iLOSCAR

Update time: 2024/01/22

A web-based interactive carbon cycle model, built upon the classic LOSCAR model. Forward and inverse model are included.

When using iLOSCAR, cite as:

Zeebe, R.E., 2012. LOSCAR: Long-term ocean-atmosphere-sediment carbon cycle reservoir model v2.0.4. Geoscientific Model
Development 5, 149–166.
Li, et al., in review, iLOSCAR: interactive Long-term Ocean-atmosphere-Sediment CArbon cycle Reservoir Model v1.0.

Author
Shihan Li | Department of Oceanography, Texas A&M University, College Station, Texas, 77843, USA

Contributors
Dr. Richard E. Zeebe | Department of Oceanography, University of Hawaii, Manoa Honolulu, Hawaii, 96822, USA
Dr. Shuang Zhang | Department of Oceanography, Texas A&M University, College Station, Texas, 77843, USA

For any questions, please contact shihan@tamu.edu

File structure

```
├── dat
├── iLOSCAR
├── Anaconda_install.md
├── README.md
├── iLOSCAR_tutorial.pdf
├── iloscar_mac.yml
├── iloscar_win.yml
```

The *dat folder* contains the data that can be used to replicate the experiments in our GPC paper.

The *iLOSCAR folder* contains the main functions of our model, which will be introduced later in this tutorial.

The files *Anaconda_install.md*, *README.md*, and *iLOSCAR_tutorial.pdf* contain comprehensive tutorials on installing the model, setting up and running experiments, and troubleshooting common issues.

The files *iloscar_mac.yml* and *iloscar_win.yml* are used to create a virtual environment using Anaconda for running the model.

Install

To successfully install iLOSCAR, please follow the tutorial provided.

To avoid the potential Python package inconsistency, we highly recommend downloading the code directly from https://github.com/Shihan150/iloscar (see below) and setting up an Anaconda virtual environment before running iLOSCAR.
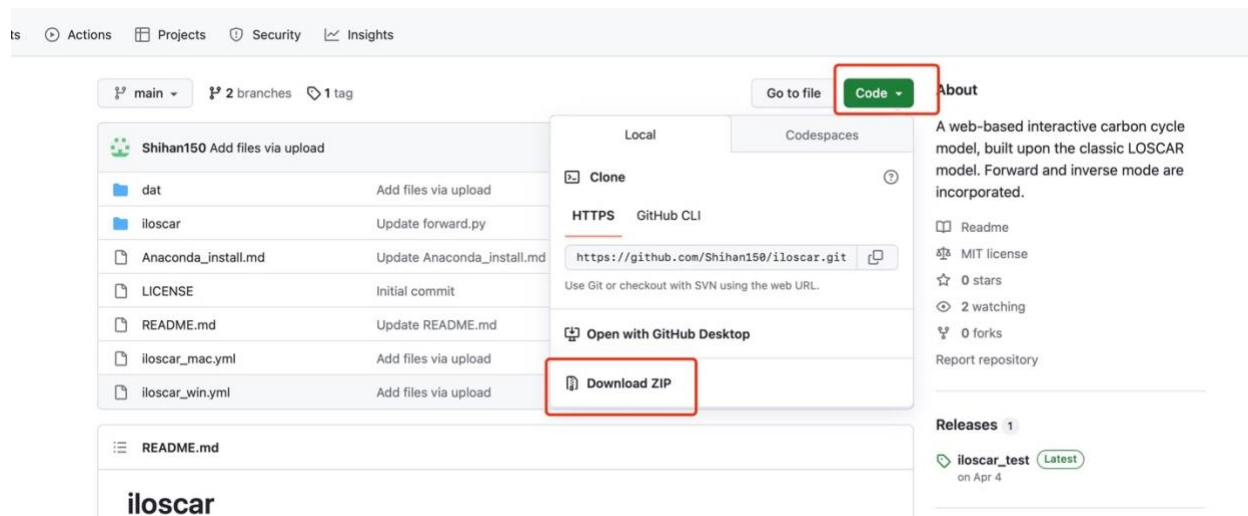
0. Anaconda install

Please refer to the Anaconda_install.md file for detailed instructions on installing Anaconda. Two options including Anaconda Distribution and Miniconda are provided in the tutorial. If you already have Anaconda installed, you can proceed to the next step.

1. Create a virtual environment and run the model

Mac system

1. Open the Terminal and go to the iloscar main directory downloaded in the previous step. One example is shown below and you need to specify your own path (the part after "cd"). You can learn more about navigating files and directories here.

2. Type ***conda env create -f iloscar_mac.yml*** to install the iloscar environment. It may take ~1 - 5 min.

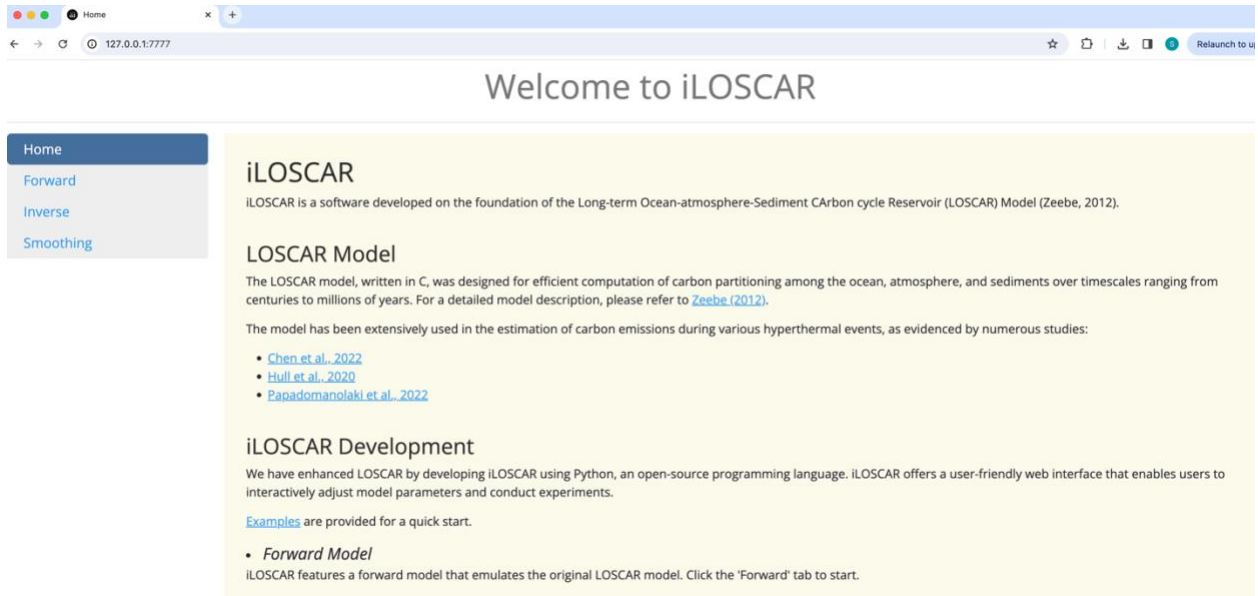
3. Type ***conda activate iloscar***

4. Go to the iloscar coding directory by typing *cd iloscar*

```
(iloscar) shihan@C02F316GMD6V iloscar-main % cd iloscar
```

