SMRT: a new modular snow active/passive microwave radiative transfer model

Ghislain Picard (IGE), Melody Sandells (Corescience, UK) and Henning Löwe (SLF)

Projet esa

What is a microwave radiative transfer model ?

Inputs :

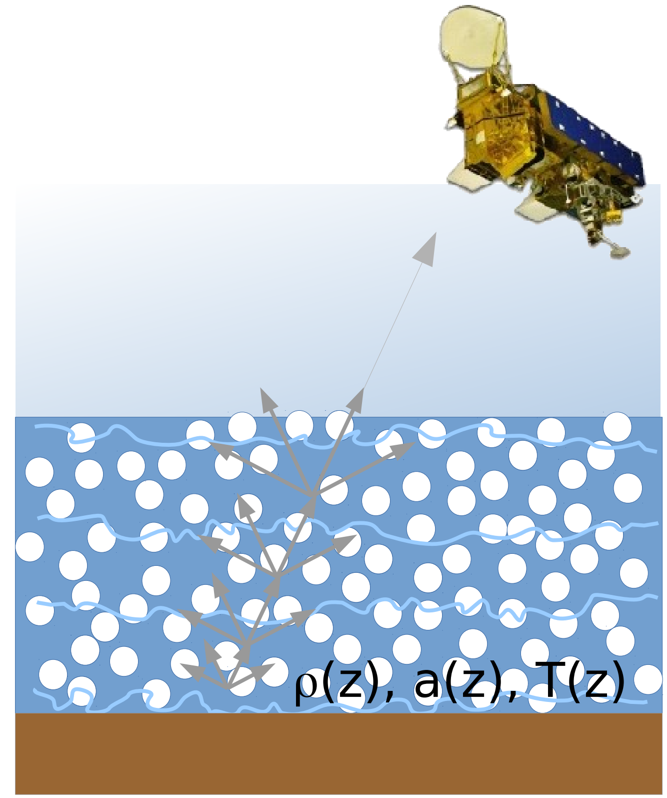
- Snowpack:
    + Homogeneous layers (density, temperature, microstructure)
    + internal interfaces
    + bottom interface (soil, ...)
    + top interface and atmosphere.

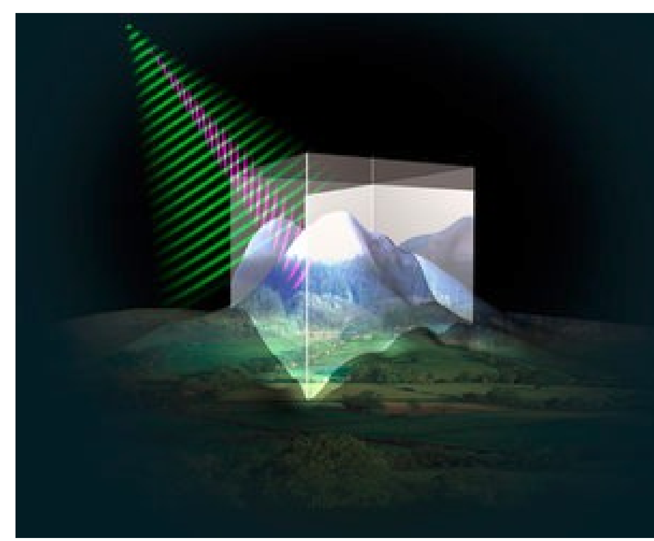
- Sensor configuration

Computations:

1) single scattering and absorption properties in each layer
2) propagation / multiple scattering between the layers

→ outgoing intensity / radiance/ brightness temperature





CoReH2O, ESA, Earth Explorer 8 Mission
(not selected)

The different microwave RT models for snow:

HUT	MEMLS	DMRT-QMS	DMRT-ML
Maximum extent (aka traditional grain size), Dmax	**Correlation length** Exponential correlation fct A(x)	**Sphere radius** (distribution), stickiness: a, τ	**Sphere radius** (distribution), stickiness: a, τ
Empirical Ks Semi-empirical Ka	IBA (Wahl=12) / W98 (Wahl<12)	DMRT	DMRT Short range Shih et al. 1997
↓ κs,κa, q	↓ κs,κa,P(Θ)	↓ κs,κa,P(Θ)	↓ κs,κa,P(Θ)
1-flux	6-flux	N-stream (spline)	N-stream (DISORT, Jin 1994
Fortran / Matlab	Matlab / Fortran	Matlab	Fortran/Python
FMI	C. Mätzler & co	L. Tsang & co	G. Picard & co

Why a new model:

