データ分析統合環境
PADOC/STAT
ver2.8.1

操作マニュアル ver1.3

AIIT

使用上の注意

Note1　本文書の文責と著作権は下記にあります。無断転載を禁じます。

産業技術大学院大学　客員研究員　中井眞人

Note2　本書のプログラムは下記の PADOC/Stat データ分析統合環境より Download できる。しかし使用上で発生した問題については責任外とする。

`http://www.padoc.info`

現状では widows 上で稼動しメモリは 512M 以上で次の OS での動作しか確認できていない。

- windows XP
- win7 32/64 bit
- win10 32/64 bit

Note3　使用上での誤記や問題があれば速やかにご連絡をお願いしたい。

mail:pionix@gmail.com

現状では以下を対応中である。

- ~~ユーザが独自に開発した分析モデルの組込み~~→対応済み 4 章参照
- データフロー図上でのインタラクティブな分析のマニュアル

0.0.1 概要

本文章はデータ分析統合環境「PADOC/Stat」の操作マニュアルである。
本文章は次の構成となっている。

1. コマンドベースの操作環境
2. コマンドベースでの操作
3. コマンド記述による分析
4. PLOT 表示
5. ユーザ登録コマンド
6. 付録
7. データフローによる操作環境 (後述)
8. データフローによる分析（後述)

目次

第 1 章

PADOC/STAT コマンドベースの操作環境

1.1　コマンドベースの操作環境の概要

図 1.1 に示す様に PADOC/STAT(以下 padoc) はコマンドベースとデータフローベースでのデータ分析環境がある。左図がコマンドベースで右図がコマンドをデータフローに置き換えたものである。本章はコマンドベースの操作環境について説明する。(データフローベースは後述の予定)

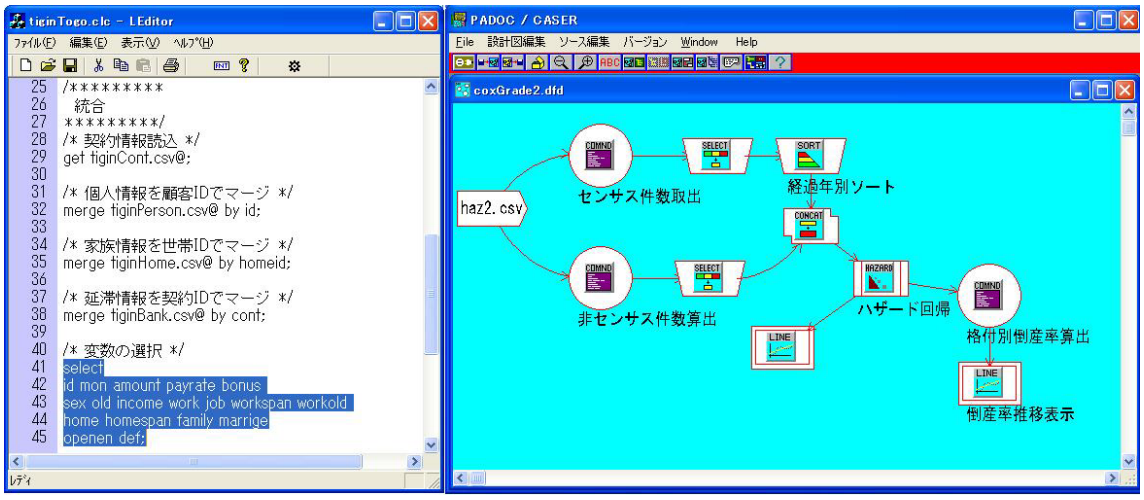


図 1.1　Left:command mode Right:graphical mode

padoc を実行すると次のメイン・コントロール画面 1.2 が表示される。コマンドベースでは矢印のアイコンを押下してコマンド編集画面とその隣のログ画面を表示して、コマンド編集画面でコマンドを記述して操作を行う。

図 1.2　Push Editor buttom in Main Contorl panel

メイン・コントロール画面で編集アイコンを押下して表示されたコマンド編集画面

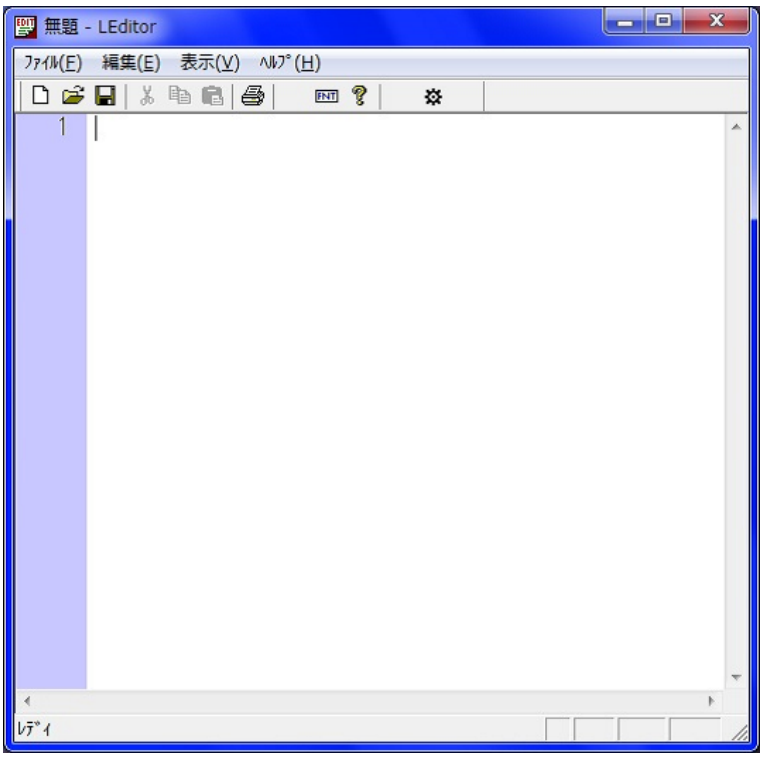


図 1.3　Command Editor panel

またメイン・画面の矢印の隣のアイコンを押下して、ログ画面を表示する。これはコマンド実行結果を表示する画面である。

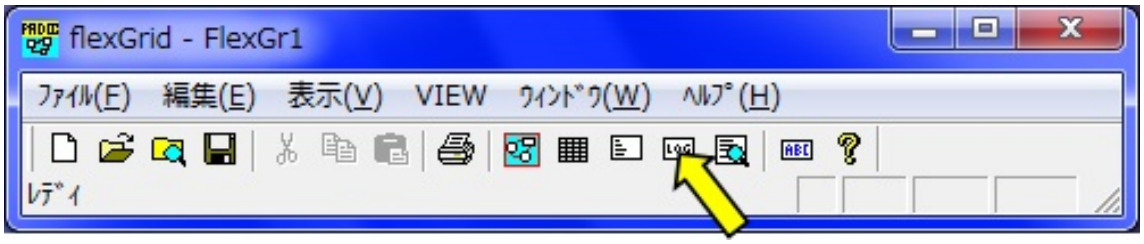


図 1.4　Push Loger buttom in Main Contorl panel

メイン・コントロール画面でログ・アイコンを押下して表示されたログ画面コマンド実行の結果が正常の場合は、実行したログ情報が表示され、エラーの場合は赤字でエラー内容表示される。使用者はこのログ状態を見て実行結果を確かめながら分析を行うことができる。

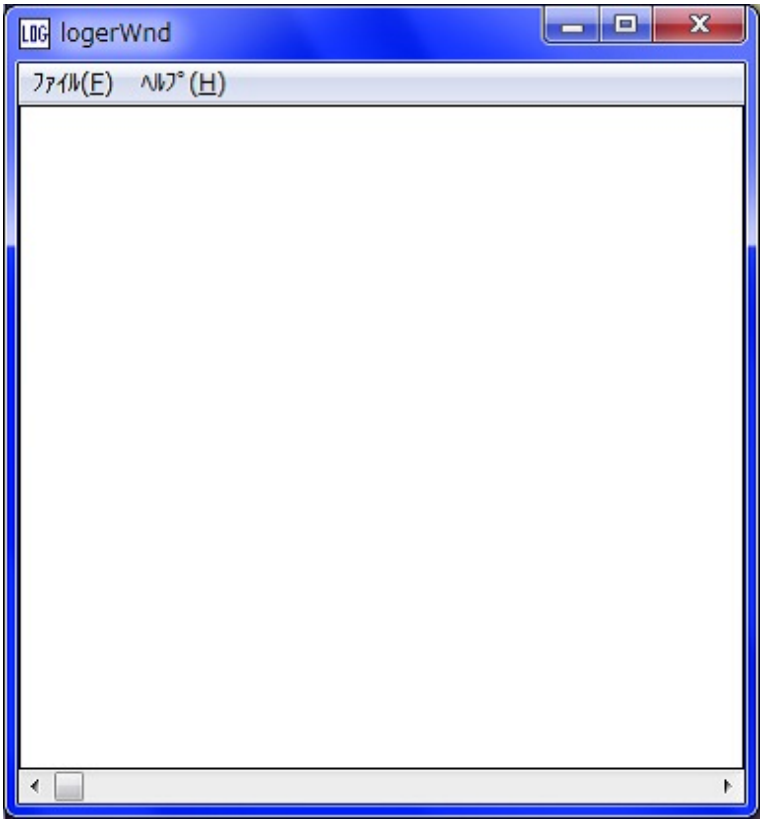


図 1.5　Loger panel

分析作業開始時には次の様な環境になっており、編集画面にコマンドを記述して、コマンド群を実行して分析を行う。

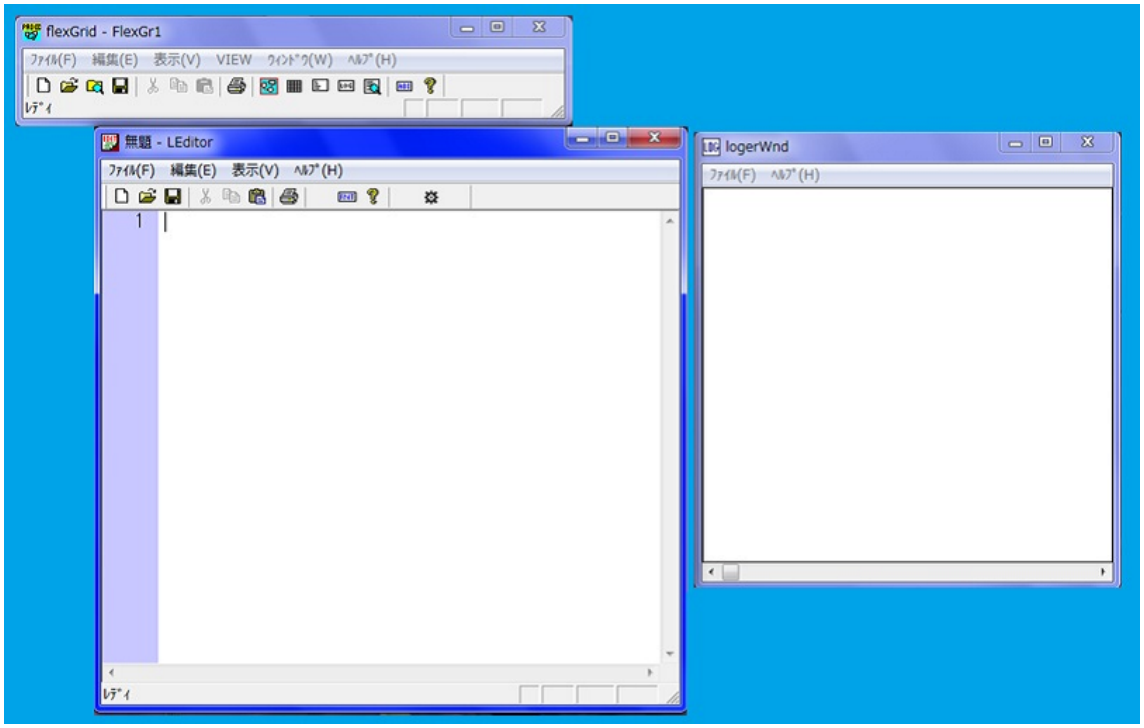


図 1.6　Padoc Anaysis Enviroment at start

まず編集画面に次の様にコマンドを記述してみる。この例は米国の銀行員 475 人について有色人種 (minority=1) と非有色人種とで給与に差があるか見る分析である。

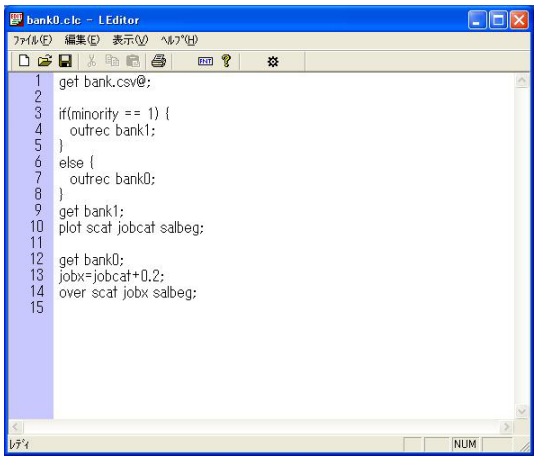


図 1.7　Discription about analysis for minority salary

- 1 行目　　米国銀行員のデータの取込み
- 3-5 行目　有色人種なら bank1 に書出し
- 6-8 行目　それ以外なら bank0 に書出し
- 9 行目　　有色人種のデータの取込み
- 10 行目　職種と初任給の散布図表示
- 12 行目　非有色人種のデータの取込み
- 13 行目　職種コードに 0.2 を加算 (重ね図で散布点が重ならない様にする)
- 14 行目　有色人種と非有色人種の散布図の重ね合せ図の表示

コマンド記述を実行するには、実行部分をマウスで選択（黒反転する）して下図の様に実行アイコンを押下する。

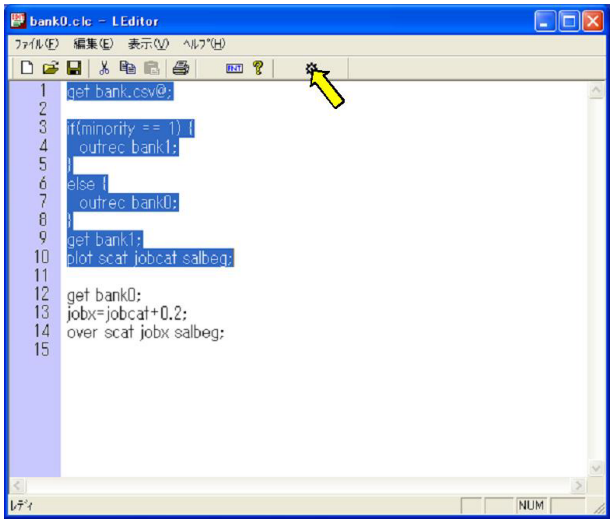


図 1.8　Execution about analysis for minority salary

実行するとログ画面に結果が表示される。エラーがあればエラーの内容が赤表示される。

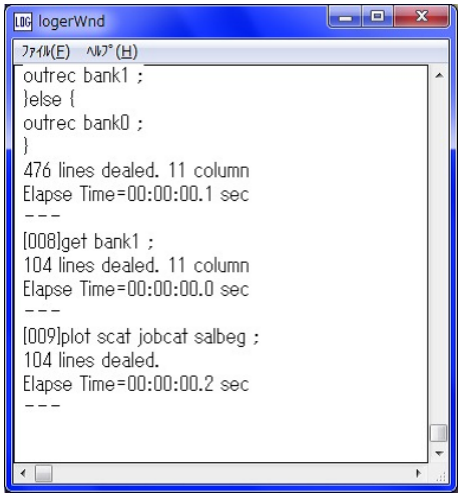


図 1.9　Output of Log about execution analysis

コマンドの手続きはデータ・テーブルを読込み、これをカレントデータとする。データの編集、分析、グラフ表示は常にカレントデータに対して行われる。図 1.10 に示す様に黄色の矩形がカレント・データである。一般に使用者はコマンドを実行する度にログとカレント・データを確認する必要がある。カレントデータは 1.7 の操作で状態を見る事ができる。

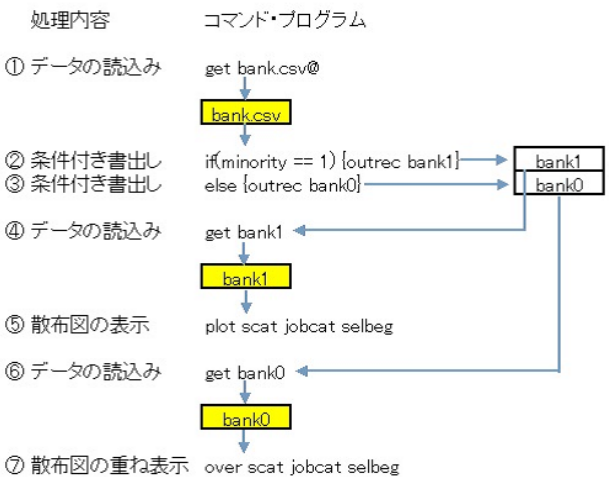
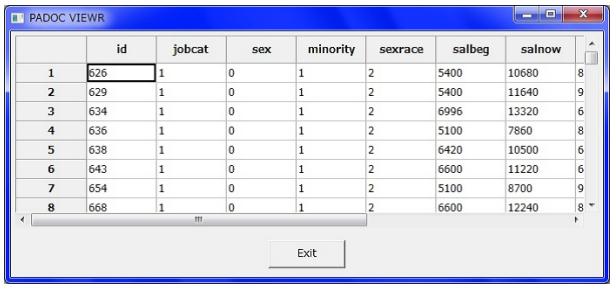


図 1.10　Relation between command process and current data

図 1.7 の 10 行目まで実行した状態で、カレント・データを表示するにはメイン・コントロールのグリッドアイコンを押下する。

図 1.11　Push buttom for showing concurrent data

カレントデータは [get] コマンドで 9 行目で (minority==1) で書出した有色人種のみのデータ・テーブルを呼出した状態となっている。



	id	jobcat	sex	minority	sexrace	salbeg	salnow	
1	626	1	0	1	2	5400	10680	8
2	629	1	0	1	2	5400	11640	9
3	634	1	0	1	2	6996	13320	6
4	636	1	0	1	2	5100	7860	8
5	638	1	0	1	2	6420	10500	6
6	643	1	0	1	2	6600	11220	6
7	654	1	0	1	2	5100	8700	9
8	668	1	0	1	2	6600	12240	8

図 1.12　View concurrent data

図 1.7 を全て実行すると図 1.13 のグラフが表示される。グラフは有色人種は赤点、白人は緑十字で示され、横は職種コードで縦は初任給の撒布図である。これによると有色人種は給与が高い職種に殆ど就いてないことが分る。有色人種と非有色人種とに給与に差があるのではなく、有色人種は高い地位につけないので給与に差が出ている様に見えている。

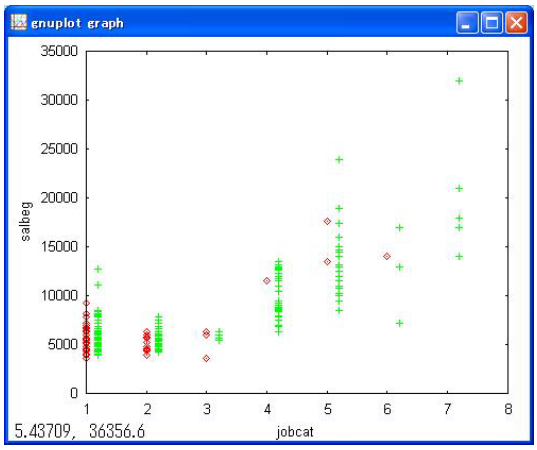


図 1.13　Output of Log about execution analysis

ここの例では職種コードは次の通りである

1. stuff　スタッフ職
2. trainning　訓練中
3. gardman　警備員
4. special　特別職
5. manager　管理職
6. engineer　技術職
7. MBA　MBA 資格保持者

1.2　コマンドベースの操作

コマンドベースの概要に示した様に、メイン・コントロール画面と編集画面を使ってデータ分析を行う。

1.2.1　メイン・コントロール画面

以下にメイン・コントロール画面の各アイコンについて説明を行う。

(a) ホームデイレクトリィの設定

データ編集や分析用のフォーム・ディレクトリィを指定する。一連の作業で作成したデータやプログラムはここに一体で保存されるので便利である。

ホーム・ディレクトリィの指定

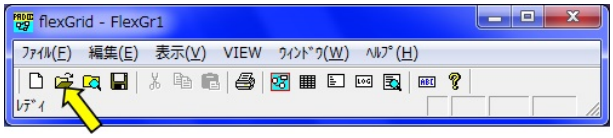


図 1.14　Push home directory icon

ホーム・ディレクトリィを指定する。編集したデータやプログラムは全てここに保存される。

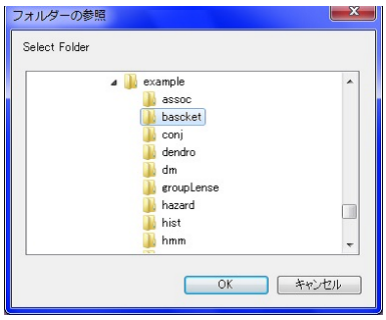


図 1.15　Appoint home directory

(b) データ・テーブル一覧表示

分析途中では多くのデータ・テーブルが生成されるので、この一覧表示機能がある。

図 1.16　Push viwing directory icon

一覧表示ボタン押下後次の選択表示がされる。

- HOME　ホーム・デイレクトリィ
- WORK　一時保存・ディレクトリィ（padoc が終了すると削除される）
- プロジェクト名 1.17 の例では bank や wcalc を示す。　これは次のコマンドで設定されたディレクトリィである。

```
xxxx     prj xxxx ''D:\user\work\data'';  コマンド prj で指定されたディレクトリィ
```

デイレクトリィ選択画面

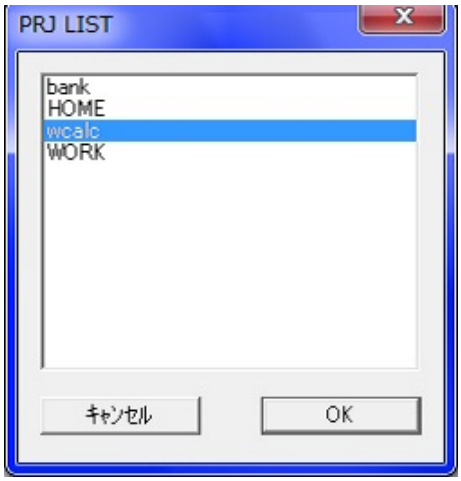


図 1.17　Push viwing directory icon

ディレクトリィを選択すると、全テーブルのヘッダーが表示される。

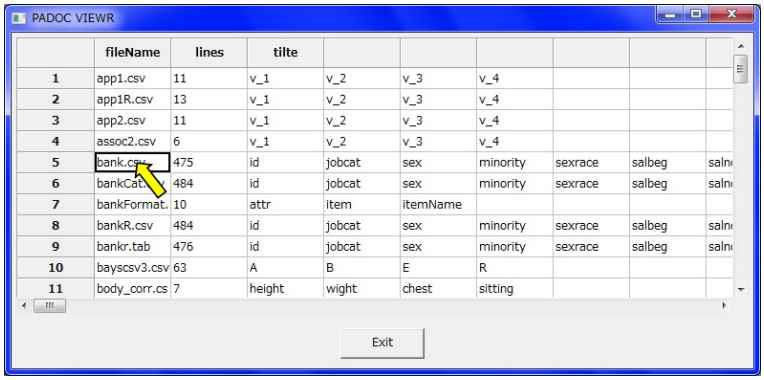


図 1.18　Push viwing directory icon

全テーブルの一覧でファイル名を指定すると、そのテーブルが表示される。

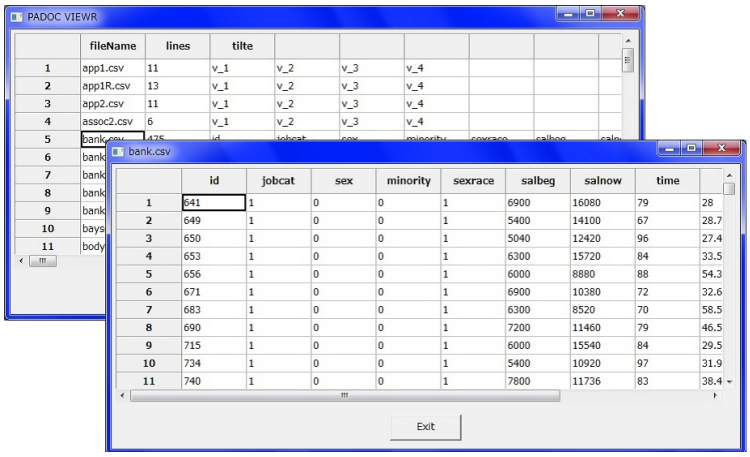


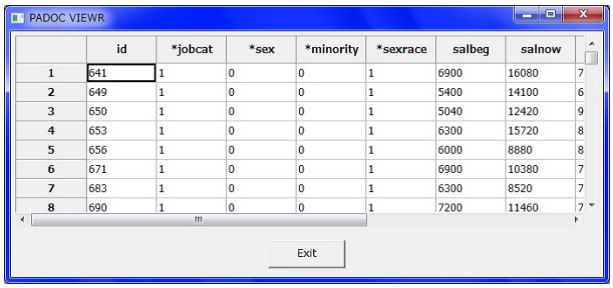
図 1.19　Push viwing directory icon

(c) カレントデータ表示

コマンドベースの操作環境の概要にも述べたが、padoc ではカレントデータに対して編集や分析が行われる。カレントデータの確認には図 1.20 のアイコンを押下する。

図 1.20　Push buttom for showing concurrent data

図 1.39 での変数名の先頭に (*) が付いているものは整数ではなくコードであることを示している。



	id	*jobcat	*sex	*minority	*sexrace	salbeg	salnow	
1	641	1	0	0	1	6900	16080	7
2	649	1	0	0	1	5400	14100	6
3	650	1	0	0	1	5040	12420	9
4	653	1	0	0	1	6300	15720	8
5	656	1	0	0	1	6000	8880	8
6	671	1	0	0	1	6900	10380	7
7	683	1	0	0	1	6300	8520	7
8	690	1	0	0	1	7200	11460	7

図 1.21　View concurrent data

(d) 分析結果の表示

分析コマンドを実行した結果を見たい場合は、このアイコンを押下する。

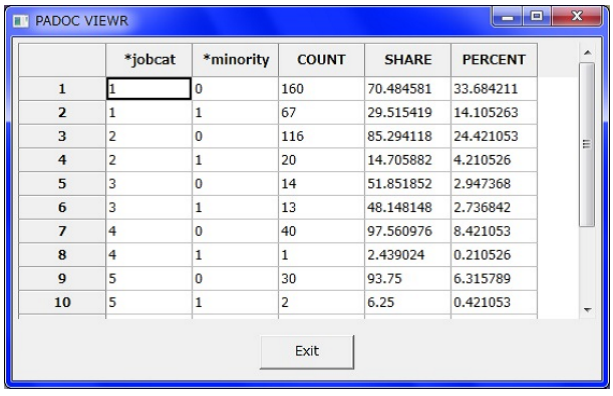
例えば 1.1 章について人種別に (minority) に職種 (jobcat) が偏っているか見たい場合編集画面で以下の様に記述できる。

```
get bank.csv@;              //米国銀行員のデータの読込み
attr name type/             //職種と人種のデータを数字型からコード型に変換する
  jobcat=code
  minority=code
:
put bankR.csv@;             //コード型にした結果を bankR.csv に保存
count jobcat minority;      //職種と人種コード別の件数
```

この結果を表示するにはコントロール画面の分析表示アイコンを押下すると図**??**が表示される。

図 1.22　Push buttom for showing concurrent data

分析結果から人種別 (minority) に見て職種 (jobcat) の比率が異なることが分る。jobcat=3 のガードマンは半々だが jobcat=5 のマネージャは 3% 未満しか有色人種がいない。



	*jobcat	*minority	COUNT	SHARE	PERCENT
1	1	0	160	70.484581	33.684211
2	1	1	67	29.515419	14.105263
3	2	0	116	85.294118	24.421053
4	2	1	20	14.705882	4.210526
5	3	0	14	51.851852	2.947368
6	3	1	13	48.148148	2.736842
7	4	0	40	97.560976	8.421053
8	4	1	1	2.439024	0.210526
9	5	0	30	93.75	6.315789
10	5	1	2	6.25	0.421053

図 1.23　View concurrent data

(e) ヘルプ画面の呼び出し

コマンドベースではコントロール画面にヘルプアイコンがある。

図 1.24　Push buttom for help in comman mode

ヘルプアイコンを押下すると Web 形式のヘルプ画面が表示される。画面内のメニューを押下すると詳細なヘルプ画面が得られる。

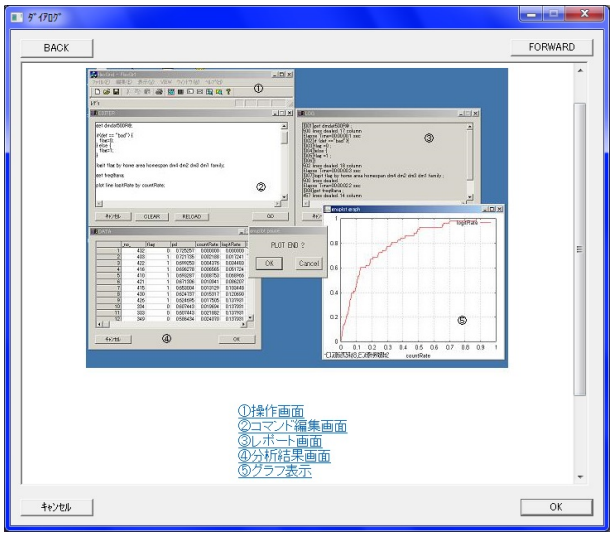


図 1.25　View help web page

1.3 編集画面

メイン・コントロール画面より編集画面アイコンを押下すると図 1.26 の編集画面が表示される。コマンド記述は編集画面で直接記述できるが、既に保存したコマンド記述ファイル（拡張子 *.clc) を呼出して修正することもできる。

1.3.1 プログラムの呼出

編集画面のコマンド記述やデータの読込み書出し先はホーム・ディレクトリィとなっており、これは図 1.15 に指定した通りである。

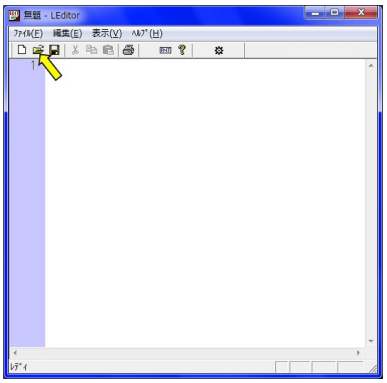


図 1.26 Push to read command script file

ホーム・ディレクトリィ内の保存されたコマンド記述の一覧が表示される。（上部のファイルの場所の指定でディレクトリィの移動も可能である）

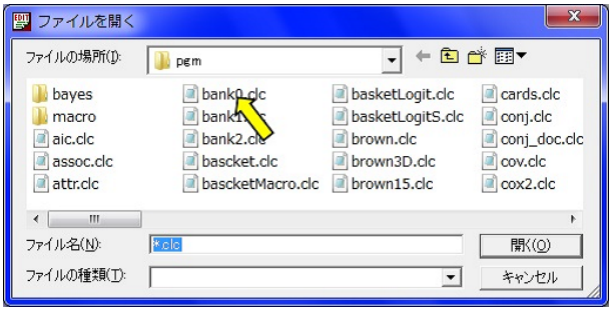


図 1.27 View command script file list in home dirctory

コマンド記述ファイルを指定するとコマンド記述が表示される。

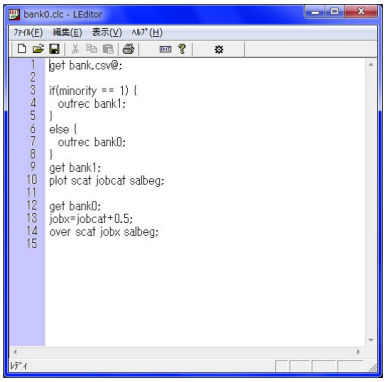


図 1.28　View command script selected from the list

1.3.2　プログラムの保存

修正したコマンド記述を保存するには編集画面の保存アイコンを押下して、ファイル名記入枠にファイル名を指定して、保存を押下する。

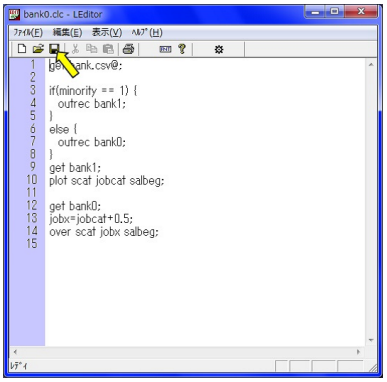


図 1.29　Push to save command script file

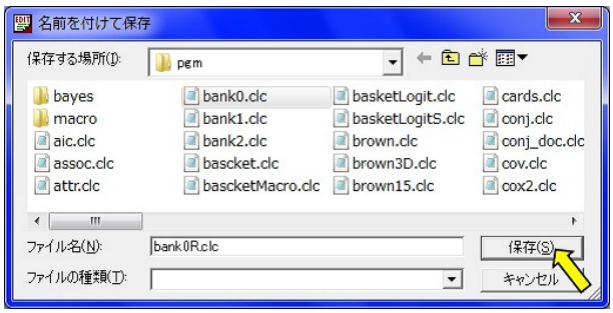


図 1.30　View save to save command script file

1.4　コマンド記述仕様

1.4.1　コマンド記述の概要

コマンド記述には以下の種類がある。

- 入出力記述：get put plot など
- 分析記述：logit reg count など
- 編集記述：データ・テーブルの全行に一律適用されるデータの加工、追加、削除等の命令

編集記述はデータの加工や編集を記述するもので、データの項目名を変数として使い、各行に同じ編集記述が適用される。（但し前行の値 (lag 関数) を使う様な行間での計算は別途関数が用意されている）一行毎に適用される編集記述の仕様はC言語仕様に近いものとなっている。即ち条件分岐や繰返し処理の記述は同じである。

例えば図 1.31 では 2〜3 行目の編集記述が全行に適用されるので minority==1 である行が全て bank1 に書出され、minority==0 の全行が bank0 に書出される。

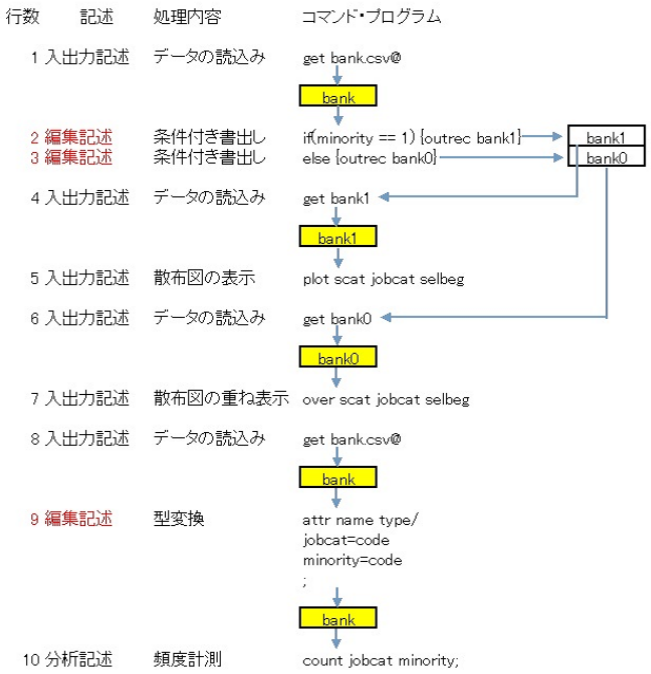


図 1.31　View kind of discription

1.4.2　入出力

padoc では入力データは基本的にはテーブル形式のデータのみ扱う。しかし次の例外がある。

- hand(手入力) コマンドで 編集画面に手入力で直接記述する

```
hand  *c1 c2 c3  c4 stat/    //c1 c2 c3 stat は変数名
1 32 31 28  good
2 21 18 25  even
3 13 12 22    bad
;
```

(注 1)hand の引数で (*) が前についている場合、数値であってもコード型と解釈される。

- 形態要素解析で指定のディレクトリィ配下*.txt を全て処理する。

```
mecab news\*.txt@;  //news デイレクトリィ配下の*.txt が対象
```

出力データは次節の保存形式で出力される。

(a) 保存形式

データ・テーブルの保存形式は次の csv とタブ形式の 2 種類がある (@は次節の保存場所を示す)。

- CSV 形式 拡張子は.csv

```
get banker.csv@;
put result.csv@;
```

- タブ区切り　拡張子は不要

```
get banker@;
put result@;
```

(b) 保存場所

データ・テーブルの保存場所には以下の 3 箇所がある。

- 一時作業領域　padoc が終了すると削除される

```
put temp1;  //一時作業領域に temp1 として書出し
get temp1;  //一時作業領域から temp1 を読込み
```

- コマンド prj で指定されたディレクトリィ

```
prj xxxx ''D:\user\work\data''; //コマンド prj で指定されたディレクトリィ
get banker.csv@xxxx;  //xxxx ディレクトリィ内の banker.csv の読込み
put result.csv@xxxx;   //xxxx ディレクトリィ内の result.csv として書出し
```

- ホーム・ディレクトリィ　最後に「@」で終わっており「@」より後を省略できる

```
get banker.csv@;   //ホーム・ディレクトリィから banker.csv を読込み
put result.csv@;    //ホーム・ディレクトリィに result.csv を書出し
```

(c)CSV ファイルの読込み

padoc ではデータレコードから始まる新規の csv ファイルを読込む場合、自動的に以下の事をする。

- 図 1.32 の様に警告が出て dummy の変数が図 1.33 の様に生成される
- 各カラムの属性を判断して数値か文字かを判断する。
  しかし数値がコードである場合は判断できないので「attr」コマンドで修正する必要がる

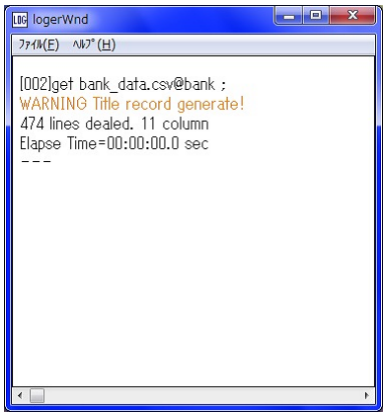


図 1.32　View warning message on generation dummy title

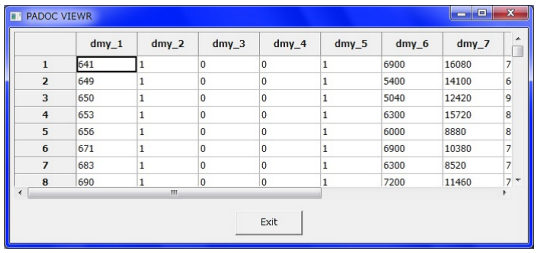


図 1.33　View readed csv file with dummy title

もしタイトル行が 1 行目にあれば、以下の opiton 記述でタイトル行を読むことができる。

```
get bank_title.csv@/
  titleline=1
;
```

例えばタイトルが無く、タイトルが dummy で生成された場合「attr」コマンドを使って全変数名と型を変更することができる。

```
prj bank ‘‘c:\temp\download’’;    //ライブラリィの定義
 get bank_data.csv@bank;          //csv の読込
 attr name rename type/           //変数名の変更と属性の設定
  dmy_1 no code
  dmy_2 jobcat code
  dmy_3 gender code
  dmy_4 minority code
  dmy_5 sexrace code
  dmy_6 selbeg num
  dmy_7 selnow num
  dmy_8 time    num
  dmy_9 age     num
  dmy_10 edlevel num
  dmy_11 work num
 :
```

「attr」コマンドの実行によってカレント・テーブルは図 1.34 の様に変更される。

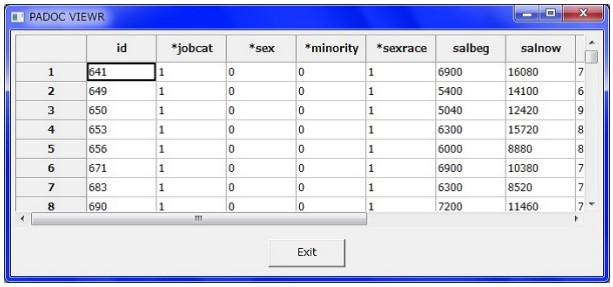


図 1.34　View concurrent data

(d) データ・テーブルの変数名

padoc が出力したデータ・テーブルを再読込みした場合、テーブルのデータ項目が変数名となっている。次の様に padoc の計算によって新たな変数が作られる場合は、図 1.35 に示す様に新たなデータ項目としてデータ・テーブルに反映される。

```
salrate=(salnow - salbeg)/work;
```