5. Type *python app.py* and open http://127.0.0.1:7777/ in your browser to run the model. It may take several to tens of seconds, depending on your machine.

```
(iloscar) shihan@0M629-99226 iloscar % python app.py
Dash is running on http://127.0.0.1:7777/

* Serving Flask app 'app'
* Debug mode: on
```
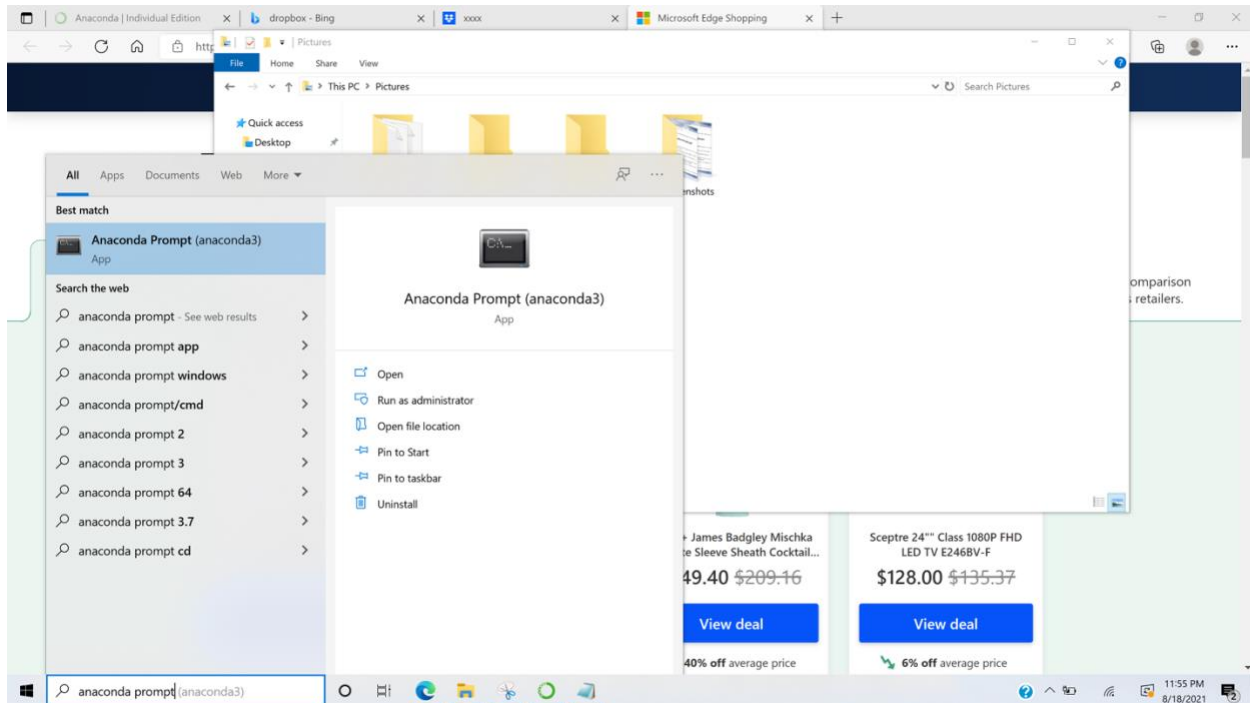

Welcome to iLOSCAR

Home
Forward
Inverse
Smoothing

iLOSCAR
iLOSCAR is a software developed on the foundation of the Long-term Ocean-atmosphere-Sediment CArbon cycle Reservoir (LOSCAR) Model (Zeebe, 2012).

LOSCAR Model
The LOSCAR model, written in C, was designed for efficient computation of carbon partitioning among the ocean, atmosphere, and sediments over timescales ranging from centuries to millions of years. For a detailed model description, please refer to Zeebe (2012).

The model has been extensively used in the estimation of carbon emissions during various hyperthermal events, as evidenced by numerous studies:

- Chen et al., 2022
- Hull et al., 2020
- Papadomanolaki et al., 2022

iLOSCAR Development
We have enhanced LOSCAR by developing iLOSCAR using Python, an open-source programming language. iLOSCAR offers a user-friendly web interface that enables users to interactively adjust model parameters and conduct experiments.

Examples are provided for a quick start.

- *Forward Model*
iLOSCAR features a forward model that emulates the original LOSCAR model. Click the 'Forward' tab to start.

Success!

Windows 10

1. Open the start menu and look for **Anaconda Prompt**.

2. Go to the iloscar main directory downloaded in the previous step. One example is shown below by typing **cd C:\Users\kaway\iloscar-main**. You need to specify your own path.

3. Type **conda env create -f iloscar_win.yml** to install the iloscar environment. It may take ~1-2 mins.

```
(base) C:\Users\kaway\iloscar-main>conda env create -f iloscar_win.yml
```

4. Type **conda activate iloscar**

```
(base) C:\Users\kaway\iloscar-main>conda activate iloscar

(iloscar) C:\Users\kaway\iloscar-main>cd iloscar
```

5. Type **python app.py** and open http://127.0.0.1:7777/ in your browser to run the model. It may take several to tens of seconds, depending on your machine.

```
(iloscar) C:\Users\kaway\iloscar-main\iloscar>python app.py
Dash is running on http://127.0.0.1:7777/

 * Serving Flask app 'app'
 * Debug mode: on
```

Success!

Model description

For details iLOSCAR, including the relevant processes, the physical meanings of parameters, model structure, and derivation of equations, please refer to our paper (in review) and [Zeebe, 2012, GMD](#).

