汎用最小二乗推定C関数ライブラリ (liblsq) 説明書

近藤哲朗

2005年6月19日版

目 次

1　概要

1991 年に FORTRAN 用に作成したサブルーチン群を C 言語に移植した。最小二乗解の計算は特異値分解法によっている。ランク落ちの起きたパラメータに対しても最小二乗最小ノルム解（答えの善し悪しは別としてももっともらしい解）が得られる。また次元あり重みを採用しているので異なる次元（単位）の観測量も同時に扱える。なお、特異値分解プログラムの出典は「計算機のための数値計算法」G.E.Forsythe/M.A.Malcolm/C.B.Moler 著、森正武訳、科学技術出版社 (1978)。

2　プログラム作成法

```
#include "liblsq.h"          … 最小二乗推定関数群のヘッダーファイル

/* test main */
int main(int argc, char *argv[])
{
    double x,xerr,y,yerr,anm;
    double xa[10];
    int n,m,nobs,mpara,omax,pmax;
    lsq_type_t lsq;                          …· 最小二乗推定で利用する変数
     ····
     ····
    omax=100; pmax=10;                       …· 観測数、パラメータ数の最大値設定
    lsqinit(&lsq,omax,pmax);                 …· liblsq 群の初期化
     ····
    lsqdebug(&lsq);                          …· デバグ出力指定
    for(n=1; n<=nobs; n++){
         ····
        lsqsetobs(&lsq,y,yerr,n);            …· 観測データをセット
        for(m=1; m<=mpara; m++){
             ····
            lsqsetjacob(&lsq,anm,n,m);       …· ヤコビアンをセット
        }
    }
     ····
    lsqsetsize(&lsq,nobs,mpara);             …· dolsq にヤコビアンのサイズを教える
    dolsq(&lsq);                             …· 最小二乗推定の実行
    lsqgetkai(&lsq,&x,&xerr,m);              …· パラメータａを推定
     ····
    xa[m]+=x              …· 真の解はアプリオリ値 xa にアジャスト値 x を加えたもの

     ····
    lsqfinish(&lsq);                         …· メモリの解放。最後に必ず呼ぶこと
}
```

3 最小二乗推定ライブラリ (liblsq) 構造体の定義および関数

3.1 lsq_type_t 構造体

定義

double *r_y	観測データ（O－Cデータ）収納配列（へのポインタ）
double *r_yerr	観測誤差データ収納配列（へのポインタ）
int *i_good	観測データフラグ配列（へのポインタ） 1：良好データ　0：悪データ（推定に使用しない）
double *d_a	オリジナル偏微分係数行列収納配列（へのポインタ）
double *r_omc	最小二乗推定後のO－C収納配列（へのポインタ）
double *x_adj	最小二乗推定パラメータ値（アジャスト値）収納配列（へのポインタ）
int *i_flag	推定パラメータフラグ収納配列（へのポインタ） 1：推定する　0：推定しない 推定後-1 となったパラメータはランク落ちの発生したパラメータ
double *d_corx	得られたパラメータ間の相関係数行列収納配列（へのポインタ）
double *d_covx	得られたパラメータ間の共分散行列収納配列（へのポインタ）
double r_sum2	推定後の残差二乗和 χ^2 収納用変数 今の場合 $\chi^2 = S(\hat{x}) = \tilde{v}(\hat{x})\Sigma^{-1}v(\hat{x})$ ただし $v(\hat{x}) = y - f(\hat{x})$ ：残差 Σ^{-1} ：推定パラメータの誤差行列 (D_COVX) の逆行列（スケーリング前）
double r_sigma	推定後の残差の標準偏差（σ_*）収納用変数 $\sigma_*^2 = S(\hat{x})/(n-m)$ ただし $n-m$ ：自由度
double r_chi2	規格化 $\chi^2 = \chi^2/$ 自由度 $(n-m) = \sigma_*^2$
int n_free	推定後の自由度
int m_pmax	最大可能パラメータ数（配列の確保に使用）
int m_omax	最大可能データ数（配列の確保に使用）
int n_data	実際のデータ数（bad データも含む）
int n_obs	Good データ数
int n_para	推定パラメータ数
int m_para	実際に推定されたパラメータ数（ランク落ちパラメータも含む）
int i_dbk	処理デバッグ情報出力の制御　1: ON 0: OFF
int ierr	関数からのエラーコード

