eigentools: A Python package for studying differential eigenvalue problems with an emphasis on robustness

Jeffrey S. Oishi[*1], **Keaton J. Burns**[2], **S. E. Clark**[3], **Evan H. Anders**[4], **Benjamin P. Brown**[5], **Geoffrey M. Vasil**[6], **and Daniel Lecoanet**[4, 7]

1 Department of Physics and Astronomy, Bates College **2** Department of Mathematics, MIT **3** School of Natural Sciences, Institute for Advanced Study **4** CIERA, Northwestern University **5** Department of Astrophysical and Planetary Sciences, University of Colorado, Boulder **6** School of Mathematics and Statistics, University of Sydney **7** Department of Engineering Sciences and Applied Mathematics, Northwestern University

DOI: 10.21105/joss.03079

Software

- Review
- Repository
- Archive

Editor: David P. Sanders

Reviewers:

- @ketch
- @caropen

Submitted: 14 January 2021
Published: 23 June 2021

Summary and Statement of need

Linear stability analysis of partial differential equations (PDEs) is a fundamental tool in chaotic dynamics, fluid dynamics, biophysics, and many other scientific disciplines. These analyses require solving eigenvalue problems. However, studying these eigenvalues is not without significant peril: discretized PDEs are poorly conditioned and many of the numerical eigenvalues reported by standard solvers are unreliable. Additionally, it is far from trivial to start with a set of PDEs and *construct* a matrix discretization in the first place. In order to solve these problems, we present `eigentools`, a Python package that extends the eigenvalue problem (EVP) capabilities of the Dedalus Project (Burns et al., 2020) to provide a complete analysis toolkit for EVPs.

`eigentools` provides a convenient, parallelized interface for both modal and non-modal stability analyses for nearly arbitrary systems of PDEs. It provides a toolkit that makes it very easy to take a model, find robust eigenvalues and eigenmodes, and find critical parameter values for stability. The only thing a user needs to do is find a state to linearize about. One constructs a Dedalus `EigenvalueProblem` object for the PDE linearized about the chosen background, and passes that to `eigentools`. `eigentools` provides robust spurious eigenvalue rejection (Boyd, 2001), spectrum and eigenmode visualization, $\varepsilon-$pseudospectra, and the ability to project a given eigenmode onto an 2- or 3-dimensional domain and save it as a Dedalus-formatted HDF5 file to use as an initial condition for a non-linear simulation of the same system.

Robust Eigenvalues and Finding Critical Parameters

The core `eigentools` class, `Eigenproblem`, provides a simple interface to accurately solve a Dedalus eigenvalue problem using sparse or dense methods. A common use of eigenvalue problems is stability analysis, in which one characterizes the exponential growth rate γ of a mode via (typically) the temporal part of the eigenvalue, $e^{\gamma t}$ with $\gamma \in \mathbb{R}$. When $\gamma > 0$, the mode is unstable. `Eigenproblem` allows the user to choose how this "growth" is defined via custom functions of the eigenvalues returned by the solver. This permits instability to correspond to positive real parts of the eigenvalue (e.g. $e^{\sigma t}$, $\sigma = \gamma + i\omega$ and $\gamma, \omega \in \mathbb{R}$), negative imaginary parts of the eigenvalue (e.g. $e^{i(kx-\omega t)}$, $\omega = \omega_r + i\omega_i$, $\omega_r, \omega_i \in \mathbb{R}$), or any other choice. Because every eigenvalue has a corresponding growth rate, `Eigenproblem`

*Corresponding Author

returns an *overall* growth rate defined as the robust eigenvalue with the highest growth rate. `Eigenproblem` also has simple tools to plot all components of eigenmodes corresponding to a selected eigenvalue. In order to find robust eigenvalues, `eigentools` performs *mode rejection* by solving the same problem twice, once at a user specifiable multiple (1.5 by default) of the original resolution. In order to ascertain which modes are good, we calculate a figure of merit called the **inverse drift ratio** one of two ways (for details, see Chapter 7 of Boyd, 2001). For simple problems with only one mode family, one can use the *ordinal* method in which the eigenvalues are compared in sorted order. However, many important problems have *multiple wave families*. By increasing the resolution, the number of resolved modes for each family increases; because of this, one must compare the drift ratios of the *nearest* eigenvalues between the two resolutions. The right panel of Figure 1 shows both ordinal and nearest drift ratios; by the ordinal criterion, all eigenvalues would be rejected, despite the fact that many eigenvalues are robust. This shows that for this problem, one must use the nearest method.

