



*System Analysis Total Environment for Laboratory
- Language and Interactive Execution*

リファレンスマニュアル

RELEASE 4.2.3

Declaration, Control, Operator, Constant, Builtin, Mathematic
library, UTILITY, SYSTEM, ISPP, GPM, BPS, NCS, NPE, DCM,
STATISTICS

Neuroinformatics Laboratry

SATELLITE: リファレンスマニュアル

RELEASE 4.2.3

Neuroinformatics Laboratory

Copyright © 2003-2005 RIKEN Japan

目次

1. DECLARATION	1
const	2
define	3
external	4
scalar	5
series	6
snapshot	7
string	8
2. CONTROL	9
break	10
continue	11
do	12
for	13
func	14
if	15
proc	16
return	17
while	18
3. OPERATOR	19
3.1. ARITHMETIC	19
3.2. ASSIGNMENT	25
3.3. LOGIC	31
3.4. RELATION	34
3.5. OTHER	40
4. CONSTANT	47
true	48
false	49
DEG	50
E	51
GAMMA	52
PHI	53
PI	54
5. BUILTIN	55
alias	56
eval	57
exit	58
history	59
index	60
inline	61
isdef	62
length	63
module	64
printf	65
read	66
strlen	67

symbols	68
system	69
typeof	70
undef	71
undefall	72
6. Mathematic Library	73
abs	74
acos	75
asin	76
atan	77
atan2	78
ceil	79
cos	80
exp	81
floor	82
int	83
log	84
log2	85
log10	86
mod	87
pow	88
round	89
sgn	90
sin	91
sqrt	92
tan	93
7. UTILITY	94
dump	95
exist	96
extract	97
indexsize	98
puts	99
revtime	100
Scalar	101
Series	102
Snapshot	103
sprintf	104
String	105
text2string	106
tolower	107
toupper	108
unit	109
8. SYSTEM	110
bm	111
cut	112
fill	113
find	114

get	116
getenv	117
header	118
help	119
max	120
maxpos	121
merge	123
min	124
maxpos	125
mkdir	127
put	128
reform	130
reverse	131
rotate	132
sampling	133
setenv	134
tempdir	135
thin	136
wait	138
9. ISPP	139
arand	140
argen	142
burg	144
butwmake	146
cep	148
dccut	149
det	150
eigen	151
fftc	153
fftn	154
fir	155
firmake	157
gauss2	159
hil	160
icep	161
iir	162
iirmake	164
interp	165
interp2	167
inv	168
levin	170
mnrand	172
mul	174
nmeq	175
norm	177
nrand	179
phase	180

pole	181
power	182
rank	183
spcf	184
sum	186
trans	187
urand	188
window	189
10. GPM	190
axis	191
chwin	193
color	194
cont	196
draw	198
egraph	200
factor	202
font	203
frame	204
ginit	205
graph	206
gsolm	208
gstat	211
label	212
line	214
ltype	216
lwidth	217
map	218
newpage	222
origin	223
prev	225
scale	226
size	228
title	229
wclose	230
we	231
wopen	232
11. BPS	234
bactload	235
bcor	236
bdisp	237
berrfunc	238
berrload	240
berror	241
bfunction	242
blalgo	243
blayer	245
blearn	246

blend	247
bpdisp	248
bpinit	249
bpload	251
bpsave	252
brec	253
brvmap	255
bsetrec	256
bsigmoid	257
bteach	258
bwalgo	259
bweight	260
bwgtset	261
bwinit	262
12. NCS	263
nassign	264
ncal	265
nchgbuff	266
ndelay	267
ne	268
nerase	269
ngetp	270
ninteg	271
nlink	272
nlist	273
nmake	274
nout	275
npara	276
npp	277
nsclist	278
nstim	279
ntime	281
13. NPE	282
cdata	283
cdel	284
cdisp	285
chistory	286
cinit	287
clist	289
cload	291
clogic	292
clsearch	293
cmethod	294
cmodel	295
cnorm	296
cnumber	297
cpenalty	298

cpoint	299
cresult	300
cscale	301
cstore	302
cterm	303
cweight	304
npe	305
npeinit	306
14. DCM	307
abf2satellite	308
atf2satellite	309
buffer2avs	310
buffer2mathematica	311
buffer2matlab	312
buffer2text	313
genesis2buffer	314
matlab2satellite	315
neuron2satellite	316
teac2satellite	317
text2buffer	318
15. STATISTICS	319
15.1. STATISTIC	319
15.2. TEST	344

第 1 章 DECLARATION

宣言文

目次

const	2
define	3
external	4
scalar	5
series	6
snapshot	7
string	8

const

機能

定数を定義する .

形式

```
const name = value
```

パラメータ

1. name : 定数名
2. value : 定数 (数値 , 文字列)

解説

定数として定義した値の変更は , undef 関数で定義を取り消すか const で再定義し直すこと以外は許可されない .

参照

undef

define

機能

システム・モジュールを定義する．

形式

```
define mod_name {  
    set Module_Dll    "dll_file";  
    set Module_Ini    "ini_file";  
}
```

パラメータ

1. mod_name : モジュール名
2. dll_file : モジュールファイルのパス (String)
3. ini_file : 設定ファイルのパス (String)

解説

モジュールファイルおよび設定ファイルの格納場所を指定し，システム・モジュールとして定義する．この書式でシステム・モジュールを定義した後，module 関数を呼び出すことにより 利用可能となる．

使用例

モジュールの定義例を示す．

```
% define SYSTEM {  
%   set Module_Dll "/usr/local/satellite4/lib/satellite4/system.so";  
%   set Module_Ini "/usr/local/satellite4/etc/satellite4/system.ini";  
% }
```

参照

module

external

機能

外部参照する変数を宣言する .

形式

```
external x, y, z, ...
```

パラメータ

x, y, z, \dots : 外部を参照する変数名

解説

宣言した変数は , トップレベルで使用している変数が参照される . これらの変数は , 関数 (func) および手続き内(proc)での宣言のみ有効である .

使用例

トップレベルで宣言した Series 変数 x を sinewave 関数内で参照する場合

```
% n = 20
%
% series x;
% x = 0~n/n;
%
% func sinwave( a, f, th ) {
%   external x;
%   series y;
%
%   y = a*sin( 2*PI*f*x+th );
%
%   return y;
% }
```

参照

func , proc

scalar

機能

Scalar 型の変数を宣言する .

形式

```
scalar x, y, z, ...
```

パラメータ

x, y, z, \dots : Scalar 型の変数として宣言される変数名

解説

1. Scalar 型の変数には , 実数値 1 つが格納される .
2. 宣言した変数型は , 任意の関数の出力値を代入することにより , 変更される可能性がある .

series

機能

Series 型の変数を宣言する .

形式

```
series x, y, z, ...
```

パラメータ

x, y, z, \dots : Series 型の変数として宣言される変数名

解説

1. Series 型の変数は、可変長の数列を格納するものであり、2 次元以上の 変数を宣言する際には、下位次元の大きさ (サブインデックス値) を示さなければならない . 例えば、`series x[64][32], y[100]` などのように設定する . 前者では $n \times 64 \times 32$ の 3 次元配列 . 後者では $n \times 100$ の 2 次元配列となる . 何も大きさを示さなかった場合には、1 次元配列となる . 下位次元についてはその大きさは固定である . 最上位次元の n については可変であり、宣言直後は 1 となっている .
2. 宣言した変数の大きさおよび型は、任意の関数の出力値を代入することにより、変更される可能性がある .
3. `reform` 関数によって変数の次元数・大きさを変更することができる .
4. `x:[3]` などのサブバッファ参照値の変数の型は、Snapshot 型となる .

参照

`reform`

snapshot

機能

Snapshot 型の変数を宣言する .

形式

```
snapshot x[n][m], ...
```

パラメータ

1. x : Snapshot 型の変数として宣言される変数名
2. n, m : 配列のインデックス値 (Scalar)

解説

1. Snapshot 型の変数は , 大きさは固定であり , 宣言時に配列のインデックス値 を必ず示さなければならない . 例えば , `snapshot x[64][32], y[100]` などのように宣言する . 前者では , 64×32 の 2 次元配列 . 後者では要素数 100 の 1 次元配列となる . 宣言直後は , 全ての配列要素が 0 に設定される .
2. `reform` 関数によって 変数の次元数・大きさを変更することができる .

参照

`reform`

string

機能

String 型の変数を宣言する .

形式

```
string x, y, z, ...
```

パラメータ

x, y, z, ... : String 型の変数として宣言する変数名

解説

1. String 型の変数には , 文字列を格納することができる . 宣言時に `string x[100]` などのインデックス値を伴って宣言することにより , 多次元の文字列配列として利用可能である .
2. 大きさの設定方法は , Snapshot 型の場合と同じであり , それぞれの要素に 文字列が格納される .
3. 宣言した変数の大きさおよび型は , 任意の関数の出力値を代入することにより , 変更される可能性がある .

第 2 章 CONTROL

制御文

目次

break	10
continue	11
do	12
for	13
func	14
if	15
proc	16
return	17
while	18

break

機能

実行を中断してループを抜ける。

形式

`break`

パラメータ

なし

解説

実行を中断して for や while , do-while の繰り返しループを抜ける。

使用例

```
% for( i=0; i<100; i++ ) {  
%     ..... statement1 .....  
%     if ( i>1e15 ) {  
%         break;  
%     }  
%     ..... statement2 .....  
% }  
%
```

参照

for , while , do , continue

continue

機能

ループの実行を中断し先頭に処理を移す。

形式

```
continue
```

パラメータ

なし

解説

処理の実行を中断して、while 文や do-while 文の終了を判断する条件式に戻る。for 文の場合は、実行を中断して for 文の 3 式目に定義されている初期値 の再設定に処理を移し、その後、終了を判断する条件式に戻る。

使用例

```
% for( i=0; i<100; i++ ) {  
%     ..... statement1 .....  
%     if ( data!=0.0 ) {  
%         continue;      /* bypass the rest of the statements */  
%     }  
%     ..... statement2 .....  
% }  
%
```

参照

while , do-while , for , break

do

機能

DO-WHILE 型のループ制御を行う。

形式

```
do { stmt } while ( expr )
```

パラメータ

1. stmt : 処理 ({ } で囲まれた複数のコマンドによる処理の記述ができる)
2. expr : 条件判定

解説

1. expr が真 (0 以外) である間 , stmt を繰り返し実行する。
2. expr において , 論理演算子には AND 演算子 "&&" や "||" , 及び全ての関係演算子が使用できる。論理演算の結果が 0 ならば偽 , それ以外は真となる。

使用例

処理を 100 回繰り返して実行する場合の記述例

```
% i=0;
% do {
% ..... statement .....
% i++;
% } while (i<100);
```

参照

while , break , continue

for

機能

for ループ制御をする。

形式

```
for ( stmt1 ; expr; stmt2 ) stmt3
```

パラメータ

1. stmt1 : カウンタの初期化
2. expr : 条件判定
3. stmt2 : カウンタの再設定
4. stmt3 : 処理 ({ } で囲まれた複数のコマンドによる処理の記述ができる)

解説

1. stmt1 , stmt2 , expr に複文を記述することはできない。
2. for ループでは、まず stmt1 が実行され、次に expr が真である間、stmt3 を実行し、stmt2 が実行される。
3. expr において、論理演算子には AND 演算子 "&&" や "||"、及び全ての関係演算子がしよ
うできる。論理演算の結果が 0 ならば偽、それ以外は真となる。

使用例

処理を 100 回繰り返して実行する場合の記述例

```
% for( i=0; i<100; i++) {
%     .....処理.....
% }
```

func

機能

関数を定義する。

形式

```
func function( [arg1,...,argn] ) stmt
```

パラメータ

1. function : 関数名
2. arg1,...,argn : 引数名
3. stmt : 処理 ({ } で囲まれた複数のコマンドによる処理の記述ができる)

解説

1. 引数, また戻り値として, Scalar, Series, Snapshot, String 型のオブジェクトを扱うことができる。
2. return 関数により, 呼び出し元にデータを返すことができる。
3. 関数内で引数の内容の書き換えた場合, この変化は呼び出し元の対応する変数に影響する。

使用例

任意の型のデータを String オブジェクトに変換する関数の記述例

```
% func data2string( var ) {
%   return " " + var
% }
```

if

機能

条件分岐をする .

形式

```
if ( expr ) stmt1 [ else { stmt2 } ]
```

パラメータ

1. expr : 条件判定
2. stmt1 : 処理
3. stmt2 : 処理

解説

1. else 文を記述する場合 ,} else { のように ,} の直後に改行を入れずに else を続けて記述する .
2. expr において , 論理演算しには AND 演算子 "&&" や "||" , 及び全ての関係演算子が使用できる . 論理演算の結果が 0 ならば偽 , それ以外は真となる .

proc

機能

手続きを定義する .

形式

```
proc method( [arg1,...,argn] ) stmt
```

パラメータ

1. method : 手続き名
2. arg1,...,argn : 引数名
3. stmt : 処理 ({ } で囲まれた複数のコマンドによる処理の記述ができる)

解説

手続き内で引数を書き換えた場合 , 呼出元の対応する変数に影響する .

使用例

引数 n に 10 加える手続きの記述例

```
% proc plusten(n) {
%   n=n+10
% }
% s=0
% plusten(s)
% s
10
```

参照

external , func

return

機能

指定されたデータを関数の呼出元に渡す。

形式

```
return stmt
```

パラメータ

stmt : 評価式 (オブジェクトのみも含む)

解説

関数の処理結果を呼出元に渡すコマンドである。複数のオブジェクトは指定できない。

参照

func

while

機能

WHILE 型のループ制御を行う。

形式

```
while( expr ) stmt1
```

パラメータ

1. `expr` : 条件判定
2. `stmt1` : 処理 (`{ }` で囲まれた複数のコマンドによる処理の記述ができる)

第 3 章 OPERATOR

演算子

目次

3.1. ARITHMETIC	19
3.2. ASSIGNMENT	25
3.3. LOGIC	31
3.4. RELATION	34
3.5. OTHER	40

+

機能

加算

形式

 $x + y$

パラメータ

 x : 入力オブジェクト (Scalar, Series, Snapshot) y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5  
% y = 10  
% x + y  
15
```

機能

減算

形式

$$x - y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 10
% y = 5
% x - y
5
```

＊

機能

乗算

形式

$$x * y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5  
% y = 10  
% x * y  
50
```

/

機能

除算

形式

$$x \ / \ y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
% y = 10
% x / y
0.5
```

%

機能

剰余

形式

$x \% y$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
```

```
% y = 2
```

```
% x % y
```

```
1
```


^

機能

べき乗

形式

$$x \wedge y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
% y = 3
% x ^ y
125
```

十二

機能

加算代入

形式

$y += x$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 出力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% y = 10
% y += 3
% y
13
```

-二

機能

減算代入

形式

```
y -= x
```

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 出力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% y = 10  
% y -= 3  
% y  
7
```

***二**

機能

乗算代入

形式

$y \ *= \ x$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 出力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% y = 5
% y *= 10
% y
50
```

/=

機能

除算代入

形式

$y \ /= \ x$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 出力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% y = 10
% y /= 5
% y
2
```

%=

機能

剰余算代入

形式

$$y \ \% = \ x$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 出力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% y = 10
% y %= 3
% y
1
```

$\wedge=$

機能

べき乗代入

形式

$$y \wedge= x$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 出力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% y = 5
% y ^= 3
% y
125
```

||

機能

論理和 (OR)

形式

$x \ || \ y$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 1
% y = 0
% x || y
1
```


&&

機能

論理積 (AND)

形式

$x \ \&\& \ y$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 1
% y = 1
% x && y
1
```

!

機能

論理否定 (NOT)

形式

$!x$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 1~5
```

```
% x
```

```
[0]:%    1    2    3    4    5
```

```
% !x
```

```
[0]:%    0    0    0    0    0
```

!=

機能

要素毎の比較演算 (等しくない)

形式

$$x \neq y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5  
% y = 10  
% x != y  
1
```

==

機能

要素毎の比較演算 (等しい)

形式

$$x == y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
% y = 5
% x == y
1
```

>

機能

要素毎の比較演算 (より大きい)

形式

$$x > y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 10
% y = 5
% x > y
1
```

>=

機能

要素毎の比較演算 (以上)

形式

$x \geq y$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 10
% y = 5
% x >= y
1
```

<

機能

要素毎の比較演算 (より小さい)

形式

$$x < y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
% y = 10
% x < y
1
```

<=

機能

要素毎の比較演算 (以下)

形式

$$x \leq y$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
% y = 10
% x <= y
1
```


++

機能

インクリメント演算．前置および後置が可能で代入文で利用すると演算実行順序が変わる．

形式

`x++`

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
% x++
% x
6
```

--

機能

デクリメント演算．前置および後置が可能で代入文で利用すると演算実行順序が変わる．

形式

`x--`

パラメータ

`x` : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 5
% x--
% x
4
```

~

機能

1 等差数列の生成

形式

$$x \sim y$$

パラメータ

x : 初項 (Scalar, Series, Snapshot)

y : 末項 (Scalar, Series, Snapshot)

使用例

```
% a = 1~5
```

```
% a
```

```
[0]:%    1    2    3    4    5
```

()

機能

変数の結合

形式

`(x, y)`

パラメータ

`x` : 入力オブジェクト (Scalar, Series, Snapshot)`y` : 入力オブジェクト (Scalar, Series, Snapshot)

使用例

```
% x = 3~7
% y = 1~10
% (x,y)
[ 0]:%    3    4    5    6    7
[ 5]:%    1    2    3    4    5
[10]:%    6    7    8    9   10
```

[]

機能

指定したインデックスの時系列 (Series) の切り出し

形式

```
x[ y ]
```

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 指定時系列 (Scalar, Series, Snapshot)

使用例

```
% a = 1~10
% b = reform(a,(5,2))
% b
[0]:[0]%    1    2
[1]:[0]%    3    4
[2]:[0]%    5    6
[3]:[0]%    7    8
[4]:[0]%    9   10
% b[1]
[0]:%    2    4    6    8   10
```

: []

機能

任意時間の要素 (Snapshot) の切り出し

形式

$x : [\quad y \quad]$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot)

y : 指定時間 (Scalar, Series, Snapshot)

使用例

```
% a = 1~10
% b = reform(a,(5,2))
% b
[0]:[0]%    1    2
[1]:[0]%    3    4
[2]:[0]%    5    6
[3]:[0]%    7    8
[4]:[0]%    9   10
% b:[1]
[0]%    3    4
```

第 4 章 CONSTANT

組み込み定数

目次

true	48
false	49
DEG	50
E	51
GAMMA	52
PHI	53
PI	54

true

機能

真値

形式

```
true
```

解説

条件判断などで利用可能な真を示す値。

SATELLITE では , true を 1 と定義している。

参照

false

false

機能

偽値

形式

```
false
```

解説

条件判断などで利用可能な偽を示す値。

SATELLITE では , false を 0 と定義している。

参照

true

DEG

機能

1 ラジアン

形式

DEG

解説

$$DEG = \frac{180}{\pi} \doteq 57.2957$$

SATELLITE では , DEG を 57.29577951308232087680 と定義している .

E

機能

自然対数の底

形式

E

解説

$$E = e \doteq 2.7182$$

SATELLITE では , E を 2.71828182845904523536 と定義している .

GAMMA

機能

Euler-Mascheroni の定数

形式

GAMMA

解説

$GAMMA = \gamma \doteq 0.5772$

SATELLITE では , GAMMA を 0.57721566490153286060 と定義している .

PHI

機能

黄金分割比

形式

PHI

解説

$$PHI = \frac{\sqrt{5} + 1}{2} \doteq 1.6180$$

SATELLITE では , PHI を 1.61803398874989484820 と定義している .

PI

機能

円周率

形式

PI

解説

$$PI = \pi \doteq 3.1415$$

SATELLITE では , PI を 3.14159265358979323846 と定義している .

第 5 章 BUILTIN

組み込み関数

目次

alias	56
eval	57
exit	58
history	59
index	60
inline	61
isdef	62
length	63
module	64
printf	65
read	66
strlen	67
symbols	68
system	69
typeof	70
undef	71
undefall	72

alias

機能

コマンドなどの別名を表示・定義・変更・削除する．

形式

```
alias( ["nick", "real"] )
```

パラメータ

1. nick : 別名文字列 (String)
2. real : 実行コマンド (String)

解説

1. 別名が実行コマンドと同じ名前である時には，最初の 1 回だけエイリアス展開する．
2. 何もパラメータを設定しない場合には，別名定義の一覧を表示する．
3. 別名を削除したい場合には，real にヌル文字 ("") を設定すればよい．

使用例

1. 別名の定義

```
% alias( "quit", "exit" )
```

2. 別名定義の一覧表示

```
% alias()
```

3. 別名の削除

```
% alias( "quit", "" )
```


eval

機能

文字列を SATELLITE の文法の式として評価する .

形式

```
eval( "str" )
```

パラメータ

str : 文字列 (String)

使用例

文字列として式を保存しておき , 後でその式を実行する例です .

```
% string expr
% expr = "z = x + y"
% x = "this is"
% y = " a pen."
% eval(expr)
% z
this is a pen.
```

exit

機能

SATELLITE シェルを終了する .

形式

exit

パラメータ

なし

history

機能

ヒストリ (コマンド実行履歴) の大きさの設定、ヒストリの一覧を表示する。

形式

```
history( [ hsiz ] )
```

パラメータ

hsiz : ヒストリのサイズ (Scalar)

解説

1. ヒストリサイズはデフォルトで 256 に設定されている。
2. ヒストリサイズを指定しない場合には、ヒストリの一覧が表示される。
3. ヒストリサイズに 0 または負値を与えると、ヒストリサイズが出力される。

index

機能

オブジェクトのインデックスを得る．

形式

```
idx = index( x )
```

パラメータ

1. x: オブジェクト (Series , Snapshot , String)
2. idx: 要素数出力 (Series , Scalar)

解説

データのインデックスを出力する．一次元データであれば Scalar 値が，多次元データであれば Series 値で出力される．

使用例

```
% t=(0~99)/100
% x1=sin(2*PI*t)
% x2=sin(10*PI*t)
% x=merge(x1,x2)
% index(x)
200
% y=reform(x,(2,100))
% index(y)
[0]:%           2           100
```

inline

機能

SATELLITE 言語のスクリプトを実行する .

形式

```
inline( "script" )
```

パラメータ

script : スクリプトファイル名 (String)

解説

1. inline 関数で使用するオブジェクトは , スクリプトの実行終了後も有効であり , func , proc とは異なる .
2. 戻り値はない .

使用例

sample.sl を実行する

```
% inline( "sample.sl" )
```

isdef

機能

オブジェクト名が使用されているかどうかを調べる．

形式

```
flag = isdef( name )
```

パラメータ

1. name : オブジェクト名
2. flag : 結果 (Scalar)

解説

1. Scalar , Series , Snapshot , String 型オブジェクト , もしくは定数として定義されている場合には 1 を返す .
2. オブジェクト毎が定義されていない場合 0 を返す .
3. モジュール名 , モジュールに含まれる関数 , func , proc , などの名前で使われている場合には , -1 を返す .

length

機能

オブジェクトの要素数を得る .

形式

```
num = length( x )
```

パラメータ

1. x : オブジェクト (Series , Snapshot , String)
2. num : 要素数 (Scalar)

解説

多次元データの時系列方向の要素数を出力する . 例えば , インデックスが [10][50] のデータであれば , 出力は 10 である .

使用例

```
% t=(0~99)/100
% x1=sin(2*PI*t)
% x2=sin(10*PI*t)
% x=merge(x1,x2)
% length(x)
200
% y=reform(x,(2,100))
% length(y)
2
```

module

機能

コマンドモジュールを登録する .

形式

```
module( module_object )
```

パラメータ

module_object : モジュール名

解説

1. モジュール名は , define 宣言によって , 諸パラメータの設定されているものでなければならない .
2. 同じコマンド名を含むモジュールを登録した場合 , 新しく登録されたモジュールのコマンドの方が実行されるようになる .
3. モジュールの定義については , define を参照 .
4. 標準モジュールとして , 現在 BUILTIN , UTILITY , SYSTEM , ISPP , GPM , BPS , NCS , NPE , DCM , STATISTICS が用意されており , モジュール名は定義済みである .

参照

define

printf

機能

オブジェクトの内容を指定した形式でする .

形式

```
printf( "format", x [, y, ...] )
```

パラメータ

1. format : フォーマット文字列 (String)
2. x, y, ... : オブジェクト (Scalar , Series , Snapshot , String)

解説

フォーマット文字列に従って文字を画面出力する . フォーマット文字列の書式は , C 言語と同様である .

使用例

整数 i , 実数 a を表示する .

```
% printf( "%4d : %8.3f\n", i, a );
```

read

機能

指定した型のデータをキーボードから読み込む。

形式

```
x = read( type )
```

パラメータ

1. x : 出力オブジェクト (Series , Snapshot , Scalar , String)
2. type : データ型

解説

x には、キーボードから入力した値を指定した型に変換した値が入る。

strlen

機能

文字列の長さを得る .

形式

```
len = strlen( "str" )
```

パラメータ

1. str : 文字列 (String)
2. len : 文字列の長さ (Scalar)

symbols

機能

シンボルテーブルを表示する .

形式

```
symbols( )
```

パラメータ

なし

解説

定義されているシンボル (オブジェクト名 , 定義名 , モジュール名) を表示する .

system

機能

外部コマンドを実行する .

形式

```
out = system( "command" )
```

パラメータ

1. command : コマンド文字列 (String)
2. out : 出力 (String)

解説

外部コマンドを実行する . 外部コマンドから出力される文字列は , 戻り値として String オブジェクトで SATELLITE に渡される .

typeof

機能

オブジェクトのクラスを得る .

形式

```
str = typeof( x )
```

パラメータ

1. x : オブジェクト (Scalar , Series , Snapshot , String)
2. str : オブジェクトのクラス出力 (String)

解説

データのオブジェクトのタイプが出力される . 出力値は String 型であり , それぞれのオブジェクトクラスによって次の値が出力される .

Scalar 型 - "scalar"

Series 型 - "series"

Snapshot 型 - "snapshot"

String 型 - "string"

undef

機能

指定したシンボルを削除する。

形式

```
undef( x [, y,...] )
```

パラメータ

x, y,... : シンボル名

解説

SATELLITE シェルのシンボルテーブルから指定したシンボルを削除する。シンボルには、変数や func , proc で定義した関数・手続きも含まれる。現在定義されているシンボルの一覧を得るコマンドは、symbols である。

undefall

機能

全ての変数の登録を削除する。

形式

```
undefall( )
```

パラメータ

なし

解説

SATELLITE シェルのシンボルテーブルに定義してある変数を全て削除する。

第 6 章 Mathematic Library

数学ライブラリ

目次

abs	74
acos	75
asin	76
atan	77
atan2	78
ceil	79
cos	80
exp	81
floor	82
int	83
log	84
log2	85
log10	86
mod	87
pow	88
round	89
sgn	90
sin	91
sqrt	92
tan	93

abs

機能

オブジェクトの内容の絶対値を求める。

形式

$$y = \text{abs}(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

定義式

$$y_i = |x_i|$$

acos

機能

逆余弦を求める。

形式

$$y = \text{acos}(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

1. x は rad 表記である
2. 定義式

$$y_i = \cos^{-1} x_i$$

asin

機能

逆正弦を求める .

形式

```
y = asin( x )
```

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

1. x は rad 表記である
2. 定義式

$$y_i = \sin^{-1} x_i$$

atan

機能

逆正接を求める。

形式

$$y = \text{atan}(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

1. x は rad 表記である
2. 定義式

$$y_i = \tan^{-1} x_i$$

atan2

機能

逆正接を求める。

形式

```
z = atan2( y, x )
```

パラメータ

1. x, y : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. z : 出力オブジェクト (Scalar, Series, Snapshot)

解説

1. x は rad 表記である
2. 定義式

$$z_i = \tan^{-1} \frac{y_i}{x_i}$$

ceil

機能

オブジェクトの小数点以下を切り上げた整数値を求める。

形式

```
y = ceil( x )
```

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

最も近い最大の整数を求める

参照

floor , int , round

COS

機能

余弦を求める。

形式

$$y = \cos(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

1. x は rad 表記である
2. 定義式

$$y_i = \cos x_i$$

exp

機能

オブジェクトの指数を計算する .

形式

$$y = \exp(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

定義式

$$y_i = e^{x_i}$$

floor

機能

オブジェクトの小数点以下を切り捨てた整数値を求める。

形式

```
y = floor( x )
```

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

最も近い最小の整数を求める

参照

ceil , int , round

int

機能

オブジェクトの小数点以下を切り捨てた整数値を求める。

形式

$$y = \text{int}(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

定義式

$$y_i = \lceil x_i \rceil$$

参照

ceil , floor , round

log

機能

データの自然対数を計算する。

形式

$y = \log(x)$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

定義式

$$y_i = \log_e x_i$$

log2

機能

データの底が 2 の対数を計算する .

形式

$$y = \log_2(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

定義式

$$y_i = \log_2 x_i$$

log10

機能

データの常用対数を計算する。

形式

$$y = \log_{10}(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

定義式

$$y_i = \log_{10} x_i$$

mod

機能

剰余を求める。

形式

$$z = \text{mod}(x, y)$$

パラメータ

1. x, y : 入力オブジェクト (Scalar , Series , Snapshot, File)
2. z : 出力オブジェクト (Scalar , Series , Snapshot)

解説

定義式

$$z_i = x_i \% y_i$$

pow

機能

累乗を求める。

形式

```
z = pow( x, y )
```

パラメータ

1. x, y : 入力オブジェクト (Scalar , Series , Snapshot, File)
2. z : 出力オブジェクト (Scalar , Series , Snapshot)

解説

定義式

$$z_i = x_i^{y_i}$$

round

機能

オブジェクトの小数点以下を四捨五入した整数値を求める。

形式

```
y = round( x )
```

パラメータ

1. x: 入力オブジェクト (Scalar , Series , Snapshot, File)
2. y: 出力オブジェクト (Scalar , Series , Snapshot)

解説

値の四捨五入を求める。ただし、内部的な 2 進数表現と 10 進数表現の差により生じる丸め誤差の影響により、必ずしも小数点以下を四捨五入した結果を返さないことがある。

参照

ceil , floor , int

sgn

機能

符号を得る .

形式

$$y = \text{sgn}(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar , Series , Snapshot, File)
2. y : 符号 (Scalar)

解説

1. 符号を判定し 1 , 0 , -1 のいずれかを返す .
 - $1 : x > 0$ の場合
 - $0 : x == 0$ の場合
 - $-1 : x < 0$ の場合
2. 定義式

$$y_i = \text{sgn}(x_i) = \frac{x_i}{|x_i|}$$

sin

機能

正弦を求める。

形式

$$y = \sin(x)$$

パラメータ

x : 入力オブジェクト (Scalar, Series, Snapshot, File)

1. x : 入力オブジェクト (Scalar, Series, Snapshot, File)
2. y : 出力オブジェクト (Scalar, Series, Snapshot)

解説

1. x は rad 表記である
2. 定義式

$$y_i = \sin x_i$$

sqrt

機能

平方根を求める .

形式