Model functions

Forward model

In the forward mode, a specific emission trajectory is applied as the forcing and the model will return the temporal evolution of various parameters in the global carbon cycle (see Output files section). The core part is to solve the Ordinary Differential Equation (ODE) problem:

$$\frac{d\vec{y}}{dt} = F(t, \vec{y})$$

This is the ODE system that governs the dynamic changes of the model's state variables over time, where t is time, $\vec{y}$ is the vector containing various biogeochemical tracers, and F is the function used to calculate the derivatives of the state variables $\vec{y}$.

Inverse model

Five inversion options are provided in the inverse model. The aim of the inverse model is to derive the time-dependent carbon emission scenario ($fcinp(t)$) and the isotopic composition of emitted carbon ($f\delta^{13}C(t)$) by minimizing the relative errors between observations and corresponding modeling results:

$$fcinp(t), f\delta^{13}C(t) = argmin \sum_{i=1}^{n} \left| \frac{x_{model}(t_i) - x_{obs}(t_i)}{x_{obs}(t_i)} \right|$$

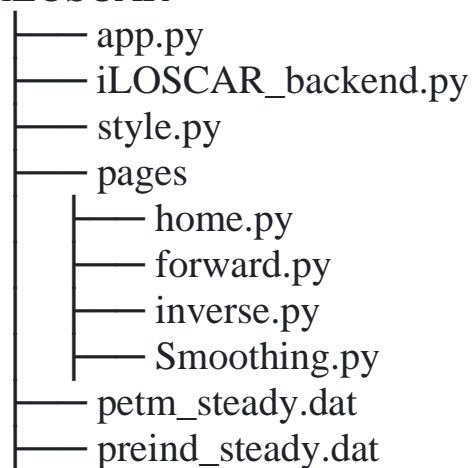| Option	Input	Output
pCO2	pCO2 proxy records	fcinp(t)
d13c	d13c proxy records	fcinp(t), needs to assume a constant d13c for the emission
GSpH	Global surface pH records	fcinp(t)

Option	Input	Output
pCO2_d13c	pCO2 and mean surface d13C proxy records	fcinp(t) + fd13C(t)
pH_d13c	pH and mean surface d13C proxy records	fcinp(t) + fd13C(t)

Smoothing function

A LOWESS smoothing function is available within the module. Users have the flexibility to upload data files and manually adjust the hyperparameters that control the fraction of the data used when estimating each y-value in LOWESS algorithm. Note that the default temporal resolution for output data is 0.2 kyr. For a comprehensive explanation of the smoothing algorithm, please refer to the following link: LOWESS Smoothing Algorithm.

Model structure

```
iLOSCAR
├── app.py
├── iLOSCAR_backend.py
├── style.py
├── pages
│   ├── home.py
│   ├── forward.py
│   ├── inverse.py
│   └── Smoothing.py
├── petm_steady.dat
├── preind_steady.dat
```

Front-end

The web-based interface in iLOSCAR is developed upon the Dash Package, which provides a low-code framework for rapidly building data apps in Python. The app.py is used to activate the model and the interface. The contents of the interface are controlled by the .py files in the 'pages' folder and their format is controlled by the style.py.

Back-end

The iLOSCAR_backen.py contains all the core functions to run the model, including setting up the model parameters and ODE functions, solutions for both forward and inverse models, saving the experiment results, as well as some tool functions such as solving the carbonate system from alkalinity (ALK) and dissolved inorganic carbon (DIC).

Initial y file

Two files (**petm_steady.dat** and **preind_steady.dat**) are provided, which are the initial state files from the original LOSCAR default settings.

For each experiment, users can update parameters from the front-end, which will be transferred to call the backend functions.

Numeric algorithms

ODE solver for the forward model

The LSODA (an acronym for Livermore Solver for Ordinary Differential equations, with Automatic method switching for stiff and nonstiff problems) algorithm is employed as the ODE solver, given its demonstrated stability when dealing with stiff problems, as highlighted by [Hindmarsh (1992)](). The algorithm is available in the [Python Scipy Package]().

Users are encouraged to explore the performance of various ODE solvers. For example, the Backward Differentiation Formula (BDF) also exhibits comparable efficiency for stiff problems. To change the solver, users can navigate to Line 53 in the ./iloscar/iLOSCAR_backend.py file and uncomment the desired solver they wish to employ.

TOMS 748, a Root-finding algorithm, for the inverse model

To solve the inverse problem, the TOMS 748 root-finding algorithm is applied, which uses a mixture of inverse cubic interpolation and Newton-quadratic steps to enclose zeros of contiguous univariate functions ([Alefeld et al., 1995]()). This algorithm offers the advantage of a rapid convergence rate, which significantly accelerates the inversion process. Additionally, Algorithm 748 is readily available in the [Python Scipy package](). Note that two parameters (a, b) need to be given to determine the boundaries of the algorithm search interval, i.e., $f(a) \times f(b) < 0$, where f is the function $\frac{x_{model}(t_i) - x_{obs}(t_i)}{x_{obs}(t_i)}$. In our context, a, b represent the potential minimum and maximum emission rates (in Gt), respectively. When a is negative, it represents the carbon burial rate. Default settings are -0.1 and 2, which should work for most applications. Increasing the absolute value can reduce the failure probability of the experiment but at the expense of running time. Therefore, it's important for users to carefully select these values based on their domain knowledge. Careful consideration in this regard will help optimize the model's performance, ensuring a balance between accuracy and computational efficiency.

External input files

Some external files are required to run the model.

Mode	File usage	Format	Requirement
Forward	Initial y0	.dat	1 column, 140 (for modern) or 184 rows (for paleo); y0 satisfies dy0/dt = F(t=t0, y0) = 0
	Emission file	.dat	When LOADFLAG == 2, two columns (age (yr) + emission mass (Gt/yr)). When LOADFLAG == 3, three columns (age (yr) + emission mass (Gt/yr)+d13c of input (per mil)).
	Save yfinal	.dat	When auto-spinning the model to the steady state (i.e., 'Save ystart' in the Table == 1), y(t=tfinal) will be saved into the specified .dat file
Inverse	pCO2 data for inversion	.csv	2 columns with headline, age (yr) + pCO2 (ppmv)
	mean surface pH data for inversion	.csv	2 columns with headline, age (yr) + pH
	mean surface d13c data for inversion	.csv	2 columns with headline, age (yr) + d13c (per mil)
Function	data to be smoothed	.csv	2 columns with headline, age (yr) + data

Output files

In each experiment, the model will output the following data files.

