NOAA Technical Memorandum NOS CS 62

Coastal Modeling Vertical Datum Package

Silver Spring, Maryland
March 2026



noaa National Oceanic and Atmospheric Administration

U.S. DEPARTMENT OF COMMERCE
National Ocean Service
Coast Survey Development Laboratory

Office of Coast Survey
National Ocean Service
National Oceanic and Atmospheric Administration
U.S. Department of Commerce

The Office of Coast Survey (OCS) is the Nation's only official chartmaker. As the oldest United States scientific organization, dating from 1807, this office has a long history. Today it promotes safe navigation by managing the National Oceanic and Atmospheric Administration's (NOAA) nautical chart and oceanographic data collection and information programs.

There are four components of OCS:

The Coast Survey Development Laboratory develops new and efficient techniques to accomplish Coast Survey missions and to produce new and improved products and services for the maritime community and other coastal users.

The Marine Chart Division acquires marine navigational data to construct and maintain nautical charts, Coast Pilots, and related marine products for the United States.

The Hydrographic Surveys Division directs programs for ship and shore-based hydrographic survey units and conducts general hydrographic survey operations.

The Navigational Services Division is the focal point for Coast Survey customer service activities, concentrating predominately on charting issues, fast-response hydrographic surveys, and Coast Pilot updates.

NOAA Technical Memorandum NOS CS 62

Coastal Modeling Vertical Datum Package

Felicio Cassalho[1, 2], Jack Riley[1], Saeed Moghimi[1], Jack Reeves Eyre[1, 3], Soroosh Mani[1], Greg Seroka[1], Liujuan Tang[1], Shachak Peeri[4], Edward Myers[1], Corey Allen[1]

1. NOAA Office of Coast Survey, Silver Spring, Maryland
2. Ocean Associates Incorporated (OAI), Arlington, Virginia
3. Earth Resources Technology (ERT), Laurel, Maryland
4. NOAA National Geodetic Survey, Silver Spring, Maryland

March 2026

noaa National Oceanic and Atmospheric Administration

U. S. DEPARTMENT OF COMMERCE Howard Lutnick, Secretary	National Oceanic and Atmospheric Administration Dr. Neil Jacobs, Administrator	National Ocean Service Nicole LeBoeuf, Assistant Administrator

Office of Coast Survey RDML Christiaan van Westendorp Director	Coast Survey Development Laboratory Corey Allen Division Chief

NOTICE

Mention of a commercial company or product does not constitute an endorsement by NOAA. Use for publicity or advertising purposes of information from this publication concerning proprietary products or the tests of such products is not authorized.

TABLE OF CONTENTS

LIST OF CODE SNIPPETS iv
LIST OF FIGURES iv
ABSTRACT v
1. INTRODUCTION 1
2. OVERVIEW OF WORKFLOWS 4
Case a 4
Case b 4
3. PACKAGE ARCHITECTURE AND IMPLEMENTATION 7
Core Processing Logic (*vdatum.py*) 7
Geodetic Transformations (*_geoid_tr.py*) 8
Datum Grids and Data Sources (*_path.py*) 8
Utilities (*utils.py*) 9
4. PACKAGE USE AND LIMITATIONS 11
Package Usage 11
Limitations 14
5. PACKAGE VALIDATION 17
6. FUTURE DEVELOPMENT AND COMMUNITY CONTRIBUTIONS 21
7. SUMMARY AND CONCLUSIONS 23
REFERENCES 24

LIST OF CODE SNIPPETS

Code Snippet 1. PROJ pipeline structure. 7
Code Snippet 2. Conda environment setup and package installation. 11
Code Snippet 3. Use of the Coastal Modeling Vertical Datum package “online”. 12
Code Snippet 4. Use of the Coastal Modeling Vertical Datum package “offline”. 12
Code Snippet 5. Updating a mesh’s bathymetry via OCSMesh. 13

LIST OF FIGURES

Figure 1. Transformation workflows supported in Coastal Modeling Vertical Datum package. 4
Figure 2. Extent of NAVD88 on top of the xGEOID20B GeoTIFF (units in meters). 14
Figure 3. Extent of tidal GeoTIFF files, hydroid heights vary from -40 to +35 meters. 15
Figure 4. Location of the 25 validation points along the US East and Gulf Coasts. 17

ABSTRACT

Accurate conversions between vertical datums are a fundamental requirement for operational coastal modeling, storm surge prediction, and inundation mapping. However, integrating standard transformation tools into automated, Python-based modeling workflows is often hindered by software complexity, lack of transparency, and difficulties in handling proprietary “black-box” logic. This report introduces the Coastal Modeling Vertical Datum package, an open-source, lightweight Python library designed to streamline these transformations for the ocean circulation community. Built upon the industry-standard PROJ library and utilizing specific NOAA-derived datum grids, the package implements rigorous, hard-coded transformation pipelines that bridge ellipsoidal, orthometric, and tidal reference frames. This report details the package architecture, the geodetic logic defining the transformation pathways, and the utilization of the tool for mesh bathymetry adjustment. A quantitative validation against NOAA’s authoritative *VDatum* software across 25 control points along the U.S. East and Gulf coasts demonstrates high reliability: the package achieves near-perfect agreement (bias < 1 cm) for tidal grid shifts and maintains high consistency (bias < 1.5 cm) for complex ellipsoidal transformations. The result is a transparent, community-extensible tool that satisfies the accuracy requirements of operational coastal forecasting.

