JCB processing system concepts

John C. Bancroft and Zaiming Jiang

ABSTRACT

A simplified processing system is presented for research that is simple to use, portable between software platforms, and independent of any other processing systems. The system is designed around input and output SEGY files and uses an input text file to define these files and other parameters. Commercial packages may then used for displaying or additional processing.

INTRODUCTION

Bancroft has developed software in a number of environments and has always had great difficulty with portability. In many cases portability was not an issue as the software was propriety information and could not be transferred. It is desired that CREWES software be portable so that it can be quickly installed and used on different platforms. Maintaining SEGY input and output enables straightforward interaction with other processing systems.

A number of platforms have been tried at CREWES, but there continues to be issues with maintainability and compatibility. For example, algorithm implemented in ProMAX, may have to be reprogrammed as the internal data structure, within ProMAX, has changed. MATLAB, a very useful development platform, does not perform as efficiently as desired when computing large volumes of data. As a result, our intention is to have a simple system that can be easily maintained, is portable, economical, and compatible with any processing system.

The system described here is not a comprehensive program bundled into one software package, but rather individual programs that will each perform a single major process such as FK poststack migration, prestack Kirchhoff migration, surface consistent statics estimation, etc. Utility functions can be used within each main program, such as filtering, decon, etc.

Simple migration programs are in development to accompany teaching with a migration course. EOM software is also under development.

BASIC CONCEPTS OF THE SYSTEM

1. The system uses the languages of FORTRAN and “C” that are available free off the internet under the Cygwin package. Only the basic generic functions of these packages are used for compatibility.

2. The system is being developed on a PC using WINDOWS so that images can be easily inserted into WORD documents.

3. Basic software design uses “C” for I/O and memory management and FORTRAN for numeric algorithms.

4. Seismic data input and output uses SEG-Y rev 0 Data Exchange format. Geometry data will be read from headers to form an internal geometry that can then be used for sorting as required. This geometry data will not be stored.

5. A unique structure file ("C") is defined for each main program that will be used for the input parameter file.

6. Structures in "C" and common memory in FORTRAN are not be used for compatibility purposes. They are OK in some systems but not all.

7. Parameters passed to and from FORTRAN should be minimized. If the number of calling parameters becomes too large, then they will be passed in integer and real arrays.

8. Character strings will not be passed from "C" to FORTRAN

9. Names should be consistent between programs, i.e. ntr, ntrIn, ntrOut, nsamp, etc

10. A unique parameter file in plain text file format is designed for each "main" program. This text file will contain the names of input and output SEGY files along with parameters.

11. The input parameter file is directed into the "main" file. i.e the input line will be of the form `./Model < ModelParameters`

12. All the required files and parameter definitions are described in a ReadParms files. Each line of text will define each data file or a parameter.

13. Each input parameter will be defined in a "C" source file, in a form that can be copied and pasted into other documents.

14. Input parameters will be unique, i.e. no multiple choices. Given TIMEms is OK, timesms, or timeMS is not allowed.

15. The order of the parameter lines are not important, but should follow a logical order. Typically SEG-Y files will be at the beginning.

16. Filenames will have no blank spaces as they have difficulty in some environment.

17. A debug line (**`idebug`**) allows various forms of text output, where 0 in minimal and 10 verbose. A debug level of 7 will output lines from each internal loop.

18. A # as the first character identifies a comment line

19. An "`end`" line in the parameter file stops the reading of the input parameters. This enables other lines to be "stored" below the "end" line.

