

---

# **Community Land Model**

***Release 1.0.0***

**Grant Johnson, Kevin Yi, Henry Stone**

**Dec 16, 2022**



**CONTENTS:**

<b>1</b>	<b>Community_Land_Model</b>	<b>1</b>
1.1	DIAGNOSTICS package . . . . .	1
1.2	EXAMPLES package . . . . .	1
1.3	SRC package . . . . .	2
1.4	TESTS package . . . . .	4
1.5	noxfile module . . . . .	5
<b>2</b>	<b>Indices and tables</b>	<b>7</b>
	<b>Python Module Index</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



## COMMUNITY\_LAND\_MODEL

### 1.1 DIAGNOSTICS package

#### 1.1.1 Submodules

#### 1.1.2 DIAGNOSTICS.diagnostics module

Routines for analysing output data

#Requires linking to the defintions:

`DIAGNOSTICS.diagnostics.diagnostics()`

`DIAGNOSTICS.diagnostics.plot_general(X, Y, str_val_x, str_val_y)`

#### 1.1.3 Module contents

### 1.2 EXAMPLES package

#### 1.2.1 Submodules

#### 1.2.2 EXAMPLES.input\_script module

Main Project loop: Inputs: None Outputs: None

#Requires linking to the definitions: initialize\_data, run\_clm

`class EXAMPLES.input_script.clm_parameters`

Bases: object

`EXAMPLES.input_script.main()`

### 1.2.3 Module contents

## 1.3 SRC package

### 1.3.1 Submodules

### 1.3.2 SRC.initialization module

Initialization routines: Inputs: structures Parameters Outputs: structures App, State, Grid

#Requires linking to the definitions:

```
class SRC.initialization.clm_app(parameters)
    Bases: object
    load()
        try load self.name.txt
    save()
        save class as self.name.txt
class SRC.initialization.clm_grid(parameters)
    Bases: object
    load()
        try load self.name.txt
    save()
        save class as self.name.txt
    time(i)
class SRC.initialization.clm_state(parameters, Grid)
    Bases: object
    load()
        try load self.name.txt
    save()
        save class as self.name.txt
class SRC.initialization.evaporation(parameters, NT, Grid)
    Bases: object
SRC.initialization.initialize_data(parameters)
class SRC.initialization.input_data(parameters, NT)
    Bases: object
class SRC.initialization.radiation(parameters, NT)
    Bases: object
class SRC.initialization.sensible_heat(parameters, NT, Grid)
    Bases: object
```

### 1.3.3 SRC.model\_absorbed\_emitted\_radiation module

model\_absorbed\_emitted\_radiation: Inputs: State, Grid, i Outputs: None

#Requires linking to the definitions:

SRC.model\_absorbed\_emitted\_radiation.run\_absorbed\_emitted\_radiation\_model(*State, Grid, App, i*)

### 1.3.4 SRC.model\_albedo module

Albedo model: Inputs: State, Grid, App, i Outputs: None

#Requires linking to the definitions: TABLES and METHODS subdirectories

SRC.model\_albedo.canopy(*State, Grid, App, i*)

SRC.model\_albedo.canopy\_model(*State, Grid, App, i, j*)

SRC.model\_albedo.lambda\_val(*d*)

SRC.model\_albedo.optical\_params(*State, i, j*)

SRC.model\_albedo.run\_albedo\_model(*State, Grid, App, i*)

SRC.model\_albedo.solar(*State, Grid, App, i*)

SRC.model\_albedo.solar\_zenith\_angle(*State, Grid, App, i*)

### 1.3.5 SRC.model\_evaporation module

Evaporation model: Inputs: State, Grid, i Outputs: None

#Requires linking to the definitions:

SRC.model\_evaporation.MO\_length(*State, Grid, App, i*)

SRC.model\_evaporation.U\_av(*State, Grid, App, i*)

SRC.model\_evaporation.air\_res(*State, Grid, App, i*)

SRC.model\_evaporation.canopy\_specific\_humidity(*State, Grid, App, i*)

