C プログラムから SSL II を利用するには

渡部 善隆 [*]　　山元 規靖 [†]

C 言語は UNIX とともに普及した (もともとは) システム記述言語でしたが、最近では数値計算に利用する人も増えてきました。しかし、 C 用に作られた汎用の数値計算ライブラリは、 Fortran に比べてまだまだ少ないのが現状です。

SSL II(Scientific Subroutine Library II) は、 Fortran 用に作成された科学技術計算用の汎用サブルーチンライブラリです。 UXP 上の SSL II には、スカラー版とベクトル版があり、ベクトル計算機向けにチューニングされた SSL II/VP の中には、理論的なピーク性能に迫る優秀なライブラリもあります。

SSL II は、 Fortran プログラムから呼び出す (CALL する) のが普通です。しかし、 Fortran と C を静的に結合して一つのプログラムを作る一般的なテクニックを用いることで、 C プログラムの中から SSL II を呼び出すことができます。

本稿では、汎用計算機 M-1800/20U およびベクトル計算機 VP2600/10 の UXP システムと、ライブラリ・サーバー S-4/1000E 上で C プログラムから SSL II を呼び出す方法を紹介します。

なお、一般的な Fortran プログラムと C プログラムとの結合方法については [7] の第 8 章に詳しい解説がありますので参照下さい。

1　環境

1.1　センターでサービスしている C 言語

ANSI 規格 (X3.159-1989) 準拠の C 言語です。 C の利用できる環境は以下の通りです (1996 年 7 月現在)。

計算機	VP2600/10	M-1800/20U	S-4/1000E
OS	UXP/M V12	UXP/M V12	SunOS 5.4
ホスト名	kyu-vpux	kyu-cc	wisdom
IP アドレス	バッチ処理のみ	133.5.9.1	133.5.9.9
浮動小数点形式	IBM(M) 形式	IBM(M) 形式	IEEE 形式
コマンド	`vcc`	`vcc`, `cc`	`cc`

詳しくは [1]–[6] を御覧下さい。コマンドは `man` コマンドで検索できます。

なお、来年 1 月に予定されているスーパーコンピュータの入れ換えにともない、 VP2600/10 は新スーパーコンピュータに置き換えられる予定です。従って、利用環境に変化が生じる可能性がありますのでご注意下さい。

変更点はセンターニュース・広報で随時お知らせします。

[*]九州大学大型計算機センター・研究開発部　E-mail : watanabe@cc.kyushu-u.ac.jp
[†]九州大学大型計算機センター・研究開発部　E-mail : yamamoto@cc.kyushu-u.ac.jp

1.2 SSL II の機能

1.2.1 SSL II とは

科学用サブルーチンライブラリ SSL II は、線型方程式や微分方程式などの数学的問題を解く、約 230 種類のサブルーチンからなる汎用数値計算ライブラリです。各サブルーチンは Fortran で記述されており、通常は Fortran プログラムから CALL 文で使用します。

SSL II は標準機能と拡張機能より構成されています。標準機能は広範囲な科学技術計算を支援することを目的としたもので、拡張機能は大規模な科学技術計算をベクトル計算機上で高速で処理することを狙ったものです。

1.2.2 標準機能の構成

標準機能は、以下の分野から構成されています。

線型計算	:	連立 1 次方程式，逆行列，最小二乗解，特異値分解
固有値問題	:	固有値・固有ベクトル
非線型方程式	:	代数方程式，超越方程式，連立非線型方程式
極値問題	:	関数の極小化，線型計画問題，非線型計画問題
補間・近似	:	補間式 / 補間値, 近似式, 平滑化式 / 平滑値, 級数展開
変換	:	フーリエ変換，ラプラス変換
数値微積分	:	離散点 / 関数入力, 有限区間 / 無限領域,1 次元 /2 次元
微分方程式	:	連立 1 階常微分方程式, 連立 1 階スティフ常微分方程式
特殊関数	:	ベッセル関数, 楕円積分, 指数積分, 正弦・余弦積分
擬似乱数	:	乱数生成 (一様 / 正規 / 指数 / ポアソン / 二項), 乱数検定

