マーケティング・サイエンス入門

上條哲男 (上智大学 経済学部)

平成 22 年 11 月 9 日

第１章　マーケティング・サイエンス

第２章　LISP-STATの利用方法

第3章　意思決定支援システムとエキスパート・システム

第４章　市場反応モデル (v014-002)

4.1　はじめに

マーケティング・サイエンスにおける基本的な考察の１つは、マーケティング変数および市場環境における他の変数に対して、市場がどのように反応するか、ということである。 市場反応の性格を理解することは、特定のマーケティング問題の分析にとって必要不可欠なものである。市場反応に対する記述的および予想的モデルは、市場反応の複雑な過程に関する理解を組織化し、進展させる際の枠組みを提供する。この章では、ある種の記述的および予想的モデルの分析ついて説明する。規範的モデルについては、後の章で取り扱うことにする。この章は、市場反応の性格を、２つのタイプのモデルを調べることで、探求することにする。伝統的に、それらのモデルは、シミュレーション・モデルと確率的モデルと言われているものである。

4.2　市場反応のシミュレーション・モデル

4.2.1　シミュレーション・モデルの定義と用途

この項では、歴史的にシミュレーション・モデルと言われているモデルについて扱うことにする。これらのモデルは、モデルの解を得るために現在利用可能な分析的手順の能力では手におえない複雑に関連する要素から成っているという特徴を持っている。一般的に、これらのモデルは、確率的部分を含んでおり、したがって、モンテ・カルロ法の手順が、分析道具として用いられてきた。

シミュレーション・モデルという名称が、きわめて適切であるというわけではない。第１章で定義されたシミュレーション技法は、事実上、どのようなモデル、すなわち、決定論的モデルあるいは確率論的モデルに対しても適用可能で

ある。その意味で、すべてのマーケティング・モデルは、「シミュレーション・モデル」として考察することができる。しかしながら、マーケティングにおいて、「シミュレーション・モデル」という用語は、そのモデルが、シミュレーション技法によって、いかなる市場反応を示すのかを探求するという目的に照らして開発されたものである、ということを示すために一般に使用されてきた。歴史的には、シミュレーション・モデルというのは、分析の主要な技法が、シミュレーションであるようなモデルである、ということなる。

シミュレーション・モデルは、２つのマネジメント機能に力を発揮するものとして作成されてきている。すなわち、（１）計画策定および（２）作業の監査とコントロールである。シミュレーション・モデルの計画策定の役割はきわめて明確である。そのモデルが現実の合理的な説明となっているならば、代替案をモデル上で調べるという「モデル上での実験」をすることによって、その代替案の現実における結果を検証できるかもしれない。「モデル上での実験」は、「現場での実験」と比較して、それほど費用がかからず、それほど危険でもない。さらに、「モデル上での実験」では、比較的容易に実験を繰り返すことができる。また、「モデル上での実験」では、実験結果を得るまでにそれほどの時間を必要としない。

シミュレーション・モデルは、現実の結果を監視しコントロールするのにも使用することができる。モデルに計画案が入力されると、モデルは各種の値を生成する。その計画を実際に実行した場合には、現実の結果が、モデルが生成する値となっているのかを監視したり、現実の結果を、モデルが生成する値に近づけようにコントロールするためにシミュレーション・モデルを使用することができる。

シミュレーション・モデルを開発する、という努力は、重要な副次効果をもたらす。シミュレーション・モデルを開発するには、市場における各構成要素の相互依存関係および構成要素間の関係を明確に規定する、という作業を必要とする。これらの作業は、市場の理解を深めるとともに、見落としていた重要な領域を発見するのにも役に立つ。

4.2.2　シミュレーション・モデルのマーケティングへの応用

(1) マーケティング・ゲーム

マーケティングにおけるシミュレーションの第１番目の応用は、マーケティ

ング・ゲームである。マーケティング・ゲームは、マーケティング問題を理解し、それに対処する方法を訓練するための道具である。マーケティング・ゲームは、ダイナミックなケース・スタディであると特徴付けられる。そのゲームへの参加者は、時間毎に決定をすることを要求される。参加者の決定によって時間毎に状況が変化し、その変化した状況に対して、参加者は、新たな決定をすることを要求される。このようなサイクルを繰り返すことによって、参加者は、ゲームが提供している現実を表現した市場について理解を深めていくことになる。このことは、現実の市場を理解するのに役立つであろう。

