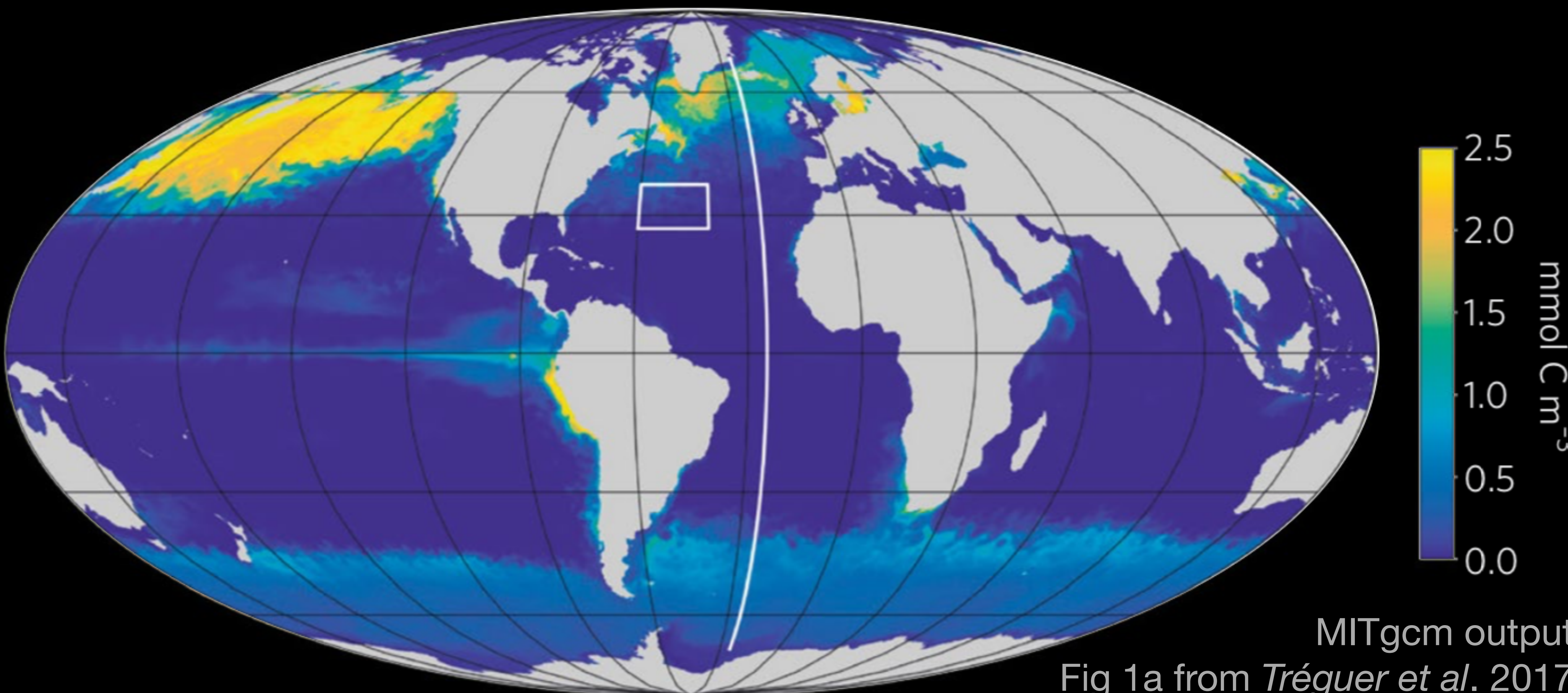
Sack Lunch Seminar Thursday 24 Jan 2019 Massachusetts Institute of Technology

# Developing a new, open-source, user-friendly, fast, modular, global marine biogeochemistry model (in julia)

Benoît Pasquier[1]

[1] Department of Earth System Science, University of California, Irvine, CA, USA.

