An Introduction to the Piecewise Parabolic Method (PPM) Library
II. I/O and Data Analysis

David H. Porter,
Sarah E. Anderson,
Paul Ralph. Woodward,
and B. Edgar,

Laboratory of Computational Science or Engineering,
Department of Astronomy, School of Physics and Astronomy
116 Church St. SE.
University of Minnesota
Minneapolis, Minnesota 55455

2. Basic Routines

Experienced FORTRAN programmers know that it inserts record marks in the binary files it creates. These record marks can be annoying when such files are intended for use with routines written in other languages. Therefore, PPMLIB has four basic routines intended to circumvent this antisocial behaviour of FORTRAN. These routines are

a. PPM98_binopen(filename, iunit)

This opens a file for reading or writing whose name is the character string contained in `filename`, which is of type CHARACTER. The FORTRAN unit number which is to be assigned to the open file is given by the integer `iunit`.

a. PPM98_binopen_trunc(filename, iunit)

This opens a file for writing whose name is the character string contained in `filename`, which is of type CHARACTER. If a file of that name already exist, its contents will be destroyed. The FORTRAN unit number which is to be assigned to the open file is given by the integer `iunit`.

b. PPM98_binclose(iunit)

This closes the file associated with the FORTRAN unit number `iunit`, which was assumed to have been previously opened by a call to **PPM98_BINOPEN**. The argument `iunit` is an integer.

c. PPM98_binwrite(iunit, data, nbytes)

This will write `nbytes` contiguous bytes from memory beginning at the memory location of the first byte of `data` to the file associated with the FORTRAN unit number `iunit`, which was assumed to have been previously opened by a call to **PPM98_BINOPEN**. Note that the requested number of bytes, `nbytes` to write, may exceed the size of the variable `data`, which is used only as a pointer to the location of the first byte to write.

d. PPM98_binread(iunit, data, nbytes)

This will read from the file associated with the FORTRAN unit number `iunit`, which was assumed to have been previously opened by a call to **PPM98_BINOPEN**. The `nbytes` read from the file are placed into contiguous bytes of memory beginning at the memory location of the first byte of `data`. Note that if the requested number of bytes `nbytes` to read exceeds the memory space allocated to `data`, unforeseen and adverse consequences usually result.

The use of the above programs is illustrated by the following two simple examples.

```
      program example1
c
c   This is a simple example of creating and writing a binary file using
c    routines from the PPMLIB Library.
c
      character*8 filename
      real*4 data(100)
c
c   create a data set consisting of the square roots of the number 1 to 100
c
      do i=1,100
      data(i) = sqrt( float(i) )
      enddo
c
c   Open a file name "DATA"  and assign it to FORTRAN unit 10
c   If an existing file  named "DATA" destroy it.
c
      filename = 'DATA'
      iunit = 10
      call ppm98_binopen_trunc( filename, iunit )
c
c   write the contents of the array "data" to it.  Data contains 100*4 bytes
c
      nbytes = 100 * 4
      call ppm98_binwrite( iunit, data(1), nbytes )
c
c   The file "DATA" should contain exactly "nbytes" bytes.
c
c   close the file
c
      call ppm98_binclose( iunit )
c
      stop
      end
c     endprogram example1
```

The program `example1` will create a binary file which contains the square roots of the integral numbers from one to ten. The name of the file is DATA which is stored in the character variable `filename`. Space is allocated for real array `data` with 100 elements. This has been declared as `real*4`, so each element of the array has four bytes, and the total size of the array is 400 bytes. The square roots are then calculated and stored in the array `data`.

```
      do i=1,100
      data(i) = sqrt( float(i) )
      enddo
```

A disk file is opened and associated with FORTRAN unit 10.

```
      filename = 'DATA'
      iunit = 10
      call ppm98_binopen_trunc( filename, iunit )
```

The array is then written to the disk file, after the number of bytes to write has been calculated, recalling that each element of the array `data` consists of four bytes.

```
      nbytes = 100 * 4
```

```
      call ppm98_binwrite( iunit, data(1), nbytes )
```

Finally, the file is closed and the program exits.

```
      call ppm98_binclose( iunit )

      stop
      end
```

```
      program example2
c
c   This is a simple example of reading a binary file using routines
c   from the PPMLIB Library.
c
      character*20 filename
      real*4 data(100)
c
c   Open a file name "DATA"  and assign it to FORTRAN unit 10
c
      filename = 'DATA'
      iunit = 10
c
      call ppm98_binopen( filename, iunit )
c
c   read the contents of the file into the 100 elements of
c   the array array "data".
c
      nbytes = 100 * 4
      call ppm98_binread( iunit, data(1), nbytes )
c
c   Display the contents of data
c
      write(6,50)
50    format('   I    SQRT(I)',/, '---------------')
      do i=1,100
      write(6,100) i,data(i)
      enddo
100   format(1h ,i4,1x,f9.6)
c
c   close the file
c
      call ppm98_binclose( iunit )
      stop
      end
c     endprogram example2
```

The program `example2` reads from the binary file which contains the square roots of the integral numbers from one to ten, created by example1. The name of the file, `DATA`, is stored in the character variable `filename`. Space is allocated for real array `data` with 100 elements. This has been declared as `real*4`, so each element of the array has four bytes, and the total size of the array is 400 bytes. The square roots are then read from the file.

```
      nbytes = 100 * 4
      call ppm98_binread( iunit, data(1), nbytes )
```

After the square roots are displayed, the file is closed and the program exits.

```
      call ppm98_binclose( iunit )
      stop
```

```
end
```

The next section will demonstrate how the basic I/O routines presented here may be used to create customized restart dumps.