



PyESPERv1.0.0: a Python implementation of empirical seawater property estimation routines (ESPERs)

Larissa M. Dias^{1,2} and Brendan R. Carter^{1,2}

¹Cooperative Institute for Climate, Ocean, and Ecosystem Studies, University of Washington, Seattle, 98105, USA

²NOAA Pacific Marine Environmental Laboratory, Seattle, 98115, USA

Correspondence: Larissa M. Dias (lmdias@uw.edu)

Received: 31 January 2025 – Discussion started: 11 April 2025

Revised: 12 July 2025 – Accepted: 15 July 2025 – Published: 15 October 2025

Abstract. This project produced a Python language implementation of locally interpolated regression (LIR) and neural network (NN) algorithms from empirical seawater property estimation routines (PyESPERv1.0.0). These routines estimate total alkalinity, dissolved inorganic carbon, total pH, nitrate, phosphate, silicate, and oxygen from geographic coordinates, depth, salinity, and 16 combinations of zero to four additional predictors (temperature and biogeochemical information) and were previously available only in the MATLAB programming language. Here, we document modifications to reduce discrepancies between the implementations, calculate the disagreements between the methods, and quantify Global Ocean Data Analysis Project (GLODAPv2.2022) reconstruction errors with PyESPER. While the PyESPER routine based on neural networks (PyESPER_NN) faithfully reproduces the corresponding MATLAB routine estimates of properties that do not require anthropogenic carbon change information, PyESPER_LIR and – to a lesser extent – PyESPER_NN estimates for total pH and dissolved inorganic carbon do not exactly reproduce the MATLAB routine estimates due to differences in interpolation and extrapolation methods between the programming languages. While the MATLAB and Python LIR-based estimates are not identical, we show that they are similarly skilled at reproducing the GLODAPv2.2022 data product and are thus comparable. This project increases the accessibility of ESPERv1.01.01 algorithms by providing users with code in the freely available Python language and enables future ESPER updates to be released in multiple coding languages.

1 Introduction

Ship-based biogeochemical data, as compiled within the Global Ocean Data Analysis Project (GLODAP; Lauvset et al., 2022), have high precision and accuracy but are seasonally biased and spatially sparse (Hauck et al., 2023). International efforts to deploy biogeochemical (BGC) profiling floats with broad spatial coverage and high temporal resolution (10 d) are ongoing (Bittig et al., 2019), with potential to greatly augment available ocean carbon cycle and biogeochemical data. These data can then support a wide variety of research topics and management applications (e.g., warming, acidification, eutrophication, deoxygenation, fisheries, and ecosystem studies). This strategy leverages the high precision and accuracy of ship-based measurements to calibrate and validate the BGC float sensors periodically throughout a float deployment. To do this, machine learning and regression algorithms – which take advantage of the strong regional correlations between seawater properties in the open ocean and, especially, the ocean interior (Bittig et al., 2018; Carter et al., 2018, 2021) – are used to map the ship-based information onto “reference depth” portions of the float profiles.

The empirical seawater property estimation routines (ESPERv1.01.01, henceforth referred to as ESPERs), originally written in the MATLAB programming language, aim to help realize the full potential of BGC float data by using machine learning techniques and regression strategies to predict total alkalinity (TA), dissolved inorganic carbon (DIC), pH on the total scale (pH_T), phosphate, nitrate, silicate, and oxygen from commonly measured physical and BGC parameters (Carter et al., 2021). The algorithms are used to calibrate float profiles (Maurer et al., 2021). In addition, because two car-

bonate system property measurements are necessary to fully quantify the carbonate system in seawater (Zeebe and Wolf-Gladrow, 2001) and BGC floats only have the capability to measure pH_T , these algorithms have the potential to provide (calculated) TA or DIC as a secondary constraint for the marine carbonate system. This method also offers an alternative to using models to estimate variables for carbonate chemistry calculations when nutrient information is unavailable, which potentially has high error values. ESPERs have also been used to map ship-based information across spatial and temporal scales for other applications, including estimation of TA for adjustments of the pH and fugacity of CO_2 ($f\text{CO}_2$) to in situ conditions for data products (Jiang et al., 2021) and estimation of the TA and seawater properties necessary for estimating ocean acidification indicators (Jiang et al., 2020; Sharp et al., 2024). Recent research has also shown that similar machine learning estimation algorithms have potential for the development of four-dimensional data products such as the Gridded Ocean Biogeochemistry from Artificial Intelligence–Oxygen (GOBAI- O_2 ; Sharp et al., 2023) and the Mapped Observation-Based Oceanic DIC (MOBO-DIC; Keppler et al., 2020).

1.1 Importance

Tanhua et al. (2021) and others have argued that researchers should utilize workflows that produce findable, accessible, interoperable, and reusable (FAIR) data products. ESPERs are publicly available (findable) on Zenodo, with updates published to GitHub, are free (accessible), and provide the option for users to cite a digital object identifier (DOI) for each version (reusable). However, until now, ESPERs were available only in the proprietary MATLAB programming language, which posed a barrier to accessibility and interoperability that we aim to address. Future updates may include even more accessible features such as a user interface.

1.2 Goals

This project aimed to create a freely available Python implementation of ESPERs (PyESPERv1.0.0, henceforth referred to as PyESPERs; Carter et al., 2021; Dias and Carter, 2025) that is equivalent to the MATLAB version within ± 2 times the estimate uncertainties (σ) for all estimated biogeochemical properties (TA, DIC, pH_T , nitrate, phosphate, silicate, and oxygen). The PyESPER code is freely available at Zenodo, and updates will be made available at the GitHub repository (see the “Code availability” section).

2 Methods

ESPER algorithms were translated into Python coding language, while the associated files were either translated into Python or read by Python as MATLAB files. Some original

methods were required to allow interpolations to be similar in Python to those of MATLAB ESPERs.

2.1 ESPERs

ESPERs allow estimation of biogeochemical seawater properties using coordinates, depth, salinity, and other optional inputs from a single function call. While sharing a similar set of equations and required input data, ESPERs have two variants that use either locally interpolated regressions (ESPER_LIR) or neural networks (ESPER_NN), along with a mixed estimate (ESPER_Mixed) that is the mean of estimates from the two functions (Carter et al., 2021). There are a couple of reasons to maintain the separate ESPER LIR, NN, or Mixed options, from an end-user perspective, and these reasons are also true for PyESPERs.

1. ESPER_LIRs predate the ESPER_NNs and have been used as a standalone data product for various research purposes (see Carter et al., 2016, 2018). Long-term users of these LIRs have previously expressed a desire for consistency between versions (e.g., when depth was taken out as a predictor for pH_T), and some of them already use CANYON-B (Bittig et al., 2019) as a neural network option for comparison. Therefore, these users who desire consistency would most likely prefer to use ESPER_LIR.
2. ESPER_LIRs are more transparent than ESPER_NNs, as it is simple to parse apart coefficients at the gridded locations and to see how the equations are a result of these. ESPER_LIRs also rely on a grid, which may appeal to some users.
3. ESPER_NNs provide improved estimates on average compared with ESPER_LIRs and behave more like a mapping product in that 3D coordinates are predictors, which may alternately appeal to some users.
4. Although the ESPER_Mixed estimates perform better on average than LIRs or NNs do independently, there are cases where they have greater bias and root mean square error (RMSE) values than LIRs or NNs (e.g., when using Eqs. 1–3 for phosphate or nitrate at all depths; Carter et al., 2021). Users may want to assess each scenario independently and choose which method is most appropriate according to their needs.
5. The NNs are more closely reproduced between the MATLAB and Python ESPER implementations.

2.1.1 Locally interpolated regressions

The most recent versions of ESPER_LIRs (version 1.01.01; version 3 of LIRs) use a standard set of equations of the format shown in Eq. (1) to estimate up to seven different biogeochemical water properties using up to 16 equations with

different combinations of input parameters (see Appendix A, Tables A1 and A2; Carter et al., 2021):

$$X = C_0 + \sum_{i=1}^n C_i P_i, \quad (1)$$

where X is the estimated property (TA, DIC, pH_T , nitrate, phosphate, silicate, or oxygen), C_0 is the intercept, and C_i is the coefficient for each of the n predictors P_i . The intercepts (C_0) and coefficients (C_i) vary with location (latitude, longitude, and depth) and are different for each of the predictor variables (P_i ; Tables A1 and A2; Carter et al., 2021). The most recent ESPERs were trained and assessed on the GLODAPv2.2020 (Olsen et al., 2020) data product, which includes data from 946 cruises and spans 1972–2019, and additional datasets from the Mediterranean Sea and Gulf of Mexico (Carter et al., 2021, Supplementary Information) taken from the Coastal Ocean Data Analysis Project (CODAP, Jiang et al., 2021) and the CARIMED data product (Álvarez et al., 2019).

When the ESPER_LIR function is called, the routines interpolate a pre-determined grid of C 's (intercepts and coefficients) to user-defined locations. Linear interpolation is used within the grid and for extrapolation, and this method utilizes an underlying Delaunay triangulation with MATLAB's scatteredInterpolant function (Carter et al., 2021). The three-dimensional interpolation and extrapolation algorithm is implemented differently in MATLAB and Python, and although both calculations are valid, this difference in implementation is the source of disagreements we find and later quantify between ESPER and PyESPER.

ESPER_LIR coefficients have been determined on a grid using a moving window regression strategy similar to the approach first outlined by Velo et al. (2013), resulting in a set of intercept and coefficient estimates for each of 16 equations for seven possible properties at 44 957 total locations on a 5° latitude (-84.5 – 85.5° N) $\times$ 5° longitude (-19.5 – 375.5° E) $\times$ 33 depth (0–5500 m) ocean interior grid subsampled from the World Ocean Atlas gridded product (Carter et al., 2016, 2018, 2021). These coefficients were fit using regressions relating the property of interest (X) to different combinations of up to five predictor properties (P , Tables A1 and A2), relating to each possible equation as in Eq. (1). Depth (scaled to $1/25$) is included as a coordinate for coefficient interpolation, but depth is not used as a predictor for the current ESPER version (it was included in an earlier version, but only when predicting pH_T ; Carter et al., 2018). Data for each regression fit are selected from “windows” of data that are within 15° latitude, within $30^\circ/\cos(\text{latitude})$ in longitude, and within either $(100 + z/10)$ m depth or 0.1 kg m^{-3} of the estimated density of seawater at that coordinate location, where z is depth in meters (Carter et al., 2021). If either the depth-based or the density-based criterion applies, data are selected for that location, which allows water masses to impact window selection along with depth. If fewer than

100 measurements fall within a window, the dimensions are doubled. In LIRv2, windows are iteratively scaled by a factor of the iteration number until at least 100 measurements are selected to train each regression (Carter et al., 2018). For ESPER_LIRs (LIRv3), it is argued that increasing window size has the following benefits: (1) includes more data for regression fits, (2) introduces more modes of oceanographic variability into fitting data, and (3) reduces multicollinearity (Carter et al., 2021). However, the risk of increasing the window size is that the ESPER_LIRs will be less appropriate locally. A weighting term is applied to help account for this by reducing the cost of regression misfits to data that are distant or at significantly different depths relative to the location, with a cap to prevent overfitting to nearby coordinates (see Carter et al., 2021). Regression coefficients (C_0 and C_i) are then fit using Eq. (2), with separate regressions for the Northern Hemisphere Atlantic, Mediterranean, and Arctic, as well as other global locations, to prevent interpolation across Central America or the Bering Strait.

$$XW = \left(C_0 + \sum_{i=1}^n C_i P_i \right) W \quad (2)$$

PyESPER_LIR does not duplicate this portion of the effort but instead builds directly upon the grid of coefficients obtained for and utilized by the MATLAB implementation of ESPER_LIR.

When the function is called, ESPER_LIR uses MATLAB's scatteredInterpolant (linear interpolation and extrapolations) function to interpolate this previously created grid of regression coefficients to the user-provided set of coordinates, resulting in coefficient estimates at the desired locations (Carter et al., 2021). This method uses a Delaunay triangulation of the scattered sample points to perform interpolations and extrapolations. Different valid mathematics can be used to obtain these Delaunay triangulations and to extrapolate and interpolate, but efforts to identify a Python method for these tasks that exactly replicates the MATLAB results have been unsuccessful. The most similar and least computationally intensive results relative to those of MATLAB's scatteredInterpolant were produced by combining Python's SciPy package functions LinearNDInterpolator (interpolate subpackage) and Delaunay (spatial subpackage; Virtanen et al., 2020). However, because LinearNDInterpolator does not extrapolate and other Python functions do not produce similar results to those of MATLAB when using similar methods, the gridded set of three-dimensional coordinates (44 957 locations based on the World Ocean Atlas) and corresponding coefficient estimates provided by ESPER_LIRs were expanded in MATLAB to 106 400 locations on a grid with estimates every 5° latitude (-94.5 – 90.5° N) and longitude (-19.5 – 375.5° E) and up to 9000 m in depth and applied to scatteredInterpolant within ESPER_LIR to provide coefficient estimates for the external locations through extrapolation. This grid, with equivalent coefficients within the

original parts of the grid and extrapolations outside of the grid, was read in Python when the LIRs were called. The expanded grid allowed Python functions to avoid extrapolations and rely solely on interpolation and triangulation methods when estimating coefficients at user-defined locations. While some of these locations are unphysical (e.g., $\pm > 90^\circ$ N or on land), the coefficients nevertheless provide valid extrapolations from MATLAB for the full possible domain that can then be interpolated in PyESPER_LIR. PyESPER_LIR otherwise replicated ESPER_LIR's separation of data from the Atlantic Ocean, Mediterranean Sea, and Arctic Ocean and data from the Indo-Pacific and Southern Ocean regions.