# Global marine BGC models are great tools…

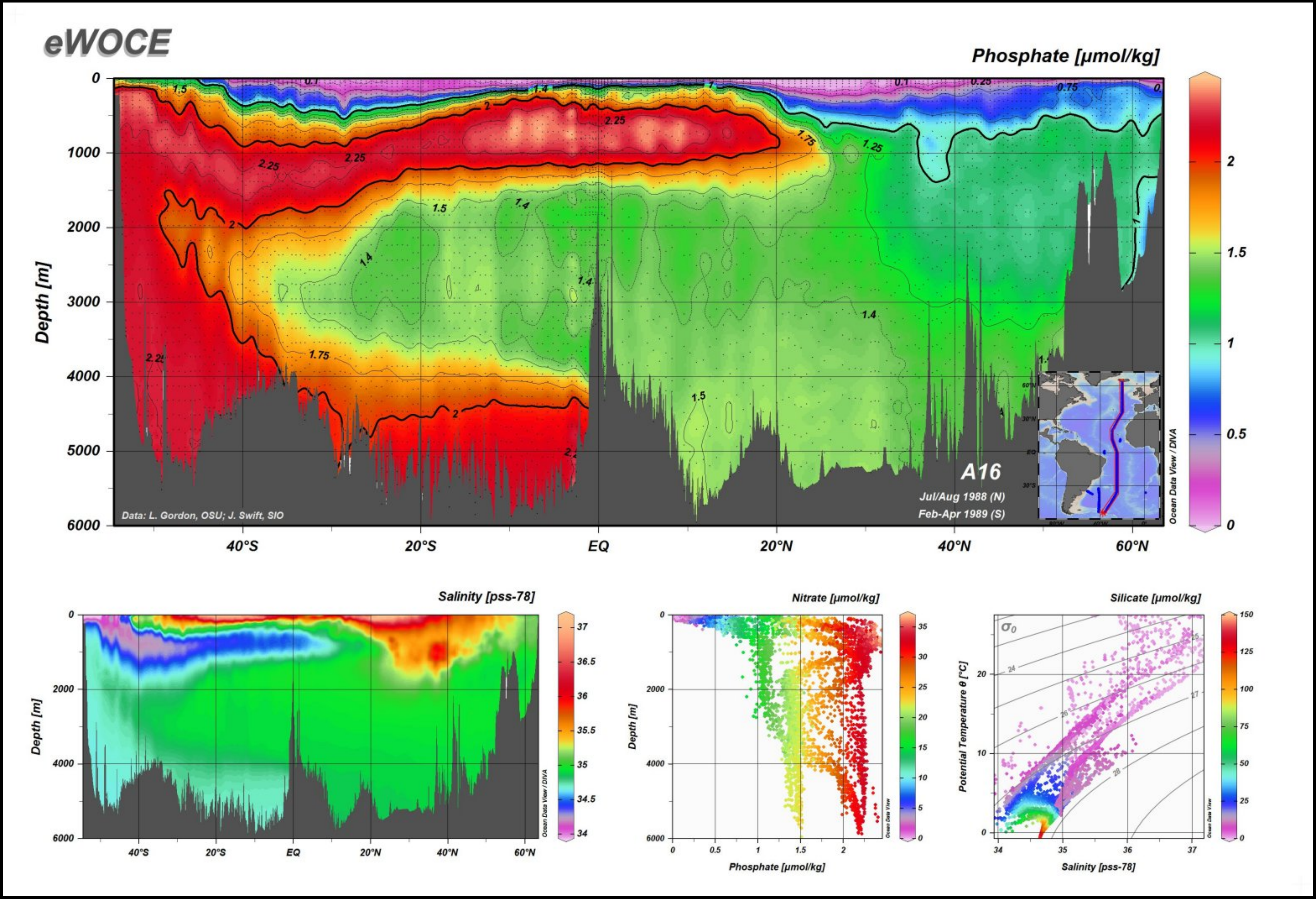



MITgcm output
Fig 1a from *Tréguer et al*. 2017

# But they are complicated to use…

# Ocean Data View (ODV)

Schlitzer, R., Ocean Data View, odv.awi.de, 2018



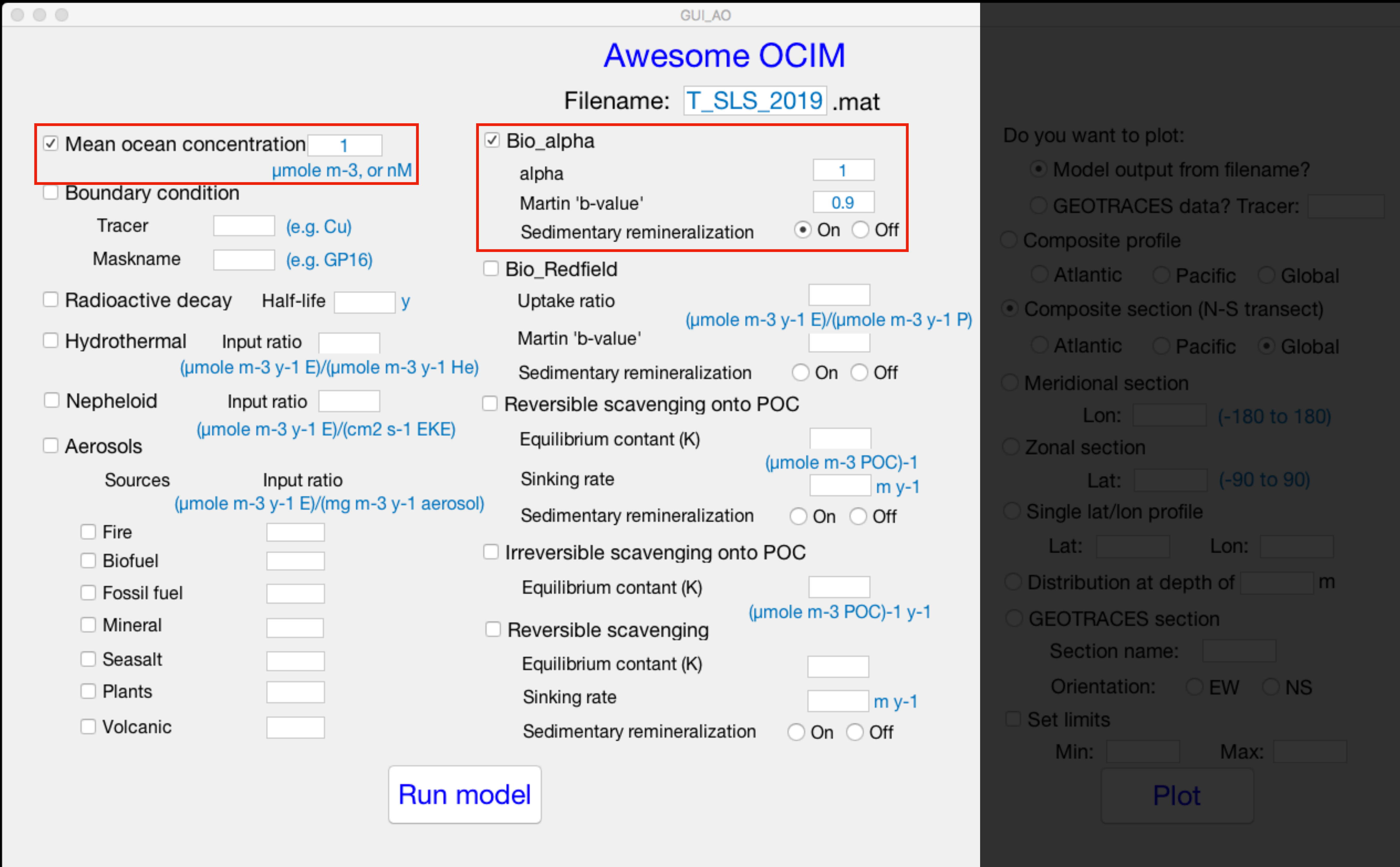
# There is room for new, simpler-to-use models…


