IOSR Journal of Applied Geology and Geophysics (IOSR-JAGG)
e-ISSN: 2321–0990, p-ISSN: 2321–0982.Volume 4, Issue 4 Ver. I (Jul. - Aug. 2016), PP 51-68
www.iosrjournals.org

SPS to Seismic Data Mapping: An AWK Script for Merging Field Geometry with Seismic Data

Anil Kumar Semwal
(Processing/ Pan India Consultants, Pvt. Ltd., India)

***Abstract**: SPS (Shell Processing Support) files contain vital acquisition information and are standard input for merging field geometry with seismic data. These files are interrelated and must conform one another and the data as well for some of the prime fields. But on many occasions, it does not hold and requires user's massive efforts to rectify this correspondence. The best way to circumvent this problem is to automate the procedure through coding a script. With this concept, an AWK script is coded, which replenishes processing geophysicist's inventory with data mapped SPS files which can straight away be input for updating trace headers of seismic data.*

***Keywords:** AWK, Field geometry, SPS files, Seismic data*

I. Introduction

SPS files are being widely used for over a couple of decades in seismic data acquisition & processing. Their use has enabled a processing geophysicist with smooth working on a structured set of fields spread among three files (R, S & X) comprising all the acquisition parameters needed.

The purpose of the SPS format is to establish a common standard for the transfer of positioning and geophysical support data from land 3D field crews to seismic processing centres. In principal the format can also be used for land 2D surveys (SEG Technical Standards Committee on Ancillary Data Formats, 1995).

Using these files one can straight away update seismic data trace headers by Line no., Pkt no., Coordinates, Statics & other near surface information, provided inter relationship among some fields conform i.e. these fields can be mapped across the SPS files and between data & SPS files as well. But, on number of instances this conformal relationship does not hold good and a hectic manual intervention is called for.

An attempt is made in this regard by coding an AWK script (SPStoSeismicDataMapping.awk) to automate the process and alleviate a processing geophysicist by saving his time. The script does exhaustive mapping, reordering & structuring in these files with reference to the data.

II. Method

AWK is an interpreted programming language designed for text processing and typically used as a data extraction and reporting tool. The language uses the string data type, associative arrays (i.e. arrays indexed by key strings), and regular expressions. It is a standard feature of most Unix-like operating systems. It is used extensively in this work with other shell command sets.

It is being aimed to map FFID from data onto X.SPS, SP from X.SPS onto S.SPS, SPIndex from X.SPS onto S.SPS, channel range onto receiver picket range for a receiver line within X.SPS, receiver picket range for a receiver line from R.SPS onto X.SPS, channel range for a FFID from data onto X.SPS. The procedure is divided into five phases:

Phase-1 scripts map FFID from data onto X.SPS and output 2 files: XfileInStandardFmt (finalized data mapped X.SPS file which can straight away be input in a merging module) & RestrictDataFFID (enlist unmapped FFIDs to be deselected from data while actual data merging with geometry is carried out).

Phase-2 scripts map SLNo & SP from data mapped X.SPS onto S.SPS such that S.SPS comply with that of X.SPS.

Phase-3 scripts map SP index from data mapped X.SPS onto S.SPS. Two choices are kept for SP index mapping session. First is manual updation and the second is automatic updation. Choosing manual updation requires one to update SP index of S.SPS file: FileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually for only entries (i.e. unmapped indices) in SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile as per his notion & understanding. While choosing automatic updation straight away furnishes finalized runnable file SfileInStandardFmt_SPIndexUpdated.

Phase-4 scripts generate a finalized R.SPS: RfileInStandardFmt.

By the end of Phase-4 all the three SPS files have been mapped among themselves and conform to data FFID as well.

Phase-5 scripts map channel range onto receiver picket range for a receiver line within X.SPS, receiver picket range for a receiver line from R.SPS onto X.SPS & channel range for a FFID from data onto X.SPS. This

phase furnishes three files:

AmbiguityInMapping_FromNoOfChannels_OntoNoOfRP_WithinXFile,AmbiguityInMappingOf_Min MaxRPInRL_FromRFile_OntoXFile & RestrictChannelRangeInDataFFID. First two ambiguity files report inconsistencies, if prevail and can be easily resolved by using observer log or other available information. User need to rerun the script after resolving these inconsistency issues and get the data mapped finalized SPS files. Third file imposes restriction on channel range corresponding to a FFID in case X.SPS channel range for a FFID does not map onto corresponding data FFID channel range. User need to select the enlisted channel range corresponding to FFID in data while actual geometry merging with data is carried out.

With this, all vital fields for geometry merging purpose are mapped across the files and at the same time comply with the data at FFID & channel level.

III. Results and Discussion

3.1 Input Dataset

Input dataset includes three SPS files (R, S & X) and two data extracted files (FFID list & Channel statistics list). All these files need to be space delimited. Moreover, SPS files must contain standard 15 fields in the designated order and they should be not null.

A specimen of an input field shot gather before merging & binning is shown below (Fig.1). Figure 1 shows that only "Field Record number" (i.e. Word 22) & "Trace number on the spread" are populated on the trace headers in this case.

S

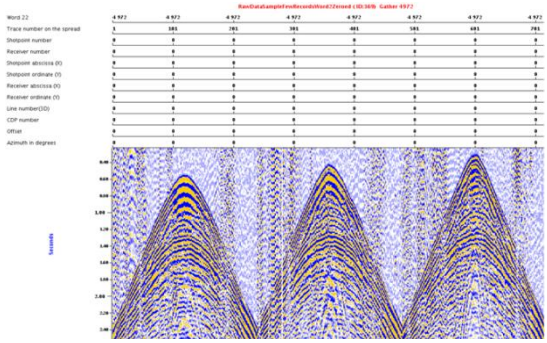

Figure 1 An input field shot gather before merging. Only Field Record number (i.e. Word 22) & Trace number on the spread only are populated in the trace headers

3.2 Output Dataset

Output dataset contains finalized SPS files in standard format & restriction files in Final sub folder and ambiguity files in Ambiguities sub folder of the current working directory.

Ambiguity files report inconsistencies, if prevail, in mapping of receiver pickets onto channels with in X.SPS and mapping of receiver pickets from R.SPS onto X.SPS as well. These reported ambiguities can be easily resolved by the user using observer log or other available information. Next, he would need to rerun the script and get, ready to run finalized SPS files.

