PYOMPA version 0.3: Technical Note

Avanti Shrikumar[1,1,1,1], Rian Lawrence[1,1,1,1], and Karen L Casciotti[1,1,1,1]

[1]Stanford University

November 30, 2022

Abstract

This document is intended as a technical note for pyompa (https://github.com/nitrogenlab/pyompa), a python package for conducting water mass mixing analysis (also known as "Optimum Multiparameter Analysis", "OMP analysis" or "OMPA" in the literature). It is being made available in advance of a formal publication as an accompaniment to an upcoming paper by Lawrence et al. on the water mass analysis of the GP15 transect, and also so that it may be used as a reference for individuals who wish to use PYOMPA in their research today. PYOMPA contains several significant improvements over previously-published methods for conducting OMP analysis. These include a hard constraint on mass conservation, flexible definitions for both end-member composition and nutrient exchange ratios, a smooth penalty-based approach to encode prior knowledge of end-member distributions, support for reporting ambiguity when the solution is underdetermined, and a way to leverage knowledge from another model (such as an Ocean Circulation Inverse Model or OCIM) to select the best PYOMPA solution. This document also contains a description of additional analyses, such as archetype analysis for selecting end-member subtypes and an automated way to define the thermocline boundary, as they are often useful in conjunction with water mass mixing analysis. The focus of this document is on the mathematical formulations. Please refer to the github README or contact the authors for more information.

Posted on 30 Nov 2022 — CC-BY 4.0 — https://doi.org/10.1002/essoar.10507053/v4 — This a preprint and has not been peer reviewed. Data may be preliminary.

PYOMPA TECHNICAL NOTE

A PREPRINT

Avanti Shrikumar
Department of Earth System Science
Stanford Data Science Institute
Stanford University

Rian Lawrence
Department of Earth System Science
Stanford University

Karen L. Casciotti
Department of Earth System Science
Stanford University

February 4, 2022

ABSTRACT

This document is intended as a technical note for PYOMPA (`https://github.com/nitrogenlab/pyompa`), a python package for conducting water mass mixing analysis (also known as "Optimum Multiparameter Analysis", "OMP analysis" or "OMPA" in the literature). It is being made available in advance of a formal publication as an accompaniment to an upcoming paper by Lawrence et al. on the water mass analysis of the GP15 transect, and also so that it may be used as a reference for individuals who wish to use PYOMPA in their research today. PYOMPA contains several significant improvements over previously-published methods for conducting OMP analysis. These include a hard constraint on mass conservation, flexible definitions for both end-member composition and nutrient exchange ratios, a smooth penalty-based approach to encode prior knowledge of end-member distributions, support for reporting ambiguity when the solution is underdetermined, and a way to leverage knowledge from another model (such as an Ocean Circulation Inverse Model or OCIM) to select the best PYOMPA solution. This document also contains a description of additional analyses, such as archetype analysis for selecting end-member subtypes and an automated way to define the thermocline boundary, as they are often useful in conjunction with water mass mixing analysis. The focus of this document is on the mathematical formulations. Please refer to the github README or contact the authors for more information.

Keywords Optimum Multi-Parameter Analysis · water mass mixing analysis · OMP · OMPA · PYOMPA · Convex Combination

1 Background on Water Mass Mixing Analysis and the OMP method

1.1 Overview of Water Mass Mixing Analysis

Suppose we collect a sample on an oceanographic cruise and measure various properties (or "tracers") of the sample that could tell us about the biogeochemistry. Before we can draw conclusion about the biogeochemistry, it is important to account for the component of the property values that is caused not by the biogeochemistry, but rather by ocean circulation. One method of accounting for the ocean circulation is known as *water mass mixing analysis*. In water mass mixing analysis, each measured sample is thought of as a mixture of canonical "water types" or "end-members". Because the chemical composition of the "end-members" is known, the expected chemical composition of the sample due to mixing of end-members can be computed. This computed value can then be compared to the in-situ measured value to check for potentially interesting biogeochemistry.

We can illustrate the basic approach with an example. Say we are interested in a property X that is informative about the biogeochemistry. Now say we have a sample s for which we have measured three properties: the temperature (s_θ),

the salinity (s_S), and property X (s_X). If we can assume that X is conservative in the absence of any biogeochemical processes, we can calculated the value of property X produced solely due to mixing as follows: first, we identify existing water types for which we know the temperature, salinity and property X, and which we expect our sample to have originated from; these are the 'end-members' – let us pretend, in our case, that we have 3 end-members. Second, we use the salinity and temperature to estimate the fraction of each end-member in the sample; let x^1, x^2 and x^3 denote the fractions for end-members 1, 2 and 3. Third, we we calculate the expected amount of X in s due to mixing as $x^1 e^1_X + x^2 e^2_X + x^3 e^3_X$, where e^i_X represents the value of property X in end-member i.

A standard method for performing water mass mixing analysis is "Optimum Multi-Parameter analysis", "OMP analysis" or "OMPA" Tomczak [1981], Thompson and Edwards [1981]. In this section, we will provide a description of the existing OMP implementation provided by the original OMP authors at `https://omp.geomar.de`.

1.2 "Classical" OMP Analysis

Classical OMP analysis [Tomczak, 1981, Thompson and Edwards, 1981] attempts to explain the measured properties (or "tracers") of a given sample as a mixture of several end-members (or "water types"), where it is assumed that the property values are conserved by mixing. Formally, let e^i_p denote the value of property p for end-member i (e.g. we will use e^1_T to denote the value of the temperature parameter $\boldsymbol{T}$ for end-member 1), let s^j_p denote the value of property p in measured sample j, and let x^i_j denote the estimated fraction of end-member i in sample j. In the *ideal* case where the properties of end-members are exactly known and the properties of the sample can be explained *perfectly*, we would have $\sum_i e^i_p x^i_j = s^j_p$. However, in practice there may be uncertainty both in the end-member properties e^i_p and in the measured sample properties s^j_p. In such a situation, our best estimates of the end-member fractions x^i_j may not be able to perfectly explain the sample properties s^j_p, resulting in a *residual*. Let us use ϵ^j_p to denote the residual in explaining property p for sample j. If we modify our earlier equation to include residuals, we have:

$$\sum_i e^i_p x^i_j = s^j_p + \epsilon^j_p \tag{1}$$

How do we solve for the values of x^i_j? Classical OMP analysis does this by minimizing the residuals in a least-squares objective - however, the residuals that are minimized are *not* the same as the residuals ϵ^j_p above (which are presented in terms of the original parameter values). Rather, classical OMP analysis first normalizes the different parameters prior to calculating the residuals, and then further reweights the residuals prior to doing the least-squares minimization. To explain the normalization procedure used, we will introduce three new symbols: μ_p to denote a mean value for parameter p across the end-members, σ_p to denote the population standard deviation of parameter p across the end-members. Letting M represents the total number of end-members, the quantities μ_p and σ_p are calculated as follows:

$$\mu_p := \frac{1}{M} \sum_i e^i_p \tag{2}$$

$$\sigma_p := \sqrt{\frac{1}{M-1} \sum_i (e^i_p - \mu_p)^2} \tag{3}$$

After subtracting the mean and normalizing by the variance, the normalized end-member parameter values become $e^{i,\text{norm}}_p := (e^i_p - \mu_p)/\sigma_p$ and the normalized sample property values become $s^{j,\text{norm}}_p := (s^j_p - \mu_p)/\sigma_p$. As mentioned, the residuals used in the least-squares objective function of classical OMP analysis are in terms of the normalized property values, and are additionally scaled up by a user-specified weight that we will denote with W^{user}_p. Putting it all together, we have:

$$\begin{aligned}
\epsilon^{j,\text{classic}}_p :=& W^{\text{user}}_p \left(\left(\sum_i e^{i,\text{norm}}_p x^i_j \right) - s^{j,\text{norm}}_p \right) \\
=& W^{\text{user}}_p \left(\left(\sum_i \frac{(e^i_p - \mu_p)}{\sigma_p} x^i_j \right) - \frac{(s^j_p - \mu_p)}{\sigma_p} \right) \\
=& \frac{W^{\text{user}}_p}{\sigma_p} \left(\left(\sum_i (e^i_p - \mu_p) x^i_j \right) - (s^j_p - \mu_p) \right)
\end{aligned} \tag{4}$$

Written this way, we can see that the impact of dividing by the standard deviation is effectively to rescale the user-provided weights W_p^{user} by a factor of $1/\sigma_p$.

