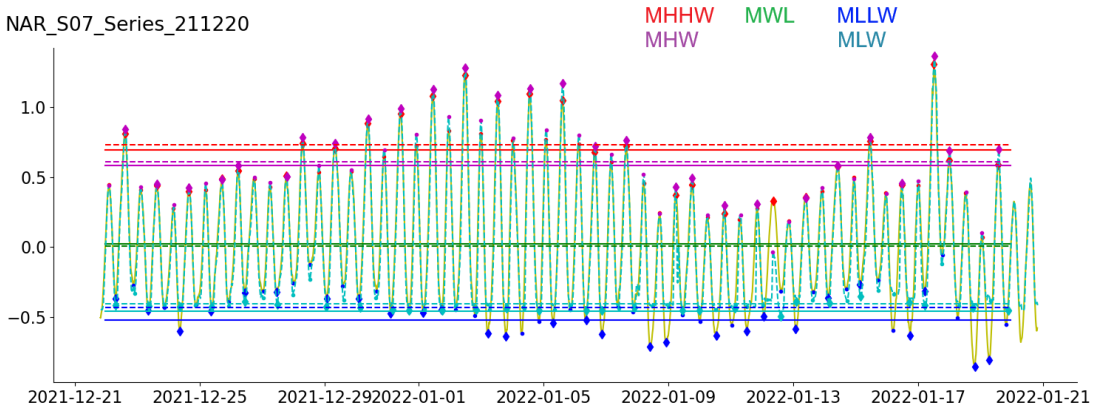
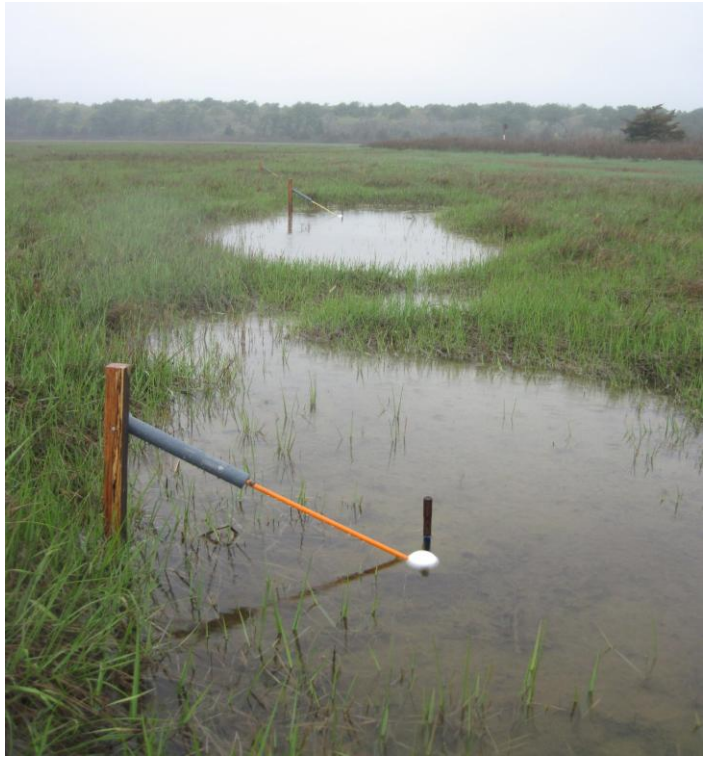
Hyperlocal
Estuarine water level monitoring





User manual and training guide

Updated September 2025 by
Robert Dunn
Original from Feb. 2025 by
Vitalii Sheremet and Megan Tyrrell

National Estuarine Research Reserve System Science Collaborative

Waquoit Bay National Estuarine Research Reserve

Great Bay National Estuarine Research Reserve

North Inlet - Winyah Bay National Estuarine Research Reserve

Narragansett Bay Research Reserve

Delaware National Estuarine Research Reserve

wellsreserve at laudholm

GTM

Funded by the NERRS Science Collaborative (2021 – 2023)

GOAL: Transfer of a low-cost tidal wetland water level monitoring system: hyperlocal calculations of inundation and tidal datums for understanding change and restoration planning

PIs: Vitalii Sheremet & Megan Tyrrell
End-user (lead): Robert Dunn

Lead Reserve: Waquoit Bay NERR

Six End-users, Recipient Reserves:

Jason Goldstein Ph.D., Wells NERR
Christopher Peter, M.S., Great Bay NERR
Kenneth Raposa Ph.D., Narraganset Bay NERR
Kari St. Laurent Ph.D., Delaware NERR
Robert Dunn Ph.D., North Inlet – Winyah Bay NERR,
Nikki Dix Ph.D., Guana Tolomato Matanzas NERR

Each reserve had 10-15 monitoring sites instrumented with either an arm-and-float or pressure sensor.

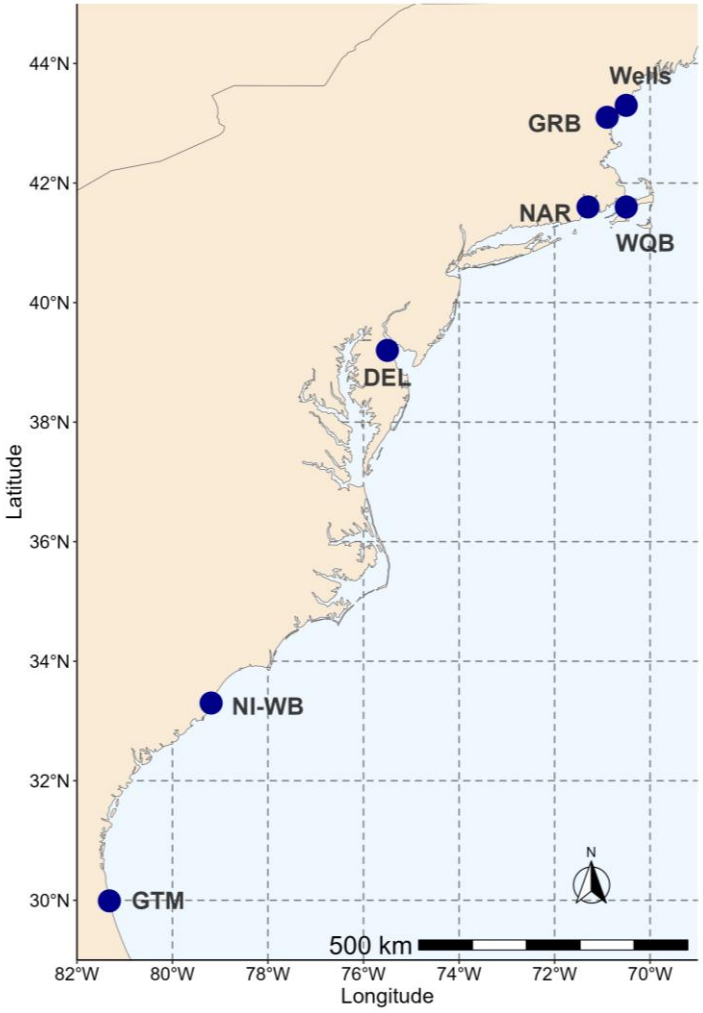
Original motivation:
New pools developing on the marsh platform in Waquoit Bay due to sea level rise.

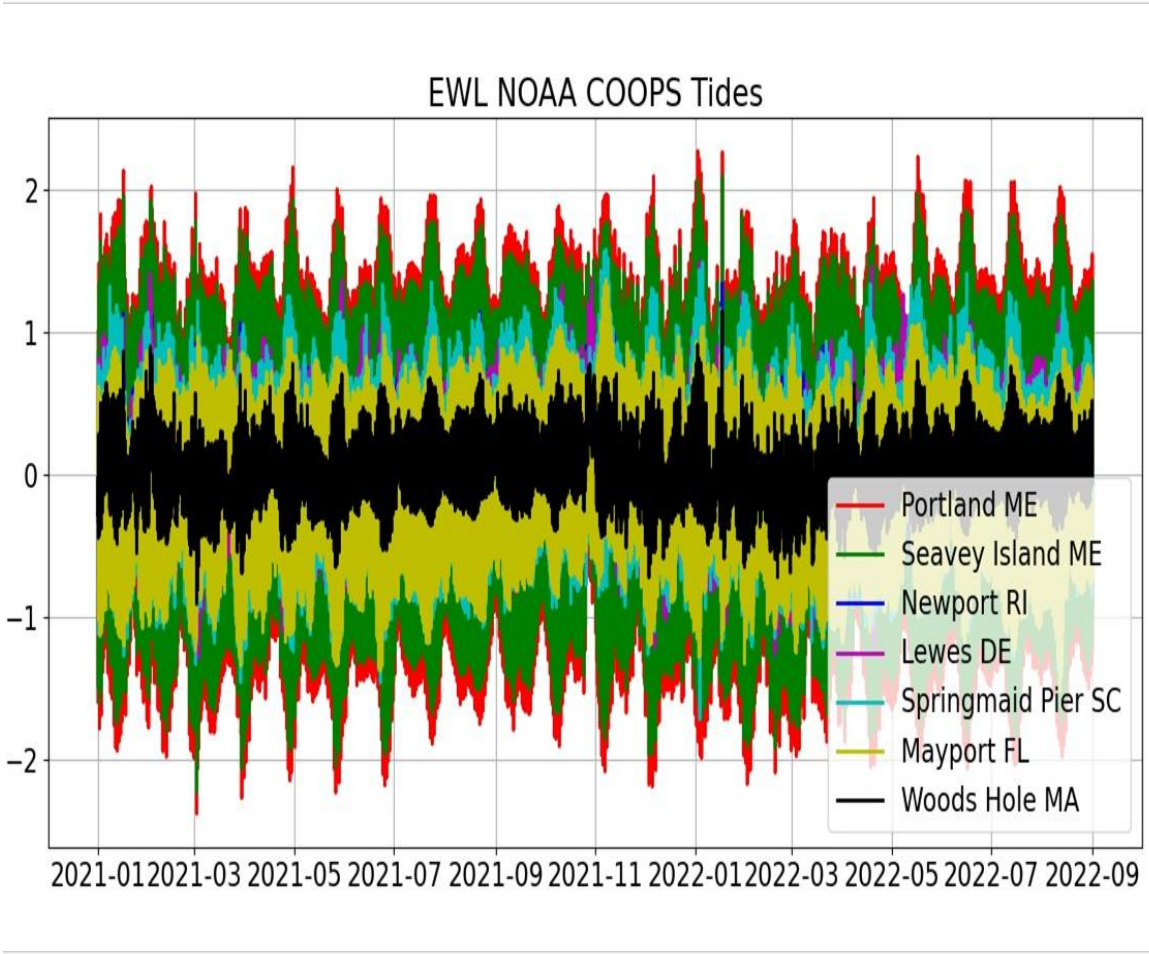
Why do we need water level information broadly?

