気象観測データの解析

１　AMeDAS データの解析

研究を進めるにあたって、データ解析用のプログラムを自分で作成する必要が生じることがあります。ここでは、自分で FORTRAN または C でプログラムを作成し、CD-ROM に入った気象観測データ（気象庁による AMeDAS の観測データ）を読みこんで解析します。データを読みこむためのサブルーチンや関数はあらかじめ作成してあります。それらのサブルーチンや関数を使って自分でプログラムを書いてデータを解析していきます。まず、サンプルプログラムを実行してみます。具体的には、

```
/home/snaoki> f77 sample.f ameadas_sub.f
```

とします。sample.f がプログラムの本体で、その中で使われているデータの読みこみ用のサブルーチンは ameadas_sub.f に書かれています。C の場合は、

```
/home/snaoki> cc sample.c ameadas_sub.c -lm -lf2c
```

とします。ameadas_sub.c は FORTRAN で作成したプログラムを自動で変換して作成したため、-lf2c というオプションをつけて専用のライブラリを参照します。コンパイルが成功すると実行ファイルが生成されているはずです。生成されていることを確認したら実行します。地点番号と年月日を聞かれるので、入力してください。府中の地点番号は 44116 です。

```
/home/snaoki> ./a.out
 Station No. ?
44116
 Year, Month, Day ?
2009,1,2
 Reading data ...
 Output was written to file 'output.txt'.
```

結果は output.txt というファイルに書き出されるので、cat などのコマンドで中身を確認してください。

```
/home/snaoki> cat output.txt
 Station No. = 44116,
 Date = Jan.  2, 2009
 Time Precip. Dir. Speed   S.D. Temp.
   hr      mm        m/s     hr     C
    1       0  C       0    0.0   1.2
    2       0  W       1    0.0   0.4
    3       0  NW      1    0.0  -0.5
```

〈中略〉

```
  24        0  NNE      2     0.0    2.9
```

次に、指定した日時の時間降水量と気温を、観測地点の一覧表に載っているすべての地点について出力するプログラムを作成してみます。観測地点の一覧表は index.txt というファイルで与えられています。このファイルを読みこむためのサブルーチンが AMDID です。AMeDAS のデータを読みこむためのサブルーチンは AMDHR です。サブルーチン AMDHR は、IYY、IMM で指定した年、月について、index.txt で指定したすべての観測地点のデータを読みこみます。サブルーチン AMDID と AMDHR は、amedas_sub.f の中に書いてあるので、コンパイルのときに、このファイル名を指定すれば、自分でサブルーチンを作る必要はありません。