(2) マクロ分析的な市場シミュレーション・モデル

マーケティングにおけるシミュレーション・モデルの第２番目の応用は、マクロ分析的な市場シミュレーション・モデルである。このモデルは、市場構造のダイナミックスを研究し、マクロ的なマーケティング問題を分析する際にマーケティング担当者を支援するために使用することができる。

(3) マイクロ分析的な市場シミュレーション・モデル

マーケティングにおけるシミュレーション・モデルの第３番目の応用は、マイクロ分析的な市場シミュレーション・モデルである。このモデルは、マーケティングにおけるマイクロ的な問題のダイナミックスを研究し、マイクロ的なマーケティング戦略を計画策定する際にマーケィング担当者を支援するために使用することができる。

(a) マイクロ分析的な市場シミュレーション・モデルの構造

マイクロ分析的な市場シミュレーション・モデルの開発は、そのシミュレーション・モデルが達成すべきであると考えられる仕様から始まる。この段階で提示される目的は、そのモデル開発の最終的な結論を示す物である。例えば、Amstutz(1967, p.1) はつぎのように述べている

(Amstutz, A.E.(1967), Computer Simulation of Competitive Market Response, MIT.
邦訳：山下隆弘訳（1969），マーケティングの計量モデル：コンピュータ・シミュレーション，新評論.)。

”This book was written to present elements of an organized behavioral theory of market interactions and to suggest an approach to management based on the use of microanalytic computer simulations of interactions within the marketing

environment.”

「本書は、市場相互作用についての組織化された行動理論の要素を提示し、市場環境における相互作用についてのマイクロ分析的なシミュレーションの使用に基づく経営へのアプローチを示唆するために書かれた。」

目的についての仕様が、分析の限界を設定するための指針を提供する。

システムの境界が定義された後に、マクロ構造を開発する過程が開始される。

図 2-1 は、製品および情報の流れで定義されたサブシステム間の相互依存関係を持つ Amstutz モデルの 1 つにおけるマクロ構造を表現している。

モデルの主要なサブシステムが識別された後に、サブシステムは、詳細に記述されることになる。Amstutz によって提案された消費者部門に対する詳細なモデル構造が図 2-2 に表示されている。

マイクロ単位のモデルの開発において、それぞれの基本的な行為が。詳細にモデル化されなければならない。例えば、広告が消費者にいかなる影響を与えるかを示す過程についての Amstutz によるモデルが図 2-3 である。消費者の決定過程のすべての局面が同様な形式で記述されなければならない、そして最終的には、それらは数学的な式あるいはコンピュータ・プログラムで表現されなければならない。

(b) マイクロ分析的な市場シミュレーション・モデルのテスト

マイクロ分析的シミュレーション・モデルに対して提案された構造および関係式は、そのモデルが市場反応の正確な記述であると明確に述べる前に、テストのための分析がなされねばならない。３種類の分析がある。すなわち、(ⅰ) 論理および安定性をテストするための初期的分析、(ⅱ) 妥当性分析、および (ⅲ) 感度分析である。

(ⅰ) 初期的分析

マイクロ分析的シミュレーション・モデルの 1 つの特徴が、分析のために非常に多くの計算を必要とする、ということであるので、ほとんどすべての場合において、そのモデルは、コンピュータ・プログラムで表現されている。最初のテストは、なんらかの論理的エラーが、コンピュータ・プログラムにおいて生じているかどうか、についてなされる。

論理的エラーが取り除かれた後に安定性に関するテストがなされる。安定性は、期間の経過およびパラメター値の取りうる範囲に照らしてモデルがいかなる結果を示すかに関わりがある。ここでの判断基準は、モデルが道理にかなっ