```
y = sqrt( x )
```

パラメータ

1. x : 入力オブジェクト (Scalar , Series , Snapshot, File)
2. y : 出力オブジェクト (Scalar , Series , Snapshot)

解説

定義式

$$y_i = \sqrt{x_i}$$

tan

機能

正接を求める。

形式

$$y = \tan(x)$$

パラメータ

1. x : 入力オブジェクト (Scalar , Series , Snapshot, File)
2. y : 出力オブジェクト (Scalar , Series , Snapshot)

解説

1. x は rad 表記である
2. 定義式

$$y_i = \tan x_i$$

第 7 章 UTILITY

ユーティリティ

目次

dump	95
exist	96
extract	97
indexsize	98
puts	99
revtime	100
Scalar	101
Series	102
Snapshot	103
sprintf	104
String	105
text2string	106
tolower	107
toupper	108
unit	109

dump

機能

変数の内容を表示する .

形式

```
dump( x )
```

パラメータ

x: 入力オブジェクト

exist

機能

ファイルがあるか調べる関数 .

形式

```
flag = exist( "filename" )
```

パラメータ

1. filename : 対象ファイル名 (String)
2. flag : 結果 (Scalar)
 - 1 : 存在
 - 0 : 存在しない

extract

機能

変数のインデックスを変更し，元の値をそのままはめ込む．

形式

```
y = extract( x, index )
```

パラメータ

1. x: 入力オブジェクト
2. index: 変更後のインデックス (Scalar, Series)
3. y: 出力オブジェクト

解説

reform 関数では，大きさは変更できるが，値は先頭から順番に詰め込まれる．本関数は元のデータの並びを保ち，且つインデックスを変更したい場合に用いるとよい．

使用例

```
% x=1  
% y=extract(x,2)  
% y  
[0]:%    1        0
```

参照

reform

indexsize

機能

変数の全体の要素数を求める .

形式

```
y = indexsize( x )
```

パラメータ

x: 入力オブジェクト

y: 出力オブジェクト (Scalar)

解説

length 関数では , 多次元の場合の全体の要素数が分からないが , この関数を使えば , 全体の要素数がわかる .

使用例

```
% x=0~99  
% indexsize(x)  
100
```

puts

機能

オブジェクトを画面出力する .

形式

```
puts( x )
```

パラメータ

x: 入力オブジェクト

使用例

```
% x=0~9
% puts(x)
[0]:%    0    1    2    3    4
[5]:%    5    6    7    8    9
```

revtime

機能

時間軸方向を逆転する .

形式

```
y = revtime( x )
```

パラメータ

1. x: 入力オブジェクト
2. y: 出力オブジェクト

解説

多次元データでは , reverse 関数の結果と異なる .

使用例

```
% x=0~9
% y=revtime(x)
% y
[0]:%    9    8    7    6    5
[5]:%    4    3    2    1    0
```

参照

reverse

Scalar

機能

指定した変数を Scalar 型の値にして返す .

形式

```
y = Scalar( x )
```

パラメータ

1. x : 入力オブジェクト
2. y : 出力オブジェクト (Scalar)

解説

1 つの実数値が返される . Series などでは , 先頭の値が変換される .

使用例

```
% x=0~9
% y=Scalar(x)
% y
0
% z=revtime(x)
% z1=Scalar(z)
% z1
9
```

参照

Snapshot , Series , String

Series

機能

指定した変数を Series 型の値にして返す .

形式

```
y = Series( x )
```

パラメータ

1. x : 入力オブジェクト
2. y : 出力オブジェクト (Series)

使用例

```
% x=1  
% x  
1  
% y=Series(x)  
% y  
[0]:%    1
```

参照

Scalar , Snapshot , String

Snapshot

機能

指定した変数を Snapshot 型の値にして返す .

形式

```
y = Snapshot ( x )
```

パラメータ

1. x : 入力オブジェクト
2. y : 出力オブジェクト (Snapshot)

使用例

```
% x=1  
% x  
1  
% y=Snapshot(x)  
% y  
[0]:%    1
```

参照

Scalar , Series , String

sprintf

機能

数値を指定した形式で，文字列に変換する．

形式

```
str = sprintf( "format", x1, [x2, ..., x15] )
```

パラメータ

1. format : 出力フォーマット (String)
2. x1,x2,...x15 : 出力したいデータ (Scalar)
3. str : 出力文字列 (String)

解説

1. format は，printf と同じ記述方法．引数が多数指定できる (15 個まで)．フォーマット文字列に含まれるスペースは，その数だけ出力される．
2. ただし，引数に指定できるのは，Scalar 値のみであり，Series，Snapshot 値が指定できた場合は，その先頭の値のみを処理する．

String

機能

指定した変数を String 型の値にして返す .

形式

```
y = String( x )
```

パラメータ

1. x : 入力オブジェクト
2. y : 出力オブジェクト (String)

参照

Scalar , Series , Snapshot

text2string

機能

テキストファイルの内容を String 型に変換する .

形式

```
str = text2string( "filename" )
```

パラメータ

1. filename : テキストファイル名 (String)
2. str : 出力文字列 (String)

解説

データの区切り文字は , スペース , タブ , 改行である .

参照

text2buffer , buffer2text

tolower

機能

小文字に変換した文字列を返す .

形式

```
out = tolower( "string" )
```

パラメータ

1. string : 入力文字列 (String)
2. out : 出力文字列 (String)

使用例

```
% tolower( "toLower" )  
tolower
```

参照

toupper

toupper

機能

大文字に変換した文字列を返す .

形式

```
out = toupper( "string" )
```

パラメータ

1. string : 入力文字列 (String)
2. out : 出力文字列 (String)

使用例

```
% toupper( "toUpper" )  
TOUPEER
```

参照

tolower

unit

機能

単位行列を生成する .

形式

```
imat = unit( dim )
```

パラメータ

1. dim : 次元数 (Scalar)
2. imat : 出力オブジェクト (Snapshot)

使用例

```
% imat = unit(2)
% imat
[0]:[0]%    1    0
[1]:[0]%    0    1
```

第 8 章 SYSTEM

システム関連モジュール

目次

bm	111
cut	112
fill	113
find	114
get	116
getenv	117
header	118
help	119
max	120
maxpos	121
merge	123
min	124
maxpos	125
mkdir	127
put	128
reform	130
reverse	131
rotate	132
sampling	133
setenv	134
tempdir	135
thin	136
wait	138

bm

機能

バッファモニター，バッファのモニタリングを行う．

形式

```
bm( x )
```

パラメータ

x : 表示するオブジェクト (Series, Snapshot)

解説

1. オブジェクト x の変化を逐次グラフ表示する．
2. バッファモニターを実行中に，モニタリング対象オブジェクトを再宣言すると，バッファモニターは機能しなくなるため，注意が必要．
3. 現在 Windows のみで動作する．

cut

機能

オブジェクトの一部を取り出す。

形式

```
y = cut( x, start, end )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 出力オブジェクト (Series)
3. start : 切り出すデータの始点座標 (Scalar, Series)
4. end : 切り出すデータの終点座標 (Scalar, Series)

解説

1. x の start から , end の位置までのデータを取り出す。
2. start , end には , x が多次元データならば index (Series) を , 2 次元データならば data point (Scalar) を設定する。
3. end の値は , \geq start でなければならない。

使用例

2 次元の場合

```
[0]:[0]%   1    2    3    4    5
[1]:[0]%   6    7    8    9   10
```

という x に対し , cut(x,(0,1),(0,3)) を実行すると ,

```
[0]:[0]%   2    3    4
```


fill

機能

オブジェクトの指定した範囲を指定した値で埋める。

形式

```
y = fill( x, start, end, value )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 出力オブジェクト (Series)
3. start : 始点座標 (Scalar, Series)
4. end : 終点座標 (Scalar, Series)
5. value : 埋める数値 (Scalar)

解説

1. x の start から , end の位置までの範囲を value の値で満たす。
2. start , end には , x が多次元ならば index (Series) を , 1 次元データならば data point (Scalar) を設定する。
3. end の値は , \geq start でなければならない。

使用例

2 次元の場合

```
[0]:[0]%   1    2    3    4    5
[1]:[0]%   6    7    8    9   10
```

という x に対し , fill(x,(0,1),(0,3),30) を実行すると ,

```
[0]:[0]%   1   30   30   30    5
[1]:[0]%   6    7    8    9   10
```

find

機能

入力オブジェクトから指定した値に最も近いデータを見つけ、値とその位置を表示し、位置を Series オブジェクトとして返す。

形式

```
ip = find( x, val, num )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. ip: データの位置 (Series)
3. val: 探し出す値 (Scalar)
4. num: 探し出すデータの個数 (Scalar)

解説

1. 入力オブジェクト内から val に最も近い値を持つインデックスの系列を取得する。
2. インデックスの系列は、次元数分の要素を持ち、時間方向に num 個得られる。
3. 取得するデータ数が 1 個で、かつ 1 次元データの位置を示す場合には戻り値は、Scalar オブジェクトである。

使用例

1. 1 次元の場合

```
[0]% 11 12 13 14 15
```

という x に対し、find(x,14,3) を実行すると、

```
[0]:[0]% 3
[1]:[0]% 4
[2]:[0]% 2
```

2. 2 次元の場合

```
[0]:[0]% 11 12 13 14 15
[1]:[0]% 16 17 18 19 20
```

という x に対し, `find(x,14,3)` を実行すると,

```
[0]:[0]% 0 3
[1]:[0]% 0 4
[2]:[0]% 0 2
```

get

機能

オブジェクトの指定した位置の値を得る .

形式

```
y = get( x, position )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 指定した位置のデータ値 (Scalar)
3. position : データの座標 (Scalar)

解説

1. x の position の位置のデータ値を返す .
2. position には , x が多次元データならば index (Series) を , 1 次元データならば data point (Scalar) を設定する .

使用例

2 次元の場合

[0]:[0]%	1	2	3	4	5
[1]:[0]%	6	7	8	9	10

という x に対し , get(x,(0,3)) を実行すると ,

```
[0]:[0]%    4
```

getenv

機能

指定した文字列の環境変数の値を得る .

形式

```
val = getenv( "name" )
```

パラメータ

1. name : 環境変数の名前 (String)
2. val : 環境変数の値 (String)

解説

name に指定されている文字列の環境変数の値を返す .

header

機能

データファイルの情報を表示，変更する．

形式

```
header( "file_name", mode )
```

パラメータ

1. file_name : データファイル名 (String)
2. mode : モード (Scalar)
 - 0 : 画面に表示する．
 - 1 : 変更する．

解説

1. データファイル file_name の情報を，mode に従って処理する．
2. 表示・変更される情報には，Byte Order，Data size，Owner，Date，Comment，Sampling frequency，Dimension，Index などがある．

help

機能

マニュアルを表示する .

形式

```
help( "com_name" )
```

パラメータ

com_name : 関数名 (String)

解説

1. 関数 com_name の機能 , 形式 , パラメータなどを表示する .

max

機能

データの最大値を得る。

形式

$$y = \max(x)$$

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 最大値 (Scalar)

maxpos

機能

データの最大値のデータ位置を得る。

形式

```
y = maxpos( x, num )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 最大値データの位置 (Scalar, Series)
3. num : 取得するデータ位置の数 (≥ 1) (Scalar)

解説

1. 入力オブジェクト内で最も大きい値を持つインデックスの系列を取得する。
2. インデックスの系列は、次元数分の要素を持ち、時間方向に num 個得られる。
3. 取得するデータ数が 1 個で、かつ 1 次元データの位置を示す場合には戻り値は、Scalar オブジェクトである。

使用例

1. 1 次元の場合

```
[0]% 13 12 15 11 14
```

という x に対し、maxpos(x,1) を実行すると、

```
2
```

maxpos(x,3) を実行すると、

```
[0]:[0]% 2
[1]:[0]% 4
[2]:[0]% 0
```

2. 2 次元の場合

```
[0]:[0]% 13 17 15 11 18
[1]:[0]% 20 12 16 19 14
```

という x に対し, $\text{maxpos}(x,3)$ を実行すると,

```
[0]:[0]% 1 0
[1]:[0]% 1 3
[2]:[0]% 0 4
```

merge

機能

2 つのオブジェクトのデータを連結する .

形式

```
z = merge( x, y )
```

パラメータ

1. x : 入力オブジェクト 1 (Series, Snapshot)
2. y : 入力オブジェクト 2 (Series, Snapshot)
3. z : 出力オブジェクト (Series)

解説

1. z には , x の終端に y を連結したデータが出力される .
2. 多次元データの場合には , サブインデックスが一致しなければ連結しない .

使用例

2 次元の場合

x :		y :	
[0]:[0]%	1 2	[0]:[0]%	30 31
[1]:[0]%	3 4	[1]:[0]%	32 33
[2]:[0]%	5 6	[2]:[0]%	34 35
[3]:[0]%	7 8		

に対し , merge(x,y) を実行すると ,

[0]:[0]%	1 2
[1]:[0]%	3 4
[2]:[0]%	5 6
[3]:[0]%	7 8
[4]:[0]%	30 31
[5]:[0]%	32 33
[6]:[0]%	34 35

min

機能

データの最小値を得る .

形式

$y = \min(x)$

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 最小値 (Scalar)

maxpos

機能

データの最小値のデータ位置を得る。

形式

```
y = minpos( x, num )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. y: 位置格納オブジェクト (Scalar, Series)
3. num: 取得するデータ位置の数 (≥ 1)

解説

1. 入力オブジェクト内で最も小さい値を持つインデックスの系列を取得する。
2. インデックスの系列は、次元数分の要素を持ち、時間方向に num 個得られる。
3. 取得するデータ数が 1 個で、かつ 1 次元データの位置を示す場合には戻り値は、Scalar オブジェクトである。

使用例

1. 1 次元の場合

```
[0]% 13 12 15 11 14
```

という x に対し、minpos(x,1) を実行すると、

```
3
```

minpos(x,3) を実行すると、

```
[0]:[0]% 3
[1]:[0]% 1
[2]:[0]% 0
```

2. 2 次元の場合

```
[0]:[0]% 13 17 15 11 18  
[1]:[0]% 20 12 16 19 14
```

という x に対し, minpos(x,3) を実行すると,

```
[0]:[0]% 0 3  
[1]:[0]% 1 1  
[2]:[0]% 0 0
```

mkdir

機能

新しくディレクトリを作成する .

形式

```
mkdir( "newdir" [, force] )
```

パラメータ

1. newdir : 作成ディレクトリ名 (String)
2. force : 強制フラグ (Scalar)

解説

1. ディレクトリの作成に失敗するとエラーを返し処理を中断する .
2. force の値が 0 でない場合 , ディレクトリの作成に失敗してもエラーを返さず処理を続行する .

使用例

```
% mkdir( "newdir" );
```

put

機能

オブジェクトの一部に，別のデータを入れる．

形式

```
z = put( x, y, index )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 挿入オブジェクト (Series, Snapshot)
3. z : 出力オブジェクト (Series)
4. index : y の値を書き込む座標

解説

1. x の index の位置から，y の内容を上書きしたデータを出力する．但し，z は，x の時間軸方向の長さに依存する．
2. x が多次元ならば index は Series 値で，1 次元ならば Scalar 値で設定する．
3. 多次元データの場合には，サブインデックスが一致しなければ上書きしない．

使用例

2 次元の場合

x :		y :	
[0]:[0]%	1 2	[0]:[0]%	30 31
[1]:[0]%	3 4	[1]:[0]%	32 33
[2]:[0]%	5 6	[2]:[0]%	34 35
[3]:[0]%	7 8		
[4]:[0]%	9 10		

に対し，put(x,y,(1,0)) を実行すると，

[0]:[0]%	1 2
[1]:[0]%	30 31


```
[2]:[0]% 32 33  
[3]:[0]% 9 10
```

が得られる .

reform

機能

オブジェクトのフォーマットを変更する。

形式

```
y = reform( x, index )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 出力オブジェクト (Series)
3. index : データサイズ (Series, Series)

解説

1. 指定したサイズが入力オブジェクトより大きい場合には、データの後部に 0 詰めを行い、入力オブジェクトより小さい場合には、データの削除を行って、オブジェクトのフォーマットを変更する。
2. x が多次元ならば index は Series 値で、1 次元ならば Scalar 値で設定する。

使用例

```
x = (1,2,3,4,5,6,7,8,9,10)
```

に対し、reform(x,(2,5))を実行すると、

```
[0]:[0]% 1 2 3 4 5
[1]:[0]% 6 7 8 9 10
```

reform(x,(2,7))を実行すると、

```
[0]:[0]% 1 2 3 4 5 6 7
[1]:[0]% 8 9 10 0 0 0 0
```

reform(x,(2,3))を実行すると、

```
[0]:[0]% 1 2 3
[1]:[0]% 4 5 6
```

reverse

機能

入力データ配列を逆にしたデータを得る。

形式

```
y = reverse( x )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. y: 出力オブジェクト (Series)

使用例

2 次元の場合

```
[0]:[0]%   1     2     3     4     5
[1]:[0]%   6     7     8     9    10
```

という x に対し, reverse(x) を実行すると,

```
[0]:[0]%  10     9     8     7     6
[1]:[0]%   5     4     3     2     1
```

rotate

機能

オブジェクトの内容を、指定した位置が先頭となるように移動する。

形式

```
y = rotate( x, index )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. y: 出力オブジェクト (Series)
3. index: 先頭にする座標

解説

1. データの先頭から index までのデータは、オブジェクトの最後に移動される。
2. index が負であった場合、|index| の位置に入力の先頭が移動される。
3. x が多次元ならば index は Series 値で、1 次元ならば Scalar 値で設定する。

使用例

2 次元の場合

[0]:[0]%	1	2	3	4	5
[1]:[0]%	6	7	8	9	10

という x に対し、rotate(x,(0,3)) を実行すると、

[0]:[0]%	4	5	1	2	3
[1]:[0]%	9	10	6	7	8

rotate(x,(0,-3)) を実行すると、

[0]:[0]%	3	4	5	1	2
[1]:[0]%	8	9	10	6	7

sampling

機能

サンプリング周波数を変更する。

形式

```
new_freq = sampling( freq )
```

パラメータ

1. freq : サンプリング周波数 (単位 : Hz) (Scalar)
2. new_freq : 設定されたサンプリング周波数 (単位 : Hz) (Scalar)

解説

1. サンプリング周波数を，指定した値に変更する．但し，freq のデフォルトは，1000.0 である．
2. 戻り値には，設定値を返す．
3. 引数に 0 以下の数値を渡した場合，サンプリング周波数の変更は行われない．
4. ここで設定するサンプリング周波数は，ISPP モジュールの関数や，GPM モジュールの graph 関数に参照される．

setenv

機能

環境変数を指定した文字列に設定する。

形式

```
setenv( "name", "val" )
```

パラメータ

1. name : 環境変数の名前 (String)
2. val : 環境変数の値 (String)

解説

環境変数 name に文字列 val を設定する。

tempdir

機能

テンポラリディレクトリを取得する。

形式

```
dir = tempdir( )
```

パラメータ

dir: テンポラリディレクトリ名 (String)

解説

SATELLITE システムが利用しているテンポラリディレクトリのディレクトリ名の絶対パスを取得する。

使用例

```
% dir = tempdir();
```

thin

機能

データを指定したステップ数で間引く。

形式

```
y = thin( x, step )
```

パラメータ

1. x : 入力時系列 (Series)
2. y : 出力時系列 (Series)
3. step : 間引きの間隔 (> 1)

解説

1. 入力時系列の step-1 点分のデータを間引いて出力する。
2. step = 0 および 1 の場合はデータは変化しない。
3. step には、x が多次元データならば Series を、1 次元データならば Scalar を設定する。
4. 多次元オブジェクトの入力に対しては、次のように間引かれる。

5x10 の 2 次元オブジェクトの場合

```
y = thin( x, (2, 3))
```

y には、

```
y:[0]  x00    x03    x06    x09
y:[1]  x20    x23    x26    x29
y:[2]  x40    x43    x46    x49
```

が、出力される。

使用例

1. 1 次元の場合

```
[0]:%    1    2    3    4    5    6    7    8    9   10
という x に対し、
```


`thin(x, 2)`を実行すると

```
[0]:%    1      3      5      7      9
```

2. 2次元の場合

```
[0]:[0]%    11      12      13      14      15
```

```
[1]:[0]%    16      17      17      18      19
```

```
[2]:[0]%    21      22      23      24      25
```

```
[3]:[0]%    26      27      28      29      39
```

```
[4]:[0]%    31      32      33      34      35
```

という x に対し,

`thin(x, (2,3))`を実行すると

```
[0]:[0]%    11      14
```

```
[1]:[0]%    21      24
```

```
[2]:[0]%    31      34
```

wait

機能

指定した時間 (秒) もしくは、リターンキーが押されるまで待つ。

形式

```
wait( [time] )
```

パラメータ

1. time : 待つ時間 (秒) (Scalar)
2. time : 入力待ちコメント (String)
3. time を与えない場合にはリターンが押されるまで待つ

解説

デモンストレーションなどで、画面を静止状態、一定時間で自動実行にしたい場合などに用いる。

使用例

1. 一秒待つ。

```
% wait(1);
```

2. 入力待ちコメントを表示し、リターンキーが押されるまで一時停止する。

```
% wait("Pause > ");
Pause >
```

3. リターンキーが押されるまで一時停止する。

```
% wait();
* * * * Hit Return key :
```

第 9 章 ISPP

ディジタル信号処理パッケージ

目次

arand	140
argen	142
burg	144
butwmake	146
cep	148
dccut	149
det	150
eigen	151
fftc	153
fftn	154
fir	155
firmake	157
gauss2	159
hil	160
icep	161
iir	162
iirmake	164
interp	165
interp2	167
inv	168
levin	170
mnrand	172
mul	174
nmeq	175
norm	177
nrand	179
phase	180
pole	181
power	182
rank	183
spcf	184
sum	186
trans	187
urand	188
window	189

arand

機能

任意の経験分布関数に従う乱数を生成する。

形式

```
x = arand( dpt, seed, f, min, max )
```

パラメータ

1. x : 出力時系列 (Series)
2. dpt : 出力データ点数 (Scalar)
3. seed : 乱数の初期値 (奇数) (Scalar)
4. f : 経験分布関数バッファ (Series , Snapshot)
5. min : 経験分布関数定義域下限 (乱数最小値) (Scalar)
6. max : 経験分布関数定義域上限 (乱数最大値) (Scalar)

解説

1. 乱数のシードは、M 系列を用いて生成している。
2. 乱数生成のアルゴリズムは、逆関数法を用いている。
3. 経験分布関数 f の値域は、 $0 \leq f \leq 1$ の区間に入っていないなければならない。

使用例

コーシー分布に従う乱数を発生させる。

```
% func cauchy(x) {                                     # コーシー分布の
%   return 1 / (PI * (x^2 + 1));                       # 密度関数定義
% }                                                     #
%                                                       #
% series f;                                             #
% x = (0~999) / 1000 - 0.5;                             # 分布関数の定義域設定
% ff = cauchy(x);                                       # 密度関数の値設定
% a = sum(ff,f);                                         # 密度関数を積算する
```

```
% f = f / a; # 経験分布関数の設定  
% data = arand(1024,1,f,min(x),max(x)); # 乱数生成
```

argen

機能

線形 AR モデルに従う信号を生成する。

形式

```
x = argen( para, dpt )
```

パラメータ

1. para : パラメータ (Scalar, Series, Snapshot)
 - 0 : パラメータを会話的に設定し，オブジェクト x に書き込む。
 - オブジェクト名 : パラメータを para から読み込んで実行する。
2. dpt : 出力データ点数 (Scalar)
3. x : 出力時系列 (Series)

解説

1. para が 0 の場合はパラメータを会話的に設定し，出力オブジェクトに返す．para がオブジェクト名の場合は，指定したパラメータバッファからパラメータを読み込んで，AR 信号を生成する．
2. パラメータ設定は，画面にコメントが表示されるので，それに従って数値を入力すればよい．
3. パラメータのフォーマットは，次の通りである．
 - [0]入力正規乱数の初期値 (nrand 関数の init と同じく，奇数)
 - [1]Ignored Iteration (ii)
 - [2]AR モデルの次数 (n)
 - [3]AR モデルの係数 (次数 1)
 -
 - [n+2]AR モデルの係数 (次数 n)
 - [n+3]入力正規乱数の分散 (入力分散)
4. 最初の ii 個は出力されず，その後の dpt 個のデータが x に出力される．
5. 入力正規乱数は nrand 関数と同じモジュールを使用．

6. 入力分散が設定されていない場合は 1.0 とする .
7. `burg` , `levin` 関数で得られたパラメータオブジェクトは直接使用できない (以下の使用例を参照) .

使用例

$$x_k + 0.8x_{k-1} = e_k, \quad e_k \sim N(0.0, 2.0)$$

に従う信号を生成する .

- パラメータバッファ `para` の生成

```
% para = argen(0,0)
Init : 3                                # 入力乱数の初期値 ( 奇数 )
Ignored Iteration : 10                  # Ignored Iteration
Order of AR : 1                         # AR モデルの次数
AR Coef[1] : 0.8                        # AR 係数
Input Variance : 2.0                   # 入力分散
% para                                 # 設定されたパラメータ
[0]:% 3    10    1    0.8    0.2      # バッファの内容
```

- 信号の生成

```
% x = argen(para,1024)                  # para の内容に従って 1024
                                          # 点データを生成
```

- `burg` , `levin` 関数で求めたパラメータ `para` を使用する場合 (入力分散 1.0)

```
% tmp = (3,10,length(para))            # Init , Order を格納
% para2 = merge(tmp,para);
% para2:[length(para2)] = 1.0          # 入力分散
```

信号生成には `para2` を用いる .

参照

`nrand` , `burg` , `levin`

burg

機能

時系列に線形 AR モデルをあてはめ，Burg 法を用いて AR 係数を求め，パワースペクトルを出力する．

形式

```
burg( x, pw, ret1, mode, odr, dpt, tola, ret2 )
```

パラメータ

1. x : 入力時系列 (Series, Snapshot)
2. pw : 出力時系列:パワースペクトル (Series)
3. ret1 : 出力 1 (Series)
4. mode : モード (0, "A", "E", "K", "F", "C")
5. odr : モデルの最大次数 (Scalar)
6. dpt : パワースペクトルの出力データ点数 (Scalar)
7. tola : 次数を下げる基準 (0, 1, 2 or 3)
8. ret2 : 出力 2 (Series)

解説

1. 各モードの計算方式と出力の関係は以下の通り．

0 - AIC_{min} で計算 : 出力 1 = AIC , 出力 2 = AR 係数

"A" - AIC_{min} で計算 : 但し AIC_{min} の値と total 以内であれば , 次数の少ない方を選択する . 出力は 0 と同じ .

"E" - odr の次数で計算 : 出力 1 = 誤差分散 , 出力 2 = AR 係数

"K" - odr の次数で計算 : 出力 1 = 反射係数 , 出力 2 = AR 係数

"F" - odr の次数で計算 : 出力 1 = 反射係数 , 出力 2 = 誤差系列

"C" - odr の次数で計算 : 出力 1 = 出力 2 = AR 係数

2. 入力データは , dccut 関数により直流成分を除去しておく .

使用例

$$x_k + 0.8x_{k-1} = e_k, \quad e_k \sim N(0.0, 1.0)$$

に従う信号のパワースペクトルと AR 係数を求める (odr: 固定) .