*Seth John*

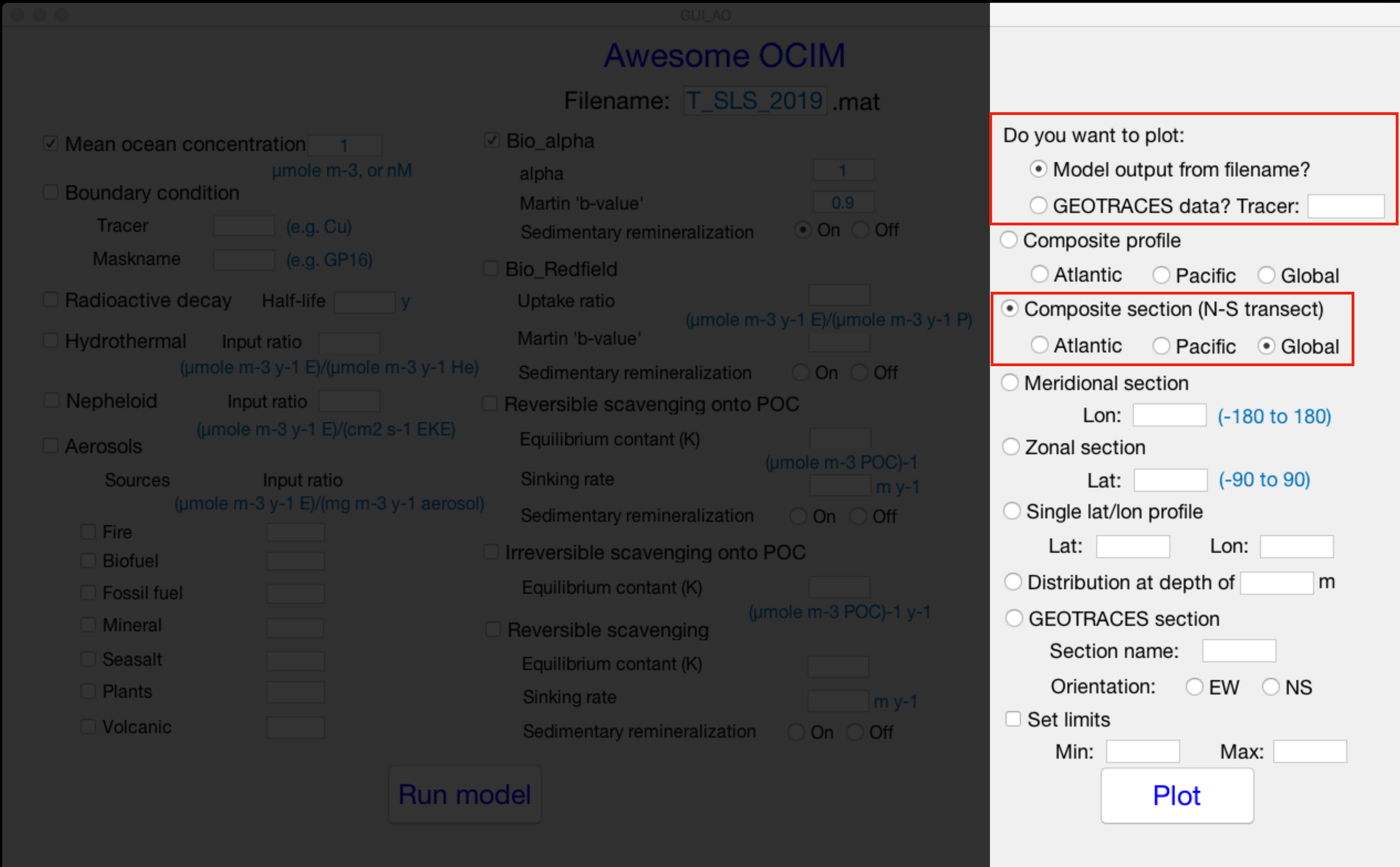
github.com/hengdiliang/AWESOME-OCIM-v1.1