File.csv	Unit	Variable
tcb	(deg C)	Ocean (OCN) temperature
dic	(mmol/kg)	OCN total dissolved inorganic carbon

File.csv	Unit	Variable
alk	(mmol/kg)	OCN total alkalinity
po4	(umol/kg)	OCN phosphate
dox	(mol/m3)	OCN dissolved oxygen
dicc	(mmol/kg)	OCN DIC-13
d13c	(per mil)	OCN delta13C(DIC)
pco2a_d13c	(ppmv, per mil)	Atmospheric pCO2 and d13c
co3	(umol/kg)	OCN carbonate ion concentration
ph	(-)	OCN pH (total scale)
pco2ocn	(uatm)	OCN ocean pCO2
omegaclc	(-)	OCN calcite saturation state
omegaarg	(-)	OCN aragonite saturation state
fca	weight %	Sediment (SED) calcite content of Atlantic Ocean
fci	weight %	SED calcite content of Indian Ocean
fcp	weight %	SED calcite content of Pacific Ocean
fct	weight %	SED calcite content Tethys (PALEO model version only)
ccda	(m)	SED calcite compensation depth Atlantic

File.csv	Unit	Variable
ccdi	(m)	SED calcite compensation depth Indian
ccdp	(m)	SED calcite compensation depth Pacific
ccdt	(m)	SED calcite compensation depth Tethys (PALEO model version only)
Surface_dic_alk_d13c_ph	(mmol/kg, mmol/kg, per mil, -)	Mean OCN surface DIC, ALK, d13c, and pH
Carbon_inventory	(mol)	Total carbon and alkalinity in the ocean

Example

(Example data files could be downloaded here).

For all tables, the first column can be adjusted manually.

1. Forward model example

The general workflow is as follows:
1. Select the desired version in Step 1 table;
2. Tune relevant parameters in Step 2 table and turn off the carbon emission in Step 3 table;
3. Change the t0 and tfinal in Step 3 table and spin up the model for 2e7 years. Check if the steady state is achieved;
4. Utilize the final steady state achieved in the previous step as the initial condition (y0). Enable the carbon emissions in Step 3 table and run the model.

To assist users in becoming familiar withthe process of running iLOSCAR in a forward manner, an example is provided.

1.1 Original PETM example from Zeebe et al., 2009.

Please note that this specific implementation does not include the prolonged carbon release following the main emission event or the inferred reverse circulation adopted in the original study by Zeebe et al., 2009.

Tuning the steady state

Please note that if you intend to run the default model for the modern and paleo setup, you can skip this part as the initial steady-state y0 values are already provided in our package (preind_steady.dat and petm_steady.dat), which can be used directly.

1. Go to the Forward page
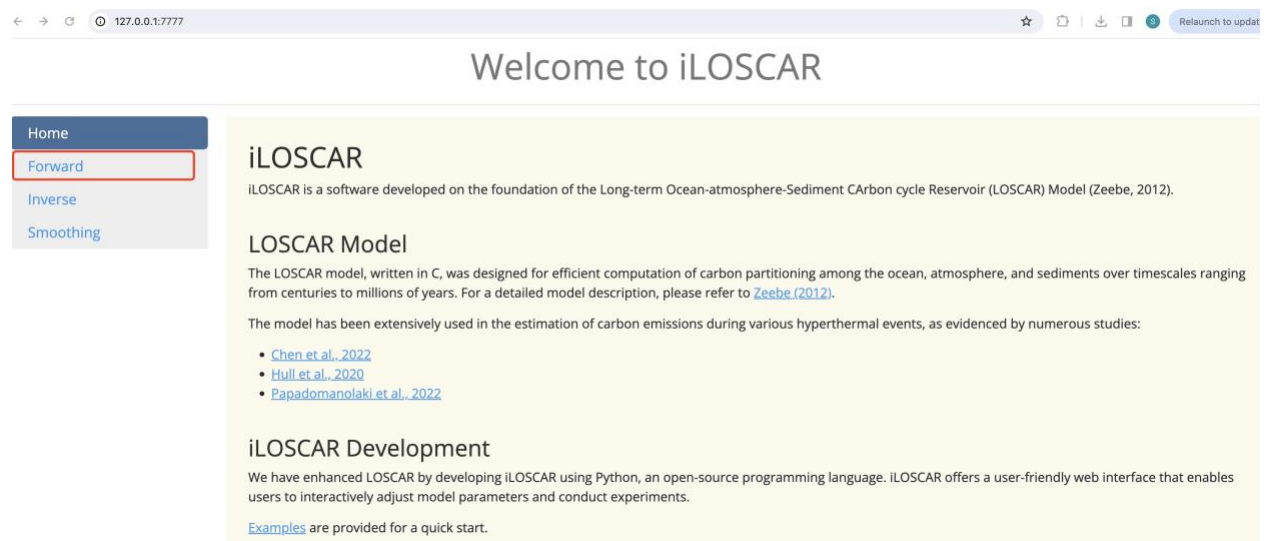
2. In Step 1 table, set the PALEO parameter to '1', LOADFLAG to '0', and Save ystart to '1'. The model parameters in Step 2 table will adjust automatically to the palaeo settings. Save ystart determines if the model will export the y values at t=tfinal. The export file name can be manually specified in Step 4 table. In this example, we use the 'petm_steady.dat'.

Step 1: select the experiment version

Value	Parameter	Options	Comment
1	PALEO	1/0	1: Paleo setup; 0: Modern setup
1	Sediment	1/0	1: Sediment box on; 0: sediment box off
0	LOADFLAG	1/0	1: load initial settings from an external file; 0: off
1	Ocaen temperature change	1/0	1: Ocean temperature change (co2-sensitivity) ON 0: Off
1	Save ystart	1/0	1: the experiment aim is to autospin to the steady state and the system variables at t=tfinal will be saved 0: off.