- Understand sea level change effects on vegetation and sediment accretion
- Understand small scale variability in inundation time across marsh habitat ‘features’
- Restoration planning
- Plan hydrological manipulations – runnels, ditch remediation, placing culverts
- Better relate inundation frequencies to flora and fauna assemblages

GOAL: Transfer technology and understanding gained at Waquoit Bay NERR to 6 other Reserves.

Macro-tidal to micro-tidal regimes assessed (based on reference NOAA tide gauges)





Salt marshes encompass a diverse set of ‘features’ that may differ in value as habitat for animals due to variability in tidal inundation.

Small channel

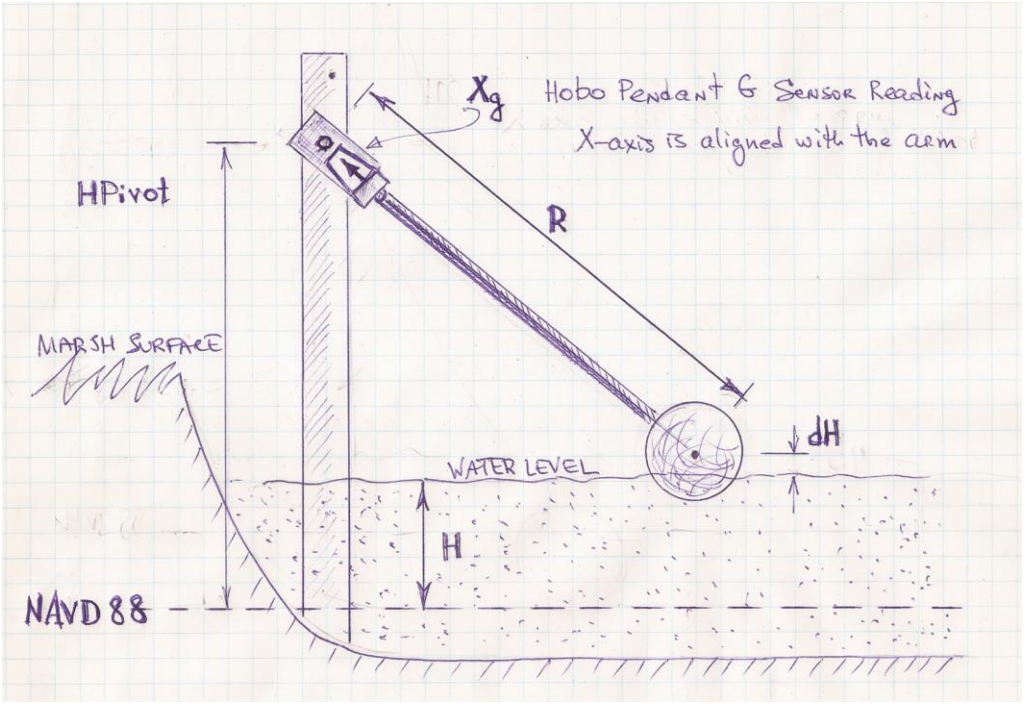
Pool

Subtidal creek

Marsh platform

FloatArm

A simple water level monitoring tool based on a standard accelerometer logger, designed by Vitalii Sheremet.



HOBO Pendant G Tilt meter accuracy 0.025, with an example arm length (R) = 50 cm, provides water level accuracy of 1.25 cm (actual accuracy depends on length of arm).

Hobo pendant G Part number: UA-004-64

FloatArm: Part Names

The FloatArm model HX58 consists of a fiberglass rod arm with a 5 cm diameter styrofoam ball at the end. A PVC logger is mounted at the pivot.

HX58 means that the arm length (distance between the pivot and the center of the ball) is 58cm.

The arm is attached to the 1.25 m wooden stake with 5/16” stainless steel bolt (the pivot point) and swings freely, allowing it to move with changing water levels.

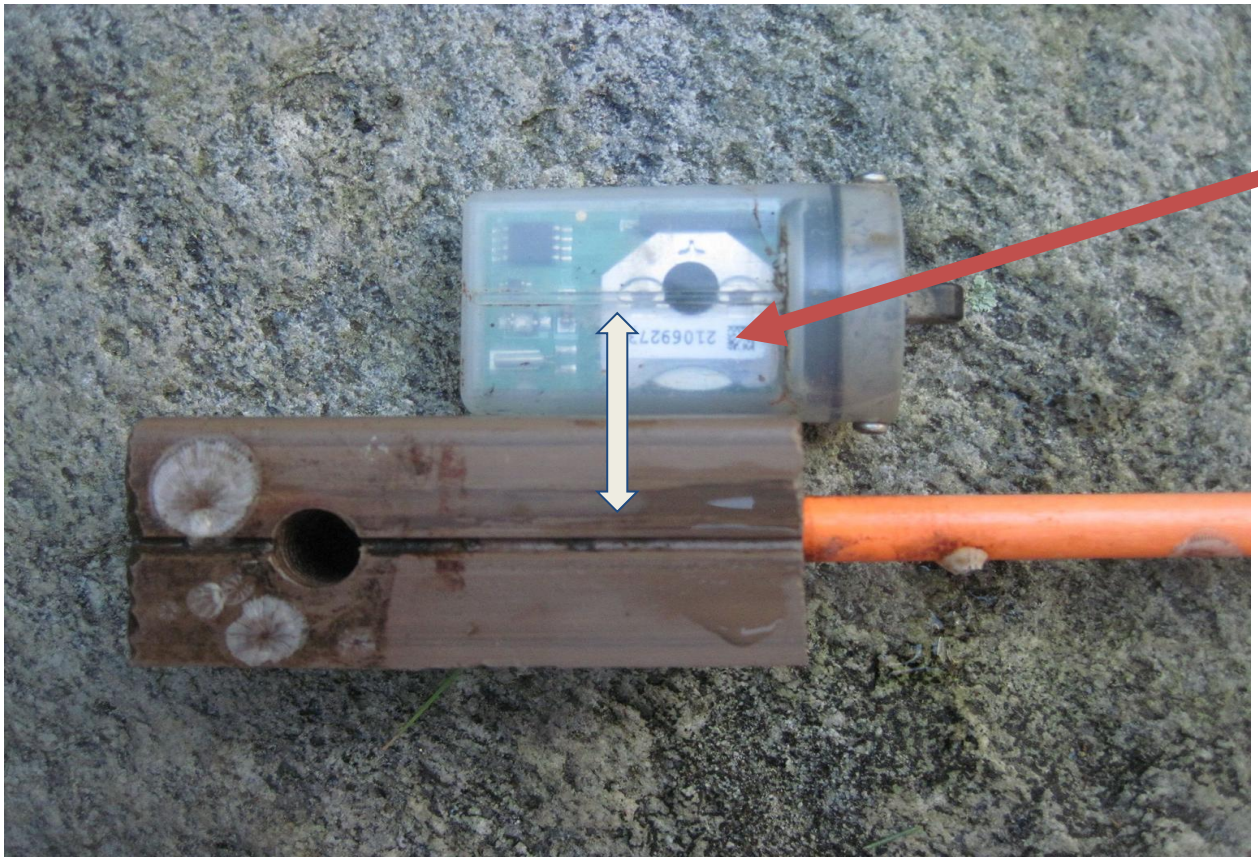
FloatArm: Part Names

This photo shows attachment of the FloatArm to the stake. Note placement of 5/16 stainless steel bolt, washer, and that the housing is completely wrapped in electric tape.

Above the main pivot is a stopper nail that prevents the arm from flipping to the other side (e.g. in a storm or other high water level situation).

The logger faces toward the Styrofoam ball and sits in a small groove notched into the mounting component (see next page).

FloatArm: Alignment of Arm and Logger

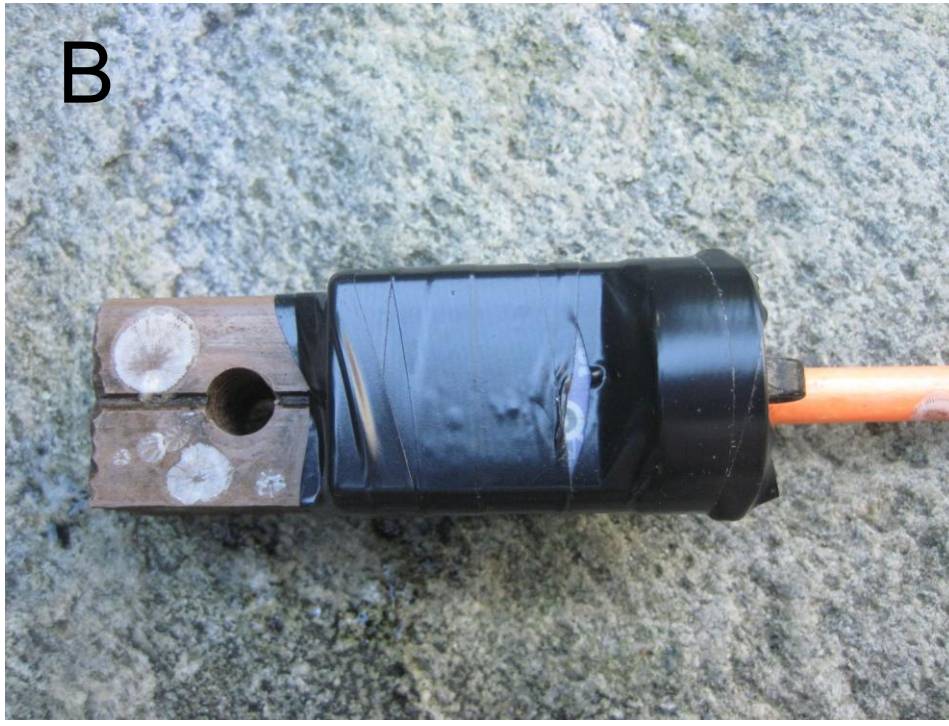


The FloatArm has a groove directed parallel to the arm and the logger housing has a ridge to allow for accurate alignment of the logger X-axis and the arm.
The X-axis (bottom, skinny end of logger) should be pointing towards the pivot (bolt).