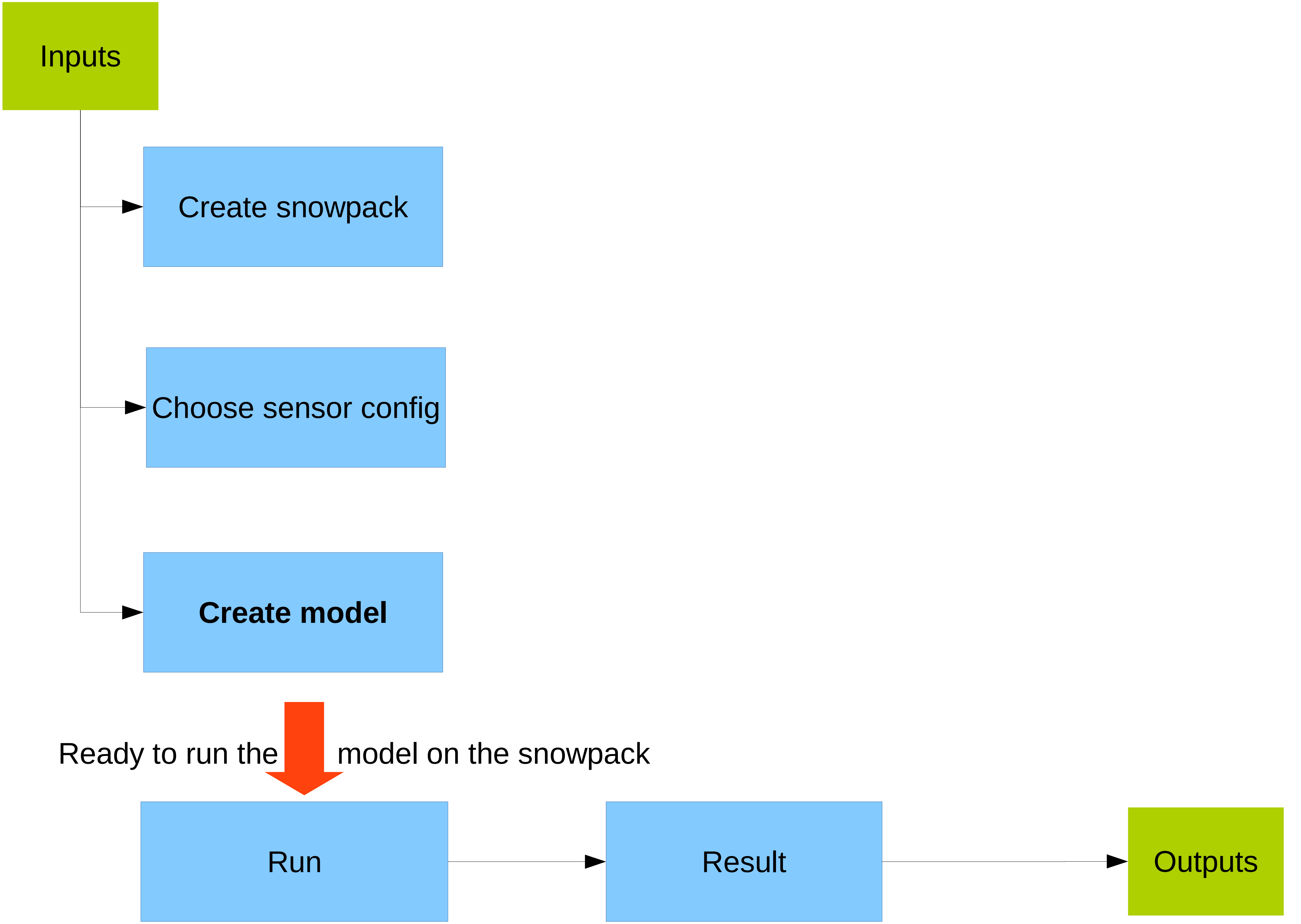
- need inter-comparisons at the level of processes, not of whole models.

- progresses in snow microstructure → need to inter-compare different microstructure representations (existing and new ones).