# Showcase of the AWESOME OCIM

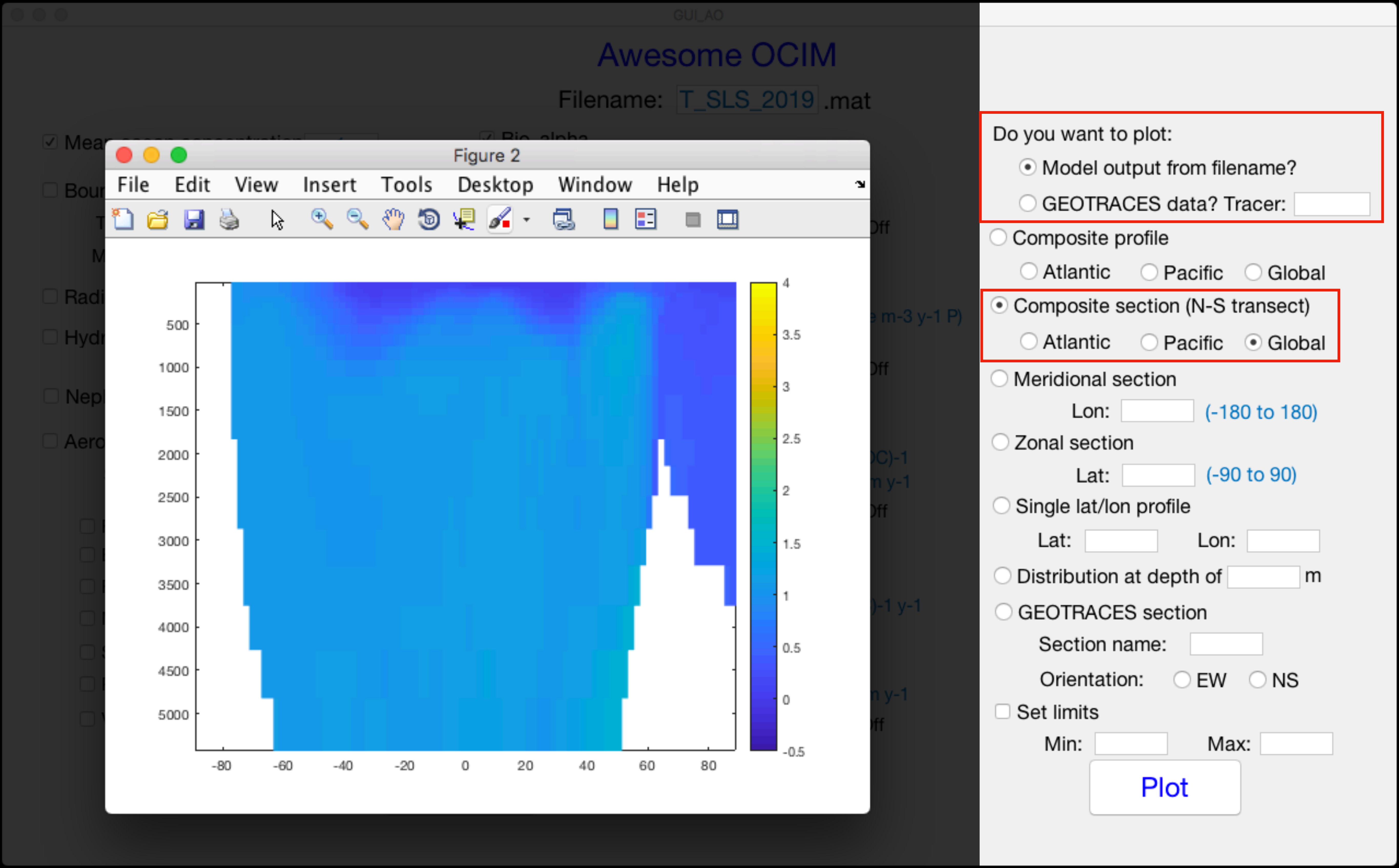
# Screenshots of the AWESOME OCIM



# Screenshots of the AWESOME OCIM

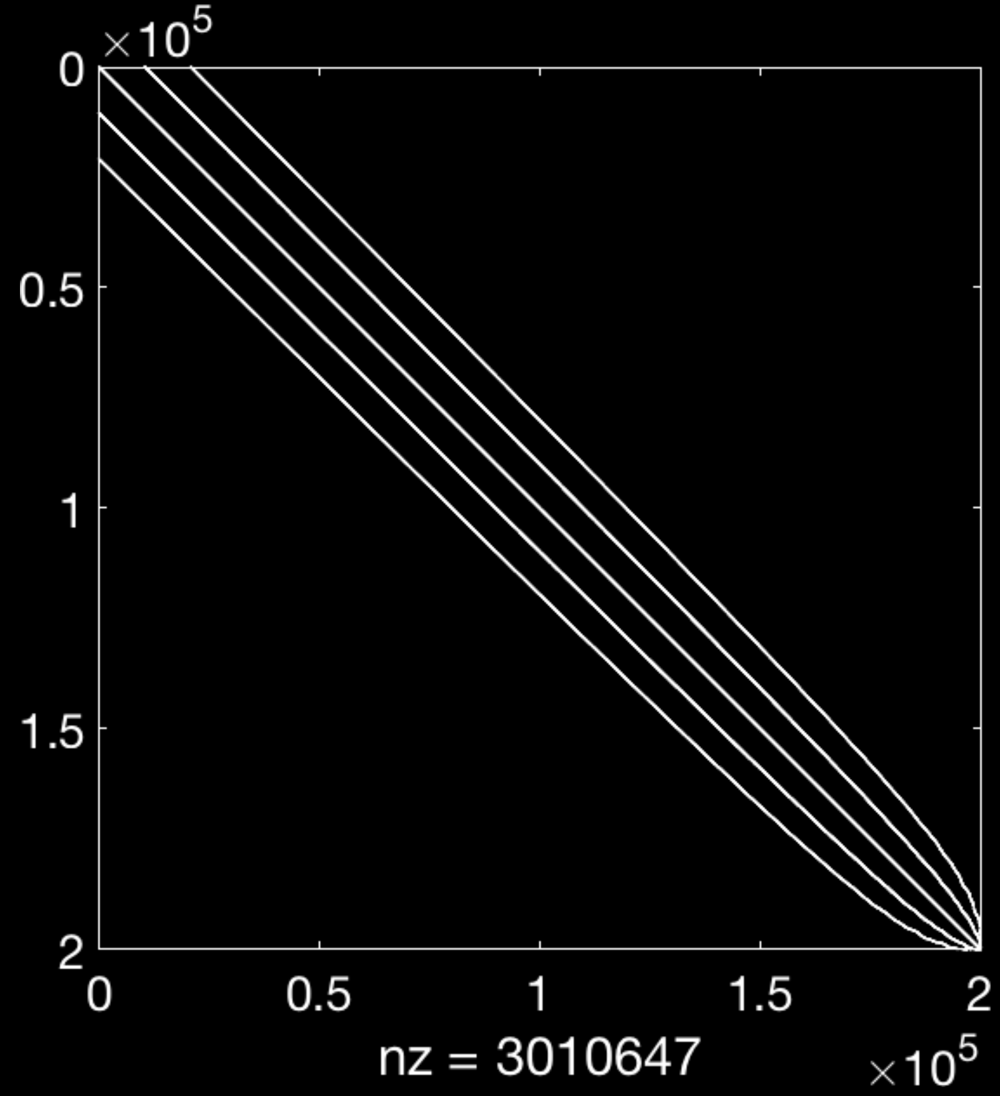


# Screenshots of the AWESOME OCIM



# The Ocean Circulation Inverse Model (OCIM) is a transport matrix

$T =$




- potential temperature
- salinity
- radiocarbon
- CFC-11

[*Devries*, 2014]

"a data-assimilation model that incorporates oceanographic tracer data to estimate a mean state of the global ocean circulation."

OCIM2 (*DeVries and Holzer*, submitted). Available early 2019.

# The AO uses linear algebra

The discrete 3D tracer, $\chi(r)$,
is rearranged into a column vector $x$.

Linear processes, represented by matrices:

- the transport operator, $T$, the OCIM [*Devries*, 2014]
- and other biogeochemical mechanisms, $L = L_1 + \ldots + L_n$

The tracer equation is then $(\partial_t + T)x = Lx + s$
where $s$ is the sum of the constant source/sink terms

The steady-state is then the solution of

$Ax = s$ —— MATLAB ——→ `A \ s`

where $A = T - L$

# A circulation-agnostic model

Matrices

- Simple few-boxes models
- OCIM1 [*DeVries*, 2014] and OCIM2 [*DeVries and Holzer*, in prep]
- CESM [e.g., *Bardin et al*., 2014]
- MITgcm [*Khatiwala et al*., 2005; *Khatiwala*, 2007]
- UVic ESCM [*Kvale et al*., 2017]
- MPAS-O [*Fu et al.,* in prep?]
- ACCESS (MOM4p1) [*Chamberlain et al*., 2019]

The ability (to easily swap in the circulation of another model) would allow us to test biogeochemistry mechanisms across circulations, maybe to disentangle the effects of the circulation from the biogeochemical transport, sources, and sinks, and to clarify the skill of the circulations and of the biogeochemistry…

Adding this functionality seems more practical than a functionality to swap the biogeochemistry module IMHO…