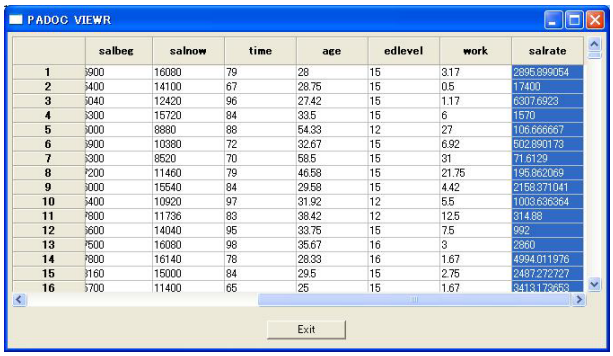


図 1.35　Added new calculated variable

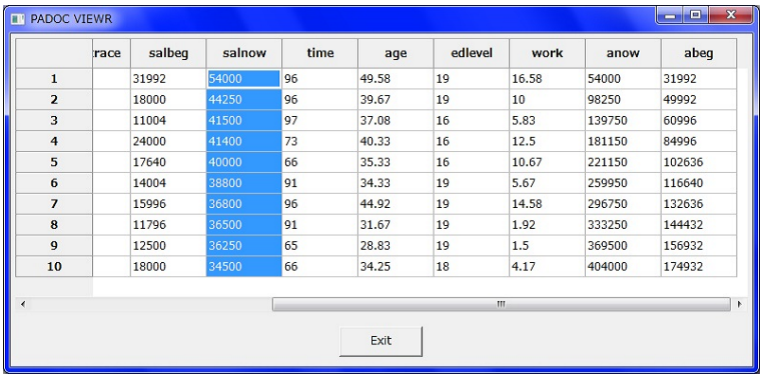
その他に次に示す変数名がある。

- #： 行番号を示す
- LAST： 最後の行番号を示す
- accume： 蓄積宣言された変数
- vector： 配列宣言された変数　配列の項参照

蓄積宣言は現在給与 (salnow) が高い 100 人について現在給与と初任給 (salbeg) の合計を求めたい場合は以下の記述になる

```
get bank.csv@;                  //データ・テーブルの読込み
sort -salnow;                    //現在給与の逆順のソート
if( # <= 100)  outrec;          //100 行目までカレントに書込み

accume anow abeg        //蓄積データの宣言
anow= salnow;               //データの蓄積
abeg = salbeg;

if(# == LAST) outrec;    //最後の行のみカレントに書込み
```

図 1.36 では salnow が降順にソートされ値が蓄積（右端) されていることが分る。



	race	salbeg	salnow	time	age	edlevel	work	anow	abeg
1		31992	54000	96	49.58	19	16.58	54000	31992
2		18000	44250	96	39.67	19	10	98250	49992
3		11004	41500	97	37.08	16	5.83	139750	60996
4		24000	41400	73	40.33	16	12.5	181150	84996
5		17640	40000	66	35.33	16	10.67	221150	102636
6		14004	38800	91	34.33	19	5.67	259950	116640
7		15996	36800	96	44.92	19	14.58	296750	132636
8		11796	36500	91	31.67	19	1.92	333250	144432
9		12500	36250	65	28.83	19	1.5	369500	156932
10		18000	34500	66	34.25	18	4.17	404000	174932

図 1.36　Reverse sort and accume data

(e) データの型

padoc では次のデータ型に区別する。

- 数値：　padoc では整数と実数には区別がない
- コード：　識別字　例えば商品コードである
- 文字 ： 漢字やアルファベットが入っているデータ

例えば図 1.37 の様な jobcat や minority は数値かコードか判断できない。

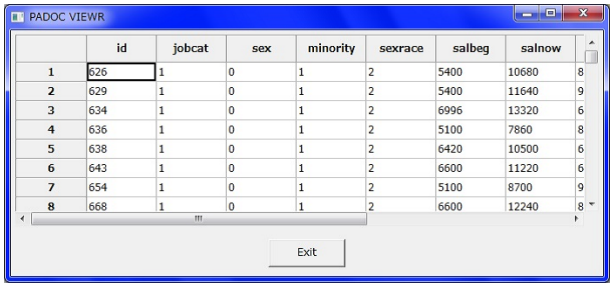


図 1.37　View concurrent data

例えば jobcat を数値のままで使用すると、図 1.38 の左図の様に jobcat 毎の件数を count コマンドで数えると次の様に数値の区間で区切られて出力される。一方コードと指定して数えると右図の様に簡潔な表示が得られる。

```
get bank.csv@; //行員データテーブルの読込み
//数値のまま件数計算
count jobcat;
//jobcat はコードと指定して件数計算
count jobcat/
  jobcat=code
;
```

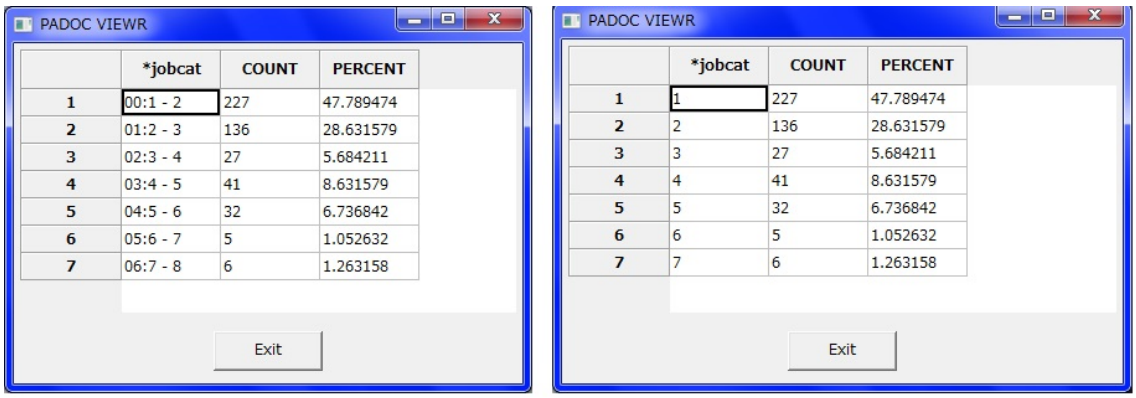


図 1.38　comparison cunt output by numeric or code

型を変更するには次の編集記述 (attr) の option を使う。padoc では option の設定は全て [/ ;] で囲んだ範囲となる。

```
attr name type/
jobcat=code
minority=code
;
```

型がコードなら、カレントデータを表示すると変数名の先頭に (*) が付いている。

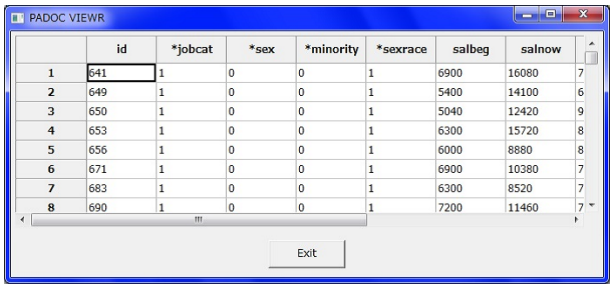


図 1.39 View concurrent data

(f) データ・テーブル読込コマンド「get」のオプション

「get」コマンドでは次のオプションがある。

- titleline ： もしカラム名の設定行があるならその行番号を指定する
- first： データの始まりの行番号を指定
- separetar： 次の区切り指定 comma/space/tab/bar
- maxline： 読込むデータ行の最大数

図 1.40 の様なヘッダー付きで区切が「|」の場合は次の様に読込む

```
prj bank "c:\temp\download\bank";
get bank_deco.txt@bank/      //9 行目のタイトル　10 行目からデータ　区切は「|」
  titleline=9
  first=10
  seperator=bar
;
```

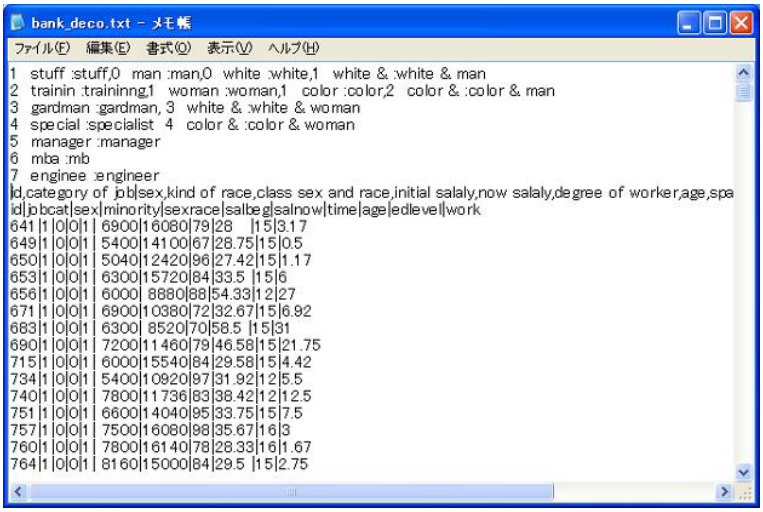


図 1.40　Data with header comment and separated by bar character

1.4.3 データ・テーブルの編集

編集記述はデータ・テーブルの加工や編集を記述するもので、データ・テーブルの項目名を変数として使い、各行に同じ編集記述が適用される。(但し前行の値 (lag 関数) を使う様な行間での計算は別途関数が用意されている) 一行毎に適用される編集記述の仕様はC言語仕様に近いものとなっている。即ち条件分岐や繰返し処理の記述は同じである。

例えばアンケートの 526 人の回答として、一行に 20 個の回答がある。もし無回答 (欠損) が 10 個以上あればそのデータは削除する編集記述は以下となる。

```
get dmail@;  //アンケート読込み 変数は id out_1 out_2 .... out_20
/* 編集記述開始 */
vector dm[20]=out_1 - 20;  //out_1～out_20 を配列 dm[1]～dm[20] に入れる
lack=0;
for(i=1;i<=20;i++) {
  if(dm[i] == ?) lack++;  //欠損の場合カウントする
}
if(lack >= 10) delrec;  //欠損が 10 個以上なら削除する
/* 編集記述終了 */
put dmailR@;  //修正結果を書出し
```

(注) 編集記述の 1 行目で変数名で数字が連続する場合 `out_1 - 20` の様に簡略な指定が可能になっている。
この例では `verbctor dm[20] = out_1-20;` は配列宣言で項目 `out_1` から `out_20` までを配列 dm[20] に入れることを示している。
上記の編集記述を実行した結果の変数 lack(右端) を見ると何れも 10 個未満になっていることが分る。

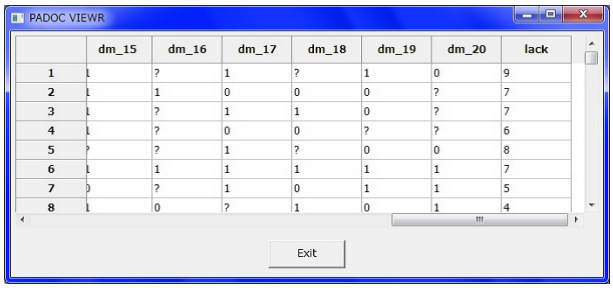


図 1.41 Result data modified by command description

(a) 計算

データ・テーブルの変数（項目) 間で計算が行え、この計算は全行に適用される。新規の変数の場合、結果は各行の最後に追加される。

例えば初任給 (salbeg) と現在給与 (salnow) を勤続年数 (work) で割った昇給率 (salrae) は以下となる。計算された変数は図 1.42 の様にテーブルの最後に追加される。

```
salrate=(salnow - salbeg)/work;
```

但し、何れかの変数が欠損もしくは計算不能な場合欠損になる（上記の場合は $work = 0$ でゼロ割の場合)

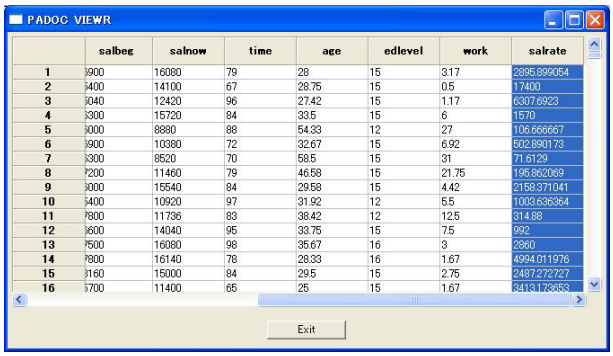


図 1.42　Added new calculated variable

演算子には表 1.1 に示すものが使える。

表 1.1　operator in usage of padoc

operater	description	usage	constraint
+	plus	a = b + c	
-	minus	x = y - c	
*	multify	c = d * e	
/	divied	a = b / c	
%	modulo	a = a % d	
**	power	a = b ** c	
++	incriment	n ++	integer only
–	decriment	n –	integer only
!	factorical	a = n!	intenger only

関数としては 1.2 が使用可能である。

表 1.2 Function spec in usage of padoc

kind	function	description	usage	argument
numeric	log	logarithm	a = log(b)	positive only
numeric	log10	base 10 log	a = log10(b)	positive only
numeric	exp	exponential	a = exp(b)	
numeric	sqrt	squre root	a = sqrt(b)	positive
numeric	cos	cosine	a = cos(b)	radian value
numeric	sin	sine	a = sin(b)	radian value
numeric	tan	tangent	a = tan(b)	radian
numeric	acos	arc cosine	a = hcos(real)	radian value
numeric	asin	arc sine	a = hsin(real)	radian value
numeric	atan	arc tangent	a = htan(real)	radian
numeric	abs	absolute	a = abs(b)	
numeric	round	round	a = round(a,n)	n is shift no
numeric	sign	sign	a = sign	
numeric	determn	determinant	determin(table)	filename
transform	ifx	integerize	a = ifix(b)	real value
trnasform	rad	radians	a = rad(b)	degree value
transform	hex	hexian	a = hex(b)	hexian value
string	num2str	no to string	a = num2str(b)	
string	str2num	string to no	a = str2num(“abc”)	
string	strsel	select character	a = strsel(“abcd”,1,2)	
string	stradd	add string	a = stradd(“abc”,”xyz”)	
date	undate	trans jurian date	a = undate(20180517)	
date	dater	trans date	a =dater(jurian)	
random	random	uniform random	a=random	no argument
random	ranstd	normal random	a=ranstd(ave,std)	
random	ranpoi	poison random	a=ranpoi(ramda)	
random	mrand	multi random	a=mrand(table)	filename
random	qnorm	accume std density	p=qnorm(real)	
random	pnorm	inverse accume std	x=pnorm(prob)	
else	prev	get previous value	a=prev(var)	
else	outlook	out look func	outlook = (table,var,val,var)	filename

(b) 配列

padoc では 1 次元と 2 次元の配列が使える。配列は *vector* 宣言を使う。しかし vector x[3][4] は `x_1, x_2,,x_12` として保存される。

(注)`x_1, x_2,,x_12` は padoc では `x_1-12` と表記できる。

以下の例では 3 × 4 の行列を読込み、図 1.44 の様に 2 次元配列 vec[3][4] として `vec_1-12` に収納している。

```
 clear;  //データの初期化
//行列の読込み
 hand c1 c2 c3 c4 stat/
   21 32 31 28  good
   10 21 18 25  even
    7 13 12 22    bad
 ;
 n=3;   //配列の行数
 m=4;  //配列の列数
 vector v[m]=c1 c2 c3 c4; //配列に変換
 vector vec[n][m];            //2 次元配列の宣言
 accume vec_1-12;           //値の蓄積の宣言

 for(i=1;i<=n;i++) {
    for(j=1;j<=m;j++) {
       if(# == i) vec[i][j]=v[i];  //#は処理中の行番号
       else       vec[i][j]=0;
    }
 }
 select vec_1-12 n m;      //変数の選択
 if(# == LAST) outrec;     //最終行のみ取出し
```

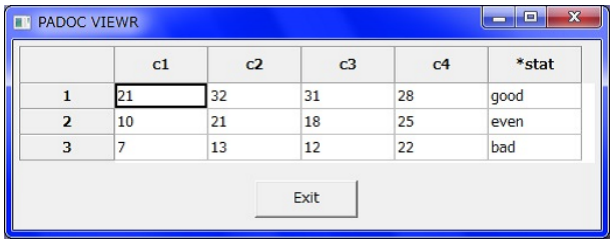


図 1.43　2*3 matrix

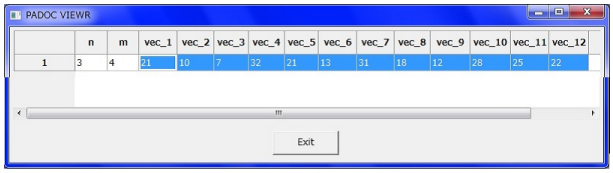
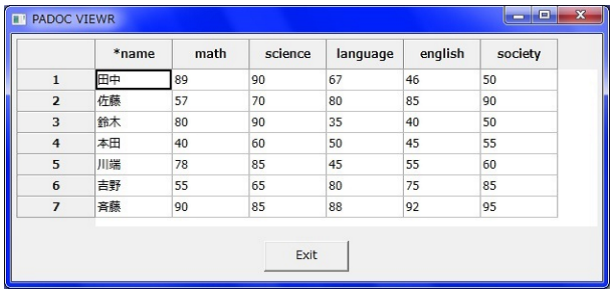


図 1.44　Transform matrix to 2D vector

(c) 制御

padoc では制御としては条件分岐と繰返しがある。図 1.45 では 5 人の 5 教科の点数である。これを格付けすることを考える。これには教科の点数を配列に入れ、教科毎に格付処理を繰返し格付けすればよい。格付けは点数を格付けの閾値で区分を行う。

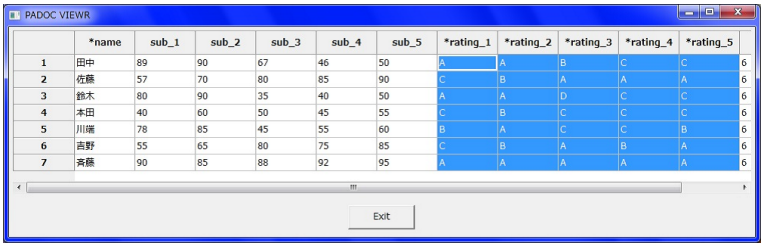


	*name	math	science	language	english	society
1	田中	89	90	67	46	50
2	佐藤	57	70	80	85	90
3	鈴木	80	90	35	40	50
4	本田	40	60	50	45	55
5	川端	78	85	45	55	60
6	吉野	55	65	80	75	85
7	斉藤	90	85	88	92	95

図 1.45　Score of subjects by classmates

```
//成績データ・テーブルの読込み
get factor1.csv@;

//5 教科の点数を配列に入れる
vector sub[5]=math science language english society;
vector rating[5];
for(i=1;i<=5;i++) {  //5 教科について繰返す
   if(sub[i] >= 80) {rating[i]="A";}        //80 点以上なら A 格
   elseif (sub[i] >= 60) {rating[i]="B";}  //60 点以上なら B 格
   elseif (sub[i] >= 40) {rating[i]="C";}  //40 点以上なら C 格
   else {rating[i]="D";}                    //40 点未満なら D 格
}
select rating_1-5;
```

点数により格付した結果は図 1.46 の様になる。



	*name	sub_1	sub_2	sub_3	sub_4	sub_5	*rating_1	*rating_2	*rating_3	*rating_4	*rating_5	
1	田中	89	90	67	46	50	A	A	B	C	C	6
2	佐藤	57	70	80	85	90	C	B	A	A	A	6
3	鈴木	80	90	35	40	50	A	A	D	C	C	6
4	本田	40	60	50	45	55	C	B	C	C	C	6
5	川端	78	85	45	55	60	B	A	C	C	B	6
6	吉野	55	65	80	75	85	C	B	A	B	A	6
7	斉藤	90	85	88	92	95	A	A	A	A	A	6

図 1.46　Rating of subjects by classmates

(d) レコードの分割と生成

(d.1) レコードの分割

padoc では読み込んだデータ・テーブルを変数の値によって分割することができる。次の例は一様乱数でランダムに 4 分割した編集記述である。

```
get bank.csv@;   //データ・テーブル読込
m=4;                //分割数
i=0;
rnd = random;    //一様乱数の発生
if(rnd <= ++i/m) {outrec bank1;}
elseif(rnd <= ++i/m) {outrec bank2;}
elseif(rnd <= ++i/m) {outrec bank3;}
else {outrec bank4;}
```

(d.2) レコードの生成

padoc では繰返し処理内でレコード生成コマンド (outrec) を使ってレコードを生成することができる。
次の例は 3 種類の乱数を振ったレコードを 1000 行生成した例である。
(注意) 繰返し制御でレコードを生成する場合、既にカレント・データが存在するなら、これを削除するため clear コマンドを設定する必要がある。これを忘れると次の例ではカレント・データの 1 行毎に 1000 回行生成することになるので注意が必要である。

```
clear;                    //カレント・データの削除
for(i=1;i<=1000;i++) {    //1000 レコードの生成
  uniform=random;         //一様乱数
   normal=ranstd(0,1);    //正規乱数
   poison=ranpoi(4);      //ポアソン乱数
   outrec;                //レコードの生成
}
plot hist uniform normal poison;  //乱数のヒストグラフ表示
```

図 1.47 は 3 種類の乱数を 1000 行生成した結果のヒストグラムである。

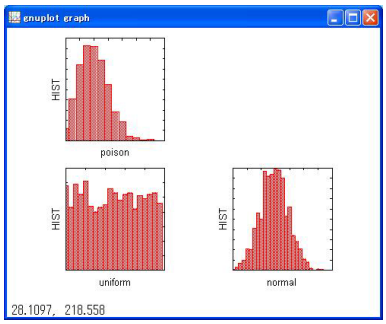


図 1.47　Gereration records for 3 kind of random

(e) レコードのソートと重複の削除

(e.1) ソート

padoc では複数の変数 (項目) についてレコードをソートすることができる。

- 昇順の場合：　変数のみか前にプラス (+) を付ける
- 降順の場合：　変数名の前にマイナス (-) を付ける

変数が文字の場合は文字コードでソートされる。
次の例は職種コードを降順に、初任給を昇順にソートした結果である。

```
get bank.csv@;     //データ・テーブルの読込
sort -jobcat salbeg;  //ソート　職種（jobcat）では降順 初任給では昇順
```

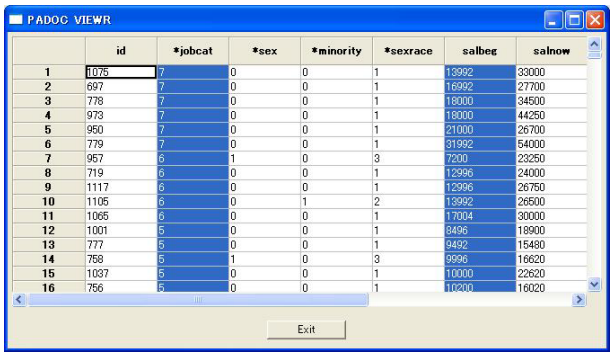
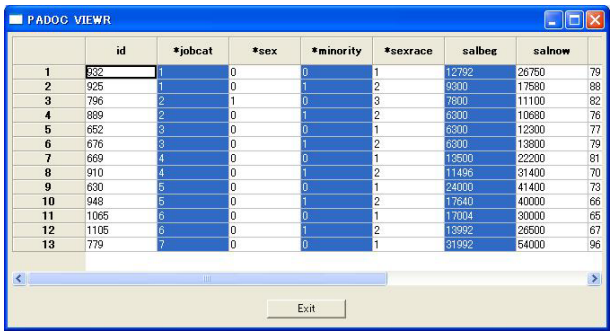


図 1.48　Record sort by multiple variable

(e.2) レコードの重複の削除

padoc では複数の変数がユニークになる様に重複レコードは削除することができる。次の例は職種と人種について初任給を降順にソートして、ユニーク (unique) コマンドより職種と人種が異なるものを残しして最高の初任給を比べている。

```
get bank.csv@;     //データ・テーブルの読込
sort jobcat minority -salbeg;  //ソート　職種では昇順 初任給では降順
unique jobcat minority    //同じ職種コードと人種は削除する

if(minority == 0) outrec bank0;
else              outrec bank1;

get bank0;
rename salbeg=minority_0;
plot bar minority_0 by jobcat ;

get bank1;
rename salbeg=minority_1;
over bar minority_1 by jobcat ;  //重ね図の表示
```



	id	*jobcat	*sex	*minority	*sexrace	salbeg	salnow	
1	932	1	0	0	1	12792	26750	79
2	925	1	0	1	2	9300	17580	88
3	796	2	1	0	3	7800	11100	82
4	889	2	0	1	2	6300	10680	76
5	652	3	0	0	1	6300	12300	77
6	676	3	0	1	2	6300	13800	79
7	669	4	0	0	1	13500	22200	81
8	910	4	0	1	2	11496	31400	70
9	630	5	0	0	1	24000	41400	73
10	948	5	0	1	2	17640	40000	66
11	1065	6	0	0	1	17004	30000	65
12	1105	6	0	1	2	13992	26500	67
13	779	7	0	0	1	31992	54000	96

図 1.49　Unique Record by multiple variable

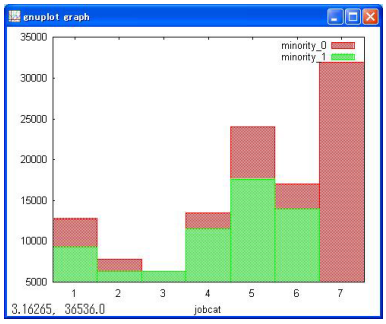


図 1.50　comparison of beginner salary between jobcat and minority

(f) テーブルの合成

padoc では 2 つのデータ・テーブルの合成には以下の 3 方法がある。

- merge： 2 テーブルを横方向に変数値が同じならば横に繋ぐ
- concat： 同じ変数名があれば 2 テーブル縦方向い繋ぐ
- append： 横方向に固定の値を付加える
- outlook： outlook 関数によって他のテーブルの値を検索し、該当行の値を設定する

(f.1) マージ (merge)

2 つのデータ・テーブルを共通する変数 (key 項目) で繋ぐ。次の例は職種コード (jobcat) の職種名のデータ・テーブルと銀行員データ・テーブルを職種コードをキーとしてマージ連結したものである。マージ後の職種名で現在給与のヒストグラムを表示すると職種名によって給与の分布が異なることが分る。

```
hand jobcat jobname/      //職種名を手入力
1 1:stuff
2 2:trainning
3 3:gardman
4 4:special
5 5:maneger
6 6:mba
7 7:engineer
;
put jobid;       //職種名データ・テーブルの生成

get bank.csv@;    //銀行員データ・テーブルの読込
merge jobid_ by jobcat;       //職種コードをキーとして職種名データ・テーブルをマージ

plot hist salnow by jobname;  //職種名で現在給与の分布を表示
```

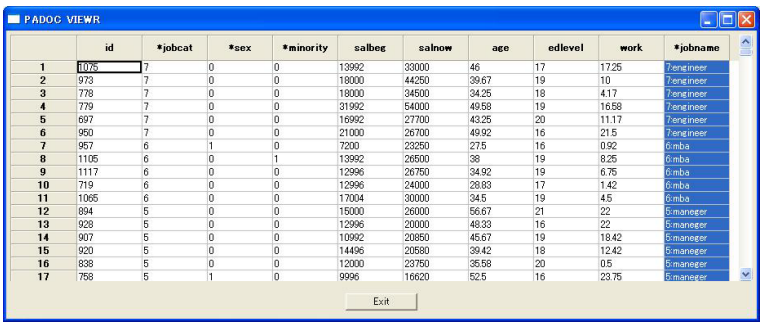


図 1.51　Result of merge for bankdata and jobname by jobcat

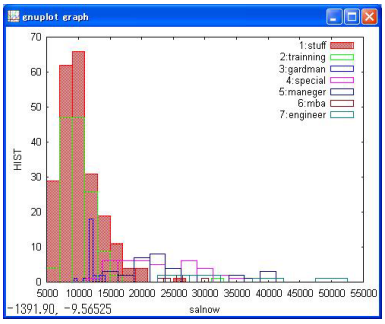


図 1.52　Histgram of salary by jobname

両データ・テーブルの合成キーが一致しない場合、合成後の結果には図 1.53 に示す状態がある。

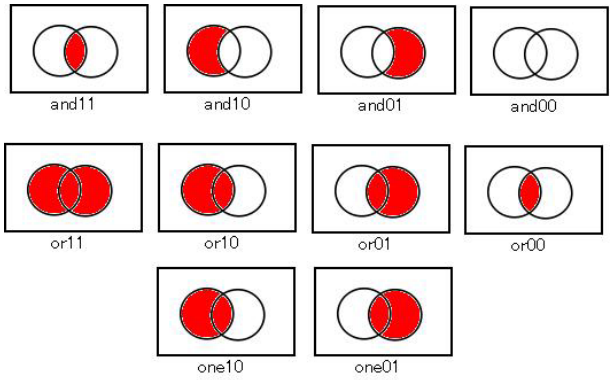


図 1.53　Type of overlapping state

図 1.53 の合成後の状態を指定するには各図の下段にあるコードで設定する。その設定は以下の編集記述の option で指定する。

```
//A のデータテーブル
hand key val_a/
a  1
b  2
c  3
y  4
z  5
;
put A;
//A のデータテーブル
hand key val_b/
x  10
y  20
a  30
c  40
;
put B;

get A;
//option を指定してデータテーブル A の元にデータテーブル B をマージする
merge B by key/
method=one10
;
```

上述の結果は図 1.54 となり、A テーブルの全てのレコードが入っている。

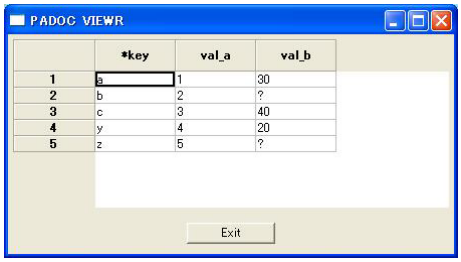


図 1.54　Result of merge option=one10

合成元のテービルのキーを A　合成するテーブルのキーを B とすれば、以下の option 記述で合成の指定を行う。但し $\bar{A}$ 及び $\bar{B}$ は集合 A の補集合、集合 B の補集合を示す。図 1.53 から分る様に or10 と one10　or01 と one01 は同じ意味である。

- and11 : $A \cap B$
- and10 : $A \cap \bar{B}$
- and01 : $\bar{A} \cap B$
- and00 : $\bar{A} \cap \bar{B}$

- or11：$A \cup B$
- or10：$A \cup \bar{B}$
- or01：$\bar{A} \cup B$
- or00：$\bar{A} \cup \bar{B}$
- one10：$A \cup \bar{B}$
- one01：$\bar{A} \cup B$

(f.2) 縦連結 (concat)

concat は 2 つのデータ・テーブルを縦に連結する。この場合は図 1.55 に示す様に、同じ項目名があると一列に連結され、同じ項目が存在しない場合は新たな項目名として連結される。

```
clear;  //5 行の 2 正規乱数とポアソン乱数データ生成
for(i=1;i<=5;i++) {
   //uniform=random;
   normal=ranstd(0,1);
   poison=ranpoi(4);
   outrec;
}
put rand5;

clear;  //10 行の正規乱数と一様乱数データ生成
for(i=1;i<=10;i++) {
   uniform=random;
   normal=ranstd(0,1);
   //poison=ranpoi(4);
   outrec;
}
put rand10;

get rand5;
concat rand10;  //10 行の乱数データを縦連結する
```

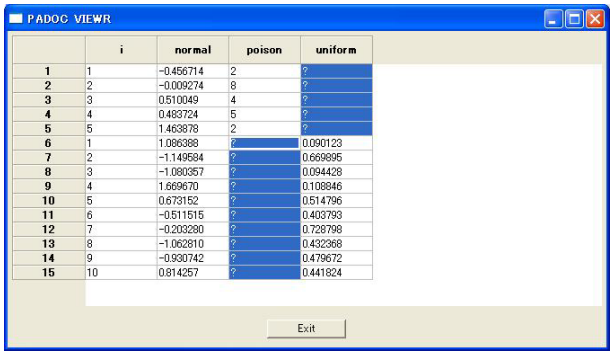


図 1.55　Result of concat of 2 random data

(f.3) 固定項目の追加

解析の結果を全行に反映し対場合がある。例えば次の例は、現在給与を行員の属性で回帰して回帰係数を求めている。この回帰係数で回帰値を計算して、実際の目的変数と比較したい場合がある。次の例では回帰係数を append で行員データ・テーブルに全行に反映して、回帰係数と説明変数値を掛けて全行で回帰値を計算している。最後に実際の値と比較している。

```
get bankR.csv@;  //行員データ・テーブルの読込
reg salnow by minority jobcat gender age work edlevel;  //現在給与を回帰

get freq@ana;    //回帰結果の取込
select beta0-6; //回帰係数の選択
put regp;    //回帰係数をテーブルに書出し

get bankR.csv@;  //行員データ・テーブルの再読込
append regp;     //回帰係数テーブルを行員データ・テーブルに追加
reg=beta0        //線形和で回帰予測値を計算
    +beta1*minority
    +beta2*jobcat
    +beta3*gender
    +beta4*age
    +beta5*work
    +beta6*edlevel;
plot line salnow reg; //回帰予測値と実値との比較
```

図 1.56 は回帰係数を append で全行に追加した結果

	age	edlevel	work	beta0	beta1	beta2	beta3	beta4	beta5	beta6
1	28	15	3.17	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
2	28.75	15	0.5	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
3	27.42	15	1.17	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
4	33.5	15	6	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
5	54.33	12	27	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
6	32.67	15	6.92	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
7	58.5	15	31	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
8	46.58	15	21.75	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
9	29.58	15	4.42	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
10	31.92	12	5.5	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
11	38.42	12	12.5	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
12	33.75	15	7.5	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
13	35.67	16	3	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
14	28.33	16	1.67	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
15	29.5	15	2.75	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333
16	25	15	1.67	-1778.042513	-790.451971	2665.9578	-2737.900564	69.993334	-115.1873	726.006333

図 1.56　Result of append for fix variable of regression coeficent

図 1.57 は回帰値と実際の値との比較した図

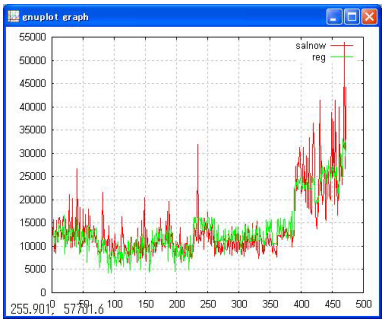


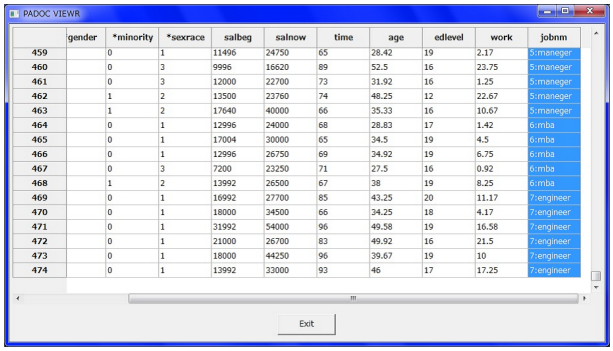
図 1.57　comparison salary and regression value

(f4)outlook 関数による検索値の反映

これは Excel の [lookup] 関数と同じ機能で他のファイルの値を検索し、該当行の値を設定する方法である。

```
hand jobcode jobname/      //職種名を手入力
1 1:stuff
2 2:trainning
3 3:gardman
4 4:special
5 5:maneger
6 6:mba
7 7:engineer
;
put jobid;        //職種名データ・テーブルの生成

get bankR.csv@; //銀行員テーブルの読込み
//(f4)jobcode が銀行員の jobcat に一致した jobname を銀行員テーブルに設定する
//第 1 引数：検索テーブル名
//第 2 引数：検索項目
//第 3 引数：反映先テーブルの検索値
//第 4 引数：一致行での反映項目
jobnm=outlook(jobid,''jobcode'',jobcat,''jobname'');
```

outlook 関数で jobcode=jobcat で一致した場合での職種名が反映されている。



	gender	*minority	*sexrace	salbeg	salnow	time	age	edlevel	work	jobnm
459		0	1	11496	24750	65	28.42	19	2.17	5:maneger
460		0	3	9996	16620	89	52.5	16	23.75	5:maneger
461		0	3	12000	22700	73	31.92	16	1.25	5:maneger
462		1	2	13500	23760	74	48.25	12	22.67	5:maneger
463		1	2	17640	40000	66	35.33	16	10.67	5:maneger
464		0	1	12996	24000	68	28.83	17	1.42	6:mba
465		0	1	17004	30000	65	34.5	19	4.5	6:mba
466		0	1	12996	26750	69	34.92	19	6.75	6:mba
467		0	3	7200	23250	71	27.5	16	0.92	6:mba
468		1	2	13992	26500	67	38	19	8.25	6:mba
469		0	1	16992	27700	85	43.25	20	11.17	7:engineer
470		0	1	18000	34500	66	34.25	18	4.17	7:engineer
471		0	1	31992	54000	96	49.58	19	16.58	7:engineer
472		0	1	21000	26700	83	49.92	16	21.5	7:engineer
473		0	1	18000	44250	96	39.67	19	10	7:engineer
474		0	1	13992	33000	93	46	17	17.25	7:engineer

図 1.58　Result of outlook function by jobcode key

(g) 変数選択と削除

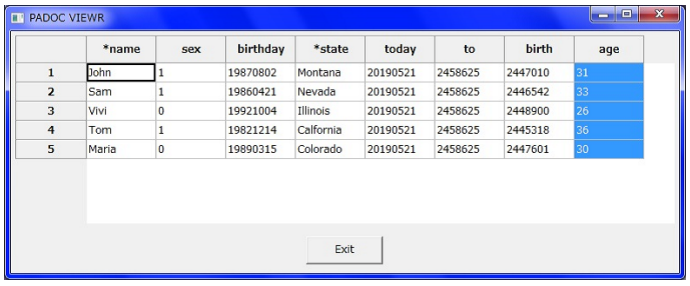
データ・テーブルの項目に不要なものは削除したい。その場合には 2 つの方法がある。

- select： 残したい変数のみ指定する
- reject： 削除したい変数のみ指定する

例えば次の例では生年月日から年齢を計算する場合、一旦ユリウス通日に変換し本日との差を取って年に直す必要がある。

```
clear;
//名簿の手入力
hand name sex birthday state/
John 1 19870802  Montana
Sam  1 19860421  Nevada
Vivi   0 19921004  Illinois
Tom  1  19821214 Calfornia
Maria 0 19890315 Colorado
;
put class;

get class;

today=20150521;
now=undate(today);  //ユリウス通日に変換

birth = undate(birthday);  //誕生日をユリウス通日に変換
age = (now-birth)/365.25;  //本日までの通算日を年に変換
age = ifix(age);    //整数に変換

reject today now birth;
```

上記を計算すると計算途中の変数も残る。これらは select または reject で必要な変数のみにすることができる。



	*name	sex	birthday	*state	today	to	birth	age
1	John	1	19870802	Montana	20190521	2458625	2447010	31
2	Sam	1	19860421	Nevada	20190521	2458625	2446542	33
3	Vivi	0	19921004	Illinois	20190521	2458625	2448900	26
4	Tom	1	19821214	Calfornia	20190521	2458625	2445318	36
5	Maria	0	19890315	Colorado	20190521	2458625	2447601	30

図 1.59 Result of age by date transform function

計算途中の変数を削除 (reject) した結果

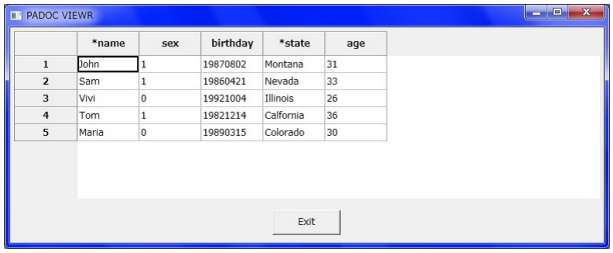


図 1.60　Rejection for work variables

(h) 行間計算

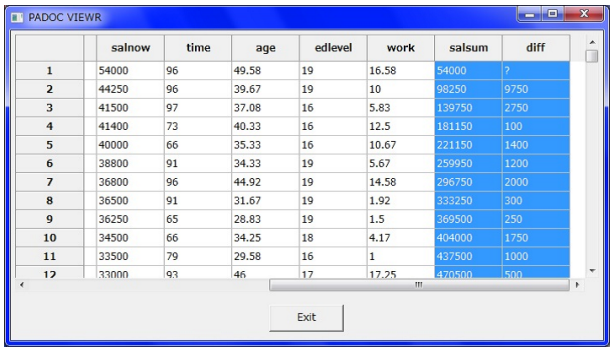
padoc の編集記述はデータ・テーブルの一行毎に適用されるが、前の行の結果を使いたい場合がある。padoc には次の様な行間計算ができる