```
% x = argen(para,1024);           # 信号の生成

% series pw,sigma,coef;           # Series 変数の宣言
% burg(x,pw,sigma,"E",1,64,1,coef); # 推定
odr = 1
ndpt = 64
tola = 1.000000
P[ 1 ] = 0.956714
    AR COEFFICIENT( 1) =      0.7914492

% coef
[0]:%    0.7914
```

信号生成には para2 を用いる .

参照

levin

butwmake

機能

バターワース特性の IIR 型フィルタを設計する。

形式

```
gain = butwmake( Type, Cutoff, order, zr, zi, pr, pi )
```

パラメータ

1. Type: フィルタ特性 (1 : LowPass, 2 : HighPass, 3 : BandPass)
2. Cutoff: 遮断周波数
 - LowPass, HighPass では Scalar 型 (周波数 Hz)
 - BandPass 特性では, Series 型で設定する。例えば,

```
gain = butwmake(3, (100, 200), 3, ...)
```

のように設定する。
3. order: フィルタ次数 (入力) (Scalar)
4. zr: 零点 (実数部) 出力 (Series)
5. zi: 零点 (虚数部) 出力 (Series)
6. pr: 極 (実数部) 出力 (Series)
7. pi: 極 (虚数部) 出力 (Series)
8. gain: ゲイン (Scalar)

解説

1. 設計値は, iirmake 関数を使用すれば直接型 IIR フィルタの係数に変換できる。
2. 設計可能フィルタ次数は最大 101 次である。

使用例

User's Manual 第 4 章 3.2 節 フィルタ処理 参照

参照

iirmake

cep

機能

指定したオブジェクトの複素ケプストラムを算出する．

形式

```
cep( x, y, u, v )
```

パラメータ

1. x : 入力時系列 : 実部 (Series, Snapshot)
2. y : 入力時系列 : 虚部 (Series, Snapshot)
3. u : 出力時系列 : 実部 (Series)
4. v : 出力時系列 : 虚部 (Series)

解説

1. y に 0 を入力した場合は，虚部のデータに全て 0 が入力されたとして処理する．
2. 出力 u , v は予め宣言しておく必要がある．

参照

icep

dccut

機能

指定したオブジェクトの内容の直流成分を除去する．

形式

```
y = dccut( x )
```

パラメータ

1. x : 入力時系列 (Series, Snapshot)
2. y : 出力時系列 (Series)

解説

データの平均値を引いたものが出力される．

det

機能

2次元データを行列とみなし、行列式の値を計算する。

形式

```
y = det( x )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. y: 計算値 (Scalar)

解説

1. 正方行列のみ計算可能
2. 3次元以上のデータは、下位の2次元データを配列とみなし、すべての配列について行列式を計算し、Seriesを返す。

使用例

3次元以上のデータの場合

```
% matrix                                # 入力データ
[0]:[0][0]%    1      0
[0]:[1][0]%    0      1

[1]:[0][0]%    0      1
[1]:[1][0]%    1      0

[2]:[0][0]%    1      2
[2]:[1][0]%   -2      1

% det(matrix)
[0]:%    1    -1    5    # 計算結果
```

参照

inv

eigen

機能

2次元データを行列とみなし，固有値および固有ベクトルを数値的に求める．

形式

```
eigen( x, vec, val )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. vec: 出力オブジェクト: 固有ベクトル (Series)
3. val: 出力オブジェクト: 固有値 (Series)

解説

1. 実対象行列のみ計算可能である．
2. 固有値はハウスホルダー法による直交変換により求める．
3. あらかじめ，vec および val は Series 変数として宣言しておく必要がある．

使用例

行列 $\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$ の固有値，固有ベクトルを求める．

```
% matrix                                # 入力
[0][0]% 2    -1
[1][0]% -1    2

% series vec, val                       # 出力を格納する変数宣言

%eigen(matrix,vec,val)
0

% val                                    # 固有値
[0]:% 3    1

% vec                                    # 固有ベクトル
```

```
[0]:[0]%  -0.7071  -0.7071
[1]:[0]%   0.7071  -0.7071
```


fftc

機能

複素数入力データの離散フーリエ変換値，または逆変換値を計算する．

形式

```
fftc( flg, x, y, u, v )
```

パラメータ

1. flg : 計算方式フラグ ("P" or "I")
"P" : 正フーリエ変換
"I" : 逆フーリエ変換
2. x : 入力時系列 : 実部 (Series, Snapshot)
3. y : 入力時系列 : 虚部 (Series , Snapshot)
4. u : 出力時系列 : 実部 (Series)
5. v : 出力時系列 : 虚部 (Series)

解説

1. 入力 x , y の長さは 2 のべき乗にしなければならない．
2. 出力 u , v は予め宣言しておく必要がある．

使用例

User's Manual 第4章 3.1 節 フーリエ変換参照

参照

spcf , fftn

fftn

機能

多次元複素数入力データの離散フーリエ変換値，または逆変換値を計算する．

形式

```
fftn( flg, x, y, u, v )
```

パラメータ

1. flg : 計算方式フラグ ("P" or "I")
"P" 正フーリエ変換
"I" 逆フーリエ変換
2. x : 入力時系列 : 実部 (Series, Snapshot)
3. y : 入力時系列 : 虚部 (Series , Snapshot)
4. u : 出力時系列 : 実部 (Series, Snapshot)
5. v : 出力時系列 : 虚部 (Series, Snapshot)

解説

u , v は予め宣言しておく必要がある．

参照

fftc

fir

機能

データに対して非再帰型 (FIR) フィルタをかける。

形式

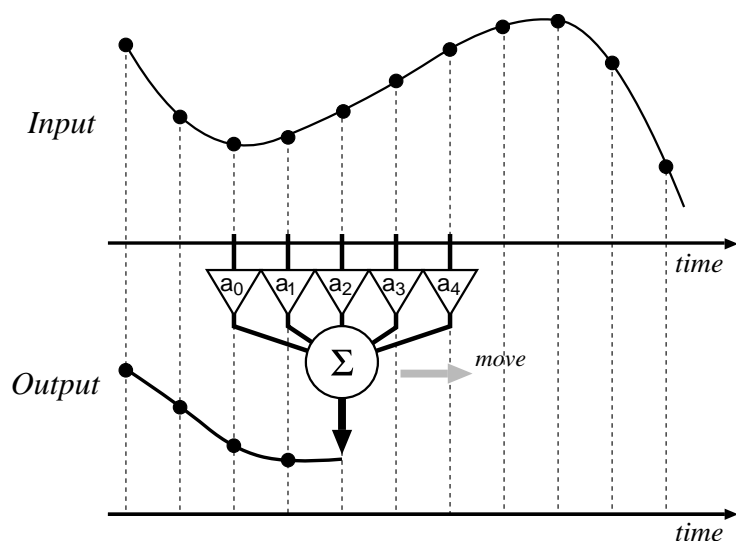
```
y = fir( prm, x[, data] )
```

パラメータ

1. prm : パラメータ系列 (Series, Snapshot)
2. x : 入力オブジェクト (Series , Snapshot)
3. y : 出力オブジェクト (Series)

解説

1. 入力が2次元のときは, 2次元 FIR フィルタをかける。
2. フィルタリング・スパン (パラメータ系列の長さ) は, 奇数値でなければならない。
3. フィルタの係数は firmake 関数を用いて生成できる。
4. FIR フィルタの処理概念図を以下に示す。



fir 関数のフィルタ処理概念図

a_0, a_1, \dots, a_4 は , 関数におけるパラメータ系列 (添字は , 系列の順番を示している) .

使用例

User's Manual 第 4 章 3.2 節 フィルタ処理参照

参照

firmake , iir

firmake

機能

非再帰型 (FIR) フィルタを設計する。

形式

```
prm = firmake( Type, Span, Cutoff, Window[, coef] )
```

パラメータ

1. Type : フィルタ特性 (1 : LowPass, 2 : HighPass, 3 : BandPass)
2. Span : フィルタ次数 (奇数)
3. Cutoff : 遮断周波数
LowPass , HighPass では Scalar 値 (周波数 Hz)
BandPass 特性では , Series で設定する。
例えば ,
Para=firmake(3,11,(100,200),3,)
のように設定する。
4. Window : 設計に使用する窓関数 (0 : 矩形窓 , 1 : ハニング窓 , 2 : ハミング窓 , 3 : ブラックマン窓 , 4 : カイザー窓)
5. coef : カイザー窓を使用する場合は , カイザー窓の係数を与える (coef > 21) 。他の窓では設定しなくてよい。
6. prm : パラメータ系列 (Series)

解説

1. フィルタ設計には , 窓関数法を用いている。
2. 2次元フィルタは設計できない。
3. フィルタリング・スパン (パラメータ系列の長さ) は , 奇数値でなければならない。

使用例

User's Manual 第4章 3.2 節 フィルタ処理参照

参照

fir , iir

gauss2

機能

2 次元ガウス関数を生成する。

形式

```
x = gauss2( size, center, r11, r12, r22, max )
```

パラメータ

1. x : 出力オブジェクト (Series)
2. size : データサイズ (インデックス)
3. center : ガウス関数の中心座標 (インデックス)
4. r11 , r12 , r22 : 共分散行列の要素 (Scalar)
5. max : ガウス関数の最大値 (Scalar)

hil

機能

ヒルベルト変換処理を行う。

形式

```
hil( x, u, v )
```

パラメータ

1. x : 入力時系列 (Series, Snapshot)
2. u : 出力時系列 : 実部 (Series)
3. v : 出力時系列 : 虚部 (Series)

解説

出力 u , v は予め宣言しておく必要がある。

icep

機能

2 つのオブジェクトのデータの逆ケプストラムを計算する .

形式

```
icep( x, y, u, v )
```

パラメータ

1. x : 入力時系列 : 実部 (Series, Snapshot)
2. y : 入力時系列 : 虚部 (Series, Snapshot)
3. u : 出力時系列 : 実部 (Series)
4. v : 出力時系列 : 虚部 (Series)

解説

1. y に 0 を入力した場合は , 虚部のデータに全て 0 が入力されたとして処理する .
2. 出力 u , v は予め宣言しておく必要がある .

参照

cep

iir

機能

データに対して再帰型 (IIR) フィルタをかける。

形式

```
y = iir( prm, x )
```

パラメータ

1. prm : パラメータ系列 (Series, Snapshot)
2. x : 入力時系列 (Series , Snapshot)
3. y : 出力時系列 (Series)

解説

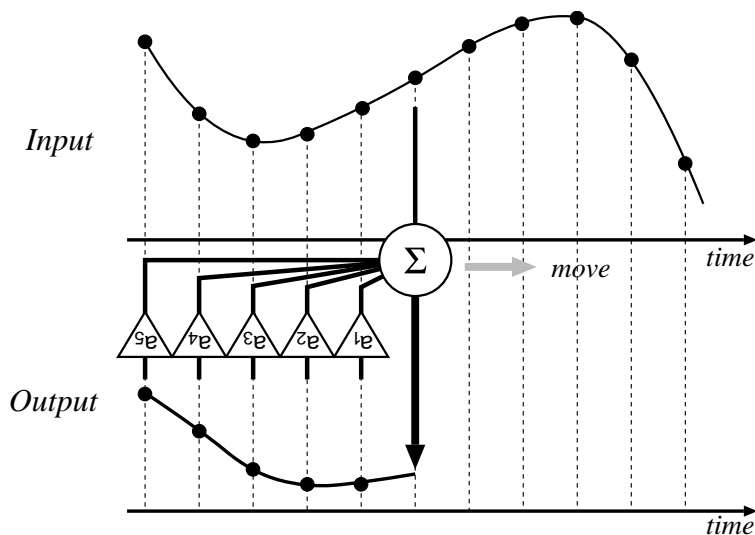
1. 入力が2次元のときは, 2次元 IIR フィルタをかける。
2. アルゴリズム

入力データを n 個の離散値 $x(i)$ (ただし, $i = 1, 2, \dots, n$) で表し, N 個の離散点からなる "重み関数" $w(j)$ (ただし, $j = 1, 2, \dots, n$) を用いる。

出力値 $y(i)$ は次のように求める。

$$y(i) = \sum_{j=1}^N w(j)y(i-j) + x(i)$$

iir 関数の処理概念図を以下に示す。



iir 関数のフィルタ処理概念図

a_0, a_1, \dots, a_4 は、関数におけるパラメータ系列 (添字は、系列の順番を示している)。

- 通常の方法で設計される直接型の IIR フィルタによる処理を行うには、次のようにする。para_a, para_b はそれぞれ伝達関数 $H(z)$ の分母, 分子の各係数を fir, iir 関数に直接入力できるように iirmake 関数で調整したものである。

```
% temp = fir(para_b, input_signal);
% output_signal = gain * iir(para_a, para_b);
```

使用例

User's Manual 第4章 3.2 節 フィルタ処理参照

参照

fir, iirmake, butwmake

iirmake

機能

零点，および極から直接型 IIR フィルタの係数を求める．

形式

```
iirmake( zr, zi, pr, pi, a, b )
```

パラメータ

1. zr: 零点の実数部入力オブジェクト (Series, Snapshot)
2. zi: 零点の虚数部入力オブジェクト (Series, Snapshot)
3. pr: 極の実数部入力オブジェクト (Series, Snapshot)
4. pi: 極の虚数部入力オブジェクト (Series, Snapshot)
5. a: 伝達関数の分母の係数の出力オブジェクト (Series)
6. b: 伝達関数の分子の係数の出力オブジェクト (Series)

解説

1. 出力される係数は，iir，fir それぞれの関数にそのまま使用できるフォーマットで出力される．
2. 出力 a，b は予め宣言しておく必要がある．

使用例

User's Manual 第4章 3.2 節 フィルタ処理参照

参照

butwmake，fir，iir

interp

機能

1次元系列の入出力データ対 x, y を補間した系列 xip, yip を求める。

形式

```
yip = interp( "algorithm", x, y, xip )
yip = interp( "algorithm", x, y, dpt [, xip2] )
```

パラメータ

1. algorithm : 補間アルゴリズム (String)
 "akima" : Akima の補間
 "spline" : 自然 3 次スプライン補間
2. x : 入力系列 (1 次元 Series)
3. y : 出力系列 (1 次元 Series)
4. dpt : 補間後のデータ点数 (>0) , サンプル周波数からデータ点数, 補間後の入力系列を求める ($=0$) .
5. xip : 補間後の入力系列 (1 次元 Series)
6. xip : 補間後の出力系列 (1 次元 Series)
7. xip2 : 自動生成された補間後の入力系列 (1 次元 Series)

解説

1. x, y は同じデータ点数でなければならない。
2. x および xip は, とともに $x_i < x_{i+1}$ の関係を満たしていなければならない。
 - 形式 1 について :
 - a. xip の系列に対する yip が求められている。
 - b. xip の値の範囲は x の値の範囲を超えてはならない。
 - 形式 2 について :

- a. $dpt = 0$ の場合は，サンプリング周波数をもとに，以下の定義より， xip が自動生成される．

$$xip_0 = x_0, xip_i = x_0 + step \times i, \dots xip_{dpt-1} = x_{n-1}$$

ここで，

$$step = 1.0/sf, dpt = (x_{n-1} - x_0) \times sf + 1$$

但し， sf はサンプリング周波数， n は x のデータ点数

- b. $dpt > 0$ の場合は，上式の $step$ を次式として xip が生成される．

$$dpt > 1 \text{ のとき } step = (x_{n-1} - x_0)/(dpt - 1)$$

$$dpt = 1 \text{ のとき } step \text{ は } 0$$

- c. 第 4 引数 $xip2$ には，自動生成された補間後の入力系列 xip が格納される．

interp2

機能

2 次元データを読み込み 2 次元補間を行い，その結果を出力する．

形式

```
y = interp2( x, s1, s2 )
```

パラメータ

1. x : 2 次元入力オブジェクト (Series, Snapshot)
2. y : 2 次元出力オブジェクト (Series)
3. s1 : X 軸データの補間数 (≥ 0)
4. s2 : Y 軸データの補間数 (≥ 0)

解説

1. 補間数は，元データの 2 点のデータ間を，何個のデータをもって補間するのかを指定する．
2. 入力オブジェクトは，2 次元データでなければならない．
3. X 軸とは，時間軸 (データ長が可変のデータ方向)

参照

cont

inv

機能

2次元データを行列とみなし，逆行列を求める．

形式

```
y = inv( x )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. y: 出力オブジェクト (Series, Snapshot)

解説

1. 正方行列のみ計算可能
2. 3次元以上のデータは，下位の2次元データを配列とみなし，すべての配列について逆行列を計算し，同じインデックスの Series を返す．

使用例

1. 3次元データの場合

```
% matrix                                # 入力データ
[0]:[0][0]%    1    0
[0]:[1][0]%    0    1

[1]:[0][0]%    0    1
[1]:[1][0]%    1    0

[2]:[0][0]%    1    2
[2]:[0][0]%   -2    1

% inv(matrix)                            # 逆行列の計算
[0]:[0][0]%    1    0
[0]:[1][0]%    0    1

[1]:[0][0]%    0    1
[1]:[1][0]%    1    0
```



```
[2]:[0][0]%    0.2   -0.4
[2]:[1][0]%    0.4    0.2
```

2. User's Manual 第 4 章 3.4 節 行列演算参照

参照

det

levin

機能

時系列に線形 AR モデルをあてはめ，Levinson-Durbin アルゴリズムを用いて AR 係数を求め，パワースペクトルを出力する．

形式

```
levin( x, pw, aic, method, odr, dpt, tola, err, coef )
```

パラメータ

1. x : 入力時系列 (Series, Snapshot)
2. pw : 出力時系列:パワースペクトル (Series)
3. aic : 出力時系列 : AIC (Series)
4. method : 次数決定方式 ("A" or 数値)
 "A" : AIC_{max} で次数決定
 数値 : Fixed Order
5. odr : AR モデルの最大次数
6. dpt : パワースペクトルの出力データ点数
7. tola : AIC に対する許容値 (burg 関数参照)
8. err : 出力時系列 : 予測誤差 (Series)
9. coef : 出力時系列 : AR 係数 (Series)

解説

1. 入力データは dcut 関数により，直流分をカットしておく．
2. pole 関数等により，極の特性を知ることができる．
3. 出力 pw , aic , err , coef は予め宣言しておく必要がある．

使用例

$$x_k + 0.8x_{k-1} = e_k, \quad e_k \sim N(0.0, 1.0)$$

に従う信号のパワースペクトルと AR 係数をもとめる (odr: 固定) .

```
%x = argen(para,1024);           # 信号の生成

% series pw,aic,err,coef;         # Series 変数の宣言
% levin(x,pw,aic,1,1,128,2,err,coef); # 推定
Order =          1      AIC =      -41.225
AR Coefficient( 1)=      0.7817198,      0.7817198

% coef
[0]:%      0.7917
```

参照

dccut , burg , pole

mnrand

機能

多次元正規分布に従う乱数ベクトルを発生する。

形式

```
x = mnrand( dpt, seed, mvec, vmat )
```

パラメータ

1. x : 出力オブジェクト (Series)
2. dpt : 出力ベクトルの数 (Scalar)
3. seed : 乱数の初期値 (奇数)
4. mvec : 入力オブジェクト : 平均値ベクトル (Series, Snapshot)
5. vmat : 入力オブジェクト : 分散共分散行列 (Series, Snapshot)

解説

1. 乱数のシードは、M 系列を用いて生成している。
2. 分散共分散行列に対してコレスキー分解を施すことにより、多次元乱数を生成している。
3. 分散共分散行列は、正定値対象行列でなければならない。

使用例

平均 $\mu_1 = \mu_2 = 0$, 分散 $\sigma_1^2 = \sigma_2^2 = 1$, 相関係数 $\rho_{12} = 0.5$ で規定される 2 次元正規分布に従う確率変数ベクトル (y_1, y_2) を 1000 点生成する。

```
% mvec = (0,0);           # 平均ベクトルの設定
% vmat = (1,0.5,0.5,1);   # 分散共分散
% vmat = reform(vmat,(2,2)); # 行列の設定
% y = mnrand(1000,1,mvec,vmat); # 乱数生成
% y1 = y[0];              # 1000 点の乱数が
% y2 = y[1];              # 格納される
```

参照

`nrnd , urand`

mul

機能

2 つの 2 次元データを行列とみなして積を求める。

形式

```
z = mul( x, y )
```

パラメータ

1. x, y : 入力オブジェクト (Series, Snapshot)
2. z : 出力オブジェクト (Series, Snapshot)

解説

1. 1 次元データは、行ベクトルとみなして計算する。
2. 3 次元以上のデータについての動作は、次のようになる。
 - a. x : 2 次元以下, y : 3 次元 - x に y の行列をそれぞれ乗ずる。
 - b. x : 3 次元, y : 2 次元以下 - x の行列のそれぞれに y を乗ずる。
 - c. x : 3 次元, y : 3 次元 - x と y の行列が 1 対 1 に対応すればそれぞれの積を求める, それ以外はエラー。

使用例

User's Manual 第 4 章 3.4 節 行列演算参照

nmeq

機能

正規方程式を解き，パラメータの最小2乗推定量を求める．

形式

```
para = nmeq( x, y )
```

パラメータ

1. para : 出力オブジェクト : パラメータベクトル (Series)
2. x : 入力オブジェクト : 計画行列 (Series, Snapshot)
3. y : 入力オブジェクト : 従属変数ベクトル (Series, Snapshot)

解説

ハウスホルダー変換を用いて計算している．

User's Manual 第4章 3.4節 行列演算では行列コマンドを組み合わせて同様の機能を実現しているが，精度等については本関数の方が優れている (ハウスホルダー変換を用いているため)．

使用例

あるシステムへの入力データ $x = (-2, -1, 0, 1, 2)$ と，出力データ $y = (7.25, 3.81, 1.88, 2.33, 5.12)$ が与えられている時，これを，以下のようなモデル，

$$y_i = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \varepsilon_i, \quad i = 1, \dots, 5$$

により同定する．

```
% x=(-2,-1,1e-10,1,2);
% y=(7.25,3.81,1.88,2.33,5.12);      # 従属変数ベクトルの設定

% series xmat[3];
% xmat[0]=x^0;                        # 計画行列の設定
% xmat[1]=x^1;
% xmat[2]=x^2;

% beta=nmeq(xmat,y);                  # パラメータ値を求める
```

パラメータの最小2乗推定量は次のようになる．

```
% beta  
[0]:%    1.958    -0.574    1.06
```


norm

機能

指定したオブジェクトの内容を正規化する .

形式

```
y = norm( x, type )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 出力オブジェクト (Series, Snapshot)
3. type : 正規化方法 (1,2,3,4 or 5)
 - 1 : 最大値で正規化
 - 2 : 最小値で正規化
 - 3 : 絶対値最大で正規化
 - 4 : 絶対値最小で正規化
 - 5 : 平均 0 , 分散 1 に正規化

解説

処理内容を以下に示す .

- type = 1 :

$$y_i = x_i / x_{max}$$

- type = 2 :

$$y_i = x_i / x_{min}, z_i = |x_i|$$

- type = 3 :

$$y_i = x_i / z_{max}$$

- type = 4 :

$$y_i = x_i / z_{min}$$

- type = 5 :

$$y_i = (x_i - x_{mean}) / \sqrt{1/n \sum_j (x_j - x_{mean})^2}$$

使用例

データの最大値を 1 , 最小値を 0 に正規化する場合の記述例

```
% y = norm(x-min(x),1);
```

nrnd

機能

正規乱数データを生成する。

形式

```
x = nrnd( dpt, init, mean, variance )
```

パラメータ

1. x : 出力時系列 (Series)
2. dpt : 出力データ点数 (Scalar)
3. init : 乱数の初期値 (奇数)
4. mean : 平均値 (Scalar)
5. variance : 分散 (Scalar)

解説

1. 平均 0.5 分散 1.0 の正規分布に従う乱数を 1024 点生成する。

```
% x = nrnd(1024,1,0.5,1.0);
```

2. User's Manual 第 4 章 3.1 節 フーリエ変換参照

参照

mnrand , urand

phase

機能

2つのオブジェクトの内容により位相を求める。

形式

```
ph = phase( x, y, d, u )
```

パラメータ

1. x: 入力オブジェクト: 実部 (Series, Snapshot)
2. y: 入力オブジェクト: 虚部 (Series, Snapshot)
3. ph: 出力オブジェクト (Series)
4. d: 出力データ形式 ("D" or "O")

"D": degree

"O": radian

5. u: 位相戻し操作フラグ ("U" or "O")

"U": 位相戻しする

"O": 位相戻ししない

解説

1. 連続するデータが π 以上変化したとき位相戻し操作が行われる。
2. 実部および虚部がともに0となるデータを含む場合には、位相戻しが正常に動作しない場合がある。

使用例

User's Manual 第4章 3.1節 フーリエ変換参照

参照

fft, power, spcf

pole

機能

burg , levin 関数で求められた AR 係数から極の位置を求める .

形式

```
pole( ar, xr, xi )
```

パラメータ

1. ar : AR 係数を格納してある時系列 (Series, Snapshot)
2. xr : 極の実数部の値を格納する時系列 (Series, Snapshot)
3. xi : 極の虚数部の値を格納する時系列 (Series, Snapshot)

使用例

線形 AR モデル $x_k + 0.8x_{k-1} = e_k$ の極の位置を求める .

```
% para                                # AR 係数
[0]:%    0.8
% series xr,xi;                        # 出力を格納する変数の宣言

% pole(para,xr,xi);
AR[    0]=                0.8
I= 0   x =                -0.8    Y =                0
I= 1   x =    3.3952e-313    Y =    3.3952e-313
I= 2   x =                0      Y =                0.8
I= 3   x =    3.3952e-313    Y =    3.3952e-313

% xr                                  # 実部
[0]:%    -0.8
% xi                                  # 虚部
[0]:%                0
```

参照

burg , levin

power

機能

2 つのオブジェクトの内容を 2 乗して加算する .

形式

```
z = power( x, y )
```

パラメータ

1. x, y : 入力オブジェクト (Series, Snapshot)
2. z : 出力オブジェクト (Series)

解説

処理 :

$$z_i = x_i^2 + y_i^2$$

使用例

User's Manual 第 4 章 3.1 節 フーリエ変換参照

参照

fft , phase , spcf

rank

機能

データの内容から，度数，分散，平均値を求め，その正規分布を算出する．

形式

```
rank( x, u, v, w, xmin, xmax, n )
```

パラメータ

1. x : 入力時系列 (Series, Snapshot)
2. u : 出力時系列 : 度数 - X 軸 (Series, Snapshot)
3. v : 出力時系列 : 度数 - X 軸 (Series, Snapshot)
4. w : 出力時系列 : 正規分布 - Y 軸 (Series, Snapshot)
5. xmin : 階級の最小値 (Scalar)
6. xmax : 階級の最大値 (Scalar)
7. n : 階級の個数 (Scalar)

使用例

```
% x = nrand(128,1,0,1);          # 入力データ
% series u,v,w;                  # 出力を格納する変数宣言

% rank(x,u,v,w,min(x),max(x),20); # 関数実行
** MEAN VALUE =                -0.18724
   VARIANCE   =                 0.82391
   STADARD D  =                 0.90769
   3-ORDER M  =                -0.16687
   SKEWNESS   =                -0.22313
   4-ORDER M  =                 1.9476
   KURTOSIS   =                -0.13089
-0.18724449
```

spcf

機能

入力データのパワースペクトル，位相を計算する．

形式

```
spcf( x, amp, phs )
```

パラメータ

1. x : 入力時系列 (Series, Snapshot)
2. amp : 出力時系列 : パワースペクトル (Series, Snapshot)
3. phs : 出力時系列 : 位相 (Series, Snapshot)

解説

1. N 点のデータを入力したときの出力データ

- データ点数 :

$$N/2 + 1$$

- パワースペクトル :

$$amp_i = \frac{2}{N \cdot sf} \{X_{\text{real}}^2(i) + X_{\text{imag}}^2(i)\}$$

Nyquist 周波数までの片側 PSD を出力するので 2 倍してパワーが同じになるようにしてある．

sf は，サンプリング周波数．

- 位相 : 位相戻し処理を行っている．

単位は，degree である．

2. 出力 amp, phs は予め宣言しておく必要がある．

離散データからの FFT によるパワースペクトル推定について簡単に触れておく．入力 $x_t, t = 0, 1, \dots, N-1$ のパワースペクトルの計算には一般的に次式が用いられている [1]．

$$P(f) = \frac{1}{T} \|X(f)\|^2 = \frac{sf}{N} \|X(f)\|^2$$

$$X(f) = \frac{1}{sf} \sum_{t=0}^{N-1} x_t e^{-j \cdot 2\pi f t / sf}$$

ただし本関数は FFTC コマンドと同一のモジュールを使用しているため以下のようになる。

$$P(f) = \frac{sf}{N} \|X(f)\|^2 = \frac{1}{N \cdot sf} \{X_{\text{real}}^2(f) + X_{\text{imag}}^2(f)\}$$

$$X(f) \cdot sf = X_{\text{real}}(f) - j X_{\text{imag}}(f)$$

上式以外の定義を用いる文献 [2] があるので注意が必要である。

使用例

User's Manual 第4章 3.1 節 フーリエ変換参照

参照

fftc , phase , power

[1] 中溝高好 : "信号解析とシステム同定", コロナ社, 1988

[2] 越川常治 : "信号解析入門", 近代科学社, 1992

sum

機能

Series の内容を積算する .

形式

```
n = sum( x [,y] )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 出力オブジェクト (Series)
3. n : 積算値 (Scalar)

解説

1. 出力 y には , 積分の途中経過が格納される .
(数学の積分とは異なる . $\sin(t) = -a \times \cos(t)$)
2. 返り値 n は , 全要素についての合計
3. 処理

$$n = \sum_{j=1}^m x(j)$$

使用例

```
% x=(-2,-1,0,1,2);          # 入力データ
% series y;                  # 途中結果を格納する変数

% sum(x,y);
0                             # 計算結果

% y
[0]:%  -2    -3    -3    -2    0    # 途中結果
```

trans

機能

2次元データを行列とみなし，転置行列を求める．

形式

```
y = trans( x )
```

パラメータ

1. x: 入力オブジェクト (Series, Snapshot)
2. y: 出力オブジェクト (Series, Snapshot)

解説

1. 1次元データは，行ベクトルとみなして処理する．
2. 3次元以上のデータについては，複数の行列とみなし，下位の2次元についてを行列として処理する．

使用例

1. 列ベクトルの転置を求める．

```
% x=(1,2,3);      # 入力データ（行ベクトル）

% trans(x);        # 転置（列ベクトル）
[0]:[0]%    1
[1]:[0]%    2
[2]:[0]%    3
```

2. User's Manual 第4章 3.4節 行列演算参照

urand

機能

一様乱数データを生成する。

形式

```
x = urand( dpt, init, lower, upper )
```

パラメータ

1. x : 出力時系列 (Series, Snapshot)
2. dpt : 出力データポイント (Scalar)
3. init : 初期値 (奇数)
4. lower : データの最小値 (Scalar)
5. upper : データの最大値 (Scalar)

解説

乱数のシードは、M 系列を用いて生成している。

使用例

区間 [0,1] の一様乱数を 1024 点生成する。

```
% x = urand(1024,1,0,1);
```

参照

mnrand , nrand

window

機能

データに対して、ウィンドウ処理 (ハニング、ハミング) を施す。

形式

```
y = window( x, type, flag )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. y : 出力オブジェクト (Series, Snapshot)
3. type : ウィンドウの種類 (1,2,3 or 4)
 - 1 : ハミングウィンドウ
 - 2 : ハニングウィンドウ
 - 3 : ブラックマン
 - 4 : トライアングル
4. flag : データ補正フラグ (0 or 1)
 - 0 : 補正しない
 - 1 : 補正する

解説

1. データ補正を行うとウィンドウを処理前後の積分値が等しくなる。
2. 入力が 2 次元データの場合には、2 次元ウィンドウ処理を行う。

使用例

User's Manual 第 4 章 3.1 節 フーリエ変換参照

第 10 章 GPM

グラフィックシステム

目次

axis	191
chwin	193
color	194
cont	196
draw	198
egraph	200
factor	202
font	203
frame	204
ginit	205
graph	206
gsolm	208
gstat	211
label	212
line	214
ltype	216
lwidth	217
map	218
newpage	222
origin	223
prev	225
scale	226
size	228
title	229
wclose	230
we	231
wopen	232

axis

機能

cont, graph, map 関数で表示したグラフの X 軸, Y 軸を Linear もしくは Log 表示する.

形式

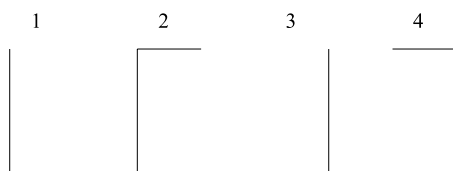
```
axis( tbuf, typ, flg1, flg2, size, grid, form, xscl, yscl, index )
```

パラメータ

1. tbuf: タイトルバッファ番号 (1, 2 or 3)

グラフィック関数群では, 軸のタイトルを登録するために, x, y, z 軸 1 組で, 3 本のタイトルバッファを持っている. その何番目のタイトルを表示するかの指定を行う. なお, 本関数では z 軸の値は使用しない.

2. typ: 軸の表示形式 (1, 2, 3 or 4)



3. flg1: 表示する軸の指定 ("X", "Y" or "XY")

"X": X 軸のみ表示

"Y": Y 軸のみ表示

"XY": XY 両軸を表示

4. flg2: 表示するスケールの指定 ("X", "Y" or "XY")

"X": X 軸のスケールのみ表示

"Y": Y 軸のスケールのみ表示

"XY": XY 両軸のスケールを表示

5. size: 表示するスケール, タイトルの文字サイズ [mm] (Scalar)

6. grid: グリッド表示の指定 (0,1,2 or 3)

0: グリッドなし

1: X 軸のみ表示

2: Y 軸のみ表示
 3: XY 両軸を表示

7. from: スケール表示の型指定 (0,1,2 or 3)

0: X 軸 . . . 実数, Y 軸 . . . 実数
 1: X 軸 . . . 整数, Y 軸 . . . 実数
 2: X 軸 . . . 実数, Y 軸 . . . 整数
 3: X 軸 . . . 整数, Y 軸 . . . 整数

8. xscl: X 軸スケール表示間隔 (Scalar)

9. yscl: Y 軸スケール表示間隔 (Scalar)

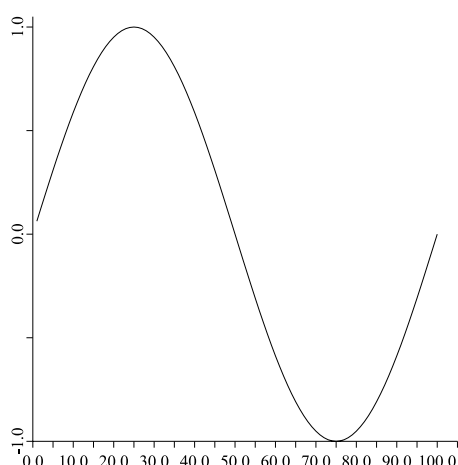
10. index: linear スケール値の指数表示指定 (0 or 1)

0: 必要ならば指数表示をする (デフォルト)
 1: スケール値に関係なく指数表示を行わない

使用例