# The nonlinear case

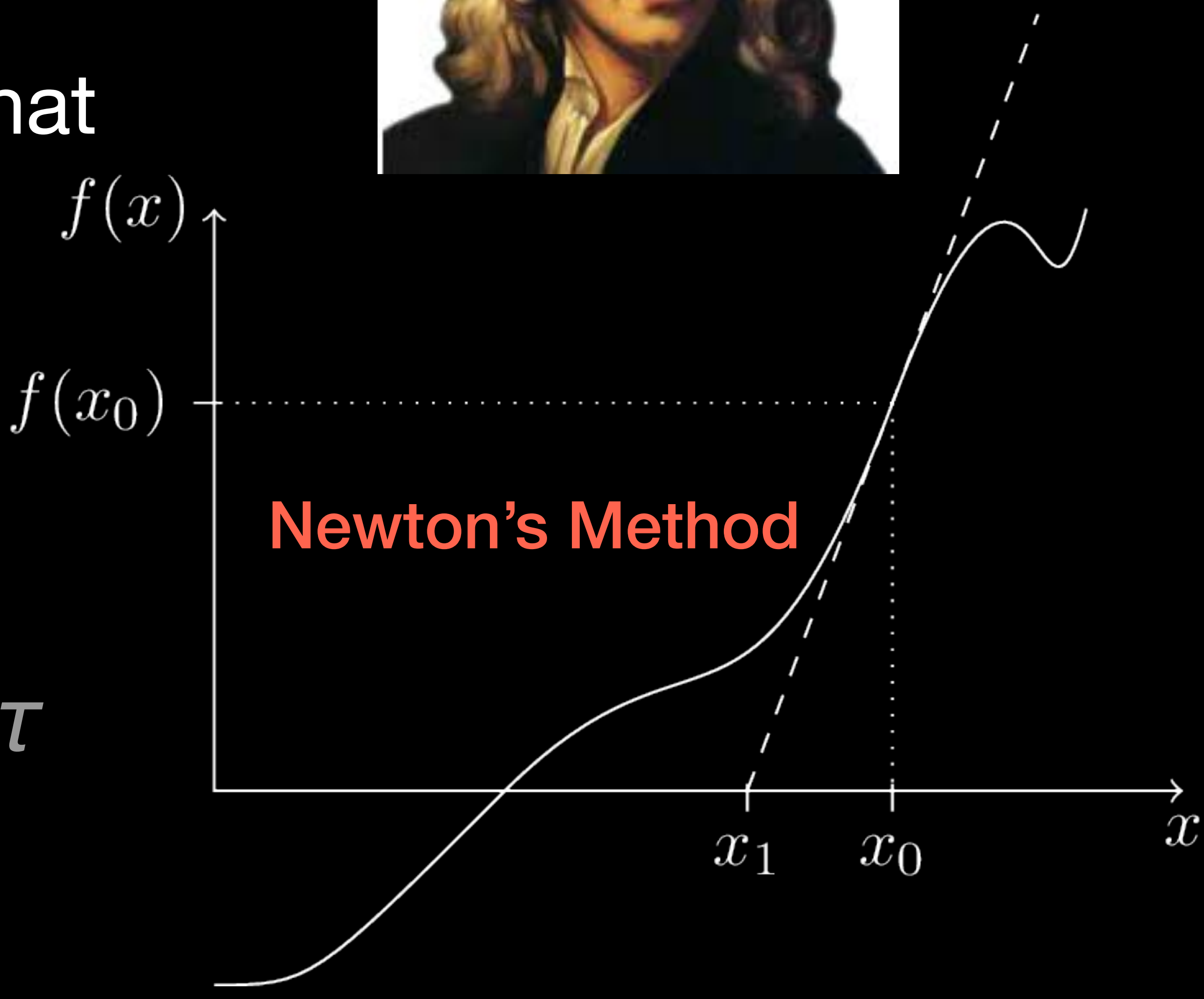
The tracer equation is not always linear. I.e., sometimes we have

$$\partial_t x = f(x)$$

instead of $(\partial_t + T)x = Lx + s$.

The steady state is the $x$ such that

$$f(x) = 0$$




Nonlinear examples:

- nutrient restoring

$$(x - x^{\mathrm{obs}})(x > x^{\mathrm{obs}})(z > z_{\mathrm{e}})/\tau$$

- Iron scavenging

$$x_{n+1} = x_n - J(x_n)^{-1} f(x_n)$$

[*Pasquier and Holzer*, 2017]

# Parameter optimization

Say we want to optimize some of the biogeochemical parameters, collected into a vector $p$. Now the problem to solve becomes

$$f(x, p) = 0$$

Denote the $p$-dependent solution by $\hat{x}(p)$.

We want to optimize $p$ by minimizing some cost functions:

- $c(x)$ — quantifies the model's state mismatch with obs.
- $c(p)$ — quantifies the mismatch of the parameters with obs.

We thus want to minimize of the form

$$q(p) = c(\hat{x}(p), p)$$

where $c(x, p) = c(x) + c(p)$

# Parameter optimization

Minimizing $q(p) = c(\hat{x}(p), p)$ can be done efficiently using another Newton's method (not root-finding)


For that, we need the gradient and Hessian of $q(p)$.

