE12	-0.3784086E-01	-0.2080000E+13
FE13	0.5784086E-01	0.000000
ZN14	0.000000	-0.8494174E+11
ZN15	0.2000000	0.000000
MN16	0.2001017	0.000000
MN17	0.9989829E-01	0.000000
MG18	0.000000	-0.4267747E+11
MG19	0.1000000E-01	0.000000
OTH1	0.000000	0.1055241E+13
OTH2	0.000000	-0.1025319E+12

図 4・2　計算結果の状態

(2)　なぜエラーか？

ここで解がある試験問題にたけた学校秀才は，びっくりする．

この会社の人は従来から勘と経験で操業しているので，解がないのは LINDO(すでに開発販売を停止している，LINDO Systems Inc.の最初の数理計画法ソルバー)が間違っているのではないかといってきた．分析結果は，正しいのである．

なぜこのようなことになるのだろう．残念だが，先方と詳しく議論しなかったことが悔やまれる．一般的に考えられることは，次のような点である．

・ モデル作成に際し，定式化の誤り．
・ 表に示された値が，だいたいの値であり，厳密な値でない．
・ 真の問題を似せて問い合わせのために作った表である．

定式化の誤りは，入力ミスのほか，X が 5 以上か 3 以下というような背反条件がある．この場合，次の 2 つの条件式をモデルに同時に入れてはいけない．

X>=5;

X<=3;

注）ただし，生産する場合は 100 単位以上作るような場合，生産しない（ｘ=0)，生産する（ｘ>=100）という 2 社択一問題になる．このような二者択一は，IP の問題になる．例えば，段取り費用が発生する場合に良く用いられる．

実行可能解がないということは，このように制約条件のどこかに間違いがあるということにある．しかし，これ位のモデルでもそれを発見することは神業である．

とにかくこの問題は，重要だ．なぜなら，数理計画法プログラムの試金石になる．このような小さな問題でも実行可能解がない場合に対応方法を示さないプログラムが多いのには驚きだ．実際使う上での配慮が足りないものが多いということだ．

(3)　どこが問題化？

図 4･2 の「Slack or Surplus」をみてみよう．不等号の左辺に制約式，右辺に定数項がくることを基本に話をする．不等号が「＜＝」制約とすれば，「右辺定数項－制約式の値」は正になる．プロダクト・ミックスの場合，「在庫－使用量」を表し，0 か正の値をとる．正の場合，在庫が余る状態であり，余裕（Slack）と呼ばれる．一方，制約式が「>=」の場合，「制約式の値－右辺定数項」で下限値からの超過分（Surplus）を表している．いずれにしても，非負の値をとる．

しかし，鉄（Fe）の下限制約が負になっている．「Value」をみると Fe=0.842 であり，これは鉄（Fe）の下限の値 0.88 より 0.038 だけ小さい．一方，上限制約は 0.9 でありスラックは 0.058＝0.9 - 0.842 である．

すなわち，鉄（Fe）の下限の値 0.88 を 0.842 より小さくすればよい．

(4)　どうすれば実行可能解がもとまるか？

そこで，〔FE12〕 の 0.88 の下限値を思い切って「0」にしても良いが，例えば「0.84」に変更して，実行可能領域を広げてもう一度 [Solve]を実行してみよう．これによって，出力の最初に「Global optimal solution found.」の表示があり，**図 4･3** の最適解(Status が Optimal)253.47 が 7 ステップ（Iteration）で求まったことが分かる．

```
  Global optimal solution found.
  Objective value:                              253.4700
  Total solver iterations:                             7
                       Variable           Value        Reduced Cost
                             X1       0.6340993E-01         0.000000
                             X2       0.1371361             0.000000
```

X3	0.3500000	0.000000
X4	0.000000	190.7464
X5	0.2490194	0.000000
X6	0.1137282	0.000000
X7	0.000000	230.6026
X8	0.4631258E-01	0.000000
X9	0.000000	41633.85
X10	0.4039380E-01	0.000000
X11	0.000000	1557.021
CU	2.200000	0.000000
SI	11.20000	0.000000
FE	0.8400000	0.000000
ZN	1.600000	0.000000
MN	0.1920125	0.000000
MG	0.3400000	0.000000
Row	Slack or Surplus	Dual Price
1	253.4700	-1.000000
CU2	0.000000	416.4190
SI3	0.000000	0.3392355E-01
FE4	0.000000	-1756.149
ZN5	0.000000	-97.73759
MN6	0.000000	0.000000
MG7	0.000000	-21.21468
CU8	0.4000000	0.000000
CU9	0.000000	416.4190
SI10	0.4000000	0.000000
SI11	0.000000	0.3392355E-01
FE12	0.000000	-1756.149
FE13	0.6000000E-01	0.000000
ZN14	0.000000	-97.73759
ZN15	0.2000000	0.000000
MN16	0.1920125	0.000000

MN17	0.1079875	0.000000
MG18	0.000000	-21.21468
MG19	0.1000000E-01	0.000000
OTH1	0.000000	534.7840
OTH2	0.000000	-188.5638

図 4･3　　目的関数と決定変数の値と減少費用を表す最終結果

4.6　出力結果の解釈

（１）目的関数と決定変数の値と減少費用

図 4･3 は，目的関数と決定変数の値と減少費用を表す最終結果である．最適解は 253.47 で最小化問題として求まったことが「OBJECTIVE VALUE」から分る．

VARIABLE の下に決定変数が表示される．VALUE の下の値は，最適化計算でもとまった配合比率である．原材料 X1 は 0.063410, X2 は 0.137136, X3 は 0.350000 だけ使用している．この場合，「減少費用（REDUCED COST)」は必ず 0 になる．

原材料 X4 は 0 である．もし強制的に使用すると，最適解でなくなる．すなわち，原材料費は高騰する．これを表わすのが「減少費用」である．X4 を 1 単位使ったとすれば，190.746414 だけ原材料費が増えることを示している．

原材料 X9 を 1 単位使用すると，原材料費は 41633.843750 も高くなる事がわかる．すなわち，この原材料を用いる事はばかげている．

X4, X7, X9, X11 の 4 個の原材料が 0 で使われていないことが分る．すなわち，これらは非基底変数である．逆に X1, X2, X3, X5, X6, X8, X10 の 7 個の原材料が使われている．このように 0 でない最適解に選ばれた決定変数を基底変数といっている．

(2)　制約式と双対価格

制約式の［CU8］は「2.2>=1.8」を満たしておりサープラスは 0.4 である．すなわち，下限値の 1.8 より 0.4 大きいことを表す．双対価格は 0 になる．［CU9］は「2.2<=2.2」になりスラックは 0 になる．銅の上限制約を，もし 1 だけ緩めて 3.2 にできれば，原材料費が 416.419 だけ減る（改善される）ことが双対価格から分かる．

次に，［FE12］と［FE13］を見てみよう．これらは，「FE＞＝0.84」と「FE＜＝0.9」を表している．FE の値は 0.84 なので下限制約式では，「FE(0.84)＞＝0.84」と等式条件になる．このような場合，0.84 を 1 単位緩めれば（1 少なくする），目的関数の値は改善される．しかし，下限制約の場合は緩めないで，上限制約と同じく下限値を 1 増やす．すなわち制約を強め，実行領域を狭めるので，最適解は悪くなる．そこで双対価格は，－1756.149 のよ

うに負で表す．読者は，下限値を 1 だけ小さくすればこの値だけ目的関数は改善されると解釈すればよいだろう．ただし，0.84 を 1 単位緩めれば（1 少なくする）－0.16 になる．双対価格や減少費用は 1 単位あたりに換算して表している．今回の場合，例えば 0.1 小さくすると 1756.149 の 1 割の 175.6 だけ改善されると考えるべきだ．

すなわち,「減少費用」の値は決定変数で 0 になったものの一つを，無理やり 1 単位使用すると，目的関数がどれだけ悪くなるかを表わしている．複数の決定変数を同時に変更した場合にはどうなるかは分からない．

一方，制約条件の「双対価格」は，スラックやサープラスが 0 になったものの一つを，どれだけ緩めれば目的関数が改善されるかを示す．複数の制約条件を同時に変更した場合にはどうなるかは分からない．これらは，数学的に偏微分のような役割である．

(3)　丸める

原材料 X1 は 0.06341 の比率で作れば最適であることがわかる．しかし，製造技術上，例えば小数 3 桁でそろえる必要があったとしよう．これらの値を小数点第 3 位で四捨五入しても，**表 4･2** のように合計は 1 になるので，最終的にはこの丸めた値で生産することになる．これらの値を元のモデルに，X1=0.063 というように加えて計算すれば良い．すなわち，単に連立方程式の計算になる．

表 4･2　実際には丸めた値で生産する

VARIABLE	VALUE	丸め
X1	0.063410	0.063
X2	0.137136	0.137
X3	0.350000	0.350
X4	0.000000	0.000
X5	0.249019	0.249
X6	0.113728	0.114
X7	0.000000	0.000
X8	0.046313	0.046
X9	0.000000	0.000
X10	0.040394	0.040
X11	0.000000	0.000
合計	1.000000	1.000

4.7　親会社に幾ら請求するか

さて，ここで次の問題を考えてみよう．

問）この会社では親会社から原材料 X3 を 35%使用することを要求されている．一体これに対して，親会社から幾らの対価を受け取るのが妥当であろうか．これに対する解は，どのようにすればよいだろうか．

答）　それは，単に X3 を **35%**使用するという制約を取り払って（![OTH2] X3=0.35;とこの制約式をコメントにすればよい）**決定変数**にして，再計算してみればよい．**図 4･4** が出力である．

目的関数の値は，214.785 である．図 4･3 では，253.47 であった．これとの差の 38.685 が，結局親会社によって拘束されることで発生する無駄な費用である．この分析結果でもって，親会社と支払いあるいは受け取り金額を冷静に交渉すればいいであろう．

これは減少費用からも計算できる．X3 は 0 であるが，これを 1 単位増やすと目的関数は 102.89575 だけ減少する．ただし，0.35 だけ増やすので，102.89575*0.35＝36.013512 になる．厳密に一致しないのは，減少費用は 1 単位に換算した場合の値を表すが，実際どこまでが有効か分からない点だ．

```
  Global optimal solution found.
  Objective value:                              214.7850
  Total solver iterations:                             7
                       Variable           Value        Reduced Cost
                             X1        0.000000            36.02483
                             X2       0.1071510            0.000000
                             X3        0.000000            102.8957
                             X4        0.000000            100.6345
                             X5       0.5924272            0.000000
                             X6       0.1889545            0.000000
                             X7        0.000000            73.57996
                             X8    0.8229685E-01           0.000000
                             X9        0.000000            4495.885
                            X10    0.2917054E-01           0.000000
                            X11        0.000000            170.4098
                             CU        2.200000            0.000000
                             SI        10.80000            0.000000
                             FE       0.8554239            0.000000
```

ZN	1.600000	0.000000
MN	0.1823328	0.000000
MG	0.3500000	0.000000
Row	Slack or Surplus	Dual Price
1	214.7850	-1.000000
CU2	0.000000	45.66194
SI3	0.000000	-0.7321034
FE4	0.000000	0.000000
ZN5	0.000000	-26.03164
MN6	0.000000	0.000000
MG7	0.000000	9.434340
CU8	0.4000000	0.000000
CU9	0.000000	45.66194
SI10	0.000000	-0.7321034
SI11	0.4000000	0.000000
FE12	0.1542386E-01	0.000000
FE13	0.4457614E-01	0.000000
ZN14	0.000000	-26.03164
ZN15	0.2000000	0.000000
MN16	0.1823328	0.000000
MN17	0.1176672	0.000000
MG18	0.1000000E-01	0.000000
MG19	0.000000	9.434340
OTH1	0.000000	-268.9860

図 4･4　親会社に幾ら請求すべきか

4.8　汎用の配合モデルを作成する

(1)　スカラー表記よ，さようなら

スカラー表記による LINGO のモデル表記は，勉強する上で分かりやすい．しかし，現実の問題に適用しようとする場合，決定変数や制約式が大きなものも扱う必要が出てくる．この場合，スカラー表記であるとモデル作成に多大な工数がかかる．この場合，これまでは制約式や決定変数のモデルのサイズを表すパラメータと条件式の係数を読み込んで，モ

デルをプログラムで生成してきた.

しかし，What'sBest!はExcelで式のコピー機能を使えば，同じ構造を持った制約式を作成することは容易である．私の研究では，2万件の制約式をコピーしたモデルをここ10年間分析してきた．一方，LINGOでは集合表記を用いれば，大規模なモデルも係数をモデルから分離し，汎用化できる．

(2)　集合によるモデル記述

次は，LINGOによる集合を用いた定式化である．LINGOのスカラーモデルでは大規模モデルの作成が大変である．そこで，次の集合モデルで，最終成分（CONTENT）と原材料（RAW）の原始集合と，それらから作られる2次元の集合MATRIXを定義する．DATA節でこれらの値をモデルから分離して与えることができるので，モデルから係数が分離される．

SETS節（SETS:で始まりENDSETSで終わる）では，原始集合CONTENTを定義する．ULとLLは，上下限値の入った要素数6の1次元配列である．従来の配列概念と異なるのは，集合は同じ要素数を持つ配列を統一し，それらの配列を集合のインデックスで繰り返し処理できる点である．RAWは原材料に対応する集合で，11個の原材料の価格（PRICE）と配合比率（PRODUCT）の1次元配列を持っている．

MATRIXは，RAWとCONTENTから作られる2次元の派生集合である．SEIBUNは11行8列の成分を入れる2次元配列である．これらは，DATA節で，各配列に割り当て文で値が割り当てられる．配合比率PRODUCTは最適計算で決まる．

@SUM(RAW(i):) は，集合RAWの要素数11だけPRICE(i)*PRODUCT(i)の合計すなわち $\Sigma_{i=1\cdots11}$ PRICE(i)*PRODUCT(i)を計算する．これで，原材料費が定義される．

「@FOR(CONTENT(j): @SUM(RAW(i): SEIBUN(i,j)*PRODUCT(i))<=UL(j));」は，集合CONTENTの要素数と同じ8個の上限制約（<=）を定義する．すなわち，同じ構造であれば，1万個の制約でもこれだけで十分だ．次の「@FOR(CONTENT(j): @SUM(RAW(i): SEIBUN(i,j)*PRODUCT(i))>=LL(j));」は，8個の下限制約を表す．

```
SETS:
 CONTENT: UL,LL;
 RAW: PRICE,PRODUCT;
 MATRIX(RAW,CONTENT):SEIBUN;
ENDSETS
DATA:
 UL=2.2 11.2 0.9 1.8 0.3 0.35;
 LL=1.8 10.8 0.88 1.6 0 0.34;
```

```
 PRICE=275 275 285 285 185 235 235 260 290 340 255;
 SEIBUN=
1.4       3.3    0.7    1.5    0.2    0.8
2.5       8      0.8    4.5    0.2    0.3
2.5       7.7    0.9    0.9    0.18   0.19
2.5       9.5    0.9    0.9    0.18   0.09
2.5       9.3    0.95   0.93   0.18   0.09
2.3       8.4    0.8    3      0.21   1.4
2.5       9      0.9    0      0      0
0.2       0.2    0.5    0      0.5    0
98 0      0      0      0      0
0  97     0.5    0      0      0
4  0.5    0.5    0.1    0.5    0.5;
ENDDATA
MIN=@SUM( RAW(i):PRICE(i)*PRODUCT(i));
@FOR( CONTENT(j): @SUM(RAW(i): SEIBUN(i,j)*PRODUCT(i))<=UL(j));
@FOR( CONTENT(j): @SUM(RAW(i): SEIBUN(i,j)*PRODUCT(i))>=LL(j));
@SUM(RAW(i): PRODUCT(i))=1;
```

(3)　完全な汎用モデル

先ほどのモデルでは，モデル式の係数を割り当て文で配列に直接定義していた．しかし図4・5のように，Excel上に問題を表すデータを記述しよう．そして，成分を表すC5:H15を選んで，［挿入］→［名前］→［定義］でセル名をSEIBUNと指定する．同様に価格のセル名をPRICE,上限値をUL,下限値をLLとセル名を与えてやる．これによって，パラメータを「SEIBUN=」で直接割り当てる必要がなくなり，「SEIBUN=@OLE(　)」でもってExcelから入力できる．読者は，「4配合4.xls」をダブルクリックすればこのファイルが表示される．しかし，その前に一度自分で作ってみよう．

このようにすれば，Excel上に解きたい問題を表すデータを作成するだけで，モデルのサイズに関係なく汎用的な配合モデルが完成する．すなわち，勉強したことがすぐに企業の大きな問題にプログラム開発することなく作成し，実行できる．

	A	B	C	D	E	F	G	H	I	J
1		含有成分	Ｃｕ	Ｓｉ	Ｆｅ	Ｚｎ	Ｍｎ	Ｍｇ		
2		下限	1.8	10.8	0.88	1.6	0	0.34		1
3		上限	2.2	11.2	0.9	1.8	0.3	0.35		1
4	材料	単価（円）								0
5	×１	275	1.4	3.3	0.7	1.5	0.2	0.8		0
6	×２	275	2.5	8	0.8	4.5	0.2	0.3		0
7	×３	285	2.5	7.7	0.9	0.9	0.18	0.19		0
8	×４	285	2.5	9.5	0.9	0.9	0.18	0.09		0
9	×５	185	2.5	9.3	0.95	0.93	0.18	0.09		0
10	×６	235	2.3	8.4	0.8	3	0.21	1.4		0
11	×７	235	2.5	9	0.9	0	0	0		0
12	×８	260	0.2	0.2	0.5	0	0.5	0		0
13	×９	290	98	0	0	0	0	0		0
14	×１０	340	0	97	0.5	0	0	0		0
15	×１１	255	4	0.5	0.5	0.1	0.5	0.5		0
16			0	0	0	0	0	0		0
17			<=	<=	<=	<=	<=	<=		Not =
18			Not >=	Not >=	Not >=	Not >=	=>=	Not >=		
19										

図４・５　Excel上の問題（4配合4.xls）

データを「＠OLE」関数で，Excelからデータを呼び込めば，企業では，Excelデータのみを更新すればよい．授業で習ったことを，学生が企業に入って実際の配合問題を解決できる．次が完全な汎用のモデルである．

```
SETS:
 CONTENT: UL,LL;
 RAW: PRICE,PRODUCT;
 MATRIX(RAW,CONTENT):SEIBUN;
ENDSETS
DATA:
 UL=@OLE();
 LL=@OLE();
 PRICE=@OLE();
 SEIBUN=@OLE();
ENDDATA
MIN=@SUM( RAW(i):PRICE(i)*PRODUCT(i));
@FOR( CONTENT(j): @SUM(RAW(i): SEIBUN(i,j)*PRODUCT(i))<=UL(j));
@FOR( CONTENT(j): @SUM(RAW(i): SEIBUN(i,j)*PRODUCT(i))>=LL(j));
@SUM(RAW(i): PRODUCT(i))=1;
```

4.9　読者へ

プロダクト・ミックスと異なり，配合問題は，石油，鉄，配合飼料で実際好く使われている．しかし，なぜか製鉄や石油以外の化学品会社での利用は進んでいない．また，金型は結構原材料費が高いらしい．代替となる原材料の成分表さえあれば，日本の中小の金型メーカーも助かるのにと思う．

原材料高の今こそ，ちょっと数理計画法を用いるだけで，あと数%の利益改善が図れるのに．残念だ,.....

5　巡回セールスマン問題

5.1　世紀の難問

巡回セールスマン問題（Traveling Salesmen Problem, TSP）は，問題の構造はいたって簡単である．しかし，大きな問題を解く場合，組み合わせの爆発により計算時間の壁にぶち当たる難問である．営業マンが指定された都市を訪問し製品の販売を行う．このとき，本拠地から出発し，一筆描きの要領で，各都市を 1 度だけ最短の距離(時間)で巡回し本拠地へ戻ってくることが望まれている．この問題を考える場合，日本の都道府県所在地を巡回することが考えられる．ただ，経度と緯度の位置情報から計算したユークリッド距離を用いるとデータ作成が一番容易である．しかし，4 つの島に散らばっているので，単なる直線距離ではうそっぽい．その点，アメリカの主要都市であれば，地続きであり，日本より矩形に近いので例として取り上げる．

そこで，Linus のテキストで取り上げている，「風とともに去りぬ」の舞台であるアトランタ（ATL），LINDO Systems Inc.のあるシカゴ（CHI），シンシナティ（CIN），石油と航空機産業の町ヒューストン（HOU），西海岸の拠点ロスアンゼルス（LA），フランス系カナダ人の中心モントリオール（MON），世界の金融やビジネスの中心ニューヨーク（NY），自由の鐘があるフィラデルフィア（PHI），かつての鉄鋼業の中心のピッツバーグ（PIT），セントルイス（STL），軍港と観光のサンディエゴ（SD），霧の町サンフランシスコ（SF）の 12 都市である．

表 5.1 はこれらの都市間の距離を表す 12 行 12 列の距離行列である．

表 5.1　北米主要 12 都市の都市間の距離行列

	ATL	CHI	CIN	HOU	LA	MON	NY	PHI	PIT	STL	SD	SF
ATL	0	702	454	842	2396	1196	864	772	714	554	2363	2679
CHI	702	0	324	1093	2136	764	845	764	459	294	2184	2187
CIN	454	324	0	1137	2180	798	664	572	284	338	2228	2463
HOU	842	1093	1137	0	1616	1857	1706	1614	1421	799	1521	2021
LA	2396	2136	2180	1616	0	2900	2844	2752	2464	1842	95	405
MON	1196	764	798	1857	2900	0	396	424	514	1058	2948	2951
NY	864	845	664	1706	2844	396	0	92	386	1002	2892	3032
PHI	772	764	572	1614	2752	424	92	0	305	910	2800	2951
PIT	714	459	284	1421	2464	514	386	305	0	622	2512	2646
STL	554	294	338	799	1842	1058	1002	910	622	0	1890	2125

SD	2363	2184	2228	1521	95	2948	2892	2800	2512	1890	0	500
SF	2679	2187	2463	2021	405	2951	3032	2951	2646	2125	500	0

対角線上は 0 であり，対角線を挟んで対称になっている．場合によって，都市 i と都市 j の距離を d_{ij} で表すと，$d_{ij} \neq d_{ji}$ と非対称な問題も扱えるが，ここでは対称と考える．

5.2 TSP の定式化

(1) 割り当て問題

TSP の問題はいたって簡単である．都市 i から都市 j へ移動する場合，$Y_{ij}=1$ と表すことにする．目的関数は，次の式になる．

$$MIN=\Sigma d_{ij}*Y_{ij} \qquad i=1,\ldots, n,\ j=1,\ldots, n$$

そして，制約式は，都市 i から他の(n - 1)都市に行けるのは 1 都市だけであるので，次の制約になる．

$$\Sigma_{j=1,\ldots, n} Y_{ij} = 1 \qquad i=1,\ldots, n$$

一方，都市 j に他の(n - 1)都市からこれるのは 1 都市だけであるので次の制約になる．

$$\Sigma_{i=1,\ldots, n} Y_{ij} = 1 \qquad j=1,\ldots, n$$

ただし，上の式で自分の都市に行くことを禁止するため次の制約を付け加える．

$$Y_{kk}=0 \qquad k=1,\ldots, n$$

すなわち，次のようなモデルになる．

[_1] $MIN=\Sigma d_{ij}* Y_{ij} \qquad i=1,\ldots, n,\ j=1,\ldots, n$

[_2] $\Sigma_{j=1,\ldots, n} Y_{ij} = 1 \qquad i=1,\ldots, n$

[_3] $\Sigma_{i=1,\ldots, n} Y_{ij} = 1 \qquad j=1,\ldots, n$

[_4] $Y_{kk} =0 \qquad k=1,\ldots, n$

[_5]ただし，Y_{ij} は 0/1 の整数変数．

このモデルは，一般には「割り当て問題」として知られている整数計画法の代表的な問題である．今，m 個の倉庫から n 個の需要地への輸送を考える．この場合，m 行 n 列の距離や輸送費を表す表 5.1 のような行列を作成すればよい．そして，倉庫の在庫の範囲内で，1 箇所の需要地へ需要を満たす輸送を計画し，輸送費を最小化するような場合である．

モデルの定式化はずいぶん簡単であるが，Y_{ij} を 0/1 の整数変数としている点である．都市数を n とすれば，$(n^2 - n)$ だけの整数変数が必要となる．10 都市で 90 個の整数変数，20 都市で 380 個の整数変数であることが次の Speakeasy の計算で分かる．300 都市もあれば，何と 89700 個の整数変数になる．

```
:_N=10 20 30 50 70 100 200 300
```

```
:_n**2-n
  N**2-N (A 8 Component Array)
  90      380     870     2450    4830    9900    39800   89700
```

もし整数変数が単に 0/1 であっても，次の計算に示すとおり，10 都市で 10 の 27 乗の組み合わせ問題になる．20 都市と 30 都市で，10 の 114 乗，10 の 261 乗通りの場合になる．40 都市になると Speakeasy でオバーフローしてしまう．

```
:_N=10 20 30
:_n**2-n
  N**2-N (A 3 Component Array)
  90    380   870
:_LOG10(2**(n**2-n))
  LOG10(2**(N**2-N)) (A 3 Component Array)
          27.93   114.39   261.9
```