20. Time zero coincides with the first sample in an array.

21. A “Makefile” file will be used to compile the appropriate software.

22. A File of all subprograms will be maintained. It will be compatible across platforms.

EXAMPLE

An example is given of running a modelling program. This is a very simple modelling program that can insert lines, points, diffractions, or semi-circles into a section that is assumed to be zero offset and with a constant velocity. The input text file is:

```
# This input file is read in as an indirect, i.e.
#./Model  < input-data.txt
file_out    aussi.sgy
ntr         128
nsamp       512
delt        0.002
delx        20
vel         3000
linear      12 0.1 95 0.35 1.0
linear      55 0.3 75 0.8 1.0
point       30 0.3 1.0
point       35 0.3 1.0
point       36 0.3 1.0
diff        32 0.1 1.0
diff        33 0.1 1.0
diff        64 0.1 1.0
diff        40 0.4 1.0
diff        60 0.5 0.5
diff        80 0.6 0.25
cir         100 0.3 0.5
cir         100 0.5 0.5
cir         100 0.7 0.5
filter      10,15,50,60,5
idebug      1
end
diff        42 0.4 1.0
diff        43 0.4 0.5
diff        44 0.4 0.5
diff        45 0.4 0.5
diff        46 0.4 0.5
diff        47 0.4 0.5
diff        48 0.4 0.5
diff        49 0.4 0.5
diff        50 0.4 0.5
```

The limited output for idebug = 1 is shown below in Figure 1.

```
$ ./Model < InputParms.txt

*** Starting read-parms subroutine
*** Finished read-parms subroutine
*** Before call to JCBmodel
   Completed the linear events.
   Completed the points.
   Completed the diffractions.
   Completed the semi-circles.
   Completed filter.
*** After call to JCBmodel
*** Finished writing SEGY file.
    Traces written: 128.

***  All done ***
```

FIG. 1. Text output from the above input text file.

Data output is a SEGY file "`aussi.sgy`" that can be read into any processing system. I use VISTA software for displaying the files as it is PC compatible and the images can be inserted directly into a this WORD document.

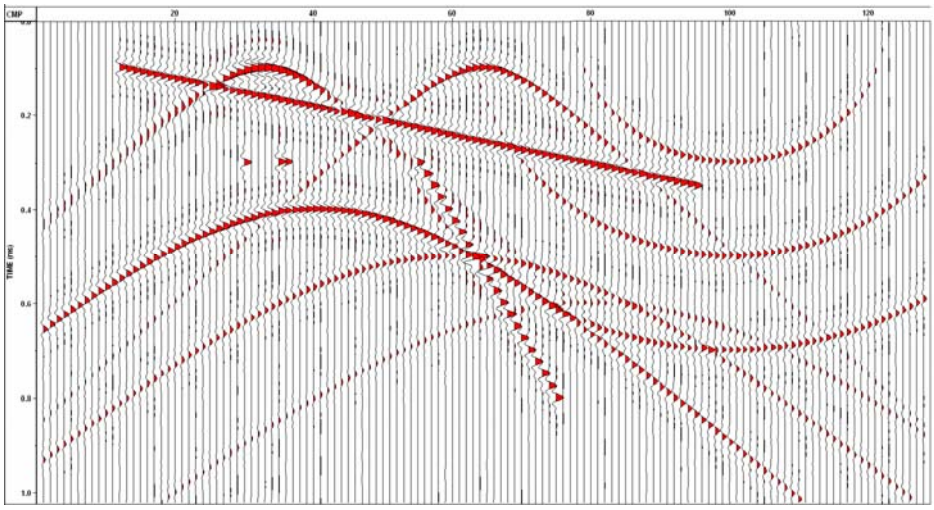

FIG. 2. VISTA display of the model program with the input and text output shown above.

Other migration programs under development are a simple Kirchhoff (diffraction stack), advanced Kirchhoff, FK, Phase Shift, Prestack Kirchhoff, and EOM.

CONCLUSIONS

A simple seismic processing "system" was presented with the intent that it be portable, economical, easy to install, simple to use, and implement. It is also intended that this "system" be used by students to unify their software development in order to make it more accessible to CREWES sponsors.

ACKNOWLEDGEMENTS

We are grateful for the financial support of the CREWES sponsors. We also thank GEDCO for providing access to VISTA, which is used for displaying seismic data in this report.