

SeReMpy library

External libraries

SeReMpy uses the following Python libraries:

- NumPy 1.19.2
- SciPy 1.6.2
- matplotlib 3.4.1

Data

The folder `Data` contains multiple synthetic datasets. All the data are synthetic and covered by the MIT license included with the code.

Output

The folder `Output` contains sample output files of the proposed scripts.

Description of modules

Functions included in the `RockPhysics.py` module

<i>Function</i>	<i>Description</i>
<code>DensityModel</code>	linear porosity-density relation to compute density
<code>MatrixFluidModel</code>	Voigt-Reuss-Hill averages to compute the elastic moduli and density of the solid and fluid phases
<code>GassmannModel</code>	Gassmann's equations to compute the elastic moduli of the fluid-saturated rock
<code>VelocityDefinitions</code>	definitions of P-wave and S-wave velocity
<code>LinearizedRockPhysicsModel</code>	linear rock physics model based on a multilinear regression to compute P-wave and S-wave velocity and density
<code>WyllieModel</code>	Wyllie equation to compute P-wave velocity
<code>RaymerModel</code>	Raymer's equation to compute P-wave velocity
<code>SoftsandModel</code>	Dvorkin's soft sand model to compute P-wave and S-wave velocity
<code>StiffsandModel</code>	Dvorkin's stiff sand model to compute P-wave and S-wave velocity
<code>SphericalInclusionModel</code>	inclusion model for spherical pores to compute P-wave and S-wave velocity
<code>BerrymanInclusionModel</code>	Berryman's inclusion model for prolate and oblate pores to compute P-wave and S-wave velocity

Functions included in the `Geostats.py` module

<i>Function</i>	<i>Description</i>
<code>ExpCov</code>	exponential spatial covariance model
<code>GauCov</code>	Gaussian spatial covariance model
<code>SphCov</code>	spherical spatial covariance model
<code>SpatialCovariance1D</code>	1D spatial covariance function according to one of the available models (exponential, Gaussian, and spherical);
<code>RadialCorrLength</code>	radial correlation length for 2D spatial covariance functions
<code>SpatialCovariance2D</code>	2D spatial covariance function according to one of the available models (exponential, Gaussian, and spherical);
<code>SimpleKriging</code>	simple kriging interpolation at a given location based on a set of measurements
<code>OrdinaryKriging</code>	ordinary kriging interpolation at a given location based on a set of measurements
<code>IndicatorKriging</code>	indicator kriging interpolation at a given location based on a set of measurements
<code>GaussianSimulation</code>	Gaussian simulation to generate a sample of a random variable at a given location based on a set of measurements
<code>RandDisc</code>	Simulation of a discrete random variable with given probability mass function
<code>SeqGaussianSimulation</code>	Sequential Gaussian Simulation method to generate spatially correlated realizations of a continuous random variable based on a set of measurements given a set of location coordinates
<code>SeqIndicatorSimulation</code>	Sequential Indicator Simulation method to generate spatially correlated realizations of a discrete random variable based on a set of measurements given a set of location coordinates
<code>CorrelatedSimulation</code>	sampling approach to simulate spatially correlated stochastic (1D) realizations of multiple random variables given a set of location coordinates
<code>MarkovChainSimulation</code>	sampling approach to simulate multiple 1-dimensional realizations of a discrete random variable based on a stationary first-order Markov chain

Functions included in the `Inversion.py` module

<i>Function</i>	<i>Description</i>
<code>RickerWavelet</code>	Ricker wavelet with given dominant frequency
<code>AkiRichardsCoefficientsMatrix</code>	Aki Richards coefficient matrix
<code>DifferentialMatrix</code>	differential matrix for discrete differentiation
<code>WaveletMatrix</code>	wavelet Toeplitz matrix for discrete convolution
<code>SeismicModel</code>	synthetic seismic data according to a linearized seismic model based on the convolution of a wavelet and the linearized approximation of Zoeppritz equations
<code>SeismicInversion</code>	posterior distribution of elastic properties according to the Bayesian linearized AVO inversion
<code>RockPhysicsLinGaussInversion</code>	posterior distribution of petrophysical properties conditioned on elastic properties assuming a Gaussian distribution and a linear rock physics model
<code>RockPhysicsLinGaussMixInversion</code>	posterior distribution of petrophysical properties conditioned on elastic properties assuming a Gaussian mixture distribution and a linear rock physics model
<code>RockPhysicsGaussInversion</code>	posterior distribution of petrophysical properties conditioned on elastic properties assuming a Gaussian distribution estimated from a training dataset
<code>RockPhysicsGaussMixInversion</code>	posterior distribution of petrophysical properties conditioned on elastic properties assuming a Gaussian mixture distribution estimated from a training dataset
<code>RockPhysicsKDEInversion</code>	posterior distribution of petrophysical properties conditioned on elastic properties assuming a non-parametric distribution estimated from a training dataset using kernel density estimation
<code>EpanechnikovKernel</code>	Epanechnikov kernel used in kernel density estimation
<code>EnsembleSmotherMDA</code>	updated model realizations of the model variables conditioned on seismic data using the ES-MDA method
<code>LogitBounded</code>	logit transformation for bounded properties
<code>InvLogitBounded</code>	inverse logit transformation for bounded properties

Functions included in the `Facies.py` module

<i>Function</i>	<i>Description</i>
<code>BayesGaussFaciesClass</code>	Bayesian facies classification assuming a multivariate Gaussian distribution of the continuous properties
<code>BayesKDEFaciesClass</code>	Bayesian facies classification assuming a multivariate non-parametric distribution of the continuous properties
<code>ConfusionMatrix</code>	classification confusion matrix