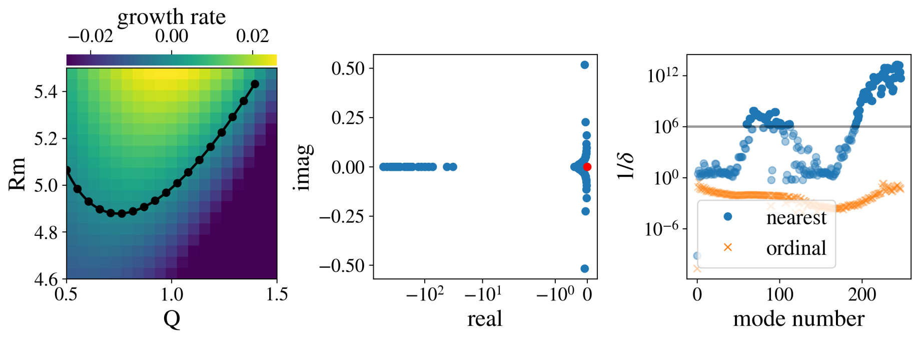

Figure 1: Magnetorotational instability. From left to right: growth rates in the $\mathrm{Rm} - Q$ plane; black line and circles show zero-growth contour. The MRI spectrum at the critical parameters. Inverse drift ratios for modes shown in the spectrum. Those below 10^6 are rejected according to nearest (blue) and ordinal (orange) criteria. For this problem, the nearest criterion must be used.

One of the original motivations for `eigentools` was to quickly and easily find critical parameters for eigenvalue stability problems. Critical parameters are those at which the fastest growing mode has zero growth rate. In order to do so, `eigentools` provides an object `CriticalFinder`, which allows users to specify an `Eigenproblem` and a 2-dimensional tuple of parameters. The user then provides a grid of points for those two parameters, and `CriticalFinder` finds the maximum growth rate for the EVP at each point in the grid, exploiting MPI parallelism on multiprocessor systems. It then interpolates to find the zero crossings of one parameter, and finally minimizes over the remaining parameter to find approximate critical values. `CriticalFinder` also provides simple visualization tools, root polishing algorithms to further refine the critical values, and the ability to save and load the grid of eigenvalues.

Figure 1 demonstrates three core features of `eigentools`: the ability to find critical parameters, the ability to use sparse and dense eigenvalue solvers, and the ability to reject spurious eigenvalues. In the left panel, the growth rate of the magnetorotational instability (defined as the positive real part of the eigenvalue σ) is plotted on a 20×20 grid of magnetic Reynolds number Rm and wavenumber Q, finding the critical values $\mathrm{Rm_c} = 4.88, Q = 0.747$ (Clark & Oishi, 2017b); in Figure 1, we used 4 cores each performing 100 *sparse* eigenvalue solves finding the 15 modes with σ closest to zero. The middle panel shows the spectrum at the critical parameters; this was solved using a *dense* eigenvalue solver to find all modes. The unstable mode is a rotationally modified Alfvén wave highlighted in red. Finally, the rightmost panel shows a plot of the inverse drift ratio $1/\delta$ for both ordinal and nearest comparisons.

Output and creation of initial conditions

`eigentools` can output eigenmodes in the Dedalus HDF5 data format so they can easily be used as initial conditions for non-linear simulations. Figure 2 highlights this capability. We solve an EVP at $\mathrm{Ra} = 10^6$ for Rayleigh-Benard convection between two no-slip plates using `eigentools` at a resolution of $n_z = 32$, and select the most unstable mode. We then project that mode on a 2-D domain, write it to disk, and load the data into a Dedalus initial value problem (IVP) solver using the full, non-linear equations for Rayleigh-Benard convection. Using Dedalus's ability to change parameters and resolutions on the fly, we then run IVP with a resolution of $(512, 64)$ until it reaches a non-linear steady state.

