

Nortek Time Series

Data files for Nortek devices (ADCPs, current meters, velocimeters) are described here. [Daily current plots](#) are also available for some Nortek devices. Parsed scalar ancillary data (e.g., temperature, pitch, roll) are also available as [time series scalar data](#) and [time series scalar plots](#).

[Oceans 3.0 API filter](#): dataProductCode=NTS

Revision History

- 20211006: netCDF file format update: added ancillary data (e.g. Beam Velocity, Correlations, Intensity), apply compression, and change format to netcdf4 classic.
- 20161101: Support for Signature 55 ADCPs
- 20150601: Added bin-mapping option (none, nearest vertical, linear).
- 20150226: Change range to be to the beam centre, add bin mapping to correct velocities and range for tilt. Add all other Nortek raw binary formats (VEC, VNO, AQD, WPR).
- 20150202: Heading correction rework (affecting ENU / uvw velocities): improve handing of mobile sensors, autonomous deployments and better documentation of processing was done to the data.
- 20111019: Initial NetCDF, [CF1.5](#) compliant product released.
- 20110302: Correction to algorithm for obtaining ENU (with respect to true North velocities). If instrument is mobile, a combination of instrument compass and local magnetic declination is used. If instrument is fixed position, the orientation determined at instrument deployment is used.
- 20110110: Initial MAT product released
- 20101130: Initial PRF product released

Formats

This data is available in manufacturers' raw binary data files (**PRF / AQD / VEC / VNO / WPR / AD2CP**), and parsed and processed time series data files (**MAT** and **NETCDF**). Content descriptions are provided below.

To produce these files, the following requirements apply:

- A new file is started at the start of each day, when the maximum records per file is exceeded (86400 for **MAT** and **NETCDF**, 86400 for **PRF / AQD / VEC / VNO / WPR / AD2CP**, or when the driver is restarted (this should account for configuration changes, site changes, etc).
- Only records with valid checksums are included.
- Only records that have a valid Nortek data structure ID, including configuration data structures and structures that may not normally be associated with file extension (Nortek software appears to handle this ok. We don't filter out valid data structures as firmware updates may cause a structure to be used by a different device). List of valid data structures was taken from the [system integrator manual, version Dec. 2014](#) and the [Signature Series System Integrator Guide](#).
- The instrument date/time field is replaced by the ONC timestamp at the beginning of the [log file](#) (since this timestamp is more accurate than the instrument clock).

All Nortek devices are sensitive to orientation. Current meters and velocimeters' data are parsed into [scalar data](#), and part of that process includes a rotation from instrument or Earth relative velocities to true East, North and Up velocities. Data from the manufacturers' raw binary files (**PRF / AQD / VEC / VNO / WPR**) is as is: no rotation or processing is done. Users should use the appropriate post-processing software to rotate the data to correct for orientation. The **MAT** and **NETCDF** processed data products described further below are generated correction for orientation that is described in detail below. If any configurations are changed, files will break. In particular, Nortek Signature 55 ADCP may operate at two different frequencies, so two sets of processed time series products will be generated, while the raw binary data file supports the two frequencies.

Raw Binary Data Files: PRF / AQD / VEC / VNO / WPR / AD2CP

These very similar binary formats are specific to Nortek acoustic devices. When using Nortek data acquisition software, data is normally stored in this way. Although we use custom-built drivers to communicate with our instruments, we can use the raw data in the [log file](#) to produce these files. The aforementioned requirements apply, as well as:

- After replacing the instrument date/time field with the ONC timestamp, the checksum is recalculated.
- For all Nortek devices except Signature series: Instrument hardware, head and user configuration strings are inserted as the first three lines of the each raw binary file and important parameters are parsed into MAT file. However, any data existing previous to accompanying configuration details being saved in the database cannot be interpreted (earlier than Fall 2009). We are actively back-filling the configurations, [contact us](#) or press the help button if your data product request indicates that the configuration does not exist, even though there is data available.
- For Nortek Signature series: the configuration and command information that you'd normally find at the start of an **AD2CP** file are reconstructed from the device configuration which is managed and stored in the device attributes. See here <http://dmas.uvic.ca/DeviceListing?DeviceId=22925> and go the additional attributes for an example. This configuration and command data is presented in the **AD2CP** file as a string data record, as per the [Signature Series System Integrator Guide](#).

VEC is the format for [Nortek Vector Current Meters](#), **VNO** is the format for [Nortek Vectrino Current Meters](#), **AQD** is the format for [Nortek Aquadopp Current Meters](#), **PRF** is the format for Nortek ADCPs, both the [Aquadopp Current Profiler](#) and the [Aquadopp-HR Current Profiler](#), **WPR** is the format for Nortek AWAC (measures currents and waves, see Nortek's description [here](#)), **AD2CP** is the format for [Nortek Signature Series ADCPs](#) (viewable in Signature Viewer is a license key). Go to each of those links for the appropriate manufacturer's software for each. Each type of Nortek device has its own software, however, they all (except Signature Series) have the same data conversion function that appears to read all of the above formats and data structures (however, it's probably best to use the appropriate software). In the appropriate Nortek device interface software, go to the 'Deployment' menu, select 'Data Conversion...', then 'Add file' and then press the arrow. Text file conversions of the raw binary data will be produced. Please note, the raw binary formats are not the .dep files the Nortek device interface software will attempt to load on click of the open file icon or on selecting File -> Open. The .dep files are deployment configuration files useful for field operations. The raw binary files are produced from live devices when the operator selects Deployment -> Recorder Data Retrieval, which is the functionality that our system emulates.

Nortek also offers post-processing software, such as *Surge*, *Prof2NDP*, *bin2mat*, etc., that will also read these formats, see here for downloads (some require license): <http://www.nortek-as.com/en/support/software> The *Surge* software will analyze and plot the velocity data, including doing bin-mapping correction, however it is not free software. The free post-processing option, other than our processed **MAT** and **NetCDF** files and plot products, is a combination of *Prof2NDP* and *ExploreP*. To post-process the data with *Prof2NDP* and *ExploreP*, begin by downloading and installing *Prof2NDP* and *ExploreP* from the [Nortek website](#). Get the **PRF** or other raw binary files from data search, convert the file with *Prof2NDP* and open the resulting .ADP file in *ExploreP*. To perform the co-ordinate transform on the velocity data from beam or instrument co-ordinates, go to Contour Plot -> Plot Param... choose East /West or N/S or Up/Down, then click on the icon that says 'insert profile graph' or 'insert time series graph'. Users can then click on the graph to chose the time or bin of the profile, then export the post-processed velocity data via the file menu.

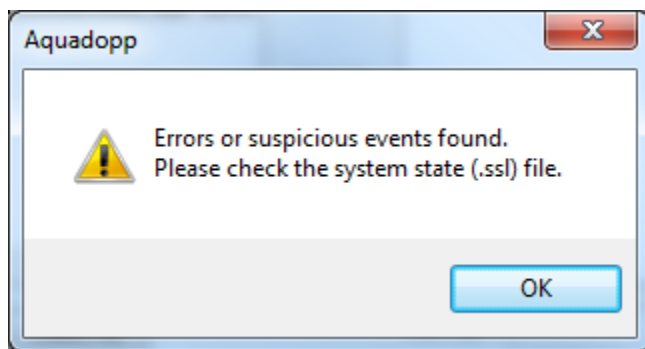
These formats are further described in the manufacturer's [System Integrator Guide](#) and [Signature Series System Integrator Guide](#).

Oceans 3.0 API filter: extension={prf,aqd,vec,vno,wpr,ad2cp}

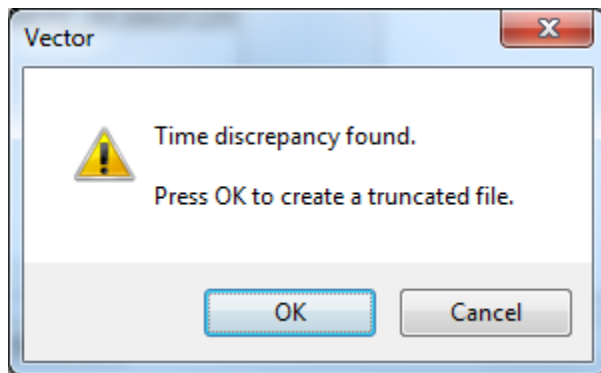
Example: **BH_POD2_AD2M_20101018T160041.704.prf**

Errors and Warnings when Processing Raw Binary Files with Nortek Device Interface Software (Except for Signature Series)

When reading and converting the raw binary data files with Nortek Software, users may encounter some warnings. In our testing, we've come across the following warnings and messages:



In the .ssl file that was generated by *Aquadopp* from the **AQD** file, we saw some warnings about out of range pitch and roll sensor data. This could be caused by a bad configuration or orientation, most likely test data, or this can occur if the orientation of the device is too extreme, as could happen if the device was deployed to a mooring or mobile deployment. In the case of the latter, users may want to ignore or mask out this data.



In the *Vector* software, we often see this warning. It will create a small subset **VEC** file that appears to duplicate data that was processed normally. The processed data appears to be ok; it is not out of order or otherwise scrambled.

[Contact us](#) if you have any questions about these errors.

Processed Time Series Data Files: MAT / NETCDF

Processed time series data files are available for Nortek ADCP-type devices: [Aquadopp Current Profiler](#), [Aquadopp-HR Current Profiler](#), [Nortek Signature Series ADCPs](#). The Nortek current meter type devices are parsed as scalar sensors. The processed data for those devices are available in generic [time series scalar data](#) and [time series scalar plot](#) data products, and from other scalar data portals such as [Plotting Utility](#).