We are almost ready to describe the objective used in classical OMP analysis, but we must first discuss one last detail pertaining to the quantities x_j^i. Recall that x_j^i represents the fraction of end-member i to sample j. In order for x_j^i to have a valid interpretation as a fractional contribution, two things must be true: the fraction must be non-negative, and the fractions must sum to 1. Classical OMP analysis satisfies non-negativity by performing *constrained* least-squares optimization on – that is, it only searches for solutions where $x_j^i \geq 0$. Curiously, it does not similarly apply a strict constraint to only search for solutions where the end-member fractions exactly sum to 1; instead, the summation of x_j^i to 1 is only *approximately* achieved by minimizing a *mass conservation residual*. Let us use $\epsilon_M^{j,\text{classic}}$ to denote this mass-conservation residual, defined as:

$$\epsilon_M^{j,\text{classic}} := W_M^{\text{user}} \left(\left(\sum_i x_j^i \right) - 1 \right) \tag{5}$$

We are now ready to fully specify the least-squares optimization procedure used by classical OMP analysis to find solutions. We will first write the *cost function* (i.e. the quantity that is minimized by the least-squares solver; also known as an "objective function") used in classical OMP analysis as:

$$C_j^{\text{classic}} = \left(\epsilon_M^{j,\text{classic}} \right)^2 + \sum_p \left(\epsilon_p^{j,\text{classic}} \right)^2 \tag{6}$$

The cost function alone does not explicitly say what constraints are used. In order to succinctly describe the constraints, we will introduce the use of bold symbols to denote vector quantities - for example, we will use $\boldsymbol{x}_j = (x_j^1, x_j^2, ...)$ to denote a vector of end-member fractions. We will also use $\boldsymbol{x}_j^{\text{classic}}$ to denote the solution obtained by classical OMP analysis for the vector of end-member fractions in sample j. If we adopt the notation "$\text{argmin}_{\boldsymbol{x}_j}[...]$" to denote the value of $\boldsymbol{x}_j$ that minimizes the quantity in brackets, $[...]$, we can write the full optimization objective for classical OMP analysis as:

$$\begin{aligned} \boldsymbol{x}_j^{\text{classic}} := \underset{\boldsymbol{x_j}}{\text{argmin}} & \left[\left(\epsilon_M^{j,\text{classic}} \right)^2 + \sum_p \left(\epsilon_p^{j,\text{classic}} \right)^2 \right] \\ & \text{subject to the constraints:} \\ & x_j^i \geq 0 \text{ for all } i \end{aligned} \tag{7}$$

1.2.1 Specifying the user-defined weights

The objective above depends on user-defined weights (denoted as W_p^{user}), and a natural question that arises is how these weights should be specified. As evidenced by Eqn. 4, these weights play a role in scaling the magnitude of the residuals associated with particular parameters, and consequently determine how heavily a particular parameter will feature in the objective function. This implies that a higher weight should be given to those parameters that are likely to have small residuals in a correct solution. There are two main considerations that go into whether a correct solution is likely to have low residuals for a particular parameter: the first is whether the parameter has been measured precisely, and the second is whether the end-member definitions for the parameter are sufficiently accurate. For example, if one is very confident in the sample measurement of temperature, and is very confident that the correct definitions of temperature have been provided in the end-member file, it would stand to reason that temperature should have a small residual in a correct OMP solution (assuming, of course, that all the end-members necessary to explain the sample have been provided in the first place). Ultimately, the choice of weighting is subjective; however, we recommend performing a sensitivity analysis [Peters et al., 2018] to ensure that the ideal solution does not change dramatically with slight changes to the parameter weights.

Note that the absolute magnitudes of the parameter weights don't matter; only the relative magnitudes matter. However, avoid setting weights that have very large magnitudes as this can cause numerical instabilities (specifically, it can cause the system of equations to have a poor condition number).

1.3 "Extended" OMP analysis

Classical OMP analysis relies on the assumption that all the parameters used in the analysis are conserved by mixing. While this generally holds true for parameters such as temperature or salinity, parameters such as oxygen, phosphate, nitrate and silicate can be subject to biogeochemical processes that can incorporate them into or release them from organic matter. Thus, if viewed solely in terms of their concentrations in their dissolved inorganic form, these parameters will not appear to be conserved.

How can we account for such biogeochemical processes? In "extended" OMP analysis [Karstensen and Tomczak, 1998], this is handled by assuming that these nutrients are exchanged with oxygen in a fixed ratio (the "Redfield ratio"). Formally, let us denote the exchange ratio of a parameter p w.r.t. phosphate as $r^p_{\mathrm{PO_4}}$. For example, the default value of $r^{\mathrm{O_2}}_{\mathrm{PO_4}}$ used in the MATLAB OMP implementation is -170 (oxygen has a negative sign because it is consumed when nitrate/phosphate are produced), while the default value of $r^{\mathrm{NO_3}}_{\mathrm{PO_4}}$ is 16. If we use the variable $\Delta^j_{\mathrm{PO_4}}$ to denote the amount of phosphate that is remineralized from organic matter into dissolved inorganic matter in sample j, we can model the effect of remineralization as:

$$\left(\sum_i e^i_p x^i_j\right) + r^p_{\mathrm{PO_4}}\Delta^{\mathrm{PO_4}}_j = s^j_p + \epsilon^j_p \tag{8}$$

The term in blue highlights how the equation above differs the corresponding Eqn. 1 from classical OMP analysis. Note that $r^{\mathrm{PO_4}}_{\mathrm{PO_4}}$ will equal 1. Further, if a particular parameter (like salinity) is modeled as being unaffected by remineralization, $r^p_{\mathrm{PO_4}}$ for that parameter would simply be 0. Based on this formulation, we can write the normalized & weighted residuals used in the optimization objective of extended OMP analysis as:

$$\epsilon^{j,\mathrm{ext}}_p = \frac{W^{\mathrm{user}}_p}{\sigma_p}\left(\left(\sum_i (e^i_p - \mu_p)x^i_j\right) + r^p_{\mathrm{PO_4}}\Delta^{\mathrm{PO_4}}_j - (s^j_p - \mu_p)\right) \tag{9}$$

Once again, the term in blue highlights how the equation above differs from the corresponding Eqn. 4 from Classical OMP analysis.

One important point to note about the MATLAB OMP implementation is that $\Delta^{\mathrm{PO_4}}_j$ is restricted to being positive, meaning that the MATLAB OMP implementation can model only remineralization (or, alternatively, can model only assimilation, if the user were to flip the sign of $r^p_{\mathrm{PO_4}}$ for all the parameters). If we reuse our definition of $\epsilon^{j,\mathrm{classic}}_M$ from Eqn. 5, we can write the optimization objective used in extended OMP analysis as:

$$\begin{aligned}\boldsymbol{x}^{\mathrm{ext}}_j, \Delta^{\mathrm{PO_4}}_j := \underset{\boldsymbol{x}_j, \Delta^{\mathrm{PO_4}}_j}{\operatorname{argmin}} & \left(\epsilon^{j,\mathrm{classic}}_M\right)^2 + \sum_p \left(\epsilon^{j,\mathrm{ext}}_p\right)^2 \\ & \text{subject to the constraints:} \\ & x^i_j \geq 0 \text{ for all } i \text{ and} \\ & \Delta^{\mathrm{PO_4}}_j \geq 0\end{aligned} \tag{10}$$

As a final aside, the current MATLAB OMP implementation appears to report the value of $\Delta^{\mathrm{PO_4}}_j$ in terms of oxygen-equivalent units (that is, it appears to report $-r^{\mathrm{O_2}}_{\mathrm{PO_4}}\Delta^{\mathrm{PO_4}}_j$) to the user in the "biogeo" field of `.mat` file written out in the solution.