Key Words: Vertical Datum Transformation, Coastal Modeling, Storm Surge, PROJ, Python

1. INTRODUCTION

Accurate vertical referencing is the backbone of reliable coastal modeling. In ocean circulation and storm surge modeling, the representation of the physical environment (bathymetry, topography, and water levels) relies on the vertical reference surface, or vertical datum, used to define the base elevation. These datums generally fall into three categories: Orthometric Datums, Tidal Datums (based on averaged water phases, e.g., MLLW), and Ellipsoidal Datums (based on geometric models of the Earth). Reliable tidal and storm surge predictions from an operational forecast system requires all geometric inputs (e.g., numerical mesh) to be converted to a single datum, and model outputs must often be translated back into different reference frames to be meaningful for navigation or emergency management.

However, storm surge modelers face a persistent challenge: the inputs and outputs of their models rarely align with the desired reference frames of end-users or observational data. For example, a modeler may need to convert results from Mean Lower Low Water (MLLW) to North American Vertical Datum of 1988 (NAVD88) for inundation mapping. These conversions are often cumbersome, relying on “black-box” software or proprietary tools where the internal logic is not immediately transparent. Furthermore, integrating these external executables into automated, Python-based modeling workflows can be difficult, creating friction when modelers need to debug discrepancies or require specific, customized transformation pathways.

To address these challenges, the Coastal Modeling Vertical Datum package was developed. This Python-based tool offers a lightweight, transparent, and open-source alternative for performing vertical datum transformations specifically tailored for the ocean circulation modeling community. This package minimizes software dependencies, relying only on *NumPy* (Harris et al. 2020) and *pyproj* (pyproj 2024). By leveraging the industry-standard *PROJ* library and predefined public datum grids (e.g., from the PROJ.org Datumgrid *CDN* - cdn.proj.org/), the package constructs rigorous, hard-coded transformation pipelines. This approach effectively democratizes the process, allowing users to clearly understand the ellipsoidal and grid-based shifts being applied and to contribute new pipelines, such as those for emerging datums like xgeoid20b, as modeling needs evolve.

This report documents the methodology, implementation, and validation of the Coastal Modeling Vertical Datum package. The following sections will detail the construction of the *PROJ* pipelines, including descriptions of the ellipsoids and datum grids used; provide a technical description of the package architecture; offer a guide on how to utilize the

package for common modeling tasks; and finally, present a quantitative validation of the package's conversions compared against NOAA's standard *VDatum* software.

2. OVERVIEW OF WORKFLOWS

The Coastal Modeling Vertical Datum package supports two different types of workflows for vertical datum conversion. The first (Case a, Figure 1) assumes the input and output datums are referenced to the same ellipsoid, and thus no *PROJ*-based geodetic transformation is needed. The second (Case b, Figure 1) requires a more complex pipeline that encompasses the respective ellipsoid transformations.