- accume 累計宣言 これで宣言された変数は累計される
- prev 関数 前行の値を使用する

次の例は現在給与の高い順にソートして、次に高い給与との差分 (diff) を求め累計しているものである。

```
get bankR.csv@;  //行員データ・テーブルの読込み

sort -salnow;  //現在給与を降順にソート

accume salsum;  //変数 salnow を累計宣言する
diff=prev(salnow) - salnow; //prev 関数で前行の値を得て差を取る
salsum=salnow;   //累計計算
```

結果は以下の様に累計と差分が計算されている。



	salnow	time	age	edlevel	work	salsum	diff
1	54000	96	49.58	19	16.58	54000	?
2	44250	96	39.67	19	10	98250	9750
3	41500	97	37.08	16	5.83	139750	2750
4	41400	73	40.33	16	12.5	181150	100
5	40000	66	35.33	16	10.67	221150	1400
6	38800	91	34.33	19	5.67	259950	1200
7	36800	96	44.92	19	14.58	296750	2000
8	36500	91	31.67	19	1.92	333250	300
9	36250	65	28.83	19	1.5	369500	250
10	34500	66	34.25	18	4.17	404000	1750
11	33500	79	29.58	16	1	437500	1000
12	33000	93	46	17	17.25	470500	500

図 1.61　Result of calculation of lines

(i) テーブルの掛算

両データテーブルが数値のみで構成され、一方の列の数ともう一方の行の数が一致する場合、行列の掛算と同じことができる。

即ち次式の第 l 項が N × M のテーブルで第 2 項が M × P のテーブルの場合、テーブルの掛算で N × P のデータ・テーブルを生成する。

$$\begin{pmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & xii & \vdots \\ 0 & \cdots & x_{nm} \end{pmatrix} \begin{pmatrix} y_{11} & \cdots & y_{1p} \\ \vdots & y_{ii} & \vdots \\ m1 & \cdots & y_{mp} \end{pmatrix} \tag{1.1}$$

(1.2)

以下の例は 16 × 5 次元のデータを主成分分析して、3 主成分の次元変換行列 (5 × 3) を元データに掛けて 16 × 3 に変換し、これを 3D で表した結果である。

```
//16 × 5 距離データテーブルの読込
get princomp1.csv@;
//主成分分析の実行
prin a b c d e;

//結果の取出し
get freq@ana;

//(I-1) 上位の 3 主成分の変換ベクトル 5 × 3 テーブルの取出し
select vector_0 vector_1 vector_2;
put prin;

//(I-2)16 × 5 の距離データテーブルの読込.
get princomp1.csv@;
select a b c d e;
//)3 主成分の変換ベクトルに観測データを乗じて 3 主成分 5 × 3 の変換を得る
//mxmult は行列 16 × 5 と 5 × 3 とを乗じて 16 × 3 の行列を得る
mxmult by prin;

//(I-3)16 × 3 主成分データのグラフ表示
plot scat vector_0 vector_1 vector_2;
```

(I-1)5 × 3 主成分の次元変換のデータ

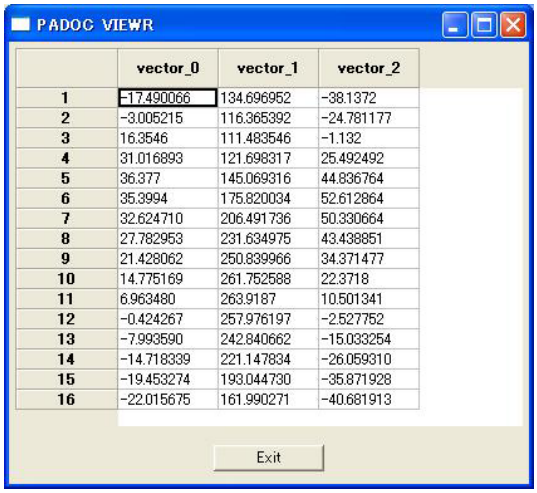


図 1.62　Result of matrix calculation by mxmult command

(I-2)16 × 5 次元の観測データ

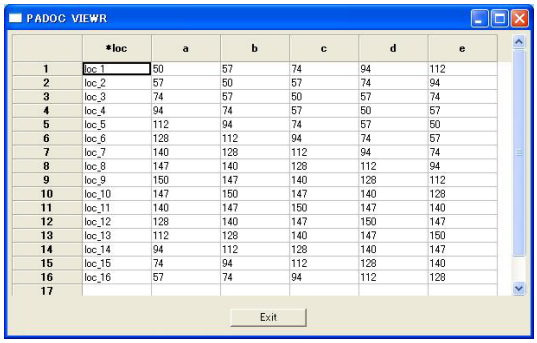


図 1.63　Distance data from 5 points to 16 trees

(I-3) テーブルの掛算で 16 × 3 のデータにして、3D プロットした結果

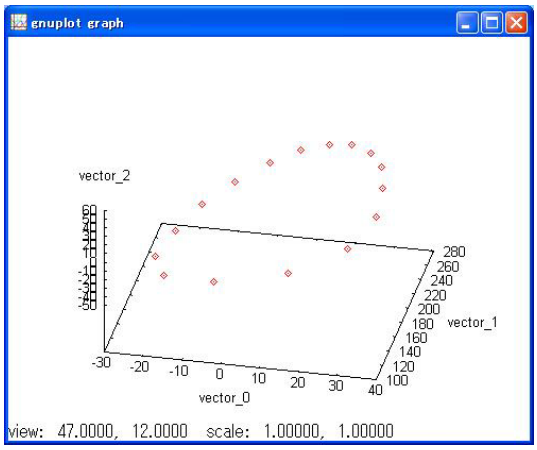


図 1.64　plot reduced 3D data by principle analysis

1.4.4 メタ記述：編集記述の再利用

padoc の編集記述では一旦使ったロジックを再利用するため、編集記述のメタ記述が使える。メタ記述には次の 2 通りがある。

- メタ関数： 関数仕様による再利用
- メタ制御： メタ制御による編集記述の制御
- メタ変数： データテーブルに反映されない大域的な変数の定義

(a) メタ関数

メタ関数とは次の例`$func irnd()` に示す様に引数を渡して呼出す再利用のための定義である。
この例ではバスケットボールのフリー・スローの投入確率をプレイヤーの格付 (level) で説明できるかという課題で、格付が高い人と低い人は人数が少ないので格付の説明力が損なわれると考えられる。実際の NBA での投入確率の人員構成は図 1.65 になっている。

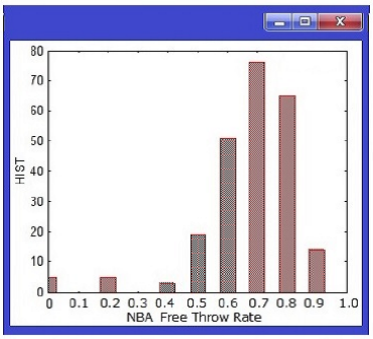


図 1.65 Composion of level in NBA

そこでこの様なデータを生成するため、格付毎にデータ生成するロジックを再利用して 5 格付について繰返す例が以下の編集記述である。
この編集記述では irnd というメタ記述を定義している。メタ関数は`$func-$end` で定義され引数を持っている。メタ関数の定義後に引数で生成件数が異なる`$irnd` でメタ記述を呼び出して 5 種類のデータ・テーブルを生成している。

```
$func irnd(n,lvl,pd,fo);  //メタ記述の定義開始 引数は（件数、格付、投入確率、出力テーブル名）
clear;
level=?lvl;   //引数の変数には?が付く
for(i=0;i< ?n;i++) {  //引数 ?n 件数だけデータを生成する
  rnd=random;  //乱数を振る
  if(rnd < ?pd) flag=0;  //乱数未満なら不成功
  else          flag=1;  //乱数以上なら成功
  outrec;
}
put ?fo;  //生成データ・テーブルの保存先（引数）
$end;  //メタ記述の定義終了

$irnd( 100,1,0.9,f1); //メタ記述の呼出し
$irnd( 300,2,0.7,f2);
$irnd( 500,3,0.5,f3);
$irnd( 300,4,0.3,f4);
$irnd( 100,5,0.1,f5);

get f1;
concat f2;
concat f3;
concat f4;
concat f5;  //生成した 5 個のデータ・テーブルの結合

plot hist level;  //レベル別のヒストグラム表示
```

定義体を 5 回繰返し生成された格付別件数の表示

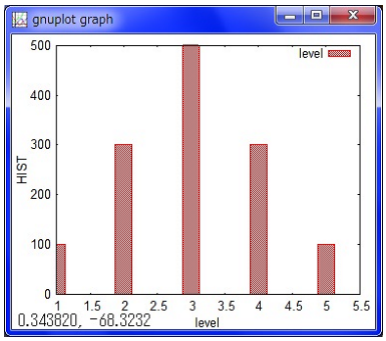


図 1.66　Generated composion of level by def

メタ関数仕様は padoc の編集記述と同じであるが、メタ関数は次の点で特有の記述となっている。

- メタ関数は`$func+` メタ記述名で始まり`$end` で終了する

- メタ関数の引数はメタ記述の中では**?**が接頭字になっている
- メタ関数の呼出しは接頭字が**$**で始まるメタ記述名と引数値の設定で行う

(b) メタ制御とメタ変数

メタ制御はコマンド編集記述を制御するロジックである。これには以下のものがある。

- `$loop`： コマンド編集記述の繰返し制御
- `$if $then $else`： コマンド編集記述の条件分岐
- `$set`： メタ変数の設定

例えばデータ・テーブル名が連番でこれらを一つに連結したい場合メタ制御を使うと以下の記述になる。
方法としては、メタ関数で生成したデータ・テーブル (`f_no1` から `f_no5`) までをメタ制御の繰返で読込む。
メタ関数による生成結果の連結は

- メタ制御 if 条件で 1 回目は fa に書出し
  put fa;
- メタ制御 else 条件で 2 回目以降は
  concat fa; //結果を連結
  put fa; //fa を更新

```
$irnd( 100,1,0.9,f_no1); //メタ記述の呼出し
$irnd( 300,2,0.7,f_no2);
$irnd( 500,3,0.5,f_no3);
$irnd( 300,4,0.3,f_no4);
$irnd( 100,5,0.1,f_no5);

$set cnt 5;   //メタ変数に繰返回数を設定

$loop i=1 to ?cnt;   //メタ制御  繰返開始
  get f_no?i;    //(注 1) メタ制御  データ・テーブルの読込み
  fileno = stradd("file_no",num2str(?i));  //(注 2) データ・テーブル名の設定

  $if ?i == 1 do;    //メタ制御 条件開始  繰返回数が 1 の場合
    put fa;        //データ・テーブルの書出し
  $end;           //メタ制御  条件終了
  $else do;       //メタ制御  条件開始
    concat fa;     //データ・テーブルの連結
    put fa;        //連結結果を更新する
  $end;         //メタ制御  条件終了
$end;              //メタ制御  繰返開始

count fileno;   //データ・テーブル名毎の件数表示
count fileno flag;  //データ・テーブル名毎の flag=1 の構成比表示
```

図 1.67 はメタ制御　5 回繰返で連結した生成したデータ・テーブルの頻度結果である。　メタ関数の呼出し (`$irnd()`) の第 1 引数で設定した件数が正しく生成されていることが分る。

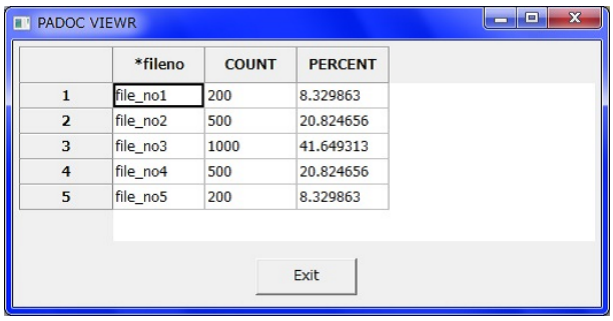


図 1.67　Result of meta control for generated count

(注 1) `get f_no?i` で`?i` はメタ制御繰返しの回数を示し、3 回目なら `?i` は 3 となっており　`get f_no3` と padoc では解釈する

(注 2) `$loop` 内の 3 行目　stradd 関数は文字列の連結で num2str は数値の文字変換である。 従って `stradd("f_no",num2str(?i))` は"`f_no3`"となる。

第 2 章

PADOC/STAT コマンド記述による分析

- 分析コマンドによる結果の表示
  padoc では次に示す様に多数の分析コマンドがある。一般に分析コマンドを実行した結果は図 2.1 のアイコンを押下すると結果が表示される。
  (注) 分析コマンドを実行してもカレント・データはそのままであるので、結果を参照してコマンド編集記述を続けることができる。

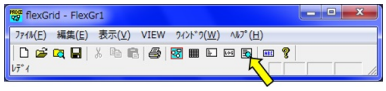


図 2.1　Push buttom for showing concurrent data

- 分析結果の利用
  分析結果のデータ・テーブルを使うには次の様に `get freq@ana` で結果テーブルを読込みカレント・データテーブルを置き換えする必要がある。

```
get bank.csv@;  //行員データテーブルの読込み
reg salnow by minority jobcat gender age edlevel work;  //重回帰

get freq@ana;  //結果テーブルの読込み
plot line salnow reg; //回帰対象と回帰結果の比較
```

- option
  また分析コマンドは分析の微調整のため option があるのが一般的である。option の指定はコマンド行の最後に [/] で改行し、セミコロン [;] で終了する。例えば頻度コマンドでは次の様になる。Option の値の設定は [:] または [=] で行う。

```
get bank.csv@;
//option の設定 結果を bank_count に出力する
count jobcat salnow/
jobcat:code
file=bank_count
;
```

- 分析モデル

  分析モデルはデータの特性を調べる基礎統計分析と用途に応じた分析に大別される。用途に応じた分析は、教師データが存在しこれを説明するモデルとベイズ統計による隠れ変数推定をする教師なしモデル分けている。

  – 基礎統計分析：データの特性を調べる
  – 教師あり分析：教師データを説明するモデル
  – 教師なし分析：隠れ変数の推定モデル

2.1 基本統計分析

2.1.1 基本統計量 (statis)

padoc ではデータ・テーブルの状態を俯瞰するため基本統計量を計算することができる。

```
//職種名テーブルの作成
hand jobcat jobname/
1 1:stuff
2 2:training
3 3:guardman
4 4;special
5 5:manager
6 6:MDA
7 engineer
;
//テーブルを jobnm の名称で保存
put jobnm;

hand minority racename/
0 0:whitle
1 1:colored
;
//テーブルを jobnm の名称で保存
put racenm;

//行員データ・テーブルの読込み
get bank.csv@;
//職種コードと職種名との結合
merge jobnm by jobcat;
//人種コードと人種名との結合
merge racenm by minority;
//職種名と人種名が付いたデータテーブルを保存
put bankRR.cat@;

statis salnow salbeg time age work;  //(A) 複数の変数の統計量
statis salnow by jobname;  //(B) カテゴリィ別の統計量
plot box salnow by jobname;  //(C) 箱髭図の表示
```

例 (A) 現在給与　初任給 年齢 勤務時間 教育年数 勤務年数の複数の統計量表示

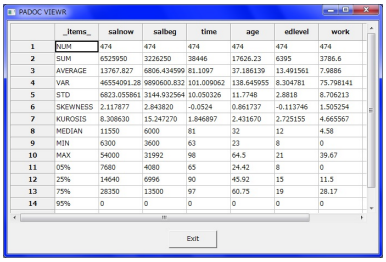


図 2.2　Result of basic statistic for variables

例 (B) カテゴリィ職種 (jobcat) 別の現在給与の基本統計量

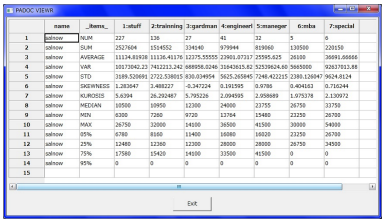


図 2.3　Basic statistic of now salary by jobcat name

例 (C) 図 2.4 のグラフは職種別の現在給与の箱髭図で基本統計量より分り易い。箱は 75% 分位点 ∗は平均値を示す。

図 2.4　Box chart of now salary by jobcat name

statis コマンドの option

表 2.1　Options for statis command

name	value	content	default
variable	code	variable as code	
variable	format	format file for variable	
file	filename	specify file name for output	

以下の例は jobcat がコードである場合、statis コマンドの option の format で jobcat コードを以下の様に設定すると jobcat 名称を毎に基礎統計量が計算される。これは上述の merge によらない直接的な方法である。結果は図 2.5 の様になる

```
//jobcat 名称のテーブルの作成
hand id idnm/
1 stuff
2 training
3 guardman
4 special
5 manager
6 MDA
7 engineer
;
//テーブルを以下の名称で保存
put jobcat_catnm;
//行員データ・テーブルの読込み
get bank.csv@;
//基礎統計量の計算　option で jobcat 名称を指定している
statis salnow by jobcat/
 jobcat:format=jobcat_catnm
 minority:code
 file=bank_statis
;
//結果の出力先を file で指定して、これを読込みしている
get bank_statis;
```

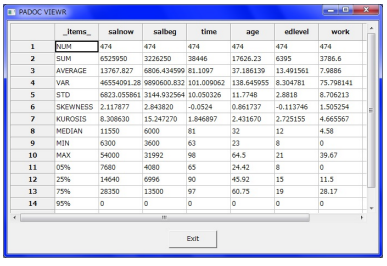


図 2.5 Result of basic statistic by format option

2.1.2　頻度分析

図 2.7 の様に複数の変数の区分について件数をカウントする。区分は以下で行われる。

- 文字変数： 同じ文字について件数をカウントする (例 A)
- 数値変数： 変数の分布に応じて自動的に分割する (例 B)
- コード変数： コード毎に件数をカウントする
  但し、コードが数値の場合コードであることを指定する必要がある (例 C)

図 2.6 読込んだ行員データには職種名や人種名が反映されている。変数が文字かコードの場合はその区分でカウントされるる。

数値変数の場合、図 2.8 の様に padoc が数値を自動的に区切ってカウントする。

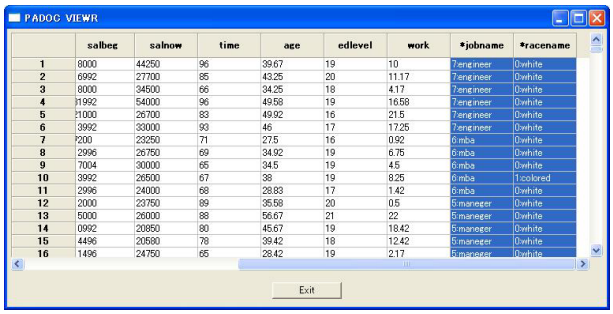


図 2.6　Data Table with jobname and racename

```
get bankRR.csv@;  //行員データテーブルの読込み
count jobname racename;  //(A) 複数の文字変数毎の件数表示
count salnow;       //(B) 数値データ自動分割による件数
count jobcat minority/   //(C) 数値データをコードと指定して件数表示
   jobcat=code
   minority=code
;
```

例 (A) 複数変数かつ文字変数での件数表示

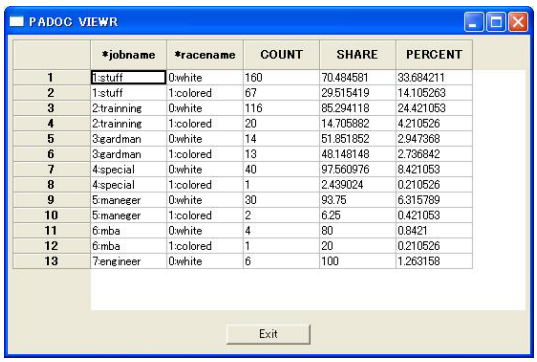


図 2.7　Result of frequency by string

例 (B) 数値変数での自動分割による件数表示

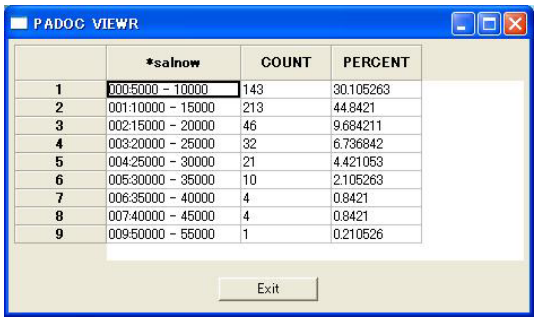


図 2.8 Result of frequency by numeric variable

例 (C) 数値変数を数値コードと指定した件数表示

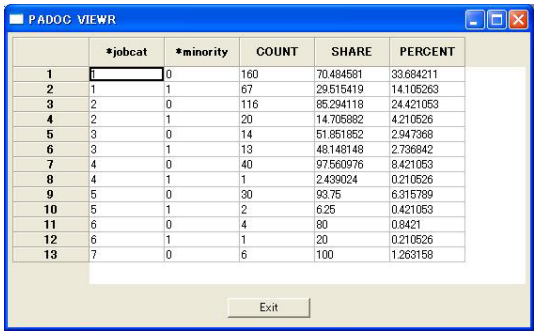


図 2.9 Result of frequency by coded numeric variable

count コマンドの option

表 2.2 Options for count command

name	value	content	default
variable	code	variable as code	
variable	format	format file for variable	
file	filename	specify file name for output	

2.1.3　集計分析

頻度分析は件数の集計であるが、同じ切り口で値の集計を行う。
次の例は米国の行員データで職種別、人種別に現在給与と初任給をサマリーしたものである。

```
//行員データテーブルの読込
get bankRR.csv@;
//職種名別の給与の平均を計算
sumup salnow salbeg by jobname racename/
 method=average
;
//サマリー結果の取出し
get freq@ana;
//人種別にテーブルを分離する。
if(strsel(racename,1,1) == 0) outrec bank0;
else                          outrec bank1;
//変数名を変更
get bank0;
rename salnow=salnow_w;
rename salbeg=salbeg_w;
reject racename;
//テーブルを連結
merge bank1 by jobname;
//職種別人種別　平均給与の棒グラフ
plot bar salnow_w salnow by jobname;
```

職種別と人種別の給与の現在給与と初任給のサマリー (平均) の結果

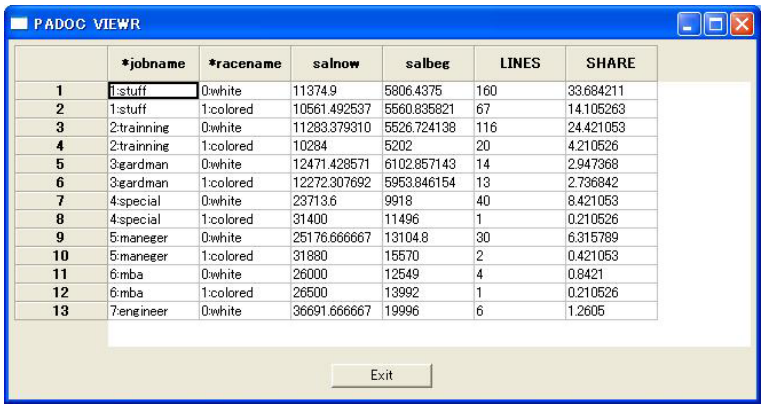


図 2.10　summary table by jobname and race for now and begin salary

職種別と人種別の給与の現在給与の平均比較結果

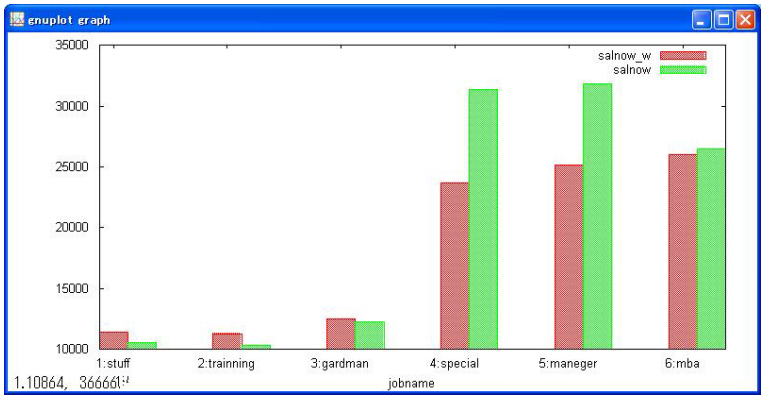


図 2.11 Comparison of avereage now salary by jobname and race

summary コマンドの option

表 2.3 Options for summary command

name	value	content	default
method	sum average	summation average	sum
variable	code	variable as code	
variable	format	format file for variable	
file	filename	specify file name for output	

2.1.4　関連分析

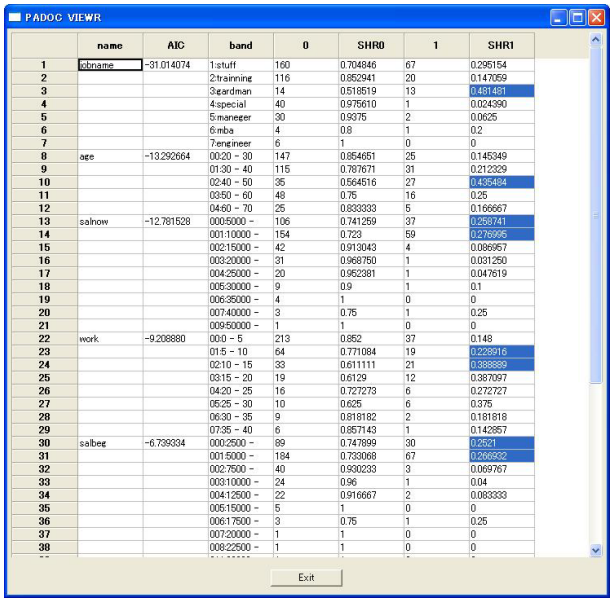
関連分析は一般に注目する変数がその他の変数との関係の強さをランキングする場合が多い、注目する変数が 2 値を摂る場合、padoc では AIC 表を提供している。AIC 表には注目する変数と i 番目の変数間での無関係な場合の AIC 値 $aic_i^{norelation}$ と実際の AIC 値 aic_i^{fact} の差が表示され、負の値が大きい順にランキング表示されている。

$$aic_i^{diff} = aic_i^{norelation} - aic_i^{fact} \tag{2.1}$$

次の例は米国行員の少数民族であることが強く関係している変数をランキング表示している。

```
get bankR.csv@;  //行員データ・テーブルの読込

//少数民族と関わりが強い変数のランキング表示
aic minority by jobcat gender salnow salbeg age time edlevel work;
```

図 2.12 では職種 (jobname)、年齢 (age)、現在給与 (salnow) の順に関係が強いことが示されている。タイトル欄にある SHR1 は少数民族=1 での各変数での構成率で、職種名では半数が警備員であり、現在給与では低い値に集中していることが分る。



	name	AIC	band	0	SHR0	1	SHR1
1	jobname	-31.014074	1:stuff	160	0.704846	67	0.295154
2			2:trainning	116	0.852941	20	0.147059
3			3:gardman	14	0.518519	13	0.481481
4			4:special	40	0.975610	1	0.024390
5			5:maneger	30	0.9375	2	0.0625
6			6:mba	4	0.8	1	0.2
7			7:engineer	6	1	0	0
8	age	-13.292664	00:20 - 30	147	0.854651	25	0.145349
9			01:30 - 40	115	0.787671	31	0.212329
10			02:40 - 50	35	0.564516	27	0.435484
11			03:50 - 60	48	0.75	16	0.25
12			04:60 - 70	25	0.833333	5	0.166667
13	salnow	-12.781528	000:5000 -	106	0.741259	37	0.258741
14			001:10000 -	154	0.723	59	0.276995
15			002:15000 -	42	0.913043	4	0.086957
16			003:20000 -	31	0.968750	1	0.031250
17			004:25000 -	20	0.952381	1	0.047619
18			005:30000 -	9	0.9	1	0.1
19			006:35000 -	4	1	0	0
20			007:40000 -	3	0.75	1	0.25
21			009:50000 -	1	1	0	0
22	work	-9.208880	00:0 - 5	213	0.852	37	0.148
23			01:5 - 10	64	0.771084	19	0.228916
24			02:10 - 15	33	0.611111	21	0.388889
25			03:15 - 20	19	0.6129	12	0.387097
26			04:20 - 25	16	0.727273	6	0.272727
27			05:25 - 30	10	0.625	6	0.375
28			06:30 - 35	9	0.818182	2	0.181818
29			07:35 - 40	6	0.857143	1	0.142857
30	salbeg	-6.739334	000:2500 -	89	0.747899	30	0.2521
31			001:5000 -	184	0.733068	67	0.266932
32			002:7500 -	40	0.930233	3	0.069767
33			003:10000 -	24	0.96	1	0.04
34			004:12500 -	22	0.916667	2	0.083333
35			005:15000 -	5	1	0	0
36			006:17500 -	3	0.75	1	0.25
37			007:20000 -	1	1	0	0
38			008:22500 -	1	1	0	0

図 2.12　Result of aic ranking for minority between other variables

aic コマンドの option

表 2.4 Options for aic command

name	value	content	default
variable	code	variable as code	
variable	format	format file for variable	
file	filename	specify file name for output	
method	aic	ranking by aic value	aic
	kai	ranking by χ^2 value	

2.1.5　相関分析

変数間の相関係数や共分散を計算する。

```
//成績データテーブルの読込み
hand id name math  sience literature english society/
1  jon    89 90 67 46 50
2  smith  57 70 80 88 90
3  kazuo  80 90 35 40 50
4  sam    40 60 50 45 55
5  mark   78 85 45 55 85
6  michel 55 65 80 75 85
7  jony   90 85 88 92 95
8  babara 79 92 65 36 48
9  rose   59 72 82 90 92
10 risa   78 88 33 38 48
11 pam    38 57 55 50 60
11 maria  75 88 24 52 83
12 moris  65 75 90 85 95
13 tom    80 75 78 82 85
;
put subject;
//Execute corration  for subjects(A)
corr math sience literature english society;

//数学と他の教科の関係図 (B)
get freq@ana;
if(# == 1) outrec;
plot bar math sience literature english society;

//文学と他の教科の関係図 (C)
get freq@ana;
if(# == 3) outrec;
plot bar math sience literature english society;

//Execute co-variance for subjects
get subject;
cov math sience literature english society;
```

(A) 相関係数の結果は次のとなる。

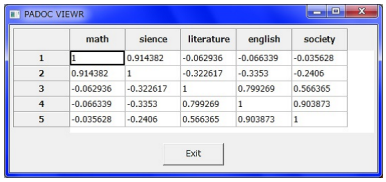


図 2.13 Result of corration between subjects

(B) 数学と他の教科の相関は科学とは強いが文系科目とは無相関である。

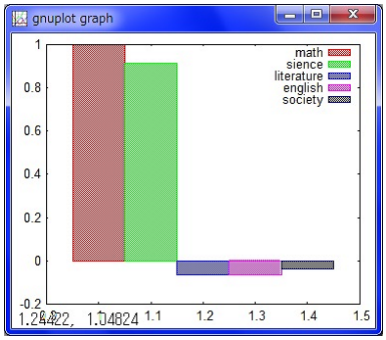


図 2.14 Corration of math between other subjects

(C) 文学と他の教科で見ると文系科目の相関は強いが科学は逆相関である

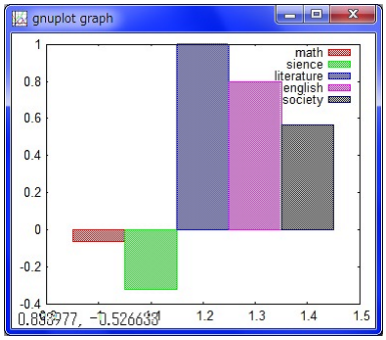


図 2.15 Corration of literature between other subjects

corr コマンドの option

表 2.5 Options for correlation command

name	value	content	default
method	corr	outputs correlation	corr
	cov	outputs covariance	

2.2　教師あり分析

このモデルは教師データ (目的変数) があり、一般的に目的変数と関わりが強いデータ (説明変数) から教師データを予測するモデルである。

2.2.1　線形回帰分析

一般に重回帰モデルと云われ教師データが実数値の場合に適応される。一方説明も実数値が一般的であるが、文字変数やコード変数の場合はダミー変数化 (onehot）すれば使える。

(注 1) onehot による文字やカテゴリィのダミー変数化は次の様に設定される。例えば A,B,C と 3 種類の区分があれば、このダミー変数は次となる。

```
variable dmy1 dmy2 dmy3
  A        1       0      0
  B        0       1      0
  C        0       0      1
  A        1       0      0
  A        1       0      0
  C        0       0      1
  B        0       1      0
```

(注 2）回帰に投入するダミー変数はカテゴリィ数から 1 つ減らした数となる。例えば職種 (jobcat) ではカテゴリィが 7 個なのでダミー変数は 7 個生成されるが、投入変数は 6 個になる。これは全部投入すると行列演算のランク落ちが発生し計算が不定になる為である。

```
get bank.csv@;    //行員データテーブルの読込み
//(A)onehot にするため整数変数をコードの型に変更
attr name type/
 jobcat code
 minority code
 gender code
;
//(B)onehot でダミー変数を生成　対象変数はコードか文字に限定
onehot gender minority jobcat;
put bank_dmy;
//(C) ダミー変数を使った回帰を実行　ダミー変数の数は 1 つ減らす
reg salnow by jobcat_dmy1-6 gender_dmy1 minority_dmy1
     age time edlevel work;
//結果の読込み
get freq@ana;
plot reg; //(D) 回帰結果と実値との比較
```

(A)attr コマンドでコード型に変換したので、ヘッダーには (*) が付いている

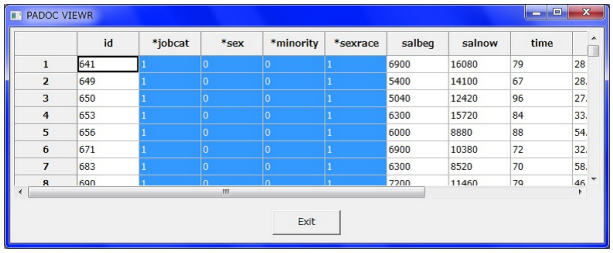


図 2.16　Transrate type of integer variable for code

(B)onehot コマンドでダミー変数を追加した結果

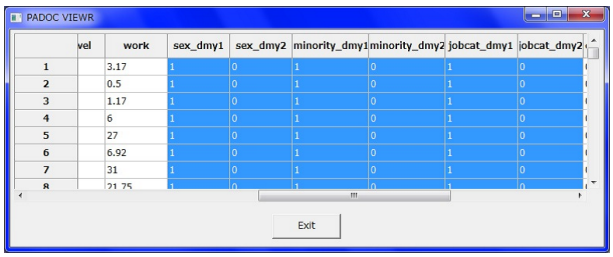


図 2.17　Generation dummy variable by onehot command

(C) 回帰の結果は図 2.18 で示す様に右端に (*) が付いているのが有意な変数である。

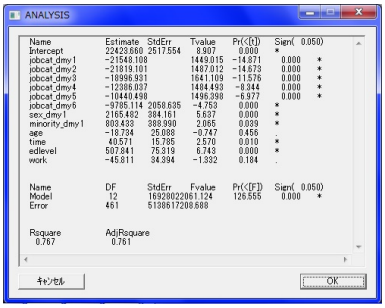


図 2.18　Result of multi-regression with dummy variables

(D) 回帰の結果で目的変数と回帰値との比較

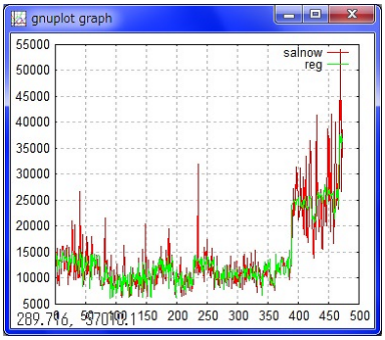


図 2.19　Comparison regression prediction and real data

reg コマンドの option

表 2.6　Options for regression command

name	value	content	default
intercept	'none'	specifies no intercept	

2.2.2 ロジステック回帰分析

線形回帰の教師データは実数だったが、ロジステック回帰では教師データ (目的変数) は 2 値の値を持つ。説明変数は線形回帰と同じで実数が一般的であるが、コード変数や文字変数は onehot でダミー変数化すれば使える。但しロジステック回帰も投入できるのはダミー変数の数は 1 個少ない数である。
ロジステック回帰での所属確率 pd は次の様な式をしているが、Z 値は回帰係数 α とデータ $x_1 \cdots x_n$ との線形和であり線形回帰モデルの一つであることが分る。

$$pd = \frac{1}{(1+\exp(-z))} \tag{2.2}$$

$$z = \beta + \sum_{i=1}^{N} \alpha_i \dot{x}_i \tag{2.3}$$

次の例はロジステック回帰で行員データから少数民族である確率を求めるものである。

```
get bank.csv@;      //行員データテーブルの読込み
attr name type/  //整数変数をコード変数とする
  jobcat code
  gender code
;
onehot jobcat gender;  //onehot でダミー変数化する

//(A) 少数民族についてロジステック回帰をする
logit minority by jobcat_dmy1-6 gender_dmy1
      salnow salbeg age time edlevel work;

get freq@ana;  //結果の取出し
plot scat countRate logitRate;  //(B) パレート図の表示
```

(A) 回帰の結果は図 2.20 で示す様に右端に (*) が付いているのが有意な変数である。
このロジット回帰の AR=49% である。

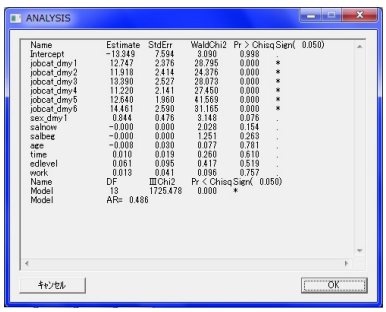


図 2.20 Result of logistic regression with dummy variables

(B) 図 2.21 はパレート図で、少数民族の確率が高い順に並べたものである。高い確率の全体の 20 %で少数民族を 40 %補足していることが分る。

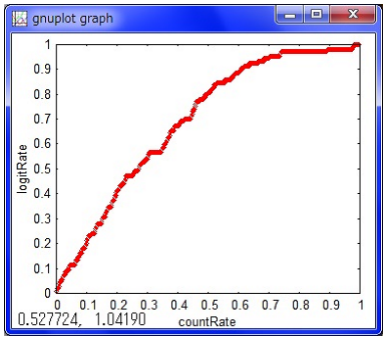


図 2.21　Result of palate graph of logistic regression

logit コマンドの option

表 2.7　Options for logistic regression command

name	value	content	default
target	0	larger pd show larger numbers	
	1	larger pd show fewer numbers	1

2.2.3 SVM

非線形の判別モデルの SVM(Support Vector Machine) は、線形回帰やロジステック回帰が線形的なモデルと異なり、判別する空間をカーネルで高次元に写像して教師データを出来るだけ分離する方法を採っている。SVM は線形では判別できない対象に適用できるが、一方問題の空間を歪めてしまうのでロジステック回帰や線形回帰の様に識別対象の所属確率を算出できないモデルとなっている。

次の例は図 2.22 の様に閉じられた閉曲面を判別する問題で、図 2.23 では非線形識別の SVM は適切に解いている。

```
get svm3.csv@;    //閉曲面識別データテーブルの読込み
plot scat x y by kbn/  //(A) 識別データの図表示
  kbn=code;
;

if(kbn == 1 or kbn == 2) flag=1;     //SVM の識別フラグの設定
else                     flag=-1;

if(kbn == ? or x == ? or y == ?) delrec;  //欠損値の排除

svm x y by flag/  //SVM の実行 ガウスカーネル使用
 type=gauss
;
get freq@ana;   //結果の読込み
plot scat x y by predict;  //(B) 識別結果の図表
```