1.3.1 Modeling remineralization within "Classical" OMP analysis using variables like PO and NO

One way to account for these biogeochemical processes while remaining within the classical OMP formulation is to instead use the "semiconserved" parameters of PO, NO, etc. To derive the formula for these semiconserved quantities, let us consider the case where we have measurements for two specific remineralized parameters, which we will denote as p_1 and p_2. Referring back to Eqn. 8, we can write:

$$\left(\sum_i e^i_{p_1} x^i_j\right) + r^{p_1}_{\mathrm{PO}_4} \Delta^{\mathrm{PO}_4}_j = s^j_{p_1} + \epsilon^j_{p_1}$$
$$\left(\sum_i e^i_{p_2} x^i_j\right) + r^{p_2}_{\mathrm{PO}_4} \Delta^{\mathrm{PO}_4}_j = s^j_{p_2} + \epsilon^j_{p_2}$$

If we multiply the second equation by $r^{p_1}_{\mathrm{PO}_4}/r^{p_2}_{\mathrm{PO}_4}$ and subtract it from the first equation, we get:

$$\left(\sum_i \left(e^i_{p_1} - \frac{r^{p_1}_{\mathrm{PO}_4}}{r^{p_2}_{\mathrm{PO}_4}} e^i_{p_2}\right) x^i_j\right) = \left(s^j_{p_1} - \frac{r^{p_1}_{\mathrm{PO}_4}}{r^{p_2}_{\mathrm{PO}_4}} s^j_{p_2}\right) + \left(\epsilon^j_{p_1} - \frac{r^{p_1}_{\mathrm{PO}_4}}{r^{p_2}_{\mathrm{PO}_4}} \epsilon^j_{p_2}\right) \tag{11}$$

Note that Eqn. 11 does not involve $\Delta^{\mathrm{PO}_4}_j$. If we define a new variable $p_{1,2}$ as:

$$p_{1,2} := p_1 - \frac{r^{p_1}_{\mathrm{PO}_4}}{r^{p_2}_{\mathrm{PO}_4}} p_2 \tag{12}$$

We can write:

$$\left(\sum_i e^i_{p_{1,2}} x^i_j\right) = s^j_{p_{1,2}} + \epsilon^j_{p_{1,2}} \tag{13}$$

Note that Eqn. 13 has the same form as Eqn. 1, and thus fits perfectly into the paradigm of "Classical" OMP analysis. In the literature, OMP parameters that have the form of Eqn. 12 have been constructed using Oxygen & Phosphate (i.e. $[\mathrm{PO}] = [\mathrm{O}_2] + r^{\mathrm{O}_2}_{\mathrm{PO}_4}[\mathrm{PO}_4]$) and Oxygen & Nitrate (i.e. $[\mathrm{NO}] = [\mathrm{O}_2] + \frac{r^{\mathrm{O}_2}_{\mathrm{PO}_4}}{r^{\mathrm{NO}_3}_{\mathrm{PO}_4}}[\mathrm{NO}_3]$), such as in Peters et al. [2018]. As we have seen here, similar equations can be constructed using any two pairs of variables that are subject to remineralization.

What are the reasons to favor the "classical" OMP formulation over the "extended" OMP formulation? As we noted earlier, the MATLAB implementation of "extended" OMP analysis places the restriction $\Delta^{\mathrm{PO}_4}_j \geq 0$; by constrast, Eqn. 11 makes no such assumption, and is therefore capable of accounting for both assimilation and remineralization. However, a more subtle difference in using the classical formulation is that the solution obtained by minimizing the combined residual $\epsilon^j_{p_{1,2}} = \epsilon^j_{p_1} - \frac{r^{p_1}_{\mathrm{PO}_4}}{r^{p_2}_{\mathrm{PO}_4}} \epsilon^j_{p_2}$ may be different from the solution obtained by minimizing $\epsilon^j_{p_1}$ and $\epsilon^j_{p_2}$ independently - indeed, it is conceivable that, for some solutions, the residuals in p_1 and p_2 might "cancel out" to yield a low residual in $p_{1,2}$; these solutions might be favored under the classical OMP formulation but might correctly be identified as undesirable under the extended formulation. Our recommendation is that the user favor the "extended" formulation where possible; note that the PYOMPA implementation (unlike the MATLAB implementation) does not enforce the restriction that $\Delta^{\mathrm{PO}_4}_j \geq 0$, and is thus capable of modeling both assimilation and remineralization.

2 PYOMPA's OMP formulation

There are currently four key ways in which PYOMPA's formulation differs from the OMP formulation described in Sec. 1. We will discuss each in turn and will finish by presenting the overall optimization objective.

2.1 A hard constraint on mass conservation

2.1.1 Impact of mass conservation residuals in the original OMP formulation

In Sec. 1.2, we noted that the MATLAB OMP implementation does not enforce that the mass conservation residuals be zero. To our knowledge, it has not previously been noted that this failure to exactly satisfy mass conservation is the reason that the MATLAB OMP solution is affected by the mean normalization. In our opinion, a major caveat of having a solution that is sensitive to mean normalization is that even though a particular solution might have small

residuals in terms of the mean-normalized parameter values, the residuals could be very large in terms of the original (non-mean-normalized) parameter values. We will illustrate this first with a mathematical proof, and then with a concrete example.

Let us reproduce the formula for the residuals used in classical OMP analysis (Eqn. 4) below - as before, s_p^j represents the value of parameter p for sample j, e_p^i represents the value of parameter p for end-member i, x_j^i represent the fraction of end-member i in sample j, and μ_p denote the mean value for parameter p. We had:

$$\epsilon_p^{j,\text{classic}} := \frac{W_p^{\text{user}}}{\sigma_p}\left(\left(\sum_i (e_p^i - \mu_p)x_j^i\right) - (s_p^j - \mu_p)\right)$$

Our goal is to rearrange this equation to show the relationship between $\epsilon_p^{j,\text{classic}}$ and ϵ_p, where $\epsilon_p := (\sum_i e_p^i x_j^i) - s_p^j$ (from Eqn. 1) is the residual in the original parameter space. Note that the mass conservation residual in the original parameter space is simply $\epsilon_M = (\sum_i x_i) - 1$. We have:

$$\begin{aligned}
\epsilon_p^{j,\text{classic}} &:= \frac{W_p^{\text{user}}}{\sigma_p}\left(\left(\sum_i (e_p^i - \mu_p)x_j^i\right) - (s_p^j - \mu_p)\right) \\
&= \frac{W_p^{\text{user}}}{\sigma_p}\left(\left(\sum_i e_p^i x_j^i\right) - \left(\sum_i \mu_p x_j^i\right) - (s_p^j - \mu_p)\right) \\
&= \frac{W_p^{\text{user}}}{\sigma_p}\left(\left(\left(\sum_i e_p^i x_j^i\right) - s_p^j\right) - \left(\left(\sum_i \mu_p x_j^i\right) - \mu_p\right)\right) \text{ (Rearranging terms)} \\
&= \frac{W_p^{\text{user}}}{\sigma_p}\left(\left(\left(\sum_i e_p^i x_j^i\right) - s_p^j\right) - \mu_p\left(\left(\sum_i x_j^i\right) - 1\right)\right) \text{ (Factoring out } \mu_p\text{)} \\
&= \frac{W_p^{\text{user}}}{\sigma_p}\left(\epsilon_p - \mu_p\epsilon_M\right) \text{ (Substituting the definitions for } \epsilon_p \text{ and } \epsilon_M\text{)}
\end{aligned}$$

This gives:

$$\epsilon_p^{j,\text{classic}}\left(\frac{\sigma_p}{W_p^{\text{user}}}\right) + \mu_p\epsilon_M = \epsilon_p \tag{14}$$

Let's reflect on the implications of Eqn. 14. Note that even when $\epsilon_p^{j,\text{classic}}$ is relatively small, if $\mu_p\epsilon_M$ is large, ϵ_p will be large. Further note that even if ϵ_M appears to be small, a large value of μ_p can result in a sizeable value for $\mu_p\epsilon_M$. Thus, allowing a residual in the mass conservation makes it possible to have large residuals in the original parameter space even when $\epsilon_p^{j,\text{classic}}$ appears small.

The issue can be illustrated with a simple example. Considered the case of 2-end-member mixing with one measured parameter - salinity (S). Let us imagine that our end-members e^1 and e^2 have salinity values of $e_S^1 = 32$ and $e_S^2 = 34$, and that the mean value for S is $\mu_S = 33$. Let us also consider an observation s with a salinity value of $s_S = 31.5$. After mean-normalization, the salinity values are $\bar{e}_S^1 = -1$, $\bar{e}_S^2 = 1$, and $\bar{s}_S = -1.5$. For added simplicitly, in order to focus on the impact of mean normalization, let us ignore the rescaling by the standard deviation in this example (i.e. let us pretend the standard deviation is 1). If the mass conservation equation is given the same weight as the salinity equation (which is the case according to the default coefficients from the MATLAB OMP package), then the solution that minimizes the sum of the squared residuals (Eqn. 7; this involves both the salinity residual and the mass conservation residual) is $x^1 = 1.25$ and $x^2 = 0$, which corresponds to $\epsilon_M = 0.25$. The residual in terms of mean-normalized salinity here is $\bar{\epsilon}_S = 0.25$ (i.e. it is equal to the mass conservation residual; this is expected if the residuals for mass conservation and salinity are given equal weight), but the residual in terms of the original salinity is $32 * 1.25 - 31.5 = 8.5$ - a very large residual for salinity! For parameters that can have even larger values of μ_p (e.g. parameters such as PO or NO), this discrepancy between the residuals in the mean-normalized parameter space and the residuals in the original parameter space can be even more striking.

