PyRISM Documentation

Release 20.03.2018

Ismail Baris

Jun 08, 2018

Contents

CHAPTER 1

Introduction

This repository contains the Python bindings to different radar and optical backscattering and reflectance models, respectively. The bindings implement the following models:

1.1 Optical Models:

- **PROSPECT**: Leaf reflectance model (versions 5 and D).
- **SAIL**: Canopy reflectance model.
- **PROSAIL**: Combination of PROSPECT and SAIL.
- **LSM**: Simple Lambertian soil reflectance model.
- **Volume Scattering**: Compute volume scattering functions and interception coefficients for given solar zenith, viewing zenith, azimuth and leaf inclination angle.

1.2 RADAR Models:

- **Rayleigh**: Calculate the extinction coefficients in terms of Rayleigh scattering.
- **Mie**: Calculate the extinction coefficients in terms of Mie scattering.
- **Dielectric Constants**: Calculate the dielectric constant of different objects like water, saline water, soil and vegetation.
- **I2EM**: RADAR soil scattering model to compute the backscattering coefficient VV and HH polarized.
- **Emissivity**: Calculate the emissivity for single-scale random surface for Bi and Mono-static acquisitions.

For the optical models the code from José Gómez-Dans was used as a benchmark. The theory of the radar models is from F.T. Ulaby

1.3 Indices and tables

- genindex
- modindex
- search

CHAPTER 2

Installation

There are currently different methods to install *prism*.

2.1 Using pip

The ‘ prism ‘ package is provided on pip. You can install it with:

```
pip install prism
```

2.2 Standard Python

You can also download the source code package from this repository or from pip. Unpack the file you obtained into some directory (it can be a temporary directory) and then run:

```
python setup.py install
```

2.3 Test installation success

Independent how you installed ‘ prism ‘, you should test that it was sucessfull by the following tests:

```
python -c "from prism import I2EM"
```

If you don’t get an error message, the module import was sucessfull.

2.4 Indices and tables

- genindex

- modindex
- search

CHAPTER 3

Technical documentation

Contents:

3.1 Core Functions

class pyrism.core.**Kernel**(*iza*, *vza*, *raa*, *normalize=False*, *nbar=0.0*, *angle_unit='DEG'*, *align=True*)

Bases: object

The kernel object defines the different models.

Parameters

- **vza, raa** (*iza,*) – Incidence (iza) and scattering (vza) zenith angle, as well as relative azimuth (raa) angle.
- **normalize** (*boolean, optional*) – Set to 'True' to make kernels 0 at nadir view illumination. Since all implemented kernels are normalized the default value is False.
- **nbar** (*float, optional*) – The sun or incidence zenith angle at which the isotropic term is set to if normalize is True. The default value is 0.0.
- **angle_unit** (*{'DEG', 'RAD'}, optional*) –
  - 'DEG': All input angles (iza, vza, raa) are in [DEG] (default).
  - 'RAD': All input angles (iza, vza, raa) are in [RAD].
- **align** (*boolean, optional*) – Expand all input values to the same length (default).

Returns

- *All returns are attributes!*
- **iza** (*ndarray*) – Sun or incidence zenith angle in [RAD].
- **vza** (*ndarray*) – View or scattering zenith angle in [RAD].
- **raa** (*ndarray*) – Relative azimuth angle in [RAD].

- **izaDeg** (*ndarray*) – Sun or incidence zenith angle in [DEG].
- **vzaDeg** (*ndarray*) – View or scattering zenith angle in [DEG].
- **raaDeg** (*ndarray*) – Relative azimuth angle in [DEG].
- **phi** (*ndarray*) – Relative azimuth angle in a range between 0 and 2pi.

Note: Hot spot direction is vza == iza and raa = 0.0

normalization(*kernel=None*, *args=None*)

class pyrism.core.**Scattering**(*frequency*, *particle_size*, *diel_constant_p*, *diel_constant_b*)
Bases: object

Calculate the extinction coefficients in terms of Rayleigh or Mie scattering from *[UL15a]* and *[UL15b]*.

Parameters

- **frequency** (*int or float*) – Frequency (GHz)
- **particle_size** (*int, float or array*) – Particle size a (cm).
- **diel_constant_p** (*complex*) – Dielectric constant of the medium.
- **diel_constant_b** (*complex*) – Dielectric constant of the background.

3.1.1 References

3.2 Manage Results

class pyrism.core.**ReflectanceResult**
Bases: dict

Represents the reflectance result.