Data Product Options For ADCPs (Ensemble averaging, bin-mapping and plan)

Note that this included page ([Ensemble Averaging, Bin-mapping, Three-beam Solutions and Filtering Options For RDI ADCPs](#)) covers both Nortek and RDI ADCPs

For [Nortek ADCP data file products \(MAT and netCDF formats\)](#)

Ensemble Period: Data not altered (none) ▼
(ping averaging)

Ensemble Period:

- Data not altered (none)
[Oceans 3.0 API filter:](#) dpo_ensemblePeriod=0
- 1 Minute
[Oceans 3.0 API filter:](#) dpo_ensemblePeriod=60
- 10 Minute
[Oceans 3.0 API filter:](#) dpo_ensemblePeriod=600
- 15 Minute
[Oceans 3.0 API filter:](#) dpo_ensemblePeriod=900
- 1 Hour
[Oceans 3.0 API filter:](#) dpo_ensemblePeriod=3600

When selecting any of the ensemble periods, this option will cause the search to perform the standard box-car average resampling on the data. 'Boxes' of time are defined based on the ensemble period, e.g. starting every 15 minutes on the 15s, with the time stamp given as the center of the 'box'. Acoustic pings that occur within that box are averaged and the summary statistics are updated. This process is often called 'ping averaging'. The process uses log scale averaging on the intensity data, which involves backing out the logarithmic scale, compute the weighted average, and then compute the logarithmic scale again. Weighted averages are used when raw files bridge an ensemble period and when the data is already an ensemble or ping average.

New files are started when the maximum records per file is exceeded (usually set to make files that will use less than 1 GB of memory when loaded), or when there is a configuration, device or site changes. In the case where there is data from either side of a configuration change within the one ensemble period, two files will be produced with the same ensemble period, with the same time stamps, but different data. Users may use the ensemble statistics on the number of pings or samples per ensemble to filter out ensembles that do not have enough data. (As an aside, we do this by default with clean averaged scalar data - each ensemble period needs to have at least 70% of it's expected data to be reported as good.)

The default value is no averaging, meaning the data is not altered. This option is only available for **MAT** and **NETCDF** files.

File-name mode field

Selecting an ensemble period will add 'Ensemble' followed by the ensemble period. For example '-Ensemble600s'.

Velocity Bin-mapping (tilt compensation EX)

For all [Nortek](#) ADCP data products (PNG/PNG and MAT and netCDF formats)

Velocity Bin-mapping: ☒ None ☐ Nearest vertical bin ☐ Linear interpolation (Ott method)
(tilt compensation)

This option specifies the bin-mapping processing method to be applied. Bin-mapping is also known as 'depth cell mapping' or 'tilt compensation' or even 'map to vertical'. There are two methods, both correct for tilt effects on ADCP velocity data, while the none option leaves the velocity data as is. For details on the two methods, see the section on [correction and rotation of velocities](#) (included below). The 'None' option is the default for Nortek ADCPs since the free version of the manufacturer's software does not apply bin-mapping (a core goal of our data products is to replicate the functionality offered by the manufacturer's software). The 'Nearest vertical bin' is the default for RDI ADCPs as *winADCP* applies this method for Instrument or Beam co-ordinate data. The 'As configured on the device' option uses the configuration onboard to determine whether to apply bin-mapping, this matches processing on-board the device (for Earth-co-ordinate data, while for Instrument or Beam co-ordinate data *winADCP* ignores the device configuration and always uses 'Nearest vertical bin'). The best method has been shown to be the linear interpolation method ([Ott, 1992](#)).

- Nearest vertical bin
[Oceans 3.0 API filter:](#) dpo_velocityBinmapping=1
- As configured on the device (matches processing on device)
[Oceans 3.0 API filter:](#) dpo_velocityBinmapping=-1
- None
[Oceans 3.0 API filter:](#) dpo_velocityBinmapping=0
- Linear interpolation (Ott method)
[Oceans 3.0 API filter:](#) dpo_velocityBinmapping=2

File-name mode field

The velocity bin-mapping option will be appended to the filename. For example: '-binMapNone', 'binMapLinearInterp', 'binMapNearest'.

Nortek Signature Series Plan

For [Nortek Signature Series ADCPs](#) only, an additional option is appended to the above:

Nortek Signature Series Plan: ☒ Normal PLAN (55 kHz) ☐ Alternate PLAN1 (75 kHz if available) ☐ Both

This option allows the user to select the specific data acquisition plan; there are two: PLAN and PLAN1. PLAN, also denoted as "PLAN0", is generally used for the base frequency (55 kHz), PLAN1 is the alternate (75 kHz). PLAN0 is always available, which is why it is the default option. If the device attribute AlternatePlanEnabled is true, and the DataFormat is DF3 (see the Nortek integrator's guide, DF3 is an internal setting), then PLAN1 data is available and may be accessed by the PLAN1 option. Otherwise requesting PLAN1 may result in no data found. Only appears to Nortek Signature Series ADCPs. When the alternate plan is active, the device will normally operate by alternating between the two frequencies, and the data products will respond by only including the data from the requested plan.

- Normal PLAN (55 kHz)
[Oceans 3.0 API filter](#): `dpo_nortekSignatureSeriesPlan=0`
- Alternate PLAN1 (75 kHz if available)
[Oceans 3.0 API filter](#): `dpo_nortekSignatureSeriesPlan=1`
- All (55 kHz and 75 kHz if available)
[Oceans 3.0 API filter](#): `dpo_nortekSignatureSeriesPlan='both'`

File-name mode field

If the option is applied, a '-PLAN0' or a '-PLAN1' will be appended to the file-name.

Nortek Correlation Screen Threshold

For [Nortek Signature Series ADCPs](#) only, an additional option is appended to the above:

Low Correlation Screen Threshold: ☒ 50% (Nortek default) ☐ 0% (off) ☐ 5% ☐ 10% ☐ 20% ☐ 40% ☐ 60% ☐ 80% ☐ 90% ☐ 95%

This option allows the user to control the correlation screening step. The default value retains the previous behaviour of ONC data products: a threshold of 50%. Beam-velocities that have associated correlation values lower than this threshold are masked / screened to NaN values. Only available on Instrument or Beam co-ordinate data and Signature series ADCPs.

- 50% (Nortek default)
[Oceans 3.0 API filter](#): `corScreen=50`
- Any percentage (integer between 1 and 100):
[Oceans 3.0 API filter](#): `corScreen=<1:100>`
- Off (0%)
[Oceans 3.0 API filter](#): `corScreen=0`

File-name mode field

If a value other than the default is used, a '-corr'<value> will be appended to the file-name, where <value> is the value of the option matching the API filter.

Overall Processing Flow

Velocity data acquired from RDI and Nortek ADCPs can take on several forms. The original, unaltered velocity data, read from the raw files is normally presented in the **MAT** file product as *adcp.velocity* (RDI) or *data.velocity* (Nortek). (Please note that the data structure names for RDI is *adcp* and for Nortek we use *data*, substitute *data* for *adcp* when looking at Nortek data). Also in the **MAT** file product is the config struct that contains information on how the data was collected. Based on the configuration, the original data is processed to present the velocities relative to East-North-Up: this is the final data in the **netCDF**, **MAT** files and in the daily current plot **PNG** and **PDF** (see *adcp.u*, *adcp.v*, *adcp.w* in the **MAT** file and in the MAT file description below). *u*, *v*, *w* velocities are always relative to geographic North and local gravity (for Up).

In general, the aim of ADCP post-processing is to replicate the functionality in the manufacturer's post-processing software: RDI's *winADCP* and Nortek's *Surge* and *Storm* (which replaced *ExploreP*). RDI provides documentation on their rotations, see: [ADCP Coordinate Transformation](#) and the more general guide [Acoustic Doppler Current Profiler: Principles of Operation - A Practical Primer](#) (as a background only). Nortek's rotations are documented online, see: <http://www.nortek-as.com/lib/forum-attachments/coordinate-transformation/view> or the matlab code directly: [post-2-71782-Xform.m](#). See [RDI software support login](#) for RDI software and <http://www.nortek-as.com/en/support/software> for Nortek software.

The overall processing flow is listed below. This sequence is as the code is written. Some of the steps listed here are expanded upon in sections following this one. Right away there is a decision point based on the co-ordinate system: Instrument & Beam versus Earth. The coordinate system parameter in each device's configuration determines the type of data that is acquired and how it is processed to produce u,v,w velocities. In the **MAT** file product, see *config.coordSys*. If the co-ordinate system is set to 'Beam', the original data contains the radial (along-beam) velocities for each of the 4 or 5 beams, otherwise the velocities are rotated to an orthogonal co-ordinate system: 'Instr' (RDI) or 'XYZ' (Nortek) means the original velocities have been rotated to a basis defined by horizontal plane of the instrument with an azimuth relative to the direction of beam 3, while 'Earth' is relative to the onboard magnetic compass or a fixed direction defined by *config.heading_EH* (RDI only). Our default setting is 'Beam' so that the most raw form of the data is collected. 'Ship' is not recognized/used. For RDI ADCPs, when the co-ordinate system is either 'Earth' or 'Instr', the 4th row in the *adcp.velocity* matrix is the error velocity (the measurement error on the velocity at that bin and time).

From Instrument or Beam data:

1. If Instrument co-ordinate data, convert from Instrument to Beam co-ordinate data
2. Exclude bad beams (RDI only, specified by device attribute, forces three-beam solutions to on)
3. Apply correlation screening (RDI only, except for a fixed 50% screen on Nortek Signature55 data, applied to match Nortek's Signature Viewer software, option available)
4. Apply fish detection screening (RDI only, if requested, tunable)
5. Apply bin-mapping (RDI default is 'nearest vertical bin', Nortek default is none, option available)
6. Apply three-beam solutions (RDI only, if requested, option available)
7. Transform from Beam co-ordinate data to Instrument co-ordinate data
8. Rotate Instrument data to true Earth (via a combination of fixed heading/pitch/roll, magnetic declination corrected compass heading and onboard pitch/roll data)
9. Apply error velocity screening (RDI only, option available)
10. Do ensemble averaging if requested (option available)

From Earth data (for RDI only):

1. Do nothing if **RDI** source file has been rotated to true Earth (via a combination of fixed heading/pitch/roll, magnetic declination corrected compass heading and onboard pitch/roll data), otherwise rotate from magnetic Earth to True Earth (via a combination of fixed heading/pitch/roll, magnetic declination corrected compass heading and onboard pitch/roll data)
2. Apply error velocity screening if the requested threshold is more stringent than the screening applied onboard the device (option available)
3. Do ensemble averaging if requested (option available)

The steps above with *option available* (in brackets above) are tuneable by the user as detailed in the data product options below. The *ADCP.processingComments* field in both Nortek and RDI **MAT** file data products provides the user with a log of the decisions made in this processing flow. The **MAT** files (RDI only) also has a structure called *ADCP.processingOptions* that details what options the user chose and what options were actually applied, as some options, such as the error velocity screening threshold, are not always applicable. See the data product options and MAT file format sections below for more information. This is complicated, [contact us](#) for help.