Restriction files provide information on constraints for FFID & channel range, which need to be incorporated during merging of geometry with actual data.

A specimen of output field shot gather after using these finalized SPS files in merging/ binning job is shown below (Fig.2). Figure 2 illustrates that all of the fields, related to geometry such as Field Record, Trace No., Shot, Receiver, X/ Y Coordinates, Inline, CDP, Offset & Azimuth are populated on the trace headers.

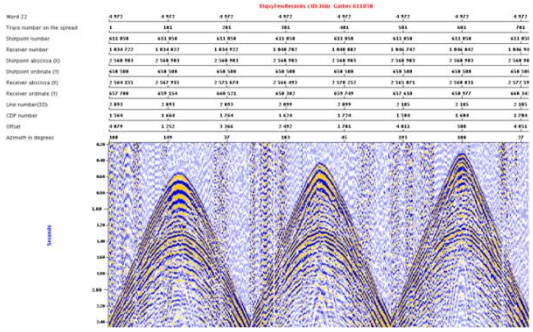

Figure 2. A geometry merged/ binned output shot gather. All surface & subsurface parameters have been populated in the trace headers

3.3 SPS to Seismic Data Mapping Script

The script works on any Linux/ Unix like platform. It's help are commented with Doc> which lets user to identify which kind of files & what values are to be input when prompted. Data processing is commented with <Process-> which lets user understand the mapping & restructuring mechanism. These help & process comments also appear when script runs and keeps the user abreast with the mechanism.

```
echo "****************************************** SPStoSeismicDataMapping.awk ******************************************"
echo
"#################################################################################################################"
echo "Doc> Author at anil_semwal@hotmail.com created this AWK Script to save couple of hours to days of a Geophysicist while fixing SPS"
echo "Doc> files for merging purposes. Author would be happy to listen any suggestion towards improvement in the script."
echo
"#################################################################################################################"
echo "Doc> This script prompts user to supply input files & few parameters. Next, it does exhaustive sorting, mapping and rearrangement: "
echo
"#################################################################################################################"
echo "Doc> ____________________________________________________"
echo "Doc> | Maps          |        From        |        Onto  |"
echo "Doc> ____________________________________________________"
echo "Doc> | FFID          |        Data        |        X.SPS |"
echo "Doc> | Channel Ranges |       Data        |        X.SPS |"
echo "Doc> | SP            |       X.SPS        |        S.SPS |"
echo "Doc> | RP Range      |       X.SPS        |        R.SPS |"
echo "Doc> | SP Index      |       X.SPS        |        S.SPS |"
echo "Doc> ____________________________________________________"
echo "Doc> Also checks for integrity between Channel Ranges & Receiver pkt Ranges within X.SPS."
echo " "
echo "Doc> Finally, the Anomaly files(with instructions) & ready to run SPS files(in standard SPS format) are output in  /Ambiguities & /Final subfolders"
echo "Doc> of the Current working directory respectively. With the help of these user can straightaway run the merge job without any error"
echo
"#################################################################################################################"
echo " "
echo "Doc> Five text files are required in this context:"
echo "Doc> Three SPS files : (i) X.SPS, (ii) S.SPS & (iii) R.SPS "
echo "Doc> Two List files extracted from raw Sgy or Sgd data which have essentially been sorted on Two keys FFID & ChannelNo: "
echo "Doc>                (i) FFID list file & (ii) FFID-Channel Statistics list file"
echo " "
echo "Doc> General law for all the above 5 files is that they should be space delimited."
echo " "
echo "Doc> Law for different files:"
echo "Doc> X.SPS: Contains 15 fields & all should be Not Null & must follow the following Field order"
echo "Doc> RecordId TapeNo FFID Increment Code SLNo SP Index FromCh ToCh Increment RLNo FrRP ToRP Increment"
echo " "
echo "Doc> S.SPS: Contains 15 fields & all should be Not Null & must follow the following Field order"
echo "Doc> RecordId SLNo SP Index Code ShotStaticsCor SPDepth Datum Uphole WaterDepth X Y SurfaceElev DOY HMS"
echo " "
echo "Doc> R.SPS: Contains 15 fields & all should be Not Null & must follow the following Field order"
```

```
echo "Doc> RecordId RLNo RP Index Code RecStaticsCor RPDepth Datum Uphole WaterDepth X Y
SurfaceElev DOY HMS"
echo " "
echo "Doc> FFID list file: It must contain Data FFID in one of the field. "
echo "Doc>                Header/ footer lines in the file are allowed. Script takes care of them."
echo " "
echo "Doc> FFID-Channel Statistics list file: It must contain Data FFID & corresponding Minimum &
Maximum Channel No. in any three fields."
echo "Doc>            Header/ footer lines in the file are allowed. Script takes care of them."
echo " "
echo
"#############################################################################################
#################"
echo " "
echo "Doc> Run the software specific module with input as the field data and get the desired FFID list &  FFID-
Channel Statistics list file "
echo
"#############################################################################################
#################"
echo "Phase-1 Begins....."
echo
"#############################################################################################
#################"
echo "Following set of Phase-1 scripts map FFID from X.SPS onto FFID List File and output 2 files:"
echo "XFileInStandardFmt: Finalized runnable X.SPS file which can straight away be input in a merging
module"
echo "RestrictDataFFID: Enlist Unmapped FFIDs to be deselected from Data while actual Data merging with
geometry is carried out"
echo
"#############################################################################################
#################"
echo " "
echo "Input X.SPS File with extension:"
read X
echo $X
echo "Input S.SPS File with extension:"
read S
echo $S
echo "Input R.SPS File with extension:"
read R
echo $R
echo ""
echo
"#############################################################################################
#################"
echo "Doc> Next User will be prompted for No.of Chars including spaces Before FFID in X.SPS & FFID length
in X.SPS."
