Marine seismology data/metadata open issues

1 General

1.1 Unified software proposal

Propose a unified (or at least validated) software as goal for the next cycle (IUGG 2027?)

Joel Simon recommends a formal proposal (what is needed?) preceded by a request for current users to log bugs, quirks or expert know-how in a less formal way (shared wiki?)

- Create FDSN "Marine Seismology Software" github site with lists of software and mechanism for people to submit their experiences and recommendations
- Activate wiki so that people can comment there
- Advertise through multiple outlets: MSROC commitee meeting, obsictech website, everybody's OBS group puts on a link, Earthquake.

1.2 Companies that correct leap seconds

Create "Dataloggers" document indicating which dataloggers already do some of the work outlined in the standards/software documents (leapseconds, drift correction...) and stating whether their solution conforms to our recommendations..

2 Metadata-specific (StationXML)

2.1 StationXML-standardized element

Currently encoded as StationXML `<Comment>`s, but there was interest in using an alternative namespace, which should allow a clearer representation. This was not possible in earlier versions of obspy, but this (unverified) obspy/obspy#3588 PR claims to resolve the issue. Should this be used in the future?

2.2 Companies that output StationXML metadata

Should add a table of companies that output StationXML metadata, and information on which standards/recommendations they do not yet respectNew version of msmod

As of August 2025, additions to msmod have been written (but not tested), allowing direct clock drift corrections for piecewise linear, polynomial, or cubic spline estimates of the clock drift.

A further, requested modification, should allow leap seconds to be inserted in miniseed files using a single call.

3 Data-specific (miniSEED)

3.1 msmod

Verify that the additions to msmod work and do not interfere with msmod'ss functioning (in cooperation with Chad Trabant)

Coordinate/fund inclusion of these modifications in the latest version of msmod

Add one-line leap-second modification to msmod.

3.2 How to apply clock drift + leapsecond corrections

In the standards document, we recommend a method in which one corrects the clock drift first. For completeness (and possible further debate), we detail the three proposed methods here.

3.2.1 Correcting clock drift first

- Apply the drift correction
- Then apply the leap-second correction.

This avoids the possibility that the record containing the leap-second will change, which would generate an extra gap and overlap in the data. The offset it will create in the time correction is of the order of the drift rate * 1 second, which will be below 1 microsecond for a drift rate below 1e-6 (< 31 seconds/year)

Note: If you use the data to calculate the drift rate, first generate LEAP-CORRECTED data from the NOT CLOCK CORRECTED DATA. Use this data to calculate the clock drift. Then throw the LEAP-CORRECTED data away.

3.2.2 Correcting leap second first first

- Apply the leap second correction
    - If you use the data to calculate the drift rate, do it here
- Then apply the clock drift correction.
- Validate that there are no new gaps/overlaps. If there are, move leap-second flags to the proper record(s)

Note: If you use the data to calculate the drift rate, you can do this on the data just after the leap-second correction.

3.2.3 Takeishi Isse proposal

Note by Wayne: This seems to me to be like the "correcting clock drift first" method, but with an explicit method for applying the leap-second correction for non-miniSEED data (Japan-standard WIN format?)

Basic flow is as follows;

1. estimate true drift from measured drift (or change the GPS measurement time to no-leap-seconds clock time)
2. correct observed data based on true drift rate (or no-leap-seconds clock time)
3. Split the time-corrected data into three segments: before_leap_seconds data, on_leaps_seconds data, after_leap_seconds data.
4. Correct the time stamp of on_leaps_seconds data, after_leap_seconds_data.

For example, if leap seconds was applied in 2012-July-1
Make no-leap-second clock drift measurement information

```
#orignal time difference measurement ( + means recorder is in advance)
station_name GPS_time_at_deployment diffrence_in_recorder(ms) GPS_time_at_recovery
differnce_in_recoder(ms)
NM18 2011/11/20 18:11:00 +8 2013/09/03 19:47:50 -883
If leap-seconds is not occurred, GPS time at recovery is 2013/09/03 19:47:51 and clock drift
is -1883 ms
NM18 2011/11/20 18:11:00 +8 2013/09/03 19:47:51 -1883
```

Apply drift correction based on no-leap-seconds time.

Split clock-corrected data into three segments.

```
1st segment record_start - 2012/06/30 23:59:59.9999
2nd segment 2012/07/01 00:00:00.0000 - 2012/07/01 00:00:00.9999
3rd segment 2012/07/01 00:00:01.0000 - record_end
```

correct time stamps to

```
1st segment : no correction
2nd segment : 2012/06/30 23:59:60.0000 - 2012/06/30 23:59:60.9999
3rd segment : 2012/07/01 00:00:00.0000 - record_end-1second
```

In my original protocol, to avoid time stamp correction of 3rd segment, I made with-leap-seconds clock drift measurement

```
(NM18 2011/11/20 18:10:59 +1008 2013/09/03 19:47:50 -883)
```

This is because win format (Japanese standard) adds time stamps in every second.

3.3 Should the miniSEED3 sampling rate be specified as double precision?

The miniSEED3 record header's sample rate word is now 64-bits, which gives a precision of about 16 decimal digits (about 3 nanoseconds in 1 year). This is good enough to specify the true sample rate of the data. For example, if the instrument clock was synchronized with GNSS time at the start, was found to be 1 second later than GNSS time after 365 days, the drift was linear, and the target sample rate was 100 sps, the true sample rate could be written as be 100*(1+1s/86400*365) = 100.000000323654.

Entering this drift-corrected sampling rate could help avoid programs finding false gaps between records/files and could allow plots of long data segments to have times that are as well-aligned at the end as at the start, but could confuse users and present problems for systems that use 32-bit sampling rates.

Jerry Carter (Earthscope) writes : “The advantages far outweigh the disadvantages in my opinion and I don't think we need to look for alternatives. MiniSeed3 will require some re-coding anyway; this is something that should be picked up.”