3. Turn off carbon emissions by changing the 'emission pattern' to 0 in Step 3 table.

Step 3: select the carbon emission scenario

Value	Parameter	Unit	Comment
0	emission pattern	3/2/1/0	3: emission scenario with time-dependant d13c 2: emission scenario from the external files 1:carbon emission in uniform distribution 0: no carbon emission

4. Modify 'tfinal' to '2e7' in Step 2 table.

Step 2: select the model parameters

Value	Parameter	Unit	Comment
0	t0	yr	start time (yr) for the experiment
20000000	tfinal	yr	end time (yr) for the experiment

5. Provide a name for your experiment and run it. I name it Zeebe2009 here.

Experiment name

Zeebe2009

The modeling results will be saved in a dictionary

after experiment name

Run

Clean the output

6. The running information will be displayed in the following chunk.

```
@ This is the PALEO setup including Tethys
@ Default initial values are used
@ The carbon injection is OFF


@ The experiment name : Zeebe2009
@ No. ocean basins : 4
@ No. ocean boxes : 13
@ No. ocean tracers : 6
@ Atmospheric Carbon : 1
@ Atmospheric Carbon-13 : 1
@ No. sediment depth levels: 13
@ No. sediment C-13 levels : 13
@ No. equations : 184



@ Starting integration
[tstart tfinal]=[0.00e+00 2.00e+05]

Progressing...
```

7. Once the integration is complete, the final steady state will be saved to the file specified in Step 4 table ('petm_steady.dat' in this case). The exported file can be used as the initial y0 for perturbation experiments later.

```
Integration finished.

46.49s used.

Starting to save the modeling results.

System variables at t=2.00e+05 has been saved to ./petm_steady.dat
```

Perturbation experiment

The above steps are to set up the initial steady state before a geologic event. Now we will simulate the perturbation driven by the carbon injection.

1. Stay on the same page. In Step 1 table, set LOADFALG to '1' and Save ystart to '0'.

Step 1: select the experiment version

Value	Parameter	Options	Comment
1	PALEO	1/0	1: Paleo setup; 0: Modern setup
1	Sediment	1/0	1: Sediment box on; 0: sediment box off
1	LOADFLAG	1/0	1: load initial settings from an external file; 0: off
1	Ocaen temperature change	1/0	1: Ocean temperature change (co2-sensitivity) ON 0: Off
0	Save ystart	1/0	1: the experiment aim is to autospin to the steady state and the system variables at t=tfinal will be saved 0: off.

2. In Step 2 table, change 'tfinal' to '2e5'.

Step 2: select the model parameters

Value	Parameter	Unit	Comment
0	t0	yr	start time (yr) for the experiment
200000	tfinal	yr	end time (yr) for the experiment

3. Select the carbon emission scenario in Step 3 table. In this example, set: 'emission pattern' == 1, 'emission amount' == 3000, 'd13c emission' == -55, 'emission start' == 0, 'emission duration' == 6000.

Step 3: select the carbon emission scenario

Value	Parameter	Unit	Comment
1	emission pattern	3/2/1/0	3: emission scenario with time-dependant d13c 2: emission scenario from the external files 1:carbon emission in uniform distribution 0: no carbon emission
3000	emission amount	Gt	total amount of carbon input, only useful when "emission pattern" == 1
-55	d13c emission	per mil	d13c of input carbon, only useful when "emission pattern" == 1
0	emission start	yr	start year for the emission, only useful when "emission pattern" == 1
6000	emission duration	yr	duration for the emission, only useful when "emission pattern" == 1

4. Run the model. Optionally, you can click the 'Clean the output' button to clear the experiment information from the previous run. However, this step is optional, You can directly create a new perturbation scenario with the same initial state and run the next experiment.

Experiment name

Zeebe2009

The modeling results will be saved in a dictionary
after experiment name

Run

Optional → Clean the output

@ This is the PALEO setup including Tethys
@ Loaded initial values are used
@ The carbon injection is ON

Help you check the experiment information

5. Once the integration is complete, the modeling results will be saved in the exp_name folder (e.g., 'Zeebe2009'). The folder will be located in the same directory where you ran your Python code. Modeling mean surface DIC, ALK, pH and d13c, pCO2, and CCD for each ocean basin will be displayed when integration succeeds.

Integration finished.

10.17s used.

It is faster than the previous steady state run. It is because we use Numba to compile our code, which may take 15-30 seconds in different machines. In this run, the compilation results from the previous run can be used directly, thus taking less time .

Starting to save the modeling results.

Modeling results have been saved to Zeebe2009 folder.

The experiment succeeds. Congratulations!

The final average surface pH is: 7.61

The final average surface d13c is: 1.83

The final pCO2 is: 1247.31
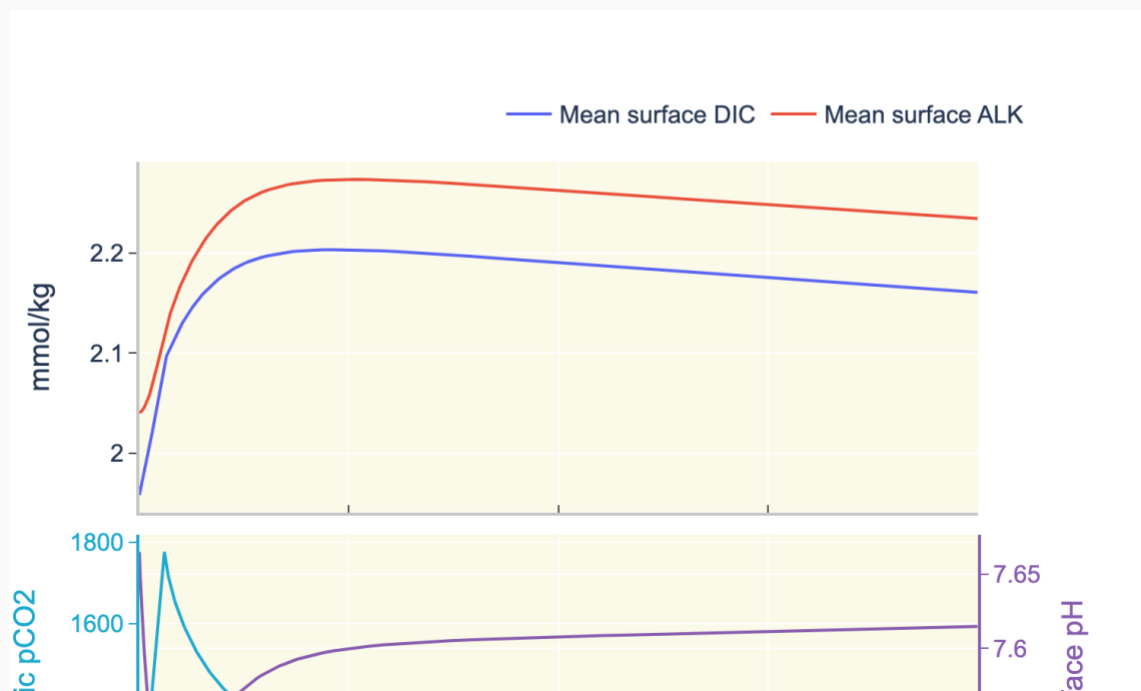
2. Inverse model example

The general workflow for the inverse model is as follows:

1. Tune the parameters and obtain the initial steady state y0 using the forward model page.
2. Go to the inverse model page and specify the file names that contain the proxy records.
3. Adjust the parameters based on the tuning results obtained from the forward model.
4. Run the inverse model.