すなわち，整数計画法は「一休さんの頓知問題」よろしく「組み合わせの爆発」が起きる．

(2)　ナップザックも問題

0/1 の整数変数でなく，非負の一般整数変数となるともっと場合の数が増える．例えば次のモデル(5TSP1.lg4) は，制約式[_2]のもとで，目的関数[_1]を最大化する単純なモデルである（制約式に番号を振りたい場合は，鍵括弧の中で数字の前にアンダースコアをつけてください）．具体的には，このようなタイプのモデルは「ナップザック問題」といわれる．個人が旅行で持っていくナップザックに荷物を容量制限内で詰め込むことを考える．そのとき，荷物の重要度の総和を最大化したい．ナップザックに限らず，運輸に用いるコンテナや，CD に映像コンテンツなどを詰め込むような問題である．この問題をナップザック問題と解釈すれば，X_1 から X_6 の 6 個の製品があり，それらの重量は，12228 , 36679 , 36682, 48908, 61139, 73365 である．そして，1 個詰め込んだ場合の利益が 81, 221, 219, 317, 385, 413 である．このとき各製品は何個詰め込んでも良いとする．そこで一般整数変数として扱うが,一応0から99999までの間の一般整数変数に限定している．しかし，これでも探索空間は$(10^5)^6 = 10^{30}$になる．

```
[ _1] MAX =  81 * X_1 + 221 * X_2 + 219 * X_3 + 317 * X_4 + 385 * X_5 + 413 * X_6;
[ _2] 12228 * X_1 + 36679 * X_2 + 36682 * X_3 + 48908 * X_4 + 61139 * X_5 + 73365
* X_6 = 89716837;
 @GIN( X_1); @GIN( X_2); @GIN( X_3); @GIN( X_4); @GIN( X_5); @GIN( X_6);
```

```
@BND( 0, X_1, 99999); @BND( 0, X_2, 99999);
@BND( 0, X_3, 99999); @BND( 0, X_4, 99999);
@BND( 0, X_5, 99999); @BND( 0, X_6, 99999);
```

これを LINGO で解くと 1 秒以内で次の解が出力される.

```
Global optimal solution found.
Objective value:                              540564.0
Extended solver steps:                               0
Total solver iterations:                             0
                     Variable           Value        Reduced Cost
                          X_1        0.000000           -81.00000
                          X_2        2445.000           -221.0000
                          X_3        1.000000           -219.0000
                          X_4        0.000000           -317.0000
                          X_5        0.000000           -385.0000
                          X_6        0.000000           -413.0000
                          Row    Slack or Surplus      Dual Price
                           _1        540564.0            1.000000
                           _2        0.000000            0.000000
```

図5.1　　ナップザック問題の解

しかし，整数計画法の世界で世界最高速を謳うソフトは，少なくとも 2007 年以前の版で永遠に解けない問題といわれていた．結局整数計画法は，問題のタイプによって計算時間が異なり，ソフトによっても得意/不得意がある．それなのに，整数計画法の研究者の一部は，整数計画法の特定の問題でどれだけのサイズが解けたかどうかを「数理計画法ソフト全体の最優先課題」にしている．私は，「良いソフトは，素人から専門家までの全ての人が使いやすく，専門家が必要とする全ての機能を備えていることがより重要である」と考えている．一流の研究者による，現実問題への適用と普及を二の次にする傾向が，統計に比べて数理計画法が世間の理解を得ることを阻害したと考えている．

5.3　TSP の真の困難さ

TSP は，計算の困難さを物語る代表選手のように紹介されてきた．その理由としては，整数変数が多いことがすぐに指摘できる．しかし，割り当て問題は 0/1 の整数変数で定式できるが，0≦Y≦1 の制約を課して LP で解ける場合が多い．

TSP の真の困難さは上の割り当て問題で解くことができないことである．割り当て問題で解くと，幾つかのサブ・ツアー(部分旅行)に分かれてしまう．すなわち,「一筆書きの条件」が入っていない．このため，Linus のテキストを 2007 年 11 月に翻訳中割り当て問題を LP で解いてできたサブツアーを，人手でステップを踏んで切断する方法（サブツアーカット）が紹介されていた．例えば，3 都市が次のように巡回（1→4→8→1）していたとする．

Y(1,4)＋Y(4,8)＋Y(8,1)=3

このとき，次のステップで次のような制約を課す．

Y(1,4)＋Y(4,8)＋Y(8.1)<=2

これが SUBTOUR　CUT である．これによって 3 都市の巡回閉ループが切断され，他のサブツアーと融合することで，最終的に一つのツアーに成長することを期待している．

私は，正直言って面白いが，何度かマニュアルでこのステップを実施することが面倒だなと思った．しかし，2007 年 12 月中旬に LINGO のモデルの中の LOOPTSP というモデルを見てびっくりした．何と，人手で行わず SUBTOUR　CUT が自動的に行える．私は，自分で確かめもしないで,「LINGO10 の衝撃」というレポートを書いた．そして，知り合いの会津にある Spansion という富士通と米国の合弁企業の所長さんに「半導体設計のデータをもらえないか?」というお願いをした．しかし，2008 年の 1 月の中旬，ひょんなことで経済学部の共同研究室で同僚と話していて，東京と神奈川の不動産物件を一筆書きで結びたいということを聞いた．早速彼からデータをもらい分析を始めた．

最初 LOOPTSP(Ver. 1) で LP に指定して解くと，アメリカの都市ではうまくいくのに，日本のデータでは小さなサブツアーがたくさんでき,それが SUBTOUR　CUT で成長していかない．どうも等間隔メッシュの区間から代表物件を選んだため，ほぼ等距離になることが問題を難しくしているのでないかと考えている．そこで，IP で指定して解くと私の貧弱な PC では 30 都市ぐらいが限度である.そこで,Linus 教授と何度かメール交換し,LOOPTSP(Ver. 4) でようやくやりたい 220 都市の問題が解けるようになった．48180 整数変数である．結局，単に整数変数が多いだけでなく，TSP の真に困難な点はモデルの組みようで計算速度が異なること，同じ PC で計算しても内部メモリーの設定の違いで極端に異なってくることである．このような計算時間のかかる問題を解く場合は，クロック数の早い 64bit の PC を準備し，LINGO のチューニングをいってからすべきであろう．私のように，薄型軽量を基準に PC を選んだことは，TSP 研究の心構えができていなかったと反省せざるを得ない．

また LINGO では整数計画法の解法に分枝限定法という手法を用いている．この解法は，高速な枝だ刈りの機能があるが，SUBTOUR　CUT ではこれが使えない点も問題である．

米国では，Greedy 法つまり最適化でなく試行錯誤的なアプローチで数千都市，数万都市の問題に対応しているようだ．その場合は，LINDO API のようなＣのライブラリーで特注のプログラムを組んで，並列処理コンピュータなどで分析すべきであろう．

5.4 簡単な実行可能解

難問の TSP も，実行可能解は読者でも手で求めることができる．私も，一時 100 都市を越えられず，クラスター分析の結果を利用した満足解で妥協しようかと思った．しかし，この分野で「最近隣法」と呼ぶ手法がすでにあることを 2008 年 2 月 27 日に成蹊大学理工学部の同僚から教えてもらった．

それは，**表 5.1** で一番距離の近い都市を探すと，ｄ（NY,PHI）=92 である．これで，ニューヨーク（NY）を出発都市として，次にフィラデルフィア（PHI）に行くことにする．次に PHI 行で NY 列は省いて一番距離の小さなピッツバーグ（PIT）を探す．これで，NY→PHI→PIT というツアーになる．その後同じような手順で PIT→CIN→CHI→STL→ATL→HOU→SD→LA→SF→MON→NY という一筆書きが完成する．

表 5.2 はツアーが行われた場合を 1 にした行列 Y である．各行と各列の合計が 1 であることを計算している．最右端の列は距離を表す．結局，総移動距離は 8063 になる．

表 5.2　最近隣法によるツアー

	ATL	CHI	CIN	HOU	LA	MON	NY	PHI	PIT	STL	SD	SF		8063
ATL	0	0	0	1	0	0	0	0	0	0	0	0	1	842
CHI	0	0	0	0	0	0	0	0	0	1	0	0	1	294
CIN	0	1	0	0	0	0	0	0	0	0	0	0	1	324
HOU	0	0	0	0	0	0	0	0	0	0	1	0	1	1521
LA	0	0	0	0	0	0	0	0	0	0	0	1	1	405
MON	0	0	0	0	0	0	1	0	0	0	0	0	1	396
NY	0	0	0	0	0	0	0	1	0	0	0	0	1	92
PHI	0	0	0	0	0	0	0	0	1	0	0	0	1	305
PIT	0	0	1	0	0	0	0	0	0	0	0	0	1	284
STL	1	0	0	0	0	0	0	0	0	0	0	0	1	554
SD	0	0	0	0	1	0	0	0	0	0	0	0	1	95
SF	0	0	0	0	0	1	0	0	0	0	0	0	1	2951
	1	1	1	1	1	1	1	1	1	1	1	1		

5.5　TSP の応用

TSP 問題は，巡回セールスマン問題と呼ばれているため，読者にとって大きな問題解決法になるとは考えないかもしれない．産業に応用したとしても，飲み物やタバコの自販機に商品を補充するルートセールスぐらいをイメージするかもしれない.また宅配便の場合，毎日経路が変わるし，顧客の指定時間の制約もあり単純に TSP を適用できないだろう．

しかし，これは製造現場で色々な応用が考えられる．

・回路を設計する場合，結線を行うロボットの移動時間を最小になるように設計したい．

・複数の作業場所と，複数の部品保管庫の間の往来を最適化したい．

・一つの機械で数種類のペンキを吹き付けている．この場合，違った色にかえると色の組み合わせで段取り費用や時間が異なってくる．このような場合，色をかえる順序を決める必要がある．

・私のように，共同研究で関東圏の不動産物件を一筆書きしたいという，思ってもみない要求にも対応できる．

私は，そのうちマウンティングマシンと呼ばれるものの最適化にチャレンジしたいと考えている．

5.6　汎用モデル 1 の紹介

先ほども触れたが，2008 年 2 月 27 日以前に LOOPTSP(Ver.4)で解決したい問題の目処がついたので，理工学部の同僚のところに整数計画法に関する情報をもらいに行った．そこで，「一筆書き」のアイデアがあることを教えてもらった．SUBTOUR　CUT の代わりに次の制約条件を入れる．U_iは都市 i が何番目に訪問するかの順位を表す．ただし，最初の出発都市はプログラムで決められる．

[__5]　$U_i - U_j + n*Y_{ij} <= (n-1)$　　i=2,..., n, j=2,..., n

これをモデルに組み込んだものが次のものである．モデルの意味は，すでに紹介してあるので，説明を省く．

```
MODEL:
SETS:
 city:U;
 LINK( city, city):   dist,    Y;
ENDSETS
DATA:
 city = @OLE( );
```

```
 dist = @OLE( );
ENDDATA
 S=@size(CITY);!U(1)=s;
 MIN =  @SUM( LINK(i,j): dist(i,j) * Y(i,j));
 @FOR( city( K):
   @SUM( city( I)| I #NE# K: Y( I, K))= 1;
   @SUM( city( J)| J #NE# K: Y( K, J))= 1;
 ! Cannot go to yourself;    Y( K, K) = 0; );
 @FOR(LINK(i,j):@BIN(y(i,j)); );
@FOR( CITY(M)| M #GE# 2: @FOR( CITY(N)|N #GE#2:
   @SUM( LINK( M,N): U(M)-U(N)+S*Y(M,N)) <= S-1;));
 !@SUM( LINK( M,N)|M #GE# 1 #AND# N#GE#1: U(M)-U(N)+S*Y(M,N)) <= S-1;
DATA:
@OLE( )=y;
ENDDATA
END
```

これを解くと，**図 5.2** の出力になる．**表 5.3** は行列 Y の結果である．最後の 2 都市の順序が 0 になっている．これは，最後の 12 番目の都市から都市 1 へ行くことを省いているためである．LINGO には剰余関数があるが，これを利用すると計算時間がかかる．一筆書きが完成すればよいので，このまま用いる．PIT(ピッツバーグ)が出発点で，次に PHI（フィラデルフィヤ）を訪問している．このあと，表 5.3 と比較して次の訪問都市順が分かる．PIT→PHI→NY→MON→SF→LA→SD→HOU→ATL→STL→CH→CIN→PIT と巡回し，総移動距離は 7577 で，私の遅い PC で 42 秒であった．前の最近距離法の 8063 と比べると(8063－7577)/7577=0.064141 になる．最近隣法は最適解より，6.4%余分な距離を移動したことがわかる．ただ，このモデルはコンパクトであるが LOOPTSP (Ver.1) より遅いことが分かった．

```
  Global optimal solution found.
  Objective value:                              7577.000
  Extended solver steps:                           3570
  Total solver iterations:                        52142
                       Variable           Value        Reduced Cost
                              S        12.00000            0.000000
```

U(ATL)	0.000000	0.000000
U(CHI)	5.000000	0.000000
U(CIN)	0.000000	0.000000
U(HOU)	10.00000	0.000000
U(LA)	8.000000	0.000000
U(MON)	4.000000	0.000000
U(NY)	3.000000	0.000000
U(PHI)	2.000000	0.000000
U(PIT)	1.000000	0.000000
U(STL)	6.000000	0.000000
U(SD)	9.000000	0.000000
U(SF)	7.000000	0.000000

図 5.2　　切断なしの方法

表 5.3　5TSP2.lg4 の行列 Y

ATL	CHI	CIN	HOU	LA	MON	NY	PHI	PIT	STL	SD	SF		7577
0	0	1	0	0	0	0	0	0	0	0	0	1	454
0	0	0	0	0	0	0	0	0	1	0	0	1	294
0	0	0	0	0	0	0	0	1	0	0	0	1	284
1	0	0	0	0	0	0	0	0	0	0	0	1	842
0	0	0	0	0	0	0	0	0	0	1	0	1	95
0	1	0	0	0	0	0	0	0	0	0	0	1	764
0	0	0	0	0	1	0	0	0	0	0	0	1	396
0	0	0	0	0	0	1	0	0	0	0	0	1	92
0	0	0	0	0	0	0	1	0	0	0	0	1	305
0	0	0	0	0	0	0	0	0	0	0	1	1	2125
0	0	0	1	0	0	0	0	0	0	0	0	1	1521
0	0	0	0	1	0	0	0	0	0	0	0	1	405
1	1	1	1	1	1	1	1	1	1	1	1		

5.7　汎用モデル 2 の紹介

次のモデルは，割り当て問題として LP で解いて，マニュアルで SUBTOUR　CUT を行うモデルである．距離行列を対角線から下側だけで示している．都市 i から j へ移動しようが

都市 j から i へ移動しようが方向性を考えていないことに特徴がある．このため，集合Y(I,j)の「Y(CITY, CITY)|&1 #GT# &2:COST, Y;」の定義の中で「|&1 #GT# &2」すなわち「i>=j」とした下三角行列だけに限定している．

「@SUM(CITY(I)|I #GE# 2: Y(I, 1)) = 2;」は，ATL を出発地とし，最初に訪問する都市と，最後の訪問都市から帰ってくるので，合計は 2 になる．

次の制約式は，出発都市以外で，出入りが 2 であることを制約している．

```
「@FOR( CITY( J)|J #GE# 2: @SUM(CITY(I)| I #GT# J:
     Y(I, J)) + @SUM(CITY(K)|K #LT# J: Y(J, K))=2);」
```

そして，次の制約「@FOR(ROUTE: Y <= 1);」で Y は 0 から 1 の間の実数すなわち LP モデルであることを指定している．

```
MODEL:
SETS:
CITY;
Y(CITY, CITY)|&1 #GT# &2:COST, Y;
ENDSETS
DATA:
 CITY=
  ATL  CHI  CIN HOU   LA  MON   NY  PHI  PIT  STL  SD  SF;
COST=
  702
  454  324
  842 1093 1137
 2396 2136 2180 1617
 1196  764  798 1857 2900
  864  845  664 1706 2844  396
  772  764  572 1614 2752  424   92
  714  459  284 1421 2464  514  386  305
  554  294  338  799 1842 1058 1002  910  622
 2363 2184 2228 1521   95 2948 2892 2800 2512 1890
 2679 2187 2463 2021  405 2951 3032 2951 2646 2125 500;
ENDDATA
MIN = @SUM( ROUTE: Y * COST);
```

```
@SUM( CITY( I)|I #GE# 2: Y(I, 1)) = 2;
@FOR( CITY( J)|J #GE# 2: @SUM(CITY(I)| I #GT# J:
  Y(I, J)) + @SUM(CITY(K)|K #LT# J: Y(J, K))=2);
@FOR( ROUTE: Y <= 1);
END
```

このモデルを実行すると，次の 3 つのサブツアーが存在する．LP であるので 1 秒以内である．ただし，最後まで追及していないので何回 SUBTOUR　CUT が必要か分からない．

PIT→PHI→NY→MON→PIT, SF→LA→SD→SF, NY→MON→PIT→PHI→NY

そして，例えば 2 番目を次のような条件を加えて切断する．

```
Y( SNF, LAX) + Y( SND, LAX) + Y( SNF, SND) <= 2;
```

これによって，この 3 都市に図 5.3 のように HOU が加わり，4 都市のサブツアーになる．そこで，また次の切断を加えることになる．

```
Y(SND, HOU) + Y(SNF, HOU) + Y(SNF, LAX) + Y(SNF, SND)<= 3;
```

以下，順次結果を見て切断していくことになる．

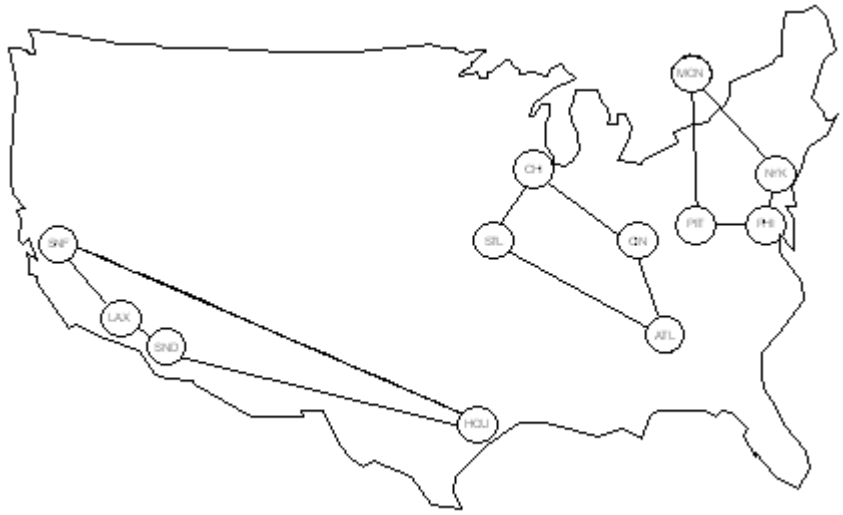

図 5.3　サブツアー

表 5.4　サブツアー

	ATL	CHI	CIN	HOU	LA	MON	NY	PHI	PIT	STL	SD	SF		5020
ATL	0	0	1	0	0	0	0	0	0	0	0	0	1	454
CHI	0	0	0	0	0	0	0	0	0	1	0	0	1	294
CIN	0	1	0	0	0	0	0	0	0	0	0	0	1	324
HOU	1	0	0	0	0	0	0	0	0	0	0	0	1	842

LA	0	0	0	0	0	0	0	0	0	0	0	1	1	405
MON	0	0	0	0	0	0	0	0	1	0	0	0	1	514
NY	0	0	0	0	0	1	0	0	0	0	0	0	1	396
PHI	0	0	0	0	0	0	1	0	0	0	0	0	1	92
PIT	0	0	0	0	0	0	0	1	0	0	0	0	1	305
STL	0	0	0	1	0	0	0	0	0	0	0	0	1	799
SD	0	0	0	0	1	0	0	0	0	0	0	0	1	95
SF	0	0	0	0	0	0	0	0	0	0	1	0	1	500
	1	1	1	1	1	1	1	1	1	1	1	1		

5.8 汎用モデル 3 の紹介

ここでは，LINGO10 のサンプルプログラム(5TSP3.lg4) で，2007 年 11 月に私が感激した LOOPTES (Ver.1) を紹介する．LOOPTES (Ver.4) は今のところ非公開にしている．

```
MODEL:
 ! (TSPCUT) Traveling Salesman Problem.  Find the shortest tour that
   visits each city exactly once. Subtour elimination method.
 ! Keywords: TSP, traveling sales person, routing, tour;
 SETS:
  CITY;
  LINK( CITY, CITY):
       DIST,  ! The distance matrix;
          Y;  ! Y( I, J) = 1 if link I, J is in tour;
  SUBTOUR: TOURSIZE;
  SXC(SUBTOUR,CITY): FLAG;
 ENDSETS
DATA:
  SUBTOUR = 1..20; ! Number subtour cuts we allow;
  CITY = @OLE( );
   ! Distance matrix for cities of National League of
      US baseball, circa 2004. It need not be symmetric;
  DIST =@OLE( );
ENDDATA
SUBMODEL TSP_CUT:
```

```
! Minimize total distance traveled;
 MIN = @SUM( LINK: DIST * Y);
! The Assignment constraints;
  @FOR( CITY( K):
  !  City K must be entered;
    @SUM( CITY( I)| I #NE# K: Y( I, K))= 1;
  !  City K must be departed;
    @SUM( CITY( J)| J #NE# K: Y( K, J))= 1;
  ! Cannot go to yourself;    Y( K, K) = 0;  );
! Subtour cuts;
  @FOR( SUBTOUR(t):
!  FLAG(t,i) = 1 if city i is in subtour t;
    @SUM( CITY(I) | FLAG(t,i) #EQ# 1:
      @SUM( CITY(J) | FLAG(t,j) #EQ# 1: Y(i,j))) <= TOURSIZE(t) - 1;     );
ENDSUBMODEL
CALC:
  @SET( 'TERSEO', 2);
  N = @SIZE( CITY);
  MXCUTS = @SIZE(SUBTOUR);
 ! Initially there are no subtour cuts;
 @FOR( SXC(t,i):
   FLAG(t,i) = 0;   );
   ICUT = 1;
! Loop over subtour cuts, ICUT;
 QUIT1 = 1;
 @WHILE( QUIT1 :
!  Solve current version;
   @SOLVE( TSP_CUT);
!  Find subtour if any;
   TOURSIZE(ICUT) = 0;
!  Loop over cities KURSTOP to find subtour starting at 1;
   KURSTOP = 1   QUIT2 = 1;
```

```
   @WHILE( QUIT2:
!    Loop over possible next cities j;
     @FOR(CITY(J):
      @IFC( Y(KURSTOP, J) #GT# .5:
        NEXT1 = J;      ););!    Next j;
      KURSTOP = NEXT1;
      TOURSIZE(ICUT) = TOURSIZE(ICUT) + 1;
      FLAG(ICUT,KURSTOP) = 1;
      !@WRITE( ' Next stop= ', CITY(KURSTOP),' Tour size=',TOURSIZE(ICUT),
@NEWLINE( 1));
      @IFC( KURSTOP #EQ# 1:  ! Back home/Completed the subtour?;
        QUIT2 = 0;     );
          ); ! End loop over cities in subtour;
   ! If subtour is in fact a full tour, or out of space, get out;
    @IFC( TOURSIZE(ICUT) #EQ# N #OR# ICUT #GE# MXCUTS:
       !We are done;
       QUIT1 = 0;
       @FOR( LINK: Y = Y); !Fix Y;   );
   ! Get ready for next cut;
    ICUT = ICUT + 1;    );  !End loop over add cuts;
ENDCALC
SETS:
   LINKOPT( LINK) | Y( &1, &2) #GT# .5;
ENDSETS
CALC:
! Give simple report;
  @IFC( TOURSIZE(ICUT-1) #LT# N:
    @WRITE(' Sorry, optimum not found', @NEWLINE(1));
  @ELSE
    KURSTOP = 1;
    CUMDIST = 0;
    K = 1;
```

```
    @WRITE( 9*' ', 'Tour:    Distance:', @NEWLINE( 1));
    @WRITE( '    1:    ',CITY( KURSTOP),'  ', 8*' ','0',@NEWLINE(1));
    @FOR( CITY:
!     Get next city;
      @FOR( LINKOPT( I, J) | I #EQ# KURSTOP:
        NEXT = J;
        CUMDIST = CUMDIST + DIST( I, NEXT);
        K = K + 1;
        @WRITE( @FORMAT( K, '5.0f'),':    ',CITY( NEXT),
         '  ', @FORMAT( CUMDIST, '9.0f'),@NEWLINE(1));  );
      KURSTOP = NEXT;    );   );
ENDCALC
END
```

これを実行すると次の解が出力される．結果は，TSP1 と同じである．1 秒で計算できた．

しかし，200 以上の都市をこのモデルで解くことは難しい．LOOPTES（Ver.4）では簡単に解ける．同じ LINGO と PC で実行しても，モデルによって数百倍以上の違いが出てくる．人間の知恵が生きてくるわけだ．読者も知のフロンティアに挑戦してみてほしい．

```
         Tour:    Distance:
    1:    NYK           0
    2:    PHL          92
    3:    PIT         397
    4:    CIN         681
    5:    ATL        1135
    6:    HOU        1977
    7:    SND        3498
    8:    LAX        3593
    9:    SNF        3998
   10:    STL        6123
   11:    CHI        6417
   12:    MON        7181
   13:    NYK        7577
```

図 5.4　5TSP4.lg4 の実行結果

6　ポートフォリオ分析

6.1　マーコウィッツの平均/分散ポートフォリオ・モデル

シカゴ大学の大学院生であったマーコウィッツ(1959)は，今日**マーコウィッツの平均/分散ポートフォリオ・モデル**として知られている博士論文を提出した．しかし，単に数学モデルの論文で現実問題を分析する経済学的な論文と評価されなかったらしい．しかし，シカゴ学派の泰斗ミルトン・フリードマン(1976 年ノーベル経済学賞受賞)に救われたらしい．後年，コンピューターや数理計画法ソフトの発展で現実の株や債権に応用できるようになり，1990 年にノーベル経済学賞を受けた，私が 1984 年に LINDO 製品の代理店になる交渉のため，シカゴ大学ビジネス・スクールの Linus Schrage 教授を訪問した．その際，フランクリン・ロイドの旧宅を案内してもらい，大学生協でシカゴ大学関連のノーベル賞受賞者の名前をプリントした T シャツをプレゼントされた．そして，Linus から「シュウイチもノーベル賞をとったら，シカゴ大学は 1 日でも訪問した人をシカゴ大学関係者としてＴシャツに名前を載せるだろう」と冗談を言われた．

実は，私はノーベル賞少年であった．多分，富山市立八人町小学校の先生から「頭が良いからノーベル賞を狙いなさい」といわれたことが原因であろう．しかし，近所の無欲な田中耕一さんが実際ノーベル賞をもらったことで私のむなしい思いも報われて感激である．

ポートフォリオ分析では，投資家は期待収益とその分散（すなわち，リスク）を考慮すると仮定している．期待収益は，株式の値上がり益と配当を考慮したものである．分散は，期待収益からの散らばり具合を表わしている．ある株式の分散が大きければ，値上がり幅も値下がり幅も大きい．これをリスクの尺度に用いている．しかし，実際の投資家は下側に振れることをリスクと考えても，上に振れることをリスクと考えにくいので，最初はこの点に戸惑うだろう．予測の不確かさをリスクとしているわけだ．

昔から「財産三分法」とか「卵を一つのかごに入れるな」といわれてきた．確率的に変動するものは，分散投資すべきことは経験的に知られていた．それを数学的にはっきりさせたわけだ．

今，3 社の株式データがあったとする．統計ソフトを用いて値上がり率を計算し，それを期待収益率とする．そして，投資対象銘柄 3 社の株価データから分散共分散行列を計算する．**図 6.1** は，ポートフォリオデータである．3 社の社名を分かりやすく X，Y，Z とする．B4：D4 には各社の期待利益が入っている．X 社は年間 3 割，Y 社は 2 割，Z 社は 8%の利益が期待される．ここで利益の一番大きな X 社に全て投資することは間違っている，というのがポートフォリオ理論の骨子である．

B6 から D8 の 3 行 3 列に 3 社の分散共分散行列のデータが与えてある.X 社単独の分散は 3，Y 社単独の分散は 2，Z 社単独の分散は 1 である．そして，X 社と Y 社の共分散は 1，X 社と Z 社の共分散は - 0.5，Y 社と Z 社の共分散は - 0.4 である.