FloatArm: Attaching Logger to Arm; Electric tape to protect from UV

The logger housing should be completely covered by electrical tape to protect from the sun's UV light.

Start with wrapping the electrical tape around the end of logger housing (A).



Then slide the logger into the groove until the rounded end is flush with the block. Continue wrapping the tape until the logger is secured (B). You may leave a small gap to see the red LED blinking during logging.

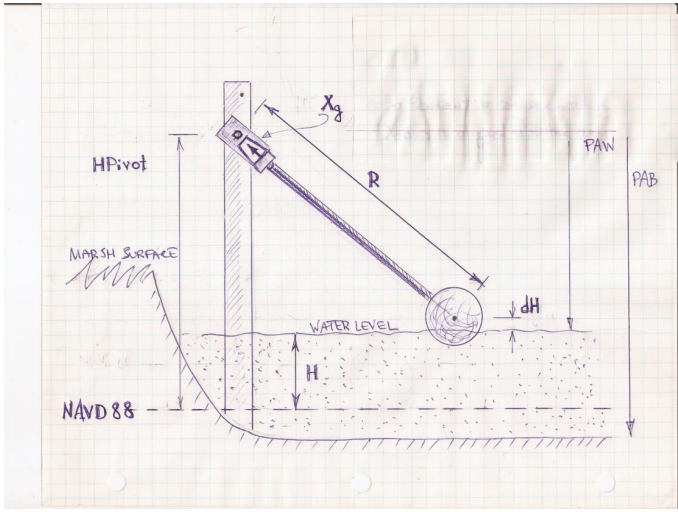
FloatArm: Field Installation

Tighten in the bolt with a ½” socket driver, but leave enough spacing for the arm to move freely. Here, the gap is left too large (for illustration).

After installing the stake at a proper height, attach the arm with a 5/16” stainless steel bolt, making sure there is a washer between the arm and the stake.

The proper height is typically selected slightly above the mean water level, Throughout the tidal cycle, the angle range of the arm movement should be +/-80 deg from the horizontal to prevent the arm getting stuck in the vertical position.

FloatArm: Principle of Operation



Conversion formula: Raw data from accelerometer to water level (m, relative to NAVD88)

$$H = Xg*R+dH+H_{pivot}$$

H - water level above NAVD88 (m).

Xg - the raw output from the logger sensor is Xg=(**X***__g__)/g, the projection of the gravitational acceleration onto the arm direction (with calibration applied). The standard orientation of the logger X-axis is toward the pivot, Xg is positive when the arm is up, negative when down.

R - the arm length (m), the distance from the pivot to the center of the ball.

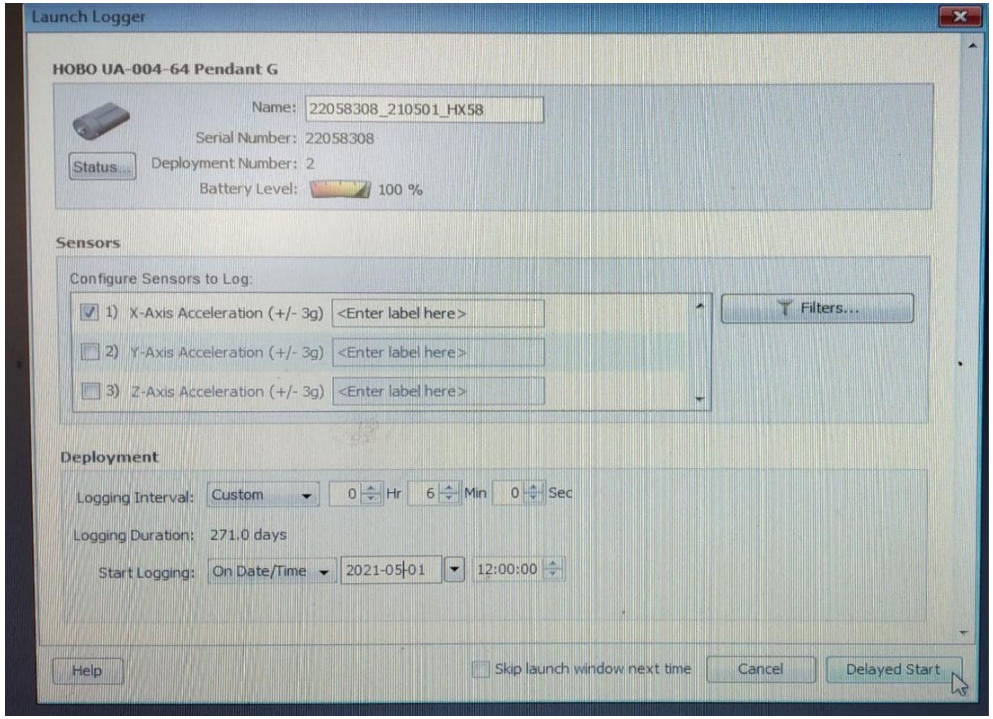
dH - a small correction to account for the difference between the water level and the center of the ball, typically several mm (see Appendix- Float ball correction).

H_{pivot} - is the elevation of the pivot center above the NAVD88 datum, determined during the RTK or optical survey. Usually, during survey the staff rests on the flat top of the hex bolt which is 6 mm higher than the center of the bolt: eg, RTK elevation = 1.0m, Hpivot = 0.994m

The water level H relative to NAVD88 is derived in the python script sh_FloatArm_HX.py

FloatArm: Starting Logger in HoboWare (Onset Inc.)

[FREE HOBOWARE DOWNLOAD]

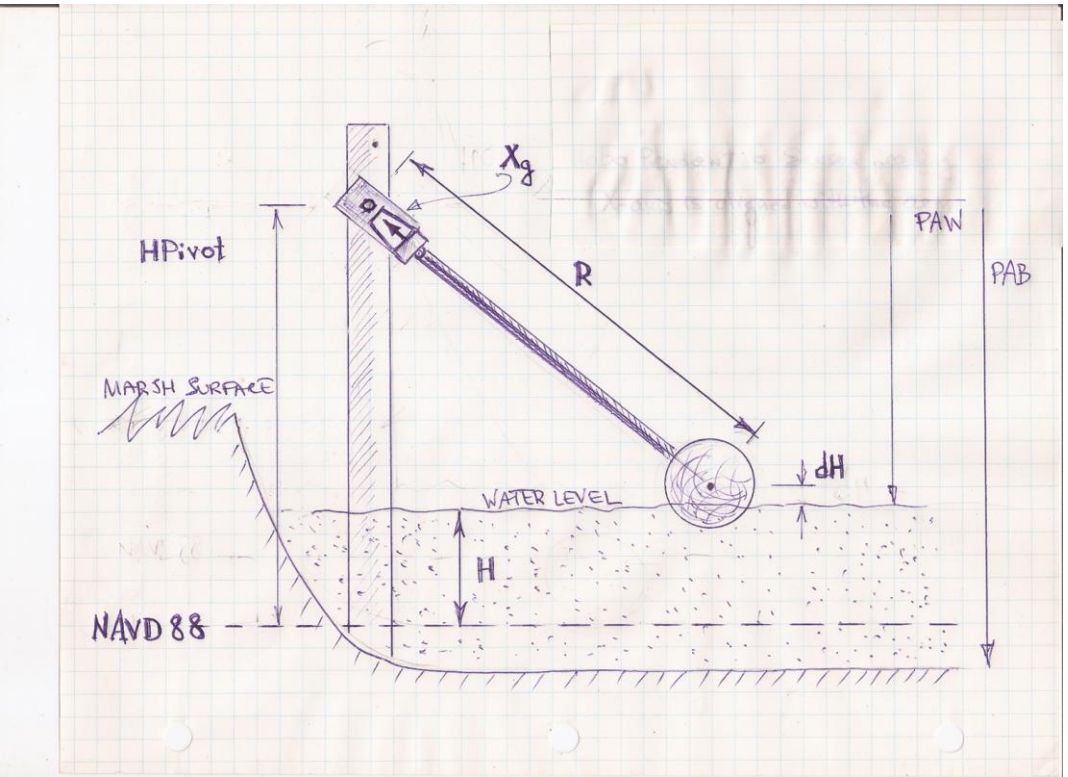


1. Recommended File Name Convention: SN_YYMMDD_MODEL_SITE
   Use Logger Serial Number SN and Launch date YYMMDD to uniquely identify the data file. MODEL such as HX58 and SITE codes help to organize the files.
1. Activate only X-axis logging.
2. Typical sampling frequency of 6 min will give 271 days of logging duration. If this level of sampling frequency is unnecessary, increase sampling interval time.
3. Select a delayed launch on a particular date and time.

