SoftwareX 24 (2023) 101567

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

rockphypy: An extensive Python library for rock physics modeling

Jiaxin Yu [a,*], Tapan Mukerji [b], Per Avseth [c,d]

[a] Norwegian University of Science and Technology, Department of Geoscience and Petroleum, Trondheim, Norway
[b] Department of Energy Science and Engineering, Stanford University, Stanford, USA
[c] Dig Science, Oslo, Norway
[d] Norwegian University of Science and Technology, Department of Electronic Systems, Trondheim, Norway

ARTICLE INFO

Keywords:
Python
Rock physics models
Interpretation
Rock physics diagnostics

ABSTRACT

Rock physics aims to understand the relationship between the physical properties of rocks and geophysical observables under various conditions. The generic knowledge provides valuable insights into the behavior of subsurface rocks and has been applied in various fields. However, the availability of comprehensive open-source Python libraries for rock physics is quite limited. To address this limitation, we present **rockphypy**: a comprehensive and streamlined Python library that offers access to a vast array of rock physics models and workflows ranging from basic to sophisticated. The library is designed to be easily embedded in interdisciplinary fields such as deep neural networks and probabilistic frameworks, leveraging the rich resources of Python. Currently, **rockphypy** implements ten modules with over 100 methods, accessible through a straightforward and user-friendly API that facilitates various modeling tasks in rock physics. Its modular design allows easy extension to incorporate new features and functionalities. In addition to the versatility of the library, we have shown that **rockphypy** also greatly simplifies practical tasks that require many different rock physics models, enabling fast experimentation and iteration of research and practical programs.

Code metadata

Current code version	0.0.1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00355
Permanent link to Reproducible Capsule	
Legal Code License	GPL-3.0
Code versioning system used	pypi, git
Software code languages, tools, and services used	python, readthedocs
Compilation requirements, operating environments	python ≥3.8, requirements provided in requirement.txt
If available Link to developer documentation/manual	https://rockphypy.readthedocs.io/en/latest/index.html
Support email for questions	jiaxin.yu@ntnu.no

1. Motivation and significance

Rock physics is a multidisciplinary field that draws upon geophysics, petrophysics, geomechanics, and geology. The term "rock" can refer to naturally occurring rocks, sediments, and granular media, synthetic rocks created in laboratory settings [1], and digital rocks created using high-resolution imaging and modeling techniques on computers [2]. The main objective of rock physics is to quantitatively describe the relationships between rock properties and physical measurables of rock under different conditions [3]. Physical measurements of rocks can be obtained from field-based remote sensing data acquisition methods [4], such as seismic, electrical, and electromagnetic measurements, downhole well logging measurements, static and dynamic laboratory experiments [5], as well as estimates from numerical simulations. [6]. The sought-after relationships can be mathematically formulated as various rock physics models (RPMs). Numerous RPMs models targeting different research problems in rock physics have been proposed in the literature, which constitutes a complex yet invaluable repository of resources.

* Corresponding author.
E-mail address: jiaxin.yu@ntnu.no (Jiaxin Yu).

https://doi.org/10.1016/j.softx.2023.101567
Received 30 May 2023; Received in revised form 22 September 2023; Accepted 15 October 2023
Available online 21 October 2023

Rock physics has a wide range of applications in various fields and is essential for making informed decisions in many industries. For example, rock physics has played a crucial role in studying the effect of fluid extraction and injection on the subsurface reservoir system and its surroundings. It helps the quantitative interpretation and monitoring of changes in seismic responses resulting from variations in reservoir rock and fluid properties [7,8]. Recently, there has been a shift in focus from energy recovery to reducing and neutralizing CO_2 emissions, the same generic rock physics knowledge has been increasingly applied to sustainable studies, such as CO_2 sequestration [9] and geothermal energy exploration [10]. Another rapidly evolving field in rock physics is the intersection of rock physics, statistics [11], machine learning [12], and deep learning approaches [13], enabling the uncertainty quantification of prediction using rock physics given limited data and promoting significantly the precision and accuracy of predictions given large volumes of data. To foster innovation and progress, it is crucial to democratize the rich resource of rock physics models, making them more adaptable to current computing power and infrastructure for AI-based inference in Python, and more accessible to the research community.