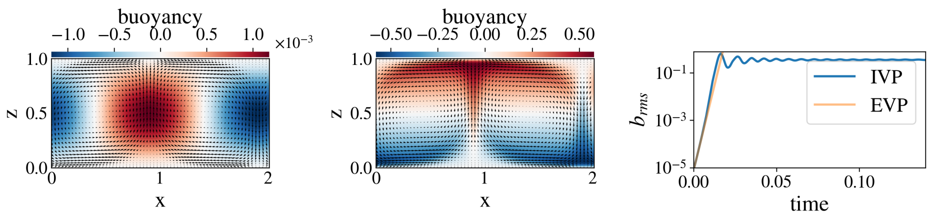

Figure 2: Rayleigh-Benard convection. From left to right: buoyancy (colormap) and velocities (arrows) for the most unstable eigenmode at $\mathrm{Ra} = 10^6$, buoyancy and velocities for the non-linear steady state for that eigenmode after evolution via an initial value problem in Dedalus, time evolution of RMS buoyancy.

The IVP in Figure 2 was run in parallel on 32 cores. In the right panel of Figure 2, we see excellent agreement between the growth rate from the non-linear IVP and the initial eigenvalue until non-linearity begins to become important around $t \approx 0.01$.

Pseudospectra

The eigentools package can also solve for the ε–pseudospectra (Trefethen & Embree, 2005) of *generalized* eigenvalue problems of the form

$$Lx = \lambda M x \tag{1}$$

using the recent algorithm given by Embree & Keeler (2017). To our knowledge, this is the first open-source system for computing $\varepsilon-$pseudospectra for arbitrary generalized EVPs, including those with singular M matrices. Such M arise quite commonly in the solution of temporally differential-algebraic equations, which occur any time there are algebraic equations in the system such as linear constraints (e.g. $\nabla \cdot \mathbf{u} = 0$) or boundary conditions. The pseudospectrum shows the sensitivity of the eigenvalue λ to a parameter ε for bounded perturbations in the underlying operators

$$L \to L + L', \qquad M \to M + M', \qquad \text{where} \qquad ||L'||,\ ||M'|| < \varepsilon. \tag{2}$$

The notion of pseudospectra relies on a particular choice of operator norm; in fluid dynamics, this is often the energy inner product. `eigentools` allow the user to implement any norm useful for their particular problem, provided it is induced by an inner product on the underlying vector space V:

$$||L|| = \sup_{x \in V} \sqrt{\frac{\langle Lx, Lx\rangle}{\langle x, x\rangle}}. \tag{3}$$

The pseudospectrum identifies the robust parts of the spectrum. Pseudospectra have numerous applications, including non-modal stability analysis in fluid dynamics (Schmid & Henningson, 2012), locating topological states (Loring, 2015), and helping to understand the unusual properties of $\mathcal{PT}$–symmetric quantum mechanics (Krejčiřík et al., 2015).

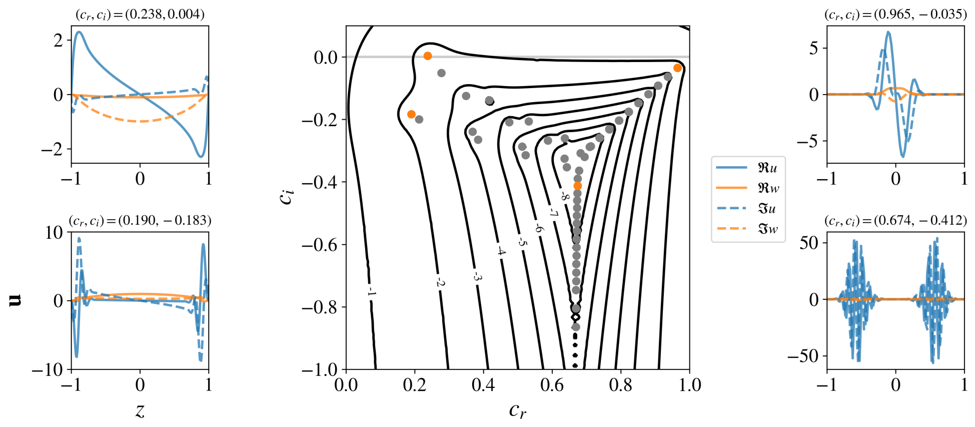

Figure 3: Spectrum, pseudospectrum, and four representative eigenmodes for the Orr-Sommerfeld problem, expressed in primitive variables (u, v). The eigenmodes correspond to the eigenvalues highlighted in orange in the middle panel; eigenvalue rejection ensures the oscillations in the modes are well-resolved. Pseudospectra contours are labeled by n, representing 10^n.