FloatArm: Manual water level check- Distance from pivot to water/sediment surface

Convenient manual measurements that help to diagnose the operation of the FloatArm:

PAW - “Pivot above Water”, the distance from the pivot center to the water surface.
PAB - “Pivot above Bottom”, the distance from the pivot center to the stake/sediment interface, wherever it is located.

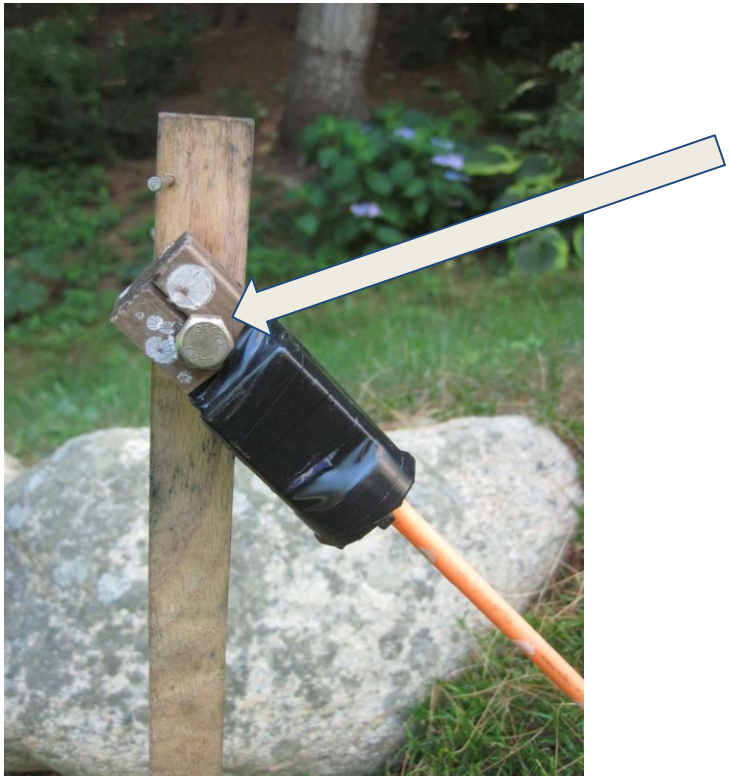


Occasional checks of PAW are used to diagnose that the logger is operating properly

PAB are used to check if the stake has moved, pulled up by ice.

PAW and PAB checks recommended every other month during deployment.

FloatArm: Pivot – Bolt Center, Surveying top of the bolt



When tightening the bolt, make sure that the flat side of the bolt head is horizontal. It is convenient to rest the staff on top of that side during the RTK or optical surveys. The flat side of the bolt head is 6 mm higher than the center of the bolt hole (height of the pivot), which is taken into account during post processing.

Absolute elevations from geodetic leveling or GPS-RTK survey (~2cm accuracy)

Elevation surveys of all loggers were conducted. As pictured here, optical bar code methods produce mm to sub-mm accuracy. RTK-GPS elevation was also measured at some Reserves, which may provide slightly less accurate values. Surveys were done to measure the parameter *Hpivot*, which allows for vertically controlled water level time series.

FloatArm: Recovery and Service

Recommended deployment period is 6 months.

This duration allows for high temporal resolution sampling without filling the memory of the logger.

Also, deployments longer than this risk extended periods of data loss if the logger has failed during deployment.

FloatArm: Recovery and Service

During recovery:

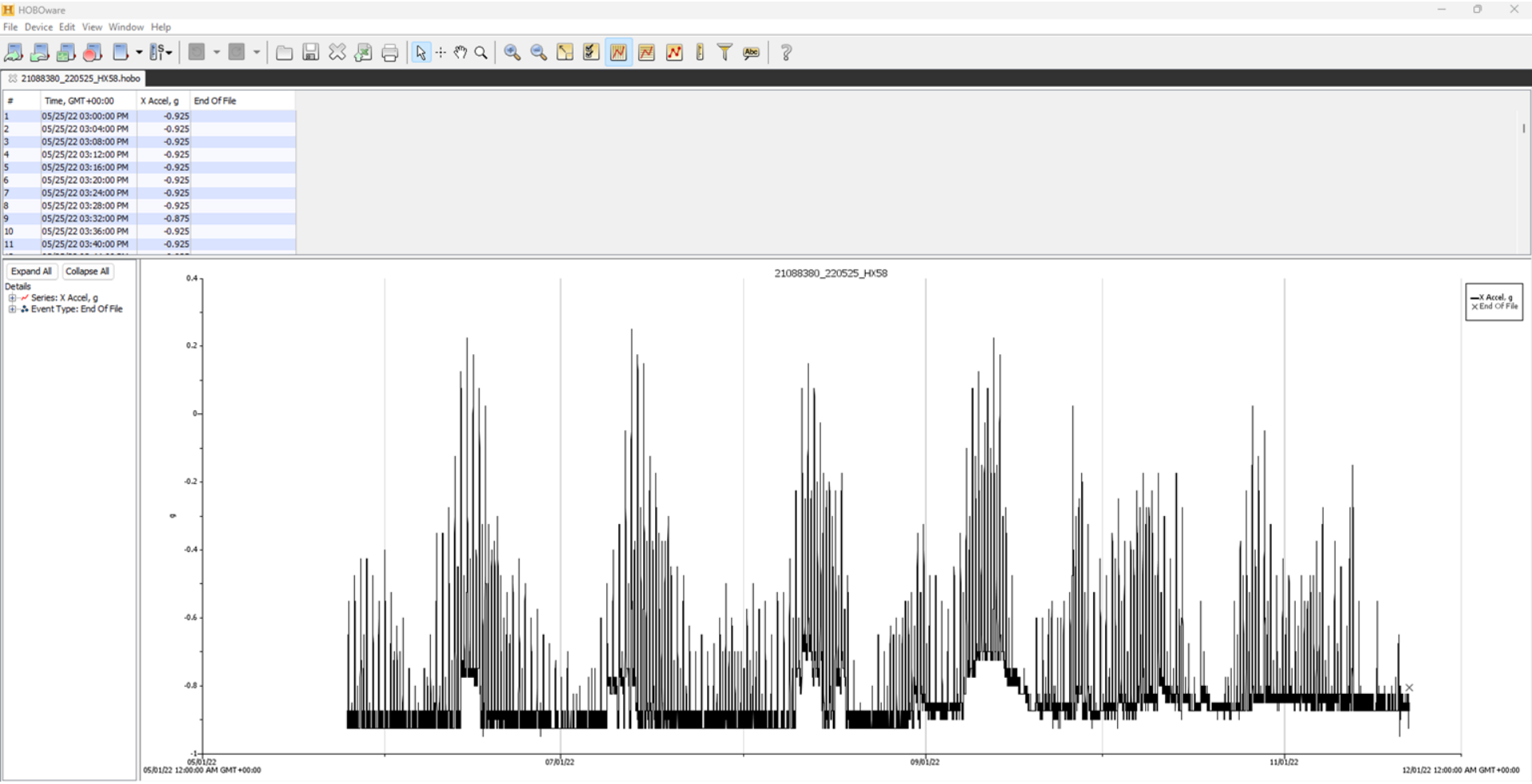
1. Remove float arm from stake by loosening bolt (pivot). Leave stake in place if possible.

2. Unwrap (cut) tape from accelerometer.

3. Use Hobo shuttle and Hoboware software on field laptop to download raw data from logger.

4. Export raw .hobo file to local laptop as .csv file. Be sure to note deployment site in file name for clarity.

5. Re-launch logger (see above page ‘Starting logger’). Re-deploy as described previously (wrap in tape, tighten bolt).



Example screen shot of HOBOware raw file from accelerometer download.
“X Accel, g” contains the data Xg needed for:
H = Xg*R+dH+HPivot
Xg is the variable needed for input to the python script. R is either 120 or 58.
HPivot is obtained from a survey and is fixed. dH is calculated within the script.

FloatArm: Data Download and Processing

The HOBO Logger raw data are saved in files SN_YYMMDD_MODEL.csv which have the following format with three columns:

"#","Date Time, GMT+00:00","X Accel, g (LGR S/N: 21088371, SEN S/N: 21088371)"
3361,2021-07-01 00:00:00,-0.275
3362,2021-07-01 00:04:00,-0.275
3363,2021-07-01 00:08:00,-0.275

where # is the scan number #, date time, and "X Accel, g" or Xg.

The conversion of Xg to water level is performed according to the formula
H = Xg*R+dH+HPivot by python function sh_FloatArm_HX.py

The output is saved in files SN_YYMMDD_MODEL_H.csv
Date_Time, Att (g), H (m)
2021-07-01T00:00:00,-0.238294,0.703626
2021-07-01T00:04:00,-0.238294,0.703626
2021-07-01T00:08:00,-0.238294,0.703626

where H is in meters above NAVD88. Att(g) is arm attitude, an auxiliary variable used for diagnostics. All times are UTC.

Tips: running scripts

If interested in running the scripts locally, recommend the anaconda distribution which has a large number of supported scientific libraries packaged with it.

The main python scripts and files all organized at the below 'XYZ' folder link:

https://drive.google.com/drive/u/0/folders/1fIGjVJTB-qLOZ_HmuqfRVZ89_iSWvAji

The main driver script is tidebytide_XYZ.py where a single reserve and deployment (DPL) variable combination are manually uncommented and file saved before running the script. For example, uncommenting the below lines will allow the script to process NIW for deployment 210422.