た行動を示すかどうかである。例えば、モデルは、爆発的なあるいはあまりにも振動的な結果を示すべきではない。

モデルが安定性のテストを通過したならば、モデルの結果が合理的であるかどうかについてチューリング・テスト的なテストを行うべきだある。それは、モデルが現実をどの程度把握しているかをテストするものである。、このテストでは、モデルが扱っている問題を熟知している人が、モデルが生成した結果を現実での結果と比較した場合にそれらの間に差を見出せない程度が高いほど、そのモデルはより現実を把握している程度が高いものと判断する、というものである。

(ⅱ) 妥当性分析

理論的に、妥当性は、モデルの絶対的な真実に関係しているが、現実には、モデルがモデル化された実際の状況の合理的な近似となっているかどうかに関連している。マイクロ分析的シミュレーション・モデルにおいては、モデル全体、サブモデル、およびそれらの構成要素である関数が妥当性分析の対象となる。

経験的な妥当性は、現実の状況における結果とモデルから得られる結果とを比較することによってなされる。この比較に対する最も単純な手順は、現実の結果とモデルの結果とを、それらが類似しているかいなかを比較するために、表形式に並べて表示するか、グラフを描くことである。比較のための統計的な手法は、カイ２乗検定あるいは Kolmogorov-Smirnov 検定などの「当てはまりのよさ」に関するテストである。そのようなテストを実施して得られた結果は、モデルのマイクロ構造を改訂するのに使用されるべきである。すなわち、方程式の関数形は、最良の当てはまりが得られるまで改良されるべきである。このテストを行うことによって、重要なパラメター、関数、関係式、そして分布に関する実行可能な最も正確な表現が獲得される。

洗練された測定値が得られ、実際の状況の現実的な表現が手に入ったことを、モデル作成者が信じたならば、マクロモデルの妥当性のテストが開始される。そのテストは、モデルが全体として現実の状況の合理的な表現であるかどうかを決定することである。そのモデルは妥当性に関するテストを通過しなければならない。そのテストでは、そのモデルが、経営者が戦略的な代替案の結果を探求する際に有用な実験室となるかをテストするものである。

大規模なシミュレーションにおけるマクロモデルの妥当性には、一般的には２つのクラの妥当性が考えられる。１つは、外見的あるいは構造的な妥当性であり、もう１つは、予想的妥当性である。

第1の妥当性は、社会科学における構成的妥当性の考えと関連を持つものである。この考えは、モデルの表現と出力は、外見上の価値に関して妥当であるというものである。例えば、もし、経営者がそのモデル構造およびそのパラメターの値を実質的に正しいものと強く信じているならば、そのモデルは、経営者にとって外見上の妥当性を持つことになる。あるモデルの、経営者の観点からの外見上の妥当性は、もし、その経営者が、そのモデルの開発過程における統合的な重要な一員でなかったとするならば、得られそうには思われない。

第2の妥当性である予想的妥当性をテストするには、まず、モデルの出力を現実の状況における対応する結果と比較する必要があり、そこでの一致の度合いがそのモデルの予想的妥当性の測度として使用されることになる。そのような分析を遂行するためには、「当てはまりのよさ」を求める手法が数多く存在する。

最初は、モデルの出力と現実の結果が、同様の大きさと時間的変動を示しているかどうかをテストする際に、統計的分布に関して明確な仮説を要求しない統計的手法を、しばしば使用することができる。そのようなアプローチの１つが、ランク・オーダー・テストである。

予想的妥当性に関するもう１つのアプローチは、モデルの時系列的出力と現実の時系列的出力を比較するという手法である。具体的な手法としては、回帰分析およびスペクトラル分析などがある。

予想的妥当性をテストするための統計的道具および手順は十分に開発されたという状況には、なっていない。

(iii) 感度分析

感度分析の目的は、関数、パラメター、分布、および入力のいずれがモデルの出力に対して感度が強いかを決定することである。感度分析は、洗練された測定値および関数の妥当性のテストの前になされる必要がある。その理由は、遂行において実際上の差異をもたらすようなモデルの局面を測定することに対して利用可能な少ない財務的、人的、および時間的資源を配分する際に、感度分析は、モデル作成者に支援を与えることができるからである。感度分析のより詳細な考察をする前に、そのような分析から得られる情報の含意を調べることは有益であろう。モデルは、2つのタイプのパラメター、関数、および分布を含んでいる、すなわち、直接的な測定値に影響を受けやすいものとそうでないものとである。もし、測定可能な関数、パラメター、あるいは分布におけるある要因が、感度分析によって、非常に重要であることがわかったとするなら