During the creation of this expanded grid, a grouping error was observed in current versions of MATLAB ESPER_LIRs. Specifically, the mirrored portion of the grid found at < 0 and $> 360^\circ$ E and north of 40° S was not correctly flagged as belonging to the Atlantic grid. The practical effect of this bug was that estimates near the Prime Meridian and near the cutoff between the Southern Ocean and the Atlantic Ocean had extrapolated coefficients instead of interpolated coefficients. This bug was fixed for both MATLAB ESPER_LIR and PyESPER_LIR comparisons for this paper, and a fixed grouping routine is now provided at the original MATLAB ESPER repository with corresponding documentation and will be included in future updates to ESPER_LIRs.

2.1.2 Neural networks

ESPER_NNs use feed-forward neural networks with latitude, depth, $\cos(\text{longitude} - 20^\circ \text{E})$, $\cos(\text{longitude} - 110^\circ \text{E})$, and the parameters from Table A2 as predictors. Four neural networks are used in each of the two ocean regions, which are the same as those used for LIRs (Atlantic–Mediterranean–Arctic and Indo–Pacific–Southern), resulting in 896 total neural networks (eight for each of 16 combinations of predictors for seven property estimates; Carter et al., 2021). An ensemble of four previously created neural networks with different combinations of neurons and hidden layers, including a single one-hidden-layer network with 40 neurons and three two-hidden-layer networks with 30/10, 25/15, and 20/20 neurons in the first/second hidden layers, is used to minimize the impact of errors from any one neural network (Carter et al., 2021).

In ESPER_NN, the neural networks are encoded as functions to avoid requiring access to the machine learning toolbox within MATLAB. Here, we further translate these functions to Python. The resultant Python functions replicate the functions in ESPER_NN to within machine precision. ESPER_NNs linearly interpolate between the two regions of neural networks by latitude across the Southern Atlantic Ocean and Bering Sea and between the North Pacific and Arctic oceans. Zonal transitions in the Southern Atlantic and Indo–Pacific–Southern Ocean network are also implemented. This interpolation uses custom-written 1D or 2D in-

terpolations that are handled identically in both programming environments.

2.1.3 Mixed estimates

The mixed estimate for each input location is the mean of the LIR and NN estimates and therefore is trivially reproduced by a simple single function call and module within Python.

2.1.4 Anthropogenic carbon

The impacts of anthropogenic carbon (C_{ant}) are approximated in ESPER and PyESPER using a $1^\circ \times 1^\circ$ gridded transit time distribution (Vaugh et al., 2006) C_{ant} product referenced to the year 2002 (Lauvset et al., 2016). ESPERs assume that oceanic C_{ant} increases proportionally to atmospheric anthropogenic CO_2 (transient steady-state assumptions; Gammon et al., 1982; Gruber et al., 2019; Tanhua et al., 2007). This implies that the “shape” of the C_{ant} vertical profile (gradient) remains constant with continuous exponential increases in atmospheric CO_2 and ocean C_{ant} according to Eq. (3) (Carter et al., 2021).

$$C_{\text{ant_year_location}} = C_{\text{ant_year_location}} e^{0.018989(\text{year}-2002)} \quad (3)$$

The coefficient in Eq. (3) is derived from Gruber et al.'s (2019) assumption of a 28 % increase in C_{ant} from 1994 to 2007 and enables estimating C_{ant} for a location in a desired year when C_{ant} is known for that same location in a reference year (2002; Carter et al., 2021). This approach does not allow for non-steady-state variations, which is accounted for in overall uncertainty estimates, and is noted as a significant source of uncertainty for projections beyond ~ 2030 .

ESPERs were trained on data for pH_T and DIC, which were converted to the equivalents for the year 2002, and then modified back to the original measurement dates using Eq. (3). ESPERs and PyESPERs estimate the C_{ant} component of DIC and pH_T in output variables for 2002 by interpolating the 2002 C_{ant} grid to user-provided coordinates and then applying Eq. (3) to estimate C_{ant} for the user-requested estimate year. As with the original ESPERs, this method is not meant to be used when C_{ant} is of primary interest but rather provides a means of quickly adjusting DIC or pH_T to a reference year (Carter et al., 2021). Likewise, these methods are not adequate for making reliable projections beyond the year 2030, or perhaps sooner in coastal or other areas where the underlying global open ocean anthropogenic carbon estimations have greater uncertainties (Carter et al., 2021).

2.2 Uncertainty estimation

ESPERs and PyESPERs return depth- and salinity-dependent uncertainties for each property at the 1σ (one standard uncertainty) level, meaning approximately 95 % of new open ocean measurements from GLODAPv2.2022 should fall within $\pm$ twice the ESPER uncertainties (Carter et al.,

2021). As in Carter et al. (2021), baseline error estimates in the depth and salinity space (E_{X_Est}) are interpolated based on RMSEs of all predictions from validation versions of the routines within bins of salinity and depth. ESPER_LIRs and PyESPER_LIRs scale these uncertainties using user-provided predictor uncertainty estimates ($E_{Pi_Provided}$). Equation (4) is used when user-provided uncertainties exceed the default assumed input uncertainties ($E_{Pi_Default}$; Table A3):

$$E_{X_Output} = \sqrt{E_{X_Est}^2 - \sum_{i=1}^n \left(\frac{\partial X}{\partial P_i} E_{Pi_Default} \right)^2 + \sum_{i=1}^n \left(\frac{\partial X}{\partial P_i} E_{Pi_Provided} \right)^2}, \quad (4)$$

where $\frac{\partial X}{\partial P_i}$ is the sensitivity of the property estimate X to the i th predictor P_i . ESPER_NNs and PyESPER_NNs estimate sensitivities by iteratively perturbing the input predictors if the user specifies uncertainties that are larger than the default. Mixed uncertainties are the minimum uncertainties assessed for LIR and NN estimates.

2.3 Assessment

For many applications, the most critical validation is a test of the reconstruction of withheld data. However, such an exercise requires training alternative versions of the method after withholding data, and, as of now, PyESPER is not separately trained but is instead reliant on the ESPER training that was performed and validated previously with MATLAB (Carter et al., 2021). For this publication, we aim to instead show that PyESPER and ESPER provide quantitatively similar results and assert that the validation presented earlier for ESPER in MATLAB can be considered to also be appropriate for PyESPER in all but a limited number of specific exceptional cases. To support this claim, PyESPER and ESPER were used to estimate values for the GLODAPv2.2022 data product (1 381 248 sets of measurements; Fig. 1) with each equation and output variable combination. This dataset included a wide range of input data, and comparison of PyESPER and ESPER was primarily considered based on application to the high-quality “open ocean” (o) portion of the GLODAP dataset as in Carter et al. (2021), defined as GLODAP data with only World Ocean Circulation Experiment (WOCE) data quality control flag categories of 2 (Acceptable) and secondary quality control flag categories of 1 (subjected to full secondary quality control) for all possible input and measurement data and for salinities between 30 and 37 ($n = 306\,227$ for TA, $343\,580$ for DIC, $199\,304$ for pH_T , and $764\,301$ for phosphate, nitrate, silicate, and oxygen). Additional comparison with the entire GLODAPv2.2022 dataset (“whole ocean” or w), including NaNs and anomalous data with salinities < 30 and temperatures $< 0^\circ\text{C}$, which are not recommended for use with ESPERs, is presented in Appendix B. These comparisons are used as a rigorous test of the fidelity of the PyESPER estimates to the ESPER esti-

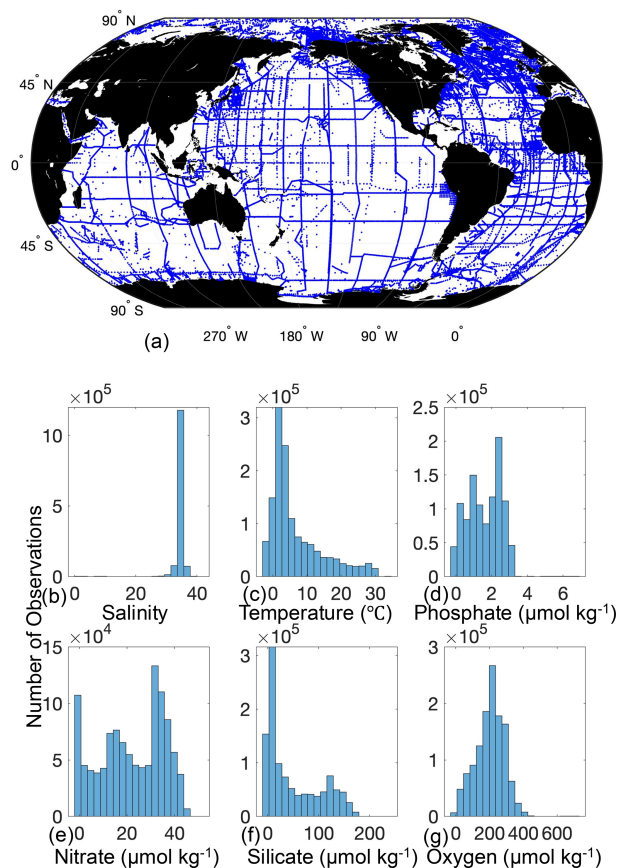


Figure 1. Location of GLODAPv2.2022 data used to compare PyESPER to MATLAB ESPER estimates (a), and histograms of the distributions of measured GLODAPv2.2022 variables used as inputs for the PyESPERv1.0.0 and ESPER algorithms (b–g).

mates. Resulting estimates were compared graphically and with the normalized root mean square error ($RMSE_n$; equivalent to RMSE divided by the mean of the MATLAB estimate for each variable) for each equation case globally and regionally and across depths. $RMSE_n$ was used because it allows for comparison between variables of different scales. Additionally, where measured values were present in the dataset, both ESPER and PyESPER were validated against the measured data, though, again, this is not a validation of the method as much as a check that both variants provide similar values.

2.3.1 DIC application

As an additional comparison of the LIR method differences, DIC estimates from both PyESPER_LIR and ESPER_LIR were applied to the Roemmich–Gilson Argo climatology (Roemmich and Gilson, 2009) to create mapped annual surface estimates of DIC.

3 Results and discussion

PyESPER and ESPER produced open ocean estimates with mean differences (Python estimate–MATLAB estimate) of $< \pm 0.04$ for all parameters, and NNs had smaller mean differences of $< \pm 0.004$ for all parameters (units are $\mu\text{mol kg}^{-1}$ except for pH_T) estimated from the open ocean GLODAPv2.2022 data, although the standard deviations of these differences and uncertainties associated with the estimates were at times larger than the mean differences (Tables 1 and 2). The greatest RMSE_n was 2.08×10^{-2} for silicate estimates using LIRs. PyESPER_NN functioned as an equivalent data product to ESPER_NN for all data. For open ocean data, PyESPER_LIRs functioned similarly to ESPER_LIRs, with a large majority of identical estimates produced between the two data products.

3.1 Data product validation

The results of comparisons between MATLAB ESPERs and PyESPERs are described below.

3.1.1 Locally interpolated regressions

When compared to the ESPER_LIR results for the open ocean ($\circ$) GLODAPv2.2022 dataset, all equation–case and desired outcome variable combinations from PyESPER (PyESPER_LIR–ESPER_LIR estimates) resulted in mean differences of $< \pm 0.04$ (Table 1). Mean ($\pm$ standard deviation; RMSE_n) PyESPER–ESPER_LIR differences for each property are shown in Table 1. The very wide range of input data resulted in a wide range of estimates from both ESPER_LIRs and PyESPER_LIRs for all variables (Table 1; Fig. 2; for w , see Appendix B, Fig. B1), representing the large range of biogeochemical property values that can be found in the oceans. The PyESPER_LIR and ESPER_LIR results worked similarly well in predicting measured values at locations, even with the outlier and unusual input data used (see Table B1), suggesting that the Python estimates, although not identical to the MATLAB estimates for these interpolations, were equivalently valid reconstructions.

PyESPER_LIRs were within 2σ ($\sim 95\%$ of measurements should fall within this uncertainty level) for most ocean regions, with a few exceptions that occurred predominantly in coastal areas or deep waters near the edges of the original MATLAB grid (Figs. 3 and 4). Spatial patterns in the distribution of outliers shown in Fig. 4 appear to reflect locations where more edge-of-grid biogeochemical measurements were collected (e.g., near coasts and in deep waters). Hence, these exceptionally different locations aligned well with places where coefficients were extrapolated in MATLAB for use in PyESPER_LIRs, compared to interpolations with distant “dummy points” within MATLAB ESPER_LIRs (see Sect. 2.1.1, *Locally interpolated regressions*, Figs. 3–5; for w , Figs. B2 and B3). Within regions where MAT-

Table 1. Mean (standard deviation), maximum, minimum, and normalized RMSE (RMSE_n) for differences between MATLAB and Python LIRs, ESPER_LIR and measured values, and PyESPER_LIR and measured values for TA, DIC, pH_T , phosphate, nitrate, silicate, and oxygen estimates (all units, except for pH_T , are $\mu\text{mol kg}^{-1}$) for open ocean ($\circ$) data and all equations combined ($n=13\,384\,096$ for TA, $13\,384\,096$ for DIC, $13\,384\,096$ for pH_T , $13\,384\,096$ for phosphate, $12\,718\,592$ for nitrate, $12\,640\,896$ for silicate, and $12\,757\,792$ for oxygen).