	A	B	C	D	E
1					
2					
3		X	Y	Z	期待利益
4	利益	1.3	1.2	1.08	1.2
5	投資比率	0	1	0	リスク
6	分散共分散	3	1	-0.5	2
7		1	2	-0.4	
8		-0.5	-0.4	1	
9					
10	投資上限	0.75	0.75	0.75	
11					

図 6.1　ポートフォリオデータ

読者もこのデータを作成し，セル E4 に 3 社にある比率(X，Y，Z)で投資した場合の次の期待利益を入れてみよう.

```
=1.3*X+1.2*Y+1.08*Z
```

次にセル E6 に，3 社に比率(X，Y，Z)で投資した場合の分散を入れてみよう．分散は，次の行列の積で計算される．t は 1 行 3 列の行列を 3 行 1 列の行列に転置することを示している.

分散=(X，Y，Z)*V*(X，Y，Z)^t=3*X^2+2*Y^2+Z^2+2*X*Y-X*Z-0.8*Y*Z

さて 3 社への投資比率に関しては，次の制約がある．すなわち，3 社合計で 1 あるいは 100%と考える.

```
X+Y+Z=1;
```

問：各社への投資比率は 0 以上 1 以下で自由に読者が選択できる．3 社の期待利益と合成された分散の上限と下限はどうなるであろうか.

あるいは，全てを X に投資した場合はどうなるであろうか?実際，B5 に 1 をいれ，C5 と D5 を 0 にすれば,利益は当然であるが 1.3 で分散は 3 になる.表は Y=1 にした場合である.これによって，読者は自由に分散投資のリスクとリターンが計算できる.

さて先ほどの質問であるが，期待利益は 1.08 から 1.3 の間にある．分散は 1 から 3 の間になる.

6.2　LINGO でモデル化する

投資分析の注意点は，リスクを最小に，利益を最大にしたいという相反する 2 つの目的

関数がある点である．これらの間のトレードオフをどうするか悩ましい問題である．一番簡単なのは，これらを加重平均し単目的化することである．しかし，リスクとリターンという異なった尺度を一つにまとめてもその意味が分からなくなる．そこで，リターンを最大化する代わりにある値以上に制限し，リスクを最小化することが考えられる．

この方針で，マーコウィッツの平均/分散ポートフォリオ・モデル(PORT02.lg4)をLINGOで定式化すると次のようになる．例えば，リターンを1.15以上でリスクを最小にしたいモデルは次のようになる．

```
MIN=3*X*X+2*Y*Y+Z*Z+2*X*Y-X*Z-0.8*Y*Z;
X+Y+Z=1;
1.3*X+1.2*Y+1.08*Z>1.15;
```

これを解くと**図 6.2**の解が得られる．Xを約18%，Yを25%，Zを57%の比率で投資すればリスクは最小化され0.42になる．

```
 Local optimal solution found.
 Objective value:                              0.4209630
 Extended solver steps:                                5
 Total solver iterations:                              4
                      Variable           Value        Reduced Cost
                             X        0.1828794            0.000000
                             Y        0.2480545            0.000000
                             Z        0.5690661            0.000000
                           Row    Slack or Surplus      Dual Price
                             1        0.4209630           -1.000000
                             2         0.000000            0.5564202
                             3         0.000000           -1.215953
```

図 6.2　リターン1.15以上

次に，リターンを1.18以上に変更すると**図 6.3**のように，Xを約32%，Yを24%，Zを44%の比率で投資すればリスクは最小化され0.55になる．リターンが1.15から1.18に増やすことで，リスクが0.42から0.55に増えた．人生，いいとこどりができないということだ．結局，自分自身でリスクとリターンのトレードオフを決める必要がある．

```
 Local optimal solution found.
 Objective value:                              0.5501556
 Extended solver steps:                                2
```

```
 Total solver iterations:                           13
                      Variable           Value        Reduced Cost
                             X        0.3252918            0.000000
                             Y        0.2369650            0.000000
                             Z        0.4377432            0.000000
                           Row    Slack or Surplus      Dual Price
                             1        0.5501556           -1.000000
                             2         0.000000            7.628016
                             3         0.000000           -7.396887
```

図 6.3　リターンを 1.18 以上

6.3　汎用モデル

実際の株式は，例えば東証 1 部でもおよそ 1690 社ある．そこで，希望する銘柄の投資分析のために汎用モデルを作成しよう．図 6.1 で分散共分散行列をセル名 V に，投資比率を INVEST に，利益を RATE にそして 1 銘柄の投資上限を 0.75 に設定する．次が，汎用のポートフォーリオモデルである．

SUBMODEL 節では，SUB1 というモデルを定式化している．そして CALC 節で@SOLVE(SUB1)でこのモデルを解いている。実は、この CALC 節を用いると，複雑な最適化モデルが簡単に記述できる．私が 12 年間かけてやってきた判別分析の研究の再現と，やれなかったことが 2007 年の 12 月 27 日から 2008 年の 1 月 4 日のわずか 1 週間でモデル開発とデータの分析が行えた．

```
MODEL:
SETS:
   ASSET: RATE, UB, INVEST;
   COVMAT( ASSET, ASSET): V;
ENDSETS
DATA:
  ASSET = GOOGLE, YAHOO, CISCO;
  RATE = @OLE( );
  UB   = @OLE( );
  V    = @OLE( );
ENDDATA
```

```
  SUBMODEL SUB1:
     MAX = @SUM( COVMAT( I, J): V( I, J) * INVEST( I) * INVEST( J));
     RETURN = @SUM( ASSET: RATE * INVEST);
     ! Must be fully invested;
     @SUM( ASSET: INVEST) = 1;
     ! Upper bounds on each;
     @FOR( ASSET: @BND( 0, INVEST, UB));
     ! Must achieve target return;
     RETURN >= RET_LIM;
  ENDSUBMODEL
  CALC:
     !@SET( 'DEFAULT');
     !@SET( 'TERSEO', 2);
     @SET( 'STAWIN', 0);
     @SOLVE(SUB1);
  ENDCALC
  END
```

出力結果は**図6.4**の通りである.

```
Objective value:                              2.187500
Extended solver steps:                               5
Total solver iterations:                            54
                     Variable           Value        Reduced Cost
                       RETURN        1.275000            0.000000
                      RET_LIM        0.000000            0.000000
                 RATE( GOOGLE)        1.300000            0.000000
                  RATE( YAHOO)        1.200000            0.000000
                  RATE( CISCO)        1.080000            0.000000
                    UB( GOOGLE)       0.7500000            0.000000
                     UB( YAHOO)       0.7500000            0.000000
                     UB( CISCO)       0.7500000            0.000000
                INVEST( GOOGLE)       0.7500000           -2.500000
                 INVEST( YAHOO)       0.2500000            0.000000
```

INVEST(CISCO)	0.000000	3.450000
V(GOOGLE, GOOGLE)	3.000000	0.000000
V(GOOGLE, YAHOO)	1.000000	0.000000
V(GOOGLE, CISCO)	-0.5000000	0.000000
V(YAHOO, GOOGLE)	1.000000	0.000000
V(YAHOO, YAHOO)	2.000000	0.000000
V(YAHOO, CISCO)	-0.4000000	0.000000
V(CISCO, GOOGLE)	-0.5000000	0.000000
V(CISCO, YAHOO)	-0.4000000	0.000000
V(CISCO, CISCO)	1.000000	0.000000

Row	Slack or Surplus	Dual Price
1	2.187500	1.000000
2	0.000000	0.000000
3	0.000000	2.500000
4	1.275000	0.000000

図6.4　汎用モデルの解

6.4　効率的フロンティア

次は10段階のリターンレベルで,Portfolio分析を行い,効率的フロンティア曲線を描く.LINGOの印刷の命令を本書で説明することは多くの読者に意味がないので省く.金融機関の人や利用したい人はマニュアルを読んでほしい.

```
MODEL:
! Solves the generic Markowitz portfolio
  model in a loop to generate the points
  on the efficient frontier;
SETS:
   ASSET: RATE, UB, INVEST;
   COVMAT( ASSET, ASSET): V;
   POINTS: XRET, YVAR;
ENDSETS
DATA:
! Number of points on the
```

```
   efficient frontier graph;
  NPOINTS = 10;
  POINTS = 1..NPOINTS;
! The stocks;
  ASSET = GOOGLE, YAHOO, CISCO;
! Expected growth rate of each asset;
  RATE = @OLE( );
! Upper bound on investment in each;  UB   = @OLE( );
! Covariance matrix;
  V    = @OLE( );
ENDDATA
! Below are the three objectives we'll use;
SUBMODEL SUB_RET_MAX:
   [OBJ_RET_MAX] MAX = RETURN;
ENDSUBMODEL
SUBMODEL SUB_RET_MIN:
   [OBJ_RET_MIN] MIN = RETURN;
ENDSUBMODEL
SUBMODEL SUB_MIN_VAR:
   [OBJ_MIN_VAR] MIN =
    @SUM( COVMAT( I, J): V( I, J) * INVEST( I) * INVEST( J));
ENDSUBMODEL
!and the constraints;
SUBMODEL SUB_CONSTRAINTS:
   ! Compute return;
   RETURN = @SUM( ASSET: RATE * INVEST);
   ! Must be fully invested;
   @SUM( ASSET: INVEST) = 1;
   ! Upper bounds on each;
   @FOR( ASSET: @BND( 0, INVEST, UB));
   ! Must achieve target return;
   RETURN >= RET_LIM;
```

```
ENDSUBMODEL
CALC:
! Set some parameters;
   ! Reset all params;
   @SET( 'DEFAULT');
   ! Output error messages only;
   @SET( 'TERSEO', 2);
   ! Suppress status window;
   @SET( 'STAWIN', 0);
! Solve to get maximum return;
   RET_LIM = 0;
   @SOLVE( SUB_RET_MAX, SUB_CONSTRAINTS);
! Save maximum return;
   RET_MAX = OBJ_RET_MAX;
! Solve to get minimum return;
   @SOLVE( SUB_RET_MIN, SUB_CONSTRAINTS);
! Save minimum return;
   RET_MIN = OBJ_RET_MIN;
! Interval between return points;
   INTERVAL =
    ( RET_MAX - RET_MIN) / ( NPOINTS-1);
! Loop over range of possible returns,   minimizing variance;
   RET_LIM = RET_MIN;
   @FOR( POINTS( I):
      @SOLVE( SUB_MIN_VAR, SUB_CONSTRAINTS);
      XRET( I) = RET_LIM;
      YVAR( I) = OBJ_MIN_VAR;
      RET_LIM = RET_LIM + INTERVAL;   );
! Display the results;
   @WRITE( '     Return     Variance', @NEWLINE( 1));
   @FOR( POINTS: @WRITE( @FORMAT( XRET, '#12.6G'),
    @FORMAT( YVAR, '#12.6G'), @NEWLINE( 1))   );
```

```
ENDCALC
CALC:
! The remainder of the model graphs the efficient frontier;
   NHASHY = 20;   NHASHX = 60;   SCALE = 10;
   V0 = @FLOOR( YVAR( 1) * SCALE);
   V0 = V0 / SCALE;
   V1 = @FLOOR( YVAR( NPOINTS) * SCALE + .5);
   V1 = V1 / SCALE;
   R0 = @FLOOR( RET_MIN * SCALE);
   R0 = R0 / SCALE;
   R1 = @FLOOR( RET_MAX * SCALE + .5);
   R1 = R1 / SCALE;
ENDCALC
SETS:
   HASHX /1..NHASHX/: XAXIS;
   HASHY /1..NHASHY/: YAXIS;
   GRID( HASHY, HASHX): CHECK;
ENDSETS
CALC:
   XWIDTH = ( R1 - R0) / NHASHX;
   XAXIS( 1) = R0 + XWIDTH;
   XAXIS( NHASHX) = R1;
   YHEIGHT = ( V1 - V0) / NHASHY;
   YAXIS( 1) = V0 + YHEIGHT;
   YAXIS( NHASHY) = V1;
   @FOR( HASHY( J) | J #GT# 1 #AND# J #LT# NHASHY:
      YAXIS( J) = YAXIS( J - 1) + YHEIGHT;   );
   @FOR( HASHX( J) | J #GT# 1 #AND# J #LT# NHASHX:
      XAXIS( J) = XAXIS( J - 1) + XWIDTH;);
   @FOR( GRID: CHECK = 0);
   @FOR( POINTS( P):
      J = 1;
```

```
      @WHILE( XRET( P) #GT# XAXIS( J): J = J + 1);
      I = 1;
      @WHILE( YVAR( P) #GT# YAXIS( I): I = I + 1);
      CHECK( I, J) = 1;);
   INDENT = 12;
   @WRITE( @NEWLINE( 2), (INDENT-3)*' ', 'Variance', @NEWLINE( 1));
   @WRITE( INDENT*' ', '^', @NEWLINE( 1));
   @FOR( HASHY( II):
      @IFC( II #EQ# 1:
         @WRITE( ( INDENT - 5) * ' ', @FORMAT( V1, '#4.2G'), ' |');
      @ELSE
         @IFC( II #EQ# @SIZE( HASHY):
            @WRITE( ( INDENT - 5) * ' ', @FORMAT( V0, '#4.2G'), ' |');
         @ELSE
            @WRITE( INDENT*' ', '|');          );      );
      @FOR( HASHX( JJ):
         @IFC( CHECK( @SIZE( HASHY) - II + 1, JJ):
            @WRITE( '*');
         @ELSE
            @WRITE( ' ');          );      );
      @WRITE( @NEWLINE( 1));   );
   @WRITE( INDENT*' ', NHASHX*'-', '>', @NEWLINE( 1));
   @WRITE( (INDENT-2)*' ', @FORMAT( R0, '4.2G'), ( NHASHX - 4)*' ', R1, @NEWLINE( 1));
   @WRITE( ( INDENT + NHASHX - 3)*' ',  'Return', @NEWLINE( 3));
ENDCALC
END
```

出力結果は次の通りである.

```
   Return     Variance
   1.11000    0.417375
   1.12833    0.417375
   1.14667    0.418054
   1.16500    0.462381
```

1.18333　　0.575957

1.20167　　0.758782

1.22000　　1.01086

1.23833　　1.33218

1.25667　　1.72275

1.27500　　2.18750

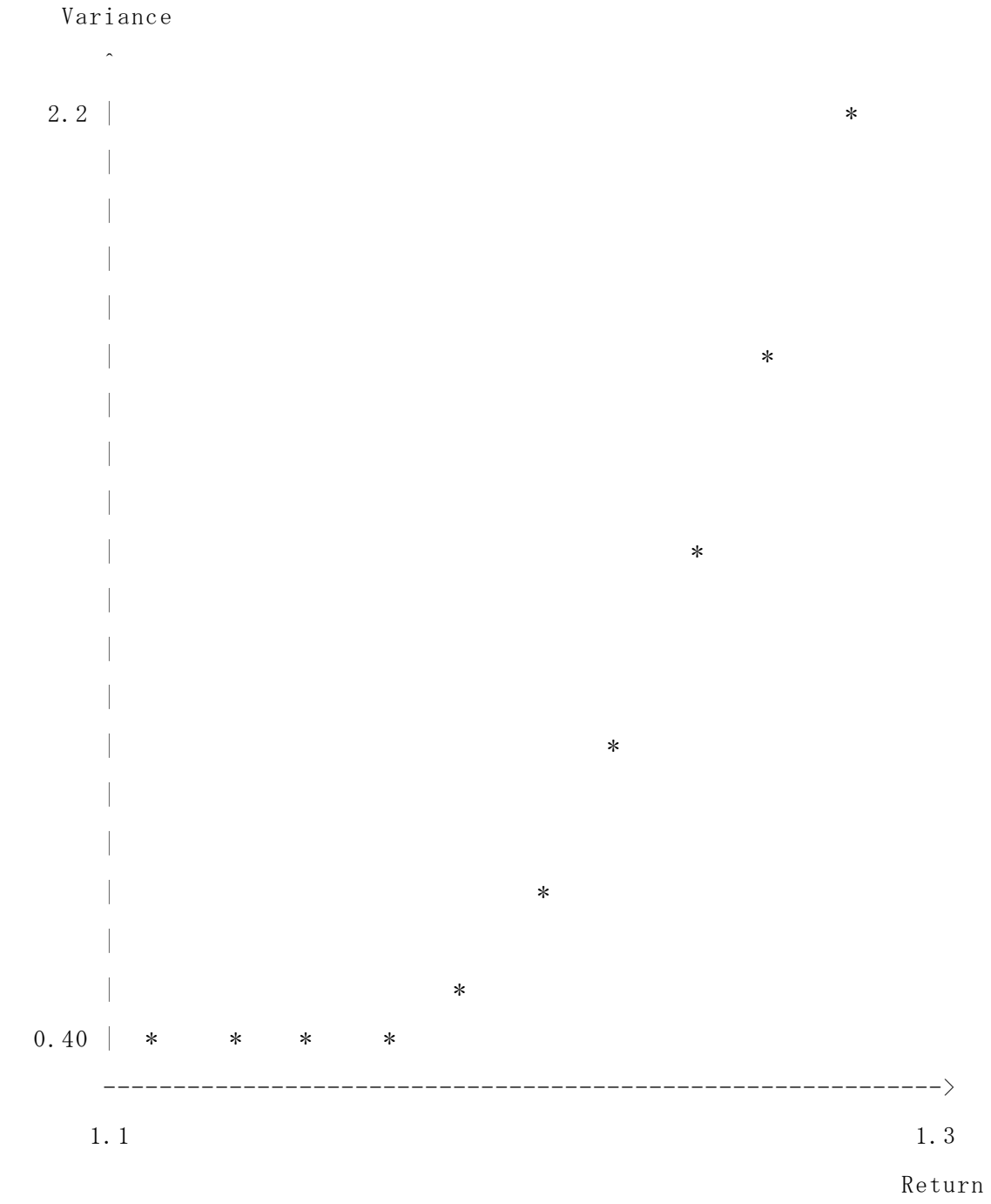

図6.5　効率的フロンティア・モデルの解

7　人生の達人

数理計画法は，自然科学に対して，人・金・物を管理し最適化する学問である．そして，人は金さえあれば，人を雇い，物を購入できるので，とかく拝金主義に走りがちである．また，金の少なきを憂え，あたふたと時間を無駄にする．

しかし,貧富の差に関係なく人間に平等に与えられているのが時間である.この時間は，人，金，物というような浮き世と違い，自然科学の範疇に属するので，管理の対象にすることの意識が少なかったようだ．すなわち，制御不可能なものと考えていたのではなかろうか．

また，生ある物は，すべからく死を迎えるという大命題の前に，限りある時間をどう有効に使うかということに対し思考が停止してしまったのだろう．

人は,金をもってしても人生の時間をあがなうことはできない.ただ知恵によってのみ，時間をうまく使い，人の２倍３倍に時間を有効に使うことができる．

時間を管理する手法の PERT（パート，Program Evaluation and Review Technique）は，20 世紀になってやっと発見された方法である．なんとそれまで，原始的な横線工程表（ガント・チャート）くらいしかなかった．ここに人間にとって最も貴重でありながら，管理することをあきらめていた時間に対する意識の欠落が読みとれる．

7.1　時間の管理

外食したとき，従業員が多い割には，もたもたした店にたまに出くわす．なんと手際の悪い店だ,あるいは能率の悪い店だという印象を持つことは一度ならずあるだろう.また，比較できないので分からないが，きっとやりくり上手な奥さんと下手な奥さんがいるに違いない．公平をきせば，忙しそうに体を動かしていても，さっぱり成果の上がらない人もいる．

そして，その理由を人間の能力というものに置き換えて納得してしまうから，それ以上なぜかという詮索は進まない．また，ブルー・カラーの作業改善は行きつくとこまでいったが，ホワイトカラーの生産性の改善は一向に進まない．

その大きな理由は，時間管理に対する無知と惰性からきているのではなかろうか．

ことほど左様に，時間を管理することは難しい．

(1) ガント・チャートと時間管理

不思議なことだが，時間を管理する工程管理の手法は，有史このかた第２次世界大戦前まで，ほどんとみるべき物はなかった．

ただ，ガント・チャートとして知られる，お馴染みの工程別に開始と終わりを示す横線を書く程度のものがあっただけだ．

例えば，次の**表 7・1** はある画期的な商品開発プロジェクトの作業工程のリストである．これを用いて，ガント・チャートを描いてみよう．ここで，作業名（またはアクティビティ）は，各作業の工程を表す．括弧内の英単語は，後で LINGO で用いる決定変数である．

表 7・1　画期的商品開発

No.	作業名	工数	開始ノード	終了ノード	先行作業
1	全体設計（Total）	10	T0	T1	－
2	詳細設計（Detail）	15	T1	T2	Total
3	アプリ開発（Appli）	8	T2	A1	Detail
4	アプリ検査（ApplTest）	8	A1	A2	Appli
5	OS アプリ検査（OS Appli）	3	A2	A3	ApplTest,D1
6	実機検査（OSApHard）	4	A3	A4	OSAppli,D2
7	最終検査（LastTest）	4	A4	H4	OSApHard,D3
8	販売活動（Sales）	3	H4	H5	LastTest,Market
9	OS 開発と OS 検査（OSTest）	21	T2	OS1	Detail
10	ダミー（D1）	0	OS1	A2	OSTest,D4
11	結合検査（OSHard）	5	OS1	H2	OSTest,D4
12	ダミー（D2）	0	H2	A3	OSHard
13	ハード改良（SysTest）	12	H2	H3	OSHard
14	ダミー（D3）	0	H3	A4	SysTest
15	ハード開発とハード検査（HardTest）	11	T2	H1	Detail
16	ダミー（D4）	0	H1	OS1	HardTest
17	マーケティング（Market）	5	H1	H4	OSHard