echo "Doc> An excerpt of this type of file is as follows for your reference to answer the prompts:"
echo "Doc> ____________________________________________________"
echo "Doc>|X 0 1339 1 1 16 1049 1 1 240 1 1034 182 421 1     |"
echo "Doc>|X 0 1339 1 1 16 1049 1 241 480 1 1040 182 421 1   |"
echo "Doc>|X 0 1339 1 1 16 1049 1 481 720 1 1046 182 421 1   |"
echo "Doc>|X 0 1339 1 1 16 1049 1 721 960 1 1052 182 421 1   |"
echo "Doc>|X 0 1339 1 1 16 1049 1 961 1200 1 1058 182 421 1  |"
echo "Doc>|X 0 1339 1 1 16 1049 1 1201 1440 1 1064 182 421 1 |"
echo "Doc>|X 0 1339 1 1 16 1049 1 1441 1680 1 1070 182 421 1 |"
echo "Doc>|X 0 1339 1 1 16 1049 1 1681 1920 1 1076 182 421 1 |"
echo "Doc>|X 0 1339 1 1 16 1049 1 1921 2160 1 1082 182 421 1 |"
echo "Doc>|X 0 1339 1 1 16 1049 1 2161 2400 1 1088 182 421 1 |"
```

```
echo "Doc>|X 0 1339 1 1 16 1049 1 2401 2640 1 1094 182 421 1 |"
echo "Doc>|X 0 1339 1 1 16 1049 1 2641 2880 1 1100 182 421 1 |"
echo "Doc>|X 0 1342 1 1 16 1050 1 1 240 1 1034 182 421 1     |"
echo "Doc>|X 0 1342 1 1 16 1050 1 241 480 1 1040 182 421 1   |"
echo "Doc>|_____________________________________________________|"
echo "Doc> In the above example, Avoid Chars i.e. No.of Chars including spaces Before FFID in X.SPS = 4"
echo "Doc>                          Max. No.of Chars for comparison i.e. FFID length in X.SPS = 4"
echo "Doc> Use above help to answer the prompts in your case"
echo
"#############################################################################################
#################"
echo " "
echo "Avoid Chars i.e. No.of Chars including spaces Before FFID in X.SPS?"
read C
echo $C
echo "Max. No.of Chars for comparison i.e. FFID length in X.SPS?"
read N
echo $N
rm -rf Final
echo "<Process-> Removing Final Folder"
mkdir Final
echo "<Process-> Creating Final Folder"
rm -rf Ambiguities
echo "<Process-> Removing Ambiguities Folder"
mkdir Ambiguities
echo "<Process-> Creating Ambiguities Folder"
echo "<Process-> Unique filtering of FFIDs in X.SPS results X1"
uniq -s$C -w$N $X> X1
echo "<Process-> Sorting on FFID & FrCh keys in X1 results X2"
sort -n --key=3 --key=9 X1 > X2
echo " "
echo "<Process-> Also, User need to have a version of X.SPS (i.e.X3 with all contents) merely sorted on FFID
& FrCh keys"
sort -n --key=3 --key=9 $X > X3
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s%1s\n",\
$1," "$2," ",$3," ",$3," ",$4," ",$5," ",$6," ",$7," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15 }' X3
> X4
echo "<Process-> FFID inserted once in X3 and results X4"
echo " "
echo
"#############################################################################################
#################"
echo "Doc> User need to have FFID List File (space delimited) generated from Raw Data. It must have Ffid in
one of the column. "
echo "Doc> It may or may not have Header Lines in the beginning and Footer Lines after the valid Listing of
Data in this context."
echo "Doc> Valid Listing of Data is the part of FFID List File, wherein, FFIDs appear."
echo "Doc> An excerpt of this type of file is as follows for your reference to answer the prompts"
echo " "
echo "Doc> _____________________________"
echo "Doc>| Header Line1            |"
echo "Doc>| Header Line2            |"
echo "Doc>| LISC\$# 1 ; 2=         1  |"
echo "Doc>| LISC\$# 1 ; 2=         2  |"
echo "Doc>| LISC\$# 1 ; 2=         3  |"
echo "Doc>| LISC\$# 1 ; 2=         4  |"
echo "Doc>| LISC\$# 1 ; 2=         5  |"
```

```
echo "Doc>| Footer Line1            |"
echo "Doc>| Footer Line2            |"
echo "Doc>| Footer Line3            |"
echo "Doc>|_____________________________|"
echo " "
echo "Doc> In the above example No. of Lines in the Header (i.e.No.of Lines before FFIDs start appearing) =
2"
echo "Doc>                Field No. of FFID = 5"
echo "Doc>                No. of Last Line upto which valid data exists (i.e.FFIDs are enlisted) in this context =
7"
echo " "
echo "Doc> Use above help to answer the prompts in your case"
echo
"#############################################################################################
#################"
echo " "
echo "Enter FFID List File Name with extension: "
read FList
echo $FList
echo " "
echo "Enter No. of Lines in the Header of $FList file: "
read nlHdr_FList
echo $nlHdr_FList
echo " "
echo "Enter Field No. of FFID in the $FList file: "
read fnF_FList
echo $fnF_FList
echo " "
echo "Enter No. of Last Line in $FList file upto which FFIDs are enlisted: "
read nlD_FList
echo $nlD_FList
echo " "
echo "nlHdr_FList " $nlHdr_FList  "fnF_FList " $fnF_FList "nlD_FList " $nlD_FList> UP_FList
echo "<Process-> $FList related User Prompted values are collected in UP_FList"
echo " "
cat UP_FList $FList > FList1
echo "<Process-> UP_FList is concatenated at the top of $FList"
echo " "
awk -F " " 'NR ==1{
                    nlHdr_FList = $2
                    nlD_FList = $6
                    fnF_FList = $4
                    next}
                    NR > nlHdr_FList+1 && NR <= nlD_FList+1{print $fnF_FList
                    }' FList1 > FList2
echo "<Process-> FFIDs from FList1 are extracted & stored in FList2"
echo " "
sort -n --key=1 FList2 > FList3
echo "<Process-> FList2 is sorted on FFID key & stored in FList3"
echo " "
awk -F" " '{printf "%1s%1s%1s%1s%1s\n",$1," "$1," "$1," "$1," "$1}' FList3 > FList4
echo "<Process-> FFID field is inserted four times into FList3 for mapping purposes & stored in FList4"
echo " "
awk -F " " 'BEGIN { OFS=" "} FNR==NR {a[$3$4]=$5;next}($3$4 in a && $5=$5" "a[$3$4])' FList4 X4 > X5
echo "<Process-> Mapping of FFIDs from FList4 onto X4 is carried out using Hash Key, which results in X5, a
Data FFID mapped version of X.SPS"
echo " "
awk -F" " '{ printf
"%1s%6i%4i%1i%1i%16s%8s%1i%4i%4i%1i%16s%8s%8s%1i\n",$1,$2,$3,$5,$7,$8,$9,$10,$11,$12,$13,$14
```

```
,$15,$16,$17 }' X5 > XFileInStandardFmt
echo "<Process-> Required fields are extracted from X5 & formatted, adhering standard X.SPS structure, results
in XFileInStandardFmt"
cp XFileInStandardFmt Final
echo " "
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s%1s\n",\
$1," "$2," ",$3," ",$3," ",$4," ",$5," ",$6," ",$7," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15 }' X2