Returns

- *All returns are attributes!*
- **BSC.ref, BSC.refdB, BSC.ms.ref, BSC.ms.refdB** (*array_like*) – Radar Backscatter values (polarization-independent).
- **BSC.VV, BSC.HH, BSC.VVdB, BSC.HHdB, BSC.ms.VV, BSC.ms.HH, BSC.ms.VVdB, BSC.ms.HHdB** (*array_like*) – Radar Backscatter values (polarization-dependent).
- **BRDF.ref, BRDF.refdB, BRDF.ms.ref, BRDF.ms.refdB** (*array_like*) – BRDF reflectance values (polarization-independent).
- **BRDF.VV, BRDF.HH, BRDF.VVdB, BRDF.HHdB, BSC.ms.VV, BRDF.ms.HH, BRDF.ms.VVdB, BRDF.ms.HHdB** (*array_like*) – BRDF reflectance values (polarization-dependent).
- **BRF.ref, BRF.refdB, BRF.ms.ref, BRF.ms.refdB** (*array_like*) – BRF reflectance values (polarization-independent).
- **BRF.VV, BRF.HH, BRF.VVdB, BRF.HHdB, BSC.ms.VV, BRF.ms.HH, BRF.ms.VVdB, BRF.ms.HHdB** (*array_like*) – BRF reflectance values (polarization-dependent).

Notes

There may be additional attributes not listed above depending of the specific solver. Since this class is essentially a subclass of dict with attribute accessors, one can see which attributes are available using the *keys()* method. adar Backscatter values of multi scattering contribution of surface and volume

The attribute 'ms' is the multi scattering contribution. This is only available if it is calculated. For detailed parametrisation one can use BSC.ms.sms or BSC.ms.smv for the multiple scattering contribution of surface or volume, respectively.

class pyrism.core.**EmissivityResult**
Bases: dict

Represents the reflectance result.

Returns

- *All returns are attributes!*
- **EMS.ref, EMS.refdB** (*array_like*) – Emission values (polarization-independent).
- **EMS.VV, EMS.HH, EMS.VVdB, EMS.HHdB** (*array_like*) – Emission values (polarization-dependent).
- **EMN.VV, EMN.HH, EMN.VVdB, EMN.HHdB** (*array_like*) – Due to the several conversions in ROM this output format delivers the emission values divided through the sensing geometry times 4pi. This attribute is only for the I2EM.Emissivity class. If you want to calculate the emissivity of a scene, use this output from I2EM.Emissivity.

Notes

There may be additional attributes not listed above depending of the specific solver. Since this class is essentially a subclass of dict with attribute accessors, one can see which attributes are available using the *keys()* method. adar Backscatter values of multi scattering contribution of surface and volume

The attribute 'ms' is the multi scattering contribution. This is only available if it is calculated. For detailed parametrisation one can use BSC.ms.sms or BSC.ms.smv for the multiple scattering contribution of surface or volume, respectively.

class pyrism.core.**SailResult**
Bases: dict

Represents the sail result.

Returns

- *All returns are attributes!*
- **SDR.ref, SDR.refdB** (*array_like*) – Directional reflectance factor.
- **BHR.ref, BHR.refdB** (*array_like*) – Bi-hemispherical reflectance factor.
- **DHR.ref, DHR.refdB** (*array_like*) – Directional-Hemispherical reflectance factor.
- **HDR.ref, HDR.refdB** (*array_like*) – Hemispherical-Directional reflectance factor.

Note: All returns have in addition the attributes *L8.Bx* and *ASTER.Bx*. L8 is the Landsat 8 average reflectance values for Bx band (B2 until B7). *ASTER* is the ASTER average reflectance for Bx band (B1 until B9).

Notes

There may be additional attributes not listed above depending of the specific solver. Since this class is essentially a subclass of dict with attribute accessors, one can see which attributes are available using the *keys()* method. adar Backscatter values of multi scattering contribution of surface and volume

The attribute 'ms' is the multi scattering contribution. This is only available if it is calculated. For detailed parametrisation one can use BSC.ms.sms or BSC.ms.smv for the multiple scattering contribution of surface or volume, respectively.

3.3 Optical Models

class pyrism.models.**VolScatt**(*iza*, *vza*, *raa*, *angle_unit='DEG'*)
Bases: pyrism.core._core.Kernel

Compute volume scattering functions and interception coefficients for given solar zenith, viewing zenith, azimuth and leaf inclination angle ([Cam86], [Cam90], [VMB98]).