Rotation of Velocities to East-North-Up Co-ordinate System (Earth step #1, Instr/Beam step #8)

RDI sensor source and sensor availability parameters determine if additional sensors are available to be used for rotation, such sensors include: magnetic compass, tilt sensor (for pitch and roll), depth, conductivity and temperature (the last three are used to determine the sound speed, however we normally fix that value, see *config.soundspeed_EC*). In the **MAT** file, see *config.sensorSource_EZ* and *config.sensorAvail*. If the magnetic compass is available, it is often used for Earth co-ordination rotation. The Nortek ADCPs always have compass and tilt sensors and prescribed sound speeds.

Magnetic compasses are not reliable when in close proximity to varying electromagnetic or magnetic fields (electric power cables or the steel instrument platform can cause the compass to be inaccurate). To improve the data, we normally supply a fixed heading from the site and device metadata. If the device is mobile, deployed on a mooring, cabled profiler or glider, a mobile position sensor is normally supplied. Here is an example of a mobile ADCP site: <http://dmas.uvic.ca/Sites?siteId=100206> and here is an example of a fixed site: <http://dmas.uvic.ca/Sites?siteId=1000359>. If the device is mobile and is attached to a device that can supply better orientation information, such as an optical gyro, the data processing code will make use of that device when its sensors are assigned to the heading/pitch/roll of the ADCP's site (we currently do not yet have an example of such a scenario). If neither fixed or mobile heading/pitch/roll are assigned, then the system defaults to the device's internal sensors. When using the onboard magnetic compass, a correction is applied for magnetic declination (if onboard heading is set manually (RDI only), magnetic declination correction is not done). For RDI only, we replicate RDI's gimbal correction for pitch.

Rotation of the input velocities to produce East-North-Up velocities (u,v,w) is performed accounting for the incoming co-ordinate system, types and sources of the data. This is somewhat complicated, therefore, all of the processing steps are documented in the **MAT** files, see *adcp.processingComments*. As an example, here is a common scenario: raw data has been collected in the 'Earth' co-ordinate system defined by the onboard magnetic compass and pitch/roll sensors, but the instrument is stationary with a known fixed heading, then difference between compass heading and the fixed is calculated and used to rotate the East and North velocities. In that case, *adcp.processingComments* would say (use the matlab command char):

```
>> char(adcp.processingComments)
```

adcp.velocity(1:3,,:) contains the unaltered velocity data relative to the co-ordination system: Earth. adcp.velocity(4,,:) is the RDI error velocity. adcp.u/v/w are processed velocities relative to true East/North/Up, respectively.

Using a fixed true heading of 228 degrees from metadata.

This device does not have a fixed value for pitch or pitch sensor assigned; using onboard sensor for pitch (pitch gimbal correction applied).

This device does not have a fixed value for roll or roll sensor assigned; using onboard sensor for roll.

Beam range bins were corrected for tilt by nearest vertical bin-mapping (RDI method) onboard the device (new pitch/roll values not applied). Average or fixed tilt is 3.14 degrees. adcp.range is the vertical range to the corrected bin centres.

Rotated Earth co-ordinate, device-supplied adcp.u/v to true North from onboard magnetic North, using a fixed or variable external source true heading (pitch/roll rotation unchanged).

adcp.backscatter (relative volume backscatter) calculated from $0.45 \cdot \text{adcp.intens} + 20 \cdot \log_{10}(\text{adcp.range}) + 2 \cdot \text{config.soundAbsorptionCoefficient} \cdot \text{adcp.range}$, adcp.meanBackscatter calculated by beam-averaging the volume backscatter, adcp.meanBackscatter and adcp.backscatter have units of relative dB.

For information on the various processing steps applied, see <http://wiki.neptunecanada.ca/display/DP/5>.

For advanced, interested users, here is the core code that does the rotation for the scenario above, rotating Earth co-ordinate data derived on-board the instrument to a fixed heading true Earth co-ordinate data set.

```
if useCompassForHeading
    ADCP.uMagnetic = squeeze(velocityInstrOrEarth(1,:,:)); % east velocity relative to magnetic North
    ADCP.vMagnetic = squeeze(velocityInstrOrEarth(2,:,:)); % north velocity relative to magnetic north
    % rotate to true Earth coords, get magnetic declination at this location
    magdevRadians = -ADCP.magneticDeclination * pi/180; % magnetic declination in radians (negation is
to convert heading to yaw)
    M = [cos(magdevRadians) -sin(magdevRadians); sin(magdevRadians) cos(magdevRadians)]; % standard 2D
rotation
    velocityTrue = M * velocityScreenedReshape(1:2,:);
    ADCP.processingComments(end+1,1) = {'Rotated Earth co-ordinate, device-supplied adcp.u/v to true
North from onboard magnetic North, by applying a magnetic declination correction (pitch/roll rotation
unchanged). '};
else
    % rotate to true Earth coords: get the difference in yaw (transforming heading to yaw first), apply
standard 2D rotation
    % Verification method: grab a bin, calculate the EN vector direction, do the correction, calculate
vector direction, take the
    % difference and it is the same difference as between the heading and the compass. The original
method failed that test.
    velocityTrue = nan(2, n*p);
    for i = 1:p
        yawCorrection = -(heading(min(length(heading),i)) - ADCP.compassHeading(i));
        M = [cosd(yawCorrection) -sind(yawCorrection); sind(yawCorrection) cosd(yawCorrection)]; %
standard 2D rotation
        velocityTrue(:, (i-1)*n+1:i*n) = M * velocityScreenedReshape(1:2, (i-1)*n+1:i*n);
    end
    ADCP.processingComments(end+1,1) = {'Rotated Earth co-ordinate, device-supplied adcp.u/v to true
North from onboard magnetic North, using a fixed or variable external source true heading (pitch/roll rotation
unchanged). '};
end
velocityTrue = reshape(velocityTrue, [2 n p]);
ADCP.u = squeeze(velocityTrue(1,:,:));
ADCP.v = squeeze(velocityTrue(2,:,:));
```

Here's the rotation to true Earth co-ordinates when the original data is in Instrument or Beam co-ordinates. This does not include the transformation from beam to instr (step #7). This is step #8 in the Instr/Beam overall process flow:

```

velocityTrue = nan(3, n*p);
maxNumPosData = max([length(heading) length(pitch) length(roll)]); % either 1 or p, the number of time
stamps
for i = 1:maxNumPosData
    if any(isnan([heading(min(length(heading),i)) pitch(min(length(pitch),i)) roll(min(length(roll),i))]))
        M = nan(3,3);
    else
        % heading/pitch/roll matrices - this isn't the standard rotations, but it matches what winADCP
does, and this is how it was originally implemented:
        % I considered various ways to get the heading index - pointers, use an eval, etc, but this is
actually faster: heading(min(length(heading), i))
        M1 = [cosd(heading(min(length(heading),i))) sind(heading(min(length(heading),i))) 0; -sind(heading
(min(length(heading),i))) cosd(heading(min(length(heading),i))) 0; 0 0 1];
        M2 = [1 0 0; 0 cosd(pitch(min(length(pitch),i))) -sind(pitch(min(length(pitch),i))); 0 sind(pitch
(min(length(pitch),i))) cosd(pitch(min(length(pitch),i)))]];
        M3 = [cosd(roll(min(length(roll),i))) 0 sind(roll(min(length(roll),i))); 0 1 0; -sind(roll(min
(length(roll),i))) 0 cosd(roll(min(length(roll),i)))]];
        M = M1 * M2 * M3;
        if strcmp(Config.orient, 'Up') % negate 1st & 3rd column
            M(1:3,1) = -M(1:3,1);
            M(1:3,3) = -M(1:3,3);
        end
    end

    % apply heading/pitch/roll rotations
    if maxNumPosData == 1
        velocityTrue = M * velocityScreenedReshape(1:3,:);
    else
        velocityTrue(:, (i-1)*n+1:i*n) = M * velocityScreenedReshape(1:3, (i-1)*n+1:i*n);
    end
end

% extract u,v,w,error values - singleton dimensions handled later
velocityTrue = reshape([velocityTrue; velocityScreenedReshape(4,:)], [m n p]);
ADCP.u = squeeze(velocityTrue(1,:,:));
ADCP.v = squeeze(velocityTrue(2,:,:));
ADCP.w = squeeze(velocityTrue(3,:,:));
ADCP.velocityError = squeeze(velocityTrue(4,:,:));

```

For Nortek ADCPs, the process is a bit different and simpler. We convert all data back to Beam co-ordinate data and then use the same beam to true Earth code to derive the true Earth co-ordinate data.

```

%% get the original beam velocities
if strcmp(Config.coordSys, 'BEAM')
    ADCP.processingComments(end+1,1) = {'Raw data matrix, data.velocity, is the beam radial velocities.'};
    beamMatrix = ADCP.velocity;
elseif strcmp(Config.coordSys, 'XYZ')
    ADCP.processingComments(end+1,1) = {'Raw data matrix, data.velocity, is the XYZ velocities (instrument co-ordinates). Used the instrument transform matrix to revert to beam radial velocities. '};
    beamMatrix = T \ ADCP.velocity;
elseif strcmp(Config.coordSys, 'ENU')
    ADCP.processingComments(end+1,1) = {'Raw data matrix, data.velocity, is the ENU velocities (Earth co-ordinates relative to onboard compass, pitch, roll sensors). Used the instrument transform matrix and raw onboard sensor data to revert to beam velocities. '};
    beamMatrix = nan(3, n*p);
    for i = 1:p
        hThis = (ADCP.compassHeading(i) - 90) * pi/180;
        pThis = ADCP.pitch(i) * pi/180;
        rThis = ADCP.roll(i) * pi/180;
        H = [cos(hThis) sin(hThis) 0; -sin(hThis) cos(hThis) 0; 0 0 1];
        PR = [cos(pThis) -sin(pThis)*sin(rThis) -cos(rThis)*sin(pThis);...
            0 cos(rThis) -sin(rThis); ...
            sin(pThis) sin(rThis)*cos(pThis) cos(pThis)*cos(rThis)];
        beamMatrix(:, (i-1)*n+1:i*n) = (H * PR * T) \ ADCP.velocity(:, (i-1)*n+1:i*n);
    end
    beamMatrix = reshape(beamMatrix, [3 p n]);
else
    error('ONC:nortekENUvelocities:unknownCoordSys', 'Unknown co-ord system');
end

