ADCP Perl Tools for UNIX Systems
V2.0

A.M. Thurnherr

December 6, 2017

This document contains incomplete documentation of a set of utilities for processing TRDI ADCP data. In addition to the general purpose utilities described here, the software distribution contains a few special purpose utilities, as well as libraries used for the utilities as well as other software. The software is written in perl and assumed to run in a UNIX environment. This software can be freely used and copied for educational or other not-for-profit purposes.

Most utilities produce output files in a whitespace-delimited ASCII file format called the ANTS format. The "#" character is used for comments and metadata header lines; the string `nan` is used to indicate missing values. ANTS files can easily be read by many software packages, possibly after manually removing the headers. The file layout (association of field names with data columns) is defined by the last header line beginning with `#ANTS#FIELDS#`. Header lines beginning with `#ANTS#PARAMS#` define meta-data parameters. The `Matlab` script `loadANTS.m` can be used to load ANTS files, including metadata, into Matlab.

Contents

1 PD02grd – create GMT grid files from ADCP data

```
Usage: PD02grd [-M)agnetic <declination>]
               [-r)ange <first_ens,last_ens>]
               [output -b)ase <name>]
               [-d)imensional coordinates]
               <PD0 file>
```

The PD02grd utility creates 2-dimensional grid files compatible with the Generic Mapping Tools from TRDI ADCP data. A total of 16 grid files are created with the following data:

`u.grd`, `v.grd`, `w.grd` Zonal, meridional and vertical velocity components, respectively.

`e.grd` TRDI error velocity (inconsistency between two separate w estimates).

`ea1.grd` `ea2.grd` `ea3.grd` `ea4.grd` Per-beam echo amplitudes.

`corr1.grd` `corr2.grd` `corr3.grd` `corr4.grd` Per-beam correlations.

`pcg1.grd` `pcg2.grd` `pcg3.grd` `pcg4.grd` Per-beam echo amplitudes.

2 editPD0 – edit TRDI PD0 files

```
Usage: editPD0 -e) <edit-file> | -x) <expr>
               [-k)eep original data-source id]
               <input file> <output file>
```

The editPD0 utility implements a set of edit operations for TRDI PD0 files. The editing instructions can be provided either in an editing file (-e option; primarily for ensemble-specific editing), or with the -x option on the command line (only for editing applied to all ensembles). The following editing functions are supported:

```
#   - Data-Editing Library:
#       p(<pitch>)              set pitch value (RDI not gimbal pitch) of current ensemble
#       r(<roll>)               set roll alue value of current ensemble
#       h(<heading>)            set heading alue value of current ensemble
#
#       swap_beams(<b1>,<b2>)   swap data from beams b1 and b2
#                                   - input in beam coords required
#                                   - beam rotation is equivalent to 3 consecutive beam swaps
#                                   - basic BT data are swapped as well (not RL and not SIGNAL_STRENG
#
#       earth2beam()            transform beam to earth coordinates
#                                   - does not handle bin-remapping
#                                   - input in earth coords required
#
#       beam2earth()            transform earth to beam coordinates
#                                   - does not handle bin-remapping
#                                   - input in beam coords required
#
#       instrument2beam()       transform instrument to earth coordinates
#                                   - does not handle bin-remapping
#                                   - input in instrument coords required
#
#       ensure_UL()             correct data for wrong transducer orientation
#       ensure_DL()                 - sets correct flag & negates roll value
#
#       dealias(<WV lim[m/s]>)  correct data for erroneously low WV setting
#                                   - HEURISTIC, i.e. may not work
```

Any of these functions can be used directly on the command line. The following syntax is used for edit files:

```
#   - Edit File Syntax:
#       - # comments ignored
#       - empty lines ignored
#       - [space] <ensemble-number|*> <space> <perl-expr>
#       - Examples:
#           162     p(3), r(4), h(3.14)
```

3 listBT – extract bottom-track data

```
Usage: listBT [use -b)ins <1st,last>]
              [write -R)aw data] [write -B)T data] [write -E)nsembles <pref>]
              [-F)ilter ensembles <script>]
              [-C)ompass correction <amp/phase/bias>]
              [-w) <max-diff|0.03>] [-a)mp <min|30>] [-e)rr-vel <max|0.05>] [-c)orrelation <min|220>]
              [-W)ater <depth> [allowed -d)epth-diff <maxdiff|20>]]
              [-f)orce (no setup tests)]
              [-M)agnetic <declination>]
              <RDI file>
```

The `listBT` utility extracts bottom-track data from TRDI PD0 files.