```
% t = (0~100)/100*2*PI
% data = sin(t)
% wopen(1,"A4",0,0)
% graph(data,"T",0,0,0,0,0)
% axis(1,1,"XY","XY",4,0,0,10,1,0)
```

と入力すると、以下ようになる。



参照

cont, graph, map

chwin

機能

アクティブウィンドウを切り替える。

形式

```
chwin( wnum )
```

パラメータ

wnum : 切り替えたいウィンドウ番号 (Scalar)

color

機能

グラフ，軸，フレームの色を指定する．

形式

```
color( pen1, pen2 )
```

パラメータ

1. pen1 : グラフの色 ($1 \leq \text{pen1} \leq 10$, 整数 or カラー名)
2. pen2 : 軸，フレームの色 ($1 \leq \text{pen2} \leq 10$, 整数 or カラー名)

解説





















色指定 (ディスプレイ)

色指定は，以下の整数値以外にカラー名を指定してもよい (小文字でも可)．カラー名 : BLACK, BLUE, RED, MAGENTA, GREEN, SKYBLUE (CYAN), YELLOW, WHITE, GLAY1, GLAY2, GLAY3

0 : 黒	1 : 青
2 : 赤	3 : 紫
4 : 緑	5 : 水色
6 : 黄色	7 : 白
8 : グレー 1	9 : グレー 2
10 : グレー 3	

使用例

プリンタに出力したときは次のようになる．左側がグレイスケールモードで出力した場合，右側がカラーモードで出力した場合である．

	0:BLACK	
	1:BLUE	
	2:RED	
	3:MAGENTA	
	4:GREEN	
	5:CYAN	
	6:YELLOW	
	7:WHITE	
	8:GLAY1	
	9:GLAY2	
	10:GLAY3	
Gray output mode		Color output mode

Version 2.9x の color() 関数とは、色番号と出力色の対応関係が異なりウィンドウ表示色がそのままされるため注意が必要となる。

cont

機能

入力された 2 次元データの等高線を表示する。

形式

```
cont( x, step, dirc, view, mode )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. step : 等高線の表示間隔 (Scalar)
3. dirc : 表示方向 ("X" or "Y")
"X" : オブジェクトの上位次元が X 軸方向になる
"Y" : オブジェクトの下位次元が Y 軸方向になる
4. view : 表示データの原点の指定 (1 or -1)
1 : 左下が原点
-1 : 左上が原点
5. mode : 表示モードの指定 (0 or 1)
0 : モノクロ
1 : カラー

解説

1. 表示する等高線の最大, 最小値の設定は, scale 関数で設定する。
2. 表示を行う際, 補間を行わない。等高線をなめらかに表示するためには interp2 関数を用いる必要がある。

参照

interp2

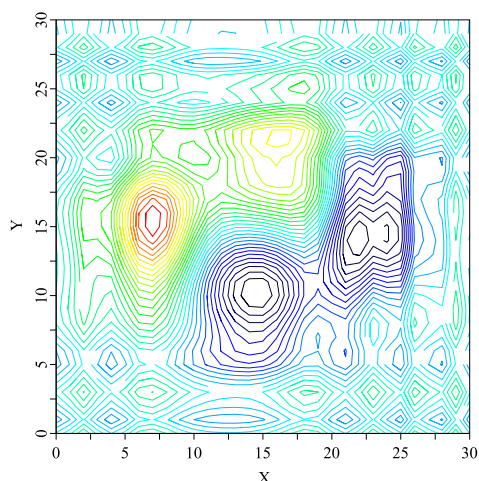
使用例

```

% t = (0~30) / 5 - 3;
% x = y = t;
% snapshot z[31][31];
% for(i = 0; i <= 30; i++){
%   for(j = 0; j <= 30; j++){
%     ptx2 = (1+x:[j])*(1+x:[j]);
%     pty2 = (1+y:[i])*(1+y:[i]);
%     tx2 = x:[j]*x:[j];
%     ty2 = y:[i]*y:[i];
%     tx = tx2*x:[j] - ty2*ty2*y:[i];
%     z[i][j]=10*(x:[j]+tx)*exp(-tx2-ty2)-sin(-ptx2-tx2)-cos(-pty2-ty2);
%   }
% }
% wopen(1,"A4",0,1);
% cont(z, 0.5,"X",1,1);
% title(1,"X", "Y");
% axis(1,1,"XY","XY",4,0,3,0,0,0);
% frame();

```

と入力すると以下ようになる .



draw

機能

cont , graph , map 関数で表示したグラフに対し , 指定したレベルのラインを表示する .

形式

```
draw( axs, val )
```

パラメータ

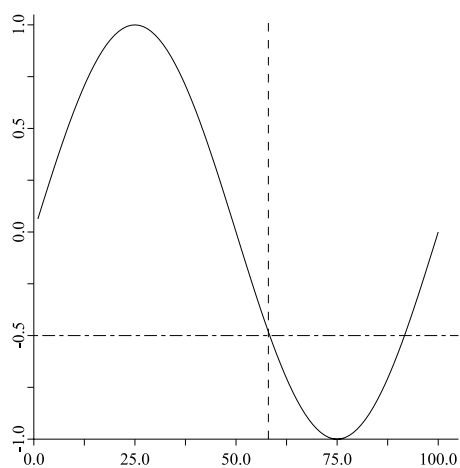
1. axs : 軸の指定 ("X" or "Y")
2. val : ライン表示する値 (Scalar)

解説

1. 第 1 引数が "X" ならば , $x = val$ となるような , "Y" ならば , $y = val$ となるような直線を作成する .
2. 色の指定は color 関数の第 2 パラメータで指定する .
3. グラフを表示しなくても , scale で設定したレベルに基づいてラインを引くこともできる .

使用例

```
% t = (0~100)/100*2*PI
% data = sin(t)
% wopen(1,"A4",0,1)
% graph(data,"T",0,0,0,0,0)
% ltype( 2, 2 )           # 長鎖線
% draw("X", 58 )         # X 軸に補助線を引く
% ltype( 3, 3 )           # 一点鎖線
% draw("Y", -0.5 )       # Y 軸に補助線を引く
```



参照

cont , graph , map

egraph

機能

エラーバー付きのグラフを表示する。

形式

```
egraph ( y, x, axs, lne, step, sym1, sym2 )
```

パラメータ

1. y : Y 軸方向のデータ (object, "T" or "F")

object : 入力オブジェクト (Series, Snapshot)

"T" : 時間軸

"F" : 周波数軸

"D" : データ点数

2. x : X 軸方向のデータ (object, "T" or "F")

object 入力オブジェクト (Series, Snapshot)

"T" : 時間軸

"F" : 周波数軸

"D" : データ点数

3. axs : XY 軸の形式 (0, 1, 2 or 3)

	X 軸	Y 軸
0	LINEAR	LINEAR
1	LOG	LINEAR
2	LINEAR	LOG
3	LOG	LOG

4. lne : グラフ描画の形式 ($0 \leq lne \leq 6$)

0 : エラーバー付きのグラフを引く

1 : センターシンボルおよびエラーバーを表示

2 : エラーバーおよびグラフ, センターシンボルを表示

3 : 棒グラフを描く

4 : lne = 0 および lne = 3 で表示する

5 : 零レベルとグラフで囲まれた領域を塗りつぶす

6 : エラーバーのみ描く

5. `step` : 表示間隔 ($\text{step} \geq 0$, 整数)
($\text{step}-1$) 点ごとのデータを対象に処理を行う
6. `sym1` : センターシンボルの指定 ($0 \leq \text{sym1} \leq 16$, 整数)
シンボルの種類については, `graph` 関数参照
7. `sym2` : センターシンボルの大きさ [mm] ($\text{sym2} \geq 0$, 実数)

解説

1. データ形式は, `index` が (n, 2) または, (n, 3) である必要がある. 例えば, `y` が表示するデータである場合, 前者の場合には, `y[0]` がグラフ, `y[0] ± y[1]` がエラーバーとして表示される. 後者の場合, `y[0]` をグラフ, `y[1]`, `y[2]` をエラーバーの端点として表示する.

2. 時間軸表示

$$\text{buff}(i) = \frac{1000.0 \times \text{float}(i - 1)}{\text{sam}}$$

3. 周波数軸表示

$$\text{delta} = \frac{\text{sam}}{\text{float}(n - 1) \times 2.0}$$

$$\text{buff}(i) = \text{delta} \times \text{float}(i - 1)$$

4. センターシンボルの大きさ

大きさは [mm] で入力する. 但し, 0 を入力したときは 0.5[mm] に自動設定される.

factor

機能

グラフィック表示倍率を指定する .

形式

```
factor( x )
```

パラメータ

x : 表示倍率 ($x \geq 0$)

解説

デフォルト値は 1.0 .

font

機能

グラフィック端末の文字種類を指定する。

形式

```
font( type )
```

パラメータ

type : 文字種類番号 ($1 \leq \text{type} \leq 12$, 整数)

文字種類と番号の対応は以下のようになっている。

	roman	italic	bold	bold-italic
times	FONT 1	<i>FONT 2</i>	FONT 3	<i>FONT 4</i>
helvetica	FONT 5	<i>FONT 6</i>	FONT 7	<i>FONT 8</i>
courier	FONT 9	<i>FONT 10</i>	FONT 11	<i>FONT 12</i>

frame

機能

グラフの枠を描く。

形式

```
frame( )
```

パラメータ

なし

解説

origin , size など指定されたパラメータに従って , グラフの枠を描く。

ginit

機能

グラフィックパラメータ初期化する。

形式

```
ginit( )
```

パラメータ

なし

解説

初期設定

カラー (グラフ) : 白

カラー (軸, フレーム) : 白

スケール設定 : 自動設定

スケールファクター : 1.0

原点 (X, Y) : (20.0, 20.0)

サイズ (X, Y) : (100.0, 100.0)

graph

機能

データをグラフ表示する。

形式

```
graph( y, x, axs, lne, step, sym1, sym2 )
```

パラメータ

1. y : Y 軸方向のデータ (object, "T" or "F")

object 入力オブジェクト (Series, Snapshot)

"T" : 時間軸

"F" : 周波数軸

"D" : データ点数

2. x : X 軸方向のデータ (object, "T" or "F")

object 入力オブジェクト (Series, Snapshot)

"T" : 時間軸

"F" : 周波数軸

"D" : データ点数

3. axs : XY 軸の形式 (0, 1, 2 or 3)

	X 軸	Y 軸
0	LINEAR	LINEAR
1	LOG	LINEAR
2	LINEAR	LOG
3	LOG	LOG

4. lne : グラフ描画の形式 ($0 \leq lne \leq 5$)

0 : 線のみを引く

1 : センターシンボルのみを表示

2 : 線とセンターシンボルを表示

3 : 棒グラフを描く

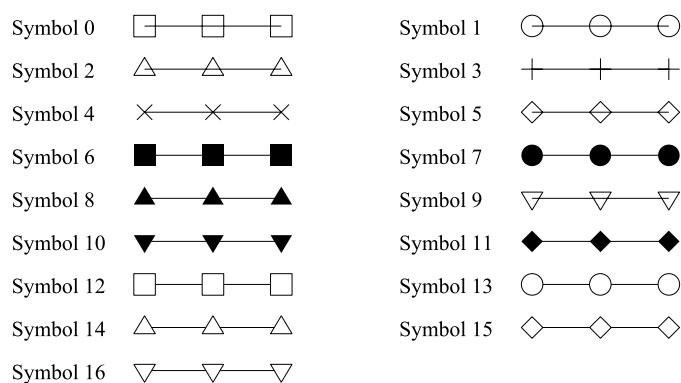
4 : lne = 0 および lne = 3 で表示する

5 : 零レベルとグラフで囲まれた領域を塗りつぶす

5. step : 表示間隔 ($step \geq 0$, 整数)

(STEP-1) 点ごとのデータを対象に処理を行う

6. sym1 : センターシンボルの指定 ($0 \leq \text{sym1} \leq 16$, 整数)



12 から 16 までは白抜きである .

7. sym2 : センターシンボルの大きさ [mm] ($\text{sym2} \geq 0$, 実数)

解説

1. 2次元データの場合には , 上位次元方向を時系列の方向と考え , 下位次元の要素分の本数のグラフを作画する .
2. 時間軸表示

$$\text{buff}(i) = \frac{1000.0 \times \text{float}(i - 1)}{\text{sam}}$$

3. 周波数軸表示

$$\text{delta} = \frac{\text{sam}}{\text{float}(n - 1) \times 2.0}$$

$$\text{buff}(i) = \text{delta} \times \text{float}(i - 1)$$

4. センターシンボルの大きさ

大きさは [mm] で入力する . 但し , 0 を入力したときは 0.5[mm] に自動設定される .

gsolm

機能

直交座標系で定義される二変数関数が矩形領域格子点上で与えられたとき，立体図を表示する．

形式

```
gsolm( x, rh1, rh2, f1, f2, f3, f4, col, div1, div2,
       dirc, view, mode [,size] )
```

パラメータ

1. x : 入力オブジェクト (Series, Snapshot)
2. rh1, rh2 : 表示形式 ($-1.0 < rh1, rh2 < 1.0$)
3. f1 : 作画条件 1 (0 or 1)
 - 0 : 隠線消去する
 - 1 : 隠線消去しない
4. f2 : 作画条件 2 (0, 1 or 2)
 - 0 : X 方向，Y 方向とも格子線を引く
 - 1 : X 方向のみ引く
 - 2 : Y 方向のみ引く
5. f3 : 作画条件 3 (0, 1 or 2)
 - 0 : 視点に近いへりを上側と下側ともに表示する
 - 1 : 上側のみ表示する
 - 2 : 下側のみ表示する
6. f4 : 作画条件 4 ($-1 \leq f4 \leq 7$)
 - 1 : 図に座標軸と枠をつけない
 - 0~7 : 指定された色で，3 次元座標軸を表示する
7. col : 零平面表示形式 ($-1 \leq col \leq 7$)
 - 1 : 零平面を表示しない
 - 0~7 : 指定された色で零平面を表示する
8. div1 : X 軸方向表示ステップ (≥ 1 , 整数型)

9. div2 : Y 軸方向表示ステップ (≥ 1 , 整数型)
10. dirc : 表示方向 ("X" or "Y")
 - "X" : オブジェクトの上位次元が X 軸方向になる
 - "Y" : オブジェクトの下位次元が Y 軸方向になる
11. view : 表示データの原点の指定 (1 or -1)
 - 1 : 左下が原点
 - 1 : 左上が原点
12. mode : 表示モードの指定 (0 or 1)
 - 0 : モノクロ
 - 1 : カラー
13. size : 座標軸につけるタイトルなどの文字の大きさ [mm] (Scalar)

解説

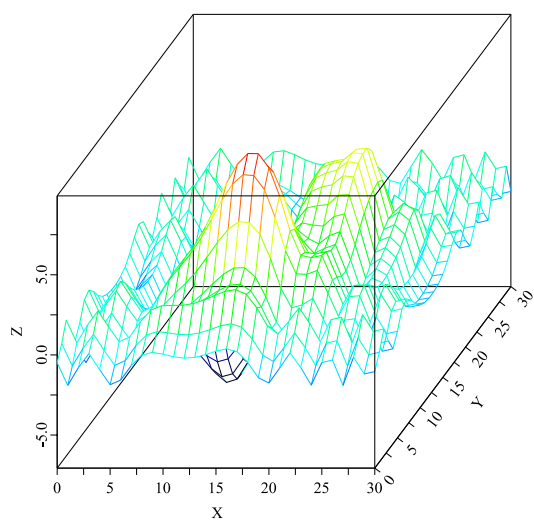
1. 表示するデータの最大, 最小値は, scale 関数の z 軸の値の設定により行う.
2. $Z_{i,j} = f(X_i, Y_j)$ $0 \leq i \leq mx, 0 \leq j \leq my$ が与えられたとき,
 - $Z_{i,j} \leq Z_{min}$ なら $Z_{i,j} = Z_{min}$ とし, $Z_{max} \geq Z_{i,j}$ なら $Z_{i,j} = Z_{max}$ とみなして表示する.
3. パラメータ f4, col に関しては Version 2.9x と異なり, 0 のとき黒で描画するため注意が必要.

使用例

```
% t=(0~30)/5-3;
% snapshot z[31][31];
% x=y=t;
% for(i=0;i<=30;i++){
%   for(j=0;j<=30;j++){
%     ptx2=(1+x:[j])*(1+x:[j]);
%     pty2=(1+y:[i])*(1+y:[i]);
%     tx2=x:[j] * x:[j];
%     ty2=y:[i] * y:[i];
%     tx = tx2*x:[j] - ty2*ty2*y:[i];
%     z[i][j]=10*(x:[j]+tx)*exp(-tx2-ty2)-sin(-ptx2-tx2)-cos(-pty2-ty2);
%   }
% }

% wopen(1,"A4",0,1);
% title(1,"X", "Y", "Z");
% gsolm(z,0.3,0.4,0,0,0,0,-1,1,1,"X",1,1);
```

と入力すると以下のようなになる .



gstat

機能

グラフィックパラメータの現在値を表示する。

形式

```
gstat( )
```

パラメータ

なし

label

機能

グラフィック画面に文字列を書く。

形式

```
label( "ctyp", xorg, yorg, high, th [, "str" [, mode]] )
```

パラメータ

1. ctyp : 設定形式 ("I" or "F")
 "I" : インデックス入力によって座標を設定
 "F" : インデックス入力によって座標を設定し, 文字列をファイルから読み込む
2. xorg, yorg : 書き始める座標 [mm] (相対座標) (Scalar)
3. high : 書く文字のサイズ [mm] (Scalar)
4. th : 書く文字の角度 (0, 90, 180, 270) [deg] (Scalar)
5. str : ラベル文字列 もしくは ファイル名 (String)
 ctyp = "I" : ラベル文字列 ("" で囲む, 30 文字以内) (省略可能)
 ctyp = "F" : ラベル文字列の入ったファイル名
6. mode : 表示モード (0, 1 or 2)
 0 : 左寄せ
 1 : 右寄せ
 2 : センタリング

指定した座標を, 文字列の左端, 右端, 中央として表示する。

省略時は 0 の左寄せとして表示する。

解説

処理

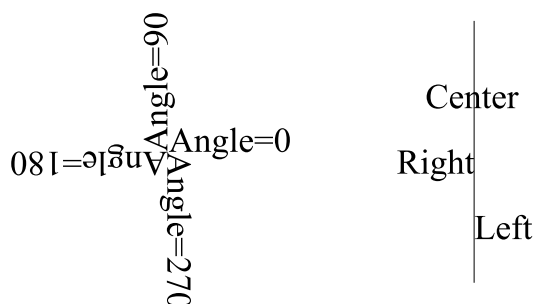
ctyp = "I" の時, 文字列を省略すると

>
が表示されて入力待ちになるので, 表示したい文字列を入力する。なお, この入力方法は 1 度の入力で 80 文字まで表示することができる。

使用例

```
% wopen(1,"A4",0,1);
% origin(60,60);
% string str, str1;
% theta=90;
% str="Angle=";
% for(i=0; i<4; i++) {
%     str1 = str + String(theta*i);
%     label("I",0,0,8,theta*i,str1);
% }
% origin(130,30);
% str1="Center";
% label("I",0,40,8,0,str1,2);
% str1="Right";
% label("I",0,25,8,0,str1,1);
% str1="Left";
% label("I",0,10,8,0,str1,0);
% line(0,0,0,60);
```

と入力すると以下ようになる .



line

機能

グラフィック画面にラインを引く .

形式

```
line( xorg, yorg, xend, yend, "mode" )
```

パラメータ

1. xorg, yorg : 始点座標 [mm] (Scalar)
2. xend, yend : 終点座標 [mm] (Scalar)
3. mode : 描画モード ("L", "B" or "BF")

"L" : 直線

"B" : 矩形

"BF" : 矩形内を塗りつぶす

解説

xorg, yorg, xend, yend で指定する座標は origin 関数で設定した座標からの相対座標である .

使用例

```
% wopen(1,"A4",0,1);
% origin(25,25);
% yoffset = 10;
% line(0,40,100,40+yoffset, "L");
% line(0,20,100,20+yoffset, "B");
% line(0, 0 100, 0+yoffset, "BF");
```

と入力すると以下ようになる .

ltype

機能

図形を出力する場合の線種を指定する。

形式

```
ltype( ilt1, ilt2 )
```

パラメータ

1. ilt1 : グラフの線種 ($1 \leq \text{ilt1} \leq 8$)
2. ilt2 : 座標軸, フレームの線種 ($1 \leq \text{ilt2} \leq 8$)

—————	1: 実線
-----	2: 長鎖線
- - - - -	3: 一点鎖線
.....	4: 二点鎖線
.....	5: 破線
.....	6: 点線 1
.....	7: 点線 2
.....	8: 点線 3

lwidth

機能

グラフや座標軸の線種を指定する。


形式

```
lwidth( ilt1, ilt2 )
```

パラメータ

1. ilt1 : グラフ線幅 ($1 \leq \text{ilt1} \leq 4$)
2. ilt2 : 座標軸 , フレームの線幅 ($1 \leq \text{ilt2} \leq 4$)

1 : 

2 : 

3 : 

4 : 

map

機能

1 次元データのカラーマップ表示を行う。

形式

```
map( x, "dirc", style, psize, view[, time, start, end, step] )
```

パラメータ

1. x : 入力オブジェクト , 2 次元または 3 次元 (Series, Snapshot)
2. dirc : 表示方向 ("X" or "Y")
 - "X" : データの上位次元方向を X 軸とする
 - "Y" : データの下位次元方向を Y 軸とする
3. style : 表示形式 ($0 \leq \text{style} \leq 14$, 整数)
 - 0 : パターンの色の変化
 - 1 : パターンのサイズが変化 (パターンの色は color 関数で設定する . データ > 0 : 第 1 パラメータ , データ < 0 : 第 2 パラメータ)
 - 2 : パターンの色とサイズが変化
 - 3 : データが min と max の範囲に正規化され , パターンのサイズが変化 (fill)
 - 4 : style = 3 において , パターンを open で表示
 - 5 : style = 3 において , パターンの色の変化
 - 6 : パターンサイズが変化し , データ > 0 : open , データ < 0 : fill
 - 7 : パターンサイズが変化し , データ > 0 : fill , データ < 0 : open
 - 8 ~ 14 : パターンの色が , 8 : 青 , 9 : 赤 , 10 : マゼンタ , 11 : 緑 , 12 : シアン , 13 : 黄色 , 14 : 白黒の強弱で変化する .
4. psize : パターンサイズ (0 or 1)
 - 0 : MAP 全体を SIZE で指定された大きさで表示 (デフォルト)
 - 1 : 各パターンを正方形で表示
5. view : 表示データの原点の指定 (1 or -1)
 - 1 : 左下が原点
 - 1 : 左上が原点
6. time : 描画する間隔 msec 単位 (Scalar)

7. start : 表示開始点 (Scalar)
8. end : 表示終了点 (0 のときはデータの最後まで表示する . Scalar)
9. step : 表示データのステップ (≥ 1 , 0 のときは 1 と同じ)

解説

1. 表示データの範囲

scale 関数の z 軸の設定に従って , 表示データの最大・最小値を設定する .

2. PostScript プリンタへの出力

style = 0, 2, 5, 8~14 では , 表示データがグレイスケールの濃淡レベルに変換される (データの大 小に対して , 黒 白に変化) .
その他の style では , color 関数と同様 .

3. time , start , end , step は , データが 3 次元の場合のみ有効である .

使用例

```
% series dat[4];
% dat:[0]=( -2, -1,  0,  1);
% dat:[1]=( -1,  0,  1,  2);
% dat:[2]=(  0,  1,  2,  3);
% dat:[3]=(  1,  2,  3,  4);

% wopen(1,"A4",0,1);
% size(70,40);
% color(4,2);
% lwidth(2,1);

% x=20;
% y=180;
% ystep=50;
% lstr="map(dat,\"Y\",");
% rstr=",0,-1)";
% string str;

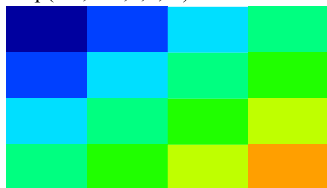
% scale("N","A","N","A","N","F",-3,5);

% for(i=0;i<8;i++){
%   origin(x,y);
%   color(4,2);
%   map(dat,"Y",i,0,-1);
%   color(4,"black");
%   str=lstr+String(i)+rstr;
%   label("I",0,42,5,0,str);
```

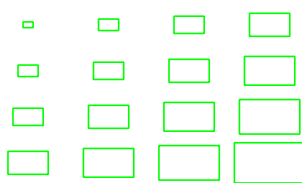
```
% if(i==3){
%     x=110;
%     y=180;
% }else{
%     y=y-ystep;
% }
% }
```

と入力したときカラーモードで出力した場合は以下のようなになる。

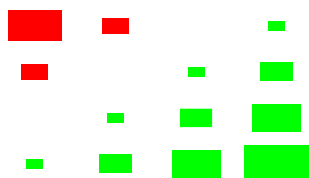
map(dat,"Y",0,0,-1)



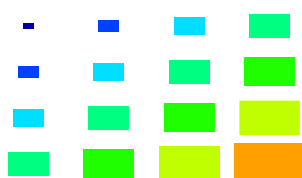
map(dat,"Y",4,0,-1)



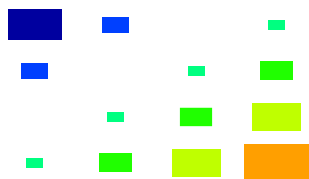
map(dat,"Y",1,0,-1)



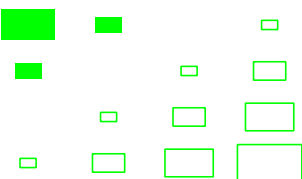
map(dat,"Y",5,0,-1)



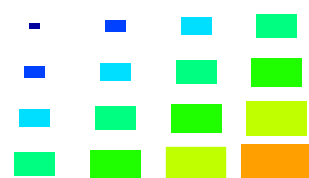
map(dat,"Y",2,0,-1)



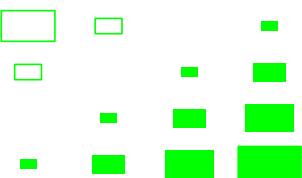
map(dat,"Y",6,0,-1)



map(dat,"Y",3,0,-1)

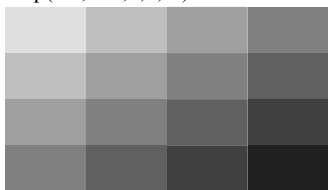


map(dat,"Y",7,0,-1)

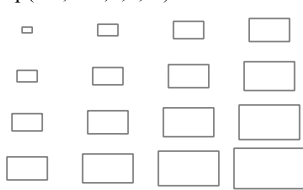


また、グレースケールモードで出力した場合は以下のようなになる。

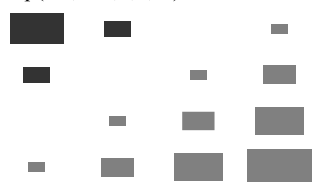
map(dat,"Y",0,0,-1)



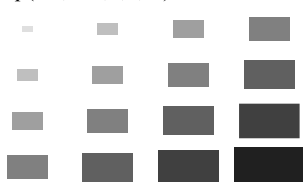
map(dat,"Y",4,0,-1)



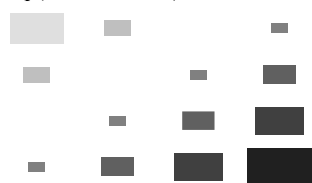
map(dat,"Y",1,0,-1)



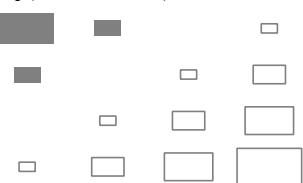
map(dat,"Y",5,0,-1)



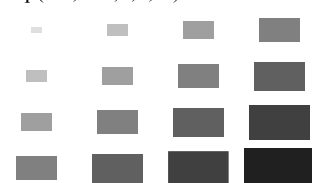
map(dat,"Y",2,0,-1)



map(dat,"Y",6,0,-1)



map(dat,"Y",3,0,-1)



map(dat,"Y",7,0,-1)



newpage

機能

プリンタの改ページを行う。

形式

```
newpage( )
```

パラメータ

なし

解説

ポストスクリプトファイルに複数のページを出力できる

origin

機能

グラフィック相対座標の原点を指定する。

形式

```
origin( xorg, yorg )
```

パラメータ

1. xorg : X 軸の原点 [mm] ($xorg \geq 0.0$, 実数)
2. yorg : Y 軸の原点 [mm] ($yorg \geq 0.0$, 実数)

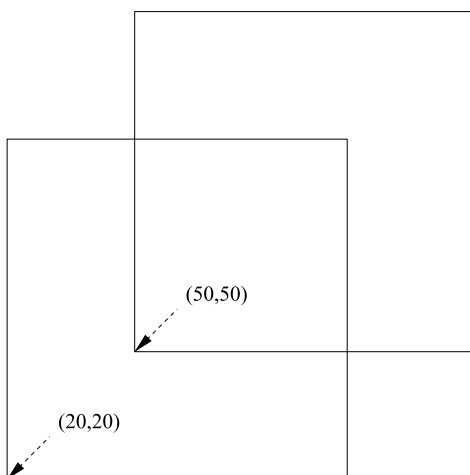
解説

原点の指定は、絶対座標 [mm]で行う。

使用例

```
% origin(20,20);  
% frame( );  
% origin(50,50);  
% frame( );
```

で、



と出力される (ただし, この図の矢印及び座標の数字は後から編集して加えたものである) .
この関数のパラメータを決定する際には, ウィンドウ左上に表示されるマウスカーソルの座標値を目安にすることができる . また, このウィンドウ左上の座標値は, ウィンドウ内の任意の座標でマウス左ボタンをクリックすることによりクリップボードへコピーすることができる .

prev

機能

中間ファイル (GPMDVIFILE) の内容を , SATELLITE ウィンドウに表示する .