Parameters

- **vza, raa** (*iza,*) – Incidence (iza) and scattering (vza) zenith angle, as well as relative azimuth (raa) angle.
- **angle_unit** (*{'DEG', 'RAD'}, optional*) –
  - 'DEG': All input angles (iza, vza, raa) are in [DEG] (default).
  - 'RAD': All input angles (iza, vza, raa) are in [RAD].

Returns

- *All returns are attributes!*
- **iza** (*ndarray*) – Sun or incidence zenith angle in [RAD].
- **vza** (*ndarray*) – View or scattering zenith angle in [RAD].
- **raa** (*ndarray*) – Relative azimuth angle in [RAD].
- **izaDeg** (*ndarray*) – Sun or incidence zenith angle in [DEG].
- **vzaDeg** (*ndarray*) – View or scattering zenith angle in [DEG].
- **raaDeg** (*ndarray*) – Relative azimuth angle in [DEG].
- **phi** (*ndarray*) – Relative azimuth angle in a range between 0 and 2pi.
- **chi_s** (*int, float or array_like*) – Interception function in the solar path.
- **chi_o** (*int, float or array_like*) – Interception function in the view path.

Note: Hot spot direction is vza == iza and raa = 0.0

See also:

VolScatt.coef, *LIDF.campbell*, *LIDF.verhoef*, *LIDF.nilson*

coef(*lidf_type='verhoef'*, *n_elements=18*, ***kwargs*)
Calculate the extinction and volume scattering coefficients ([Cam86], [Cam90], [VMB98]).

Parameters

- **lidf_type** (*{'verhoef', 'campbell'}*) – Define with which method the LIDF is calculated
- **n_elements** (*int, optional*) – Total number of equally spaced inclination angles. Default is 18.
- **kwargs** (*dict*) –

  Possible **kwargs from campbell method:

  - a : Mean leaf angle (degrees) use 57 for a spherical LIDF.

  Possible **kwargs from verhoef method:

  - a : Parameter a controls the average leaf inclination.
  - **b** [Parameter b influences the shape of the distribution (bimodality), but has no effect on the] average leaf inclination.

Returns

- *All returns are attributes!*
- **self.ks** (*int, float or array_like*) – Volume scattering coeffient in incidence path.
- **self.ko** (*int, float or array_like*) – Volume scattering coeffient in scattering path.
- **self.Fs** (*int, float or array_like*) – Function to be multiplied by leaf reflectance to obtain the volume scattering.
- **self.Ft** (*int, float or array_like*) – Function to be multiplied by leaf transmittance to obtain the volume scattering.
- **self.Fst** (*int, float or array_like*) – Sum of Fs and Ft.

See also:

LIDF.campbell(), *LIDF.verhoef()*, *LIDF.nilson()*

volume(*lza*)

Compute volume scattering functions and interception coefficients for given solar zenith, viewing zenith, azimuth and leaf inclination angle (*[VMB98]*, *[Cam90]*).

Returns

- *All returns are attributes!*
- **chi_s** (*float*) – Interception function in the solar path.
- **chi_o** (*float*) – Interception function in the view path.
- **frho** (*float*) – Function to be multiplied by leaf reflectance to obtain the volume scattering.
- **ftau** (*float*) – Function to be multiplied by leaf transmittance to obtain the volume scattering.

class pyrism.models.**LIDF**

Calculate several leaf area inclination density function based on *[Cam90]*, *[VMB98]* or *[NK89]*.

See also:

LIDF.campell, *LIDF.verhoef*, *LIDF.nilson*

Note: This class contains only static methods.

static campbell(*n_elements=18*)

Calculate the Leaf Inclination Distribution Function based on the mean angle of ellipsoidal LIDF distribution. :param a: Mean leaf angle (degrees) use 57 for a spherical LIDF. :type a: float :param n_elements: Total number of equally spaced inclination angles . :type n_elements: int

Returns **lidf** – Leaf Inclination Distribution Function for 18 equally spaced angles.

Return type list

static nilson(*lza*, *mla=None*, *eccentricity=0.5*, *scaling_factor=0.5*, *distribution='random'*)

Leaf Angle Distributions (LAD) from Nilson and Kuusk.