```

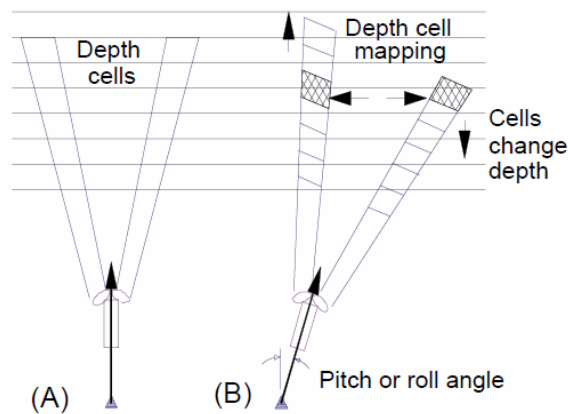
```

velocityTrue = nan(3, n*p);
maxNumPosData = max([length(heading) length(pitch) length(roll)]); % either 1 or p, the number of time stamps
for i = 1:maxNumPosData
    % heading/pitch/roll matrices
    % this rotation is from Nortek: http://www.nortek-as.com/lib/forum-attachments/coordinate-transformation/view
    % I considered various ways to get the index - pointers, use an eval, etc, but using 'min(length(heading), i)' is actually faster
    hThis = (heading(min(length(heading),i)) - 90) * pi/180; % convert heading to yaw, the Nortek code doesn't have a '-' here, but the sine term is negated below
    pThis = pitch(min(length(pitch),i)) * pi/180;
    rThis = roll(min(length(roll),i)) * pi/180;
    H = [cos(hThis) sin(hThis) 0; -sin(hThis) cos(hThis) 0; 0 0 1];
    PR = [cos(pThis) -sin(pThis)*sin(rThis) -cos(rThis)*sin(pThis);...
        0 cos(rThis) -sin(rThis); ...
        sin(pThis) sin(rThis)*cos(pThis) cos(pThis)*cos(rThis)];
    % apply heading/pitch/roll rotations
    if maxNumPosData == 1
        velocityTrue = H * PR * T * beamReshape;
    else
        velocityTrue(:, (i-1)*n+1:i*n) = H * PR * T * beamReshape(:, (i-1)*n+1:i*n);
    end
end
% extract w values (u,v below) - use permute to ensure we don't lose singleton dimensions
velocityTrue = reshape(velocityTrue, [3 p n]);
ADCP.u = permute(velocityTrue(1,:,:), [3 2 1]);
ADCP.v = permute(velocityTrue(2,:,:), [3 2 1]);
ADCP.w = permute(velocityTrue(3,:,:), [3 2 1]);

```

Correction for Tilt: Bin-mapping (For Instr/Beam data only, step #5)

In addition to coordinate system rotations, the orientation of the instrument affects the vertical range on which the measurement bins are spaced (depth cell and measurement bin are synonymous). If the instrument is significantly tilted, the bins used to determine the velocities at each range will be at different water depths. Here is Figure 21 from [Acoustic Doppler Current Profiler: Principles of Operation - A Practical Primer](#), notice in (B) that the highlighted depth cells are not the same as in (A):



RDI ADCP data collected in Earth co-ordinates normally has had a depth cell mapping algorithm applied to align the depth cells to correct for tilt, while the other modes have not had it applied. Depth cell mapping is applied in post-processing in *winADCP*. Nortek's *Storm / Surge* post-processing software also applies bin-mapping when doing co-ordinate system rotations/transforms. (The term 'bin-mapping' is used by Nortek, RDI uses 'depth cell mapping'. We use 'bin-mapping' as it's the most common term in the literature.) Conversely, Nortek ADCP data collected in ENU co-ordinates has not had bin mapping applied. To apply bin-mapping, ENU data is converted back to Beam data, bin-mapping is applied and then converted back to ENU, doing any rotations as needed. The approach originated as described by [Pulkkinen \(1992\)](#). The Pulkkinen / RDI bin-mapping method is based on matching up the nearest vertical bin in each beam for each of the original range steps, then the radial velocities from those bins are used in the rotation to true Earth velocities. For compatibility with original VENUS ADCP data products, an alternative method is offered as a option as well, as described by [Ott \(2002\)](#). In that paper, the Ott method is shown to be an improvement over the discrete nearest vertical bin matching method. The Ott method uses a linear interpolation between the nearest vertical bins for each beam for each of the original range steps. The Ott method also interpolates over a small number of missing bins, otherwise missing data nullifies a greater extent of the data than it does in the nearest vertical bin method. The inner and outer most bins can end up with NaN values when there are less than 4 bins at the same vertical level, see the above graphic. This affects the nearest vertical bin method more so than the Ott method, and more bins are affected with increasing tilts.

Users are offered the option of 'None', 'Nearest vertical bin', or 'Linear interpolation (Ott method)'. When the data is in Earth co-ordinates, bin-mapping cannot be applied/modified. For RDI Earth data, the nearest vertical bin method has normally been applied, while Nortek Earth data does not have bin-mapping applied. In both cases, the users' preference may be overridden. The RDI manual indicates that pitch or roll values greater than 20 degrees are not possible and nominal processing has, in the past, applied NaN values when the pitch or roll exceeds that limit. However, we have found that RDI ADCPs can accurately report higher tilts. The limit is now set to 60 degrees for pitch/roll and at or above 90 degrees of average tilt, bin-mapping is overridden to the 'None' option.

Bin-mapping is not applied to the backscatter and intensity data. Instead, we offer a vertically corrected range in the mat file products and use this range for the plots when plotting against depth. For example, if the bin spacing is 1 m, but the tilt is 10 degrees, the vertical spacing is 0.98 m.

For advanced, interested users, here is the core code that does bin-mapping:

```

%% apply nearest vertical bin-mapping (RDI method)
if isOrientUp
    ZSG = [+1 -1 +1 -1]; % ADCP up/convex
else
    ZSG = [+1 -1 -1 +1]; % ADCP down/convex
end
beamElAbsRad = abs(beamElevation) * pi/180; % calcs below, from beam2uvw.m, take the abs() and convert to
rad, so do it here for speed
beamElRad = beamElevation * pi/180;
beamNearestVert = nan(size(beam));
maxNumPosData = max([length(pitch) length(roll)]); % either 1 or p, the number of time stamps - gotta be
the same length, error if not!
for k = 1:maxNumPosData % loop over time stamps and pitch/roll values
    % NaN any pings that don't have good pitch/roll
    if any([pitchOverLim(k) rollOverLim(k)])
        doMaxPitchRollWarning = true;
        continue
    end
    if any([isnan(pitch(k)) isnan(roll(k))])
        doNaNPitchRollWarning = true;
        continue
    end

    % use the bin-mapping math from beam2uvw.m (not sure what ZSG and SC are supposed to be named for, but
    keep them for historical reasons)
    SC = sin(beamElAbsRad)*cos(pitch(k))*cos(roll(k)) + ...
        cos(beamElRad) .* ZSG .* [-sin(roll(k)) -sin(roll(k)) sin(pitch(k))*cos(roll(k)) sin(pitch(k))*cos
(roll(k))];
    SC = abs(sin(beamElAbsRad) ./ SC); % the abs wasn't in beam2uvw.m but even with nominal conditions it
needs to be there
    vertBinIndex = (round(binIndex.' * SC)).';
    indexOutOfBoundsFlag = vertBinIndex < 1 | vertBinIndex > n;
    vertBinIndex(indexOutOfBoundsFlag) = 1;
    for i = 1:m
        beamNearestVert(i, :, k) = beam(i, vertBinIndex(i,:), k);
        beamNearestVert(i, indexOutOfBoundsFlag(i), k) = NaN;
    end
end
beam = beamNearestVert;

```

```

%% do the Ott method bin mapping, including Ott's filling/smoothing method
binSpacing = mean(diff(range));
minRangeExtrap = min(range) - binSpacing;
maxRangeExtrap = max(range) + binSpacing;

for i = 1:m % loop over beams
    % do this calc here since it doesn't change in time
    matrixBeamElevationAzimuth = [-cosd(beamElevation(i))*sind(beamAzimuth(i)); cosd(beamElevation(i))*cosd
(beamAzimuth(i)); -sind(beamElevation(i))] ./ sind(abs(beamElevation(i)));
    for k = 1:p % loop over time stamps
        iP = min(length(pitch),k);
        iR = min(length(roll),k);
        % NaN any pings that don't have good pitch/roll
        pThis = pitch(iP);
        rThis = roll(iR);
        if any([pitchOverLim(iP) rollOverLim(iR)])
            beam(i,:,k) = NaN;
            doMaxPitchRollWarning = true;
            continue
        end
        if any([isnan(pThis) isnan(rThis)])
            beam(i,:,k) = NaN;
            doNaNPitchRollWarning = true;
            continue
        end
        % skip any pings that are all NaN
        thisBeamTime = squeeze(beam(i,:,k));
        iNaNthisBeamTime = isnan(thisBeamTime);
    end
end