形式

```
prev( [ "filename" ] )
```

パラメータ

filename : GPMDVIFILE の名前 (String)

解説

filename を省略した場合は , "GPMDVIFILE1" のデータを表示する .

scale

機能

グラフィック表示のスケールの設定。

形式

```
scale( "xtyp", "xaxs", "ytyp", "yaxs"
      [, scl1, scl2 [, scl3, scl4]] )

scale( "xtyp", "xaxs", "ytyp", "yaxs", "ztyp", "zaxs"
      [, scl1, scl2 [, scl3, scl4 [, scl5, scl6]]] )
```

パラメータ

1. xtyp : X 軸のスケール形式 ("N" or "L")
 - "N" : LINEAR 軸
 - "L" : LOG 軸
2. yaxs : X 軸のスケール設定方式 ("A", "F" or "D")
 - "A" : 自動設定
 - "F" : マニュアル設定
 - "D" : デフォルト値設定 (前回処理したときの値)
3. ytyp : Y 軸のスケール形式 ("N" or "L")
 - "N" : LINEAR 軸
 - "L" : LOG 軸
4. yaxs : Y 軸のスケール設定方式 ("A", "F" or "D")
 - "A" : 自動設定
 - "F" : マニュアル設定
 - "D" : デフォルト値設定 (前回処理したときの値)
5. ztyp : Z 軸のスケール形式 ("N" or "L")
 - "N" : LINEAR 軸
 - "L" : LOG 軸
6. zaxs : Z 軸のスケール設定方式 ("A", "F" or "D")
 - "A" : 自動設定

"F" : マニュアル設定

"D" : デフォルト値設定 (前回処理したときの値)

7. scl1 : x 軸の最小スケール値 (Scalar)
8. scl2 : x 軸の最大スケール値 (Scalar)
9. scl3 : y 軸の最小スケール値 (Scalar)
10. scl4 : y 軸の最大スケール値 (Scalar)
11. scl5 : z 軸の最小スケール値 (Scalar)
12. scl6 : z 軸の最大スケール値 (Scalar)

解説

1. 入力を省略した場合 , 現在のスケール値を表示して入力待ちとなる .
2. SATELLITE 起動時には全軸自動設定になっている .

size

機能

描画領域のサイズを指定する .

形式

```
size( len1, len2 )
```

パラメータ

1. len1 : X 軸方向の長さ [mm] (Scalar)
2. len2 : Y 軸方向の長さ [mm] (Scalar)

解説

SATELLITE 起動時には len1 = 100[mm] , len2 = 100[mm] に設定されている .

title

機能

軸を表示する際のタイトルを指定する。

形式

```
title( tbuf [, "x-title", "y-title" [, "z-title"]] )
```

パラメータ

1. tbuf : タイトル番号 (1, 2 or 3)
2. x-title : X 軸のタイトル (30 文字以内)
3. y-title : Y 軸のタイトル (30 文字以内)
4. z-title : Z 軸のタイトル (30 文字以内)

解説

1. タイトル設定方法

x-title , y-title が省略された場合 , 現在設定されているタイトルを表示して入力待ちになる。

2. z 軸のタイトル設定について

z-title は , x-title , y-title とともに省略した場合 , 現在設定されているタイトルを表示して入力待ちになる。

wclose

機能

SATELLITE がグラフィックウィンドウをクローズする .

形式

```
wclose( wnum )
```

パラメータ

wnum : グラフィックウィンドウ番号 (Scalar) なお , "ALL" (String) を入力した場合は , 開いている全てのウィンドウをクローズする .

we

機能

SATELLITE グラフィックウィンドウの画面を消去する。

形式

`we()`

パラメータ

なし

wopen

機能

SATELLITE グラフィックウィンドウをオープンする .

形式

```
wopen( wnum, "size", orientation, device [, X, Y] )
```

パラメータ

1. wnum : グラフィックウィンドウ番号 (Scalar)
2. size : 用紙サイズ ("A4", "B4", "Free")
 - "A4" : A4 (210mm × 294mm)
 - "B4" : B4 (257mm × 364mm)
 - "Free" : フリー (X , Y で指定)
3. orientation : 用紙の向き (0 or 1)
 - 0 : Portrait
 - 1 : Landscape
4. device : 出力デバイス (0, 1 or 2)
 - 0 : ディスプレイのみ
 - 1 : ディスプレイ および ファイル(GPMDVIFILE)
 - 2 : ファイルのみ
5. X : オープンするウィンドウの X 方向サイズ [pixels] (Scalar)

device を 1 に指定した場合は , GPMDVIFILE のファイル名とみなされる .
省略した場合は , GPMDVIFILE_n で n は , wnum の値が使われる .
6. Y : オープンするウィンドウの Y 方向サイズ [pixels] (Scalar)

解説

1. マウスをウィンドウ内に移動した時には , ウィンドウ左上にポインタの座標が表示される (size = "A4" or "B4" のときは単位は [mm] , Free の場合は [pixels]) .
2. ウィンドウ内でマウスをクリックすると , クリップボードに (x,y) 座標値がコピーされる .

3. size を "Free" とした場合 , device はディスプレイのみに設定される .

第 11 章 BPS

バックプロパゲーションシミュレータ

目次

bactload	235
bcor	236
bdisp	237
berrfunc	238
berrload	240
berror	241
bfunction	242
blalgo	243
blayer	245
blearn	246
blend	247
bpdisp	248
bpinit	249
bpload	251
bpsave	252
brec	253
brvmap	255
bsetrec	256
bsigmoid	257
bteach	258
bwalgo	259
bweight	260
bwgtset	261
bwinit	262

bactload

機能

テスト (認識) 結果ファイルから Snapshot へ活性値をロードする .

形式

```
buf = bactload( pnum, unum, "fname" )
```

パラメータ

1. pnum : パターンの軸の指定 , またはパターン番号の指定 ("X", "Y" or 数値)

"X" : パターンを X 軸にとる

"Y" : パターンを Y 軸にとる

数値 : その番号の活性値を対象とする

2. unum : ユニットの軸の指定 , またはユニット番号の指定 ("X", "Y" or 数値)

"X" : ユニットを X 軸にとる

"Y" : ユニットを Y 軸にとる

数値 : その番号の活性値を対象とする

3. fname : テスト (認識) 結果ファイル名 (String)

戻り値

buf : 活性値を出力するオブジェクト (Snapshot)

bcor

機能

バッファのデータに対し，相関を計算し，データを 2 次元バッファに相関マトリクスとして格納する．

形式

```
bufout = bcor( buf, "mode" )
```

パラメータ

1. buf : 入力バッファ (Snapshot)
2. mode : 相関モード ("C" or "D")

"C" : 相関

"D" : ユークリッド距離

戻り値

1. bufout : ストア先のバッファ (Snapshot)

bdisp

機能

学習回数，誤差値などの表示間隔を設定する．

形式

```
bdisp( dint, "str" )
```

パラメータ

1. dint : 学習回数，誤差，コメント等の表示間隔 (Scalar)
2. str : 学習時に表示するコメント (String)

berrfunc

機能

評価関数を演算し，データを 2 次元バッファへ格納する．

形式

```
buf1 = berrfunc( his1, l1, us1, ud1, min1, max1, d1, l2,  
                 us2, ud2, min2, max2, d2 )
```

パラメータ

1. his1 : ウェイトヒストリ (ブロック) 番号 (Scalar)
2. l1 : 結合元層番号 (Scalar)
3. us1 : 結合元ユニット番号 (Scalar)
4. ud1 : 結合先ユニット番号 (Scalar)
5. min1 : 結合重み最小値 (Scalar)
6. max1 : 結合重み最大値 (Scalar)
7. d1 : データポイント数 (Scalar)
8. his2 : ウェイトヒストリ (ブロック) 番号 (Scalar)
9. l2 : 結合元層番号 (Scalar)
10. us2 : 結合元ユニット番号 (Scalar)
11. ud2 : 結合先ユニット番号 (Scalar)
12. min2 : 結合重み最小値 (Scalar)
13. max2 : 結合重み最大値 (Scalar)
14. d2 : データポイント数 (Scalar)

戻り値

1. buf1 : 出力バッファ (Snapshot)

解説

1. `buf1` : 指定した結合重み (1 and 2) を変化させ , 評価関数を計算する . このとき , 指定した結合重み以外は `bsetrec` で指定したウェイト履歴番号の結合重みに固定してある .

berrload

機能

エラーヒストリファイルから誤差値を Snapshot へロードする。

形式

```
buf = berrload( it, unum, "fname" )
```

パラメータ

1. `it` : 学習回数の軸の指定, またはヒストリ番号の指定 ("X", "Y" or 数値)
"X" : 学習回数を X 軸にとる
"Y" : 学習回数を Y 軸にとる
数値 : その番号の誤差を対象とする
2. `unum` : ユニットの軸の指定, またはユニット番号の指定 ("X", "Y", "S" or 数値)
"X" : ユニットの X 軸にとる
"Y" : ユニットの Y 軸にとる
"S" : 総和誤差を対象とする
数値 : その番号のユニット誤差をを対象とする
3. `fname` : エラーヒストリファイル名

戻り値

`buf` : 誤差値を格納するオブジェクト (Snapshot)

berror

機能

エラーヒストリファイルに関する設定を行う。

形式

```
berror( "fname", sint, "sdir" [, "smod" ] )
```

パラメータ

1. fname : エラーヒストリファイル名 (String)
2. sint : エラーヒストリストア間隔 (Scalar)
3. sdir : エラーヒストリストア方向 ("R" or "D")
"R" : レコード方向
"D" : データポイント方向
4. smod : エラーヒストリストアモード ("A" or "O")
"A" : 追加
"O" : 上書き

解説

1. エラーヒストリファイルの識別子には自動的に ".bhe" が付けられる。
2. smod (エラーヒストリストアモード) は, sdir が "R" (レコード方向)の時のみ設定する。

bfunction

機能

各層のユニット特性とバイアスユニットの有無を設定する．

形式

```
bfunction( [ "p1", "p2", "p3", ..., "p10" ] )
```

パラメータ

1. p1: 入力層のユニットの特性，バイアスユニットの有無の設定 ("LN", "LA", "SN" or "SA")

	"LN"	"LA"	"SN"	"SA"
ユニット特性	Linear	Linear	Sigmoid	Sigmoid
バイアスユニット	無	有	無	有

2. p2: 第 2 層目のユニットの特性，バイアスユニットの有無の設定 ("LN", "LA", "SN" or "SA")
3. pi: 第 i 層目のユニットの特性，バイアスユニットの有無の設定 ("LN", "LA", "SN" or "SA")

解説

1. blayer で設定した層数分のパラメータを入力する．入力を略した場合，現在のユニット特性，バイアスの有無を表示して入力待ちとなる．
2. blayer で層数を設定した後に，本コマンドを実行すること．

参照

blayer

blalgo

機能

ネットワークの学習アルゴリズムと学習パラメータを設定する。

形式

```
blalgo( "mode", alg, p1 [, p2, p3, p4, p5, p6 ] )
```

パラメータ

1. mode : 一括学習, 逐次学習の選択 ("S" or "P")
 - "S" : 一括学習
 - "P" : 逐次学習
2. alg : 学習アルゴリズム ($1 \leq \text{alg} \leq 6$)
 - 1 : steep 法
 - 2 : momentum 法
 - 3 : Vogl 法
 - 4 : Jacobs 法
 - 5 : momentum Vogl 係数法
 - 6 : 落合法
3. p1 : 学習率 (Scalar)
4. p2 : 慣性率 (Scalar)
5. p3 : Vogl & 落合アルゴリズム用学習率増加係数 (Scalar)
6. p4 : Vogl & 落合アルゴリズム用学習率減少係数 (Scalar)
7. p5 : Vogl アルゴリズム用閾値 (Scalar)
8. p6 : 落合アルゴリズム用平滑化微係数 (Scalar)
9. str : 構造学習法 (0, 1, 2 or 3)
 - 0 : 構造学習無し
 - 1 : Weight Decay 法
 - 2 : 忘却付学習法
 - 3 : 側抑制学習法

10. ps : 構造学習用効果率係数 (Scalar)

解説

1. 学習パラメータ (p1 ~ p6, str, ps) は, alg により次のように設定する .

alg	p1	p2	p3	p4	p5	p6	str	ps
1		-	-	-	-	-	-	-
2			-	-	-	-		
3						-	-	-
4					-	-	-	-
5			-	-	-	-	-	-
6					-			

2. 入力を略した場合, 現在の学習パラメータを表示して入力待ちとなる .

3. 構造学習法 str を指定し, sp の入力を略した場合, 現在の学習パラメータを表示して入力待ちとなる . str 以前を略した場合, 構造学習は行われないものと判断され, str, sp は入力待ちにならない .

blayer

機能

ネットワークの層数, 各層のユニット数を設定する .

形式

```
blayer( n [ , lnum1, lnum2, lnum3, ..., lnum10 ] )
```

パラメータ

1. `n` : 層数 ($3 \leq n \leq 10$)
2. `lnum1` : 入力層のユニット数 (Scalar)
3. `lnum2` : 第 2 層目のユニット数 (Scalar)
4. `lnumi` : 第 i 層目のユニット数 (Scalar)

解説

1. 指定した層数分だけユニット数を入力する . ユニット数の入力を略した場合 , 現在のユニット数を表示して入力待ちとなる .

blearn

機能

ネットワークの学習を行う。

形式

```
blearn( buf )
```

パラメータ

1. buf : バッファ名 (0 or object)
0 : 総和誤差をバッファに出力しない
object : 総和誤差を出力するバッファ名

解説

1. あらかじめ設定コマンド群を実行し, 学習に必要なパラメータを設定するか, bpload を実行し, パラメータファイルからパラメータを読み込んでおく必要がある.
2. SATELLITE のバッファモニタ機能を利用して, 学習誤差のリアルタイム表示が可能である. 利用時には, あらかじめ buf と対応するモニタ機能を起動しておく必要がある.

参照

blayer , bfunction , bweight , berror , bteach , blalgo , blend , bdisp , bm

blend

機能

ネットワークの学習の終了条件を設定する .

形式

```
blend( err, n )
```

パラメータ

1. err : 学習終了誤差 (Scalar)
2. n : 学習回数 (Scalar)

解説

1. 総和誤差が err 以下か, 学習回数が n 回になれば学習を終了する .

bpdisp

機能

BPS パラメータの現在の設定値を表示する.

形式

```
bpdisp( )
```

パラメータ

なし

参照

bpload , bpsave

bpinit

機能

シミュレーションに必要な全てのパラメータの初期設定を行う。

形式

```
bpinit( )
```

パラメータ

なし

解説

各パラメータの初期設定状態は、以下のようにになっている。

1. 層数 : 0
2. ユニット数 : 0
3. ユニットの特性 : EOS
4. 結合重みの初期値生成アルゴリズム : random
5. 乱数の種 : 1
6. 結合重み初期値 最大値 : 1.0
7. 結合重み初期値 最小値 : -1.0
8. イニシャライズウェイトファイル名 : EOS
9. ウェイトヒストリファイル名 : EOS
10. ウェイトストアインターバル : 1
11. ウェイトストアモード : append
12. エラーヒストリファイル名 : EOS
13. エラーストアインターバル : 1
14. エラーストアディレクション : record

15. エラーストアモード : append
16. 入力データファイル名 : EOS
17. 教師データファイル名 : EOS
18. 入力開始パターン番号 : 0
19. 入力最終パターン番号 : 0
20. 学習モード : 一括学習
21. 学習アルゴリズム : momentum
22. 学習率 : 0.0
23. 慣性率 : 0.0
24. Vogl アルゴリズム用増加係数 : 0.0
25. Vogl アルゴリズム用減少係数 : 0.0
26. Vogl アルゴリズム用閾値 : 0.0
27. 落合アルゴリズム用ファクタ : 0.0
28. 学習終了誤差 : 0.0
29. 学習回数 : 0
30. 表示間隔 : 0
31. コメント : EOS
32. テスト用ウェイトヒストリファイル名 : EOS
33. ウェイトヒストリ番号 : 0
34. テストデータファイル名 : EOS
35. テストデータ入力開始パターン番号 : 0
36. テストデータ入力最終パターン番号 : 0
37. 入力層番号 : 0
38. 出力層番号 : 0
39. テスト結果ファイル名 : EOS

bpload

機能

シミュレーションに必要なパラメータをパラメータファイルから読み込む。

形式

```
bpload( "fname" )
```

パラメータ

1. fname : パラメータファイル名 (String)

解説

1. パラメータファイルの識別子には自動的に ".prm" が付けられる。

参照

bpsave

bpsave

機能

現在設定されているシミュレーションに必要なパラメータをパラメータファイルに保存する。

形式

```
bpsave( "fname" )
```

パラメータ

1. fname : パラメータファイル名 (String)

解説

1. パラメータファイルの識別子には自動的に ".prm" が付けられる。
2. パラメータはコマンドリファレンス bpinit もしくは User's Manual 表 6.3 を参照。

参照

bpload

brec

機能

ネットワークのテスト (認識) 及びネットワーク構造表示を行う。

形式

```
brec( stor, disp [, mode, min, max, "fname1", "fname2" ] )
```

パラメータ

1. stor : bsetrec により設定したテスト (認識) 結果ファイルへのストア (0 or 1)
 - 0 : ストアしない
 - 1 : ストアする
2. disp : ネットワーク構造表示 (0 or 1)
 - 0 : 表示しない
 - 1 : 表示する
3. mode : 表示モード (1,2,3 or 4)
 - 1 : 活性値を色の变化で表示する
 - 2 : 活性値を正方形の大きさで表示する
 - 3 : 認識を行わず, 構造表示のみを行う (特性関数のマークを書く)
 - 4 : 認識を行わず, 構造表示のみを行う (特性関数のマークは書かない)
4. min : スケール最小値 (Scalar)
5. max : スケール最大値 (Scalar)
6. fname1 : 教師データファイル名 (教師データも共に表示する) (String)
7. fname2 : エラーファイル名 (教師データとの誤差をこのファイルにストアする) (String)

解説

1. あらかじめ bpload を実行してパラメータを読み込んでおくか, 設定コマンドを実行して必要なパラメータを設定しておく必要がある。
2. 教師データファイル名, エラーファイル名は省略しても入力待ちにはならない。

参照

blayer , bfunction , bsetrec

brvmap

機能

結合加重の逆投影演算を行い, 2 次元バッファ格納する.

形式

```
buf = brvmap( lay1, lay2, his )
```

パラメータ

1. lay1 : 出力層番号 (Scalar)
2. lay2 : 入力層番号 (Scalar)
3. his : ウェイト履歴番号 (Scalar)

戻り値

buf : 出力バッファ (Snapshot)

bsetrec

機能

ネットワークのテスト (認識) に関するパラメータを設定する .

形式

```
bsetrec( "fname1", hisno, "fname2", ps, pe, ilay, olay, "fname3" )
```

パラメータ

1. fname1 : ウェイトファイル名 (String)
2. hisno : ウェイトヒストリ番号 (Scalar)
3. fname2 : 入力データファイル名 (String)
4. ps : 入力開始パターン番号 (Scalar)
5. pe : 入力最終パターン番号 (Scalar)
6. ilay : 入力層番号 (Scalar)
7. olay : 出力層番号 (Scalar)
8. fname3 : テスト (認識) 結果出力ファイル名 (String)

解説

1. ウェイトファイル (fname1) の識別子には自動的に ".bhw" が付けられる .
2. ウェイトヒストリ番号 (hisno) は, ブロック番号に対応する .
3. 入力データファイル (fname2) の識別子には自動的に ".dat" が付けられる .
4. 出力ファイル (fname3) の識別子には自動的に ".brc" が付けられる .
5. 入力層を第 0 層とする .

bsigmoid

機能

SIGMOID 曲線上に活性値をプロットする. または, 入力総和値のヒストグラムを描く .

形式

```
bsigmoid( lay, unit, "mode", n )
```

パラメータ

1. lay : 対象ユニットの層番号 (Scalar)
2. unit : 対象ユニット番号 (Scalar)
3. mode : モード ("P" or "H")
"P" : シグモイド曲線上に活性値をプロットする
"H" : 入力総和のヒストグラムを描く
4. n : mode が "P" のときにはシンボル番号 (graph コマンド参照) , "H" のときには分割数 .

bteach

機能

入力データ，教師データに関する設定を行う。

形式

```
bteach( "fname1", "fname2", ps, pe )
```

パラメータ

1. fname1 : 入力データファイル名 (String)
2. fname2 : 教師データファイル名 (String)
3. ps : 入力開始パターン番号 (Scalar)
4. pe : 入力最終パターン番号 (Scalar)

解説

1. データファイルの識別子には自動的に ".dat" が付けられる。

bwalgo

機能

ウェイトの初期値生成に必要なパラメータを設定する.

形式

```
bwalgo( "alg", "fname", seed, rmax, rmin )
```

パラメータ

1. alg : 初期値生成アルゴリズム ("J" or "R")
"J" : JIA のアルゴリズム
"R" : 一様乱数
2. fname : 出力する初期値ファイル名 (String)
3. seed : 初期値の種 (Scalar)
4. rmax : 生成する初期値の範囲 (最大値) (Scalar)
5. rmin : 生成する初期値の範囲 (最小値) (Scalar)

解説

1. パラメータファイルの識別子には自動的に ".bhw" が付けられる .

bweight

機能

ウェイトヒストリファイルに関する設定を行う。

形式

```
bweight( "fname1", "fname2", sint, smod )
```

パラメータ

1. fname1 : ネットワークを構成するためのウェイトが格納されているファイル名 (String)
2. fname2 : ウェイトヒストリーファイル名 (String)
3. sint : ウェイトヒストリーストア間隔 (Scalar)
4. smod : ウェイトヒストリーストアモード ("A" or "O")

"A" : 追加 (レコード方向にアペンド)

"O" : 上書き (オーバーライト)

解説

1. パラメータファイルの識別子には自動的に ".bhw" が付けられる。

bwgtset

機能

ウェイトファイルに任意のリンクの重みを設定する。

形式

```
bwgtset( lay, us, ud, data )
```

パラメータ

1. lay : 層番号 (Scalar)
2. us : リンク結合元のユニット番号 (Scalar)
3. ud : リンク結合先のユニット番号 (Scalar)
4. data : 重み値 (Scalar)

bwinit

機能

ウェイトの初期値を生成し，設定したファイルに格納する．

形式

```
bwinit( )
```

パラメータ

なし

解説

1. あらかじめ bpload を実行してパラメータを読み込んでおくか，設定コマンドを実行して必要なパラメータを設定しておく必要がある．

参照

blayer , bfunction , bwalgo

第 12 章 NCS

神経回路シミュレータ

目次

nassign	264
ncal	265
nchgbuff	266
ndelay	267
ne	268
nerase	269
ngetp	270
ninteg	271
nlink	272
nlist	273
nmake	274
nout	275
npara	276
npp	277
nsclist	278
nstim	279
ntime	281

nassign

機能

NCS で使用するモデルファイル名を定義する .

形式

```
nassign( "filename" )
```

パラメータ

filename : モデルファイル名 (String)

解説

1. モデルファイルの識別子は自動的に ".mdl" が付けられる .

ncal

機能

計算を実行する .

形式

```
ncal( )
```

パラメータ

なし

解説

この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

npp , nmake

nchgbuff

機能

nout で指定できる変数の最大個数の変更

形式

```
nchgbuff( number )
```

パラメータ

number : 変数の個数の最大数 (Scalar)

解説

1. モデル記述内で exinput , output , observable 文によって指定した変数を , nout 関数で指定できる個数はある値 (デフォルトは 255) に制限されている . 大規模モデルなどで , 多くの変数を取り出すようにしたいときなどに使用する .
2. この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

nout , npp , nmake

ndelay

機能

ディレイ情報を設定する。

形式

```
ndelay( "mdl", "var", dt, [, init] )
```

パラメータ

1. mdl : モジュール名 (String)
2. var : 内部変数名 (String)
3. dt : ディレイ時間 (Scalar)
4. init : 出力の初期値 (省略時は、モデル記述で指定した値になる) ("AUTO" or 数値)

解説

1. init に "AUTO" が設定された場合、時刻 0 の出力値を初期値とする
2. ディレイはモジュール入力のみ設定可能
3. ディレイは QUEUE を使用することによって実現している。時刻 0 から dt までの間は、init の値が出力される
4. NETWORK() 中で、2 度以上呼び出しのあるモジュールでの動作は補償しない
5. この関数実行時には、npp 関数または nmake 関数の実行によって、シミュレーション条件ファイル群が生成されている必要がある。

参照

npp , nmake

ne

機能

エディタを起動する。

形式

```
ne( )
```

パラメータ

なし

解説

1. アサインされたモデルファイルを編集の対象として、エディタを起動する。
2. 環境変数 EDITOR が設定されている場合それを起動し、設定されていない場合、Windows ではメモ帳、その他の環境では vi エディタが起動される。

nerase

機能

シミュレーション条件を削除する。

形式

```
nerase( "type", "var" [, dim] )
```

パラメータ

1. type : 削除内容 ("S", "O" or "D")
"S" : 刺激情報
"O" : 出力情報
"D" : デイレイ情報
2. var : 変数名 (String)
3. dim : 配列番号 (Scalar)

解説

1. var に "ALL" を指定すれば、全ての条件を削除する。
2. 出力に配列を使用している場合、dim に配列番号も指定する。
3. この関数実行時には、npp 関数または nmake 関数の実行によって、シミュレーション条件ファイル群が生成されている必要がある。

参照

npp , nmake

ngetp

機能

パラメータを取得する .

形式

```
para = ngetp( "mdl", "var" )
```

パラメータ

1. mdl : モジュール名 (String)
2. var : パラメータ名 (String)
3. para : パラメータ値 (Scalar)

解説

1. NCS 言語記述においてパラメータ宣言された内部変数のみに有効である .
2. この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

npp , nmake

ninteg

機能

積分方法を設定する .

形式

```
ninteg( "type" [, mcell, relerr] )
```

パラメータ

1. type : 積分方式の指定 ("F","R" or "E")
"F" : 自動刻み積分法
"R" : ルンゲ-クッタ-ギル法
"E" : オイラー法
2. mcell : 最大セル数 (Scalar)
3. relerr : 数値積分の相対許容誤差 (Scalar)

解説

1. 最大セル数とは , cell , synapse , gap 文で定義した配列の最大値 .
2. この関数を指定しなかった場合 , 自動刻み積分法で計算される .
3. この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

npp , nmake

nlink

機能

npp により変換されたファイルからシミュレーション用実行形式を作成する。

形式

```
nlink( [ [ [ "opt" ] , "flag" ] , "library" ] )
```

パラメータ

1. opt : コンパイル時の最適化レベルの指定 (コンパイラに依存) (String)
"-g" : デバッグフラグ
"-O1" ~ "-O4" : 最適化レベル
2. flag : エラー or オブジェクトファイル用フラグ ("E", "O" or "N")
"E" : リンクエラー時にエラーファイルを出力する
"O" : リンクされるオブジェクトファイルを出力する
"N" : "E" と "O" の両方ファイルを出力する
3. library : リンク時に追加するライブラリの指定 (String)

解説

1. npp で変換された C 言語のソースファイルをコンパイルし, NCS ライブラリとのリンクを行い, 実行形式を作成する。
2. opt が省略された場合, デフォルトで "-g -O2" を設定する。

参照

npp

nlist

機能

パラメータリストを表示する .

形式

```
nlist( "mdl" )
```

パラメータ

1. mdl : モジュール名 (String)

解説

1. mdl に "ALL" を指定すると , 全てのパラメータを表示する .
2. この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

npp , nmake

nmake

機能

モデルファイルからシミュレーション用の実行形式を作成する．

形式

`nmake()`

パラメータ

なし

解説

1. モデルファイルに対して，プリプロセス(npp)，コンパイル・リンク(nlink)を一括で行い実行形式のファイルを作成する．
2. nassign 関数で assign されているモデルファイルが対象となる．

参照

npp , nlink , nassign

nout

機能

出力情報を設定する。

形式

```
nout( buff [, num], "mdl", com, type [, "val"] )
```

パラメータ

1. buff : 出力値を格納するバッファ名 (Series)
2. num : buff に配列を指定した場合の要素番号 (Scalar)
3. mdl : モジュール名 (String)
4. com : コンポーネント番号 (Scalar)
5. type : 出力情報 (1, 2 or 3)
 - 1 : 出力値
 - 2 : 入力値
 - 3 : 内部変数値
6. val : type = 3 を指定したとき , 内部変数名を指定する . (String)

解説

1. 引数に指定するバッファ名は , 事前に series 宣言されたものをしようする必要がある .
2. num を指定した場合 , series 変数の配列の num 番目に出力される . 例えば , buf に out , num に 3 を指定した場合 , series 変数 out[3] に結果が出力される .
3. この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

npp , nmake

npara

機能

パラメータを変更する .

形式

```
npara( "mdl", "var", num )
```

パラメータ

1. mdl : モジュール名 (String)
2. var : パラメータ名 (String)
3. num : パラメータの設定値 (Scalar)

解説

1. NCS 言語記述においてパラメータ宣言された内部変数のみに有効である .
2. この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

npp , nmake

npp

機能

NCS 言語を C 言語に変換するプリプロセッサ。

形式

```
npp( [ "model" ] )
```

パラメータ

model : モデルファイル名 (String)

nsclist

機能

シミュレーション条件を表示する。

形式

```
nsclist( "type" )
```

パラメータ

type : 表示内容 ("S","O","T" or "D")

"S" : 刺激条件の表示

"O" : 出力条件の表示

"T" : 時間情報の表示

"D" : デレイ情報の表示

解説

この関数実行時には、npp 関数または nmake 関数の実行によって、シミュレーション条件ファイル群が生成されている必要がある。

参照

npp , nmake

nstim

機能

外部入力 (刺激) を設定する .

形式

```
nstim( "mdl", com, "type", p1, [, p2, p3, p4, p5] )
```

パラメータ

1. mdl : モジュール名 (String)
2. com : コンポーネント番号 (Scalar)
3. type : 入力波形の指定 ("P","R" or "B")

"P" : パルス関数

"R" : ランプ関数

"B" : バッファ入力

4. p1 ~ p5 : パラメータ列

入力波形	p1	p2	p3	p4	p5
パルス関数 ("P")	入力開始時刻	入力初期値	高さ	時間幅	周期
ランプ関数 ("R")	入力開始時刻	入力初期値	傾き		
バッファ入力 ("B")	バッファ名	[配列番号]	[時間情報]		

解説

1. ファイル及びバッファ入力においては, Euler 法, 自動刻み幅積分法, RKG 法を選択した場合いずれにおいても, データにない点は補完して用いられる. なお, 補間には 4 次のラグランジュ補間を使用している.
2. ntime で指定する last/cal は, バッファ入力のデータ点数を越えてはならない.
3. バッファ入力の「配列番号」とは, 入力バッファに series 型の 1 次元配列を用いた場合, 何番目の series データを使用するのかを指定する. デフォルトは 0.
4. バッファ入力の「時間情報」には, 全部の入力系列の時刻を格納したバッファ名を指定する. ntime 関数で指定された計算刻みで計算されるが, ここで指定されたバッファを調べ, 計算時刻のデータがあるのであれば, 補間せず計算を行う (通常は Euler 法だと $h/2$

の点が , RGK 法だと $h/4$, $h/2$, $3h/4$ の点が補間されている [h : ntime で指定した計算刻み]).

5. この関数実行時には , npp 関数または nmake 関数の実行によって , シミュレーション条件ファイル群が生成されている必要がある .