Tips: running scripts

The setting of the RESERVE and DPL values sets up for each site the csv filename (FN) based on choices selected and their associated datetime format (sh_helper.time_format) and folderpath (DATAFOLDER), associated SiteCode and SiteName ID, BoltCenter value, associated HFlood(HFloodRef,HFloodSite) values, and 'green-dot' manually chosen values from review of the produced 'Flood' graph

```
RESERVE='NIW'
DPL='210422'
#DPL='211028'

#RESERVE='GTM'
#DPL='210427'
#DPL='211022'
```

```
if RESERVE=='NIW' and DPL=='210422':

#############################################################################
######
    DATAFOLDER='NIW/2021_NIW_rawfiles/';
    sh_helper.time_format = '%Y-%m-%d %H:%M:%S'

    TBL=[
    #note BoltCenter is -6 mm of measured RTK/Elev value, for example Elev
= 1.000, BoltCenter = 0.994
    # FN,SiteCode,SiteName,BoltCenter
    '10331920_210422_HX1193.csv','OC','Oyster Creek',0.178,
    '21069273_210422_HX58.csv','BC-HX58','Blythe Creek HX58',0.291,
    '21069272_210422_HX58.csv','CB','Clambank',0.530
    ]

 HFlood={
 # Site: (HFloodRef, HFloodSite)
 'OC' :(-1.0,-0.75),
 'BC-HX58' :(0.15,0.85),
 'CB' :(0.0,0.2)
 }
```

Tips: running scripts

From the command line to run the script:
C:\XYZ>python tidebytide_XYZ.py

This will produce several outputs for each site and in summary.

For each site deployment, the below graphs are produce comparing local site to reference site data:

NIW_OC_Series_210422.png - 'Series' graph comparing time series data

NIW_OC_TimeLag_210422.png - 'TimeLag' graph showing tide time lag difference between local and reference site

NIW_OC_Flood_210422.png - 'Flood' correlation graph showing flood elevation differences between local and reference site

The filename convention allows related graphs to be sorted by filename and viewed/compared in succession.

The summary csv file for all sites for a deployment is listed with the below filename convention

TideByTide_NIW_210422_wide.csv

Tips: running scripts

Within this '...wide.csv' file, the 'Ref_' column tide datum values are static and pulled from a reference website lookup and manually entered at the end of the tidebytide_XYZ.py script, for example
https://tidesandcurrents.noaa.gov/datums.html?datum=NAVD88&units=1&epoch=0&id=8661070

The tidebytide_XYZ.py script is run (or rerun) once for each deployment, changing the deployment (DPL) that is uncommented to choose the deployment analyzed. Once all deployments have been run/processed, there will be an associated '...wide.csv' file for each one which are then subsequently used when running the below command to produce a 'combined_metrics' file for the target chosen reserve (NIW in the command below producing combined_metrics_all_sites_NIW.pdf). The 'combined metrics' file shows graphs of the chosen metrics across all deployments for comparison.

C:\XYZ>python combined_metric_all_sites.py --metrics Ref_MHHW,Ref_MHW,Site_MHHW,Site_MHW,FloodFrac,HighTideLag_min,HighTideLagStd_min,HFloodRef,HFloodSite --out combined_metrics_all_sites.pdf --target NIW

Tips: running scripts

Reference station data

The scripts currently get their station reference tide data from previously downloaded data stored as .npy files (a binary/compressed file format for python arrays), one file per associated station. These current reference station data files were generated for the years 2021 to 2023, but other stations or years can be provided using the sh_tide_datums_ref.py and changing the below lines of code to match the time range and stations of interest. The script when run will save the stations for the time range listed as .npy files for reference in the tidebytide_XYZ.py file and also show plots of the reference stations data.

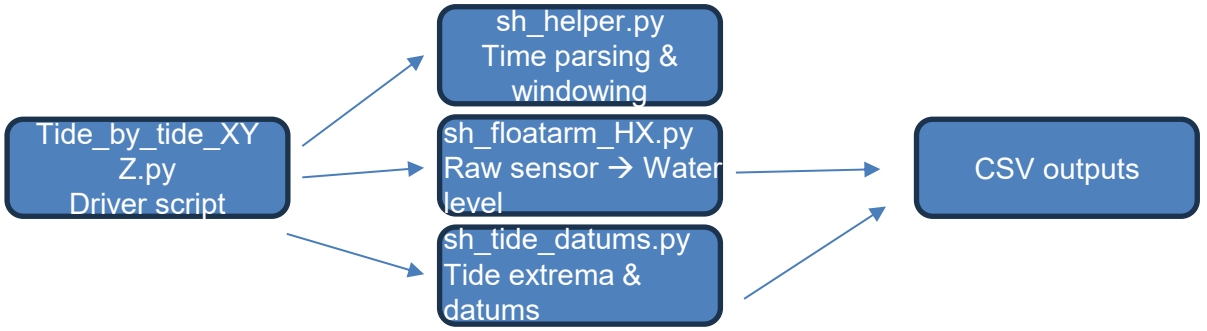
```
t1=datetime(2021,1,1);t2=datetime(2024,1,1)
FN2='_WL_NAVD88_2021_2023.npy'
DN=True # True - Download Data from NOAA Site first time; False - Read
from file
#DN=False  # p02 - work with downloaded data

TBL=[
'8418150','Portland ME'      ,'WEL','r',
#'8419870','Seavey Island ME'  ,'GRB','g', # starting 2020-06-11
#'8452660','Newport RI'       ,'NAR','b',
#'8557380','Lewes DE'         ,'DEL','m',
#'8661070','Springmaid Pier SC','NIW','c',
#'8720218','Mayport FL'       ,'GTM','y',
#'8447930','Woods Hole MA'    ,'WQB','k', # plot last <- smallest
range
]
```

Tips: running scripts

Note also the section of code in tidebytide_XYZ.py where each reserve has station datums manually copied/listed from their matching webpage data

```
# CUSTOM - Station specific references, FN2, STAID, datum values
# note: the .npy file is produced by sh_tide_datums_ref.py
# note: epoch(1983-2001) datums, sub id=STAID at
https://tidesandcurrents.noaa.gov/datums.html?datum=NAVD88&units=1&epoch=0
&id=8661070

FN2='_WL_NAVD88_2021_2023.npy'

NAVD88=     0.000 #North American Vertical Datum of 1988
CFGFOLDER = None #default if none


if RESERVE == 'NIW':
    STAID='8661070';STALOC='Springmaid Pier, SC'

    MHHW=0.745    #Mean Higher-High Water
    MHW=0.626     #Mean High Water
    MTL=-0.139    #Mean Tide Level
    MSL=-0.136    #Mean Sea Level
    DTL=-0.108    #Mean Diurnal Tide Level
    MLW=-0.905    #Mean Low Water
    MLLW=-0.962   #Mean Lower-Low Water
```

Tips: running scripts



Further script, function, workflow details

For additional script, function and workflow details see Appendix 'Workflow Documentation for Tide-by-Tide Scripts'

Note also that 2 of the 3 main files (tidebytide_XYZ.R, sh_FloatArm_HX.R also sh_helper.R) were successfully converted to R language format, where interested in reviewing in that format. Currently they will produce the water level height conversions correctly, but not the datums. The third datum python file(sh_tide_datums_Rwrapper.R) could be converted to R if there is strong interest. The method of conversion was using chatGPT to iteratively translate from python to R, providing feedback to chatGPT by spot-checking where functions or outputs were not matching from the original python script runs.

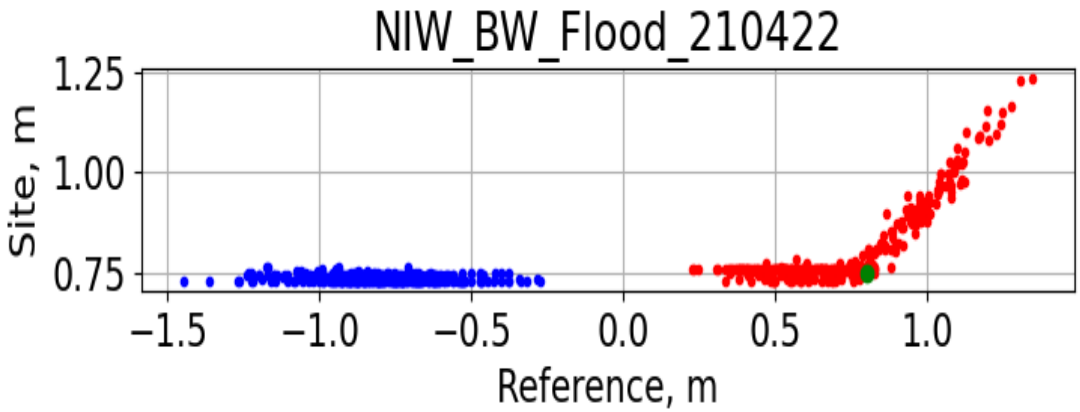
Note also the python script preview_plots.py was used to get a simple time series plot preview of csv files when csv files with flatlining or bad data were causing the regular scripts to fail due to this bad data.

Tips: running scripts to generate water level time series.

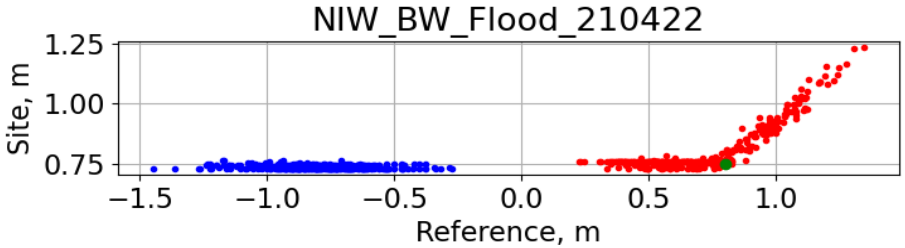
In the tidebytide_XYZ.py script (screenshot below), each site/deployment has an associated **HFlood** point value associated with it. When the script is initially run, this HFlood value can be set to some arbitrary value such a 1 as it will be need to be manually chosen based on the produced 'Flood' graph review for the site/deployment.

