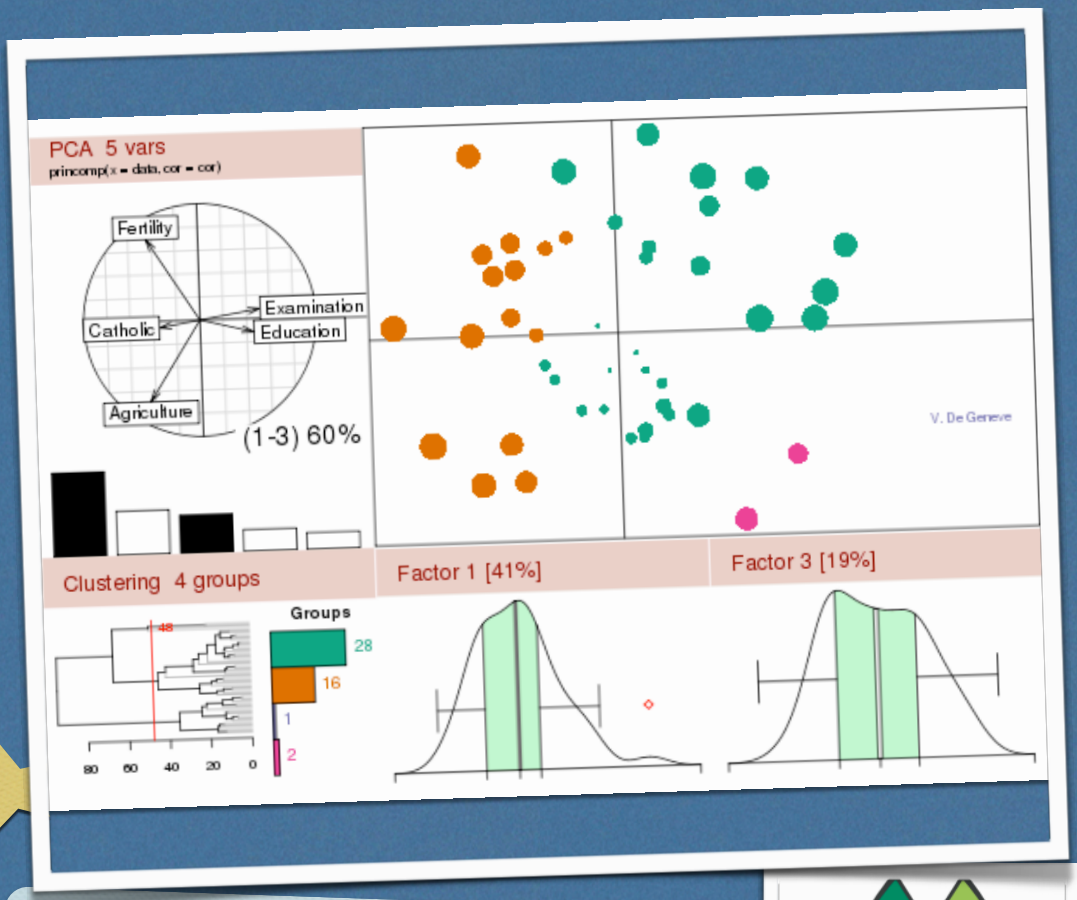
中四国心理学会第69回大会特別企画

Rチュートリアルセミナー



丁寧なテキストつき！

本セミナーでは，Rをまだ始めたことのない人，Rをまだ使い始めてまもない人を対象に，導入の仕方から丁寧に解説し，t検定，回帰分析，因子分析など心理学で用いられる主たる統計解析を実例とともに解説いたします。

2013年11月16日（土・大会一日目）　09:00〜12:00

※チュートリアルセミナーへの参加は，大会参加とは別に

Webサイトにてエントリー手続きを行なって頂く必要があります（参加費は無料です）

大会ウェブサイト　https://sites.google.com/site/chushi2013
大会Facebookサイト　https://www.facebook.com/chushi2013

中国四国心理学会第 69 回大会特別企画
R チュートリアルセミナー特別テキスト

小杉考司[*1]・押江隆[*2]

[*1] E-mail;kosugi@yamaguchi-u.ac.jp/TwitterID; @kosugitti
[*2] E-mail:oshie@yamaguchi-u.ac.jp/Twitter ID; @oshie

目次

第 1 章

What is R?

1.1　R，R 言語，R 環境

R は R 言語，R 環境等とも呼ばれます。これらのことばにはどういう違い，メッセージがあるのでしょうか。辞書的に調べてみましょう。まずは「言語」からです。

言語

言語（げんご 英: Language）とは、コミュニケーションのための記号の体系である。狭義には人間の音声による音声言語を指すが、広義には身振りなど音声以外の要素も含む。また、動物間のコミュニケーションや、コンピュータに指示するための記号体系を指す場合もある。

【Wikipedia より】

次に「環境」を調べてみます。

環境

環境（かんきょう）は、広義においては人、生物を取り巻く家庭・社会・自然などの外的な事の総体であり、狭義においてはその中で人や生物に何らかの影響を与えるものだけを指す場合もある。特に限定しない場合、人間を中心とする生物に関するおおざっぱな環境のことである場合が多い。
パソコンにおいては、オペレーティングシステムやアプリケーションの設定を環境設定などと呼ぶことがある。

【Wikipedia より】

なるほど，パソコン上での環境のことをさすのでしょうか。それではここで，R の開発母体である R-project 自身による，R とは何か？という解説を見てみましょう。

R 環境

R はデータ操作、計算、そしてグラフィックス表示のためのソフト機能の統合されたまとまりです。 それには以下のものが含まれます。

- データを効率的に操作し、保管する機能
- 配列、とくに行列の計算のための演算子のセット
- データ解析の媒介に使う道具の大規模で一貫した集り
- データ解析のためのグラフィカルな機能と、画面または印刷物への出力
- 条件分岐、ループ、ユーザー定義の再帰的関数や入出力機能を含む、開発の進んだ、簡潔で効率的なプログラミング言語

「環境」という言葉が使われているのは、R は完全に計画され一貫性を持ったシステムであり、 固有の目的のみを持って融通のきかない道具の積み重ねではない (他のデータ解析ソフトウェアはしばしばそうなのですが) という特徴を示すためです。多くのユーザーは R を統計システムと考えるでしょう。 しかし我々は R は環境であり、その中で統計的手法が実装されていると考えたいと思います。

【What is R?(http://www.r-project.org/about.html) より抜粋】

このように，R を統計ソフトとして狭い意味でとらえるのではなく，広い意味で考えたいというメッセージが込められているのです。

1.2　統計ソフトはなぜ必要か

これからはビッグデータの時代，統計学が最強の学問だ，等といわれますが，それをするためにどうして専用のソフトウエア，環境が必要なのでしょうか。「数字を扱うなら，Excel でいいじゃない，間に合ってるよ」と考える方もいらっしゃるかもしれません。

しかし，Excel などに代表される表計算ソフトではできないことがあります。一つは行列計算です。行や列，それぞれについて集計をするのは得意な表計算ソフトも，行列全体の計算（固有値分解等に代表される線形代数的処理です）は不得手です。統計モデルは基本的にデータの行・列それぞれの絡み合いをひもとく技術ですので，行列計算ができないのでは統計計算ができないことに等しいのです[*1]。

また，あまり知られていないことですが，Microsoft 社の Excel は数値計算処理において計算精度，乱数発生アルゴリズム等の点で問題があり，専門的用途には耐えない，ということがあります。簡単に数を数え，グラフを書くことは得意でも，高度な数値計算は苦手だと考えた方が良いでしょう。こうしたバグについては，開発元の Microsoft 社も気づいているかもしれませんが，修正されるかどうか，機能が追加されるかどうかは，一私企業の経営方針に依存しているということは問題があるかもしれません。

専門的な統計ソフトを，ということで本書では R 環境を推奨しているわけですが，他にソフトウェアがないわけではありません。SPSS や SAS,JUMP,S-plus,Stata,HALBAW for WIN,M-*p*lus など有名なソフトは多く，長所短所もそれぞれにあります。このなかで，R だけにある特徴が，フリーソフトウェアであるということです。フリーソフトウェアの特徴については次の章で触れたいと思います。

[*1] もちろん表計算ソフトでも，マクロと呼ばれる言語を使って行列計算ができるように工夫することは可能ですが，最初からその機能を持っていないこと，をここでは指摘しています。

1.3　その他の R の特徴

R について，その他の特徴として，最も大きなものは CUI(キャラクタ・ユーザ・インターフェイス，あるいはコマンド・ユーザ・インターフェイス) であるということでしょうか。CUI の対義語は GUI(グラフィカル・ユーザ・インターフェイス）で，ポインタがボタンを押すと分析などが実行される，という画面を持っていることを指します。CUI はそうしたポインタやボタンのような，目に見えた（グラフィカルな）表現をせず，命令文 (コマンド) を文書として打ち込んで (キャラクタ，とは文字列の意味です)，実行させます。こうした入力方法を持っていますので，次のような特徴があります。

1. 大文字と小文字が区別されます
2. コマンドの書き間違えが許されません
3. コマンドがわからないと実行できません
4. 日本語の処理は不得手です
5. ヘルプが英語です

これらの特徴は短所に思えるかもしれませんが，実はそのまま長所にもなります。

1. 大文字と小文字を区別して豊かな表現
2. 正しいコードは必ず完璧に再現します
3. コピー＆ペーストでも動くので覚える必要はありません
4. 全角と半角の違いがないので便利です
5. 世界共通の言語で表現されており，一般性が高く，またヘルプの読み方もコツをつかめば簡単です

どういうソフトウェアにも癖はあるものですから，慣れてしまうことも大事です。そして R は汎用性が高いので，一度慣れてしまうとデータを扱うあらゆるシーンで利用することができるのです。

1.4　なぜ R なのか。まとめ。

統計処理は，データという素材を調理するようなものだと考えてみましょう。

面白そうな素材に対して，様々な調理法で味を変えながら表現し，隅々まで味わってみませんか。そして良い素材であればあるほど，素材の特徴を生かした調理法が必要になってきます。

R 環境はいわば，プロの使用にも耐える調理場です。R 環境の作法をまねることで，クリエイティブな活用をすることができます。R 環境に慣れてしまうと，表計算ソフトや市販の統計ソフトが，電子レンジで温めるだけの料理や値段もカロリーも高いだけのコンビニ弁当のように思えるでしょう。

R 環境は，本体だけでも基本的な統計処理は可能です。しかし，地方の料理や新しい調理法，調理器具等，特殊な手法をしたい場合はどうすればよいのでしょうか。R 環境では，【パッケージ】と呼ばれる「新しいレシピ」を手に入れることで，同じキッチン（R 環境）で実行することができます。パッケージはインターネットを介して提供され，これもフリーソフトウェアの精神に則って公開・配布されています。世界中の R ユーザがレシピを公開していますので，最新・最先端の技術を手元に取り入れることができるのです。

R 本体もインターネット環境を通じて配布されます。追加パッケージや本体のバージョンアップもそうです。もっとも，ダウンロードの際にインターネットへの接続が必要なだけで，一旦必要なものをダウンロードしておけばローカルだけでの環境の構築ができます。

皆さんも今すぐ R 環境に飛び込むべきです。

第 2 章

自由で開かれているということ

2.1 フリーソフトウェアってなんだろう

R は無料で使用できるソフトウェアです。しかし，そもそもなぜ無料で使用できるのでしょうか。また，R を起動すると「R は，自由なソフトウェアであり，『完全に無保証』です」というメッセージが表示されます。このメッセージはいったいどういう意味なのでしょうか。この節では R そのものの使い方に入る前に，「自由なソフトウェア」または「フリーソフトウェア」とはいったい何なのか，ごく簡単ではありますが解説します。

「自分はソフト屋ではないのだから使い方さえわかればそれでいい」，「無料で分析ができたらそれで十分」という考えもあるでしょう。しかし，フリーソフトウェアの発想は，研究にとっても重要な示唆を与えてくれると筆者は考えています。少しの間だけお付き合いください。

2.2 「フリー」とは「無料」のことではありません

「フリーソフトウェア」という表現における「フリー」とは「無料」という意味ではありません。“free beer” というときは「無料のビール」という意味ですが，“free speech” というときは「自由な言論」を意味します。フリーソフトウェアのフリーとは後者，すなわち「自由」を意味します。

では「ソフトウェアが自由である」とはどういう意味なのでしょうか。GNU[1] によれば，ここでいう自由とは次の 4 つの自由を指します。

第 0 の自由　いかなる目的に対しても、プログラムを実行する自由。

第 1 の自由　プログラムがどのように動作しているか研究し，必要に応じて改造する自由。ソースコードへのアクセスは，この前提条件となります。

第 2 の自由　身近な人を助けられるよう，コピーを再配布する自由。

第 3 の自由　改変した版を他に配布する自由。これにより，変更がコミュニティ全体にとって利益となる機会を提供できます。ソースコードへのアクセスは、この前提条件となります。

ここで「ソースコード」という言葉が出てきました。ソースコードとは，そのプログラムのいわば設計書です。第 1 の自由は，プログラムの設計書が入手可能であることを前提としています。たとえば Microsoft の Excel というソフトウェアは，ソースコードが非公開となっています。そのため，Excel を使っていて何らかの問題が問題が生じた場合，利用者が問題の設計書を見て何が原因なのか知ることはできません。できることといえば Microsoft に問題を報告して，直してもらえることを祈る程度です。

「Excel のような有名で高価なソフトウェアに問題が放置されることなんてない」とお考えの方もいるかもしれません。しかし，Excel の統計関数がいまいちなのは有名な話です [2][3]。長らく放置された問題も多いようです [4]。一方

フリーソフトウェアの場合，問題があればソースコードを手に入れて自らの手で修正することが可能です。たとえば LibreOffice というオフィスソフトウェアは，ソースコードが全て公開されていますから，技量さえあれば誰でも修正することが可能なのです。

ここで，フリーソフトウェアの定義に「無料であること」が含まれていないことに注意してください。商用のフリーソフトウェアも存在します。フリーソフトウェアのコピーを売っても構わないのです。

「フリーソフトウェア」とほぼ同じものとして「オープンソースソフトウェア（Open Source Software: OSS）」という表現も存在します。両者はほぼ同様の意味ですが，「自由」であることをより強調した表現が「フリーソフトウェア」，「ソースコードが公開されている」ことを強調した表現が「オープンソースソフトウェア」だと考えてください。この両者を組み合わせて “FOSS(Free and Open Source Software)”，“FLOSS(Free/Libre and Open Source Software)” などとも呼ばれます。次からは FLOSS という表現を用います。なお，FLOSS の対義語として「プロプライエタリソフトウェア」という用語が存在します。これは先に例に挙げた Excel のように，ソースコードが公開されていないソフトウェアです。

ここまで読んでみて，「FLOSS ってなんだか難しそうだな」とお感じになられた方もいるかもしれません。しかし，すでにみなさんのほとんどが FLOSS のお世話になっているはずです。Google で検索をしたことのない人はほぼいないでしょう。Google は Linux という FLOSS で動いていますし，筆者の所属する山口大学の Web サイトは FreeBSD と Apache HTTP Server という FLOSS で動いています。知らず知らずのうちに，みんな FLOSS を使っているのです。

FLOSS ではプロプライエタリソフトウェアとは異なり，開発者と利用者が必ずしも分離しません。ソースコードが公開されているため，利用者の誰もが開発に参加することができます。利用者がほしいと思った機能を自ら実装することもできるのです。FLOSS に関わる人々で構成される団体は多くの場合「コミュニティ」と呼ばれます。

2.3　R はタダではありません？

「FLOSS はすべてタダで使えるのがいい。でもタダだから本物ではない。大したことはできない」というような声を時折目にします。しかし，FLOSS は決してタダではありません。先にも書きましたように，FLOSS は売ってもよいわけです。FLOSS のコミュニティが開発に必要な寄付を募っている事例も数多く存在します。したがって，FLOSS は決して貧乏人のためのソフトウェアではありません。FLOSS における “コスト” の支払い方にはたとえば次のようなものが考えられます。

- 自らソースコードを修正してコミュニティにフィードバックする
- 翻訳する。FLOSS の多くが全世界で開発されており，その多くは英語です。より使いやすくするためには，母国語で使えるように翻訳する必要があります
- 問題を見つけてコミュニティに報告する
- コミュニティにお金を寄付する

このように，FLOSS はお金を支払うか，問題報告をするかしかできない（しかも実際に解決されるかどうかは開発元次第）プロプライエタリソフトウェアと比べて，コストの支払い方がはるかにバリエーションに富んでいます。これは FLOSS の大きな強みです。

しかしこれは裏を返すと，ソフトウェアに問題を見つけたら報告するとか，新たにほしい機能があったら自らプログラムを書いて実装するとかのように，自ら行動する必要のあることを意味します。問題を見つけても報告しなければ，他の誰かが報告しない限り放置されます。英語の FLOSS を日本語で使いたいと思っても，誰も翻訳しなければ英語のままです。「誰かがいつかはやってくれる」ことを期待していても，どうにもならないものはどうにもならないわけです。FLOSS の利用者は同時にそのソフトウェアの当事者であることを求められるわけです。

2.4　自由で開かれた研究

学術研究にプロプライエタリソフトウェアを使うことは，様々な危険をはらんでいます。プロプライエタリソフトウェアは，製造元の都合で消失しえます。「昔のコンピューターで作成した論文のファイルが，対応ソフトがなくて開けない」なんてことはよくある話です[*1]。

SPSS はプロプライエタリソフトウェアですが，SPSS がもしなくなってしまったら，当然のことながら SPSS を使った統計処理はできなくなってしまいます。SPSS で出力した分析結果のファイルも開けなくなってしまうかもしれません。SPSS の分析に誤りを見つけても，利用者にはそれを直すすべはありません。報告しても放置されてしまう可能性だってないとはいえないのです。誤った分析をし続けている可能性だって，決してゼロとは言えないのです。

一方で，FLOSS が消滅することはありません。ソースコードが公開され，自由にコピーすることができるため，古いソフトだってその気になればいまのコンピューターで動くよう細工して動かすことができるのです。

そもそも，学術研究自体が自由で開かれているべきです。公開されている先行研究を自由に利用することで，あらゆる学術研究は発展してきました。研究成果はどこかの一族に伝わる「秘伝の術」ではありません。「自由に閲覧・引用できない論文」に，はたしてどんな意味があるというのでしょうか？

追試可能性に言及するまでもなく，論文に書かれている分析結果は，その手法が明らかでないと再現できません。研究が自由で開かれているためには，その手法も自由で開かれているべきなのです。そういう意味でも，統計処理には FLOSS の一種である R を使ってほしいのです。もちろん「無料で使える」ことも強みですが，研究が自由で開かれているためにも，R が FLOSS であること自体が大きなアドバンテージなのです。

[*1] このように考えると，論文はオープンなフォーマット（TEX や ODF（LibreOffice や Apache OpenOffice で主に使われるフォーマット），OOXML（MS Office で主に使われるフォーマット），PDF など）で作成・公開されるべきです。「自分の書いた論文が 10 年後には読んでもらえなくなっても構わない」と考える読者はいないでしょう？）

第 3 章

はじめよう，R

3.1　R 環境の導入

3.1.1　R に関するインターネットサイト

既に述べたように，R やそのパッケージはインターネット上で公開されています。

R は The Comprehensive R Archive Network(CRAN; 包括的 R アーカイブネットワーク）というネットワークで配信されています。CRAN，あるいは R-Project のサイトである http://cran.r-project.org/ にアクセスすると，R をダウンロードすることができます。自分のプラットフォーム (OS) に合ったものをダウンロードし，インストールしましょう。

インストールの方法等については，インターネット上で検索していただければいろいろ情報が得られるかと思います。

日本語で利用できる，R 情報の中心的取りまとめ役としては RjpWiki(http://www.okada.jp.org/RWiki/) が有名です。こちらのサイトには入門から応用，FAQ まで，様々な情報が寄せられます。

また，山口大学教育学部教育心理学コースの有志がつくった，R オンラインガイド (http://psycho.edu.yamaguchi-u.ac.jp/?page_id=242) は動画でインストールや R 環境の導入，使い方を開設してあります。

他にも，青木繁伸先生のサイト，R による統計解析 (http://aoki2.si.gunma-u.ac.jp/R/) は，様々な分析の解説だけではなく，R のコードも記述してありますので，自分でコードを書く時の勉強にもなります。このサイトはあまりにも出来が良かったため，インターネット上で公開されているにもかかわらず，書籍化されているほどです。

多くのユーザが R を使い，ネットで情報交換をしていますので，キーワードを入れて検索すると何かしらヒントがつかめることが少なくありません。もっとも，「R」というキーワードはアルファベット一つですので，検索キーワードに入れても，それだけではヒットしない（あるいはヒットしすぎてしまう）ことがあります[*1]。そういう時は，R に関する情報を専門的に検索する，seekR(http://seekr.jp/) を使うといいでしょう。

インターネットや書物の力を借りて，無事に R がインストールできることを祈っております。なお，本稿執筆時点 (2013.10.23) で，R の最新バージョンは 3.0.2 です。時々バージョンが上がりますが，本質的な計算機能は変わりありません。しかし環境によっては R を入れ直すことでパッケージ・ライブラリを作り直す必要があったり，パッケージが動かなくなったりすることがありますので，注意してください。

[*1] 検索にかかる力のことを Googlability といいますが，R は Googlability が低いといえます。

3.2　RStudio のすすめ

R を使う時は，この後で詳しく説明しますが，画面にコードを書いていくことで計算したり，グラフを描画したりします。このコードは一行いれると，一行結果が返ってくるということになりますので，エラーが生じてもすぐに気づくことができるという利点があります。

コードを書いて実行するときに感じる不便のひとつが，どこにコードを記録しておこうか迷う，というものです。R にはエディタもありますが，メモ帳などでコードを書いて，コピー・ペーストで実行し，うまくいったものだけ残して保存する，という方法を取っている方もいるでしょう。