2.1 Inverse twin experiment

To evaluate the performance of the inverse algorithm, an identical twin test can be conducted. In this type of test, a preliminary run of the forward model is performed to generate a synthetic 'truth' dataset that can be used for subsequent inversion experiments. This allows for a straightforward assessment of the accuracy of the inverse algorithm.

2.1.1. Preliminary run

- In the Forward page, maintain all the default settings in Step 1 table and 2, except for setting 'tfinal' to '4e4'.

- Set the 'emission pattern' to 3 in Step 3 table, and input 'pulse_emi.dat' in the second row of Step 4 table. Note that the file name should be provided as a relative path.

- 'pulse_emi.dat' file represents two emission events:
  a fast and short event (3000 Gt in 3 kyr) and a slow and long event (10000 Gt in 35 kyr). This dataset serves as an excellent example for checking the performance of the inversion algorithm.

- Provide a name for the experiment and run it.

Step 3: select the carbon emission scenario

Value	Parameter	Unit	Comment
3	emission pattern	3/2/1/0	3: emission scenario with time-dependant d13c 2: emission scenario from the external files 1:carbon emission in uniform distribution 0: no carbon emission
1000	emission amount	Gt	total amount of carbon input, only useful when "emission pattern" == 1
-55	d13c emission	per mil	d13c of input carbon, only useful when "emission pattern" == 1
0	emission start	yr	start year for the emission, only useful when "emission pattern" == 1
3000	emission duration	yr	duration for the emission, only useful when "emission pattern" == 1

Step 4: select required external file, path information included

| File name	Comment
./preind_steady.dat	initial steady state file, required only when LOADFLAG == 1 in Step 1
../dat/pulse_emi.dat	time-resolved carbon input file, required only when "emission pattern" == 2 or 3 in Step 3
./preind_steady.dat	file to save the y when t = tfinal, required only when Save ystart == 1

Experiment name

twin_exp

The modeling results will be saved in a dictionary after experiment name

Run

Clean the output

2.1.2. Prepare data for inversion

- Navigate to the twin_exp folder.
- Locate the pCO2 results in the pCO2_d13c.csv file.
- Select the desired pCO2 values and save them as twin_pco2_for_inv.csv.
- Keep in mind that the forward modeling results may have a high temporal resolution (thus a lot of data points), so it is recommended to select a subset of data to ensure reasonable inversion times.
- Repeat the same process for the mean surface pH and surface d13C results. Select the relevant values and save them accordingly for use in the inversion experiment.

2.1.3. Inverse experiment

- Go to the Inverse page.
  From the dropdown menu, select 'pCO2 + mean surface d13c'. Manually input the target file names.
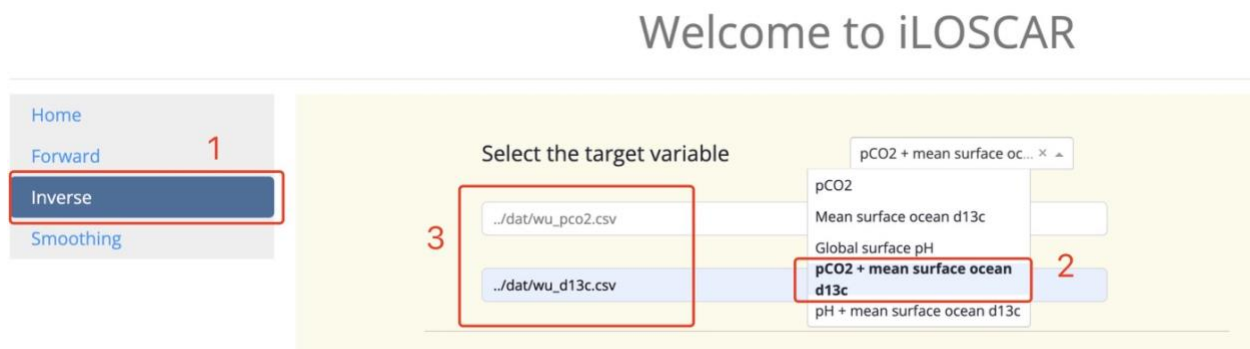
- In Step 3 table, specify the boundary values for the Toms748 root-finding algorithm. These values represent the expected minimum and maximum degassing rates. The narrower the range, the faster the experiment will run, but there is a higher chance of failure. The default values of -0.1 and 2 Gt/yr should be suitable for most application.

Step 3: choose the carbon emission scenario

Value	Parameter	Unit	Comment
-55	d13c emission	per mil	d13c of input carbon, required for the single version
-0.1	Lower boundary in bracket for toms748 method. Value equivalent to expected minimum carbon degassing rate. Negative values represent organic carbon burial	Gt/yr	Default settings can work for most applications. Adjust the value for extreme case
2	Higher boundary in bracket for toms748 method. Value equivalent to maximum degassing rate	Gt/yr	Increase the absolute value can reduce the failure probability of experiment, but at the expense of computation time

- Provide a name for the experiment and run the model.

- If you need to terminate the ongoing experiment, simply click the 'Cancel' button.

Experiment name

twin_inverse

The modeling results will be saved in a dictionary after experiment name

Run

Cancel

Clean the output

- If you encounter an error similar to the figure below, it means that some degassing rates exceed the default upper boundary in Step 3 table. Adjust the values in the second and third rows of Step 3 table accordingly.