> X6
echo "<Process-> FFID field is inserted one time into X2 (having unique FFID entries) for mapping purposes &
stored in X6"
echo " "
awk -F " " 'BEGIN { OFS=" "} FNR==NR {a[$3$4]=$5;next}($3$4 in a && $5=$5" "a[$3$4])' FList4 X6 > X7
echo "<Process-> For Qc purposes: Mapping of Unique FFIDs from FList4 onto X6 is carried out using Hash
Key, which results in X7"
echo " "
echo "<Process-> This code maps field 3 & 4 from X6 onto corresponding fields of FList4 using Hask Key and
prints fields 3 & 4 if FFIDs map,"
echo "          else prints all 5 fields i.e. entire row of FList4 in T"
awk -F " " 'BEGIN { OFS=" "} NR==FNR && a[$3" "$4]=$5 {} NR>FNR { k= $3" "$4; if(k in a) print k; else
print $0}' X6 FList4 > T
awk -F " " 'BEGIN { OFS=" "} NR==1{printf "%180s\n","Important-
>>:While_actual_Data_merging_with_geometry_is_carried_out,\
following_FFIDs_need_to_be_deselected_from_Input_Data_File_as_these_FFIDs_are_not_common_among_D
ata_&_X.SPS_File"}' T > T1
awk -F " " '{printf "%1s\n",$5}' T >> T1
grep -v " " T1 > T2
echo " "
echo "<Process-> Sorting on FFID key in T2 results RestrictDataFFID"
sort -n --key=1 T2 > RestrictDataFFID
cp RestrictDataFFID Final
echo " "
echo "Phase-1 Ends."
echo " "
echo
"##########################################################################################
#################"
echo "Phase-2 Begins."
echo "Following set of Phase-2 scripts map SLNo & SP from Data Mapped X.SPS (i.e.X5) onto S.SPS such that
S.SPS comply with that of X.SPS"
echo " "
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s%1s%1s%1s%1s\n",\
$2," ",$3," "$1," "$2," ",$3," ",$2,""$3," ",$4," ",$5," ",$6," ",$7," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13,"
",$14," ",$15 }' $S > S1
echo "<Process-> Inserted fields such as SL,SP & concatenated version of SL,SP in S.SPS and stored in S1"
echo " "
awk -F" " '{printf "%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s\n",$1," "$2," "$3," "$5,"
"$7," "$8," "$9," "$10," "$11," "$12," "$13," "$14," "$15," "$16," "$17 }' \
X5 > X8
echo "<Process-> Fields are extracted from X5 & formatted as single space delimited, furnishes X8."
echo " "
echo "<Process-> Unique filtering of FFID in X8 furnishes X9"
uniq -s$C -w$N X8 > X9
echo " "
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
```

```
s%1s%1s%1s%1s%1s%1s%1s\n",\
$6," ",$7," "$1," "$2," ",$3," ",$4," ",$5," ",$6," ",$7," ",$6,""$7" ",$8," ",$9," ",$10," ",$11," ",$12," ",$13,"
",$14," ",$15 }' X9 > X10
echo "<Process-> Fields such as SLNo,SP & concatenated SLNo,SP from X9 are further inserted and stored in
X10"
echo " "
echo "<Process-> This code maps SLNo,SP from X10 onto S1 using Hask Key and print entire row of S1 with
FFID extracted from X10 into S2, if SLNo,SP map"
awk -F " " 'BEGIN { OFS=" "} FNR==NR {a[$1$2]=$5;next}($1$2 in a && $5=$5" "a[$1$2])' X10 S1 > S2
echo " "
echo "<Process-> Sorting on FFID key in S2 results S3"
sort -n --key=6 S2 > S3
echo " "
awk -F" " '{ printf
"%1s%16s%8s%1i%2s%4i%4.1f%4i%2i%4.1f%9.1f%10.1f%6.1f%3i%6i\n",$3,$4,$5,$8,$9,$10,$11,$12,$13,
$14,$15,$16,$17,$18,$19 }' \
S3 > SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually
echo "<Process-> Required fields are extracted from S3 & formatted, adhering standard S.SPS structure, results
in SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually"
echo " "
echo "Phase-2 Ends."
echo " "
echo
"#############################################################################################
#################"
echo " "
echo "Phase-3 Begins."
echo "Following set of Phase-3 scripts map SPindex from Data Mapped X.SPS onto S.SPS"
echo " "
awk -F" " '{ printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s\n",\
$3," ",$4," ",$5," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15," ",$16," ",$17," ",$18," ",$19 }' S3 >
S4
echo "<Process-> Required fields are extracted from S3 & formatted as single space delimited, furnishes S4"
echo " "
awk -F" " '{ printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s\n",\
$1," ",$2," ",$3," ",$5," ",$7," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15," ",$16," ",$17 }' X5 >
X11
echo "<Process-> Required fields are extracted from X5 & formatted as single space delimited, furnishes X11"
echo " "
echo
"#############################################################################################
#################"
echo "Doc> Now I keep 2 choices for SP index mapping session. First is manual editing and the Second is
automatic updation."
echo "Doc> I would strongly recommend to go for Second i.e. automatic updation."
echo " "
echo "Doc> If One wants to manually edit the SP Index in S.SPS file not matching with corresponding SP Index
in X.SPS"
echo "Doc> Then use entries in file: SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile to edit SP
Index in"
echo "Doc> file: SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually as per your notion &
understanding."
echo "Doc> But beware of that SP Index in Files: XFileInStandardFmt &
SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually must match."
echo "Doc> All of these 3 files can be found in /Final sub folder."
```

```
echo
"#############################################################################################
#################"
echo " "
echo "If Manual editing option is chosen"
echo "Then at prompt ENTER: 1"
echo " "
echo "Else if One wants that SP Index in S.SPS are automatically updated by script corresponding to SP Index
in X.SPS"
echo "Then at prompt ENTER: 2"
echo " "
read I
echo $I
if [ $I == 1 ]
then
echo "<Process-> Unique filtering of FFID in X11 furnishes X12"
uniq -s$C -w$N X11 > X12
echo " "
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s%1s\n",\
$1," "$2," ",$3," ",$4," ",$5," ",$6," ",$7," ",$8," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15 }' X12
> X13
echo "<Process-> Inserted field SPIndex in X12 and stored in X13"
echo " "
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s\n",\
$1," "$2," ",$3," ",$4," ",$5," ",$6," ",$7," ",$4," ",$4," ",$4," ",$4," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13,"