For example, the below graph was produced and the HFlood (green dot) value was estimated to say 'BW' :(0.8,0.75) based on estimating where the right hand graph starts an upward slope.

```
if RESERVE=='NIW' and DPL=='210422':

##########################################################################
######
    DATAFOLDER='NIW/2021_NIW_rawfiles/';
    sh_helper.time_format = '%Y-%m-%d %H:%M:%S'

    TBL=[
    #note BoltCenter is -6 mm of measured RTK/Elev value, for example Elev
= 1.000, BoltCenter = 0.994
    # FN,SiteCode,SiteName,BoltCenter
    '10331920_210422_HX1193.csv','OC','Oyster Creek',0.178,
    '21069273_210422_HX58.csv','BC-HX58','Blythe Creek HX58',0.291,
    '21069272_210422_HX58.csv','CB','Clambank',0.530
    ]

    HFlood={
    # Site: (HFloodRef, HFloodSite)
    'OC' :(-1.0,-0.75),
    'BC-HX58' :(0.15,0.85),
    'CB' :(0.0,0.2)
    }
```



Tips: running scripts to generate water level time series.



What am I looking at?

Red points/curve: Represents fraction of high tides above a given elevation.
For each potential flood threshold (y-axis, elevation), the red point shows the proportion of **tide cycles** that reached or surpassed that level.

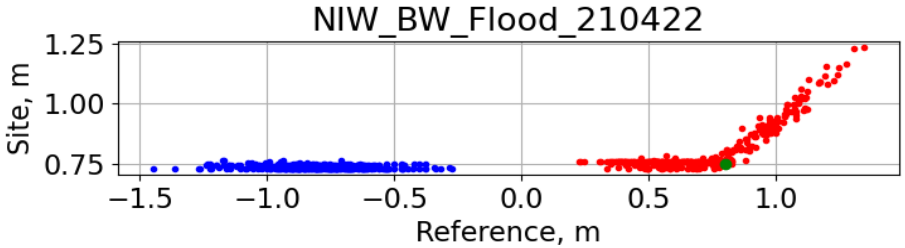
That's why it looks **flat** at low elevations (100% exceedance: every tide goes above that) and then starts an **upward slope** as you get higher elevations that only *some* tides reach.

Blue points/curve: Represents the reference station's fraction for comparison.
Same idea (fraction of high tides above a given elevation), but computed from the **NOAA reference series**. The blue gives you a baseline — so you can see if your marsh station floods more often or at different levels than the reference tide gauge.

How they're used together: Blue curve (reference): a stable, long-term tide series.
Red curve (site): your logger-derived water levels at that deployment.

Where the red curve diverges from the blue, you learn about **local flooding differences** — like lower/higher marsh surfaces or altered hydrology. The **HFlood green dot** is placed along the **red curve**, since that's site-specific.

Tips: running scripts to generate water level time series.



Interpretation:
In the **Flood plot**, the **red curve** (fraction of cycles above threshold) stays **flat with zero slope** at first — meaning *every tide* exceeds those low elevations. Then at some higher elevation, the curve begins an **upward slope** — because fewer and fewer tides exceed those levels.

The **green dot (HFlood)** is set at this *transition point*: the lowest elevation where the flat portion ends and the curve starts to rise.

General rule
HFlood should be set at the elevation where the exceedance curve (red line) transitions from a flat horizontal line to the beginning of an upward slope. This represents the threshold between levels that are *always flooded* and levels that are only *sometimes reached*.

Why this upward break matters
Flat section (left side) = levels below the marsh surface/bank, always inundated, not a true "flood" threshold. **Upward slope** = levels only reached during higher tides, capturing meaningful flood events.

HFlood at the break ensures you're anchoring the analysis to the *ecologically relevant surface elevation* at that site/deployment.

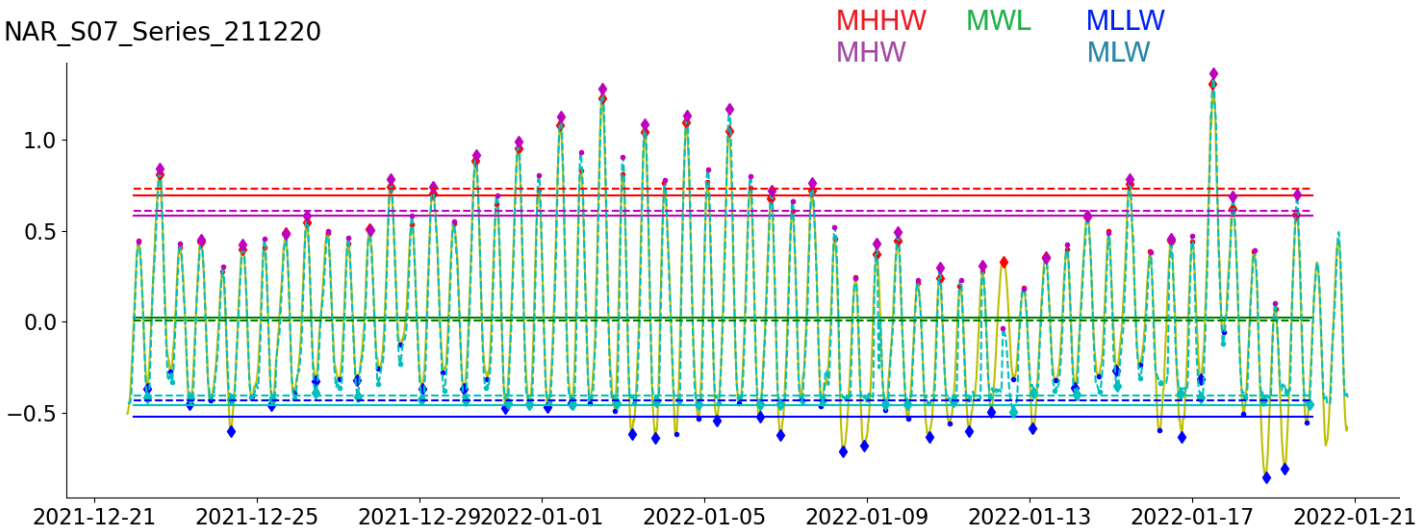
FloatArm Datum Calculations: Tide-by-tide analysis

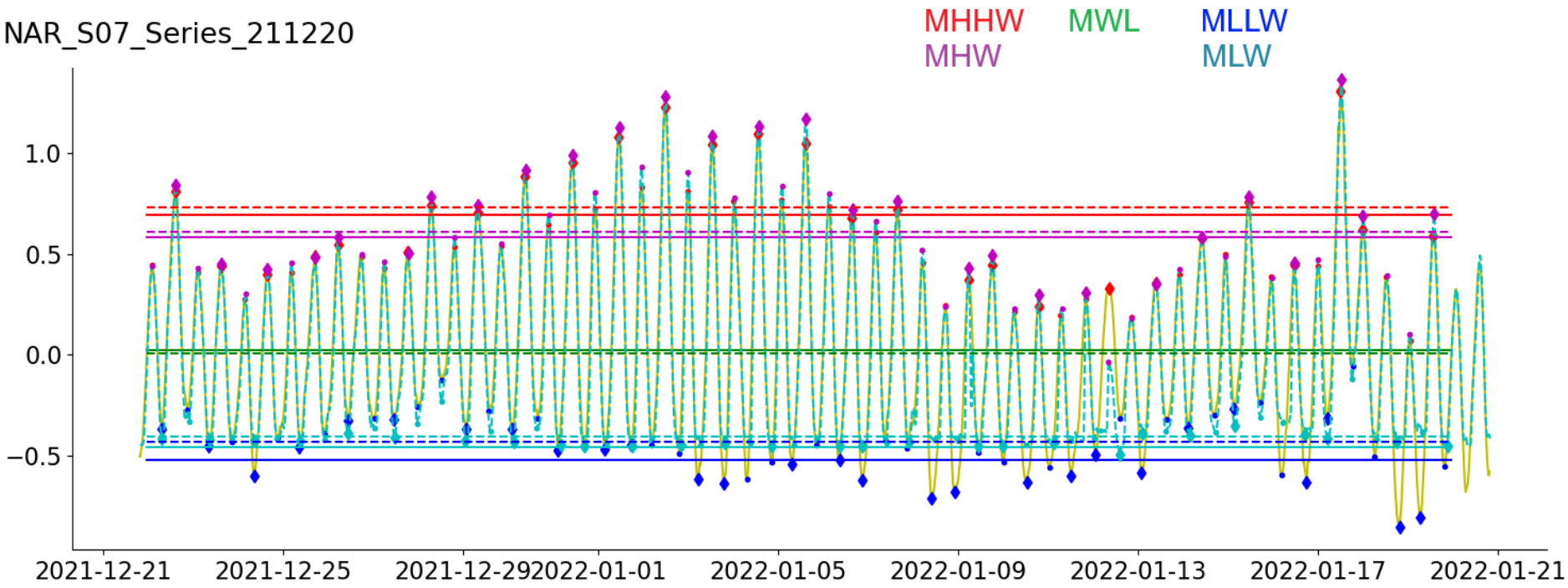
6 mo. hyperlocal deployments are insufficient to calculate tidal datums because of the ~19 year tidal epoch.

To get around this problem, we can use a tide-by-tide analysis, leveraging water level data from ‘reference’ NOAA tide stations that have sufficiently long time series to estimate datums.

A different, local ‘reference’ station is used for each Reserve.

In the image to the right, the green line is the hyperlocal float arm water level. It is plotted on top of the yellow line representing the NOAA tide gauge station water level. This is a visual representation of the ‘tide-by-tide’ analysis. See next slide for larger version.





All measurements are relative to NAVD88.

Dashed horizontal lines: hyperlocal datums
Solid horizontal lines: NOAA reference station datums
Datum colors are shown in the figure legend.