```

```

        numNaNthisBeamTime = sum(iNaNthisBeamTime);
        if numNaNthisBeamTime >= min(n * maxFracNaN, n - 1) % skip this beam time if too much data is NaN
            (need at least 2 valid points)
            beam(i,:,k) = NaN;
            continue
        end
        % for this time and beam, fill the NaNs with linear interpolation, except where there are too many
        NaNs in a row
        % this won't fill on outside of valid data - linear interpolation doesn't extrapolate by default
        if any(iNaNthisBeamTime)
            % find consecutive NaNs for this beam and time, only if there are enough of them to flag
            iTooManyConsecNaNs = false(1, n);
            if numNaNthisBeamTime >= maxConsecutiveNaNs
                numConsec = 0;
                for j = find(iNaNthisBeamTime, 1, 'first') : find(iNaNthisBeamTime, 1, 'last') % loop
                    over depth bins
                        if iNaNthisBeamTime(j)
                            numConsec = numConsec + 1;
                            if numConsec > maxConsecutiveNaNs
                                iTooManyConsecNaNs((j-numConsec+1):j) = true;
                            end
                        else
                            numConsec = 0;
                        end
                    end
                end
                iNaNsInterp = iNaNthisBeamTime & ~iTooManyConsecNaNs;
                thisBeamTime(iNaNsInterp) = interp1(binIndex(~iNaNthisBeamTime), thisBeamTime
                (~iNaNthisBeamTime), binIndex(iNaNsInterp));
            end
            % this is simplication of the commented code below
            Mp = [1 0 0; 0 cos(pThis) -sin(pThis); 0 sin(pThis) cos(pThis)];
            Mr = [cos(rThis) 0 sin(rThis); 0 1 0; -sin(rThis) 0 cos(rThis)];
            beamXYZ = Mr * Mp * matrixBeamElevationAzimuth;
            actualVerticalRange = range * abs(beamXYZ(3));

%           % range calculation code from beam2uvw.m
%           Mp = [1 0 0; 0 cos(pThis) -sin(pThis); 0 sin(pThis) cos(pThis)];
%           Mr = [cos(rThis) 0 sin(rThis); 0 1 0; -sin(rThis) 0 cos(rThis)];
%           beamXYZ = Mr * Mp * [-cosd(beamElevation(i))*sind(beamAzimuth(i)); cosd(beamElevation(i))*cosd
(beamAzimuth(i)); -sind(beamElevation(i))];
%           actualVerticalRange = range * abs(beamXYZ(3) ./ sind(abs(beamElevation(i))));
%           % map the beam bins to the vertical range - allow some extrapolation - won't extrapol through NaNs
though (good).
%           % Extrapolation limited to +/- one bin spacing.
            beam(i,:,k) = interp1( actualVerticalRange, thisBeamTime, range, 'linear', 'extrap' );
            beam(i, actualVerticalRange > maxRangeExtrap | actualVerticalRange < minRangeExtrap, k) = NaN;
        end
    end
end

```

Three-Beam Solutions (For RDI ADCPs Only, Instr/Beam step #5)

In systems with four beams, the transformation from radial velocities to Earth co-ordinates velocities has redundant information; only three beams are needed to resolve currents in the three orthogonal directions, while the 4th beam effectively provides error estimation. In the case where exactly one bin out of the four at any level is flagged and NaN'ed, then currents can still be calculated from the remaining three radial velocities, using the three-beam transformation solution. The screening steps that can flag the data 'NaN' prior to transformation include fish detection and correlation threshold, as well as any bin-mapping (particularly the RDI nearest vertical bin which can set many bins to NaN). The three-beam solution works by assigning the error velocity to be zero and then solving the transformation for the missing bin (for more details see Three Beam Solutions in [adcp coordinate transformation_Jan08.pdf](#)). With the missing data filled in, the normal transform from Beam to Instrument co-ordinate data is applied.

In cases where the data quality team has determined that an entire beam is bad, they can supply a device attribute specifying the bad beam number. That beam is then excluded in the processing from beam or instr co-ords and the values are filled in with the three-beam solution. Only the U,V,W results are affected, the raw velocity data as recorded by the instrument is presented without the exclusion. This forces the 3 beam option to on, regardless of the user input. The excluded beam number is mentioned in a comment on both the [currents plot](#) and [intensity plots](#) and in the processing comments in the [mat files](#). This is currently only available for RDI ADCPs with 4 beams.

Here is the code for three-beam solutions and transform from Beam to Instrument (XYZ) co-ordinate data:

```

%% apply the three beam solution
% Note, for instr data, this may be a bit redundant unless some data has been screened or bin-mapped out
if Config.coord_EX(4) == '1' % only 3-beam if the EX parameter is set, this replicates winADCP's behaviour
    isnanBM = isnan(beamMatrix);
    is3beam = squeeze(sum(isnanBM, 1)) == 1; % index of all bins over time that have 1 NaN only (3 beam)
    transformMatrixRow4 = transformMatrix(4,:).';
    for j = find(any(is3beam,2)).' % loop over the bin indices that have at least one 3 beam instance
        for k = find(is3beam(j,:)) % loop over the times when the current bin has a 3 beam instance
            isnanThis = isnan(beamMatrix(:,j,k));
            temp = beamMatrix(:,j,k) .* transformMatrixRow4;
            beamMatrix(isnanThis,j,k) = -sum(temp(~isnanThis)) / transformMatrix(4, isnanThis);
        end
    end
    ADCP.processingComments(end+1,1) = {'Three beam solution applied to ' num2str(sum(is3beam(:))) '
individual bins (of ' num2str(numel(beamMatrix)) ' total). '};
else
    ADCP.processingComments(end+1,1) = {'Three beam solution not applied as the EX command was set to
xxx0x. (This matches the behaviour of winADCP which responds to EX(4). We can override this on request, contact
us. As a side note, winADCP does not respond to EX(5), the command for bin-mapping; winADCP always does bin-
mapping on beam co-ord data.'};
end

%% convert bin-mapped, screened and 3-beamed data to XYZ
vTemp = reshape(beamMatrix, 4, n * p);
velocityInstrOrEarth = reshape(transformMatrix * vTemp, [m n p]);

```

Screening

Correlation screening is the most widely applied and effective screening step, noted as step #3 in the Instr/Beam overall processing flow. It is also applied onboard RDI ADCPs when in Earth co-ordinate configuration, the value applied is available in the config structure.

Nortek ADCPs are generally three beam (so the three-beam solution option is not offered) and screening is not applied in the data products, except for a fixed 50% correlation screen on Nortek Signature55 data only, applied to match Nortek's Signature Viewer software.

RDI ADCPs screening steps also include a fish detection algorithm and screening, Instr/Beam step #4 in the overall process flow.

Once in Earth co-ordinates, the data can also be screened by the error velocity. Error velocity screening only applies to data with four valid beams (and non-zero error velocities). For further information on the screening steps, including algorithms, see [adcp coordinate transformation_Jan08.pdf](#) (RDI ADCPs only). This is step #9 in Instr/Beam and #2 in Earth data process flows.

Data Verification

As noted earlier, the ultimate requirement for the ADCP data products is that they replicate the results of the manufacturer's software. We test against the manufacturer's software to verify the data at every software release (regression testing). There are cases where this adherence is intentionally not true: if users choose any non-default option (or that says it's not the RDI default), the results will not match the default processing in the manufacturer's software, in particular, the Ott bin-mapping option is not available on winADCP at all. The testing suite includes manual and automated testing where the output of the manufacturer's software is compared to the data products and where data products are compared to the captured files from the previous release. The test suite is regularly updated for any new scenarios. Improvements in the test procedure are also made as needed. In addition to testing the software, there has been a significant effort to ensure that the metadata, in particular the heading metadata, is correct. These steps include careful measurement and documentation at deployment time, verifying and vetting ROV heading sensors, comparing data to nearby ADCPs, comparing results between various processing methods, and perhaps the best check is to compare the currents with the known/modelled tides. [Contact us](#) for information on the testing procedures and metadata verification.

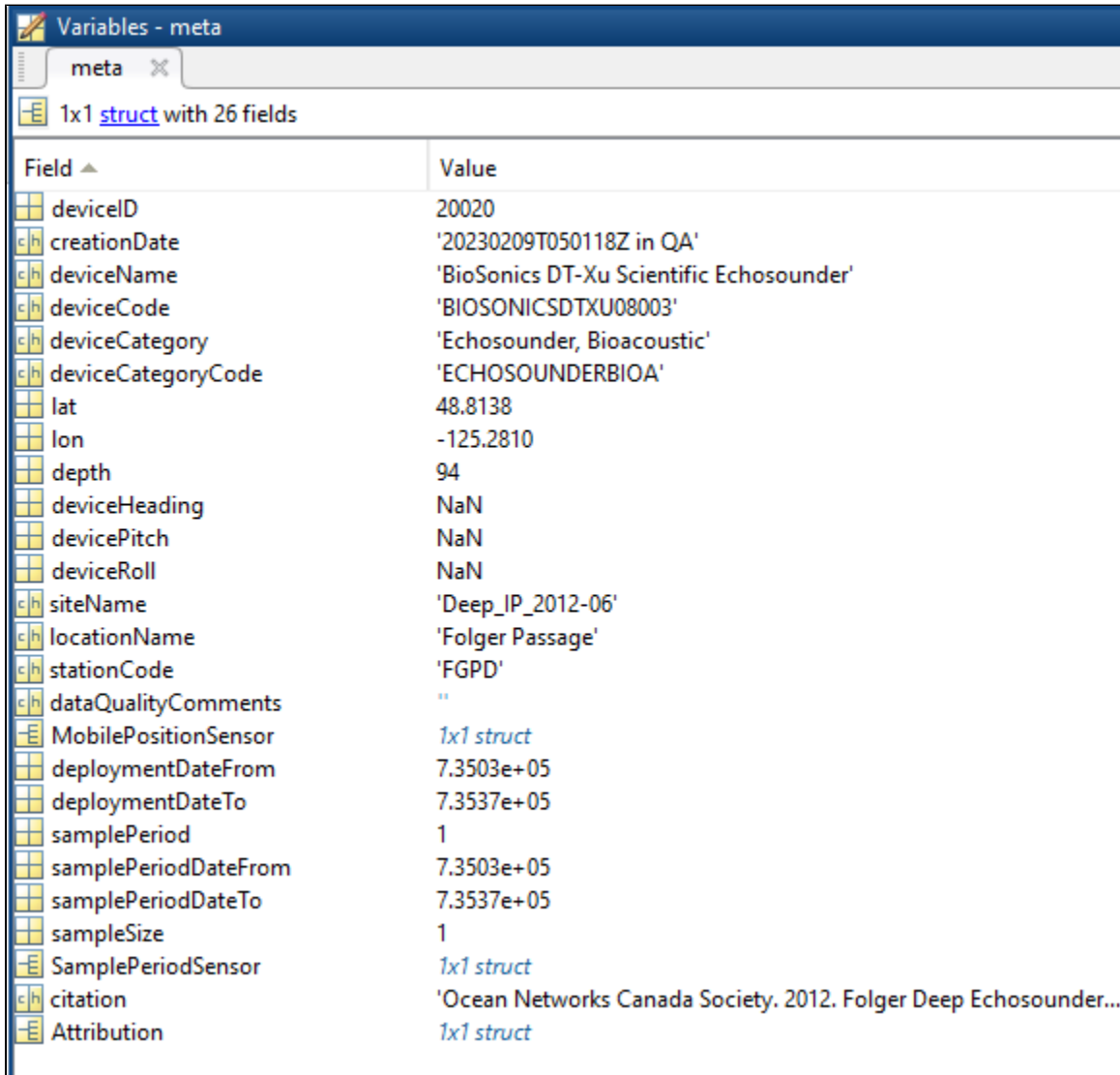
A recent simplification in testing came about because of an improvement in the generation of **RDI** file data products, which we archive to speed up generation of subsequent products, **MAT/NC** files and plots. As of November 2015, **RDI** files with Beam or Instr co-ordinate data will incorporate the same heading/pitch/roll values as used for rotation and bin-mapping, that way, users and testers may load and process these files with RDI software, matching the values they can obtain from **MAT** and **NC** file products. For Earth co-ordinate data in **RDI** files, if we update the heading, we also need to rotate the East (u) and North (v) velocities as the new heading will be different from the internal compass heading (either fixed value, mobile position sensor or a magnetic declination is applied) on which the onboard Earth co-ordinate rotation was done. In this case, we don't update the pitch/roll as we cannot update the bin-mapping done for Earth data. Fortunately, the pitch/roll data is generally good and we do not yet have any examples where we do not use the internal pitch/roll data. Once the **RDI** files produced prior to November 2015 are re-generated, all **RDI** files will be made available in Data Search.