2.1.2 PYOMPA's solution: enforcing mass conservation as a hard constraint

The analysis in the previous section illustrates what (to our knowledge) is a previously unappreciated drawback of allowing residuals in the mass conservation equation. It is sometimes argued that it is important to allow residuals in

the mass conservation equation because the analysis may have missing end-members - however, we contend that the possibility of missing end-members is already accounted for by allowing residuals in the tracer conservation equations. Thus, it is not clear why allowing a residual in the mass conservation equation would improve the quality of the solutions. For these reasons, we argue that mass conservation should ideally be strictly enforced in the solution.

In theory, one could achieve a near-zero mass conservation residual within the MATLAB OMP implementation by setting the weight W_M^{user} of the mass conservation equation to be very high. However, this would cause the resulting system of equations to have a poor condition number, which would in turn lead to numerical difficulties during optimization. Fortunately, modern least squares solvers (such as MATLAB's `lsqlin`) have support for enforcing linear equality constraints, and these can be leveraged to strictly satisfy mass conservation without running into numerical difficulties. PYOMPA uses this formulation; specifically, we use linear constraints with the OSQP solver [Stellato et al., 2020], which is called via the cvxpy python package [Diamond and Boyd, 2016, Agrawal et al., 2018] (OSQP is the default cvxpy solver for quadratic programs). As an aside, the OSQP solver has support for a variety of languages including MATLAB, R, Julia, and C/C++.

Revisiting our previous example of two end-members with salinity values of $e_S^1 = 32$ and $e_S^2 = 34$, PYOMPA would produce the solution $x^1 = 1$ and $x^2 = 0$ to explain an observation with salinity $s_S = 31.5$. This would correspond to salinity residual of 0.5, both in terms of the mean-normalized salinity values and the original salinity values.

2.2 Modeling both assimilation and remineralization in "extended" OMP analysis

As mentioned in Sec. 1.3, the MATLAB OMP implementation for extended OMP analysis includes the $\Delta_j^{\mathrm{PO_4}}$ term in its non-negativity constraint, thereby restricting the term to modeling only remineralization and not assimilation. PYOMPA's formulation of extended OMP analysis does not impose this restriction; thus, the equivalent term of $\Delta_j^{\mathrm{PO_4}}$ can adopt either positive or negative signs. As an aside, this remineralization variable is currently labeled as "oxygen deficit" in pyompa's plotting functions and the codebase, meaning that it is intended to represent $-\Delta_j^{\mathrm{O_2}}$. However, depending on the exchange ratios that the users specified, this variable could be used to represent any remineralized/assimilated quantity (e.g. if the user specified the exchange ratios in terms of phosphate rather than oxygen, this would cause the variable to effectively represent $\Delta_j^{\mathrm{PO_4}}$, even if it was labeled as "oxygen deficit" in the code). In later versions of PYOMPA, the user will be allowed to define what the variable should be called, and will further be given an option to restrict the sign of this remineralization parameter to be only positive (consistent with the MATLAB OMP formulation).

2.3 Support for flexible nutrient exchange ratios

Traditional OMP analysis presumes that variables are remineralized in a fixed Redfield ratio – however, it is known that the nutrient ratios can vary. For example, phytoplankton growing in phosphate-limited waters are known to use less phosphate per unit of nitrate [Mills and Arrigo, 2010]. Modeling flexibility in the nutrient exchange ratio allows us to account for such deviations.

How can we achieve flexibility in the exchange ratios? One idea is that we could define the equivalent of "end-members" for exchange ratios, and then structure our formulation such that the "effective" exchange ratio is a linear combination of the exchange-ratio "end-members". How do we do this? Let us first consider what happens if we introduce *multiple* remineralization variables per sample. We will denote the k^{th} remineralization variable for sample j as $\Delta_j^{\mathrm{PO_4},k}$. Let us further presume that each remineralization variable is associated with its own exchange ratio; we will denote the exchange ratio of parameter p to $\mathrm{PO_4}$ for the k^{th} remineralization variable as $r_{\mathrm{PO_4}}^{p,k}$. The OMP equations then become:

$$\left(\sum_i e_p^i x_j^i\right) + \left(\sum_k r_{\mathrm{PO_4}}^{p,k} \Delta_j^{\mathrm{PO_4},k}\right) = s_p^j + \epsilon_p^j \tag{15}$$

What is the "effective" exchange ratio for sample j according to this system of equations? Let us use $\Delta_j^{\mathrm{PO_4,tot}} = \sum_k \Delta_j^{\mathrm{PO_4},k}$ to denote the total amount of phosphate remineralized/assimilated for sample j. Accordingly, the total quantity of parameter p that is remineralized is $\Delta_j^{p,\text{tot}} = \sum_k r_{\mathrm{PO_4}}^{p,k} \Delta_j^{\mathrm{PO_4},k}$; thus, the effective exchange ratio of parameter p to $\mathrm{PO_4}$ becomes:

$$
\begin{aligned}
r^{p,\text{eff}}_{\text{PO}_4,j} &= \Delta^{p,\text{tot}}_j / \Delta^{\text{PO}_4,\text{tot}}_j \\
&= (\sum_k r^{p,k}_{\text{PO}_4} \Delta^{\text{PO}_4,k}_j) / \Delta^{\text{PO}_4,\text{tot}}_j \\
&= \sum_k r^{p,k}_{\text{PO}_4} (\Delta^{\text{PO}_4,k}_j / \Delta^{\text{PO}_4,\text{tot}}_j)
\end{aligned} \tag{16}
$$

Observe what happens if we assume that all $\Delta^{\text{PO}_4,k}_j$ have the same sign for all k; in that case, we are guaranteed that $\Delta^{\text{PO}_4,\text{tot}}_j$ will have the same sign as $\Delta^{\text{PO}_4,k}_j$, and so the ratio $(\Delta^{\text{PO}_4,k}_j / \Delta^{\text{PO}_4,\text{tot}}_j)$ will be nonnegative. Combined with the fact that $\sum_k (\Delta^{\text{PO}_4,k}_j / \Delta^{\text{PO}_4,\text{tot}}_j) = (\Delta^{\text{PO}_4,\text{tot}}_j / \Delta^{\text{PO}_4,\text{tot}}_j) = 1$, we can conclude that constraining $\Delta^{\text{PO}_4,k}_j$ to have the same sign for all k means that we can interpret $(\Delta^{\text{PO}_4,k}_j / \Delta^{\text{PO}_4,\text{tot}}_j)$ as the *proportion* of phosphate that is remineralized in the exchange ratio associated with the k^{th} remineralization variable. Note here that there is now an analogy to end-member mixing; in the same way that e^i_p represented the value of property p for end-member i, we can say that $r^{p,k}_{\text{PO}_4}$ represents the value of the exchange ratio $r^p_{\text{PO}_4}$ for the k^{th} "exchange-ratio end-member"; further, in the same way that x^i_j represented the proportion of the i^{th} end-member in sample j, we now have that $(\Delta^{\text{PO}_4,k}_j / \Delta^{\text{PO}_4,\text{tot}}_j)$ represents the proportion of the k^{th} "exchange-ratio end-member" in sample j. Thus, we conclude that the effective exchange ratio $r^{p,\text{eff}}_{\text{PO}_4,j}$ is a linear mixture (or, formally, a *convex combination*[1]) of the k "exchange-ratio end-members", and the only requirement for this conclusion is that $\Delta^{\text{PO}_4,k}_j$ must have the same sign for all k. In PYOMPA, we meet this requirement by computing the solution twice; once with $\Delta^{\text{PO}_4,k}_j \geq 0$ for all k, and once with $\Delta^{\text{PO}_4,k}_j \leq 0$ for all k, and we report the solution that achieves lower residuals.

2.4 Soft end-member usage penalties

In the situation where the number of unknowns being solved for exceeds the number of OMP equations (including both the parameter conservation equations and the mass conservation constraint), there can be ambiguity in the ideal solution. Prior work (e.g. Evans et al. [2020] and Peters et al. [2018], to name a couple) has avoided this ambiguity by separating the analysis into discrete sections (e.g. separating "intermediate" waters from "deep" waters), where only certain end-members are permitted to contribute to the observations from a particular section. However, this can result in high residuals at the boundaries separating different sections [Peters et al., 2018].

As an alternative to hard boundaries between sections, we propose introducing sample-specific penalties that encode prior knowledge of the likely end-member distributions. These penalties would be zero in regions where a water type is expected to occur, positive but small in regions where there is some uncertainty about whether a water type can occur, and high in regions where a water type is not expected to occur. They can be viewed as a generalization of the hard boundaries used in prior work, because the hard boundaries effectively correspond to a penalty of zero in sections that are allowed to contain a particular water type, and infinity in sections that are not allowed to contain the water type. Formally, let P^i_j denote the penalty on the presence of water type i for sample j. Building on our notation from the previous sections, where x^i_j represents the fraction of end-member i in sample j, we can define the following additional residuals terms:

$$
\epsilon^{j,\text{penalty}}_i := P^i_j x^i_j \tag{17}
$$

These residual terms will be included in the least squares minimization objective - in effect, $\epsilon^{j,\text{penalty}}_i$ can be thought of as the "cost" associated with the penalty P^i_j. By introducing additional residual equations into the OMP analysis in this way, the ambiguity in the optimal solution is reduced.