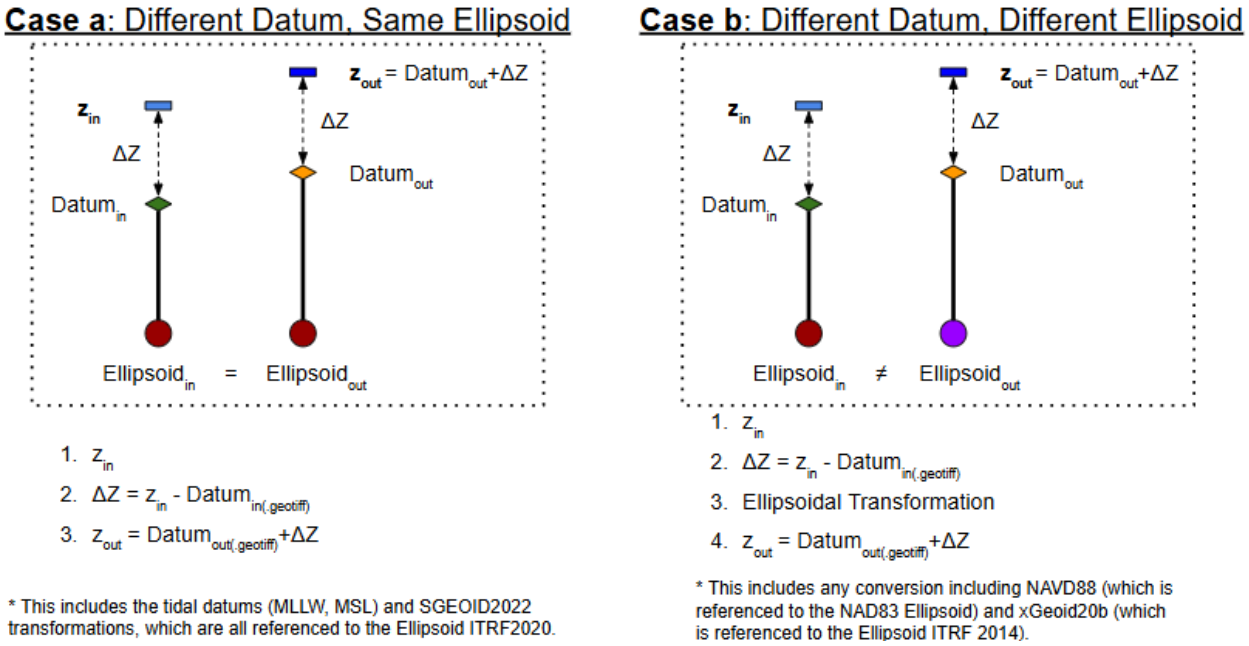


Figure 1. Transformation workflows supported in Coastal Modeling Vertical Datum package.

Case a

In the case where ellipsoid transformation is not needed, the workflow goes as follows:

1. **User Input:** the user provides the elevation (z_{in}) and the respective longitude and latitude for the reference vertical datum ($Datum_{in}$).
2. **Vertical Grid Shift:** the provided elevation (z_{in}) is subtracted by the grid value (*GeoTIFF* file for the respective input vertical datum).
3. **Inverse Vertical Grid Shift:** the ellipsoidal height calculated in step 2 is then applied over the target vertical datum ($Datum_{out}$).

Case b

In the case where ellipsoid transformation is needed, the workflow goes as follows:

1. **User Input:** the user provides the elevation (z_{in}) and the respective longitude and latitude for the reference vertical datum ($Datum_{in}$).

2. **Vertical Grid Shift:** the provided elevation (z_{in}) is subtracted by the grid value (*GeoTIFF* file for the respective input vertical datum).
3. **Ellipsoidal transformation:**
    a. **Conversion to 3D Cartesian:** the longitude and latitude provided by the user, and ellipsoidal height calculated in step 2 are converted to 3D Cartesian coordinates prior to the ellipsoid transformation.
    b. **Helmert Transform:** this *PROJ* step shifts, rotates, and scales the 3D coordinates from the input referencing to the output. It applies a 7-parameter transformation (or 14-parameter if time-dependent) that includes: i) Translation (dx, dy, dz), by moving the center of the Earth model; ii) Rotation (rx, ry, rz), by tilting the Earth axis as needed; iii) Scale (s), by inflating or shrinking the model slightly; and iv) Epoch (t) by accounting for tectonic shifts.
    c. **Conversion to Ellipsoid:** the resulting 3D Cartesian coordinates are converted to the new longitude, latitude, and height.
4. **Inverse Vertical Grid Shift:** the ellipsoidal height from step 5 is then applied over the target vertical datum ($Datum_{out}$).

By strictly defining these two workflow pathways, the Coastal Modeling Vertical Datum package ensures that transformations are applied with the appropriate level of geodetic rigor without imposing unnecessary computational overhead. While “Case a” provides a streamlined approach for co-ellipsoidal shifts, “Case b” handles the complex geometric adjustments required when crossing reference frames, ensuring high-accuracy results even when tectonic velocities and epoch differences are factored in. Crucially, this selection logic is abstracted from the user; the package automatically determines the requisite pathway based on the defined source and target datums. The subsequent section provides a detailed breakdown of the package’s internal architecture, explaining how these conceptual workflows are translated into specific *PROJ* pipeline syntaxes and Python implementation.

3. PACKAGE ARCHITECTURE AND IMPLEMENTATION

The Coastal Modeling Vertical Datum package is designed as a lightweight, modular Python library. Its architecture isolates the transformation logic from the data sources, ensuring that the geodetic operations remain transparent while allowing flexibility in how datum grids are accessed (online vs. offline).

The package is hosted on GitHub (oceanmodeling/coastalmodeling-vdatum) and relies on a minimal set of dependencies: *NumPy* for array manipulation and *pyproj* for interfacing with the *PROJ* engine. The codebase is organized into four primary modules:

1. ***vdatum.py*:** The core processing engine and entry point.
2. ***_geoid_tr.py*:** Definitions of ellipsoidal (Helmert) transformations.
3. ***_path.py*:** Configuration of datum grid file locations (S3 buckets or local paths).
4. ***utils.py*:** Helper functions for file I/O specific to coastal modeling formats.

Core Processing Logic (*vdatum.py*)

The primary functionality is exposed through the convert function within *vdatum.py*. This function acts as a dispatcher that implements the logic described in Section 2. Based on the user's specified *vd_from* (source datum) and *vd_to* (target datum), the function queries an internal mapping system (the inputs function) to determine the necessary components for the transformation pipeline.

Once the components are identified, the *build_pipe* function constructs a standard *PROJ* pipeline string. A characteristic pipeline generated by the package follows this rigorous structure (Code Snippet 1):

```
+proj=pipeline
  +step +proj=axisswap +order=2,1
  +step +proj=unitconvert +xy_in=deg +z_in=m +xy_out=rad +z_out=m
  +step +proj=vgridshift +grids={height_to_geoid_grid} +multiplier=1
  {ellipsoid_transformation_step}
  +step +proj=vgridshift +grids={geoid_to_height_grid} +multiplier=-1
  +step +proj=unitconvert +xy_in=rad +z_in=m +xy_out=deg +z_out=m
  +step +proj=axisswap +order=2,1
```

Code Snippet 1. PROJ pipeline structure.