この表の作業の所要時間と，作業者の都合を聞いて，適当に作業工程を決めることは，普段よく行われている光景である．しかし，各作業の間には，ある作業が終わってから初めて開始でき，終わらないのに開始できない物がある．これらの関係を，先行と後続作業と呼ぶ．この情報を入れて**図 7.1** のような工程表が一つの例としてできあがる．

ここまでが，なんと人類の 2000 年以上の知恵なのだ．

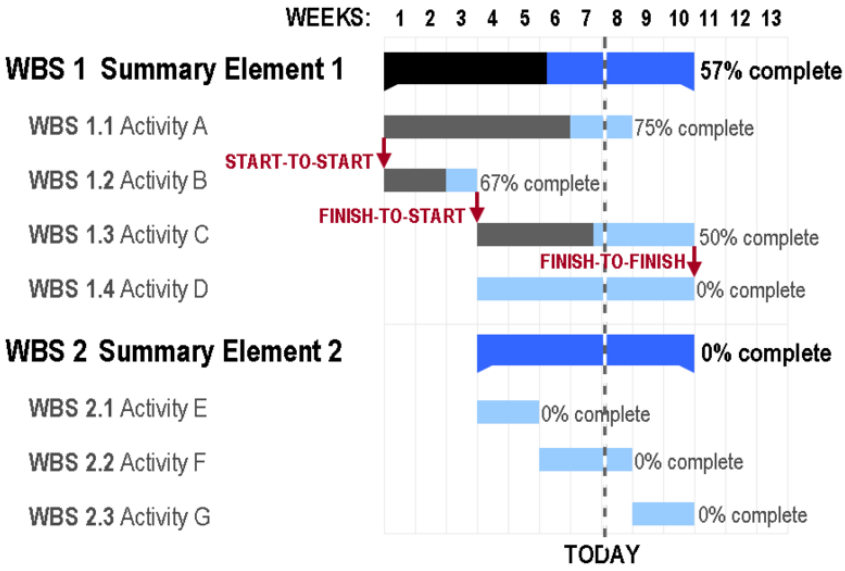

図 7・1　ガント・チャート（Wikipedia より転載）

（2）ロッキード事件の遠因

私が敬愛する近藤次郎先生によれば，ロッキード事件の遠因は，ガント・チャートにあるということだ．その心は，次の通りだ．

ロールス・ロイス社といえば，倒産してしまったが，英国の名門高級自動車メーカーである．この会社はジエット・エンジン・メーカーとしても有名だった．

そして，ロッキードから開発依頼を受けていたトライスターのエンジンの開発プロジェクト管理に，このようなハイテク産業でガント・チャートが用いられていた．これによって，開発が大きく遅れ，同社の倒産の原因となった．

一方，これが原因でトライスター開発に後れをとったロッキード社は，失地回復のためピーナッツ（田中角栄元首相に対する賄賂）を用いて日本に売り込まざるをえなかった．これが例のロッキード事件につながる一連の流れである．

かように，ガント・チャートの利用は，名門企業を倒産に追い込み，一国の総理の犯罪まで生んだ訳だ．

近代産業に，古めかしい管理手法．笑ってはおられない．

注:インターネットでガント・チャートを検索すれば，沢山のフリーソフトやシェアウエアのガント・チャートがあることが分かる．しかし，PERT でクリティカル・パスを調べた後で，これらを利用してもらいたいものだ．

（3）時間を制す

これに対して，PERT は，先ほどの表の情報だけを用いて著しい成果を上げる画期的な時間を管理する手法だ．別に難しい理論も何もないのに，なぜ我々の先祖は長らく発見でき

なかったのだろう．そのような思いで，この章を読んでほしい．

PERT は，1956 年に米海軍の SOP(Special Project Office)の艦船弾道ミサイル計画に始まっている．この計画の主要部分は，ポラリス型原子力潜水艦の開発だ．この時代，米国はソ連のスプートニク１号に後れをとり，国を挙げて，技術の遅れを取り戻すべく注力していた時代背景がある．

多くの企業を巻き込んだプロジェクトは，古今東西を問わず，納期遅れや開発費用の増大はあたりまえだ．そこで，SOP は，ロッキード社とブーツ・アレン・ハミルトン社から技術者派遣を受け，米海軍の中に OR 班を作った．

そこで開発されたのが，PERT だ．主な開発責任者は，数学者の C.E.クラーク氏といわれている．この手法は，ポラリス型原子力潜水艦建造で，当初予定していた７年を２年短縮するという成果を収めたといわれる．

この成果をふまえて,米国政府は各種プロジェクトのプロポーザルに PERT を義務づけた．これによって，広く知られることになった．

PERT は，当初日程管理が中心だった．その後，米国宇宙開発局（NASA）が，時間のほか，人，金，物を含めた PERT/COST と呼ばれる手法に発展させた．

公共事業に限らないが，入札金額の妥当性だけでなく，工期の妥当性に関しても，アメリカのようにできるだけ客観的な提案書の内容で決めるという姿勢が重要だ．

7.2　画期的な商品開発プロジェクト(本章は，坂村教授への個人的オマージュです)

パソコンのガレージ産業から大きくなった（株）新村コンピュータでは，新規事業として，東京大学の坂村健教授の提案するトロン OS を用いたゲーム専用機を作ることにした．

（1）PC の世界

コンピュータの世界は，**図 7･2** に示すように，ハードウェアの上にオペレーティングシステム（Operating System, OS）が乗り，その上にアプリケーションソフトがあるピラミッド構造になっている．

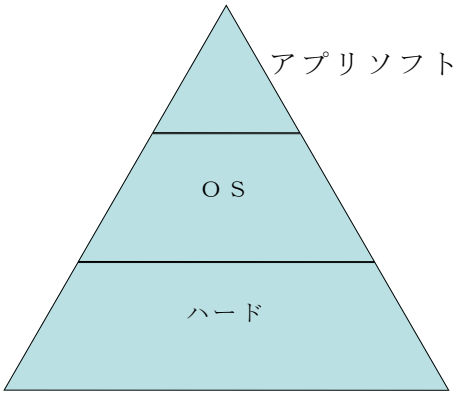

図 7･2　コンピュータのピラミッド

皆さんになじみやすいのは，ハードウェアとしては一般に PC であろう．そして，その上に Windows というマイクロソフト社の OS が搭載されている.このような PC は，IBM PC/AT 互換機と呼ばれ，IBM が設計図を公開し，IBM 以外の日本の

メーカーや，デルやHP（ヒューレット・パッカード）などが生産しているものをいう．

昔は日本語処理をハードで行う必要があり，NECはPC98という独自のPCの上にマイクロソフト社のMS/DOSやWindowsを搭載したものが，一時日本では50%以上のシェアを持っていた．

そこで日本IBMは，PCの処理能力が向上したことを受け，ソフト的に日本語を処理することを可能にし，それを他社に公開することでNECの寡占状態を崩すことに成功した．しかし，日用品化したPCは世界的なコスト競争の世界である．DELLやHPに差をつけられたIBMは，2002年の12月になって自ら新しく作り出したPC事業を中国の企業に売却するというリストラを行い世界の人々をびっくりさせた．

一方，アプリケーションソフトは，本書で扱うような理数系のソフトからゲームソフトまで多種多様である．

そして，OSは，このアプリケーションソフトとハードの仲立ちをする，言わば共通の基本ソフトウェアである．OSがなければ，アプリケーションソフト自体が，それぞれOSが果たしているハードとのやりとりを行う共通ソフトを独自で開発する必要がある．

（2）IBM PC/AT互換機とMacの世界

先程，IBMのPC事業の売却に触れた．なぜ，このようなことになったのだろうか．それは,IBMはPCの心臓部であるMPU(マイクロ･プロセッサー･ユニット)をインテルに外注し，OSをマイクロソフト社に外注し，これらの開発費用のリスクを最小化しようとしたためである．また，IBM PC/AT 互換機開発当時，IBM より先行し大きな市場を持っていた Appleを追撃する為，IBM PC/AT互換機の仕様を公開したためである．これによって，IBM PC/AT互換機市場はMacより大きくなった．しかし，この市場において，インテルとマイクロソフト社のウインテル連合しか儲からなくなった．これらの重要部品を購入し単に組み立てるPCメーカーは，スケールメリットを生かしコストダウン競争の世界になってしまった．

PCにはIBM PC/AT互換機の他，アップルコンピュータ社のMac/OSで動くPCがある．Macのハードと OSはいずれもアップルコンピュータ社の独自製品のため，IBM PC/AT互換機のように多くの会社が作っていないことに注意すべきである．このため，一時攻勢を極めたが,多くのアプリケーションソフトの開発会社がMac/OSからWindowsで稼働するIBM PC/AT互換機で稼動するソフトの開発に移行したため，現時点では劣勢である．

（3）企業で使われるコンピュータ

一方，企業では個人用としてPCは使われているが，共通の業務処理には，UNIXや，IBMやNECや富士通などのメーカーが独自に開発した汎用コンピュータが用いられている．

日本のソフトウエア産業は,なぜかこのUNIXや汎用機上で動く個別企業から発注される

業務用のアプリケーション開発に比重がある.

これに対して，本書で紹介する数理計画法ソフトや統計ソフトのような不特定多数をユーザーとする，いわゆるパッケージソフトの開発に弱いという特徴がある.

そしてそれ以上に，OS はこれまで米国の企業に独占されてきた感がある．もちろん，情報処理産業で一番儲かるのは,この OS を独占的に販売するマイクロソフトのような企業であり，次いで Oracle のような DBMS や，SAP のような業務ソフトそして SAS のような汎用統計パッケージソフトの開発企業である.

これに対して，最近では UNIX や汎用機の OS として，無償で公開されている Linux が有名である.

しかしそれ以前に，日本人研究者が開発し，米国の巧みな産業政策によって一次抹殺されたが，しぶとく日本の産業界に根付いて，不死鳥の如く甦った OS がある．それがトロン（TRON）である.

トロンは一般に，家電製品や産業機械の中に組み込まれているため，我々の目につきにくいのである.

MPU（マイクロプロセッサーユニット）と呼ばれる IC チップの上に，トロン OS を載せ，その上で稼働するアプリケーションソフトが産業界で開発されている.

皆さんが手にしている携帯電話などが高度な機能を持つのも，デジタルカメラが便利なのも，あるいは自動車の制御に用いられ快適なドライブができるのも，この小さなコンピュータのおかげである．カーナビなどは，衛星からの位置情報を受信し，現在位置を地図上に表示する代表的なものである.

日本人は,グランドキャニオンのような壮大な景観を持つ Windows のような OS 開発にはなぜ向いていないのかと不思議に思うことがある．一方，日本人は TRON やそれを応用した商品開発のように,幕の内弁当と同じく,小さなものに神が宿る製品開発が得意なようだ.

（4）プロジェクトの紹介

さて，**表 7・1** に示した，画期的と社長の新村が自画自賛している内容を，企業秘密に触れない範囲で紹介しよう．開発する製品は，例えばゲーム専用機や携帯電話の新製品の開発をイメージすればよい.

このような大型の開発は，かっこ良くプロジェクトと呼ばれる．最近では，NHK の「プロジェクト X」でプロジェクトという言葉も一般的になった．そして，プロジェクトは，同一の仕事内容である作業工程（アクティビティ）に分割される.

プロジェクトの最初の作業は，ハードウェア，OS，開発するアプリケーションの全体の役割と関連を決める作業である．これを「全体設計（Total)」と呼ぶことにする.

このような作業は，開始点を表わす開始ノード（T0）と終了ノード（T1）を結ぶ**図 7･3** の矢印線（方向を持ったアーク）で表わすことにする．ただし，これ以降は矢印を省く．

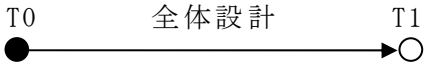

図 7･3　ノードとアーク

このように，ノードとアークのネットワークで表わされるものには PERT の他，石油やガスなどの輸送問題や交通量などにも利用できる．

そして，作業の内容を決め，それにかかる工数を見積ることになる．ここでは 10（ヶ月）としているが，楽観値，悲観値，平均値（あるいは最頻値）の 3 点見積りを行うこともある．

次に全体設計が終ると，それに基づいて作業 No.2 のハード，OS，アプリのより詳しい詳細設計（Detail）に入る．見積り工数は 15 である．

ここで重要なことは，詳細設計は，全体設計が終ってはじめて開始できるという事実（制約）である．このとき，全体設計を詳細設計の先行作業という．逆に，詳細設計は全体設計の後行作業になる．

この後プロジェクトは，No.3 から No.7 までのアプリケーション開発と，No.9 の OS 開発と検査と，No.15 のハード開発と検査とが並行して行われる．3 つの開発は，例えば OS 開発と OS 検査というように，それ単独での検査が行われ不具合が直される．そして，ハードウェアと OS が組み合わされ，No.11 のハードと OS の結合検査が行われる．このためには，OS 検査とハード検査が先行作業として終っている必要がある．

No.16 のダミー（D4）は，実際にはハード検査の終了ノードの H1 が，OS 検査の終了ノード OS1 と同じことを表わす．もしこのダミー作業を用いないと，T2 から OS1 を開始ノードと終了ノードとする作業が２つあり，コンピュータプログラムが識別できなくなるからである．D2 と D3 は，これと同じ理由でダミーを用いている．

アプリケーションの検査が終れば，開発時間の短縮のため，ハードと OS の結合検査前にこのハードと OS の上にアプリケーションをインストールし，不具合などを調べることになる．このため No.10 のダミー作業（D1）が発生する．もちろんこの場合，OS1 を A2 と同じにしてもいいが，ここでは図を見やすくするために用いた．

一方，ハードと OS の結合検査が終ると，マーケティング部門はそれをもって重要顧客や展示会に出展することになる．時折，展示デモで動かないので説明員が慌てている姿を見るのは，PERT などを使わない為，プロジェクト管理が上手く行なわれずこれらの検査が終了しないまま出展されるためである．

7.3　PERT

（1）ネットワークを描く

さて，前述の**表 7・1** のプロジェクトの情報をネットワークで表してみよう．

各作業を，PERT では，アクティビティと呼び，**図 7･3** のような矢線で表すことにする．矢線の尾は作業の開始を表し，頭は終了を示す．矢線の開始と終わりは，丸で示され，イベントという．すなわち，作業と作業の結合点，あるいは開始と終了を表す．

そして，**図 7･4** のようなネットワーク図が描かれる．これが PERT 図である．

TOTAL は,DETAIL の先行作業であるから,図のようになることは自然である.DETAIL は,APPLI と OSTest と HardTest の先行作業になっている，などである．

これらのネットワークを作成する規則は簡単だ．次の３つの規則がある．

・イベントの役割

イベントは，そこに入ってくる先行アクティビティがすべて終了しないと，そのイベントから出ていくアクティビティを開始できない．

イベント T2 に注目しよう．このとき，図のように，そのイベントに入ってくる Detail が先行アクティビティである．このイベントからは，Appli と OSTest と HardTest の３つのアクティビティがでている．これらは，先行のアクティビティ T2 が終了しないと作業が開始できない．

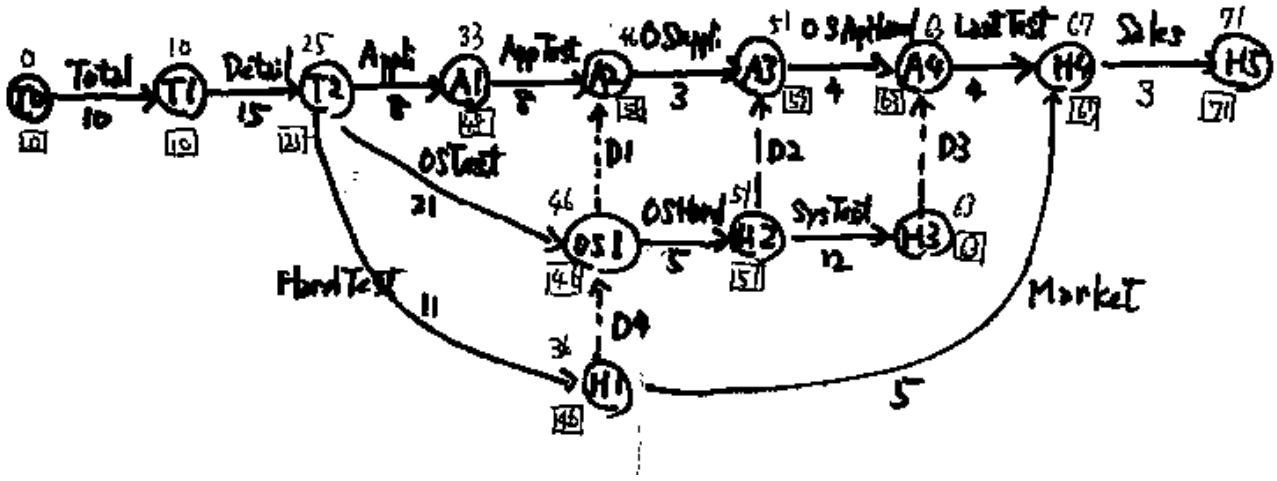

図 7･4　PERT 図

・見やすくするためにダミー・アローの設定

見やすくするため所要時間ゼロの擬似的な作業,ダミー･アローを設定することもある．

例えば，OS1 から A2 にダミー・アローD1 を設定する．これは，PERT 図を見やすくするためである．このとき，AppliTest と D1 が終わらないと OSAppli は開始できない．

これら２つの規則は，本質的である．次の規則は，コンピュータ化のために必要である．

・同一イベントからの矢線の制限のためにダミー・アローを用いる

同一イベントから入ってくる矢線が複数の場合（イベントOS１,A3,A4 ），コンピュータ処理に困るので，図のようなダミー・アローD2，D3，D4を用いて変更する．

さて，ネットワーク図を作成したら，次にイベントの２つのノード･タイムを考えよう．これが，PERTの核心だ．

（2）イベントの最早開始時刻

最早開始時刻は，出発点を０として，各アクティビティを予定通りの日程で実施した場合，そのイベントを最も早く着手できる時間である．

さて，ネットワークが完成したら，イベントの最早開始時刻を，**図 7･4** のイベントの上に書き込もう．

ネットワークの出発点のイベントT0は，0である．この値に，作業TOTALの所要時間10を足した10が，イベントT1の最早開始時刻になる．イベントT2は，25である．

しかし，イベントOS1のように，複数のイベントが入ってくる場合，一番大きな値（MAX演算）がこの値になる．この場合は，OS1の値は，OSTestは21でHardTestは11なので，T2の25に21を足した46になる．すなわちHardTestの作業が完成しても，OSTestが終わらない限りOS1が開始できない為である．

この場合，最初は最早と言う言葉に違和感を覚えるかもしれない．このような規則で，各イベントの下に最早開始時刻を書き込んである．ゴールの H5 点の最早開始時刻は，70になる．これらは表7･2の最早開始時刻にまとめてある．

（3）イベントの最遅完了時刻

次に，ゴールの最早開始時刻70から出発し，次の要領で出発点までの最遅完了時刻を計算する．このノード・タイムは，イベントの下に，四角の枠組みの中に書き込むことにする．

イベントH5が70に終わるためには，イベントH4は，遅くても67までに作業を完成していなければいけない．もし終わっていなければ，70に作業を完成できない．

イベントH2からは，２つのアクティビティがでている．この場合は，SysTestに対する値51と，D2に対する値59の最小値51をとることになる．すなわち，MIN演算が働き，51を採用する．もし，59を採用すると，H3は71，A4は71，H4は75，H5は79になって70で作業が終了しない．

（4）ノード・タイムとクリティカル・パス

以上求められた最早開始時刻と最遅完了時刻をあわせてノード・タイムという．２つの

ノード・タイムの一致する経路をなぞってみよう．これが，クリティカル・パスになる．隘路とか，ボトル・ネック・パスともいわれる．各作業が予定通り順調に進めば，このクリティカル・パスが，プロジェクトの最長経路になる．ただし，最長経路という言葉にだまされてはいけない．ガント・チャートを使って，経験と勘で作業手順を考えていては，このような最短経路を発見することはおぼつかない．

（5）余裕時間

ノード・タイムが計算されたら，次の定義に従って，各アクティビティのフロート（余裕時間）を計算しよう．

トータル・フロート（全余裕）＝（最遅完了時刻）－（最早開始時刻）－（作業時間）

フリー・フロート（自由余裕）＝（プロジェクトの終端の最早開始時刻）－（プロジェクトの開始ノードの最早開始時刻）－（作業時間）

この式に従い，各アローの下に，２つのフロートを書き込んであるので，確認してほしい．枠組みの数字が，自由余裕である．

全余裕は，各アローが属する経路全体の余裕である．これに対し，自由余裕は，先行作業が予定通り進んでおれば，そのアローで許される最大の遅れを表している．

クリティカル・パスは，２つの余裕時間ともゼロという特徴を持っている．以上まとめると，次の表 7･2 が得られる．そして，Total，Detail，OSTest，OSHard，SysTest，D3，LastTest，Sales がクリティカル・パスになる．

表 7・2　画期的商品開発の手計算の結果

No.	作業名	工数	最早開始時刻	最遅完了時刻	全余裕	自由余裕
1	全体設計（Total）	10	0	10	10-0-10=0	10-0-10=0
2	詳細設計（Detail）	15	10	25	25-10-15=0	25-10-15=0
3	アプリ開発（Appli）	8	25	48	48-25-8=15	33-25-8=0
4	アプリ検査（ApplTest）	8	33	56	56-33-8=15	46-33-8=5
5	OS アプリ検査（OS Appli）	3	46	59	59-46-3=10	51-46-3=2
6	実機検査（OSApHard）	4	51	63	63-51-4=8	63-51-4=8
7	最終検査（LastTest）	4	63	67	67-63-4=0	67-63-4=0
8	販売活動（Sales）	3	67	70	70-67-3=0	70-67-3=0
9	OS 開発と OS 検査（OSTest）	21	25	46	46-25-21=0	46-25-21=0
10	ダミー（D1）	0	46	56	56-46-0=10	46-46-0=0

11	結合検査（OSHard）	5	46	51	51-46-5=0	51-46-5=0
12	ダミー（D2）	0	51	59	59-51-0=8	51-51-0=0
13	ハード改良（SysTest）	12	51	63	63-51-12=0	63-51-12=0
14	ダミー（D3）	0	63	63	63-63-0=0	63-63-0=0
15	ハード開発とハード検査（HardTest）	11	25	46	46-25-11=10	36-25-11=0
16	ダミー（D4）	0	36	46	46-36-0=10	46-36-0=10
17	マーケティング（Market）	5	36	46	46-36-5=5	46-36-5=5

7.4 PERT ネットワークと LP

この例の場合,クリティカル･パスを紙と鉛筆で計算できたが,LP で定式化してみよう.この場合,ほとんどの人が異なった２つの定式化のうちの最初の１つを考えることだろう.

（1）万人による定式化

１つめの定式化を以下に示す.

変数 TOTAL, DETAIL 等は，その仕事がクリティカル・パス上にあるかないかにより１または０の整数値をとるものとする.

目的関数は，クリティカル・パスを求めることに対応し，次のようになる.

```
Max=10*Total+15*Detail+8*Appli+8*ApplTest+3*OSAppli+4*OSApHard+4*LastTest
   +3*Sales+21*OSTest+5*OSHard+12*SysTest+11*HardTest+5*Market;
```

この目的関数は一見間違っているように思える．何故ならプロジェクトの長さを最大化したいわけではないからである．しかし，クリティカル・パス上の作業工数は，他のどのパス上の作業工数の合計より大きくならなければいけない．そして，適当な制約式を設定すると，クリティカル・パスを求めるための目的関数となる.

制約式は，次の条件を満たすように設定する.

①TOTAL と Sales は必ずクリティカル・パス上にある.

②先行作業の１つがクリティカル・パス上にあるときのみ，その仕事がクリティカル・パス上にある可能性がある．また，ある仕事がクリティカル・パス上にあるとき，その後続作業のうちただ１つがクリティカル・パス上にある．これは入力と出力が等しいという保存法則を考えれば良い.