ば、このことは、モデル作成者に、その要因について注意深い測定を行うことを示唆するものであると考えられる。測定可能でない要因に含まれている意味は、つぎの２つである。１つ目は、経営者の洗練された判断は、有意として識別されてきた主観的要因において見つけられるはずである。2 つ目は、最終的実行（すなわち、政策意味を持つ実行）における有意な要因に関する感度分析の実行に対して考慮が払われるべきであるおいうこと。その結果、判断的入力における推定誤差の決定上の含意を探求することができる。

市場反応のマイクロ分析的シミュレーション・モデルにおける感度分析の使用は、報告されていないが、このタイプの分析の潜在力がどのようなものであるかについては議論する必要があろう。市場反応のマイクロ分析的シミュレーション・モデルにおける感度分析の基本的な考察が以下に示される。

感度分析を遂行する際に利用可能な基本的な方法が３つある。すなわち、解析的方法 (analytic methods)、非実験的摂動方法 (nonexperimental perturbations methods)、および実験的方法 (experimental methods) である。

解析的方法は一般的に感度分析可能な関数に対してのみ利用可能である。この場合、あるパラメターに関する反応の感度は、そのパラメターに反応関数のそのパラメターに関する微分を計算することによってテストすることができる。

解析的方法は通常マイクロ分析に限定される。

非実験的摂動方法は、モデルの要因のどれが重要な役割を果たすかについてのモデル作成者および経営者における判断を含むものである。モデル作成者あるいは経営者は、重要と考えられるいくつかの要因を選択し、それらの要因について水準を決定する。それら要因および水準の組み合わせをモデルで実行し、モデルの出力から、どの要因が感度が高いのかを判断する。

実験的方法は、マイクロレベルでの最も科学的な形式の感度分析である。モデル作成者はどの要因をテストし、その水準はいくらするかを決定する。実験的方法と非実験的摂動方法の間の差異は、実験的方法では、選択された実験計画手順が分析を実行する際の要因水準の組み合わせを決めることにある。

感度分析における実験的方法の利点は、分散分析の結果の精度を主張できることである。

4.2.3　モデルの改訂と保守

モデルを一度にすべて開発するというわけにはいかない。モデルが計画策定およびコントロールの目的にとって適切であり続けるためには、モデルの保守および改訂の過程を継続して行う必要がある。市場は、動的な現象であり、市場反応における変化は、常に生起する。

4.2.4　シミュレーション・モデルにおける将来の研究課題

マイクロ分析的シミュレーション・モデルの重要な研究課題は、２つある。1つは、感度分析であり、もう１つは、妥当性分析である。

4.3　消費者反応の確率的モデル

4.3.1　確率的モデルの定義と用途

シミュレーション・モデル、特に、マイクロ分析的シミュレーション・モデルは、市場反応をモデル化する際の非常に詳細な包括的なアプローチであるけれども、分析の他の方法も効果的に使用されてきている。その１つが、消費者反応の確率的モデルである。

そのモデルでは、消費者反応は、ある確率過程の結果として生起するものとして考えている。市場反応のマイクロ分析的シミュレーション・モデルにおいてもある反応は、確率的なものとしてモデルに組み入れられている場合がある。したがって、そのシミュレーション・モデルも確率的モデルということができる。

ここでは、確率的モデルは、通常、解析的方法によって、分析されるモデルであり、シミュレーション・モデルは、通常、モンテ・カルロ法によって分析されるモデルであると考えておくことにする。

確率的モデルは、市場データを組織化し、説明するための構成概念として使用される場合がある。

この文脈から、確率的モデルには、２つの主要な使い道がある。１つ目は、構造的仮説をテストするための使い道、そして２つ目は、条件付き予想をする

ための使い道である。

構造的仮説をテストするための使い道という観点からの確率的モデルとしては、ブランド・ロイヤルテー (brand loyalty)、学習 (learning)、および普及効果 (diffusion effects) などに関するモデルがその良い例であろう。