In all, the results from RDI winADCP, and Nortek software will match the processed data products. In addition, the original data, particularly the Beam co-ordinate data is always available in the **RDI** and [Nortek raw binary files](#) and as *data.velocity* (Nortek) and *adcp.velocity* (RDI) in the **MAT** files. The *processingComments* will completely describe the steps applied to produce the *u,v,w* processed velocities. In addition to the code snippets above, we are happy to supply more code, collaborate on that code, etc. We are interested in collaborating on QAQC processes, data verification, processing, analysis and research. [Contact us](#) if you have any questions.

MAT

MAT files (v7) can be opened using MathWorks MATLAB 7.0 or later. The file contains four structures: meta, data, config, and units. For short duration searches (less than one day) or short duration data returned, the files will be concatenated, otherwise, expect one file per day.

Meta: a structure array containing the following metadata fields:



Field ▲	Value
deviceID	20020
creationDate	'20230209T050118Z in QA'
deviceName	'BioSonics DT-Xu Scientific Echosounder'
deviceCode	'BIOSONICSSTXU08003'
deviceCategory	'Echosounder, Bioacoustic'
deviceCategoryCode	'ECHOSOUNDERBIOA'
lat	48.8138
lon	-125.2810
depth	94
deviceHeading	NaN
devicePitch	NaN
deviceRoll	NaN
siteName	'Deep_IP_2012-06'
locationName	'Folger Passage'
stationCode	'FGPD'
dataQualityComments	''
MobilePositionSensor	1x1 struct
deploymentDateFrom	7.3503e+05
deploymentDateTo	7.3537e+05
samplePeriod	1
samplePeriodDateFrom	7.3503e+05
samplePeriodDateTo	7.3537e+05
sampleSize	1
SamplePeriodSensor	1x1 struct
citation	'Ocean Networks Canada Society. 2012. Folger Deep Echosounder...'
Attribution	1x1 struct

- deviceID: A unique identifier to represent the instrument within the Ocean Networks Canada data management and archiving system.
- creationDate: Date and time (using ISO8601 format) that the data product was produced. This is a valuable indicator for comparing to other revisions of the same data product.
- deviceName: A name given to the instrument.
- deviceCode: A unique string for the instrument which is used to generate data product filenames.
- deviceCategory: Device category to list under data search ('Echosounder').
- deviceCategoryCode: Code representing the device category. Used for accessing webservices, as described here: [API / webservice documentation](#) (log in to see this link).
- lat: Fixed value obtained at time of deployment. Will be NaN if mobile or if both site latitude and device offset are null. If mobile, sensor information will be available in mobilePositionSensor structure..
- lon: Fixed value obtained at time of deployment. Will be NaN if mobile or if both site longitude and device offset are null. If mobile, sensor information will be available in mobilePositionSensor structure.
- depth: Fixed value obtained at time of deployment. Will be NaN if mobile or if both site depth and device offset are null. If mobile, sensor information will be available in mobilePositionSensor structure.
- deviceHeading: Fixed value obtained at time of deployment. Will be NaN if mobile or if both site heading and device offset are null. If mobile, sensor information will be available in mobilePositionSensor structure.
- devicePitch: Fixed value obtained at time of deployment. Will be NaN if mobile or if both site pitch and device offset are null. If mobile, sensor information will be available in mobilePositionSensor structure.
- deviceRoll: Fixed value obtained at time of deployment. Will be NaN if mobile or if both site roll and device offset are null. If mobile, sensor information will be available in mobilePositionSensor structure.
- siteName: Name corresponding to its latitude, longitude, depth position.
- locationName: The node of the Ocean Networks Canada observatory. Each location contains many sites.
- stationCode: Code representing the station or site. Used for accessing webservices, as described here: [API / webservice documentation](#) (log in to see this link).
- dataQualityComments: In some cases, there are particular quality-related issues that are mentioned here.
- MobilePositionSensor: A structure with information about sensors that provide additional scalar data on positioning and attitude (latitude, longitude, depth below sea surface, heading, pitch, yaw, etc).

meta	
meta.MobilePositionSensor	
Field	Value
name	1x1 cell
sensorID	NaN
deviceID	NaN
dateFrom	NaN
dateTo	NaN
typeName	1x1 cell
offset	NaN
units	1x1 cell
sensorTypeID	NaN
correctedSensorID	NaN

- name: A cell array of sensor names for mobile position sensors. If not a mobile device, this will be an empty cell string.
- sensorID: An array of unique identifiers of sensors that provide position data for mobile devices - this data may be used in this data product.
- deviceID: An array of unique identifiers of devices that provide position data for mobile devices - this data may be used in this data product.
- dateFrom: An array of datenums denoting the range of applicability of each mobile position sensor - this data may be used in this data product.
- dateTo: An array of datenums denoting the range of applicability of each mobile position sensor - this data may be used in this data product.
- typeName: A cell array of sensor names for mobile position sensors. If not a mobile device, this will be an empty cell string. One of: Latitude, Longitude, Depth, COMPASS_SENSOR, Pitch, Roll.
- offset: An array of offsets between the mobile position sensors' values and the position of the device (for instance, if cabled profiler has a depth sensor that is 1.2 m above the device, the offset will be -1.2m).
- sensorTypeID: An array of unique identifiers for the sensor type.
- correctedSensorID: An array of unique identifiers of sensors that provide corrected mobile positioning data. This is generally used for profiling deployments where the latency is corrected for: CTD casts primarily.
- deploymentDateFrom: The date of the deployment on which the data was acquired.
- deploymentDateTo: The date of the end of the deployment on which the data was acquired (will be NaN if still deployed).
- samplingPeriod: Sample period / data rating of the device in seconds, this is the sample period that controls the polling or reporting rate of the device (some parsed scalar sensors may report faster, some devices report in bursts) (may be omitted for some data products).
- samplingPeriodDateFrom: matlab datenum of the start of the corresponding sample period (may be omitted for some data products).
- samplingPeriodDateTo: matlab datenum of the end of the corresponding sample period (may be omitted for some data products).
- sampleSize: the number of readings per sample period, normally 1, except for instruments that report in bursts. Will be zero for intermittent devices (may be omitted for some data products).
- SamplePeriodSensor: A structure array with an entry for each scalar sensor on the device (even though this metadata is for complex data products that don't use scalar sensors).

meta

meta.MobilePositionSensor

meta.Attribution

meta.SamplePeriodSensor

meta.SamplePeriodSensor

Fields	sp	dateFrom	dateTo	sampleSize	deviceID	sensorID	isDeviceLevel	sensorName
1	1	7.3695e+05	7.3727e+05	1	20001	8214	1	'Sound Speed'
2	1	7.3695e+05	7.3727e+05	1	20001	8215	1	'Magnetic Com...
3	1	7.3695e+05	7.3727e+05	1	20001	8216	1	'Pitch'
4	1	7.3695e+05	7.3727e+05	1	20001	8219	1	'Pressure'
5	1	7.3695e+05	7.3727e+05	1	20001	8217	1	'Roll'
6	1	7.3695e+05	7.3727e+05	1	20001	8218	1	'Temperature'
7								

- sp: sample period in seconds (array), unless sensorid is NaN then this is the device sample period
- dateFrom: array of date from / start date (inclusive) for each sample period in MATLAB datenum format.
- dateTo: array of date to / end date (exclusive) for each sample period in MATLAB datenum format.
- sampleSize: the number of readings per sample period (array). Normally 1, except for instruments that report in bursts. Will be zero for intermittent devices.
- deviceID: array of unique identifiers of devices (should all be the same).
- sensorID: array of unique identifiers of sensors on this device.
- isDeviceLevel: flag (logical) that indicates, when true or 1, if the corresponding sample period/size is from the device-level information (i.e. applies to all sensors and the device driver's poll rate).
- sensorName: the name of the sensor for which the sample period/size applies (much more user friendly than a sensorID).
- citation: a char array containing the DOI citation text as it appears on the [Dataset Landing Page](#). The citation text is formatted as follows: <Author(s) in alphabetical order>. <Publication Year>. <Title, consisting of Location Name (from searchTreeNodeName or siteName in ONC database) Deployed <Deployment Date (sitedevicefrom in ONC database)>. <Repository>. <Persistent Identifier, which is either a DOI URL or the queryPID (search_dtlid in ONC database)>. Accessed Date <query creation date (search.datecreated in ONC database)>

- Attribution: A structure array with information on any contributors, ordered by importance and date. If an organization has more than one role it will be collated. If there are gaps in the date ranges, they are filled in with the default Ocean Networks Canada citation. If the "Attribution Required?" field is set to "No" on the [Network Console](#) then the citation will not appear. Here are the fields:

meta.Attribution	
Field ▲	Value
acknowledgement	'Ocean Networks Canada Data Archive, http://www.oceannetworks.ca , University of Victoria, Canada'
startDate	[]
endDate	[]
organizationName	'Ocean Networks Canada'
organizationRole	'Owner'
roleComment	''

- acknowledgement: the acknowledgement text, usually formatted as "<organizationName> (<organizationRole>)", except for when there are no attributions and the default is used (as shown above).
- startDate: datenum format
- endDate: datenum format
- organizationName
- organizationRole: comma separated list of roles
- roleComment: primarily for internal use, usually used to reference relevant parts of the data agreement (may not appear)

data: structure containing the ADCP data, having the following fields.