	Python–MATLAB					MATLAB–Measured					Python–Measured				
	Mean	Max	Min	RMSE_n	Mean	Max	Min	RMSE_n	Mean	Max	Min	RMSE_n	Mean	Max	Min
TA	-4.75×10^{-4} (1.08)	6.44×10^1	-7.03×10^1	4.64×10^{-4}	2.71×10^{-1} (6.34)	8.13×10^2	-1.69×10^2	2.72×10^{-3}	2.70×10^{-1} (6.32)	8.13×10^2	-1.73×10^2	2.71×10^{-3}	-4.02×10^{-1} (8.47)	6.20×10^2	-3.16×10^2
DIC	3.39×10^{-2} (1.60)	2.01×10^2	-2.61×10^2	7.29×10^{-4}	-4.40×10^{-1} (8.55)	6.20×10^2	-3.20×10^2	3.90×10^{-3}	-4.02×10^{-1} (8.47)	6.20×10^2	-3.16×10^2	3.90×10^{-3}	-2.56×10^{-3} (2.23)	1.14×10^0	-5.46×10^{-1}
pH	-5.65×10^{-5} (4.24)	5.05×10^{-1}	-3.77×10^{-1}	5.36×10^{-4}	-2.51×10^{-3} (2.24)	1.14×10^0	-6.80×10^{-1}	2.86×10^{-3}	-2.56×10^{-3} (2.23)	1.14×10^0	-5.46×10^{-1}	2.84×10^{-3}	-1.61×10^{-4} (6.09)	2.57×10^0	-3.50×10^0
Phosphate	3.08×10^{-4} (1.41)	1.65	-2.17	8.44×10^{-3}	-1.54×10^{-4} (6.21)	2.90	-3.12×10^0	3.90×10^{-2}	-1.61×10^{-4} (6.09)	2.57×10^0	-3.50×10^0	3.61×10^{-2}	-7.67×10^{-3} (8.14)	3.04×10^1	-4.24×10^{-1}
Nitrate	2.20×10^{-3} (3.08)	1.89×10^1	-4.13×10^1	1.30×10^{-2}	-5.43×10^{-3} (8.58)	4.23×10^1	-3.45×10^1	3.62×10^{-2}	-7.67×10^{-3} (8.14)	3.04×10^1	-4.24×10^{-1}	3.43×10^{-2}	-4.28×10^{-2} (3.01)	8.20×10^1	-2.08×10^2
Silicate	2.27×10^{-2} (1.18)	5.92×10^1	-5.85×10^1	2.08×10^{-2}	-6.60×10^{-2} (3.19)	8.42×10^1	-2.08×10^2	5.57×10^{-2}	-4.28×10^{-2} (3.01)	8.20×10^1	-2.08×10^2	5.25×10^{-2}	3.21×10^2	-3.75×10^2	4.55×10^{-2}
Oxygen	3.98×10^{-3} (2.11)	3.08×10^2	-2.31×10^2	1.10×10^{-2}	6.06×10^{-2} (8.90)	3.28×10^2	-4.23×10^2	4.66×10^{-2}	6.32×10^{-2} (8.70)	3.21×10^2	-3.75×10^2	4.55×10^{-2}			

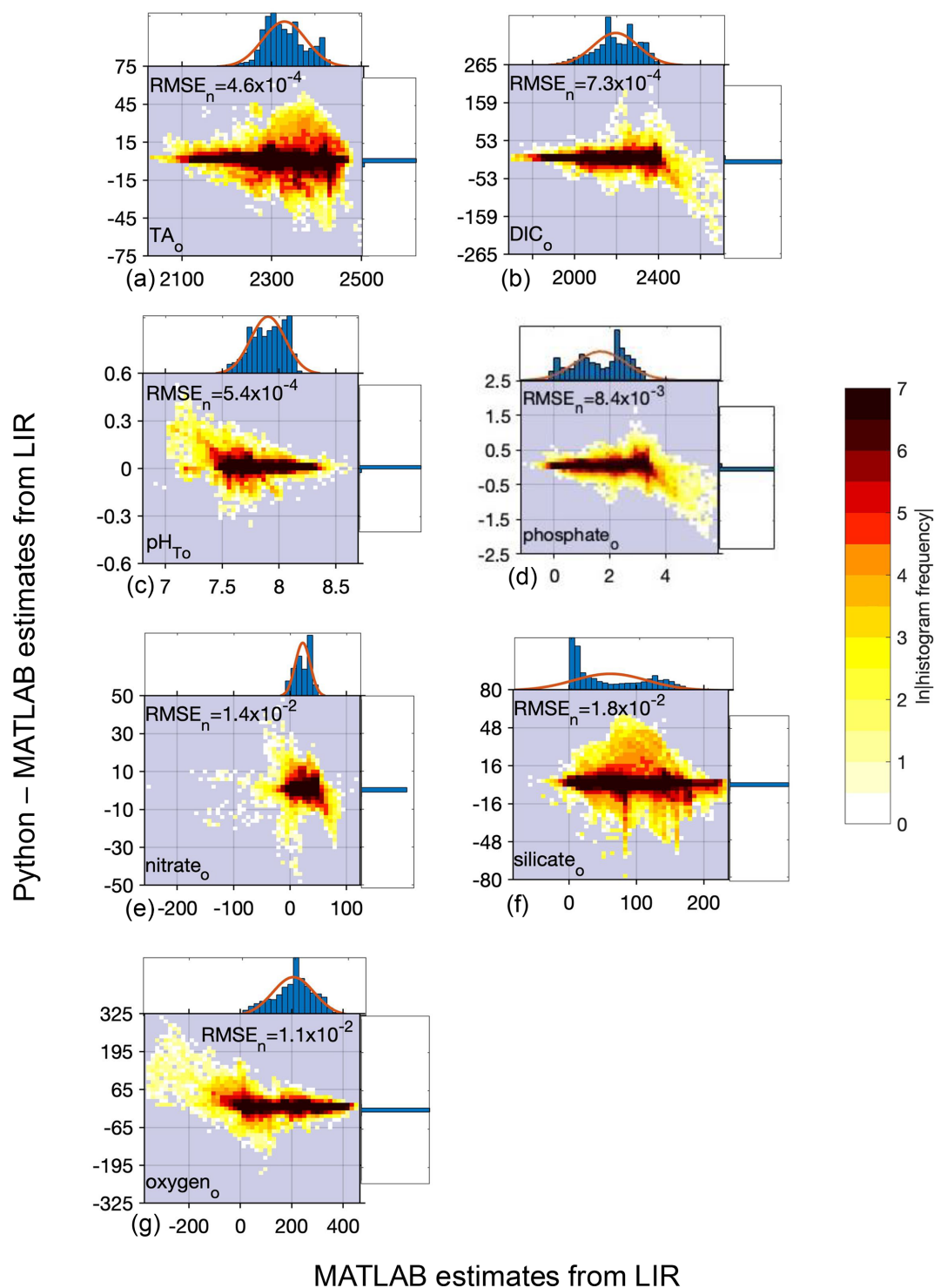


Figure 2. Difference between Python and MATLAB locally interpolated regression estimates (y axis) compared to MATLAB estimates (x axis) for open ocean (o) data and all equations combined for TA (a, 13 384 096 total estimates from all equations), DIC (b, 13 384 096 estimates), pH_T (c, 13 384 096 estimates), phosphate (d, 13 384 096 estimates), nitrate (e, 12 718 592 estimates), silicate (f, 12 640 896), and oxygen (g, 12 757 792 estimates); $n = 306\,227$ for TA, 343 580 for DIC, 199 304 for pH_T, and 764 301 for phosphate, nitrate, silicate, and oxygen. All units, except for pH_T, are $\mu\text{mol kg}^{-1}$. Top and side histograms represent the distribution of the x and y axes, respectively. Note the differences in the x- and y-axis scales. RMSE_n is the normalized root mean square error, or the RMSE of all estimates divided by the mean of all estimates.

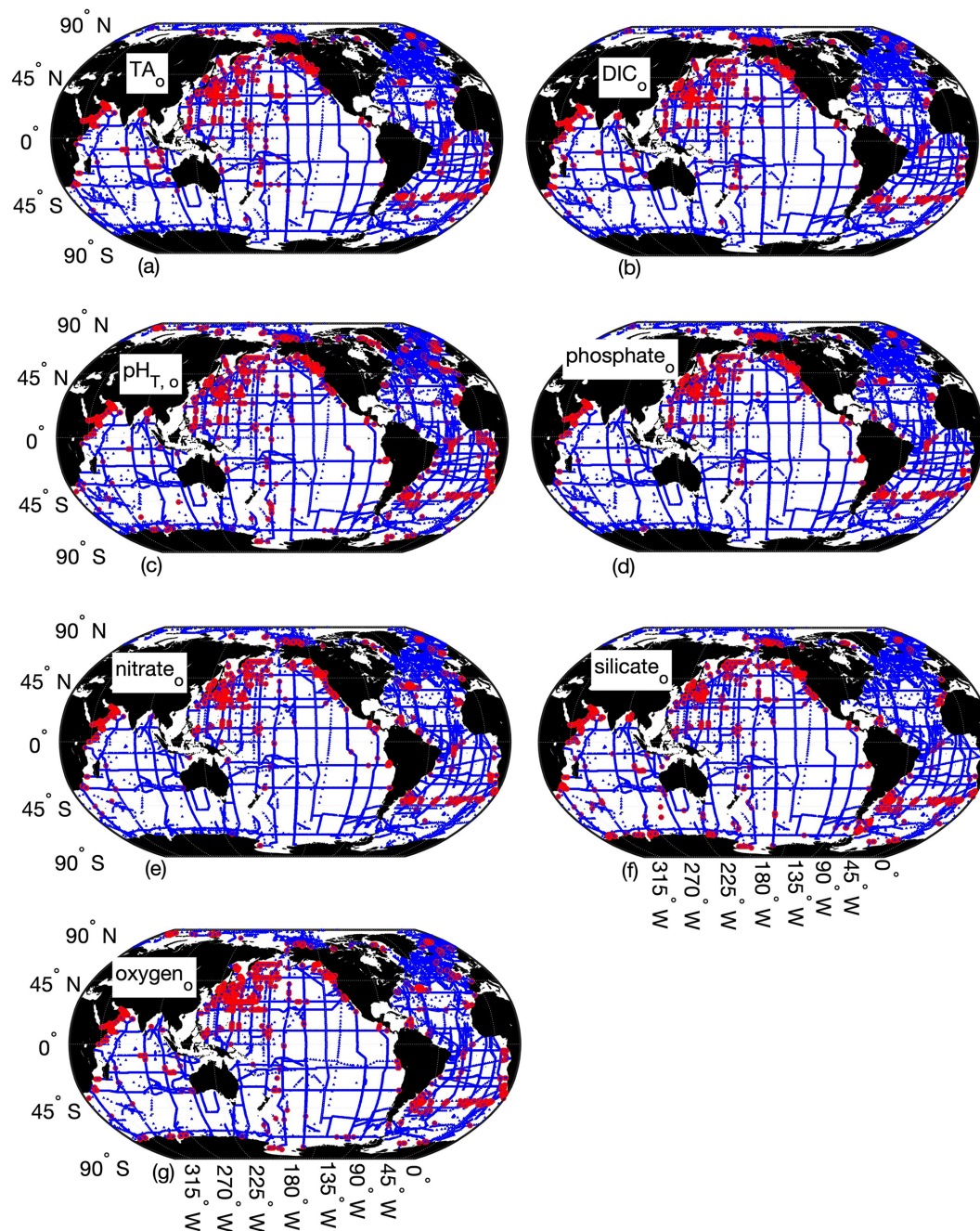


Figure 3. Map of differences between Python and MATLAB ESPER locally interpolated regression estimates (total estimates $n = 13\,384\,096$ for TA (a), DIC (b), pH_T (c), and phosphate, 12 718 592 for nitrate (d), 12 640 896 for silicate (e), and 12 757 792 for oxygen (f)) for the open ocean (_o), where the small blue circles represent differences < 2 times the uncertainties of the MATLAB estimates ($n = 13\,344\,924$ for TA, 13 354 980 for DIC, 13 349 438 for pH_T , 13 357 843 for phosphate, 12 688 861 for nitrate, 12 597 608 for silicate, and 12 721 483 for oxygen) and red circles represent differences > 2 times the uncertainties of the MATLAB estimates ($n = 39\,172$ for TA, 29 116 for DIC, 34 658 for pH_T , 26 253 for phosphate, 29 731 for nitrate, 43 288 for silicate, and 36 309 for oxygen).