参照

ntime , npp , nmake

ntime

機能

時間情報を設定する。

形式

```
ntime( last, cal, str [, interval] )
```

パラメータ

1. last : 計算時間 (Scalar)
2. cal : 計算刻み (Scalar)
3. str : データをストアする間隔 (Scalar)
4. interval : データをファイルに書き込む間隔 (Scalar)

解説

1. 一度のデータファイル書き込みで，str で設定した値によって決まる回数分のデータが書き込まれる。
2. $cal \leq str < last$
3. 書き込み刻みとは計算結果を出力バッファに書き込む刻み時間である． $itv > last$ の時 last 時に書き込まれる。
4. 書き込み刻みを小さく取りすぎると，シミュレーション時間が長く必要になるので適切な値を選ぶこと。
5. この関数実行時には，npp 関数または nmake 関数の実行によって，シミュレーション条件ファイル群が生成されている必要がある。

参照

npp , nmake

第 13 章 NPE

非線型パラメータ推定

目次

cdata	283
cdel	284
cdisp	285
chistory	286
cinit	287
clist	289
cload	291
clogic	292
clsearch	293
cmethod	294
cmodel	295
cnorm	296
cnumber	297
cpenalty	298
cpoint	299
cresult	300
cscale	301
cstore	302
cterm	303
cweight	304
npe	305
npeinit	306

cdata

機能

データファイルを指定する .

形式

```
cdata( "file_name" )
```

パラメータ

file_name : データファイル名 (String)

解説

1. モデル出力の数だけ , 連続したレコードに実験データ (教師データ) が格納されていなければならない .
2. データファイルの格納形式は , モデルの出力 1 個に対して 1 レコードを対応させる .

使用例

出力モデルのデータとしてサイン波 , 1000 点を与える .

```
% series x,y[1]
% x = (0~(1000-1))/(1000-1)
% y[0] = sin(2*PI*x)
% $"sin1.dat" = trans(y)
% index($"sin1.dat")
[0]:%    1    1000
```

参照

cweight

cdel

機能

設定されている推定条件を削除する。

形式

```
cdel( "type" )
```

パラメータ

type : 削除する推定条件の種類 (String)

"METHOD" : 最適化アルゴリズム

"LSEARCH" : 直線探索法

"MODEL" : モデル

"PENALTY" : 制約条件

"INIT" : パラメータの初期値

"SCALE" : スケーリング法

"TERM" : 停止基準

"NUMBER" : モデル出力の数

"POINT" : データ点数

"DATA" : データファイル名

"WIEGHT" : 評価関数の重みづけデータファイル名

"RESULT" : モデル出力の履歴ファイル名

"HISTORY" : パラメータおよび評価関数値の履歴ファイル名

"INTEG" : 積分方法

"DISP" : 表示する評価関数値の計算方法

"NORM" : 評価関数のノルムの計算方法

"ALL" : 上記全ての条件

解説

推定条件の指定には大文字，小文字どちらでも構わない。また，頭の 3 文字だけでもよい。

cdisp

機能

表示する評価関数値の計算方法を指定する．

形式

```
cdisp( type )
```

パラメータ

type : 表示する評価関数値の計算方法 (0 or 1)

0 : 重みなし

1 : 重みつき

解説

1. 与えるデータを $t_{i,j}$, モデルからの出力を $o_{i,j}$ 重みを $w_{i,j}$ とすると評価関数値 $Value$ (2 - ノルム) は

$$Value = \sum_{i,j} w_{i,j} (t_{i,j} - o_{i,j})^2$$

である．評価関数値を表示する際に $Value$ を使うが，重みを全て 1 として計算した $Value'$ を用いるかを指定する．

2. 表示する評価関数値の計算方法であって，停止基準・その他には関係しない．
3. 省略された場合は，1 (重みつき) が設定される．

参照

chistory , cnorm

chistory

機能

パラメータおよび評価関数値の履歴を格納する方法を指定する．

形式

```
chistory( "file_name", interval )
```

パラメータ

1. file_name : 格納するファイル名 (String)
2. interval : 格納間隔 (interval = 0 は interval = 1 と同じ) (Scalar)

解説

1. 推定対象のパラメータ値，評価関数値の履歴を指定した格納間隔でファイル及びバッファにアPENDモードで格納する．停止条件を満たした場合は，指定した格納間隔に関わらず格納される．
2. 評価関数値の履歴は指定したファイルの最後のレコードに格納される．
3. 格納される評価関数値は，cdisp 関数で指定した方法で計算される．また，制約項の値は加えられない．

参照

cdisp

cinit

機能

パラメータの初期値等の情報を指定する．

形式

```
cinit( ninit, inum, val, "flag", "name", span )
```

パラメータ

1. ninit : 推定するパラメータの数 (Scalar)
2. inum : パラメータ番号 (Scalar)
3. val : パラメータの初期値 (Scalar)
4. flag : パラメータの固定 / 可変の指定 (String)
 "FIX" : パラメータ値を初期値で固定 (推定しない)
 "VAR" : パラメータ値を可変 (推定する)
5. name : パラメータの名前 (String)
6. span : 差分間隔 (Scalar)

解説

1. パラメータ数 ninit の値を変更する場合は，パラメータ情報は失われる．
2. ninit 個あるパラメータのうち，1 から数えて inum 番目のパラメータの情報を指定する．
3. パラメータの名前は，使用モデルにより次のように付ける．
 - a. C 言語で記述したモデルの場合
 推定実行時に表示される名前を指定する．(モデルファイルと対応させる必要はない)
 - b. NCS モデルの場合
 各種情報を設定したパラメータを，NCS モデル記述内のモジュール名とパラメータ名を@で接続した文字列で指定する．
 (例) BHL モジュールの Tau_L というパラメータを推定する場合

```
name = Tau_L@BHL
```

また、パラメータの名前は最大で 25 文字である。

4. 各パラメータに対する偏微分は中心差分で行うため span は十分に小さい値でなければならない。

clist

機能

設定されている推定条件を表示する。

形式

```
clist( ["elmnttype"] )
```

パラメータ

elmnttype : 表示する推定条件の種類 (String)

"METHOD" : 最適化アルゴリズム

"LSEARCH" : 直線探索法

"MODEL" : モデル

"PENALTY" : 制約条件

"INIT" : パラメータの初期値

"SCALE" : スケーリング法

"TERM" : 停止基準

"NUMBER" : モデル出力の数

"POINT" : データ点数

"DATA" : データファイル名

"WIEGHT" : 評価関数の重みづけデータファイル名

"RESULT" : モデル出力の履歴ファイル名

"HISTORY" : パラメータおよび評価関数値の履歴ファイル名

"INTEG" : 積分方法

"DISP" : 表示する評価関数値の計算方法

"NORM" : 評価関数のノルムの計算方法

"ALL" : 上記全ての条件

解説

1. 表示する推定条件の指定がなければ全て表示する。
2. 推定条件の指定には大文字，小文字どちらでも構わない。また，頭の 3 文字だけでもよい。

使用例

LSEARCH の内容を表示させる。

```
% clist("lsearch")  
lsearch : golden      0.0001
```

cloud

機能

設定ファイルの内容 (推定条件) をコモン領域へ読み込む。

形式

```
cloud( "file_name" )
```

パラメータ

file_name : 設定ファイル名 (String)

解説

1. 追加，変更されたコモン領域の内容を，設定ファイルに格納 (cstore) せずに新たに読み込もうとすると，警告を発して確認を求めるフェイルセーフ機能を有している。
2. 各種コマンドの組み合わせのみで必要な条件を設定することが可能ですが，毎回変更しない条件を設定ファイルに書き込んでおくとう便利である。
3. 設定ファイルをはじめて作る時は，各コマンドで条件を設定し，cstore 関数を利用するとよい。

参照

cstore

clogic

機能

停止基準の論理式を指定する .

形式

```
clogic( "logic" )
```

パラメータ

logic : 停止基準の論理式 (String)

解説

1. cterm 関数で設定された停止基準を論理和 (\vee) , 論理積 (\wedge) および括弧を用いて組み合わせる .
2. cterm 関数より先に実行する必要がある .

使用例

```
% clogic("0|1")
% cterm(0, 1000)
% cterm(1, 1.0e-04)
% clist("term")
term      :    0|1
           TERM 0 = 1000
           TERM 1 = 0.0001
```

参照

cterm

clsearch

機能

直線探索法を指定する。

形式

```
clsearch( "linear_method", bracket_init )
```

パラメータ

1. linear_method : 直線探索法 ("golden" or "cubic")
 "golden" : 黄金分割法
 "cubic" : 三次補間法
2. bracket_init : 囲い込みの幅の初期値 (Scalar)

解説

1. ほとんどの最適化法は下位のルーチンとして直線上で評価関数値を最小にするパラメータを求めるルーチンを必要とする。黄金分割法は、パラメータに関する偏導関数を計算する必要がないが、三次補間法は計算しなければならない。前者はなめらかでない関数でも計算可能であるし、後者は滑らかならば少ないステップ数で計算が終了する。どちらも試してみるのが無難である。
2. 囲い込みの幅 (最大探索幅) は自動的に拡大する。

使用例

```
% clsearch("golden",1.0e-03)
% clist("lsearch")
```

```
lsearch :    golden          0.001
```

参照

cmethod

cmethod

機能

最適化アルゴリズムを指定する。

形式

```
cmethod( "method" )
```

パラメータ

method : 最適化法 (String)

"simplex" : Simplex 法

"bfgs" : BFGS (Broyden-Fletcher-Goldfarb-Shanno) 法

"dfp" : DFP (Davidon-Fletcher-Powell) 法

"ssvm" : SSVM (Self-Scaling Variables Metric) 法

"conjfr" : 共役勾配法 (Fletcher-Reeves の更新式)

"conjprp" : 共役勾配法 (Polak-Ribiere-Polyak の更新式)

解説

1. simplex 法以外では直線探索法も指定しなければならない。
2. 最適化法に一般的にベストだと言える方法は発見されていないため、いくつかの方法を試して見るのが常識となっている。経験的には BFGS 法が優れていると言われている。

使用例

```
% cmethod("bfgs")
% clist("method")
```

```
method : bfgs
```

参照

clsearch

cmodel

機能

モデルを指定する .

形式

```
cmodel( "model_type", "file_name" )
```

パラメータ

1. mdl_type : モデルの形式 ("USR" or "NCS")
"USR" C 言語で作成したモデル関数
"NCS" NCS で作成したモデル
2. file_name : C 言語の場合 , モデルの C ファイルもしくはオブジェクトファイル . NCS モデルの場合はオブジェクトファイル名 (String)

解説

1. モデルの記述方法については , User's Manual 第 8 章 項 2 . 「モデル記述」 を参照 .
2. NCS モデルでオブジェクトファイルを作成するには nlink 関数で第二引数に "O" を指定することで作成できる .

参照

nlink

cnorm

機能

評価関数のノルムの計算方法を指定する．

形式

```
cnorm( norm )
```

パラメータ

norm : ノルムのタイプ (0,1 or 2)

0 : 無限大ノルム

1 : 1 - ノルム

2 : 2 - ノルム

解説

1. 与えるデータを $t_{i,j}$, モデルの出力を $o_{i,j}$, 重みデータを $w_{i,j}$ とすると , 評価関数値は各ノルム毎に以下のように定義される .

- a. 0 - ノルム

$$Value = \max_{i,j} \{ w_{i,j} |t_{i,j} - o_{i,j}| \}$$

- b. 1 - ノルム

$$Value = \sum_{i,j} w_{i,j} |t_{i,j} - o_{i,j}|$$

- c. 2 - ノルム

$$Value = \sum_{i,j} w_{i,j} (t_{i,j} - o_{i,j})^2$$

2. タイプの指定がなければ , 2 - ノルムが設定される .

参照

cdisp

cnumber

機能

モデル出力の数を指定する .

形式

```
cnumber( numwave )
```

パラメータ

numwave : モデル出力の数

解説

実験データや重みデータは cnumber で指定しただけのレコード数を持たなければならない

使用例

1. 1 - 出力モデル

```
% cnumber(1);  
% clist("number");  
number : 1
```

2. データファイルから設定する .

```
% cdata("sin1.dat")  
% cnumber(length($"sin1.dat"));
```

参照

cdata , cweight

cpenalty

機能

制約条件を設定する .

形式

```
cpenalty ( "file_name" )
```

パラメータ

file_name : 制約を C または C++ 言語で記述したファイル名 (String)

解説

1. NPE で利用可能な制約の課し方はペナルティ法である . ペナルティ法はその名の通り , パラメータが制約条件を満たしていなければ評価関数値に大きな値 (ペナルティ) を加える . これによって , 制約条件を満たすように推定されることが期待される .
2. オブジェクトファイルの形式でも可 .

cpoint

機能

データ点数を指定する .

形式

```
cpoint( datapoint )
```

パラメータ

datapoint : データ点数 (Scalar)

解説

実験データや重みデータは , cpoint で指定しただけのデータ点数を持たなければならない .

使用例

ファイルに格納されているデータを指定する .

```
% cdata("sin1.dat");  
% cpoint( length( $"sin1.dat":[0] ) );
```

参照

cdata , cnumber , cweight

cresult

機能

モデル出力の履歴を格納するファイル名を指定する．

形式

```
cresult( "file_name" )
```

パラメータ

file_name : 格納するファイル名 (String)

解説

1. 推定したパラメータを用いてモデル出力を計算し，ファイル及びバッファに格納する．
2. 推定が終了した時点で実行する．

cscale

機能

スケーリング法を指定する .

形式

```
cscale( scalingmethod )
```

パラメータ

scalingmethod : スケーリング法 (0,1,2 or 3)

0 : スケーリングなし

1 : 桁スケーリング有り (初回のみ)

2 : 桁スケーリング有り (毎ステップ)

3 : re-scaling

解説

1. スケーリングを設定することで , 推定が難しい場合 (ill-condition) を改善することが期待される .
2. 推定されなかった場合は , 0 (スケーリングなし) が設定される .

cstore

機能

設定されている推定条件を指定したファイルに格納する .

形式

```
cstore( "file_name" )
```

パラメータ

file_name : 推定環境を格納する設定ファイル名 (String)

解説

設定ファイルをはじめて作成する時に利用すると便利である .

参照

cload

cterm

機能

停止基準を設定する .

形式

```
cterm( termnum, termvalue )
```

パラメータ

1. termnum : 停止基準の番号 (0,1 or 9)
 - 0 : 最大反復回数 \geq termvalue
 - 1 : 評価関数値 \leq termvalue
 - 9 : Simplex の評価関数値の標準偏差 \leq termvalue
2. termvalue : 停止基準の値 (Scalar)

解説

1. この関数で指定した停止基準は , clogic 関数で組み合わせられ , 停止基準として用いられる .
2. 必ず clogic 関数の後に実行する .

参照

cllogic

cweight

機能

評価関数に重みづけをする重みデータファイルを指定する．

形式

```
cweight( "file_name" )
```

パラメータ

file_name : 重みデータファイル名 (String)

解説

1. モデル出力の数だけ，連続したレコードに重みデータが格納されていなければならない．
2. モデルの出力 1 個に対して 1 レコードを対応させる．

使用例

1 - 出力モデルの重みデータ (1000 点，すべて 1) を生成する．

```
% series wgt[1];  
% wgt[0] = (1~1000)*0 + 1  
% $"sin1.dat" = trans(wgt)  
% index($"sin1.wgt");  
[0]:%    1      1000
```

参照

cdata

npe

機能

パラメータ推定を実行する。

形式

```
npe( res, hist )
```

パラメータ

1. res : 推定したパラメータを用いて計算したモデルの出力 (Series)
2. hist : パラメータ・評価関数値の履歴 (Series)

解説

1. 設定された推定条件に従って、パラメータ推定を行う。
2. res に格納される内容は cresult 関数で設定したファイルに格納される内容と同じ。
3. hist に格納される内容は chistory 関数で設定したファイルに格納される内容と同じ。

npeinit

機能

推定条件を初期化する .

形式

```
npeinit( )
```

パラメータ

なし

解説

cdel("ALL") と同等の機能 .

第 14 章 DCM

データ変換モジュール

目次

abf2satellite	308
atf2satellite	309
buffer2avs	310
buffer2mathematica	311
buffer2matlab	312
buffer2text	313
genesis2buffer	314
matlab2satellite	315
neuron2satellite	316
teac2satellite	317
text2buffer	318

abf2satellite

機能

AXON pClamp の ABF 形式のファイルを SATELLITE のファイルオブジェクトに変換する .

形式

```
abf2satellite( "filename.abf" )
```

パラメータ

filename.abf : 変換する ABF 形式のファイルのファイル名 (String)

解説

1. ABF 形式のファイルを SATELLITE のファイルオブジェクトに変換する .
2. 出力されるファイルオブジェクトのファイル名は , 入力された ABF 形式のファイルのファイル名の拡張子 ".abf" を ".dat" に変えたファイル名となる . 例えば , "filename.abf" の場合 "filename.dat" となる .

使用例

abf ファイル 2000-07-01-00000.abf を SATELLITE のファイルオブジェクトに変換する .

```
% abf2satellite("000-07-01-00000.abf")
```

atf2satellite

機能

AXON pClamp の ATF 形式のファイルを SATELLITE のファイルオブジェクトに変換する。

形式

```
atf2satellite( "filename.atf" )
```

パラメータ

filename.atf : 変換する ATF 形式のファイルのファイル名 (String)

解説

1. ATF 形式のファイルを SATELLITE のファイルオブジェクトに変換する。
2. 出力されるファイルオブジェクトのファイル名は、入力された ATF 形式のファイルのファイル名の拡張子 ".atf" を ".dat" に変えたファイル名となる。例えば、"filename.atf" の場合 "filename.dat" となる。

使用例

atf ファイル 2000-07-01-00000.atf を SATELLITE のファイルオブジェクトに変換する。

```
% atf2satellite("000-07-01-00000.atf")
```

buffer2avs

機能

SATELLITE の Series 型の変数を AVS のフィールドデータ形式のファイルに変換する .

形式

```
buffer2avs( "basename", x )
```

パラメータ

1. `basename` : 出力したいファイル名の拡張子を省いた文字列 (String)
2. `x` : 入力オブジェクト (Series)

解説

1. 引数に与えるファイル名には拡張子を含ませないでおく .
2. 出力されるファイルは 2 つあり , "basename" にそれぞれ ".data" と ".fld" の拡張子が付加される .

使用例

Series オブジェクト `x` を AVS のフィールドデータ形式に変換する .

```
% buffer2avs("FiledData", x)
```

buffer2mathematica

機能

SATELLITE の Series 変数を Mathematica で使用可能なリスト形式に変換する .

形式

```
buffer2mathematica( "filename", x )
```

パラメータ

1. filename : 出力ファイル名 (String)
2. x : 入力オブジェクト (Series)

解説

オブジェクトが 2 次元の場合 , 値を行列とみなしてファイルを変換する .

使用例

Series オブジェクト x を Mathematica のリスト形式に変換する .

```
% buffer2mathematica( "Mathematica.dat", x)
```

buffer2matlab

機能

SATELLITE の series 変数を MATLAB の Version 4 の MAT-ファイル形式に変換する .

形式

```
buffer2matlab( "filename.mat", "variable", x )
```

パラメータ

1. filename.mat : 出力したいファイル名 (String)
2. variable : MATLAB において実際に用いられている変数名 (Series)
3. x : 入力オブジェクト (Series)

解説

オブジェクトが 2 次元の場合 , 値を行列とみなしてファイルを変換する .

使用例

series オブジェクト x を MATLAB の MAT-ファイル形式に変換する.

```
% buffer2matlab("Matlab.mat", "Satellite",x)
```


buffer2text

機能

SATELLITE の Series 変数をテキスト形式のファイルに変換する .

形式

```
buffer2text( "filename.txt", "format", x, y,...)
```

パラメータ

1. filename.txt : テキストファイル名 (String)
2. format : フォーマット文字列 (String)
3. x, y,... : 入力オブジェクト (Series)

解説

フォーマット文字列の書式は , 整数及び実数を扱う部分のみ C 言語と同様である .

使用例

Series オブジェクト x , y をテキストファイル test.txt に出力する .

```
% buffer2text("test.txt", "%f %e\n",x,y)
```

genesis2buffer

機能

Genesis の disk_out オブジェクトによって出力されたファイルを SATELLITE の Series 変数に変換する。

形式

```
x = genesis2buffer( "filename" [, start, step] )
```

パラメータ

1. x : 出力オブジェクト (Series)
2. filename : 入力したい Genesis のファイル名 (String)
3. start : Genesis でシミュレーションした際の開始時間 (Series)
4. step : Genesis でシミュレーションした際の時間刻み幅 (Series)

解説

start および step は、任意に付加できる引数となっているが、この引数を使用する際は、あらかじめ、Series で宣言しておかなければならない。

使用例

Genesis によって出力されたファイル Genesis.dat を SATELLITE の Series 変数 x に変換する。

```
% series start,step  
% x = genesis2buffer("Genesis.dat",start,step)
```

matlab2satellite

機能

MATLAB によって出力された Version 4 の MAT-ファイル形式のファイルを SATELLITE のファイルオブジェクトに変換する。

形式

```
matlab2satellite( "filename" )
```

パラメータ

filename : 入力したいファイル名 (String)

解説

1. この関数により出力されるファイルは、実際に MATLAB で使用されていた変数名 ?? を含む mat_?.dat という名前になる。
2. また、MAT-ファイルが複素数のデータである場合、出力されるファイル名は、実数部が mat_?.r.dat、虚数部が mat_?.i.dat となる。

使用例

MATLAB の Version 4 の MAT-ファイル形式で出力されたファイル "Matlab.mat" を SATELLITE のファイルオブジェクト形式に変換する。

```
% matlab2satellite("Matlab.dat")
```

neuron2satellite

機能

Neuron によって出力されたファイルを SATELLITE のファイルオブジェクトに変換する。

形式

```
neuron2satellite( "filename" )
```

パラメータ

filename : 入力したいファイル名 (String)

解説

この関数により出力されるファイルは、実際に Neuron で使用されていたラベルを用いたファイル名になる。

使用例

Neuron で出力されたファイル Neuron.dat を SATELLITE のファイルオブジェクト形式に変換する。

```
% neuron2satellite("Neuro.dat")
```

teac2satellite

機能

TEAC 製デジタルデータレコーダー DR-M2a, DR-M3a のファイルを SATELLITE のファイルオブジェクトに変換する。

形式

```
x = teac2satellite( "basename" )
```

パラメータ

1. x : 出力オブジェクト (Series)
2. basename : DR-M[2,3]a で作られたファイル名 (拡張子は不必要) (String)

解説

作成されるファイルオブジェクトは "filename_?.dat" というファイル名で作成される。ただし, "??" の部分は DR-M[2,3]a のチャンネル番号となる。

text2buffer

機能

テキスト形式のファイルを SATELLITE の Series 型変数に変換する .

形式

```
x = text2buffer( "filename.txt" )
```

パラメータ

1. x : 出力オブジェクト (Series)
2. filename.txt : 入力したいファイル名 (String)

解説

この関数により出力されたオブジェクトはテキストファイルが $n \times m$ の場合 , 時系列を 2 次元の行列にみたてた形で表現される .

使用例

テキストファイル text.txt を SATELLITE の Series オブジェクト x に変換する .

```
% x = text2buffer( "text.txt" )
```

第 15 章 STATISTICS

統計解析パッケージ

目次

15.1. STATISTIC	319
15.2. TEST	344

ave

機能

平均値 (average) を求める .

形式

$$y = \text{ave}(x)$$

パラメータ

1. x : 入力データ (Series , Snapshot)
2. y : 平均値 (Scalar)

解説

$$\bar{x} = \frac{\sum x}{n}$$

var

機能

分散 (variance) を求める .

形式

```
y = var( x )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. y : 分散値 (Scalar)

解説

$$s^2 = \frac{\sum (X - \bar{x})^2}{n - 1}$$

stddev

機能

標準偏差 (standard deviation) を求める .

形式

```
y = stddev( x )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. y : 標準偏差値 (Scalar)

解説

$$s = \sqrt{\frac{\sum (X - \bar{x})^2}{n - 1}}$$

stderr

機能

標準誤差 (standard error) を求める .

形式

```
y = stderr( x )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. y : 標準誤差値 (Scalar)

解説

$$s\bar{x} = \frac{s}{\sqrt{n}}$$

corrcoef

機能

相関係数 (correlation coefficient) を求める .

形式

```
z = corrcoef( x , y )
```

パラメータ

1. x, y : 入力データ (Series , Snapshot)
2. z : 相関係数値 (Scalar)

解説

$$\begin{aligned}
 r_{xy} &= \frac{\sum_i^N (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i^N (x_i - \bar{x})^2} \sqrt{\sum_i^N (y_i - \bar{y})^2}} \\
 &= \frac{\frac{\sum_i^N (x_i - \bar{x})(y_i - \bar{y})}{N}}{\sqrt{\frac{\sum_i^N (x_i - \bar{x})^2}{N}} \sqrt{\frac{\sum_i^N (y_i - \bar{y})^2}{N}}}
 \end{aligned}$$

varcovmat

機能

分散共分散行列 (variance covariance matrix) を求める .

形式

```
y = varcovmat( x )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. y : 分散共分散行列 (Series)

解説

サンプル数 n , 変数の数 p の入力データ x

$$\begin{pmatrix} x_{11} & x_{12} \cdots & x_{1p} \\ x_{21} & x_{22} \cdots & x_{2p} \\ \vdots & \vdots \ddots & \vdots \\ x_{n1} & x_{n2} \cdots & x_{np} \end{pmatrix}$$

の分散共分散行列は ,

$$\begin{pmatrix} V_{11} & V_{12} \cdots & V_{1p} \\ V_{21} & V_{22} \cdots & V_{2p} \\ \vdots & \vdots \ddots & \vdots \\ V_{n1} & V_{n2} \cdots & V_{np} \end{pmatrix}$$

但し ,

$$V_{ij} = \frac{\sum_{a=1}^n (x_{ai} - \bar{x}_i)(x_{aj} - \bar{x}_j)}{n-1} \quad (i, j = 1 \cdots p)$$

対角要素は各変数の分散 , その他は変数同士の共分散を返す .

normpdf

機能

平均 μ , 標準偏差 σ で表される正規分布 (normal distribution) の入力データ x に対する確率分布関数値 (probability density function) を返す .

形式

```
y = normpdf( x , mu , sig )
```

パラメータ

1. x : 入力データ (Scalar , Series , Snapshot)
2. μ : 平均値 (Scalar)
3. σ : 標準偏差 (Scalar) 正の数に限る
4. y : 正規確率密度関数値 (Scalar , Series , Snapshot)

解説

$$y = f(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『 C 言語による最新アルゴリズム』 , pp133-135 , 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』 , pp64-71 , 東海大学出版会 .

normcdf

機能

平均 μ , 標準偏差 σ で表される正規分布 (normal distribution) の入力データ x に対する正規累積分布関数値 (cumulative distribution function) を返す .

形式

```
y = normcdf( x , mu , sig )
```

パラメータ

1. x : 入力データ (Scalar , Series , Snapshot)
2. μ : 平均値 (Scalar)
3. σ : 標準偏差 (Scalar) 正の数に限る
4. y : 正規累積分布関数値 (Scalar , Series , Snapshot)

解説

$$y = f(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt$$

実際の計算には , Hastings の近似式を使用する .

$$w = |x|$$

$$a_1 = 0.049867347$$

$$a_2 = 0.0211410061$$

$$a_3 = 0.0032776263$$

$$a_4 = 3.80036 * 10^{-5}$$

$$a_5 = 4.88906 * 10^{-5}$$

$$a_6 = 5.383 * 10^{-6}$$

$$p_1 = \{1 + w * (a_1 + w * (a_2 + w * (a_3 + w * (a_4 + w * (a_5 + w * a_6)))))\}^{16}$$

$$p_2 = 1 - \frac{1}{2 * p}$$

$$p_3 = \frac{1}{2} + \left(p_2 - \frac{1}{2}\right) * \operatorname{sgn}(x)$$

$$p = 1 - p_3$$

参照

1. 奥村晴彦 (1991): ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』, pp133-135, 技術評論社.
2. 大滝厚 (1984): マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp64-71, 東海大学出版会.
3. 超幾何関数を用いた確率分布の計算

<http://www.mcc.pref.miyagi.jp/people/ikuro/koramu/tyokika.htm>

norminv

機能

平均 μ , 標準偏差 σ で表される正規累積分布関数に対して , 確率 p を与える値 (inverse of cumulative distribution function) を返す .

形式

```
y = norminv( p , mu , sig )
```

パラメータ

1. p : 確率 (Scalar , Series , Snapshot) $0 < p < 1$
2. μ : 平均値 (Scalar)
3. σ : 標準偏差 (Scalar) 正の数に限る
4. y : 正規累積分布関数値 (Scalar , Series , Snapshot)

解説

$$x = F^{-1}(p|\mu, \sigma) = \{x : F(x|\mu, \sigma) = p\}$$

但し ,

$$p = F(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt$$

実際の計算には , 戸田の近似式を使用する .

正規分布の上側確率が Q となるパーセント点 $u(Q)$ は ,

$$u(Q) = [x(b_0 + x(b_1 + x(b_2 + x(b_3 + x(b_4 + x(b_5 + x(b_6 + x(b_7 + x(b_8 + x(b_9 + x*b_{10})))))))))))]^{\frac{1}{2}}$$

但し , $x = -\log[4Q(1-Q)]$

$$b_0 = 0.1570796288 * 10$$

$$b_1 = 0.3706987906 * 10^{-1}$$

$$b_2 = -0.8364353589 * 10^{-3}$$

$$b_3 = -0.2250947176 * 10^{-3}$$

$$b_4 = 0.6841218299 * 10^{-5}$$

$$b_5 = 0.5824238515 * 10^{-5}$$

$$b_6 = -0.1045274970 * 10^{-5}$$

$$b_7 = 0.8360937017 * 10^{-7}$$

$$b_8 = -0.3231081277 * 10^{-8}$$

$$b_9 = 0.3657763036 * 10^{-10}$$

$$b_{10} = 0.6936233982 * 10^{-12}$$

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』, pp133-135, 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp64-71, 東海大学出版会 .
3. 超幾何関数を用いた確率分布の計算

<http://www.mcc.pref.miyagi.jp/people/ikuro/koramu/tyokika.htm>

fpdf

機能

分子の自由度 v_1 , 分母の自由度 v_2 で表される F 分布 (F distribution) の入力データ x に対する確率密度関数値 (probability density function) を返す .

形式

$$y = \text{fpdf}(x, v_1, v_2)$$

パラメータ

1. x : 入力データ (Scalar , Series , Snapshot)
2. v_1 : 分子の自由度 (Scalar) 正の整数に限る
3. v_2 : 分母の自由度 (Scalar) 正の整数に限る
4. y : F 確率密度関数 (Scalar , Series , Snapshot)

解説

$$y = f(x|v_1, v_2) = \frac{\Gamma\left(\frac{v_1+v_2}{2}\right)}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_2}{2}\right)} \left(\frac{v_1}{v_2}\right)^{\frac{v_2}{2}} \frac{x^{\frac{v_1-2}{2}}}{\left[1 + \left(\frac{v_1}{v_2}\right)x\right]^{\frac{v_1+v_2}{2}}}$$

ガンマ関数 $\Gamma(x)$ の算出には , Hastings の 8 次多項式近似

$$\Gamma(x+1) \doteq 1 - 0.577191652x + 0.988205891x^2 - 0.897056937x^3 + 0.918206857x^4 \\ - 0.756704078x^5 + 0.482199394x^6 - 0.193527818x^7 + 0.035868343x^8$$

を用いる .