(A) 閉曲面の識別課題データの図表示 □印の点が閉曲面でそれ以外の点を分断している。

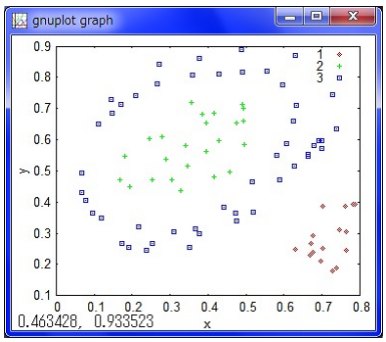


図 2.22 Classification of close surface

(B)SVM による判別結果の図表示　SVM は非線形な区分でも可能であることが分る。

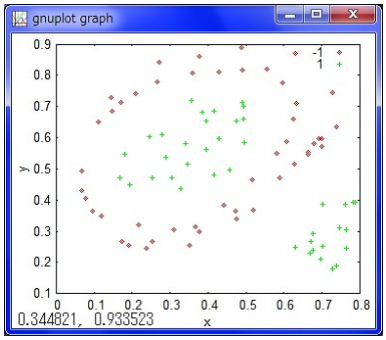


図 2.23　Result of classification by SVM

SVM のオプション

表 2.8　Options for SVM command

name	value	content	default
method	linear	linear	
	multi	polynomal	
	gauss	gaussian	gauss
	laplase	laplase	
	sigmoid	sigmoid	
margin	psitive	regularize margin	1000
kkt	decimal	tolerance of KKT	0.001
eps	decimal	convergency condition	0.001
delta	decimal	δ for gaussian kernel	0.1
degree	positive	degree for multi kernel	4
coef	positive	coef for liner or sigmoid kernel	0
test	decimal	share of test data	0

(注 1)SVM は線形分離不可能な識別を行うが、解けない場合が多い。その場合上述の option で調整が必要である。
(注 2) カーネル関数の定義は以下である。ここで u は v データの各行のベクトルを示す。

- type=liner: liner function $u^T * v + coef$　　u and v are vector
- type=multi: polynomial function $(u^T * v + coef)^{degree}$
- type=gauss: gaussian function $\exp(\frac{|u-v|^2}{\delta})$　　(default)
- type=laplase:laplase function $\exp(\frac{|u-v|}{\delta})$
- type=sigmoid:sigmoid function $tanh = \frac{1}{\exp(1+\exp(-u' * v + coef))}$

2.2.4 判別木

判別木は別名 decsion tree と云われ、ロジステック回帰と同様に目的変数が 2 値を持つ場合に適用される。判別木は J.Ross Quinlan による情報量 (entrophy) が減少する（説明力が高くなる) 様にデータ分割を繰返して細分化する方法である。padoc では Quinlan の C4.5 のモデルを採用している。 ロジット回帰はコードや文字変数について特別な配慮 (onehot) が必要であったが、判別ツリーはその様な配慮が不要である

次の例は 500 人にダイレクトメール出し実際にモデルハウスを訪れた人に見込み客として good のフラグが付いたデータである。AIC 表示コマンドで図 2.24 訪れた人についての関係の深かい変数についてランキング表示している。AIC 表で上位にランキングした変数を判別木モデルに投入している。

```
//ダイレクトメールに応じて訪問した客のデータテーブルの読込み
get dmdatKNJ.csv@;

//(A) 訪問客について関連が深い項目をＡＩＣ表でランキングしている
aic def by old area dm1 dm2 dm3 dm4 dm5
  income work home workspan homespan gender family job amount;
//(B) 判別木でどの様な客層が訪問するか分析している
tree def by old area income work home
    workspan homespan gender family job amount/
target/good,bad
terminal=10
;
//判別木の結果の取出し
get freq@ana;
plot line distincRate by countRate;   //パレート図の表示 (C)
```

(A)AIC ランキング表を見ると現在の持家でない人で北部地域に住み子供が 2 人以上の先が訪問していることが分る。

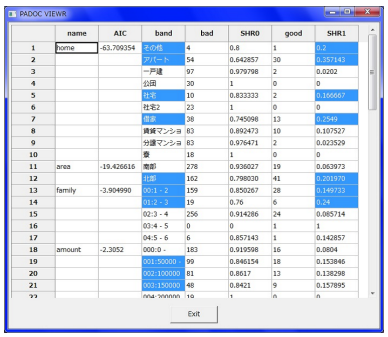


図 2.24 Result of AIC table which shows stronger relations with good action

(B) 図 2.25 の生成された判別木を見ると持家でなく北部地域に住み月収が 24 万超で家族が 1 一人超 (子供がいる) であれば 100 %訪問していることが分る。

図 2.25　Result of decsion tree of good action

(C) 図 2.26 はパレート図で判別木で訪問確率が高い順に客層を並べて、実際に訪問客の累計構成比をプロットしている。この図から訪問確率の高い全体の 20 %で実際の訪問客の 70 %補足していることが分る。営業はこの客層に行えば非常に効率的であることが判明する。

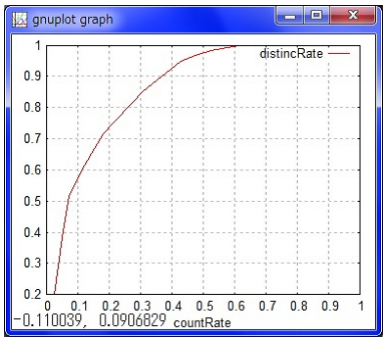


図 2.26　Palate graph of decsion tree

(注 1) 判別木で末端の数を設定することが一般的である。上記の例では tree コマンドのオプションとして terminal=10 で末端が 10 人以上としている。

tree コマンドの option

表 2.9 Options for decsion tree command

name	value	content	default
target	value,value	target values	required
terminal	value	minmum at teminal node	100
test	percent	share of data for test	0
make	percent	share of data for making	100
variable	code	code type	
	continueous	continuous type	
	format	format type	
	char	character type	

(注 2)option の target は必須である。[target/] の後は教師データの値をカンマで区切って入れる必要がある。
次の例は教師データの値が OK か NG になっており、最初の指定 OK について判別木を生成する。

```
target/OK,NG
```

2.2.5　回帰木

判別木は目的変数が 2 値であったが、回帰木は目的変数が実数で、目的変数の数値が増加や減少する方向に判別木と同様に分岐を繰返して分割している。これも線形回帰ではコードや文字変数に配慮が必要であったが、回帰木では不要である。

次の例は同じ 500 人ダイレクトメールに応じた住居購入金額別の客層である。

```
//ダイレクトメールに応じて訪問した客のデータテーブルの読込み
get dmdatKNJ.csv@;
//(A) 回帰木による住居購入金額別の客層の分析
tree amount by income old area work home
      workspan homespan gender family  job/
target/continuous
terminal=20
;
```

(A) 回帰木によると現在一戸建でサービス・土建業で 51 歳超が最も高い物件を希望している。

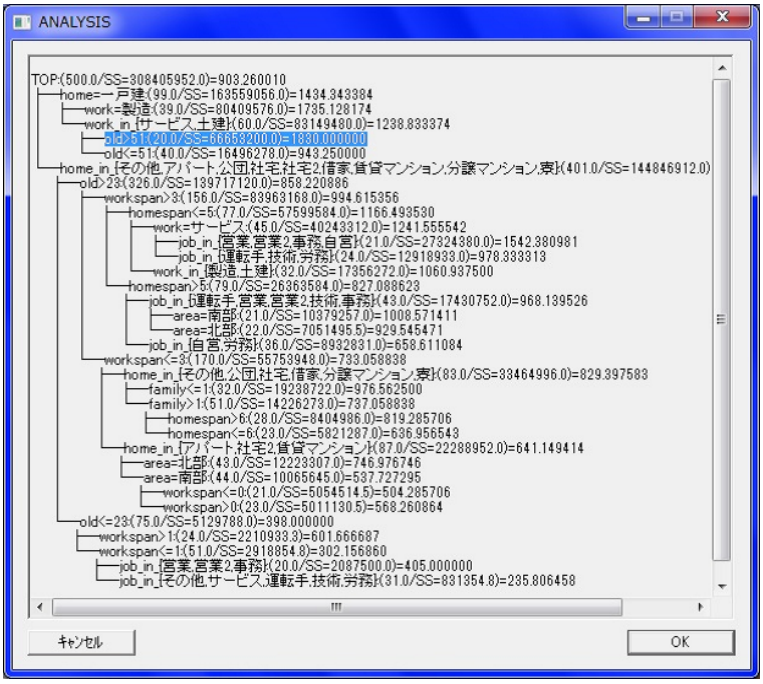


図 2.27　Result of regression tree for house amount

回帰 tree コマンドの option

表 2.10 Options for regression tree command

name	value	content	default
target	'continuous'	specify regression	required
terminal	value	minmum at teminal node	100
test	percent	share of data for test	0
make	percent	share of data for making	100
variable	code	code type	
	continueous	continuous type	
	format	format type	
	char	character type	

(注 1)option の target は必須である。[target/] の後は continueous と記述必要がある。

```
target/continuous
```

2.2.6　ナイーブベイズ

naive bayes は教師区分 C が存在し、説明変数 F_i 間が独立と仮定する場合にベイズの定理より次式が成立し、識別は各変数の所属確率の積で表される。しかしナイーブベイズは説明変数間に多少の相関あっても分別できる。

$$
\begin{aligned}
p(C|F_1,\cdots,F_n) &= \frac{p(F_1,\cdots,F_n|C)p(C)}{p(F_1,\cdots,F_n)} \\
&= \frac{p(C,F_1,\cdots,F_n)}{Z} \\
p(C,F_1,\cdots,F_n) &= p(C)p(F_1|C)p(F_2|C)\cdots p(F_n|C) \\
&= p(C)\Pi_{i=1}^{n} p(F_i|C) \\
p(C|F_1,\cdots,F_n) &= \frac{1}{Z}p(C)\Pi_{i=1}^{n} p(F_i|C)
\end{aligned}
$$

次の例は横に 92 個の文書があり、縦に単語毎の頻度を示したデータ・テーブルである。この文書の種類 (0 と 5) の識別を頻度の分布でナイーブベイズで行う。

この分別では aic の関連ランキング機能でどの単語が強く関連しているか分析し、3D 図にプロットするため 3 単語を選択する。ナイーブベイズはこの選択した単語で行い、高い確率順に文書を並べてパレート図を描き精度を確認する。

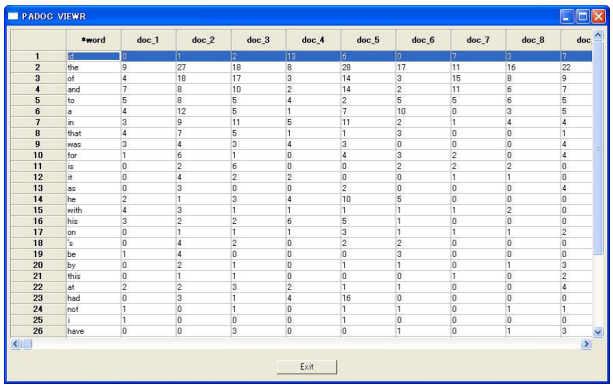
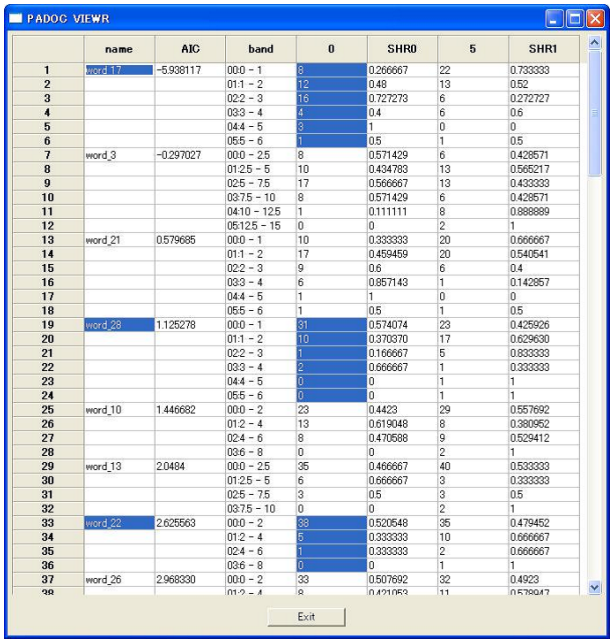


図 2.28　Input data which has word frequency by document id

```
//文書種類と単語の頻度データテーブルを読む
get brownlexR.csv@;
//2 種類の文書を取り出す
if(id == 0 or id == 5) {
  outrec;
}
//(A)aic 関連ランキング表の作成
aic id  by word_1-32;
//(B) 選択した 3 変数での 3D プロット図
plot scat word_17 word_23 word_22 by id;
//(C)naive bayes の実行
nbayes id by word_17 word_23 word_22;
//結果の取り出し
get freq@ana;
//(D) パレート図の表示
plot line bayesRate by countRate;
```

(A) 図 2.29 は aic コマンドによる文書種類と変数の関連ランキングの表示



	name	AIC	band	0	SHR0	5	SHR1
1	word_17	-5.938117	00:0 - 1	8	0.266667	22	0.733333
2			01:1 - 2	12	0.48	13	0.52
3			02:2 - 3	16	0.727273	6	0.272727
4			03:3 - 4	4	0.4	6	0.6
5			04:4 - 5	3	1	0	0
6			05:5 - 6	1	0.5	1	0.5
7	word_3	-0.297027	00:0 - 2.5	8	0.571429	6	0.428571
8			01:2.5 - 5	10	0.434783	13	0.565217
9			02:5 - 7.5	17	0.566667	13	0.433333
10			03:7.5 - 10	8	0.571429	6	0.428571
11			04:10 - 12.5	1	0.111111	8	0.888889
12			05:12.5 - 15	0	0	2	1
13	word_21	0.579685	00:0 - 1	10	0.333333	20	0.666667
14			01:1 - 2	17	0.459459	20	0.540541
15			02:2 - 3	9	0.6	6	0.4
16			03:3 - 4	6	0.857143	1	0.142857
17			04:4 - 5	1	1	0	0
18			05:5 - 6	1	0.5	1	0.5
19	word_28	1.125278	00:0 - 1	31	0.574074	23	0.425926
20			01:1 - 2	10	0.370370	17	0.629630
21			02:2 - 3	1	0.166667	5	0.833333
22			03:3 - 4	2	0.666667	1	0.333333
23			04:4 - 5	0	0	1	1
24			05:5 - 6	0	0	1	1
25	word_10	1.446682	00:0 - 2	23	0.4423	29	0.557692
26			01:2 - 4	13	0.619048	8	0.380952
27			02:4 - 6	8	0.470588	9	0.529412
28			03:6 - 8	0	0	2	1
29	word_13	2.0484	00:0 - 2.5	35	0.466667	40	0.533333
30			01:2.5 - 5	6	0.666667	3	0.333333
31			02:5 - 7.5	3	0.5	3	0.5
32			03:7.5 - 10	0	0	2	1
33	word_22	2.625563	00:0 - 2	38	0.520548	35	0.479452
34			01:2 - 4	5	0.333333	10	0.666667
35			02:4 - 6	1	0.333333	2	0.666667
36			03:6 - 8	0	0	1	1
37	word_26	2.968330	00:0 - 2	33	0.507692	32	0.4923
38			01:2 - 4	8	0.421053	11	0.578947

図 2.29　Result of aic analysis for ducument id

(B) 図 2.30 は選択した 3 単語での文書種類 (0 と 5) とを 3D プロットした図。分布に相違が確認できる。

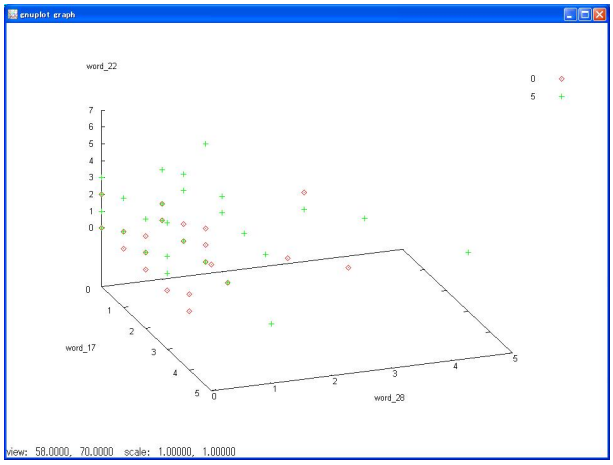
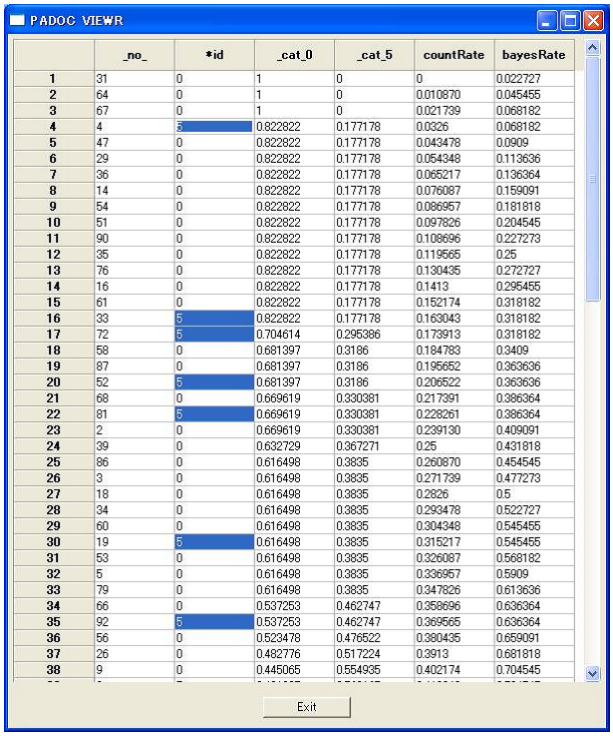


図 2.30　3D plot by selected 3 variable

(C) ナイーブベイズの結果。図 2.31 文書種類 (0) の確率が高い順に文書番号を並べている。文書種類 (5) は少ないことが分かる。



	_no_	*id	_cat_0	_cat_5	countRate	bayesRate
1	31	0	1	0	0	0.022727
2	64	0	1	0	0.010870	0.045455
3	67	0	1	0	0.021739	0.068182
4	4	5	0.822822	0.177178	0.0326	0.068182
5	47	0	0.822822	0.177178	0.043478	0.0909
6	29	0	0.822822	0.177178	0.054348	0.113636
7	36	0	0.822822	0.177178	0.065217	0.136364
8	14	0	0.822822	0.177178	0.076087	0.159091
9	54	0	0.822822	0.177178	0.086957	0.181818
10	51	0	0.822822	0.177178	0.097826	0.204545
11	90	0	0.822822	0.177178	0.108696	0.227273
12	35	0	0.822822	0.177178	0.119565	0.25
13	76	0	0.822822	0.177178	0.130435	0.272727
14	16	0	0.822822	0.177178	0.1413	0.295455
15	61	0	0.822822	0.177178	0.152174	0.318182
16	33	5	0.822822	0.177178	0.163043	0.318182
17	72	5	0.704614	0.295386	0.173913	0.318182
18	58	0	0.681397	0.3186	0.184783	0.3409
19	87	0	0.681397	0.3186	0.195652	0.363636
20	52	5	0.681397	0.3186	0.206522	0.363636
21	68	0	0.669619	0.330381	0.217391	0.386364
22	81	5	0.669619	0.330381	0.228261	0.386364
23	2	0	0.669619	0.330381	0.239130	0.409091
24	39	0	0.632729	0.367271	0.25	0.431818
25	86	0	0.616498	0.3835	0.260870	0.454545
26	3	0	0.616498	0.3835	0.271739	0.477273
27	18	0	0.616498	0.3835	0.2826	0.5
28	34	0	0.616498	0.3835	0.293478	0.522727
29	60	0	0.616498	0.3835	0.304348	0.545455
30	19	5	0.616498	0.3835	0.315217	0.545455
31	53	0	0.616498	0.3835	0.326087	0.568182
32	5	0	0.616498	0.3835	0.336957	0.5909
33	79	0	0.616498	0.3835	0.347826	0.613636
34	66	0	0.537253	0.462747	0.358696	0.636364
35	92	5	0.537253	0.462747	0.369565	0.636364
36	56	0	0.523478	0.476522	0.380435	0.659091
37	26	0	0.482776	0.517224	0.3913	0.681818
38	9	0	0.445065	0.554935	0.402174	0.704545

図 2.31　Result of naive bayes which shows high probability about each document no

(D) 図 2.32 は文書種類 (0) の確率が高い順位ならべ、文書種類 (0) の捕捉を示したパレート図である。文書種類 (0) は全体の 41 %あるので完全に判別するモデルでは Perfect Model の線になる。一方全く判別しないモデルでは 45 度の Random Model となる。Naive Model は黄色で示すモデルであり、僅か 3 単語でも高い識別を達成している。

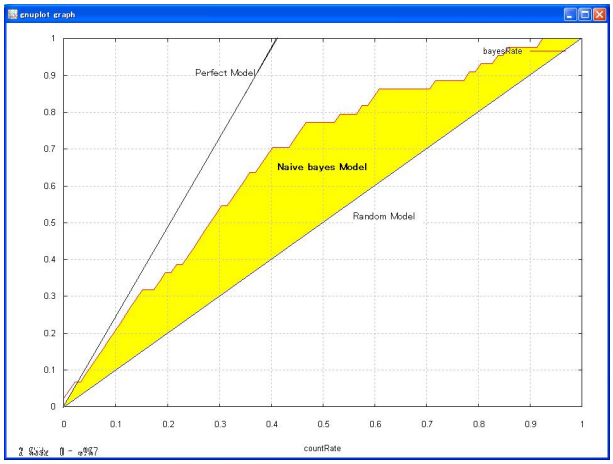


図 2.32 Result of naive bayes about palete figure

2.2.7　生存分析

padoc は次式の Cox の比例ハザードモデルで生存分析を提供している。$h_0(t)$ はベースラインで共変量に依存しない経過時間に関する関数で、$x_1 \cdots x_n$ が共変量で α_i が共変量の係数である。この式より線形モデルの一種であることが分る。Cox の比例ハザードは $h_0(t)$ と α_i の両方を共変量に基づいて計算する。

$$h(t) = h_0(t) \exp(\alpha_1 * x_1 + \cdots + \alpha_n * x_n) \tag{2.4}$$

図 2.33 の例は米国の格付 (grade) 毎の最初の社数と 12 年間の倒産件数である。格付 (grade) を共変量として、Cox の比例ハザードで格付毎で経過年別倒産発生率の予測を行う。

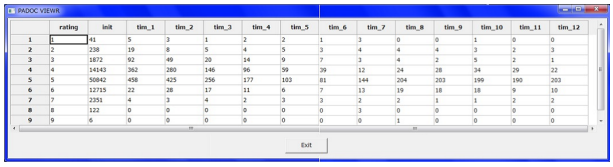


図 2.33　Initial number of companies and bankruptcy in USA

Cox の比例ハザードの分析には, コマンド coxgen で図 2.34 のセンサーデータに変換してから cox コマンドを使う。センサーデータとは共変量毎に観測できる時系列の倒産件数とセンサー打切り時の残余社数のデータである。

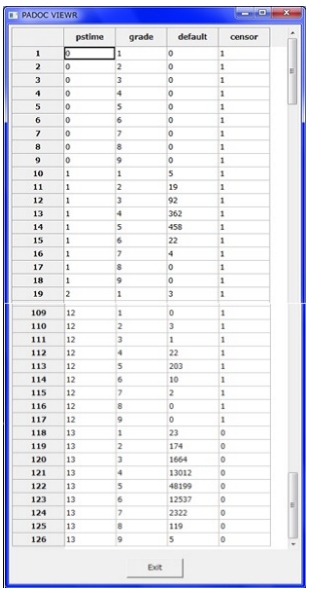


図 2.34　Transration bankruptcy to cencer data

```
//ハザードデータテーブルの読込み
get haz2.csv@;
//センサーデータテーブルの作成
coxgen init tim_1-12 by rating;
get freq@ana;  //センサーデータテーブルの読込み
//Cox 比例ハザードの実行
cox default by rating;
//結果の読込み
get freq@ana;
//共変量係数の取出し
select beta_0;
put beta;
//ベースラインの取出し
get freq@ana;
select time base;
//ベースラインと共変量係数の連結
append beta;
//(A)Cox ハザードの式に合わせて格付毎・経過年別の発生率を計算
vector grade[9];
for(i=1;i<=9;i++) {
  grade[i]=base*exp(i*beta_0);
}
//格付毎・経過年別の発生率のグラフ表示
plot line base grade_1 - 9 by time;
```

(A) 格付が悪い程倒産発生率が高いことが分る。また経過 2 年でピークを迎え経過 6 年まで減衰し、以降安定的に推移することが分る。強いストレスを恒常的に受けると弱い生物は早期に死滅するが、適応力のある生物は粘り強く生き抜く自然界と同じ傾向が見られる。

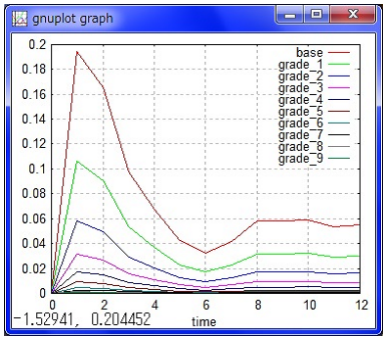


図 2.35 Result of bunkrupt transition by grade

2.2.8　k-nn

K 個近傍探索 (K nearest neighbor) は所属先が未定の点の k 個の近傍が複数の所属先にどれだけ多く所属しているかで、所属先を決めるものである。
例えば図 2.36 には 3 集団あり、円で囲まれた区分が 0 の 3 点について、この点の k 個の近傍が所属する件数を数え、多く所属した集団に属すると見做す。

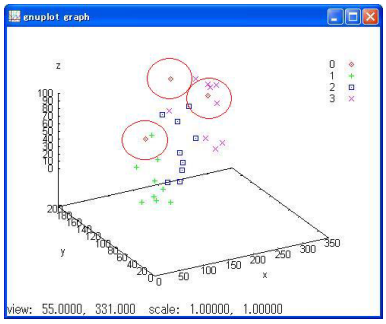


図 2.36　k-nn conception

```
//20 点の座標位置と所属区分データテーブルの読込
get maha2R.csv@;
//(A) 所属区分が 0 の点について、近傍の 4 個の点が所属する数で確率を計算する
k_nn by kbn/
count=4
;
//計算結果の取込み
get freq@ana;
//所属確率が最大の区分を見つける
vector p[3]=prob_0-2;
pd=0;
kbn=0;
for(i=1;i<=3;i++) {
   if(pd < p[i]) {
     pd = p[i];
     kbn=10*i;
   }
}
//(B) 推定された所属での 3D 表示
concat maha2R.csv@;
plot scat x y z by kbn/
kbn=code
;
```

(A)k-nn の実行結果は図 2.37 で得られ、所属確率が計算されている。

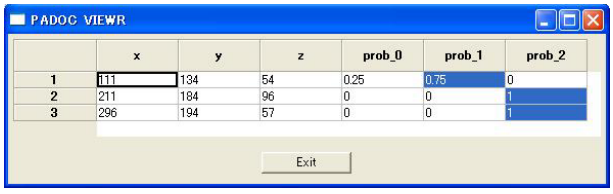


図 2.37　Output of k-nn

(B) 所属確率が大きい集団に分類した結果が図 2.38 である。判例の 20 の 1 点は所属が 2 判例の 30 の 2 点は所属 3 と判断している。

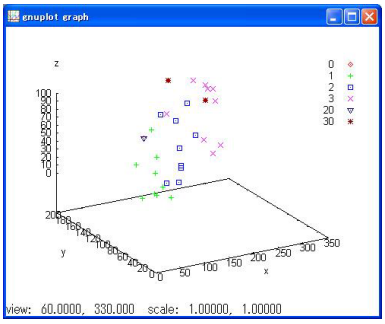


図 2.38　Result of k-nn in 3D figure

knn コマンドの option

表 2.11　Options for k nearest nighbor command

name	value	content	default
count	number	number of neighbor	4

2.2.9　マハラノビスによる識別

マハラノビス距離は図 2.39 の様に集団が多変量正規分布すると仮定して、所属が未知の点と各集団の中央点との距離の近さで属する集団を推定する。

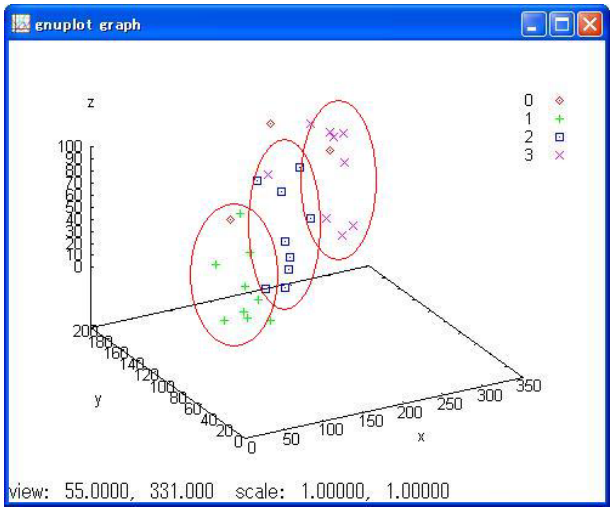


図 2.39　Conception for mahalanobis distance

```
//20 点の座標位置と所属区分データテーブルの読込
get maha2R.csv@;
//(A) 所属区分が 0 の 3 点について
maha by kbn;

//計算結果の取込み
get freq@ana;
//距離が最小の区分を見つける
vector d[3]=dis_0-2;
dd=999;
kbn=11;
for(i=1;i<=3;i++) {
  if(dd > d[i]) {
     dd=d[i];
     kbn=10+i;
  }
}
//(B) 推定された所属での 3D 表示
concat maha2R.csv@;
plot scat x y z by kbn/
kbn=code
;
```

(A) マハラノビスの距離の実行結果は図 2.40 で得られ、所属確率が計算されている。

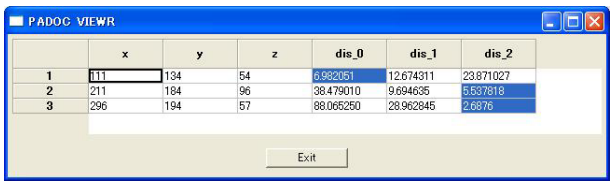


図 2.40　output of mahanobis distance

(B) 距離が短い集団に分類した結果が図 2.41 である。判例の 12 の 1 点は所属 2 判例の 13 の点は所属 3 と判断している。

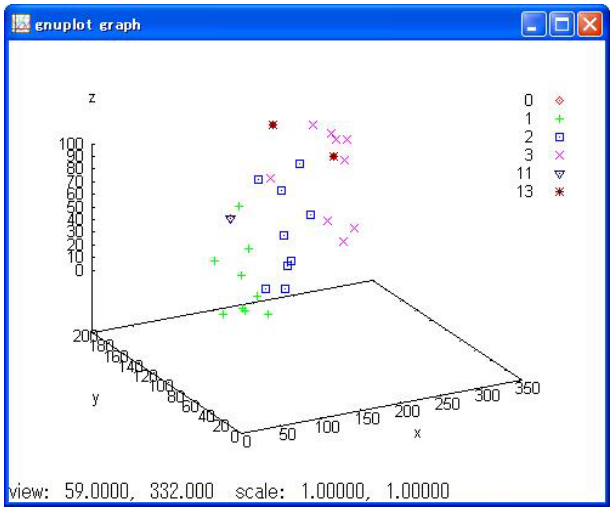


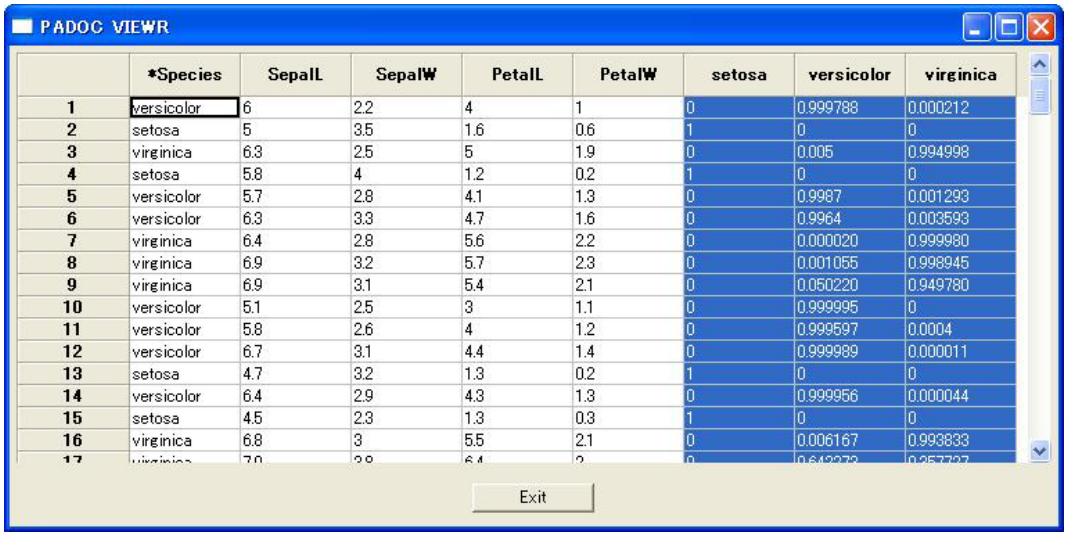
図 2.41 Result of segmentation for mahalanobis distance

2.2.10　softmax

ロジット回帰は 2 値だが softmax は 2 値超の区分を回帰して予測する。

```
//菖蒲データの読込
get iris.csv@;

//ユーザ登録コマンド重回帰の実行
softmax Species by SepalL SepalW PetalL PetalW;

get freq@ana;
//(A) 種別毎の確率の平均
statis setosa versicolor virginica by Species;
get freq@ana;
if(_items_ == "AVERAGE") outrec;
//(B) 種別毎の確率の平均の棒グラフ
plot bar setosa versicolor virginica by name;
```

(A)softmax の結果表 2.42 では種別の確率が出力される。



	*Species	SepalL	SepalW	PetalL	PetalW	setosa	versicolor	virginica
1	versicolor	6	2.2	4	1	0	0.999788	0.000212
2	setosa	5	3.5	1.6	0.6	1	0	0
3	virginica	6.3	2.5	5	1.9	0	0.005	0.994998
4	setosa	5.8	4	1.2	0.2	1	0	0
5	versicolor	5.7	2.8	4.1	1.3	0	0.9987	0.001293
6	versicolor	6.3	3.3	4.7	1.6	0	0.9964	0.003593
7	virginica	6.4	2.8	5.6	2.2	0	0.000020	0.999980
8	virginica	6.9	3.2	5.7	2.3	0	0.001055	0.998945
9	virginica	6.9	3.1	5.4	2.1	0	0.050220	0.949780
10	versicolor	5.1	2.5	3	1.1	0	0.999995	0
11	versicolor	5.8	2.6	4	1.2	0	0.999597	0.0004
12	versicolor	6.7	3.1	4.4	1.4	0	0.999989	0.000011
13	setosa	4.7	3.2	1.3	0.2	1	0	0
14	versicolor	6.4	2.9	4.3	1.3	0	0.999956	0.000044
15	setosa	4.5	2.3	1.3	0.3	1	0	0
16	virginica	6.8	3	5.5	2.1	0	0.006167	0.993833
17	virginica	7.0	3.2	6.4	2	0	0.642272	0.357727

図 2.42　Output of softmax for multi-class regression

(B) この種別毎の平均確率の結果を図 2.43 を表示すると殆ど正しく推定していることが分る。

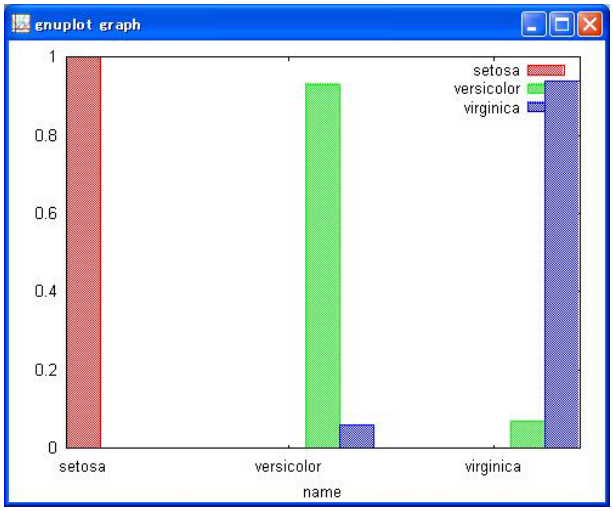


図 2.43 Figure multi-class regression probability

2.3　教師なし分析

教師なし分析とは、正解のデータが無く、データの状態から有意な情報を取り出すモデルになる。一般には隠れ変数をデータから推定するベイズ統計モデルとなる。

2.3.1　主成分分析

主成分分析は多次元のデータを次元を減らした直交する軸の空間に最適にデータを写像するものである。次の例は図 2.44 の様にＡからＥの 5 つの観測点から円形に植林された 16 本の木の距離を測った図 2.45 の 5 次元データから、主成分分析で木の配置を 3 次元の配置のデータに変換して 3D で見える状態にしている。

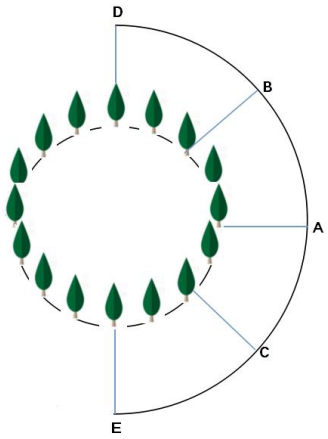


図 2.44　Meseament distance from 5 points to 16 trees

PADOC VIEWR

	*loc	a	b	c	d	e
1	loc_1	50	57	74	94	112
2	loc_2	57	50	57	74	94
3	loc_3	74	57	50	57	74
4	loc_4	94	74	57	50	57
5	loc_5	112	94	74	57	50
6	loc_6	128	112	94	74	57
7	loc_7	140	128	112	94	74
8	loc_8	147	140	128	112	94
9	loc_9	150	147	140	128	112
10	loc_10	147	150	147	140	128
11	loc_11	140	147	150	147	140
12	loc_12	128	140	147	150	147
13	loc_13	112	128	140	147	150
14	loc_14	94	112	128	140	147
15	loc_15	74	94	112	128	140
16	loc_16	57	74	94	112	128
17						

Exit

図 2.45　Distance data from 5 points to 16 trees

```
//5 観測点から 16 本の樹木までの距離データテーブルの読込
get princomp1.csv@;
//(A) 主成分分析の実行
prin a b c d e;

//結果の取出し
get freq@ana;

//上位の 3 主成分の変換ベクトルの取出し
select vector_0 vector_1 vector_2;
put prin;

//再度の距離データテーブルの読込.
get princomp1.csv@;
select a b c d e;
//(B) 主成分の変換ベクトルに観測データを乗じて 3 次元の主成分を得る
//mxmult は行列 16*5 と 5*3 とを乗じて 16*3 の行列を得る
mxmult by prin;

//(C)16 樹木の 3 主成分データのグラフ表示
plot scat vector_0 vector_1 vector_2;
```

(A) 主成分分析の結果は 5 観測点について、5 個の固有値と 5 主成分 (軸) の変換ベクトルである。固有値は 5 軸の情報量を示しており、第一主成分が殆どを占めている。

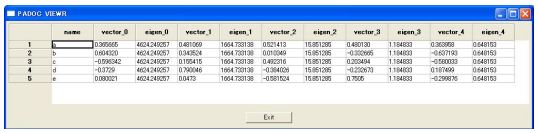
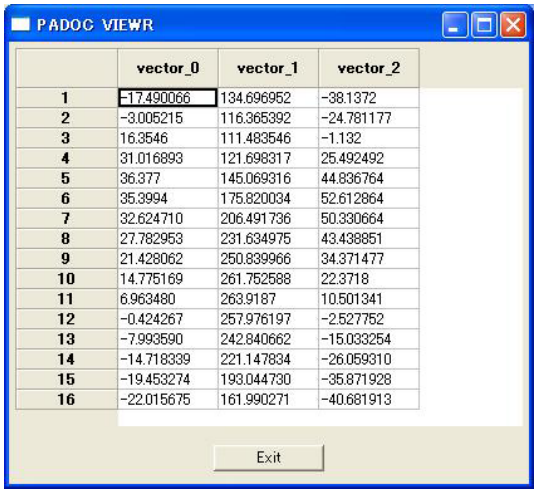