Note:

If mla is None, the default values are alculated by following distributions:

- 'erectophile': 0
- 'planophile': pi/2
- 'plagiophile': pi/4
- 'erectophile': 0
- 'random' : This determines the output to 1
- 'uniform' : This determines the output to 0.5

Parameters

- **lza** (*int, float or ndarray*) – Leaf zenith angle (lza).
- **mla** (*int or float, optional*) – Modal leaf angle in [Deg], Default is None (See Note).
- **eccentricity** (*int or float (default = 0.5), optional*) – Zero eccentricity is a spherical leaf angle distribution. An eccentricity of 1 is a 'needle'.
- **scaling_factor** (*int or float (default = 0.5), optional*) – Scaling factor (reflectance if lza = 0)
- **distribution** (*{'erectophile', 'planophile', 'plagiophile', 'random', 'uniform'}, optional*) – Default distribution which set the mla. Default is 'random'

Returns **LAD** – LAD integrated over a sphere (0 - pi/2)

Return type int, float or array_like

static verhoef(*b*, *n_elements=18*)

Calculate the Leaf Inclination Distribution Function based on the Verhoef's bimodal LIDF distribution.

Parameters

- **b** (*a,*) – Parameter a controls the average leaf inclination. Parameter b influences the shape of the distribution (bimodality), but has no effect on the average leaf inclination.
- **n_elements** (*int*) – Total number of equally spaced inclination angles.

Returns **LAD** – Leaf Inclination Distribution Function at equally spaced angles.

Return type list

Note: The parameters must be chosen such that |a| + |b| < 1. Some possible distributions are [a, b]:

- Planophile: [1, 0].
- Erectophile: [-1, 0].
- Plagiophile: [0,-1].
- Extremophile: [0,1].
- Spherical: [-0.35,-0.15].
- Uniform: [0,0].

---

class pyrism.models.**PROSPECT**(*N*, *Cab*, *Cxc*, *Cbr*, *Cw*, *Cm*, *Can=0*, *alpha=40*, *version='5'*)

PROSPECT D and 5 (including carotenoids and brown pigments) version b (october, 20th 2009) ([JB90], [FFranccoisA+08], [Bar])

Parameters

- **N** (*int or float*) – Leaf structure parameter.
- **Cab** (*int or float*) – Chlorophyll a+b content.
- **Cxc** (*int or float*) – Carotenoids content.
- **Cbr** (*int or float*) – Brown pigments content in arbitrary units.
- **Cw** (*int or float*) – Equivalent water thickness.
- **Cm** (*int or float*) – Dry matter content
- **alpha** (*int*) – Mean leaf angle (degrees) use 57 for a spherical LIDF. Default is 40.
- **version** (*{'5', 'D'}*) – PROSPECT version. Default is '5'.

Returns

- *All returns are attributes!*
- **L8.Bx.kx** (*namedtuple (with dot access)*) – Landsat 8 average kx (ks, kt, ke) values for Bx band (B2 until B7):
- **ASTER.Bx.kx** (*namedtuple (with dot access)*) – ASTER average kx (ks, kt, ke) values for Bx band (B1 until B9):
- **l** (*array_like*) – Continuous Wavelength from 400 until 2500 nm.
- **kt** (*array_like*) – Continuous Transmission from 400 until 2500 nm.
- **ks** (*array_like*) – Continuous Scattering from 400 until 2500 nm.
- **ke** (*array_like*) – Continuous Extinction from 400 until 2500 nm.
- **ka** (*array_like*) – Continuous Absorption from 400 until 2500 nm.
- **om** (*array_like*) – Continuous Omega value in terms of Radar from 400 until 2500 nm.

cleanup(*name*)

Do cleanup for an attribute

indices()

select(*mins=None*, *maxs=None*, *function='mean'*)

Returns the means of the coefficients in range between min and max.

Parameters

- **mins** (*int*) – Lower bound of the wavelength (400 - 2500)
- **maxs** (*int*) – Upper bound of the wavelength (400 - 2500)
- **function** (*{'mean'}, optional*) – Specify how the bands are calculated.

Returns **Band** – Reflectance in the selected range.

Return type array_like

class pyrism.models.**LSM**(*reflectance*, *moisture*)

In optical wavelengths the Lambertian Linear Model (LSM) is used. If you want to calculate the RO Model in optical terms you will calculate the surface reflexion previous.

Equation: Total Soil Reflectance = Reflectance*(Moisture*soil_spectrum1+(1-Moisture)*soil_spectrum2)

By default, soil_spectrum1 is a dry soil, and soil_spectrum2 is a wet soil, so in that case, 'moisture' is a surface soil moisture parameter. `reflectance` is a soil brightness term. You can provide one or the two soil spectra if you want. The calculation is between 400 and 2500 nm with 1nm spacing.

Parameters

- **reflectance** (*int or float*) – Surface (Lambertian) reflectance in optical wavelength.
- **moisture** (*int or float*) – Surface moisture content between 0 and 1.