3.2 lsqinit

機能

lsq_type_t 型変数の初期化および動的メモリの確保を行う。観測フラグはすべて０、パラメータフラグはすべて１に初期化される。この関数は必ず最初に呼んでおくこと

関数プロトタイプ

void lsqinit(lsq_type_t *lt, int maxobs, int maxpara);

引数

lt	入出力	lsq_type_t 型変数
maxobs	入力	確保する最大データ数
maxpara	入力	確保する最大パラーメータ数

戻り値

なし

3.3 lsqfinish

機能

確保した動的メモリ領域の解放。この関数はプログラム終了前に必ず呼ぶこと

関数プロトタイプ

void lsqfinish(lsq_type_t *lt);

引数

lt	lsq_type_t 型変数

戻り値

なし

3.4 lsqdebug

機能

デバグ情報表示フラグのセット

関数プロトタイプ

void lsqdebug(lsq_type_t *lt);

引数

lt	入出力	lsq_type_t 型変数

戻り値

なし

3.5 lsqgetmax

機能

使用可能な最大データ数および最大パラメータ数を得る

関数プロトタイプ

void lsqgetmax(lsq_type_t *lt, int *maxobs, int *maxpara);

引数

lt	入力	lsq_type_t 型変数
maxobs	出力	最大可能データ数
maxpara	出力	最大可能パラメータ数

戻り値

なし

3.6 lsqsetsize

機能

ヤコビアン行列のサイズ（実データ数および実パラメータ数）を最小二乗推定関数 dolsq に教える。dolsq に先立って必ずこの関数を呼ぶこと

関数プロトタイプ

void lsqsetsize(lsq_type_t *lt, int nobs, int npara);

引数

lt	入出力	lsq_type_t 型変数
nobs	入力	実データ数
npara	入力	実パラメータ数

戻り値

なし

3.7 lsqgetsize

機能

ヤコビアン行列のサイズ（実データ数および実パラメータ数）を得る

関数プロトタイプ

void lsqgetsize(lsq_type_t *lt, int *nobs, int *npara);

引数

lt	入力	lsq_type_t 型変数
nobs	出力	実データ数
npara	出力	実パラメータ数

戻り値

なし

3.8 lsqsetflag

機能

推定パラメータフラグのセット、リセットや観測データフラグのセット、リセットを行う

関数プロトタイプ

void lsqsetflag(lsq_type_t *lt, char *key, int k, char *aflag);

引数

lt	入出力	lsq_type_t 型変数
key	入力	セットするフラグの指定コード ”PARAM”：パラメータフラグ指定 ”OBS”：観測データフラグ指定
k	入力	パラメータ番号または観測番号　（いずれも 1 から始まる番号）
aflag	入力	セットするフラグの中身 パラメータフラグ指定の場合 ”YES”：推定する　”NO”：推定しない 観測データフラグ指定の場合 ”GOOD”：Good データとする　”BAD”：Bad データとする

戻り値

なし

3.9 lsqgetflag

機能

推定パラメータフラグおよび観測データフラグの値を得る

関数プロトタイプ

void lsqgetflag(lsq_type_t *lt, char *key, int k, int *iflag);

引数

lt　　入力　lsq_type_t 型変数
key　入力　値を得るフラグの指定コード
　　”PARAM”：パラメータフラグ指定
　　”OBS”：観測データフラグ指定
k　　入力　パラメータ番号または観測番号　（いずれも１から始まる番号）
iflag　出力　フラグの中身
　　パラメータフラグ指定の場合
　　1：推定する　0：推定しない
　　-1：推定パラメータとして指定したがランク落ち発生
　　観測データフラグ指定の場合
　　1：Good データ　”0”：Bad データ

戻り値

なし

3.10　lsqpflag

機能

推定パラメータフラグをセットする

関数プロトタイプ

void lsqpflag(lsq_type_t *lt, int m);

引数

lt　入出力　lsq_type_t 型変数
m　入力　推定フラグをセットするパラメータ番号（１から始まる番号）

戻り値

なし

3.11　lsqpflagr

機能

推定パラメータフラグをリセットする

関数プロトタイプ

void lsqpflagr(lsq_type_t *lt, int m);