プログラム：

FORTRAN

```
C  配列の大きさ: IHXは1日の時間数(=24)、IDXは1月の日数(=31)、
C                 NXは地点数である。
C                 NXには、観測地点リストindex.txtに載っている
C                 観測地点数と等しい値を指定する。
      PARAMETER (IHX=24,IDX=31,NX=146)

C  配列(入力データ)の宣言:
C    ISTA (NX): 観測地点番号。東京(大手町)は44132など。
C    ITYPE (IDX,NX): 観測地点の種類。
C                    降水量のみの場合は1、4要素の場合は4。
C                    データが存在しない場合は0。
C    XX (NX): 観測地点の経度 [度]。
C    YY (NX): 観測地点の緯度 [度]。
C    IR (IHX,IDX,NX): 降水量 [時間]。
C    ID (IHX,IDX,NX): 風向 [16方位、北北東=1、東=4]。
C    IW (IHX,IDX,NX): 風速 [m/s]。
C    S (IHX,IDX,NX): 日照時間 [時間]。
C    T (IHX,IDX,NX): 気温 [℃]。
      INTEGER ISTA(NX),ITYPE(IDX,NX),
     +        IR(IHX,IDX,NX),ID(IHX,IDX,NX),IW(IHX,IDX,NX)
      REAL XX(NX),YY(NX),HH(NX),
     +     S(IHX,IDX,NX),T(IHX,IDX,NX)

C  欠損値: データが無効のとき、IMISS、RMISSで指定された値が格納される。
C          整数型の場合はIMISS、実数型の場合はRMISSである。
      PARAMETER (IMISS=999,RMISS=999.9)

C  サブルーチンAMDIDによって、観測地点リストを読みこむ。
```

```
      CALL AMDID (NX,ISTA,XX,YY,HH)

C  年月日、時刻を指定する。
C  入力を求めるメッセージを標準出力に書き出す。
      WRITE(6,*) 'Year, Month, Day, Hour ?'

C  READ文で入力値を読みこむ。
      READ (5,*) IYY,IMM,IDD,IHH

C  サブルーチンAMDHRによって、
C  指定された年、月の観測データ(時別値)を読むこむ。
C  ※サブルーチンAMDHRは、IYY、IMMで指定された年、月について
C    データを読みこむ。
      CALL AMDHR
     +  (IHX,IDX,NX,ISTA,IYY,IMM,IMISS,RMISS,ITYPE,IR,ID,IW,S,T)

C  出力ファイルを開く。
C  機番は10以降の番号を指定する。
C  STATUSは書き出しの場合は'UNKNOWN'を指定する。
C  FORMはテキストファイルの場合は'FORMATTED'を指定する。
      OPEN(10,FILE='output.txt',STATUS='UNKNOWN',
     +     FORM='FORMATTED')

C  ここからDOループ11が始まる。
C  各地点について、結果を出力する。
      DO 11 N=1,NX

C  データが存在する場合だけ、つまり、ITYPE(ID,N)がゼロでない場合だけ、
C  結果を出力する。
      IF (ITYPE(IDD,N).NE.0) THEN

C  地点番号、緯度、経度、降水量、気温を出力する。
        WRITE(10,'(1X,I5,1X,F8.3,1X,F8.3,1X,I3,1X,F5.1)')
     +    ISTA(N),YY(N),XX(N),IR(IHH,IDD,N),T(IHH,IDD,N)

      ENDIF

C  ここでDOループ11が終了する。
   11 CONTINUE
```

```
C  ファイルを閉じる。
      CLOSE(10)

      STOP
      END
```