コードを保存しておいても，複数のデータを扱ったり，同じデータでも異なる分析をする（異なるコードを書く）ことがあると，混乱してしまうことがあるかもしれません。

そこでご紹介するのが RStudio です。RStudio は統合開発環境とよばれ，R と R にまつわるファイル，関数，変数，パッケージ，図版などをこのソフトだけで管理することができるのです。例えて言うなら，R は飯ごう炊爨のように一つ一つ手作りで進んでいく楽しさがありますが，RStudio はキッチンスタジアム，とまではいかないまでも，システムキッチンでお料理する程度には整えられた環境なのだ，とイメージしていただければと思います。

RStudio は，その計算部分は R 本体をつかいますので，R をインストールした上で，RStudio もインストールしていただきますと，RStudio が自動的に R の場所を探り当て，計算に利用します。

RStudio ではコードやデータファイルのセットを「プロジェクト」としてまとめます。RStudio を起動したら，メニューバーの Project から「Create new project（新しいプロジェクトを始める）」を選んで，ファイル等を置いておくフォルダを指定してから，分析を始めましょう（二回目以降は，Open project で，ファイルを置いておくフォルダの中に作られる.proj ファイルを選択することでプロジェクトを続けることができるようになります）。

3.3　R はインタプリタ

R を起動するとコンソール画面が現れます。あるいは，Rstudio を起動すると四分割した窓（ペイン）が現れますが，そのうちの一つがコンソール・ペイン (Console Pane) になっています。コンソールには「>」の記号が出ているかと思いますが，この記号はプロンプトと呼ばれる「入力待ち状態」を表しています。R はインタプリタ言語です。つまり，我々のやってほしいこと（統計処理）を入力すれば，それをひとつひとつ翻訳・実行して，回答が得られるという，逐次やり取り型の処理体系になっています。一つ命令（コード[*2]）を入力すれば、一つ答えが返ってくる。これの繰り返しですので，一歩ずつ進めていきましょう。また，こちらからの入力に何らかの間違いが含まれている場合，エラーが返ってくることがありますので，すぐに「どこで間違えたのか」がわかるようにもなっています。

R は CUI，すなわち文字入力で命令を実行していきますので，一つ一つの分析命令を文字で書かなければなりません．しかも，これが初心者にとって敷居が高いと感じさせるところです。例えば，R は文字の書き間違いを許しません。特に R では大文字と小文字を区別します ので，たとえば sample/Sample/SAMPLE はすべて異なる対象（命令）をさすことになります。また，日本語も使えないわけではありませんが，全角・半角を入れ替えることが面倒であったり，場所がずれたりすることがありますので，あまりお勧めできません。

しかし，何も全ての命令を覚えて使わなければならないわけではありません。典型的な使い方のパターンがありますし，書籍に掲載されているコードを書き写したり，ヘルプの例やインターネット上のコードをコピー&ペーストで実行してもいいのです。また，RStudio には入力補完機能がついていますから，途中で「この続きどう書くんだっけ」と思えばタブキーでサポートしてくれることもあります。

[*2] 一行の命令をコード，複数のコードをスクリプト，ソース，プログラムなどと呼ぶことがありますが，本書ではコードで表現を統一しています。

また，自分の入力したことを全て記録しておくことが出来ます。GUI，すなわちマウスでクリックしながら分析を進めていくやり方は，確かに最初は簡単に思えますが，「どこのボタンを押して，このような結果になったのかわからない」とか「なんだか勝手に答えが出た」ということがよくあります。これでは周囲の上級者に質問することも出来ませんし，自分で反省して学習することも出来ません。逆に，全て正しく実行されたコードさえ保存しておけば，異なる場所，異なるパソコン上で実行することも可能です。実際，筆者は学生とのやり取りにおいて，元データとコードファイルを共有し，結果はそれぞれのパソコン上で再現するという方法をとっています。この方法ですと，例えば筆者が出張中であったとしても，出先のパソコンで実行して問題点・改善点を発見し，メールで指導するということが可能なのです。

R は教育ツールとしても大変有用であると思います。

3.4　R で計算（四則演算）

さてそれでは実際に R を使ってみましょう。簡単なところで，コード 3.1 の四則演算を実行してみてください。

ソースコード 3.1　R の四則演算

```
> 1+3
[1] 4
> 3-5
[1] -2
> 5*7
[1] 35
> 8/2
[1] 4
> 9\8
 エラー:  予想外の  入力  です  in "9\"
```

加減乗除，がそれぞれ +,-,*,/の記号です。一行目でプロンプト（>）に「1+3 は？」と聞いたところ，二行目で R から「4 ですよ」と答えが返ってきた，というわけです。R からの返信に「[1]」とついていますが，これは気にしないでください（要素 1 のベクトルであるという意味です）。もし入力ミスがあった場合，10 行目にあるようにエラーが返されます。こうして，問題があればすぐにわかりますので，効率よく R 言語を学習していくことが出来ます。

3.5　R で関数

3.5.1　関数の引数

次に関数の使い方を練習しましょう。統計処理は加減乗除の複雑な組み合わせですが，全ての計算を加減乗除だけでやるわけにはいきません。例えば平方根を算出したい場合は，平方根を求める関数を使うことになります。

ソースコード 3.2　平方根の関数

```
> sqrt(16)
[1] 4
```

平方根を求めるには，sqrt（スクェア・ルート）と書き，続く括弧の中に数字を入れます。例では 16 の（正の）平方根を求めよ，と命令し，4 という答えを得ています。ここで，sqrt が関数名です。続く括弧の中身が引数（ひきすう）と呼ばれるもので，関数に与える情報になります。sqrt 関数は平方根を求めるだけですから、引数として数字を一つ渡してやるだけでいいのですが，複雑な統計処理をしようとすると、この引数が複数つくことになります。例えば「A というデータセットをつかって，最尤法で，6 つの因子を抽出し，バリマックス回転した因子分析を実行したい」となると，因子分析関数にデータセット名，因子抽出法，因子数，回転数のオプションを引数として渡してやることになります。

多くの関数には、引数のデフォルト値（既定値）がありますので，全ての引数を指定してやる必要はなく，必要最低限だけ指定するという使い方が一般的です。とはいえ，デフォルト値がどのようになっているかは，しっかりと把握し

ておいた方がよいでしょう。

3.5.2 ヘルプの活用

関数をどのように使えばよいか、どのような引数を与えればよいか，デフォルト値がどのようになっているか，といったこと知るために，関数のヘルプをみることを心がけましょう。R ではヘルプを見るのも help 関数で呼び出す、ということになります。例えば，先ほどの sqrt 関数のヘルプを見るためには、コード 3.3 のように入力します。

ソースコード 3.3　平方根の関数

```
> help(sqrt)
```

これで，例えば RStudio ではヘルプウィンドウに sqrt の使い方が示されます。

ヘルプは一般的に，次の項目からなります。

Description　関数の概略説明
Usage　関数の使い方。引数とデフォルト値の記述
Arguments　引数の解説。
Value　戻り値の解説。複数の結果がある関数の場合，どのような名称で結果が返ってくるかが示されます。
Details　関数の詳細。
References　関数や統計モデルについての出典。
Examples　実行例。

英語での解説になりますが，例えば Details はたいてい丁寧に統計モデルの解説をしてくれていますので，ここを読むだけでも統計モデルの勉強になります。また，Usage のところで引数=”XXXX”, という記述があれば，その引数は「XXXX」をデフォルト値として実行しますという意味ですので，その引数は明示的に記述しなくてもかまわないということでもあります。もちろん，データセット名のように省略できない引数もありますので，注意して見て下さい。

論文などで R を使って分析したことを報告する場合，R のバージョンやパッケージのバージョンについて，記述する必要があるかもしれません。関数ではなくパッケージそのもののヘルプを見るためには，引数として package を使います。

ソースコード 3.4　パッケージのヘルプを表示する

```
> help(package="psych")
```

これでヘルプ画面にパッケージの概略や含まれる関数のリストが表示されるので，一つ一つの関数の使い方を調べることが出来ます。

3.5.3 サンプルデータの活用

R は教育的にも大変優れているなあ，と評価できることの一つに、豊富なサンプルデータをもっていることを挙げることが出来ます。プロンプトで「data()」とすると，R の基本システムが持っているデータの一覧を見ることが出来ます。それがどういったデータなのかは、もちろんヘルプで確認することが出来ます。使ってみたいデータセットを data 関数で呼び出し，あとはそのデータ変数名を使って分析を進めていくことが出来ます。統計モデルの例も，サンプルデータセットをもとに解説されていることが多く，すぐにでも分析を試してみることが出来ますね。

ソースコード 3.5　iris データの呼び出しと要約関数 summary の実行例

```
> data(iris)
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
```

```
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

3.6　R のデータと型

3.6.1　R で代入

R では様々な演算をすることができますが，そのすべてが表示され、プロンプトと結果の中に消えていったのでは，わかりにくくて困ります。そこで，データや結果をいったん R のメモリ内に保持しておくこと，を考えます。これを代入といいます。例えば，四則演算の結果を名前を付けて保持しておくことにしましょう。代入には小なり・ハイフンの記号（< –）を使います。これは矢印を模したものです。

ソースコード 3.6　計算結果の代入

```
> test <- 2+5
>
```

ここでは 2+5 の結果を test というオブジェクト（変数）に代入してみました。画面上は，すぐ次のプロンプトが出て入力待ち状態になっています。計算結果を代入しなさい，という命令をしただけで、表示しなさい，という命令が実行されていないからです。代入されたオブジェクトの中身を知りたい場合は、その名称をそのまま記述します。

ソースコード 3.7　代入結果の表示

```
> test
[1] 7
```

ここでも大文字・小文字，全角・半角の区別がされますので注意が必要です。こうして保管された情報は，そのセッションの間いつでも呼び出すことが出来ますし，オブジェクト名を使って計算を処理することも出来ます。また，同じ名前に別のものを代入すると、その結果は上書きされます。

ソースコード 3.8　オブジェクトの操作と上書き

```
> test * 3
[1] 21
> test <- 5-2
> test
[1] 3
```

3.6.2　数字の型

ここで，R で扱う数字の種類について説明しておきます。R では次の 5 種類の数値を扱うことが出来ます。

実数　numeric，一般的な数字（小数点含む，double 型と integer 型の両方をまとめていうこともある）

整数　integer, 整数型。

複素数　complex, 複素数（実数 + 虚数 i の形）型

文字列　character, 文字列

論理値　logical, 論理型で TRUE と FALSE の二値しかない。数字では TRUE=1,FALSE=0 に対応する。

これに NULL（ヌル，空っぽ）を加えて，全 6 種類の数値のいずれかがオブジェクトに含まれていることになります。論理値はあまり使い慣れない数字かもしれませんが，これはスイッチオン・オフの意味だと考えればいいでしょう。関

数のオプションで使うか使わないか、という二択になる場合は，TRUE/FALSE で指定することがあります。また，大文字の T/F で省略することも出来ます。

3.6.3　変数の型

R では数字にも種類がありますが，この数字をどのように扱うか、というところにも種類があります。まず基本的に，R では数値をベクトル（複数の数字のセット）として処理しようとします。四則演算の例で 1+2 という単純な数と数の足し算をしましたが，これは要素が一つしかないベクトル（スカラー）とベクトルの和，と捉えます。基本はベクトルなのです。

例えば 1,2,3,4,5 という五つの数字をセットとして変数に代入し，その変数に計算操作をすると，全ての要素が処理されます。

ソースコード 3.9　ベクトルの計算

```
> test <- c(1,2,3,4,5)
> test+3
[1] 4 5 6 7 8
```

ここで，c() という関数は「つなげて渡す，combine」というものです。連続した数字の場合は，特にコロンを使って表現することも出来ます。

ソースコード 3.10　連続を表すコロン

```
> 2:8
[1] 2 3 4 5 6 7 8
```

このように数字のセット＝ベクトルが基本ですが，ベクトルのセットは行列 matrix 型になります。

ソースコード 3.11　matrix 型

```
> test <- matrix(c(1:8),nrow=2)
> test
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

matrix 関数で 1 から 8 までの連続数を，2 行 (nrow=2，必然的に 4 列）で表現しています。この matrix 型で作られた変数の要素を指定するには，どの行，列なのかを中括弧 [ ] 付きで指定してやることになります。

ソースコード 3.12　要素の指定

```
> test[1,2]
[1] 3
> test[1,]
[1] 1 3 5 7
> test[,2]
[1] 3 4
```

このようにして数字やベクトルのセットを扱うことが出来ますが，いろいろなものをもっとまとめておきたい，ということがあるかもしれません。例えばデータセットとして，ID, 性別，身長，体重，のように，です。このような変数のセットは list 型と呼ばれます。

ソースコード 3.13　list 型

```
> test <- list(id=c(1,2,3,4),sex=c("male","female"),
height=c(155,163,172),weight=c(70.6,80.9,50.3,40.7))
> test$id
[1] 1 2 3 4
> test$weight
[1] 70.6,80.9,50.3,40.7
```

ここでは test というオブジェクトに，id,sex,height,weight という変数名のついたベクトル（要素は character 型だったり numeric 型だったり様々）を list 型として代入しています。作られたオブジェクトの要素指定は，ドルマーク $ で変数名をつなげて表記します。

しかし実際，我々がデータを扱うときは，上の例のように長さがバラバラになっていることはあまりありませんね。普通の統計データは，一行に一人分のデータが入り，各列に変数が入っている個体×変数の矩形（長方形）のデータです。R ではこれを特に data.frame 型として扱います。

ソースコード 3.14　data.frame 型

```
> test <- data.frame(list(name=c("kosugi","tanaka","suzuki"),
sex=c(1,2,1),height=c(170,160,170),weight=c(70.6,80.9,90.6)))
> test
    name sex height weight
1 kosugi   1   170   70.6
2 tanaka   2   160   80.9
3 suzuki   1   170   90.6
```

data.frame 型は list 型の特殊形で，行番号，変数名のついた行列でもあります。data.frame 型でも変数の指定はドルマーク$，中括弧による行番号・列番号指定も可能です。

さて，この例では性別が 1,2 とコード化されています。名義尺度水準のデータは，このように例えば男性を 1，女性を 2 とコード化することがありますが，この数字は数値型としてあつかってはいけません。数字であって数字でない，連続量ではなく分類番号でしかないものである，ということを示すために，R ではこれを factor 型として指定することが出来ます。factor 型は数字にラベルを付け，ラベルで表現してくれます。

作られた変数がどのような型として扱われているのかを確認するために，str 関数というものがあります。これを見ると，オブジェクトがどのような構造をしているのかが示されます。例では変数 name も factor 型として扱われていること，height,weight は整数・実数ですが numeric 型として扱われていることがわかります。

ソースコード 3.15　factor 型の指定と str 関数

```
> test$sex <- factor(test$sex, labels = c("male", "female"))
> test
    name    sex height weight
1 kosugi   male   170   70.6
2 tanaka female   160   80.9
3 suzuki   male   170   90.6
> str(test)
'data.frame': 3 obs. of  4 variables:
 $ name  : Factor w/ 3 levels "kosugi","suzuki",..: 1 3 2
 $ sex   : Factor w/ 2 levels "male","female": 1 2 1
 $ height : num  170 160 170
 $ weight: num  70.6 80.9 90.6
```

3.7　R とデータ操作

3.7.1　ファイルからデータを読み込む

統計ソフトで統計処理をするとき，データは表計算ソフトなどで作成し，そこから読み込む形にすることが多いかと思います。R でも同様で，データの作成そのものは外部にまかせ，出来上がったデータファイルを読み込む，というのが一般的なやり方になります。

データファイルの作り方に関しては，カンマ区切りのテキスト形式ファイルから読み込むのがいいでしょう。このファイル形式は.csv ファイルといいますが，読み込みには次の関数を使います。

ソースコード 3.16　ファイル読み込み関数

```
> sample <- read.csv(file.choose(),head=TRUE,na.strings="*")
```

この read.csv 関数を実行すると，見たことのあるファイル選択画面があらわれますので，ここで読み込みたい csv ファイルをクリックして選択します。すると，ファイルがデータフレーム型で読み込まれます。read.csv 関数の第一引数である file.choose 関数は，あるファイルのパスを取得する関数なのです。第二引数の head=TRUE は，データの一行目が変数名になっていることを意味し，第三引数はデータの中の欠損値をアスタリスク (*) にしてあることを指定しています。第二引数はデフォルトで TRUE ですが, 第三引数のデフォルトはありませんので，指定しなければ欠損値記号もデータとして読み込み，例えばアスタリスクのある列は文字列変数として扱われることになりますので，注意が必要です。str() 関数や summary() 関数で構造，要約を表示させ，読み込まれた内容に間違いがないか確認しておきましょう。

ところで，ファイルパスの指定は長くなると面倒なものですが，RStudio のプロジェクトを使う場合，プロジェクトフォルダが自動的にワークディレクトリになっており，パスの指定が必要なくなります。プロジェクトフォルダの中にデータファイルを入れ，その中で一括管理すると大変便利です。

3.7.2　データの整形

変数の作成

読み込まれたデータを使って計算する際，新たに変数を作成したり，計算して変数として保存することが，実際場面ではよくあります。例えば身長と体重のデータから，新たに BMI を変数として算出したいとします。test データセットに新たに BMI 変数を作りたい場合，ドルマーク$で新しい変数名を指定して代入すれば出来ることになります。

ソースコード 3.17　計算による変数の作成

```
> test <- data.frame(list(name=c("kosugi","tanaka","suzuki"),
+ sex=c(1,2,1),height=c(170,160,170),weight=c(70.6,80.9,90.6)))
> test
    name sex height weight
1 kosugi   1    170   70.6
2 tanaka   2    160   80.9
3 suzuki   1    170   90.6
> test$BMI <- test$weight/(test$height/100)^2
> test
    name sex height weight      BMI
1 kosugi   1    170   70.6 24.42907
2 tanaka   2    160   80.9 31.60156
3 suzuki   1    170   90.6 31.34948
```

邪魔な変数を削除したい場合は，NULL によって変数を空にしてください。

ソースコード 3.18　変数の削除

```
> test$sex <- NULL
> test
    name height weight      BMI
1 kosugi    170   70.6 24.42907
2 tanaka    160   80.9 31.60156
3 suzuki    170   90.6 31.34948
```

subset() 関数による変数の抜き出し

変数が多くある中で，途中の数行だけ抜き出して別のデータセットを作りたいことがあります。その場合は，変数列が連続している場合は列の指定でできますが，選択する必要がある場合は subset() 関数を使い，引き抜く変数を select オプションで指定します。。

ソースコード 3.19　部分的に変数を取り出す

```
> subdata <- test[2:4]
> subdata2 <- subset(test,select=c("name","BMI")
```

subset 関数のオプション指定は他にも可能で，条件を指定して抜き出すといったことも可能です。あるいは，complete.cases() 関数を使うことで欠損値のないデータセットを作ることも可能です。次の例では，一行目は身長 170cm 以上のデータを抜き出しています。下の行は欠損値のないデータセットにしています。

ソースコード 3.20　subset() による抜き出し方様々

```
> subdata3 <- subset(test,test$height>=170)
> subdata4 <- subset(test,complete.cases(test))
```

rbind(),cbind(),merge() 関数によるデータの結合

データセットに別のデータセットを引っ付ける場合は，rbind() 関数や cbind() 関数です。row 方向の結合，column 方向の結合の略ですので，rbind() 関数はデータセットの下に結合，cbind() はデータセットの右に結合します。それぞれ列数，行数が等しくない場合はエラーが返されます。

このようなデータの結合の他に，「名寄せ」のような，同じ個体が別々のデータセットに入っているが対応するデータ同士は合わせて結合したい，ということもあるかもしれません。このようなときは，merge() 関数を使います。「merge(x,y,by=キーになる変数)」という形式で指定し，データセット X の A という変数，データセット Y の B という変数で同じ値を持つものは同じケースとしてまとめる，という操作が可能です。詳しくは merge() 関数のヘルプをご覧ください。

ifelse() 関数による条件分岐

最後に，ifelse() 関数によるデータセットの編集について触れておきます。ifelse() 関数は「ifelse(条件，真，偽)」のように記述し，「もし条件〜が真（該当する）ならばこの処理を，偽（該当しない）ならばこの処理を」という条件分岐関数です。

工夫次第で様々な用途がある関数ですが，データ処理の場合よくあるのは，元データにおかしな数値が紛れ込んでいる場合です。例えば 5 件法のデータであるはずなのに，変数に 23 という数字が入っている，といった場合です。このようなときは，元データに戻って入力し直すことになりますが，戻れない場合は例えば欠損値として処理することになるかもしれません。そのような場合は，次のように記述します。

ソースコード 3.21　ifelse() による外れ値の処理

```
> subdata$item <- ifelse(subdata$item>5,NA,subdata$item)
```

このコードは，subdata というデータセットの item 変数がもし 5 を超えていたら，NA(欠損値) をその箇所に代入し，そうでなければ元の数値をそのまま使う，という命令文です。

もちろん左辺を他の変数名にしたり，他の条件，処理を書くことが可能です。大事なのは，R はこれらの処理も全て記録としてコードに残しておくことが可能ですので，元データは完成した時点で保持し，変数の作成，変更，処理などのプロセスを記録し可視化しておくことで客観性を担保することでしょう。

3.8　図やグラフの描画

R で図やグラフを描画する練習をしてみましょう。

まずは元になるデータセットが必要ですが、R がもっているサンプルデータでもっとも有名なもののひとつ，iris（アヤメ）データを使うことにしましょう。iris データは次のようにして読み込めます。

ソースコード 3.22　iris データの読み込みと概要

```
> data(iris)
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

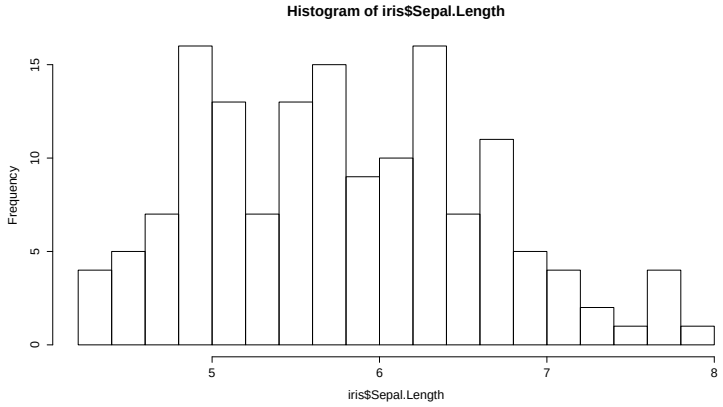


図 3.1　20 分割で表示したヒストグラム

Sepal.Length はがく片の長さ，Sepal.Width はがく片の幅，Petal.Length は花びらの長さ，Petal.Width は花びらの幅を意味します。Species はアヤメの三つの種類（セトサ，バージカラー，バージニカ）で，名義尺度水準のデータとして用意されています。

まずは，がく片の長さがどのような分布をしているか、ヒストグラムを描いてみましょう。

ソースコード 3.23　ヒストグラムによるデータの表現