LAB and Python were interpolating similarly, far outliers were uncommon (Figs. 3–5, B2, and B3). When ESPER_LIR and PyESPER_LIR were applied to temperature and salinity from the Roemmich and Gilson climatology for the year 2023 (Roemmich and Gilson, 2009), the patterns of the sur-

face DIC distribution were similar, with a few minor nuances (Fig. C1). Notably, low DIC estimates covered a broader spatial extent in the western equatorial Pacific and Indian oceans for the PyESPER_LIR estimates, and PyESPER_LIR appeared to have a slightly low bias in some places rela-

Table 2. Mean (standard deviation), maximum, minimum, and normalized RMSE ($RMSE_n$) are shown for three scenarios: (1) between Python and MATLAB NNs, (2) between MATLAB ES-NN and measured values, and (3) between PyESPER-NN and measured values. Separate rows exist for TA, DIC, pH_T , phosphate, nitrate, silicate, and oxygen estimates. All units, except for pH_T , are $\mu\text{mol kg}^{-1}$, and data are for open oceans (o) and all equations combined.

	Python–MATLAB				MATLAB–Measured				Python–Measured			
	Mean	Max	Min	$RMSE_n$	Mean	Max	Min	$RMSE_n$	Mean	Max	Min	$RMSE_n$
TA	-4.49×10^{-12} (5.89×10^{-9})	4.00×10^{-6}	-2.00×10^{-6}	2.53×10^{-12}	3.40×10^{-1} (5.21)	8.15×10^2	-1.78×10^2	2.24×10^{-3}	3.40×10^{-1} (5.21)	8.15×10^2	-1.78×10^2	2.24×10^{-3}
DIC	-3.01×10^{-3} (9.29×10^{-2})	2.31	−3.69	4.22×10^{-5}	-2.94×10^{-1} (7.67)	6.17×10^2	-3.37×10^2	3.49×10^{-3}	-2.97×10^{-1} (7.67)	6.18×10^2	-3.37×10^2	3.49×10^{-3}
pH_T	1.07×10^{-5} (2.34×10^{-4})	5.60×10^{-3}	-7.65×10^{-3}	2.98×10^{-5}	-4.59×10^{-3} (1.59×10^{-2})	4.71×10^{-1}	-6.58×10^{-1}	2.10×10^{-3}	-4.58×10^{-3} (1.59×10^{-2})	4.71×10^{-1}	-6.58×10^{-1}	2.10×10^{-3}
Phosphate	-6.19×10^{-14} (6.60×10^{-11})	2.50×10^{-8}	-1.25×10^{-7}	1.31×10^{-10}	1.15×10^{-3} (5.19×10^{-2})	2.12	−2.81	3.06×10^{-2}	1.15×10^{-3} (5.19×10^{-2})	2.12	−2.81	3.06×10^{-2}
Nitrate	-7.80×10^{-13} (8.91×10^{-10})	1.35×10^{-7}	-2.28×10^{-6}	3.76×10^{-11}	-2.24×10^{-3} (7.17×10^{-1})	4.06×10^1	-3.40×10^1	2.93×10^{-2}	-2.24×10^{-3} (7.17×10^{-1})	4.06×10^1	-3.40×10^1	2.93×10^{-2}
Silicate	-1.24×10^{-12} (1.98×10^{-9})	2.11×10^{-7}	-2.97×10^{-6}	3.42×10^{-11}	4.96×10^{-3} (2.55)	1.23×10^2	-8.29×10^1	4.42×10^{-2}	4.96×10^{-3} (2.55)	1.23×10^2	-8.29×10^1	4.42×10^{-2}
Oxygen	-4.42×10^{-13} (2.09×10^{-10})	1.00×10^{-8}	-1.00×10^{-7}	1.06×10^{-12}	5.33×10^{-2} (7.29)	3.54×10^2	-2.06×10^2	3.82×10^{-2}	5.33×10^{-2} (7.29)	3.54×10^2	-2.06×10^2	3.82×10^{-2}

tive to ES-NN. Beyond these minor differences, the mapped DIC demonstrates the similarity of the data products’ functionality in an applied setting. While ES-NN and PyESPER-NN do not produce quantitatively identical estimates, it should be noted that both routines perform similarly well at reconstructing the GLODAPv2.2022 data product (Table 1; for w , Table B1). These routines should not be considered identical but are comparable.

3.1.2 Neural networks

When compared to the ES-NN results for the open ocean (o) GLODAPv2.2022 dataset, all equation–case and desired outcome variable combinations from PyESPER-NN (PyESPER–ES-NN estimates) resulted in mean differences of $< \pm 0.004$ (Table 2), a much smaller difference than for the LIR comparisons. The mean ($\pm$ standard deviation; $RMSE_n$) offset for each property is shown in Table 2. Because a very wide range of input data were used, a wide range of estimates were produced from both ES-NNs and PyESPER-NNs for all variables (Fig. 6), representing the high variability that can be found in the oceans (especially coastal regions, some of which were included in the “open ocean” dataset due to having salinities between 30 and 37 and quality-controlled data). Both the PyESPER-NN and ES-NN results were nearly identical, even when outlier results were obtained from unusual input data from environments where ES-NNs are not recommended for use (for example, resulting in negative DIC estimates in Fig. B4; see also Table B2). The largest relative disagreements were found for DIC and pH_T , though these disagreements remained small relative to the measurement uncertainties. These minor offsets are attributed to the programming language differences in the interpolation of the C_{ant} adjustment, which is applied only to these two properties.

3.1.3 Anthropogenic carbon estimates

Although inconsistencies in the results occur between Python and MATLAB when interpolating (same issue noted in Sect. 2.1.4, *Anthropogenic carbon*), the anthropogenic carbon (C_{ant}) estimates were similar between the two versions of ES-NN. This was demonstrated by differences in the DIC and pH_T estimates for NNs, which interpolate only when estimating the contribution of C_{ant} to the estimates (Fig. 6). The next generation of ES-NN updates will include a new method for estimating C_{ant} (Tracer-Based Rapid Anthropogenic Carbon Estimation, or TRACEv1; Carter et al., 2025), which uses neural networks and should eliminate the need for interpolation. Currently, when C_{ant} estimates are required, the results from PyESPER-NNs remain functionally identical to those from ES-NNs, despite minor offsets from the interpolation methods.

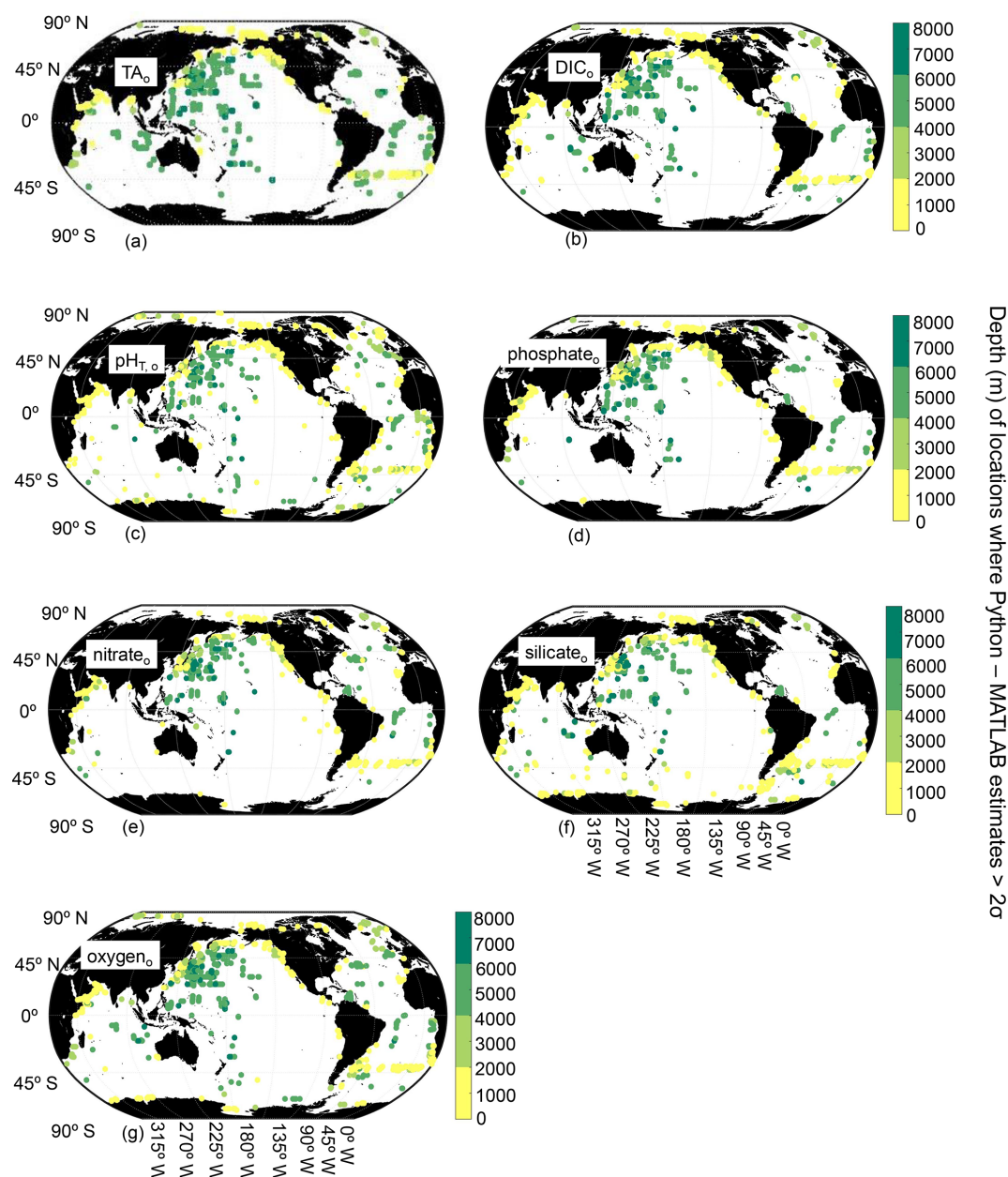


Figure 4. Map of locations and depths (color bar) where differences between Python and MATLAB ESPER locally interpolated regression estimates are greater than 2 times the estimate uncertainties for the open ocean (_o; $n = 13\,344\,924$ for TA (a), $13\,354\,980$ for DIC (b), $13\,349\,438$ for pH_T (c), $13\,357\,843$ for phosphate (d), $12\,688\,861$ for nitrate (e), $12\,597\,608$ for silicate (f), and $12\,721\,483$ for oxygen (g)).

3.2 Speed of calculation

PyESPERs take considerably longer than ESPERs to produce estimates. On a MacBook Air using Python in the terminal with a standard internet connection, PyESPER_NN produced results 0–1500 times slower than ESPER_NN, while PyESPER_LIR produced results about 7–500 times slower than ESPER_LIRs, with the magnitude of the slowdown dependent upon the number of variable inputs and equation cases requested and the number of estimates re-

quired (Table 3). ESPER_NNs were the fastest to execute and took < 2 s for all time tests, even when large datasets and all variable–equation case scenarios were requested. ESPER_LIRs were the next-fastest, requiring < 33 s for all time tests, followed by PyESPER_NNs, which typically required 5–15 s to execute but required > 1400 s (23 min) for running large datasets and all variable–equation case scenarios. PyESPER_LIRs were the slowest and typically required 22–500 s to execute, but the longest scenario required 7530 s

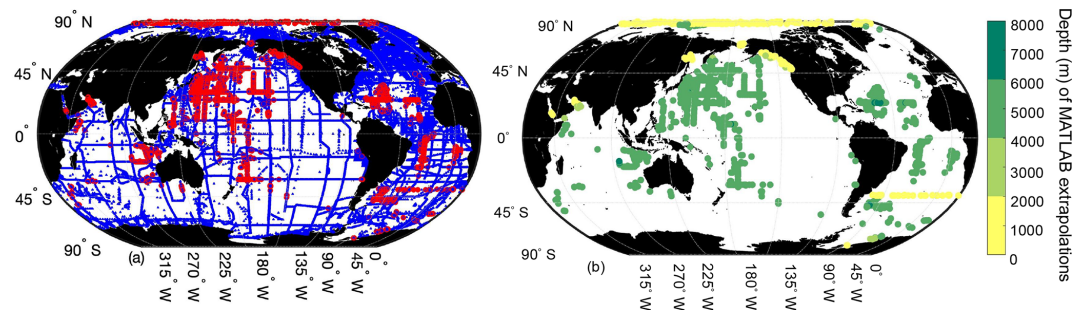


Figure 5. Map of locations where MATLAB was interpolating ($n = 1\,365\,170$, blue) and extrapolating ($n = 16\,078$, red) from the grid to the GLODAPv2.2022 data (a), and the depth of the extrapolations (b).

Table 3. Time required to produce estimates for PyESPERv1.0.0 and ESPERs (LIRs and NNs) for different desired variables, equation–case combinations, and numbers of estimates.