This structure ensures that unit conversions (degrees to radians) and axis ordering (latitude/longitude) are handled consistently before any vertical shifts are applied. The *ellipsoid_transformation_step* is inserted dynamically only when the source and target datums reside on different ellipsoids.

Geodetic Transformations (*_geoid_tr.py*)

To ensure reproducibility and adherence to specific modeling epochs, ellipsoidal transformations are hard-coded within *_geoid_tr.py* rather than relying on *PROJ*'s default dynamic lookup. This module defines 14-parameter time-dependent Helmert transformations between major reference frames.

Key transformations included in this module are:

1. **ITRF2014 to/from ITRF2020:** Used for connecting modern satellite-derived datasets (i.e., xGeoid20b).
2. **NAD83(2011) epoch 2010.0 to/from ITRF2020 epoch 2020.0:** Critical for linking the National Spatial Reference System to tidal models.

By explicitly defining parameters such as translation (dx, dy, dz), rotation (rx, ry, rz), scale (s), and their respective rates of change, the package guarantees that the geometric relationship between ellipsoids remains constant across different computing environments.

Datum Grids and Data Sources (*_path.py*)

The accuracy of the vertical transformation is driven by the underlying grid files (*GeoTIFF*). The module *_path.py* manages the locations of these grids.

A distinguishing feature of this package is its use of specialized tidal datum grids. Unlike standard public grids, the tidal datum surfaces (e.g., MLLW, MHHW, LMSL) used here were developed internally by NOAA experts. These grids are explicitly referenced to ITRF2020, allowing for direct, high-precision geometric transformations without intermediate orthometric steps.

The package supports two modes of operation via the online flag:

1. **Online Mode (Default):** Grids are streamed directly from a public NOAA S3 bucket (noaa-nos-stofs2d-pds). This ensures the user is always applying the most up-to-date surfaces without manual file management.
2. **Offline Mode:** For High-Performance Computing (HPC) environments where compute nodes may lack internet access, the package allows users to override the S3 paths and point to a local directory containing the *GeoTIFFs*.

Utilities (*utils.py*)

To facilitate integration with specific ocean modeling software, *utils.py* provides specialized I/O functions. This includes parsers for the *station.in* format (targeting SCHISM modeling system), allowing modelers to read station metadata (index, longitude, latitude, elevation) directly into Python lists for immediate processing.

4. PACKAGE USE AND LIMITATIONS

While the architecture described above provides a robust framework for calculation, the physical validity of any transformation is strictly limited by the spatial extent of the datum grids defined in *_path.py*. Because the package relies on grid-based lookups for tidal and orthometric adjustments, conversions are only possible where these grids possess valid data. The following section details the usage of the package, demonstrates the spatial domains of these supported grids, and provides examples of handling data both within and outside these valid bounds.

Package Usage

The package is hosted on GitHub and is best managed within a dedicated Conda environment to ensure dependency isolation. It requires Python 3.11 or later. To install the package and its dependencies (*NumPy*, *pyproj*), execute the following commands in a terminal:

```
# Create and activate a new conda environment
conda create --name cmvd python=3.11
conda activate cmvd

# Install the package directly from the source repository
pip install "git+https://github.com/oceanmodeling/coastalmodeling-vdatum.git"
```

Code Snippet 2. Conda environment setup and package installation.

The core functionality is exposed via the *vdatum.convert* function. This function accepts scalars, lists, or *NumPy* arrays for coordinates, making it flexible for point-checks or bulk data processing. The following example demonstrates converting a water level elevation from Mean Lower Low Water (MLLW) to North American Vertical Datum of 1988 (NAVD88):

```python
from coastalmodeling_vdatum import vdatum

# Define input coordinates (Latitude, Longitude, Elevation)
lat = [28.8, 29.1]
lon = [-80.0, -80.5]
z_mllw = [1.5, 1.2]  # Elevation in meters relative to MLLW

# Perform conversion
# online=True fetches grids from NOAA S3 buckets automatically
lat_out, lon_out, z_navd88 = vdatum.convert(
    vd_from='mllw',
    vd_to='navd88',
    lat=lat,
    lon=lon,
    z=z_mllw,
    online=True,
    epoch=None
)

print(f"Elevation in NAVD88: {z_navd88}")
```

Code Snippet 3. Use of the Coastal Modeling Vertical Datum package "online".