```
> hist(iris$Sepal.Length)
```

適当なサイズに区切ったヒストグラムが描かれるかと思います。区切りの単位を変えるには，breaks オプションで何分割するかを指定します。

ソースコード 3.24　ヒストグラムによるデータの表現

```
> hist(iris$Sepal.Length,breaks=20)
```

棒グラフを描くのは barplot 関数です。

ソースコード 3.25　棒グラフ

```
> barplot(table(iris$Species))
```

データの群ごとの散らばりを見るには箱ひげ図，boxplot が便利です。

ソースコード 3.26　箱ひげ図によるデータの表現

```
> boxplot(Sepal.Length~Species,data=iris)
```

箱ひげ図は，箱の中央太線が平均値，箱の上端と下端が第 3・第 1 の四分位，上下の線の先端の値は、それぞれ四分位数プラスマイナス 1.5*(第 3 四分位数-第 1 四分位数) になっています。それ以上の所にあるデータは外れ値として，○が表示されています。

次に二変数の関係を表す例を示します。まずは散布図を描きましょう。

ソースコード 3.27　散布図のプロット

```
> plot(iris$Sepal.Length,iris$Sepal.Width)
```

plot 関数は様々な描画ができます。iris データセット全体を plot 関数にいれると，変数同士の組み合わせによる散布図を複数描きます。

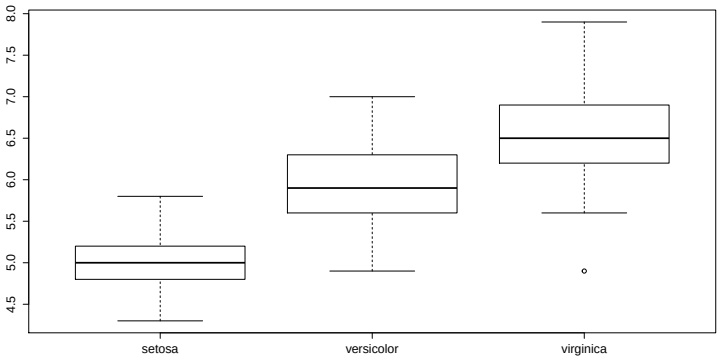


図 3.2　箱ひげ図

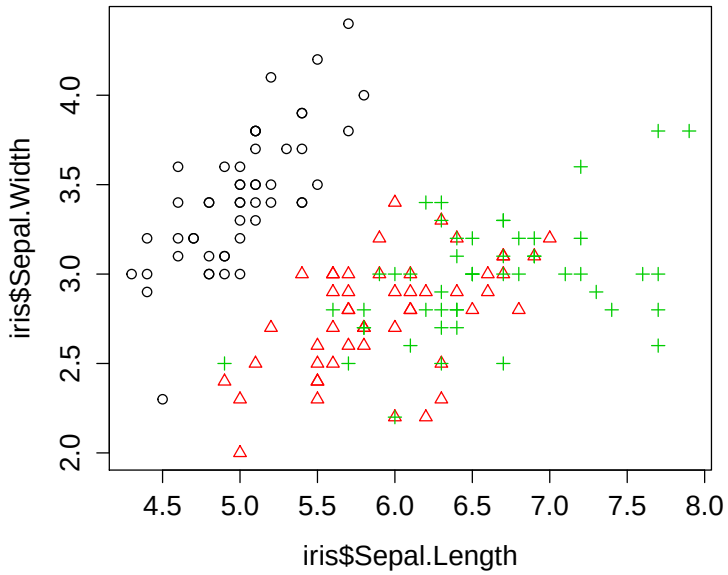


図 3.3　散布図の例

ソースコード 3.28　散布図のプロットその 2

```
> plot(iris)
```

プロットする図形を変えるには，pch オプションを使います。色々な数字を試してみてください。色の指定は col オプション，散布図の点を線で結ぶとき等は type オプションを使います。

ソースコード 3.29　散布図のプロットその 3

```
> plot(iris$Sepal.Length,pch=3)
> plot(iris$Sepal.Length,col="blue")
> plot(iris$Sepal.Length,type="l")
```

これらのオプションを複合的に用いたのが次のコードです。結果は図 3.3 にある通りです。

ソースコード 3.30　散布図のプロットその 3

```
> plot(iris$Sepal.Length, iris$Sepal.Width,pch=as.numeric(iris$Species),col=iris$Species)
```

Rstudio を使うと，作図されたものは Export ボタンを使って，画像ファイル，PDF ファイル，クリップボードに出力できます。サイズも指定できますので，大変便利です。

R には他にも，円グラフを書くための pie 関数，三次元プロットのための image 関数，contour 関数，psesp 関数などがありますが，多すぎてとても紹介しきれません。そこで，どのような図がかけるのか，デモンストレーションを見てみましょう。次のコードを実行すると，return キーを押すたびに次々と図が変わっていくので，見ているだけでも大変面白いですよ。

ソースコード 3.31　demo 関数

```
demo(graphics)
```

第 4 章

これは使える!パッケージ紹介

ここでは便利なパッケージをご紹介します。R は 4000 を超えるパッケージが提供されていますが，これほど多いとどのパッケージを入れれば便利なのか，迷子になってしまうことは少なくありません。筆者の実感として，「このような分析がしたい」というイメージや，「こんな分析があるんだ」と新しい分析名に出会った場合，キーワード＋ package で検索すればほぼ確実に R で実行できていますが，できれば事前に知っておきたいものです。このセクションが皆様のパッケージ迷子を少しでも解消する手助けになればと願っております。

4.1 初心者のための GUI ラッパーたち

まずは R 初心者のための GUI ラッパーパッケージをご紹介します。R はコマンドを入力して結果が返ってくる，という CUI を採択していますが，パソコン初心者にとってこれは大変不便なものであるように受け止められます。例えば t 検定がしたい，というときでも，t.test 関数という関数名を知っていなければ実行できませんし，ttest や T.test のように表記をチョコッと間違えるだけで実行できないからです[*1]。

そんな時，GUI であれば，例えば「分析＞平均の差の検定＞ t 検定＞対応あり/なし」のようにメニューバーから選んでいけると大変便利ですね。R には十分な分析力がありますが，見た目で人を寄せ付けないのは大変残念です。

そこで，GUI でコマンドを「代筆」してくれる GUI ラッパーパッケージがいくつか開発されています[*2]。

4.1.1 ラッパーの古株，Rcmdr

R コマンダーは R 業界では古株の GUI ラッパーです。実は筆者も R を始めた頃は，R コマンダーを使って R のコマンドや表記法を勉強しました。R コマンダーはパッケージで提供されています。大文字と小文字の区別に注意して，Rcmdr パッケージをインストールし，library 関数で実装すると R コマンダーの画面が表示されます（最初のインストール時は，関連パッケージを読み込むよう要請されるので，少し時間がかかります)。R コマンダーを使うと，例えばファイルを読み込むときもマウスをクリックして，画面と対話しながら進めていくことができます (図 4.4)。

画面の指示に従って，どういうファイルが，どこに必要なのかを選んで進んで実行すると，R コマンダーの画面上にコードが書かれます。この R コードを自動的に実行した結果が，画面下に表示されます。ファイルの読み込みをすれば，R コマンダーの中にファイルセットが読み込まれますので，これを使って分析したり，描画したりすることができます (図 4.6)。

[*1] もっとも t 検定と T 検定は別物ですが。

[*2] ラッパーはラップ演奏者ではなく，かぶせる，という意味の wrap です。

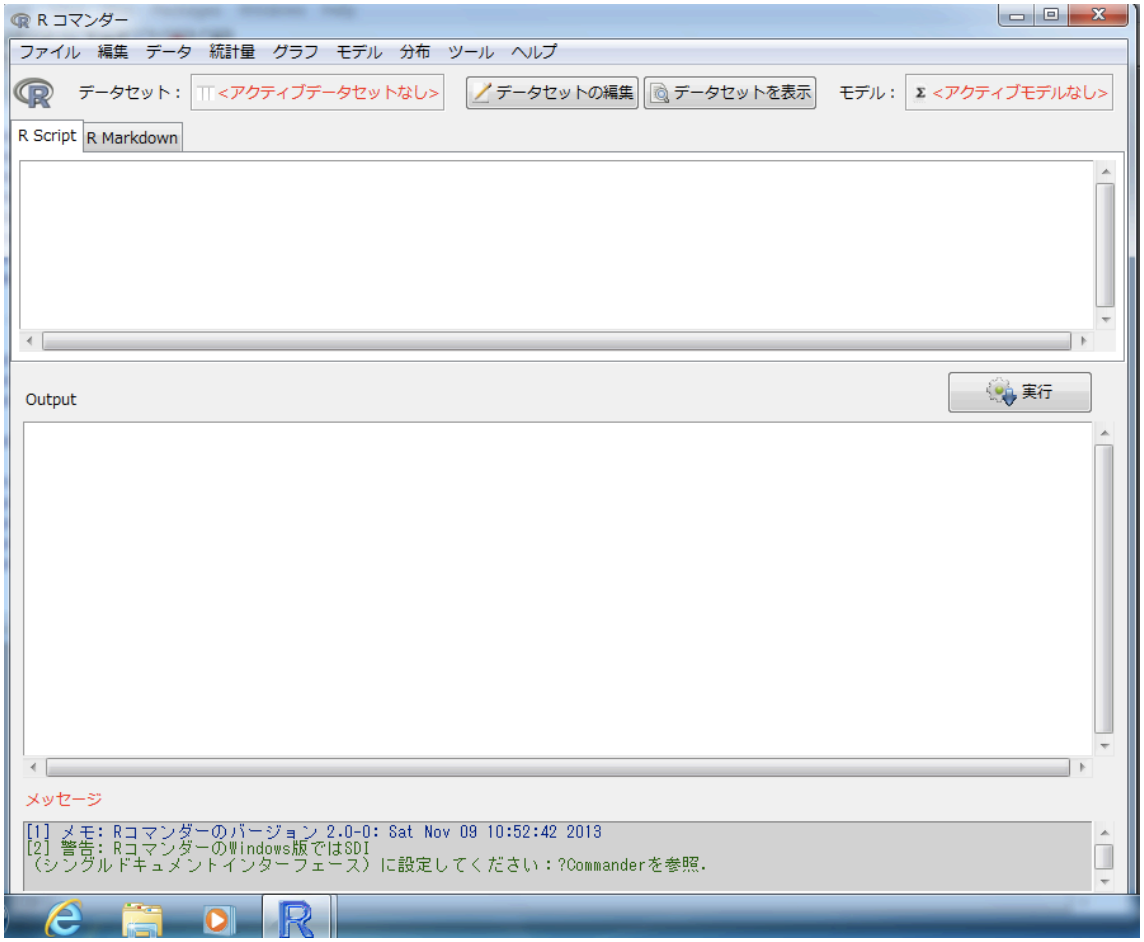


図 4.1　R コマンダーの画面

4.1.2　R コマンダーをさらに高機能に！ EZR

R コマンダーは，多くのプラグインを有しており，プラグインを追加することでさらに様々な分析が R コマンダーに追加されることになります。プラグインも R のパッケージとして提供されており，RcmdrPlugin.***という名称で統一されています。中でも，RcmdrPlugin.EZR として提供されている EazyR は，R コマンダーをさらに高機能にしたもので，かなり複雑な分析もできるようになっています。興味がある方は是非試してみてください。

この他にも，GUI ラッパーとして Rz というものがあります。林真広氏が開発したパッケージで，様々なファイル形式のデータの読み込みや，変数の加工，管理を得意とするパッケージです (図 4.8)。また，Mac ユーザには MacR という大変優れた GUI ラッパーがあります (図 4.9)。MacR は Mac のもつ美しさを損なうことなく，R の優れた統計解析力を利用することができ，市販の統計アプリケーションを買うよりよいとの声もあります。

これらの GUI ラッパーは，R 本体に GUI というカバーで覆うタイプのもので，R のコードをラッパーの中で書いているものです。一方，Rstudio は操作そのものは分析者にゆだね，その周辺として関数やパッケージの管理，ファイル操作，ソース・履歴の管理，プロットの出力補助など R 利用環境の周辺を整える統合環境です。お好みに応じて導入を検討してみてください。

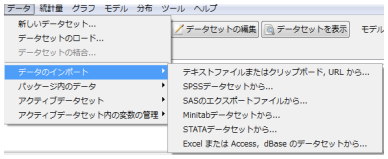


図 4.2　R コマンダーでデータを読み込む

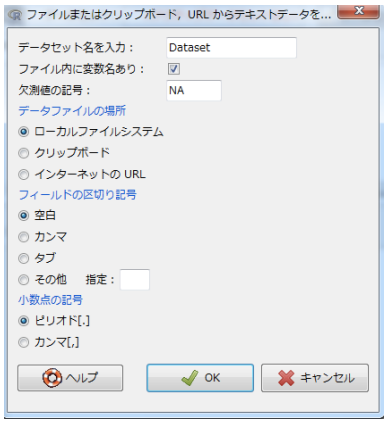


図 4.3　データファイルの指定

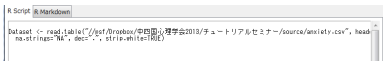


図 4.4　代筆されます

4.2　心理学ユーザのための psych パッケージ

このあとの 7 章，やその中の 7.3 節でもたびたびでてきますが，統計を心理学領域で使う人に取って必要な関数，便利な関数をまとめていることで有名なのが，psych パッケージです。

psych パッケージは，パーソナリティ，サイコメトリクス，実験心理学のための多くの決まりきった手順を R で簡単にするためのパッケージで，因子分析で尺度を作ったり，クラスター分析や信頼性分析，記述統計，順序尺度・名義尺度水準の因子分析（項目反応理論，polycor パッケージを使います），などが含まれています。

いくつか特徴的な関数の例を示しましょう。データは iris データを使います。iris データはアヤメの 3 つの種類がありますが，この 3 つの群ごとの記述統計をしたい時，describeBy 関数が便利です。

ソースコード 4.1　describeBy 関数

```
> describeBy(iris$Sepal.Length,iris$Species)
group: setosa
  var  n mean   sd median trimmed mad min max range skew kurtosis   se
1   1 50 5.01 0.35      5       5 0.3 4.3 5.8   1.5 0.11    -0.45 0.05
---------------------------------------------------------------------------
group: versicolor
  var  n mean   sd median trimmed  mad min max range skew kurtosis   se
1   1 50 5.94 0.52    5.9    5.94 0.52 4.9   7   2.1  0.1    -0.69 0.07
---------------------------------------------------------------------------
group: virginica
  var  n mean   sd median trimmed  mad min max range skew kurtosis   se
1   1 50 6.59 0.64    6.5    6.57 0.59 4.9 7.9     3 0.11     -0.2 0.09
```

群ごとの平均，標準偏差，中央値などが示されています。

偏相関係数は，統制変数で回帰した残差同士の相関係数です。この偏相関係数を算出するのも，psych パッケージに含まれる partial.r を使います。iris データの 4 つの量的変数で相関係数を算出するのは，cor 関数でいいのですが，4 番

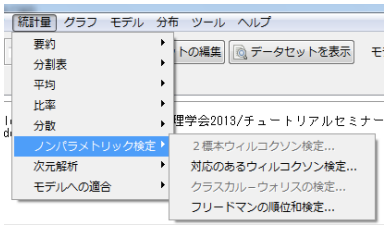


図 4.5　R コマンダーの分析画面

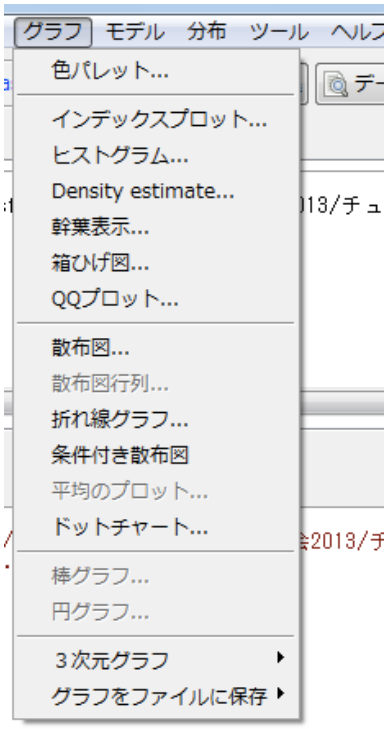


図 4.6　R コマンダーでグラフ描画

目の変数 Petal.Width で統制した偏相関係数を算出するためには，partial.r 関数を使います。

ソースコード 4.2　partial.r 関数

```
> cor.mat <- cor(iris[,-5])
> cor.mat
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
> partial.r(cor.mat,c(1,2,3),4)
partial correlations
             Sepal.Length Sepal.Width Petal.Length
Sepal.Length         1.00        0.34         0.54
Sepal.Width          0.34        1.00        -0.30
Petal.Length         0.54       -0.30         1.00
```

この他にも様々な関数が用意されているパッケージです。興味を持たれた方は，test.psych() と入力するとデモンストレーションを見ることができますので，楽しんでみてください。

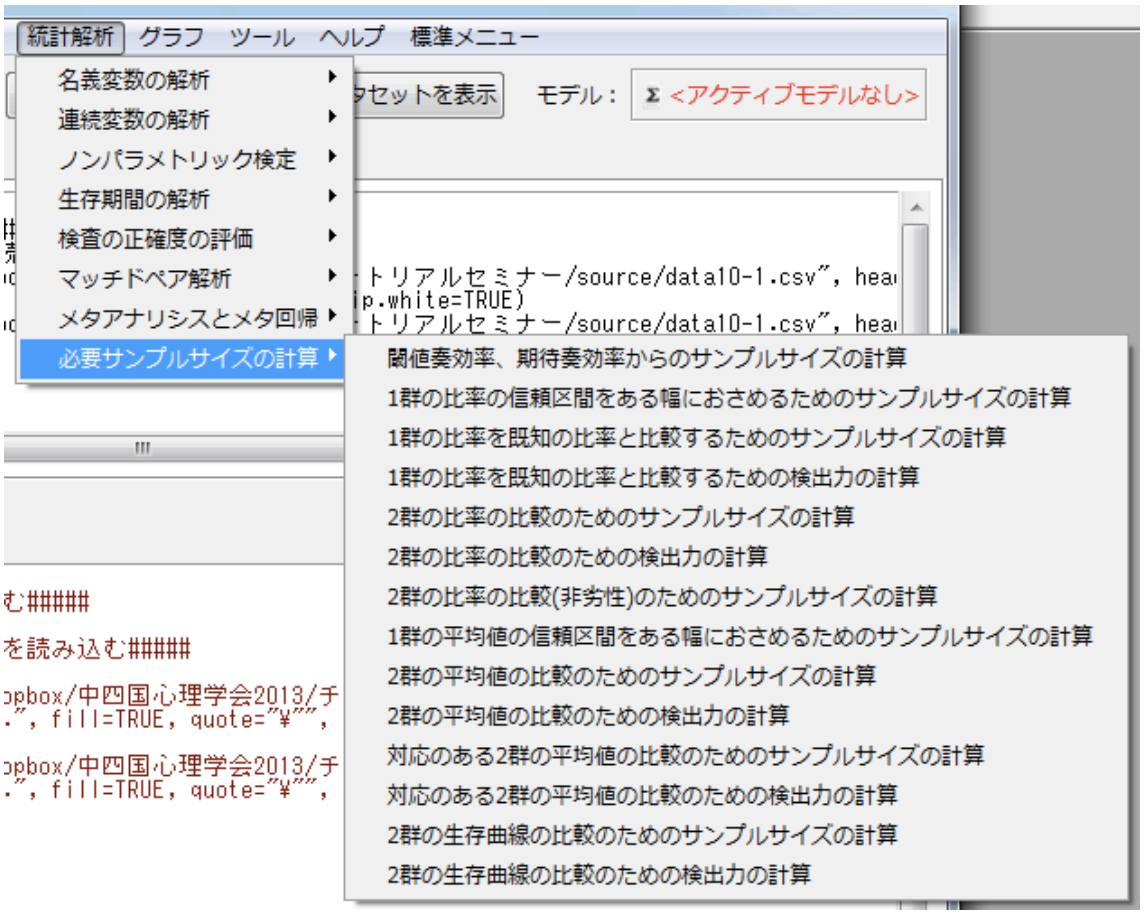


図 4.7　EZR の画面

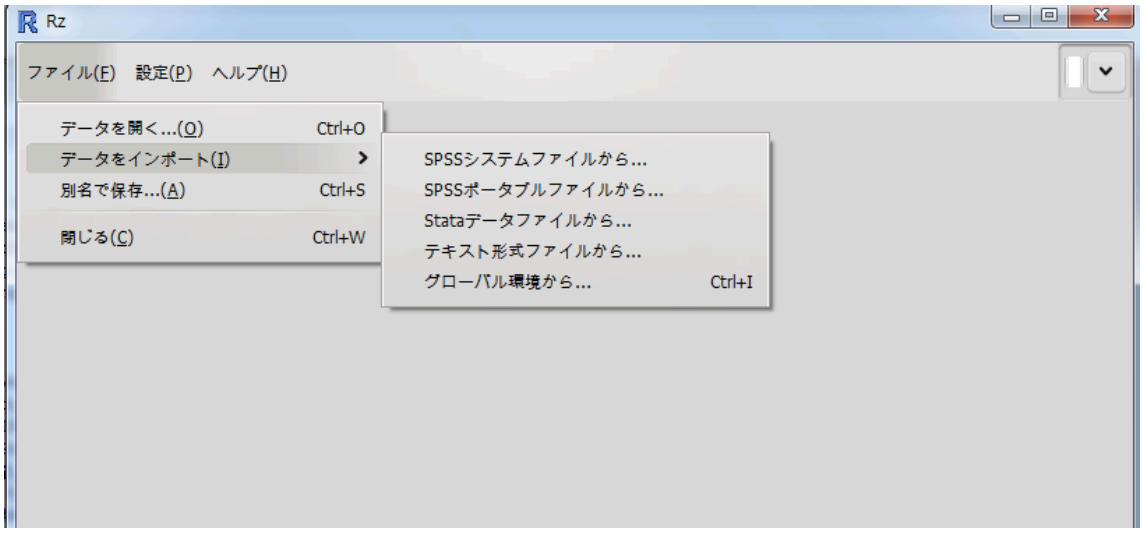


図 4.8　Rz の画面

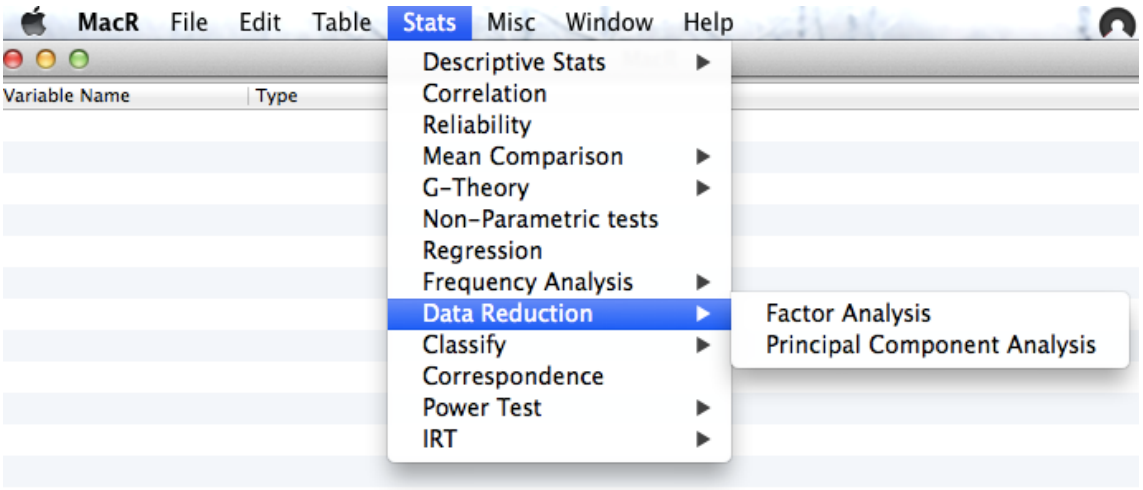


図 4.9　MacR の画面

4.3　っていうか心理学系のパッケージを全部いれたい

もちろん psych パッケージ以外にも，心理学系のパッケージは色々ありますが，ひとつひとつ紹介したりインストールしたりするのは大変です。まとめて同じ分野のパッケージを入れたいという人は，ctv パッケージを使うといいでしょう。このパッケージをダウンロードし，library 関数で実装，心理学関係のパッケージが欲しい場合は install.views 関数で次のように入力します。

ソースコード 4.3　ctv パッケージの install.views 関数