Variable	Equation(s)	Number of Estimates	PyESPER_NN time (s)	ESPER_NN time (s)	PyESPER_LIR time (s)	ESPER_LIR time (s)
TA	1	10	0.06	0.01	193.16	0.77
TA	1	100	0.12	0.01	207.68	0.60
TA	2	100	0.13	0.01	204.92	0.79
TA	3	100	0.14	0.01	216.22	0.81
TA	4	100	0.14	0.01	209.98	0.78
TA	5	100	0.12	0.00	194.39	0.75
TA	6	100	0.12	0.01	194.30	0.79
TA	7	100	0.13	0.00	201.98	0.78
TA	8	100	0.12	0.00	195.30	0.80
TA	9	100	0.14	0.00	195.42	0.81
TA	10	100	0.11	0.01	194.58	0.74
TA	11	100	0.13	0.00	193.33	0.74
TA	12	100	0.13	0.00	201.39	0.75
TA	13	100	5.88	0.00	22.37	0.79
TA	14	100	5.82	0.01	22.46	0.77
TA	15	100	5.81	0.00	22.35	0.84
TA	16	100	5.81	0.01	22.57	0.74
TA	1–16	100	11.06	0.04	312.13	0.62
TA	1	1000	11.50	0.03	29.69	0.76
TA	1	10 000	61.54	0.12	57.59	0.83
TA	1	100 000	950.78	0.62	325.87	1.55
DIC	1	100	5.86	1.55	32.51	2.69
DIC	1–16	100	10.86	1.53	365.58	1.54
pH	1	100	6.09	0.06	54.65	0.81
pH	1–16	100	15.37	0.46	766.74	3.41
Phosphate	1	100	5.85	0.01	23.46	3.39
Phosphate	1–16	100	11.01	0.06	376.30	0.80
Nitrate	1	100	5.85	0.01	23.07	0.74
Nitrate	1–16	100	11.04	0.05	364.13	3.56
Silicate	1	100	5.84	0.02	26.84	3.64
Silicate	1–16	100	11.02	0.04	365.34	0.82
Oxygen	1	100	6.97	0.01	24.60	0.78
Oxygen	1–16	100	10.98	0.04	385.28	2.15
All Variables	1	100	11.81	0.01	194.31	13.86
All Variables	1	10 000	147.26	0.10	561.29	15.17
All Variables	1–16	100	49.53	0.09	3182.56	15.26
All Variables	1–16	10 000	1443.63	1.67	7530.23	32.13

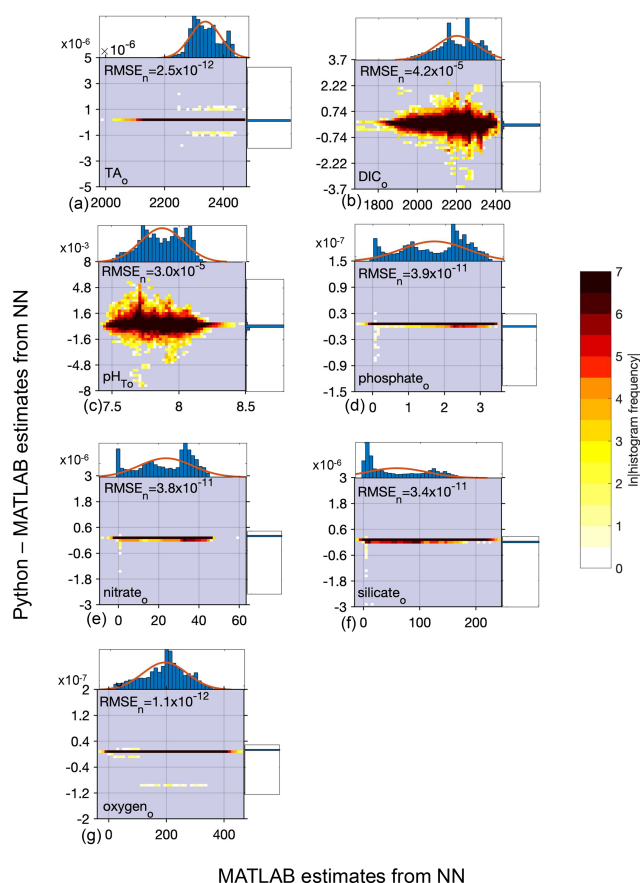


Figure 6. Difference between Python and MATLAB neural network estimates (y axis) compared to MATLAB estimates (x axis) for open ocean (_o) data and all equations combined for TA (a, 4 899 512 total estimates from all equations), DIC (b, 5 497 004 estimates), pH_T (c, 3 188 864 estimates), phosphate (d, 12 228 432 estimates), nitrate (e, 12 228 432 estimates), silicate (f, 12 228 432 estimates), and oxygen (g, 12 228 560 estimates); $n = 306\,227$ for TA, 343 580 for DIC, 199 304 for pH_T , and 764 301 for phosphate, nitrate, silicate, and oxygen. All units, except for pH_T , are $\mu\text{mol kg}^{-1}$. Top and side histograms represent the distribution of the x and y axes, respectively. Note the differences in the x- and y-axis scales. RMSE_n is the normalized root mean square error, or the RMSE divided by the mean of all estimates from MATLAB_NN.

(125 min; Table 3). It is possible that this code can be further optimized for speed in future updates.

3.3 Future improvements

Updated ESPERs will be trained and assessed using GLODAPv2.2023 (or later versions), which includes 1108 cruises (compared to 946 cruises from GLODAPv2.2020, the current data product used). Additionally, future ESPERs will incorporate depth (z) as an optional predictor variable for consistency with LIPHR, a prior version for estimating pH_T (Carter et al., 2018). The implementation of updated C_{ant} estimation methods should additionally improve the accuracy and effi-

ciency of both ESPERs and PyESPERs when C_{ant} estimates are required. Future versions of ESPER written in MATLAB may be modified to improve interoperability with the Python implementation (i.e., to ensure the interpolation routines are identical in all instances between languages).

4 Conclusions

A near-replicate of ESPERs has been produced in the freely available Python programming language. This algorithm data product will offer Python users or researchers with limited funds an alternate, free method for using ESPERs (other than the proprietary MATLAB), increasing the accessibility of the original ESPER algorithms. The same logic applied to the original MATLAB ESPERs was applied within the Python coding language (PyESPERs, version 1.0.0), and the results have demonstrated comparability to the ESPER estimates. Estimates from PyESPER_NNs precisely align with those from ESPER_NNs for all equations and desired outcome variable combinations (Fig. 6), and estimates from these two routines align very closely for all estimates and to within machine precision for all but pH_T and DIC, which exhibit slight differences due to the impacts of interpolating for C_{ant} . The PyESPER_LIR estimates differ from the ESPER_LIR estimates for some coastal and deep-water regions between the two coding languages due to triangulation, extrapolation, and interpolation differences, but they were more similar throughout all portions of the open ocean (Figs. 2–4). Notably, PyESPER_LIR performs equivalently to ESPER_LIR when reconstructing the training data from GLODAPv2.v2022, so estimates produced from these two routines should be considered comparable rather than identical. Nevertheless, we do not recommend using PyESPER_LIR in coastal or deep ($> 5500\text{ m}$) waters when primarily interested in comparing results with those of the MATLAB implementation of ESPER_LIR. Future updates to ESPERs will include updates to PyESPERs, with adjustments to allow for greater consistency and speed.

Appendix A: ESPer specifications

The sets of equations, predictor variables, and measurement uncertainties used in ESPer and PyESPer (adapted from Carter et al., 2021) are shown below.

Table A1. Input predictor variable combinations used for each ESPer equation (adapted from Carter et al., 2021), where S is salinity, T is temperature, and A , B , and C are as defined in Table A2 (below).

Equation number	Predictor variables
1	S, T, A, B, C
2	S, T, A, C
3	S, T, B, C
4	S, T, C
5	S, T, A, B
6	S, T, A
7	S, T, B
8	S, T
9	S, A, B, C
10	S, A, C
11	S, B, C
12	S, C
13	S, A, B
14	S, A
15	S, B
16	S

Table A2. Input predictor variables (A , B , and C) for each estimated property (adapted from Carter et al., 2021).

Estimated property	A	B	C
TA	Nitrate	Oxygen	Silicate
DIC	Nitrate	Oxygen	Silicate
pH _T	Nitrate	Oxygen	Silicate
Phosphate	Nitrate	Oxygen	Silicate
Nitrate	Phosphate	Oxygen	Silicate
Silicate	Phosphate	Oxygen	Nitrate
Oxygen	Phosphate	Nitrate	Silicate

Table A3. Default measurement uncertainties ($E_{Pi_Default}$) for ESPer and PyESPer (adapted from Carter et al., 2021), where θ is the potential temperature.

Property	Units	Uncertainty
S	—	0.003, absolute
θ	°C	0.003, absolute
Phosphate	$\mu\text{mol kg}^{-1}$	2 %, relative
Nitrate	$\mu\text{mol kg}^{-1}$	2 %, relative
Silicate	$\mu\text{mol kg}^{-1}$	2 %, relative
Oxygen	$\mu\text{mol kg}^{-1}$	1 %, relative

Appendix B: Comparison using the entire GLODAPv2.2022 dataset

The results of comparisons of PyESPer with ESPer for the entire GLODAPv2.2022 dataset, including the entire oceanic and coastal salinity range and data of all quality control flag categories, are shown below.

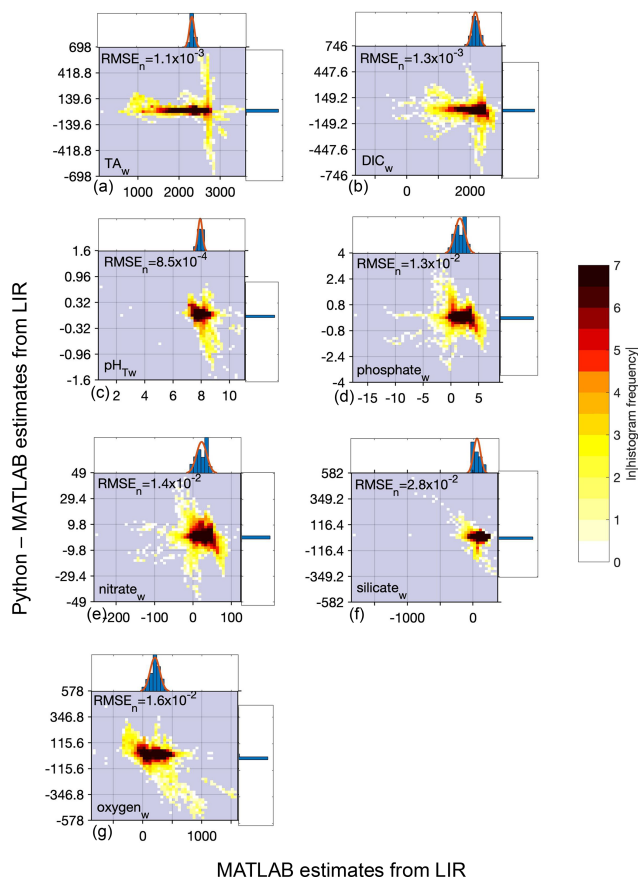


Figure B1. Difference between Python and MATLAB locally interpolated regression estimates (y axis) compared to MATLAB estimates (x axis) for whole ocean (_w) data and all equations combined (22 099 968 total estimates from all equations for each variable) for TA (a), DIC (b), pH_T (c), phosphate (d), nitrate (e), silicate (f), and oxygen (g) derived using all equations and calculated from the entire GLODAPv2.2022 dataset ($n = 1\,381\,248$). All units, except for pH_T, are $\mu\text{mol kg}^{-1}$. Top and side histograms represent the distribution of the x and y axes, respectively. Note the differences in the x - and y -axis scales. RMSE_n is the normalized root mean square error, or the RMSE of all estimates divided by the mean of all MATLAB estimates. The large range of sometimes unrealistic estimates along the x axis can be attributed to the anomalous and sometimes erroneous input data used for predictions.