For operational forecasting or HPC environments where compute nodes do not have internet access, the package supports an "offline" mode. This allows users to manually point the package to local copies of the required *GeoTIFF* datum grids. To use this feature, the user must override the file paths in the *_path* module before *calling convert*:

```python
from coastalmodeling_vdatum import vdatum, _path

# Define local paths to datum grids (downloaded previously)
LOCAL_DIR = "/path/to/datum/grids"
_path.NAVD88_G2018 = f"{LOCAL_DIR}/us_noaa_g2018u0.tif"
_path.XGEOID20B = f"{LOCAL_DIR}/xGEOID20B.tif"
_path.MLLW_ITRF2020_2020 = f"{LOCAL_DIR}/us_noaa_nos_MLLW-ITRF2020_2020.

# Perform conversion with online=False
x, y, z_out = vdatum.convert(
    vd_from='navd88',
    vd_to='mllw',
    lat=lat,
    lon=lon,
    z=z_in,
    online=False
)
```

Code Snippet 4. Use of the Coastal Modeling Vertical Datum package "offline".

A primary use case for this package is correcting the bathymetry of unstructured meshes used by models like SCHISM. The package integrates seamlessly with mesh-handling libraries such as *OCSMesh* (github.com/noaa-ocs-modeling/OCSMesh) and *pylib* (github.com/pyproj4/pyproj). The package expects standard elevation convention (positive up). Since many hydrodynamic models represent depth as positive values (positive down), users must invert the sign of the vertical coordinate before and after the transformation:

```python
import numpy as np
import ocsmesh
from coastalmodeling_vdatum import vdatum

# Load mesh and extract coordinates (invert depth to elevation)
gd = ocsmesh.Mesh.open("hgrid.gr3", crs=4326)
lat = gd.vert2['coord'][:, -1]
lon = gd.vert2['coord'][:, 0]
z_in = gd.value * -1

# Perform conversion (e.g., LMSL to NAVD88)
_, _, z_out = vdatum.convert(
    vd_from='lmsl',
    vd_to='navd88',
    lat=lat,
    lon=lon,
    z=z_in
)

# Handle outliers (inf) where conversion failed
z_out[np.isinf(z_out)] = z_in[np.isinf(z_out)]

# Reconstruct and save mesh
mesh_msht = ocsmesh.utils.msht_from_numpy(
    coordinates=gd.vert2['coord'],
    triangles=gd.tria3['index'],
    quadrilaterals=(
        gd.quad4['index'] if len(gd.quad4['index']) > 0 else None
    ),
    crs=4326
)

mesh_msht.value = np.array(z_out)
ocsmesh.Mesh(mesh_msht).write(
    "mesh_navd88.2dm",
    format='2dm',
    overwrite=True
)
```

Code Snippet 5. Updating a mesh's bathymetry via OCSMesh.

These workflows illustrate how the Coastal Modeling Vertical Datum package simplifies the integration of vertical datum transformations into operational modeling pipelines. Whether processing isolated station data or adjusting bathymetry for large-scale unstructured meshes, the package abstracts the complexity of *PROJ* pipeline construction, ensuring consistent and reproducible results. However, it is important to note that the validity of these transformations is strictly bounded by the spatial extent of the underlying datum grids. The subsequent section examines these geographical limits, detailing the coverage of the supported tidal and orthometric surfaces and the package's behavior when encountering data outside these defined domains.

Limitations

Since the package relies on the *PROJ vgridshift* operator, transformations are essentially lookup operations. If a coordinate pair (latitude, longitude) falls outside the defined spatial extent of the underlying grid files (*GeoTIFFs*), the transformation cannot be resolved.

The package distinguishes between two broad categories of grids, each with distinct spatial characteristics:

1. **Gravimetric and Hybrid Geoid Grids (e.g., xGEOID20b, NAVD88):** These grids typically possess broad spatial coverage, extending across the entire continental United States (CONUS), Alaska, Hawaii, and often well into the offshore Exclusive Economic Zone. Figure 2 show the extent of these Raster files:

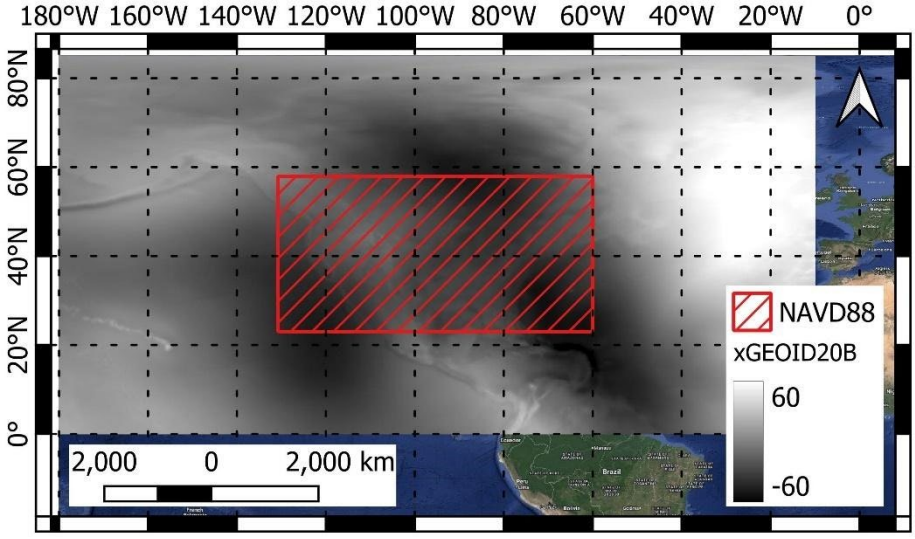


Figure 2. Extent of NAVD88 on top of the xGEOID20B GeoTIFF (units in meters).