図 2.46 Result of principle component analysis

(B) 行列演算の結果は図 2.47 の様に 16 樹木について 3 主成分データとなっている。



	vector_0	vector_1	vector_2
1	-17.490066	134.696952	-38.1372
2	-3.005215	116.365392	-24.781177
3	16.3546	111.483546	-1.132
4	31.016893	121.698317	25.492492
5	36.377	145.069316	44.836764
6	35.3994	175.820034	52.612864
7	32.624710	206.491736	50.330664
8	27.782953	231.634975	43.438851
9	21.428062	250.839966	34.371477
10	14.775169	261.752588	22.3718
11	6.963480	263.9187	10.501341
12	-0.424267	257.976197	-2.527752
13	-7.993590	242.840662	-15.033254
14	-14.718339	221.147834	-26.059310
15	-19.453274	193.044730	-35.871928
16	-22.015675	161.990271	-40.681913

図 2.47　Result of matrix calculation by mxmult command

(C) 上図 2.47 の `vector_0 vecotor_1 vector_2` を 3D 表示した結果　5 次元を 3 次元に下げても円構造が生成できている。

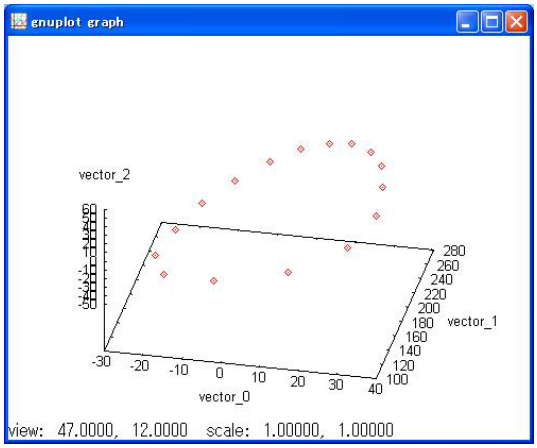


図 2.48　plot reduced 3D data by principle analysis

prin コマンドの option

表 2.12　Options for prin command

name	value	content	default
input	data	input is real data	data
	corr	input is correlation	
	cov	input is covariance	

2.3.2 分類分析

データを分類する主な手法としては以下がある。

- k-means 法
- 樹系図

(a)K-means 法

この手法は分類数を指定して、分類したデータ正規分布に従って分布しているとして、平均と分散を隠れ変数としてＥＭ (Expect Maximum) 法最適な隠れ変数を求め分類するものである。

次の図 2.49 は R.A.Fisher が提供した菖蒲の 3 種類の顎辺の幅と長さ、花弁の幅と長さでのデータである。これを K-means 法で区分して実際の種別と合うか試してみる。

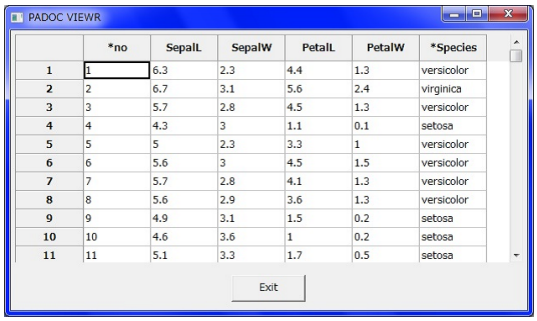
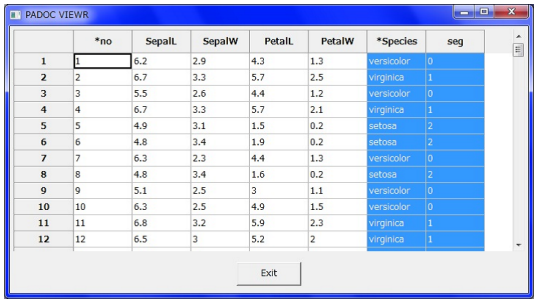


図 2.49 Iris data with species name

```
//菖蒲データテーブルの読込み
get iris.csv@;
//(A)K-means コマンドの実行
kmeans SepalL SepalW PetalL PetalW by 3;
//分類結果の取出し
get freq@ana;
select seg;
put seg;
//分類を元のデータに連結
get iris.csv@;;
merge seg by #;
//(B) 実際の菖蒲の種類と分類との比較 seg はコード変数指定
count Species seg/
 seg=code
;
```

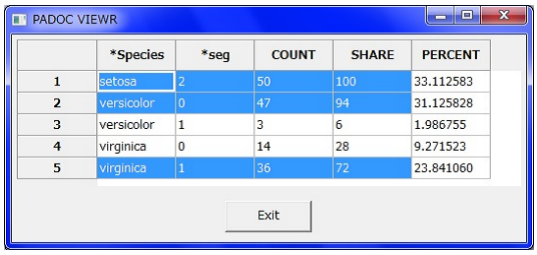
(A)K-means の結果は seg 変数で表示される。



	*no	SepalL	SepalW	PetalL	PetalW	*Species	seg
1	1	6.2	2.9	4.3	1.3	versicolor	0
2	2	6.7	3.3	5.7	2.5	virginica	1
3	3	5.5	2.6	4.4	1.2	versicolor	0
4	4	6.7	3.3	5.7	2.1	virginica	1
5	5	4.9	3.1	1.5	0.2	setosa	2
6	6	4.8	3.4	1.9	0.2	setosa	2
7	7	6.3	2.3	4.4	1.3	versicolor	0
8	8	4.8	3.4	1.6	0.2	setosa	2
9	9	5.1	2.5	3	1.1	versicolor	0
10	10	6.3	2.5	4.9	1.5	versicolor	0
11	11	6.8	3.2	5.9	2.3	virginica	1
12	12	6.5	3	5.2	2	virginica	1

図 2.50　Result of K-means by 3 segment

(B) 実際の菖蒲の種類との検証では setosa は 100 %識別、versicolor は 94 %識別、virginica は 72 %の識別であった。正解データを使わずともデータの分布状態では分類できることが分る。



	*Species	*seg	COUNT	SHARE	PERCENT
1	setosa	2	50	100	33.112583
2	versicolor	0	47	94	31.125828
3	versicolor	1	3	6	1.986755
4	virginica	0	14	28	9.271523
5	virginica	1	36	72	23.841060

図 2.51　Comparison between true segment and K-means

(b) 樹系図

樹系図はまずデータ間の類似性でグループ化して、次にグループ間の類似でグループ化を繰返して系統化するものである。例えば 3 分類する場合樹系図の頂点からの 3 サブグループで分類する。

次の例は図**??**の 10 人の 5 教化の成績を類似度で樹系図で分類したものである。分析の結果図 2.53 が得られた。これを確かめるため、5 教科の成績を 3 主成にして 3D 図で 10 人の関係を図 2.54 で示してみた。

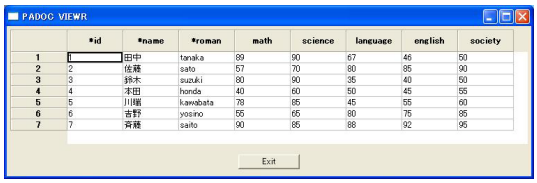


図 2.52　5 subject's grades of 10 classmate

```
//10 人の成績の読込
get factor1R.csv@;
//(A) 樹系図で 2 分類に分ける
dendro math science language engish scoiety/
  level=2
;
//主成分分析をする
prin math science language english society;

//3 主成分の軸ベクトルを取出す
get freq@ana;
select vector_0-2;

put prin_v;

//再度 10 人の成績を読込む
get factor1R.csv@;
//10 人の成績と 3 主成分の軸で 10 人の成績を 3 軸で表現
mxmult math science language english society by prin_v;
put prin_3d;

//再度 10 人の成績を読み、名前を取出す
get factor1R.csv@;
select roman;
//名前と 3 軸での成績を連結する
merge prin_3d by #;
//(B)3 軸での 10 人の位置を表示し、グループ化する
plot scat vector_0-2 by roman;
```

(A)5 教科の成績の類似度で樹系図として分類した結果

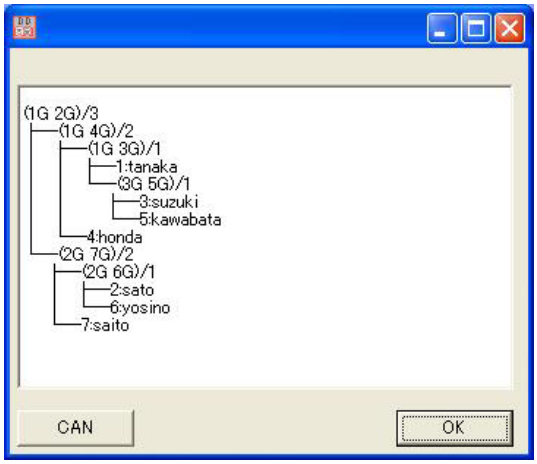


図 2.53 Result dendro gram of 10 classmate by 5 subject's grade

(B) 主成分分析して 3 主成分で 10 人を 3 D表現した結果。この図より正しく樹系図で分類されていることが分る。

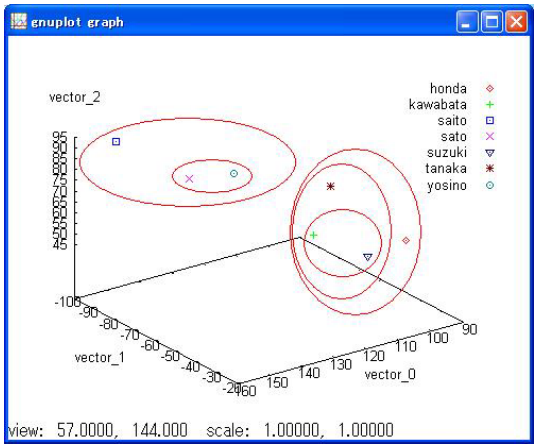


図 2.54 Result of principle component analysis with 3 axis and grouping by distance

dendro コマンドの option

表 2.13 Options for dendro command

name	value	content	default
count	positive	number of neighbor	4
level	positive	num of tree branch	2
method	1	min	1
	2	max	
	3	median	
	4	center of gravity	
	5	center of group	

2.3.3　協調フィルタリング

図 2.55 の様に縦にユーザ、横に商品の評点が反映されているテーブルがある。一般に全てのユーザが全商品を評価していないので、そこは欠損値になっている。協調フィルタリングは別名 GroupLense と云われ、以下の補正をする。

- 欠損の補填
- 評点の補正

手法としては例えばユーザ u_i で商品 $item_j$ が欠損または適切な値でなければ、その商品 $item_j$ を評価しているユーザ群で補正する。その補正はユーザ間の相関により、u_i ユーザに類似したユーザの評点に近い値にしている。

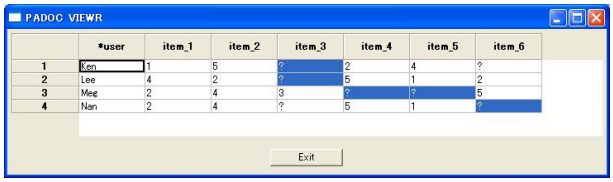


図 2.55　Score matrix of users and items with lack value

```
//欠損を含んだユーザ・商品テーブルの作成
hand user item_1-6/
Ken 1 5 ? 2 4 ?
Lee 4 2 ? 5 1 2
Meg 2 4 3 ? ? 5
Nan 2 4 ? 5 1 ?
;
put grpout.csv@;
//(A) 欠損の商品を排除したユーザ・商品評点リストを生成する
vector item[6];

for(i=1;i<=6;i++) {
  if(item[i] == ?) {
  }
  else {
    items=i;
    score=item[i];
    outrec;
  }
}
select user items score;
//(B) 協調フィルタリングを実行する
grouplens score by user items;
//結果の取出し
get freq@ana;
//(C) 欠損を補填したユーザ・商品テーブルの生成
table score_calc by items user;
//ユーザ・商品テーブルの生成結果の取出し
get freq@ana;
```

(A) 図 2.56 の様に欠損を除いた商品の評点リストが協調フィルタリングに投入される。

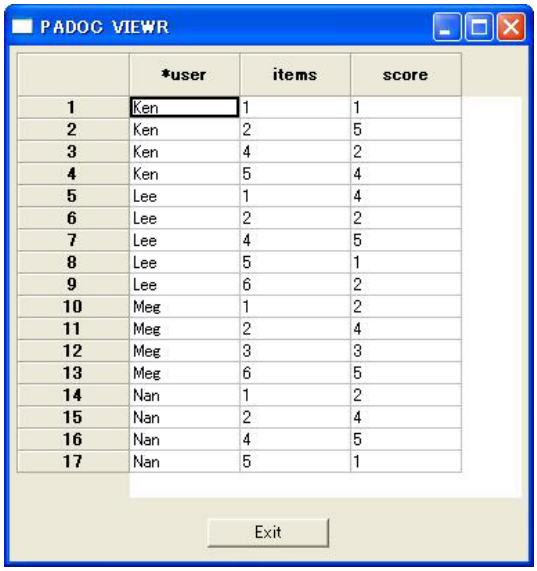


図 2.56　List of no lack score by each users

(B) 図 2.57 は協調フィルタリングの補正結果が `score_calc` 欄に表示される。

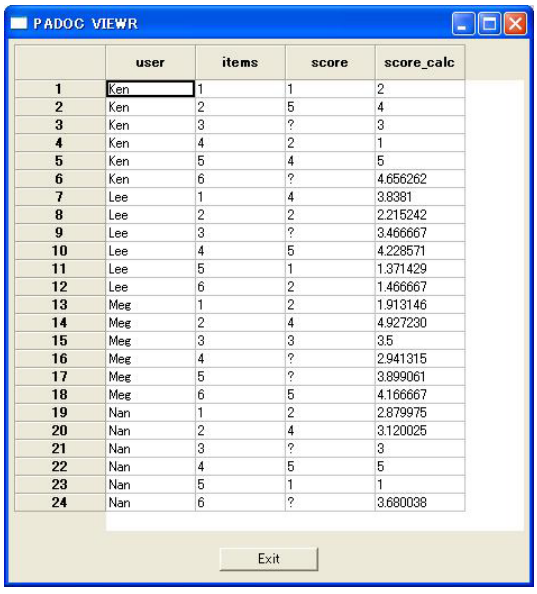


図 2.57　Result of colavorae filtering

(C) 図 2.58 は table コマンドで欠損値を補填したユーザ・商品テーブルの計算結果

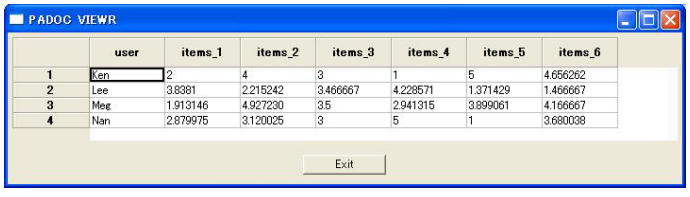


図 2.58　Result of users and items score table without lack value

grouplens コマンドの option

表 2.14 Options for grouplens command

name	value	content	default
method	ave	estimate average points	ave
	no	estimate freqency without points	
	bool	estimate exist or not	
	sum	estimate point*frequency	

2.3.4　コンジョイント分析

客の選択は感性に因ることが多く、特に商品の組合わせがある場合、適切な選択が容易でないことが多い。この分析は顧客に商品の組合わせについての相対評価のデータを使って絶対評価に変換するモデルである。

- 商品の選択の組合わせの評点から　商品を選択
- 商品の組合わせの評点テーブルから　最適な組合わせの選択

(a) 選択組合わせからの商品の推薦

次の様に旅行で買物、温泉、食事、バス旅行、列車旅行を複数選択し、その評価をランキングして、本当に何が最も好まれるか分析した例である。

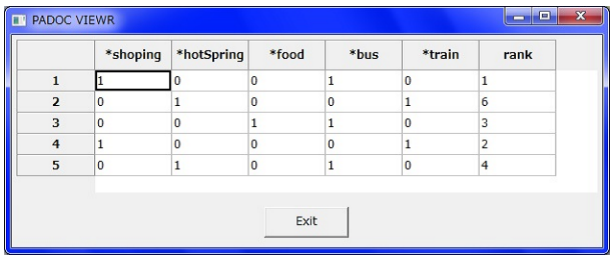


図 2.59　Conbination of tours and ranking table

```
//商品の組合わせとランキングの読込み
get conjoint1.csv@;
//コンジョイント分析
conj shoping hotSpring food bus train by rank;
//結果の取出し
get freq@ana;
transpose; //テーブルの縦横の転置
//(A) 商品の評価の表示
plot bar col_1 by _items_;
```

(A) 分析の結果図 2.60 が得られ、温泉と列車旅行が好まれていることが分る。この場合は温泉を目的地とした列車旅行が推薦できる。

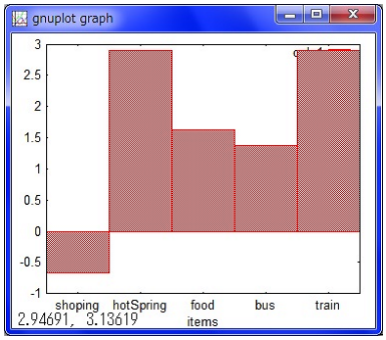


図 2.60 Point of tours by conjoint analysis

(b) 組合わせテーブルからの商品の推薦

次の例は顧客に旅行先と滞在数の選択テーブルで適当にランキングしてもらったものである。このデータから最適な旅行先と滞在数を推薦する分析である。

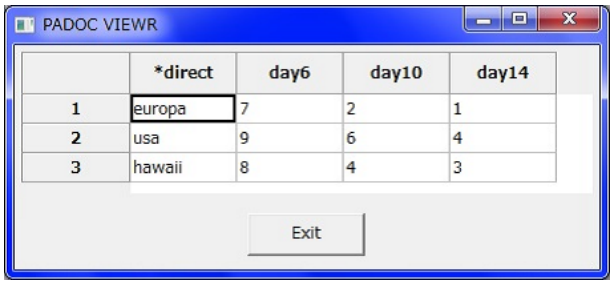


図 2.61　Conbination table of direction and dates

```
//商品の組み合わせのランキングテーブルを読込む
get conjoint2.csv@;
//コンジョイント分析を行う
conj day6 day10 day14 by direct;
//結果の取出し
get freq@ana;
transpose;//テーブルの縦横の転置
//(B) 商品の組合わせの評価の表示
plot bar col_1 by _items_;
```

(B) この結果から 2 週間のヨーロッパ滞在が最も評価が高いことが判明する。

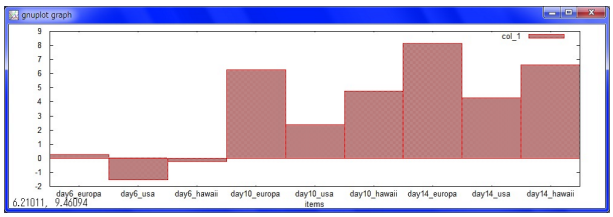


図 2.62　Result of conjoint for direction and dates

2.3.5 アソシエーション分析

図 2.63 の様なレコードの長さが異なる購買実績がある場合、同時に買う物を以下の形式で類推するモデルである。

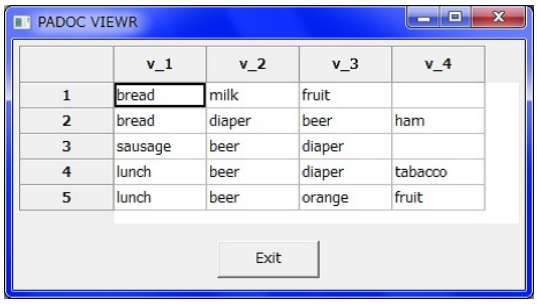


図 2.63 Purchase data for association analysis

例えば `beer <- diaper(60.0, 60.0, 100.0)` はオムツを買う人がビールを買う確率を括弧内の数値で視している。

```
beer <-  (80.0, 100.0, 80.0)
beer <- lunch  (40.0, 40.0, 100.0)
beer <- diaper  (60.0, 60.0, 100.0)
```

括弧内の数値は次の (member,support,confidence) の確率である。

- member A ¡- B $A \cap B/ALL$
- support A ¡- B B/ALL
- confidence A ¡- B $A \cap B/B$
- lift (A ¡- B) / $(A \cap B/B)/A$

support はA と B を同時に買う確率、member は B を購入する確率、confidence は B を買った人がA を買う確率である。
この様な組合わせは大量に発生するので、確率の閾値を次の様に assoc コマンドにパラメータを指定して、出力を抑制するのが普通である。

```
assoc/
  smin=xx
  conf=yy
;
```

smin=xx は support の確率が xx %以上でかつ conf=yy は confidence の確率が y y %以上が出力対象となる。一般的には上記の様な条件で出力を絞り、次式の Lift の値が高いものを見つけるのが大事である。何故なら大多数が A を購入するなら A を購入する確率は B に依存しないからである。

$$Lift = confidence/member \tag{2.5}$$

```
get assoc2.csv@;
//(A) アソシエーション分析
assoc/
 smin:0.2
 conf:0.4
;
//結果の取出し
get freq@ana;
//(B)Lift で降順にソート
sort -lift;
```

(A) 上記の例では図 2.64 とテーブル形式のの結果が得られる。

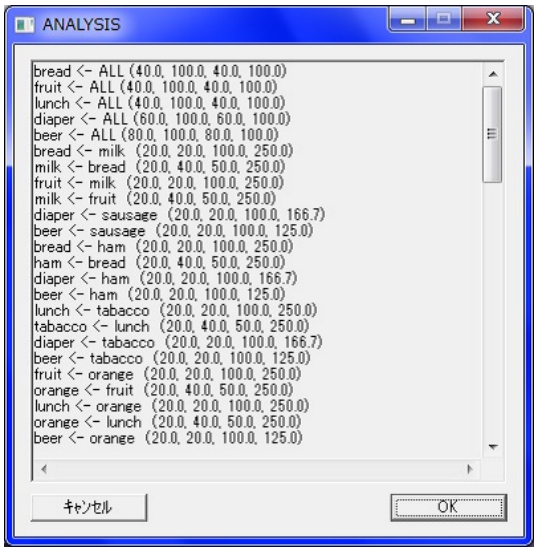
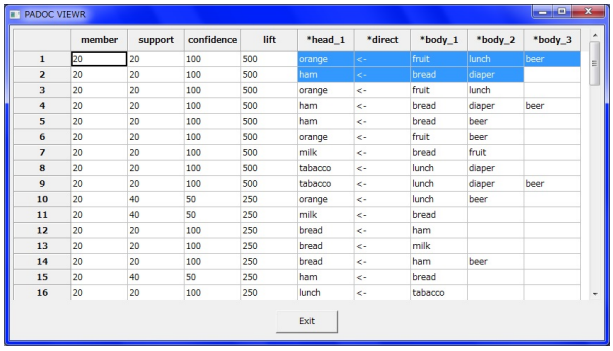


図 2.64　Result of association analysis in a market goods

(B) 図 2.65 の結果からは以下の様な事がわかる。

- 果物とランチとビールを買うと必ずオレンジを買っている
- パンとオムツを買うと必ずハムを買っている



	member	support	confidence	lift	*head_1	*direct	*body_1	*body_2	*body_3
1	20	20	100	500	orange	<-	fruit	lunch	beer
2	20	20	100	500	ham	<-	bread	diaper	
3	20	20	100	500	orange	<-	fruit	lunch	
4	20	20	100	500	ham	<-	bread	diaper	beer
5	20	20	100	500	ham	<-	bread	beer	
6	20	20	100	500	orange	<-	fruit	beer	
7	20	20	100	500	milk	<-	bread	fruit	
8	20	20	100	500	tabacco	<-	lunch	diaper	
9	20	20	100	500	tabacco	<-	lunch	diaper	beer
10	20	40	50	250	orange	<-	lunch	beer	
11	20	40	50	250	milk	<-	bread		
12	20	20	100	250	bread	<-	ham		
13	20	20	100	250	bread	<-	milk		
14	20	20	100	250	bread	<-	ham	beer	
15	20	40	50	250	ham	<-	bread		
16	20	20	100	250	lunch	<-	tabacco		

図 2.65 Result of association analysis as table view

assoc コマンドの option

表 2.15 Options for assoc command

name	value	content	default
smin	decimal	$A \cap B/ALL$	0.1
smax	decimal	$A \cap B/ALL$	0.999
conf	decimal	$A \cap B/B$	0.5

)

2.3.6　偏相関分析（ガウシアン・グラフィカル・モデル)

見かけ上では相関があるが、両方に影響する存在のため相関がある様に見えてる可能性がある。その様な存在から独立させる条件付独立を使って真の因果関係があるか見る分析である。

例えば図 2.66 の近代 4 種陸上競技の記録では、100m 走 (m100) は 110m ハードル (h110) との相関は高い。しかし図 2.67 に示す様に、400m 走の記録が早い人は、100m 走や 110m ハードルが早いだけで、互いに関係は無いかもしれない。

第 3 の影響を除去した関係は偏相関係数で計算できる。各変数間で条件付き独立を仮定しても、その相関関係の尤度が劣化しないなら図 2.68 の様に偏相関関係 0 として無関係と見做せる。

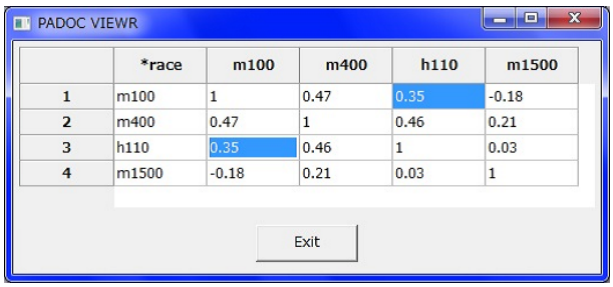


図 2.66　Correlation between Modern field 5 race

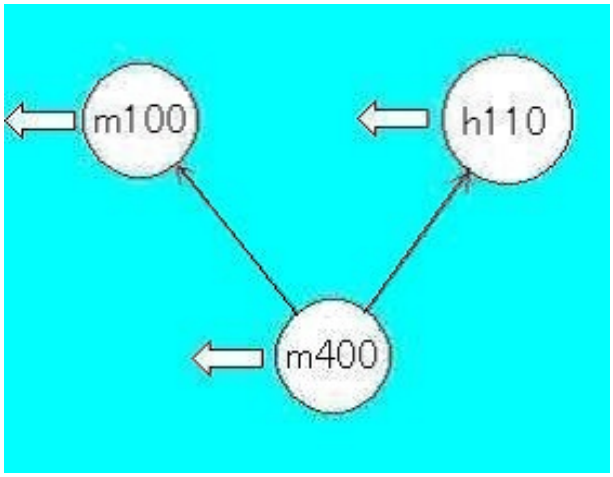


図 2.67　Apparent relationship of 3 race

```
//4 種競技の記録の相関係数の読込み
get ggm_4race.csv@
//(A) 互いに独立として仮定して相関の尤度が劣化しないか検討
ggm m100 m400 h110 m1500;
//結果の取出し
get freq@ana;
//(B)100m 走の関係を表示
plot bar m100 by _item_;
```

(A) 図 2.67 に示す様に第 3 の影響を除去した 100m 走との真の関係は 110m ハードルには無く、400m 走には強い相関、1500m 走には負の相関があることが分った。

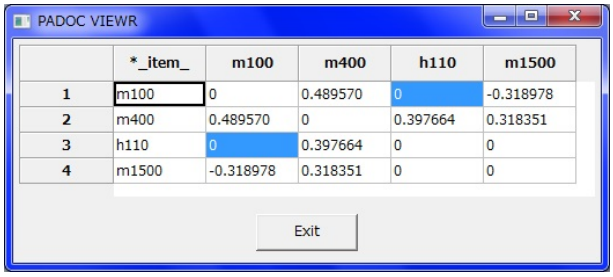


図 2.68 Result of partial correlation for field 5 race

(B)100m 走と他の 3 種との第 3 の影響を除去した関係は図 2.69 となる。

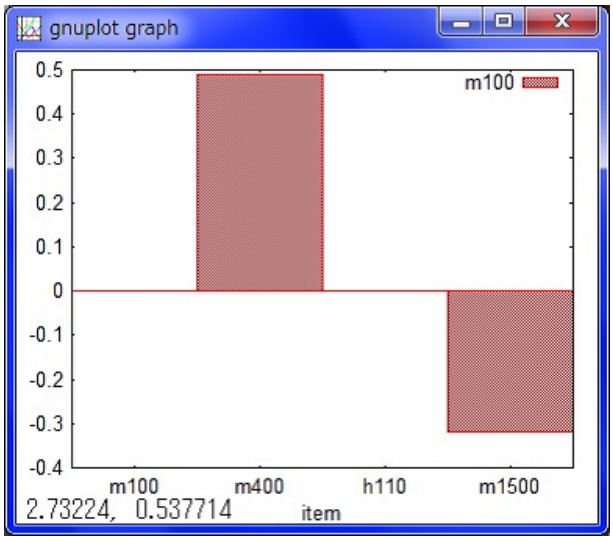


図 2.69 Result of association analysis as table view

ggm コマンドの option

表 2.16　Options for gaussian graphical model command

name	value	content	default
method	raw	input is real data	raw
	corr	input is correlation	
num	integer	num of record using corr method	101

2.3.7 異状検知

padoc では異常検知として、マハラノビス距離と One Class SVM を提供している。

(a) マハラノビス距離

マハラノビス距離はN次元の複数の点があった場合に次式にて中心からの距離を計算する。この場合共分散 Σ^{-1} を使うのは、各次元間のスケールが異なる場合、これらを標準化するためである。

$$distance = \sqrt{(\boldsymbol{x}-\boldsymbol{u})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{u})}$$

次の例は 200 個の 2 次元のデータについてマハラノビス距離を計算した例である。

```
//2D データテーブルの読込
get data2d.csv@;
//100 個の点についてマハラノビスの距離を実行
mahadist dmy_1-2;
get freq@ana;
//(A)-1 の結果の表示
plot scat dmy_1-2 mahara;

//(A)-2 異常値の表示
sort -mahara;
if(# < 50) outrec;
plot scat dmy_1-2;
```

(A)-1 2D 平面が点の分布でZ軸がマハラノビス距離、末端程高いマハラノビス距離になっている事が分る。

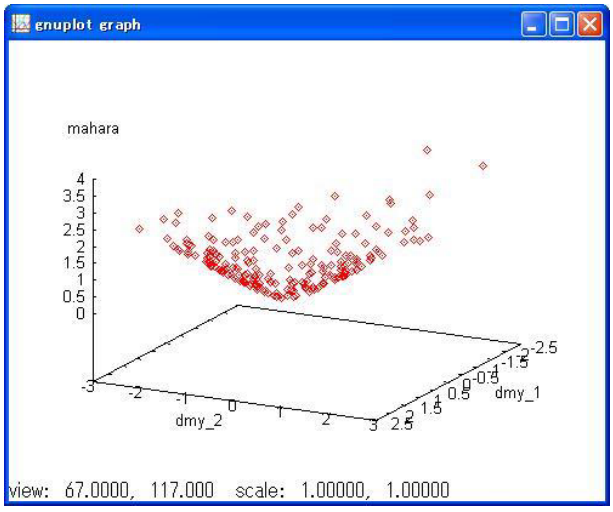


図 2.70 Result of Maharanobis distance for 3D view

(A)-2 マハラノビス距離が大きい 50 点を抽出すると異常値として周辺の点が表示される。

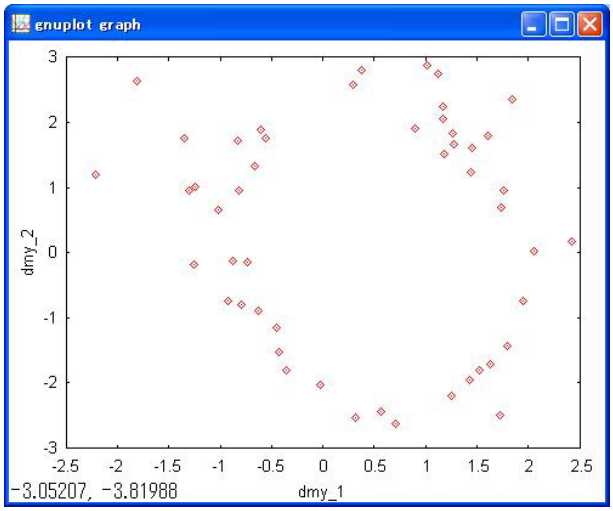


図 2.71　Result of larger Maharanobis distance 50 points for anomary value

(b)One Class SVM

SVM は一般に教師データを必要とするが、この One Class SVM は非教師型モデルである。この SVM は教師データを用いず N 次元のデータの分布で異常点がより低い確率になる様にカーネル関数の重みを最適化している。
次の例は 100 個の 2 次元のデータでは末端が異常点となる。One Class SVM でデータの異常点 (末端が高いカーネル値になる様に最適化した例である。

```
//2D データテーブルの読込
get data2d.csv@;
//100 個の点について oneClassSVM の実行
onesvm dmy_1-2/
  num=100
  eps=0.05
  loop=100
;
get freq@ana;
//(B)oneClassClassSVM の結果の表示
plot scat dmy_1-2 fx;
```

(B)3D の分布がすり鉢形になっており、末端程が高いカーネル値になっている事が分る。

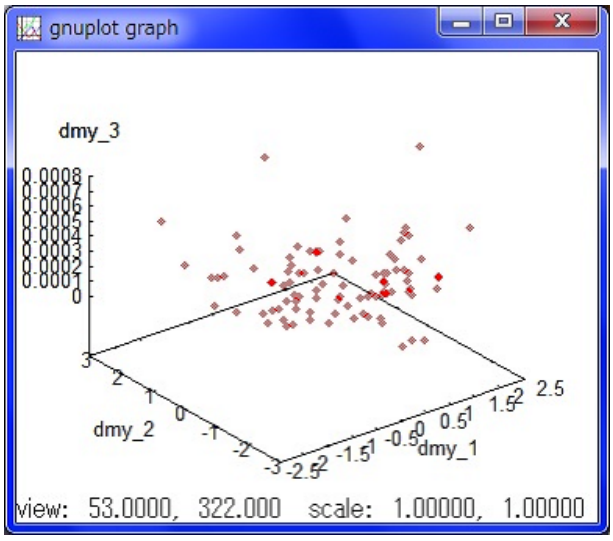


図 2.72 Result of one Class SVM for 60 points

onesvml コマンドの option

表 2.17　Options for one class SVM command

name	value	content	default
nyu	decimal	param of margin	1
num	integer	number of data	all
eps	decimal	tolerance convergency	0.005
loop	integer	maximum loop count	100

2.4 計画法

一般には計画法は、制約条件を満たしながら、最適な配置や配分を計算する。計画法には課題の設定により次の様な手法がある。

- 線形計画法 (重回帰)　最適化式・制約式も 1 次式
- 2 次計画法　最適化式が 2 次式
- 整数計画法　解が整数

2.4.1 線形計画法

課題が最適解と制約式も線形和で表されるものである。

ここに示す例は 100 ヘクタールの土地の最適開墾計画を求めるものである。次の様な制約の基で、最小の初期の所持金額で済む、開墾計画を求める。

- 開墾面積は 100 ヘクタール
- 開墾した土地は翌年から作物を植えて収入を得れる。
- 百万円の投資額により 5 ヘクタールの土地が開墾できる。
- 1 ヘクタールから農作物 25 万円の収入が得られる。
- 借入金額は銀行との約束で 5 年間で次の様な返済がある。
  - 2 年目　10 百万円
  - 4 年目　40 百万円
  - 5 年目　20 百万円

この課題は年初の最小所持金額で制約は次の様になる。

- t 年末所持金=t-1 年末の所持金　-　t 年使用金額　-　t 年返済額　+　0.25 × t-1 年末開墾面積
- t 年末開墾面積=t-1 年末開墾面積　+　5*t 年使用金額
- 5 年末開墾面積 $\geq$ 100
- t 年末所持金　$\geq$ 0
- t 年使用金額 $\leq$ t-1 年末所持金額

ここで次の値を使うと次式の最適化問題で表される。

x_0：初年度の所持金額 (百万円単位)

x_t：t 年の使用金額 (百万円単位)

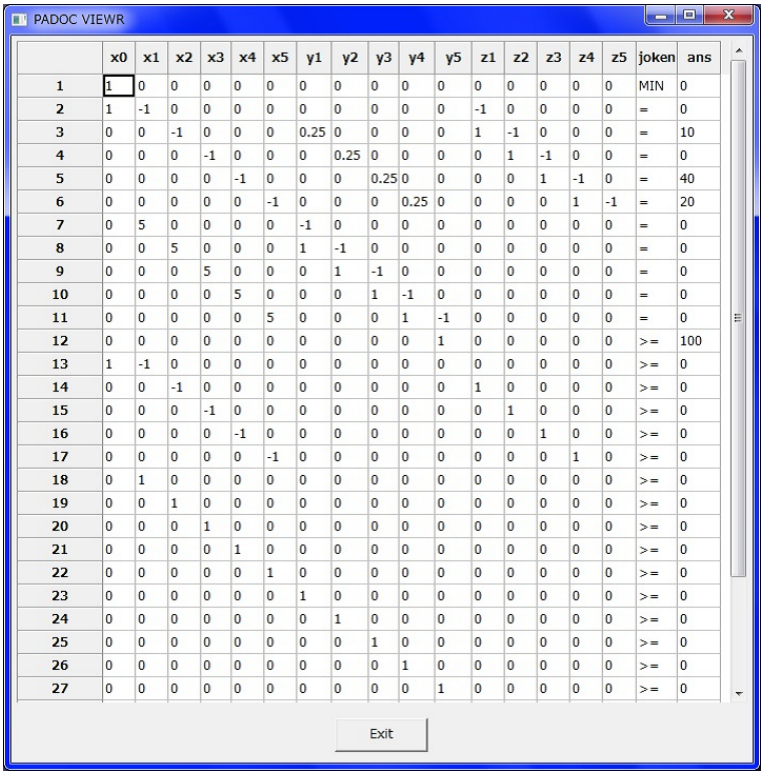
y_t：t 年末の開墾面積

z_t：t 年末の所持金額 (百万円単位)

$$
\begin{aligned}
&min\, x_0 && (2.6)\\
&s.t.\\
&z_1 = x_0 - x_1\\
&z_2 = z_1 - 10 - x_2 + 0.25y_1\\
&z_3 = z_2 - x_3 + 0.25y_2\\
&z_4 = z_3 - 40 - x_4 + 0.25y_3\\
&z_5 = z_4 - 20 - x_5 + 0.25y_4\\
&y_1 = 5x_1\\
&y_2 = y_1 + 5x_2\\
&y_3 = y_2 + 5x_3\\
&y_4 = y_3 + 5x_4\\
&y_5 = y_4 + 5x_5\\
&y_5 \geq 100\\
&x_1 \leq x_0\\
&x_{t+1} \leq z_t\, t = 1, \ldots, 5\\
&x_t \geq 0\, t = 1, \ldots, 5\\
&y_t \geq 0\, t = 1, \ldots, 5\\
&z_t \geq 0\, t = 1, \ldots, 5 && (2.7)
\end{aligned}
$$

上式 2.7 をデータ・テーブル化すると図 2.73 となる。この設定方法は以下である。

- ヘッダー　上式の変数をヘッダーとして記述する。
- 1 行目　最適化対象式である。この数値は、ヘッダーに対応する最適化式の係数値となっている。右端から 2 列目は以下の最適化の方向である。ans は 0 を設定する。
  - MIN:最小値を求める
  - MAX:最大値を求める
- 2 行目以降　ヘッダーの変数に対応する制約式の係数を設定する。　右端の 2 列は制約式の条件と値である。

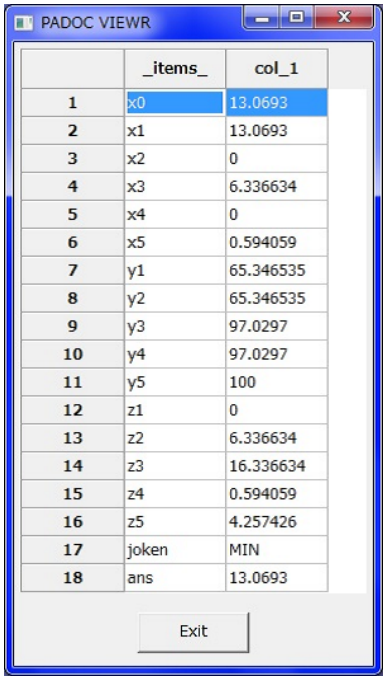


	x0	x1	x2	x3	x4	x5	y1	y2	y3	y4	y5	z1	z2	z3	z4	z5	joken	ans
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MIN	0
2	1	-1	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	=	0
3	0	0	-1	0	0	0	0.25	0	0	0	0	1	-1	0	0	0	=	10
4	0	0	0	-1	0	0	0	0.25	0	0	0	0	1	-1	0	0	=	0
5	0	0	0	0	-1	0	0	0	0.25	0	0	0	0	1	-1	0	=	40
6	0	0	0	0	0	-1	0	0	0	0.25	0	0	0	0	1	-1	=	20
7	0	5	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	=	0
8	0	0	5	0	0	0	1	-1	0	0	0	0	0	0	0	0	=	0
9	0	0	0	5	0	0	0	1	-1	0	0	0	0	0	0	0	=	0
10	0	0	0	0	5	0	0	0	1	-1	0	0	0	0	0	0	=	0
11	0	0	0	0	0	5	0	0	0	1	-1	0	0	0	0	0	=	0
12	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	>=	100
13	1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	0
14	0	0	-1	0	0	0	0	0	0	0	0	1	0	0	0	0	>=	0
15	0	0	0	-1	0	0	0	0	0	0	0	0	1	0	0	0	>=	0
16	0	0	0	0	-1	0	0	0	0	0	0	0	0	1	0	0	>=	0
17	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	1	0	>=	0
18	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	0
19	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	0
20	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	>=	0
21	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	>=	0
22	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	>=	0
23	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	>=	0
24	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	>=	0
25	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	>=	0
26	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	>=	0
27	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	>=	0