data						
1x1 struct with 23 fields						
Field ▲	Value	Class	Min	Max	Mean	
time	1x48 double	double	7.3651e+05	7.3651e+05	7.3651e+05	
range	1x65 double	double	30	1310	670	
u	65x48 double	double	NaN	NaN	NaN	
v	65x48 double	double	NaN	NaN	NaN	
w	65x48 double	double	NaN	NaN	NaN	
velocity	3x48x65 double	double	-1.2240	1.2660	0.0344	
amplitude	3x48x65 double	double	68	175	109.5233	
correlation	3x48x65 double	double	0	100	57.6686	
compassHeading	1x48 double	double	317.8000	318.3000	318.0250	
pitch	1x48 double	double	1	1	1	
roll	1x48 double	double	0.4000	0.5000	0.4979	
pressure	1x48 double	double	961.6156	963.7613	962.8348	
temperature	1x48 double	double	3.5800	3.6700	3.6125	
soundSpeed	1x48 double	double	1481	1.4814e+03	1.4811e+03	
voltage	1x48 double	double	46.3000	46.3000	46.3000	
error	1x48 double	double	NaN	NaN	NaN	
backscatter	3x48x65 double	double	96.5880	162.3478	122.9555	
meanBackscatter	65x48 double	double	98.5483	160.5832	123.2065	
processingComments	9x1 cell	cell				
magneticDeclination	16.7006	double	16.7006	16.7006	16.7006	
corScreenThresholdUsed	50	double	50	50	50	
amplitudeVerticalRange	1x65 double	double	29.9943	1.3098e+03	669.8724	
binMappingOption	'None'	char				

- time: vector, timestamp in datenum format (obtained from time the reading reached the shore station)
- range: vector of distance to each bin centre. If bin mapping compensation for tilt is active, this is then a vertical range to the bin centres, see data.processingComments to see if bin mapping was applied. (Prior to Feb 26, 2015: vector of distance to start of each bin)
- cellDepth: vector, starting depth of each cell calculated from range and deployment depth (*may be inaccurate if instrument is vertically mobile*)
- u,v,w: 2D matrices, East/North/Up velocities relative to True North, derived using raw velocity matrix, orientation, coordinate system, transformation matrix and one of a) calculated magnetic declination based on time and location (for mobile adcps) or b) meta.deviceHeading (fixed position ADCPs)
- u_std, v_std, w_std: standard deviations for ensemble averaged u,v,w (averaged products only).
- pingsPerEnsemble: vector, number of pings that contributed to the ensemble (averaged products only).
- velocity: 3D matrix, corresponds directly to output of instrument and so depends on configuration coordinate system
- amplitude: 3D matrix, amplitude time-series for each of the three receivers. From [Nortek Technical Note No. 3](#): "All Nortek systems use the same component family to measure the amplitude for the echo. This component outputs a signal that is referred to as the RSSI and that is proportional to logarithm of the echo strength. The dynamic range of this signal is about 90 dB and it is linear within an accuracy of about 1-2 dB over a range of 70 dB. Inside this range, the scaling factor is about 0.45 counts/dB but with some variation (about 0.40 to 0.47)."
- correlation: 3D matrix, signal correlation coefficient time-series for each of the three receivers (*only available for HR-Profilers*)
- compassHeading: vector, magnetic compass heading time-series. (Strictly from onboard sensor. See data.processingComments to determine which source was used for rotation: one of internal or external sensor or fixed value.)

- pitch: vector, pitch time-series. (Strictly from onboard sensor. See data.processingComments to determine which source was used for rotation: one of internal or external sensor or fixed value.)
- roll: vector, roll time-series. (Strictly from onboard sensor. See data.processingComments to determine which source was used for rotation: one of internal or external sensor or fixed value.)
- pressure: vector, pressure time-series
- temperature: vector, temperature time-series
- soundSpeed: vector, sound speed time-series (often constant, depending on configuration)
- voltage: vector, battery voltage time-series
- error: vector, error code (see the manufacturer's system integrator guides to decode)
- backscatter: 3D matrix based on received signal strength intensity (adcp.intensity), compensated for two-way spreading (20LogR) and absorption. Equation based on [Gostiaux and van Haren](#) ("Extracting Meaningful Information from Uncalibrated Backscattered Echo Intensity Data, Journal of Atmospheric and Oceanic Technology, 72, 943-949, 2010). The absorption computation follows Ainslie and Malcolm ("A simplified formula for viscous and chemical absorption in sea water", Journal of the Acoustical Society of America, 103(3), 1671-1672, 1998). Absorption coefficient is based on mean depth, temperature and salinity in adcp structure.
- meanBackscatter: same as above, except averaged over the four beams to create a 2D matrix. Averaging is done by converting to standard intensity, averaging, then converting back to decibels.
- processingComments: cell array char field of comments, documenting all of the processing steps taken on the data (such as transform from beam co-ordinates to East-North-Up) and the sources of positioning data applied (heading, pitch, roll, depth)
- magneticDeclination: value that can be applied to correct the compassHeading to true North.
- corScreenThresholdUsed: value used for the correlation screen, i.e. data with correlation less than this are screened (replaced with NaN) from the final u,v,w velocities.
- amplitudeVerticalRange: tilt-corrected range for plotting backscatter (calculated as $\cos(\text{average tilt}) * \text{range}$).

config: structure containing configuration details, parsed from the binary configuration or string data structures that are placed at the beginning of the raw binary files (as described earlier). For details about the configuration parameters, refer to the manufacturer documentation. * - denotes that these fields are not (yet) populated for Signature Series.

config					
1x1 struct with 25 fields					
Field	Value	Class	Min	Max	Mean
SN	"	char			
headSN	"	char			
boardFrequency	[]	double			
headFrequency	55	double	55	55	55
fwVersion	"	char			
hwRevision	[]	double			
PICversion	[]	double			
avgInterval	2	double	2	2	2
measurementInterval	2	double	2	2	2
compassUpdateRate	[]	double			
nbeams	3	double	3	3	3
ncells	65	double	65	65	65
npings	1	double	1	1	1
velocityScaling	-3	double	-3	-3	-3
beamAngle	20	double	20	20	20
blankingDistance	20	double	20	20	20
cellSize	20	double	20	20	20
coordSys	'BEAM'	char			
orientation	'Up'	char			
transformationMatrix	[1.9492,-0.9746,-0.9746;0,-1.6880,1.6880;0.3547,0.3547,0.3547]	double	-1.6880	1.9492	0.1182
correlationThreshold	[]	double			
rawDataVelocityUnits	'mm/s'	char			
plan	"	char			
soundAbsorptionCoefficient	0.0145	double	0.0145	0.0145	0.0145
configBit	'11111111'	char			

- SN: instrument type and serial number*
- headSN: head serial number*
- boardFrequency: board frequency*
- headFrequency: head frequency
- fwVersion: firmware version*
- hwRevision: hardware revision*
- PICversion: PIC code version*
- avgInterval: time the instrument is actively measuring within the profile interval
- measurementInterval: time between each measurement (i.e., data output rate)
- compassUpdateRate: rate at which compass readings are updated
- nbeams: number of transducers/beams.
- ncells: number of cells
- npings: number of pings per profile
- velocityScaling: scale conversion from raw data to data in mat file - always converts all velocities to m/s, from either mm/s or 0.1 mm/s, by multiplying by $10^{\text{config.velocityScaling}}$. (Nortek Software does this as well).

- beamAngle: angle of beams
- blankingDistance: distance from the sensor head to the start of the first measurement cell. See [Nortek forum discussion](#) for calculation details.
- cellSize: size of each cell. See [Nortek forum discussion](#) for calculation details.
- coordSys: coordinates system of raw data (BEAM, XYZ or ENU)
- orientation: Up or Down indicates the direction transducers are facing
- transformationMatrix: matrix, used to convert from beam to xyz coordinates
- correlationThreshold: correlation threshold for resolving ambiguities
- rawDataVelocityUnits: records the original unit of measurement for the raw velocity, can be either 'mm/s' or '0.1 mm/s', see velocityScaling above
- soundAbsorptionCoefficient: : sound absorption coefficient used for computing backscatter
- configBit: various configuration flags, see the Nortek documentation for details.

units: structure containing unit of measure for fields in structures above. For instance, units.pressure='decibar'.

units					
1x1 struct with 31 fields					
Field ▲	Value	Class	Min	Max	Mean
amplitude	'counts'	char			
avgInterval	's'	char			
backscatter	'rel. dB'	1x6 char			
meanBackscatter	'rel. dB'	char			
beamAngle	'degrees'	char			
blankingDistance	'm'	char			
boardFrequency	'kHz'	char			
cellSize	'm'	char			
compassHeading	'degrees'	char			
compassUpdateRate	's'	char			
correlation	'%'	char			
headFrequency	'kHz'	char			
lat	'degrees N'	char			
lon	'degrees E'	char			
magneticDeclination	'degrees'	char			
measurementInterval	's'	char			
pitch	'degrees'	char			
pressure	'decibar'	char			
range	'm'	char			
amplitudeVerticalRange	'm'	char			
roll	'degrees'	char			
samplingPeriod	'sec'	char			
soundSpeed	'm/s'	char			
soundAbsorptionCoefficient	'dB/m'	char			
temperature	'C'	char			
time	'Matlab time in UTC'	char			
u	'm/s'	char			
v	'm/s'	char			
velocity	'm/s'	char			
voltage	'Volts'	char			
w	'm/s'	char			

Oceans 3.0 API filter: extension=mat

Example: [NortekADCP1504_20101022T000021Z.mat](#)

NETCDF

NetCDF is a machine-independent data format offered by numerous institutions, particularly within the earth and ocean science communities. Additional resources are noted [here](#). For short duration searches (less than one day) or short duration data returned, the files will be concatenated, otherwise, expect one file per day.