2. **Tidal Datum Grids (e.g., MLLW, MHHW, LMSL):** The tidal datum grids used by this package are derived from hydrodynamic modeling and observational data. Consequently, their valid spatial domain is restricted to the coastal shelf and nearshore waters. They do not extend into the deep ocean nor do they cover

inland topography beyond the extent of tidal influence. Figure 3 show the extent of these Raster files:

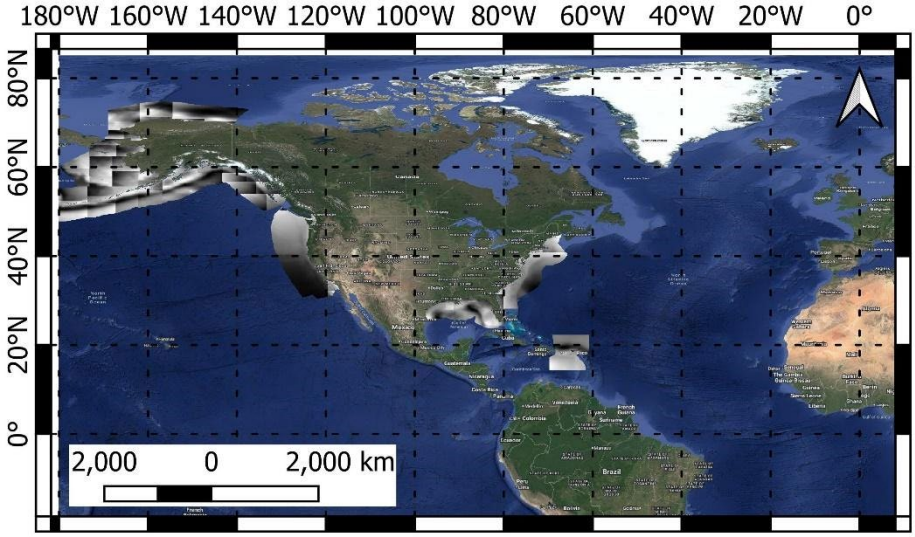


Figure 3. Extent of tidal GeoTIFF files, hydroid heights vary from -40 to +35 meters.

When the package encounters a coordinate that lies outside the valid domain of the required grid, it does not attempt to extrapolate or apply a fallback value. Instead, it returns an infinite value (*inf*). This behavior is intentional. It forces the modeler to make an explicit decision regarding boundary conditions rather than silently propagating potentially erroneous extrapolated values.

5. PACKAGE VALIDATION

To assess the accuracy and consistency of the Coastal Modeling Vertical Datum package, a comparative analysis was performed against NOAA's authoritative *VDatum* software. It is important to emphasize that the primary objective of this validation is to verify the package's internal mechanics. Specifically, ensuring that the underlying PROJ pipelines, geodetic logic, and mathematical grid shifts are constructed and executed correctly. Consequently, this section focuses on software-to-software agreement to confirm technical fidelity, rather than absolute validation against real-world observational data or specific local bathymetric surveys.

The validation dataset consisted of 25 control points randomly distributed along the U.S. East Coast and Gulf Coast, spanning from Maine to Texas. These locations were selected to cover a wide range of latitudes and coastal geometries, ensuring that the package was tested against various regional grid definitions.

Two distinct transformation scenarios were evaluated for each control point to test different internal logic pathways:

1. **Orthometric to Geopotential (NAVD88 to xGEOID20b):** This transformation represents a "complex" pathway (Case b), requiring both a vertical grid shift and a 3D ellipsoidal transformation (Helmert transformation) to bridge the reference frames.
2. **Tidal Datum Conversion (LMSL to MLLW):** This transformation represents a "co-ellipsoidal" pathway (Case a), requiring only direct vertical grid arithmetic without geometric reference frame changes.

The geographical distribution of the validation points is shown in Figure 4.

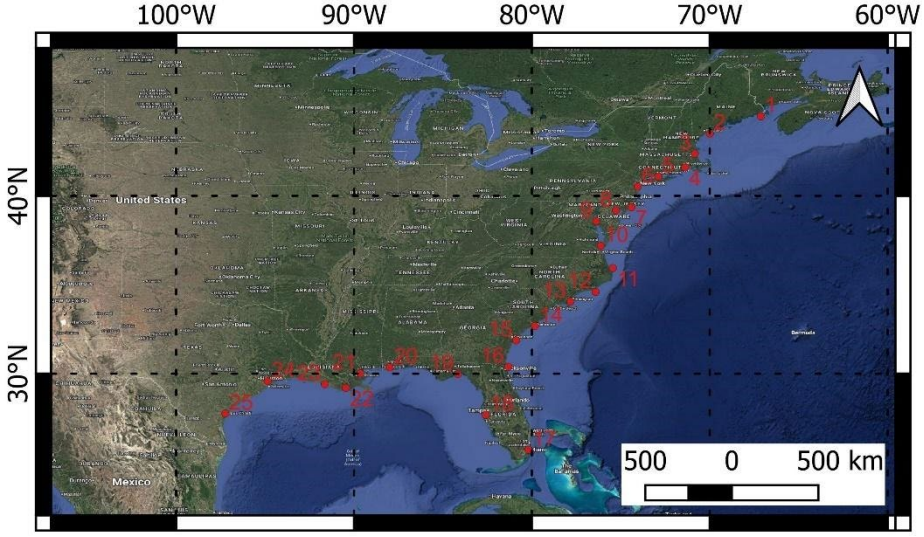


Figure 4. Location of the 25 validation points along the US East and Gulf Coasts.