図 2.73 Input data of linear planning for land development

```
//開墾計画問題のデータテーブルの読込み
get lp_develp.csv@;
//線形計画法の実行
lp x0-5 y1-5 z1-5 by joken ans;
//結果の取出し
get freq@ana;
transpose;
```

線形計画法を上記のコマンド記述で実行すると図 2.74 の結果が得られる。初期に投資する金額は 16.07 百万円となり、$x1, \cdots, x5$ が各年で使用する金額となり $z5$ が返済後の最終所持金額となる。



	items	col_1
1	x0	13.0693
2	x1	13.0693
3	x2	0
4	x3	6.336634
5	x4	0
6	x5	0.594059
7	y1	65.346535
8	y2	65.346535
9	y3	97.0297
10	y4	97.0297
11	y5	100
12	z1	0
13	z2	6.336634
14	z3	16.336634
15	z4	0.594059
16	z5	4.257426
17	joken	MIN
18	ans	13.0693

図 2.74　Result of linear planning for land development

2.4.2 整数計画法

線形計画の解は実数だが、整数計画法は解が人数など整数に限定される。しかし入力のデータは線形計画法と同じである。padoc では分岐制限法を採用している。
表 2.18 は所謂ナップザック問題である。ナップザックの容量以内で、容量と価値が異なる物品を数個ずつ入れ、総価値が一番高くなる様にする問題である。

$$
\begin{aligned}
&max\ value \\
&\quad value = 9x_1 + 30x_2 + 21x_3 + 15x_4 + 23x_5 + 28x_6 + 7x_7 \\
&\quad s.t. \\
&\quad 34 \geq 1x_1 + 8x_2 + 6x_3 + 9x_4 + 15x_5 + 24x_6 + 21x_7 \\
&\quad 1 \geq x_t\ t = 1, \cdots, 7
\end{aligned}
\tag{2.8}
$$

表 2.18 item feature for napzack problem

variable	price	volume
x_1	9	1
x_2	30	8
x_3	21	6
x_4	15	9
x_5	23	15
x_6	28	24
x_7	7	21

padoc のコマンド記述は以下となる。

```
hand x1 - 7 cond val/
9 30 21 15 23 28 7 MAX 0
1 8 6 9 15 24 21 <= 34
1 0 0 0 0 0 0 <= 1
0 1 0 0 0 0 0 <= 1
0 0 1 0 0 0 0 <= 1
0 0 0 1 0 0 0 <= 1
0 0 0 0 1 0 0 <= 1
0 0 0 0 0 1 0 <= 1
0 0 0 0 0 0 1 <= 1
;
//整数計画法
intp x1 - 7 by cond val;
//結果の取出し
get freq@ana;
transpose;
```

整数計画法の結果での最適解は 83 であった。

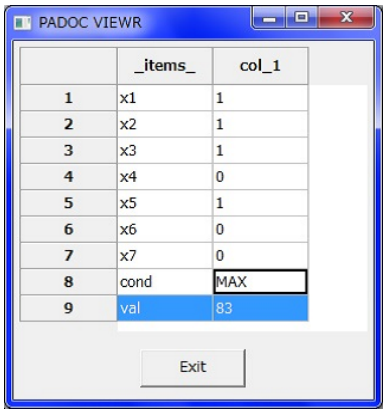


図 2.75　Result of integer planning for knapsack problem

一方線形計画法の最適解は 90.333 であり整数解でないので少し高い価値となっている。

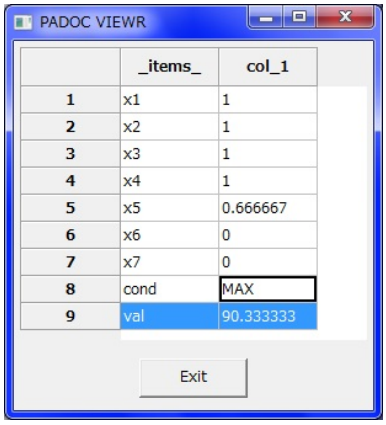


図 2.76 Result of integer planning for knapsack problem

2.4.3　2 次計画法

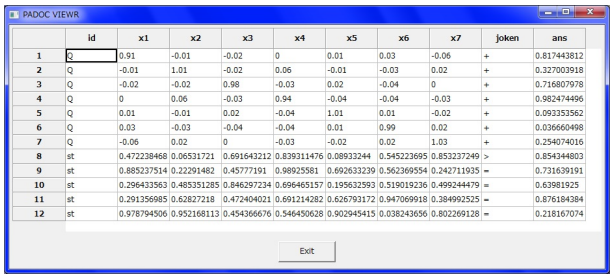
一般に最適化対象の分散を最小にする最適化である。分散は 2 次変数なので 2 次式の最適化問題となる。
分散を最小にする課題としては、ロボテックや制御装置で誤差を最小にして効率の良い動作をさせる場合や、変動する資産の組合わせの最適化に使われる場合が多い。

2 次形式の最適化式は次の様に表現でき、最初の式は n この変数 $x_1 \cdots x_n$ についての最適化式で $Q_{i,j}$ は変数間の共分散を示し一次項を持つ。2 番目の式は m 個の線形の制約式であり、等号の制約も有り得る。

$$max \begin{pmatrix} Q_{11} & \cdots & Q_{1n} \\ \vdots & Q_{i,i} & \vdots \\ Q_{n1} & \cdots & Q_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} \tag{2.9}$$

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & a_{i,j} & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \geq \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix} \tag{2.10}$$

次の例は 7 個の変数で左端の Q は相関係数 st は制約式を示し、上記の式と同じ形式で設定ができる。右端の 2 列は線形計画法と同じ形式で条件を示している。



	id	x1	x2	x3	x4	x5	x6	x7	joken	ans
1	Q	0.91	-0.01	-0.02	0	0.01	0.03	-0.06	+	0.817443812
2	Q	-0.01	1.01	-0.02	0.06	-0.01	-0.03	0.02	+	0.327003918
3	Q	-0.02	-0.02	0.98	-0.03	0.02	-0.04	0	+	0.716807978
4	Q	0	0.06	-0.03	0.94	-0.04	-0.04	-0.03	+	0.982474496
5	Q	0.01	-0.01	0.02	-0.04	1.01	0.01	-0.02	+	0.093353562
6	Q	0.03	-0.03	-0.04	-0.04	0.01	0.99	0.02	+	0.036660498
7	Q	-0.06	0.02	0	-0.03	-0.02	0.02	1.03	+	0.254074016
8	st	0.472238468	0.06531721	0.691643212	0.839311476	0.08933244	0.545223695	0.853237249	>	0.854344803
9	st	0.885237514	0.22291482	0.45777191	0.98925581	0.692633239	0.562369554	0.242711935	=	0.731639191
10	st	0.296433563	0.485351285	0.846297234	0.696465157	0.195632593	0.519019236	0.499244479	=	0.63981925
11	st	0.291356985	0.62827218	0.472404021	0.691214282	0.626793172	0.947069918	0.384992525	=	0.876184384
12	st	0.978794506	0.952168113	0.454366676	0.546450628	0.902945415	0.038243656	0.802269128	=	0.218167074

図 2.77　Input data is made same as equation

```
//2 次計画法用データテーブルの読込み
get qpl8R.csv@;
//2 次計画法の実行　収束閾値 eps 繰返回数 loop 設定
qpl x1-7 by joken ans/
  eps=0.0005
  loop=100
;
//結果の取出し
get freq@ana;
transpose;
select col_1;
put ans;
//検算
get qpl8R.csv@;
select id x1-7;
if(id == "st") outrec;
//拘束式と解を乗じて満たすか見る。
mxmult x1-7 by ans;
put ans1;
//拘束式の拘束条件と比較する
get qpl8R.csv@;
if(id == "st") outrec;
merge ans1 by #;
```

図 2.78 に 2 次計画法の解 $x_1 \cdots x_7$ の結果を示す。

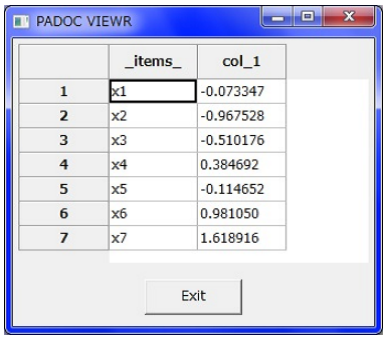


図 2.78　Result of square planning

図 2.79 の右端 3 列に示す様に拘束条件との比較すると正しく解が得られていることが分る。

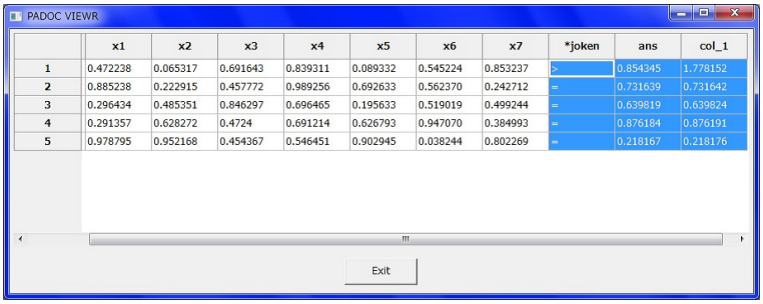


図 2.79　compalison between constraint and calculation

qpl コマンドの option

表 2.19　Options for quadratic plan command

name	value	content	default
eps	decimal	tolerance convergency	0.005
loop	integer	maximum loop count	100

2.5　時系列分析

時系列分析は大別して扱う波動が定常波と非定常波があり、padoc では以下のものがある。また時系列分析の目的は将来予測や波動の特性を検出するのが一般的である。

- 定常波解析
  - 自己相関
  - 相互相関
  - 自己回帰 AR Auto Regression
  - 移動平均
  - ADF 検定
- 非定常波
  - トレンド成分の除去
  - 季節成分の除去
  - フーリエ変換と逆変換
  - 隠れマルコフ
  - ガウシアン過程

例題として図 2.80 の F 社の株価と為替レートを使う。データのスケールは比較し易い様に平均 0、分散 1 に正規化している。

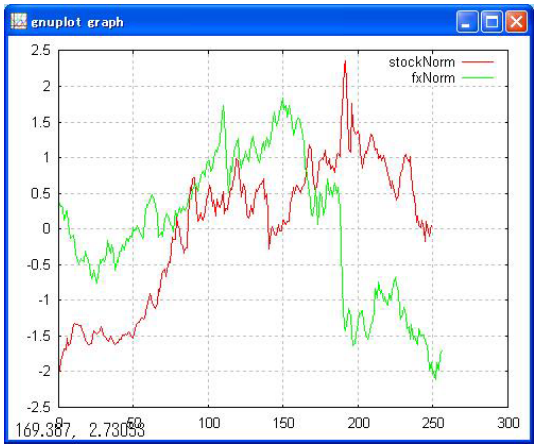


図 2.80　Stock and Fx data for time series analysis

2.5.1　定常波解析

(a) 自己相関

自己相関とは自己の波動と一期毎にずらした波動との相関を採ったものである。一般に波動に n 期毎に繰返し性があれば、 n 期毎に大きな相関が得られる。次の株価の例では自己相関は直近との相関が大きい。これは前の傾向を保持する傾向がある。再び 200 期で相関が高いので長い周期の波動であることが示されている。

```
get stock_fx.csv@;

plot line stockNorm fxNorm;  //時系列の表示

tmcov stockNorm;
get freq@ana;
plot line stockNorm;  //(A) 自己相関の結果表示
```

(A) 自己相関の結果表示

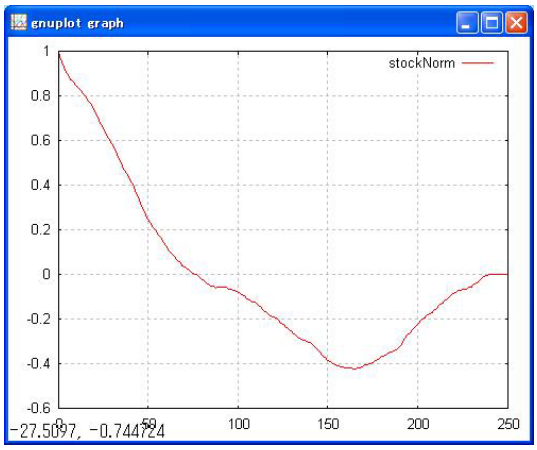


図 2.81　Result of Auto covarians of stock time series

tmcov コマンドの option

表 2.20　Options for time correlation command

name	value	content	default
kind	corr	output is correlation	cor
	cov	output is covariance	
legs	integer	span of correlation	100

(b) 相互相関

相互相関は 2 つの時系列の相関を見るもので、他方を一期毎にずらして相関を採っている。次の株価と為替の例では 50 期にピークがあるので 50 期ずれた波動であることが示されている。これは図 2.80 の波動からも確認できる。

```
get stock_fx.csv@;
tmcrs stockNorm fxNorm;
get freq@ana;
plot line stockNorm_fxNorm;  //(B) 相互相関の結果表示
```

(B) 相互相関の結果表示

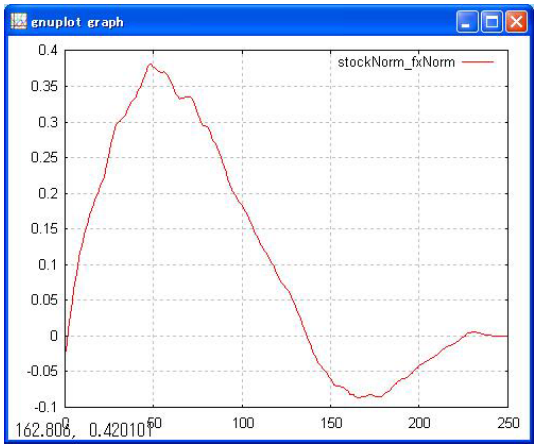


図 2.82 Result of correlation between stock and fx time series

tmcrs コマンドの option

表 2.21 Options for time inter correlation command

name	value	content	default
kind	cor	output is correlation	cor
	cov	output is covariance	
legs	integer	span of correlation	100

(c) 自己回帰

自己回帰は一般に定常波に適用され、過去の波動から現在の波動を予測するものである。自己回帰は過去の波動と現在の波動との回帰モデルである。波動の値は次式の回帰式で表される。目的変数は時刻 t での波の高さ w_t、説明変数は過去の t-1 から t-n までの波の高さ $w_{t-1} \cdots w_{t-n}$ で、$a_0 a_1 \cdots a_n$ は回帰係数である。

$$w_t = a_0 + a_1 * w_{t-1} + a_2 * w_{t-2} + \cdots + a_n * w_{t-n}$$

```
get stock_fx.csv@;
tmar stockNorm;
get freq@ana;
plot line stockNorm;  //(C) 自己回帰の表示
```

(C) 自己回帰の表示

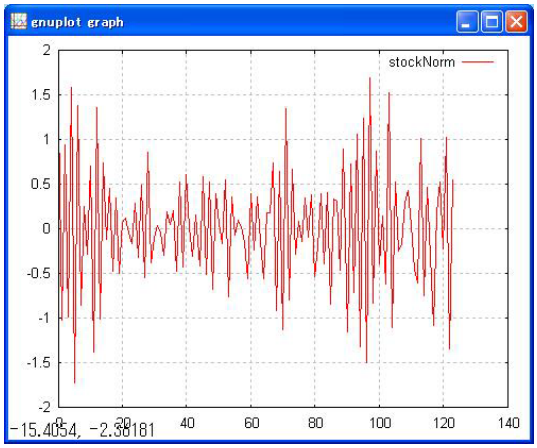


図 2.83　Result of Auto correlation between stock time series

tmar コマンドの option

表 2.22　Options for time auto regression command

name	value	content	default
size	integer	size of QR decomposition	1000
dim	integer	num of dimension	5

(d) 移動平均

移動平均は区間をずらしながらその間の平均値の値とする。移動平均は偶然で発生した波の高低を消去して滑らかにするが、長い区間での移動平均は波動の特性を失う可能性があるので注意すべきである。

```
get stock_fx.csv@;
plot line stockNorm;

tmave stockNorm/
span=20
;
get freq@ana;
over line stockNorm;  //(D) 移動平均の表示
```

(D) 移動平均の表示

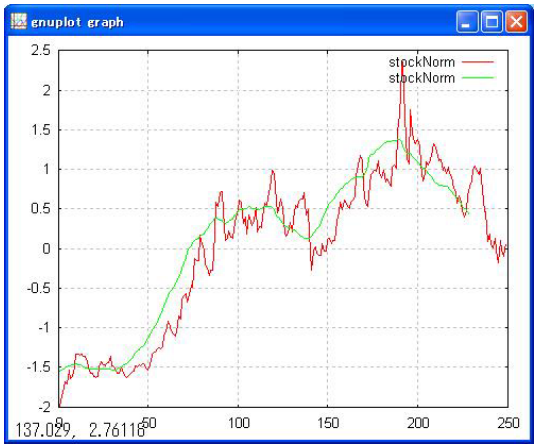


図 2.84 Comparison between smoothing 20 days and fact stock time series

tmave コマンドの option

表 2.23 Options for time average command

name	value	content	default
method	ave	smoothing method by average	ave
	samp	smoothing method by sampling span	
span	integer	samplin span for samp option	3

(e)ADF 検定

時系列解析では一般に定常時系列を扱う。ADF(Augmented Dickey-Fuller) 検定は時系列が定常か判定する検定である。定常時系列は過去の傾向を反映しない時系列である。ADF 検定は定数項、トレンド項、一次自己回帰項で線形回帰してそれらが有意であれば、定常と見做なさない。

```
//非定常時系列の読込み
get stock_fx.csv@;
//(E-1)ADF 検定で非定常である事を確認
tmadf stockNorm;
//トレンド項を除去する
tmtrend stockNorm;
get freq@ana;
//トレンド項を除去した状態を表示
plot line ;
//(E-2) 再び ADF 検定で非定常か検定
tmadf stockNormTC;
```

(E-1)ADF 検定の結果定数、トレンド、自己回帰項が有意で非定常波であることを示している。

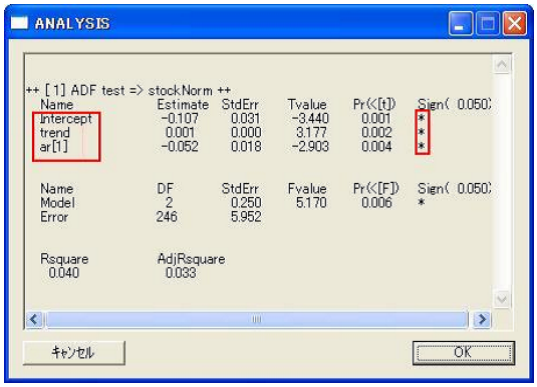


図 2.85　Result of Nonstationary time series

トレンド成分を除去した結果の表示は図 2.87 を参照

(E-2) 再び ADF 検定の結果では定数とトレンド成分が除去されていることが、自己回帰成分が残っていることが分かる。

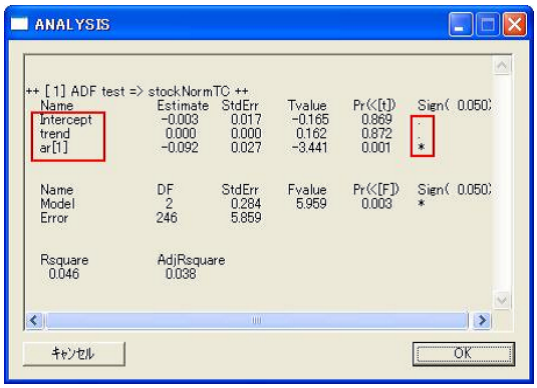


図 2.86　Result of exclusion constant and trend componet

tmadf コマンドの option

表 2.24　Options for ADF test command

name	value	content	default
plevel	decimal	significant level	0.05

2.5.2 非定常波解析

非定常波は一般に定常波にしてから、定常波で予測を行い、逆の手順で定常波を非定常波に戻す方法が採られる。

(a) トレンド成分の除去

非定常波を定常波にするためトレンドを除去する。トレンドの種類には直線や曲線があり、適切なトレンドを選択してから除去する。以下の例は 2 次曲線を当てはめ除去している。

```
get stock_fx.csv@;
tmtrend stockNorm;
get freq@ana;
plot line;  //(A) 2 次のトレンドとそれを除去した結果の表示
```

(A) 2 次のトレンドとそれを除去した結果の表示

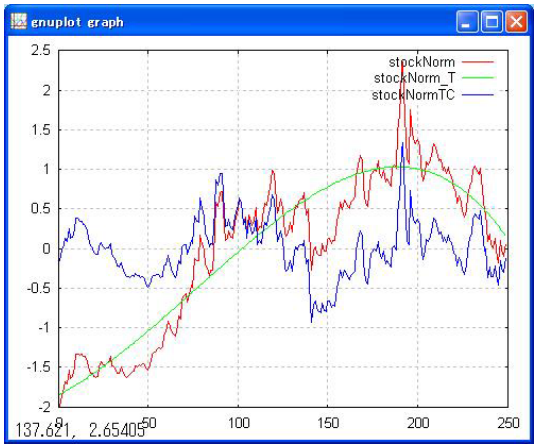


図 2.87 Result of trend componet for stock time series

tmtrend コマンドの option

表 2.25 Options for exclusion of trend command

name	value	content	default
dim	integer	dimension of trend component	5
predict	integer	span of prediction	0

(b) 季節成分の除去

一般に時系列は長期の繰返し上に定常波が乗っている場合があり、長期の波長を除去する。

```
get stock_fx.csv@;
season stockNorm;
get freq@ana;
plot line;  //(B)  季節成分の除去
```

(B) 季節成分の当てはめとそれを除去した結果

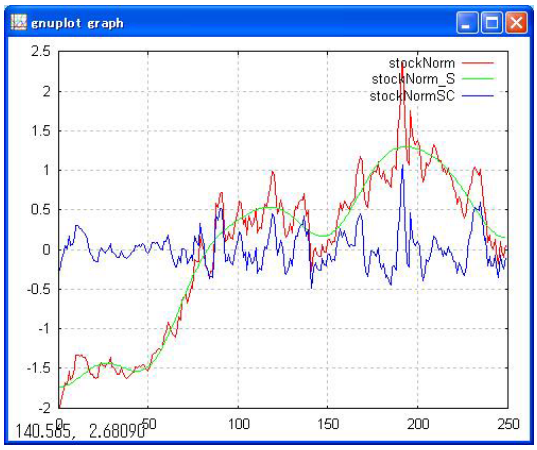


図 2.88　Result of season componet for stock time series

season コマンドの option

表 2.26　Options for exclusion of season component command

name	value	content	default
from	number	start cut frequency component	0
to	positive	end cut frequency componet	5
predict	integer	span of prediction	0

(c) 時系列の差分

一般にトレンドがある非定常波は差分を採るとトレンドが無くなることが知られている。しかし 1 回の差分では自己相関は残ることが多いので、複数回の差分を採ることがある。しかし差分を採ると重要な時系列の特性が損なわれるので注意が必要である。

```
//時系列データ・テーブルの読込み
get stock_fx.csv@;
plot line stockNorm;  //時系列の表示
//一回差分を採る
tmlag stockNorm/
lag = 1
;
get freq@ana;
over line stockNorm;  //(C)   1 回差分を採った結果を重ね合わす
```

(C)1 回差分を採った結果からトレンドが除去されていることが分る。

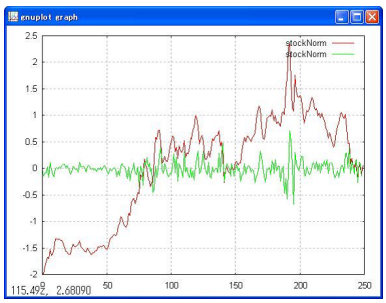


図 2.89 Result of one time difference

tmlag コマンドの option

表 2.27 Options for time lag

name	value	content	default
lag	number	number of lag differecnce	1

(d) フーリエ変換と逆変換

フーリエ変換は波動を周波数毎の強さ（スペクトル）に変換するもである。逆フーリエはスペクトルを波動に変換するもので、この変換と逆変換とで元の波動に戻ることができる。

```
get stock_fx.csv@;
//フーリエ変換
fourie stockNorm/
kind=trans
method=fourie
;

get freq@ana;
plot line;  //(D-1) フーリエ変換したスペクトル表示

fourie stockNorm/
kind=rev
method=fourie
;
get freq@ana;
plot line;  //(D-2) 逆フーリエ変換して波動に戻した結果

get stock_fx.csv@;
over line stockNorm;  //(D-3) 元の波動と逆フーリエ変換で戻した結果
```

(D-1) フーリエ変換による周波成分表示

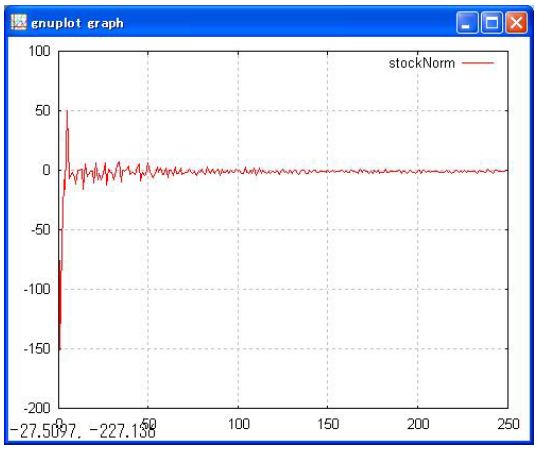


図 2.90　Result of fourie translation for stock time series

(D-2) 周波成分を逆フーリエで時系列に再変換した結果

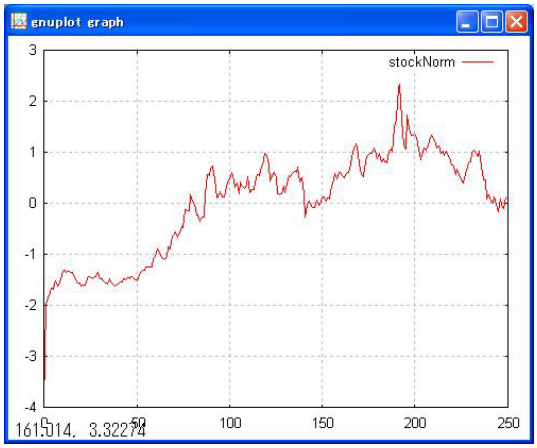


図 2.91 Result of reverse fourie translation for stock time series

(D-3) 変換前のデータと再変換した結果の重ね合わせ図、一致している事が分る。

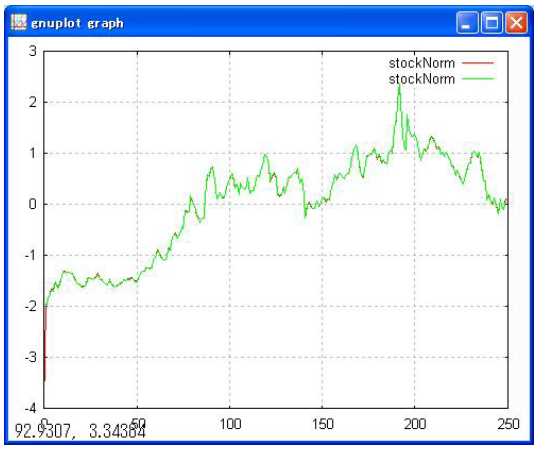


図 2.92 Comprison between true stock time series and result of reverse fourie translation

fourie コマンドの option

表 2.28 Options for fourie transfomation command

name	value	content	default
method	fourie	fourie of transformation	fourie
	fft	Fast Furie transformation	
kind	trans	wave to frequency transformation	
	period	period transformation	period
	rev	frequency to wave transformation	
dim	integer	dimension of time series	0

(e) 隠れマルコフ

隠れマルコフは図 2.93 の様に最下段にある観察値が得られた場合、これは最上段の 3 個の隠れモード α, β, γ があり、これが或る推移確率で変化し、そのモードがさらに或る出現確率で観察値になっていると仮定する。隠れマルコフは実際に得られたデータより、推移確率 M と出現確率 F 及び、初期のモード Π を推定する。

例として図 2.97 の株式推移で隠れマルコフを適用する。隠れマルコフは観測値は離散なので、図 2.97 の様に推移を 5 分類する。

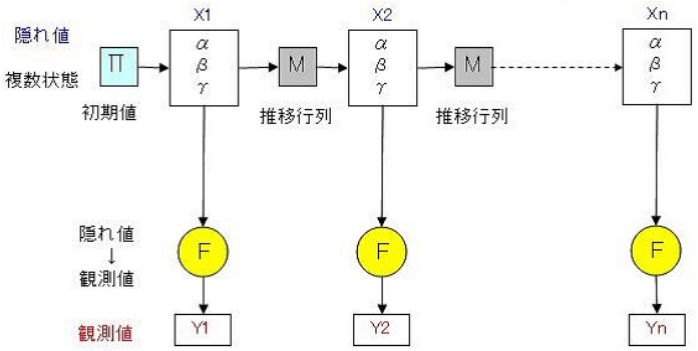


図 2.93　Hidden Marcov Model which has 3 latent mode

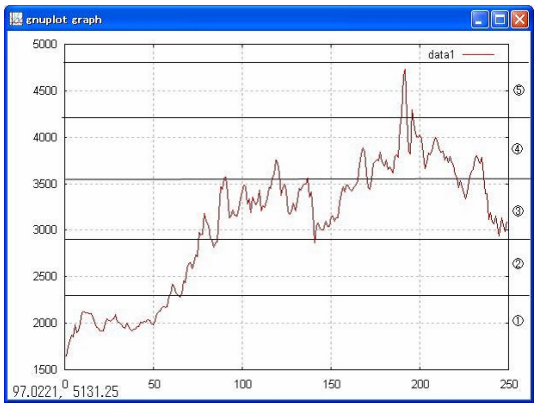


図 2.94　Division of 5 segment of stock time series

```
get stock_fx.csv@;
//(E-1)3 個の隠れ要因による隠れマルコフを実行
hmm stockNorm by 3;

get freq@ana; //(E-2) 隠れマルコフの結果の取出し
put trans0;

if(strsel(id,1,2) == "MM") outrec;
plot bar hmm_0-2 by id;  //(E-3) 隠れ要因の推移確率の表示

get trans0;
if(strsel(id,1,2) == "TR") outrec;
put trans;

plot bar hmm_0-2 by id;  //(E-4) 隠れ要因と現象の出現確率の表示
```

(E-1) 隠れマルコフの実行の結果、図 2.95 と図 2.96 が得られた。図 2.95 に示す様に、次の結果が得られている。

- Π 初期の隠れモード
- M 推移確率行列
- F 隠れモード毎の出現確率

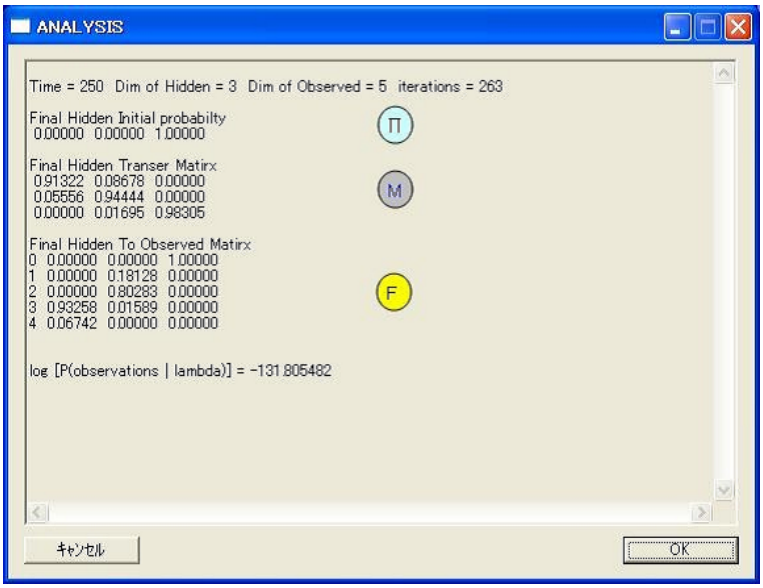
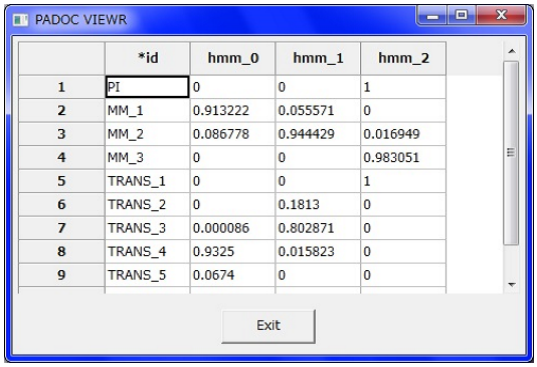


図 2.95 Result of Hidden Marcov

(E-2) 隠れマルコフの結果の取出し



	*id	hmm_0	hmm_1	hmm_2
1	PI	0	0	1
2	MM_1	0.913222	0.055571	0
3	MM_2	0.086778	0.944429	0.016949
4	MM_3	0	0	0.983051
5	TRANS_1	0	0	1
6	TRANS_2	0	0.1813	0
7	TRANS_3	0.000086	0.802871	0
8	TRANS_4	0.9325	0.015823	0
9	TRANS_5	0.0674	0	0

図 2.96　Result of Hidden Marcov for csv file

(E-3) 隠れモードの推移行列のグラフ　隠れモードは他のモードに移らない事を示しており、図 2.97 の時系列推移の平滑な推移が多い事と矛盾しない。

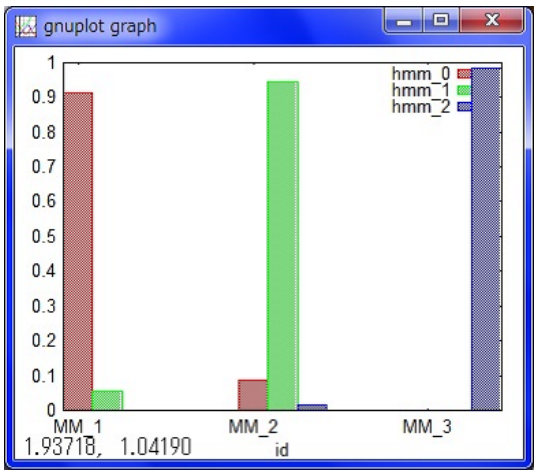


図 2.97　Division of 5 segment of stock time series

(E-4) 各隠れモードと現象との出現確率

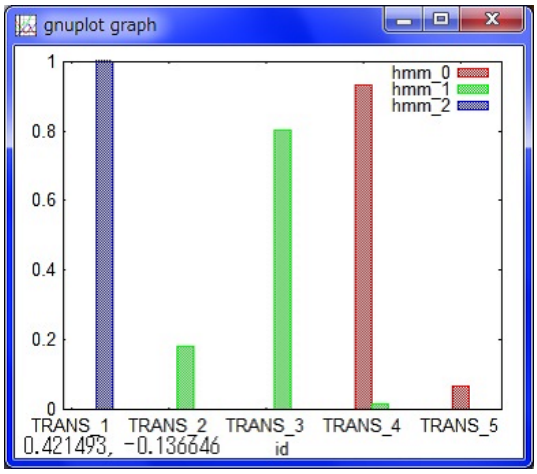


図 2.98　Division of 5 segment of stock time series

(f) ガウス過程

ガウシアン過程 (Gaussian Process) は図 2.99 の様な間隔が一定でない点過程が得られた場合、その間をガウス分布の推移として補間する。

一般に N 次元空間上で実数値の点過程が得られた場合、その間を補填は複雑な補間が必要だが、ガウス過程は容易に補間することができる。

```
//点過程の読込み
hand t stock20/
1 -1.992561
21 -1.620277
41 -1.591967
61 -1.046988
81 0.057125
101 0.495939
181 0.892287
201 1.373567
221 0.76489
241 0.184523
;
//(F-1) 点過程の表示
plot scat t stock20;
//ガウス過程の実行
gproc stock20 by t;
get freq@ana;
//(F-2) ガウス過程の重ね図表示
over line gaussUp gaussProc gaussLow by t;
```

(F-1) 点過程の表示　中央の点過程が抜け落ちて、間が広がっている。

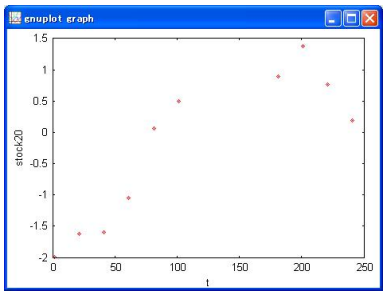


図 2.99　Point process of 10point

(F-2) ガウス過程の重ね図表示

図 2.100 からガウス過程は点過程を適切に補間していることがわかる。また抜け落ちた中央は信頼区間が増幅していることが分かる。

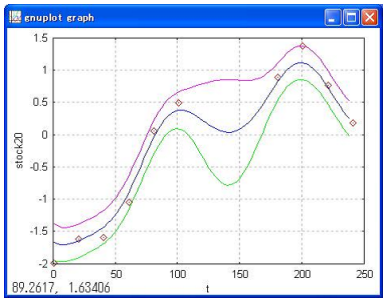


図 2.100　Result of Gaussian Process

(注 1) ガウス過程は次の 2 点 u と v のガウスカーネルで共分散を計算している。

$$kernel(u,v) = \theta \exp\{-\frac{\sigma}{2}(u-v)^2\} - \gamma - \delta * u^T v$$

(注 2) 逆行列計算をしているので、ランク落ち防止のため $\frac{1}{\beta}$ のノイズを入れている。

gproc コマンドの option

表 2.29　Options for gauss process

name	value	content	default
theta	positive	θ of kaernel	0.33
sigma	positive	σ of kernel	25.0
gamma	positive	γ of kernel	1
delta	positive	δ of kernel	0.03
beta	positive	additional noise for inverse matrix	12
num	integer	number of gaussian process length	100

2.6 自然言語分析

padoc では自然言語分析としては次のものがある。

- 形態要素解析
- LDA(Latent dirichlet allocation)

2.6.1 形態要素解析

図 2.101 の様な日本語文が複数存在するディレクトリィと拡張子を指定して、該当する複数の文書について形態要素解析を行う。

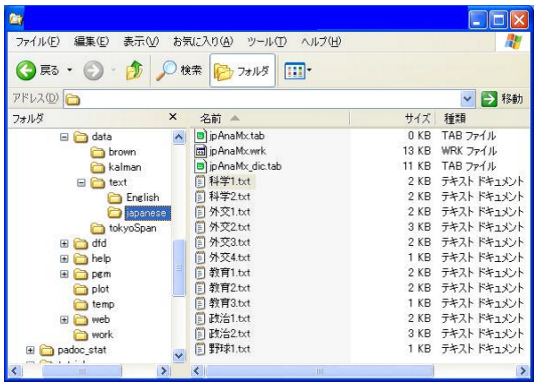


図 2.101 Input directory and extender for mecab

```
//ディレクトリィと拡張子を指定して形態素解析を行う
//filter オプションは抜き出す品詞を指定する。この場合は名詞と形容詞が対象となっている。(option 記述参照)
mecab text\japanese\*.txt@/
    filter=noun
    filter=adjective
    anaout=1
;
//(A) 結果の取出し
get freq@ana;
```