すなわち,クリティカル･パスにだけ,１単位の水を流してやるようなものだ.次の LINGO モデル（PERT01.lg4）の制約式は上の条件を満足する.

```
Max=10*Total+15*Detail+8*Appli+8*ApplTest+3*OSAppli+4*OSApHard+4*LastTest
```

```
+3*Sales+21*OSTest+5*OSHard+12*SysTest+11*HardTest+5*Market;
-Total=-1;
Total-Detail=0;
Detail-Appli-OSTest-HardTest=0;
Appli-ApplTest=0;
ApplTest+D1-OSAppli=0;
OSAppli+D2-OSApHard=0;
Sales=1;
HardTest-D4=0;
OSTest+D4-D1-OSHard=0;
OSHard-D2-SysTest-Market=0;
SysTest-D3=0;
OSApHard+D3-LastTest=0;
LastTest+Market-SalEs=0;
END
```

この問題の解は，図 7･5 のようになる．決定係数の TOTAL, DETAIL, LASTTEST, SALES, OSTEST, OSHARD, SYSTEST がクリティカル・パスになっている．そしてプロジェクトは何も無ければ 70 で終わる．

```
  Global optimal solution found.
  Objective value:                              70.00000
  Total solver iterations:                             0
                       Variable           Value        Reduced Cost
                          TOTAL        1.000000            0.000000
                         DETAIL        1.000000            0.000000
                          APPLI        0.000000            0.000000
                       APPLTEST        0.000000            15.00000
                        OSAPPLI        0.000000            0.000000
                       OSAPHARD        0.000000            0.000000
                       LASTTEST        1.000000            0.000000
                          SALES        1.000000            0.000000
                         OSTEST        1.000000            0.000000
                         OSHARD        1.000000            0.000000
```

SYSTEST	1.000000	0.000000
HARDTEST	0.000000	0.000000
MARKET	0.000000	11.00000
D1	0.000000	10.00000
D2	0.000000	8.000000
D4	0.000000	10.00000
D3	1.000000	0.000000
Row	Slack or Surplus	Dual Price
1	70.00000	1.000000
2	0.000000	31.00000
3	0.000000	-41.00000
4	0.000000	-26.00000
5	0.000000	-18.00000
6	0.000000	5.000000
7	0.000000	8.000000
8	0.000000	19.00000
9	0.000000	-15.00000
10	0.000000	-5.000000
11	0.000000	0.000000
12	0.000000	12.00000
13	0.000000	12.00000
14	0.000000	16.00000

図 7･5　PERT の解

クリティカル・パス上の作業に相当する変数の値が１になっている．最初の２つの制約式「 -TOTAL = -1 と Sales=1」が重要だ．

図 7･6 は，この問題の構造を表している．各行は，あるノードに１の水が入ってこれば，そこから出ていくアクティビティの１つにその水が流れていくことを表している．注意すべきは，制約式の各列には＋１と－１が必ず一対現れていることだ．

制約式中で，各変数が多くとも２つの係数しか持たないことに注目したい．そのうちの１つは +1，もう１つは -1 である．これはネットワーク型の LP の顕著な特徴である．

このような特徴を持つ問題は，整数変数の指定をしなくても，自然に整数解が求められるという特徴を持っている．

```
          A   O L         H
          P O S A       S A
        D   P S A S   O O Y R M
      T E A L A P T S S S S D A
      O T P T P H T A T H T T R
      T A P E P A E L E A E E K
      A I L S L R S E S R S S E D D D D
      L L I T I D T S T D T T T 1 2 4 3
 1: A B 8 8 3 4 4 3 B 5 B B 5     '   MAX
 2:-1     '     '     '     '     '   =-1
 3: 1-1' ' ' ' ' ' ' ' ' ' ' ' ' '=
 4:   1-1 '     ' -1 ' -1 '     '   =
 5:     1-1     '     '     '     '   =
 6: ' ' 1-1' ' ' ' ' ' 1' ' ' '=
 7:       ' 1-1 '     '     '   1 '   =
 8:       '     ' 1   '     '     '   = 1
 9: ' ' ' ' ' ' ' ' ' '1 ' ' -1  '=
10:       '     '   1-1     '-1   1   =
11:       '     '     1-1  -1  -1 '   =
12: ' ' ' ' ' ' ' ' 1' ' ' ' '-1'=
13:       '   1-1     '     '     ' 1 =
14:       '     1-1   '     1     '   =
```

図 7･6　モデルの構造

(2) 技巧的定式化

さて次に２つめの定式化を見てみよう．この定式化の目的関数は，プロジェクトにかかる時間を最小にすることである．このため，アクティビティを中心に考えるのではなく，イベントを中心にする．例えば次のようである．

T1：全体設計が終了する時点

T2：詳細設計が終了する時点

OS1：OSTest と HardTest が終了する時点

変数 T1,T2,…,H5 をこれらのイベントが起こる時点と定義する．すると，次のイベント型のネットワークが作られる．目的関数は次のようになる．

```
Min=H5-T0 ;
```

イベントがおこるのが予定より遅れれば，それに続くイベントも少なくともその時間だけ遅れるという制約がある．そこで各々の作業について１つずつ制約を得る．次が LINGO によるモデル（PERT02.lg4）である．

```
Min=H5-T0;
T1-T0>10;
T2-T1>15;
A1- T2>8;
A2-A1>8;
A3-A2>3;
A4-A3>4;
H4-A4>4
H5-H4>3;
OS1- T2>21;
H2-OS1>5;
H3-H2>12;
H1- T2>11;
H2-OS1>5;
H3-H2>12;
H4-H2>5;
H5-H4>3;
A2-OS1>0;
A3-H2>0;
A4-H3>0;
OS1-H1>0;
```

この問題の解は，**図 7･7** のようになる．

```
LP OPTIMUM FOUND AT STEP     12

        OBJECTIVE FUNCTION VALUE

        1)      70.00000

  VARIABLE        VALUE          REDUCED COST

        H5        70.000000          0.000000

        T0         0.000000          0.000000
```

T1	10.000000	0.000000
T2	25.000000	0.000000
A1	48.000000	0.000000
A2	56.000000	0.000000
A3	59.000000	0.000000
A4	63.000000	0.000000
H4	67.000000	0.000000
OS1	46.000000	0.000000
H2	51.000000	0.000000
H3	63.000000	0.000000
H1	46.000000	0.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	0.000000	-1.000000
3)	0.000000	-1.000000
4)	15.000000	0.000000
5)	0.000000	0.000000
6)	0.000000	0.000000
7)	0.000000	0.000000
8)	0.000000	-1.000000
9)	0.000000	0.000000
10)	0.000000	-1.000000
11)	0.000000	0.000000
12)	0.000000	0.000000
13)	10.000000	0.000000
14)	0.000000	-1.000000
15)	0.000000	-1.000000
16)	11.000000	0.000000
17)	0.000000	-1.000000
18)	10.000000	0.000000
19)	8.000000	0.000000
20)	0.000000	-1.000000
21)	0.000000	0.000000

```
NO. ITERATIONS=      12
```

図 7･7 別の定式化

目的関数値が，クリティカル・パスの長さに等しい．

双対価格が非ゼロの制約に注目することにより，間接的にクリティカル・パス上の作業を求めることができる．つまりこれらの制約に相当する作業は，クリティカル・パス上にある．この一致は偶然ではない．制約の右辺は作業時間である．もしクリティカル・パス上の作業の時間を増やしたならば，プロジェクトの期間も増やすことになる．これは制約の双対価格が非ゼロだからである．この問題の係数行列は，**図 7･8** のようになる．

```
                          O
       H T T T A A A A H S H H H
       5 0 1 2 1 2 3 4 4 1 2 3 1
 1: 1-1   ’     ’     ’     ’ MIN
 2:  -1 1 ’     ’     ’     ’ > A
 3: ’  -1 1  ’  ’  ’  ’  ’  ’ > B
 4:      -1 1   ’     ’     ’ > 8
 5:       ’-1 1 ’     ’     ’ > 8
 6: ’  ’  ’  -1 1  ’  ’  ’  ’ > 3
 7:       ’    -1 1   ’     ’ > 4
 8:       ’     ’-1 1 ’     ’ > 4
 9: 1  ’  ’  ’  ’  -1 ’  ’  ’ > 3
10:      -1     ’     1     ’ > B
11:       ’     ’    -1 1   ’ > 5
12: ’  ’  ’  ’  ’  ’  ’ -1’1 ’ > B
13:      -1     ’     ’     1 > B
14:       ’     ’    -1 1   ’ > 5
15: ’  ’  ’  ’  ’  ’  ’ -1’1 ’ > B
16:       ’     ’   1 ’-1   ’ > 5
17: 1     ’     ’  -1 ’     ’ > 3
18: ’  ’  ’  ’1 ’  ’ -1  ’  ’ >
19:       ’     1     ’-1   ’ >
20:       ’     ’ 1   ’  -1 ’ >
21: ’  ’  ’  ’  ’  ’  1  ’ -1 >
```

図 7･8　別の定式化のグラフ表現

（3）双対問題

この２つの定式化の構造図を比較してみよう．この問題の係数行列は，**図 7･9** のようになる．

```
                    0
    H T T T A A A A H S H H H
    5 0 1 2 1 2 3 4 4 1 2 3 1
 1: 1-1  ’    ’    ’    ’ MIN
 2:  -1 1’    ’    ’    ’ > A
 3: ’  -1 1  ’  ’  ’  ’  ’  ’ > B
 4:     -1 1  ’    ’    ’ > 8
 5:      ’-1 1’    ’    ’ > 8
 6: ’  ’  ’  -1 1  ’  ’  ’  ’ > 3
 7:      ’    -1 1  ’    ’ > 4
 8:      ’    ’-1 1’    ’ > 4
 9: 1  ’  ’  ’  ’  -1 ’  ’  ’ > 3
10:     -1    ’    1    ’ > B
11:      ’    ’    -1 1  ’ > 5
12: ’  ’  ’  ’  ’  ’  ’-1’1 ’ > B
13:     -1    ’    ’    1 > B
14:      ’    ’    -1 1  ’ > 5
15: ’  ’  ’  ’  ’  ’  ’-1’1 ’ > B
16:      ’    ’   1 ’-1   ’ > 5
17: 1    ’    ’  -1 ’    ’ > 3
18: ’  ’  ’  ’1 ’  ’ -1  ’  ’ >
19:      ’    1    ’-1   ’ >
20:      ’    ’ 1   ’  -1 ’ >
21: ’  ’  ’  ’  ’  ’  1  ’ -1 >
```

図 7･9　別の定式化のグラフ表現

（3）双対問題

この２つの定式化の構造図を比較してみよう．１番目の定式化の図を 90° 回転したものが，２番目の図になっている．この２つの定式化は一見なんの関係もないように思えたが，

実は密接な関係がある．数学者はこの関係を「双対関係」と呼んでいる．

7.5　汎用モデル（本節は，LINGO のマニュアルからの転載である）

(1)　新製品の販売

次に，PERT の汎用モデル（PERT03.lg4）を紹介する．

新村コンピュータ社では，製品開発に成功した．そこで，この新製品を売り出そうと考えている．発売予定日に遅れがでないよう，S 社はその予定日までのタスクを PERT で分析したいと考えている．PERT で制限時間内にしなければならない仕事－クリティカル・パスを確認することができ，新製品を適時に売り出すことができる．表 7.3 は，タスクの作業時間である．

表 7.3　タスクの作業時間

タスク	週
デザイン完成	10
需要の予測	14
競合の完了	3
価格設定	3
生産工程スケジュール	7
費用の見積り	4
販売員のトレーニング	10

特定のタスクは，他のタスクが始まる前に終わらせなければいけない．**図 7.10** に，優先順位を表す関係図がある．

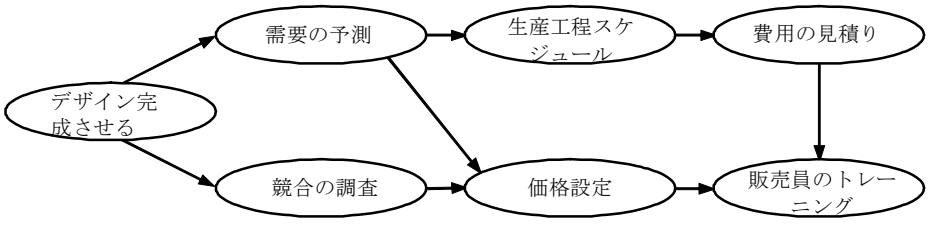

図 7.10　製品発売－優先関係

例えば,「需要の予測」から出ている２つの矢印は「生産工程スケジュール」と「価格設定」以前に「需要の予測」を終わらせなければならないことを示している．目的は，新製品の発売にむけて PERT モデルを使い,タスクのクリティカル･パスを確定することである．

(2)　定式化

プロジェクトのタスクを表すには，原始集合が必要となる．集合の定義をすることで，

モデルに集合を追加することができる.

```
TASKS / DESIGN, FORECAST, SURVEY, PRICE,
SCHEDULE, COSTOUT, TRAIN/: TIME, ES, LS, SLACK;
```

TASKS 集合には，次の４つの属性がある.

TIME　タスクを完了させる時間

ES　タスクを開始する可能な限り早い時間

LS　タスクを開始する可能な限り遅い時間

SLACK　そのタスクの LS と ES の差

TIME はデータとして提示されている. 残りの３つの属性の値はこれから計算できる. もし，あるタスクのスラック時間が 0 ならば，そのタスクは決められた日時に開始しないとプロジェクト全体が遅れることになる. この 0 スラックタイムとなっているタスクが，クリティカル・パスになる.

タスクの開始時を計算するには，優先関係を考える必要があるため，モデルに優先順位の関係を入力しなければならない. 例えば，DESIGN が FORECAST の前に終わっていなければならないという関係を表すと，(DESIGN,FORECAST) となる. TASKS 集合に 2 次元の派生集合を作る事で，優先順位の関係を入力できる. このモデルには，次のようなペアを作成した.

```
PRED(TASKS, TASKS) /
 DESIGN,FORECAST,
 DESIGN,SURVEY,
 FORECAST,PRICE,
 FORECAST,SCHEDULE,
 SURVEY,PRICE,
 SCHEDULE,COSTOUT,
 PRICE,TRAIN,
 COSTOUT,TRAIN /;
```

上の集合は，ただのタスクではなく，(DESIGN, FORECAST)というような 8 個の順位付けられたペアタスクである. よって，この集合には合計 8 個の集合があり，それら全ては優先関係の図にあるアークに該当している.

PRED 集合は，今回強調したい疎な派生集合に明示的リスト法を使っている. これは，集合に含みたいメンバーを全て書き出しているので「明示的」なリストと呼ばれる. しかし，

多くのメンバーを含まなくてはならない場合大変な手間がかかり，不便である．しかし，疎な集合が比較的小さい場合，はっきりと集合メンバーを定義するのがよい．

次に，タスクの開始時間を Data 節で入力する．

```
DATA:
   TIME = 10, 14, 3, 3, 7, 4, 10;
ENDDATA
```

集合とデータが入力されたので定式化を始める．先ず，このモデルには ES，LS，SLACK という計算しなければならない３つの属性があり，面倒なのは ES と LS の計算である．しかし，一度計算されれば SLAC はただの ES と LS の差になる．

ES の計算を考えてみよう．タスクはそのタスクを始める前に完了していなければならない全てのタスクが終わらないと開始できない．よって，最後に完了するタスクが完了する時間がわかれば，開始する時間もわかります．LINGO 形式で表現すると次のようになる．

```
@FOR(TASKS(J)| J #GT# 1:
 ES(J) = @MAX(PRED(I, J): ES(I) + TIME(I)));
```

条件（J#GT#I）を加えているので，最初のタスクの計算を飛ばしたことに注意してほしい．飛ばした理由は，最初のタスクには，それ以前のタスクがないからである．最初のタスクには，次のように任意の開始時間を与える．

LS の計算は多少難しいが ES と似ている．もし，タスク t がこれ以上遅くに開始すると，その後のタスク（少なくても１つ）の一番速い開始時間に始めることを禁止します．これを LINGO で表現すると次のようになる．

```
@FOR(TASKS(I)| I #LT# LTASK:
 LS(I) = @MIN(PRED(I, J): LS(J) - TIME(I)));
```

後に続くべく仕事がないので，最後のタスクの計算は省く．スラック時間は，単に LS と ES の差になる．

```
@FOR(TASKS(I): SLACK(I) = LS(I) - ES(I));
```

開始時間に適当な値を入れる．ここでは 0 を入力する．

```
ES(1) = 0;
```

最後のタスクの一番遅い開始時間以外の変数の値を求める式を入力した．もし，最後のプロジェクトが一番早い時間より遅く始まるとプロジェクト全体に遅れが生じる．

```
LS(7) = ES(7);
```

これでも意味はあるが，これは関係を表すよい表現ではない．モデルにタスクを増すの

に，上記の式の７を新しい数字に代えなければならない．LINGO で使われる集合ベースのモデル言語は，データを式から独立させるためにある．上記のやり方だと，データの独立性が成り立っていないので，下記の方法を用いる．

LTASK = @SIZE(TASKS);

LS(LTASK) = ES(LTASK);

@SIZE は，集合のサイズを返す機能である．この例の場合は「７」を出力する．タスクの数を変更すると，@SIZE の出力も変わる．このように，モデル内の方程式を変えずにデータを保存できる．

PERT の汎用モデル（PERT03.lg4）と**図 7.11** に解を表示する．

```
SETS:
   TASKS / DESIGN, FORECAST, SURVEY, PRICE,
    SCHEDULE, COSTOUT, TRAIN/: TIME, ES, LS, SLACK;
   PRED(TASKS, TASKS) /
    DESIGN,FORECAST,
    DESIGN,SURVEY,
    FORECAST,PRICE,
    FORECAST,SCHEDULE,
    SURVEY,PRICE,
    SCHEDULE,COSTOUT,
    PRICE,TRAIN,
    COSTOUT,TRAIN /;
ENDSETS
DATA:
   TIME = 10, 14, 3, 3, 7, 4, 10;
ENDDATA
@FOR(TASKS(J)| J #GT# 1:
 ES(J) = @MAX(PRED(I, J): ES(I) + TIME(I))
);
@FOR(TASKS(I)| I #LT# LTASK:
 LS(I) = @MIN(PRED(I, J): LS(J) - TIME(I));
);
@FOR(TASKS(I): SLACK(I) = LS(I) - ES(I));
```

```
ES(1) = 0;
LTASK = @SIZE(TASKS);
LS(LTASK) = ES(LTASK);
```

次が解である.

```
Feasible solution found at step:  0

          Variable           Value
             LTASK        7.000000
        ES(DESIGN)       0.0000000
      ES(FORECAST)        10.00000
        ES(SURVEY)        10.00000
         ES(PRICE)        24.00000
      ES(SCHEDULE)        24.00000
       ES(COSTOUT)        31.00000
         ES(TRAIN)        35.00000
        LS(DESIGN)       0.0000000
      LS(FORECAST)        10.00000
        LS(SURVEY)        29.00000
         LS(PRICE)        32.00000
      LS(SCHEDULE)        24.00000
       LS(COSTOUT)        31.00000
         LS(TRAIN)        35.00000
     SLACK(DESIGN)       0.0000000
   SLACK(FORECAST)       0.0000000
     SLACK(SURVEY)        19.00000
      SLACK(PRICE)        8.000000
   SLACK(SCHEDULE)       0.0000000
    SLACK(COSTOUT)       0.0000000
      SLACK(TRAIN)       0.0000000
```

図 7.11　PERT の解

タスクのスラックを見てみると, SURVEY と PRICE は両方ともそれぞれ 19 週と 8 週という開始時点に余裕があることがわかる. プロジェクト全体の完了予定日を遅らせることなくスラックの分だけSURVEYとPRICEの開始時に遅れが出てもかまわない.DESIGN,FORECAST,

SCHEDULE, COSTOUT と TRAIN は，SURVEY や PRICE と違いスラック値が 0 である．これらのタスクがプロジェクトのクリティカル・パスを構成しているので，開始が遅れるとプロジェクト全体に影響する．プロジェクトマネージャーは，こういったクリティカル・パスが決められた日時に開始して予定された時間どおりに完了するよう気を配らなければいけない．最後に，ES(TRAIN)の値が 35 であるということは，この新製品の販売開始まで 45 週間かかることを示す．35 週間ではなく，45 週間なのは，要員のトレーニング開始まで 35 週間かかり，トレーニング自体が 10 週間なので，合計 45 週間かかるということを示している．

(3)　汎用化

それでは，モデルからデータを切り離し，汎用化（PERT04.lg4）する．データは，**表 7.4** のように Excel 上に定義してある．B4:C11 に PRED，E4:E10 に TASKS，F4:F10 に TIME というセル名を与える．

	A	B	C	D	E	F
1						
2						
3						
4		DESIGN	FORECAST		DESIGN	10
5		DESIGN	SURVEY		FORCAST	14
6		FORECAST	PRICE		SURVEY	3
7		FORECAST	SCHEDULE		PRICE	3
8		SURVEY	PRICE		SCHEDULE	7
9		SCHEDULE	COSTOUT		COSTOUT	4
10		PRICE	TRAIN		TRAIN	10
11		COSTOUT	TRAIN			

図 7.12　Excel 上のデータ

```
SETS:
   TASKS / DESIGN, FORECAST, SURVEY, PRICE,
    SCHEDULE, COSTOUT, TRAIN/: TIME, ES, LS, SLACK;
   PRED(TASKS, TASKS) ;
ENDSETS
DATA:
   TASKS=@OLE( );
   PRED= @OLE( );
   TIME = @OLE( );
ENDDATA
@FOR(TASKS(J)| J #GT# 1:
 ES(J) = @MAX(PRED(I, J): ES(I) + TIME(I))
```

```
);
@FOR(TASKS(I)| I #LT# LTASK:
 LS(I) = @MIN(PRED(I, J): LS(J) - TIME(I));
);
@FOR(TASKS(I): SLACK(I) = LS(I) - ES(I));
ES(1) = 0;
LTASK = @SIZE(TASKS);
LS(LTASK) = ES(LTASK);
```

8　判別分析のニューフェース（SVM）

8.1　統計手法も数理計画法の領域だ

私は，2008 年の年賀状に「統計を卒業し，今後の人生を数理計画法の普及に専念する」という内容の抱負を知人に出した．しかし，よく考えてみると，統計の研究に LINGO が一番役に立つことを十分認識していなかった．何しろ数式で定式化できるもの全てが数理計画法の対象である．回帰分析，判別分析，コンジョイント分析の雛形モデルも LINGO のマニュアルにすでに紹介されている．結局，汎用統計ソフトでカバーできないものを LINGO で行えばよいことを本書で紹介したい．

早くから，線形回帰分析が数理計画法で定式化できることは日本オペレーションズ・リサーチ学会の発表で行ってきた．詳細は省くが，最小二乗法で求める回帰分析は，2 次計画法で定式化できる．LAD(Least Absolute Deviation)回帰分析は，誤差の絶対値の和を最小化する回帰手法であり LP モデルになる.SAS の創業者の一人で,パソコン統計ソフト JMP の開発者である J.Sall 博士の拙著翻訳本『SAS による回帰分析入門(朝倉，1983)』は回帰分析の名著である．再販されないので，同氏から私が自由に書き直して出版することの了解を取っている．この中で，L1 ノルムすなわち LAD 回帰をより汎用化した Lp ノルム回帰分析が紹介されている．これらは，数理計画法のモデルそのものである．

そこで，1980 年代から数理計画法を用いた判別分析の研究も数多く行われてきた．一番研究成果の多いのは，LP を用いた Lp ノルム判別分析モデルである．私は，1996 年から整数計画法を用いて，誤分類数最小化基準(Minimum Misclassification Number, MMN)による最適線形判別関数（Optimal Linear Discriminant Function, OLDF)の研究を行ってきた．実は，確率分布を基準にしてすでに数多くの成果の出ている判別分析であるが，整数計画法による組み合わせアプローチで,びっくりするような面白い結果が得られている.今回，そのさわりの紹介をしようかと思ったが，別の機会にしたい．その理由は，まだ学会で広く認知されていないものを，読者に中途半端に紹介するのもどうかと思ったわけである．

そこで,1990 年代以降,判別分析のニューフェースとして注目を集めている SVM (Support Vector Machine)を紹介する．

8.2　SVM の考え方　－マージン概念－

SVM は，変なネーミングである．統計的な背景でなく，パターン認識の分野でも文字認識などで，判別やクラスター分析と同じことが研究されてきた．この分野では最終的にパターン認識する機械を作ることに最終目標があり機械という言葉が自然に出てくるのであ

ろう.

この手法は，マージン最大化という面白い概念と，Kernel トリックという信じられないマジックでこの分野の研究者による強烈な宗教集団を作っている.

それまでの Lp ノルム研究では，時代背景もあるが，単にモデルの提案で終わり実データによる検証がほどんと行われてこなかった．これに対し SVM は，開発されたモデルが「どんな新しい知見があるのか?」,「役に立つのか?」といった視点でアプローチしており，私も大いに共感を覚えるが，ある点では批判的でもある.

SVM は，3 つのカテゴリーがある．導入は，判別対象が線形分離可能な場合である．すなわち，線形判別関数で誤分類数 0 の場合である．私のような，統計から出発すると，線形分離可能な場合は余りにも簡単なので，考慮の対象でなかった．SVM の創始者は，この問題に対し，パターン認識で考えられてきた「マージン概念」を取り入れ，図 8.1 のような「マージン最大化」ということを提案した．これをハードマージン最大化 SVM という.

