PYOMPA VERSION 0.3: 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

May 14, 2021

ABSTRACT

This document is intended as a technical note for PYOMPA version 0.3 (`https://github.com/nitrogenlab/pyompa/tree/v0.3.1.0-alpha`), a python package for conducting optimum multi-parameter (OMP) analysis. It is being made available in advance of a formal publication so that it may be used as a reference for individuals who wish to use PYOMPA in their research today. The focus of this document is on the mathematical formulation. Please refer to the github README for usage instructions and examples.

Keywords Optimum Multi-Parameter Analysis · OMP · OMPA · PYOMPA · Convex Combination

1 Background on the OMP method

In this section, we will go over the OMP formulation currently available as a MATLAB implementation at `https://omp.geomar.de`.

1.1 "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_p^i denote the value of property p for end-member i (e.g. we will use e_T^1 to denote the value of the temperature parameter T for end-member 1), let s_p^j denote the value of property p in measured sample j, and let x_j^i 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 $\sum_i e_p^i x_j^i = s_p^j$. However, in practice there may be uncertainty both in the end-member properties e_p^i and in the measured sample properties s_p^j. In such a situation, our best estimates of the end-member fractions x_j^i may not be able to perfectly explain the sample properties s_p^j, resulting in a *residual*. Let us use ϵ_p^j to denote the residual in explaining property p for sample j. If we modify our earlier equation to include residuals, we have:

$$\sum_i e_p^i x_j^i = s_p^j + \epsilon_p^j \tag{1}$$

How do we solve for the values of x_j^i? 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_p^j 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:

$$\mu_p := \frac{1}{M} \sum_i e_p^i \tag{2}$$

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

After subtracting the mean and normalizing by the variance, the normalized end-member parameter values become $e_p^{i,\text{norm}} := (e_p^i - \mu_p)/\sigma_p$ and the normalized sample property values become $s_p^{j,\text{norm}} := (s_p^j - \mu_p)/\sigma_p$. 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_p^{user}. Putting it all together, we have:

$$\begin{aligned}
\epsilon_p^{j,\text{classic}} :=& W_p^{\text{user}} \left(\left(\sum_i e_p^{i,\text{norm}} x_j^i \right) - s_p^{j,\text{norm}} \right) \\
=& W_p^{\text{user}} \left(\left(\sum_i \frac{(e_p^i - \mu_p)}{\sigma_p} x_j^i \right) - \frac{(s_p^j - \mu_p)}{\sigma_p} \right) \\
=& \frac{W_p^{\text{user}}}{\sigma_p} \left(\left(\sum_i (e_p^i - \mu_p) x_j^i \right) - (s_p^j - \mu_p) \right)
\end{aligned} \tag{4}$$

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 [...], we can write the full optimization objective for classical OMP analysis as:

$$\boldsymbol{x}_j^{\text{classic}} := \underset{\boldsymbol{x_j}}{\operatorname{argmin}} \left[\left(\epsilon_M^{j,\text{classic}}\right)^2 + \sum_p \left(\epsilon_p^{j,\text{classic}}\right)^2 \right] \tag{7}$$

subject to the constraints:

$x_j^i \geq 0$ for all i

1.1.1 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).

1.2 "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 $r_p^{\text{PO}_4}$. For example, the default value of $r_{\text{O}_2}^{\text{PO}_4}$ 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 $r_{\text{NO}_3}^{\text{PO}_4}$ is 16. If we use the variable $\Delta_{\text{PO}_4}^j$ 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:

$$\left(\sum_i e_p^i x_j^i\right) + r_p^{\text{PO}_4} \Delta_j^{\text{PO}_4} = s_p^j + \epsilon_p^j \tag{8}$$

The term in blue highlights how the equation above differs the corresponding Eqn. 1 from classical OMP analysis. Note that $r_{\text{PO}_4}^{\text{PO}_4}$ will equal 1. Further, if a particular parameter (like salinity) is modeled as being unaffected by remineralization, $r_p^{\text{PO}_4}$ 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:

$$\epsilon_p^{j,\text{ext}} = \frac{W_p^{\text{user}}}{\sigma_p} \left(\left(\sum_i (e_p^i - \mu_p) x_j^i \right) + r_p^{\text{PO}_4} \Delta_j^{\text{PO}_4} - (s_p^j - \mu_p) \right) \tag{9}$$