図**??**の形態素解析の結果を見るには、上記の記述の様に $anaout = 1$ を指定する必要があるが、大量に出力されるので注意が必要である。

図 2.102　Result of morphological analysis

(A) 形態素解析の結果は、抜き出した品詞の単語について文書毎の頻度テーブルとなる。

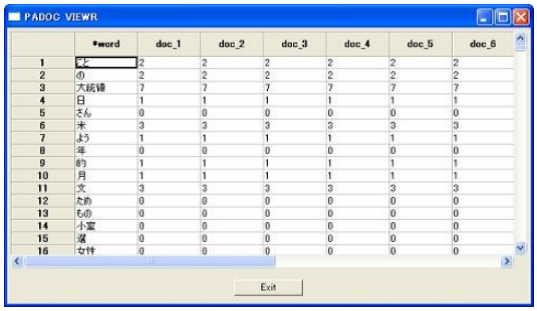


図 2.103　Result of frequency of word by each document

mecab コマンドの option

表 2.30　Options for mecab command

name	value	content	default
filter	noun	extract noun	noun
filter	verb	extract verb	
filter	adjective	extract adjective	
filter	auxiliary	extract auxiliary	
count	integer	minmum frequecy to be outputed	1
anaout	0 1	no morphological result output morphological result	0

(注 1) filter option は例にある様に複数指定が可能

2.6.2 LDA(Latent dirichlet allocation)

LDA は文章には隠れたクラスがディレクテットの分布で存在することを仮定して、文書群の所属するクラスの確率と各単語の所属するクラスの確率を求めるものである。LDA は図 2.104 の様に文書と単語の頻度テーブルが用いられる。この単語、文書テーブルは日本語、英語を問わず mecab コマンドで生成することができる。

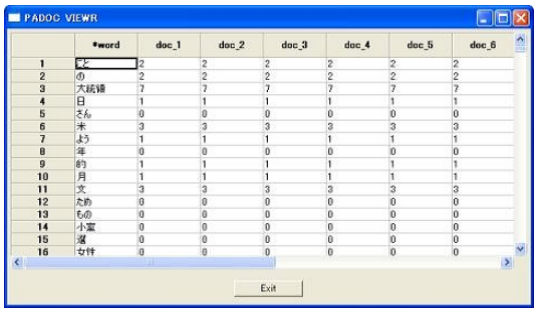


図 2.104 Result of frequency of word by each document

```
//単語・文書頻度テーブルの生成
mecab text\japanese\*.txt@/
 filter=noun
;
//LDA の 12 の文書について実行
lda doc_1-12/
nclass=10
;

get freq@ana;     //(A) 隠れクラスへの所属確率を表示
get score@ana;  //(B) 単語の発生確率を表示
```

(A) 計算結果として、隠れクラスへの所属確率の結果

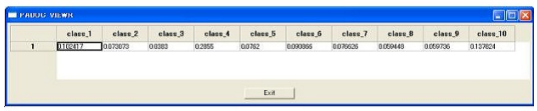


図 2.105 Result of probability of latent class

(B) 隠れクラス毎の単語の発生確率の結果

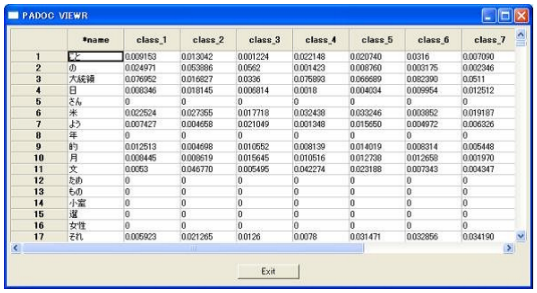


図 2.106　Result of probability of word by each latent class

lda コマンドの option

表 2.31　Options for lda command

name	value	content	default
nclass	integer	number of latent class	50
emmax	integer	maximum loop count for VB-EM logic	100
demmax	integer	maxmum inner loop for VB-EM logic	20

2.7　行列計算

padoc では行列分解として以下のものがある。

- 非負値分解
- 特異値分解 (SVD)
- コレスキー分解
- LU 分解
- QR 分解

行列演算として以下のものがある。

- 逆行列
- 固有値
- 行列式
- 行列の加減演算

2.7.1　行列分解

(a) 非負値分解

購買履歴データより推薦商品を選別する場合、同様な商品を買っている人を見つけ、未購買かつ売れ筋を推薦するのが有効と考えられる。しかし実際の人と購買商品の行列ガ巨大過ぎるため、殆ど 0 か欠損となって所謂スパース (疎) データとなって分析が難しい。 そこで行列を分解する手法が効果的であるが、特に非負値分解は行列の特性を保存することが知られている。

次の例は図 2.107 の 20 × 8 の行列を非負値分解で 20 × 4 と 4 × 20 の行列に分解して、それらを掛け合わして基に戻るか試したものである。

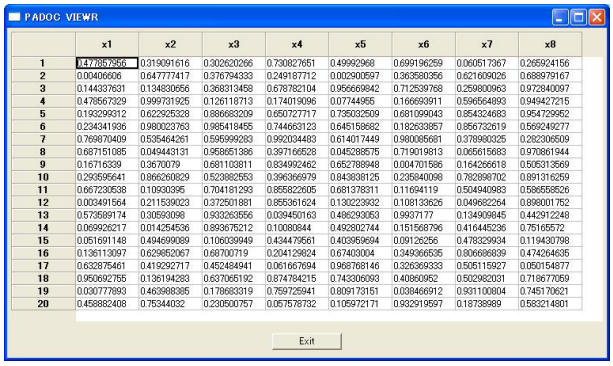


図 2.107　Input data for non negative matrix factarization

```
//データ・テーブルの読込
get nfmx20_8.csv@;
//非負値分解 4 列（20*4）と 4 行（4*20）の 2 つの行列に分解
nfmx x1-8 by 4/
  loop=1000
  kind=KL
;

get score@ana;   //(A-1) 分解した結果 4*20 の行列の取出し
put m4_8;

get freq@ana; //(A-2) 分解した結果 20*4 の行列の取出し
put m20_4;

//(A-3) 分解結果を掛け合わせて元に戻るか検算
get m20_4;
mxmult base0-3 by m4_8;
put m20_8;
//(A-4) 相関係数の表示
corr x1-8;

//(A-5) 元の相関係数の表示
get nfmx20_8.csv@;
corr x1-8;
```

(A-1) 非負値分解 20 × 4 行列の表示

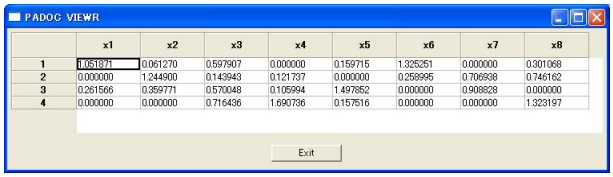
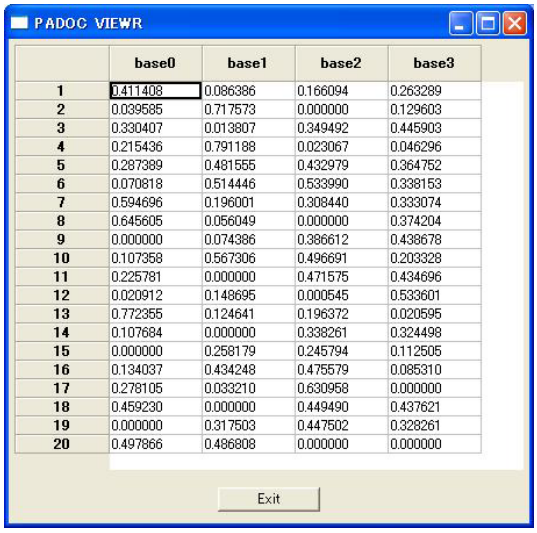


図 2.108　Result of 20 × 4 factarized matrix

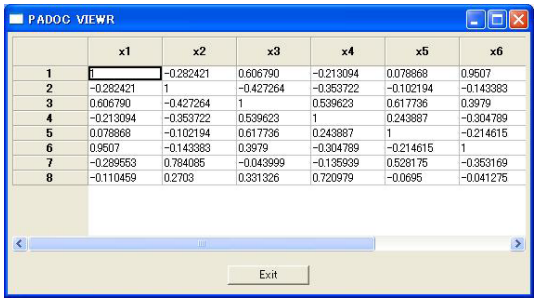
(A-2) 非負値分解 4 × 20 行列の表示



	base0	base1	base2	base3
1	0.411408	0.086386	0.166094	0.263289
2	0.039585	0.717573	0.000000	0.129603
3	0.330407	0.013807	0.349492	0.445903
4	0.215436	0.791188	0.023067	0.046296
5	0.287389	0.481555	0.432979	0.364752
6	0.070818	0.514446	0.533990	0.338153
7	0.594696	0.196001	0.308440	0.333074
8	0.645605	0.056049	0.000000	0.374204
9	0.000000	0.074386	0.386612	0.438678
10	0.107358	0.567306	0.496691	0.203328
11	0.225781	0.000000	0.471575	0.434696
12	0.020912	0.148695	0.000545	0.533601
13	0.772355	0.124641	0.196372	0.020595
14	0.107684	0.000000	0.338261	0.324498
15	0.000000	0.258179	0.245794	0.112505
16	0.134037	0.434248	0.475579	0.085310
17	0.278105	0.033210	0.630958	0.000000
18	0.459230	0.000000	0.449490	0.437621
19	0.000000	0.317503	0.447502	0.328261
20	0.497866	0.486808	0.000000	0.000000

図 2.109　Result of 4 × 20 factarized matrix

(A-3) 分解した結果を掛け合わせて元に戻るかの確認



	x1	x2	x3	x4	x5	x6
1	1	-0.282421	0.606790	-0.213094	0.078868	0.9507
2	-0.282421	1	-0.427264	-0.353722	-0.102194	-0.143383
3	0.606790	-0.427264	1	0.539623	0.617736	0.3979
4	-0.213094	-0.353722	0.539623	1	0.243887	-0.304789
5	0.078868	-0.102194	0.617736	0.243887	1	-0.214615
6	0.9507	-0.143383	0.3979	-0.304789	-0.214615	1
7	-0.289553	0.784085	-0.043999	-0.135939	0.528175	-0.353169
8	-0.110459	0.2703	0.331326	0.720979	-0.0695	-0.041275

図 2.110　Result of multified matrixs

(A-4) 非負値分解の行列を掛け合わせた行列の $x2$ 相関係数の表示

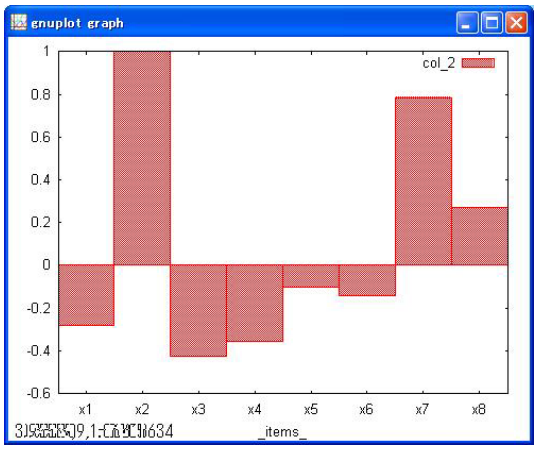


図 2.111　Correlation of x2 and others by multified matrix

(A-5) もとの行列の $x2$ 相関係数の表示。図 2.111 の結果と殆ど同じであり、正しく復元していることが分る。

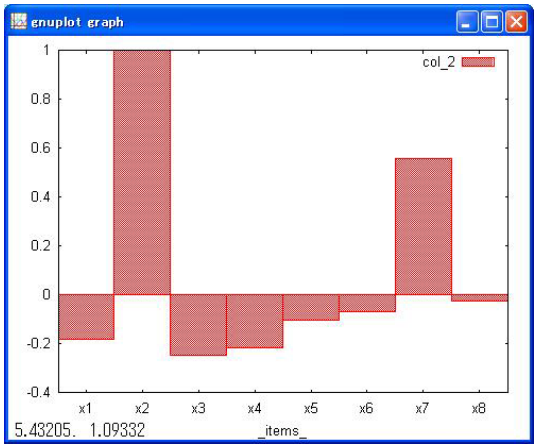


図 2.112　Correlation of x2 and others by original matrix

nfmx コマンドの option

表 2.32　Options for non-negative matrix factarization command

name	value	content	default
kind	EU	Eucid distance	
	KL	Kullback-Leibler divergence	KL
	IS	ItakuraSaito pseudo distance	
loop	integer	loop count	1000

(注 1) 非負値分解の近似距離の定義は以下である。

- EU $(a||b) = (a-b)^2$
- KL $(a||b) = a\log\frac{a}{b} + b - a$
- IS $(a||b) = \frac{a}{b} - \log\frac{a}{b} - 1$

(b) 特異値分解 (SVD)

一般に特異値分解は図 2.113 にある様にN×Mの矩形の行列を 3 個の行列に分解する。中央の Σ 行列は方型であり、この行列の次元を減じて次元圧縮に使われる。(主成分分析も次元圧縮に使えるが、こちらは方型の行列しか適用できない)

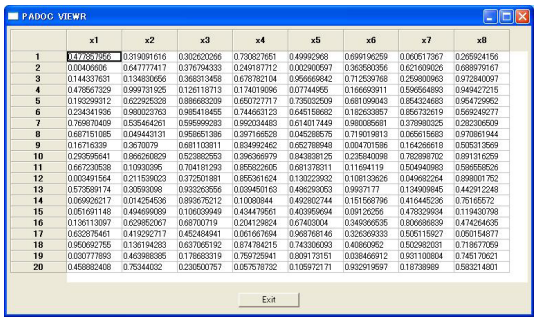


図 2.113　Explaination for Singular value decomposition(SVD)

次の記述は図**??**の行列を特異値分解し、Σ の次元を 4 に減じて、再び合成して同じ行列が得られるか試したものである。

```
get nfmx20_8.csv@;
put nfx;
//特異値分解の実行
svd x1-8/
output:uwv
;
//(B-1) 結果の取出し
get freq@ana;
put vwu;
```

(B-1) 特異値分解した 3 の行列は図 2.114 で得られる。ここで第 1 列の識別は次の通りである。

- w： Σ　行列の対角成分
- u： U 8 × 8 行列の結果
- v： V 20 × 8 行列の結果

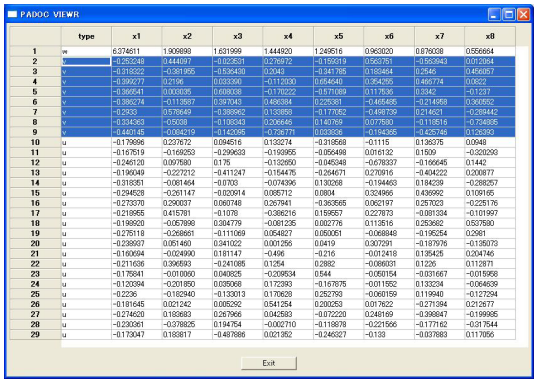


図 2.114 Result of Singular value decomposition(SVD)

特異値分解した結果を使って以下に次元を下げた結果について見当する。
対角成分を 8 次元から 4 次元にする。

```
get vwu;
//対角行列を 8 から 4 次元に減じる
select type x1-4;

//対角化
if(type == "w") outrec;
n=4;

vector w[n]=x1-4;
for(i=1;i<=n;i++) outrec;
for(i=1;i<=n;i++) {
  if(i != #) w[i]=0.0;
}
select w_1-4;
put w; //(B-2)4 次元の対角行列
```

(B-2) 4 次元に圧縮した Σ 行列の結果

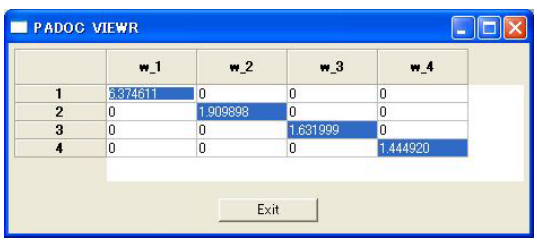


図 2.115 reduced 4 dimension of diagonal matrix

特異値分解した結果で次元を 8 から 4 に下げた結果について検討する。

```
//V 行列の転置
get vwu;
if(type == "v") outrec;
select x1-4; //8 から 4 次元に減じる
transpose;
select col_1-8;
put v;   //svd4

//(B-3)4 次元の対角行列× U 行列
get w;
mxmult w_1-4 by v;
put wv;
```

(B-3)4 次元に圧縮した *Sigma* 行列とV行列の積の結果

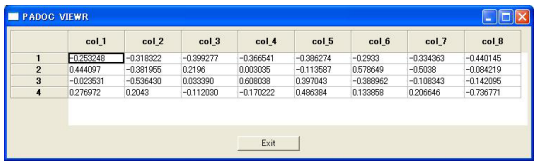


図 2.116　Result of mulitiplied diagonal matorix and V matorix

```
//(B-4) U×対角行列×V行列で元の　20 ×　8 の行列を復元
get vwu;
if(type == "u") outrec;
mxmult x1-4 by wv;
put svdx;  //svd6

//復元行列の相関係数計算
vector z[8]=col_1-8;
corr;  //相関係数計算
get freq@ana;
transpose;
plot bar col_2 by _items_;  //(B-5)x2 と他の相関係数の表示

//元の行列の再読込
get nfmx20_8.csv@;
corr; //相関係数計算
get freq@ana;
transpose;
plot bar col_2 by _items_; //(B-6)x2 と他の相関係数の表示
```

(B-4)4 次元に圧縮した Σ 行列でU×　Σ ×　V で元に戻した結果

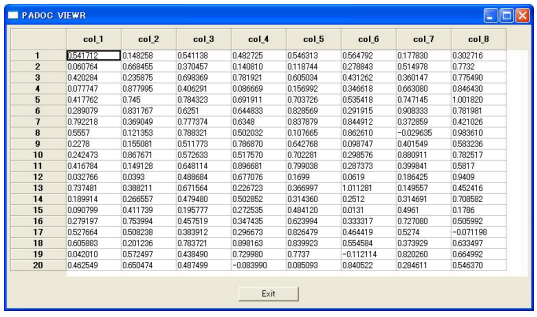


図 2.117　Result of restored matrix by reduced diagonal matrix

(B-5)4 次元に圧縮した Σ 行列で元に戻した行列の x2 とその他の相関係数の結果

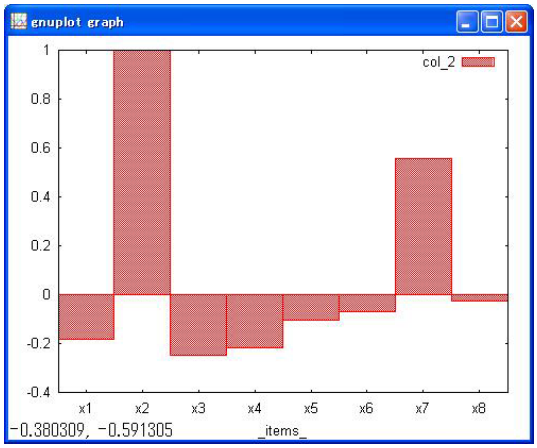


図 2.118　Correlation of x2 between others in restored matorix

(B-6) 元の行列の相関係数の結果とほぼ近似しており、Σ 行列を半分にしても元に近似できることが分る。

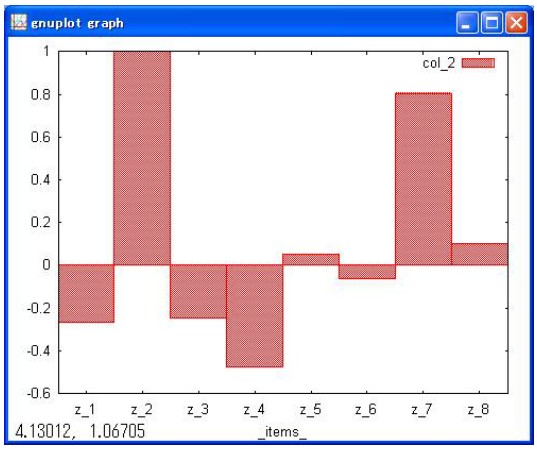


図 2.119　Correlation of x2 between others in original matorix

svd コマンドの option

表 2.33　Options for singure value decomposition command

name	value	content	default
output	rev	solution for pseudo reverse matrix	rev
	uwv	decompositon UwV^T matrix	

(c)LU 分解

LU 分解は正方行列を次の様に分解する。

$$U = \begin{pmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & uii & \vdots \\ 0 & \cdots & u_{nn} \end{pmatrix} \tag{2.11}$$

$$L = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & 1 & \vdots \\ l_{n1} & \cdots & 1 \end{pmatrix} \tag{2.12}$$

$$A = LU \tag{2.13}$$

連立方程式は逆行列が必要となるが、LU 分解を使うと次式となる。L と U は下三角行列　上三角行列なので $U\boldsymbol{x}$ と $L\hat{\boldsymbol{x}}$ は簡単な四則演算で計算できる。

$$\begin{aligned} A\boldsymbol{x} &= LU\boldsymbol{x} = y \\ U\boldsymbol{x} &= \hat{\boldsymbol{x}} \\ A\boldsymbol{x} &= L\hat{\boldsymbol{x}} = y \end{aligned} \tag{2.14}$$

次の編集記述では図 2.120 の行列について $x_1 \cdots x_7$ の連立方程式を解き、検算をしてみる。

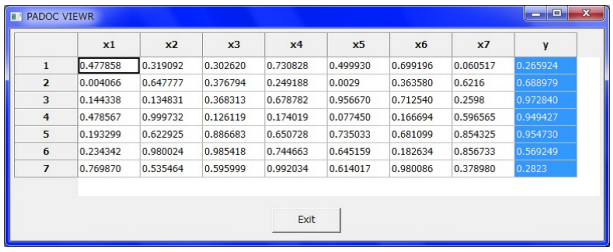
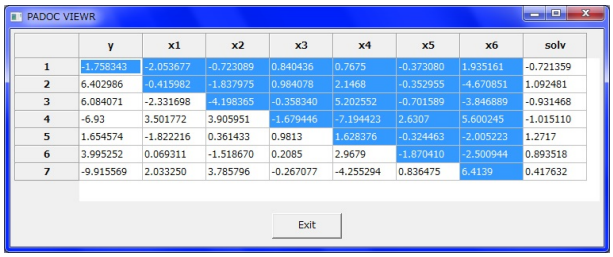


図 2.120　Input for LU composition

```
//連立方程式のデータ・テーブルを読込む
get lu8.csv@;
//LU 分解を行う
ludecomp x8 by x1-8;
get freq@ana;
select solv; //(C-1) 解を取出す
put solv;

//検算
get lu8.csv@; //再度連立方程式のデータテーブルを読込む
select x1-7;
//連立方程式の係数と LU 分解の解で y を算出
mxmult x1-7 by solv;
put luans;  //(C-2) 算出 y が一致している
```

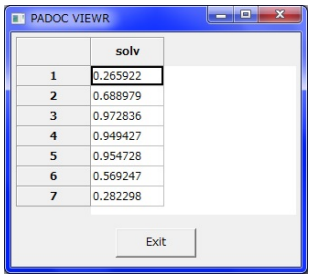
(C-1) 図 2.121 はＬＵ分解の結果で上下三角行列で分解されている。右端は連立方程式 $x_1 \cdots x_7$ の解である。



	y	x1	x2	x3	x4	x5	x6	solv
1	-1.758343	-2.053677	-0.723089	0.840436	0.7675	-0.373080	1.935161	-0.721359
2	6.402986	-0.415982	-1.837975	0.984078	2.1468	-0.352955	-4.670851	1.092481
3	6.084071	-2.331698	-4.198365	-0.358340	5.202552	-0.701589	-3.846889	-0.931468
4	-6.93	3.501772	3.905951	-1.679446	-7.194423	2.6307	5.600245	-1.015110
5	1.654574	-1.822216	0.361433	0.9813	1.628376	-0.324463	-2.005223	1.2717
6	3.995252	0.069311	-1.518670	0.2085	2.9679	-1.870410	-2.500944	0.893518
7	-9.915569	2.033250	3.785796	-0.267077	-4.255294	0.836475	6.4139	0.417632

図 2.121　Result of LU composition

(C-2) 図 2.122 は LU 分解の解で $A\boldsymbol{x} = y$ を求めた結果である。入力の行列の y 列と同じであることが分る。



	solv
1	0.265922
2	0.688979
3	0.972836
4	0.949427
5	0.954728
6	0.569247
7	0.282298

図 2.122　Test of LU composition

(d)QR 分解

QR 分解は N × M の行列を直交行列Qと上三角行列Rとに分解する。

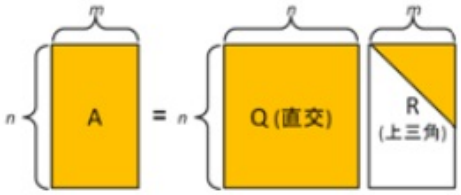


図 2.123　Explaination for QR decomposition

次の例では図 2.124 の 20 × 8 の行列を QR 分解する

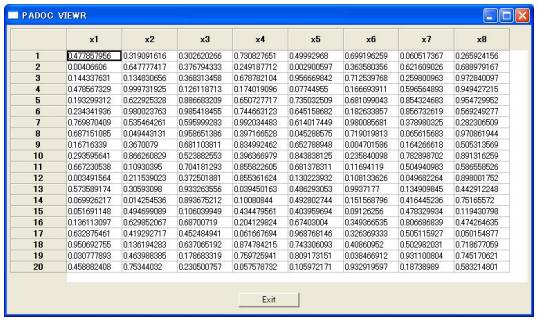


図 2.124　Input data for QR decomposition

次の例は 20 × 8 の行列を QR 分解して、Q 直行行列が直行しているか確かめる。

```
//20 × 8 の行列を読込む
get nfmx20_8.csv@;
//(D-1)QR 分解を行う
qrdecomp x1-8;
get freq@ana;
//QR 分解の逆行列を計算する
mxinv x1-8;
get freq@ana;
put iqr;
//QR 分解の逆行列を元のデータに掛けて Q 直行行列を求める
get nfmx20_8.csv@;
mxmult x1-8 by iqr;

corr x1-8;   //(D-2)Q 直行行列の相関係数を表示
```

(D-1)QR 分解した上三角行列の結果

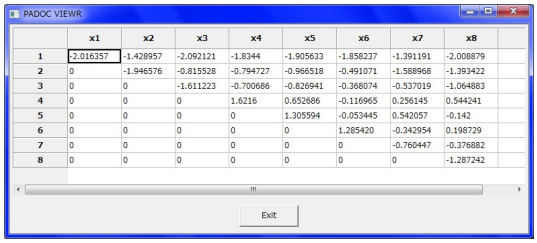


図 2.125　Result of QR decomposition

(D-2)Q 直行行列の相関係数は対角のみ 1 なので直行していることが分かる。

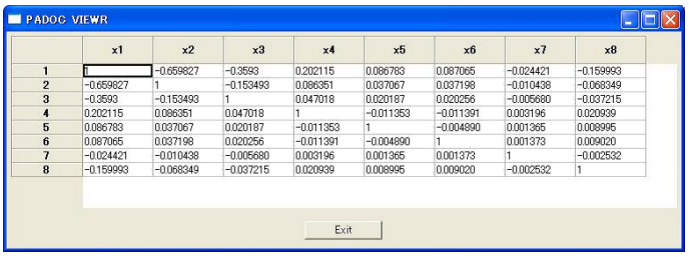
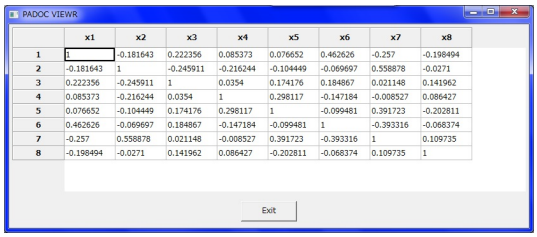


図 2.126　Result of QR decomposition

(e) コレスキー分解

コレスキー分解は相関を保存した正規乱数の発生に用いられる。図 2.127 の 20 × 8 の相関行列をコレスキー分解する。正規乱数を 1000 個作成し、これにコレスキー分解を乗ずることにより同じ相関行列を持つ正規乱数を 1000 個を生成する。



	x1	x2	x3	x4	x5	x6	x7	x8
1	1	-0.181643	0.222356	0.085373	0.076652	0.462626	-0.257	-0.198494
2	-0.181643	1	-0.245911	-0.216244	-0.104449	-0.069697	0.558878	-0.0271
3	0.222356	-0.245911	1	0.0354	0.174176	0.184867	0.021148	0.141962
4	0.085373	-0.216244	0.0354	1	0.298117	-0.147184	-0.008527	0.086427
5	0.076652	-0.104449	0.174176	0.298117	1	-0.099481	0.391723	-0.202811
6	0.462626	-0.069697	0.184867	-0.147184	-0.099481	1	-0.393316	-0.068374
7	-0.257	0.558878	0.021148	-0.008527	0.391723	-0.393316	1	0.109735
8	-0.198494	-0.0271	0.141962	0.086427	-0.202811	-0.068374	0.109735	1

図 2.127　Correlation of 20 × 8 行列

```
//20 × 8 の行列を読込む
get nfmx20_8.csv@;
corr; //相関行列作成
get freq@ana;
plot bar x2;  //(E-3) 相関行列のグラフ表示
//(E-1) コレスキー分解
chole x1-8;
get freq@ana;
put chole;

//8 次元 1000 個の正規乱数作成
clear;
vector v[8];
for(i=1;i<=1000;i++) {
  for(j=1;j<=8;j++) {
     v[j]=ranstd(0,1); //正規乱数
  }
  outrec;
}
put rand;
plot hist v_1; //(E-2)1000 個の正規乱数

get rand;
//生成した正規乱数にコレスキー行列を掛ける
mxmult v_1-8 by chole;

corr; //相関行列を計算
get freq@ana;
plot bar x2; //(E-4) 生成したデータの相関行列のグラフ表示
```

(E-1) 図 2.128 はコレスキー分解した行列

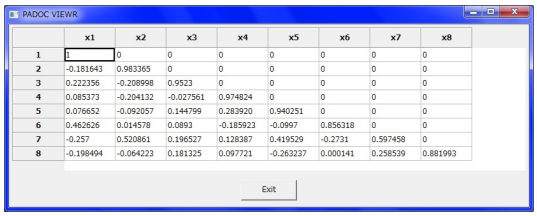


図 2.128 Result of Cholesky decomposeion

(E-2)1000 個生成した乱数の x_1 のヒストグラム

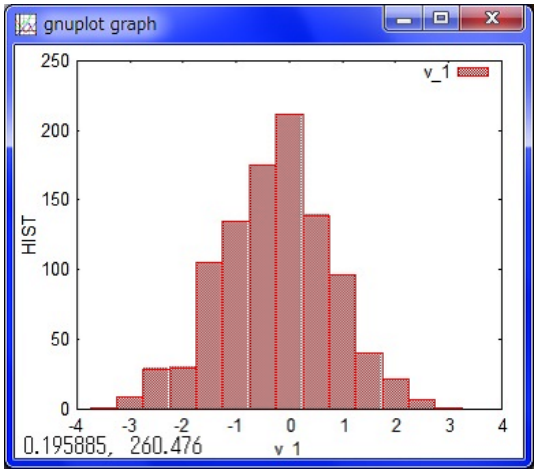


図 2.129　Histgram of generated 1000 record normal distribution

(E-3)20 × 8 行列の x_2 とその他の相関の表示

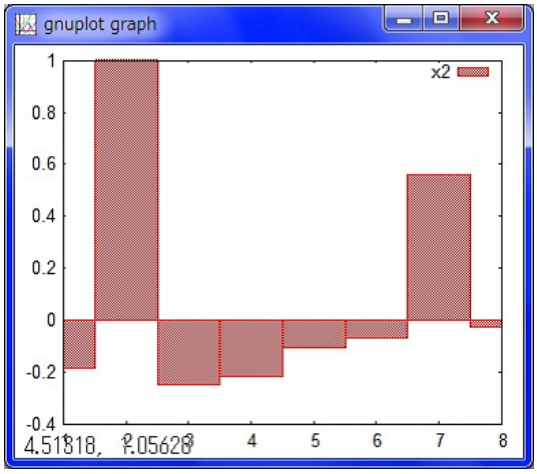


図 2.130　Correlation of x2 between others in original data

(E-4)1000 個生成した x_2 とその他の相関の表示、図 2.130 と同様の相関が得られていることが分る。

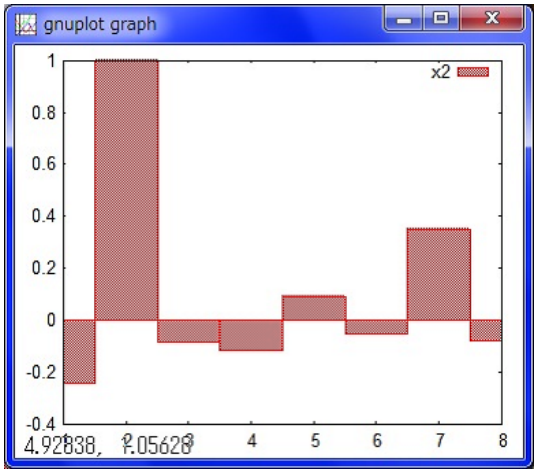


図 2.131 Correlation of x2 between others in generated data by Cholesky decoposition

2.7.2 行列演算

(a) 逆行列

次の記述は正方行列の逆行列を求め、両行列を掛け合せることで単位行列になっているか確かめる。

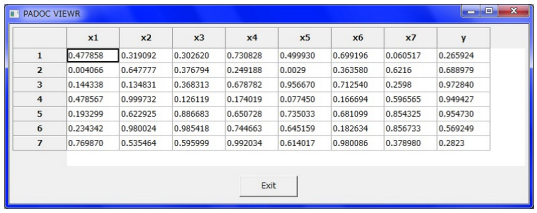


図 2.132 input matrix for inverse matrix

```
//7 × 7 の行列の読込み
get lu8.csv@;
//(A-1) 逆行列計算
mxinv x1-7; //1 から 7 列まで使う
get freq@ana;
put inv;

//7*7 行列の読込み
get lu8.csv@;
select x1-7;
put org;

//(A-2) 逆行列×正方行列
get inv;
mxmult x1-7 by org;
```

(A-1) 図 2.133 は計算された逆行列

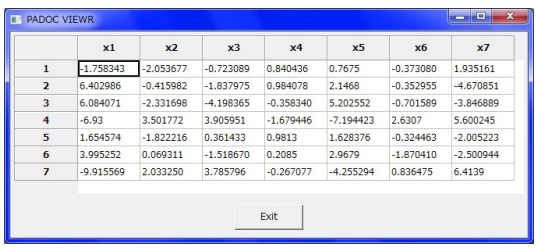


図 2.133 Result of inverse matrix

(A-2) 逆行列に元の行列を掛けて単位行列になるか確かめた結果

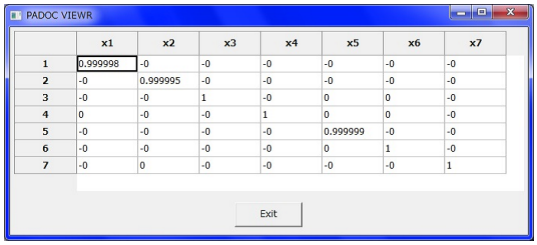


図 2.134 Verification unit matrix by inverse by original matrix

(b) 擬似逆行列 (一般逆行列)

逆行列を算出するには対象の行列が正定値 (固有値が全て正) である必要がある。
この擬似逆行列は正定値である必要がない。このロジックは特異値分解 (SVD) を使った [Numerical Recipes (2..6.6)] に詳述されている

$$SVD(\boldsymbol{A}) = \boldsymbol{U} \cdot \boldsymbol{w} \cdot \boldsymbol{V}^T$$
$$\boldsymbol{A}^+ = \boldsymbol{V} \cdot [diag(\frac{1}{w_{ij}})] \cdot \boldsymbol{U}^T$$

次の例は固有値に例がある場合は逆行列は発散してしまうが、擬似逆行列は求まることを示した例である。

```
hand x1-10/
0  6  7  2 3 3 1 0 0 1
1  1 11 6 0 2 1 4 1 2
2 12 32 5 0 1 3 4 1 1
3  3  7 3 2 2 2 1 2 5
4  6  6 3 5 1 1 1 1 3
5  7  9 5 0 2 2 1 1 2
;
//共分散行列
cov x1-10;
get freq@ana;
put cov;

//(B)-1 共分散行列の固有値
eigen x1-10;
get freq@ana;
select eigen_0-9;

//(B)-2 逆行列
get cov;
mxinv x1-10;

//(B)-3 擬似逆行列
get cov;
mxpinv x1-10;
//(B)-4 擬似逆行列と共分散行列を掛けた結果
mxmult x1-10 by cov;
```

(B-1) 固有値の結果 0 の値となって正定値ではない

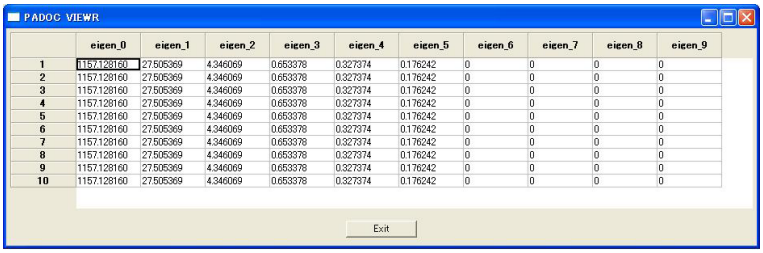


図 2.135　Result of inverse matrix

(B-2) 逆行列の結果　値が発散している

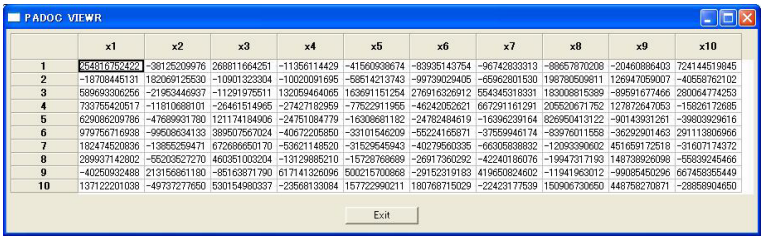


図 2.136　Result of inverse matrix

(B-3) 擬似逆行列の結果では値が発散していない

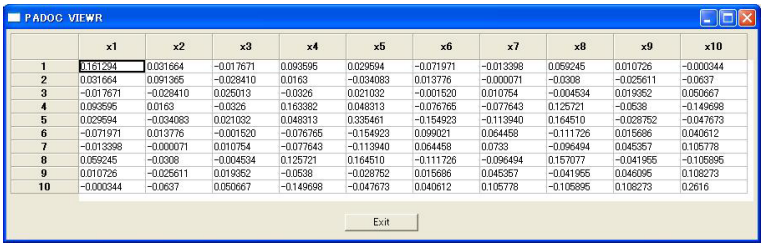


図 2.137　Result of pseudo inverse matrix

(B-3) 擬似逆行列と共分散行列を掛けた結果　対角成分は 1 に近くなっている。

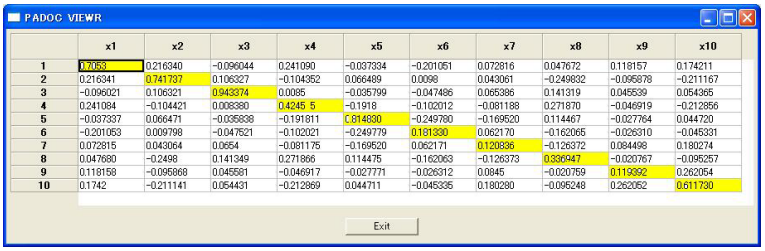


図 2.138　Result of pseudo inv * cov

(c) 固有値

図 2.139 の 20 × 8 の行列の固有値計算を行う.。固有値は多次元のデータの直交軸を回転させて、データをカバーする方向を計測するものである。行列が 8 列の場合、固有値は 8 個計算され、最初に計算される固有値から順番に小さくなる。

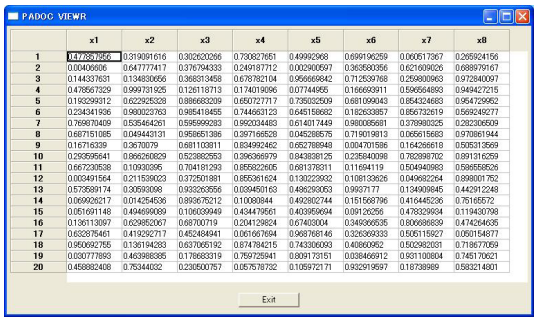


図 2.139 Input data for eigen value

```
//20 × 8 行列を読込む
get nfmx20_8.csv@;
//(C-1) 固有値計算する
eigen x1-8;
get freq@ana;
//(C-2) 固有値のグラフ表示
plot bar eigen_0-8 by name;
select vector_0-2;  //固有ベクトルの 3 成分の取出し
put eigen;

//8 次元の 20 行のデータに固有ベクトル 3 成分を乗じる
get nfmx20_8.csv@;
mxmult x1-8 by eigen;
plot scat vector_0-2;  //(B-3)20 点の 3D 表示;
```