Diamonds: daily higher high tide and lower low tide
Circles: daily lower high tide and higher low tide

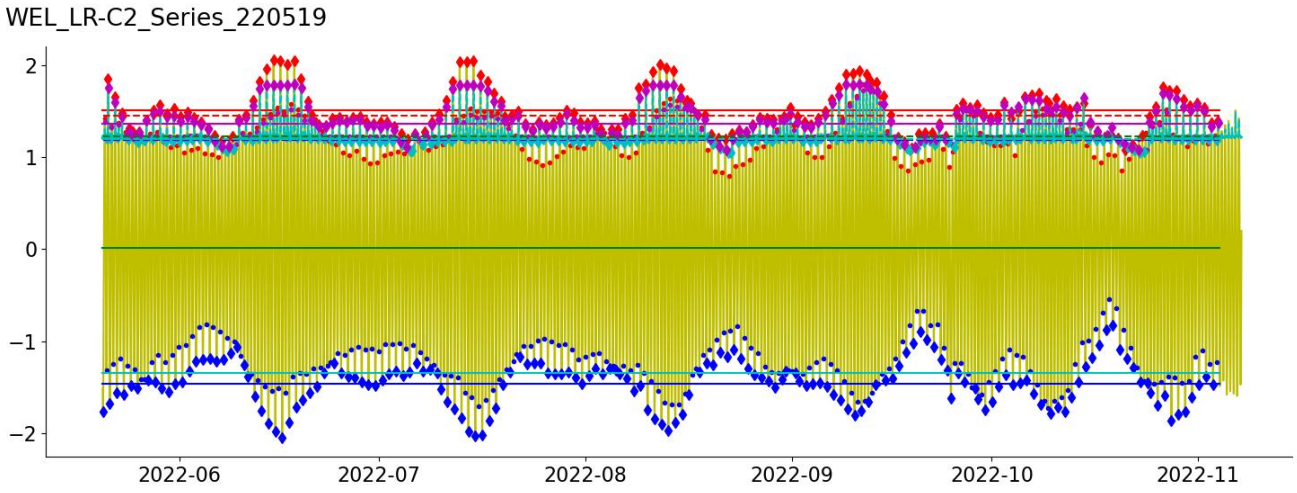
FloatArm Datum Calculations: Tide-by-tide analysis

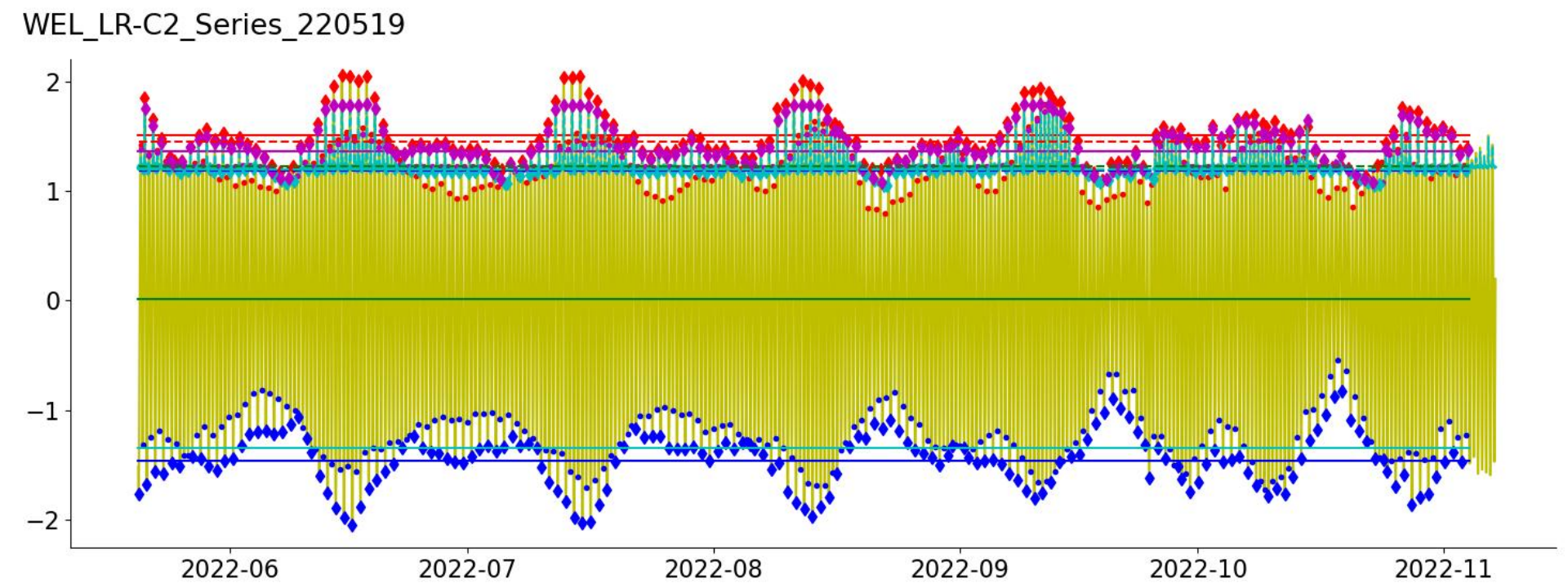
For each ~6 month hyperlocal deployment, we pull the corresponding 6 month water level time series from the NOAA reference station.

We then line up the hyperlocal water level time series with the reference water levels and calculate the difference in heights for each high tide and low tide.

*Note: low tides are generally unreliable from hyperlocal due to clipping of the water level data when water drains from the marsh platform or from tidal creeks.

In the image to the right, note how the hyperlocal water level data is clipped during much of the tide cycle with the exception of high water periods. As a result, low tide datums are unreliable. See next slide for larger version.



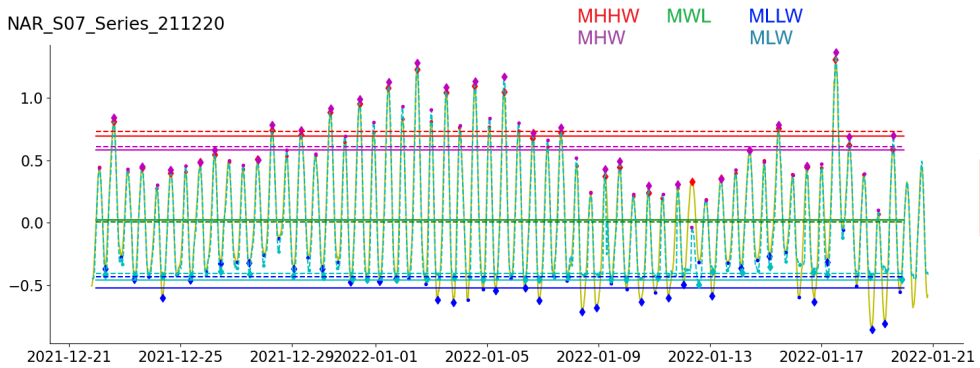


Dashed horizontal lines: hyperlocal datums
Solid horizontal lines: NOAA reference station datums
Datum colors are same as previous time series image.
Note that hyperlocal datums for lower water levels are incorrect due to clipping.

Diamonds: daily higher high tide and lower low tide
Circles: daily lower high tide and higher low tide

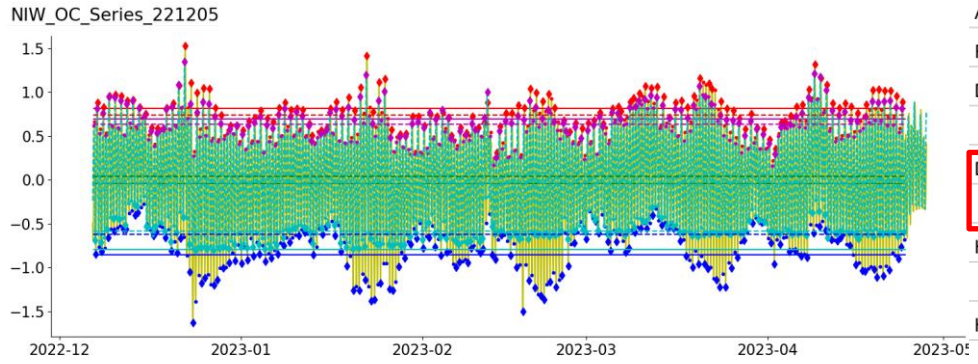
Example Tide-by-tide visualization and output

1 month



S07				
Analysis Interval: 2021-12-21 19:40:00 2022-01-20 19:40:00				
FloatArm File: 21088365_211220_HX120.csv				
Datums Ref: MHHW	MHW	MLW	MLLW	MTL
0.6921	0.5809	-0.4567	-0.5176	0.0621
Datums S07: MHHW	MHW	MLW	MLLW	MTL
0.729	0.61	-0.406	-0.4281	0.102
HFloodRef	HFloodSite	FloodFrac		
-0.367	-0.465	0.81365924		
HighTideLag_min	HighTideLagStd_min			
7.9	14.45			

6 months



OC				
Analysis Interval: 2022-12-06 16:00:00 2023-04-28 04:00:00				
FloatArm File: 10331920_221205_HX1193.csv				
Datums Ref: MHHW	MHW	MLW	MLLW	MTL
0.815	0.6938	-0.793	-0.8541	-0.0496
Datums OC: MHHW	MHW	MLW	MLLW	MTL
0.7382	0.6316	-0.5832	-0.6189	0.0242
HFloodRef	HFloodSite	FloodFrac		
-1	-0.75	0.96776423		
HighTideLag_min	HighTideLagStd_min			
-163.1	13.39			

Key parameters circled in Red. See next page for descriptions.

FloatArm Datum Calculations: Tide-by-tide analysis

Key parameters estimated:

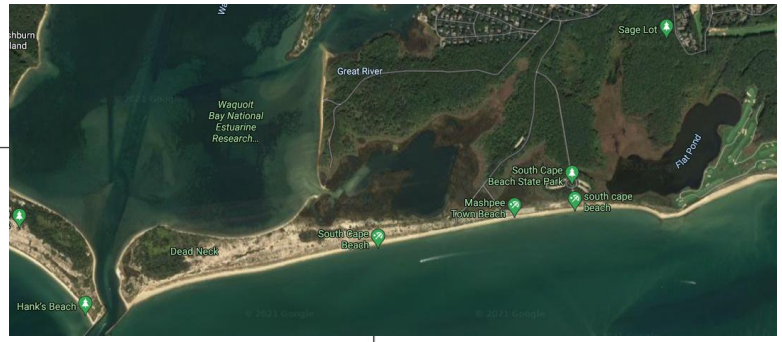
Flood fraction – proportion/% of time that tide is flooding the site; i.e., water is above some baseline (Hflood), which could be the sediment surface or water surface in a pool.