Returns

- *All returns are attributes!*
- **self.L8** (*namedtuple (with dot access)*) – Landsat 8 average kx (ks, kt, ke) values for Bx band (B2 until B7)
- **self.ASTER** (*namedtuple (with dot access)*) – ASTER average kx (ks, kt, ke) values for Bx band (B1 until B9)
- **self.ref** (*dict (with dot access)*) – Continuous surface reflectance values from 400 until 2500 nm
- **self.l** (*dict (with dot access)*) – Continuous Wavelength values from 400 until 2500 nm

cleanup(*name*)

Do cleanup for an attribute

select(*mins*, *maxs*, *function='mean'*)

Returns the means of the coefficients in range between min and max.

Args:

min (int) Lower bound of the wavelength (400 - 2500)

max (int) Upper bound of the wavelength (400 - 2500)

class pyrism.models.**SAIL**(*iza*, *vza*, *raa*, *ks*, *kt*, *lai*, *hotspot*, *rho_surface*, *lidf_type='campbell'*, *a=57*, *b=0*, *normalize=False*, *nbar=0.0*, *angle_unit='DEG'*)

Bases: pyrism.core._core.Kernel

Run the SAIL radiative transfer model (See Note) ([*GomezD18*]).

Parameters

- **vza, raa** (*iza,*) – Incidence (iza) and scattering (vza) zenith angle, as well as relative azimuth (raa) angle.
- **kt** (*ks,*) – Continuous leaf reflection (ks) and leaf transmission (kt) values from from 400 until 2500 nm. One can use the output from PROSPECT class instance.

- **lai** (*float*) – Leaf area index.
- **hotspot** (*float*) – The hotspot parameter.
- **rho_surface** (*array_like*) – Continuous surface reflectance values from from 400 until 2500 nm. One can use the output from LSM class instance.
- **lidf_type** (*{'verhoef', 'campbell'}, optional*) – Define with which method the LIDF is calculated. Default is 'campbell'
- **b** (*a,*) –

  Parameter a and b depends on which lidf_type is applied:

  - If lidf_type is 'verhoef': Parameter a controls the average leaf inclination. Parameter b influences the shape of the distribution (bimodality), but has no effect on the average leaf inclination. The default values are for a uniform leaf distribution a = 0, b = 0.
  - If lidf_type is 'campbell': Parameter a is the mean leaf angle (degrees) use 57 for a spherical LIDF. The default value represents a spherical leaf distribution a = 57.
- **normalize** (*boolean, optional*) – Set to 'True' to make kernels 0 at nadir view illumination. Since all implemented kernels are normalized the default value is False.
- **nbar** (*float, optional*) – The sun or incidence zenith angle at which the isotropic term is set to if normalize is True. The default value is 0.0.
- **angle_unit** (*{'DEG', 'RAD'}, optional*) –
  - 'DEG': All input angles (iza, vza, raa) are in [DEG] (default).
  - 'RAD': All input angles (iza, vza, raa) are in [RAD].

Returns

Return type For more attributes see also pyrism.core.Kernel and pyrism.core.SailResult.

See also:

pyrism.core.Kernel, *pyrism.core.SailResult*

Note: If the input parameter for ks and kt are the output from the class PROSPECT, SAIL will calculate the PROSAIL model.

3.3.1 References

3.4 RADAR Models

class pyrism.models.**Rayleigh**(*frequency*, *particle_size*, *diel_constant_p*, *diel_constant_b=(1+1j)*)

Bases: pyrism.core._core.Scattering

Calculate the extinction coefficients in terms of Rayleigh scattering (*[UL15a]* and *[UL15b]*).

Parameters

- **frequency** (*int or float*) – Frequency (GHz)
- **particle_size** (*int, float or array*) – Particle size a [m].
- **diel_constant_p** (*complex*) – Dielectric constant of the medium.

- **diel_constant_b** (*complex*) – Dielectric constant of the background.

Returns

- *All returns are attributes!*
- **self.ke** (*int, float or array_like*) – Extinction coefficient.
- **self.ks** (*int, float or array_like*) – Scattering coefficient.
- **self.ka** (*int, float or array_like*) – Absorption coefficient.
- **self.om** (*int, float or array_like*) – Omega.
- **self.s0** (*int, float or array_like*) – Backscatter coefficient sigma 0.

class pyrism.models.**Mie**(*frequency*, *particle_size*, *diel_constant_p*, *diel_constant_b=(1+1j)*)

Bases: pyrism.core._core.Scattering

Calculate the extinction coefficients in terms of Mie scattering (*[UL15a]* and *[UL15b]*).