",$14," ",$15 }' S4 > S5
echo "<Process-> Inserted field SPIndex four times in S4 and stored in S5"
echo " "
echo "<Process-> This code maps field 8 & 9 from X13 onto corresponding fields of S5 using Hask Key and
prints fields 8 & 9 if FFIDs map,"
echo "          else prints all fields i.e. entire row of S5 in T3"
awk -F " " 'BEGIN { OFS=" "} NR==FNR && a[$8" "$9]=$10 {} NR>FNR { k= $8" "$9; if(k in a) print k;
else print $0}' X13 S5 > T3
echo " "
echo "<Process-> Creating header for SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile"
awk -F " " 'BEGIN { OFS=" "} NR==1{printf "%1s\n","Important: Field 4 of this file is SP Index of S.SPS File:
\
SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually not matching with SP Index of X.SPS File:
XFileInStandardFmt, \
so correct the SP Index in S.SPS File: SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually for
the following entries only"}' T3 > TempHdr1
grep "S" T3 > T4
echo " "
awk -F " " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s\n",\
$1," ",$2," ",$3," ",$4," ",$5," ",$6," ",$7," ",$12," ",$13," ",$14," ",$15," ",$16," ",$17," ",$18," ",$19}' T4 > T5
echo "<Process-> Required fields are extracted from T4 & formatted as single space delimited, furnishes T5"
echo " "
echo "<Process-> Sorting on SLNo & SP keys in T5 results T6"
sort -n --key=2 --key=3 T5 > T6
echo " "
echo "<Process-> Concatenating header on top of T6 results
SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile"
cat TempHdr1 T6 > SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile
```

```
cp SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile Final
cp SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually Final
echo " "
echo
"#############################################################################################
################"
echo "Now one can manually update SP Index of S.SPS file:
SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually for only entries (i.e. unmapped"
echo "indices)in SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile as per his notion &
understanding. These files are present in /Final Sub folder."
echo
"#############################################################################################
################"
echo " "
else
echo " "
echo "<Process-> Concatenate fields SLNo,SP & Insert in X9 furnishes X14"
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s%1s%1s\n",\
$1," "$2," ",$3," ",$4," ",$5," ",$6," ",$7," ",$6,""$7," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15
}' X9 > X14
echo " "
echo "<Process-> Concatenate fields SLNo,SP & Insert in S4 furnishes S6"
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s%1s%1s%1s\n",\
$1," "$2," ",$3," ",$4," ",$5," ",$6," ",$7," ",$2,""$3," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15
}' S4 > S6
echo " "
echo "<Process-> Join lines of S6 & X14 on a common field i.e.concatenated SLNo & SP results T7"
join -1 8 -2 8 S6 X14 > T7
echo " "
awk -F" " '{printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1
s%1s%1s\n",\
$2," "$3," ",$4," ",$24," ",$6," ",$7," ",$8," ",$9," ",$10," ",$11," ",$12," ",$13," ",$14," ",$15," ",$16 }' T7 > T8
echo "<Process-> Required fields are extracted from T7 & formatted as single space delimited, furnishes T8"
echo " "
awk -F" " '{ printf
"%1s%16s%8s%1i%2s%4i%4.1f%4i%2i%4.1f%9.1f%10.1f%6.1f%3i%6i\n",$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$
11,$12,$13,$14,$15 }' \
T8 > SFileInStandardFmt_SPIndexUpdated
echo "<Process-> Required fields are extracted from T8 & formatted, adhering standard S.SPS structure, results
in SFileInStandardFmt_SPIndexUpdated"
cp SFileInStandardFmt_SPIndexUpdated Final
echo " "
echo
"#############################################################################################
################"
echo " SP Index of S.SPS file are updated from SP Index of X.SPS file. Find the updated file
SFileInStandardFmt_SPIndexUpdated in /Final sub folder"
echo
"#############################################################################################
################"
echo " "
fi
echo "Phase-3 Ends."
echo " "
```

```
echo
"#############################################################################################
#################"
echo "Phase-4 Begins."
echo "Following set of Phase-4 scripts generate a runnable R.SPS"
echo
"#############################################################################################
#################"
echo " "
echo "<Process-> Sorting on RLNo & RP keys in R.SPS results R1"
sort -n --key=2 --key=3 $R > R1
echo " "
echo "<Process-> Formatting R1, adhering standard R.SPS structure, results in RFileInStandardFmt"
awk -F" " '{ printf
"%1s%16s%8s%1i%2s%4i%4i%4i%2i%4i%9.1f%10.1f%6.1f%3i%6i\n",$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$
12,$13,$14,$15 }' R1 > RFileInStandardFmt
cp RFileInStandardFmt Final
echo "Phase-4 Ends."
echo
"#############################################################################################
#################"
echo "By the end of Phase-4 all the three SPS files have been mapped among themselves and conform to
DataFFID in $FList as well."
echo
"#############################################################################################
#################"
echo " "
echo "Phase-5 Begins."
echo " "
echo
"#############################################################################################
#################"
echo "Phase-5 script maps:"
echo "1. Channel range onto Receiver range for a Receiver Line within X.SPS"
echo "2. Receiver Pkt range for a Receiver Line from R.SPS onto X.SPS"
echo "3. Channel Range for a FFID from Data onto X.SPS"
echo
"#############################################################################################
#################"
echo "This segment of script checks within X.SPS, that, if Total no.of RP & Total no.of Channels for a RL are
equal or not."
echo "In case they are not equal then write anomaly in
AmbiguityInMapping_FromNoOfChannels_OntoNoOfRP_WithinXFile."
echo
"#############################################################################################
#################"