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』 , pp334-346 , 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』 , pp87-89 , 東海大学出版会 .

fcdf

機能

分子の自由度 ν_1 , 分母の自由度 ν_2 で表される F 累積分布関数の入力データ x に対する F 累積分布関数値 (cumulative distribution function) を返す .

形式

```
y = fcdf( x ,  $\nu_1$  ,  $\nu_2$  )
```

パラメータ

1. x : 入力データ (Scalar , Series , Snapshot)
2. ν_1 : 分子の自由度 (Scalar) 正の整数に限る
3. ν_2 : 分母の自由度 (Scalar) 正の整数に限る
4. y : F 累積分布関数値 (Scalar , Series , Snapshot)

解説

$$p = F(\alpha|\nu_1, \nu_2) = \int_0^x \frac{\Gamma\left(\frac{\nu_1+\nu_2}{2}\right)}{\Gamma\left(\frac{\nu_1}{2}\right)\Gamma\left(\frac{\nu_2}{2}\right)} \left(\frac{\nu_1}{\nu_2}\right)^{\frac{\nu_2}{2}} \frac{x^{\frac{\nu_1-2}{2}}}{\left[1 + \left(\frac{\nu_1}{\nu_2}\right)x\right]^{\frac{\nu_1+\nu_2}{2}}} dt$$

実際の計算には , 不完全ベータ関数比を用いる .

不完全ベータ関数比

$$I_x(a, b) = \frac{1}{B(a, b)} \int_0^x t^{a-1}(1-t)^{b-1} dt \quad (0 \leq x \leq 1)$$

F 分布の上側確率を不完全ベータ関数比を用いて表すと ,

$$F(\alpha|\nu_1, \nu_2) = I_x\left(\frac{\nu_2}{2}, \frac{\nu_1}{2}\right) = 1 - I_{1-x}\left(\frac{\nu_1}{2}, \frac{\nu_2}{2}\right), \quad x = \frac{\nu_2}{\nu_2 + \nu_1\alpha}$$

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』 , pp344-346 , 技術評論社 .

2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp87-89 , 東海大学出版会 .
3. F 分布の計算

http://www.sist.ac.jp/~suganuma/cpp/2-bu/7-sho/Java/f_j.txt

finv

機能

分子の自由度 ν_1 , 分母の自由度 ν_2 で表される F 累積分布関数に対して , 確率 p を与える値 (inverse of cumulative distribution function) を返す .

形式

$$y = \text{finv}(p, \nu_1, \nu_2)$$

パラメータ

1. p : 確率 (Scalar , Series , Snapshot) $0 < p < 1$
2. ν_1 : 分子の自由度 (Scalar) 正の整数に限る
3. ν_2 : 分母の自由度 (Scalar) 正の整数に限る
4. y : F 累積分布逆関数 (Scalar , Series , Snapshot)

解説

$$\alpha = F^{-1}(p|\nu_1, \nu_2) = \{\alpha : F(\alpha|\nu_1, \nu_2) = p\}$$

但し ,

$$p = F(\alpha|\nu_1, \nu_2) = \int_0^x \frac{\Gamma\left(\frac{\nu_1 + \nu_2}{2}\right)}{\Gamma\left(\frac{\nu_1}{2}\right)\Gamma\left(\frac{\nu_2}{2}\right)} \left(\frac{\nu_1}{\nu_2}\right)^{\frac{\nu_2}{2}} \frac{x^{\frac{\nu_1 - 2}{2}}}{\left[1 + \left(\frac{\nu_1}{\nu_2}\right)x\right]^{\frac{\nu_1 + \nu_2}{2}}} dt$$

実際の計算には , $((1 - b)^2 - bu_\alpha^2 > 0.8)$ が成り立つ場合には , Paulson・竹内の近似式を使用する .

$$F(\alpha|\nu_1, \nu_2) \doteq \left[\frac{(1 - a)(1 - b) + u_\alpha \sqrt{(1 - a)^2 b + (1 - b)^2 a - abu_\alpha^2}}{(1 - b)^2 - bu_\alpha^2} \right]^3 \quad \left(a = \frac{2}{9\nu_1}, \quad b = \frac{2}{9\nu_2} \right)$$

$((1 - b)^2 - bu_\alpha^2 > 0.8)$ の場合は , 累積分布逆関数の公式について , ニュートン法により一定の誤差範囲内まで補正を行う .

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』, pp344-346, 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp87-89, 東海大学出版会 .
3. F 分布の計算
http://www.sist.ac.jp/~suganuma/cpp/2-bu/7-sho/Java/f_j.txt

tpdf

機能

自由度 v で表されるスチューデントの t 分布の, 入力データ x に対する確率密度関数値 (probability density function) を返す.

形式

```
y = tpdf( x , v )
```

パラメータ

1. x : 入力データ (Scalar, Series, Snapshot)
2. v : 自由度 (Scalar) 正の整数に限る
3. y : スチューデントの t 確率密度関数 (Scalar, Series, Snapshot)

解説

$$y = f(x|\nu) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})} \frac{1}{\sqrt{\nu\pi}} \frac{1}{(1 + \frac{x^2}{\nu})^{\frac{\nu+1}{2}}}$$

参照

1. 奥村晴彦 (1991): ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』, pp428-430, 技術評論社.
2. 大滝厚 (1984): マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp78-87, 東海大学出版会.

tcdf

機能

自由度 v で表されるスチューデントの t 累積分布関数の入力データ x に対する t 累積分布関数値 (cumulative distribution function) を返す。

形式

$$y = \text{tcdf}(x, v)$$

パラメータ

1. x : 入力データ (Scalar, Series, Snapshot)
2. v : 自由度 (Scalar) 正の整数に限る
3. y : スチューデントの t 累積分布関数 (Scalar, Series, Snapshot)

解説

$$p = F(\alpha|\nu) = \int_{-\infty}^x \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})} \frac{1}{\sqrt{\nu\pi}} \frac{1}{(1 + \frac{t^2}{\nu})^{\frac{\nu+1}{2}}} dt$$

実際の計算には, Wallace の近似式を用いる。

$$u = \left(1 - \frac{2}{3 + 8 * \nu}\right) * \sqrt{\nu * \log\left(\frac{x^2}{\nu + 1}\right)}$$

$$p = \text{normcdf}(u, 1, 0)$$

参照

1. 奥村晴彦 (1991): ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』, pp428-430, 技術評論社。
2. 大滝厚 (1984): マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp78-87, 東海大学出版会。
3. 超幾何関数を用いた確率分布の計算

<http://www.mcc.pref.miyagi.jp/people/ikuro/koramu/tyokika.htm>

tinvs

機能

自由度 ν で表されるスチューデントの累積分布関数に対して、確率 p を与える値 (inverse of cumulative distribution function) を返す。

形式

$$y = \text{tinvs}(p, \nu)$$

パラメータ

1. p : 確率 (Scalar, Series, Snapshot) $0 < p < 1$
2. ν : 自由度 (Scalar) 正の整数に限る
3. y : スチューデントの t 累積分布関数値 (Scalar, Series, Snapshot)

解説

$$\alpha = F^{-1}(p|\nu) = \{\alpha : F(\alpha|\nu) = p\}$$

但し、

$$p = F(\alpha|\nu) = \int_{-\infty}^{\alpha} \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})} \frac{1}{\sqrt{\nu\pi}} \frac{1}{(1 + \frac{t^2}{\nu})^{\frac{\nu+1}{2}}} dt$$

実際の計算には、山内の近似式を使用する。

正規分布 $N(0,1)$ で確率 α となるパーセント点を u_α とすると、

$$F(\alpha|\nu) = u_\alpha \left(1 + \frac{Y_1(u_\alpha)}{\nu} + \frac{Y_2(u_\alpha)}{\nu^2} + \frac{Y_3(u_\alpha)}{\nu^3} + \dots + \frac{Y_i(u_\alpha)}{\nu^i} \right)$$

但し、

$$Y_1(u) = \frac{1}{2^2}(u^3 + u)$$

$$Y_2(u) = \frac{1}{2^5 * 3}((5u^2 + 16) * u^2 + 3)$$

$$Y_3(u) = \frac{1}{2^7 * 3}(((3u^2 + 19) * u^2 + 17) * u^2 - 15)$$

$$Y_4(u) = \frac{1}{2^{11} * 3^2 * 5} (((((79u^2 + 766) * u^2 + 1482) * u^2 - 1920) * u^2 - 945)$$

$$Y_5(u) = \frac{1}{2^{13} * 3^2 * 5} ((((((27u^2 + 339) * u^2 + 930) * u^2 - 1782) * u^2 - 765) * u^2 + 17955)$$

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』, pp428-430, 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp78-87, 東海大学出版会 .
3. 超幾何関数を用いた確率分布の計算

<http://www.mcc.pref.miyagi.jp/people/ikuro/koramu/tyokika.htm>

chi2pdf

機能

自由度 ν で表される χ^2 分布の入力データ x に対する確率密度関数 (probability density function) を返す .

形式

```
y = chi2pdf( x , v )
```

パラメータ

1. x : 入力データ (Scalar , Series , Snapshot)
2. ν : 自由度 (Scalar) 正の整数に限る
3. y : χ^2 分布確率密度関数 (Scalar , Series , Snapshot)

解説

$$y = f(x|\nu) = \frac{x^{\frac{\nu-2}{2}} e^{-\frac{x}{2}}}{2^{\frac{\nu}{2}} \Gamma(\frac{\nu}{2})}$$

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』 , pp26-27 , 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』 , pp71-78 , 東海大学出版会 .
3. 確率分布ライブラリ

<http://www.mcc.pref.miyagi.jp/people/ikuro/koramu/tyokika.htm>

chi2cdf

機能

自由度 ν で表される χ^2 累積分布関数の入力データ x に対する t 累積分布関数値 (cumulative distribution function) を返す .

形式

```
y = chi2cdf( x , ν )
```

パラメータ

1. x : 入力データ (Scalar , Series , Snapshot)
2. ν : 自由度 (Scalar) 正の整数に限る
3. y : χ^2 分布確率密度関数 (Scalar , Series , Snapshot)

解説

$$p = F(\alpha|\nu) = \int_0^x \frac{t^{\frac{\nu-2}{2}} e^{-\frac{t}{2}}}{2^{\frac{\nu}{2}} \Gamma(\frac{\nu}{2})} dt$$

実際の計算には , Wilson-Hilferty の近似式を使用する .

正規分布 $N(0,1)$ で確率 α となるパーセント点を u_α とすると ,

$$F(\alpha|1) = \left(u \left(\frac{\alpha}{2} \right) \right)^2$$

$$F(\alpha|2) = -2 \log \alpha$$

$$F(\alpha|\nu) = \nu \left\{ \left(1 - \frac{2}{9\nu} + u_\alpha \sqrt{\frac{2}{9\nu}} \right)^3 \right\} \quad (\nu > 2)$$

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『 C 言語による最新アルゴリズム 』 , pp26-27 , 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『 パソコン BASIC 統計解析 』 , pp71-78 , 東海大学出版会 .

3. 確率分布ライブラリ

<http://www5.airnet.ne.jp/tomy/cpro/sslib11.htm>

chi2inv

機能

自由度 ν で表される χ^2 累積分布関数に対して確率 p を与える値 (inverse of cumulative distribution function) を返す .

形式

$$y = \text{chi2inv}(p, \nu)$$

パラメータ

1. p : 確率 (Scalar , Series , Snapshot) $0 < p < 1$
2. ν : 自由度 (Scalar) 正の整数に限る
3. y : χ^2 分布確率密度関数 (Scalar , Series , Snapshot)

解説

$$\alpha = F^{-1}(p|\nu) = \{\alpha : F(\alpha|\nu) = p\}$$

但し ,

$$p = F(\alpha|\nu) = \int_0^\alpha \frac{t^{\frac{\nu-2}{2}} e^{-\frac{t}{2}}}{2^{\frac{\nu}{2}} \Gamma(\frac{\nu}{2})} dt$$

実際の計算には , Cornish-Fisher の近似式を使用する .

正規分布 $N(0,1)$ で確率 α となるパーセント点を u_α とすると ,

$$F^{-1}(\alpha|1) = \left(-u\left(\frac{\alpha}{2}\right)\right)^2$$

$$F^{-1}(\alpha|2) = -2 \log \alpha$$

$$c1 = \frac{(x^2 - 7)x}{9\sqrt{2\nu}}$$

$$c2 = \frac{((3x^2 + 7)x^2 - 16)2}{405\nu}$$

$$F^{-1}(\alpha|\nu) = \nu + x\sqrt{2\nu} + \frac{2(x^2 - 1)}{3} + c1 - c2 \quad (\nu > 2)$$

参照

1. 奥村晴彦 (1991) : ソフトウェアテクノロジー 13 『C 言語による最新アルゴリズム』, pp26-27, 技術評論社 .
2. 大滝厚 (1984) : マイコン活用シリーズ 『パソコン BASIC 統計解析』, pp71-78, 東海大学出版会 .
3. 確率分布ライブラリ
<http://www5.airnet.ne.jp/tomy/cpro/sslib11.htm>

normal

機能

入力データが正規分布 (normal distribution) に従うかどうか, Kolmogorov-Smirnov の検定を行う.

形式

```
y = normal( x )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. y : 検定結果 (Series)
 - a. y : [0] 検定統計量
 - b. y : [1] p 値

解説

データの値を $x_i (i = 1, 2, \dots, n)$, 標準得点を z_i とする.

(標準得点: $z_i = \frac{x_i - \bar{x}}{s}$ \bar{x} : 平均値 s : 標準偏差)

各 z_i について標本分布関数 $S_i(x)$ (累積相対度数) を計算する. 度数 1 の相対度数は, $\frac{1}{n}$, $S_i = \frac{i}{n}$ となる. 2 つの z_i が同じ値を持つ所では, 度数 2 に相当する分だけ増加する.

各 z_i について, 標準正規分布の累積確率の値を $F_i(x)$ とする.

検定統計量 $D = \max \{|S_i(x) - F_i(x)|\}$

このとき,

$$\lim_{n \rightarrow \infty} P\left(D_n \leq \frac{\lambda}{\sqrt{n}}\right) = Q(\lambda), \quad Q(\lambda) = \sum_{k=-\infty}^{\infty} (-1)^k e^{-2k^2 \lambda^2}$$

が成り立ち, 検定量 $\sqrt{n}D$ は漸近的に $Q(\lambda)$ に従って分布する.

Kolmogorov-Smirnov の λ 分布表の値より p 値を算出する.

参照

1. 天野健太郎 (編) (1985): 『数学ハンドブック』, pp31 , 787 , 森北出版 .
2. 奥野忠一 (編) (1986): 『応用統計ハンドブック』, pp85-87 , 養賢堂 .

homogeneity

機能

入力データの分散が等しい (homogeneity) かどうか , Bartlett の検定を行う .

形式

```
y = homogeneity( x, n )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : 標本数 (Series)
3. y : 検定結果 (Series)
 - a. y : [0] 検定統計量
 - b. y : [1] p 値

解説

サンプル数を $n_i (i = 1, 2, \dots, k)$ とする .

$$Q_i = \sum_j (x_{ij} - \bar{x}_i)^2 (i = 1, 2, \dots, k; j = 1, 2, \dots, n_i)$$

$$V_i = \frac{Q_i}{n_i - 1}$$

$$f = \sum_{i=1}^k (n_i - 1)$$

$$M = f \log \left(\frac{1}{f} \sum_i Q_i \right) - \sum_i (n_i - 1) \log V_i$$

$$B = \frac{M}{1 + \frac{1}{3(k-1)} \left(\sum_i \frac{1}{n_i - 1} - \frac{1}{f} \right)}$$

検定統計量 B は , 自由度 (k-1) の χ^2 分布に従うため , p 値は自由度 (k-1) の χ^2 分布の累積分布関数より算出する .

参照

池田央 (編) (1989) : 『統計ガイドブック』, pp89 , 新曜社 .

ttest

機能

対応のない 2 つの入力データ x, y が同じ平均を持つかどうか t 検定 (non-paired t-test) を行う。2 群の等分散性を仮定する。

形式

```
z = ttest ( x, y, type )
```

パラメータ

1. x, y : 入力データ (Series, Snapshot)
2. type : 片側 / 両側検定 ("T" or "O")
 - a. "T" : two-sided
 - b. "O" : one-sided
3. z : 検定結果 (Series)
 - a. y : [0] 検定統計量
 - b. y : [1] p 値

解説

両群のサンプル数 n_1, n_2 , 平均値 \bar{X}_1, \bar{X}_2 , 不偏分散 u_1, u_2 とする。

検定統計量

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{\sum_{i=1}^{n_1} (X_{1i} - \bar{X}_1)^2 + \sum_{i=1}^{n_2} (X_{2i} - \bar{X}_2)^2}{n_1 + n_2 - 1} \left(\frac{1}{n_1} + \frac{1}{n_2} \right)}}$$

$$= \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{(n_1 - 1)u_1^2 + (n_2 - 1)u_2^2}{n_1 + n_2 - 2} \left(\frac{1}{n_1} + \frac{1}{n_2} \right)}}$$

自由度 $df = n_1 + n_2 - 2$

p 値は、自由度 df の t 分布の累積分布関数より算出する。

参照

森 敏昭，吉田寿夫 (1990)：『心理学のためのデータ解析テクニカルブック』，pp59-65，北大路書房．

ttestp

機能

対応のある 2 つの入力データ x, y が同じ平均を持つかどうか t 検定 (paired t-test) を行う。

形式

```
z = ttestp ( x , y , type )
```

パラメータ

1. x, y : 入力データ (Series , Snapshot)
2. type : 片側 / 両側検定 ("T" or "O")
 - a. "T" : two-sided
 - b. "O" : one-sided
3. z : 検定結果 (Series)
 - a. y : [0] 検定統計量
 - b. y : [1] p 値

解説

両群のサンプル数 n , 平均値 $\overline{X_1}, \overline{X_2}$, 不偏分散 u_1, u_2 とする。

検定統計量

$$t = \frac{\overline{D}}{\sqrt{\frac{\sum_i^n (D_i - \overline{D})^2}{n(n-1)}}}$$

$$= \frac{\overline{D}}{\sqrt{\frac{n \sum_i^n D_i^2 - \left(\sum_i^n D_i\right)^2}{n^2(n-1)}}}$$

自由度 $df = n_1$

p 値は , 自由度 df の t 分布の累積分布関数より算出する。

参照

森 敏昭，吉田寿夫 (1990)：『心理学のためのデータ解析テクニカルブック』，pp59-65，北大路書房．

welch

機能

対応のない 2 つの入力データ x, y が同じ平均を持つかどうか Welch の検定 (welch's test) を行う。2 群の等分散性を仮定しない。

形式

```
z = welch ( x , y , type )
```

パラメータ

1. x, y : 入力データ (Series , Snapshot)
2. type : 片側 / 両側検定 ("T" or "O")
 - a. "T" : two-sided
 - b. "O" : one-sided
3. z : 検定結果 (Series)
 - a. y : [0] 検定統計量
 - b. y : [1] p 値

解説

両群のサンプル数 n_1, n_2 , 平均値 $\overline{X_1}, \overline{X_2}$, 不偏分散 u_1, u_2 とする。

検定統計量

$$t' = \frac{\overline{X_1} - \overline{X_2}}{\sqrt{\frac{u_1^2}{n_1} + \frac{u_2^2}{n_2}}}$$

自由度

$$df = \frac{\left(\frac{u_1^2}{n_1} + \frac{u_2^2}{n_2} \right)}{\frac{u_1^4}{n_1^2(n_1-1)} + \frac{u_2^4}{n_2^2(n_2-1)}}$$

p 値は , 自由度 df の t 分布の累積分布関数より算出する。

参照

森 敏昭，吉田寿夫 (1990)：『心理学のためのデータ解析テクニカルブック』，pp59-65，北大路書房．

wilcoxon

機能

対応のない 2 つの入力データ x, y の差について Wilcoxon の順位和検定 (wilcoxon rank sum test)を行う。特定の分布を仮定せずに、データの順位情報のみを利用する。

形式

```
z = wilcoxon ( x , y )
```

パラメータ

1. x, y : 入力データ (Series , Snapshot)
2. z : 検定結果 (Series)
 - a. $z[0]$ 検定統計量
 - b. $z[1]$ p 値 ($n > 20$) , または $p = .05$ の棄却値の下限 ($n \leq 20$)
 - c. $z[2]$ $p = .05$ の棄却値の上限 ($n \leq 20$)

解説

両群のサンプル数 $n_1, n_2 (n_1 \leq n_2)$ とする。

両群のデータを合わせて小さい順に順位 r_{1i}, r_{2j} を付ける。但し、同じ値には、それらの値が異なると考えた場合の順位の平均値を付ける。データが少ないほうの一群 (n_1) について順位の総和を取り、 W とする。

検定統計量

$$W = \sum_{i=1}^{N_1} r_{1i}$$

$n_2 \leq 20$ の場合、

Wilcoxon の数表を参照し、棄却限界値 $\frac{u_1}{u_2}$ を求める。

$W \leq u_1$ または $W \geq u_2$ のとき、 $p = .05$ で有意に差があるといえる。

$u_2 > 20$ の場合、

検定統計量 W は近似的に $N(0, 1)$ に従うため、

$$Z_0 = \frac{\left| W - \frac{n_1(n_1+n_2+1)}{2} \right| - \frac{1}{2}}{\sqrt{\frac{n_1 n_2 (n_1+n_2+1)}{12}}}$$

について , $p = Pr \{ |Z| \geq Z_0 \}$ を算出する .

参照

石村貞夫 (1990) : 『統計解析のはなし』 , pp258-264 , 東京図書 .

wilcoxonp

機能

対応のある 2 つの入力データ x, y の差について Wilcoxon の符号付き順位和検定 (wilcoxon sign rank test) を行う。

形式

```
z = wilcoxonp ( x , y )
```

パラメータ

1. x, y : 入力データ (Series , Snapshot)
2. z : 検定結果 (Series)
 - a. $z[0]$ 検定統計量
 - b. $z[1]$ p 値 , または $p = .05$ の棄却限界値 ($n \leq 50$)

解説

両群のサンプル数 $n_1, n_2 (n = n_1 + n_2)$ とする。

対応する各標本の差 $z_i = x_i - y_i$ について, 0 を除いて $|z_i|$ の小さい順に順位 r_i を付け, z_i の正負で 2 群 R_+, R_- に分類する。但し, 同じ値の場合は順位の平均をとる。 R_+, R_- のうち, 順位和の小さいほうを R とする。

$n \leq 50$ の場合,

Wilcoxon の符号付き順位和検定量の数表より棄却限界値 r を求める。 $R \leq r$ のとき, $p = .05$ で有意に差があるといえる。

$n > 50$ の場合,

検定統計量 R は近似的に $N(0, 1)$ に従うため,

$$Z = \frac{|R - \frac{n(n+1)}{4}| - \frac{1}{2}}{\sqrt{\frac{n(n+1)(2n+1)}{24}}}$$

について, $p = Pr \{|Z| \geq Z_0\}$

参照

ウィルコクスの符号順位検定

<http://aoki2.si.gunma-u.ac.jp/lecture/Average/mpsr-test.html>

anova1

機能

一元配置分散分析 (one way analysis of variance) を行い, 分散分析表を表示する. x の列の平均と行の平均が等しいという帰無仮説によって p の値を出力する.

形式

```
y = anova1( x, n, type )
```

パラメータ

1. x : 入力データ (Series, Snapshot)
2. n : 標本数 (Series)
3. $type$: 対応あり / なし ("P" or "N")
 - a. "P" paired data
 - b. "N" non-paired data
4. y : 検定結果 (Series)
 - a. $y[0]$: 要因 1 の主効果の検定統計量, p 値
 - b. $y[1]$: 繰り返し測定 of 検定統計量, p 値 ($type = "P"$ の時)

ANOVA 表: 6 つの列を表示

- 1 列: 変化量の要因
- 2 列: 各要因の平方和
- 3 列: 各要因の自由度
- 4 列: 各要因の平均平方
- 5 列: F 値
- 6 列: p 値

解説

	要因				
群	1	2		k	
	X_{11}	X_{12}		X_{1k}	
	X_{21}	:		:	

	要因				
	\vdots $X_{n_1 1}$	\vdots $X_{n_2 2}$		\vdots $X_{n_1 k}$	
要素数	n_1	n_2		n_k	n
平均	\bar{X}_1	\bar{X}_2		\bar{X}_k	\bar{X}

1. 対応がない場合 (データ数が等しいケース)

群の数を k , 各群のデータ数を n , 全体の平均値を \bar{X} 第 j 群の平均値を \bar{X}_j とする .
 $(j = 1, 2, \dots, k; \sum n_j = n)$

$$[X] = \frac{\left(\sum_{j=1}^k \sum_{i=1}^{n_j} X_{ij} \right)^2}{nk}$$

$$[AS] = \sum_{j=1}^k \sum_{i=1}^{n_j} X_{ij}^2$$

$$[A] = \frac{\sum_{j=1}^k \left(\sum_{i=1}^{n_j} X_{ij} \right)^2}{n}$$

$$SS_M = [A] - [X]$$

$$SS_W = [AS] - [A]$$

$$SS_T = [AS] - [X]$$

分散分析表

変動因	平方和 : SS	自由度 : df	平均平方 : MS	F 値 : F
群間	SS_M	$df_M = k - 1$	$MS_M = \frac{SS_M}{df_M}$	$F = \frac{MS_M}{MS_W}$
群内	SS_W	$df_W = k(n - 1)$	$MS_W = \frac{SS_W}{df_W}$	----
全体	SS_T	$df_T = nk - 1$	$MS_T = \frac{SS_T}{df_T}$	----

p 値は , 分子の自由度 , df_M 分母の自由度 df_W の F 分布の累積分布関数より算出する .

2. 対応がない場合 (データ数が異なるケース)

群の数を k , 全データ数を n , 各群のデータ数を n_j , 全体の平均値を \bar{X} 第 j 群の平均値を \bar{X}_j とする . $(j = 1, 2, \dots, k; \sum n_j = n)$

$$[X] = \frac{\left(\sum_{j=1}^k \sum_{i=1}^{n_j} X_{ij} \right)^2}{n}$$

$$[AS] = \sum_{j=1}^k \sum_{i=1}^{n_j} X^2_{ij}$$

$$[A] = \frac{\sum_{j=1}^k \left(\sum_{i=1}^{n_j} X_{ij} \right)^2}{n_j}$$

$$SS_M = [A] - [X]$$

$$SS_W = [AS] - [A]$$

$$SS_T = [AS] - [X]$$

分散分析表

変動因	平方和 : SS	自由度 : df	平均平方 : MS	F 値 : F
群間	SS_M	$df_M = k - 1$	$MS_M = \frac{SS_M}{df_M}$	$F = \frac{MS_M}{MS_W}$
群内	SS_W	$df_W = n - k$	$MS_W = \frac{SS_W}{df_W}$	----
全体	SS_T	$df_T = n - 1$	$MS_T = \frac{SS_T}{df_T}$	----

3. 対応がある場合

群の数を k , 各群のデータ数を n , 全体の平均値を \bar{X} 第 j 群の平均値を \bar{X}_j とする .
 $(j = 1, 2, \dots, k; \sum n_j = n)$

$$[X] = \frac{\left(\sum_{j=1}^k \sum_{i=1}^{n_j} X_{ij} \right)^2}{nk}$$

$$[AS] = \sum_{j=1}^k \sum_{i=1}^{n_j} X^2_{ij}$$

$$[A] = \frac{\sum_{j=1}^k \left(\sum_{i=1}^{n_j} X_{ij} \right)^2}{n}$$

$$[S] = \frac{\sum_{i=1}^n \left(\sum_{j=1}^k X_{ij} \right)^2}{k}$$

$$SS_S = [S] - [X]$$

$$SS_M = [A] - [X]$$

$$SS_W = [AS] - [A]$$

$$SS_T = [AS] - [X]$$

分散分析表

変動因	平方和 : SS	自由度 : df	平均平方 : MS	F 値 : F
群間	SS_M	$df_M = k - 1$	$MS_M = \frac{SS_M}{df_M}$	$F_M = \frac{MS_M}{MS_W}$
被験者内	SS_S	$df_S = n - 1$	$MS_S = \frac{SS_S}{df_S}$	$F_M = \frac{MS_M}{MS_W}$
群内	SS_W	$df_W = (k - 1)(n - 1)$	$MS_W = \frac{SS_W}{df_W}$	----
全体	SS_T	$df_T = nk - 1$	$MS_T = \frac{SS_T}{df_T}$	----

参照

森 敏昭 , 吉田寿夫 (1990) : 『心理学のためのデータ解析テクニカルブック』 , pp86-94 , 北大路書房 .

anova2

機能

二元配置分散分析 (two way analysis of variance) を行い, 分散分析表を表示する. x の列の平均と行の平均が等しいという帰無仮説によって p の値を出力する.