A few useful julia packages for doing just that:

- SparseArrays.jl, SuiteSparse.jl, LinearAlgebra.jl
- DualNumbers.jl, ForwardDiff.jl, HyperDualNumbers.jl
- Optim.jl, DifferentialEquations.jl
- BenchmarkTools.jl, Cassette.jl
- DualMatrixTools.jl, HyperDualMatixTools.jl

Hopefully a publication early 2019?

# Dual numbers

Dual numbers are a form of generalized complex numbers.

The basis is to consider a new number, $\varepsilon$, such that $\varepsilon^2=0$.
I.e., $\varepsilon$ behaves like an infinitesimally small number.

Dual numbers are a tool to compute derivatives. This can be seen from the Taylor expansion:

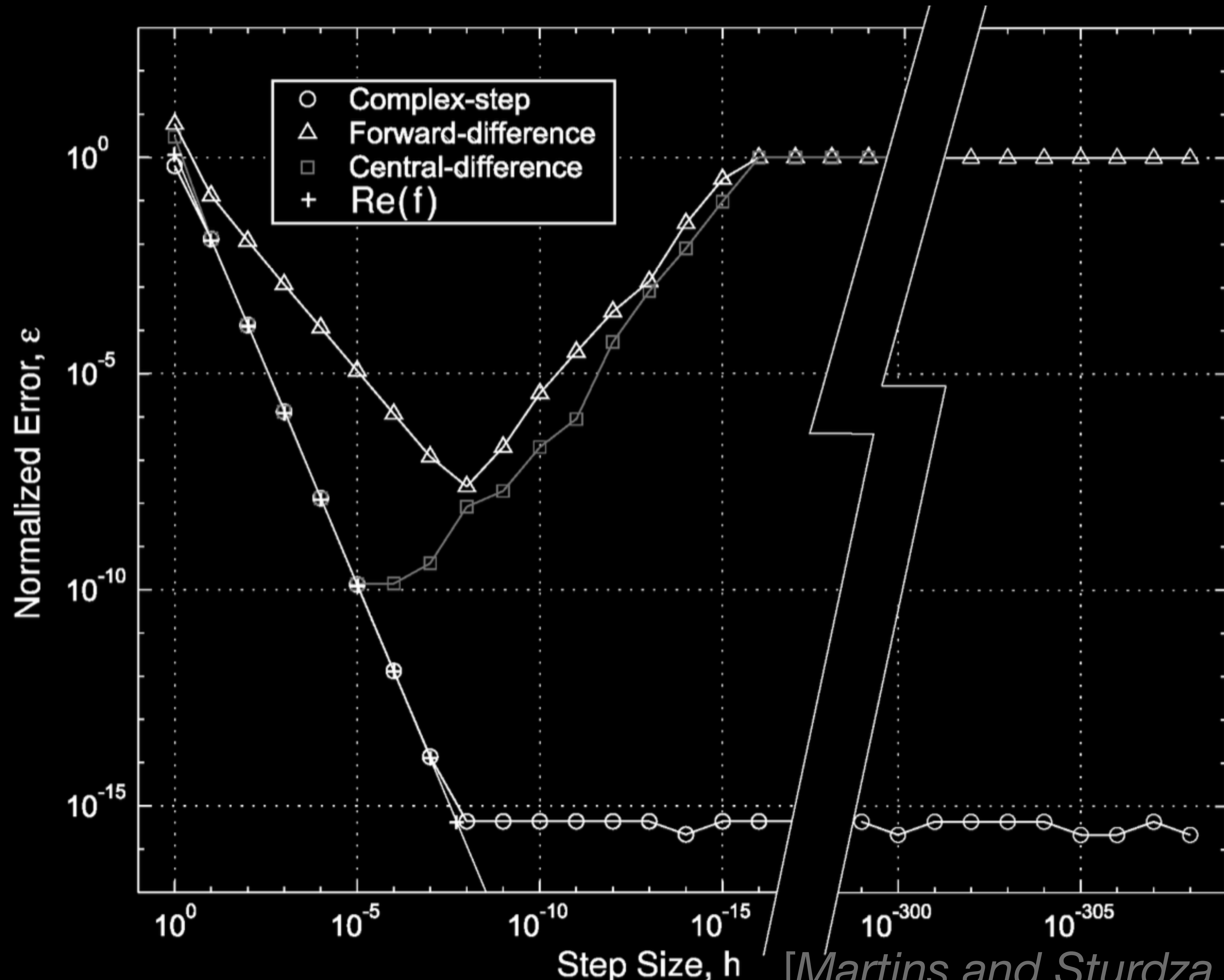
$$f(x+\varepsilon) = f(x) + \frac{\mathrm{d}f}{\mathrm{d}x}(x)\varepsilon$$

Real part Dual part

Thus, the derivative is simply the dual part of $f(x+\varepsilon)$.