The Stanford SRB Matlab toolbox [3] was one of the earliest open-source projects of an extensive implementation of various rock physics models. There have been other resources of rock physics models and applied examples written in Matlab [7]. Researchers [11,14] sometimes will publish Python codes containing a limited number of RPMs dedicated to the workflow present in the research papers. There are also a few open-source Github projects offering a selection of commonly used RPMs such as open_petro_elastic and rppy. Many of the projects mentioned above however do not regard RPMs as the centerpiece. More often, rock physics models are integrated into commercial software and remain prioritized. There is still a lack of comprehensive, standardized streamlined Python integration of the vast array of rock physics models available. To address this gap, we present **rockphypy**, an open-source Python library that provides an extensive, cohesive, and structured implementation of RPMs, supporting e.g. the modeling of elastic behavior of rock, fluid, and their coupling under various considerations, with over 100 methods. **rockphypy** is crafted to embody flexibility and versatility allowing easy extension and remaining adaptable to evolving rock physics field. There are two ways to use **rockphypy**, individual models and extensions are designed to be called without initializing an instance, on the other hand, given data, an instance of the practical workflows built upon different RPMs can be created, the instance is then used to do data analysis. In this way, **rockphypy** permits a great reusability of the code which allows for rapid experimentation and iteration of research and practical programs. This library is expected to be useful for both researchers and practitioners interested in various rock physics applications.

2. Software description

rockphypy is an open-source Python library for rock physics modeling. It is a trove of useful models and workflows that aims to offer a convenient and efficient way to perform complex tasks in rock physics studies. For that purpose, a consistent API (Application Programming Interface) and image galleries of examples and tutorials are documented on ReadTheDocs https://rockphypy.readthedocs.io/en/latest/. All examples can be downloaded as both Python files and Jupyter notebooks. The **rockphypy** is currently released under the GNU General Public License (GPLv3) [15]. To foster a community-driven innovation and collaboration, the codebase is publicly accessible on GitHub https://github.com/yujiaxin666/rockphypy and encourages contributions and continued development from the community.

2.1. Software architecture

rockphypy utilizes objected oriented programming (OOP) paradigm in Python to create modular, maintainable, and reusable code. The library is built upon two Python libraries *numpy* and *scipy* and consists of 10 core modules, each of which is associated with a class that holds the implementation of a specific group of models and workflows. Fig. 1 depicts module organization and some keywords of the generic functions. The hierarchical classification of **rockphypy** is designed for easy extension to introduce new rock physics modeling approaches.

In rock physics modeling, it is often necessary to use multiple models to complete a task. Additionally, it is common for a newly proposed model to depend on existing models. As a result, the classes in **rockphypy** are designed to interact with one another and are mutually called. These interactions are illustrated and color-coded in Fig. 1.

2.2. Software functionalities

rockphypy includes 10 classes and over 100 methods that can be used to perform a variety of rock physics modeling tasks. The list below provides a detailed description of each class and its purpose. Users are recommended to check the API interface of the library, which provides an exhaustive explanation of each individual method. Referred link of API: https://rockphypy.readthedocs.io/en/latest/autoapi/index.html

AVO: The AVO class consists of methods that model the Reflectivity, Amplitude Variations with Offset (AVO), and Amplitude Variations with Azimuth (AVOAz) in both Isotropic and anisotropic Media.

Anisotropy: Anisotropy is generally used to describe the properties of the material or systems that are directionally dependent [16]. The Anisotropy class focuses primarily on seismic anisotropy, which concerns the directional dependence of wave velocity and elastic properties of rock. This class includes various models for computing the effective elastic constants and velocities of transversely isotropic (TI) media.

BW: The acronym BW stands for Batzle–Wang [17] who developed models for predicting the density and bulk moduli of reservoir fluids by combining thermodynamic relationships and empirical trends from published data. The BW class implements the original Batzle–Wang models to compute water, brine, oil, and gas properties at various pressures and temperatures. Additionally, the modified Batzle–Wang equations [18] that yield more accurate CO_2 properties are also included.

EM: Effective medium (EM) models treat rock as a composite of different constituents such as pore and matrix. The EM class includes most well-known EM models that pertain to the theoretical modeling of macroscopic rock properties by incorporating the individual elastic properties, the volume fractions, and the assumed spatial geometry of the different phases that make up the rock. EM models requiring the recursive or iterative computation of the effective properties such as Differential Effective Models (DEM) and Self-Consistent (SC) models are also included. Note that this class is primarily concerned with elastic mixing laws and inclusion effective medium models, while the effective medium models for the granular medium are collected in **GM** class.

Empirical: This class contains models that are empirically derived to best fit certain types of data. Examples include the Krief model [19] for computing the elasticity of stiff sandstone, the Greenberg–Castagna model for S-wave velocity prediction, and various rock physics depth trends of different lithologies in different sedimentary basins. These models usually have limited ranges of applicability and should be used with consideration.