1.2.3 拡張機能の構成

拡張機能は、以下の分野から構成され、ベクトル計算機専用のアルゴリズムが用いられています。

線型計算	:	実行列の積，逆行列，連立 1 次方程式 (実行列, 3 項行列, 正値対称行列)
固有値問題	:	固有値・固有ベクトル
変換	:	フーリエ変換 (実フーリエ変換, 複素フーリエ変換)

マニュアル [8], [9] は、富士通株式会社提供の手引書としては出色の出来です。詳しい機能やアルゴリズムを知りたい方は是非一度お読み下さい。センター 2 階のプログラム相談室および 4 階の図書室で参照出来ます。

また、機能と引数だけを調べたい場合は、オンラインマニュアルで検索できます。M-1800/20U の UXP(ホスト名 kyu-cc) および S-4/1000E(ホスト名 wisdom) の man コマンドで検索できます。

```
kyu-cc% man ssl2  ↵              <--- 機能一覧を見る
kyu-cc% man subroutine-name  ↵   <--- 各ライブラリの引数を調べる
```

“*subroutine-name*” は、検索したいサブルーチンの名前です。残念ながら、SSL II の日本語のオンラインマニュアルは全角英数字を多用しているうえにレイアウトも上手とはいえませんので、英語のマニュアルの方が読みやすかったりします。日本語と英語との切替えは以下のようにします。

```
kyu-cc% unsetenv LANG  ↵        <--- 英語へ表示を切替える
kyu-cc% man ssl2  ↵             <--- オンライン検索
          :
kyu-cc% setenv LANG japan  ↵    <--- 日本語表示に戻す
```

1.3 Fortran を知らない方へ

オンラインマニュアルを見ても意味がわからなかった方は以下を参考にしてみて下さい。

整合寸法とは何ですか？

SSL II のサブルーチンで 2 次元配列を扱う場合は、配列、次数だけでなく「整合寸法」を入力する必要があります。配列の整合寸法 (adjustable dimension of array) とは、 Fortran のメインプログラムで配列を用意する時に宣言する行列の 行数 のことです。

ただし、 C と Fortran では 2 次元以上の配列要素の記憶順序が異なるので (3 章参照)、引数で渡す整合寸法の値は注意が必要です。

倍精度 / 単精度の切替えはどうするのですか？

SSL II では、一部を除いて単精度と倍精度用のルーチンが利用できます。倍精度演算用ルーチン名は、先頭に “D” を付加します。例えば離散型 Fourier 変換のサブルーチンは、単精度が VRFT1 , 倍精度が DVRFT1 となります[1]。

パラメータの型がよくわかりません...

パラメータとして与えるサブルーチンの引数の型は次の規則に従っています。

- I, J, K, L, N, M で始まるものは 4 バイト長の整数です[2]。
- Z で始まるものは複素数です。 C には複素数型に対応する型が存在しませんが、構造体として表現することが可能です。
- それ以外で始まるものは実数型です (2 章参照)。単精度ルーチンの場合は 4 バイトの実数型、倍精度ルーチンの場合は 8 バイトの実数型として宣言します。

それから

どうしてこんなに引数がたくさんあるの！

それぞれ理由があって、無駄なものは何一つありません。精度と安定性と速度とデバッグのために必要なものばかりですので、文句をいわないで下さい。

2 属性

SSL II で使用するデータは、 4 バイト整数型、実数型、倍精度実数型、複素数型、倍精度複素数型の 5 つです[3]。 C プログラムから SSL II を呼び出すときは、 SSL II の引数と属性を一致させる必要があります。対応関係は次の通りです。

[1] 大規模な数値計算をする方には、倍精度ルーチンを使用することをお勧めします。
[2] 悪名 (?) 高い Fortran の「暗黙の型宣言」です。バグの温床となり、 Fortran プログラマを苦しめます。
[3] 4 倍精度実数 / 複素数型も Fortran から C へ受け渡しできますが、この記事の範囲外なので説明を省略します。

型	Fortran	C
4 バイト整数	INTEGER*4 INTEGER	long int int
単精度実数	REAL*4 REAL	float
倍精度実数	REAL*8 DOUBLE PRECISION	double
単精度複素数	COMPLEX COMPLEX*8	struct { float r,i; }
倍精度複素数	COMPLEX*16	struct { double r,i; }