Figure 3 shows an example pseudospectrum, its corresponding spectrum, and four representative eigenvectors for the classic Orr-Sommerfeld problem in hydrodynamic stability theory (Reddy et al., 1993; Trefethen et al., 1993). As a twist on the standard problem, we demonstrate Dedalus and `eigentools` ability to solve the problem using the standard Navier-Stokes equations linearized about a background velocity, rather than in the traditional, single fourth-order equation for wall-normal velocity. This is not possible without using the generalized eigenvalue pseudospectra algorithm implemented above. Note that for the four eigenvectors, we plot u and w, the stream wise and wall-normal directions, respectively, rather than w and η, the vorticity as would be the case in the reduced Orr-Sommerfeld form. The solid and dashed lines represent the real and imaginary parts of the eigenvectors, respectively.

Example

Here we present a script that computes the spectra and pseudospectra for the Orr-Sommerfeld problem, producing a simplified version of the center plot in Figure 3. The first block of code sets up the Navier-Stokes equations in Dedalus, making use of its expressive substitution mechanism.

```
import matplotlib.pyplot as plt
from eigentools import Eigenproblem
import dedalus.public as de
import numpy as np

# define Navier-Stokes equations in Dedalus
z = de.Chebyshev('z', 128)
d = de.Domain([z])
os = de.EVP(d,['u','w','uz','wz', 'p'],'c')
```

```
os.parameters['alpha'] = 1.
os.parameters['Re'] = 10000
os.substitutions['umean'] = '1 - z**2'
os.substitutions['umeanz'] = '-2*z'
os.substitutions['dx(A)'] = '1j*alpha*A'
os.substitutions['dt(A)'] = '-1j*alpha*c*A'
os.substitutions['Lap(A,Az)'] = 'dx(dx(A)) + dz(Az)'
os.add_equation('dt(u) + umean*dx(u) + w*umeanz + dx(p) - Lap(u, uz)/Re = 0')
os.add_equation('dt(w) + umean*dx(w) + dz(p) - Lap(w, wz)/Re = 0')
os.add_equation('dx(u) + wz = 0')
os.add_equation('uz - dz(u) = 0')
os.add_equation('wz - dz(w) = 0')
os.add_bc('left(w) = 0')
os.add_bc('right(w) = 0')
os.add_bc('left(u) = 0')
os.add_bc('right(u) = 0')

os_EP = Eigenproblem(os) # the main eigentools interface

# define the energy inner product
def energy_ip(Q1, Q2):
    u1, w1 = Q1['u'], Q1['w']
    u2, w2 = Q2['u'], Q2['w']

    field = (np.conj(u1)*u2 + np.conj(w1)*w2).evaluate().integrate()
    return field['g'][0]

# Calculate pseudospectrum
k = 100 # size of invariant subspace
psize = 100 # number of points in real, imaginary points
real_points = np.linspace(0,1, psize)
imag_points = np.linspace(-1,0.1, psize)
os_EP.calc_ps(k, (real_points, imag_points), inner_product=energy_ip)

# plot
P_CS = plt.contour(os_EP.ps_real,os_EP.ps_imag, np.log10(os_EP.pseudospectrum),
                   levels=np.arange(-8,0),linestyles='solid')
plt.scatter(os_EP.evalues.real, os_EP.evalues.imag,color='blue',marker='x')
plt.xlim(0,1); plt.ylim(-1,0.1)
plt.xlabel(r"$c_r$"); plt.ylabel(r"$c_i$")
plt.tight_layout(); plt.savefig("OS_pseudospectra.png", dpi=300)
```

Related Work

There are a few other packages dedicated to the automatic construction of eigenvalue problems, including Chebfun, which can also produce pseudospectra. Chebfun, while itself released under the standard 3-Clause BSD license, is written in the proprietary MATLAB language. For computing spectra and pseudospectra for existing matrices, the venerable `EigTool` package is another open-source option, also written in MATLAB. It does not feature parallelism nor the ability to construct matrices from PDEs. `EigTool` has also been ported to the open-source Julia language in the `Pseudospectra.jl` package.