The Nortek **NetCDF** file is extracted from the data contained in the above **MAT** file. The **NetCDF** file contains the following main variables: *u*, *v*, *w*, *time*, *binmap_depth*, where the *binmap_depth* variable here is distinct from the pressure sensor measurement in the **MAT** file (*data.depth*). When there is bin-

mapping performed on the data, the *binmap_depth* is calculated as *meta.depth - data.range*; it is the approximate water depth at which the measurement bins are located. If bin-mapping is 'none', then the *binmap_depth* and the *depth* are equal. *depth* is calculated as *meta.depth - data.amplitudeVerticalRange*, which is the range corrected for tilt. *depth* is used to plot the beam-averaged backscatter and any other beam-averaged values that are not bin-mapped. The structure and content of the netCDF file is very similar to that of the **MAT** file, including the variable naming. The **BDI** and **Nortek** netCDF files are intended to be very similar, even interchangeable with the same structure and definitions.

The **NetCDF** files are intended to be in compliance with the CF-1.7 standard. Depending on the compliance checker that is used the data may not meet this standard for the following reasons: The units dB are not defined, and the feature type of the data is not defined.

Oceans 3.0 API filter: extension=nc

Example: NORTEKSIGNATURE200066_20191027T000000Z_20191028T000000Z-Ensemble60s_binMapNone_PLAN0.nc

```
>> ncdisp('C:\Users\ONC\Test\NORTEKSIGNATURE200066_20191027T000000Z_20191028T000000Z-
Ensemble60s_binMapNone_PLAN0.nc')
Source:
      C:\Users\ONC\Test\NORTEKSIGNATURE200066_20191027T000000Z_20191028T000000Z-
Ensemble60s_binMapNone_PLAN0.nc
Format:
      netcdf4_classic
Global Attributes:
      Conventions              = 'CF-1.7'
      title                    = 'Ocean Networks Canada Nortek Profiler Data'
      institution              = 'Ocean Networks Canada'
      source                   = 'Nortek Profiler 55 kHz'
      history                   = 'Includes data extracted from raw output, minimally processed,
and processed data'
      references               = 'http://www.oceannetworks.ca/'
      comment                  = 'Detailed documentation: https://wiki.oceannetworks.ca/display
/DP/22'
      processing_comments      = 'Using a fixed true heading of 297 degrees from metadata.
Using internal pitch sensor. Using internal roll sensor. Raw data matrix, data.velocity, is the beam radial
velocities. A correlation screen of 50% was applied. Number of NaN beam-pings before: 0, number after: 259633
(1883520 total). Bin-mapping option set to 'None'. Average or fixed tilt is 2.7376 degrees. data.range is
instrument normal range to each bin centre. Transformed and rotated bin-mapped beam radial velocities to true
East-North-Up velocities, using instrument transform matrix and aforementioned heading/pitch/roll data for
rotation. adcp.backscatter (relative volume backscatter) calculated from 0.45*adcp.amplitude + 20*log10(adcp.
range) + 2*config.soundAbsorptionCoefficient*adcp.range, adcp.meanBackscatter calculated by beam-averaging the
volume backscatter (accounting for dB scale). adcp.meanBackscatter and adcp.backscatter have units of relative
dB. Ensemble file created by box-car average resampling to a ensemble interval of 60 seconds (raw data ensemble
interval: 6 seconds). data.amplitude was averaged normally, while data.backscatter and data.meanBackscatter
were averaged by first converting the logarithmic dB scale to linear, averaging, then converting back. For
information on the various processing steps applied, see https://wiki.oceannetworks.ca/display/DP/22 '
      CREATION_DATE            = '20210927T201637Z in QA'
      time_coverage_start      = '20191027T000130Z'
      time_coverage_end        = '20191027T235530Z'
      device_id                = 24420
      device_heading            = 297
      platform_depth           = 981
      site_name                 = 'CanyonAxis_IP_2018-06'
      device_name               = 'Nortek Signature55 Current Profiler 200066'
      location_name             = 'Barkley Canyon'
      search_id                 = 10439788
      firmware_version          = ''
      hardware_revision         = ''
      frequency                 = 55
      serial_number             = ''
      orientation               = 'Up'
      beam_angle                = 20
      adcp_setup_measurement_interval_sec = 6
      adcp_setup_average_interval_sec   = 240
      adcp_setup_number_beams          = 3
      adcp_setup_number_cells          = 109
      adcp_setup_number_pings          = 40
      adcp_setup_blanking_distance_meters = 0.2
      adcp_setup_cell_size_meters      = 10
Dimensions:
      time      = 1435
      depth     = 109
      latitude  = 1
      longitude = 1
Variables:
```

```

time
    Size:      1435x1
    Dimensions: time
    Datatype:  double
    Attributes:
        sdn_parameter_name = 'time'
        long_name           = 'time of measurement'
        units               = 'days since 19700101T000000Z'
        axis                = 'T'
        calendar            = 'gregorian'

binmap_depth
    Size:      109x1
    Dimensions: depth
    Datatype:  single
    Attributes:
        sdn_parameter_name = 'depth'
        long_name           = 'water depth of final velocity measurement bins'
        units               = 'meters'
        axis                = 'Z'
        positive            = 'down'
        comment             = 'Water depth for the u (east), v (north), w (up), velocityError
seawater velocity bins, accounting for bin-mapping'
depth
    Size:      109x1
    Dimensions: depth
    Datatype:  single
    Attributes:
        sdn_parameter_name = 'depth'
        long_name           = 'water depth of beam-averaged measurement bins, tilt-corrected'
        units               = 'meters'
        axis                = 'Z'
        positive            = 'down'
        comment             = 'Water depth of measurement bins corrected for tilt, applies to
meanBackscatter (beam-averaged backscatter)'
range
    Size:      109x1
    Dimensions: depth
    Datatype:  single
    Attributes:
        sdn_parameter_name = 'range'
        long_name           = 'range from transducer'
        units               = 'meters'
        axis                = 'Z'
        comment             = 'Range of measurement bins from the transducer, applies to all
parameters except: u, v, w, velocityError when bin-mapping is on and meanBackscatter'
latitude
    Size:      1x1
    Dimensions: latitude
    Datatype:  single
    Attributes:
        sdn_parameter_name = 'latitude'
        long_name           = 'latitude'
        units               = 'degrees_north'
        axis                = 'Y'

longitude
    Size:      1x1
    Dimensions: longitude
    Datatype:  single
    Attributes:
        sdn_parameter_name = 'longitude'
        long_name           = 'longitude'
        units               = 'degrees_east'
        axis                = 'X'

u
    Size:      109x1435
    Dimensions: depth,time
    Datatype:  single
    Attributes:
        sdn_parameter_name = 'eastward_sea_water_velocity'
        long_name           = 'eastward sea water velocity'
        units               = 'meters/second'

```

```

        _FillValue          = -9999999

v
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'northward_sea_water_velocity'
        long_name          = 'northward sea water velocity'
        units              = 'meters/second'
        _FillValue        = -9999999

w
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'upward_sea_water_velocity'
        long_name          = 'upward sea water velocity'
        units              = 'meters/second'
        _FillValue        = -9999999

meanBackscatter
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'sound_intensity_level_in_water'
        long_name          = 'acoustic doppler current profiler beam-averaged corrected acoustic
backscatter'
        units              = 'dB'
        _FillValue        = -9999999
        comment           = 'beam averaged and corrected for two-way beam spreading and
attenuation, not bin-mapped, use with tilt-corrected water depth'

intens_beam1
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name =
'signal_intensity_from_multibeam_acoustic_doppler_velocity_sensor_in_sea_water'
        long_name          = 'acoustic doppler current profiler return signal strength intensity
beam 1'
        units              = 'counts'
        _FillValue        = -9999999

intens_beam2
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name =
'signal_intensity_from_multibeam_acoustic_doppler_velocity_sensor_in_sea_water'
        long_name          = 'acoustic doppler current profiler return signal strength intensity
beam 2'
        units              = 'counts'
        _FillValue        = -9999999

intens_beam3
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name =
'signal_intensity_from_multibeam_acoustic_doppler_velocity_sensor_in_sea_water'
        long_name          = 'acoustic doppler current profiler return signal strength intensity
beam 3'
        units              = 'counts'
        _FillValue        = -9999999

corr_beam1
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name =
'beam_consistency_indicator_from_multibeam_acoustic_doppler_velocity_profiler_in_sea_water'

```

```

                long_name      = 'acoustic doppler current profiler correlation magnitude beam 1'
                units          = 'percent'
                _FillValue     = -9999999
corr_beam2
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name =
'beam_consistency_indicator_from_multibeam_acoustic_doppler_velocity_profiler_in_sea_water'
                long_name      = 'acoustic doppler current profiler correlation magnitude beam 2'
                units          = 'percent'
                _FillValue     = -9999999
corr_beam3
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name =
'beam_consistency_indicator_from_multibeam_acoustic_doppler_velocity_profiler_in_sea_water'
                long_name      = 'acoustic doppler current profiler correlation magnitude beam 3'
                units          = 'percent'
                _FillValue     = -9999999
velocity_beam1
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'radial_sea_water_velocity_away_from_instrument'
        long_name          = 'radial velocity beam 1'
        units              = 'meters/second'
        _FillValue         = -9999999
velocity_beam2
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'radial_sea_water_velocity_away_from_instrument'
        long_name          = 'radial velocity beam 2'
        units              = 'meters/second'
        _FillValue         = -9999999
velocity_beam3
    Size:          109x1435
    Dimensions:    depth,time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'radial_sea_water_velocity_away_from_instrument'
        long_name          = 'radial velocity beam 3'
        units              = 'meters/second'
        _FillValue         = -9999999
temperature
    Size:          1435x1
    Dimensions:    time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'sea_water_temperature'
        long_name          = 'sea water temperature'
        units              = 'K'
        _FillValue         = -9999999
pitch
    Size:          1435x1
    Dimensions:    time
    Datatype:      single
    Attributes:
        sdn_parameter_name = 'platform_pitch_angle'
        long_name          = 'pitch'
        units              = 'degrees'
        _FillValue         = -9999999
roll
    Size:          1435x1
    Dimensions:    time

```

```
Datatype:  single
Attributes:
    sdn_parameter_name = 'platform_roll_angle'
    long_name           = 'roll'
    units               = 'degrees'
    _FillValue          = -9999999

pressure
    Size:              1435x1
    Dimensions:        time
    Datatype:          single
    Attributes:
        sdn_parameter_name = 'sea_water_pressure'
        long_name           = 'sea water pressure'
        units               = 'dbar'
        _FillValue          = -9999999
```

Above example may be out of date. Recent changes include: added compass heading, updated units.

Discussion

To comment on this product, click *Add Comment* below.