Fluid: Fluid presence has a significant impact on the effective elastic properties of rocks and the propagation of waves, and it is a crucial aspect of rock physics. This class includes several methods for performing fluid substitutions (E.g., isotropic and anisotropic Biot–Gassmann and Brown–Korringa theories), as well as various poroelastic models that describe the attenuation and dispersion caused by wave-induced fluid flow.

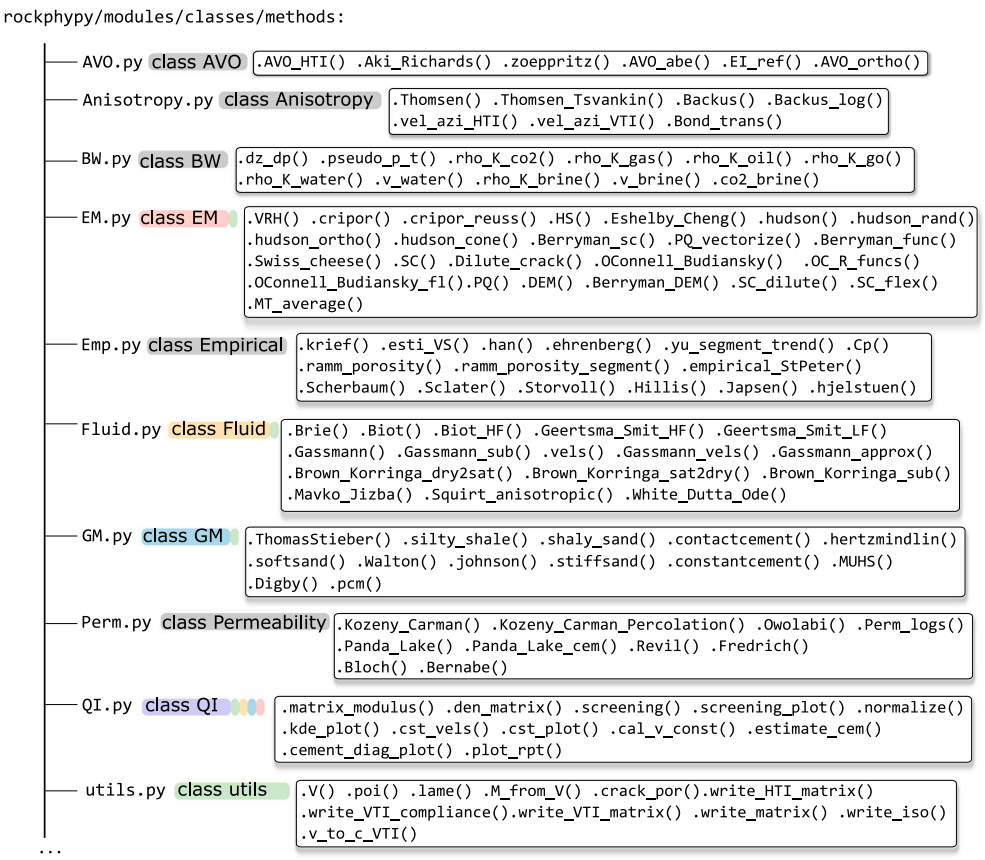

Fig. 1. Codebase structure of rockphypy 0.0.1 version. A color-coding scheme is employed to differentiate classes based on their interdependencies. Classes that exhibit mutual calling relationships have been assigned distinct colors, whereas those classes that operate independently of others are denoted in gray. Taking the red-coded EM class as an example, it invokes the **utils** class, hence it has a green color tag. Notably, the **QI** class exhibits dependencies on multiple classes, one of which being the **EM** class, consequently leading to the assignment of a red tag to the **QI** class. Readers are referred to the API page https://rockphypy.readthedocs.io/en/latest/autoapi/index.html for a comprehensive listing of all methods and their usage. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

GM: The study of granular media is essential in rock physics as it plays a crucial role in determining the mechanical and elastic properties of many types of rocks. Sands and sandstones, for example, can be represented by the granular packing of unbounded and bounded clastic particles, respectively. The literature contains numerous models, including theoretical, hybrid, and heuristic models [7], which have gained popularity in both academia and industry. The **GM** class provides a comprehensive collection of these models, including their heuristic extensions, which can be used to model the elastic properties of different lithologies, such as sand-shale mixtures.

Permeability: This class includes the Kozeny–Carman relation [20], as well as several empirical estimations of permeability in porous media based on it.

