SHDR User Manual

Manuel Franco Pire

May 12, 2023

Contents

1 Description

The SHDR algorithm (González-Pola et al., 2007) fits an upper oceanic profile to a physically inspired analytical form. This analytical form is obtained assuming the thermocline is a result of diffusive processes, and defines a constant region (mixed layer) . This manual aims to be a guide for using the codes in the repository. For further, please refer to the original work by by González-Pola et al.

SHDR stands for *Sharp Homogenization/Diffusive Retreat*. The thermocline is assumed as being a result of diffusive processes, with and the mixed layer depth (MLD) defining the boundary and the following analytical form can be found to describe an upper oceanic profile:

$$f(z) = \begin{cases} a_1 & \text{if } z < D_1, \\ a_3 + b_3(z - D_1) + a_2 e^{\left(-b_2(z-D_1) - c_2(z-D_1)^2\right)} & \text{if } z > D_1. \end{cases} \tag{1}$$

The first part of eq. 1 defines the mixed layer and the second the seasonal and permanent thermoclines. The seasonal thermocline is parameterized as a combination of exponential and gaussian decays. These decay to a straight line that describes the permanent thermocline. The different parameters in equation 1 are presented graphically in figure 1, and can be interpreted as:

- D_1 - Mixed layer depth.
- a_1 - Temperature of the mixed layer depth.
- b_2 - Exponential decay coefficient at the seasonal thermocline.
- c_2 - Gaussian decay coefficient at the seasonal thermocline.
- b_3 - Gradient of the permanent thermocline.
- a_2 - Temperature difference between the mixed layer and the base of the thermocline.
- a_3 - Temperature at the base of the thermocline.

The optimization method used by SHDR is a differential evolution algorithm. To find the combination of parameters that minimize the mean squared error (MSE), a population of individuals is randomly generated. Each individual represents a certain combination of the parameters in equation 1. By means of crossing and mutations, the "genetics" of the individuals that perform best are favored, and the MSE is progressively minimized. A description of the differential evolution algorithm as implemented in SHDR can be found in appendix A.

2 Using SHDR

The source code of SHDR is very simple. All the routines needed to use SHDR are included in the file `SHDR.py`. To use it, download it and place it in your working directory. This file is meant to be used as a module. It contains two main functions: `fit_profile` and `fit_time_series`. Using your preferred python programming environment, import them and pass your data to them.

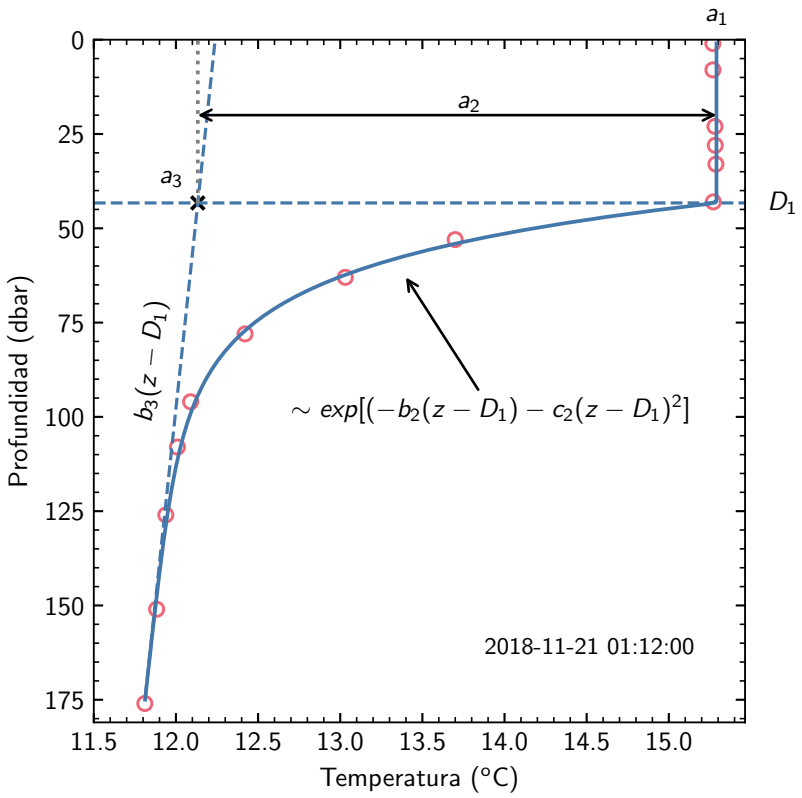

Figure 1: Caption

The examples folder in the code repository contains scripts and notebooks showcasing different use cases of the SHDR module. Here we will present the functions, their use case, arguments and output.

The main function in SHDR is `fit_profile`. It performs the differential evolution search to find the optimal parameters of equation 1 that minimize the RMS for a given profile:

`fit_profile(y, z, **opts)`.

- `y` - 1-D array defining the values of the profile.
- `z` - 1-D array defining the depths of the profile.
- `opts` - Optional arguments. Described bellow.

`fit_profile` returns an array containing the optimal parameters for the profile in the following order: [`D1, b2, c2, b3, a2, a1, a3, em`]. `em` is the RMS error of the fit.