確率的モデルは、ブランドの究極的な市場占有度あるいはそのブランドが究極的な市場占有度を達成するまでの時間についてのような条件付き予想をするために使用することもできる。

この節では、確率的モデルのアプローチについての２つの主要な応用について議論することにする。最初は、消費者のブランド選択過程の分析であり、第２は、購買時点および購買間隔の考察である。

4.3.2　ブランド選択の確率的モデル

ブランド選択モデルは、いくつかのブランドが存在する状況において、それぞれのブランドを選択する過程を記述しようとするものである。基本的なブランド選択モデルは、実際の購買時点に関心があるわけではなく、むしろ購買がなされた際のブランド選択に関心がある。ブランド選択の考察に適用されてきた３つのタイプの確率的モデルがある。すなわち、0 次モデル、マルコフ・モデル、および線形学習モデルである。

それぞれのモデルは、消費者購買行動に関して異なる仮定を設けている。

0 次モデルは、過去のブランド選択は、現在および将来のブランド選択に影響を与えない、ということを仮定している。マルコフ・モデルは、もっとも最近になされたブランド購買のみが、現在のブランド選択に影響を与える、ということを仮定している。線形学習モデルは、ブランド選択は、過去のブランド購買の完全な歴史に依存する、ということを仮定している。

(1) 0 次モデル

(2) マルコフ・モデル

(3) 学習モデル

4.3.3　ブランド購買時点・間隔に関する確率的モデル

前項で記述された 0 次モデル、マルコフ・モデル、および線形学習モデルは、ブランド選択の記述に注意が向けられたものである。それらは、購買時点および購買間隔を明示的には取り扱っていない。この項では、消費者反応における購買時点および購買間隔の問題を考えることにする。

購買間隔行動に関する仮説の 1 つは、Kuehn によって提示されたものである。彼は、購買間の時間が増加するにつれて、再購買の確率は、指数的に減少するということを発見した。これは、オレンジ・ジュースのブランド選択および購買間隔のデータの分析から得られたものである。Morrison のコーヒーに関する経験的分析は、購買間隔が変化することによるブランド・ロイヤルテーにおける差は、有意でないことを指摘している。このことは、Kuehn が冷凍オレンジ・ジュースのデータを使用し、Morrison がコーヒーのデータを使用したことによるものと考えられる。Carman による経験的分析は、異なる購買頻度をもつサブグループに対する学習パラメターを推定したものであるが、その分析は、購買間隔が増加するにつれての再購買確率の減衰を示していない。製品クラスにおける差に関して検討したが、Carman と Morrison の結果は、再購買確率における減衰はブランドがつけられた製品の市場において普遍的に妥当する結果であるわけではない、ということを指摘している。

購買時点の影響は、ブランド選択モデルにおいては、部分的に考慮されている。例えば、単純マルコフ過程のモデルにおいては、購買は特定の時間間隔毎になされるものと過程されている。したがって、購買がなされない時点については、人為的に無購買の状態を設定する必要がある。Howard は、単純マルコフ過程モデルにおける無購買の状態の使用を避けるために、セミ・マルコフ過程モデルを提案した。セミ・マルコフ過程モデルにおいては、購買時点も確率的に決定されるものとしている。　Ehrenberg は、各消費者が、ある期間に、ある製品を購買する際に、その購買単位数は Poisson 分布に従うということを仮定したモデルを提示している。

4.3.4　確率的モデルの将来の課題

確率的モデルに関して、検討されるべき 4 領域がある。

(1) 経験的可能性およびモデル間の比較

(2) 複数ブランドからなる市場のためのモデルの開発

単純マルコフ過程のモデルおよびセミ・マルコフ過程のモデルは同質的マーケットにおける複数ブランドを考慮している。異質的マーケットにおける複数ブランドに対するマルコフ・モデル、普及モデル、および学習モデルが必要であろう。