QI: Rock physics plays a crucial role in Quantitative Interpretation (QI) of geophysical measurements. The QI class offers some of the most widely used techniques and newly developed workflows in practice, including rock physics templates, rock physics diagnostics, AVO synthetics, and diagenetic modeling with rock physics constraints.

utils: Short for utilities, contains fundamental functions and tools that are commonly used throughout the entire codebase. These include velocity computation, conversion between elastic modulus and stiffness matrix formulations for isotropic and anisotropic media, as well as other basic functions. The **utils** class helps reduce code duplication and enhances the maintainability of the codebase.

3. Illustrative examples

The classes in the API that were previously described mostly consist of different rock physics models. In the **rockphypy** library, all of the rock physics models, except for workflows, are written as static methods within each Python class. This makes it flexible to use the models with or without a dataset, depending on the task at hand. The following examples demonstrate how to use **rockphypy** to perform various rock physics modeling tasks. All of the datasets and scripts used in these examples are available for download from the GitHub repository and ReadTheDocs page of the library.

3.1. Rock physics modeling of CO_2 sequestration

CO_2 injection involves injecting carbon dioxide into a storage reservoir, where it usually mixes with the preexisting pore fluids, To monitor the injection process using time-lapsed seismic methods, it is necessary to have knowledge of the seismic properties of the fluid mixture consisting of CO_2 and *in-situ* pore fluid at different reservoir pressures and temperatures.

The acoustic properties of pure carbon dioxide can be computed using different equations of state, depending on the preferred method. Complex models, such as EOS-CG model [21] based on Helmholtz energy equations of state, provide accurate estimations for both pure and impure CO_2 as a function of temperature and pressure. However, these models require numerical optimization and are usually only available through specialized software.

A common practice is to use the Batzle–Wang equations for gas to compute CO_2 properties. However, this can be highly inaccurate, particularly at higher fluid pressures. Xu et al. [18] modified the original Batzle–Wang (B–W) equations, making them more accurate for computing CO_2 properties at various P-T conditions. The discrepancy between using the Batzle–Wang equation and its modified version can

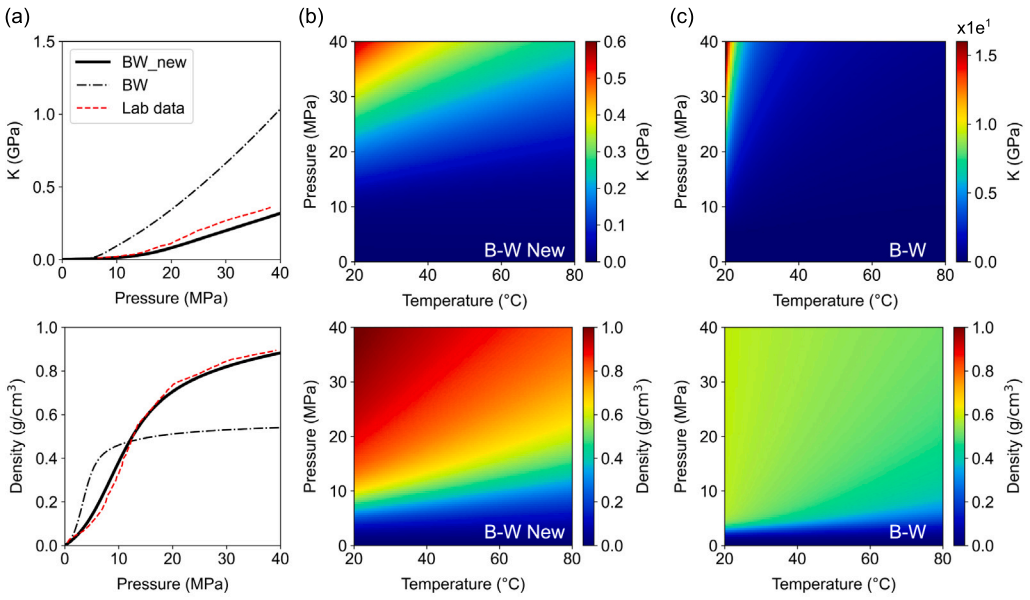

Fig. 2. (a) Comparison of acoustic properties of CO_2 computed using both the original and modified B–W methods at a temperature of 57 °C. The experimental data [22] of the CO_2 properties are digitized from the Ref. [18]. (b) Bulk modulus and density of pure CO_2 computed using new B–W method at various P-T conditions. (c)The same results computed using the original B–W method for gas.