形式

```
y = anova2( x , n , type )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : 標本数 (Series)
3. $type$: 対応あり / なし ("P" or "N")
 - a. "P" paired data
 - b. "N" non-paired data
4. y : 検定結果 (Series)
 - a. $y[0]$: 要因 1 の主効果の検定統計量, p 値
 - b. $y[1]$: 要因 2 の主効果の検定統計量, p 値
 - c. $y[2]$: 要因 1 \times 2 の交互作用の検定統計量, p 値

ANOVA 表: 6 つの列を表示

- 1 列: 変化量の要因
- 2 列: 各要因の平方和
- 3 列: 各要因の自由度
- 4 列: 各要因の平均平方
- 5 列: F 値
- 6 列: p 値

解説

要因	B_1		B_2	
A_1	X_{11}		X_{1k}	

	X_{21}			
\vdots	\vdots			
A_2	$X_{n_1 1}$		$X_{n_1 k}$	
要素数	n_1		n_k	n
平均	\bar{X}_1		\bar{X}_k	\bar{X}

1. 2 要因とも対応が無い場合 (データ数が等しいケース)

要因 A の群の数を p , 要因 B の群の数を q , 各群のデータ数を n , 全体の平均値を \bar{X} , 第 j 群の平均値を \bar{X}_j とする . ($j = 1, 2, \dots, k; \sum n_j = n$)

$$[ABS] = \sum_{i=1}^p \sum_{j=1}^q \sum_{k=1}^n X_{ijk}^2$$

$$[X] = \frac{\left(\sum_{i=1}^p \sum_{j=1}^q \sum_{k=1}^n X_{ijk} \right)^2}{npq}$$

$$[A] = \frac{\sum_{i=1}^p \left(\sum_{k=1}^n X_{ik} \right)^2}{nq}$$

$$[B] = \frac{\sum_{j=1}^q \left(\sum_{k=1}^n X_{jk} \right)^2}{np}$$

$$[AB] = \frac{\sum_{i=1}^p \sum_{j=1}^q \left(\sum_{k=1}^n X_{ijk} \right)^2}{n}$$

$$SS_A = [A] - [X]$$

$$SS_B = [B] - [X]$$

$$SS_{A \times B} = [AB] - [A] - [B] + [X]$$

$$SS_W = [ABS] - [AB]$$

$$SS_T = [ABS] - [X]$$

分散分析表

変動因	平方和 : SS	自由度 : df	平均平方 : MS	F 値 : F
A	SS_A	$df_A = p - 1$	$MS_A = \frac{SS_A}{df_A}$	$F_A = \frac{MS_A}{MS_R}$
B	SS_B	$df_B = q - 1$	$MS_B = \frac{SS_B}{df_B}$	$F_B = \frac{MS_B}{MS_R}$
A \times B	$SS_{A \times B}$	$df_{A \times B} = (p - 1)(q - 1)$	$MS_{A \times B} = \frac{SS_{A \times B}}{df_{A \times B}}$	$F_{A \times B} = \frac{MS_{A \times B}}{MS_R}$

誤差	SS_E	$df_E = pq(n-1)$	$MS_E = \frac{SS_E}{df_E}$	----
全体	SS_T	$df_T = pqn - 1$	----	----

2. 2 要因とも対応が無い場合 (データ数が異なるケース)

要因 A の群の数を p , 要因 B の群の数を q , 各群のデータ数を n_{ij} , 全体の平均値を \bar{X} , 各群の平均値を \bar{X}_{ij} とする . ($j = 1, 2, \dots, q; \sum n_j = n$)

$$[ABS] = \sum_{i=1}^p \sum_{j=1}^q \sum_{k=1}^{n_{ij}} X_{ijk}^2$$

$$[AB] = \frac{\sum_{i=1}^p \sum_{j=1}^q \left(\sum_{k=1}^{n_{ij}} X_{ijk} \right)^2}{n_{ij}}$$

$$[X] = \frac{\sum_{i=1}^p \sum_{j=1}^q \bar{X}_{ij}}{pq}$$

$$[A] = q \sum_{i=1}^p \bar{X}_i^2$$

$$[B] = p \sum_{j=1}^q \bar{X}_j^2$$

$$[AB] = \sum_{i=1}^p \sum_{j=1}^q \bar{X}_{ij}^2$$

$$\tilde{n} = \frac{pq}{\sum_{i=1}^p \sum_{j=1}^q \frac{1}{n_{ij}}}$$

$$SS_A = \tilde{n}([A] - [X])$$

$$SS_B = \tilde{n}([B] - [X])$$

$$SS_{A \times B} = \tilde{n}([AB] - [A] - [B] + [X])$$

$$SS_W = \tilde{n}([ABS] - [AB])$$

分散分析表

変動因	平方和 : SS	自由度 : df	平均平方 : MS	F 値 : F
A	SS_A	$df_A = p - 1$	$MS_A = \frac{SS_A}{df_A}$	$F_A = \frac{MS_A}{MS_R}$
B	SS_B	$df_B = q - 1$	$MS_B = \frac{SS_B}{df_B}$	$F_B = \frac{MS_B}{MS_R}$
A × B	$SS_{A \times B}$	$df_{A \times B} = (p-1)(q-1)$	$MS_{A \times B} = \frac{SS_{A \times B}}{df_{A \times B}}$	$F_{A \times B} = \frac{MS_{A \times B}}{MS_R}$

誤差	SS_E	$df_E = pq(n-1)$	$MS_E = \frac{SS_E}{df_E}$	----
----	--------	------------------	----------------------------	------

3. 2 要因とも対応がある場合

要因 A の群の数を p , 要因 B の群の数を q , 各群のデータ数を n , 全体の平均値を \bar{X} , 第 j 群の平均値を \bar{X}_j とする . ($j = 1, 2, \dots, k; \sum n_j = n$)

$$[ABS] = \sum_{i=1}^p \sum_{j=1}^q \sum_{k=1}^n X_{ijk}^2$$

$$[X] = \frac{\left(\sum_{i=1}^p \sum_{j=1}^q \sum_{k=1}^n X_{ijk} \right)^2}{npq}$$

$$[A] = \frac{\sum_{i=1}^p \left(\sum_{k=1}^n X_{ik} \right)^2}{nq}$$

$$[B] = \frac{\sum_{j=1}^q \left(\sum_{k=1}^n X_{jk} \right)^2}{np}$$

$$[AB] = \frac{\sum_{i=1}^p \sum_{j=1}^q \left(\sum_{k=1}^n X_{ijk} \right)^2}{n}$$

$$[AS] = \frac{\sum_{i=1}^p \sum_{k=1}^n (\bar{X}_{ik})^2}{n}$$

$$[S] = \frac{\sum_{k=1}^n (\bar{X}_k)^2}{pq}$$

$$[BS] = \frac{\sum_{k=1}^n \sum_{j=1}^q (X_{kj})^2}{p}$$

$$[SS] = [S] - [X]$$

$$SS_{A \times S} = [AS] - [A] - [S] + [X]$$

$$SS_{B \times S} = [BS] - [B] - [S] + [X]$$

$$SS_{A \times B \times S} = [ABS] - [AB] - [AS] - [BS] + [A] + [B] + [S] - [X]$$

分散分析表

変動因	平方和 : SS	自由度 : df	平均平方 : MS	F 値 : F
S	SS_S	$df_S = n - 1$	$MS_S = \frac{SS_S}{df_S}$	----
A	SS_A	$df_A = p - 1$	$MS_A = \frac{SS_A}{df_A}$	$F_A = \frac{MS_A}{MS_{A \times S}}$
A × S	$SS_{A \times S}$	$df_{A \times S} = (p - 1)(n - 1)$	$MS_{A \times S} = \frac{SS_{A \times S}}{df_{A \times S}}$	----
B	SS_B	$df_B = q - 1$	$MS_B = \frac{SS_B}{df_B}$	$F_B = \frac{MS_B}{MS_{B \times S}}$
B × S	$SS_{B \times S}$	$df_{B \times S} = (q - 1)(n - 1)$	$MS_{B \times S} = \frac{SS_{B \times S}}{df_{B \times S}}$	----
A × B	$SS_{A \times B}$	$df_{A \times B} = (p - 1)(q - 1)$	$MS_{A \times B} = \frac{SS_{A \times B}}{df_{A \times B}}$	$F_{A \times B} = \frac{MS_{A \times B}}{MS_{A \times B \times S}}$
A × B × S	$SS_{A \times B \times S}$	$df_{A \times B \times S} = (p - 1)(q - 1)(n - 1)$	$MS_{A \times B \times S} = \frac{SS_{A \times B \times S}}{df_{A \times B \times S}}$	----

p 値は、分子の自由度、 df_M 分母の自由度 df_W の F 分布の累積分布関数より算出する。

使用例

1. x : 標本 2 次元 Series オブジェクト、または、3 次元 Snapshot オブジェクト
2. y : 標本数 1 次元 Series オブジェクト

要因	A_1	A_2		A_n
B_1	7.0 5.7 6.6	4.6 5.6 6.6		7.3 6.3 7.8
B_2	5.8 8.9	10.0 7.1		6.5 9.8 8.0
:	:			
B_m	8.1 6.1 10.2 6.9	5.2 5.7 11.8 6.1		5.3 9.7 5.5 10.2 13.8 7.0

上記のデータを anova2(x, n) の引数として渡す場合

例 1)

```
series x[m][n]
```

```
x[0][0] = (7, 5.7)           A1 × B1
```

```
x[1][0] = (5.8, 8.9)        A1 × B2
```

```
:
```

```
x[m-1][0] = (8.1, 6.1, 10.2, 6.9)   A1 × Bm
```

```
x[0][1] = (4.6, 5.6, 6.6)           A2 × B1
```

```

x[1][1] = (10 , 7.1)           A2 × B2
:
x[m-1][1] = (5.2 , 5.7 , 11.8 , 6.1)   A2 × Bm
x[0][n-1] = (7.3 , 6.3 , 7.8)         An × B1
x[1][n-1] = (6.5 , 9.8 , 8)           An × B2
x[m-1][n-1] = (5.3 , 9.7 , 5.5 , 10.2 , 7) An × Bm

```

例 2)

snapshot X[k][m][n]

k はセルに対して最大の要素数 .

ここでは , (m , n) のセルの要素数が最大の場合 6 を指定 .

```

X[0][0][0] = 7           A1 × B1
X[1][0][0] = 5.7         A1 × B1
X[0][1][0] = 5.8         A1 × B2
X[1][1][0] = 8.9         A1 × B2
:
X[0][m-1][0] = 8.1       A1 × Bm
X[1][m-1][0] = 6.1       A1 × Bm
X[2][m-1][0] = 10.2      A1 × Bm
X[3][m-1][0] = 6.9       A1 × Bm
X[0][0][1] = 4.6         A2 × B1
X[1][0][1] = 5.6         A2 × B1
X[1][0][1] = 6.6         A2 × B1
X[0][1][1] = 10          A2 × B2
X[1][1][1] = 7.1         A2 × B2
:
X[0][m-1][1] = 5.2       A2 × Bm
X[1][m-1][1] = 5.7       A2 × Bm
X[2][m-1][1] = 11.8      A2 × Bm
X[3][m-1][1] = 6.1       A2 × Bm
:
X[0][m-1][n-1] = 5.3     An × Bm
X[1][m-1][n-1] = 9.7     An × Bm
X[2][m-1][n-1] = 5.5     An × Bm
X[3][m-1][n-1] = 10.2    An × Bm
X[4][m-1][n-1] = 13.8    An × Bm
X[5][m-1][n-1] = 7       An × Bm

```

標本数 n の指定

series N[n]

```

N[0] = ( 2 , 2 , ... , 4)   A1
N[1] = ( 3 , 2 , ... , 4)   A2
:
N[n-1] = ( 3 , 3 , ... , 6) An

```


参照

森 敏昭，吉田寿夫 (1990)：『心理学のためのデータ解析テクニカルブック』，pp64-121，北大路書房．

tukey

機能

分散分析表によって表される主効果や交互作用の検定とは別に，群間の差異を Tukey 法によって検定する．

形式

```
y = tukey( x , n )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : 標本数 (Series)
3. y : 検定結果 (Series)
 - a. y : [0] : 群 1 , 2 の検定統計量 , p 値
 - b. y : [1] : 群 1 , 3 の検定統計量 , p 値
 - c. y : [k*(k-1)] : 群 k-1 , k の検定統計量 , p 値

解説

x の形式は anova1 (x , n) と同じ

k : 群数

n_i : 群 i の観測値数

\bar{x}_i : 群 i の観測値の平均値

$w = \frac{1}{n_i} + \frac{1}{n_j}$ 群 i と群 j の比較

MS_w : 誤差分散 (群内不偏分散)

ν : 誤差の自由度

全ての 2 群の組み合わせについて ,

検定統計量

$$t_{ij} = \frac{\bar{x}_i - \bar{x}_j}{\sqrt{MS_w \left(\frac{1}{n_i} + \frac{1}{n_j} \right)}}$$

p 値は，群数 k ，自由度 v の Student 化した範囲の統計量より算出する．

参照

永田 靖，吉田道弘 (1997)：『統計的多重比較法の基礎』，pp35-40，サイエンティスト社．

scheffe

機能

分散分析表によって表される主効果や交互作用の検定とは別に，群間の差異を Scheffe 法によって検定する．

形式

```
y = scheffe( x , n )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : 標本数 (Series)
3. y : 検定結果 (Series)
 - a. y : [0] : 群 1 , 2 の検定統計量 , p 値
 - b. y : [1] : 群 1 , 3 の検定統計量 , p 値
 - c. y : [k*(k-1)] : 群 k-1 , k の検定統計量 , p 値

解説

x の形式は anova1 (x , n) と同じ

k : 群数

n_i : 群 i の観測値数

\bar{x}_i : 群 i の観測値の平均値

$w = \frac{1}{n_i} + \frac{1}{n_j}$ 群 i と群 j の比較

MS_w : 誤差分散 (群内不偏分散)

ν : 誤差の自由度

全ての 2 群の組み合わせについて ,
検定統計量

$$F = \frac{\left(\sum_j^k w_j \bar{x}_j \right)^2}{\frac{k-1}{MS_w \sum_j^k \frac{w_j^2}{n_j}}}$$

p 値は , 第 1 自由度 $df = k - 1$, 第 2 自由度 $df_w = n - k$ の F 分布より算出する .

参照

永田 靖 , 吉田道弘 (1997) : 『統計的多重比較法の基礎』 , pp52-58 , サイエンティスト社 .

bonferroni

機能

分散分析表によって表される主効果や交互作用の検定とは別に，群間の差異を Bonferroni 法によって検定する．

形式

```
y = bonferroni( x , n )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : 標本数 (Series)
3. y : 検定結果 (Series)
 - a. y : [0] : 群 1 , 2 の検定統計量 , p 値
 - b. y : [1] : 群 1 , 3 の検定統計量 , p 値
 - c. y : [k*(k-1)] : 群 k-1 , k の検定統計量 , p 値

解説

x の形式は anova1 (x , n) と同じ

k : 群数

n_i : 群 i の観測値数

\bar{x}_i : 群 i の観測値の平均値

$w = \frac{1}{n_i} + \frac{1}{n_j}$ 群 i と群 j の比較

MS_w : 誤差分散 (群内不偏分散)

ν : 誤差の自由度

全ての 2 群の組み合わせについて ,

検定統計量

$$t_{ij} = \frac{\bar{x}_i - \bar{x}_j}{\sqrt{MS_w \left(\frac{1}{n_i} + \frac{1}{n_j} \right)}}$$

p 値は，自由度 v の t 分布より算出するが，組み合わせの数を f とすると，有意水準は α/f となるため，各有意水準 $1/f$ とする代わりに， p 値を f 倍して表示する（但し， 1 を超える場合は 1 となる）。

参照

永田 靖，吉田道弘 (1997)：『統計的多重比較法の基礎』，pp81-87，サイエンティスト社。

dunnet

機能

分散分析表によって表される主効果や交互作用の検定とは別に，群間の差異を Dunnet 法によって検定する．

形式

```
y = dunnet( x , n )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : 標本数 (Series)
3. y : 検定結果 (Series)
 - a. y : [0] : 群 1 , 2 の検定統計量 , p = .05 の棄却限界値
 - b. y : [1] : 群 1 , 3 の検定統計量 , p = .05 の棄却限界値
 - c. y : [k*(k-1)] : 群 k-1 , k の検定統計量 , p = .05 の棄却限界値

解説

x の形式は anova1 (x , n) と同じ

k : 群数

n_i : 群 i の観測値数

\bar{x}_i : 群 i の観測値の平均値

$w = \frac{1}{n_i} + \frac{1}{n_j}$ 群 i と群 j の比較

MS_w : 誤差分散 (群内不偏分散)

ν : 誤差の自由度

データ内で最後の群 k を対照群とし，対照群とその他の群の組み合わせについて，
検定統計量

$$t_{ik} = \frac{\bar{x}_i - \bar{x}_k}{\sqrt{MS_w \left(\frac{1}{n_i} + \frac{1}{n_k} \right)}}$$

群数 k , 自由度 v の Dunnet の数表より棄却限界値 d を求める .

$d \leq t_{ik}$ のとき , $p = .05$ で有意に差があるといえる .

参照

永田 靖 , 吉田道弘 (1997) : 『統計的多重比較法の基礎』 , pp40-45 , サイエンティスト社 .

pearson

機能

入力データ x, y の直線的関係について Pearson の積率相関係数を算出し、母相関係数 $\rho = 0$ について検定 (Pearson's product moment correlation coefficient test) を行う。

形式

```
z = pearson( x, y, type )
```

パラメータ

1. x, y : 入力データ (Series , Snapshot)
2. $type$: 片側 / 両側検定 (String)
 - a. "T" : two-sided
 - b. "O" : one-sided
3. z : 検定結果 (Series)
 - a. $z[0]$: 検定統計量
 - b. $z[1]$: p 値

解説

相関係数 r を算出する。(corrcoef 関数を参照)

標本数を n とすると、

検定統計量

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}}$$

自由度 $df = n - 2$

p 値は、自由度 df の t 分布の累積分布関数より算出する。

参照

ピアソンの積率相関係数の検定

<http://aoki2.si.gunma-u.ac.jp/lecture/Corr/corr.html>

mregression

機能

多項式重回帰分析 (Pearson's product moment correlation coefficient test) を行う。

形式

```
z = mregression( x , y , m , type )
```

パラメータ

1. x : 説明変数 (Series , Snapshot)
2. y : 目的変数 (Series , Snapshot)
3. m : 最大次数 (Scalar)
4. type : 片側 / 両側検定 ("T" or "O")
 - a. "T" : two-sided
 - b. "O" : one-sided
5. z : 検定結果 (Series)
 - a. z : [0] : 検定統計量
 - b. z : [1] : p 値

出力結果

分散分析表
 重相関係数
 寄与率
 自由度調節済み寄与率
 回帰係数
 標準化回帰係数
 偏相関係数
 標準誤差
 t 値
 p 値

解説

データに対して, m 次の最小 2 乗多項式

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_mx^m$$

を求める.

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^m \\ 1 & x_2 & x_2^2 & \cdots & x_2^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^m \end{bmatrix} \quad X^t = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^m & x_2^m & \cdots & x_n^m \end{bmatrix} \quad A = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

とすると, 正規方程式

$$X^t X A = X^t Y$$

$$A = (X^t X)^{-1} X^t Y$$

より回帰係数を求める.

1. 分散分析表

m 個の項からなる多項式のとき, 目的変数を $y_i (i = 1, 2, \dots, n)$, その予測値を Y_i それぞれの平均を \bar{y}, \bar{Y} とすると,

変動 因	平方和 : SS	自由度 : df	平均平方 : MS	F 値 : F	p 値
回帰	$SS_R = \sum_{i=1}^n (Y_i - \bar{Y})^2$	$df_R = m$	$MS_R = \frac{SS_R}{df_R}$	$F = \frac{MS_R}{MS_E}$	1
残差	$SS_E = \sum_{i=1}^n (y_i - Y_i)^2$	$df_E = n - m - 1$	$MS_E = \frac{SS_E}{df_E}$	----	----
全体	$SS_T = \sum_{i=1}^n (y_i - \bar{y})^2$	$df_T = n - 1$	$MS_T = \frac{SS_T}{df_T}$	----	----

1. 第 1 自由度 df_R , 第 2 自由度 df_E に従う F 分布の F 値の上側確率.

2. 寄与率

- 重相関係数 ($|R|$): 目的変数とその予測値の相関係数 $R = \sqrt{(R^2)}$
- 寄与率 (R^2): 重相関係数の 2 乗値. 目的変数とその多項式のセットで何 % 説明できるかを表す.

$$R^2 = 1 - \frac{SS_E}{SS_T}$$

- 自由度調節済み寄与率: 寄与率は多項式の数が多くなるほど大きくなるため, それを調節した値.

$$R^{2*} = 1 - \frac{MS_E}{MS_T}$$

3. 回帰係数 (1 ~ m 次まで)

a. 回帰係数

多項式回帰モデルの定数項と，それぞれの多項式に対する回帰係数．

b. 標準化回帰係数

回帰係数を b_i , S_{ij} を偏差平方和積和行列の要素とすると，

$$b_i' = b_i * \sqrt{\frac{S_{ii}}{S_{yy}}}$$

c. 偏相関係数

相関行列の逆行列の要素を R_{ij} とすると，

$$d. PCC_i = -\frac{R_{yi}}{\sqrt{R_{yy} * R_{jj}}}$$

e. 標準誤差

回帰係数を b_i ，標準誤差を $\sigma(b_i)$, S_{ij} を偏差平方和積和行列の逆行列の要素とすると，
定数項：

$$\sigma(b_0) = \sqrt{\left(\frac{1}{n} + \sum \sum \bar{x}_i * \bar{x}_j * S_{ij}\right) * MS_E}$$

その他の項：

$$\sigma(b_i) = \sqrt{S_{ii} * MS_E}$$

f. t 値

$$t_i = \frac{|b_i|}{\sigma(b_i)}$$

p 値

t_i は，自由度 $n-m-1$ の t 分布に従う．(上側確率)

参照

1. 多項式回帰分析

<http://aoki2.si.gunma-u.ac.jp/lecture/Regression/preg/preg.html>

2. 2001 年度版 基礎数学ワークブック

<http://www.ele.kochi-tech.ac.jp/inoue/work/2001/ban/02/info.html>

princomp

機能

主成分分析 (principal component analysis) を行う .

形式

```
y = princomp( x )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. y : 主成分値 (Snapshot)

コマンドへの出力結果

相関係数行列
固有値・固有ベクトル
係数ベクトル・定数項
因子付加量
寄与率・累積寄与率

解説

n 個の個体に対して , m 種類の特性の測定値がある .

$x^{(1)}, x^{(2)}, \dots, x^{(m)}$

$$\begin{matrix} 1 \\ 2 \\ \vdots \\ n \end{matrix} X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

x_1, x_2, \dots, x_m の一次式

$$z_1 = a_{11}x^{(1)} + a_{12}x^{(2)} + \cdots + a_{1m}x^{(m)} + a_{10}$$

\vdots

$$z_m = a_{m1}x^{(1)} + a_{m2}x^{(2)} + \cdots + a_{mm}x^{(m)} + a_{m0}$$

を次の条件を満たすように求める .

1. 各 z_i の相関係数は全て 0 である .
2. $Var(z_1) \geq Var(z_2) \geq \cdots \geq Var(z_m)$ が最大になる .

- 固有値・固有ベクトル

m 個の変数の相関係数行列 A より , パワー法により算出する .

1. 固有ベクトルの初期値を $U_1 = (1 \ \cdots \ 0 \ 0)^t$ とする .
2. A と U_1 の積を算出し , 得られたベクトルのノルム (要素の二乗和) を算出する .
3. ノルムの平方根を算出する . これが固有値の近似値となる .
4. 得られたベクトルの要素をノルムの平方根で割ったものを U_2 とする . このベクトルが固有ベクトルの近似値となる .
5. A と U_2 の積を求める . 以下 , 2 ~ 4 を固有ベクトルの各要素の変化が $1E-6$ 以下になるまで繰り返す .
6. A の各要素に対して , 以下の演算を行った行列 (残差行列) を求め , これを新たな A とする .

A_{ij} - 固有値 $\times U_i \times U_j \rightarrow A_{ij}$ (U_i, U_j は固有ベクトルの i, j 要素)

$n \times n$ 行列の固有値・固有ベクトルは n 対得られるので , n 番目まで上述の手続きを繰り返す .

算出した相関係数行列の固有値を $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$ とし , λ_h に対する固有ベクトルを u_h とする .

- 係数ベクトル

$$a_{hi} = \frac{u_{hi}}{s_i}$$

- 定数項

$$a_{h0} = - \sum a_{hi} \cdot \bar{x}_i$$

- 因子付加量

$$r(z_h, x^{(i)}) = \sqrt{\lambda_h} \cdot u_{hi} \quad (i = 1, 2, \cdots, m)$$

- 寄与率

$$p_h = \frac{\lambda_h}{m}$$

- 累積寄与率

$$P_h = P_{h-1} + p_h$$

参照

芳賀敏郎，橋本茂司 (1980)：統計解析プログラム講座 2 『回帰分析と主成分分析』，pp170-184，日科技連出版社．

ansaribradley

機能

2 つのデータ x, y について, それぞれの母集団の分布の広がりには差があるかどうか Ansari-Bradley 検定を行う.

形式

```
z = ansaribradley( x, y )
```

パラメータ

1. x, y : 入力データ (Series, Snapshot)
2. z : 検定結果 (Series)
 - a. $z[0]$: 検定統計量
 - b. $z[1]$: p 値

解説

両群の標本数 n_1, n_2 ($n_1 \leq n_2$) とする.

両群のデータを合わせて大きい順・小さい順に順位 r_{1i}, r_{2j} を付ける. 但し, 同じ値には, それらの値が異なると考えた場合の順位の平均値を付ける. データが少ないほうの一群 (n_1) について順位の総和を取り, 検定統計量 A とする.

1. $n_1 < 10, n_2 < 10$ の場合,

Ansari-Bradley の数表を参照する.

$$A \leq \bar{a}_{n_1, n_2} \text{ または } A \leq \underline{a}_{n_1, n_2}$$

2. $n_1 \geq 10, n_2 \geq 10$ の場合,

- a. $n_1 + n_2$ が偶数のとき,

検定統計量 A は正規分布

$$N\left(\frac{n_1(n_1 + n_2 + 2)}{4}, \frac{n_1 n_2 (n_1 + n_2 - 2)(n_1 + n_2 + 2)}{48(n_1 + n_2 - 1)}\right)$$

で近似されるため,

$$Z_0 = \frac{\left| A - \frac{n_1(n_1+n_2+2)}{4} \right| - \frac{1}{2}}{\sqrt{\frac{n_1 n_2 (n_1+n_2-2)(n_1+n_2+2)}{48(n_1+n_2-1)}}}$$

について , $p = Pr\{|Z| \geq Z_0\}$ を算出する .

b. $n_1 + n_2$ が奇数のとき ,

$$Z_0 = \frac{\left| A - \frac{n_1(n_1+n_2+2)^2}{4(n_1+n_2)} \right| - \frac{1}{2}}{\sqrt{\frac{n_1 n_2 (n_1+n_2-2)((n_1+n_2)^2+3)}{48(n_1+n_2)^2}}}$$

について , $p = Pr\{|Z| \geq Z_0\}$ を算出する .

参照

石村貞夫 (1990) : 『統計解析のはなし』 , pp265-271 , 東京図書 .

lepage

機能

2 つのデータ x, y について, それぞれの母集団の分布の位置のずれや散らばり具合に差があるかどうか Lepage の検定を行う.

形式

```
z = lepage( x, y )
```

パラメータ

1. x, y : 入力データ (Series, Snapshot)
2. z : 検定結果 (Series)
 - a. $z[0]$: 検定統計量
 - b. $z[1]$: p 値

解説

両群のサンプル数 n_1, n_2 ($n_1 \leq n_2$) とする.

このデータについて, Wilcoxon 検定統計量 W と Ansari-Bradley 検定統計量 A を算出し, 2 つの値より Lepage の検定統計量

$$L = \frac{(W - E(W))^2}{Var(W)} + \frac{(A - E(A))^2}{Var(A)}$$

但し,

$$E(W) = \frac{n_1(n_1 + n_2 + 1)}{2}$$

$$E(A) = \frac{n_1(n_1 + n_2 + 2)}{4}$$

$$Var(W) = \frac{n_1 n_2 (n_1 + n_2 + 2)}{12}$$

$$Var(A) = \frac{n_1 n_2 (n_1 + n_2 - 2)(n_1 + n_2 + 2)}{48(n_1 + n_2 - 1)}$$

を算出する.

$n_1 \geq 10, n_2 \geq 10$ のとき，検定統計量 L は自由度 2 の χ^2 分布に従うため， χ^2 分布より p 値を算出する．

参照

石村貞夫 (1990)：『統計解析のはなし』，pp272-276，東京図書．

steeldwass

機能

複数群の入力データ x について，正規性を仮定しない多重比較 (Steel-Dwass の方法) を行う．

形式

```
y = steeldwass( x , n )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : x の各群の標本数 (Series)
3. y : 検定結果 (Series)
 - a. $y[0]$: 群 1 , 2 の検定統計量 p 値
 - b. $y[1]$: 群 1 , 3 の検定統計量 p 値
 - c. $y[k*(k-1)]$: 群 $k-1$, k の検定統計量 p 値

解説

すべての群に対して i と j (ただし, $i > j$) の組み合わせを作る．

第 i 群と第 j 群を併せて順位をつけ，第 i 群の第 k 番目のデータの順位を r_{ik} とし，第 i 群の順位和 $R_{ij} = r_{i1} + r_{i2} + \cdots + r_{in_i}$ とする．

両群の標本数 n_i, n_j ($N_{ij} = n_i + n_j$) とする．

帰無仮説の下での期待値 $E(R_{ij})$ と分散 $V(R_{ij})$ を計算する．

$$E(R_{ij}) = \frac{n_i(N_{ij} + 1)}{2}$$

$$V(R_{ij}) = \frac{n_i n_j}{N_{ij}(N_{ij} - 1)} \left\{ \sum_{k=1}^{n_i} r_{ik}^2 + \sum_{k=1}^{n_j} r_{jk}^2 - \frac{N_{ij}(N_{ij} + 1)^2}{4} \right\}$$

検定統計量 t_{ij} を計算する．

$$t_{ij} = \frac{R_{ij} - E(R_{ij})}{\sqrt{V(R_{ij})}}$$

自由度 の Student 化された範囲の分布より p 値を算出する .

参照

永田靖 , 吉田道弘 (1997) : 『統計的多重比較法の基礎』 , pp67-70 , サイエンティスト社 .

steel

機能

複数群の入力データ x について，等分散性を仮定しない多重比較 (Steel の方法) を行う．

形式

```
y = steel( x , n )
```

パラメータ

1. x : 入力データ (Series , Snapshot)
2. n : x の各群の標本数 (Series)
3. y : 検定結果 (Series)
 - a. $y[0]$: 群 1 , 2 の検定統計量 $p=.05$ の棄却限界値
 - b. $y[1]$: 群 1 , 3 の検定統計量 $p=.05$ の棄却限界値
 - c. $y[k*(k-1)]$: 群 $k-1$, k の検定統計量 $p=.05$ の棄却限界値

解説

データ内で最後の群を対照群，その他を処理群とする．

対照群と第 i 群を併せて順位をつけ，対照群の第 k 番目のデータの順位を r_{ik} とし，対照群の順位和 $R_{1i} = r_{11} + r_{12} + \cdots + r_{1n_i}$ とする．

両群の標本数 n_1, n_i ($N_{1i} = n_1 + n_i$) とする．

帰無仮説の下での期待値 $E(R_{ik})$ と分散 $V(R_{ik})$ を計算する．

$$E(R_{1i}) = \frac{n_1(N_{1i} + 1)}{2}$$

$$V(R_{1i}) = \frac{n_1 n_i}{N_{1i}(N_{1i} - 1)} \left\{ \sum_{k=1}^{n_1} r_{1k}^2 + \sum_{k=1}^{n_i} r_{ik}^2 - \frac{N_{1i}(N_{1i} + 1)^2}{4} \right\}$$

検定統計量 t_{1i} を計算する．

$$t_{1i} = \frac{R_{1i} - E(R_{1i})}{\sqrt{V(R_{1i})}}$$

群数 k , 自由度 ν の Dunnett の数表より棄却限界値 d を求める .

$d \leq t_{ik}$ のとき , $p=.05$ で有意に差があるといえる .

参照

永田靖 , 吉田道弘 (1997) : 『統計的多重比較法の基礎』 , pp70-74 , サイエンティスト社 .