2.5 Summarizing the PYOMPA objective

Bringing it all together, we can fully specify PYOMPA's optimization objective. As noted in Sec. 2.2, PYOMPA's codebase currently calls the remineralization variable "oxygen deficit", but ultimately the remineralization variable could be taken to represent a surplus/deficit of any exchanged nutrient depending on how the user specifies the associated exchange ratios. To reflect this, we will just use the symbol Q to denote the remineralized quantity - be it an oxygen

[1] a *convex combination* is a linear combination where all the coefficients are non-negative and sum to 1

deficit, a phosphate surplus, or whatever else the user chooses to model. In principle, multiple types of processes could be modeled, and each would be associated with its own variables to denote the metabolized quantity - but for simplicity, we will stick to one process associated with Q in our equations.

In Eqn. 17 from Sec. 2.4, we defined the residuals associated with end-member usage penalties. Here, we additionally define the updated parameter residual equation used in PYOMPA that accomodates flexible nutrient exchange ratios (Sec. 2.3):

$$\epsilon_p^{j,\text{pyompa}} := W_p^{\text{user}} \left(\left(\sum_i e_p^i x_j^i \right) + \left(\sum_k r_{\text{Q}}^{p,k} \Delta_j^{\text{Q},k} \right) - s_p^j \right) \tag{18}$$

Notice that Eqn. 18 does not involve normalization by μ_p and σ_p; the reason for this is discussed in Sec. 2.5.1.

As before, we will use $\boldsymbol{x_j} = (x_j^1, x_j^2, ...)$ to denote a vector of fractional end-member contributions. Additionally, we will introduce $\boldsymbol{\Delta_j^{\text{Q}}} = (\Delta_j^{\text{Q},1}, \Delta_j^{\text{Q},2}, ...)$ to denote a vector of remineralization variables. The pyompa solution can then be written as follows:

$$\begin{aligned} \boldsymbol{x_j}^{\text{pyompa}}, \boldsymbol{\Delta_j^{\text{Q}}}^{,\text{pyompa}} := & \underset{\boldsymbol{x_j}, \boldsymbol{\Delta_j^{\text{Q}}}}{\operatorname{argmin}} \left[\left(\sum_p \left(\epsilon_p^{j,\text{pyompa}} \right)^2 \right) + \left(\sum_i (\epsilon_i^{j,\text{penalty}})^2 \right) \right] \\ & \text{Subject to the constraints:} \\ & x_j^i \geq 0 \text{ for all } i \text{ and} \\ & \sum_i x_j^i = 1 \text{ and} \\ & \text{EITHER } \Delta_j^{\text{Q},k} \geq 0 \text{ for all } k \text{ OR } \Delta_j^{\text{Q},k} \leq 0 \text{ for all } k \end{aligned} \tag{19}$$

Note that if the user chooses to work only with parameters that are conserved (analogous to the "classical" OMP formulation), then $\boldsymbol{\Delta_j^{\text{Q}}}$ will simply be an empty vector; thus, the optimization equation above encapsulates both the "classical" and the "extended" OMP formulations.

2.5.1 On the absence of explicit normalization of the PYOMPA residuals

Unlike the MATLAB OMP residuals from Eqn. 4, the PYOMPA residual equation (Eqn. 18) does not involve any normalization by μ_p or σ_p. Regarding the choice to not subtract μ_p: as discussed in Sec. 2.1, when the residual in the mass conservation is zero (as is achieved in PYOMPA by using a hard constraint), the optimal solution is invariant to mean normalization. Regarding the choice to not divide by σ_p: as noted in Sec. 1.2, the impact of dividing by the standard deviation is effectively to rescale the user-provided weights W_p^{user} by a factor of $1/\sigma_p$; thus, rescaling by the standard deviation is equivalent to the user having provided slightly different weights. To improve transparency, PYOMPA does not do this rescaling by the standard deviation, such that the "effective" residual weighting is equivalent to the user-provided weights; in a sense, one could think of the $1/\sigma_p$ as being "included in" PYOMPA's user-provided weights. We designed the package this way because it allows the user to tweak their end-member definitions at will without worrying about how the altered end-member definitions will impact the parameter weightings. In subsequent PYOMPA versions, there will be settings to allow the user to reproduce the MATLAB OMP implementation exactly.

Because of the choice to let the user have full control over the effective weights, a third consideration arises when deciding how to select weights, and that is: what is the inherent range of variation of the parameters? If a parameter varies over a large range, its residuals will likely also tend to be larger (for example: if you were to change the units of a parameter such that all the terms got multiplied by a factor of 1000, the residuals for that parameter would also become 1000 times larger); however, parameters that happen to vary over a larger range need not inherently contain more information. We recommend keeping this consideration in mind when setting the weights; as a starting point, a user could initially set the weights to be inversely proportional to the range over which each parameter tends to vary, and can then further increase the weights for those parameters that the user believes to be more informative (as per the considerations discussed in Sec. 1.2.1).

3 Handling ambiguity when the objective is underdetermined

Ambiguity in the solution can arise when the number of end-members we wish to consider is larger than the number of constraining equations. In these situations, multiple different end-member combinations result in mixtures with identical property values. This means that multiple different solutions may be optimal in terms of the residuals they achieve. The MATLAB OMP implementation described in Sec. 1 refuses to compute any solution in such cases, placing a hard limit on the maximum number of end-members it allows. In the case of PYOMPA, the solver used for Eqn. 19 will return an optimal solution according to where it converges, but without further computation it would not be clear which other solutions would have achieved equivalent residuals. Thus, in this section we consider the question of how to navigate situations where there is ambiguity in the solution.

To make the scenario more concrete, let us revisit the example from Section 1.1 where we were interested in the value of property X would be produced solely due to mixing. As an aside, having seen in Sec. 1.3 and 2.3 that our PYOMPA formulation can model first-order metabolic processes in addition to mixing, we can upgrade our question to ask about the value of property X that would be produced by the combination of mixing and any modeled metabolism. In the case where there is ambiguity in the solution, we can more specifically ask "according to the space of PYOMPA solutions under consideration, what are the maximum and minimum values of property X that would be produced solely by mixing and any modeled metabolism?".

To answer this question, we must first formalize the notion of "the space of PYOMPA solutions under consideration". Imagine we run PYOMPA for sample j and get an optimal solution for Eqn. 19, which we will denote as $\boldsymbol{x_j}^{\text{opt}}, \boldsymbol{\Delta_j^{\mathbf{Q}}}^{,\text{opt}}$. Let's use $\epsilon_p^{j,\text{opt}}$ to represent the unweighted residuals for property p and sample j in the original parameter space that are obtained by this solution. Formally, we define $\epsilon_p^{j,\text{opt}}$ as:

$$\epsilon_p^{j,\text{opt}} := \left(\sum_i e_p^i x_j^{i,\text{opt}}\right) + \left(\sum_k r_{\text{Q}}^{p,k} \Delta_j^{\text{Q},\text{opt},k}\right) - s_p^j \tag{20}$$

If we now want to restrict ourselves to the space of solutions that achieve equivalent residuals to this optimal solution, we can specify that we will only consider values for $\boldsymbol{x_j}$ and $\boldsymbol{\Delta_j^{\mathbf{Q}}}$ for which the residuals match $\epsilon_p^{j,\text{opt}}$. Formally, the constraints are written as follows :

$$\left(\sum_i e_p^i x_j^i\right) + \left(\sum_k r_{\text{Q}}^{p,k} \Delta_j^{\text{Q},k}\right) - s_p^j = \epsilon_p^{j,\text{opt}} \text{ for all } p \tag{21}$$

Alternatively, we may also be interested in the space of solutions that achieve residuals that are almost as good as the optimal solution but not necessarily equivalent to it. For example, if we don't have a lot of confidence in the measurements for a particular parameter, we may be willing to tolerate higher residuals. The pyompa codebase has functionality whereby a user can specify a maximum tolerated magnitude for each parameter's residual, and the solver will only consider solutions for which the residuals are within the tolerated magnitude (except in cases where the original solution's residuals exceed the specified tolerated magnitude, in which case pyompa will search for solutions that have equivalent residuals to the original solution). Formally, if we use ϵ_p^{max} to denote the user-specified maximum on the magnitude of the residual for parameter p, we have the constraints:

$$\min\left(\epsilon_p^{j,\text{opt}}, -\epsilon_p^{\text{max}}\right) <= \left(\sum_i e_p^i x_j^i\right) + \left(\sum_k r_{\text{Q}}^{p,k} \Delta_j^{\text{Q},k}\right) - s_p^j <= \max\left(\epsilon_p^{j,\text{opt}}, \epsilon_p^{\text{max}}\right) \text{ for all } p \tag{22}$$

When $\epsilon_p^{\text{max}} = 0$, the constraints in Eqn 22 reduce to the constraints in Eqn 21 for when the residuals are set to equal those from the optimal solution.