🔴 Callback error updating ..info_integration_inv.... 2:22:13 PM ⌄

```
Callback error updating
..info_integration_inv.children...ysol_inv.data...tsol_inv.data..

line 358, in model_run
ems_new, results = toms748(cost_function, toms_low , toms_high,
File "/Users/shihan/opt/anaconda3/envs/iloscar/lib/python3.9/site-
packages/scipy/optimize/_zeros_py.py", line 1374, in toms748
result = solver.solve(f, a, b, args=args, k=k, xtol=xtol, rtol=rtol,
File "/Users/shihan/opt/anaconda3/envs/iloscar/lib/python3.9/site-
packages/scipy/optimize/_zeros_py.py", line 1221, in solve
status, xn = self.start(f, a, b, args)
File "/Users/shihan/opt/anaconda3/envs/iloscar/lib/python3.9/site-
packages/scipy/optimize/_zeros_py.py", line 1121, in start
raise ValueError("a, b must bracket a root f(%e)=%e, f(%e)=%e " %
ValueError: a, b must bracket a root f(-2.000000e+00)=-5.335239e+02,
f(5.000000e+00)=-4.983734e+01
```

- Once the inverse experiment is successful, the results will be displayed.

Progressing...

450.16s used.

The calculated emission scenario has been saved to double_inversion_emission.csv and
double_inversion_emission_d13c.csv

Starting to save the modeling results.

Modeling results have been saved to twin_inverse folder.   Inversed emission trajectory
                                                                  is saved here
The experiment succeeds. Congratulations!

The final average surface pH is: 7.79

The final average surface d13c is: -0.12

The final pCO2 is: 1173.31
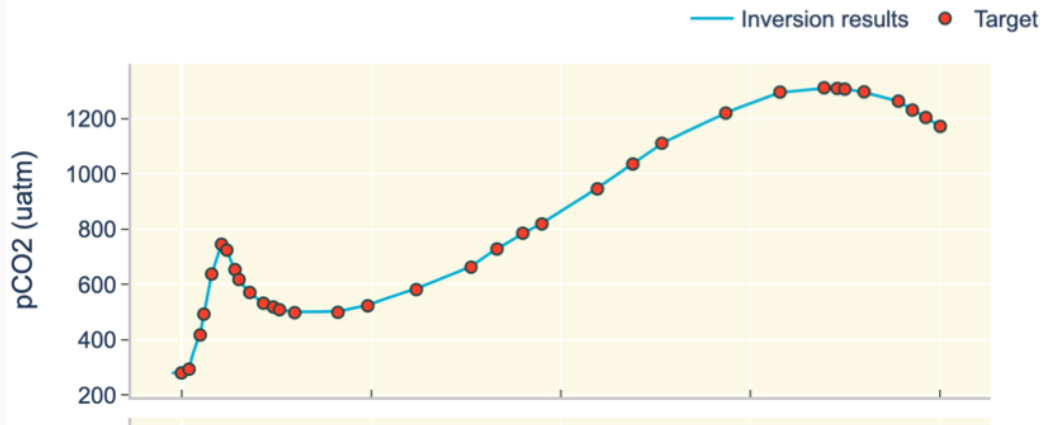
2.2 PETM experiment after [Gutjahr et al., 2017](#)

1. Derive the steady state y0

- Go to the Forward page.
- In Step 1 table, set PALEO == 1, LOADFLAG == 0, Save ystart == 1
- In Step 2 table, set tfinal == 1e7, pCO2_ref == 834, pCO2_initial == 834, silicate
  weathering0 = 7.5, carbonate weathering0 = 17.5, d13c volcanic == -1.5
- In Step 4 table, input './gutjahr2017.dat' in the third row.
- Provide a name for the experiment and run the model.

2. Inversion experiment

- Go to the Inverse page.
- Download the 'Gutjahr_pH.csv' and 'Gutjahr_d13c.csv' from the [link](#).
- In Step 1 table, set PALEO == 1, LOADFLAG == 1
- In Step 2 table, set pCO2_ref == 834, pCO2_initial == 834, silicate weathering0 = 7.5, carbonate weathering0 = 17.5, d13c volcanic == -1.5
- In Step 4 table, input './gutjahr2017.dat'
- Provide a name for the experiment and run the model.
- Success!

@ Starting inversion

Progressing...

838.00s used.

The calculated emission scenario has been saved to double_inversion_emission_pH.csv and double_inversion_emission_d13c_pH.csv
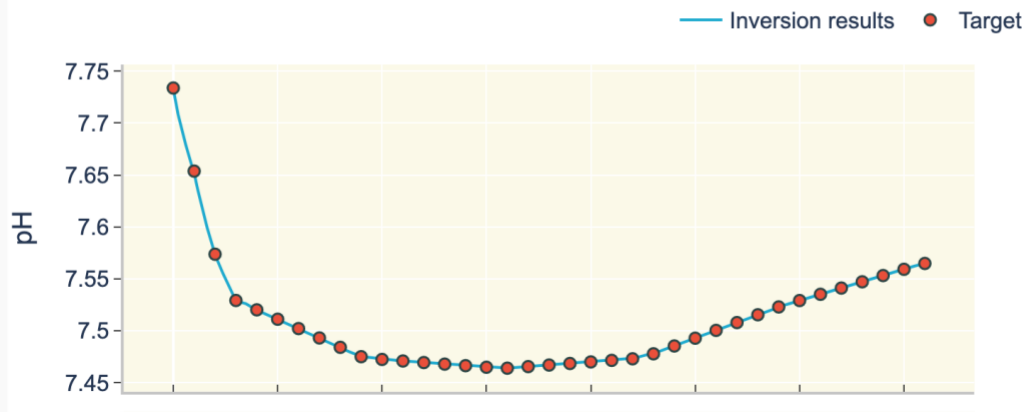
Starting to save the modeling results.

Modeling results have been saved to gutjahr_test folder.

The experiment succeeds. Congratulations!

The final average surface pH is: 7.56

The final average surface d13c is: 1.35

The final pCO2 is: 1787.31

2.3 PTB experiment after [Wu et al., 2023](#)

1. Derive the steady state y0

- Go to the Forward page.
- In Step 1 table, set PALEO == 0, LOADFLAG == 0, Save ystart == 1
- In Step 2 table, set tfinal == 1e7, pCO2_ref == 425, pCO2_initial == 449, , fsh == 5, silicate weathering0 = 12, carbonate weathering0 = 17, d13c volcanic == -1.3, ca concentration == 0.013, mg concentration == 0.042
- In Step 4 table, input './wu2023.dat' into the third row.
- Provide a name for the experiment and run the model.