The results of the validation are summarized in Table 1. The table provides the computed vertical separation (datum shift) for both software tools and the resulting difference (residual). In both cases input height was set to zero.

Table 1. Comparison of vertical datum transformations between VDatum and the Coastal Modeling Vertical Datum (CMVD) package. All units are in meters.

ID	Lon	Lat	NAVD88 to xGEOID20b	NAVD88 to xGEOID20b	LMSL to MLLW	LMSL to MLLW	Bias	Bias
(int)	(Deg)	(Deg)	VDatum	CMVD	VDatum	CMVD	Case b	Case a
1	-67.1178	44.47637	-0.332	-0.318	2.111	2.112	-0.014	-0.001
2	-69.9898	43.50265	-0.325	-0.311	1.448	1.448	-0.014	0.000
3	-70.8357	42.36911	-0.326	-0.312	1.489	1.489	-0.014	0.000
4	-71.3754	41.60562	-0.353	-0.339	0.581	0.581	-0.014	0.000
5	-72.8876	41.0868	-0.34	-0.326	0.984	0.984	-0.014	0.000
6	-74.0482	40.52617	-0.371	-0.357	0.79	0.790	-0.014	0.000
7	-74.3992	39.36066	-0.353	-0.339	0.667	0.666	-0.014	0.001
8	-75.269	39.17993	-0.377	-0.363	0.906	0.905	-0.014	0.001
9	-76.3755	38.57996	-0.372	-0.358	0.23	0.230	-0.014	0.000
10	-76.1247	37.18823	-0.321	-0.308	0.365	0.364	-0.013	0.001
11	-75.4412	35.94403	-0.294	-0.281	0.509	0.510	-0.013	-0.001
12	-76.4149	34.60148	-0.227	-0.215	0.538	0.538	-0.012	0.000
13	-77.8275	34.05438	-0.202	-0.201	0.668	0.668	-0.001	0.000
14	-79.8241	32.68846	-0.29	-0.277	0.818	0.818	-0.013	0.000
15	-80.8814	31.87703	-0.24	-0.227	1.06	1.060	-0.013	0.000
16	-81.3167	30.39186	-0.157	-0.146	0.832	0.832	-0.011	0.000
17	-80.189	25.72305	0.045	0.055	0.32	0.320	-0.010	0.000
18	-82.5891	27.68647	-0.02	-0.009	0.353	0.353	-0.011	0.000
19	-84.1665	29.9843	-0.123	-0.111	0.536	0.536	-0.012	0.000
20	-87.9926	30.36297	-0.153	-0.141	0.215	0.206	-0.012	0.009
21	-89.6462	30.02241	-0.086	-0.074	0.206	0.206	-0.012	0.000
22	-90.4551	29.20729	0.007	0.019	0.201	0.201	-0.012	0.000
23	-91.6354	29.41384	0.043	0.056	0.305	0.305	-0.013	0.000
24	-94.8516	29.58104	-0.171	-0.158	0.205	0.204	-0.013	0.001
25	-97.2294	27.76515	-0.178	-0.165	0.107	0.108	-0.013	-0.001

The comparison confirms that the package provides a highly consistent alternative to *VDatum* for coastal modeling applications.

For purely grid-based tidal transformations (LMSL to MLLW), the package achieves near-perfect agreement. The residuals are negligible (typically 0.000 m) with a maximum deviation of only 0.9 cm (Point 20). This confirms that the package is correctly reading and interpolating the underlying tidal datum grids, offering bit-wise consistency with the authoritative standard for co-ellipsoidal shifts. For complex transformations involving ellipsoidal changes (NAVD88 to xGEOID20b), the results show a small, systematic difference ranging between 1.0 cm and 1.4cm. This consistency is expected and

acceptable. The deviation possibly arises from the specifics of the 3D Helmert transformation implementation. Specifically, how the package hard-codes the rigorous ITRF2020 epoch parameters via *PROJ* pipelines compared to *VDatum*'s internal frame realization. Crucially, this difference is uniform across the entire spatial domain (from Maine to Texas) and remains consistently below 1.5 cm. Given that uncertainties in coastal model boundary conditions and bathymetric surveys often exceed these magnitudes, the Coastal Modeling Vertical Datum package demonstrates sufficient accuracy for operational ocean circulation and storm surge modeling, providing a transparent and automatable solution with errors well within acceptable tolerance limits.