We may also want to limit our attention to solutions that achieved certain values for the end-member usage penalties introduced in Sec. 2.4. To keep our constraints linear in $\boldsymbol{x_j}$ (which we will be useful when the time comes to select a computational solver for the problem), we will specify a constraint such that the sum of the penalties incurred is no

larger than the sum of the penalties incurred in the original optimal solution. Using P_j^i to denote the penalty associated with the i^{th} end-member in the j^{th} sample, our constraint is:

$$\sum_i P_j^i x_j^i \leq \sum_i P_j^i x_j^{i,\text{opt}} \tag{23}$$

Now that we have seen how to formally specify a space of possible solutions using linear constraints, we can interrogate the space for answers to specific queries. In each case, the answer to are query will come from solving a constrained optimization problem, where the constraints are a combination of the PYOMPA constraints we saw in Eqn. 19 and the constraints from Eqn. 22 & 23 that we introduced here. Note that all the constraints are linear in $\boldsymbol{x_j}$ and $\boldsymbol{\Delta_j^Q}$, which means they can be readily incorporated into most solvers. If we use the function $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q})$ to denote the objective that we are minimizing, our optimization problem can be written as:

$$\begin{aligned}
&\underset{\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}}{\operatorname{argmin}}\, o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) \\
&\text{Subject to the constraints:} \\
\min\left(\epsilon_p^{j,\text{opt}}, -\epsilon_p^{\text{max}}\right) &\leq \left(\sum_i e_p^i x_j^i\right) + \left(\sum_k r_{\text{Q}}^{p,k} \Delta_j^{\text{Q},k}\right) - s_p^j \text{ for all } p \text{ and} \\
\max\left(\epsilon_p^{j,\text{opt}}, \epsilon_p^{\text{max}}\right) &\geq \left(\sum_i e_p^i x_j^i\right) + \left(\sum_k r_{\text{Q}}^{p,k} \Delta_j^{\text{Q},k}\right) - s_p^j \text{ for all } p \text{ and} \\
\sum_i P_j^i x_j^i &\leq \sum_i P_j^i x_j^{i,\text{opt}} \text{ and} \\
x_j^i &\geq 0 \text{ for all } i \text{ and} \\
\sum_i x_j^i &= 1 \text{ and} \\
\text{EITHER} \Delta_j^{\text{Q},k} &\geq 0 \text{ for all } k \text{ OR } \Delta_j^{\text{Q},k} \leq 0 \text{ for all } k
\end{aligned} \tag{24}$$

When $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q})$ is linear, our problem will have the form of a linear program, and when $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q})$ is quadratic, it will have the form of a constrained least-squares optimization problem. Thus, the problems are readily solvable by off-the-shelf solvers (in our case, we used the solvers that come with cvxpy).

3.1 Finding the max/min property values produced solely by mixing and modeled metabolism

We now consider the specific question we posed earlier in the section: "within the space of pyompa solutions under consideration, what are the maximum and minimum values of a property X that can be produced solely due to mixing and any modeled metabolism?". To answer this question, let us first recap how we calculate the value of property X is predicted. If we are given the water mass fractions $\boldsymbol{x_j}$ and nutrient metabolism values $\boldsymbol{\Delta_j^Q}$ for a sample j, and we know the end-member compositions e_X^i for property X as well as the nutrient exchange ratios r_{Q}^X (if applicable) of X w.r.t. the metabolized nutrient Q, then we can calculate the expected value $\hat{s}_X^j$ of property X in sample j that would be produced solely mixing and modeled metabolism as:

$$\hat{s}_X^j := \left(\sum_i e_X^i x_j^i\right) + \left(\sum_k r_{\text{Q}}^{X,k} \Delta_j^{\text{Q},k}\right) \tag{25}$$

Thus, we would set our objective function (which is minimized) to be $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := \hat{s}_X^j$ when we are trying to find the minimum value of property X, and $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := -\hat{s}_X^j$ when we are trying to find the maximum value.

3.2 Reporting maximum and minimum limits of a water type in a sample

Suppose our question has the form "within the space of PYOMPA solutions, what is the maximum and minimum fraction of the sample comprised by a particular water type?". If we only have a single end-member representing this water-type (which we will denote as the i^*_{th} end-member), we can find the minimum using the objective function $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := x_j^{i^*}$ and the maximum with $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := -x_j^{i^*}$. In the case where multiple end-members are each representing subtypes of the water type (e.g. subtypes selected via archetype analysis in Sec. 5, and we are interested in the overall amount of that water type, then if we use the set I to represent the indices of all the subtypes, we can find the minimum with the objective function $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := \sum_{i^* \in I} x_j^{i^*}$ and the maximum with $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := -\sum_{i^* \in I} x_j^{i^*}$

3.3 Hybridizing the PYOMPA solution with the solution from another model (e.g. an OCIM)

Supposing we conduct water mass mixing analysis using a different type of model, such as an Ocean Circulation Inverse Model (OCIM). Let $\boldsymbol{x_j}^{\text{ocim}}$ denote the ocim-derived values for the end-member fractions in in sample j. We can use this to select a unique pyompa solution from the space of solutions with "acceptable" residuals, thereby combining the strengths of OCIM and PYOMPA. Specifically, we would set our objective function (which is minimized) to be the sum of squared residuals between the end-member fractions and the OCIM end-member fractions, i.e. $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := \sum_{i^* \in I} x_j^{i^*}$ and the maximum with $o(\boldsymbol{x_j}, \boldsymbol{\Delta_j^Q}) := \sum_i (x_j^i - x_j^{i,\text{ocim}})$.

3.3.1 Performing water mass mixing analysis with an OCIM

An Ocean Circulation Inverse Models (OCIM) [DeVries and Primeau, 2011] is a model that divides the ocean into a 3D grid and incorporates both empirical tracer observations and dynamical constraints (i.e. constraints on the physics of ocean circulation) to determine global ocean currents. Let us use $\boldsymbol{x_t}$ to denote a vector of the tracer concentrations in each gridbox at timestep t. An OCIM yields a "tracer transport matrix" T such that, in the absence of any sources and sinks, $\boldsymbol{x_{t+1}} = x_t + \boldsymbol{T x_t}$. If $s(x)$ is a vector representing the sources and sinks, then $\boldsymbol{x_{t+1}} = x_t + T\boldsymbol{x_t} + s(x_t)$. If x represents a steady-state vector, then we have $x_{t+1} = x_t$, yielding $x = x + Tx + s(x)$ and $0 = Tx + s(x)$.

How can we perform water mass mixing analysis given a tracer transport matrix $\boldsymbol{T}$? We will base our approach on an existing method in the literature (e.g. used in Holzer et al. [2021]) and will show how it can be adapted to our water mass mixing question. Intuitively, the existing approach in the literature is to designate a fictional tracer that is created very rapidly at gridboxes corresponding to a particular end-member and is also destroyed very rapidly at gridboxes corresponding to *any* end-member, such that the maximum concentration of this tracer is 1. Under this scheme, the steady-state concentration of this tracer in each gridbox will represent the steady-state mass fractions corresponding to that end-member.

Formally, let n represent the number of ocean gridboxes, and let $\boldsymbol{m_i}$ be an 'end-member vector' of length n that is 1 at gridboxes corresponding to the ith end-member and 0 elsewhere (that is, it is a 1 at index j if the jth gridbox belongs to the ith end-member and 0 otherwise). Let $\boldsymbol{M}$ be an 'end-member masking matrix' of size nxn that is 0 off the diagonal and is 1 at only at those diagonal positions corresponding to end-members (that is, it is a 1 at position (j, j) if the jth gridbox belongs to *any* end-member, and 0 otherwise). Left-multiplying by matrix M has the effect of masking out any entries that do not correspond to end-members. Let j be the index of a particular gridbox. If j belongs to end-member i, we will declare that the tracer is created at gridbox j a rate of $1/\tau$, where τ is a number much smaller than 1 representing the 'timescale' of creation. Similarly, if j belongs to *any* end-member (including end-member i), we will also declare that the tracer is destroyed at this gridbox at a rate x_j/τ. That is, the 'net' rate of creation is 0 if j does not belong to any end-member, $-x_j/\tau$ if j corresponds to an end-member other than end-member i, and $(1 - x_j)/\tau$ if j corresponds to end-member i. We can therefore write our steady-state equation as:

$$
\begin{aligned}
0 &= \boldsymbol{T x} + \frac{\boldsymbol{m_i}}{\tau} - \frac{\boldsymbol{M x}}{\tau} \\
\left(\frac{\boldsymbol{M}}{\tau} - \boldsymbol{T}\right) x &= \frac{\boldsymbol{m_i}}{\tau}
\end{aligned} \tag{26}
$$

Because τ is a constant, Eqn. 26 has the form $\boldsymbol{Ax} = \boldsymbol{b}$ ($\boldsymbol{A}$ is a matrix, $\boldsymbol{b}$ is a constant vector, and $\boldsymbol{x}$ is the vector of variables to solve for), and can thus be solved using standard solvers for linear systems of equations. In the literature, τ is typically set to be some small fractional value (e.g. if $\boldsymbol{T}$ is on the timescale of years, then τ is set to be the fraction of one second to a year). Here, we will show that we can eliminate the dependence of the solution on τ by computing the solution in the limit where τ approaches zero. To our knowledge, the ability to eliminate the dependence on τ has not been noted before in the literature.