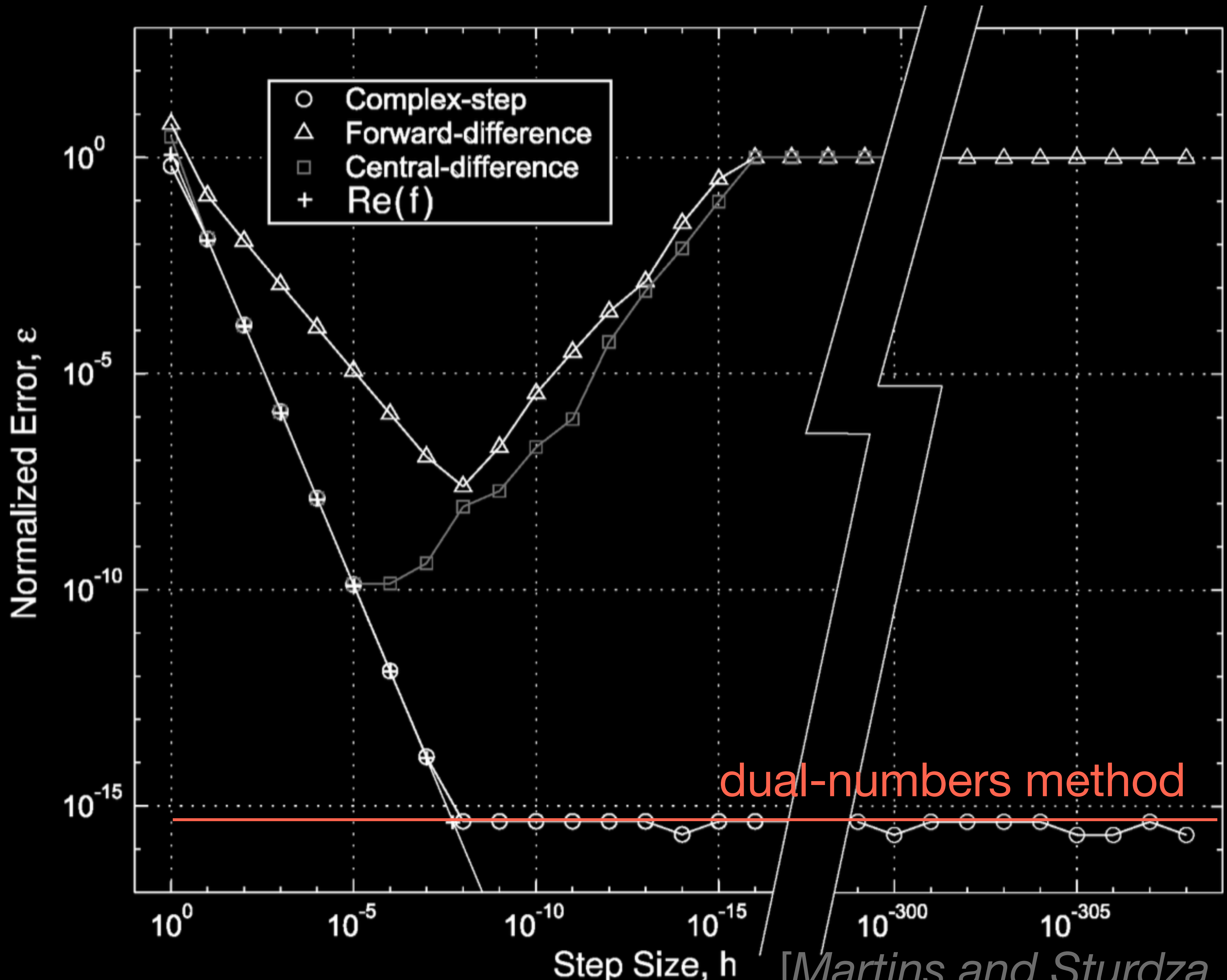
In julia, dual numbers are available through the DualNumbers.jl package.

# Dual numbers




[*Martins and Sturdza*, 2003]

# Dual numbers




[*Martins and Sturdza*, 2003]

# Dual-valued factorization

To solve a linear system of the form $Mx = y$ where $M = A + \varepsilon B$, only $A$ needs to be factorized. This is because

$$(A + \varepsilon B)^{-1} = (I - \varepsilon A^{-1}B)A^{-1}$$

The julia DualMatrixTools.jl package implements this via a new type `DualFactors` containing both

- the factors of the matrix $A$
- and the matrix $B$,

and by overloading the `factorize` function
and the backslash operator `\`
to apply the formula above when dealing with dual-valued dense and sparse matrices.

Contributions welcome on the GitHub repository:
github.com/briochemc/DualMatrixTools.jl


# Other features for the future

- Data management (DataDeps.jl for, e.g., the observational data and the swappable circulations)
  Seems to work with, e.g., World Ocean Atlas data:

  github.com/briochemc/WorldOceanDataTools.jl

- Multiple tracers. If many tracers, may need block-Newton or Newton–Krylov, matrix free methods for the solver)
- Seasonality: from matrix extraction or from CYCLOCIM)
  (Also probably need for a Newton–Krylov type of solver)
- Plotting routines / package is mandatory
- Graphical User Interface

Thank you!