センターの C では long int および int は 4 バイトです。複素数型は構造体で表現して SSL II に渡します。

3　2 次元配列の記憶順序

C と Fortran では 2 次元配列の記憶順序が異なります。 SSL II の使いたいサブルーチンで 2 次元配列を使用するときは、 Fortran の記憶順序に従った 配列を用意して下さい。

3.1　C の記憶順序

例として、 $m \times n$ 行列 $\hat{A}$ を倍精度実数で考えることにします。まず、 C で "double a[m][n];" と宣言すると、 $\hat{A}$ を表現する 2 次元配列 a が

$$a = \begin{pmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,n-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m-1,0} & a_{m-1,1} & \cdots & a_{m-1,n-1} \end{pmatrix}$$ C の行列イメージ

として (頭の中に) できます。ここでは各要素 a[i][j] が $a_{i,j}$ に対応しています。

計算機の中では、このような 2 次元的な記憶ではなくて、 $m \times n$ 個の 8 バイトデータが何らかの規則で順番に格納されています。問題は記憶の順序です。

C の記憶の順序は、まず行列 a の 1 行目を $a_{0,0}, a_{0,1}, a_{0,2}, \cdots, a_{0,n-1}$ といった順番で「列方向」に格納し、次に 2 行目に移り $a_{1,0}, a_{1,1}, a_{1,2}, \cdots, a_{1,n-1}$ と格納していきます。

$$a = \begin{pmatrix} \overset{\longrightarrow}{a_{0,0}} & a_{0,1} & \cdots & a_{0,n-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m-1,0} & a_{m-1,1} & \cdots & a_{m-1,n-1} \end{pmatrix}$$ C の記憶順序

従って、行列 a =a[m][n] の記憶順序は左から順番に

$$\underbrace{a_{0,0}\ a_{0,1}\ a_{0,2} \cdots\ a_{0,n-1}}_{1\text{ 行目}}\ \underbrace{a_{1,0}\ a_{1,1}\ a_{1,2} \cdots\ a_{1,n-1}}_{2\text{ 行目}}\ \cdots\ \underbrace{a_{m-1,0}\ a_{m-1,1}\ a_{m-1,2} \cdots\ a_{m-1,n-1}}_{m\text{ 行目}}$$

となります。

3.2 Fortran の記憶順序

一方、 Fortran で $m \times n$ 行列 $\hat{A}$ を倍精度実数で考える場合、 “DOUBLE PRECISION A(M,N)” または “REAL*8 A(M,N)” と宣言します。

すると、 $\hat{A}$ を表現する 2 次元配列 A が

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \cdots & A_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m,1} & A_{m,2} & \cdots & A_{m,n} \end{pmatrix}$$ Fortran の行列イメージ

として (頭の中で) 出来上がります。ここでは各要素 A(i,j) が $A_{i,j}$ に対応しています。

すぐに気づかれる通り、添字の動きが C と Fortran では違います。すなわち、 C では a[0][0] から a[m-1][n-1] まで動くのに対し、 Fortran は a[1][1] から a[m][n] まで動きます。この添字の始点・終点の違いはプログラムを移植する際には重要ですが、 SSL II を利用する際には気にすることはありません。

問題は記憶順序です。 Fortran の記憶の順序は、まず行列 A の 1 列目を $A_{1,1}, A_{2,1}, A_{3,1}, \cdots, A_{m,1}$ といった順番で「行方向」に格納し、次に 2 列目に移り $A_{1,2}, A_{2,2}, A_{3,2}, \cdots, A_{m,2}$ と格納していきます。

$$A = \begin{pmatrix} A_{1,1} & \boxed{A_{1,2}} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m,1} & \boxed{A_{m,2}} & \cdots & A_{m,n} \end{pmatrix} \downarrow$$ Fortran の記憶順序