引数

lt　入出力　lsq_type_t 型変数
m　入力　推定フラグをリセットするパラメータ番号（１から始まる番号）

戻り値

なし

3.12　lsqgetpflag

機能

推定パラメータフラグ値を得る

関数プロトタイプ

void lsqgetpflag(lsq_type_t *lt, int m, int *iflag);

引数

lt	入力	lsq_type_t 型変数
m	入力	推定フラグ値を得るパラメータ番号（1から始まる番号）
iflag	出力	推定フラグ値
		1：推定する　0：推定しない
		-1：推定パラメータとして指定したがランク落ち発生

戻り値

なし

3.13 lsqoflag

機能

観測データフラグをセット (Good データ）する

関数プロトタイプ

void lsqoflag(lsq_type_t *lt, int n);

引数

lt	入出力	lsq_type_t 型変数
n	入力	観測データフラグをセットする観測番号（1から始まる番号）

戻り値

なし

3.14 lsqoflagr

機能

観測データフラグをリセット (Bad データ扱い）する

関数プロトタイプ

void lsqoflagr(lsq_type_t *lt, int n);

引数

lt	入出力	lsq_type_t 型変数
n	入力	観測データフラグをリセットする観測番号（1から始まる番号）

戻り値

なし

3.15 lsqgetoflag

機能

観測データフラグ値を得る

関数プロトタイプ

void lsqgetoflag(lsq_type_t *lt, int n, int *igood);

引数

lt	入力	lsq_type_t 型変数
n	入力	観測データフラグ値を得る観測番号（1から始まる番号）
igood	出力	推定フラグ値
		1：Good データ推定する　0：Bad データ

戻り値

なし

3.16 lsqsetjacob

機能

N 番目の観測のパラメータ番号 M のヤコビアン（偏微分係数）A をセットする

関数プロトタイプ

void lsqsetjacob(lsq_type_t *lt, double a, int n, int m);

引数

lt	入出力	lsq_type_t 型変数
a	入力	セットするヤコビアン値
n	入力	ヤコビアンをセットする観測番号（１から始まる番号）
m	入力	ヤコビアンをセットするパラメータ番号（１から始まる番号）
lt.ierr	出力	エラーコード 0：正常終了 -1：観測番号が最大可能データ数を超えている -2：パラメータ番号が最大可能パラメータ数を超えている

戻り値

なし

3.17 lsqgetjacob

機能

N 番目の観測のパラメータ番号 M のヤコビアン（偏微分係数）A を得る

関数プロトタイプ

void lsqgetjacob(lsq_type_t *lt, double *a, int n, int m);

引数

lt	入出力	lsq_type_t 型変数
a	出力	ヤコビアン値
n	入力	ヤコビアンを得る観測番号（１から始まる番号）
m	入力	ヤコビアンを得るパラメータ番号（１から始まる番号）
lt.ierr	出力	エラーコード 0：正常終了 -1：観測番号が最大可能データ数を超えている -2：パラメータ番号が最大可能パラメータ数を超えている

戻り値

なし

3.18 lsqsetobs

機能

N 番目の観測データ (Y) および誤差 (YERR) をセットする。Y は通常 O-C をセットする。この関数で良好なデータを示すフラグもセットされる。悪データ（推定に使用しないデータ）としたいときは lsqsetflag または lsqoflagr を使用してフラグを変更する。

関数プロトタイプ

void lsqsetobs(lsq_type_t *lt, double y, double yerr, int n);

引数

lt	入出力	lsq_type_t 型変数
y	入力	観測データ
yerr	入力	観測誤差
n	入力	データをセットする観測番号（1から始まる番号）

戻り値

なし

3.19 lsqgetobs

機能

N 番目の観測データ (Y) および誤差 (YERR) を得る

関数プロトタイプ

void lsqsetobs(lsq_type_t *lt, double *y, double *yerr, int n);

引数

lt	入力	lsq_type_t 型変数
y	出力	観測データ
yerr	出力	観測誤差
n	入力	データを得る観測番号（1から始まる番号）

戻り値

なし

3.20 dolsq

機能

最小二乗推定を実施する

関数プロトタイプ

void dolsq(lsq_type_t *lt);

引数

lt	入出力	lsq_type_t 型変数
lt.ierr	出力	エラーコード 0：正常終了 -1：推定でエラー発生

戻り値

なし

3.21 lsqgetkai

機能

最小二乗推定で得たパラメータ番号 M の推定値（アジャスト値）X と 1 σ 誤差 XERR を得る。この誤差はすでに σ_* が乗じられている値（いわゆる scaled σ）。従ってそのまま推定誤差として用いることができる