4 listBins – create per-bin time series

```
Usage: listBins [-r)ange <first_ens,last_ens>]
                [-R)enumber ensembles]
                [-o)utput <redirection[>bin%d.raw]>]
                [output -a)ll ens (not just those with good vels)]
                [-M)agnetic <declination>]
                [-S)oundspeed correction <salin|*,temp|*,depth|*>
                [-P)itch/Roll <bias/bias>]
                [-B)eamvel <bias/bias/bias/bias>]
                [require -4)-beam solutions]
                [-d)iscard <beam#>]
                [-p)ct-good <min>]
                <RDI file>U
```

The `listBins` utility extracts per-bin time series from a PD0 file.

5 listEns – list ensemble information

```
Usage: listEns  [-A)nts]
                [-Q)uiet (errcheck only)]
                [-f)ields <[name=]FIELD[,...]>]
                [require -4)-beam solutions]
                [-d)iscard <beam#>]
                [write -E)nsemples <.suff>
                [use -B)T]
                [-M)agnetic <declination>]
                [min -p)ercent-good <#>]
                [keep -b)eam coords]]
                [-r)ange <first_ens,last_ens>]
                [in-w)ater ensembles only]
                <RDI file...>
```

The `listEns` utility lists ensemble data from a PD0 file. The following shows an excerpt from the beginning of a LADCP downlooker profile (the transducer `XD` orientation is downward) when the ADCP enters the water (the number of valid velocities `#vv` jumps from 0 in air to 4 in water). Note also how the ping intervals alternate between 1.3 s and 1.6 s (staggered pinging). `DSID` and `ESW` denote data source id and error status work, respectively.

```
    # Date       Time         XD  Temp Headng Pitch  Roll #vv DSID ESW
-------------------------------------------------------------------------
  828 07/28/2017 05:12:37.95 DN  17.0  144.7  -0.6   1.9   0 0x7F 0x00000000
  829 07/28/2017 05:12:39.53 DN  17.0  149.4  -0.8   1.3   0 0x7F 0x00000000
  830 07/28/2017 05:12:40.87 DN  16.9  158.3  -0.8   1.2   0 0x7F 0x00000000
  831 07/28/2017 05:12:42.44 DN  16.9  171.4  -3.5   9.6   4 0x7F 0x00000000
  832 07/28/2017 05:12:43.78 DN  17.0  164.7  -2.3   7.0   4 0x7F 0x00000000
  833 07/28/2017 05:12:45.35 DN  17.0  152.3  -4.5   7.5   4 0x7F 0x00000000
  834 07/28/2017 05:12:46.68 DN  16.9  151.0  -2.7   6.6   4 0x7F 0x00000000
```

6 listHdr – list header information

```
Usage: listHdr [-s)ummary]
           [-l)ong listing]
               <PD0 file[...]>
```

The `listHdr` utility lists header data from one or more PD0 files. The following is the default output (without the `-l` option) of an example file:

```
097DL000.000:
    Instrument Characteristics:
        PRODUCER                        = TRDI ADCP
        INSTRUMENT                      = Workhorse #24544
        FIRMWARE                        = 50.41
        BEAM_FREQUENCY                  = 153.6 kHz
        SPEED_OF_SOUND                  = 1500 m/s [from settings]
        AMBIGUITY_VELOCITY              = 3.91052 m/s
        Sensors                         : PRESSURE TEMPERATURE COMPASS PITCH ROLL
    Coordinate System:
        Flags                           : BEAM_COORDINATES PITCH_AND_ROLL_USED
    Bin Setup:
        N_BINS                          = 25
        BLANKING_DISTANCE               = 0 m
        DISTANCE_TO_BIN1_CENTER         = 8.37 m
        BIN_LENGTH                      = 8 m
        TRANSMITTED_PULSE_LENGTH        = 8.29 m
    Water-Track Setup:
        PINGS_PER_ENSEMBLE              = 1
        TIME_BETWEEN_PINGS              = 0 s
        TRANSMIT_POWER                  = 255
        MIN_CORRELATION                 = 64
        MIN_PERCENT_GOOD                = 0 %
        MAX_ERROR_VELOCITY              = 2 m/s
        FALSE_TARGET_THRESHOLD          = 50
        Flags                           : NARROW_BANDWIDTH TRANSMIT_POWER_HIGH
```

7 listW – extract vertical-velocity time series

```
Usage: listW [-A)nts]
             [-F)ilter <script>]
             [bin -r)ange  <bin|0,bin|*>] [-e)rr-vel <max|0.1>] [-c)orrelation <min|70>]
             [-S)alin <val|35>]
             [-t)emp <bias>]
             [output -f)ields <field[,...]>
             <RDI file>
```

The `listW` utility dumps the vertical velocities from a PD0 file.