be easily compared by utilizing the built-in methods in **rockphypy** as follows.

```
# import the modules
from rockphypy import BW, GM, Fluid
import numpy as np
# Computing bulk modulus and density of
# pure CO2 properties  as a function of pressure
# and temperature.
G = 1.5349  # gas gravity of CO2
pressure = np.linspace(0, 40, 100)
temperature = np.linspace(20, 80, 100)
P, T = np.meshgrid(pressure, temperature)
# new BW prediction
rho_co2, K_co2 = BW.rho_K_co2(P, T, G)
# original BW prediction
rho_co2_BW, K_co2_BW = BW.rho_K_gas(P, T, G)
```

The results are shown in Fig. 2. The new B–W method gives more accurate predictions compared to the original B–W methods, and the discrepancy between the two models exponentially grows as pressure increases.

The following experiment investigates the impact of errors in CO_2 properties on the seismic response of an injected reservoir. The seismic properties of a CO_2-brine mixture are calculated using the Brie fluid mixing method, based on the reservoir temperature, post-injection pore pressure, and known brine salinity. Both the original and modified Batzle–Wang equations are used to calculate the mixture properties. The reservoir is assumed to be homogeneous and isotropic, and a granular medium model is utilized to compute the dry rock properties at the corresponding effective stress. Finally, the P and S wave velocities of an unconsolidated sandstone reservoir saturated with CO_2-brine mixtures at various levels of CO_2 saturation are calculated using Biot–Gassmann theory.

```
# Modeling of CO2 sequestration
# in unconsolidated reservoir.
# grain density, bulk and shear modulus
D0, K0, G0 = 2.65, 36, 42
# brine density, bulk modulus
Db, Kb = 1, 2.2
# Reservoir condition and brine salinity
overburden_stress = 40 # MPa
pore_pressure = 20 # MPa
temperature = 45 # Degree Celsius
# effective stress
sigma = overburden_stress-pore_pressure
salinity = 35000/1000000
sw = np.linspace(0, 1, 50) # water saturation
sco2 = 1-sw # CO2 saturation
# Parameter for granular medium
phi_c = 0.4 # critical porosity
Cn = 6 # coordination number
# saturation condition: patchy saturation
brie = 4
# Using softsand model to compute the dry
# rock properties
Kdry, Gdry = GM.softsand(K0, G0, phi_c, phi_c,
    Cn,
sigma, f=1)
# CO2-brine mixture properties computed using
# original B-W and Brie mixing law
den1, Kf_mix_1 = BW.co2_brine(temperature,
pore_pressure, salinity, sco2, brie_component=
    brie,
bw=True)
# CO2-brine mixture properties computed using
    new
# B-W and Brie mixing law
den2, Kf_mix_2 = BW.co2_brine(temperature,
pore_pressure, salinity, sco2, brie_component=
    brie,
bw=False)
# Seismic properties of the reservoir rock
    saturated
# with Brine-CO2 mixture
vp1, vs1, rho1 = Fluid.vels(Kdry, Gdry, K0, D0,
Kf_mix_1, den1, phi_c)# B-W
vp2, vs2, rho2 = Fluid.vels(Kdry, Gdry, K0, D0,
Kf_mix_2, den2, phi_c)# B-W new
```

The impact of CO_2 properties on the modeling of time-lapse effects of CO_2 sequestration is shown in Fig. 3. The use of **rockphypy** makes it easy and intuitive to perform modeling tasks, enabling users to focus on specific features. It provides great flexibility by allowing users to customize functions with a wide range of pre-built functionalities. In

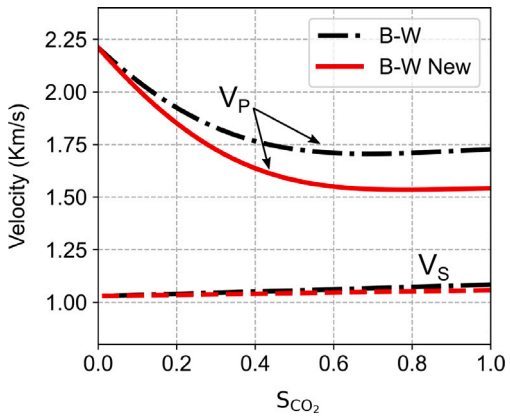

Fig. 3. The P and S wave velocities of an unconsolidated sandstone reservoir saturated with CO_2-brine mixture are affected by the accuracy of the CO_2 model. The use of the less accurate B–W gas model leads up to about 15% error in P wave velocity as the CO_2 saturation increases. Mistakes in density calculation of CO_2-brine mixture also impact the shear wave velocity, particularly for high CO_2 saturations.

this example, the `BW.co2_brine` method is built upon several basic building blocks to model a brine-fluid mixture using patchy mixing.