(3) 無作為な購買間隔を考慮したモデルの開発

異質性と非定常性の考察を含むモデル

(4) 確率的モデルにマーケティング変数含めたモデルの開発

4.4　シミュレーションと確率的モデルの比較

確率的モデルの長所	確率的モデルの短所
(1) 適用がより容易である (a) プログラミング努力がより少ない (b) コンピュータ使用時間がより少ない	(1) 経営者が履行するのが困難である (a) 直観に合う度合いがよ り小さい (b) 経営者の取り込みおよび肩入れがより少ない
(2) 利用可能なデータを使用する	(2) 関数関連およびデータ入力がより少ない。モデル化され た関連はより不確実で ある
(3) 履行するのに費用がより少ない	(3) 数学と統計学についてより高度な知識がより必要である
(4) 企業および製品クラス間での移転がより容易である	
(5) 感度分析について (a) より容易である (b) より費用が少ない (c) 分析の機会がより多い	

表 4.1: シミュレーション・モデルとの対比での確率的モデルの長所と短所

4.5 LISPによるプログラミング (v014-002)

LISP プログラミングの基本的な操作は、関数定義である。ここでは、関数定義のための道具および LISP プログラミングにおいて使用されるプログラミング技法と原理を、Tierney 著の LISP-STAT に基づいて説明する (Tierney(1990) 参照)。

4.5.1 単純な関数の作成

LISP 関数は、特殊形式 defun を用いて定義される。defun に対する基本的なシンタックスは、

$$(defun\ \langle name\rangle\ \langle parameter\rangle\ \langle body\rangle)$$

である。ここに、$\langle name\rangle$ は関数を参照するために使用される記号であり、$\langle parameter\rangle$ は関数のパラメターに名前を付けるために使用される記号のリストであり、$\langle body\rangle$ は 1 つあるいはそれ以上の式から構成される。関数が呼び出されると、本体 ($\langle body\rangle$) 内の式が次々に評価され、最後の式を評価することによって得られる値が関数の結果として返される。最初のうちは、関数本体内に 2 つ以上の式を持つ関数は使用しないことにする。defun はその引数を評価しないので、defun のいずれの引数にも引用符を付ける必要はない。

数字 (number) の 2 乗 (square) を計算する関数は、

```
(defun square (number)
   (* number number)
)
```

と定義される。5 の 2 乗は、

```
> (square 5)
25
```

と計算される。

[例題 4.1]

データセット x の平均値を計算する関数 mean-tk を作成しなさい (tk の部分は、作成者の氏名のイニシャルとする)。LISP-STAT には、平均値を計算する

関数 mean が組み込まれているので、その関数の定義を失わないように、最後に作成者のイニシャルを追加した関数名 mean-tk を使用することにする。

x のデータの個数（大きさ）を n とすると、x の平均値は、

$$\sum_{i=1}^{n} x_i/n$$

と定義される。

[解答]

LISP-STAT による関数は、

```
(defun mean-tk (x)
   (/ (sum x) (length x)
   )
)
```

となる。関数 *sum* は、その引数の要素の合計を計算し。関数 *length* は、その引数の要素の個数 (大きさ) を計算している。1,2,3,4,5, および 6 の平均値は、

```
> (mean-tk (list 1 2 3 4 5 6))
3.5
```

と計算される。

4.5.2 述語と論理式

述語は条件が真であるかないかを決定する関数である。それらは偽に対しては NIL を返し、真に対しては、NON-NIL 値、しばしば記号 T を返す。数の比較において、述語 <, >, =, / =, およびその他の比較のための述語を使用することができる。これらの述語はいずれも、2 つあるいはそれ以上の引数をとる。

述語 < はすべての引数が昇順になっているときは真で、T を返し、そうでないときは偽で、NIL を返す。例えば、つぎのとおりである。

```
> (< 1 2 3)
T
> (< 1 3 2)
NIL
```

述語 > も同様である。述語 = は全ての引数が等しいときは真で、T を返し、そうでないときは偽で、NIL を返す。例えば、つぎのとおりである。

```
> (= 1 1)
T
> (= 1 1 1)
T
> (= 1 2)
NIL
```