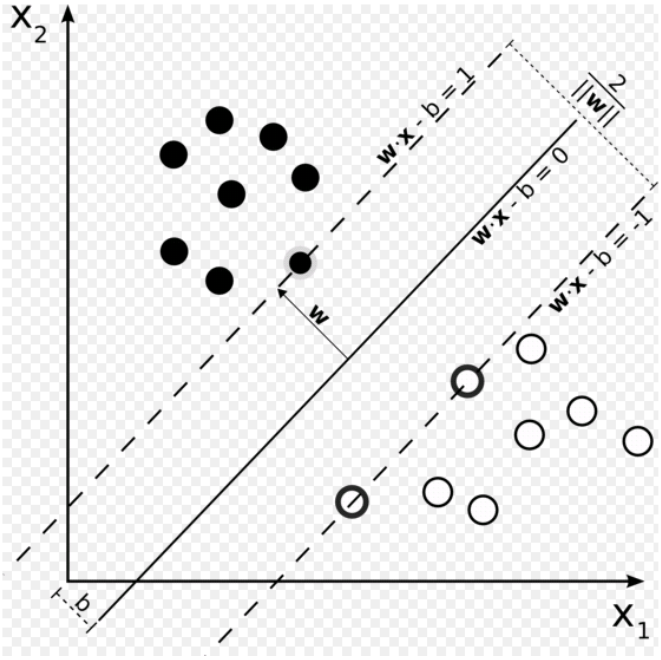

図 8.1　マージン最大化（Wikipedia より引用）

判別分析は，次のような 1 次式で表される判別関数を考える.

$$y = f(\mathbf{x}) = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_p x_p$$

判別すべき 2 群を SVM の用語に従い，クラス 1 とクラス 2 と呼ぶことにする．クラス 1 に属するケース $\mathbf{x}_1$ が $y = f(\mathbf{x}_1) > 0$ であれば，正しくクラス 1 に判別され，$y = f(\mathbf{x}_1) < 0$ であれば，クラス 2 に誤判別されたと考える．一方，クラス 2 に属するケース $\mathbf{x}_2$ が $y = f(\mathbf{x}_2) < 0$ であれば正しくクラス 2 に判別され，$y = f(\mathbf{x}_2) > 0$ であればクラス 1 に誤判別されたと考える．ここで y_i がクラス 1 の場合 1 とし，クラス 2 の場合 - 1 とすれば，次の定式化で両ク

ラスの不等号の向きの不一致が解消でき，0 以上であれば正しく判別されたと考えることができる．

$y_i * f(\mathbf{x_i}) > 0$

データが線形分離可能であれば,それらを分離する判別超平面は一意に決まらないので，サポートベクター間の距離を最大化する．ただし，判別スコアが 0 になる判別超平面上のケース $\mathbf{x_i}$（$f(\mathbf{x_i})=0$）をどう扱うかは，統計や数理計画法による判別研究で考慮されてこなかった．

8.3　ハードマージン最大化 SVM

SVM は，パターン認識で古くから考えられてきた「マージン概念」を取り入れている．2 群が線形分離可能であれば，判別超平面の両側にサポート・ベクター（SV）と呼ばれる超平面を設けて，データ空間をクラス 1 だけが含まれる空間，クラス 1 も 2 もまったく含まない空間，クラス 2 だけが含まれる空間の 3 つに分割できる．ここで，クラス 1 と 2 の SV は，必ず各 SV 上に少なくとも 1 ケース以上のケースが含まれている．

SVM は，この 2 つの SV の距離を最大化するものを選ぶ．これをマージン最大化 SVM という．さて，このマージンの距離は，図に書き込んだように，判別係数を $\mathbf{w}=(a_1, a_2, \cdots, a_p)$ とすれば，$2/\|\mathbf{w}\| = 2/\mathrm{SQRT}(a_1^2 + a_2^2 + , \cdots + a_p^2)$ になる．最初，どうしてこうなるのか分からずあせったが，しばらくして高校数学で習った「判別超平面上にある点$(x_1, x_2, \cdots, x_p)$から SV（$y = a_0 + a_1x_1 + a_2x_2 + \cdots + a_px_p$）の張る（超）平面に降ろした場合の次の距離の公式」を適用すればよいことに気づいた．

$|a_0 + a_1x_1 + a_2x_2 + \cdots + a_px_p| / \mathrm{SQRT}(a_1^2 + a_2^2 + , \cdots + a_p^2)$

そして，SV が判別超平面からの距離が 1 になるように制限する．これによって分子が 1 になる．結局マージン最大化 SVM は次のように定式化できる．

$\mathrm{MAX} = 2/\ \mathrm{SQRT}(a_1^2 + a_2^2 + , \cdots + a_p^2)$

$y_i * f(\mathbf{x_i}) > 1 \quad i=1, \cdots, n$

しかし,このままでは非線形最適化になるので,実際には次のように最小化問題で扱えば，非線形計画法より扱いやすい 2 次計画法になり，計算も容易になる．

$\mathrm{MIN} = (a_1^2 + a_2^2 + , \cdots + a_p^2)/2$

$y_i * f(\mathbf{x_i}) > 1 \quad i=1, \cdots, n$

多くの現実の問題では線形分離可能な場合はまれである．この場合，幾つかのケースが SV の反対側にくることを許す.すなわち 1 以上という拘束を少し緩めて$(1-e_i)$にしてやる．これによって SV で正しく判別されるケースのマージンを最大化し,判別されないケースの

誤差の和（Σe_i）を最小化してやればよい．この様に最適化したい基準が 2 つ以上あれば多目的最適化という．一番単純なやり方は，これらの多目的な基準の加重和を求め，見かけ上単一目的化することである．これをソフトマージン最大化 SVM という．そこで，誤差の和に重み c をかける．これは一種のペナルティであり，SVM では「ペナルティ C」と呼ばれている．本当は，ポートフォリオ分析で紹介した，リターンを制約式に取り込み，リスクを目的関数にするほうが理にかなっていると考えている．この点を指摘しても，SVM の研究者からは無視かブーイングされる．

$$MIN=(a_1^2+a_2^2{}_+,\cdots+a_p^2)/2+c*\Sigma e_i$$

$$y_i * f(\mathbf{x_i}) > 1-e_i \qquad i=1,\cdots,n$$

このあと，SVM はさらに進化し，p 次元のデータ空間にあるケースを無限次元の空間に変換し，できるだけ線形分離可能にするよう努力して元の空間に戻せば，誤分類数を少なくできる.これが Kernel トリックと呼ばれ,多くの優秀な研究者をとりこにしているようだ.しかし,私は単にデータにカーブフィッティングしているだけでないかと疑っているため,私自身の頭と行動はソフトマージン最大化 SVM で停滞したままである.

この点に関しては，この分野の研究を行っておられる佐藤義治北海道大学教授の研究でも指摘されている.

8.4　モデル

次がソフトマージン最大化 SVM の LINGO によるモデル（SVM01.lg4）である．教師データ（訓練データ，内部標本）として 40 人の「学生データ」を用いる．説明変数は,「勉強時間と支出」の 2 変数である（定数項を集合節で最後に定義している）．すなわち，40 制約式で 2 変数であり，双対問題に治すと 2 制約式 40 変数になり，計算が早くなる．25 人の試験の合格をクラス 1 とし，15 人の不合格者をクラス 2 とする．このデータは，SAS の入門書である『統計処理エッセンシャル(丸善)』,SPSS の入門書である『SPSS　for　Windows 入門（丸善)』, JMP の入門書である『JMP 活用　統計学とっておき勉強法(講談社)』で用いている.

評価用データとして，このデータから Speakeasy で一様乱数を発生させ，各クラス 1 万件の合計 2 万件のデータを作成した.

```
MODEL:
SETS:
 P/X1..X3/: VAR;
 N/1..40/:E,SCORE;
```

```
 D(N,P):IS;
ENDSETS
DATA:
 IS=@OLE( );
ENDDATA
SUBMODEL sub1:
 MIN=OBJ ;
  OBJ = SVM1/2+c*SVM2;
  SVM1=@SUM(P(j)| j #NE# pn : VAR(j)^2) ;
  SVM2= @SUM(N(i):E(i));
 @FOR(N(i): @SUM(P(j):IS(i,j)*VAR(j)) > 1-E(i));
 @FOR(P(j):@FREE(VAR(j)));
ENDSUBMODEL
CALC:
 !@SET('DEFAULT');@SET('TERSEO',2);
 !C=0.1;
 !pn=@size(p);
 !MNI=0;
 @solve(sub1);
  !@FOR(N(i): SCORE(i)=@SUM(P(j):IS(i,j)*VAR(j)));
  !@FOR(N(i):  @IFC(SCORE(i) #LT# 0 : MNI=MNI+1));
ENDCALC
DATA:
 !@OLE( )=MNI;
 !@OLE( )=SVM1;
 !@OLE( )=SVM2;
 @OLE( )=VAR;
ENDDATA
END
```

SUBMODEL節では，SVMのモデルを定義している．SVM1では目的関数のマージン最大部分を計算している．SVM2はSVの反対側にくるケースのSVからの距離の和である．そしてペナルティCでもってこれらが結合され，単目的化されている．SETS節で集合Pは2個の判別係数と

定数項を含む3個の要素をもっている．「SVM1=@SUM(P(j)| j #NE# pn : VAR(j)^2) ;」の「j #NE# pn」は，CALC節でpnは最後の係数すなわち3番目の定数項を省いて，判別係数だけの自乗和をとっている．このような論理式で，部分集合だけの演算が容易に行える．

OBJ=SVM1+C*SVM2;

集合Nは，40件の学生を表す．集合D(N,P)は，行が集合Nを，列が集合Pから作られる派生集合で，40行3列のデータの2次元配列ISを定義している．この配列は，次のDATA節でExcelからデータが入力される．

次の制約式「@FOR(N(i): @SUM(P(j):IS(i,j)*VAR(j)) > 1-E(i));」は，SVが判別超平面から距離が1だけ離れていることを示す．

次の「@FOR(P(j):@FREE(VAR(j)));」は，判別係数VAR(j))が負の値もとるので，自由変数に指定している．

CALC節の最初の「@SET('DEFAULT');@SET('TERSEO',2);」は，LINGOの出力を抑える命令である．考えてもみてほしい,変数は3個と少ないが，教師データは40個，評価データは2万個の制約式をもっている．最初，ｃを10^{6}から10^{-6}までの12段階で変えて一気に教師データで線形判別関数を求め,それを2万件の評価データに適用し誤分類数を計算するモデルを作った．しかし，どうも計算が10分以上かかりおかしい．様子を見ているとまず分析後，結果の出力準備に異常な時間がかかっている．これは減少費用や双対価格などを計算する必要があるためである．また，数理計画法の出力に合わせると各変数や制約は1行に表示されるので膨大な頁数になる．そこで2008年3月1日（土）に最初に構想したモデルを2日(日)の朝あきらめ．教師データをExcelから読み込み，判別係数をExcelに出力し，Excel上で誤分類数を計算することに変更した．またCの値も「C=0.1;」のようにその都度分析結果を見ながら行うことにした．これで，各計算は1秒以内になった．

「！」をつけると「；」までがコメントになる．最初はこれをとって計算結果をExcelと比較する．結果が同じであるので,図のようにコメント化するか省けばよい．

「pn=@size(p);」で集合Pの要素数3をpnに割り当てている．「MNI=0;」は，Excelで計算する誤分類数を初期値0に設定している．最初にモデルを開発する場合，LINGOとExcelの両方で計算結果をチェックするように慎重にした方がよいだろう．「 @solve(sub1);」で，サブモデルを計算している．次の2つの文は，教師データの誤分類数を計算している．Excelの結果と付き合わせ同じなので，省いても良い．

@FOR(N(i): SCORE(i)=@SUM(P(j):IS(i,j)*VAR(j)));

@FOR(N(i): @IFC(SCORE(i) #LT# 0 : MNI=MNI+1));

次のDATA節で，計算結果MNI，SVM1，SVM2，VARをExcelの同名のセルに出力する．

図8.2はExcel上のデータの一部である．セルA1：C1に判別係数が入っている．その下に2万件の評価データが入っている．ただしクラス2のデータはy_i=-1を掛ける代わりにデータすべてに－1をかけてある．H2:H20001は判別スコアの計算式が入っていて，I2:I20001でその値が負(誤分類される)の場合を1に，それ以外を0にしている．I1は誤分類数の和である．

J列からL列には，40件の教師データが入っていて，後ろ15件はマイナス1をかけてある．結局読者は，このようなデータを準備さえすれば，SVMを堪能できる．

	A	B	C	H	I	J	K	L	M	N
1	0.02	-0.02	0.92	1201.63	3770	0.02	-0.02	0.92	11.55198	15
2	8	3	1	0.10731	0	9	2	1	1.06829	0
3	7	3	1	0.08536	0	12	4	1	1.088615	0
4	3	3	1	-0.0024	1	7	3	1	1.001626	0
5	5	2	1	0.06423	0	10	2	1	1.090241	0
6	6	3	1	0.06341	0	7	3	1	1.001626	0
7	3	5	1	-0.048	1	7	3	1	1.001626	0
8	7	3	1	0.08536	0	6	5	1	0.934149	0
9	10	2	1	0.17398	0	3	3	1	0.913824	0
10	7	3	1	0.08536	0	6	3	1	0.979675	0
11	3	3	1	-0.0024	1	8	3	1	1.023576	0
12	3	3	1	-0.0024	1	5	4	1	0.934961	0

図 8.2　データ例（SVM02.xls）

さすがExcelである．LINGOの計算が1秒以内で終わり，Excelに切り替えると40件と2万件の計算が終わっている．表計算に関する計算速度については脱帽である．結局計算時間がかかると思えば，幾つかのソフトウエアを組み合わせて用いるべきである．実際これまで私はExcelのアドインソフトウエアであるWhat’sBest!を用いてSVMを研究してきた．大規模な表形式のデータを分析対象とする場合，LINGOよりExcelのアドインであるWhat’sBest!の方が適しているのかも知れない．私は，この1月からLINDO Systems Inc.の日本の代表になったので，LINGOとWhat’sBest!を使い分けできる．その点で，研究費の乏しい特に文科系の研究者は気の毒だと思っている．学部や研究科単位でサイトライセンスすれば，最上位版でも1台1万円程度で導入できるように価格を設定した．

表 8.1　誤分類数の比較

c	MNI	MNE	SVM1	SVM2	X1	X2	C
1E+06	6	2556	0.50	14.50	0.50	-0.50	0.50
1E+05	6	2556	0.50	14.50	0.50	-0.50	0.50
10000	6	2556	0.50	14.50	0.50	-0.50	0.50
1000	6	2556	0.50	14.50	0.50	-0.50	0.50
100	6	2556	0.50	14.50	0.50	-0.50	0.50
10	6	2556	0.50	14.50	0.50	-0.50	0.50

1	6	2556	0.50	14.50	0.50	-0.50	0.50
0.1	6	4002	0.31	15.00	0.50	-0.25	-0.75
0.001	5	2556	0.08	19.00	0.20	-0.20	0.40
1E-04	15	3770	0.00	28.49	0.03	-0.03	0.92

表 8.1 の「MNI 列」は教師データの誤分類数を表す．40 件なので，2.5 倍すれば誤分類率(%)になる．15%から 42%の間でばらついている．「MNE」は評価データにより誤分類数である．2 万件なので 200 で割れば誤分類率(%)になる．評価データでは 13%から 20%の間でばらついている．教師データは，元青山学院大学国際政治経済学部の高森教授が作成したデータで，5 点刻みで重なりもありこのような結果になった．

手法の評価は，研究分野では乱数などがこのまれている．再現性があり結果に矛盾の少ないデータであるためだ．さらに，実データ，実データを模倣した今回のような良くできたデータなど，色々な種類のデータで検証すべきであろう．

SVM では，多くの場合，評価データで判別成績の良い結果を選ぶことになる．今回の例では，C=0.001 の教師データで誤分類数 5(12.5%)，評価データで誤分類数 2556 件(12.78%)が良いことになる．教師データでも評価データでもそれほど誤分類確率に差がないのでめでたしとなる．

しかし，私は SVM が C の値を色々変えて探索することは，結局は場当たり的に色々な判別手法を試し，その中で評価データによる判別結果が良いものを選んでいるので，判別成績は良くなって当たり前だと思う．しかし，このような指摘は宗教信仰者の前に通用しない．

8.5 多目的最適化の扱い

SVM が 2 目的最適化を恣意的な重みを掛け単目的化していることは問題であろう．それは，表 8.1 をみれば分かるとおり，C を 1000000 から 1 まで 7 段階で変えても同じ結果になり，非効率である．また，単位の異なるものを単一目的関数に合成するのは間違いである．表の SVM1 と SVM2 をプロットすれば分かるとおり，SVM1 が増えれば，SVM2 は減少するという効率的フロンティアが得られる．

マージンは 0 から 0.5 で増加すると SV の反対側にくるケースの距離の和は減少する．そこで次のようにモデル（SVM03.lg4）を修正し，マージンを制約式に取り込み 0 から 0.5 の間で変えてみる．目的関数は，SV の反対側にくるケースの距離の和の最小化である．

```
MODEL:
SETS:
```

```
 P/X1..X3/: VAR;
 N/1..40/:E,SCORE;
 D(N,P):IS;
ENDSETS
DATA:
 IS=@OLE( );
ENDDATA
SUBMODEL sub1:
 MIN=@SUM(N(i):E(i));
 SVM2=@SUM(N(i):E(i));
 @SUM(P(j)| j #NE# pn : VAR(j)^2)<=C;
 @FOR(N(i): @SUM(P(j):IS(i,j)*VAR(j)) > 1-E(i));
 @FOR(P(j):@FREE(VAR(j)));
ENDSUBMODEL
CALC:
 @SET('DEFAULT');@SET('TERSEO',2);
 C=0.001;
 pn=@size(p);
 MNI=0;
 @solve(sub1);
  SVM1=@SUM(P(j)| j #NE# pn : VAR(j)^2);
  @FOR(N(i): SCORE(i)=@SUM(P(j):IS(i,j)*VAR(j)));
  @FOR(N(i):  @IFC(SCORE(i) #LT# 0 : MNI=MNI+1));
ENDCALC
DATA:
 @OLE( )=MNI;
 @OLE( )=SVM1;
 @OLE( )=SVM2;
 @OLE( )=VAR;
ENDDATA
END
```

そして，マージンを 0.5 以上から 0.1 刻みで 0 まで下げていくと表 8.2 のようになる．

表 8.2　SVM の目的関数を訂正する

SVM1	MNI	MNE	SVM1	SVM2	X1	X2	C
0.5	6	2556	0.50	14.50	0.50	-0.50	0.50
0.4	6	3421	0.40	14.73	0.50	-0.39	-0.06
0.3	7	4389	0.30	15.07	0.48	-0.26	-0.64
0.2	7	3851	0.20	15.88	0.34	-0.29	0.02
0.1	6	3851	0.10	18.17	0.23	-0.22	0.27
0.08	6	3851	0.08	19.00	0.20	-0.20	0.40
0.06	6	3770	0.06	20.47	0.17	-0.18	0.42
0.04	12	3770	0.04	22.22	0.14	-0.14	0.59
0.02	13	3770	0.02	24.50	0.10	-0.10	0.63
0.01	15	3770	0.01	26.11	0.07	-0.07	0.74
0.001	15	3770	0.00	28.77	0.02	-0.02	0.92

結果は表 8.1 と余り代わり映えがしないが，SVM1 と SVM2 の組み合わせは，表 8.1 より大きくばらついている．結局，このデータではそれほど大きな違いがないことが表 8.2 で確認できる．

9　評価の科学（松井の年俸は高すぎる？）

9.1　経営効率性分析あるいは包絡分析法とは

包絡分析法（Data Envelopment Analysis, DEA）は，企業の事業部，百貨店などの複数の店舗，自治体の複数の図書館などの各種事業体の意思決定主体（Decision Making Unit, DMU)の効率性を評価する手法である．あるいは，野球選手の年俸が成績とどう関係しているか評価できる．1978 年にテキサス大学の Charnes,Cooper and Rhodes によって提案されたので，CCR モデルと呼ばれている．

評価手法としては，回帰分析や判別分析などの手法が思いつく．これらは，分析対象のデータに共通の重みをつけて行われる．

しかし，DEA 法はガソリンや電機で動くモーターをイメージすればよい．入力であるガソリンに対し，モータの出力はロスが発生するので，出力／入力の比は 1 以下になる．1 に近いほど，エネルギー効率が良いモータと考えられる．この場合，1 入力 1 出力であるが，多くの場合は多入力多出力であることが多い．例えば野球選手を考えてみよう．出場回数を入力とすれば，ヒット数や得点数など複数の出力項目が考えられる．そこで，m 個の入力項目を $x_1,....,x_m$ とし，n 個の出力項目を $y_1,...,y_n$ とする．そして，k 人の選手（DMU）がいるとする．I 番目の選手(DMUi)の測定値を $x_{1i},....,x_{mi}$ とし，n 個の出力項目を $y_{1i},...,y_{ni}$ とする．この場合，DMU_h の効率性は次の比になる．

$DMU_h = (b_1*y_{1h}+・・・+b_n*y_{nh})/(a_1*x_{1h}, +.... +a_m*x_{mh})$

ここで，この比が 1 以下になるように重み$(b_1,・・・, b_n)$と$(a_1, x_1,...., x_m)$を決めてやればよい．しかし，重みを DMUi に無関係に一定(固定)にすれば，これまでの統計アプローチになる．DEA 法の一番の特徴は，DMUi ごとにその効率を最大になるように個別の重みを与える点である．しかし，その重みを他の DMU にも適用し，その効率を 1 以下にするという制約を課すことにする．ここで h 番目の DMU_h を考える．

$MAX=(b_1*y_{1h}+・・・+b_n*y_{nh})/(a_1*x_{1h}, +.... +a_m*x_{mh})$;

$(b_1*y_{1p}+・・・+b_n*y_{np})/(a_1*x_{1p}, +.... +a_m*x_{mp})<=1$;　for　p=1,…,k

このモデルは，数理計画法で分数計画法と呼ばれている．これを次の同値な LP モデルに置き換えたものが一般的に CCR モデルとして知られている．

$MAX=b_1*y_{1h}+・・・+b_n*y_{nh}$;

$a_1*x_{1h}, +.... +a_m*x_{mh}=1$;

$(b_1*y_{1p}+・・・+b_n*y_{np})<=(a_1*x_{1p}, +.... +a_m*x_{mp})$;　for　p=1,…,k

ここで注意したいことは，各 DMU に対して上の LP モデルを解く必要がある．すなわち，k

人の選手（DMU）に対してｋ個の異なったLPモデルを繰り返し解く必要がある．

私はこれまでDEA法と統計手法の比較を行ってみたいと思っていた．しかし，今回取り上げる67人の野球選手の場合,67個の異なったLPモデルを検証することに躊躇していた．それがLINGOを用いれば簡単にDEAの汎用モデルで検証できる．わずか2日間で行えた．DEAに関しては，Excelなどのモデルが公開されている．人生でたった1度しかない貴重な時間を，わずかなお金の節約のため浪費するのはいかがなものかと思う．

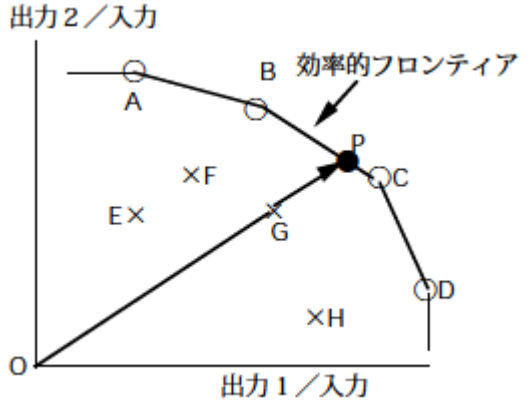

図 9．1　1入力2出力

DEA法の特徴を，図9.1の1入力2出力を例にして説明する．例えば，野球選手の場合，打席数を入力とし，得点と年俸を出力と考えればよい．各DMUpに対してCCRモデルを1回解く．A，B，C，Dの効率値が1で，Gは例えば0.7であったとする．効率値が1のA，B，C，Dを結んだ線分は，全体として凸体になり，効率的フロンティアと呼ばれている．どのような重み付けを行っても，考えている全てのDMUはこの凸体に内包される．これが包絡と呼ばれるゆえんである．

DMUgは，Gの良いところを取り入れて重み付けしても非効率的であり，出力を改善する必要がある．すなわちP点が努力目標になる．

一方，次のようなモデルを考えることができる．

$$MIN=b_1*y_{1h}+\cdots+b_n*y_{nh}\ ;$$

$$a_1*x_{1h},\ +\ldots.\ +a_m*x_{mh}=1\ ;$$

$$(b_1*y_{1p}+\cdots+b_n*y_{np})>=(\ a_1*x_{1p},\ +\ldots.\ +a_m*x_{mp})\ ;\quad for\quad p=1,\cdots,k$$

このモデルは，図9･2に示すように，非効率的なフロンティアを見つけてくれる．出力をどこまで落とせばもっとも非効率であるかが分かる．A，B，Cの非効率値は1であり，Gは1.5だったとしよう．これが図のP点まで非効率的になれば，1に下がる．CCRモデルで効率的なDMUに飴（評価をあげる），逆CCRモデルで非効率的なDMUに鞭（評価を下げる）の様な使い分けをすればよい．