```
install.packages("ctv")
library(ctv)
install.views("Psychometrics")
```

すると次々と関係あるパッケージが導入されます。時間があるときにパッケージのヘルプを見て回るだけで，色々な勉強になること請け合いです。

Psychometrics 以外にも，Bayesian,MachineLarning,Multivariate,Spatial,Survival などのパッケージもいれてみましょう。どういったパッケージ群があるか知りたければ available.views 関数を実行するといいでしょう。

ソースコード 4.4　install.views 関数

```
> available.views()

CRAN Task Views
---------------
Name: Bayesian
Topic: Bayesian Inference
Maintainer: Jong Hee Park
Repository: http://cran.ism.ac.jp/
---------------
Name: ChemPhys
Topic: Chemometrics and Computational Physics
Maintainer: Katharine Mullen
... (以下略)
```

RjpWiki2 等で ctv パッケージを検索すれば，どのようなパッケージグループがあるか確認することができます。ctv パッケージで指定できるパッケージ群の一覧を表 4.1 にまとめました

表 4.1　ctv パッケージで指定できるパッケージ群 (2013 年 4 月現在)

引数	関係領域
Bayesian	ベイズ推定
ChemPhys	計量化学やコンピュータ物理
ClinicalTrials	臨床実験デザイン，モニタリング，分析
Cluster	クラスター分析と有限混合モデル
DifferentialEquations	微分方程式
Distributions	確率分布
Econometrics	コンピュータ経済学
Environmetrics	生態学や環境データの分析
ExperimentalDesign	実験デザインと実験データの分析
Finance	ファイナンス
Genetics	遺伝統計
Graphics	画面描写，ダイナミック・グラフィクス，視覚化
HighPerformanceComputing	高機能かつ並列コンピューティング
MachineLearning	機械学習・統計学習
MedicalImaging	医療イメージ分析
Multivariate	多変量統計
NaturalLanguageProcessing	自然言語処理
OfficialStatistics	官庁統計と調査手法
Optimization	最適化と数値計算
Pharmacokinetics	薬物動態データの分析
Phylogenetics	系統学，特に比較法
Psychometrics	心理統計
ReproducibleResearch	再現可能性の研究
Robust	ロバスト統計
SocialSciences	社会科学のための統計学
Spatial	空間データの分析
SpatioTemporal	時空間データの把握と分析
Survival	生存分析
TimeSeries	時系列分析
gR	グラフィカルモデル

4.4　その他のパッケージ一挙紹介

その他，便利なパッケージを列記します。ctv の Psychometrics に含まれているものもありますが，興味があるものにチャレンジしてみてください。

e1071　サポートベクターマシンやナイーブベイズなど，複雑な分類モデルを実行するためのパッケージです。

ggplot2　図形描画パッケージで，うまく使うと画像出力が格段にきれいになります。

glmmML　一般化線形混合分布モデル (Generalized Linear Mixed Model:GLMM) を実行するためのパッケージです。お勧めはこちらですが，lme4 パッケージも参考にしてください。

GPArotation　7 の章でも触れましたが，様々な因子軸の回転を実行するためのパッケージです。

kohonen　自己組織化マップ Self-Organization Mapping をするためのパッケージです。お勧めはこちらですが，som パッケージも参考

lavaan　構造方程式モデリングを実装するパッケージです。お勧めはこちらですが，sem パッケージも参考にしてください。

lme4　一般化線形混合分布モデル (Generalized Linear Mixed Model:GLMM) を実行するためのパッケージです。glmmML パッケージも参考にしてください。

ltm　潜在特性モデル (Latent Trait Model)，別名項目反応理論 (Item Response Theory) を実行するパッケージです。多次元 IRT については mirt パッケージを参考にしてください。

MBESS　効果量や信頼区間の推定につかう関数が含まれているパッケージです。

Mclust　モデルに基づいたクラスタリング，すなわち混合分布を仮定した分類をするためのパッケージです。お勧めはこちらですが，poLCA パッケージも参考にしてください。

MCMCpack　マルコフ連鎖モンテカルロ法によるベイズ推定を行う関数群をまとめたパッケージです。

mi　欠損値を補完する，多重代入法をするためのパッケージです。

mirt　多次元項目反応理論，すなわちカテゴリカルな観測変数に対する多因子モデルを実行するパッケージです。単因子の IRT は ltm モデルを参考にしてください。構造方程式モデリングの中でカテゴリカルな観測変数を扱う場合は lavaan パッケージが便利です。

mlogit　多重ロジスティック回帰分析（多項カテゴリカル変数を従属変数にした回帰分析）をする場合のパッケージです。

mvpart　決定木分析をするためのパッケージです。

sem　構造方程式モデリングを実装するパッケージです。lavaan パッケージも参考にしてください。

som　自己組織化マップ Self-Organization Mapping をするためのパッケージです。kohonen パッケージも参考にしてください。

poLCA　モデルに基づいたクラスタリング，すなわち混合分布を仮定した分類をするためのパッケージです。お勧めはこちらですが，Mclust パッケージも参考にしてください。

PVAClone　尤度に基づく個体群生存可能性分析を行うパッケージです。

pwr　効果量を算出する関数群が含まれているパッケージです。

quantreg　分位点回帰をするためのパッケージです。

RandomForest　複数の回帰木を発生させ（森のように），より精度の高い機械学習をするランダムフォレストモデルを実行するためのパッケージです。

4.5　パッケージではないけど便利なもの

R は関数を書くことで，自分のやりたいモデルを作り上げることができます。行列演算や最適化の関数が最初から含まれているので，自分で全ての計算アルゴリズムを書く必要もありません。車輪の再発見をする必要はないのです。

パッケージは便利な関数をまとめたものですが，自作の関数も便利なものであればインターネットを介して共有することができます。ここでは関数を公開しているサイトのうち，大変有用な二つのサイト，関数を紹介します。

4.5.1 青木先生のサイトは本当に役に立ちます

群馬大学の青木繁伸先生が公開しているサイト (http://aoki2.si.gunma-u.ac.jp/R/) には，数多くの統計関数が用意されており，そのソースも公開されています。実際に使うときは，ソースファイルをダウンロードして保存するか，インターネットに繋がる環境でサイトに用意されているコードを R にはりつけ，実行するとできます。

全ての関数を一括ダウンロードするには，次のコードを書きます。

ソースコード 4.5 青木先生の R ソースを一括ダウンロードする

```
source("http://aoki2.si.gunma-u.ac.jp/R/src/all.R", encoding="euc-jp")
```

関数の使い方も詳しく説明されていますし，解説のページにもリンクされているので，大変勉強になるサイトです。ソースコードまで完全に公開されているので，関数を書く書き方の勉強にもなります。是非活用してください。

4.5.2 anovakun で捗ります

R で分散分析を行う場合，aov 関数が最初から実装されています。しかし，aov 関数の結果はよく言えば非常にシンプル，悪く言えばあっさりしすぎているところがあります。特に心理学系の分散分析の場合，複雑な実験計画になっていたり，その際の下位検定，効果量の算出となると，様々な修正や仮定のチェックが必要で，一般的な統計ソフトウェアでも完全に対応しているものは少ないでしょう。

広島女学院大学の桐木建始先生が開発された ANOVA4 というソフトウェアは，群内要因をもつ実験デザイン（反復測定）でも下位検定までしっかり算出してくれるソフトウェアでしたが，知る人ぞ知る，という存在で，筆者も直接面識のある先生経由でフロッピー媒体で（！）譲っていただいたことがあります。その後，その大変便利な ANOVA4 が web 上でできるように改訂され，ANOVA4 on the Web(http://www.hju.ac.jp/~kiriki/anova4/) として公開されました。データをブラウザから直接入力してもよし，クリップボード経由で貼付けてもよしで，インターネットに接続されている環境であれば誰でも手軽に分散分析ができます。

ANOVA4 の「4」は，群内要因と群間要因を合わせて 4 要因までの分散分析に対応している，という意味です。分散分析の考え方からいえば，4 要因で十分ですが，こうした上限をこえ，群内要因の修正も最新の理論に対応し，R で分析できる関数として公開されているのが現在理化学研究所 理研 BSI-トヨタ連携センターの井関龍太先生の「ANOVA 君」です（http://riseki.php.xdomain.jp/index.php）。

ANOVA 君の特徴は次のようなものです。

- 被験者間要因（独立測度），被験者内要因（反復測度）のいずれか，または，両方を含むタイプの分散分析を扱います。
- 単純主効果の検定を行います（一次の交互作用についてのみ）。
- 多重比較を行います（修正 Bonferroni の方法による）。
- 非釣り合い型計画（unbalanced design）に対応しています（タイプ II，III 平方和の計算）。
- 球面性検定と自由度調整を行います。
- 効果量をオプションひとつで算出できます。

ANOVA 君のインストールは，サイトで公開されているソースファイルをダウンロードし，source 関数で読み込むことで分析を取り込むことができます。使い方は非常にシンプルで，anovakun 関数に，データファイルを ANOVA4 と同様の形式にして引き渡すこと，分析デザインを A,B,C などのアルファベットを使った記号で引き渡すこと，という二点を抑えておけば非常に丁寧で詳しい結果を得ることができます。

是非研究に活用してみてください。

第 5 章

平均値の差の検定

ここでは心理学に置ける基本的な統計処理のひとつ，平均の差の検定を R でどのように行うのかを解説します。

5.1 分析の流れ

統計的仮説検定の流れ，すなわち

1. 帰無仮説と対立仮説の設定
2. 危険率すなわち有意水準の設定
3. 統計量の算出
4. 仮説の採択、棄却の判断

という概略は既に知っているという前提でお話しします。この点が既に不明な方は，一般的な統計のテキストを参照してください。

5.2 対応のない t 検定の場合

5.2.1 従来の方法

平均値の差の検定，特に二群間の平均の差の検定では一般に t 検定が用いられます。t 検定を行うには，次の手順で行われます。

1. 二群が正規分布しているかどうか，正規性の検定
2. 分散の等質性の検定。等分散性の仮定が保たれているようであれば，Student の t 分布をつかった t 検定。等分散性の仮定が保たれていないようであれば，ウェルチの補正をした t 検定
3. 正規性が確認できないようであれば，ウィルコクスンの順位和検定などノンパラメトリック検定

もっとも，正規性の検定をいちいち確認することは少ないかもしれません (良いことではありませんが)。また，等分散性の仮定については，仮定が保たれていようといまいと，Welch の補正をすることが多いです。少なくとも，R のデフォルト関数 t.test() では，デフォルトが等分散性の仮定をみないようになっており，等分散性を仮定する場合は特に var.equal=T として実行することになります。

実際に，t 検定を行った際の出力例を見てみましょう。ここでは逐一，上の手順に沿って分析を行うものとします。正規性の検定を行うには，シャピロ・ウィルク検定かコルモゴロフ・スミルノフ検定で、関数はそれぞれ shapiro.test または ks.test 関数です (ソースコード 5.1)。

ソースコード 5.1　正規性の検定

```
> shapiro.test(experiment$E)

  Shapiro-Wilk normality test

data:  experiment$E
W = 0.9193, p-value = 0.2457

> ks.test(experiment$E,"pnorm",mean=mean(experiment$E),sd=sd(experiment$E))

  One-sample Kolmogorov-Smirnov test

data:  experiment$E
D = 0.1622, p-value = 0.8838
alternative hypothesis: two-sided

 警告メッセージ:
In ks.test(experiment$E, "pnorm", mean = mean(experiment$E), sd = sd(experiment$E)) :
   コルモゴロフ・スミノフ検定において、タイは現れるべきではありません
```

警告にあるように，コルモゴロフ・スミルノフ検定は連続変数に対して行われる検定ですので，同じ値を取る数字 (タイ) がデータセットに含まれているのは良くない，という警告が出ています。いずれにせよ，どちらの検定でも「正規分布に従っていない」という帰無仮説が棄却される水準の確率ですので、正規分布していると見なして分析を進めます。

次に等分散性の検定は var.test 関数を使います (ソースコード 5.2)

ソースコード 5.2　等分散性の仮定

```
> var.test(experiment$E,experiment$C)

  F test to compare two variances

data:  experiment$E and experiment$C
F = 0.7337, num df = 12, denom df = 12, p-value = 0.6001
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.2238735 2.4045240
sample estimates:
ratio of variances
         0.7336957
```

ここでは p 値が 0.600 と「分散が異なる」という帰無仮説を棄却する大きさでしたので、等分散性が仮定されたと考えます。

それでは実際に t 検定をしましょう (ソースコード 5.3)。

ソースコード 5.3　等分散性の仮定

```
> t.test(experiment$E,experiment$C)

  Welch Two Sample t-test

data:  experiment$E and experiment$C
t = 1.6457, df = 23.447, p-value = 0.1132
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.2359867  2.0821405
sample estimates:
mean of x mean of y
 7.692308  6.769231
```

等分散性は仮定されていますが，ここでは特にオプションの指定はしませんでした。デフォルトでウェルチの t 検定が行われており，結果として $t(23.447) = 1.6457, n.s.$ であったことがわかります。

その下に，95% 信用区間が示されています。これは二群の平均の差が母集団でどの程度あると見積もられたか，の信頼区間です。結果から、母平均の差は-0.23 から 2.08 の間にある確率が 95% あり，この区間に 0.0 を含んでいる＝差がない状態も含まれているため，有意であるとはいえないわけです。

これが一般的な t 検定の流れです。

5.2.2　効果量の算出

近年では，検定をした場合は効果量も付記するような慣習が広まってきました。R で効果量を求めるには pwr パッケージの力を借りることになります。pwr パッケージをインストールし，t 検定で検定力が強いと言われるのはどの程度なのかを cohen.ES 関数で調べてみましょう (コード 5.4)。

ソースコード 5.4　検定力の目安を知る

```
> library(pwr)
> cohen.ES(test="t",size="large")

     Conventional effect size from Cohen (1982)

           test = t
           size = large
    effect.size = 0.8
```

効果量が 0.8 ぐらいあれば，十分な検定力を持った検定ができると言えるようです。それでは今回の効果量 es はどれぐらいだったのか，ですが，これは自由度と t 統計量を使って計算することになります (コード 5.5)。

ソースコード 5.5　検定力の目安を知る

```
> es <- 1.6457 * sqrt((13+13)/(13*13))
> power.t.test(n=13,es)

     Two-sample t test power calculation

              n = 13
          delta = 0.6454966
             sd = 1
      sig.level = 0.05
          power = 0.3519796
    alternative = two.sided

NOTE: n is number in *each* group
```

検定力は 0.35 で，弱い検定力しかなかったことがわかります。つまりこのデータで有意差を出そうとするゲームの勝率は決して高くなく，そもそも無謀なチャレンジだったのかもしれませんね。

5.3　対応のある t 検定の場合

対応のある二群の t 検定の場合は，同じく t.test 関数を使うのですが，オプションとして paired=T と指定する必要があります。また，この検定の場合は，例えば事前・事後の検定の場合，事前に比べて事後の数値が上がった、下がった、という仮説を検定したいことが少なくありません。こうした変化の向きを特定する検定を片側検定と言います。R ではこれもオプションで，alternative=”greater”とすることで前の変数が後ろの変数よりも有意に大きいかどうか，alternative=”less”とすることでその逆向きの検定を行うことになります。

ちなみに，前節の対応のない t 検定の時は両側検定をしていましたが，これは alternetive オプションがデフォルトで”two.sided”になっているので，特に指定せず実行できたものです。

ソースコード 5.6　対応のある t 検定

```
> t.test(anxiety$pre,anxiety$post,paired=T,alternative="greater")

  Paired t-test

data:  anxiety$pre and anxiety$post
t = 4.14, df = 20, p-value = 0.0002535
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 4.306085      Inf
sample estimates:
mean of the differences
               7.380952
```

ここでは $t(20) = 4.14, p < .05$ で有意な結果になり，信頼区間も母平均の差が 4.3〜inf までの間にある，と天井なしの値になっているので、最低でも 4.3 点分の差はあるといっていいでしょう。検定力も十分あったようです (ソースコード 5.6)。

ソースコード 5.7　検定力のチェック

```
> es <- 4.14 *sqrt((21+21)/(21*21))
> pwr.t.test(n=21,es,type="paired")

     Paired t test power calculation

              n = 21
              d = 1.277632
      sig.level = 0.05
          power = 0.9998386
    alternative = two.sided

NOTE: n is number of *pairs*
```

5.4　分散分析を行う関数

三群以上の平均の差の検定，あるいは二要因以上の研究デザインによる平均の差の検定には分散分析を用います。

分散分析は平均の差を検定するものですが，考え方としては回帰分析の独立変数がカテゴリカルなもの，としてとらえることもでき，一般線形モデリングの枠組みの中に含まれるものです。R では aov 関数を使うと分散分析が実行されますが、実際これは線形回帰モデルの lm 関数のラッパーでしかありません。

結果の出力についても，aov 関数は分散分析としての出力という意味では少々シンプルすぎるというか，不十分に感じるところも少なくありません。そこで，4.5.2 の節でも触れたように，分散分析の文脈で必要な情報を多く含んだ anovakun 関数を使う例を紹介していきます。

aov 関数を使った分散分析

まずは aov 関数を使った例を示します。iris データのアヤメの種類毎に，がく片の長さが異なるかどうかを検証します。

ソースコード 5.8　aov 関数による分散分析

```
> data(iris)
> aov(Sepal.Length~Species,data=iris)
Call:
   aov(formula = Sepal.Length ~ Species, data = iris)

Terms:
                Species Residuals
Sum of Squares  63.21213  38.95620
Deg. of Freedom       2       147

Residual standard error: 0.5147894
Estimated effects may be unbalanced
```

結果は非常にシンプルに表示されますが，平方和と自由度だけで，このままでは分散分析として必要な F 値や p 値を算出してくれません。しかし，結果を一旦オブジェクトに預け，summary 関数で出力してやると、分散分析表を作成してくれます。

ソースコード 5.9　aov 関数による分散分析その 2

```
> result.aov <- aov(Sepal.Length~Species,data=iris)
> summary(result.aov)
            Df Sum Sq Mean Sq F value Pr(>F)
Species      2  63.21  31.606   119.3 <2e-16 ***
Residuals  147  38.96   0.265
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

これで分散分析ができました。aov 関数の中に与える式は従属変数~独立変数の形をしており，複数の独立変数や交互作用を記述することもできます。

しかしこの出力では物足りなさを感じた心理学者も少なくないのではないでしょうか。それでは痒い所に手が届く，anovakun による実行例を示しましょう。

anovakun をつかった分散分析

anovakun の導入については 4.5.2 を見ていただくとして，ここでは実際の使い方を例示してみたいと思います。まずは aov 関数の例と同じく，一要因三水準群間デザイン (As デザイン) の例です。