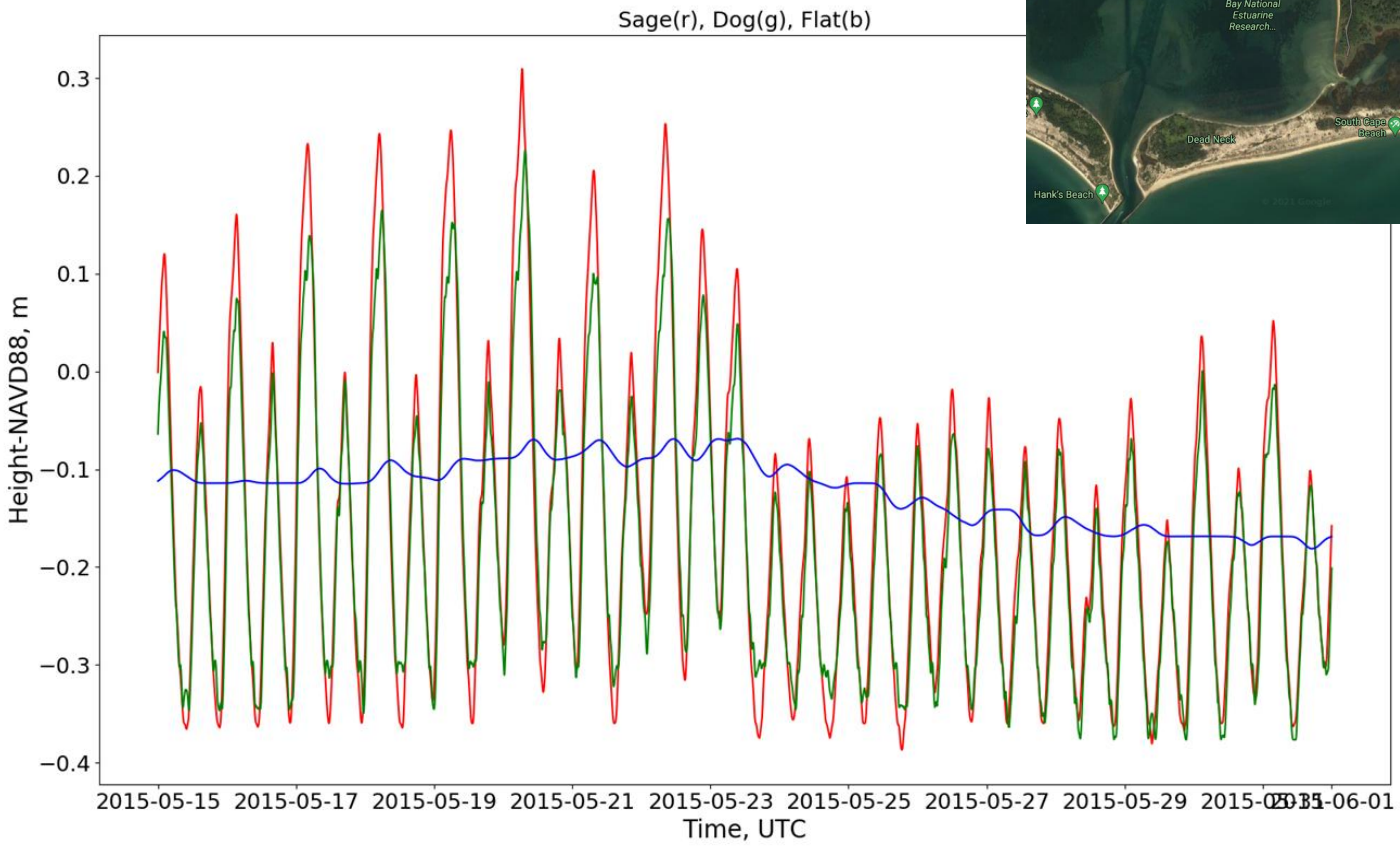
Mean higher high water (MHHW) – average of daily higher high tides

Mean high water (MHW) – average of all high tides observed

All other datums (MSL, MLW, MLLW) are heavily impacted by clipping during low water periods and so are not broadly useful for our hyperlocal sites.

Example of water level variability across marsh features in Waquoit Bay:
Sage Lot Marsh (red),
tidal range reduced in Dog Marsh (green),
or nearly absent in Flat Pond (blue) due to flow restrictions





Realized outcomes and products

Multiple deployments of armfloat and pressure sensors across marsh features within 7 NERR sites.

Computation of tidal datums and inundation statistics for each Reserve.

Comparison of water level variability (relative to reference stations) across marsh features, pooling among Reserves.

Training manual for future re-use (this guide).

Peer-reviewed manuscript with all participants invited as co-authors.

Summary

The FloatArm is an inexpensive and relatively robust instrument that is easy to deploy.

A large number of FloatArms can be deployed simultaneously to study flooding and draining of individual features within a marsh.

Pools, restricted features, and other sub-habitats have non-sinusoidal inundation curves, necessitating a tide-by-tide analysis using reference tide stations and elevation data.

Appendix.
Correction of float ball height

FloatArm: Ball Flotation, Arm Immersion Correction

During normal operation, when the arm is above water, the center of the ball is 4 mm below water. When water level is higher than the pivot, the arm is underwater leading to additional buoyancy force; in this case, the center of the ball floats 4 mm higher than the surface of water. The python script sh_FloatArm_HX.py makes the arm submersion correction (dH) automatically and derives the water level relative to pivot depending on the orientation of the arm. See next slide for details.

FloatArm: Ball Flotation, Arm Immersion Correction cont'd

A 1 ~4 mm distance between water level and center of ball

The python script sh_FloatArm_HX.py makes the arm submersion correction (dH) automatically and derives the water level relative to pivot depending on the orientation of the arm:

dH=(a1+a2)*0.5+(a2-a1)*0.5*np.tanh((Att-AttSub)/w) ,

where a1 and a2 parameters indicating the water level relative to the center of the ball for two different attitudes, w - width of the transition region.

Appendix.
Workflow documentation for tide-by-tide scripts

Big-picture workflow

1. **tidebytide_XYZ.py (driver/orchestrator)**
- Configures a deployment (RESERVE/DPL), loads reference data, loops through sites, and calls
processing functions.
- Uses sh_helper.extract_t0_t1_mld, sh_FloatArm_HX, and sh_tide_tidebytide.
2. **sh_FloatArm_HX.py (sensor → water level)**
- Parses HX logger data, applies calibration, converts accelerometer attitude to water level.
3. **sh_tide_datums.py (tide analysis)**
- Provides routines for tide extrema detection, datums, and tide-by-tide site vs reference comparisons.
4. **sh_helper.py (helpers/config)**
- Provides time_format and extract_t0_t1_mld to establish analysis windows from raw CSVs

Script Roles and Functions

tidebytide_XYZ.py
- Main driver; sets up RESERVE/DPL config and tables.
- Defines NOAA reference datums and loads reference time series.
- Opens CSV outputs (narrative and wide).
- Loops through site entries, calls helper and analysis functions, writes results.

sh_FloatArm_HX.py
- **sh_getcfg(FN)**: Load calibration coefficients from .cfg file.
- **sh_FloatArm_HX(...)**: Read logger CSV, smooth, clip window, convert attitude to water level (H), apply corrections, save _H.csv.

sh_tide_datums.py
- **sh_tide_datums**: Compute standard tidal datums (MLLW, MLW, MSL, MTL, MHW, MHHW).
- **sh_tide_mdatums**: Compute monthly means over multi-lunar cycles.
- - **sh_tide_tidebytide**: Compare site vs reference tide-by-tide; flood fraction and lag.
- - **sh_tide_find_M2_extrema***: Algorithms for detecting minima/maxima per tide cycle.
- sh_helper.py
- - **time_format**: Global strftime string, set per deployment.
- - **extract_t0_t1_mld**: Parse CSV start/end times, adjust to integer multiples of lunar cycles, return (t0, t1, MLD, ND).

Inputs/Outputs

- Inputs: Raw HX CSVs, per-site pivot elevations, config files, NOAA reference series.
- Outputs: Per-logger _H.csv, deployment-wide narrative CSV, and wide summary CSV.

Conventions

- Time parsing differs by deployment; set via sh_helper.time_format.
- Calibration can be overridden by .cfg file.
- Analysis windows aligned to complete lunar cycles.
- High-tide lag accounted for in comparisons.

Appendix.
Issues encountered and solutions

FloatArm: Stake movement due to ice altered height of pivot.

In this example from Waquoit Bay, during the winter months freezing in the creek pulled the stake upward, thus increasing the distance from the pivot to ground compared to the original deployment.

As a result, this stake needs to be redeployed and its pivot height resurveyed following spring thaw.

It is recommended to discard data known to be affected by ice (eg, clip dataset to end at timing of the first hard freeze).

FloatArm: Styrofoam float mistaken for prey by alligator.

In the North Inlet estuary, one floatarm showed evidence of alligator bite marks. This sort of disturbance should have little impact on data quality.

FloatArm: Arm Stoppers

Problem: Arm flipping at high tide.

Without the top stopper (nail) the arm could flip to the other side. The instrument reading will not change much unless the arm lands on the ground or gets stuck. The arm stopper prevents the flipping and ensures the correct operation.

Appendix.
Example deployment locations

Delaware NERR St John Reserve


Small Tidal Channel


Mud Hole


Developing Bare Spot

North Inlet Winyah Bay NERR

North Inlet Winyah Bay NERR