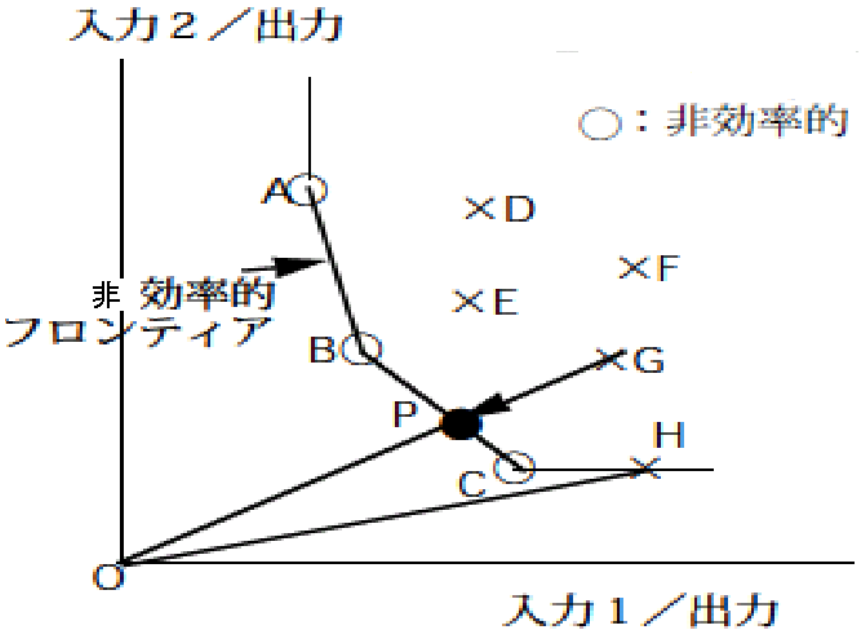

図 9・2　逆 CCR モデル

9.2　LINGO の汎用モデル

図 9・3 は LINGO の汎用モデルである．今，後で紹介する 67 人の野球選手の打率，本塁打，四球を入力とし，推定年俸を出力と考えている．SUBMODEL 節で CCR モデルを DEA1 で，逆 CCR モデルを DEA2 で定義している．DEA も双対モデルのほうが早く計算できる．

```
MODEL:
! Data Envelopment Analysis of Decision Maker Efficiency ;
! Keywords: DEA, Data Envelopment Analysis;
SETS:
  DMU/1..67/: !The decisionmaking units;
    SCORE,SCORE2; ! Each decision making unit has a
                  score to be computed;
  FACTOR/1..4/: TW,TW2;
! There is a set of factors, input & output;
  DXF( DMU, FACTOR):  F, ! F( I, J) = Jth factor of DMU I;
          W, WW2;  ! Weights used to compute DMU I's score;
ENDSETS
 DATA:
```

```
  !F=@OLE();
! Inputs are spending/pupil, % not low income;
! Outputs are Writing score and Science score;
  NINPUTS = 3;  ! The first NINPUTS factors are inputs;
 !FACTOR= COST RICH     WRIT SCIN;
!      The inputs,    the outputs;
  F=@OLE( );
  WGTMIN = .00004; ! Min weight applied to every factor;
  BIGM = 999999;  ! Biggest a weight can be;
 ENDDATA
!-------------------------------------------------------;
! The Model;
SUBMODEL DEA1:
 ! IU = DMU we are currently considering;
! Try to make the score of DMU IU as high as possible;
  MAX = TSCORE;
    TSCORE = @SUM( FACTOR(J)|J #GT# NINPUTS:
                          F(INOW, J)* TW( J));
! Sum of inputs(denominator) = 1;
   [SUM21] @SUM( FACTOR( J)| J #LE# NINPUTS:
                     F( INOW, J)* TW( J)) = 1;
! Using DMU IU's weights, no DMU can score better than 1
  Note, Numer/Denom <= 1 implies Numer <= Denom;
   @FOR( DMU( K):
   [LE1]     @SUM( FACTOR( J)| J #GT# NINPUTS: F( K, J) * TW( J))
          <= @SUM( FACTOR( J)| J #LE# NINPUTS: F( K, J) * TW( J))
       );
! The weights must be greater than zero;
  @FOR( FACTOR( J): @BND( WGTMIN, TW, BIGM));
ENDSUBMODEL
CALC:
   !@SET( 'TERSEO', 2);
```

```
   @SET( 'STAWIN', 0);
   ! Solve the DEA model for every DMU;
   @FOR( DMU( IU):
      INOW = IU;
      @SOLVE(DEA1);
     ! Store the results;
      SCORE( IU) = TSCORE;
      @FOR( FACTOR( J):
         W(IU,J) = TW( J)
      );
   );
ENDCALC
DATA:
@OLE( )=score;
@OLE( )=w;
ENDDATA
 SUBMODEL DEA2:
 ! IU = DMU we are currently considering;
! Try to make the score of DMU IU as high as possible;
  MIN = TSCORE;
    TSCORE = @SUM( FACTOR(J)|J #GT# NINPUTS:
                          F(INOW, J)* TW2( J));
! Sum of inputs(denominator) = 1;
   [SUM22] @SUM( FACTOR( J)| J #LE# NINPUTS:
                     F( INOW, J)* TW2( J)) = 1;
! Using DMU IU's weights, no DMU can score better than 1
  Note, Numer/Denom <= 1 implies Numer <= Denom;
   @FOR( DMU( K):
   [LE2]     @SUM( FACTOR( J)| J #GT# NINPUTS: F( K, J) * TW2( J))
          >= @SUM( FACTOR( J)| J #LE# NINPUTS: F( K, J) * TW2( J))
       );
! The weights must be greater than zero;
```

```
  @FOR( FACTOR( J): @BND( WGTMIN, TW2, BIGM));
ENDSUBMODEL
DATA:
@OLE( )=score2;
@OLE( )=WW2;
ENDDATA
END
```

図9・3　LINGOの汎用モデル

9.3　分析データ

私の情報科学Ⅱは,成蹊大学経済学部で2年次に配当されている半期2単位の科目である.一つのデータで，統計ソフトの分析方法を教えている．評価は，各自に自分でテーマを設定し,データを集め,統計ソフトで分析し,20頁以上の統計レポートの提出を課している.2001年度の受講生に，田中祐輔君がいた．彼は野球選手のデータを集め，推定年俸との関係を調べたレポートを提出してくれた．ここでは，彼の集めたデータを用いる.

データは，67名の野球選手名，チーム，リーグ(セリーグを１，パリーグを0)，打率，試合出場数，打数，得点，安打，２塁打，3塁打，本塁打，打点，盗塁，四球，三振と推定年俸である.

これをJMPで推定年俸を目的変数にして,リーグ以降の13変数を用いて変数増加法を行うと，四球，打率，リーグ，本塁打の4変数が順にモデルに入ってきて停止する．次に，13変数から変数減少法を行うと四球，リーグ，本塁打の3変数がモデルに残る．リーグのような0/1の2値変数はDEAでは取り扱い上問題を起こすので，恣意的であるが四球，打率，本塁打の3変数を入力項目と考え，推定年俸を出力と考えDEAで評価することにする.

DEA法は，入出力の重みは原則正とし,データも正の値が大きいほど入力と出力のパワーが大きいことを想定している.

これらの値を67行4列のセルに入れ,セル名をFとした.**表9.1**に使用したデータを示す.

表9.1　使用データ

選手	チーム	リーグ	打率	本塁打	四球	推定年俸
福浦	ロ	0	0.346	18	58	5400
小笠原	日	0	0.339	32	63	11000
松中	ダ	0	0.334	36	57	15000
松井	巨	1	0.333	36	120	50000

ﾛｰｽﾞ	近	0	0.327	55	83	17850
谷	オ	0	0.325	13	65	8000
古田	ヤ	1	0.324	15	43	20000
ﾍﾟﾀｼﾞｰﾆ	ヤ	1	0.322	39	120	21000
中村	近	0	0.320	46	104	30000
礒部	近	0	0.320	17	52	4000
鈴木	横	1	0.315	6	51	18000
金本	広	1	0.314	25	128	22200
真中	ヤ	1	0.312	7	38	8450
稲葉	ヤ	1	0.311	25	43	4700
ﾊﾞﾙﾃﾞｽ	ダ	0	0.310	21	77	4000
ﾛﾍﾟｽ	広	1	0.308	32	51	8000
松井	西	0	0.308	24	46	25000
ﾃﾞｨｱｽ	広	1	0.304	32	39	3630
高橋	巨	1	0.302	27	49	12000
佐伯	横	1	0.302	14	46	8300
柴原	ダ	0	0.302	7	46	11000
桧山	阪	1	0.300	12	29	4300
清原	巨	1	0.298	29	65	30000
石井	横	1	0.295	8	54	25000
元木	巨	1	0.292	9	37	8800
赤星	阪	1	0.292	1	50	1200
立浪	中	1	0.292	9	54	19000
小久保	ダ	0	0.290	44	62	18000
水口	近	0	0.290	3	60	5600
井出	日	0	0.288	11	46	5800
岩村	ヤ	1	0.287	18	32	5000
江藤	巨	1	0.285	30	73	24000
ｶﾌﾞﾚﾗ	西	0	0.282	49	84	15000
ﾒｲ	ロ	0	0.282	31	46	3000
ﾗﾐﾚｽ	ヤ	1	0.280	29	27	5400
小関	西	0	0.280	3	46	5100

田口	オ	0	0.280	8	43	12000
ﾎﾞｰﾘｯｸ	ロ	0	0.279	31	107	16500
ﾋﾞﾃｨｴﾛ	オ	0	0.275	22	36	6900
二志	巨	1	0.273	20	36	16000
野村	広	1	0.273	9	31	17500
金城	横	1	0.271	3	56	3900
大村	近	0	0.271	16	31	5200
宮本	ヤ	1	0.270	1	27	11800
葛城	オ	0	0.268	14	43	1200
今岡	阪	1	0.268	4	29	3700
吉岡	近	0	0.265	26	66	6700
中村	中	1	0.265	2	31	12500
小川	横	1	0.264	15	57	7000
大島	オ	0	0.263	1	75	8500
木村	広	1	0.263	7	61	4400
濱中	阪	1	0.263	13	53	900
ｱﾘｱｽ	オ	0	0.262	38	49	6000
井端	中	1	0.262	1	49	3200
塩崎	オ	0	0.262	4	54	3800
小坂	ロ	0	0.262	1	77	8500
谷繁	横	1	0.262	20	65	14000
東出	広	1	0.262	5	35	2100
井口	ダ	0	0.261	30	61	4100
城島	ダ	0	0.258	31	31	18000
田中	日	0	0.255	20	57	14000
片岡	日	0	0.254	16	57	18000
金子	日	0	0.253	8	38	6200
初芝	ロ	0	0.253	16	55	10000
福留	中	1	0.251	15	56	4200
土橋	ヤ	1	0.249	2	39	7500
ﾏｸﾚｰﾝ	西	0	0.247	39	78	10000

9.4　分析結果

そして，67行1列のCCRモデルのスコア（SCORE）と逆CCRモデルのスコア（SCORE2），67行4列のCCRモデルの重みWと逆CCRモデルの重みWW2のセルをExcel上に定義した．

このデータを**図9・3**のLINGOで分析すると，**表9.2**の分析結果が得られた． SCOREはCCRモデルで得られたスコアである．松井(巨人)，松井(西部)ら10選手のスコアが１である．これらの選手は，彼らにもっとも適した重みで評価した結果である．最小値の選手は，彼を評価する重みを用いても，他の多くの選手が彼より効率的と評価された．次の４列が，各選手を評価するのに用いられた重みである．

次のSCORE2は，逆CCRのスコアである．１が最も非効率な選手であり，一般にSCOREが1に近い選手ほど値が大きい．次の４列が，各選手を評価するのに用いられた重みである．

表9.2　分析結果

選手	score	打率	本塁打	四球	推定年俸	score2	打率	本塁打	四球	推定年俸
福浦	0.251	0.4746	0.0364	0.0031	0.0000	4.334	0.0000	0.0555	0.0000	0.0008
小笠原	0.387	2.4811	0.0000	0.0025	0.0000	4.967	0.0000	0.0312	0.0000	0.0005
松中	0.543	2.5513	0.0000	0.0026	0.0000	6.021	0.0000	0.0278	0.0000	0.0004
松井	1.000	0.5975	0.0132	0.0000	0.0000	20.064	0.0000	0.0278	0.0000	0.0004
ﾛｰｽﾞ	0.616	2.4307	0.0000	0.0025	0.0000	4.691	0.0000	0.0182	0.0000	0.0003
谷	0.448	0.5718	0.0438	0.0038	0.0001	7.193	3.0760	0.0000	0.0000	0.0009
古田	0.860	1.0605	0.0058	0.0133	0.0000	18.040	3.0857	0.0000	0.0000	0.0009
ﾍﾟﾀｼﾞｰﾆ	0.434	3.1036	0.0000	0.0000	0.0000	7.779	0.0000	0.0256	0.0000	0.0004
中村	1.000	3.1231	0.0000	0.0000	0.0000	9.424	0.0000	0.0217	0.0000	0.0003
礒部	0.199	0.5089	0.0390	0.0033	0.0000	3.399	0.0000	0.0588	0.0000	0.0008
鈴木	0.831	0.0000	0.0705	0.0043	0.0000	16.700	3.1739	0.0000	0.0000	0.0009
金本	0.556	1.5735	0.0202	0.0000	0.0000	10.214	0.0000	0.0000	0.0078	0.0005
真中	0.448	0.0000	0.0285	0.0185	0.0001	7.915	3.2045	0.0000	0.0000	0.0009
稲葉	0.198	0.5666	0.0013	0.0184	0.0000	2.716	0.0000	0.0400	0.0000	0.0006
ﾊﾞﾙﾃﾞｽ	0.173	1.2952	0.0285	0.0000	0.0000	2.751	0.0000	0.0476	0.0000	0.0007
ﾛﾍﾟｽ	0.300	1.2141	0.0000	0.0123	0.0000	3.612	0.0000	0.0312	0.0000	0.0005
松井	1.000	0.0000	0.0031	0.0201	0.0000	15.050	0.0000	0.0417	0.0000	0.0006
ﾃﾞｨｱｽ	0.164	0.6080	0.0014	0.0197	0.0000	1.639	0.0000	0.0312	0.0000	0.0005
高橋	0.465	1.2541	0.0000	0.0127	0.0000	6.421	0.0000	0.0370	0.0000	0.0005

佐伯	0.353	1.0493	0.0057	0.0131	0.0000	8.032	3.3105	0.0000	0.0000	0.0010
柴原	0.970	0.0000	0.0846	0.0089	0.0001	10.645	3.3106	0.0000	0.0000	0.0010
桧山	0.261	0.0000	0.0014	0.0339	0.0001	4.189	3.3328	0.0000	0.0000	0.0010
清原	0.971	1.0469	0.0000	0.0106	0.0000	14.946	0.0000	0.0345	0.0000	0.0005
石井	1.000	1.0226	0.0718	0.0000	0.0000	24.766	3.3890	0.0000	0.0000	0.0010
元木	0.450	0.0000	0.0197	0.0175	0.0001	8.808	3.4240	0.0000	0.0000	0.0010
赤星	0.102	0.0000	0.7200	0.0000	0.0001	1.201	3.4239	0.0000	0.0000	0.0010
立浪	0.758	0.9888	0.0053	0.0123	0.0000	19.015	3.4238	0.0000	0.0000	0.0010
小久保	0.724	2.8325	0.0000	0.0029	0.0000	5.912	0.0000	0.0227	0.0000	0.0003
水口	0.628	0.0000	0.1077	0.0113	0.0001	5.497	0.0000	0.0000	0.0167	0.0010
井出	0.396	0.6976	0.0535	0.0046	0.0001	5.885	3.4714	0.0000	0.0000	0.0010
岩村	0.274	0.0000	0.0012	0.0305	0.0001	4.013	0.0000	0.0555	0.0000	0.0008
江藤	0.725	0.9775	0.0000	0.0099	0.0000	11.558	0.0000	0.0333	0.0000	0.0005
ｶﾌﾞﾚﾗ	0.579	2.7218	0.0000	0.0028	0.0000	4.424	0.0000	0.0204	0.0000	0.0003
ﾒｲ	0.129	3.0416	0.0000	0.0031	0.0000	1.398	0.0000	0.0323	0.0000	0.0005
ﾗﾐﾚｽ	0.344	0.0000	0.0000	0.0370	0.0001	2.691	0.0000	0.0345	0.0000	0.0005
小関	0.680	0.0000	0.1279	0.0134	0.0001	5.323	3.5707	0.0000	0.0000	0.0010
田口	1.000	0.0000	0.0800	0.0084	0.0001	12.525	3.5707	0.0000	0.0000	0.0010
ﾎﾞｰﾘｯｸ	0.676	2.7143	0.0078	0.0000	0.0000	7.689	0.0000	0.0322	0.0000	0.0005
ﾋﾞﾃｨｴﾛ	0.348	0.0000	0.0039	0.0254	0.0001	4.531	0.0000	0.0454	0.0000	0.0007
二志	0.799	0.6720	0.0016	0.0218	0.0000	11.558	0.0000	0.0500	0.0000	0.0007
野村	1.000	0.0000	0.0307	0.0200	0.0001	18.734	3.6624	0.0000	0.0000	0.0011
金城	0.254	2.4248	0.1142	0.0000	0.0001	4.101	0.0000	0.0000	0.0179	0.0011
大村	0.311	0.0000	0.0471	0.0079	0.0001	4.695	0.0000	0.0625	0.0000	0.0009
宮本	1.000	3.1532	0.1485	0.0000	0.0001	12.773	3.7033	0.0000	0.0000	0.0011
葛城	0.072	0.6159	0.0472	0.0041	0.0001	1.238	0.0000	0.0714	0.0000	0.0010
今岡	0.263	0.5306	0.0382	0.0243	0.0001	4.035	3.7308	0.0000	0.0000	0.0011
吉岡	0.297	2.9415	0.0085	0.0000	0.0000	3.723	0.0000	0.0385	0.0000	0.0006
中村	0.935	2.7835	0.1311	0.0000	0.0001	13.786	3.7731	0.0000	0.0000	0.0011
小川	0.268	0.9497	0.0051	0.0118	0.0000	6.741	0.0000	0.0667	0.0000	0.0010
大島	1.000	1.2006	0.0920	0.0079	0.0001	6.675	0.0000	0.0000	0.0133	0.0008
木村	0.199	1.6867	0.0795	0.0000	0.0000	4.248	0.0000	0.0000	0.0164	0.0010

濱中	0.037	1.0086	0.0054	0.0125	0.0000	1.000	3.8013	0.0000	0.0000	0.0011
ｱﾘｱｽ	0.273	3.2069	0.0000	0.0033	0.0000	2.282	0.0000	0.0263	0.0000	0.0004
井端	0.278	3.2345	0.1524	0.0000	0.0001	3.556	0.0000	0.0000	0.0000	0.0011
塩崎	0.410	0.0000	0.1035	0.0108	0.0001	4.144	0.0000	0.0000	0.0185	0.0011
小坂	1.000	3.0604	0.0803	0.0015	0.0001	6.501	0.0000	0.0000	0.0130	0.0008
谷繁	0.480	0.8494	0.0046	0.0105	0.0000	10.112	0.0000	0.0500	0.0000	0.0007
東出	0.126	0.4493	0.0324	0.0206	0.0001	2.334	0.0000	0.0000	0.0000	0.0011
井口	0.180	3.0959	0.0000	0.0031	0.0000	1.975	0.0000	0.0333	0.0000	0.0005
城島	1.000	0.0000	0.0000	0.0323	0.0001	8.390	0.0000	0.0323	0.0000	0.0005
田中	0.676	3.1982	0.0092	0.0000	0.0000	10.112	0.0000	0.0500	0.0000	0.0007
片岡	0.993	1.6503	0.0363	0.0000	0.0001	16.251	0.0000	0.0625	0.0000	0.0009
金子	0.547	0.0000	0.0695	0.0117	0.0001	7.162	3.9518	0.0000	0.0000	0.0012
初芝	0.553	1.6531	0.0363	0.0000	0.0001	9.028	0.0000	0.0625	0.0000	0.0009
福留	0.165	0.9732	0.0053	0.0121	0.0000	4.045	0.0000	0.0667	0.0000	0.0010
土橋	0.587	2.9132	0.1372	0.0000	0.0001	8.334	0.0000	0.0000	0.0000	0.0011
ﾏｸﾚｰﾝ	0.435	3.0647	0.0000	0.0031	0.0000	3.705	0.0000	0.0256	0.0000	0.0004

9.5　松井の年収は妥当か

松井(巨人)の推定年収は，他の選手に比べてどうなのだろう．SCOREが１で最も推定年収の安い城島（ダイエー）と比較した結果が表9.3である．松井の推定年収の5億円の下が，松井の重みで計算した各選手のスコアである．松井の次に高スコアなのは，清原（巨人）の0.67061である．いかに，松井の重みが，松井だけに有利に働いているかを示している．また，パリーグよりセリーグ，セリーグの中でも巨人の選手が優遇されていることが分かる．このスコアに松井の推定年俸の5億円をかけたものが松井基準である．差額は，本人の推定年俸から松井基準による推定年俸を引いたものである．わずか3選手が正で，松井選手並の評価基準であれば年収がアップすることになる．残り63選手が松井選手並みに評価されていないことになる．これがスター選手たるゆえんなのであろう．

一方，城島は推定年俸が18000万円である．城島基準より評価が低いのは，わずか4選手である．パリーグに属しているだけで，収入的にはかなり不利なことが分かる．

野球選手の皆さん，契約更改時のとき，こっそりDEAで自分の評価を調べ，有利に交渉を進めたらいかがだろうか．何なら私がお手伝いします．

表9.3　松井と城島を比較する

選手	50000	松井基準	差額	18000	城島基準	差額
福浦	0.24335	12168	-6768	0.160	2886	2514
小笠原	0.35265	17632	-6632	0.301	5413	5587
松中	0.44545	22272	-7272	0.453	8158	6842
松井	1	50000	0	0.000	0	50000
ﾛｰｽﾞ	0.38829	19415	-1565	0.370	6667	11183
谷	0.43777	21888	-13888	0.212	3816	4184
古田	0.55711	27856	-7856	0.000	0	20000
ﾍﾟﾀｼﾞｰﾆ	0.40662	20331	669	0.000	0	21000
中村	0.75293	37647	-7647	0.497	8943	21057
礒部	0.19273	9637	-5637	0.132	2385	1615
鈴木	0.60584	30292	-12292	0.000	0	18000
金本	0.5261	26305	-4105	0.000	0	22200
真中	0.27909	13955	-5505	0.000	0	8450
稲葉	0.11166	5583	-883	0.000	0	4700
ﾊﾞﾙﾃﾞｽ	0.17322	8661	-4661	0.089	1611	2389
ﾛﾍﾟｽ	0.17165	8582	-582	0.000	0	8000
松井	1	50000	-25000	0.936	16849	8151
ﾃﾞｨｱｽ	0.07809	3904	-274	0.000	0	3630
高橋	0.27817	13909	-1909	0.000	0	12000
佐伯	0.23999	11999	-3699	0.000	0	8300
柴原	0.80668	40334	-29334	0.412	7414	3586
桧山	0.12949	6475	-2175	0.000	0	4300
清原	0.67662	33831	-3831	0.000	0	30000
石井	0.82156	41078	-16078	0.000	0	25000
元木	0.28392	14196	-5396	0.000	0	8800
赤星	0.04663	2332	-1132	0.000	0	1200
立浪	0.61294	30647	-11647	0.000	0	19000
小久保	0.47842	23921	-5921	0.500	9000	9000
水口	0.52588	26294	-20694	0.161	2894	2706
井出	0.36594	18297	-12497	0.217	3909	1891

岩村	0.136	6800	-1800	0.000	0	5000
江藤	0.53801	26900	-2900	0.000	0	24000
ｶﾌﾞﾚﾗ	0.36874	18437	-3437	0.308	5536	9464
ﾒｲ	0.10406	5203	-2203	0.112	2022	978
ﾗﾐﾚｽ	0.12331	6165	-765	0.000	0	5400
小関	0.49288	24644	-19544	0.191	3437	1663
田口	0.88	44000	-32000	0.481	8652	3348
ﾎﾞｰﾘｯｸ	0.57389	28694	-12194	0.266	4781	11719
ﾋﾞﾃｨｴﾛ	0.30401	15201	-8301	0.330	5942	958
二志	0.42481	21240	-5240	0.000	0	16000
野村	0.57516	28758	-11258	0.000	0	17500
金城	0.14759	7380	-3480	0.000	0	3900
大村	0.27914	13957	-8757	0.289	5200	0
宮本	0.47064	23532	-11732	0.000	0	11800
葛城	0.06966	3483	-2283	0.048	865	335
今岡	0.1371	6855	-3155	0.000	0	3700
吉岡	0.26762	13381	-6681	0.175	3147	3553
中村	0.48863	24431	-11931	0.000	0	12500
小川	0.20522	10261	-3261	0.000	0	7000
大島	0.99654	49827	-41327	0.195	3514	4986
木村	0.15268	7634	-3234	0.000	0	4400
濱中	0.02747	1373	-473	0.000	0	900
ｱﾘｱｽ	0.18272	9136	-3136	0.211	3796	2204
井端	0.12884	6442	-3242	0.000	0	3200
塩崎	0.36297	18149	-14349	0.121	2182	1618
小坂	1	50000	-41500	0.190	3422	5078
谷繁	0.37492	18746	-4746	0.000	0	14000
東出	0.07646	3823	-1723	0.000	0	2100
井口	0.14884	7442	-3342	0.116	2084	2016
城島	0.64037	32018	-14018	1.000	18000	0
田中	0.67349	33674	-19674	0.423	7615	6385
片岡	0.99306	49653	-31653	0.544	9790	8210

金子	0.48329	24164	-17964	0.281	5058	1142
初芝	0.55262	27631	-17631	0.313	5637	4363
福留	0.12455	6227	-2027	0.000	0	4200
土橋	0.29874	14937	-7437	0.000	0	7500
ﾏｸﾚｰﾝ	0.30255	15127	-5127	0.221	3975	6025

9.6　2006年度でどう変わったか

野球は，男子学生にとって興味のあるテーマのようだ．私は過去の学生のレポートをお手本にするようにといって,田中君のものを含め何名かのコピーを渡している.最近では，HPに掲載するようにしている．そして，すでに提出されたもの以上のレポートであれば，同じテーマでも許している.