（参考）C

```
#include <stdio.h>

/*========  関数プロトタイプ  ========*/

/*  プログラムの中で別に作成した関数を参照するときは、
    関数プロトタイプを作成する。  */
int amdhr( int ihmax, int idmax, int nmax, int *ista, int iyy, int imm,
           int imiss, float rmiss,
           int *itype, int *ir, int *id, int *iw,
           float *s, float *t );
int amdid( int nmax, int *ista, float *xx, float *yy, float *hh );

/*========  main 関数  ========*/

/*  プログラムは main 関数から実行される。  */
int main( void )
{

/*  配列の大きさ: ihmax は 1 日の時間数(=24)、idmax は 1 月の日数(=31)、
                  nmax は地点数である。
                  nmax には、観測地点リスト index.txt に載っている
                  観測地点数と等しい値を指定する。  */
  int ihmax=24, idmax=31, nmax=146;

/*  配列(入力データ)の宣言:
      ista [nmax]: 観測地点番号。東京(大手町)は 44132 など。
      itype [idmax, nmax]: 観測地点の種類。
                           降水量のみの場合は 1、4 要素の場合は 4。
                           データが存在しない場合は 0。
      xx [nmax]: 観測地点の経度 [度]。
      yy [nmax]: 観測地点の緯度 [度]。
      ir [ihmax, idmax, nmax]: 降水量 [時間]。
      id [ihmax, idmax, nmax]: 風向 [16 方位、北北東=1、東=4]。
```

```
      iw [ihmax, idmax, nmax]: 風速 [m/s]。
      s [ihmax, idmax, nmax]: 日照時間 [時間]。
      t [ihmax, idmax, nmax]: 気温 [℃]。
      配列のサイズが大きいので、静的変数(static)として宣言する。
      static では、関数が呼ばれるごとに新たにメモリを確保するのではなく、
      プログラムが終了するまで固定されたメモリを用いる。  */
  static int ista[146], itype[31*146],
    id[24*31*146], ir[24*31*146], iw[24*31*146];
  static float xx[146], yy[146], hh[146],
    s[24*31*146], t[24*31*146];

  int ihh, idd, imm, iyy, n, status;
  FILE *fp;

/*  欠損値: データが無効のとき、imiss、rmiss で指定された値が格納される。
            整数型の場合は imiss、浮動小数点型の場合は rmiss である。  */
  int imiss = 999;
  float rmiss = 999.9;

/*  サブルーチン amdid によって、観測地点リストを読みこむ。  */
  status = amdid (nmax, ista, xx, yy, hh);

/*  年月日、時刻を指定する。
    入力を求めるメッセージを標準出力に書き出す。  */
  printf( "%s¥n", "Year, Month, Day, Hour ?" );

/*  scanf 文で入力値を読みこむ。  */
  scanf( "%d,%d,%d,%d", &iyy, &imm, &idd, &ihh );

/*  関数 amdhr によって、指定された年、月の観測データ(時別値)を読むこむ。
    ※関数 amdhr は、iyy、imm で指定された年、月について
      データを読みこむ。  */
  status = amdhr (ihmax, idmax, nmax, ista, iyy, imm, imiss, rmiss,
                  itype, ir, id, iw, s, t);

/*  出力ファイルを開く。
    モードは"w"(テキストファイルへの書き出し)を指定する。  */
  fp = fopen( "output.txt", "w" );

/*  ここから for ループが始まる。
```

```
    各地点について、結果を出力する。  */
  for (n=1; n<=nmax; n++)
  {

/*  データが存在する場合だけ、
    つまり、itype[(idd-1)+(n-1)*idmax]がゼロでない場合だけ、
    結果を出力する。*/
    if (itype[(idd-1)+(n-1)*idmax] != 0){

/*  地点番号、緯度、経度、降水量、気温を出力する。  */
      fprintf( fp, " %5d %8.3f %8.3f %3d %5.1f¥n",
               ista[n-1], yy[n-1], xx[n-1],
               ir[(ihh-1)+(idd-1)*ihmax+(n-1)*idmax*ihmax],
               t[(ihh-1)+(idd-1)*ihmax+(n-1)*idmax*ihmax] );

    }

/*  ここで for ループが終了する。  */
  }

/*  ファイルを閉じる。  */
  fclose( fp );

  return 0;

}
```

実行例：

```
/home/snaoki> f77 prog01_1.f amedas_sub.f
/home/snaoki> ./a.out
 Year, Month, Day, Hour ?
2009,1,2,6
/home/snaoki> cat output.txt
 40041    36.867  140.640    0 999.9
 40046    36.840  140.775    0   3.5
 40061    36.775  140.350    0  -4.4
<中略>
 46211    35.175  139.633    0   1.7
```

※C の場合は、コンパイル時に-lm 、-lf2c オプションを指定する必要があります。

さらに、指定した年月日における各地点の日降水量を計算するプログラムを作成します。与えられたデータは時別値なので、自分でプログラムを書いて 24 時間分の降水量を合計します。