(C-1) 固有値計算の結果は固有値と変換ベクトルのペアーで図 2.140 の様に計算される。

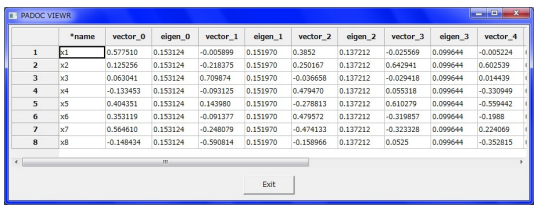


図 2.140 Result of eigen values and transrate vectors

(C-2) 固有値の値は第 1 軸から 8 軸まで図 2.141 の様になっている。

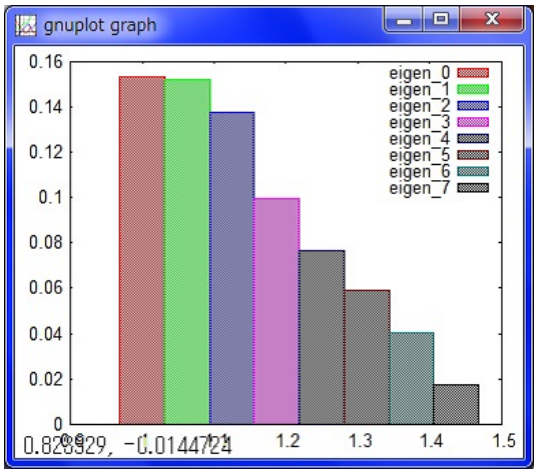


図 2.141　Result of eigen value

(C-3)3 軸までの固有ベクトルで表した 20 点の状態

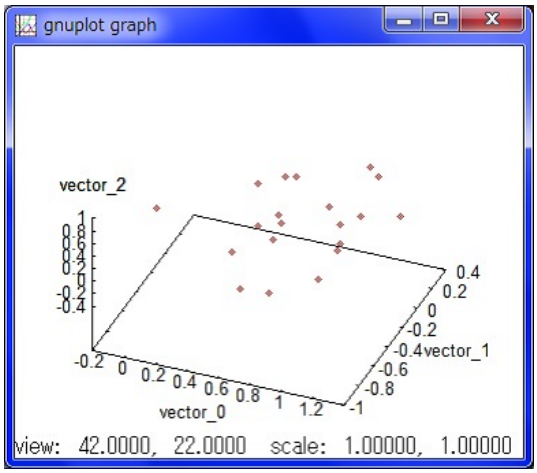


図 2.142　3D view of 20 points by eigen vector

(d) 行列式

padoc では図 2.143 正方行列の行列式を次の記述で計算する。

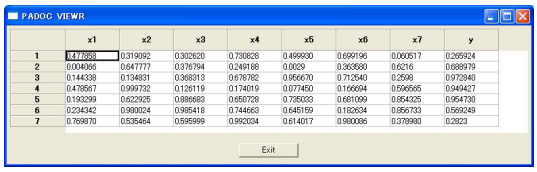


図 2.143　Input data for determinent

```
//正方行列の読込
get lu8.csv@;
//7 × 7 行列式の計算
determ x1-7;
//(D-1) 結果の取出し
get freq@ana;
```

(D-1) 計算結果は図 2.144 の様に唯一の値となる。

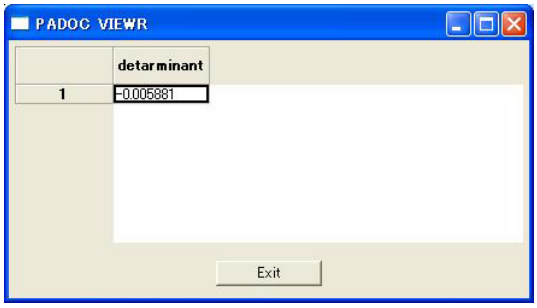


図 2.144　Result of determinent

(e) 行列の加減

padoc では同じ行と列が一致すれば加減ができる。

```
//20 × 8 の行列の読込
get nfmx20_8.csv@;
//同じ行列での減算し全てが 0 の行列とする
mxsub x1-8 by nfmx20_8.csv@;
//(E-1) 同じ行列を加算して回復する
mxadd x1-8 by nfmx20_8.csv@;
```

(E-1) 同じ行列を減算して全て 0 にした結果

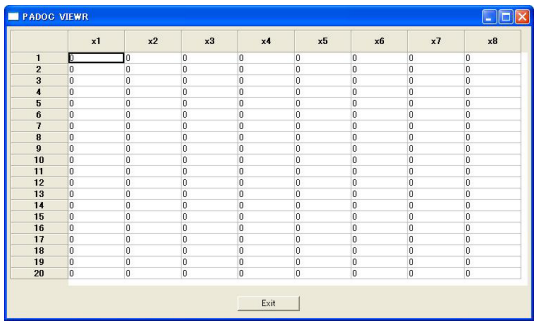


図 2.145　Result of matrix calculation

第 3 章

PADOC/STAT PLOT 表示

padoc ではグラフ表示には GNUPLOT を使っている。シンタックス（文法) としては以下の記述となる。
GNUPLOT の仕様として図表示と同時に図消去のダイアログが現われる。これを押下すると図は消える。

```
plot plot 種別 変数名 1 $\cdots$ 変数名_n by 区分/
  オプション
;
```

plot 種別としては以下のものがある。

- hist ヒストグラム
- scat 散布図 (2D 3D 相関図)
- line 折線グラフ
- bar 棒グラフ
- box ろうそく図

なお、plot の変わりに over と記述すると前の plot に重ね合わす事ができる。

plot コマンドの option

表 3.1　Options for plot command

name	value	content	default
variable	code	variable as code	
variable	format	format file for variable	
file	filename	specify file name for output	

3.1　ヒストグラム

padoc では一つの変数についてヒストグラムを表示する。次の記述の様に by を使うとそのカテゴリィ別にヒストグラムが重ね合わせて表示できる。下記の例は行員のデータテーブルには職種コード (jobcat) があり、これに職種名 (jobname) をマージして職種名毎に現在給与 (salnow) のヒストグラムを描いたものである。

```
hand*jobcat jobname/      //職種名を手入力
1 1:stuff
2 2:trainning
3 3:gardman
4 4:special
5 5:maneger
6 6:mba
7 7:engineer
;
put jobid_;       //職種名データ・テーブルの生成

get bank.csv@;    //銀行員データ・テーブルの読込
merge jobid_ by jobcat;       //(A-1) 職種コードをキーとして職種名データ・テーブルをマージ

plot hist salnow by jobname;  //(A-2) 職種名で現在給与の分布を表示
```

(A-1) 図 3.1 は職種名 (jobname) をマージした結果

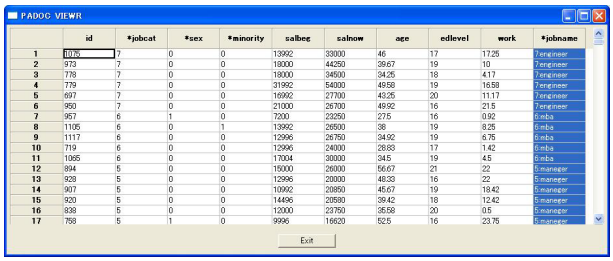


図 3.1　Result of merge for bankdata and jobname by jobcat

(A-2) 図 3.2 は職種名毎の現在給与のヒストグラム

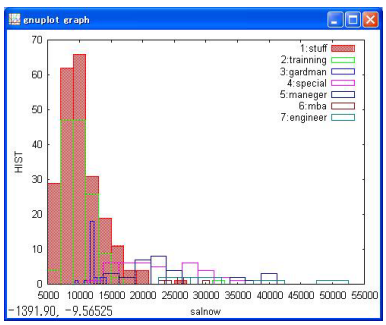


図 3.2　Histgram of salary by jobname

3.2　散布図

padoc では変数の数によって散布図が異なる。

- 変数が 2 個　2D 散布図
- 変数が 3 個　3D 散布図
- 変数が 4 個以上　散布相関図

次の例は菖蒲の散布図でこれも by によるカテゴリィ毎の散布図が得られる。但し散布相関図は by は無視される。

```
get iris.csv@;
  plot scat SepalL SepalW by Species; //(B-1)2D 散布図
  plot scat SepalL SepalW PetalL by Species; //(B-2)3D 散布図
  plot scat SepalL SepalW PetalL PetalW;  //(B-3) 散布相関図
```

(B-1)2D 散布図

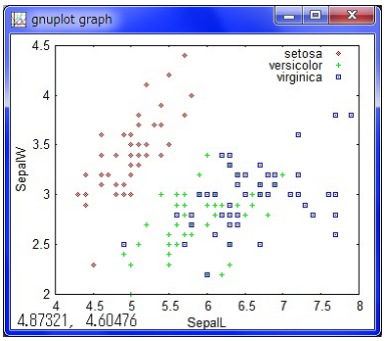


図 3.3　2D scatter for 2 variable of iris data by species

(B-2)3D 散布図

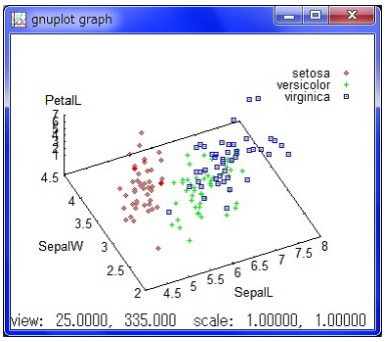


図 3.4　3D scatter for 3 variable iris data by species

(B-3) 散布相関図

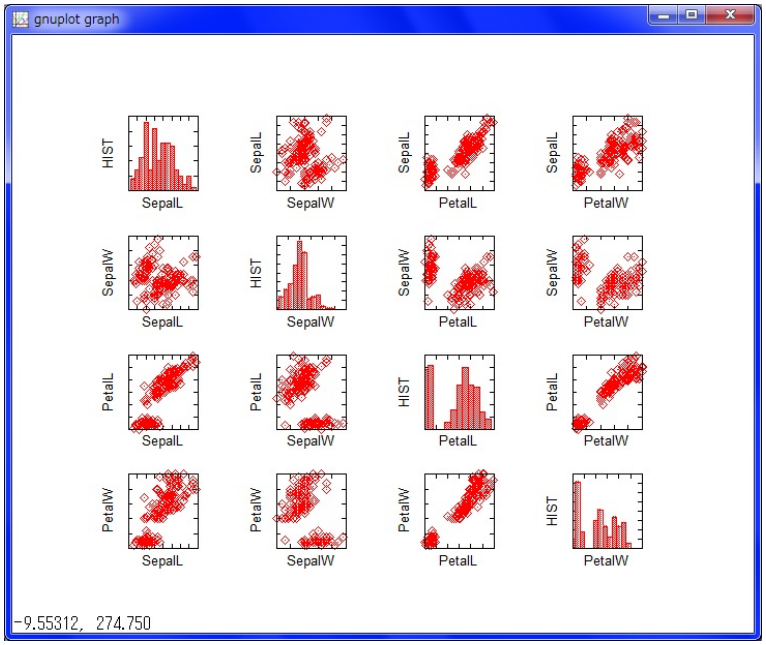


図 3.5　scatter correlation for 4 variable of iris data

3.3　折れ線グラフ

padoc では複数の変数について折れ線グラフを表示することができます。
ここでの例は、ドル円の為替の変動に対するＦ社の株価の変動を折れ線グラフで示したものである。両値は比較するため数値は正規化（平均０、分散１）してある。これによると為替が 0(平均) 近傍の場合はこの株価は低く、為替が上下に変動するにつれて、株価が上昇する現象がミ見られる。

```
get stock_fx.csv@;
plot line stockNorm fxNorm;  //(C-1) 複数折れ線図
sort fxNorm;
plot line stockNorm by fxNorm;  //(C-2)XY 折れ線図
```

(C-1) 複数折れ線図

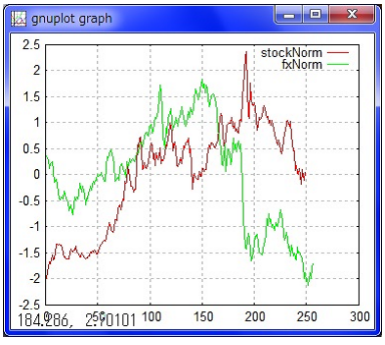


図 3.6　Stock and Fx time series

(C-2)XY 折れ線図

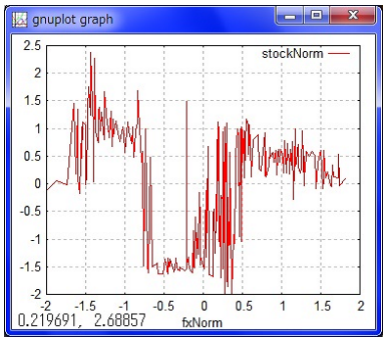


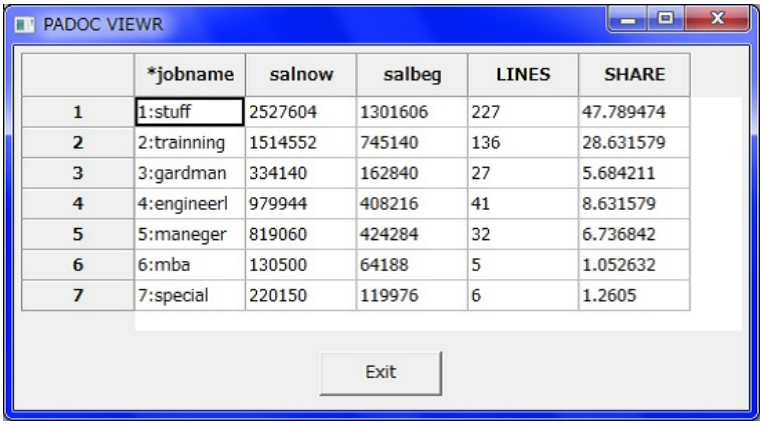
図 3.7 Movement of stock by fx movement

3.4 棒グラフ

棒グラフの例は米国の行員データテーブルの職種別の現在給与と初任給を表示したものである。まず職種毎に給与をサマリーして、その件数で割って平均給与にしたもを棒グラフで表示した。

```
//職種名テーブルの設定
hand jobcat jobname/
1 1:stuff
2 2:trainning
3 3:gardman
4 4:engineerl
5 5:maneger
6 6:mba
7 7:special
;
put jobcat_;
//行員データテーブルの読込み
get bank.csv@;
merge jobcat_ by jobcat;
//(D-1) 職種名で給与をサマリー
sumup salnow salbeg by jobname;
//サマリー結果の取出し
get freq@ana;
//平均給与の計算
salnave=salnow/LINES;
salbave=salbeg/LINES;
//(D-2) 職種別　平均給与の棒グラフ
plot bar salnave salbave by jobname;
```

(D-1) 職種別の給与の現在給与と初任給のサマリー結果



	*jobname	salnow	salbeg	LINES	SHARE
1	1:stuff	2527604	1301606	227	47.789474
2	2:trainning	1514552	745140	136	28.631579
3	3:gardman	334140	162840	27	5.684211
4	4:engineerl	979944	408216	41	8.631579
5	5:maneger	819060	424284	32	6.736842
6	6:mba	130500	64188	5	1.052632
7	7:special	220150	119976	6	1.2605

図 3.8　summary table by jobname for now and begin salary

(D-2) 職種別の平均給与の表示

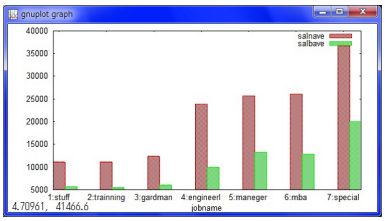


図 3.9　bar graph for average of now and begin salaly by jobname

3.5　箱髭図

padoc では分布を表す箱髭図を表示する。この例は行員の現在給与と初任給の分布を箱髭図で示し、次に職種毎の現在給与の分布を箱髭図で示したものである。中央の (*) が平均値で箱の上辺が 75 %点の値、下辺が 25 %の値である。上端の矢印は最大値、下端の矢印は最小値を示す。

```
//行員データテーブルの読込み
get bank.csv@;
//(E-1) 現在給与と初任給の箱髭図
plot box salnow salbeg

//職種名テーブルの設定
hand jobcat jobname/
1 1:stuff
2 2:trainning
3 3:gardman
4 4:engineerl
5 5:maneger
6 6:mba
7 7:special
;
put jobcat_;
//(E-1) 職種コードをキーに職種名をマージ
get bank.csv@;
merge jobcat_ by jobcat;
//(E-2) 職種名毎の現在給与の箱髭図を表示
plot box salnow by jobname;
```

(E-1) 現在給与と初任給の箱髭図

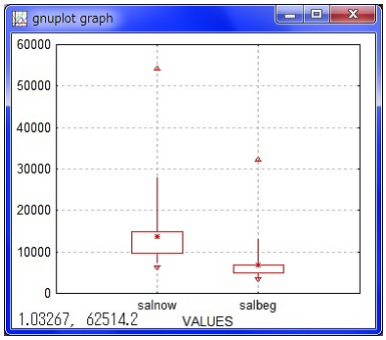


図 3.10 box figure for comparison of now salary and begin salary

(E-2) 職種名毎の現在給与の箱髭図を表示

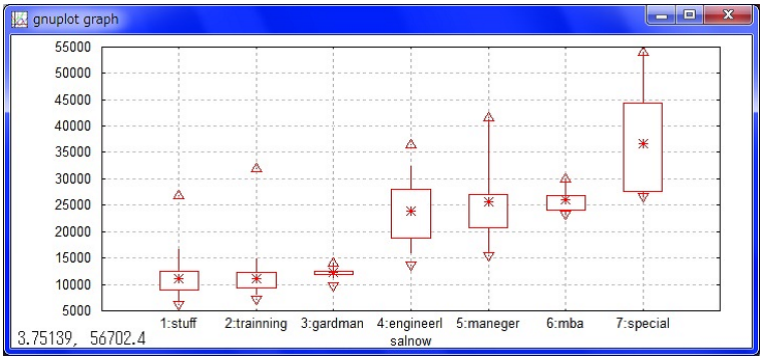


図 3.11　box figure for now salary distribution by jobname

3.6　重ね合わせ図

padoc では一旦　plot　コマンドで表示したグラフを　over　コマンドで図を重ねあわすことができる。

この例は株価をフーリエ変換で周波数領域に変換し、50 と 10 以上の高周波をカットし、逆フーリエ変換し時系列にして、重ね合わせた図である。

```
//株価と為替のデータテーブルの読込
get stock_fx.csv@;
//株価のグラフ表示
plot line stockNorm;
//フーリエ変換
fourie stockNorm/
 kind=trans
;
//周波数領域に変換した結果の読込
get freq@ana;
put freq;
//50 以上の周波数をカット
if(# >= 50) {
  stockNorm=0;
}
//逆フーリエ変換
fourie stockNorm/
 kind=rev
;
//時系列に変換結果の読込
get freq@ana;
rename stockNorm=cut50;
//重ね合わせ図の表示
over line cut50;
//再度周波数領域の読込
get freq;
//10 以上の周波数をカット
if(# >= 10) {
  stockNorm=0;
}
//逆フーリエ変換
fourie stockNorm/
kind=rev
;
get freq@ana;
rename stockNrom=cut10;
//重ね合わせ図の表示
over line cut10;
//(F-1)2 回目の重ね合わせ図の表示
over line stockNorm;
```

(F-1) 高周波の除去で波動が平滑化されている事が分る。

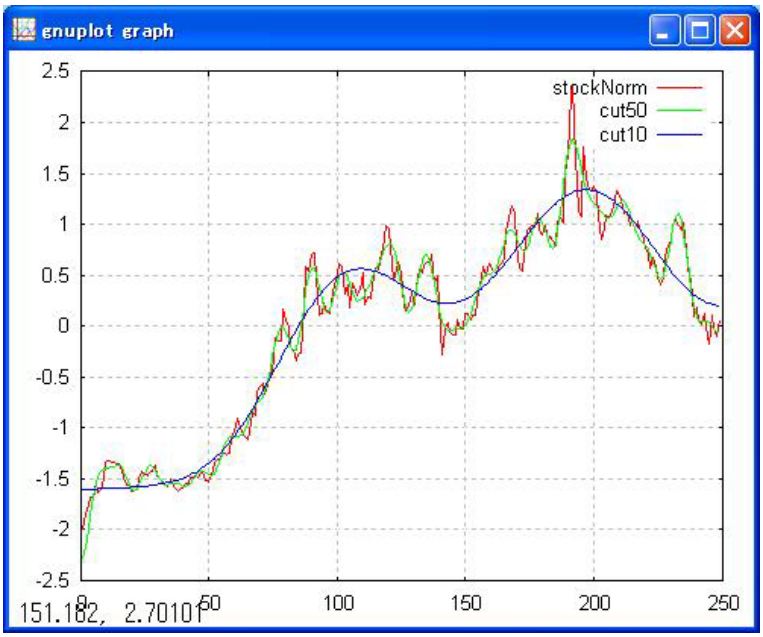


図 3.12　Over plot of fourie inverse transformation by cut hight frequency

第 4 章

ユーザ登録コマンド

padoc ではユーザ独自のコマンドを登録し使うことができる。これは Padoc の提供するコマンド群と全く同じ操作でユーザが定義できるものである。
ユーザ組込みコマンドは以下の既約に従うことで Padoc 環境から呼び出すことができる。

- プログラムは C C++ Python で記述してもよいがコンパイルして exe 形式である必要がある。
- コマンド名は名称の衝突を避けるため My から始まることを推奨する。
- 環境変数 PADOC_FHLP で指定しているフォルダーは以下に　exe 形式で配置する必要がある。
- 環境変数 PADOC_FHLP で指定しているフォルダーの mycommand.csv に登録する必要がある。

参考のため 環境変数 PADOC_FHLP の source フォルダーにユーザ登録コマンドの C 言語ソースの例が置いてある。

4.1　コマンドの登録テーブル

ユーザ組込みコマンドの登録テーブル mycommand.csv は以下の様に記述されている

表 4.1　Table of user's registration command

$command	function	tag	date	author	options1	options2	options3	options4	option5
myNorm	変数の正規化	0	20190928	mabo	debug				
myReg	重回帰	1	20190928	mabo	debug	loop	alpha		
myLogit	ロジット回帰	1	20190928	mabo	debug	iter	lambda		
myMLogit	softMax 回帰	1	20190928	mabo	debug	loop	lambda	eps	

環境変数 PADOC_FHLP で指定しているフォルダーでは既に 3 種類のユーザ登録コマンドが設置してある。各項目の意味は以下である。

- $command：ユーザ登録コマンド名　My〜で始まる　(必須)
- function：コマンドの機能
- tag :目的変数の数 (必須)
- date :登録日
- author: 作成者
- option1〜 :コマンド内で使用されるオプション 任意の英数字　数は登録されているだけ使用できる

最後の option は次の様な padoc のコマンドの Option 設定でユーザ登録コマンドに引渡される

例えば重回帰の例では以下で設定すると、myReg コマンド内で設定値が使える。

```
//菖蒲データの読込
get iris.csv@;
//ユーザ登録コマンド重回帰の実行
myReg SepalL by SepalW PetalW/
loop=100
alpha=0.01
;
plot scat SepalW PetalW SepalL; //実際の 3D 散布図
get freq@ana; //回帰結果の取込み
over scat SepalW PetalW predict;  //予測の重ね図
```

上記のスクリプトを実行したユーザ登録コマンド（重回帰）の結果 ・が実値 + が予測値

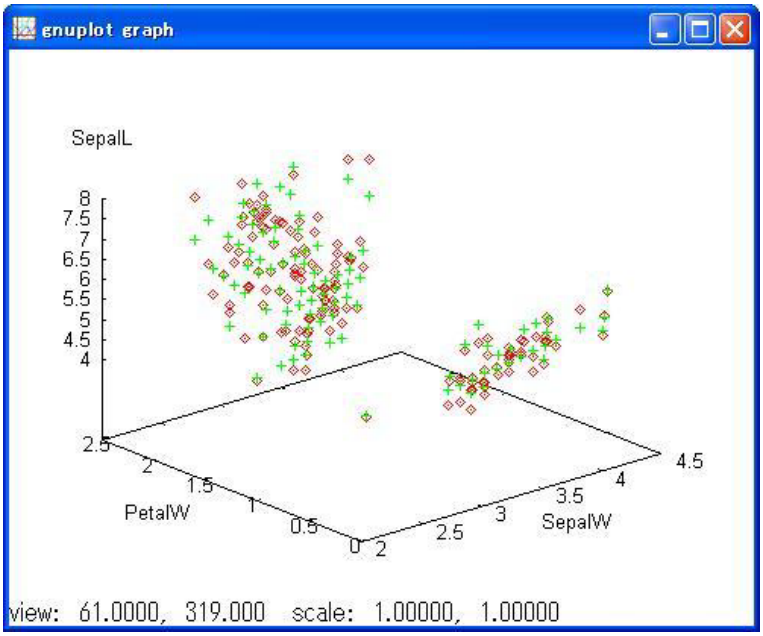


図 4.1　Regression by user registration command

4.2 ユーザ登録コマンドの I/F

ユーザ登録関数はどの様な言語で作成してもよいが、最後は My〜で始まる exe ファイルを所定の場所に置かなければならない。
padoc 環境からユーザ登録コマンドとのデータの受け渡しは　コマンドの引数で行う。もしコマンド側でエラーがあればエラーの戻り値を返す必要がある。
もしコマンド側でエラーがあればエラー戻り値に従って padoc 環境でエラーを Log に表示する。

4.2.1 Padoc 環境から渡される引数

padoc 環境からユーザ定義のコマンドをすには場合は 6 個の引数で渡す。

1. データが入ったファイル名 (データはタブ区切
2. option が入ったファイル名
3. 結果出力用ファイル名 (タブ区切のテーブル形式のファイル名)
4. 結果コメント用ファイル名
5. 目的変数の数　(上記の例だと 1)
6. 説明変数の数 （上記の例だと 2)

ユーザ登録コマンド側のプログラムでは上記の 4 つの入出力ファイル名と 2 個のパラメータを受取り、
ファイル経由でデータを取り出す。ここでは C 言語での例を示す。

```
#include <stdio.h>
#include <string.h>

#define DLM \t

int main{argc,argv}
int argc; //引数の数 自身のコマンドを入れて argc=7 になる
char *argv[]; //引数の値
{
   FILE *fp,*fparm,*fd,*fa;
   int ntag,nexp;
   char record[4098],*pc;
   int n,m;
   int i,j;

   char **names;
   double **data;

   if(argc != 7) return(-1) //引数の数が合わないのでエラーで返る
   if(!(fp=fopen(argv[1],"r")))      return(-1);  //入力データ用のファイル Open
   if(!(fparm=fopen(argv[2],"r")))  return(-1);  //Option 定義用のファイル Open
   if(!(fd=fopen(argv[3],"w")))     return(-1);  //出力テーブル用のファイル Open
   if(!(fa=fopen(argv[4],"w")))      return(-1);  //出力コメント用のファイル Open
   ntag=atoi(argv[5]);  //目的変数の数
   nexp=atoi(argv[6]); //説明変数の数

   if(ntag < 1) return(-1);
   if(nexp < 1) return(-1);
```

4.2.2　データの読込

padoc から渡されるデータは入力ファイル経由で読込む。
このファイルはタブ区切のテーブル形式になっている 表 4.2。
左から順番に目的変数の項目、説明変数の項目となっている。
但しこの入力ファイルはレコードの先頭の文字によって次の意味があり、適宜処理する必要がある。

- ＃　コメント行を示す
- ＆　項目の型を示す
- ＄　項目名を示す

上記 MyReg での padoc から渡されるデータは入力ファイルの例

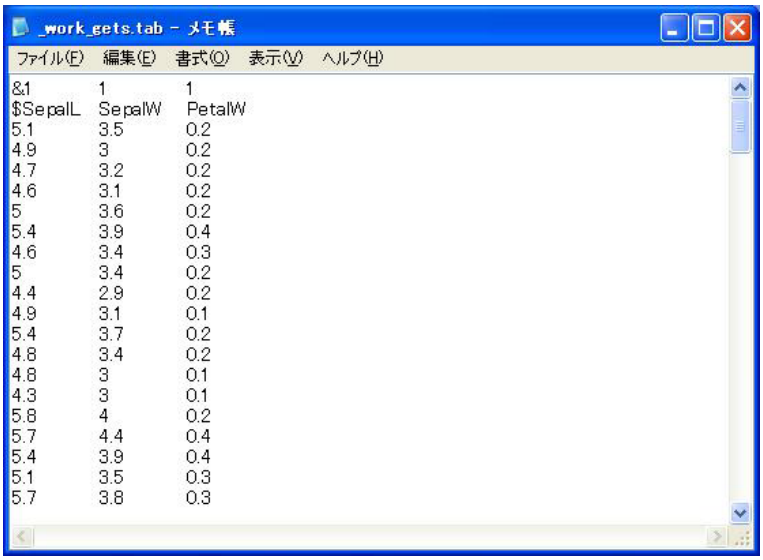


図 4.2　Interface file to user registration command

以降に領域を獲得して、データを一行毎に読込み data 変数に格納する例を示す。

```
//レコード数を計算する
n=0;
while(fgets(record,4096,fp) {
  if(record[0] == '#') continue;
  if(record[0] == '&') continue;
  if(record[0] == '$') conitnue;
  n++;
}
//項目名とデータの格納領域を確保する
names=(char **)malloc(sizeof(char *)*(ntag+nexp));
data[i]=(double **)malloc(sizeof(double *)*n);
for(i=0;i<n;i++) data[i]=(double *)malloc(sizeof(double)*(ntag+nexp));

rewind(pc);  //もう一度先頭から読込む
i=0;
while(fgets(record,4096,fp) {
   record[strlen(record)-1]='\0';
   if(record[0] == '#') continue;
   if(record[0] == '&') continue;
   if(record[0] == '$') {  //項目名業の処理
      pc=strtok(record);
      j=0;
      while(pc) {
         names[j]=(char *)malloc(sizeof(pc));
         strcpy(name[j++],pc);  //項目名を格納する
         pc=strtok(NULL,DLM); //次の項目名を読む
      }
      m=j
      continue;
   }
   pc=strtok(record,DLM);
   j=0;
   while(pc) {
      data[i][j++]=atof(pc);    //データを数値型で読む
      pc=strtok(NULL,DLM);  //次の項目名を読む
   }
  i++;
}
fclose(fp);
```

4.2.3 Option の取込み

padoc 環境で設定した Option の設定を Option ファイルから読込む。
Option ファイルは次の形式で格納されている。
option 名 = option 値
C 言語では Option 値を設定する例を以下に示す。

```
iter=1000;        //暗黙の Option 値
lambda=0.001;
debug=0;

//Option 値の読込
while(fgets(record,8192,fparm)) {
  if(record[strlen(record)-1] == '\n') record[strlen(record)-1]='\0';
  pc=strtok(record," :=");
  if(pc && !strcmp(pc,"iter")) {
    pc=strtok(NULL," \n");
    if(pc) iter=atoi(pc);
  }
  if(pc && !strcmp(pc,"lambda")) {
    pc=strtok(NULL," \n");
    if(pc) lambda=atof(pc);
  }
  if(pc && !strcmp(pc,"debug")) {
    pc=strtok(NULL," \n");
    if(pc) debug=atoi(pc);
  }
}
fclose(fparm);
```

4.2.4　結果の書出

計算結果をテーブル形式とテキスト形式に書出す。
テーブル形式のファイルでは先頭に項目名行を出力し数値行を出力する。何れもタブ形式である。

```
//項目名の書き出し
 for(j=0;j<m;j++) {
    if(!j) fprintf(fd,"$");
    fprintf(fd,"%s",names[j]);
    fprintf(fd,DLM);
 }
 fprintf(fd,"predict\n");

 //計算結果の書出し
 for(i=0;i<n;i++) {
   for(j=0;j<m;j++) {
     fprint_d(fd,data[i][j]);
     fprintf(fd,DLM);
   }
   sum=w[0];
   for(j=1;j<m;j++) {
     sum += w[j]*data[i][j];
   }
   fprint_d(fd,sum);
   fprintf(fd,"\n");
 }
 fclose(fd);
 //テキスト形式の書出し
 fprintf(fa,"coefficient of determination=%lf\n",1.0-dis/var);
 for(j=0;j<m;j++) {
   fprintf(fa,"w[%2d]=",j);
   fprint_d(fa,w[j]);
   fprintf(fa,"\n");
 }
```

4.2.5 領域の開放

padoc からは exe 形式の呼び出しなので、領域は開放されるが、確保した領域は開放するのば望ましい。

```
    //Free;
    for(i=0;i<n;I++) {
      free(data[i]);
    }
    for(j=0;j<m;j++) {
      free(names[j]);
    }
    free(w);

    return(0);
}
```

4.2.6　ユーザ登録コマンド側からのエラー戻値

padoc 環境ではユーザコマンド側で以下の戻り値を認識し Log に表示する。
ユーザコマンド側はこれらのエラーがあれば return 文で終了する必要がある。

```
#define E_bad_parameter      - 1
#define E_file_no_record      - 2
#define E_short_of_row_for_col - 3
#define E_not_convergeny      -18
#define E_out_of_condition      -20
#define E_file_cannot_open      -35
#define E_data_not_boolean     -21
#define E_data_not_square_matrix -22
#define E_target_val_not_valid  -23
#define E_short_of_data          -24
#define E_file_bad_colm_char   -25
#define E_file_bad_colm_numb   -26
#define E_file_vals_rec_notFound -27
#define E_file_type_rec_notFound -28
#define E_file_vals_rec_numShort -29
#define E_file_type_rec_numShort -30
#define E_bad_option           -31
#define E_short_of_column      -121
#define E_lack_of_start_point   -124
#define E_lack_of_end_point     -125
#define E_not_enough_memory   -253
```

4.2.7 ユーザ登録コマンドのデバッグ方法

ユーザ登録コマンドは exe 形式なので任意の点でのデバッグは困難であるが、Option の debug を使って結果テキストに書出すことができる。

padoc 側の debug オプションの設定

```
get iris.csv@;
myReg SepalL by PetalL PetalW/
debug=1
;
```

ユーザコマンド側のデバッグライト

```
if(dbg) {
  fprintf(fa,"w[%d]=%lf\n",i,w[i]);
}
```

付録 A

補足

A.1　各章で例示したサンプル・コマンド記述

　下記のサンプル・コマンド記述はコマンド編集画面のファイル一覧アイコンを押下すると表示され、何れかのコマンド記述ファイルを選択すると、編集画面に表示される。

表 A.1 Reference of table chapters and sample command description No.1

chapters	command description	remarks
1.2.2	doc1_bank1.clc	
1.3.1	bank0.clc	
1.4.2	doc1_hand.clc	
1.4.2	doc1_nlp.clc	
1.4.2(c)	doc1_title.clc	
	doc1_dmy.clc	
	doc1_accume.clc	
1.4.2(e)	doc1_type.clc	
1.4.2(f)	doc1_get.clc	
1.4.3	doc1_edit.clc	
1.4.3(b)	doc1_vector.clc	
1.4.3(c)	doc1_cntl.clc	
1.4.3(d1)	doc1_divide.clc	
1.4.3(d2)	doc1_gen.clc	
1.4.3(e1)	doc1_sort.clc	
1.4.3(e2)	doc1_unique.clc	
1.4.3(f1)	doc1_merge.clc	
	doc1_mergeOption.clc	
1.4.3(f2)	doc1_concat.clc	
1.4.3(f3)	doc1_append.clc	
1.4.3(g)	doc1_select.clc	
1.4.3(h)	doc1_prev.clc	
1.4.3(i)	doc1_mxmult.clc	
1.4.4(a)	doc1_metaFunc.clc	
1.4.4.(b)	doc1_metaCtrl.clc	

表 A.2 Reference of table chapters and sample command description No.2

chapters	command description	remarks
2	doc2_resultUse.clc	
2	doc2_option.clc	
2.1.1	doc2_statis.clc	
	doc2_statisForm.clc	
2,1,2	doc2_freq.clc	
2.1.3	doc2_summary.clc	
2.1.4	doc2_aic.clc	
2.1.5	doc2_corr.clc	
2.2.1	doc2_reg.clc	
2.2.2	doc2_logit.clc	
2.2.3	doc2_svm.clc	
2.2.4	doc2_tree.clc	
2.2.5	doc2_regTree.clc	
2.2.6	doc2_naive.clc	
2.2.7	doc2_hazard.clc	
2.2.8	doc2_knn.clc	
2.2.9	doc2_mahalanobis.clc	
2.2.10	doc2_softmax.clc	
2.3.1	doc2_prin.clc	
2.3.2(a)	doc2_kmeans.clc	
2.3.2(b)	doc2_dendro.clc	
2.3.3	doc2_groupLense.clc	
2.3.4(a)	doc2_conjoint1.clc	
2.3.4(b)	doc2_conjoint2.clc	
2.3.5	doc2_assoc.clc	
2.3.6	doc2_ggm.clc	
2.3.7(a)	doc2_mahadist.clc	
2.3.7(b)	doc2_oneSVM.clc	
2.4.1	doc2_lp.clc	
2.4.2	doc2_intpl.clc	
2.4.3	doc2_squrepl.clc	

表 A.3 Reference of table chapters and sample command description No.3

chapters	command description	remarks
2.5.1(a)	doc2_tmcov.clc	
2.5.1(b)	doc2_tmcrs.clc	
2.5.1(c)	doc2_tmar.clc	
2.5.1(d)	doc2_tmave.clc	
2.5.1(e)	doc2_tmADF.clc	
2.5.2(a)	doc2_tmtrend.clc	
2.5.2(b)	doc2_season.clc	
2.5.2(c)	doc2_tmlag.clc	
2.5.2(d)	doc2_fourie.clc	
2.5.2(e)	doc2_hmm.clc	
2.5.2(f)	doc2_gproc.clc	
2.6.1	doc2_mecab.clc	
2.6.2	doc2_lda.clc	
2.7.1(a)	doc2_nnmf.clc	
2.7.1(b)	doc2_svd.clc	
	doc2_svd1.clc	
	doc2_svd2.clc	
	doc2_svd3.clc	
2.7.1(c)	doc2_LU.clc	
2.7.1(d)	doc2_QR.clc	
2.7.1(e)	doc2_cholesky.clc	
2.7.2(a)	doc2_mxinv.clc	
2.7.2(b)	doc2_mxpinv.clc	
2.7.2(c)	doc2_eigen.clc	
2.7.2(d)	doc2_determ.clc	
2.7.2(e)	doc2_mxcalc.clc	
3.1	doc3_hist.clc	
3.2	doc3_scat.clc	
3.3	doc3_line.clc	
3.4	doc3_bar.clc	
3.5	doc3_box.clc	
3.6	doc3_over.clc	
4.1	dog4_myReg.clc	

A.2　サンプル・コマンド記述 使用例

A.2.1　オプション価格の計算

金融資産のオプション価格では、期日でオプション行使するヨーロピアン型と期日までに行使価格をに至れば権利行使するアメリカン型がある。前者はブラック・ショルズの公式が適用できるが、後者はシュミレーションでしかオプション価格を決定できない。以下に両方法の算出コマンド記述を示す。

```
clear;
//アメリカン・オプション初期値
r=0.1;           //金利
sigma=0.2;   //原資産価格の変動の標準偏差
T = 5.0/12;  //期日までの月数
K = 60;        //行使価格
So = 62;       //原資産価格
N = 30;        //シュミレーションの刻み数

vector s[N+1][N+1];
vector v[N+1][N+1];

for(i=1;i<=N+1;i++) {
  for(j=1;j<=N+1;j++) {
     s[i][j] = 0.0;
     v[i][j] = 0.0;
  }
}
//株価ツリーの作成
delta = T/N;
u = exp(  sigma * sqrt(delta));
d = exp(-sigma * sqrt(delta));
//リスク中立
//u * p + d * (1-p) =exp(r*dt) -> p = {exp(r*dt) - d} / (u -d);
p = (exp(r * delta) - d)/(u - d);
for(i=0;i<=N;i++) {
  for(j=0;j<=i;j++) {
    s[i+1][j+1] = So * u**j * d**(i-j);
  }
}
//満期でのオプション価格
for(j=0;j<=N;j++) {
  x = K - s[N+1][j+1];
  if(x < 0) v[N+1][j+1] = 0;
  else     v[N+1][j+1] = x;
}
//後ろ向きにオプション価格を計算
for(i = N;i >=1;i--) {
  for(j=1;j <= i;j++) {
    x = K - s[i][j];
    y = exp(-r*delta) * (p * v[i+1][j+1] + (1-p) * v[i+1][j]);
    if(x >= y) v[i][j] = x;
    else       v[i][j] = y;
  }
}
A       [1][1]
```

付録 B

参考文献