`fit_profile` only fits a single profile. For datasets containing multiple profiles, iterate through them with `fit_profile` and store the results as convenient. For time series datasets, the `SHDR.py` contains a helper function to simplify this process (see section 2.2).

2.1 Optional arguments

The fitting algorithm has different parameters that can be passed as arguments to the `fit_profile` function. For most use cases,

- `max_depth`: Maximum depth in the profile to be considered for the fitting. Use this argument to specify the depth that you consider defines the upper ocean in your

dataset. Default value is 450 dbar.

- `min_depth`:
- `min_obs`: Minimum number of points in the profile to perform the fit.

For a more advanced usage, the parameters of the differential evolution algorithm can be modified. The default values should reproduce consistent results for most CTD profiles:

- `CR` - Cross probability. Default: 0.7
- `FF` - Mutation factor. Default: 0.7
- `num_generations` - Maximum number of generations. Default: 1200
- `num_individuals` - Number of individuals in each generation. Default: 60
- `tol` - tolerance to stop evolution. Default: 0.00025

2.2 Fitting a time series

As many datasets are organized as time series, the `SHDR` module includes a function to fit a time series. The function is called `fit_time_series`. Its optional parameters are the same as above. Its main parameters are:

```
fit_time_series(time, variable, depth, lat=None, lon=None, save=None,
                **opts)
```

- `time` - 1D Array defining the timestamps of the measurements
- `variable` - 2D Array containing the measurements. The first dimension of the array must be time, and the same length as `time`
- `depth` - 2D Array defining the depths of the measurements. The first dimension must be temporal.
- `lat` - 1D Array defining the latitude of each measurement.
- `lon` - 1D Array defining the longitude of each measurement.
- `save` - Defines a relative path to store the results as a `.csv` file. If this argument is not passed, the results won't be stored.

`fit_time_series` returns a Pandas DataFrame containing the results of the fit. The DataFrame columns are: [`time, lat, lon, D1, b2, c2, b3, a2, a1, a3, em`]. The stored `.csv` follows the same structure.

3 Exploring SHDR results

The code repository contains another file `SHDR_utils.py`. This file contains routines to analyze the output of SHDR:

- `fit_function(z, params)`. Returns the value at depth z of an analytical profile following equation 1 with parameters `params`. The `params` argument must be a `np.ndarray` containing the output of `fit_profile` or a `pd.Series` containing a specific profile fit of `fit_time_series`.
- `compute_stratification(params, alpha=0.05)`. Returns the G_α stratification index as defined in González-Pola et al. for a given combination of parameters.

- `time_series_stratification(time_series_fit, alpha=0.05)`. Wrapper to apply `compute_stratification` to a fitted time series. The `time_series_fit` argument can be a path where the `.csv` file of the fit is stored or the output of `time_series_fit`. In the first case, an extra column containing the stratification index is added to selected results file. Otherwise a `pd.Series` object is returned containing the stratification for each measurement.

References

González-Pola, C., Fernández-Díaz, J. M., & Lavín, A. (2007). Vertical structure of the upper ocean from profiles fitted to physically consistent functional forms. *Deep Sea Research Part I: Oceanographic Research Papers*, *54*(11), 1985–2004. https://doi.org/10.1016/j.dsr.2007.08.007

A Differential evolution algorithm

The differential evolution algorithm implemented in SHDR is as follows following. Firstly, an initial population is generated randomly following a uniform distribution inside the limits for each parameter. Each generation g evolves according to the following algorithm:

1. The MSE is computed for each individual, and the best is detected, $\mathbf{x}_{b,g}$.
2. For each individual $\mathbf{x}_{i,g}$, two individuals $\mathbf{x}_{k_1,g}, \mathbf{x}_{k_2,g}$ are selected randomly with $i \neq k_1 \neq k_2$, and its mutation $\mathbf{y}_i$ is computed:

$$\mathbf{y}_{i,g} = (1-\mu)\mathbf{x}_{i,g} + \mu\mathbf{x}_{b,g} + F(\mathbf{x}_{k_1,g} - \mathbf{x}_{k_2,g}), \tag{2}$$

   where $FF \in (0,1)$ is the mutation factor and $\mu \in (0,1)$ the weight of the best individual.
3. In the next generation, the individual will evolve from $\mathbf{x}$ a $\mathbf{y}$ following

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{y}_{i,g} & \text{if } r_i < CR, \\ \mathbf{x}_{i,g} & \text{otherwise,} \end{cases} \tag{3}$$

   where $CR \in (0,1)$ is the cross probability $r_i \sim U[0,1)$.
4. If a parameter of a new individual excedes the limits, it is returned to the closest limit.
5. If $g+1 = g_{\max}$ with $g_{\max}$ the maximun number of generation, or if the mean $\overline{em} < \text{tol} \cdot \sigma_{em}$, where e_m is the MSE, σ_{em} is the standard deviation of the population MSE, and tol is a tolerance previously defined, the interation is halted and the best individual of the new generation $g+1$ is taken as solution. Otherwise, ..