Parameters

- **frequency** (*int or float*) – Frequency (GHz)
- **particle_size** (*int, float or array*) – Particle size a [m].
- **diel_constant_p** (*complex*) – Dielectric constant of the medium.
- **diel_constant_b** (*complex*) – Dielectric constant of the background.

Returns

- *All returns are attributes!*
- **self.ke** (*int, float or array_like*) – Extinction coefficient.
- **self.ks** (*int, float or array_like*) – Scattering coefficient.
- **self.ka** (*int, float or array_like*) – Absorption coefficient.
- **self.om** (*int, float or array_like*) – Omega.
- **self.s0** (*int, float or array_like*) – Backscatter coefficient sigma 0.

class pyrism.models.**DielConstant**

Class to calculate the Dielectric Constant of different objects (*[UL15a]* and *[UL15b]*).

See also:

DielConstant.pureWater, DielConstant.salineWater, *DielConstant.soil*, *DielConstant.vegetation*, *DielConstant.combine*

static combine(*mg*, *temp*, *S*, *C*, *mv*, *rho_b=1.7*)

Combine the Relative Dielectric Constant of Vegetation with Soil. Computes the real and imaginary parts of the relative dielectric constant of vegetation material, such as corn leaves, in the microwave region.

Computes the real and imaginary parts of the relative dielectric constant of soil at a given temperature 0<t<40C, frequency, volumetric moisture content, soil bulk density, sand and clay fractions.

Parameters

- **frequency** (*int, float or array_like*) – Frequency (GHz).
- **mg** (*int or float*) – Gravimetric moisture content (0<mg< 1).
- **temp** (*int, float or array*) – Temperature in C° (0 - 30).
- **S** (*int or float*) – Sand fraction in %.

- **C** (*int or float*) – Clay fraction in %.
- **mv** (*int or float*) – Volumetric Water Content (0<mv<1)
- **rho_b** (*int or float (default = 1.7)*) – Bulk density in g/cm3 (typical value is 1.7 g/cm3).

static saline_water(*temp*, *salinity*)

Relative Dielectric Constant of Saline Water. Computes the real and imaginary parts of the relative dielectric constant of water at any temperature 0<t<30, Salinity 0<Salinity<40 0/00, and frequency 0<f<1000GHz

Parameters

- **frequency** (*int, float or array_like*) – Frequency (GHz).
- **temp** (*int, float or array*) – Temperature in C° (0 - 30).
- **salinity** (*int, float or array*) – Salinity in parts per thousand.

Returns **Dielectric Constant**

Return type complex

static soil(*temp*, *S*, *C*, *mv*, *rho_b=1.7*)

Relative Dielectric Constant of soil. Computes the real and imaginary parts of the relative dielectric constant of soil at a given temperature 0<t<40C, frequency, volumetric moisture content, soil bulk density, sand and clay fractions.

Parameters

- **frequency** (*int, float or array_like*) – Frequency (GHz).
- **temp** (*int, float or array*) – Temperature in C° (0 - 30).
- **S** (*int or float*) – Sand fraction in %.
- **C** (*int or float*) – Clay fraction in %.
- **mv** (*int or float*) – Volumetric Water Content (0<mv<1)
- **rho_b** (*int or float (default = 1.7)*) – Bulk density in g/cm3 (typical value is 1.7 g/cm3).

Returns **Dielectric Constant**

Return type complex

static vegetation(*mg*)

Relative Dielectric Constant of Vegetation. Computes the real and imaginary parts of the relative dielectric constant of vegetation material, such as corn leaves, in the microwave region.

Parameters

- **frequency** (*int, float or array_like*) – Frequency (GHz).
- **mg** (*int or float*) – Gravimetric moisture content (0<mg< 1).

Returns **Dielectric Constant**

Return type complex

static water(*temp*)

Relative Dielectric Constant of Pure Water. Computes the real and imaginary parts of the relative dielectric constant of water at any temperature 0<t<30 and frequency 0<f<1000 GHz. Uses the double-Debye model.

Parameters

- **frequency** (*int, float or array_like*) – Frequency (GHz).
- **temp** (*int, float or array*) – Temperature in C° (0 - 30).

Returns Dielectric Constant

Return type complex

class pyrism.models.**I2EM**(*iza*, *vza*, *raa*, *normalize=True*, *nbar=0.0*, *angle_unit='DEG'*, *frequency=None*, *diel_constant=None*, *corrlength=None*, *sigma=None*, *n=10*, *corrfunc='exponential'*)

Bases: pyrism.core._core.Kernel

RADAR Surface Scatter Based Kernel (I2EM). Compute BSC VV and BSC HH and the emissivity for single-scale random surface for Bi and Mono-static acquisitions (*[UL15a]* and *[UL15b]*).