プログラム：

FORTRAN

```
C  配列の大きさ: IHXは1日の時間数(=24)、IDXは1月の日数(=31)、
C                 NXは地点数である。
C                 NXには、観測地点リストindex.txtに載っている
C                 観測地点数と等しい値を指定する。
      PARAMETER (IHX=24,IDX=31,NX=146)

C  配列(入力データ)の宣言:
C    ISTA (NX): 観測地点番号。東京(大手町)は44132など。
C    ITYPE (IDX,NX): 観測地点の種類。
C                    降水量のみの場合は1、4要素の場合は4。
C                    データが存在しない場合は0。
C    XX (NX): 観測地点の経度 [度]。
C    YY (NX): 観測地点の緯度 [度]。
C    IR (IHX,IDX,NX): 降水量 [時間]。
C    ID (IHX,IDX,NX): 風向 [16方位、北北東=1、東=4]。
C    IW (IHX,IDX,NX): 風速 [m/s]。
C    S (IHX,IDX,NX): 日照時間 [時間]。
C    T (IHX,IDX,NX): 気温 [℃]。
      INTEGER ISTA(NX),ITYPE(IDX,NX),
     +        IR(IHX,IDX,NX),ID(IHX,IDX,NX),IW(IHX,IDX,NX)
      REAL XX(NX),YY(NX),HH(NX),
     +     S(IHX,IDX,NX),T(IHX,IDX,NX)

C  配列(出力データ)の宣言:
C    IRD (NX): 指定した日の日降水量 [mm]。
      INTEGER IRD(NX)

C  欠損値: データが無効のとき、IMISS、RMISSで指定された値が格納される。
C          整数型の場合はIMISS、実数型の場合はRMISSである。
      PARAMETER (IMISS=999,RMISS=999.9)

C  サブルーチンAMDIDによって、観測地点リストを読みこむ。
      CALL AMDID (NX,ISTA,XX,YY,HH)

C  年月日を指定する。
```

```
C  入力を求めるメッセージを標準出力に書き出す。
      WRITE(6,*) 'Year, Month, Day ?'

C  READ文で入力値を読みこむ。
      READ (5,*) IYY,IMM,IDD

C  サブルーチンAMDHRによって、
C  指定された年、月の観測データ(時別値)を読むこむ。
C  ※サブルーチンAMDHRは、IYY、IMMで指定された年、月について
C    データを読みこむ。
      CALL AMDHR
     +  (IHX,IDX,NX,ISTA,IYY,IMM,IMISS,RMISS,ITYPE,IR,ID,IW,S,T)

C  ここからDOループ11が始まる。
C  各地点について、日降水量を計算する。
      DO 11 N=1,NX

C  出力データを格納する配列に、あらかじめ欠損値を入れておく。
        IRD(N) = IMISS

C  データが存在する場合だけ、つまり、ITYPE(ID,N)がゼロでない場合だけ、
C  計算を実行する。
      IF (ITYPE(IDD,N).NE.0) THEN

C  和の値にゼロを代入する。
        ISUM = 0

C  ここからDOループ21が始まる。
C  各地点について、1時から24時までの時間降水量を合計する。
      DO 21 IHH=1,IHX

C  欠損値を見つけたときはその地点の計算を中止する。
C  GO TO文でDOループから出る。
      IF (IR(IHH,IDD,N).EQ.IMISS) GO TO 29

        ISUM = ISUM + IR(IHH,IDD,N)

C  ここでDOループ21が終了する。
 21   CONTINUE
```

```
C  合計値をIRDに代入する。
         IRD(N) = ISUM

 29    CONTINUE

       ENDIF

C  ここでDOループ11が終了する。
 11    CONTINUE

C  出力ファイルを開く。
C  機番は10以降の番号を指定する。
C  STATUSは書き出しの場合は'UNKNOWN'を指定する。
C  FORMはテキストファイルの場合は'FORMATTED'を指定する。
       OPEN(10,FILE='output.txt',STATUS='UNKNOWN',
      +      FORM='FORMATTED')

C  ここからDOループ31が始まる。
C  各地点について、結果を出力する。
       DO 31 N=1,NX

C  データが存在する場合だけ、つまり、IRD(N)が欠損値でない場合だけ、
C  結果を出力する。
       IF (IRD(N).NE.IMISS) THEN

C  地点番号、緯度、経度、降水量を出力する。
         WRITE(10,'(1X,I5,1X,F8.3,1X,F8.3,1X,I3)')
      +     ISTA(N),YY(N),XX(N),IRD(N)

       ENDIF

C  ここでDOループ31が終了する。
 31    CONTINUE

C  ファイルを閉じる。
       CLOSE(10)

       STOP
       END
```