6. FUTURE DEVELOPMENT AND COMMUNITY CONTRIBUTIONS

The Coastal Modeling Vertical Datum package is an open-source project designed to evolve with the needs of the ocean modeling community. While the current version supports a robust set of transformations for the U.S. East and Gulf Coasts, we encourage users to contribute new pipelines for different regions or emerging vertical datums.

While the initial validation confirms the package's technical fidelity against VDatum, we recognize the need for broader geographic and observational testing. Future development efforts, include:

1. **Observational Validation:** Comparing package outputs directly against physical observation records. For example, verifying "Case a" co-ellipsoidal shifts against datum conversions based on tidal epochs from NOAA CO-OPS stations to demonstrate real-world accuracy relative to physical observations.
2. **Geographic Expansion:** Extending validation to regions that present significantly more complex vertical datum transformation challenges, particularly the U.S. West Coast and Alaska .
3. **Software Enhancements:** To improve the tool's accessibility, reliability, and ease of use, planned updates include publishing the package to the Python Package Index (PyPI), developing Command Line Interface (CLI) entry points for use in bash-scripted workflows, and implementing a comprehensive suite of automated unit tests.

Adding a new transformation is a straightforward process that leverages standard open-source tools. We recommend the following workflow for contributors wishing to extend the package:

4. **Identify Coordinate Reference Systems (CRS):** Use the EPSG Registry (epsg.org/home) to identify the specific codes for your source and target datums (e.g., EPSG:6318 for NAVD88). This ensures you are referencing the correct geodetic definitions.
5. **Construct the Pipeline:** The command-line tool *projinfo* is the primary utility for generating valid *PROJ* strings. By running *projinfo -s <source_crs> -t <target_crs>*, you can retrieve a list of available operations. This command outputs the exact *PROJ* pipeline syntax needed to mathematically bridge the two datums.
6. **Select the Correct Grid:** For transformations requiring grid shifts (e.g., geoid modulations), verify the availability and spatial extent of the necessary grid files via the *PROJ CDN* (cdn.proj.org/). This ensures that the transformation is valid for your specific modeling domain.

7. **Implement in Python:** Once the pipeline steps are identified, the workflow is simply a matter of concatenating the *PROJ* strings and adding them to the package's *vdatum.py* module.

By following this transparent approach, the modeling community can collectively build a comprehensive, reliable, and transparent library of vertical datum transformations.

7. SUMMARY AND CONCLUSIONS

The development of the Coastal Modeling Vertical Datum package addresses a critical need within the ocean modeling community for a transparent, flexible, and automatable solution to vertical referencing challenges. By abstracting the complexities of geodetic transformations into a lightweight Python framework, the package allows modelers to seamlessly integrate accurate datum shifts into mesh generation and post-processing workflows without relying on opaque, external executables.

The validation results presented in this report confirm that the package delivers accuracy comparable to authoritative standards. With deviations from NOAA's *VDatum* software remaining consistently below 1.5 cm, which is well within the uncertainty margins of typical bathymetric surveys and hydrodynamic boundary. The clear distinction between co-ellipsoidal grid shifts ("Case a") and complex geometric transformations ("Case b") ensures that the appropriate level of geodetic rigor is applied to every conversion.

However, users must remain cognizant of the physical limitations imposed by the underlying datum grids. As a lookup-based tool, the package is strictly bound by the spatial extent of available observation and model data; it does not extrapolate beyond the valid domains of the provided *GeoTIFFs*.

Ultimately, the open-source nature of this package represents a shift toward community-driven operational tools. By providing a clear architecture for defining *PROJ* pipelines and managing grid sources, the package empowers users not only to verify the logic behind their data manipulations but also to actively contribute to the tool's evolution as new datums and vertical reference frames are adopted by the scientific community.

By following this transparent approach, the modeling community can collectively build a comprehensive, reliable, and transparent library of vertical datum transformations.

REFERENCES

Coastal Modeling Vertical Datum, 2024. GitHub repository, URL: https://github.com/oceanmodeling/coastalmodeling-vdatum

Harris, C.R., Millman, K.J., van der Walt, S.J. et al., 2020. Array programming with NumPy. Nature, 585, pp.357–362.

International Association of Oil & Gas Producers (IOGP), 2024. EPSG Geodetic Parameter Dataset, URL: https://epsg.org/home

National Oceanic and Atmospheric Administration (NOAA), 2024. VDatum: Vertical Datum Transformation. National Ocean Service, URL: https://vdatum.noaa.gov/

OCSMesh, 2024. URL: https://github.com/noaa-ocs-modeling/OCSMesh

PROJ contributors, 2024. PROJ coordinate transformation software library. Open Source Geospatial Foundation, URL: https://proj.org/

PROJ.org, 2024. PROJ Datumgrid CDN, URL: https://cdn.proj.org/

pylib, 2024. URL: https://github.com/wzhengui/pylibs

pyproj contributors, 2024. pyproj: Python interface to PROJ, URL: https://github.com/pyproj4/pyproj

Zhang, Y., Ye, F., Stanev, E.V. and Grashorn, S., 2016. Seamless cross-scale modeling with SCHISM. Ocean Modelling, 102, pp.64-81.