`eigentools` has been used in several papers, including Clark & Oishi (2017b); Clark & Oishi

(2017a); Anders & Brown (2017); Anders et al. (2019); Oishi et al. (2020); Burns et al. (2020); and Lin (2021).

Acknowledgments

Eigentools was developed with support from the Research Corporation under the Scialog Collaborative Award (TDA) ID# 24231 and from NASA under LWS Grant No. NNX16AC92G and SSW Grant No. 80NSSC19K0026.

References

Anders, E. H., & Brown, B. P. (2017). Convective heat transport in stratified atmospheres at low and high Mach number. *Physical Review Fluids*, *2*(8), 083501. https://doi.org/10.1103/PhysRevFluids.2.083501

Anders, E. H., Manduca, C. M., Brown, B. P., Oishi, J. S., & Vasil, G. M. (2019). Predicting the Rossby Number in Convective Experiments. *Astrophysical Journal*, *872*(2), 138. https://doi.org/10.3847/1538-4357/aaff61

Boyd, J. P. (2001). *Chebyshev and Fourier Spectral Methods: Second revised edition*. Dover Publications. ISBN: 9780486411835

Burns, K. J., Vasil, G. M., Oishi, J. S., Lecoanet, D., & Brown, B. P. (2020). Dedalus: A flexible framework for numerical simulations with spectral methods. *Phys. Rev. Research*, *2*, 023068. https://doi.org/10.1103/PhysRevResearch.2.023068

Clark, S. E., & Oishi, J. S. (2017a). The Weakly Nonlinear Magnetorotational Instability in a Global, Cylindrical Taylor-Couette Flow. *Astrophysical Journal*, *841*(1), 2. https://doi.org/10.3847/1538-4357/aa6ff6

Clark, S. E., & Oishi, J. S. (2017b). The Weakly Nonlinear Magnetorotational Instability in a Local Geometry. *Astrophysical Journal*, *841*(1), 1. https://doi.org/10.3847/1538-4357/aa6ff1

Embree, M., & Keeler, B. (2017). Pseudospectra of matrix pencils for transient analysis of differential-algebraic equations. *SIAM Journal on Matrix Analysis and Applications*, *38*(3), 1028–1054. https://doi.org/10.1137/15M1055012

Krejčiřík, D., Siegl, P., Tater, M., & Viola, J. (2015). Pseudospectra in non-hermitian quantum mechanics. *Journal of Mathematical Physics*, *56*(10), 103513. https://doi.org/10.1063/1.4934378

Lin, M.-K. (2021). Stratified and Vertically Shearing Streaming Instabilities in Protoplanetary Disks. *Astrophysical Journal*, *907*(2), 64. https://doi.org/10.3847/1538-4357/abcd9b

Loring, T. A. (2015). K-theory and pseudospectra for topological insulators. *Annals of Physics*, *356*, 383–416. https://doi.org/10.1016/j.aop.2015.02.031

Oishi, J. S., Vasil, G. M., Baxter, M., Swan, A., Burns, K. J., Lecoanet, D., & Brown, B. P. (2020). The magnetorotational instability prefers three dimensions. *Proceedings of the Royal Society of London Series A*, *476*(2233), 20190622. https://doi.org/10.1098/rspa.2019.0622

Reddy, S. C., Schmid, P. J., & Henningson, D. S. (1993). Pseudospectra of the Orr–Sommerfeld operator. *SIAM Journal on Applied Mathematics*, *53*(1), 15–47. https://doi.org/10.1137/0153002

Schmid, P. J., & Henningson, D. S. (2012). *Stability and Transition in Shear Flows*. Springer New York. ISBN: 9781461301851

Trefethen, L. N., & Embree, M. (2005). *Spectra and Pseudospectra: The behavior of non-normal matrices and operators*. Princeton University Press. ISBN: 9780691119465

Trefethen, L. N., Trefethen, A. E., Reddy, S. C., & Driscoll, T. A. (1993). Hydrodynamic stability without eigenvalues. *Science*, *261*(5121), 578–584. https://doi.org/10.1126/science.261.5121.578