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 $\Delta_j^{\mathrm{PO_4}}$ 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 $\epsilon_M^{j,\text{classic}}$ from Eqn. 5, we can write the optimization objective used in extended OMP analysis as:

$$\begin{aligned} \boldsymbol{x}_j^{\text{ext}}, \Delta_j^{\mathrm{PO_4}} := \operatorname*{argmin}_{\boldsymbol{x}_j, \Delta_j^{\mathrm{PO_4}}} &\left(\epsilon_M^{j,\text{classic}}\right)^2 + \sum_p \left(\epsilon_p^{j,\text{ext}}\right)^2 \\ &\text{subject to the constraints:} \\ &x_j^i \geq 0 \text{ for all } i \text{ and} \\ &\Delta_j^{\mathrm{PO_4}} \geq 0 \end{aligned} \tag{10}$$

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

1.2.1 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:

$$\begin{aligned} \left(\sum_i e_{p_1}^i x_j^i\right) + r_{p_1}^{\mathrm{PO_4}}\Delta_j^{\mathrm{PO_4}} &= s_{p_1}^j + \epsilon_{p_1}^j \\ \left(\sum_i e_{p_2}^i x_j^i\right) + r_{p_2}^{\mathrm{PO_4}}\Delta_j^{\mathrm{PO_4}} &= s_{p_2}^j + \epsilon_{p_2}^j \end{aligned}$$

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:

$$\left(\sum_i \left(e_{p_1}^i - \frac{r_{p_1}^{\mathrm{PO_4}}}{r_{p_2}^{\mathrm{PO_4}}} e_{p_2}^i\right) x_j^i\right) = \left(s_{p_1}^j - \frac{r_{p_1}^{\mathrm{PO_4}}}{r_{p_2}^{\mathrm{PO_4}}} s_{p_2}^j\right) + \left(\epsilon_{p_1}^j - \frac{r_{p_1}^{\mathrm{PO_4}}}{r_{p_2}^{\mathrm{PO_4}}} \epsilon_{p_2}^j\right) \tag{11}$$

Note that Eqn. 11 does not involve $\Delta_j^{\mathrm{PO_4}}$. 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:

$$\left(\sum_i e_{p_{1,2}}^i x_j^i\right) = s_{p_{1,2}}^j + \epsilon_{p_{1,2}}^j \tag{13}$$

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 $\Delta_j^{\mathrm{PO}_4} \geq 0$; 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 $\epsilon^j_{p_{1,2}} = \epsilon^j_{p_1} - \frac{r^{\mathrm{PO}_4}_{p_1}}{r^{\mathrm{PO}_4}_{p_2}}\epsilon^j_{p_2}$ 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 $\Delta_j^{\mathrm{PO}_4} \geq 0$, and is thus capable of modeling both assimilation and remineralization.

2 PYOMPA's OMP formulation (as of version 0.3)

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 section 1.1, 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 $e^1_S = 32$ and $e^2_S = 34$, 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 section 1.2, the MATLAB OMP implementation for extended OMP analysis includes the $\Delta^{\mathrm{PO}_4}_j$ 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 $\Delta^{\mathrm{PO}_4}_j$ 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 $-\Delta^{\mathrm{O}_2}_j$. 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 $\Delta^{\mathrm{PO}_4}_j$, 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 fixed "redfield ratios" - 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 $\Delta^{\mathrm{PO}_4,k}_j$. 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 PO_4 for the k^{th} remineralization variable as $r^{\mathrm{PO}_4,k}_p$. The OMP equations then become:

$$\left(\sum_i e^i_p x^i_j\right) + \left(\sum_k r^{\mathrm{PO}_4,k}_p \Delta^{\mathrm{PO}_4,k}_j\right) = s^j_p + \epsilon^j_p \tag{15}$$

What is the "effective" exchange ratio for sample j according to this system of equations? Let us use $\Delta^{\mathrm{PO}_4,\mathrm{tot}}_j = \sum_k \Delta^{\mathrm{PO}_4,k}_j$ to denote the total amount of phosphate remineralized/assimilated for sample j. Accordingly, the total quantity of parameter p that is remineralized is $\Delta^{p,\mathrm{tot}}_j = \sum_k r^{\mathrm{PO}_4,k}_p \Delta^{\mathrm{PO}_4,k}_j$; thus, the effective exchange ratio of parameter p to PO_4 becomes:

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

Observe what happens if we assume that all $\Delta^{\mathrm{PO}_4,k}_j$ have the same sign for all k; in that case, we are guaranteed that $\Delta^{\mathrm{PO}_4,\mathrm{tot}}_j$ will have the same sign as $\Delta^{\mathrm{PO}_4,k}_j$, and so the ratio $(\Delta^{\mathrm{PO}_4,k}_j / \Delta^{\mathrm{PO}_4,\mathrm{tot}}_j)$ will be nonnegative. Combined with the fact that $\sum_k (\Delta^{\mathrm{PO}_4,k}_j / \Delta^{\mathrm{PO}_4,\mathrm{tot}}_j) = (\Delta^{\mathrm{PO}_4,\mathrm{tot}}_j / \Delta^{\mathrm{PO}_4,\mathrm{tot}}_j) = 1$, we can conclude that constraining $\Delta^{\mathrm{PO}_4,k}_j$ to have the same sign for all k means that we can interpret $(\Delta^{\mathrm{PO}_4,k}_j / \Delta^{\mathrm{PO}_4,\mathrm{tot}}_j)$ 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 $r^{\mathrm{PO}_4,k}_p$ represents the value of the exchange ratio $r^{\mathrm{PO}_4}_p$ for the k^{th} "exchange-ratio end-member"; further, in the same way that x_j^i represented the proportion of the i^{th} end-member in sample j, we now have that $(\Delta_j^{\mathrm{PO_4},k}/\Delta_j^{\mathrm{PO_4},\mathrm{tot}})$ represents the proportion of the k^{th} "exchange-ratio end-member" in sample j. Thus, we conclude that the effective exchange ratio $r_{p,j}^{\mathrm{PO_4},\mathrm{eff}}$ 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 $\Delta_j^{\mathrm{PO_4},k}$ must have the same sign for all k. In PYOMPA, we meet this requirement by computing the solution twice; once with $\Delta_j^{\mathrm{PO_4},k} \geq 0$ for all k, and once with $\Delta_j^{\mathrm{PO_4},k} \leq 0$ 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_j^i denote the penalty on the presence of water type i for sample j. Building on our notation from the previous sections, where x_j^i represents the fraction of end-member i in sample j, we can define the following additional residuals terms:

$$\epsilon_i^{j,\mathrm{penalty}} := P_j^i x_j^i \tag{17}$$

These residual terms will be included in the least squares minimization objective - in effect, $\epsilon_i^{j,\mathrm{penalty}}$ can be thought of as the "cost" associated with the penalty P_j^i. 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 section 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 deficit, a phosphate surplus, or whatever else the user chooses to specify.

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):

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

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 $\boldsymbol{\Delta_j^{\mathrm{Q}}} = (\Delta_j^{\mathrm{Q},1}, \Delta_j^{\mathrm{Q},2}, ...)$ to denote a vector of remineralization variables. The pyompa solution can then be written as follows:

---

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

$$\boldsymbol{x_j}^{\text{pyompa}}, \boldsymbol{\Delta}_{\boldsymbol{j}}^{\mathbf{Q},\text{pyompa}} := \underset{\boldsymbol{x_j}, \boldsymbol{\Delta}_{\boldsymbol{j}}^{\mathbf{Q}}}{\operatorname{argmin}} \left[\left(\sum_p \left(\epsilon_p^{j,\text{pyompa}} \right)^2 \right) + \left(\sum_i (\epsilon_i^{j,\text{penalty}})^2 \right) \right] \tag{19}$$

Subject to the constraints:

$x_j^i \geq 0$ for all i and

EITHER $\Delta_j^{\mathrm{Q},k} \geq 0$ for all k OR $\Delta_j^{\mathrm{Q},k} \leq 0$ for all k

Note that if the user chooses to work only with parameters that are conserved (analogous to the "classical" OMP formulation), then $\boldsymbol{\Delta}_{\boldsymbol{j}}^{\mathbf{Q}}$ 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. 1.1, 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 Section 1.1.1).

3 Author Contributions

Avanti Shrikumar conceived of PYOMPA's mathematical formulation and developed the 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.

4 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 Peters, Rachel Horak, Alan Devol, Clara Fuchsman, Matthew Forbes, Calvin W Mordy, and Karen L Casciotti. Estimating fixed nitrogen loss and associated isotope effects using concentration and isotope measurements of NO3–, NO2–, and N2 from the eastern tropical south pacific oxygen deficient zone. *Deep Sea Res. Part 2 Top. Stud. Oceanogr.*, 156:121–136, 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.