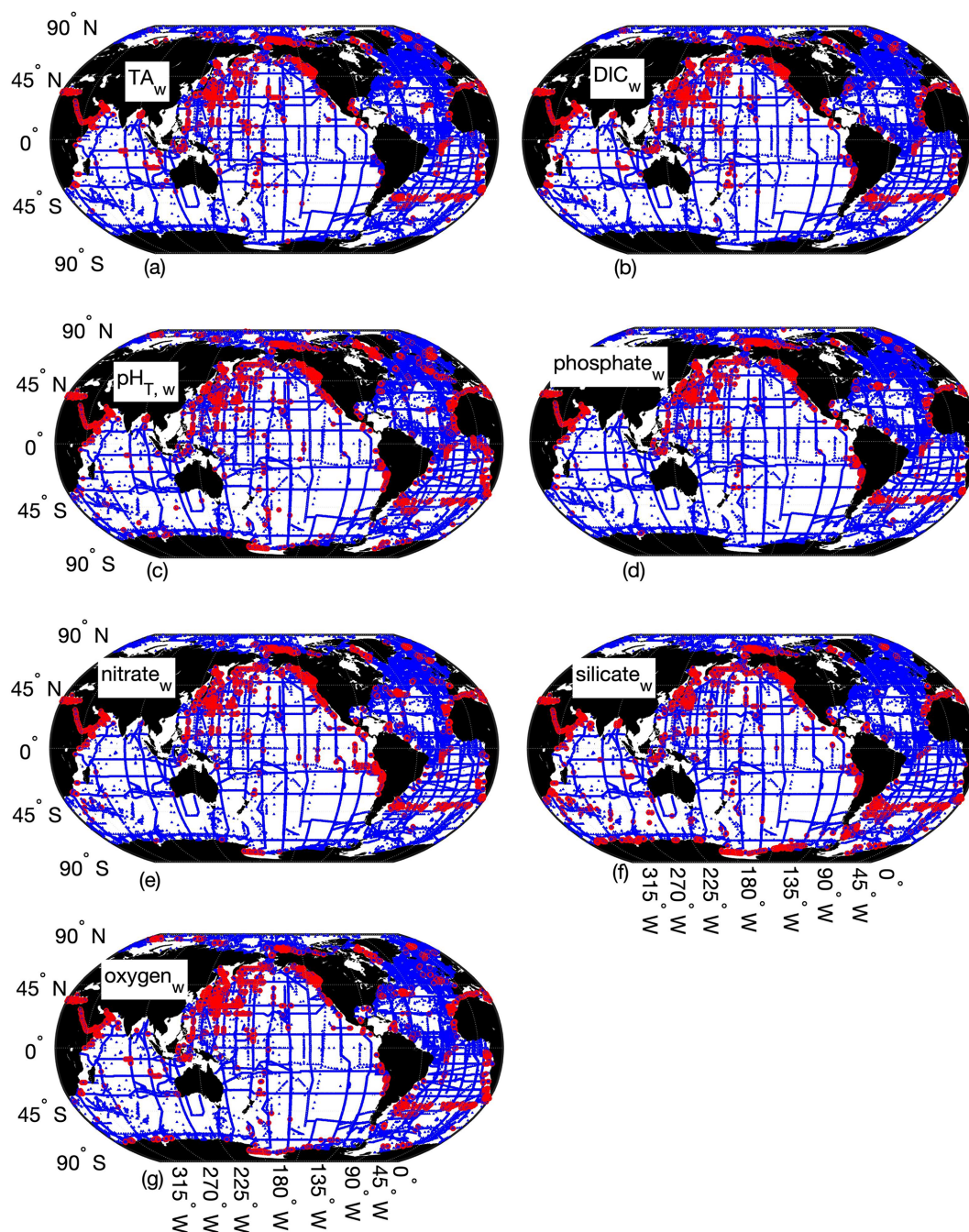


Figure B2. Map of differences between Python and MATLAB ESPER locally interpolated regression estimates (total estimates $n = 22\,099\,968$ for all variables) for the whole ocean (w), where the small blue circles represent differences < 2 times the uncertainties of the MATLAB estimates ($n = 22\,034\,967$ for TA (a), $22\,054\,048$ for DIC (b), $22\,045\,316$ for $\text{pH}_{T,w}$ (c), $22\,057\,220$ for phosphate (d), $22\,045\,770$ for nitrate (e), $22\,024\,674$ for silicate (f), and $22\,045\,827$ for oxygen (g)) and red circles represent differences > 2 times the uncertainties of the MATLAB estimates ($n = 65\,001$ for TA, $45\,920$ for DIC, $54\,642$ for pH, $42\,748$ for phosphate, $54\,198$ for nitrate, $75\,294$ for silicate, and $54\,141$ for oxygen; $n = 1\,381\,248$).

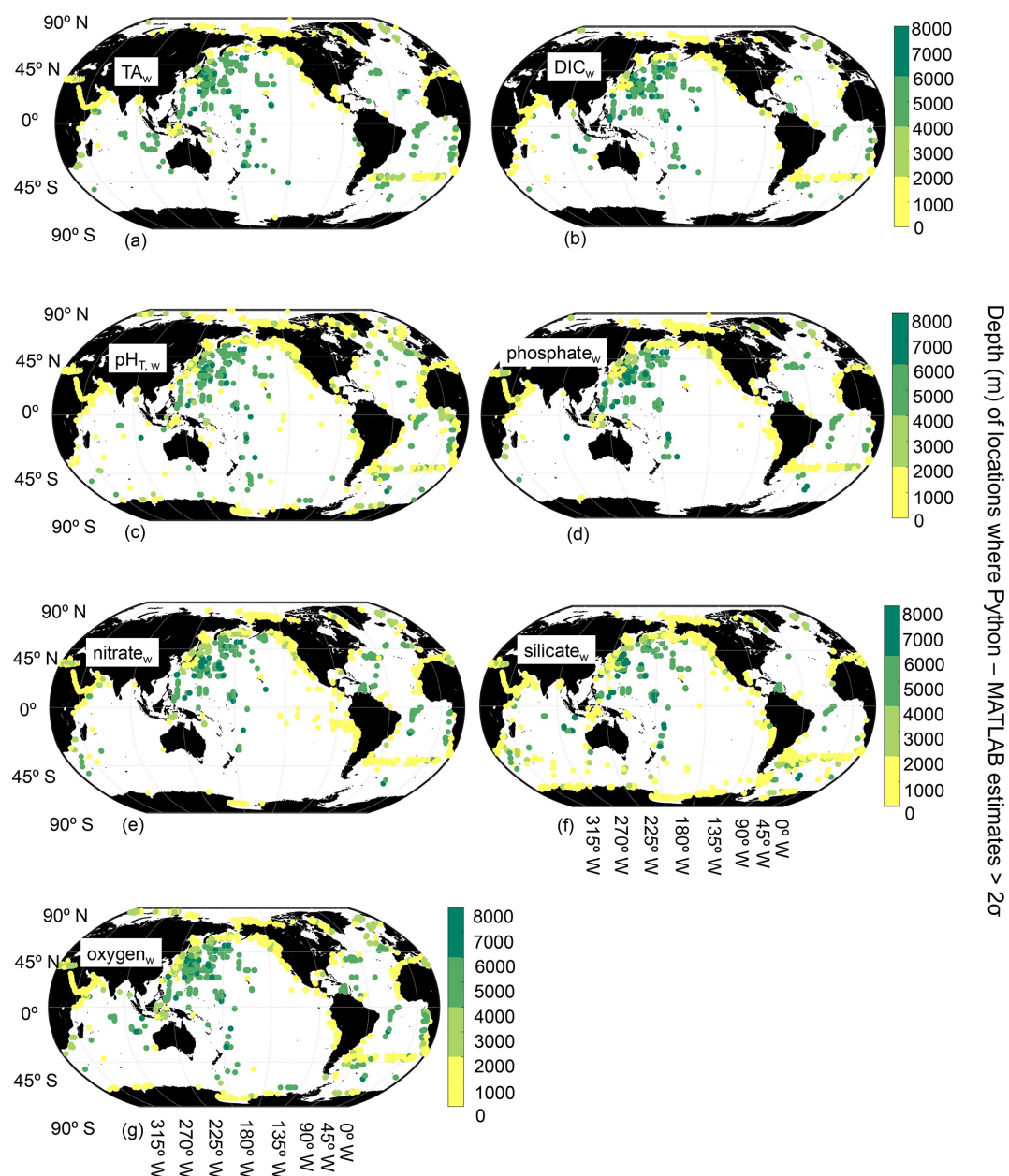


Figure B3. Map of locations and depths (color bar) where the differences between Python and MATLAB ESPER locally interpolated regression estimates are greater than 2 times the estimate uncertainties for the whole ocean (w , $n = 22\,034\,967$ for TA (a), $22\,054\,048$ for DIC (b), $22\,045\,316$ for pH_T (c), $22\,057\,220$ for phosphate (d), $22\,045\,770$ for nitrate (e), $22\,024\,674$ for silicate (f), and $22\,045\,827$ for oxygen (g); $n = 1\,381\,248$).

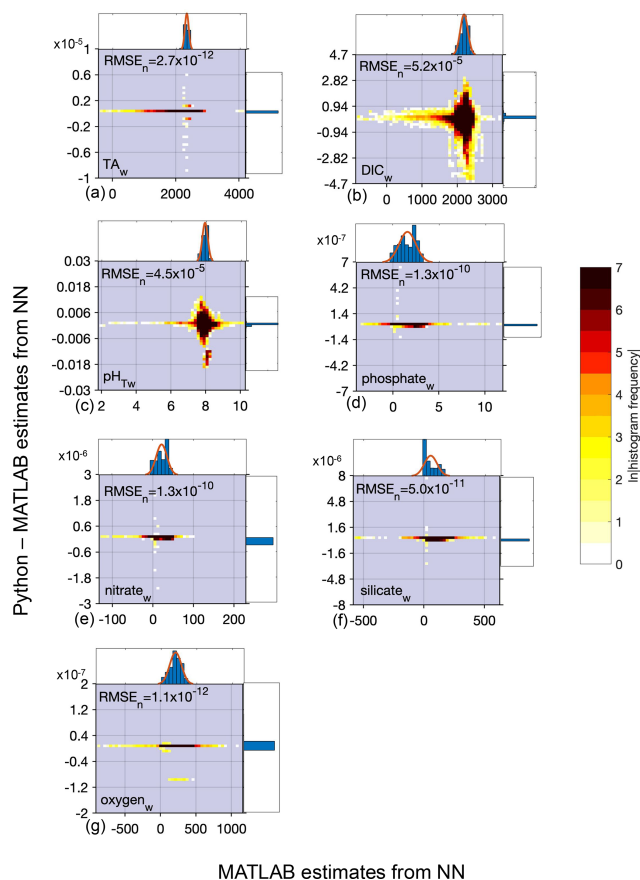


Figure B4. Difference between Python and MATLAB neural network estimates (y axis) compared to MATLAB estimates (x axis) for whole ocean (w) data and all equations combined for TA (**a**, 17 802 134 total estimates from all equations), DIC (**b**, 17 802 134 estimates), pH_T (**c**, 17 799 566 estimates), phosphate (**d**, 17 802 134 estimates), nitrate (**e**, 17 395 954 estimates), silicate (**f**, 17 445 310 estimates), and oxygen (**g**, 17 220 360 estimates) derived using all equations and calculated from the entire GLODAPv2.2022 dataset ($n = 1\,381\,248$). All units, except for pH_T , are $\mu\text{mol kg}^{-1}$. Top and side histograms represent the distribution of the x and y axes, respectively. Note the differences in the x - and y -axis scales. RMSE_n is the normalized root mean square error, or the RMSE of all estimates divided by the mean of all estimates. The large range of sometimes unrealistic estimates along the x axis can be attributed to the anomalous and sometimes erroneous input data used for predictions.

Table B1. Mean (standard deviation), maximum, minimum, and normalized RMSE (RMSE_n) for differences between MATLAB and Python LIRs, ESPer_LIR and measured values, and PyESPer_LIR and measured values for TA, DIC, pH_T , phosphate, nitrate, and oxygen estimates (all units, except for pH_T , are $\mu\text{mol kg}^{-1}$) for all equations combined from the entire GLODAPv2.2022 dataset (w ; $n = 1\,381\,248$).

	Python-MATLAB				MATLAB-Measured				Python-Measured			
	Mean	Max	Min	RMSE_n	Mean	Max	Min	RMSE_n	Mean	Max	Min	RMSE_n
TA	-2.76×10^{-2} (2.61)	6.46×10^2	-6.98×10^2	1.12×10^{-3}	3.36×10^{-1} (1.19×10^1)	1.19×10^3	-7.34×10^2	5.12×10^{-3}	3.22×10^{-1} (1.21×10^1)	1.19×10^3	-7.34×10^2	5.22×10^{-3}
DIC	-5.35×10^{-3} (2.82)	5.17×10^2	-7.46×10^2	1.29×10^{-3}	-6.85×10^{-1} (1.63×10^1)	9.79×10^2	-1.75×10^3	7.46×10^{-3}	-6.69×10^{-1} (1.63×10^1)	9.79×10^2	-1.75×10^3	7.45×10^{-3}
pH	-6.05×10^{-5} (6.74×10^{-3})	7.65×10^{-1}	-1.59×10^0	8.52×10^{-4}	2.03×10^{-3} (3.39×10^{-2})	3.24×10^0	-1.39×10^0	4.29×10^{-3}	2.14×10^{-3} (3.37×10^{-2})	2.85×10^0	-1.40×10^0	4.26×10^{-3}
Phosphate	8.88×10^{-5} (2.02×10^{-2})	3.96	-3.31	1.27×10^{-2}	-1.04×10^{-3} (9.26×10^{-2})	4.77	-1.37×10^1	5.74×10^{-2}	-9.66×10^{-4} (9.27×10^{-2})	4.66	-1.39×10^1	5.74×10^{-2}
Nitrate	-7.07×10^{-4} (4.37×10^{-1})	6.18×10^1	-6.35×10^1	1.99×10^{-2}	-7.11×10^{-3} (1.27)	8.17×10^1	-1.82×10^2	5.70×10^{-2}	-7.79×10^{-3} (1.29)	7.55×10^1	-1.76×10^2	5.76×10^{-2}
Silicate	3.78×10^{-3} (2.24)	5.49×10^2	-5.09×10^2	4.39×10^{-2}	-1.22×10^{-1} (4.06)	3.11×10^2	-1.11×10^3	7.76×10^{-2}	-1.16×10^{-1} (4.03)	1.16×10^2	-5.61×10^2	7.77×10^{-2}
Oxygen	-4.47×10^{-2} (3.20)	4.28×10^2	-5.78×10^2	1.58×10^{-2}	2.33×10^{-1} (1.27×10^1)	1.33×10^3	-9.02×10^2	6.37×10^{-2}	1.87×10^{-1} (1.23×10^1)	9.69×10^2	-9.02×10^2	6.19×10^{-2}

Table B2. Mean (standard deviation), maximum, minimum, and normalized RMSE (RMSE_n) for differences between MATLAB and Python NNs, ESPer_NN and measured values, and PyESPer_NN and measured values for TA, DIC, pH_T , phosphate, nitrate, and oxygen estimates (all units, except for pH_T , are $\mu\text{mol kg}^{-1}$) for all equations combined from the entire GLODAPv2.2022 dataset (w ; where necessary input data were available, $n = 1\,381\,248$).