First, we will multiply both sides of the equation by M to focus on end-member gridboxes. This gives:

$$\boldsymbol{M}\left(\frac{\boldsymbol{M}}{\tau}-\boldsymbol{T}\right)\boldsymbol{x}=\boldsymbol{M}\frac{\boldsymbol{m_i}}{\tau}$$

Because $\boldsymbol{M}$ has the effect of masking out any rows that do not correspond to end-member gridboxes, left-multiplying by $\boldsymbol{M}$ does not change the matrix $\boldsymbol{M}$ or the vector $\boldsymbol{m_i}$. Thus, we have $\boldsymbol{MM}=\boldsymbol{M}$ and $\boldsymbol{Mm_i}=\boldsymbol{m_i}$, giving:

$$\frac{\boldsymbol{M}}{\tau}\boldsymbol{x}-\boldsymbol{T}\boldsymbol{x}=\frac{\boldsymbol{m_i}}{\tau}$$

If we now multiply through by τ, we have:

$$\boldsymbol{M}\boldsymbol{x}-\tau\boldsymbol{T}\boldsymbol{x}=\boldsymbol{m_i}$$

If we take the limit where $\tau \to 0$, we have:

$$\boldsymbol{M}\boldsymbol{x}=\boldsymbol{m_i} \tag{27}$$

Note that Eqn. 27 can be read as a hard constraint on the vector x, where gridboxes corresponding to end-members are constrained to have a value of 1 if they are part of end-member i and 0 otherwise.

Now that we have determined the solution to x for gridboxes corresponding to end-members, we can turn our attention to the remaining gridboxes. To identify these gridboxes, we can simply multiply both sides of the equation by $(\boldsymbol{I}-\boldsymbol{M})$ where $\boldsymbol{I}$ is the identity matrix. Recall that $\boldsymbol{M}$ was a diagonal matrix such that left-multiplying by $\boldsymbol{M}$ masked out non-end-member rows; thus, left-multiplying by $(\boldsymbol{I}-\boldsymbol{M})$ has the effect of masking out end-member rows. In other words, $(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{M}=\boldsymbol{0}$ and $(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{m_i}=\boldsymbol{0}$ (here, $\boldsymbol{0}$ represents the all-zeros matrix and all-zeros vector, respectively). We have:

$$\begin{aligned}(\boldsymbol{I}-\boldsymbol{M})\left(\frac{\boldsymbol{M}}{\tau}-\boldsymbol{T}\right)\boldsymbol{x}&=(\boldsymbol{I}-\boldsymbol{M})\frac{\boldsymbol{m_i}}{\tau}\\(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{T}\boldsymbol{x}&=0\end{aligned} \tag{28}$$

If we now simply add Eqn. 27 and Eqn. 28, we get:

$$\begin{aligned}\boldsymbol{M}\boldsymbol{x}+(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{T}\boldsymbol{x}&=\boldsymbol{m_i}\\(\boldsymbol{M}+(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{T})\boldsymbol{x}&=\boldsymbol{m_i}\end{aligned} \tag{29}$$

Once again, Eqn. 29 has the form $\boldsymbol{Ax}=\boldsymbol{b}$ and can be solved with standard solvers for linear systems - except now without the dependence on the timescale constant τ. We found that the numerical difference between using Eqn. 26 and Eqn. 29 was small in practice, but we anticipate that using 29 improves the numerical stability of the results (particularly when τ is small).

Finally, we note that Eqn. 29 can be easily adapted not just to calculate steady-state end-member fractions, but also the steady-state distributions of any tracer of interest using the end-members as sources for the tracer; all that we need to know is the value of the tracer conentrations at the end-member gridboxes. In the cases of water-mass mixing analysis, we used $\boldsymbol{m_i}$ as the distribution of tracer conentrations (that is why $\boldsymbol{m_i}$ was 1 at gridboxes corresponding to the end-member of interest and 0 elsewhere); instead, if we use a vector $\boldsymbol{c}$ such that c_j equals the concentration of a particular tracer (such as salinity) at an end-member gridbox when j belongs to an end-member, and c_j is 0 otherwise, then we can solve:

$$(\boldsymbol{M}+(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{T})\boldsymbol{x}=\boldsymbol{c} \tag{30}$$

to get the steady-state distribution of the tracer. Note the residuals obtained by using Eqn. 30 (relative to the ground-truth distribution of the tracer) may be quite large if the end-members used do not correspond to the original end-members used in the OCIM to calcualate the matrix T (e.g. if the OCIM model was build using the surface gridboxes as the end-members, and the user switches to using deep waters as end-members, the resulting errors may be much larger than what was found using the original OCIM model). In other words, caution must be applied when changing end-members. In these scenarios, we recommend using the hybridization approach described here, where PYOMPA is combined with the results of the OCIM (rather than directly relying on the OCIM).

4 Solving the reverse problem: end-member properties from end-member fractions

We have seen how to solve the problem "given the property values for end-members, find a fractional combination of end-members for each sample that best fits the observed sample properties". What if, having solved this problem for several samples, we consider the inverse problem "given the fractional combination of end-members for every sample, find the end-member value for each property that best fits the values of the property in the samples"?

More formally, given the pyompa solution $(\boldsymbol{x_j^{\text{pyompa}}}, \boldsymbol{\Delta_j^{\mathbf{Q},\text{pyompa}}})$ for each sample j, we define the residuals $\epsilon_p^{j,\text{inv}}$ for each sample in the inverse problem as:

$$\epsilon_p^{j,\text{inv}} := W_p^{\text{user}} \left(\left(\sum_i e_p^i x_j^{i,\text{pyompa}} \right) + \left(\sum_k r_{\text{Q}}^{p,k} \Delta_j^{\text{Q,pyompa},k} \right) - s_P^j \right) \tag{31}$$

Then, for a given property p, we optimize for the values of e_p^i and $r_{\text{Q}}^{p,k}$ that minimize the sum of the squared residuals. Formally, if we use $\boldsymbol{e_p}$ to represent the vector $(e_p^1, e_p^2, ...)$ and $\boldsymbol{r_{\mathbf{Q}}^p}$ to represent the vector $(r_{\text{Q}}^{p,1}, r_{\text{Q}}^{p,2}, ...)$, we can write the solution to the reverse problem for property p as:

$$\begin{aligned} &\underset{\boldsymbol{e_p}, \boldsymbol{r_{\mathbf{Q}}^p}}{\operatorname{argmin}} \sum_j (\epsilon_p^{j,\text{inv}})^2 \\ &\text{Subject to the constraints:} \\ &\text{Range constraints on } \boldsymbol{e_p} \\ &\text{Range constraints on } \boldsymbol{r_{\mathbf{Q}}^p} \text{ (if applicable)} \end{aligned} \tag{32}$$

As noted in Eqn. 32 above, there will very likely be range constraints on e_p (e.g. if p is salinity, we would likely want to constrain each end-member to have a salinity value that is compatible with prior knowledge). Similarly, the nutrient exchange ratios would also likely have limitations on their plausible ranges based on prior knowledge. Users could also consider adding soft penalties on end-member properties (analogous to the soft penalties on end-member fractions from Sec. 2.4).