echo " "
echo "<Process-> Computes Total no.of RP & Total no.of Channels for a RL in X8 furnishes X15"
awk -F " " '{printf "%6s%1s%6s%1s%80s\n",1+(($14-$13) / $15) ," ",1+(($10-$9) / $11), " ",$0}' X8 > X15
echo " "
echo "<Process-> Creates a gross view of mapping between Total no.of RP & Total no.of Channels for a RL in
X15 furnishes T9"
awk -F " " 'NR==FNR{a[$2" "$1]=$3;next} {k=$1" "$2; if(k in a) printf
"%1s%1s%1s%1s%1s%1s%1s\n","<For FFID in X.SPS : " $5, ">  No.of Receivers: ",$1,\
" Matching with No.of Channels: ",$2," For the Receiver Line: ",$14 ;else printf
"%1s%1s%1s%1s%1s%1s%1s%1s\n",\
"<For FFID in X.SPS : " $5, ">  No.of Receivers: ",$1," Not Matching with No.of Channels: ",$2," For the
Receiver Line: ",$14,\
"  ***Rectify this in your original X.SPS file & Rerun the script"}' X15 X15 > T9
```

```
echo " "
grep "Not Matching" T9 > AmbiguityInMapping_FromNoOfChannels_OntoNoOfRP_WithinXFile
echo "<Process-> Unmatched Total no.of RP & Total no.Channels in a RL are extracted from T9 and stored in
AmbiguityInMapping_FromNoOfChannels_OntoNoOfRP_WithinXFile"
cp AmbiguityInMapping_FromNoOfChannels_OntoNoOfRP_WithinXFile Ambiguities
echo " "
echo
"#############################################################################################
#################"
echo "This segment of script maps MinMax range of RP in a RL from R.SPS onto X.SPS."
echo "In case they do not map then write anomaly in
AmbiguityInMappingOf_MinMaxRPInRL_FromRFile_OntoXFile."
echo
"#############################################################################################
#################"
echo " "
echo "<Process-> Computes MinMax of RP for a RL in R.SPS and collect in T10"
awk -F " " '{
                        if( NF == 15 )
                        { if(array_min[$2]>$3 || array_min[$2]==""){
                        array_min[$2]=$3;
                        }
                        }
                        { if(array_max[$2]<$3 || array_max[$2]==""){
                        array_max[$2]=$3;
                        }
                        }
                        }
                        END{
                        for(i in array_min) {
                                print i " " array_min[i] " " array_max[i]
                                }
                                }' $R > T10
echo " "
echo "<Process-> Computes MinMax of RP for a RL in X8 and collect in T11"
awk -F " " '{
                        if( NF == 15 )
                        { if(array_min[$12]>$13 || array_min[$12]==""){
                        array_min[$12]=$13;
                        }
                        }
                        { if(array_max[$12]<$14 || array_max[$12]==""){
                        array_max[$12]=$14;
                        }
                        }
                        }
                        END{
                        for(i in array_min) {
                                print i " " array_min[i] " " array_max[i]
                                }
                                }' X8 > T11
echo " "
echo "<Process-> Join lines of T10 & T11 on a common field RLNo results T12. Basically T12 is a collection
of MinMax Range of RP in a RL for R.SPS & X.SPS"
join -1 1 -2 1 T10  T11 > T12
echo " "
echo "<Process-> Maps MinMax Range of RP for a RL from R.SPS onto X.SPS."
awk -F " " 'BEGIN { OFS=" "} NR==FNR && a[$1" "$2" "$3]=$4 {} NR>FNR { k= $1" "$4" "$5; if(k in a)
printf "%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s\n",\
```

```
"OK     :> X.SPS & R.SPS map for RL: [",$1,"], As corresponding Receiver Picket Range in X.SPS: [",$4,"-
",$5,"] \
is mapping onto R.SPS Picket Range : [",$2,"-",$3,"]"; else printf
"%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s%1s\n","Problem:> \
XPS & RPS do not map for RL: [",$1,"], As corresponding Receiver Picket Range in XPS: [",$4,"-",$5,"] is not
mapping onto R.SPS Picket Range : [",$2,"-",$3,"] \
****Rectify this anomaly in original R.SPS or X.SPS file using Observer Log & rerun the script"}' T12 T12 >
T13
echo " "
grep "XPS & RPS do not map" T13 > AmbiguityInMappingOf_MinMaxRPInRL_FromRFile_OntoXFile
echo "<Process-> MinMax Range of RP in a RL which does not map from X.SPS onto R.SPS are extracted
from T13 and collected in AmbiguityInMappingOf_MinMaxRPInRL_FromRFile_OntoXFile"
cp AmbiguityInMappingOf_MinMaxRPInRL_FromRFile_OntoXFile Ambiguities
echo " "
echo
"#############################################################################################
################"
echo "This segment of script maps Channel range for a FFID from Data extracted $CSList onto X.SPS."
echo
"#############################################################################################
################"
echo " "
echo "<Process-> Computes Min of FrCh & Max of ToCh for a FFID from X8 & stores in T14"
awk -F " " '{
                        if( NF == 15 )
                        { if(array_min[$3]>$9 || array_min[$3]==""){
                        array_min[$3]=$9;
                        }
                        }
                        { if(array_max[$3]<$10 || array_max[$3]==""){
                        array_max[$3]=$10;
                        }
                        }
                        }
                        END{
                        for(i in array_min) {
                                print i " " array_min[i] " " array_max[i]
                                }
                                }' X8 > T14
echo " "
echo "<Process-> Sorting on FFID key in T14 furnishes T15"
sort -n --key=1 T14 > T15
echo " "
echo "Enter No. of Receiver Lines per Shot: "
read NRL
echo $NRL
echo " "
echo "Enter Total No. of Channels including Auxillarry Channels in the Data corresponding to a FFID or SP: "
read TNChans
echo $TNChans
echo " "
echo "Enter No. of Auxillarry Channels in Data: "
read NAuxChans
echo $NAuxChans
echo " "
echo "If Auxillarry Channels are kept in the Beginning of the Receiver cable i.e. their absolute nos. are 1,2,
..etc."
echo "Then Enter S : "
echo " "
```

```
echo "Else If Auxillarry Channels are kept in the End of the Receiver cable i.e. their absolute nos. are