	Python-MATLAB				MATLAB-Measured				Python-Measured			
	Mean	Max	Min	RMSE_n	Mean	Max	Min	RMSE_n	Mean	Max	Min	RMSE_n
TA	-6.35×10^{-12} (6.24×10^{-9})	6.00×10^{-6}	-9.00×10^{-6}	2.69×10^{-12}	4.99×10^{-1} (1.46×10^1)	2.12×10^3	-2.24×10^3	6.30×10^{-3}	4.99×10^{-1} (1.46×10^1)	2.12×10^3	-2.24×10^3	6.30×10^{-3}
DIC	-3.24×10^{-3} (1.13×10^{-1})	2.88	-4.68	5.19×10^{-5}	-4.82×10^{-1} (1.75×10^1)	1.97×10^3	-2.22×10^3	8.01×10^{-3}	-4.82×10^{-1} (1.75×10^1)	1.97×10^3	-2.22×10^3	8.01×10^{-3}
pH_T	6.08×10^{-6} (3.58×10^{-4})	1.21×10^{-2}	-2.03×10^{-2}	4.52×10^{-5}	-3.01×10^{-3} (3.54×10^{-3})	2.53	-5.74	4.49×10^{-3}	-3.00×10^{-3} (3.54×10^{-2})	2.53	-5.74	4.49×10^{-3}
Phosphate	6.32×10^{-14} (2.08×10^{-10})	6.39×10^{-7}	-1.25×10^{-7}	1.31×10^{-10}	-5.84×10^{-4} (8.25×10^{-2})	1.14×10^1	-6.02	5.06×10^{-2}	-5.84×10^{-4} (8.25×10^{-2})	1.14×10^1	-6.02	5.06×10^{-2}
Nitrate	5.12×10^{-13} (3.06×10^{-9})	1.17×10^{-5}	-2.28×10^{-6}	1.39×10^{-10}	-1.07×10^{-2} (1.17)	1.97×10^2	-1.45×10^2	5.06×10^{-2}	-1.07×10^{-2} (1.17)	1.97×10^2	-1.45×10^2	5.06×10^{-2}
Silicate	-2.35×10^{-13} (2.55×10^{-9})	7.46×10^{-6}	-2.97×10^{-6}	4.96×10^{-11}	-2.37×10^{-2} (3.71)	6.25×10^2	-7.32×10^2	7.06×10^{-2}	-2.37×10^{-2} (3.71)	6.25×10^2	-7.32×10^2	7.06×10^{-2}
Oxygen	-4.65×10^{-13} (2.15×10^{-10})	1.00×10^{-9}	-1.00×10^{-7}	1.06×10^{-12}	-3.46×10^{-3} (1.13×10^1)	7.12×10^2	-1.22×10^3	5.65×10^{-2}	-3.46×10^{-2} (1.13×10^1)	7.12×10^2	-1.22×10^3	5.65×10^{-2}

Appendix C: Example of mapped DIC estimates from PyESPER and ESPER

Surface ocean DIC estimates from PyESPER_LIR and ESPER_LIR were applied to the Roemmich and Gilson climatology (Roemmich and Gilson, 2009). Differences in the surface ocean DIC between the two coding languages (c) illustrate the need to avoid using PyESPER_LIR for DIC in the surface ocean when comparing to MATLAB ESPER_LIR.

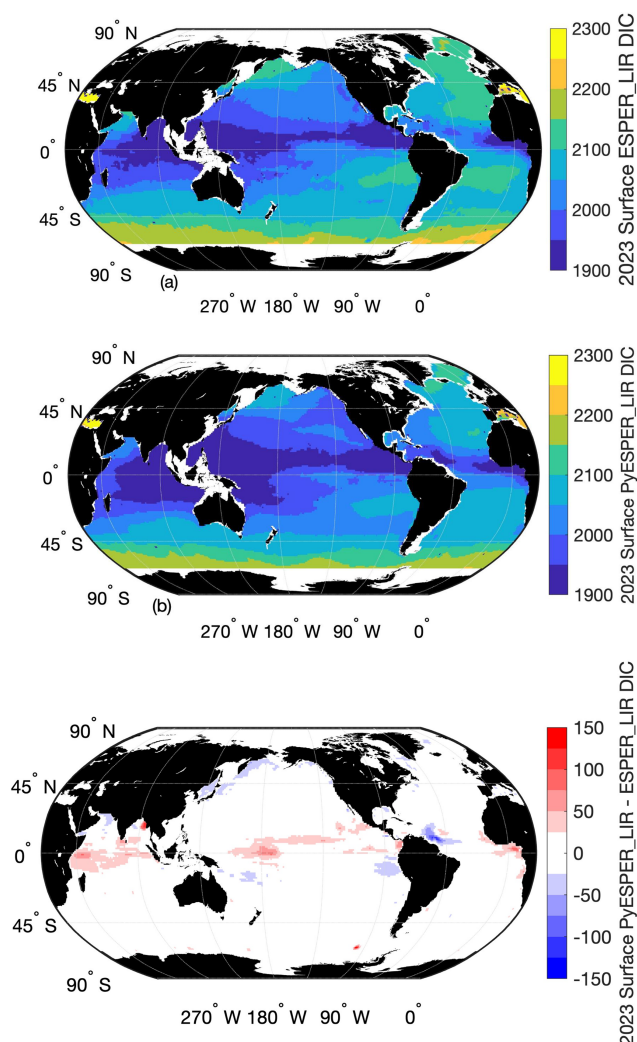


Figure C1. Maps of 2023 mean annual surface estimates of MATLAB ESPER_LIR DIC (a), Python PyESPER_LIR DIC (b), and PyESPER_LIR–ESPER_LIR DIC (c; units are $\mu\text{mol kg}^{-1}$) from the application of ESPERs to the Roemmich–Gilson Argo (Argo, 2000) climatology (Roemmich and Gilson, 2009).

Appendix D: Comparison of interpolation and extrapolation values between MATLAB and Python

MATLAB ESPER_LIRs avoid extrapolation by addition of a false set of data points at very far distances from the grid. However, when this method was implemented in Python, significant errors were introduced due to the differences in triangulation (which were both valid) between the coding languages. Therefore, it was necessary to find another means of calculating extrapolations in PyESPER_LIRs that was more similar to those of ESPER_LIRs. We did this by producing a larger grid in MATLAB and reading that into Python. A simple demonstration of the errors introduced by this method is described below.

For this comparison, we imagine a hypothetical cube, with x , y , and z coordinates, upon which we wish to provide estimates for a fourth variable (p) via both interpolation and extrapolation (Fig. D1a). We have created a random dataset of points and values within this cube for these demonstration purposes. We then followed the same procedure as in the PyESPER data product creation, whereby we extended this grid in three-dimensional space and used MATLAB scatteredInterpolant extrapolations to estimate values on the expanded grid (Fig. D1b). This method conducts a Delaunay triangulation and then uses both linear interpolation and extrapolation to estimate values. These extrapolated values were then used for interpolation only within Python using SciPy's Delaunay and LinearNDInterpolator functions, which produced more consistent results than interpolation and extrapolation within Python.

When interpolations within Python were compared to locations on the hypothetical grid where interpolations occurred in MATLAB also, the results were more similar than those where the grid was extrapolated within MATLAB. This is because different, but equally valid, mathematics are used to interpolate and extrapolate. Namely, a triangulation is used as the basis for interpolations, whereas extrapolations are based on boundary gradients. Despite these differences, the results were still more similar with this method between the two coding languages than when extrapolations were done in both Python and MATLAB.

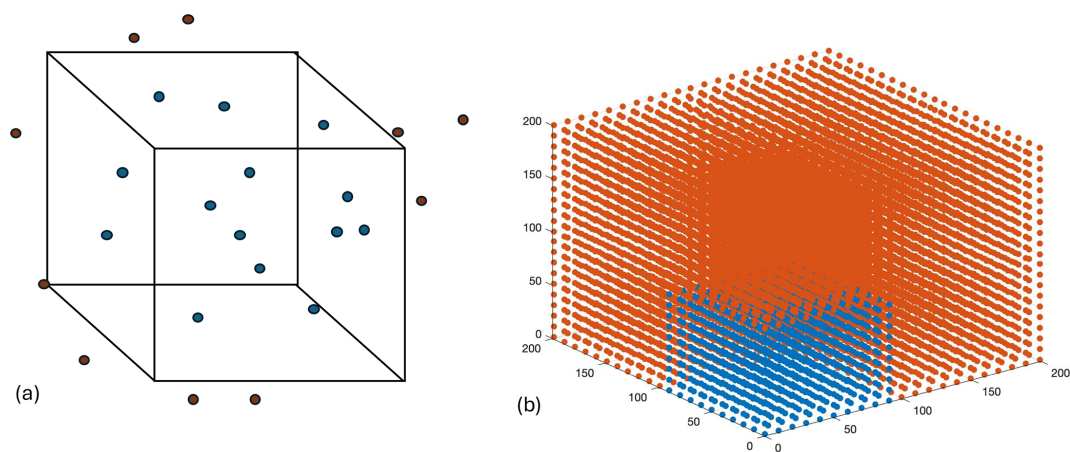


Figure D1. Hypothetical “grid” whereby estimates (p) interpolated within the grid are shown in blue and extrapolations are shown in red (a). Grid created for demonstration purposes, with interpolated values in blue and areas where we extrapolated values in red (b).

Table D1. Comparison of differences between MATLAB interpolations and extrapolations and Python results (all interpolations).

	MATLAB interpolation– Python interpolation	MATLAB extrapolation– Python interpolation
Mean	0.0004	−0.6693
Standard Deviation	0.9559	5.2088
Max	2.2582	13.3083
Min	−2.4593	−15.6633

Code availability. PyESPERv1.0.0, affiliated files, and analyses files are available through LMD’s GitHub page (<https://github.com/LarissaMDias>, last access: 29 July 2025) and archived through Zenodo (<https://doi.org/10.5281/zenodo.15929902>, LarissaMDias and Humphreys, 2025). Updates to PyESPERv1.0.0 will also be published through LMD’s GitHub page and archived through Zenodo. ESPERs (Carter, 2021) and the original associated files used in the creation of PyESPERv1.0.0 are available at BRC’s GitHub page at <https://github.com/BRCScienceProducts> (Carter, 2021). The input data used for comparisons are available through the GLODAP website (<https://glodap.info>, last access: 29 July 2025; Lauvset et al., 2022).

Data availability. The data used for the reconstruction and estimate comparisons are available through GLODAP (<https://glodap.info>, last access: 29 July 2025; see Lauvset et al., 2022, Olsen et al., 2020). The temperature and salinity gridded climatology created by Roemmich and Gilson (2009) was created with data from the Argo Program.

Author contributions. LMD was primarily responsible for Python data product development, validation, formal analysis, investigation, data curation, writing, and visualization. BRC was primarily responsible for project conceptualization, MATLAB data product development, supervision, project administration, providing

resources, funding acquisition, and editing. Methods were devised by both LMD and BRC.

Competing interests. The contact author has declared that neither of the authors has any competing interests.

Disclaimer. Publisher’s note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors.

Acknowledgements. The University of Washington Cooperative Institute for Climate, Ocean, and Ecosystem Studies (CICOES) has assigned CICOES Publication Contribution Number 2024-1382. The National Oceanic and Atmospheric Administration (NOAA) Pacific Marine Environmental Laboratory has assigned PMEL Contribution Number 5707. The data used for DIC data products were collected and made freely available by the International Argo Program and the national programs that contribute to it (<http://www.argo.ucsd.edu>, last access: 29 September 2025). The Argo Program is part of the Global Ocean Observing System. BRC and LMD also

would like to sincerely thank Matthew Humphreys, who served not only as a reviewer but also provided careful editing and help with packaging the PyESPER code. BRC and LMD also thank Daniel Sandborn, who provided useful Python coding tips.

Financial support. This research has been supported by the Global Ocean Monitoring and Observing Program (grant no. NA21OAR4310251).

Review statement. This paper was edited by Andrew Yool and reviewed by Matthew P. Humphreys and three anonymous referees.