述語 / = は全ての引数が等しくないときは真で、T を返し、そうでないときは偽で、NIL を返す。例えば、つぎのとおりである。

```
> (/= 1 2)
T
> (/= 1 2 3)
T
> (/= 1 1)
NIL
> (/= 1 1 1)
NIL
```

特殊形式 and と or および関数 not は、それらを組み合わせることによって、より複雑な述語を作成することができる。

4.5.3 条件付き評価

LISP における条件付き評価の基本的な構成体は、cond である。
cond 式の一般形式は、

$$
\begin{array}{l}
(\ \mathit{cond}\ (\ \langle\ p\ \ 1\ \rangle\ \langle\ e\ \ 1\ \rangle\ ) \\
\qquad\quad (\ \langle\ p\ \ 2\ \rangle\ \langle\ e\ \ 2\ \rangle\ ) \\
\qquad\quad \ldots \\
\qquad\quad (\ \langle\ p\ \ n\ \rangle\ \langle\ e\ \ n\ \rangle\ ) \\
)
\end{array}
$$

である。リスト ($\langle p\ 1\ \rangle\ \langle e\ 1\ \rangle$) などは、条件節と呼ばれる。条件節のそれぞれの第 1 の式 $\langle p\ 1\ \rangle$, ..., $\langle p\ n\ \rangle$ は、真に対しては NON-NIL, あるいは T を与え、偽にたいしては NIL を与える述語式である。cond は、1 つの述語式が真になるまで述語式を順に評価する。もし、ある述語、例えば、$\langle p\ i\ \rangle$ が真であるとすれば、それに対応する式 $\langle e\ i\ \rangle$ が評価され、その式の結果が cond の式の結果として返される。もしすべての述語式が真でないならば、NIL が返される。

cond を用いて、引数の絶対値を求める関数を定義すると、

```
(defun abs-cond-01-tk (x)
   (cond ( (> x 0) x)
         ( (= x 0) 0)
         ( (< x 0) (- x)
         )
   )
)
```

となる。abs-cond-01-tk を用いると、

```
> (abs-cond-01-tk 2)
2
> (abs-cond-01-tk 0)
0
> (abs-cond-01-tk -3)
3
```

となる。

cond に加えて LISP にはいくつかの他の条件付き評価の構成体がある。それらのうちで最も重要なものは if である。特殊形式 if は 3 つの式、述語式 (predicate)、結果式 (consequent)、および代替式 (alternative) をとる。最初に述語式が評価される。もしそれが真であるなば、結果式が評価され、その結果が if 式の結果として返される。そうでなければ代替式が評価され、その結果が返される。もし、オプションである代替式がないならば、NIL が返される。if 式の一般形式は、

$(if\ \langle predicate\rangle\ \langle consequent\rangle\ \langle consequent\rangle)$

である。if 式を用いて、絶対値を計算する関数 abs-if-01-tk を定義すると、

```
(defun abs-if-01-tk (x)
       (if (> x 0)
           x
           (- x)
       )
)
```

となる。abs-if-01-tk を用いると、

```
> (abs-if-01-tk 2)
2
> (abs-if-01-tk 0)
0
> (abs-if-01-tk -3)
3
```

となる。

4.5.4 再帰と繰返し (v014)

LISP は再帰的言語 (recursive language) である。その結果として、LISP では、繰返しを必要とするような数値計算を、再帰的に実行することができる。

ここで、正整数 n の階乗 $n!$ の計算について考えてみることにする。階乗 $n!$ は、

```
0! = 1
n! = n*(n-1)*(n-2)* ··· *3*2*1
```

と定義される。$n!$ は、

```
n! = n*(n-1)!   （ただし、0! = 1）
```

と計算される。すなわち、n の階乗は、$(n-1)$ の階乗に n を乗ずることよって計算される。

正整数 n の階乗を計算する LISP 関数 factorial-01-tk を定義すると、

```
(defun factorial-01-tk (n)
   (if (= n 0)
       1
       (* n (factorial-01-tk (- n 1)
            )
       )
   )
)
```