ソースコード 5.10　anovakun 関数による分散分析その 1

```
> source("anovakun_440.txt")
> subdat <- subset(iris,select=c("Species","Sepal.Length"))
> anovakun(subdat,"As",3,peta=T)

[ As-Type Design ]

This output was generated via anovakun 4.4.0 at R version 3.0.2.
It was executed on Mon Nov 11 17:08:56 2013.


<< DESCRIPTIVE STATISTICS >>

--------------------------
  A   N    Mean    S.D.
--------------------------
 a1  50  5.0060  0.3525
 a2  50  5.9360  0.5162
 a3  50  6.5880  0.6359
--------------------------


<< ANOVA TABLE >>

----------------------------------------------------------------
Source        SS  df       MS   F-ratio p-value      p.eta^2
----------------------------------------------------------------
     A   63.2121   2  31.6061  119.2645  0.0000 ***   0.6187
 Error   38.9562 147   0.2650
----------------------------------------------------------------
 Total  102.1683 149
                  +p < .10, *p < .05, **p < .01, ***p < .001


<< POST ANALYSES >>

< MULTIPLE COMPARISON for "A" >

== Shaffer's Modified Sequentially Rejective Bonferroni Procedure ==
== The factor < A > is analysed as independent means. ==
== Alpha level is 0.05. ==

--------------------------
  A   N    Mean    S.D.
--------------------------
 a1  50  5.0060  0.3525
 a2  50  5.9360  0.5162
 a3  50  6.5880  0.6359
--------------------------

----------------------------------------------------------------------
   Pair  Interval   t-value         df        p    adj.p
----------------------------------------------------------------------
  a1-a3   -1.5820   15.3655        147   0.0000   0.0000   a1 < a3 *
  a1-a2   -0.9300    9.0328        147   0.0000   0.0000   a1 < a2 *
  a2-a3   -0.6520    6.3327        147   0.0000   0.0000   a2 < a3 *
----------------------------------------------------------------------


output is over --------------------///
```

ソースコード 5.10 の例に示されているように，データとデザイン，水準数を anovakun 関数に渡すと自動的に下位検定まで行ってくれます。anovakun の下位検定は，デフォルトでは修正ボンフェローニの検定になりますが，他の方法を

指定することも可能です。

また，peta=T のオプションをつけることで，分散分析の効果量である偏 η^2 を算出してくれます。今回は 0.618 となり，アヤメの種別ががく片の分散の 61% を説明したことがわかりました。

二要因以上の分散分析でも，anovakun は対応してくれます。しかも，群内要因に対してモークリーの球面性検定をし，必要とあればオプションで補正をすることもできます。もちろん効果量も算出してくれますので，論文に記載すべき豊富な情報量を得ることができます。

ソースコード 5.11　anovakun 関数による分散分析その 2

```
# AsBデザイン
# ２水準の被験者間要因１つ，３水準の被験者内要因１つの混合要因計画の実行例
> dat <- read.csv("anovaSamp.txt",head=FALSE)
> anovakun(dat,"AsB",2,3,peta=T)

[ AsB-Type Design ]

This output was generated via anovakun 4.4.0 at R version 3.0.2.
It was executed on Mon Nov 11 21:06:36 2013.


<< DESCRIPTIVE STATISTICS >>

------------------------------
  A   B   N    Mean    S.D.
------------------------------
 a1  b1   4  6.2500  1.7078
 a1  b2   4  6.0000  2.1602
 a1  b3   4  4.5000  1.2910
 a2  b1   6  3.0000  1.5492
 a2  b2   6  4.1667  2.1370
 a2  b3   6  8.0000  0.8944
------------------------------


<< SPHERICITY INDICES >>

== Mendoza's Multisample Sphericity Test and Epsilons ==

-----------------------------------------------------------------
 Effect  Lambda  approx.Chi df       p         LB     GG     HF
-----------------------------------------------------------------
      B  0.2238      2.3148  5  0.8041 ns  0.5000 0.9621 1.2606
-----------------------------------------------------------------
    LB = lower.bound, GG = Greenhouse-Geisser, HF = Huynh-Feldt


<< ANOVA TABLE >>

== This data is UNBALANCED!! ==
== Type III SS is applied. ==

-----------------------------------------------------------------
    Source        SS  df       MS  F-ratio p-value      p.eta^2
-----------------------------------------------------------------
         A    2.0056   1   2.0056   0.6048  0.4591 ns    0.0703
     s x A   26.5278   8   3.3160
-----------------------------------------------------------------
         B   13.4778   2   6.7389   2.6918  0.0982 +     0.2518
     A x B   60.8111   2  30.4056  12.1454  0.0006 ***   0.6029
 s x A x B   40.0556  16   2.5035
-----------------------------------------------------------------
     Total  157.8667  29
                     +p < .10, *p < .05, **p < .01, ***p < .001


<< POST ANALYSES >>

< MULTIPLE COMPARISON for "B" >

== Shaffer's Modified Sequentially Rejective Bonferroni Procedure ==
== The factor < B > is analysed as dependent means. ==
== Alpha level is 0.05. ==

--------------------------
  B   N    Mean    S.D.
--------------------------
 b1  10  4.6250  2.2632
```

```
 b2  10  5.0833  2.2336
 b3  10  6.2500  2.0656
--------------------------

-------------------------------------------------------------------
   Pair  Interval  t-value       df        p    adj.p
-------------------------------------------------------------------
  b1-b3   -1.6250   2.4158        8   0.0421   0.1264   b1 = b3
  b2-b3   -1.1667   1.4783        8   0.1776   0.1776   b2 = b3
  b1-b2   -0.4583   0.6552        8   0.5307   0.5307   b1 = b2
-------------------------------------------------------------------


< SIMPLE EFFECTS for "A x B" INTERACTION >

-----------------------------------------------------------------
  Effect  Lambda  approx.Chi df       p         LB     GG     HF
-----------------------------------------------------------------
 B at a1  0.8702      0.1854  2  0.9114 ns  0.5000 0.9186 2.3003
 B at a2  0.9964      0.0058  2  0.9971 ns  0.5000 0.9985 1.6621
-----------------------------------------------------------------
     LB = lower.bound, GG = Greenhouse-Geisser, HF = Huynh-Feldt


-----------------------------------------------------------------------
      Source       SS  df       MS  F-ratio  p-value       p.eta^2
-----------------------------------------------------------------------
     A at b1  25.3500   1  25.3500   9.7735   0.0141 *      0.5499
    Er at b1  20.7500   8   2.5938
-----------------------------------------------------------------------
     A at b2   8.0667   1   8.0667   1.7520   0.2222 ns     0.1797
    Er at b2  36.8333   8   4.6042
-----------------------------------------------------------------------
     A at b3  29.4000   1  29.4000  26.1333   0.0009 ***    0.7656
    Er at b3   9.0000   8   1.1250
-----------------------------------------------------------------------
     B at a1   7.1667   2   3.5833   0.8431   0.4757 ns     0.2194
 s x B at a1  25.5000   6   4.2500
-----------------------------------------------------------------------
     B at a2  82.1111   2  41.0556  28.2061   0.0001 ***    0.8494
 s x B at a2  14.5556  10   1.4556
-----------------------------------------------------------------------
                       +p < .10, *p < .05, **p < .01, ***p < .001


< MULTIPLE COMPARISON for "B at a2" >

== Shaffer's Modified Sequentially Rejective Bonferroni Procedure ==
== The factor < B at a2 > is analysed as dependent means. ==
== Alpha level is 0.05. ==

-------------------------------------------------------------------
   Pair  Interval  t-value       df        p    adj.p
-------------------------------------------------------------------
  b1-b3   -5.0000   7.3193        5   0.0007   0.0022   b1 < b3 *
  b2-b3   -3.8333   5.4515        5   0.0028   0.0028   b2 < b3 *
  b1-b2   -1.1667   1.6592        5   0.1580   0.1580   b1 = b2
-------------------------------------------------------------------


output is over --------------------///
```

第 6 章

回帰分析について

6.1 回帰分析の基礎

まずは回帰分析について，基本的なことをおさらいしておきましょう。

■概略 回帰分析とは，ある観測変数を他の観測変数を用いて予測・説明するためのものです。説明する変数を予測変数または独立変数といい，説明される変数を目的変数，被説明変数または従属変数といいます。

ひとつの独立変数しか用いない回帰分析を単回帰分析，複数の独立変数を用いる回帰分析を重回帰分析といいます。回帰 regression という言葉は，一回目の測定で平均から逸脱した値を取ったとしても，次の測定では平均値に近い値が得られる「平均への回帰」現象から名付けられています。

■回帰式の立て方 目的変数 Y に対して，予測変数 X で回帰をするというのは，$Y = f(X)$ の関数関係をデータから見いだすことであり，もっとも単純な関数として $Y = aX + b$ の一次式が使われます。

$Y = aX + b$ の一次式による単回帰分析において，傾き a は回帰係数，パス係数と呼ばれます。切片 b は独立変数と従属変数の平均値の差を調整するための数字になっています[*1]

回帰係数の推定は，予測式で算出される予測値 $\hat{Y}_i$ と実際の目的変数 Y_i との差 $\hat{Y}_i - Y_i = e_i$ の総和 $\sum e_i$ が最小になるように a, b の値を求める最小二乗法による方法が用いられます。

また，回帰分析モデルは基本的に連続変数と連続変数の関係をつなぐ式になっていますが，従属変数が二値変数であるとか，正の値しか取らない歪んだ分布をする場合には，線形回帰モデルの条件が成立するように数値を変換するリンク関数をかませる，という手法も開発されています (一般化線形モデル,GLM)。

6.1.1 回帰分析とプロット

この分析法を実際に R を使って実行するには，特に追加のパッケージは必要なく，R の基本的な関数である lm 関数 (Liner Model の略) でできます。

ソースコード 6.1 lm 関数による回帰分析

```
> hw <- read.csv(file.choose()) # hw.csv を[開く]
> plot(weight ~ height,data=hw)
> reg1 <- lm(weight~height,data=hw)
> summary(reg1)

Call:
lm(formula = weight ~ height, data = hw)

Residuals:
    Min      1Q  Median      3Q     Max
-6.8044 -4.4821  0.3781  2.4732  8.7127
```

[*1] 用いる関数は $Y = aX + b$ の線形一次式に限らず，非線形な回帰式が使ってもかまいません (非線形回帰分析)。

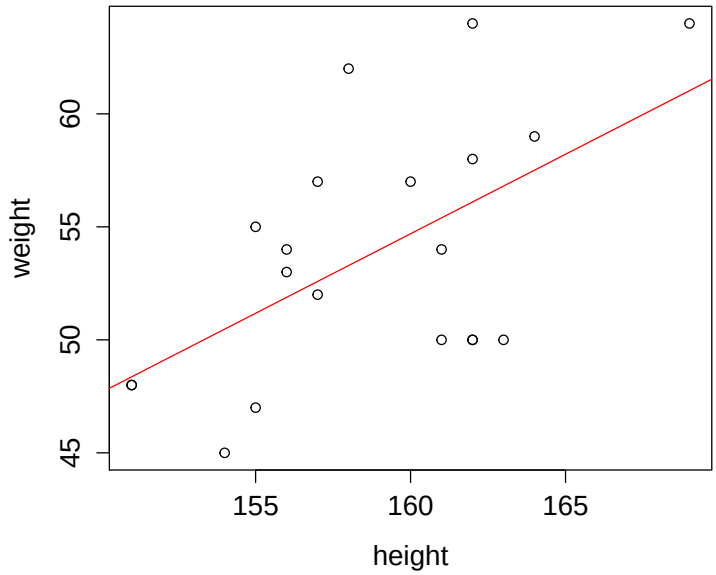


図 6.1　回帰分析のプロット (回帰直線)

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -57.8537    37.4826  -1.543    0.140
height        0.7034     0.2359   2.981    0.008 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.702 on 18 degrees of freedom
Multiple R-squared:  0.3306,  Adjusted R-squared:  0.2934
F-statistic: 8.888 on 1 and 18 DF,  p-value: 0.008004
```

モデルの書き方は，従属変数と独立変数をチルダ ( ) でつなぐというもので，このモデル式の書き方は lm 関数に限らず R のなかで一般的に使われているものです。

分析の結果，$weight = 0.7034 \times hight - 57.8537$ という回帰式が得られました。この回帰式において，標準誤差が 0.2359，検定の結果このパス係数が 0.00 であるという帰無仮説が 5% 水準で棄却されたことがわかります。

重相関係数 R^2 は 0.3306, 自由度調整済み R^2 は 0.2934 で，この回帰式による独立変数が説明変数を説明する割合が 33% であることが示されています。

回帰分析は式で見るよりも図で見た方が分かりやすいかもしれません。plot 関数で二変数の散布図は描かれていますので，これに abline 関数でえられた予測式を追加してみたものが，図 6.1 です。図を見ると，直線から外れた値がいくつもあることが分かります。それどころか，直線の上に乗っている (ぴったりと予測関数が当てはまっている) データはなく，大目に見てもあまり当てはまりの良いモデルであったとはいえないでしょう。R^2 の値がそれを雄弁に語っています。

実際に，予測関数の算出する予測値と，データのとズレ (残差) を確認してみましょう。

ソースコード 6.2　予測値と残差

```
> predict(reg1) # 予測値 ¥hat{y} を算出
       1        2        3        4        5        6        7        8        9       10       11
54.69411 51.17699 56.10096 52.58384 56.10096 56.80438 61.02492 48.36329 56.10096 48.36329 50.47356
      12       13       14       15       16       17       18       19       20
55.39753 55.39753 57.50780 52.58384 53.28726 51.88041 51.17699 56.10096 51.88041
```

```
> reg1$residuals # 残差を出力
         1          2          3          4          5          6          7          8          9
 2.3058912  3.8230111  1.8990433 -0.5838369  7.8990433 -6.8043807  2.9750755 -0.3632931 -6.1009567
        10         11         12         13         14         15         16         17         18
-0.3632931 -5.4735650 -5.3975327 -1.3975327  1.4921954  4.4161631  8.7127392  1.1195871 -4.1769889
        19         20
-6.1009567  2.1195871
```

これを見ると，最初のケースでは予測値は 54.69kg と実際の値よりも 2.31kg 低く予測されています (実際のデータは 57kg)。

6.2　重回帰分析の基礎

重回帰分析は複数の独立変数をもつ回帰分析です。最初に用語や手法の説明をしておきます。

■偏回帰係数　重回帰分析において，例えば従属変数が Y で独立変数に A,B があった場合，A で Y を予測した残差を B が予測する，ということになります。その結果，A で Y を予測した単回帰分析よりも，A, B で予測した重回帰分析の方が必ず R^2 は大きくなります。独立変数を増やせば増やすほど，説明される量は増えていくのです。また，A で説明した残りを B が説明しますので，B の回帰係数は A と Y の関係を統制した (Y に対する A の影響力を取り除いた) 上で算出した値になります。こうしたモデル上の統制をパーシャル・アウト (Partial out) と言いますが，パーシャル・アウトされた回帰係数のことを特に偏回帰係数 (partial regression coefficient) と呼びます。単回帰分析のときのパス係数とは少し名称が異なることに注意してください。

■標準偏回帰係数　重回帰分析は複数の独立変数を持っていますから，どの独立変数が予測に貢献したか，といった独立変数間の比較をすることも必要です。そのためには単純にパス係数だけを比較していてはいけません。パス係数は単位に依存する値ですので，例えば身長 (三桁の数字) と年収 (七・八桁の数字) で体重 (二桁の数字) を予測すると，年収のパス係数の方が小さくなってしまいます。そこで，単位に影響されないように，独立変数と従属変数を標準化した，標準化回帰係数が用いられることが多いです。(これに対し，素データの回帰係数のことを非標準化回帰係数といいます)。

上で述べたように，重回帰分析の場合は偏回帰係数ですから，標準化偏回帰係数 (standardised partial regession coefficient) という名称になります。

■独立変数同士が独立であること　重回帰分析を行うにあたって，独立変数同士が独立であること，言い換えると独立変数同士が無相関であること，が条件になります。例えば A, B 二つの独立変数を使って Y を説明する場合，A から Y への影響と B から Y への影響を見るわけですが，A と B が相関関係にあると $A \to Y$ の影響力の中に $B \to A \to Y$ の影響力が含まれてしまい，純粋に $A \to Y$ の影響力だけ取り出せていないことになります。この問題のことを，多重共線性 (multicollinearity, 通称マルチコ) の問題といいます。

具体的な症状としては，回帰係数が求められない，求められた回帰係数が理論的予測と符号が異なる，変数を増やしたり減らしたりすると回帰係数が大きく変わって安定しない，などのものがあり，こうしたときに多重共線性の疑いがもたれます。

多重共線性の問題が生じていないかどうかを検証するためには，VIF(Variance Inflation Factor，分散拡大要因) という数値を算出し，この値が 10 を超えていたら問題が生じている，と判断します。

■導入する順番がある場合，ない場合　重回帰分析は複数の独立変数を持っていますが，それらの変数を一度に投入する方法だけでなく，順番をつけて投入する方法があります。順番をつけて投入する方法のうち，モデルの適合度を評価基準にしながら変数を徐々に増やしていく，あるいは徐々に減らしていくようなものを「変数選択法」といいます。ステップワイズ法と呼ばれる変数選択法を使うと，最終的に適合度のよいモデルをひとつ以上提案してくれるので，説明変数が多いときにだいたいのアタリをつけるのにいいでしょう。

また，まず複数の変数を投入してその効果を確認し，次にその説明変数同士の交互作用項を投入して効果を検証する方法を，階層的重回帰分析と呼びます。ここでの階層的，とはこのように手続きが階層的になっているという意味で，データの階層性 (ネストされたデータ) を扱うモデルではないことに注意してください[*2]。

さらに，順に変数を投入するモデルのうち，$A \to Y$ の効果を確認した上で，$A \to B$ と $B \to Y$ の効果を確認する方法を，媒介分析と呼びます。変数 B が媒介となって，$A \to B \to Y$ という関係を検証しているからです。

このように，重回帰分析は様々な広がりをもっています。

6.2.1　重回帰分析とプロット

重回帰分析を実際に R を使って実行するには，単回帰分析と同じ lm 関数で，独立変数同士を + でつなぐことでできます。中古車価格の変動を走行距離と車両の状態で回帰するモデルを実行してみましょう。

ソースコード 6.3　lm 関数による重回帰分析

```
> reg2 <- lm(kakaku~kyori+jotai, data=used_car)
> summary(reg2)

Call:
lm(formula = kakaku ~ kyori + jotai, data = used_car)

Residuals:
   Min     1Q Median     3Q    Max
-9.805 -4.107 -1.268  2.388 13.842

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   70.588     21.583   3.271  0.00967 **
kyori         -6.353      1.720  -3.694  0.00496 **
jotai          5.628      2.000   2.814  0.02024 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.786 on 9 degrees of freedom
Multiple R-squared:  0.9566,  Adjusted R-squared:  0.947
F-statistic: 99.31 on 2 and 9 DF,  p-value: 7.353e-07
```

分析の結果，$kakaku = -0.6353 \times kyori - 5.628 \times jotai + 70.588$ という回帰式が得られました。独立変数はいずれも統計的に有意であり，重相関係数 R^2 は 0.9566 と，いい当てはまりのモデルであったことが伺えます。

回帰分析は式で見るよりも図で見た方が分かりやすいですが (図 6.2)，三変数以上の重回帰分析は三次元プロットでも描画できません。

6.2.2　標準偏回帰係数を求めるには

複数の独立変数間の比較をするために，標準偏回帰係数を算出しましょう。独立変数だけ標準化する場合，独立変数も従属変数も標準化する場合がありますが，ここでは独立変数も従属変数も標準化する場合をやってみます。

標準化には，R のもつ scale 関数を使います。scale 関数で元データを標準化された新しいデータセットにし，それを使って重回帰分析をするだけです。ただし，scale 関数はリスト型で結果を返しますので，あらためてデータフレーム型に変換する必要があります。

ソースコード 6.4　標準偏回帰係数を算出する

```
> z <- scale(used_car)
> z <- data.frame(z)  # データフレーム形式に戻す
> summary(lm(kakaku~kyori+jotai, z))

Call:
lm(formula = kakaku ~ kyori + jotai, data = z)
```

[*2] ネストされたデータを，下位カテゴリの分散，上位カテゴリの分散に分けてモデリングすることは階層線形モデリングといいます。

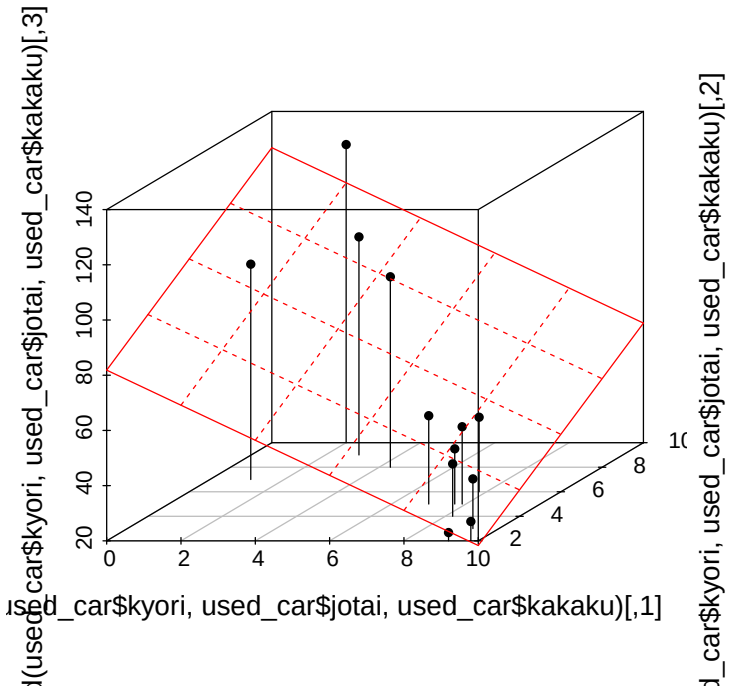


図 6.2　重回帰分析のプロット (予測平面)

```
Residuals:
     Min       1Q   Median       3Q      Max
-0.28989 -0.12143 -0.03748  0.07059  0.40925