SRC.model\_evaporation.friction\_velocity(*State, Grid, App, i, psi*)

L = MO\_length(State,Grid,App,i)[1] Ksi = MO\_length(State,Grid,App,i)[2] psi = psi(ksi)

SRC.model\_evaporation.humidity\_ratio(*State, Grid, App, i, psi*)

L = MO\_length(State,Grid,App,i)[1] Ksi = MO\_length(State,Grid,App,i)[2] psi = psi(ksi)

SRC.model\_evaporation.latent\_heat\_water\_vapor\_flux(*State, Grid, App, i*)

SRC.model\_evaporation.psi(*ksi*)

SRC.model\_evaporation.q\_sat(*State, Grid, App, i*)

SRC.model\_evaporation.r\_b(*State, Grid, App, i*)

SRC.model\_evaporation.ra(*State, Grid, App, i*)

```
SRC.model_evaporation.ra_prime(State, Grid, App, i)
SRC.model_evaporation.run_evaporation_model(State, Grid, App, i)
SRC.model_evaporation.temperature_ratio(State, Grid, App, i, psi)
    L = MO_length(State, Grid, App, i)[1] Ksi = MO_length(State, Grid, App, i)[2] psi = psi(ksi)
SRC.model_evaporation.vegetation_water_vapor_flux(State, Grid, App, i)
SRC.model_evaporation.water_vapor_flux(State, Grid, App, i)
```

### 1.3.6 SRC.model\_sensible\_heat\_flux module

Sensible heat flux model: Inputs: State, Grid, i Outputs: None

#Requires linking to the defintions:

```
SRC.model_sensible_heat_flux.canopy_air_temperature(State, Grid, App, i)
SRC.model_sensible_heat_flux.run_sensible_heat_flux_model(State, Grid, App, i)
SRC.model_sensible_heat_flux.sensible_heat_flux_tot(State, Grid, App, i)
SRC.model_sensible_heat_flux.sensible_heat_flux_v(State, Grid, App, i)
```

### 1.3.7 SRC.run\_routine module

Run routine, main time loop: Inputs: structures App, State, Grid Outputs: None (Calls: save to file)

#Requires linking to the defintions: file\_saves, modules: run\_XX

```
SRC.run_routine.run_clm(App, State, Grid)
```

### 1.3.8 Module contents

## 1.4 TESTS package

### 1.4.1 Submodules

#### 1.4.2 TESTS.test\_albedo module

Test for verifying the albedo module

#Requires linking to the defintions: SRC/model\_albedo.py

```
TESTS.test_albedo.test_albedo()
```



### 1.4.3 TESTS.test\_time\_integration module

Test for verifying the integration\_schemes module

#Requires linking to the definitions: SRC/METHODS/time\_integration.py

TESTS.test\_time\_integration.test\_integration\_schemes()

### 1.4.4 Module contents

## 1.5 noxfile module



## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### d

DIAGNOSTICS, 1  
DIAGNOSTICS.diagnostics, 1

### e

EXAMPLES, 2  
EXAMPLES.input\_script, 1

### S

SRC, 4  
SRC.initialization, 2  
SRC.model\_absorbed\_emitted\_radiation, 3  
SRC.model\_albedo, 3  
SRC.model\_evaporation, 3  
SRC.model\_sensible\_heat\_flux, 4  
SRC.run\_routine, 4