2006年度の卒業生の高橋君が同じテーマでレポートを書きたいという．彼が集めたデータは,次の通り3年間分の58項目とかなり努力してくれた.以下が彼のレポートに関する記述である.

・・・

２－１．収集したデータについて

今回の分析では 2006 年のペナントレースで「規定打席を越えて活躍した打者」について分析を行う．規定打席を越えていない選手については分析を行わない．尚，規定打席を越えた選手は 59 人（うちセ・リーグ 29 人，パ・リーグ 29 人）となった.

規定打席とは（所属球団の試合数 × ３．１）という式で求められる打席数だ．これを越えていることは「首位打者」の条件にもなっている．打席数が少ないと「打率」が極端に高くなったり，低くなったりする．例えば，40 打席で 16 安打では打率は４割になり，首位打者以上の打率になってしまう．そこで，今回の分析では除外した．また，投手の分析を行わないのは，記録の性質上，打者とは成績内容が大きく異なるためだ.

データは 2005 年から 2007 年まで，４年間に渡って収集した．尚，2007 年のデータについては試合がまだ行われていないため,「推定年俸」のみとなっている.

多年度に渡ってデータを収集することで,例えば,ある年度のデータで作った回帰式を,別の年度のデータに割り当て,回帰式がどこまで信頼できるか検証することも可能になる.また，別の年度のデータを回帰分析の説明変数にすることも可能になり，回帰式の精度を更に高めることが期待できる.

２－２．データの項目

データ項目は,「選手名」・「所属球団」・「所属リーグ」・「推定年俸」・「打率」・「試合数」・

「打数」・「得点」・「安打」・「二塁打」・「三塁打」・「本塁打」・「打点」・「盗塁」・「四球」・「死球」・「三振」・「長打率」・「出塁率」の19項目．これを３年間分収集し，更に2007年度の「推定年俸」を加えたため，項目数は19×３+１で，58項目という膨大なものになった．

データの出展については，ＮＰＢ（日本プロ野球機構）の公式ホームページ（http://www.npb.or.jp/）から参照し，推定年俸についてはサンケイスポーツの公式ホームページ（http://www.sanspo.com/）から参照し，データを合成した．

・・・

実は，2006年度の年俸を2006年度の15項目の成績で回帰分析すると選ばれる項目が異なっている．変数増加法では，四球，打点，死球，三振，三塁打が選ばれる．変数減少法では，打率，打数，安打，三塁打，打点，四球，死球が選ばれる．

ここでは2つのデータの違いを見るため，打率，本塁打，四球を入力とし，推定年俸を出力とする．表9.4に58選手のデータを示す．松井らが大リーグに移り，選手らもかなり入れ替わっている．

表9.4　2006年度の58選手のデータ

選手名	球団	リーグ	打率	本塁打	四球	推定年俸
福留　孝介	（中）	セリーグ	0.351	31	76	25,500
李　承ヨプ	（巨）	セリーグ	0.323	41	56	16,000
青木　宣親	（ヤ）	セリーグ	0.321	13	68	6,800
前田　智徳	（広）	セリーグ	0.314	23	40	18,800
岩村　明憲	（ヤ）	セリーグ	0.311	32	70	21,600
シーツ	（神）	セリーグ	0.31	19	42	21,000
タイロン・ウッズ	（中）	セリーグ	0.31	47	84	50,000
金本　知憲	（神）	セリーグ	0.303	26	79	29,000
濱中　治	（神）	セリーグ	0.302	20	43	4,200
荒木　雅博	（中）	セリーグ	0.3	2	26	13,000
新井　貴浩	（広）	セリーグ	0.299	25	32	8,800
リグス	（ヤ）	セリーグ	0.294	39	33	8,000
阿部　慎之助	（巨）	セリーグ	0.294	10	35	14,000
鳥谷　敬	（神）	セリーグ	0.289	15	60	3,800
梵　英心	（広）	セリーグ	0.289	8	27	1,400
二岡　智宏	（巨）	セリーグ	0.289	25	30	16,000
石井　琢朗	（横）	セリーグ	0.288	6	63	17,500

井端　弘和	（中）	セリーグ	0.283	8	61	20,000
東出　輝裕	（広）	セリーグ	0.282	0	27	1,500
森野　将彦	（中）	セリーグ	0.28	10	26	3,600
宮出　隆自	（ヤ）	セリーグ	0.275	9	38	3,500
矢野　輝弘	（神）	セリーグ	0.274	17	32	22,000
アレックス	（中）	セリーグ	0.273	15	52	26,250
嶋　重宣	（広）	セリーグ	0.269	24	27	5,650
赤星　憲広	（神）	セリーグ	0.269	0	60	13,500
金城　龍彦	（横）	セリーグ	0.268	11	48	13,500
ラミレス	（ヤ）	セリーグ	0.267	26	19	30,000
村田　修一	（横）	セリーグ	0.266	34	39	4,150
谷繁　元信	（中）	セリーグ	0.234	9	71	22,500
松中　信彦	（ソ）	パリーグ	0.324	19	102	50,000
カブレラ	（西）	パリーグ	0.315	31	68	60,000
リック	（楽）	パリーグ	0.314	4	22	4,000
小笠原　道大	（日）	パリーグ	0.313	32	73	38,000
福浦　和也	（ロ）	パリーグ	0.312	4	33	18,000
川崎　宗則	（ソ）	パリーグ	0.312	3	32	9,000
稲葉　篤紀	（日）	パリーグ	0.307	26	27	8,000
中島　裕之	（西）	パリーグ	0.306	16	30	4,400
鉄平	（楽）	パリーグ	0.303	2	21	1,000
村松　有人	（オ）	パリーグ	0.303	3	21	10,000
フェルナンデス	（楽）	パリーグ	0.302	28	53	20,000
田中　賢介	（日）	パリーグ	0.301	7	30	6,000
高須　洋介	（楽）	パリーグ	0.3	1	40	3,000
和田　一浩	（西）	パリーグ	0.298	19	78	27,000
セギノール	（日）	パリーグ	0.295	26	40	20,000
大村　直之	（ソ）	パリーグ	0.294	6	41	17,000
赤田　将吾	（西）	パリーグ	0.293	2	36	3,900
片岡　易之	（西）	パリーグ	0.292	4	24	1,900
森本　稀哲	（日）	パリーグ	0.285	9	46	3,000
西岡　剛	（ロ）	パリーグ	0.282	4	49	5,100

ズレータ	（ソ）	パリーグ	0.281	29	47	10,000
ベニー	（ロ）	パリーグ	0.281	17	49	13,000
塩崎　真	（オ）	パリーグ	0.278	9	38	5,000
谷　佳知	（オ）	パリーグ	0.267	6	30	28,000
今江　敏晃	（ロ）	パリーグ	0.267	9	17	5,500
里崎　智也	（ロ）	パリーグ	0.264	17	45	5,500
ＳＨＩＮＪＯ	（日）	パリーグ	0.258	16	24	22,000
金子　誠	（日）	パリーグ	0.254	6	24	7,000
山崎　武司	（楽）	パリーグ	0.241	19	51	8,000

この打率から推定年俸のデータをセル名Fにして分析すると，**表9.5**のスコアと重みが計算される．

表9.5　スコアと重み

選手名	球団	SCORE	W			
福留　孝介	（中）	0.404	1.186	0.017	0.001	0.000
李　承ヨプ	（巨）	0.296	1.366	0.000	0.010	0.000
青木　宣親	（ヤ）	0.164	1.803	0.025	0.001	0.000
前田　智徳	（広）	0.450	0.854	0.009	0.013	0.000
岩村　明憲	（ヤ）	0.365	3.214	0.000	0.000	0.000
シーツ	（神）	0.511	0.867	0.010	0.013	0.000
タイロン・ウッズ	（中）	0.847	3.224	0.000	0.000	0.000
金本　知憲	（神）	0.535	1.860	0.017	0.000	0.000
濱中　治	（神）	0.101	0.854	0.009	0.013	0.000
荒木　雅博	（中）	1.000	0.037	0.272	0.017	0.000
新井　貴浩	（広）	0.233	0.945	0.010	0.014	0.000
リグス	（ヤ）	0.202	1.869	0.000	0.014	0.000
阿部　慎之助	（巨）	0.421	1.073	0.012	0.016	0.000
鳥谷　敬	（神）	0.093	1.836	0.026	0.001	0.000
梵　英心	（広）	0.053	0.000	0.021	0.031	0.000
二岡　智宏	（巨）	0.441	0.982	0.011	0.015	0.000
石井　琢朗	（横）	0.602	1.726	0.084	0.000	0.000
井端　弘和	（中）	0.627	2.323	0.043	0.000	0.000

東出　輝裕	（広）	0.247	0.000	#######	0.037	0.000
森野　将彦	（中）	0.134	0.000	0.021	0.031	0.000
宮出　隆自	（ヤ）	0.109	2.326	0.033	0.002	0.000
矢野　輝弘	（神）	0.653	1.059	0.012	0.016	0.000
アレックス	（中）	0.671	1.913	0.027	0.001	0.000
嶋　重宣	（広）	0.168	1.062	0.012	0.016	0.000
赤星　憲広	（神）	1.000	3.717	#######	0.000	0.000
金城　龍彦	（横）	0.393	2.181	0.031	0.002	0.000
ラミレス	（ヤ）	1.000	0.000	0.000	0.053	0.000
村田　修一	（横）	0.102	1.815	0.000	0.013	0.000
谷繁　元信	（中）	0.759	2.500	0.046	0.000	0.000
松中　信彦	（ソ）	1.000	1.483	0.027	0.000	0.000
カブレラ	（西）	1.000	3.173	0.000	0.000	0.000
リック	（楽）	0.209	0.000	0.184	0.012	0.000
小笠原　道大	（日）	0.637	3.194	0.000	0.000	0.000
福浦　和也	（ロ）	0.832	0.022	0.163	0.010	0.000
川崎　宗則	（ソ）	0.504	0.027	0.198	0.012	0.000
稲葉　篤紀	（日）	0.224	0.998	0.011	0.015	0.000
中島　裕之	（西）	0.132	1.069	0.012	0.016	0.000
鉄平	（楽）	0.084	0.000	0.297	0.019	0.000
村松　有人	（オ）	0.650	0.000	0.229	0.015	0.000
フェルナンデス	（楽）	0.398	0.709	0.008	0.011	0.000
田中　賢介	（日）	0.210	0.000	0.019	0.029	0.000
高須　洋介	（楽）	0.239	0.000	0.284	0.018	0.000
和田　一浩	（西）	0.578	1.604	0.022	0.001	0.000
セギノール	（日）	0.473	0.843	0.009	0.013	0.000
大村　直之	（ソ）	0.579	1.709	0.083	0.000	0.000
赤田　将吾	（西）	0.256	0.031	0.232	0.015	0.000
片岡　易之	（西）	0.097	0.000	0.180	0.012	0.000
森本　稀哲	（日）	0.090	2.242	0.031	0.002	0.000
西岡　剛	（ロ）	0.214	2.100	0.102	0.000	0.000
ズレータ	（ソ）	0.217	1.602	0.000	0.012	0.000

ベニー	（ロ）	0.312	1.797	0.025	0.001	0.000
塩崎　真	（オ）	0.154	2.310	0.032	0.002	0.000
谷　佳知	（オ）	1.000	0.017	0.126	0.008	0.000
今江　敏晃	（ロ）	0.290	0.000	0.029	0.043	0.000
里崎　智也	（ロ）	0.137	1.864	0.026	0.001	0.000
ＳＨＩＮＪＯ	（日）	0.774	1.255	0.014	0.019	0.000
金子　誠	（日）	0.304	0.000	0.024	0.036	0.000
山崎　武司	（楽）	0.196	1.832	0.026	0.001	0.000

この結果を見ると，スコアが1の選手が5人いて，カブレラが推定年俸5億円で一番高く，荒木が1億3000万円で一番低い．この2人の重みを用いて，他の選手の年俸を**表9.6**で示す．

表9.6　カブレラと荒木の評価基準で比較する

選手名	球団	カブレラ	60000.000		荒木	13000.000	
福留　孝介	（中）	0.381	22884.888	2615.112	0.201	2619.127	22880.873
李　承ヨプ	（巨）	0.260	15603.990	396.010	0.102	1321.395	14678.605
青木　宣親	（ヤ）	0.111	6673.418	126.582	0.111	1444.162	5355.838
前田　智徳	（広）	0.314	18862.574	-62.574	0.208	2707.144	16092.856
岩村　明憲	（ヤ）	0.365	21877.436	-277.436	0.168	2181.078	19418.922
シーツ	（神）	0.356	21341.870	-341.870	0.274	3564.012	17435.988
タイロン・ウッズ	（中）	0.847	50799.520	-799.520	0.271	3516.753	46483.247
金本　知憲	（神）	0.502	30147.290	-1147.290	0.265	3441.069	25558.931
濱中　治	（神）	0.073	4381.378	-181.378	0.052	679.534	3520.466
荒木　雅博	（中）	0.228	13653.801	-653.801	1.000	13000.000	0.000
新井　貴浩	（広）	0.155	9272.348	-472.348	0.092	1197.256	7602.744
リグス	（ヤ）	0.143	8572.178	-572.178	0.055	716.216	7283.784
阿部　慎之助	（巨）	0.250	15003.048	-1003.048	0.324	4207.939	9792.061
鳥谷　敬	（神）	0.069	4142.154	-342.154	0.057	743.125	3056.875
梵　英心	（広）	0.025	1526.323	-126.323	0.041	529.006	870.994
二岡　智宏	（巨）	0.291	17442.171	-1442.171	0.168	2187.140	13812.860
石井　琢朗	（横）	0.319	19142.427	-1642.427	0.495	6433.054	11066.946
井端　弘和	（中）	0.371	22263.461	-2263.461	0.476	6193.673	13806.327
東出　輝裕	（広）	0.028	1675.993	-175.993	0.244	3169.677	-1669.677

森野　将彦	（中）	0.068	4050.948	-450.948	0.087	1134.843	2465.157
宮出　隆自	（ヤ）	0.067	4009.815	-509.815	0.087	1126.797	2373.203
矢野　輝弘	（神）	0.422	25296.288	-3296.288	0.327	4250.190	17749.810
アレックス	（中）	0.505	30291.090	-4041.090	0.406	5275.524	20974.476
嶋　重宣	（広）	0.110	6617.211	-967.211	0.062	808.101	4841.899
赤星　憲広	（神）	0.264	15810.368	-2310.368	1.000	13000.000	500.000
金城　龍彦	（横）	0.264	15869.420	-2369.420	0.272	3533.510	9966.490
ラミレス	（ヤ）	0.590	35399.759	-5399.759	0.312	4055.297	25944.703
村田　修一	（横）	0.082	4914.720	-764.720	0.032	418.620	3731.380
谷繁　元信	（中）	0.505	30287.408	-7787.408	0.472	6130.212	16369.788
松中　信彦	（ソ）	0.810	48607.487	1392.513	0.556	7224.068	42775.932
カブレラ	（西）	1.000	60000.000	0.000	0.481	6251.562	53748.438
リック	（楽）	0.067	4013.910	-13.910	0.209	2711.514	1288.486
小笠原　道大	（日）	0.637	38241.791	-241.791	0.294	3817.230	34182.770
福浦　和也	（ロ）	0.303	18177.559	-177.559	0.832	10819.352	7180.648
川崎　宗則	（ソ）	0.151	9088.853	-88.853	0.504	6545.834	2454.166
稲葉　篤紀	（日）	0.137	8209.934	-209.934	0.082	1061.522	6938.478
中島　裕之	（西）	0.076	4530.348	-130.348	0.069	903.211	3496.789
鉄平	（楽）	0.017	1039.916	-39.916	0.084	1093.616	-93.616
村松　有人	（オ）	0.173	10399.120	-399.120	0.649	8431.560	1568.440
フェルナンデス	（楽）	0.348	20862.138	-862.138	0.180	2346.009	17653.991
田中　賢介	（日）	0.105	6280.584	-280.584	0.190	2472.472	3527.528
高須　洋介	（楽）	0.053	3150.705	-150.705	0.238	3097.978	-97.978
和田　一浩	（西）	0.476	28539.865	-1539.865	0.319	4148.148	22851.852
セギノール	（日）	0.356	21358.369	-1358.369	0.198	2577.732	17422.268
大村　直之	（ソ）	0.304	18217.830	-1217.830	0.558	7254.389	9745.611
赤田　将吾	（西）	0.070	4193.808	-293.808	0.256	3330.008	569.992
片岡　易之	（西）	0.034	2050.222	-150.222	0.097	1259.391	640.609
森本　稀哲	（日）	0.055	3316.296	-316.296	0.071	924.881	2075.119
西岡　剛	（ロ）	0.095	5697.716	-597.716	0.203	2633.124	2466.876
ズレータ	（ソ）	0.187	11210.583	-1210.583	0.088	1150.336	8849.664
ベニー	（ロ）	0.243	14574.412	-1574.412	0.183	2377.503	10622.497

塩崎　真	（オ）	0.094	5666.504	-666.504	0.124	1609.653	3390.347
谷　佳知	（オ）	0.551	33041.179	-5041.179	1.000	13000.000	15000.000
今江　敏晃	（ロ）	0.108	6490.538	-990.538	0.154	2003.015	3496.985
里崎　智也	（ロ）	0.109	6563.157	-1063.157	0.078	1018.759	4481.241
ＳＨＩＮＪＯ	（日）	0.448	26865.856	-4865.856	0.355	4615.172	17384.828
金子　誠	（日）	0.145	8683.249	-1683.249	0.263	3413.782	3586.218
山崎　武司	（楽）	0.174	10456.745	-2456.745	0.102	1323.631	6676.369

カブレラの基準で評価すると分かるように，68名中4選手がもらいすぎであるが，残り63選手の評価が悪いことが分かる．一方，荒木の重みで見ると，64選手がもらいすぎであり3選手の評価が悪いことが分かる．

以上から，野球選手の年俸は「松井」や「カブレラ」のような最高年俸をもらっているスター選手は，他の選手より優遇され，「城島」や「荒木」のように評価が低い選手の評定方式を用いると他の多くの選手がもらいすぎという構造になっていることが分かる．

あとがき

2007 年 9 月 6 日，私は神戸ポートタワーホテルで目が覚めた．神戸大学で開催された発表会で，ここ 12 年間 LINDO Systems Inc.の開発した What'sBest!で判別分析の新手法の開発に没頭していた成果を発布するためである．いつもの通り，メールをチェックすると，シカゴ大学の Linus 教授から 12 月に代理店契約を解消するので，2008 年 1 月から新しい代理店を探してほしい旨の依頼である．

「とうとうこの日がやってきたか」と，複雑な思いである．私は研究者になろうと思い，京大の理学部に入ったが，入学当初教養部のある吉田キャンパスを歩いていると，中田さんという大男が「ちょっと水泳部の説明会をやっているので聞いていきませんか」と誘われ，そのまま断ることもできず入部してしまった．まじめに部活をやったおかげで，大学院に落ちて，できたばかりの住商情報システム㈱という企業に勤めた．採用試験は終わっていたが，電話を掛けるとすぐに来てくださいということである．眼光の鋭い専務は津田直治さんといって，住友の中興の祖の伊庭貞剛のお孫さんで京大の法学部出身であることが後でわかった．後日，日本電気に受かったと断りに行くと「優秀なのが五万といるのでやめとき，私が断ってやる」といわれ，電話で東京の日本電気の人事担当役員に電話し，「こちらでもらっていいな」という声が聞こえてきた．面接の際は，「優が少ないとか，余り勉強していない」と言っておきながらなので，悪い気はしなかった．入社すると新人研究の途中で，その後社長になった中川課長に日本電気の大阪支社に連れて行かれ，そのまま大阪府立成人病センターの循環器医長の野村裕先生の下に派遣された．今で言えば，二重派遣である．そこで，多変量解析を用いた「計量診断」が私の研究者のスタートになった．

30 歳を前にして，理数系の学問で個人が一からプログラムを開発し，研究するのは非効率であると悟った．そこで，当時まだ全世界で 500 ユーザーほどしかなかった汎用統計ソフトの SAS を日本に紹介するとともに，SAS を相手に統計の勉強を行った．それと前後し，森村英典東京工業大学名誉教授から，OR 学会で活躍してくださいという葉書きをなぜかいただいたので，数理計画法にも興味を持った．そこで，シカゴ大学ビジネススクールの Linus 教授にアポをとり，同教授の研究室で代理店の許可をもらった．それ以降，私にとって統計と数理計画法の二股人生が始まった．森村先生の葉書きは，ノーベル賞少年と違い，多少先生の期待にこたえられたと自負している．

そして，いつのころからか，**難しい統計や数理計画法や数学は，素人から専門家までが容易に使え，専門家が納得する機能を備えたソフトを用いれば，多くの人が容易に問題解**

決ができるという信念を持った．そして，「それらの学問を 21 世紀の一般教養にしたい」というドンキホーテの夢を見るようになった．

統計に関しては，30 歳から 40 歳頃まで，統計が理論としての「統計学」から実用の学問として統計ソフトを使った利用者の学問としての「データの科学」が確立し，今日に至っている．私自身，統計が先行し，その後を数理計画法や数学が追従すると想定していた．しかし，48 歳のとき成蹊大学に奉職したため，その願いを実現することに貢献できなかった．

そこで，数社に代理店になるようコンタクトしたが，30 年以上ずっとまってくれた Linus 教授の期待にこたえるべき，私が 10 年ぶりに「数理計画法を統計に続く 21 世紀の一般教養」にすべく LINDO　JAPAN の代表になり，国内価格や各種の営業方針の枠組みを決めることにした．最近，国立大学の教員が企業を作る時代である．しかし，私立大学であるが，やはり教育や研究を優先すべきだと思っている．そこで，私が，SAS を日本で普及するに当って私と二人三脚で市場を開拓し，例えば製薬メーカー32 社に統計分析システム（SAS，ORACLE，VAX）を販売してくれた（有）MICE の市川さんに LINDO　JAPAN の運営を全面委託することにした．また，数理計画法システムの開発に実績のある㈱構造計画研究所の OR グループにコンサルティングなどを行ってもらう体制を決めた．

私が，大学の教員を退官する 5 年以内に，日本の多くの大学で数理計画法ソフトを用いた問題解決学が学部を問わず 21 世紀の一般教養になっているよう邁進したい．

また，産業界が「感，コツ，経験」の上に，数理計画法で最後の数％の改善に用いてほしいと切に願っている．

文献

Linus Schrage(2007).Optimization Modeling with LINGO(Sixth Edition).LINDO Systems Inc

LINDO Systems Inc(新村秀一訳). LINGO User Manal.

新村秀一（2008）.OLDF と SVM の比較研究(6)-LINGOniyoru 改定 IP-OLDFto 改定 IPLP-OLDF の比較-. 成蹊大学経済学部論集, 38(2). 111-154.

新村秀一（2007）.JMP による　統計レポート作成法.丸善,　東京.

新村秀一（2004）.JMP 活用　統計学とっておき勉強法.講談社,　東京.

新村秀一（1999).パソコンらくらく数学.講談社,　東京.

新村秀一（1997).パソコン楽々統計学.講談社,　東京.

OR 学会編(分担執筆)（1999).経営科学 OR 用語大事典. 朝倉書店.

新村秀一（1995).パソコンによるデータ解析.講談社,　東京.

新村秀一（1994).SPSS for Windows 入門.丸善,　東京.

新村秀一（1994).SAS 言語入門.丸善,　東京.

新村秀一（2002）.パソコン活用　3 日でわかる・使える統計学 .講談社, 東京.

新村秀一（1993).意思決定支援システムの鍵.講談社, 東京.

Ｌ．シュラージ（新村秀一・高森寛訳）(1992).実践数理計画法.朝倉書店,　東京.

Ｌ．シュラージ（青沼龍雄･新村秀一訳）(1990).GINO によるモデリングと最適化.共立出版,　東京.

新村秀一（1989).易しく実践データ解析の進め方.共立出版,　東京.

高森寛・新村秀一 (1987).統計処理エッセンシャル.丸善, 東京.

J. Sall（新村訳）(1986) .SAS による回帰分析の実践.朝倉書店, 東京.

森村・牧野他編(分担執筆)(1984).統計・OR 活用事典.東京書籍, 東京.

OR 学会編(分担執筆)(1983).OR 事例集.日科技連, 東京.