（参考）C

```
#include <stdio.h>

/*========  関数プロトタイプ  ========*/

/*  プログラムの中で別に作成した関数を参照するときは、
    関数プロトタイプを作成する。  */
int amdhr( int ihmax, int idmax, int nmax, int *ista, int iyy, int imm,
           int imiss, float rmiss,
           int *itype, int *ir, int *id, int *iw,
           float *s, float *t );
int amdid( int nmax, int *ista, float *xx, float *yy, float *hh );

/*========  main 関数  ========*/

/*  プログラムは main 関数から実行される。  */
int main( void )
{

/*  配列の大きさ: ihmax は 1 日の時間数(=24)、idmax は 1 月の日数(=31)、
                  nmax は地点数である。
                  nmax には、観測地点リスト index.txt に載っている
                  観測地点数と等しい値を指定する。  */
  int ihmax=24, idmax=31, nmax=146;

/*  配列(入力データ)の宣言:
      ista [nmax]: 観測地点番号。東京(大手町)は 44132 など。
      itype [idmax, nmax]: 観測地点の種類。
                           降水量のみの場合は 1、4 要素の場合は 4。
                           データが存在しない場合は 0。
      xx [nmax]: 観測地点の経度 [度]。
      yy [nmax]: 観測地点の緯度 [度]。
      ir [ihmax, idmax, nmax]: 降水量 [時間]。
      id [ihmax, idmax, nmax]: 風向 [16 方位、北北東=1、東=4]。
      iw [ihmax, idmax, nmax]: 風速 [m/s]。
      s [ihmax, idmax, nmax]: 日照時間 [時間]。
      t [ihmax, idmax, nmax]: 気温 [℃]。
      配列のサイズが大きいので、静的変数(static)として宣言する。
      static では、関数が呼ばれるごとに新たにメモリを確保するのではなく、
      プログラムが終了するまで固定されたメモリを用いる。  */
```

```
  static int ista[146], itype[31*146],
    id[24*31*146], ir[24*31*146], iw[24*31*146];
  static float xx[146], yy[146], hh[146],
    s[24*31*146], t[24*31*146];

/*  配列(出力データ)の宣言:
      ird [nmax]: 指定した日の日降水量 [mm]。  */
  static int ird[146];

  int ihh, idd, imm, iyy, n, isum, status;
  FILE *fp;

/*  欠損値: データが無効のとき、imiss、rmiss で指定された値が格納される。
            整数型の場合は imiss、浮動小数点型の場合は rmiss である。  */
  int imiss = 999;
  float rmiss = 999.9;

/*  サブルーチン amdid によって、観測地点リストを読みこむ。  */
  status = amdid (nmax, ista, xx, yy, hh);

/*  年月日を指定する。
    入力を求めるメッセージを標準出力に書き出す。  */
  printf( "%s¥n", "Year, Month, Day ?" );

/*  scanf 文で入力値を読みこむ。  */
  scanf( "%d,%d,%d", &iyy, &imm, &idd );

/*  関数 amdhr によって、指定された年、月の観測データ(時別値)を読むこむ。
    ※関数 amdhr は、iyy、imm で指定された年、月について
      データを読みこむ。  */
  status = amdhr (ihmax, idmax, nmax, ista, iyy, imm, imiss, rmiss,
                  itype, ir, id, iw, s, t);

/*  ここから for ループが始まる。
    各地点について、日降水量を計算する。  */
  for (n=1; n<=nmax; n++)
  {

/*  出力データを格納する配列に、あらかじめ欠損値を入れておく。  */
    ird[n-1] = imiss;
```

```
/*  データが存在する場合だけ、
    つまり、itype[(idd-1)+(n-1)*idmax]がゼロでない場合だけ、
    計算を実行する。*/
    if (itype[(idd-1)+(n-1)*idmax] != 0){

/*  和の値にゼロを代入する。  */
      isum = 0;

/*  ここから for ループが始まる。
    各地点について、1 時から 24 時までの時間降水量を合計する。  */
      for (ihh=1; ihh<=ihmax; ihh++)
      {

/*  欠損値を見つけたときはその地点の計算を中止する。
    break 文で for ループから出る。  */
        if (ir[(ihh-1)+(idd-1)*ihmax+(n-1)*idmax*ihmax] == imiss){
          break;
        }

        isum = isum + ir[(ihh-1)+(idd-1)*ihmax+(n-1)*idmax*ihmax];

/*  ここで for ループが終了する。  */
      }

/*  欠損値を見つけたときはその地点の計算を中止する。
    break 文で for ループから出たときは ird に isum を代入しない。  */
      if (ir[(ihh-1)+(idd-1)*ihmax+(n-1)*idmax*ihmax] != imiss){

/*  合計値を ird に代入する。  */
        ird[n-1] = isum;

      }

    }

/*  ここで for ループが終了する。  */
  }

/*  出力ファイルを開く。
```

```
    モードは"w"(テキストファイルへの書き出し)を指定する。  */
  fp = fopen( "output.txt", "w" );

/*  ここから for ループが始まる。
    各地点について、結果を出力する。  */
  for (n=1; n<=nmax; n++)
  {

/*  データが存在する場合だけ、
    つまり、itype[(idd-1)+(n-1)*idmax]がゼロでない場合だけ、
    結果を出力する。*/
    if (itype[(idd-1)+(n-1)*idmax] != 0){

/*  地点番号、緯度、経度、降水量を出力する。  */
      fprintf( fp, " %5d %8.3f %8.3f %3d¥n",
               ista[n-1], yy[n-1], xx[n-1], ird[n-1] );

    }

/*  ここで for ループが終了する。  */
  }

/*  ファイルを閉じる。  */
  fclose( fp );

  return 0;

}
```