Parameters

- **vza, raa** (*iza,*) – Incidence (iza) and scattering (vza) zenith angle, as well as relative azimuth (raa) angle.
- **normalize** (*boolean, optional*) – Set to 'True' to make kernels 0 at nadir view illumination. Since all implemented kernels are normalized the default value is False.
- **nbar** (*float, optional*) – The sun or incidence zenith angle at which the isotropic term is set to if normalize is True. The default value is 0.0.
- **angle_unit** (*{'DEG', 'RAD'}, optional*) –
  - 'DEG': All input angles (iza, vza, raa) are in [DEG] (default).
  - 'RAD': All input angles (iza, vza, raa) are in [RAD].
- **frequency** (*int or float*) – RADAR Frequency (GHz).
- **diel_constant** (*int or float*) – Complex dielectric constant of soil.
- **corrlength** (*int or float*) – Correlation length (cm).
- **sigma** (*int or float*) – RMS Height (cm)
- **n** (*int (default = 10), optinal*) – Coefficient needed for x-power and x-exponential correlation function.
- **corrfunc** (*{'exponential', 'gaussian', 'xpower', 'mixed'}, optional*) – Correlation distribution functions. The *mixed* correlation function is the result of the division of gaussian correlation function with exponential correlation function. Default is 'exponential'.

Returns

Return type For more attributes see also pyrism.core.Kernel and pyrism.core.ReflectanceResult.

See also:

I2EM.Emissivity, *pyrism.core.Kernel*, *pyrism.core.ReflectanceResult*

Note: The model is constrained to realistic surfaces with (rms height / correlation length) 0.25. Hot spot direction is vza == iza and raa = 0.0

---

class Emissivity(*iza*, *vza*, *raa*, *normalize=False*, *nbar=0.0*, *angle_unit='DEG'*, *frequency=1.26*, *diel_constant=(10+1j)*, *corrlength=10*, *sigma=0.3*, *corrfunc='exponential'*)

Bases: pyrism.core._core.Kernel

This Class calculates the emission from rough surfaces using the I2EM Model.

Parameters

- **vza, raa** (*iza,*) – Incidence (iza) and scattering (vza) zenith angle, as well as relative azimuth (raa) angle.
- **normalize** (*boolean, optional*) – Set to 'True' to make kernels 0 at nadir view illumination. Since all implemented kernels are normalized the default value is False.
- **nbar** (*float, optional*) – The sun or incidence zenith angle at which the isotropic term is set to if normalize is True. The default value is 0.0.
- **angle_unit** (*{'DEG', 'RAD'}, optional*) –
  - 'DEG': All input angles (iza, vza, raa) are in [DEG] (default).
  - 'RAD': All input angles (iza, vza, raa) are in [RAD].
- **frequency** (*int or float*) – RADAR Frequency (GHz).
- **diel_constant** (*int or float*) – Complex dielectric constant of soil.
- **corrlength** (*int or float*) – Correlation length (cm).
- **sigma** (*int or float*) – RMS Height (cm)
- **corrfunc** (*{'exponential', 'gaussian', 'mixed'}, optional*) – Correlation distribution functions. The *mixed* correlation function is the result of the division of gaussian correlation function with exponential correlation function. Default is 'exponential'.

Returns

Return type For attributes see also core.Kernel and core.EmissivityResult.

See also:

pyrism.core.EmissivityResult

emsv_integralfunc(*x*, *y*)

3.4.1 References

3.5 Indices and tables

- genindex
- modindex
- search

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Bibliography

[UL15a] Fawwaz T. Ulaby and David G. Long. *Microwave Radar and Radiometric Remote Sensing*. Artech House, Norwood, 2015. ISBN 978-0-472-11935-6. URL: http://gbv.eblib.com/patron/FullRecord.aspx?p=4537961.

[UL15b] Fawwaz T. Ulaby and David G. Long. Microwave radar and radiometric remote sensing: remote sensing computer codes. 2015. URL: http://mrs.eecs.umich.edu/microwave_remote_sensing_computer_codes.html.

[Bar] Frederic Baret. The specific absorption coefficient corresponding to brown pigment. emmah, inra avignon.

[Cam86] G. S. Campbell. Extinction coefficients for radiation in plant canopies calculated using an ellipsoidal inclination angle distribution. *Agricultural and Forest Meteorology*, 36(4):317–321, 1986. doi:10.1016/0168-1923(86)90010-9.