8 meanProf – create mean profile from moored data

```
Usage: meanProf [-r)ange <first_ens,last_ens>]
                [-Q)uiet (stats only)]
                [-S)oundspeed correction <salin|*,temp|*,depth|*>
                [require -4)-beam solutions]
                [-d)iscard <beam#>]
                [-%)good <min>]
                [output -b)eam coordinates]
                [-M)agnetic <declination>]
                [-D)epth <depth>]
                <RDI file>
```

The `meanProf` utility creates a mean velocity profile from a PD0 file. This is mainly useful for moored records.

9 `mkProfile` – create time series of depth from LADCP data

```
Usage: mkProfile [-Q)uiet]
                 [-F)ilter <script>]
                 [require -4)-beam solutions]
                 [-d)iscard <beam#>]
                 [apply beamvel-m)ask <file>]
                 [-r)ef-layer <bin|1,bin|6>]
                 [-n) vels <min|2>]
                 [-e)rr-vel <max[0.1]] [-c)orrelation <min>] [-p)ct-good <min[100]>]
                 [max -g)ap <len>]
                 [output -f)ields <field[,...]>
                 [-M)agnetic <declination>]
                 [profile -B)ottom <depth>]
                 <RDI file>
```

The `mkProfile` utility takes a PD0 file from a LADCP cast as input and creates a time series of estimated depth by time-integrating the reference-layer vertical LADCP velocities. This allows for a quick test of the LADCP data quality, as the depth at the bottom (`zmax`) and at the end of the cast (`zend`) have to agree approximately with the maximum CTD depth and the CTD depth at the end of the cast (zero, presumably). Note that the agreement has to be only approximate with the errors depending significantly on the sea state — for deep casts in rough seas, agreement within 100–200 m is acceptable.

By default, the time series is written to `STDOUT` and a set of profile statistics is written to `STDERR`. Output of the time series can be suppressed with the `-Q` option. The following example output is from a profile with good data but with a bad beam:

```
Reading 097DL000.000...done
# of ensembles        : 12275
Ping intervals        : 1.3s/1.6s (1.3s-1.5s/1.4s-1.6s)
WARNING: long-ish w gap (dt=40.6s)
WARNING: long-ish w gap (dt=53.8s)
WARNING: long-ish w gap (dt=92.9s)
Start of cast         : 05:06:29.11 (#  574) at    0.0m
Bottom of cast (zmax) : 07:05:51.00 (# 5507) at 5824.1m
Seabed                :                      at 5835.3m (+-1m)
End of cast (zend)    : 09:41:24.91 (#11936) at  -40.9m
Cast Duration         : 4.6 hours (pinging for 5.0 hours)
Minimum range         : 8m at ensemble 780, beam 2
80%-valid bins        : 9.0
80%-valid range       : 72m
3-beam solutions      : 328 47 47546 160
net rotations         : [0]/-11/19/[0]
rms pitch/roll        : 2.8deg/1.7deg
rms heave acceleration: 0.22m/s^2
```

Notes:

1. A staggered pinging rate, alternating 1.3 s and 1.6 s intervals, was used.

2. The warnings indicate several gaps in the time series of reference-layer w, probably while the instrument was jerked around near the sea surface. They can be ignored because `zmax` and `zend` are correct.

3. The profile was detected correctly. Start, bottom and end times are correct. The values of `zmax` and `zend` are reasonable and consistent with the CTD measurements. This indicates that the LADCP data are of good quality.

4. Minimum instrument range during the profile was not great but likely sufficient for processing.

5. Beam #3 has many more 3-beam solutions than the other three beams, indicating that the beam is either weak or broken.

6. The rosette executed 11 counterclockwise rotations on the downcast and 19 clockwise rotations on the upcasts.

7. The rosette was ballasted nicely, resulting in small tilts.

8. The sea state was neither particularly rough (rms heave acceleration $0.3\,\mathrm{m\cdot s^{-2}}$ or higher) nor calm (rms heave acceleration $0.1\,\mathrm{m\cdot s^{-2}}$ or lower.