3.2. Rock physics interpretation

Grain contact rock physics models, in combination with fluid substitution, can be utilized for both qualitative and quantitative interpretation tasks [7]. In the following, we demonstrate how to efficiently perform rock physics screening, diagnostics, lithology, and fluid predictions using the **rockphypy** with a synthetic dataset. The well log dataset used in this section is synthesized based on the empirical depth trends for different seismic properties of sandstones in Norwegian offshore.

In practice, given the abundance of well log measurements, data screening is essential to identify and address any errors, inconsistencies, or missing data in the dataset before further analysis is conducted. Fig. 4a depicts the rock physics screening [12] results for the sandstone data. The elastic bounds generated using contact-based elastic models can determine whether the data is consistent with known principles of physics.

```
import pandas as pd
from rockphypy import QI
df=pd.read_csv('../../data/well/sandstone.csv')
# grain density, bulk and shear modulus
Dqz, Kqz, Gqz = 2.65, 36.6, 45
# clay density, bulk and shear modulus
Dsh, Ksh, Gsh = 2.7, 21, 7
# cement density, bulk and shear modulus
Dc,Kc, Gc = 2.65, 36.6, 45
Db, Kb = 1, 2.2 # brine density, bulk modulus
phi_c = 0.4 # critical porosity
sigma = 20 # effective pressure
scheme = 2 # cement distribution
Cn = 8.6 # coordination number
vsh = 0 # shale volume
phib = 0.3 # adjust porosity
f = 0.5 # slip factor
# compute elastic bounds
phi,vp1,vp2,vp3,_,_,_ = QI.screening(Dqz,Kqz,Gqz
    ,
Dsh,Ksh,Gsh,Dc,Kc,Gc,Db,Kb,phib,phi_c,sigma,vsh,
scheme,f, Cn)
# initialize the object
qi = QI(df.VP, phi = df.PHIT_ND,Vsh = df.VSH_GR)
fig_a = qi.screening_plot(phi,vp1,vp2,vp3)
```

Probability density functions can also be applied to provide better visualization of the data, which reveals more information about the distribution of rock properties in certain datasets. Fig. 4b shows the KDE (Kernel Density Estimation) plot of data generated using the following commands:

```
fig_b = qi.kde_plot(phi,vp1,vp2,vp3)
```

Diagenetic cement is commonly found in reservoir sandstone. The volume of cement can be estimated through rock physics diagnostics using constant cement models. Fig. 4c displays the estimation for the example dataset, assuming silica cement. The code used to generate the examples is presented below:

```
# estimate cement
vcem_seeds = np.array
    ([0,0.005,0.01,0.02,0.03,0.04,0.1] )
vcem = qi.estimate_cem(vcem_seeds,Kqz,Gqz,Ksh,
Gsh,phi_c,Cn,Kc,Gc,Db,Kb,scheme,vsh,Dsh,Dqz,Dc)
# drawing the constant cement lines
phib_p = [0.3,0.37,0.38,0.39,0.395]
fig_c = qi.cement_diag_plot(vcem,Dqz,Kqz,Gqz,Dsh
    ,Ksh,
Gsh,Dc,Kc,Gc,Db,Kb,phib,phib_p,phi_c,sigma,vsh,
    Cn, scheme,f)
```

Apart from the aforementioned functionalities, Rock Physics Templates (RPTs) can also be utilized to comprehend the elastic attributes derived from well log data, as depicted in Fig. 4d. Rock physics models can calculate elastic properties with various combinations of lithology and fluid parameters. Based on these characteristics, Rock Physics Templates (RPTs) provide a reference framework of all the possible variations of a specific rock with different fluid saturations. RPTs can be constructed using various grain contact models. The following code uses the stiffsand model to construct the RPT. By transforming the same data to the RPT domain, it becomes evident that the data are primarily dominated by brine-saturated sandstone, which complies with the empirical trends used to generate the data.

```
# stiffsand model
Kdry, Gdry = GM.stiffsand(K0, G0, phi, phi_c, Cn
    , sigma, f=0)
fig_d = plot_rpt(Kdry,Gdry,K0,D0,Kb,Db,Kg,Dg,phi
    ,sw)
IP= df.VP*df.DEN
PS= df.VP/df.VS
plt.scatter(IP,PS,c=df.eff_stress,edgecolors='
    grey',
s=80,alpha=1,cmap='Greens_r')
```

4. Impact

Rock physics is a broad field of research with a wide range of applications. As demonstrated by previous examples, **rockphypy** is a versatile and flexible Python library that enables users to perform various modeling tasks simply and efficiently while avoiding the hard coding of many rock physics models and workflows.