実行例：

```
/home/snaoki> f77 prog01_2.f amedas_sub.f
/home/snaoki> ./a.out
 Year, Month, Day ?
2009,1,2
/home/snaoki> cat output.txt
 40041    36.867  140.640    0
 40046    36.840  140.775    0
 40061    36.775  140.350    0
<中略>
 46211    35.175  139.633    0
```

※C の場合は、コンパイル時に-lm 、-lf2c オプションを指定する必要があります。

課題：関東地方の各地点における 2009 年 1 月の月降水量と月平均気温を、AMeDAS の時別値データから計算し、結果を GMT のようなアプリケーションを用いて地図上にプロットし、印刷して提出せよ。24×31 個の時別値がすべて有効な地点だけを表示すること。また、図のレイアウトの都合上、北緯 34° より南にある地点は作図しなくてよい（それ以外の地点はすべて表示できるように地図の範囲を適切に設定せよ）。

ヒント：サンプルプログラムでは、24 時間で合計（平均）したが、この課題では、ループを 2 段にして、さらに 31 日で合計（平均）すればよい。ただし、欠損値の扱いに関しては、サンプルプログラムでの処理手順をそのままコピーしてもうまく動作しない。この課題の場合、合計するときに、合計した回数を数えるための整数型の変数を別に用意する。つまり、

```
ISUM=ISUM+...
```

という形で合計していくときに、

```
ICOUNT=ICOUNT+1
```

のような処理を同時に行なう（もちろん初期にはゼロを代入しておく）。ただし、これらの処理は、ITYPE がゼロではなくて、かつ、データの値が欠損値ではないときだけ実行する（IF 文を用いる）。正常に合計の計算が終了すれば、ICOUNT の値は 24*31 になっているはずである。もし、ICOUNT の値が 24*31 になっていなかったら、途中に 1 回以上欠損があったことになるので、合計値（平均値）としては欠損値を出力すればよい。