[Cam90] G.S Campbell. Derivation of an angle density function for canopies with ellipsoidal leaf angle distributions. *Agricultural and Forest Meteorology*, 49(3):173–176, 1990. doi:10.1016/0168-1923(90)90030-A.

[FFranccoisA+08] Jean-Baptiste Feret, Christophe François, Gregory P. Asner, Anatoly A. Gitelson, Roberta E. Martin, Luc P.R. Bidel, Susan L. Ustin, Guerric Le Maire, and Stéphane Jacquemoud. Prospect-4 and 5: advances in the leaf optical properties model separating photosynthetic pigments. *Remote Sensing of Environment*, 112(6):3030–3043, 2008. doi:10.1016/j.rse.2008.02.012.

[GomezD18] José Gómez-Dans. Prosail implementation for python. 2018. URL: https://github.com/jgomezdans/prosail.

[JB90] S. Jacquemoud and F. Baret. Prospect: a model of leaf optical properties spectra. *Remote Sensing of Environment*, 34(2):75–91, 1990. doi:10.1016/0034-4257(90)90100-Z.

[NK89] Tiit Nilson and Andres Kuusk. A reflectance model for the homogeneous plant canopy and its inversion. *Remote Sensing of Environment*, 27(2):157–167, 1989. doi:10.1016/0034-4257(89)90015-1.

[UL15a] Fawwaz T. Ulaby and David G. Long. *Microwave Radar and Radiometric Remote Sensing*. Artech House, Norwood, 2015. ISBN 978-0-472-11935-6. URL: http://gbv.eblib.com/patron/FullRecord.aspx?p=4537961.

[UL15b] Fawwaz T. Ulaby and David G. Long. Microwave radar and radiometric remote sensing: remote sensing computer codes. 2015. URL: http://mrs.eecs.umich.edu/microwave_remote_sensing_computer_codes.html.

[VMB98] W. Verhoef, M. Molenaar, and N.J.J Bunnik. *Theory of radiative transfer models applied in optical remote sensing of vegetation canopies*. Landbouwuniversiteit Wageningen (LUW), Wageningen, 1998. ISBN 9054858044.

[Bar] Frederic Baret. The specific absorption coefficient corresponding to brown pigment. emmah, inra avignon.

[Cam86] G. S. Campbell. Extinction coefficients for radiation in plant canopies calculated using an ellipsoidal inclination angle distribution. *Agricultural and Forest Meteorology*, 36(4):317–321, 1986. doi:10.1016/0168-1923(86)90010-9.

[Cam90] G.S Campbell. Derivation of an angle density function for canopies with ellipsoidal leaf angle distributions. *Agricultural and Forest Meteorology*, 49(3):173–176, 1990. doi:10.1016/0168-1923(90)90030-A.

[FFranccoisA+08] Jean-Baptiste Feret, Christophe François, Gregory P. Asner, Anatoly A. Gitelson, Roberta E. Martin, Luc P.R. Bidel, Susan L. Ustin, Guerric Le Maire, and Stéphane Jacquemoud. Prospect-4 and 5: advances in the leaf optical properties model separating photosynthetic pigments. *Remote Sensing of Environment*, 112(6):3030–3043, 2008. doi:10.1016/j.rse.2008.02.012.

[GomezD18] José Gómez-Dans. Prosail implementation for python. 2018. URL: https://github.com/jgomezdans/prosail.

[JB90] S. Jacquemoud and F. Baret. Prospect: a model of leaf optical properties spectra. *Remote Sensing of Environment*, 34(2):75–91, 1990. doi:10.1016/0034-4257(90)90100-Z.

[NK89] Tiit Nilson and Andres Kuusk. A reflectance model for the homogeneous plant canopy and its inversion. *Remote Sensing of Environment*, 27(2):157–167, 1989. doi:10.1016/0034-4257(89)90015-1.

[UL15a] Fawwaz T. Ulaby and David G. Long. *Microwave Radar and Radiometric Remote Sensing*. Artech House, Norwood, 2015. ISBN 978-0-472-11935-6. URL: http://gbv.eblib.com/patron/FullRecord.aspx?p=4537961.

[UL15b] Fawwaz T. Ulaby and David G. Long. Microwave radar and radiometric remote sensing: remote sensing computer codes. 2015. URL: http://mrs.eecs.umich.edu/microwave_remote_sensing_computer_codes.html.

[VMB98] W. Verhoef, M. Molenaar, and N.J.J Bunnik. *Theory of radiative transfer models applied in optical remote sensing of vegetation canopies*. Landbouwuniversiteit Wageningen (LUW), Wageningen, 1998. ISBN 9054858044.

Python Module Index

p

Index