2. Inversion experiment

- Go to the Inverse page.
- Download the 'wu_pco2.csv' and 'wu_d13c.csv' from the [link](#).
- In Step 1 table, set PALEO == 0, LOADFLAG == 1
- In Step 2 table, set pCO2_ref == 425, pCO2_initial == 449, fsh == 5, silicate weathering0 = 12, carbonate weathering0 = 17, d13c volcanic == -1.3, ca concentration == 0.013, mg concentration == 0.042, nsi == 0.4, ncc == 0.4
- In Step 3 table, set lower and higher boundaries as [-0.1, 1], which will accelerate the model
- In Step 4 table, input './wu2023.dat'
- Provide a name for the experiment and run the model.
- Success!
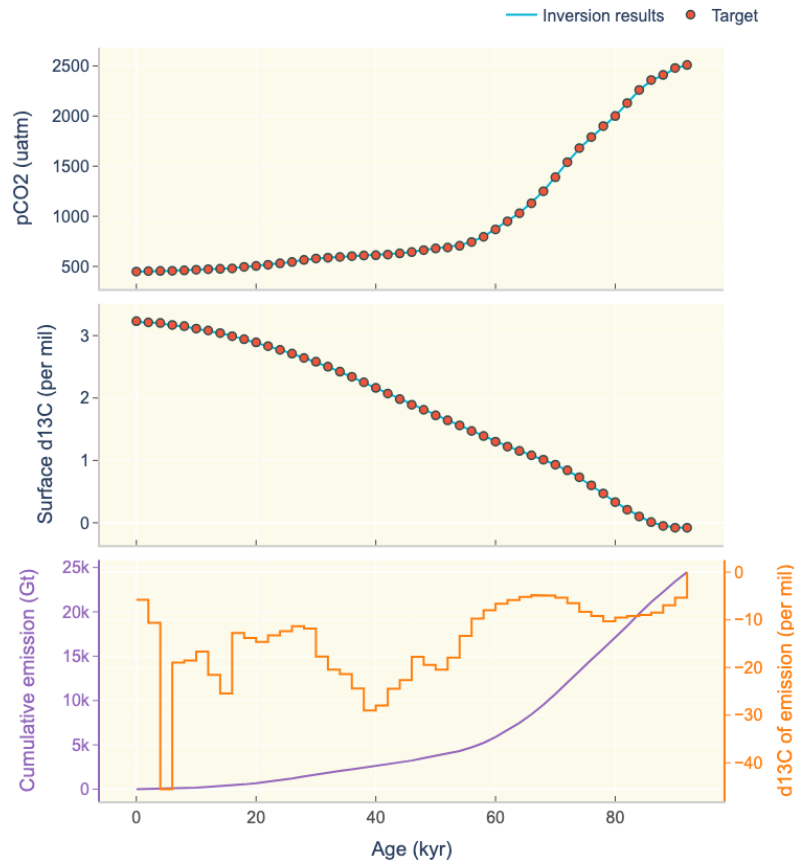
Benchmark

We welcome contributions from various operation systems and processors.

Experiment	OS	Processor	Time (seconds)	Source
1.1 Paleo steady state tuning	Mac	Apple M1 8 cores @3.2 GHz	46	This tutorial
1.1 Paleo steady state tuning	Windows 11 Pro x64	Intel Core i7-9700 CPU @ 3GHz with 8 cores	61	User
1.1 Paleo perturbation	Mac	Apple M1 8 cores @3.2 GHz	10	This tutorial

Experiment	OS	Processor	Time (seconds)	Source
1.1 Paelo perturbation	Windows 11 Pro x64	Intel Core i7-9700 CPU @ 3GHz with 8 cores	3	User
2.2 PETM Gutjahr2017	Mac	Apple M1 8 cores @3.2 GHz	838	This tutorial
2.2 PETM Gutjahr2017	Windows 11 Pro x64	Intel Core i7-9700 CPU @ 3GHz with 8 cores	760	User
2.3 PTB Wu2023	Mac	Apple M1 8 cores @3.2 GHz	816	This tutorial

Troubleshooting

This section provides a summary of common issues that may arise while running the iLOSCAR model. It's important to note that this is not an exhaustive list and may be supplemented with additional information in the future. If the troubleshooting strategies outlined here do not resolve your problems, consider reaching out to shihan@tamu.edu for further assistance and discussion.

No response from the forward model

Normally, a forward experiment should be completed within 2 to 3 minutes, depending on your machine. However, if the experiment takes an excessively long time without yielding results, it could be due to an inappropriate selection of certain parameters. This improper parameter choice might cause instability in the model and disrupt the solution process. For instance, altering the 'fsh' parameter to 10 in the default paleo settings can prevent the model from reaching a steady state. In such situations, follow these steps for troubleshooting:

i. Write down the parameter settings used in the current experiment.

ii. Interrupt the ongoing experiment by pressing Ctrl + C in the Terminal window.

iii. Restart the model by entering 'python app.py' in the Terminal window.

iv. Upon restart, the model parameters will revert to their default settings. Instead of applying all the previous parameters at once, adjust them individually. This step-by-step approach should help identify which specific parameter is causing instability.

Inversion algorithm fails to converge

If the inversion algorithm does not converge and you encounter an error message similar to the one shown in the figure below, consider the following steps to resolve the issue:

i. Ensure that the initial modeling proxy values (such as pH, pCO2, d13c) are aligned with the initial proxy records provided.

ii. Modify the values in the second and third rows of Step 3 table as needed. A reliable approach is to increase the absolute values of these parameters. While this adjustment is generally safe, it is important to note that it may slow down the process. This balance between accuracy and computational efficiency needs to be considered when making adjustments.

🔴 Callback error updating ..info_integration_inv.…   2:22:13 PM   ∨

Callback error updating
..info_integration_inv.children...ysol_inv.data...tsol_inv.data..

line 358, in model_run
ems_new, results = toms748(cost_function, toms_low , toms_high,
File "/Users/shihan/opt/anaconda3/envs/iloscar/lib/python3.9/site-packages/scipy/optimize/_zeros_py.py", line 1374, in toms748
result = solver.solve(f, a, b, args=args, k=k, xtol=xtol, rtol=rtol,
File "/Users/shihan/opt/anaconda3/envs/iloscar/lib/python3.9/site-packages/scipy/optimize/_zeros_py.py", line 1221, in solve
status, xn = self.start(f, a, b, args)
File "/Users/shihan/opt/anaconda3/envs/iloscar/lib/python3.9/site-packages/scipy/optimize/_zeros_py.py", line 1121, in start
raise ValueError("a, b must bracket a root f(%e)=%e, f(%e)=%e " %
ValueError: a, b must bracket a root f(-2.000000e+00)=-5.335239e+02,
f(5.000000e+00)=-4.983734e+01