従って A = A(M,N) の記憶順序は、左から順番に

$$\underbrace{A_{1,1}\ A_{2,1}\ A_{3,1} \cdots\ A_{m,1}}_{1\text{ 列目}} \underbrace{A_{1,2}\ A_{2,2}\ A_{3,2} \cdots\ A_{m,2}}_{2\text{ 列目}} \cdots \underbrace{A_{1,n}\ A_{2,n}\ A_{3,n} \cdots\ A_{m,n}}_{n\text{ 列目}}$$

となります。

3.3 SSL II に渡す 2 次元配列

C プログラムの中で SSL II に渡す (C で宣言する)2 次元配列は、 Fortran の記憶順序に従って渡します。

C と Fortran の 2 次元配列の記憶順序の違いは、列方向に格納するか行方向に格納するかです。従って、すぐにわかるように、 C で宣言した行列を 転置 してしまえば、転置行列は Fortran の記憶順序に従うことになります。

例えば、 C で宣言した a[m][n] を転置した行列 b=b[n][m] を作ってみます。b[i,j]=a[j,i] ですので、 $b = a^T$ のイメージは

$$b = \begin{pmatrix} a_{0,0} & a_{1,0} & \cdots & a_{m-1,0} \\ a_{0,1} & a_{1,1} & \cdots & a_{m-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{0,n-1} & a_{0,n-1} & \cdots & a_{m-1,n-1} \end{pmatrix}$$ $b = a^T$ の行列イメージ

となります。

従って、行列 b =b[n][m] の記憶順序は左から順番に

$$\underbrace{a_{0,0}\ a_{1,0}\ a_{2,0}\cdots\ a_{m-1,0}}_{1\text{ 行目}}\underbrace{a_{0,1}\ a_{1,1}\ a_{2,1}\cdots\ a_{m-1,1}}_{2\text{ 行目}}\cdots\underbrace{a_{0,n-1}\ a_{1,n-1}\ a_{2,n-1}\cdots\ a_{m-1,n-1}}_{n\text{ 行目}}$$

となり、添字をひとつ増やして考えれば、 Fortran の記憶順序と一致します。
つまり、 SSL II に渡す 2 次元配列の作成方法として、以下の方法が考えられます。

1. $m \times n$ 行列 a とは別に $n \times m$ 行列 b を用意し、 a を転置した値を b に代入し、 b を SSL II に渡す。ただし整合寸法は m。
2. 始めから C プログラム内で行と列が逆転したイメージでプログラムを組む。正方行列 $(n = m)$ の場合は、添字の入れ換えだけでそれほど苦労せずに作れるはず。
3. 考える行列が対称行列ならば何もしなくていい。

3.4 閑話

行列の転置とは、 $a_{i,j}$ の添字を入れ換えて $a_{j,i}$ とすることです。イメージとしては、右に 90 度回転した後、縦を軸に「くるっ」とひっくり返すと転置行列ができます。色の 2 次元配列としてのイメージデータを例にとると、こうなります。

元の行列

右に 90 度回転

反転

4 呼出

C プログラムから SSL II を呼び出す規則は次の通りです。

1. C プログラムの名前は UXP では MAIN__、 S-4/1000E では MAIN_ とする。
2. サブルーチン名は小文字で記述し、最後に “_” を付加する (例：dvlax_, dvmggm_)
3. extern 文で SSL II のサブルーチン名を宣言する。
4. 各パラメータはポインタで渡す。

具体的な例をあげます。

```
#include <stdio.h>
#include <math.h>
#define MAXN 2001

MAIN__()
{
     extern void dvlax_();
     static double a[MAXN][MAXN],f[MAXN],vw[MAXN],pi,eps;
     static int    n,i,j,is,ip[MAXN],icon,maxl,isw;

     n=2000;
     eps=0.;
     isw=1;
     maxl=MAXN;

       pi=4.*atan(1.);
       for(i=0; i<n; i++)
          for(j=0; j<n; j++)
             a[i][j] = sqrt(2./(n+1))*sin((i+1)*(j+1)*Pi/(n+1));

       for(i=0; i<n; i++)
          f[i] = (double) (i+1);

       dvlax_(a,&maxl,&n,f,&eps,&isw,&is,vw,ip,&icon);

       for(i=0; i<n; i++)
          printf("%5d %20.14f\n",i,f[i]);
   }
```