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.154e-16  6.645e-02   0.000  1.00000
kyori       -5.703e-01  1.544e-01  -3.694  0.00496 **
jotai        4.344e-01  1.544e-01   2.814  0.02024 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2302 on 9 degrees of freedom
Multiple R-squared:  0.9566,  Adjusted R-squared:  0.947
F-statistic: 99.31 on 2 and 9 DF,  p-value: 7.353e-07
```

標準化してもモデルの当てはまり (R^2) に変化はありません。パス係数に記号が含まれていますが，この e を使った表記はコンピューター一般に用いられる表記法で，e-01 は 10^{-1} を意味しています。ここでは，距離変数のパス係数は $-5.703 \times 10^{-1} = -5.703 \times 0.1 = 0.5703$ であったことを意味します。

6.2.3　変数選択法をつかう

それでは次に，独立変数の投入に順番をつける方法，なかでもステップワイズ法による変数選択法を試してみましょう。同じ中古車価格のデータで，独立変数に年数，距離，状態の三変数を持っている重回帰分析をいったん行います。

ソースコード 6.5　ステップワイズ法その 1

```
> reg3 <- lm(kakaku~nensu+kyori+jotai, data=used_car)
> summary(reg3)

Call:
lm(formula = kakaku ~ nensu + kyori + jotai, data = used_car)

Residuals:
   Min     1Q Median     3Q    Max
-8.555 -4.120 -1.068  1.941 13.948
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   70.932     22.673   3.128   0.0140 *
nensu          1.593      3.899   0.408   0.6936
kyori         -7.984      4.381  -1.822   0.1059
jotai          5.590      2.101   2.660   0.0288 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 8.173 on 8 degrees of freedom
Multiple R-squared:  0.9575,  Adjusted R-squared:  0.9416
F-statistic: 60.13 on 3 and 8 DF,  p-value: 7.865e-06
```

次に，この分析結果が格納されているオブジェクト reg3 を step 関数に渡します。step 関数は統計的指標 (AIC) を基準に変数選択をする関数です。

ソースコード 6.6　ステップワイズ法その 2

```
> step(reg3)
Start:  AIC=53.55
kakaku ~ nensu + kyori + jotai

        Df Sum of Sq     RSS    AIC
- nensu  1     11.15  545.55 51.803
<none>                534.40 53.555
- kyori  1    221.85  756.25 55.722
- jotai  1    472.79 1007.19 59.160

Step:  AIC=51.8
kakaku ~ kyori + jotai

        Df Sum of Sq     RSS    AIC
<none>                545.55 51.803
- jotai  1    480.05 1025.59 57.377
- kyori  1    827.27 1372.81 60.877

Call:
lm(formula = kakaku ~ kyori + jotai, data = used_car)

Coefficients:
(Intercept)        kyori        jotai
     70.588       -6.353        5.628
```

まず最初の段階で 3 つの独立変数が全て入れられたモデルが検証され，次に nensu 変数が除外されたモデルになりました。AIC が 53.55 から 51.8 に改善され (AIC は小さいほどモデルの当てはまりが良いとされます)，これでステップが終了しています。最終的に提案されたのは，距離と状態で価格を予測するモデルだったことがわかります。

6.2.4　多重共線性のチェック

ところで，中古車価格の決まり方に三つの独立変数を仮定しましたが，独立変数同士の相関関係に問題はなかったでしょうか。多重共線性の問題を検証するには，car パッケージの vif 関数を使います。

ソースコード 6.7　VIF の検証

```
> library(car)
> vif(reg3)
    nensu     kyori     jotai
24.346714 29.137302  4.957389
```

VIF の基準は 4 以上，5 以上，10 以上なら問題である，などといわれ，目安となる数値の共通見解はありませんが，少なくとも 2 より小さければ問題無し，とされています。また，今回のデータは大きめの 10 を基準にしても，年数と距離の VIF が 10 よりもかなり大きな値になっているので，多重共線性の問題が生じていたことは明らかです。中古車の場合，使用年数と走行距離に高い相関があるのは十分考えられることですから，独立変数として同時に投入するのはふさわしくないですね。

6.3　回帰分析の広がり

回帰分析の基本は単回帰，重回帰分析ですが，様々な応用的モデルがあります。ここでは分位点回帰と一般化線形モデル GLM について説明します。

6.3.1　分位点回帰 Quantile Regression

回帰分析は独立変数や従属変数が正規分布のような左右対称の分布をしている時，その真ん中，平均点あたりの変数間関係を予測するのに適しています。心理学で扱う態度などの心理的変数は，生態学的な理論的背景から正規分布を仮定しているため，回帰分析を使うことに問題はありませんが，実態的なデータの場合は正規分布していないものが多くみられます。

例えば，昨今の日本社会はジニ係数が大きくなった格差社会ですから，年収等のデータを取ってみると所謂お金持ちは多く存在せず，あまり裕福でない層の人数が多いわけです。年収の分布を見てみると，左側に大きく歪んでおり，左右対称の正規分布データとはとても言えません。

このようなデータを用いて回帰分析をし，平均的サンプルの特徴を描いたとしても，その平均像が既に珍しいケースになってしまっているので，結果の解釈が実態や実感と合わないことも少なくないでしょう。

平均値が歪みに弱い代表値であることは統計学の初歩で習いますが，それでは歪みに強い代表値と言えば何でしょうか？　例えばそれは，中央値 median であったり，四分位偏差 Quantile であったりするわけです。分位点は，分布を密度にそって分割しますから，例えば四分位は全体を 25% ずつに区切って，上位 25%, 上位 50%(=中央値), 下位 25% の区切りとなる値を示すものです。この分位点を用いた回帰が分位点回帰であり，任意の分位点では回帰係数がどうなっているか，分位点ごとの傾きの違いを検証する分析方法です。これを使うと，例えば収入が少ない世帯では影響の強い変数も収入が多い世帯では影響が少ない，といったことが分かるようになります。

R では quantreg パッケージを導入すると実行することができます。

quantreg パッケージに含まれるエンゲル係数のデータをつかって分位点回帰を行った例を示します。分位点回帰の関数は rq 関数です[*3]。年収から食費を予測する式を立てて，分位点ごとに傾きを検証します。分位点の区切りはいくつでもいいので，今回は seq 関数を使って 0 から 1 までを 0.1 刻みにしてみました。

ソースコード 6.8　分位点回帰

```
> library(quantreg)
> data(engel)
> hist(engel$income)
> rq(foodexp~income,data=engel,tau=seq(0,1,0.1))
Call:
rq(formula = foodexp ~ income, tau = seq(0, 1, 0.1), data = engel)

Coefficients:
               tau= 0.0    tau= 0.1    tau= 0.2 tau= 0.3    tau= 0.4   tau= 0.5   tau= 0.6  tau= 0.7
(Intercept) 113.1406322 110.1415742 102.3138823 99.11058 101.9598824 81.4822474 79.7022726 79.283617
income        0.2942315   0.4017658   0.4468995  0.48124   0.5098965  0.5601806  0.5858492  0.608851
             tau= 0.8   tau= 0.9    tau= 1.0
(Intercept) 58.0066635 67.3508721 225.3824790
income       0.6595106  0.6862995   0.6403102

Degrees of freedom: 235 total; 233 residual
```

結果から，年収の分位点が 30% の時は回帰係数が 0.48 であるのに，60% であれば 0.58,90% であれば 0.68 と大きく変わっていくことが見て取れます。これを図にしたのが図 6.3 です[*4]。このように，平均だけではなく様々な広がりと

[*3] Quantile Regression なら qr 関数じゃないのか，と思いますが，R には既に固有値分解のひとつ，QR 法の関数が qr の名称で実装されているために違う名称になっているようです。

[*4] quantreg パッケージの rq 関数のヘルプより。

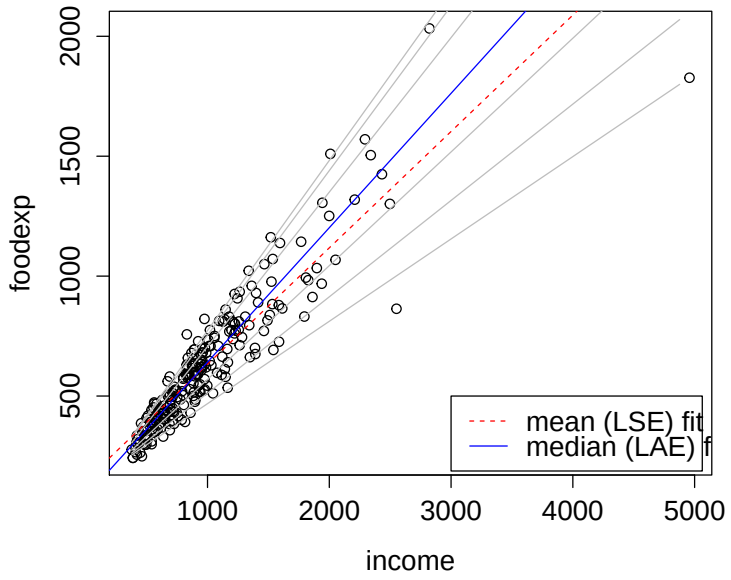


図 6.3　分位点ごとの回帰係数の変化

ともに見ることで，より面白い知見が得られるかもしれません。

6.3.2　一般化線形モデル Generarized Liner Model

分布の形についてさらに拡張したのが一般化線形モデルです。回帰分析は，残差が正規分布に従うという仮定に基づいていますが，既に述べたようにデータが常に正規分布にしたがうという保証はないわけです。昔の心理学統計法では，例えば左に歪んだ分布をしていれば，元データを対数変換したものを分析データにする，ということが行われていました。こうしたデータの変換をモデル式のレベルで表現し，かつ一般化したのが GLM と呼ばれる手法です。

通常の線形回帰は $y = X\beta + e$ で表現されますが[*5]，ここに $g(\mu) = X\beta$ のような変換の拡張を行います。ここで μ は被説明変数の平均 (期待値) で，これを線形モデルにあうように $g(\cdot)$ という何らかの関数，一般にリンク関数と呼ばれますが，で帳尻を合わせ，右辺の線形結合を予測する式にするのです。

変換は被説明変数の分布の形状から適したものを考えます。例えば正の値しか取らない，観測度数のようなデータであれば，ポアソン分布に従うと考えられますので，$\log(\mu)$ がリンク関数になります。またあるいは，二値データ (Yes/No,True/False,Dead/Alive のような) であれば二項分布ですので，$\log(\mu/(1-\mu))$ をリンク関数にしたロジスティック回帰分析になります。

R でこの一般化線形モデルを適用するには特別な準備は必要なく，回帰分析を実行する lm 関数の代わりに glm 関数を用い，オプションとして引数 family(族と呼ばれます) にふさわしい分布族の名称を渡してやることで対応します。

R のもつサンプルデータ，airquality を使って，lm 関数と glm 関数の比較をしてみましょう。このデータは 1973 年 5 月から 9 月までのニューヨークの大気状態を，オゾンの量，日射量，風力など 6 つの変数で観測した 154 の観測値です。オゾンの量を日射量や風力で回帰してみようと思うのですが，日射量は正規分布しているというより右に歪んでおり (図 6.4)，被説明変数の分布を $(0, +\mathrm{inf})$ にとるガンマ分布を分布族とした GLM を行ってみました。

[*5] ここで X や β はデータ行列，回帰係数行列の表現になっています。

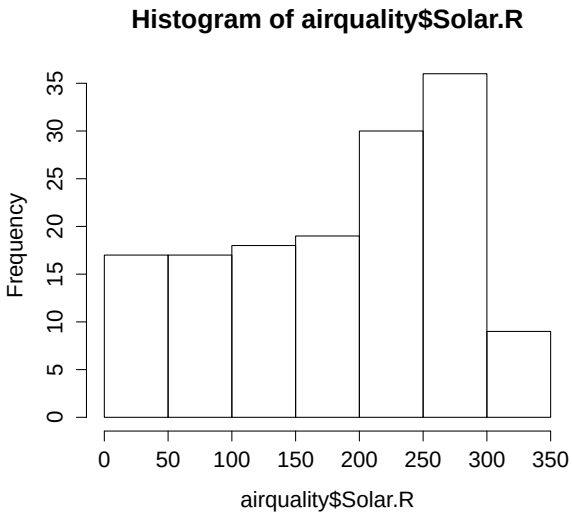


図 6.4　日射量の分布は正規分布ではない

ソースコード 6.9　線形回帰と一般化線形回帰の比較

```
> data(airquality)
> result.lm <- lm(Ozone~Solar.R+Wind,data=airquality)
> result.glm <- glm(Ozone~Solar.R+Wind,data=airquality,family=Gamma)
> AIC(result.lm)
[1] 1033.816
> AIC(result.glm)
[1] 971.8726
```

分析結果を細かく解釈することはしませんが，モデルの当てはまりを見る情報量基準 (AIC) を線形回帰と GLM とで算出してみると，GLM のモデルのほうが改善されていることが見て取れます。

この他にも，階層線形モデリングや一般化混合線形モデリングなど回帰分析の幅はまだまだ広がっており，あらためてその力強い分析力に驚かされます。

第 7 章

因子分析について

7.1 因子分析の概要

因子分析をいくつかの角度から簡単に説明すると，次のようになります。

1. 因子分析とは，回帰分析でいうところの説明変数が潜在変数であることを仮定した統計モデルです。
2. 複数の測定された項目の背後に複数の潜在変数を仮定し，潜在変数の数や影響度などの構造を検証するものです。
3. 試験などにおける個々人の回答から，潜在的な能力の値をより正確に測定するため誤差要因を除外し，因子得点として表現するものです。
4. 項目の分散を共通性と独自性の比率で表現し，主たる共通成分を解釈する手法です。
5. 相関行列を因子負荷量の積に分解し，項目と関係の深い因子を抽出する方法です。
6. 標準化された得点を因子負荷量と因子得点の積和の形で表現するもの，すなわち $z_{ij} = a_{j1}f_{i1} + a_{j2}f_{i2} + \cdot + a_{jm}f_{im} + d_j u_{ij}$ です。

7.1.1 結局やっていることと言えば

因子分析の説明の仕方は色々考えられますが，やっていることは次の通りです。

1. データから相関行列を作成します。
2. 相関行列を固有値・固有ベクトルに分解することで共通因子として抽出できる因子の数を決定します。
3. 想定された共通因子に基づいて因子負荷量を算出します。
4. 因子負荷量の解釈をしやすくするために項目との関係を強調します（因子軸の回転）。
5. 因子負荷量に対応する個々人の因子得点を推定します。

想定された因子構造がデータに当てはまっているかどうか，適合度の観点から評価することもできます。

7.2 psych パッケージを使った一連の流れ

このように書くと非常にやることが多いようですが，実際に行う分析はたった 3 行のコードでできます。関数としては R の factanal 関数を使うこともできるのですが，より豊富なオプションのある psych パッケージを使う方が便利です。psych パッケージを使った因子分析の流れは，

1. fa.parallel 関数で因子数を決定する。

2. fa 関数で因子分析を実行する。
3. print 関数で結果を表示する。

となります。

7.2.1 psych パッケージの導入とサンプルデータ

それでは psych パッケージを導入し，サンプルデータを使って因子分析を実行してみましょう。パッケージの導入は，install.package 関数を使ってインストールし，library 関数をつかって実装します。Rstudio をお使いの場合は，Package タグでインストールしたり実装したりすることもできます。

ソースコード 7.1　psych パッケージの導入

```
install.packages("psych")
library(psych)
```

psych パッケージに含まれている bfi データをサンプルとして使いましょう。

ソースコード 7.2　bfi データの読み込み

```
data(bfi) # サンプルデータ bfi の読み込み
help(bfi)
head(bfi)
```

bfi データは 2800 人分のデータで，25 項目からなり，5 因子構造（いわゆるビッグファイブ）であることが分かっています。help 関数や head 関数で，データの詳細を確認し，また最初の数行を確認してみましょう。

7.2.2 因子数の決定

因子数を決定するには，データから相関行列を作成し固有値分解する必要がありますが，R 上では面倒な手間はありません。VSS.scree 関数で最大固有値から順に折れ線グラフで表現するスクリープロットを描きます。また，上で述べた fa.parallel 関数を使うことで，平行分析によって適切な因子数を提案してくれます。スクリープロットは折れ線グラフの形状，角度などから主観的に因子数を決定するものですが，平行分析では元データと同じサイズのデータ行列をランダムに並べ替えた仮想行列を作り，その固有値と比較して因子数を決定します。データ行列のサイズが同じであるとはいえ，ランダムに並べ替えたものからは意味のある因子を抽出できるとは考えられませんから，仮想データのスクリープロットよりも大きな固有値を持つ因子は意味がある因子だと判断するのです[*1]。

ソースコード 7.3　因子数の決定

```
> VSS.scree(bfi[1:25]) # スクリーテスト
> fa.parallel(bfi[1:25]) # より高機能。平行分析により因子数を提案してくれる
Parallel analysis suggests that the number of factors =  6  and the number of components =  6
```

今回のデータでは，因子分析としては 6 因子構造が良いと提案されました。

7.2.3 共通性の推定と回転オプション

因子数が決まると，それに基づいた因子分析を行います。ここでは fa 関数を使った因子分析を行い，その結果を fa.result オブジェクトに代入します。

ソースコード 7.4　因子分析の実際

```
fa.result <- fa(bfi[1:25],nfactors=6,fm="ml",rotate="promax",scores=TRUE)
```

[*1] カテゴリカルなデータを因子分析する場合は，fa.parallel.poly 関数を使います。

fa 関数は，データ，因子数，因子推定法，回転法，因子スコアを保存するかどうかのオプション，を指定できます。

因子推定法は fm オプションで，デフォルトでは minres 法（最小二乗法）になっていますが，データ数が十分にある場合は最尤法 (ml) を使うのが良いでしょう。

因子分析で因子負荷行列を得たら，因子がどの項目と関係が深いかを確認する必要があります。因子と観測変数との相関係数ともいえる因子負荷行列ですが，そのままではどことどこの関係が強いのか，分かりにくいことがあります。そこである基準に沿ってその特徴を強調する方法，いわゆる因子軸の回転が行われることが一般的です。fa 関数のオプションでは，”varimax”, ”quartimax”, ”bentlerT”, ”geominT”, ”bifactor”,”promax”, ”oblimin”, ”simplimax”, ”bentlerQ, ”geominQ”, ”biquartimin”, ”cluster”が選択できます。詳細はヘルプや専門書を参照してください。ここでは promax 回転を指定しています。

因子軸の回転はこの他にも色々考案されています。GPArotation パッケージを使うと，目標行列を設定した回転 (targetT,targetQ,pstT,pstQ) や情報量基準に基づいた回転 (entropy,infomaxT,infomaxQ,mccammon)，Tandem 基準に基づく回転 (tandemI,tandemII)，Crawford-Ferguson family の回転 (cfT,cfQ) など，さらに豊富な回転が選択できます。

最後の scores オプションは，分析結果に基づく因子得点を保存するときに TRUE とします。こうすることで，結果を格納した fa.result オブジェクトの中に因子得点が保存されることになります。

fa 関数のオプションとしては他にも，欠損値の代入法として中央値や平均値を代入して分析する impute オプションなどがありますが，欠損値の推定については近年多重代入法などが推奨されています。R では mi パッケージで多重代入法ができますので，興味のある方は試してみてください。

7.2.4　出力のときのオプション

それでは分析結果を出力してみましょう。出力に際しては，因子負荷量の大きな順に並べ替えたり (sort オプションを TRUE に)，表示する桁数を整えたり（digit=3 で小数点下 3 桁まで表示），小さな因子負荷量は表示しないようにしたり（cut=0.3 で因子負荷量が 0.3 より小さなものは表示されません）することができます。

まず因子分析の結果として，因子負荷量，共通性 (h^2)，独自性 ($u^2 = 1 - h^2$)，ホフマンの複雑性指標からなる行列が表示されます。続いて負荷量の二乗和，寄与率，累積寄与率や，因子間相関行列などが示されます。

ソースコード 7.5　因子分析の結果 1

```
> print(fa.result,sort=TRUE,digit=3,cut=.3)
Factor Analysis using method =  ml
Call: fa(r = bfi[1:25], nfactors = 6, rotate = "promax", scores = TRUE,
    fm = "ml")
Standardized loadings (pattern matrix) based upon correlation matrix
   item    ML1    ML2    ML3    ML5    ML4    ML6    h2    u2  com
E2   12 -0.742                                      0.550 0.450 1.57
E4   14  0.716                                      0.558 0.442 1.10
E1   11 -0.655                                      0.385 0.615 1.56
E3   13  0.555                              0.379   0.477 0.523 2.73
E5   15  0.484                                      0.397 0.603 3.26
A5    5  0.370                0.366                 0.477 0.523 1.14
N4   19 -0.356  0.353                               0.477 0.523 1.04
N2   17         0.927                               0.691 0.309 1.30
N1   16         0.917                               0.705 0.295 2.08
N3   18         0.646                               0.521 0.479 1.88
N5   20         0.376                               0.345 0.655 1.44
C2    7                0.750                        0.498 0.502 1.04
C4    9               -0.651                0.546   0.555 0.445 2.11
C3    8                0.596                        0.308 0.692 1.33
C1    6                0.595                        0.346 0.654 2.23
C5   10               -0.551                0.340   0.426 0.574 1.10
A2    2                       0.661                 0.495 0.505 1.08
A1    1                      -0.603                 0.330 0.670 1.08
A3    3                       0.553                 0.511 0.489 2.88
A4    4                       0.339                 0.281 0.719 2.47
O5   25                               0.599         0.370 0.630 2.32
O3   23                              -0.554  0.360  0.484 0.516 1.25
O2   22                               0.498         0.290 0.710 2.20
O1   21                              -0.461  0.321  0.343 0.657 3.17
O4   24                              -0.352         0.251 0.749 1.17
```

```
                        ML1   ML2   ML3   ML5   ML4   ML6
SS loadings           2.682 2.451 1.966 1.597 1.484 0.891
Proportion Var        0.107 0.098 0.079 0.064 0.059 0.036
Cumulative Var        0.107 0.205 0.284 0.348 0.407 0.443
Proportion Explained  0.242 0.221 0.178 0.144 0.134 0.080
Cumulative Proportion 0.242 0.464 0.641 0.785 0.920 1.000

 With factor correlations of
       ML1    ML2    ML3    ML5    ML4    ML6
ML1  1.000 -0.376  0.341  0.320 -0.217 -0.149
ML2 -0.376  1.000 -0.092 -0.104  0.145  0.469
ML3  0.341 -0.092  1.000  0.312 -0.192  0.249
ML5  0.320 -0.104  0.312  1.000 -0.075  0.222
ML4 -0.217  0.145 -0.192 -0.075  1.000  0.114
ML6 -0.149  0.469  0.249  0.222  0.114  1.000
```

続く出力は, 因子分析の適合度指標です。構造方程式モデリング等で用いられる, χ^2 分析の結果や, RMSR,RMSEA,BIC などの適合度が示されます。

ソースコード 7.6　因子分析の結果 2；適合度

```
Mean item complexity =  1.8
Test of the hypothesis that 6 factors are sufficient.