References

- Álvarez, M., Velo, A., Tanhua, T., Key, R., and Van Heuven, S.: Carbon, tracer and ancillary data in the MEDsea, CARIMED: An internally consistent data product for the Mediterranean Sea, N43, Rapp. Comm. int. Mer Médit., 42, 8, 2019.
- Argo: Argo float data and metadata from GLOBAL Data Assembly Centre (Argo GDAC), SEANOE [data set], <https://doi.org/10.17882/42182>, 2000.
- Bittig, H. C., Steinhoff, T., Claustre, H., Fiedler, B., Williams, N. L., Sauzède, R., Körtzinger, A., and Gattuso, J.-P.: An alternative to static climatologies: Robust estimation of open ocean CO₂ variables and nutrient concentrations from T, S, and O₂ data using Bayesian neural networks, *Frontiers in Marine Science*, 5, 1–29, <https://doi.org/10.3389/fmars.2018.00328>, 2018.
- Bittig, H. C., Maurer, T. L., Plant, J. N., Wong, A. P., Schmechtig, C., Claustre, H., Trull, T. W., Bhaskar, T. V. S. U., Boss, E., Dall'Olmo, G., Organelli, E., Poteau, A., Johnson, K. S., Hanstein, C., Leymarie, E., Le Reste, S. L., Riser, S. C., Rupan, A. R., Taillandier, V., Thierry, V., and Xing, X.: A BGC-Argo guide: Planning, deployment, data handling and usage, *Frontiers in Marine Science*, 6, 502, <https://doi.org/10.3389/fmars.2019.00502>, 2019.
- Carter, B. R.: Empirical seawater property estimation routines, GitHub [code], <https://github.com/BRCScienceProducts> (last access: 30 June 2025), 2021.
- Carter, B. R., Williams, N. L., Gray, A. R., and Feely, R. A.: Locally interpolated alkalinity regression for global alkalinity estimation, *Limnol. Oceanogr.-Meth.*, 14, 268–277, <https://doi.org/10.1002/lom3.10087>, 2016.
- Carter, B. R., Feely, R. A., Williams, N. L., Dickson, A. G., Fong, M. B., and Takeshita, Y.: Updated methods for global locally interpolated estimation of alkalinity, pH, and nitrate, *Limnol. Oceanogr.-Meth.*, 16, 119–131, <https://doi.org/10.1002/lom3.10232>, 2018.
- Carter, B. R., Bittig, H. C., Fassbender, A. J., Sharp, J. D., Takeshita, Y., Xu, Y., Álvarez, M., Wanninkhof, R., Feely, R. A., and Barbero, L.: New and updated global empirical seawater property estimation routines, *Limnology and Ocean Methods*, 19, 785–809, <https://doi.org/10.1002/lom3.10461>, 2021.
- Carter, B. R., Schwinger, J., Sonnerup, R., Fassbender, A. J., Sharp, J. D., Dias, L. M., and Sandborn, D. E.: Tracer-based Rapid Anthropogenic Carbon Estimation (TRACE), *Earth Syst. Sci. Data*, 17, 3073–3088, <https://doi.org/10.5194/essd-17-3073-2025>, 2025.
- Dias, L. M. and Carter, B.: PyESPER: A Python version of Empirical Seawater Property Estimation Routines (ESPERs) (version 0), Zenodo [code], <https://doi.org/10.5281/ZENODO.5512697>, 2025.
- Gammon, R. H., Cline, J., and Wisegarver, D.: Chlorofluoromethanes in the northeast Pacific Ocean: Measured vertical distributions and application as transient tracers of upper ocean mixing, *J. Geophys. Res.-Oceans*, 87, 9441–9454, <https://doi.org/10.1029/JC087iC12p09441>, 1982.
- Gruber, N., Clement, D., Carter, B. R., Feely, R. A., van Heuven, S., Hoppema, M., Ishii, M., Key, R. M., Kozyr, A., Lauvset, S. K., Monaco, C. L., Mathis, J. T., Murata, A., Olsen, A., Perez, F. F., Sabine, C. L., Tanhua, T., and Wanninkhof, R.: The oceanic sink for anthropogenic CO₂ from 1994–2007, *Science*, 363, 1193–1199, <https://doi.org/10.1126/science.aau5153>, 2019.
- Hauck, J., Hauck, J., Nissen, C., Landschützer, P., Rödenbeck, C., and Bushinsky, S.: Sparse observations induce large biases in estimates of the global ocean CO₂ sink: An ocean model subsampling experiment, *Philos. T. R. Soc. A*, 381, 20220063, <https://doi.org/10.1098/rsta.2022.0063>, 2023.
- Jiang, L.-Q., Dunne, J., Carter, B. R., Tjiputra, J. F., Terhaar, J., Sharp, J. D., Olsen, A., and Alin, S.: Global surface ocean acidification indicators from 1750–2100, *J. Adv. Model. Earth Sy.*, 15, 1–23, <https://doi.org/10.1029/2022MS003563>, 2020.
- Jiang, L.-Q., Feely, R. A., Wanninkhof, R., Greeley, D., Barbero, L., Alin, S., Carter, B. R., Pierrot, D., Featherstone, C., Hooper, J., Melrose, C., Monacchi, N., Sharp, J. D., Shellito, S., Xu, Y.-Y., Kozyr, A., Byrne, R. H., Cai, W.-J., Cross, J., Johnson, G. C., Hales, B., Langdon, C., Mathis, J., Salisbury, J., and Townsend, D. W.: Coastal Ocean Data Analysis Product in North America (CODAP-NA) – an internally consistent data product for discrete inorganic carbon, oxygen, and nutrients on the North American ocean margins, *Earth Syst. Sci. Data*, 13, 2777–2799, <https://doi.org/10.5194/essd-13-2777-2021>, 2021.
- Keppler, L., Landschützer, P., Gruber, N., Lauvset, S. K., and Stemmler, I.: Seasonal carbon dynamics in the near-global ocean, *Global Biogeochem. Cy.*, 34, e2020GB006571, <https://doi.org/10.1029/2020GB006571>, 2020.
- LarissaMDias and Humphreys, M.: LarissaMDias/PyESPER: PyESPER v1.0.0 (1.0.0), Zenodo [code], <https://doi.org/10.5281/zenodo.15929902>, 2025.
- Lauvset, S. K., Key, R. M., Olsen, A., van Heuven, S., Velo, A., Lin, X., Schirnack, C., Kozyr, A., Tanhua, T., Hoppema, M., Jutterström, S., Steinfeldt, R., Jeansson, E., Ishii, M., Perez, F. F., Suzuki, T., and Watelet, S.: A new global interior ocean mapped climatology: the 1° × 1° GLODAP version 2, *Earth Syst. Sci. Data*, 8, 325–340, <https://doi.org/10.5194/essd-8-325-2016>, 2016.
- Lauvset, S. K., Lange, N., Tanhua, T., Bittig, H. C., Olsen, A., Kozyr, A., Alin, S., Álvarez, M., Azetsu-Scott, K., Barbero, L., Becker, S., Brown, P. J., Carter, B. R., da Cunha, L. C., Feely, R. A., Hoppema, M., Humphreys, M. P., Ishii, M., Jeansson, E., Jiang, L.-Q., Jones, S. D., Lo Monaco, C., Murata, A., Müller, J. D., Pérez, F. F., Pfeil, B., Schirnack, C., Steinfeldt, R., Suzuki, T., Tilbrook, B., Ulfssbo, A., Velo, A., Woosley, R. J., and Key, R. M.: GLODAPv2.2022: the latest version of the global interior ocean biogeochemical data product, *Earth Syst.*

- Sci. Data, 14, 5543–5572, <https://doi.org/10.5194/essd-14-5543-2022>, 2022.
- Maurer, T. L., Plant, J. N., and Johnson, K. S.: Delayed-mode quality control of oxygen, nitrate and pH data on SOCCOM biogeochemical profiling floats, *Frontiers in Marine Science*, 8, 683207, <https://doi.org/10.3389/fmars.2021.683207>, 2021.
- Olsen, A., Lange, N., Key, R. M., Tanhua, T., Bittig, H. C., Kozyr, A., Álvarez, M., Azetsu-Scott, K., Becker, S., Brown, P. J., Carter, B. R., Cotrim da Cunha, L., Feely, R. A., van Heuven, S., Hoppema, M., Ishii, M., Jeansson, E., Jutterström, S., Landa, C. S., Lauvset, S. K., Michaelis, P., Murata, A., Pérez, F. F., Pfeil, B., Schirnack, C., Steinfeldt, R., Suzuki, T., Tilbrook, B., Velo, A., Wanninkhof, R., and Woosley, R. J.: An updated version of the global interior ocean biogeochemical data product, GLODAPv2.2020, *Earth Syst. Sci. Data*, 12, 3653–3678, <https://doi.org/10.5194/essd-12-3653-2020>, 2020.
- Roemmich, D. and Gilson, J.: The 2004–2008 mean and annual cycle of temperature, salinity, and steric height in the global ocean from the Argo Program, *Prog. Oceanogr.*, 82, 81–100, <https://doi.org/10.1016/j.pocean.2009.03.004>, 2009.
- Sharp, J. D., Fassbender, A. J., Carter, B. R., Johnson, G. C., Schultz, C., and Dunne, J. P.: GOBAI–O₂: temporally and spatially resolved fields of ocean interior dissolved oxygen over nearly 2 decades, *Earth Syst. Sci. Data*, 15, 4481–4518, <https://doi.org/10.5194/essd-15-4481-2023>, 2023.
- Sharp, J. D., Jiang, L.-Q., Carter, B. R., Lavin, P. D., Yoo, H., and Cross, S. L.: A mapped dataset of surface ocean acidification indicators in large marine ecosystems of the United States, *Sci. Data*, 11, 715, <https://doi.org/10.1038/s41597-024-03530-7>, 2024.
- Tanhua, T., Körtzinger, A., Friis, K., Waugh, D. W., and Wallace, D. W. R.: An estimate of anthropogenic CO₂ inventory from decadal changes in oceanic carbon content, *P. Natl. Acad. Sci. USA*, 104, 3037–3042, <https://doi.org/10.1073/pnas.0606574104>, 2007.
- Tanhua, T., Lauvset, S. K., Lange, N., Olsen, A., Álvarez, M., Diggs, S., Bittig, H. C., Brown, P. J., Carter, B. R., da Cunha, L. C., Feely, R. A., Hoppema, M., Ishii, M., Jeansson, E., Kozyr, A., Murata, A., Pérez, F. F., Pfeil, B., Schirnack, C., Steinfeldt, R., Telszewski, M., Tilbrook, B., Velo, A., Wanninkhof, R., Burger, E., O'Brien, K., and Key, R. M.: A vision for FAIR ocean data products, *Communications Earth and Environment*, 2, 19–22, <https://doi.org/10.1038/s43247-021-00209-4>, 2021.
- Velo, A., Pérez, F. F., Tanhua, T., Gilcoto, M., Ríos, A. F., and Key, R. M.: Total alkalinity estimation using MLR and neural network techniques, *J. Marine Syst.*, 111–112, 11–18, <https://doi.org/10.1016/j.jmarsys.2012.09.002>, 2013.
- Virtanen, P., Gommers, R., Olifant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, I., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., Vijaykumar, A., Bardelli, A. P., Rothberg, A., Hilboll, A., Kloeckner, A., Scopatz, A., Lee, A., Rokem, A., Woods, C. N., Fulton, C., Masson, C., Haggstrom, C., Fitzgerald, C., Nicholson, D. A., Hagen, D. R., Pasechnik, D. V., Olivetti, E., Martin, E., Wieser, E., Silva, F., Lenders, F., Wilhelm, F., Young, G., Price, G., Ingold, G.-L., Allen, G. E., Lee, G. R., Audren, H., Probst, I., Dietrich, J. P., Silterra, J., Webber, J. T., Slavic, J., Nothman, J., Buchner, J., Kulick, J., Schonberger, J. L., Crodoso, J. V. M., Reimer, J., Harrington, J., Rodriguez, J. L. C., Nunez-Iglesias, J., Kaczynski, J., Tritz, K., Thoma, M., Newville, M., Kummerer, M., Bolingbroke, M., Tartre, M., Pak, M., Smith, N. J., Nowaczyk, N., Shebanov, N., Pavlyk, O., Brodtkorb, P. A., Lee, P., McGibbon, R. T., Feldbauer, R., Lewis, S., Tygier, S., Sievert, S., Vigna, S., Peterson, S., More, S., Pudlik, T., Oshima, T., Pingel, T. J., Robitaille, T. P., Spura, T., Jones, T. R., Cera, T., Leslie, T., Zito, T., Krauss, T., Upadhyay, U., Halchenko, Y. O., and Vazquez-Baeza, Y.: SciPy 1.0: fundamental algorithms for scientific computing in Python, *Nat. Methods*, 17, 261–272, <https://doi.org/10.1038/s41592-019-0686-2>, 2020.
- Waugh, D. W., Hall, T. M., McNeil, B. I., Key, R., and Matear, R. J.: Anthropogenic CO₂ in the oceans estimated using transit time distributions, *Tellus B*, 58, 376–389, <https://doi.org/10.1111/j.1600-0889.2006.00222.x>, 2006.
- Zeebe, R. E. and Wolf-Gladrow, D. A.: CO₂ in Seawater: Equilibrium, Kinetics, Isotopes, Elsevier Science B. V., Amsterdam, the Netherlands, ISBN 9780444505798, 2001.