関数プロトタイプ

void lsqgetkai(lsq_type_t *lt, double *x, double *xerr, int m);

引数

lt	入力	lsq_type_t 型変数
x	出力	パラメータの推定値（アジャスト値）
xerr	出力	推定誤差
n	入力	推定値を得るパラメータ番号（1 から始まる番号）

戻り値

なし

3.22 lsqgetcov

機能

推定パラメータ M1 と M2 間の共分散値 V を得る。共分散値の推定値とするにはスケーリング定数 σ_*^2（=lt.r_sigma^2) を乗じること

関数プロトタイプ

void lsqgetcov(lsq_type_t *lt, double *v, int m1, int m2);

引数

lt	入力	lsq_type_t 型変数
v	出力	共分散値
m1	入力	パラメータ番号 1（1 から始まる番号）
m2	入力	パラメータ番号 2（1 から始まる番号）

戻り値

なし

3.23 lsqgetcor

機能

推定パラメータ M1 と M2 間の相関係数 C を得る

関数プロトタイプ

void lsqgetcor(lsq_type_t *lt, double *c, int m1, int m2);

引数

lt	入力	lsq_type_t 型変数
c	出力	相関係数
m1	入力	パラメータ番号 1（1 から始まる番号）
m2	入力	パラメータ番号 2（1 から始まる番号）

戻り値

なし

3.24 lsqgetstd

機能

最小二乗推定後の各種統計量を得る

関数プロトタイプ

void lsqgetstd(lsq_type_t *lt, double *rsum2, double *sigma, double *rchi2, int *nfree, int *ndata, int *mpara);

引数

lt	入力	lsq_type_t 型変数
rsum2	出力	規格化残差二乗和（χ^2）
		今の場合 $\chi^2 = S(\hat{x}) = \tilde{v}(\hat{x})\Sigma^{-1}v(\hat{x})$
		ただし $v(\hat{x}) = y - f(\hat{x})$ ：残差
		Σ^{-1} ：推定パラメータの誤差行列 (D_COVX) の逆行列（スケーリング前）
sigma	出力	推定後の規格化残差の標準偏差（σ_*）
		$\sigma_*^2 = S(\hat{x})/(n-m)$
		ただし $n-m$ ：自由度
rchi2	出力	規格化 $\chi^2 = \chi^2/$ 自由度 $(n-m) = \sigma_*^2$
nfree	出力	自由度（＝データ数−パラメータ数）
ndata	出力	実際に推定に用いられたデータ数
mpara	出力	実際に推定されたパラメータ数（ランク落ちパラメータも含む）

戻り値

なし

3.25 lsqgetomc

機能

最小二乗推定後の観測番号 N の残差 R を得る

関数プロトタイプ

void lsqgetomc(lsq_type_t *lt, double *r, int n);

引数

lt	入力	lsq_type_t 型変数
r	出力	最小二乗推定後の観測番号 N の残差
n	入力	残差を得る観測番号（１から始まる番号）

戻り値

なし

3.26 dwsvd

機能

特異値分解の実施（出典:「計算機のための数値計算法」G.E.Forsythe/M.A.Malcolm/C.B.Moler 著、森正武訳、科学技術出版社 (1978)）

関数プロトタイプ

void dwsvd(int nm, int m, int nn, int n, double *a, double *w, int matu, double *u, int matv, double *v, int *kerr, double *rv1);

引数

nm	入力	行列 A,U,V の行サイズ A(NM,NN),U(NM,NN),V(NM,NN)
m	入力	行列 A および U の実際の行サイズ (m<nm)
nn	入力	行列 A,U,V の列サイズ
n	入力	行列 A,U および V の実列サイズ (n<nn)
a	入力	特異値分解を行う行列
w	出力	特異値を収納 W(NN)
matu	入力	1: U 行列分解を行う　0: 行わない
u	出力	U 行列
matv	入力	1: V 行列分解を行う　0: 行わない
v	出力	V 行列
kerr	出力	エラーコード 0：正常終了 k：k 番目の特異値が３０回のイタレーションでも決定できない
rv1	入力	サイズ nn の作業用配列 RV1(NN)

戻り値

なし