### t

TESTS, 5  
TESTS.test\_albedo, 4  
TESTS.test\_time\_integration, 5



## A

air\_res() (in module SRC.model\_evaporation), 3

## C

canopy() (in module SRC.model\_albedo), 3

canopy\_air\_temperature() (in module SRC.model\_sensible\_heat\_flux), 4

canopy\_model() (in module SRC.model\_albedo), 3

canopy\_specific\_humidity() (in module SRC.model\_evaporation), 3

clm\_app (class in SRC.initialization), 2

clm\_grid (class in SRC.initialization), 2

clm\_parameters (class in EXAMPLES.input\_script), 1

clm\_state (class in SRC.initialization), 2

## D

DIAGNOSTICS

module, 1

diagnostics() (in module DIAGNOSTICS.diagnostics), 1

DIAGNOSTICS.diagnostics  
module, 1

## E

evaporation (class in SRC.initialization), 2

EXAMPLES

module, 2

EXAMPLES.input\_script  
module, 1

## F

friction\_velocity() (in module SRC.model\_evaporation), 3

## H

humidity\_ratio() (in module SRC.model\_evaporation), 3

## I

initialize\_data() (in module SRC.initialization), 2

input\_data (class in SRC.initialization), 2

## L

lambda\_val() (in module SRC.model\_albedo), 3

latent\_heat\_water\_vapor\_flux() (in module SRC.model\_evaporation), 3

load() (SRC.initialization.clm\_app method), 2

load() (SRC.initialization.clm\_grid method), 2

load() (SRC.initialization.clm\_state method), 2

## M

main() (in module EXAMPLES.input\_script), 1

MO\_length() (in module SRC.model\_evaporation), 3

module

DIAGNOSTICS, 1

DIAGNOSTICS.diagnostics, 1

EXAMPLES, 2

EXAMPLES.input\_script, 1

SRC, 4

SRC.initialization, 2

SRC.model\_absorbed\_emitted\_radiation, 3

SRC.model\_albedo, 3

SRC.model\_evaporation, 3

SRC.model\_sensible\_heat\_flux, 4

SRC.run\_routine, 4

TESTS, 5

TESTS.test\_albedo, 4

TESTS.test\_time\_integration, 5

## O

optical\_params() (in module SRC.model\_albedo), 3

## P

plot\_general() (in module DIAGNOSTICS.diagnostics), 1

psi() (in module SRC.model\_evaporation), 3

## Q

q\_sat() (in module SRC.model\_evaporation), 3

## R

r\_b() (in module SRC.model\_evaporation), 3

ra() (in module SRC.model\_evaporation), 3

ra\_prime() (in module SRC.model\_evaporation), 3  
radiation (class in SRC.initialization), 2  
run\_absorbed\_emitted\_radiation\_model() (in module SRC.model\_absorbed\_emitted\_radiation), 3  
run\_albedo\_model() (in module SRC.model\_albedo), 3  
run\_clm() (in module SRC.run\_routine), 4  
run\_evaporation\_model() (in module SRC.model\_evaporation), 4  
run\_sensible\_heat\_flux\_model() (in module SRC.model\_sensible\_heat\_flux), 4

U  
U\_av() (in module SRC.model\_evaporation), 3

V  
vegetation\_water\_vapor\_flux() (in module SRC.model\_evaporation), 4

W  
water\_vapor\_flux() (in module SRC.model\_evaporation), 4

## S

save() (SRC.initialization.clm\_app method), 2  
save() (SRC.initialization.clm\_grid method), 2  
save() (SRC.initialization.clm\_state method), 2  
sensible\_heat (class in SRC.initialization), 2  
sensible\_heat\_flux\_tot() (in module SRC.model\_sensible\_heat\_flux), 4  
sensible\_heat\_flux\_v() (in module SRC.model\_sensible\_heat\_flux), 4  
solar() (in module SRC.model\_albedo), 3  
solar\_zenith\_angle() (in module SRC.model\_albedo), 3  
SRC  
    module, 4  
SRC.initialization  
    module, 2  
SRC.model\_absorbed\_emitted\_radiation  
    module, 3  
SRC.model\_albedo  
    module, 3  
SRC.model\_evaporation  
    module, 3  
SRC.model\_sensible\_heat\_flux  
    module, 4  
SRC.run\_routine  
    module, 4

## T

temperature\_ratio() (in module SRC.model\_evaporation), 4  
test\_albedo() (in module TESTS.test\_albedo), 4  
test\_integration\_schemes() (in module TESTS.test\_time\_integration), 5  
TESTS  
    module, 5  
TESTS.test\_albedo  
    module, 4  
TESTS.test\_time\_integration  
    module, 5  
time() (SRC.initialization.clm\_grid method), 2