The degrees of freedom for the null model are  300  and the objective function was  7.228
 with Chi Square of  20163.79
The degrees of freedom for the model are 165  and the objective function was  0.364

The root mean square of the residuals (RMSR) is  0.02
The df corrected root mean square of the residuals is  0.038

The harmonic number of observations is  2762 with the empirical chi square  661.285 with prob <  1.36e-60
The total number of observations was  2800  with MLE Chi Square =  1013.785  with prob <  4.64e-122

Tucker Lewis Index of factoring reliability =  0.9222
RMSEA index =  0.043  and the 90 % confidence intervals are  0.0403 0.0454
BIC =  -295.882
Fit based upon off diagonal values = 0.991
Measures of factor score adequacy
                                                   ML1   ML2   ML3   ML5   ML4   ML6
Correlation of scores with factors               0.916 0.935 0.879 0.860 0.837 0.807
Multiple R square of scores with factors         0.839 0.875 0.772 0.739 0.701 0.651
Minimum correlation of possible factor scores  0.678 0.750 0.545 0.478 0.402 0.302
```

格納された因子得点を確認するには，オブジェクトの scores 変数を指定しましょう。ここでは head 関数を使って，最初の数人分の因子得点を確認します。

ソースコード 7.7　因子得点

```
> head(fa.result$scores) # 因子得点
               ML1         ML2        ML3        ML5        ML4          ML6
61617 -0.233869576 -0.09660374 -1.6427054 -0.7478329  1.3297857 -0.913788881
61618  0.349364870  0.03948985 -0.5163990 -0.1956834  0.2265806  0.053653296
61620  0.006487571  0.69876718 -0.0214491 -0.7157791 -0.2559985 -0.004667037
61621 -0.163451729 -0.01110545 -0.8806610 -0.3484800  1.2996553  0.388638549
61622  0.288477162 -0.29990788 -0.2982969 -0.7639309  0.5447726 -0.479501495
61623  1.168699158  0.16655125  1.4715096  0.1273263 -0.5252744  0.112873954
```

7.3　信頼性係数の算出

因子分析によって，尺度のなかにどのような因子が含まれているか（下位因子）が検討できますが，尺度としては一次元性を持っている必要があります。いわゆる尺度の内的整合性信頼性の検証と言われるもので，α 係数を算出することで検証します。一般に α 係数が 0.8 を超えていると内的整合性信頼性がある，といってよいとされています。

α 係数を求めるには，psych パッケージの alpha 関数を使います。

ソースコード 7.8　信頼性係数の算出

```
# クロンバックのα
```

```
> alpha(bfi[1:25])

Reliability analysis
Call: alpha(x = bfi[1:25])

  raw_alpha std.alpha G6(smc) average_r mean   sd
      0.82      0.82    0.86      0.15  4.2 0.61

（中略）

 警告メッセージ:
In alpha(bfi[1:25]) :
  Some items were negatively correlated with total scale and were automatically reversed.
```

この結果を見ると，0.82 ですのでひとつの尺度として十分なまとまりがあったことが分かります。ところで，alpha 関数を実行すると最後に警告メッセージが示されています。α 係数は，全ての項目が同じ方向を向いていることを前提していますから，項目間の相関係数は全て正になるはずです。もっとも，心理尺度の場合わざと逆転項目を入れて聞くことがありますので，その場合は負の相関係数がでます。統計ソフトによっては，逆転項目を指定したり，逆転項目の数値を反転させた別変数を作って信頼性係数を計算したりする必要がありますが，psych パッケージの alpha 関数はこの警告メッセージにあるように，「合計得点と負の相関をしている項目は自動的に逆転」してくれるのです。大変便利ですね。

信頼性係数のとしてはこの α 係数が有名ですが，この係数はあくまでも真の信頼性の近似値でしかありません。因子分析によって 6 因子構造だとわかっていても，一次元性を検証していることからも，厳密な意味で項目の構造を反映した信頼性係数になっていないことがわかります。論文によっては，因子の項目ごとに α 係数を算出しているものもありますが，α 係数は項目の数が少なくなると小さくなる傾向がありますから，下位因子の中では信頼性係数が十分な数値でなかった，ということもあります。

こんなとき，因子の構造をもふまえた信頼性係数として ω 係数というものがあります。この ω 係数を算出する関数も，psych パッケージには含まれており，関数名はそのまま omega です。実際には次のようにして使います。

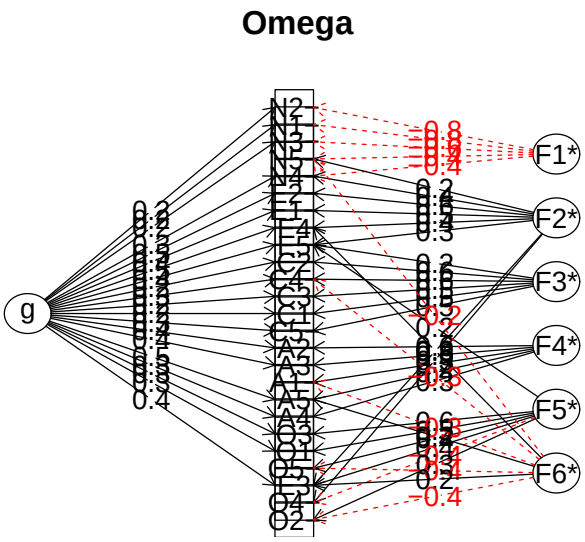


図 7.1　ω 係数のイメージ

ソースコード 7.9　信頼性係数の算出 2

```
> omega(bfi[1:25],6,fm="ml")
Omega
Call: omega(m = bfi[1:25], nfactors = 6, fm = "ml")
Alpha:                 0.82
G.6:                   0.86
Omega Hierarchical:    0.47
Omega H asymptotic:    0.53
Omega Total            0.88

(後略)
```

ω 係数は因子構造を考慮しますので，因子数や因子抽出法を指定することができます（抽出法のデフォルトは fa 関数と同じで，fm=”minres”になっています）。今回は因子分析と同じ設定にしました。同時にプロットも示されますが (図 7.1)，これを見ていただければ分かるように，因子構造を仮定した上で全体の一次元性を評価している（左側の全体共通因子 g）ことが分かるかと思います。今回の ω 係数は 0.86 でした。ω 係数は常に α 係数より大きくなることが知られているので，信頼性の下限として α でいいではないか，と主張する向きもありますが，これほど因子分析が使われている現状を考えると，これからは ω 係数の方が良いかもしれません（[5])。

第 8 章

構造方程式モデリングについて

8.1　構造方程式モデリングとは

「構造方程式モデリング」は Structural Equation Modeling の訳です。頭文字をとって “SEM” とも呼ばれます。SEM では複数の変数間の関係を記述して，データとのあてはまりを評価したり，変数間の影響力の大きさを求めたりすることができます。SEM は因子分析モデルや回帰モデルに加え，分散分析など様々な分析モデルを包含しています [6]。

変数間の関係を図に描き起こしたものを「パス図」と呼びます。パス図では 2 つの変数の間に因果関係を仮定する場合には「→」を，双方向の因果関係を仮定する場合には「⇄」を，相関関係を仮定する場合には「↔」を描きます。

SEM に関する用語を簡単におさらいしておきましょう。

- 観測変数と潜在変数
  観測変数　直接観測できる変数（例：身長，体重，テストの得点）。パス図では四角形で表す
  潜在変数　構成概念（例：知能，抑うつ）。パス図では円で表す
- 測定方程式と構造方程式
  測定方程式　潜在変数が観測変数によってどのように測定されているかを示す方程式
  構造方程式　因果関係を示す方程式
- 外生変数と内生変数
  外生変数　モデルの中で一度も他の変数の結果となっていない変数。パス図では「→」が一度も刺さっていない変数
  内生変数　少なくとも一度は他の変数の結果となっている変数。パス図では「→」が刺さっている変数

8.1.1　パス図の例

先に述べたように，単回帰分析は $Y = aX + b$ で表されます。これをパス図で表現すると図 8.1 のようになります。

因果関係を記述していますから，これは構造方程式です。図 8.1 において，a は潜在変数に，X と Y は観測変数に相当します。また X と a は外生変数に，Y は内生変数に相当します。

次に，因子分析のモデルをパス図で表してみます（図 8.2）。

図 8.2 は項目 $X1$，$X2$，$X3$ の背後に因子 F が存在していることを示しています。$X1$，$X2$，$X3$ は観測変数，F は潜

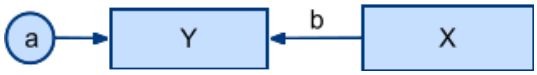


図 8.1　単回帰分析のパス図

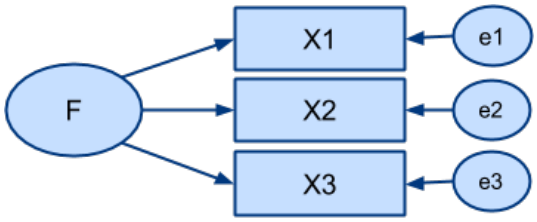


図 8.2　因子分析のパス図

表 8.1　適合度指標

適合度指標		説明
GFI	Goodness of Fit Index	
AGFI	Adjusted GFI	0.9 以上であてはまりがよい
NFI	Normed Fit Index	
CFI	Comparative Fit Index	
RMSEA	Root Mean Square Error of Approximation	0.05 以下であてはまりがよい，0.1 以上であてはまりがよくない
AIC	Akaike Information Criterion	値が小さいほどデータとモデルの乖離度の小さい，よいモデルである

在変数に相当します。潜在変数が観測変数によってどのように測定されているかを記述していますから，これは測定方程式です。

SEM では研究者が立てた仮説をもとにモデルを作成し，それをデータとつきあわせながら評価します。あてはまりが悪ければ，モデルの修正が必要になってきます。次に，モデルの評価と修正についてみていきましょう。

8.1.2　モデルの評価

SEM ではモデルのあてはまりのよさを検討することができます。χ^2 検定は古くからしばしば用いられる方法です。「分析したモデルが正しい」という仮説を帰無仮説として，検定を行います。一般的な検定では「帰無仮説が棄却される」ことが目的とされますが，SEM における χ^2 検定では逆に「帰無仮説が棄却されない」ことが目的とされることになります。

ただし，χ^2 検定では「分析したモデルが正しい」という仮説を棄却しない，つまり「分析したモデルが間違っているとは言い切れなかった」という，不明確なロジックによってモデルの適合を主張することになります。またモデルが複雑な場合や，データが多変量正規分布に従っていない場合，標本数が多い場合に，検定の精度が低下することが知られています [7]。

そこで現在は χ^2 検定に加えてさまざまな「適合度指標」が主に用いられています。表 8.1 に適合度指標とその値の見方を示します。

SEM では，適合度指標を見ながらよりあてはまりのよい，理論的にも整合性のとれたモデルを，パスを加えたり削除したりと試行錯誤しながら作っていくことになります。

8.1.3　モデルの修正

モデルを修正する上では，修正指標を参考にすることができます。修正指標とは，モデル上で新たにパスを引いた場合の χ^2 値の減少量の予測値です。この数値が大きいほど新たにパスを設定することの意味が大きいといえます。

ただし，SEM は「修正指標をもとに，適合度指標のもっともよいモデルを作成する分析」ではありません。修正指標は，仮説に基づく理論的な観点からの修正を提案してくれるわけではないのです。修正指標のもと，新たなパスを付け加えたり，パスを削除したりする場合は，それが理論的に説明できるかが前提になるといえるでしょう。

ところで，試行錯誤しながらモデルを修正していくうちに，誤差項の間に相関を設定することで適合度が向上する場合があることに気づくこともあるでしょう。では，モデルの適合度向上のためだけに，事後的に誤差相関を設定してよいのでしょうか。

誤差項の間に相関を設定することは，その観測変数間にモデルで導入された変数以外の共通変動要因が存在することを意味し，そのような仮説のもと設定されるべきです [8]。したがって，適合度指標を上げるためだけに闇雲に誤差項の間に相関を設定するべきではありません。

8.2　構造方程式モデリングで用いるパッケージ

R には構造方程式モデリングを実行するパッケージとして sem パッケージと lavaan パッケージが用意されています。ここでは lavaan パッケージを用いた方法を取り上げます。まず lavaan パッケージをインストールして読み込みましょう。

ソースコード 8.1　lavaan パッケージの導入

```
> install.packages("lavaan")
> library("lavaan")
```

8.2.1　モデルの記法

lavaan パッケージでは以下のようにモデルを式で表現します。

- 潜在変数= 観測変数 1+2+3+... で測定方程式
- 変数 X 変数 Y1+Y2+Y3+... で構造方程式
- 変数 X　変数 Y で相関関係（共分散）
- 他にも 1 で切片，f1　0*f2 で直交，など

8.3　分析事例

ここでは小塩真司先生の Amos の分析事例 [9] を R で再現しながら，lavaan パッケージの使い方を学んでいきましょう。

8.3.1　測定変数を用いたパス解析

「分析例１：測定変数を用いたパス解析 [10]」と同じ分析を R で実行してみましょう。以下のコマンドを実行してください。

ソースコード 8.2　測定変数を用いたパス解析 1:ファイルの読み込み

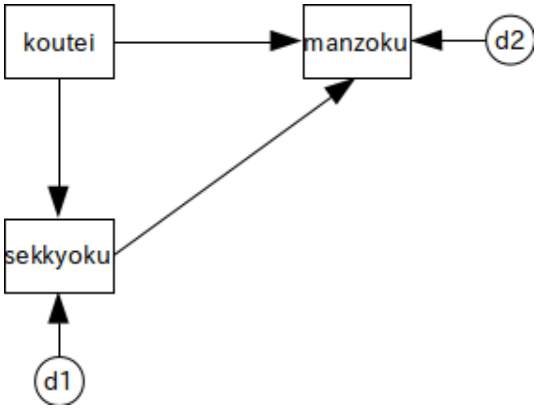


図 8.3　パス解析のモデル

```
> ip <- read.csv(file.choose()) # data10-1.csv を[開く]
> ip
   koutei sekkyoku manzoku
1      34        7       8
2      31        6       4
3      30        4       3
4      17        3       5
5      13        2       2
6      18        4       5
7      17        4       2
8      28        4       7
9      23        5       5
10     29        3       4
11     33        7       8
12     25        3       2
13     37        5       6
14     24        6       5
15     33        6       7
16     38        8       7
17     23        5       3
18     20        3       2
19     26        6       8
20     37        3       7
```

ここで koutei は「自己肯定感」の観測変数，sekkyoku は「対人積極性」の観測変数，manzoku は「対人関係の満足度」の観測変数を表します。「自己肯定感は対人関係の満足度に影響を及ぼす」，「自己肯定感は対人積極性に影響を及ぼす」，「対人積極性は対人関係の満足度に影響を及ぼす」であろうという仮説を設定し，その仮説が成り立つかどうかを検証します。この仮説モデルをパス図に表すと図 8.3 のようになります。

このモデルを lavaan の記法で表すと次のようになります。

ソースコード 8.3　測定変数を用いたパス解析 2:モデルの記述

```
> model.ip <- '
+ manzoku ~ koutei + sekkyoku
+ sekkyoku ~ koutei
+ '
```

図 8.3 を見ると，koutei と sekkyoku から伸びる矢印が manzoku に突き刺さっていますよね。これを “manzoku koutei + sekkyoku” と表現しています。同様に koutei からの矢印が sekkyoku に伸びていますから，“sekkyoku koutei” となるわけです。回帰分析のときと同じですね。

では早速このモデルが代入された model.ip を使って，SEM によるパス解析を実行してみましょう。lavaan パッケージに含まれている sem 関数を用います。

ソースコード 8.4　測定変数を用いたパス解析 3:モデルのフィッティング

```
> result.ip <- sem(model.ip,data=ip)
```

結果をみるには summary 関数を用います。オプションとして standardized=TRUE をつけると標準化係数を，fit.measures=TRUE をつけると適合度指標を，modindices=TRUE をつけると修正指標をあわせて出力することができます。

ソースコード 8.5　測定変数を用いたパス解析 4:出力関数

```
 > summary(result.ip, standardized=TRUE)

lavaan (0.5-14) converged normally after  16 iterations

  Number of observations                            20

  Estimator                                         ML
  Minimum Function Test Statistic                0.000
  Degrees of freedom                                 0
  P-value (Chi-square)                           0.000

Parameter estimates:

  Information                                 Expected
  Standard Errors                             Standard

                   Estimate  Std.err  Z-value  P(>|z|)   Std.lv  Std.all
Regressions:
  manzoku ~
    koutei            0.120    0.056    2.159    0.031    0.120    0.408
    sekkyoku          0.533    0.248    2.147    0.032    0.533    0.406
  sekkyoku ~
    koutei            0.129    0.041    3.165    0.002    0.129    0.578

Variances:
    manzoku           2.146    0.678                      2.146    0.477
    sekkyoku          1.739    0.550                      1.739    0.666
```

ここでは “Parameter estimates:” に注目しましょう。“Regressions:” の “Estimate” の列に書かれている値（0.120, 0.533, 0.129）が非標準化推定値，“Std.all” の列に書かれている値（0.408, 0.406, 0.578）が標準化推定値です。それぞれのパスが有意であるかどうかは “P(> |z|)” を見てください。すべて有意になっています。それぞれ SPSS の出力 [10] とほぼ一致していますね。

適合度指標や修正指標については次の節で見ていきます。

8.3.2　潜在変数間の因果関係

次に，「分析例 2（潜在変数間の因果関係）[11]」と同じ分析を R で実行してみましょう。

ソースコード 8.6　潜在変数感の因果関係モデル 1:ファイルの読み込み

```
 > test <- read.csv(file.choose()) # data10-2.csv を[開く]
 > head(test)
  benkyo_a benkyo_b kitai_a kitai_b jishin_a jishin_b
1        5        6       2       3       36       31
2        4        4       5       6       51       45
3        4        7       6       5       62       41
4        5        5       5       4       50       28
5        4        5       4       5       60       38
6        5        7       3       2       50       34
```

それぞれ次のような変数となっています。

- `benkyo_a`: 勉強量に関する項目得点
- `benkyo_b`: 勉強量に関する項目得点
- `kitai_a`: 成績への期待に関する項目得点
- `kitai_b`: 成績への期待に関する項目得点
- `jishin_a`: テスト結果の自信に関する 10 項目からなる下位尺度得点
- `jishin_b`: テスト結果の自信に関する 10 項目からなる下位尺度得点

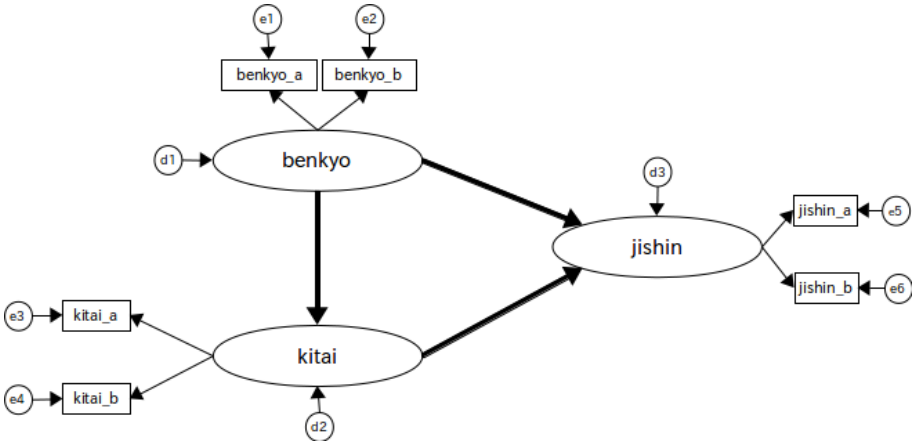


図 8.4　テストに関するモデル

ここでは「勉強量が成績への期待に影響を及ぼす」,「勉強量はテストの自信に影響を及ぼす」,「成績への期待はテストの自信に影響を及ぼす」というモデルを検証します。このモデルをパス図に表すと図 8.4 のようになります。

では，このモデルを lavaan の記法で表してみましょう。

ソースコード 8.7　潜在変数感の因果関係モデル 2:モデルの記述

```
> model.test1 <- '
+ benkyo =~ benkyo_a + benkyo_b
+ kitai =~ kitai_a + kitai_b
+ jishin =~ jishin_a + jishin_b
+ jishin ~ benkyo + kitai
+ kitai ~ benkyo
+ '
```

benkyo =~ benkyo_a + benkyo_b は測定方程式の記法です。“観測変数 benkyo_a と benkyo_b が潜在変数 benkyo から影響を受けている” ことを表します。その下 2 行も同様です。“ jishin ~ benkyo + kitai” は構造方程式の記法で，“jishin は勉強と期待から影響を受ける” ことを表します。やはり回帰分析と同じですね。最後の行も同様です。

ではこのモデルを sem 関数で分析して，summary 関数で結果を出力してみましょう。ここでは適合度指標と修正指標を出力するために fit.measures オプションと modindices オプションを TRUE に指定しました。次のように出力されます。

ソースコード 8.8　潜在変数感の因果関係モデル 3:モデルのフィッティング

```
> result.test1 <- sem(model.test1,data=test)
> summary(result.test1, standardized=TRUE, fit.measures=TRUE, modindices=TRUE)

lavaan (0.5-14) converged normally after  64 iterations

  Number of observations                            30

  Estimator                                         ML
  Minimum Function Test Statistic                9.162
  Degrees of freedom                                 6
  P-value (Chi-square)                           0.165