10 patchPD0 – patch PD0 file with external attitude data

```
patchPD0 -- patch TRDI PD0 file with external attitude data

Options & Arguments:
        [patch -p)itch] [-r)oll] [-h)eading] (none patches all)
        [-o) <heading-offset>] [-k)eep velocities of unpatched ensembles]
        [keep original -d)ata-source id]
        <original PD0 file> <patched PD0 file> [external attitude file]
```

The `patchPD0` utility combines the data from a PD0 input file and a time series file with replacement attitude measurements (typically from an external IMU), creating a patched PD0 file. The file with the replacement attitude values must contain meta data in the header, as well as exactly one record for each ensemble in the PD0 input file. The replacement attitude data must be in the ADCP reference frame. The input file must contain the following three meta-data variables:

`LADCP_pitch.mu` Profile-averaged LADCP *gimbal pitch,* not including the pre- and post-cast on-deck measurements. The start and end ensembles of a profile can be found with the `mkProfile` utility. Refer to the TRDI manuals to calculate gimbal pitch from the pitch/roll measurements.

`LADCP_roll.mu` Profile-averaged LADCP roll, not including the pre- and post-cast on-deck measurements. The start and end ensembles of a profile can be found with the `mkProfile` utility.

`IMU_hdg_offset` Clockwise offset (in degrees) of the external attitude sensor with respect to the ADCP. Note that this rotation has already been applied to the IMU data — the value is only used with the `-o` command-line option.

Additionally, the input file requires the following four fields:

`LADCP_ens` Ensemble number in the PD0 file. Note that there must be exactly one IMU record for each ensemble in the PD0 file.

`pitch` Replacement *gimbal pitch anomalies* (deviations from the profile-averaged mean) rotated into the ADCP frame of reference. Again, the pre- and post-cast on-deck measurements must not be included when calculating the averages. The string `nan` is used to mark missing values.

`roll` Replacement roll *anomalies* (deviations from the profile-averaged mean) rotated into the ADCP frame of reference. Again, the pre- and post-cast on-deck measurements must not be included when calculating the averages. The string `nan` is used to mark missing values.

`hdg` Replacement headings from the IMU rotated into the ADCP frame of reference. The string `nan` is used to mark missing values.

The following example shows the start of a valid attitude replacement file. Note that the first three ensembles (#1000–1002) do not have any valid heading data — the patched PD0 file will not have any valid velocities in those ensembles unless the `-k` command-line option is used.

```
#ANTS#PARAMS# LADCP_pitch.mu{2.1}
#ANTS#PARAMS# LADCP_roll.mu{-1.6}
#ANTS#PARAMS# IMU_hdg_offset{-91}
#ANTS#FIELDS# {LADCP_ens} {pitch} {roll} {hdg}
1000    1.42    -1.68   nan
```

```
1001    1.38    -1.73   nan
1002    1.34    -1.69   nan
1003    1.58    -1.62   223.8
1004    1.56    -1.78   223.6
1005    1.34    -1.98   227.8
1006    1.33    -1.96   218.3
1007    1.39    -1.89   221.8
```

By default, the patched PD0 files are marked with different `DATA_SOURCE_ID` values, i.e. they no longer fully conform to the TRDI specifications. (Both the `listEns` and the `listHdr` utilities use this information.) Since most LADCP processing software ignore modified `DATA_SOURCE_ID` values, the patched files can usually be processed without problems. If fully conformant PD0 files are required, however, the `-d` command-line option can be used.

The `-o` command-line option allows overriding the `IMU_hdg_offset` value from the IMU file header. When this option is used, the replacement attitude values are first rotated back into the IMU coordinate system (using `IMU_hdg_offset`) before being rotated into the orientation given by `-o`.

11 splitPD0 – split PD0 file at ensemble boundaries

```
Usage: splitPD0 [-o)ut-file <fmt[e.g. 017DL_%02d.000]>]
                [-f)irst <cast #>]
                [require -m)in <ens> to produce output]
                <RDI file> <ens> <ens[...]>
```

The `splitPD0` utility splits a PD0 file into several parts. Notes:

1. The utility is primarily useful for extracting individual down-up-casts from LADCP yo-yo and tow-yo profiles, but it can also be used to remove invalid on-deck data from the beginning or end of a data file, to split a data file into smaller pieces for emailing etc.
2. The ensemble numbers at which the file is to be split are supplied as command-line arguments.
3. Each output file is a self-contained PD0 file.
4. The original file can be reconstituted by concatenating the split PD0 files, e.g. with the UNIX `cat` tool.