..2881,2882 etc."
echo "Then Enter E :"
echo " "
read D
echo $D
echo " "
echo
"#############################################################################################
#################"
echo "Doc> Now User need to have a file which lists FFID & Channel statistics i.e.: Ffid, Minimum Channel
No., Maximum Channel No."
echo "Doc> It may or may not have Header Lines in the beginning and Footer Lines after the valid Listing of
Data in this context"
echo "Doc> Valid Listing of Data is the part of Channel Statistics File, wherein, FFID,Minimum Channel
No.,Maximum Channel No.appear."

echo "Doc> An excerpt of this type of file is as follows for your reference to answer the prompts"
echo "       "
echo "Doc>
_____________________________________________________________________________________________
_____________"
echo "Doc>| Header Line1                                                                    |"
echo "Doc>| Header Line2                                                                    |"
echo "Doc>| Header Line3                                                                    |"
echo "Doc>| Q                  0.    1.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0.    2.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0.    3.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0.    4.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0.    5.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0.    6.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0.  793.     1.00    2870.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0.  794.     1.00    2870.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0. 1003.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Q                  0. 5009.     1.00    2882.00    0.00 0.0000    0.0000    0.0000 |"
echo "Doc>| Footer Line1                                                                    |"
echo "Doc>| Footer Line2                                                                    |"
echo "Doc>| Footer Line3                                                                    |"
echo "Doc>
_____________________________________________________________________________________________
_____________"
echo " "
echo "Doc> In the above example No. of Lines in the Header = 3"
echo "Doc>                No. of Last Line upto which valid data exists in this context = 13"
echo "Doc>                Field No. of FFID = 3"
echo "Doc>                Field No. of Minimum Channel = 4"
echo "Doc>                Field No. of Maximum Channel = 5"
echo " "
echo "Doc> Use above help to answer the prompts in your case"
echo
"#############################################################################################
#################"
echo " "
echo "Enter Channel Statistics file name with extension: "
read CSList
echo $CSList
echo " "
echo "Enter No. of Lines in the Header of $CSList file: "
read nlHdr_CSList
```

```
echo $nlHdr_CSList
echo " "
echo "Enter No. of Last Line in $CSList file uptill which valid data exists in this context : "
read nlD_CSList
echo $nlD_CSList
echo " "
echo "Enter Field No. of FFID in the $CSList file: "
read fnF_CSList
echo $fnF_CSList
echo " "
echo "Enter Field No. of Minimum Channel in the $CSList file: "
read fnMinC_CSList
echo $fnMinC_CSList
echo " "
echo "Enter Field No. of Maximum Channel in the $CSList file: "
read fnMaxC_CSList
echo $fnMaxC_CSList
echo " "
echo "<Process-> Collects User prompted values in UP_CSList."
echo "Aux Chans " $NAuxChans "NRL " $NRL "AuxChanSide " $D "TNChans " $TNChans "nlHdr_CSList "
$nlHdr_CSList \
"fnF_CSList " $fnF_CSList "fnMinC_CSList " $fnMinC_CSList "fnMaxC_CSList " $fnMaxC_CSList
"nlD_CSList " $nlD_CSList> UP_CSList
echo " "
echo "<Process-> Concatenates UP_CSList on top of $CSList results in CSList1"
cat UP_CSList $CSList > CSList1
echo " "
echo "<Process-> Gathers User prompted values from the first row & Channel Statistics from next row onwards
of CSList1 and print to CSList2"
awk -F " " 'NR ==1{
                        AuxChan = $3
                        NRL = $5
                        D = $7
                        TNChans = $9
                        nlHdr_CSList = $11
                        fnF_CSList = $13
                        fnMinC_CSList = $15
                        fnMaxC_CSList = $17
                        nlD_CSList = $19
                        next}
                        NR > nlHdr_CSList+1 && NR <= nlD_CSList+1{print AuxChan " " NRL " " D " " TNChans
" " nlHdr_CSList " " $fnF_CSList " " $fnMinC_CSList " " $fnMaxC_CSList
                        }' CSList1 > CSList2
echo " "
echo "<Process-> Reorders all fields of CSList2, format them to Integer type & print to CSList3"
awk -F " " '{printf "%9i%1s%8i%1s%8i%1s%8i%1s%8i%1s%8s%1s%8i%1s%8i\n",$6," ",$7," ",$8," ",$1,"
",$2," ",$3," ",$4," ",$5}' CSList2 > CSList3
echo " "
echo "<Process-> Creates Header"
echo "Ffid        FSeisChan     LSeisChan     AuxChan       NoOfChanPerRL"> TempHdr2
echo " "
echo "<Process-> Computes No.of Seismic Channels per RL corresponding to a FFID in CSList3 & prints
FFID, FSeisChan, LSeisChan, AuxChan, NoOfChanPerRL to CSList4"
awk -F " " '{if($6 == "E"){printf "%-13s%1s%-13s%1s%-13s%1s%-13s%1s%-13s\n",$1," ",$2," ",($3-$4),"
",$4," ",(($3-$4)/$5)} \
          else if($6 == "S"){printf "%-13s%1s%-13s%1s%-13s%1s%-13s%1s%-13s\n",$1," ",($2+$4)," ",$3,"
",$4," ",(($3-$4)/$5)} \
                }' CSList3 > CSList4
echo " "
```

```
echo "<Process-> Concatenates TempHdr2 on top of CSList4 results in CSList5"
cat TempHdr2 CSList4 > CSList5
echo " "
echo "<Process-> Maps FFIDs from RestrictDataFFID onto CSList4, if they map then print only FFID else print
entire row of CSList4 in CSList6"
awk -F " " 'BEGIN { OFS=" "} NR==FNR && a[$1]=$2 {} NR>FNR { k= $1; if(k in a) print k; else print $0}'
\
RestrictDataFFID CSList4 > CSList6
echo " "
echo "<Process-> Extracts Channel statistics for X.SPS mapped Data FFIDs from CSList6 and print to CSList7"
awk -F " " '{if(NF > 1) {printf "%1s%1s%1s%1s%1s%1s%1s%1s%1s\n",$1," ",$2," ",$3," ",$4," ",$5}}'
CSList6 > CSList7
echo " "