Model test baseline model:

  Minimum Function Test Statistic               94.929
  Degrees of freedom                                15
  P-value                                        0.000

Full model versus baseline model:

  Comparative Fit Index (CFI)                    0.960
  Tucker-Lewis Index (TLI)                       0.901
```

```
Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)               -378.582
  Loglikelihood unrestricted model (H1)       -374.001

  Number of free parameters                         15
  Akaike (AIC)                                 787.164
  Bayesian (BIC)                               808.182
  Sample-size adjusted Bayesian (BIC)          761.479

Root Mean Square Error of Approximation:

  RMSEA                                          0.133
  90 Percent Confidence Interval          0.000  0.294
  P-value RMSEA <= 0.05                          0.202

Standardized Root Mean Square Residual:

  SRMR                                           0.052

Parameter estimates:

  Information                                 Expected
  Standard Errors                             Standard

                   Estimate  Std.err  Z-value  P(>|z|)   Std.lv  Std.all
Latent variables:
  benkyo =~
    benkyo_a          1.000                               0.970    0.806
    benkyo_b          1.213    0.313    3.883    0.000    1.177    0.918
  kitai =~
    kitai_a           1.000                               1.161    0.874
    kitai_b           0.787    0.240    3.278    0.001    0.914    0.753
  jishin =~
    jishin_a          1.000                               9.251    0.875
    jishin_b          0.901    0.177    5.082    0.000    8.331    0.865

Regressions:
  jishin ~
    benkyo            5.468    1.730    3.161    0.002    0.573    0.573
    kitai             4.659    1.625    2.867    0.004    0.585    0.585
  kitai ~
    benkyo           -0.029    0.258   -0.112    0.911   -0.024   -0.024

Variances:
    benkyo_a          0.508    0.240                       0.508    0.350
    benkyo_b          0.260    0.303                       0.260    0.158
    kitai_a           0.418    0.361                       0.418    0.237
    kitai_b           0.637    0.269                       0.637    0.433
    jishin_a         26.182   14.055                      26.182    0.234
    jishin_b         23.355   11.672                      23.355    0.252
    benkyo            0.941    0.404                       1.000    1.000
    kitai             1.347    0.561                       0.999    0.999
    jishin           29.555   16.169                       0.345    0.345

Modification Indices:

        lhs op      rhs    mi    epc sepc.lv sepc.all sepc.nox
1    benkyo =~ benkyo_a    NA     NA      NA       NA       NA
2    benkyo =~ benkyo_b 0.000  0.000   0.000    0.000    0.000
3    benkyo =~  kitai_a 0.808 -0.217  -0.210   -0.158   -0.158
4    benkyo =~  kitai_b 0.808  0.171   0.165    0.136    0.136
5    benkyo =~ jishin_a 1.478  2.545   2.469    0.234    0.234
6    benkyo =~ jishin_b 1.478 -2.292  -2.224   -0.231   -0.231
7     kitai =~ benkyo_a 0.010  0.014   0.017    0.014    0.014
8     kitai =~ benkyo_b 0.010 -0.017  -0.020   -0.016   -0.016
9     kitai =~  kitai_a    NA     NA      NA       NA       NA
10    kitai =~  kitai_b 0.000  0.000   0.000    0.000    0.000
11    kitai =~ jishin_a 1.478 -2.169  -2.518   -0.238   -0.238
12    kitai =~ jishin_b 1.478  1.953   2.267    0.235    0.235
13   jishin =~ benkyo_a 0.010  0.003   0.028    0.023    0.023
14   jishin =~ benkyo_b 0.010 -0.004  -0.034   -0.026   -0.026
15   jishin =~  kitai_a 0.808 -0.039  -0.357   -0.269   -0.269
16   jishin =~  kitai_b 0.808  0.030   0.281    0.232    0.232
17   jishin =~ jishin_a    NA     NA      NA       NA       NA
18   jishin =~ jishin_b 0.000  0.000   0.000    0.000    0.000
19 benkyo_a ~~ benkyo_a 0.000  0.000   0.000    0.000    0.000
20 benkyo_a ~~ benkyo_b    NA     NA      NA       NA       NA
21 benkyo_a ~~  kitai_a 1.047 -0.153  -0.153   -0.096   -0.096
22 benkyo_a ~~  kitai_b 1.595  0.171   0.171    0.117    0.117
23 benkyo_a ~~ jishin_a 3.024 -1.927  -1.927   -0.151   -0.151
24 benkyo_a ~~ jishin_b 3.035  1.738   1.738    0.150    0.150
25 benkyo_b ~~ benkyo_b 0.000  0.000   0.000    0.000    0.000
```

```
26 benkyo_b ~~  kitai_a 0.163  0.067   0.067    0.039    0.039
27 benkyo_b ~~  kitai_b 0.280 -0.077  -0.077   -0.050   -0.050
28 benkyo_b ~~ jishin_a 5.055  2.823   2.823    0.208    0.208
29 benkyo_b ~~ jishin_b 5.068 -2.546  -2.546   -0.206   -0.206
30  kitai_a ~~  kitai_a 0.000  0.000   0.000    0.000    0.000
31  kitai_a ~~  kitai_b    NA     NA      NA       NA       NA
32  kitai_a ~~ jishin_a 1.473 -1.787  -1.787   -0.127   -0.127
33  kitai_a ~~ jishin_b 1.412  1.576   1.576    0.123    0.123
34  kitai_b ~~  kitai_b 0.000  0.000   0.000    0.000    0.000
35  kitai_b ~~ jishin_a 0.279  0.664   0.664    0.052    0.052
36  kitai_b ~~ jishin_b 0.250 -0.567  -0.567   -0.049   -0.049
37 jishin_a ~~ jishin_a 0.000  0.000   0.000    0.000    0.000
38 jishin_a ~~ jishin_b    NA     NA      NA       NA       NA
39 jishin_b ~~ jishin_b 0.000  0.000   0.000    0.000    0.000
40   benkyo ~~   benkyo 0.000  0.000   0.000    0.000    0.000
41   benkyo ~~    kitai    NA     NA      NA       NA       NA
42   benkyo ~~   jishin    NA     NA      NA       NA       NA
43    kitai ~~    kitai 0.000  0.000   0.000    0.000    0.000
44    kitai ~~   jishin    NA     NA      NA       NA       NA
45   jishin ~~   jishin 0.000  0.000   0.000    0.000    0.000
46   jishin  ~    kitai 0.000  0.000   0.000    0.000    0.000
47   jishin  ~   benkyo 0.000  0.000   0.000    0.000    0.000
48    kitai  ~   jishin    NA     NA      NA       NA       NA
49    kitai  ~   benkyo 0.000  0.000   0.000    0.000    0.000
50   benkyo  ~   jishin    NA     NA      NA       NA       NA
51   benkyo  ~    kitai    NA     NA      NA       NA       NA
```

小塩先生の Web ページ [11] では GFI と AGFI，RMR を算出しています。lavaan パッケージでは次のように入力することで得られます。

ソースコード 8.9　潜在変数感の因果関係モデル 4:適合度指標の出力

```
> fitMeasures(result.test1, c("gfi", "agfi","rmr"))
  gfi  agfi   rmr
0.920 0.719 0.471
```

ではここで出力結果をみていきましょう。

χ^2 検定については Minimum Function Test Statistic を見ます。P-value が 0.165 ですから，帰無仮説は棄却されません。したがって，χ^2 検定の結果からはこのモデルは正しいといえそうです。しかし適合度指標をみてみると，GFI は 0.920 と良好であるものの，AGFI が 0.719 と不良です。RMSEA は 0.1 以上であてはまりがよくないと判断（表 8.1）しますし，あまり適合度はよくないようです。

出力結果を眺めていると，“Regressions:” の “kitai　benkyo” の有意確率 ($P > |z|$) は 0.911 となっています。したがって，“kitai　benkyo” のパスは有意ではありません。どうやらモデルの見直しが必要のようです。

また，修正指標（Modification Indices: mi）をみてみましょう。これは，因果関係（~）や相関関係（~~）のパスを新たに設定することで予測される χ^2 値の減少量を示しています。たとえば 29 列目をみると，“ benkyo_b ~~ jishin_b ” の修正指標が 5.068 となっています。新たに “ benkyo_b ~~ jishin_b” を設定すれば，モデルが改善されるかもしれません。しかし，観測変数 “　benkyo_b” と “ jishin_b” の間に相関があるとする理論的根拠は特にありませんから，ここではスルーすることとします。

以上の結果をもとに，モデルを見直してみましょう。有意でないパスは “kitai　benkyo” でした。これを削除したモデル model.test2 を新たに作成してみましょう。

ソースコード 8.10　潜在変数感の因果関係モデル 5:モデルの修正

```
> model.test2 <- '
+ benkyo =~ benkyo_a + benkyo_b
+ kitai =~ kitai_a + kitai_b
+ jishin =~ jishin_a + jishin_b
+ jishin ~ benkyo + kitai
+ kitai ~~ 0*benkyo
+ '
```

ここで “kitai ~~ 0*benkyo” という行を加えていることに注意してください。これは kitai と benkyo が無相関であることを明示するものです。lavaan パッケージでは，デフォルトですべての外生的潜在変数に相関を設定するように

なっています。この行を加えないと，lavaan パッケージは “kitai ~~ benkyo” という相関のパスを勝手に加えてしまします。model.test1 から有意でないパス “kitai ~ benkyo” をただ削除したかっただけですから，kitai と benkyo が無相関であると宣言する必要があるのです。

では先程と同様に sem 関数で分析して summary 関数や fitMeasures 関数で出力してみましょう。

ソースコード 8.11　潜在変数感の因果関係モデル 5:モデルのフィッティング

```
 > result.test2 <- sem(model.test2,data=test)
 > summary(result.test2, standardized=TRUE, fit.measures=TRUE, modindices=TRUE)

lavaan (0.5-14) converged normally after  66 iterations

  Number of observations                            30

  Estimator                                         ML
  Minimum Function Test Statistic                9.173
  Degrees of freedom                                 7
  P-value (Chi-square)                           0.240

Model test baseline model:

  Minimum Function Test Statistic               94.929
  Degrees of freedom                                15
  P-value                                        0.000

Full model versus baseline model:

  Comparative Fit Index (CFI)                    0.973
  Tucker-Lewis Index (TLI)                       0.942

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)               -378.588
  Loglikelihood unrestricted model (H1)       -374.001

  Number of free parameters                         14
  Akaike (AIC)                                 785.176
  Bayesian (BIC)                               804.793
  Sample-size adjusted Bayesian (BIC)          761.203

Root Mean Square Error of Approximation:

  RMSEA                                          0.102
  90 Percent Confidence Interval          0.000  0.261
  P-value RMSEA <= 0.05                          0.288

Standardized Root Mean Square Residual:

  SRMR                                           0.051

Parameter estimates:

  Information                                 Expected
  Standard Errors                             Standard

                   Estimate  Std.err  Z-value  P(>|z|)   Std.lv  Std.all
Latent variables:
  benkyo =~
    benkyo_a          1.000                               0.969    0.805
    benkyo_b          1.216    0.314    3.876    0.000    1.178    0.919
  kitai =~
    kitai_a           1.000                               1.156    0.870
    kitai_b           0.794    0.241    3.286    0.001    0.918    0.756
  jishin =~
    jishin_a          1.000                               9.307    0.877
    jishin_b          0.899    0.175    5.131    0.000    8.369    0.866

Regressions:
  jishin ~
    benkyo            5.446    1.713    3.180    0.001    0.567    0.567
    kitai             4.672    1.621    2.882    0.004    0.580    0.580

Covariances:
  benkyo ~~
    kitai             0.000                               0.000    0.000

Variances:
    benkyo_a          0.509    0.240                      0.509    0.352
    benkyo_b          0.257    0.304                      0.257    0.156
```

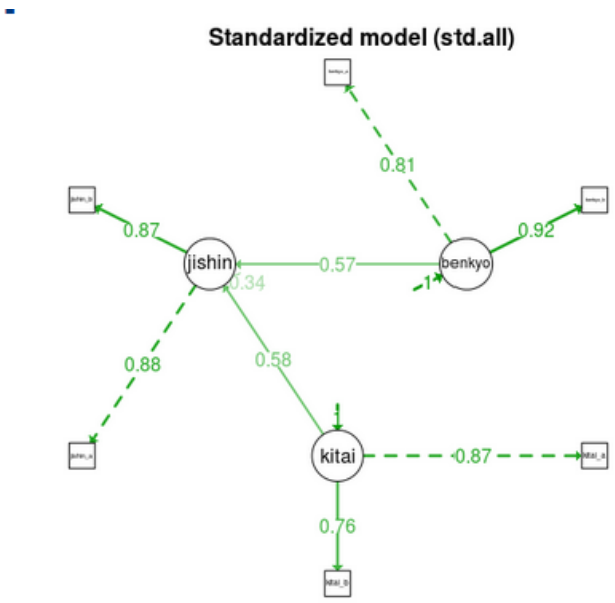


図 8.5　qgraph パッケージで描いたパス図

```
    kitai_a          0.429    0.358                      0.429    0.243
    kitai_b          0.630    0.269                      0.630    0.428
    jishin_a        26.050   14.017                     26.050    0.231
    jishin_b        23.461   11.643                     23.461    0.251
    benkyo           0.940    0.404                      1.000    1.000
    kitai            1.337    0.558                      1.000    1.000
    jishin          29.576   16.141                      0.341    0.341

Modification Indices:
 （略）

 > fitMeasures(result.test2, c("gfi", "agfi","rmr"))

  gfi  agfi   rmr
0.919 0.758 0.565
```

model.test1 と同様，model.test2 でも χ^2 検定の結果からはこのモデルは正しいといえそうです。適合度指標をみてみると，AGFI が 0.719 から 0.758 に改善しています。RMSEA も RMSEA は 0.133 から 0.102 に減少し，改善がみられます。AIC は 787.164 から 785.176 に減少しています。AIC は値が小さいほどデータとモデルの乖離度の小さい，よいモデルであると判断します（表 8.1）。さらに，すべてのパスが有意となっています。

ところで，小塩先生の Web ページ [11] で書かれている値と，今回 lavaan パッケージで出力された値に異なる点があることに注意してください。特に AIC はかなり異なる値となっています。これは両者の計算方法が異なるためです。

8.3.3　パス図を描くには

R でパス図を描くことは可能ですが，まだまだ発展途上といえます。ここでは qgraph パッケージと semPlot パッケージをご紹介します。

qgraph パッケージを用いてパス図を描くには次のように入力します。

```
> install.packages("qgraph")
> library(qgraph)
> qgraph(result.test2)
```

図 8.5 のように出力されます。

……やや期待外れのパス図だったでしょうか。

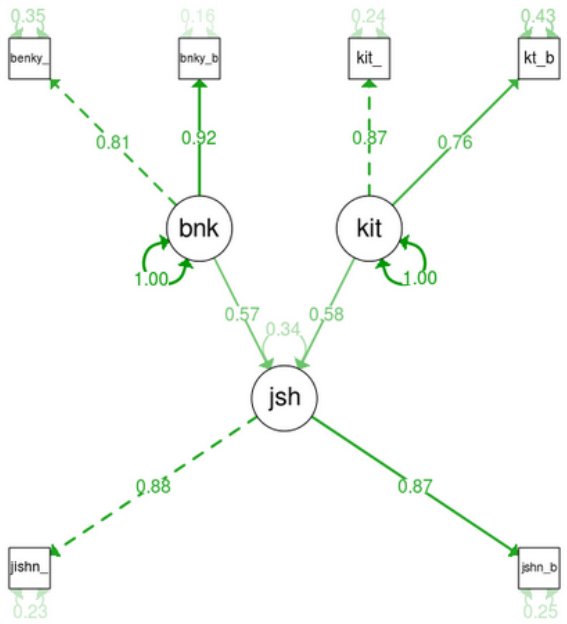


図 8.6　semPlot パッケージで描いたパス図

なお，qgraph 関数を実行すると “This funcion is deprecated, Use the ’semPlot’ package instead.（参考訳：この関数は将来的に廃止される予定です。代わりに semPlot パッケージを使ってね）” という警告メッセージが表示されます。では semPlot パッケージではどうでしょうか。

ソースコード 8.12　semPlot パッケージの利用

```
> install.packages("semPlot")
> library(semPlot)
> semPaths(result.test2, "std")
```

図 8.6 のように出力されます。

……やはりまだまだ，といったところですかね。まだまだバージョンも低い[*1]ですし，今後に期待といったところでしょうか。

現実解としては R にこだわらず，ドロー系のソフトを使って自分で描いたほうが確実だと思われます。FLOSS ですと，LibreOffice Draw が定番でしょうか。図 8.7 のような美しいパス図を描くことができます[*2]。

[*1] 2013 年 11 月 12 日現在，バージョン 0.3.3

[*2] なお，PowerPoint を使ってパス図を描くのはオススメしません。PowerPoint はプレゼンテーションソフトです。

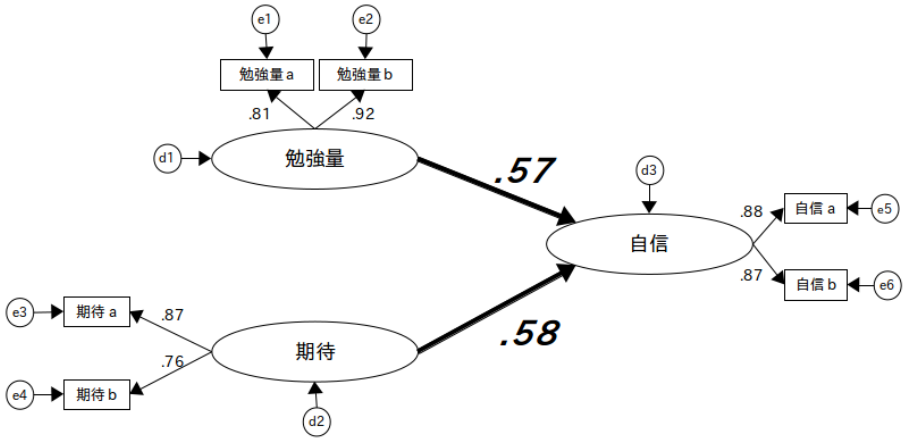


図 8.7　LibreOffice Draw で描いたパス図

第 9 章

あとがき

このテキストは，2013 年 11 月 16 日に山口大学教育学部で行われた，中国四国心理学会第 69 回大会特別企画，R チュートリアルセミナーで配布されたテキストです。

チュートリアルセミナーは午前 9 時から 3 時間にわたって開催され，R のインストールから構造方程式モデリングまで，非常に幅広い内容を集中的に行いました。幸い多くの方にご参加いただきましたが，こちらの伝えたいことが時間内に十分伝わったかどうか，と言われると少し不安になるのも事実です。

限られた時間での紹介でしたので，十分に伝えられなかった分はテキストの文中でご理解いただこうとしたため，少しボリュームのあるテキストになってしまいました。これをその一回だけの資料とするのはもったいないように思いましたので，こうして電子書籍として公開することとなりました。

本書が少しでも皆様のお役に立てればと願っております。

（小杉考司）

小杉先生のお誘いで，「R チュートリアルセミナー」の講師を務めさせていただきました。本書では「自由で開かれていること」と「構造方程式モデリングについて」の章を担当しました。私は心理統計が専門というわけではないのですが，セミナーや本書に関わらせていただくことを通して，とてもよい勉強になりました。セミナーに参加して下さった皆様，また本書をお読みいただいた皆様，さらに R を含む FLOSS コミュニティの皆様に感謝致します。

（押江隆）

カバーアートは Vector Art (http://www.thevectorart.com/) の画像 (http://www.thevectorart.com/abstract-blue-background-vector-art-100.html) (©2011 Vector Art) と R Logo (http://developer.r-project.org/Logo/) (©2010 Tobias Wolf) をもとに作成しました。

参考文献

[1] 自由ソフトウェアとは? - GNU プロジェクト - フリーソフトウェアファウンデーション, http://www.gnu.org/philosophy/free-sw.ja.html（閲覧日：2013 年 11 月 11 日）

[2] Excel 使うな — Okumura's Blog, http://oku.edu.mie-u.ac.jp/~okumura/blog/node/2287（閲覧日：2013 年 11 月 11 日）

[3] Excel 使うな：2010 版 — Okumura's Blog http://oku.edu.mie-u.ac.jp/~okumura/blog/node/2585（閲覧日：2013 年 11 月 11 日）

[4] 田中亨「Excel の謎」（第 72 回）:「2007」に関数の不具合？ーー過ちを繰り返す「DATEDIF」の受難 http://pc.nikkeibp.co.jp/article/column/20081118/1009774/（閲覧日：2013 年 11 月 11 日）

[5] 清水和秋 2007 αはやめてωにしよう ー因子分析で構成した尺度の共通性と信頼性ー, 日本心理学会第 71 回大会発表論文集 2007.9 pp.416,http://kuir.jm.kansai-u.ac.jp/dspace/handle/10112/2329

[6] 狩野裕 (2002): 構造方程式モデリングは，因子分析，分散分析，パス解析のすべてにとって代わるのか？, 行動計量学，**29**(2), 138-159.

[7] 室橋弘人 (2003): χ^2 検定, 豊田秀樹 (編) 共分散構造分析 [疑問編] ——構造方程式モデリング, 朝倉書店, pp.120-121.

[8] 米村大介 (2003): 誤差間に相関を設定する場合, 豊田秀樹 (編) 共分散構造分析 [疑問編] ——構造方程式モデリング, 朝倉書店, p.25.

[9] 小塩真司：心理データ解析 Basic, http://www.f.waseda.jp/oshio.at/edu/data_b/top.html（閲覧日：2013 年 11 月 12 日）

[10] 小塩真司：心理データ解析第 10 回 (2) 分析例 1（測定変数を用いたパス解析）, http://www.f.waseda.jp/oshio.at/edu/data_b/10_folder/da10_02.html（閲覧日：2013 年 11 月 12 日）

[11] 小塩真司：心理データ解析第 10 回 (3) 分析例 2（潜在変数間の因果関係）, http://www.f.waseda.jp/oshio.at/edu/data_b/10_folder/da10_02.html（閲覧日：2013 年 11 月 12 日）