Introducing this Python library is instrumental in its adoption by modern researchers seeking to perform sophisticated tasks in rock physics, leveraging the Python infrastructure for statistical and machine learning inference. To our knowledge, **rockphypy** is the first publicly available Python library that provides comprehensive and easy-to-use functionalities in rock physics. Our long-term goal is to standardize base-level rock physics modeling using **rockphypy**, making it a go-to tool for the research community.

rockphypy has already been utilized as teaching material for Reservoir seismic at NTNU since 2022, reaching out to a broad audience beyond the authors' research group. The library is also currently being used in the authors' ongoing project, as mentioned in the Acknowledgments section. The research paper has utilized **rockphypy** to build new rock physics models and there are other papers under revision or in preparation where **rockphypy** is presented.

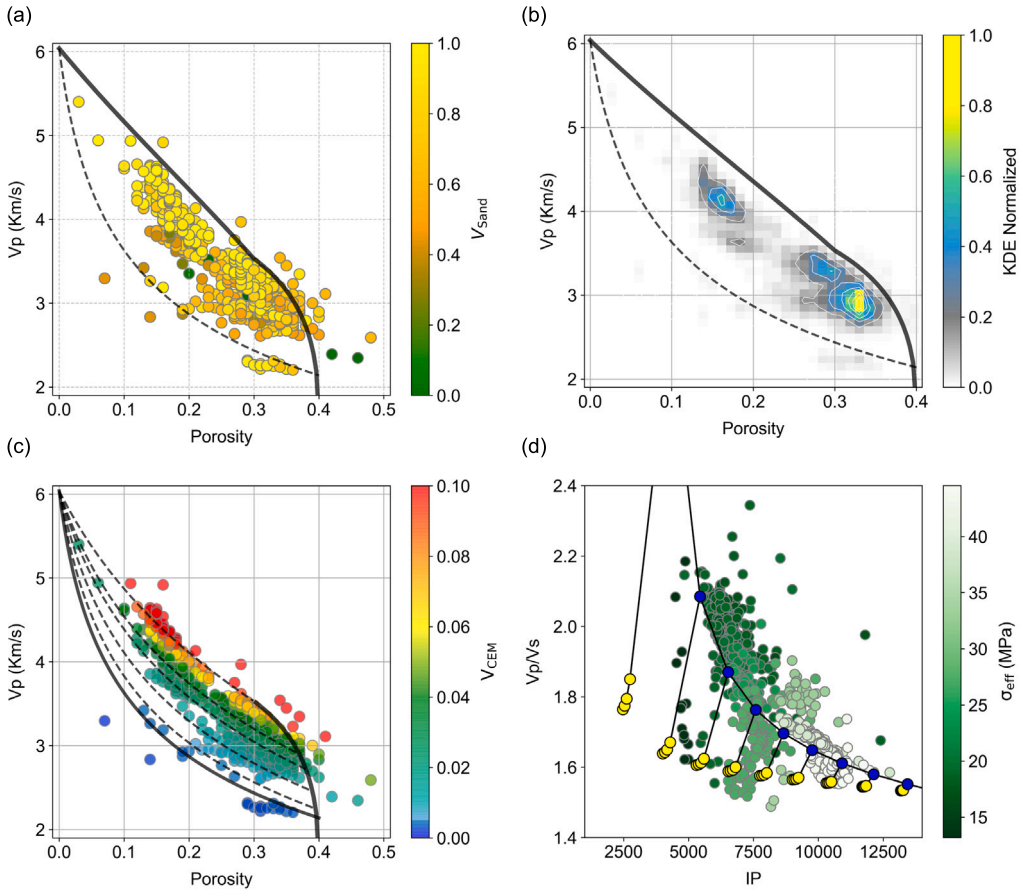

Fig. 4. (a) Rock physics screening using elastic bounds. (b) 2-D KDE plot of the data within elastic bounds defined by several rock physics models. (c) Cement amount estimation using constant cement model. (d) Rock physics template overlays the data.

5. Conclusion

We present an extensive yet easy-to-use Python library **rockphypy** for rock physics modeling. It offers a toolbox for computing elastic properties of rock, fluid, and their interactions under various conditions. The built-in workflows also simplify the process of performing broadband data analysis tasks. The library is created in a modular fashion which allows users to customize functions using pre-built functionalities, build workflows, and easily extend the library's applicability by incorporating new models. We foresee contributions from across different areas of geoscientific research and practice where rock physics modeling is involved.