The following table summarizes the differences between the forward and reverse problems:

Value type	Forward Problem	Reverse Problem
Variables to solve for	End-member fractions for a sample & (if applicable) metabolism delta values	End-member values & (if applicable) nutrient exchange ratios for a property
Other provided information	End-member property values & (if applicable) nutrient exchange ratios	End-member fractions & (if applicable) metabolism delta values
Objective function (same)	Residual difference between observed vs. predicted sample values	Residual difference between observed vs. predicted sample values
Independence	Each sample can be solved for independently	Each property can be solved for independently
Underdetermined solution	Having too-few types of properties leads to an underdetermined solution	Having too-few samples leads to an underdetermined solution
Constraints/Penalties	Constraints/penalties ensure valid/plausible values for end-member fractions and metabolism values	Constraints/penalties ensure valid/plausible values for end-member properties and nutrient exchange ratios

In principle, a user could adopt a dual iteration scheme where the forward problem is solved in the first iteration, followed by the reverse problem (which would use the values computed by the forward problem), followed again by the forward problem (which would use updated property values from the most recent solve of the reverse problem), and so on until convergence. This would allow the end-member properties and end-member fractions to be tuned simultaneously, and could be beneficial in cases where there is limited knowledge about the end-members. However, such a method should be used with caution because the optimization problem is not necessarily convex (making it possible for the solver to find a suboptimal solution). One should also be cautious of finding spurious patterns in limited data (also known as "over-fitting", where the optimization problem achieves very low residuals while the solution produced does not resemble the ground-truth).

5 Archetype analysis for selecting end-member subtypes

In many settings, the water types that we wish to use in our mixing analysis may not be well-characterized by a single end-member; rather, we would prefer to define multiple end-member "subtypes" such that the effective span of property values represented by the water type is the mixture of end-members corresponding to the subtypes (analogous to how represent our collected sample as a mixture of all the end-members). This leads us to ask: given a region of the ocean corresponding to an "water type", how can we define the corresponding end-member subtypes?

One possibility is to use a data-driven approach called *archetype analysis* [Cutler and Breiman, 1994]. Say we have a dataset of observations corresponding to the i^{th} water type. Let d_p^k denote the value of parameter p for the k^{th} observation in the dataset. Let us further say that we want to define n end-members that characterize this water type. Archetypal analysis will seek n end-members such that (1) each observation in the dataset can be approximately explained as a mixture of the end-members, and (2) the end-members themselves are defined as a mixture of observations in the dataset. Formally, let $e_p^{i,j}$ denote the value of parameter p for the j^{th} end-member subtype for the i^{th} water type, and let $\hat{d}^k$ denote the approximated version of observation d^k. Archetype analysis introduces two sets of parameters α_k^j and β_k^j, where α_k^j denotes the amount of end-member $e^{i,j}$ in the approximation of observation k (that is, $\hat{d}_p^k = \sum_i \alpha_k^j e_p^j$), and β_k^j denotes the amount of the observation k in the end-member $e^{i,j}$ (that is, $e_p^{i,j} = \sum_k \beta_k^j d_p^k$). These parameters are tuned such that the residual sum of squares between the approximated observations and the true observations is minimized. The objective is defined as:

$$\underset{\alpha,\beta}{\operatorname{argmin}} \sum_k \sum_p (\hat{d}_p^k - d_p^k)^2$$

Subject to the following constraints (which enforce that we have a valid mixture, i.e. the fractions are non-negative and they sum to 1):

$$\beta_k^j \geq 0, \sum_k \beta_k^j = 1$$

$$\alpha_k^j \geq 0, \sum_j \alpha_k^j = 1$$

Note that this kind of archetype analysis does not have to be restricted to identifying end-members for a single water type; it could instead be applied to any dataset to find the best end members as defined by the data. However, some limitations would need to be kept in mind. First, the core assumption of archetype analysis is that the end members can be described as a mixture of the data. While this is a reasonable assumption in cases where every end-member has some points within the dataset that are comprised almost exclusively of that end-member, it may not always be the case (e.g. if considering data from only a single cruise's transect). Second, because archetype analysis focuses on minimizing the residual sum of squares, it will emphasize archetypes that characterize regions of the data that are more densely sampled, and may leave out archetypes important for regions that are more sparsely sampled; to get archetype analysis to prioritize all regions of the data equally irrespective of density, a subsampling or reweighting scheme would need to be implemented. Third, there is no guarantee that the end members identified by archetype analysis will correspond in an obvious way to water types from the literature. For this reason, we suggest that users interested in archetype analysis first leverage domain knowledge to identify portions of a dataset that plausibly correspond to a water type, and then use archetype analysis to choose the best end-members to define the water type.

6 Methods for thermocline analysis

Many oceanographic datasets contain observations from a region of the ocean that lies below the mixed layer and which is subject to strong density stratification. This region is called the "pycnocline", or (in cases where the density

stratification is driven by a temperature gradient) the "thermocline". Because it is energetically unfavorable to have mixing between density strata, we can assume that the end-members that comprise a sample from the pycnocline have approximately the same density as the sample. This in turn suggests we should solve a different OMP problem for samples from each layer in the density stratification, where each problem has the end-member definitions appropriate to the density layer. This technique (of solving different OMP problems for each layer) was used in Jenkins et al. [2015] and Peters et al. [2018] and is often called the "thermocline array" technique. The pyompa codebase has support for this technique via the `ThermoclineArrayOMPAProblem` class.

7 Conclusion

Water mass analyses are most often used to understand which sample chemical properties can be explained by water mass mixing to identify samples that might have a biogeochemical process impacting its properties. PYOMPA is an evolving software package with the aim of reducing limitations seen in past water mass analyses and making the tool more accessible. Multiple options and uses are proposed in the hope that the reader will be able to use PYOMPA to conduct a water mass analysis in a way most useful for their application.

8 Author Contributions

Avanti Shrikumar developed the mathematical methods and the PYOMPA python package. Rian Lawrence helped apply PYOMPA to GP15 data, which motivated the creation of PYOMPA's key features. Karen Casciotti provided guidance and feedback. Avanti Shrikumar wrote this technical note, with input from Rian Lawrence and Karen Casciotti.

9 Funding and Acknowledgments

Avanti Shrikumar is supported by the Ram and Vijay Shriram fellowship from the Stanford Data Science Institute. Rian Lawrence is supported by NSF grant number 1657944.

We thank Emilie LeRoy (WHOI) for assistance and suggestions with applying pyompa to GP15 and GA01 data, which played an important role in developing and refining the work. We thank Natalya Evans (USC) for several helpful pointers on the MATLAB OMP implementation.

References

Matthias Tomczak. A multi-parameter extension of temperature/salinity diagram techniques for the analysis of non-isopycnal mixing. *Prog. Oceanogr.*, 10(3):147–171, January 1981.

Rory Ory Thompson and R J Edwards. Mixing and water-mass formation in the australian subantarctic. *J. Phys. Oceanogr.*, 11(10):1399–1406, 1981.

Brian D Peters, William J Jenkins, James H Swift, Christopher R German, James W Moffett, Gregory A Cutter, Mark A Brzezinski, and Karen L Casciotti. Water mass analysis of the 2013 US GEOTRACES eastern pacific zonal transect (GP16). *Mar. Chem.*, 201:6–19, April 2018.

Johannes Karstensen and Matthias Tomczak. Age determination of mixed water masses using CFC and oxygen data. *J. Geophys. Res.*, 103(C9):18599–18609, August 1998.

B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. doi:10.1007/s12532-020-00179-2. URL `https://doi.org/10.1007/s12532-020-00179-2`.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Akshay Agrawal, Robin Verschueren, Steven Diamond, and Stephen Boyd. A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60, 2018.

Matthew M Mills and Kevin R Arrigo. Magnitude of oceanic nitrogen fixation influenced by the nutrient uptake ratio of phytoplankton. *Nat. Geosci.*, 3(6):412–416, May 2010.

Natalya Evans, Elisabeth Boles, Jarek V Kwiecinski, Susan Mullen, Martin Wolf, Allan H Devol, Rintaro Moriyasu, Sunghyun Nam, Andrew R Babbin, and James W Moffett. The role of water masses in shaping the distribution of redox active compounds in the eastern tropical north pacific oxygen deficient zone and influencing low oxygen concentrations in the eastern pacific ocean. *Limnol. Oceanogr.*, 65(8):1688–1705, 2020.

Tim DeVries and François Primeau. Dynamically and observationally constrained estimates of Water-Mass distributions and ages in the global ocean. *J. Phys. Oceanogr.*, 41(12):2381–2401, December 2011.

Mark Holzer, Tim DeVries, and Casimir de Lavergne. Diffusion controls the ventilation of a pacific shadow zone above abyssal overturning. *Nat. Commun.*, 12(1):4348, July 2021.

Adele Cutler and Leo Breiman. Archetypal analysis. *Technometrics*, 36(4):338–347, November 1994.

W J Jenkins, W M Smethie, E A Boyle, and G A Cutter. Water mass analysis for the U.S. GEOTRACES (GA03) north atlantic sections. *Deep Sea Res. Part 2 Top. Stud. Oceanogr.*, 116:6–20, June 2015.