となる。factorial-01-tk を用いると、

```
> (factorial-01-tk 3)
6
> (factorial-01-tk 10)
3628800
> (factorial-01-tk 20)
2432902008176640000
```

となる。

反復的な計算のための特殊形式に do がある。do 式の一般形式は、

```
(  do  (  (   ⟨ name 1⟩  ⟨ initial 1⟩  ⟨ update 1⟩)
          ...
          (   ⟨ name n⟩  ⟨ initial n⟩  ⟨ update n⟩)
       )
       (      ⟨ test⟩       ⟨ result⟩      )
)
```

である。do には、2 つの引数がある、終了テスト式 (test) を従えている 3 つの要素から成るリストおよび結果式 (result) である。3 つの要素のそれぞれは、状態変数に対する名前 (name)、初期設定式 (initial)、および更新式 (update) である。do は初期設定式を評価し、状態変数をこれらの値とする。つぎに、終了テスト式を評価する。もし終了テスト式を評価して真であるならば、結果式が評価され、その結果が返される。もし終了テスト式が真でなければ、更新式が評価され、変数が新しい値となり、終了テスト式が評価される。同様な手順が繰り返される。do を使用して階乗関数を定義すると、

```
(defun factorial-do-01-tk (n)
  (do ( (counter 1 (+ 1 counter)
        )
        (product 1 (* counter product)
        )
      )
      ( (> counter n) product)
  )
)
```

となる。factorial-do-01-tk を用いると、

```
> (factorial-do-01-tk 3)
6
> (factorial-do-01-tk 10)
3628800
> (factorial-do-01-tk 20)
2432902008176640000
```

となる。

繰返し構成体に loop がある。loop の一般形式は、

$$(loop\ \langle body \rangle)$$

である。ループ内で、各回において順番に $\langle body \rangle$ 内の各式の実行を無限に繰り返す。ループを終了するためには、本体 (body) に、return 式を含めておけばよい。loop を使用して、階乗関数を定義すると、

```
(defun factorial-loop-01-tk (n)
  (let ( (i 1)
         (n-fac 1) ;;; 初期値
       )
       (loop (if (> i n) (return n-fac)
             )   ;;; n 回の繰返し
             (setf n-fac (* i n-fac)
             )
```

```
            (setf i (+ i 1)
            )
        ) ;;; loop
    ) ;;; let
) ;;; defun
```

となる。factorial-loop-01-tk を用いると、

```
> (factorial-loop-01-tk 3)
6
> (factorial-loop-01-tk 10)
3628800
> (factorial-loop-01-tk 20)
2432902008176640000
```

となる。

4.6 価格に関する単純な市場反応モデル (v014)

4.6.1 価格を引数とする販売数量関数

価格 P を引数とする販売数量関数 Q が、

$$Q = \alpha - \beta P \tag{4.1}$$

で与えられるものとする。販売数量関数 Q を quantity-for-price-one-element-type001-fun, 価格 P を price-one-element, α を alpha-pri, および β を beta-pri として、式 (4.1) に対する LISP 関数を定義すると、

```
(defun quantity-for-price-one-element-type001-fun
          (alpha-pri beta-pri price-one-element)
            (+ alpha-pri (* beta-pri price-one-element)
            )
)
```

となる。

[問題 4.1]

関数 quantity-for-price-one-element-type001-fun をエディタ（例えば、メモ帳、秀丸、WZ、あるいは MIFES など）で作成し、tkamijo.lsp に相当する各自のファイルに保存しなさい。

関数 quantity-for-price-one-element-type001-fun において、alpha-pri=2000, beta-pri=-8 とし、price-one-element を 170,175, および 180 とした場合の販売数量の計算を LISP-STAT で実行すると、

```
> (def alpha-pri 2000)
ALPHA-PRI
> (def beta-pri -8)
BETA-PRI
> (quantity-for-price-one-element-type001-fun
     alpha-pri beta-pri 170)
640
> (quantity-for-price-one-element-type001-fun
     alpha-pri beta-pri 175)
600
> (quantity-for-price-one-element-type001-fun
     alpha-pri beta-pri 180)
560
```

となる。