この C プログラムは、連立 1 次方程式 DVLAX を使用したプログラムです。プログラムでは 2 次元配列 a は対称行列ですので何も問題ありませんが、行列が非対称の場合は 3 章の記憶順序に配慮したプログラミングが要求されます。また、定数 maxn はそのままでは SSL II に渡せませんのでint maxl に値を代入しています。

複素数変数を扱う場合は、構造体で定義します。

```
#include <stdio.h>
MAIN__()
{
  static struct { double r,i; } z[2];
  static double a0,a1,a2;
  extern void drqdr_();
  int icon;

      a0 =  3.;
      a1 = -1.;
      a2 =  4.;

      drqdr_(&a0, &a1, &a2, z, &icon);

      printf("%20.14f %20.14f\n",z[0].r,z[0].i);
      printf("%20.14f %20.14f\n",z[1].r,z[1].i);
    }
```

これは実係数 2 次方程式の根を求めるサブルーチン DRQDR を呼び出した例です。根となる複素数 z の定義に注意して下さい。 DRQDR の使用方法は DVLAX と同じです。

5 実行

5.1 M-1800/20U の場合

汎用機 M-1800/20U で対話的に実行する場合は、vcc コマンドで翻訳し、オブジェクトファイルと SSL II を frt コマンドで -lssl2vp オプションを付加して結合させます。 C プログラムは test.c とします。

```
kyu-cc% vcc -c test.c  ↵
kyu-cc% frt -J test.o -lssl2vp -lcvp -lm  ↵
kyu-cc% a.out  ↵
```

vcc はベクトル演算用の C/VP を起動するコマンドです。frt コマンドの -J はベクトル用のライブラリを使用することを意味します。また、-lcvp -lm は C のライブラリの参照です。

スカラー演算の場合は、cc コマンドで翻訳し、オブジェクトファイルと SSL II を frt コマンドで -lssl2 オプションを付加して結合させます。

```
kyu-cc% cc -c test.c  ↵
kyu-cc% frt test.o -lssl2 -lm  ↵
```

5.2 VP2600/10 の場合

ベクトル計算機 VP2600/10 で実行する場合はバッチ処理となります。バッチジョブの投入方法は [10] の 3 章の 3 節以降と全く同じですので、そちらを御覧下さい。

```
#
cd EXAMPLE
vcc -c test.c
frt -J test.o -lssl2vp -lcvp -lm
a.out
```

5.3 S-4/1000E の場合

ライブラリ・サーバー S-4/1000E で実行する場合は以下のようにします。

```
wisdom% cc -c test.c  ↵
wisdom% frt test.o -lfssl2  ↵
wisdom% a.out  ↵
```

5.4 注意

-l で結合するライブラリ名は計算機と Fortran 言語によって異なりますので、今後変更になることが十分予想されます。また、全ての SSL II のサブルーチンで動作確認はしていませんので、「おかしいなぁ」と思われた時は、センターまで御連絡願います。

参考文献

[1] C 言語文法書, 99SP-8051, 富士通株式会社 (1992).

[2] UXP/M C 言語使用手引書, 94SP-5051, 富士通株式会社 (1992).

[3] UXP/M C/VP 使用手引書, 99SP-5070, 富士通株式会社 (1992).

[4] UXP/M C 言語メッセージ説明書, 94SP-5060, 富士通株式会社 (1991).

[5] UXP/M ANSI C 移行手引書, 93SP-1140, 富士通株式会社 (1991).

[6] UXP/M ANSI C プログラミング支援ツール使用手引書, 93SP-1041, 富士通株式会社 (1994).

[7] UXP/M FORTRAN77 EX 使用手引書 V12 用, 94SP–5010, 富士通株式会社 (1991).

[8] SSL II 使用手引書 (科学用サブルーチンライブラリ), 99SP–4020, 富士通株式会社 (1987).

[9] SSL II 拡張機能使用手引書 (科学用サブルーチンライブラリ), 99SP–4070, 富士通株式会社 (1991) .

[10] 渡部 善隆 : UXP/Fortran 利用法, 九州大学大型計算機センター広報, Vol.29, No.2, pp.93–149 (1996).