echo "<Process-> Computes MinOfFirstSeisChan & print Data mapped FFID, FirstSeisChan, LastSeisChan,
MinOfFirstSeisChan to CSList8"
awk -F " " 'NR ==1{
                    FMinChan = $2
                    print $1 " " $2 " " $3 " " FMinChan
                    next}
                    {if($2<=FMinChan){FMinChan=$2;print $1 " " $2 " " $3 " " FMinChan}
                     else if ($2>FMinChan){$4=FMinChan;print $1 " " $2 " " $3 " " $4}
                    }' CSList7 > CSList8
echo " "
echo "<Process-> Computes MaxOfLastSeisChan & print Data mapped FFID, FirstSeisChan, LastSeisChan,
MinOfFirstSeisChan, MaxOfLastSeisChan to CSList9"
awk -F " " 'NR ==1{
                    LMaxChan = $3
                    print $1 " " $2 " " $3 " " $4 " " LMaxChan
                    next}
                    {if($3>=LMaxChan){LMaxChan=$3;print $1 " " $2 " " $3 " " $4 " " LMaxChan}
                     else if ($3<LMaxChan){$5=LMaxChan;print $1 " " $2 " " $3 " " $4 " " $5}
                    }' CSList8 > CSList9
echo " "
awk -F " " 'BEGIN { OFS=" "} NR==FNR && a[$1" "$2" "$3]=$6 {} NR>FNR { k= $1" "$4" "$5; if(k in a)
print k; else print $0}' \
CSList9 CSList9> CSList10
echo "<Process-> Hash Key fields: FFID, FirstSeisChan, LastSeisChan are mapped on to fields: FFID,
MinOfFirstSeisChan, MaxOfLastSeisChan."
echo "          If they map then print 3 fields: FFID, FirstSeisChan, LastSeisChan to CSList10"
echo "          Else, print 5 fields: FFID, FirstSeisChan, LastSeisChan, MinOfFirstSeisChan,
MaxOfLastSeisChan to CSList10"
echo " "
echo "<Process-> Extracts FFID, FirstSeisChan, LastSeisChan for unmapped cases in CSList10 and print to
CSList11"
awk -F " " '{if(NF > 3) {printf "%-12s%1s%-12s%1s%-12s\n",$1," ",$2," ",$3}}' \
CSList10 > CSList11
echo " "
echo
"#############################################################################################
###" > TempHdr3
echo "Important:In general, Channel Ranges corresponding to FFIDs are mapped from Data onto X.SPS," >>
TempHdr3
echo "yet some Channel Ranges could not be mapped and they are needed to be restricted in data.   " >>
TempHdr3
echo "So, Select following Channel Range only for the corresponding FFID in your Data while actual" >>
TempHdr3
echo "Geometry Merging with Data is carried out.                                    " >> TempHdr3
echo
"#############################################################################################
```

```
###" >> TempHdr3
echo "<Process-> Creating Header TempHdr3"
echo "FFid        FromChannel  ToChannel   " >> TempHdr3
echo " "
echo "<Process-> Concatenating TempHdr3 on top of CSList11 and print to RestrictChannelRangeInDataFFID"
cat TempHdr3 CSList11 > RestrictChannelRangeInDataFFID
echo " "
cp RestrictChannelRangeInDataFFID Final
rm -rf TemporaryFiles
echo "<Process-> Removing TemporaryFiles Folder"
echo " "
mkdir TemporaryFiles
echo "<Process-> Creating TemporaryFiles Folder"
echo " "
mv Ambiguity* CSL* FL* R1* Re* RF* S? SF* T T? T?? TempH* UP* X? X?? XF* TemporaryFiles/
echo "<Process-> Garbage Collection: Moving temporary files in /TemporaryFiles Folder"
echo " "
echo
"#############################################################################################
#################"
echo " "
echo "1.  Your Finalized files are output in /Final & /Ambiguities sub folder."
echo "2a. If you have chosen Automatic SPIndex updation for S.SPS complying with X.SPS, then /Final
subfolder contains five files:"
echo "    Finalized SPS files: XFileInStandardFmt, RFileInStandardFmt, SFileInStandardFmt_SPIndexUpdated"
echo "    & Restriction files : RestrictDataFFID & RestrictChannelRangeInDataFFID. Use them while Geometry
Merging with actual Seismic Data is carried out."
echo
"#############################################################################################
#################"
echo "2b. If you have chosen Manual SPIndex updation for S.SPS, then /Final subfolder contains six files:"
echo "    Finalized SPS files: XFileInStandardFmt, RFileInStandardFmt,
SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually"
echo "    But you need to manually update SPIndex in
SFileInStandardFmt_SPIndexNotMatched_ToBeUpdatedManually for only entries in"
echo "    SFileEntriesWhoseSPIndexDoNotMatchWithSPIndexOfXFile."
echo "    & Restriction files : RestrictDataFFID & RestrictChannelRangeInDataFFID. Use them while Geometry
Merging with actual Seismic Data is carried out."
echo
"#############################################################################################
#################"
echo "3. In your /Ambiguities sub folder, you have two files:
AmbiguityInMappingOf_MinMaxRPInRL_FromRFile_OntoXFile & "
echo "    AmbiguityInMapping_FromNoOfChannels_OntoNoOfRP_WithinXFile and if they contain data then
you have to analyse the R.SPS & X.SPS files for"
echo "    the depicted problem hints in these ambiguity files & correct for them in respective SPS files using
observer log or any other info and "
echo "    rerun the script. RP & RL stand for Receiver Picket & Receiver Line respectively. File names are self
explanatory."
echo
"#############################################################################################
#################"
echo "4. You must sort the Seismic Data available in SEGD, SEGY or System's Internal Format on two keys:
FFID & ChannelNo. before you begin to merge"
echo "    Geometry with Data."
echo " "
echo "Author at anil_semwal@hotmail.com admires your patience during this exercise and ensures that you will
have accurate results in the end."
echo " "
```

```
echo
"#############################################################################################
#################"
```

IV. Conclusion

An effort is attempted by coding a script to get away from drudgery, which is incumbent otherwise, while SPS information is updated in data trace headers.

Acknowledgments

I thank the management of Pan India Consultants, Pvt. Ltd. to facilitate and inspire me during this work.

References

[1]. SEG Technical Standards Committee on Ancillary Data Formats, 1995, Shell Processing Support Format For Land 3D Surveys: Geophysics, 60, No.2, 596-610.