Current work focuses on expanding the test case suite for automated testing. We strive to maintain the readability and reliability of the codebase while continuously enriching the library's capabilities through both internal and external contributions that adhere to strict contribution guidelines.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to the data/code in the attached file step.

Acknowledgments

This work was an exchange project by J.Y at ESE department, Stanford University in the spring of 2023 under T.M.. The authors thank the Norwegian Research Council and the industry partners of the Geophysics and Applied Mathematics in Exploration and Safe production roject (GAMES) at NTNU (Grant No. 294404) for the financial support. J.Y was also supported by the Research Stay Abroad grant funded by Norwegian Research Council. The authors want to acknowledge a number of people who contributed, directly or indirectly, to this work, particularly Kenneth Duffaut, Martin Landrø, and Rune M. Holt.

References

[1] Rathore J, Fjær E, Holt R, Renlie L. P-and S-wave anisotropy of a synthetic sandstone with controlled crack geometry. Geophys Prospect 1995;43(6):711–28.
[2] Andrä H, Combaret N, Dvorkin J, Glatt E, Han J, Kabel M, et al. Digital rock physics benchmarks—Part I: Imaging and segmentation. Comput Geosci 2013;50:25–32.
[3] Mavko G, Mukerji T, Dvorkin J. The rock physics handbook. Cambridge University Press; 2020.
[4] Yilmaz Ö. Seismic data analysis: Processing, inversion, and interpretation of seismic data. Society of Exploration Geophysicists; 2001.
[5] Fjær E, Holt RM, Horsrud P, Raaen AM. Petroleum related rock mechanics. Elsevier; 2008.
[6] Saenger EH, Enzmann F, Keehm Y, Steeb H. Digital rock physics: Effect of fluid viscosity on effective elastic properties. J Appl Geophys 2011;74(4):236–41.
[7] Avseth P, Mukerji T, Mavko G. Quantitative seismic interpretation: Applying rock physics tools to reduce interpretation risk. Cambridge University Press; 2010.
[8] Wang Z. Y2K tutorial: Fundamentals of seismic rock physics. Geophysics 2001;66(2):398–412.
[9] Daley TM. Rock physics of CO_2 storage monitoring in porous media. In: Davis TL, Landrø M, Wilson M, editors. Geophysics and geosequestration, chapter 4. Cambridge University Press; 2019, p. 71–82.

[10] Bredesen K, Rasmussen R, Mathiesen A, Nielsen LH. Seismic amplitude analysis and rock physics modeling of a geothermal sandstone reservoir in the southern part of the danish basin. Geothermics 2021;89:101974.
[11] Grana D, De Figueiredo L. SeReMpy: Seismic reservoir modeling Python library. Geophysics 2021;86(6):F61–9.
[12] Avseth P, Lehocki I, Kjøsnes Ø, Sandstad O. Data-driven rock physics analysis of north sea tertiary reservoir sands. Geophys Prospect 2021;69(3):608–21.
[13] Weinzierl W, Wiese B. Deep learning a poroelastic rock-physics model for pressure and saturation discrimination. Geophysics 2021;86(1):MR53–66.
[14] Amato del Monte A. Seismic rock physics. Leading Edge 2017;36(6):523–5.
[15] GNU general public license, version 3. 2007, Last retrieved 2020-01-01. http://www.gnu.org/licenses/gpl.html.
[16] Newnham RE. Properties of materials: Anisotropy, symmetry, structure. Oxford University Press on Demand; 2005.
[17] Batzle M, Wang Z. Seismic properties of pore fluids. Geophysics 1992;57(11):1396–408.
[18] Xu H. Calculation of CO_2 acoustic properties using Batzle-Wang equations. Geophysics 2006;71(2):F21–3.
[19] Krief M, Garat J, Stellingwerff J, Ventre J. A petrophysical interpretation using the velocities of P and S waves (full-waveform sonic). Log Anal 1990;31(06).
[20] Carman PC, Machefer J. L'écoulement des gaz à travers les milieux poreux. Paris: Bibliotheeque des Sci et Tech Nucl, Presses Universitaires de France 1961.
[21] Gernert J, Span R. EOS–CG: A Helmholtz energy mixture model for humid gases and CCS mixtures. J Chem Thermodyn 2016;93:274–93.
[22] Wang Z, Nur AM. Effects of CO_2 flooding on wave velocities in rocks with hydrocarbons. SPE Reserv Eng 1989;4(04):429–36.