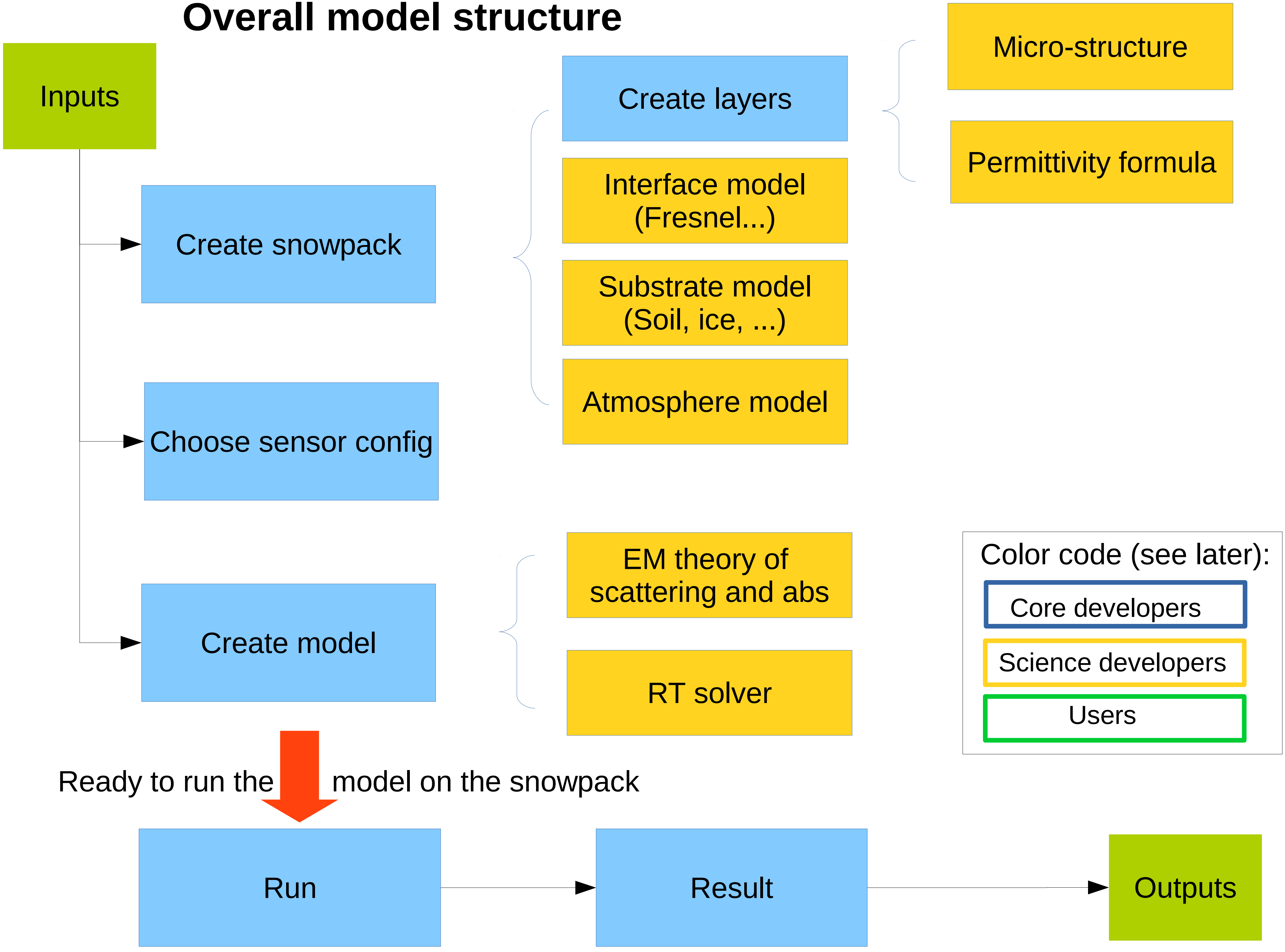
- need a unified passive/active model (this is not new).

- need a community model.

Overall model structure



Overall model structure



Example:

Use simple, intuitive naming of the functions, variables
Strict rules (uppercase, underscore, limited abbreviations)
A lot of python magic under the hood

Inputs

Create snowpack

Choose sensor config

Create model

Run

Outputs

```
from smrt import make_snowpack, make_model, sensor

# prepare inputs
thickness = [100]
corr_length = [5e-5]
temperature = [270]
density = [320]

# create the snowpack
snowpack = make_snowpack(thickness=thickness,
                         microstructure_model="exponential",
                         density=density,
                         temperature=temperature,
                         corr_length=corr_length)

# create the sensor
radiometer = sensor.amsre('37V')

# create the model
m = make_model("iba", "dort")

# run the model
result = m.run(radiometer, snowpack)

# outputs
print(result.TbV(), result.TbH())
```

Modularization & Extensibility:

Each block is as autonomous as possible (encapsulation):

Why:

- Known to be a good programming practice → less bugs.
- Each block can have multiple implementations.
- Easier to extend (limited knowledge about the whole code is needed).

```
# create the model
m1 = make_model("iba", "dort")

# create another model
m2 = make_model("dmt_shortrange", "dort")
```

<u>Each block = one directory:</u>

smrt/emmodel/
iba.py and **dmrt_shortrange.py** are files in smrt/emmodel/.

Adding a new theory to compute scattering is as simple as adding a new file in this directory!

smrt/microstructure/

Legacy:

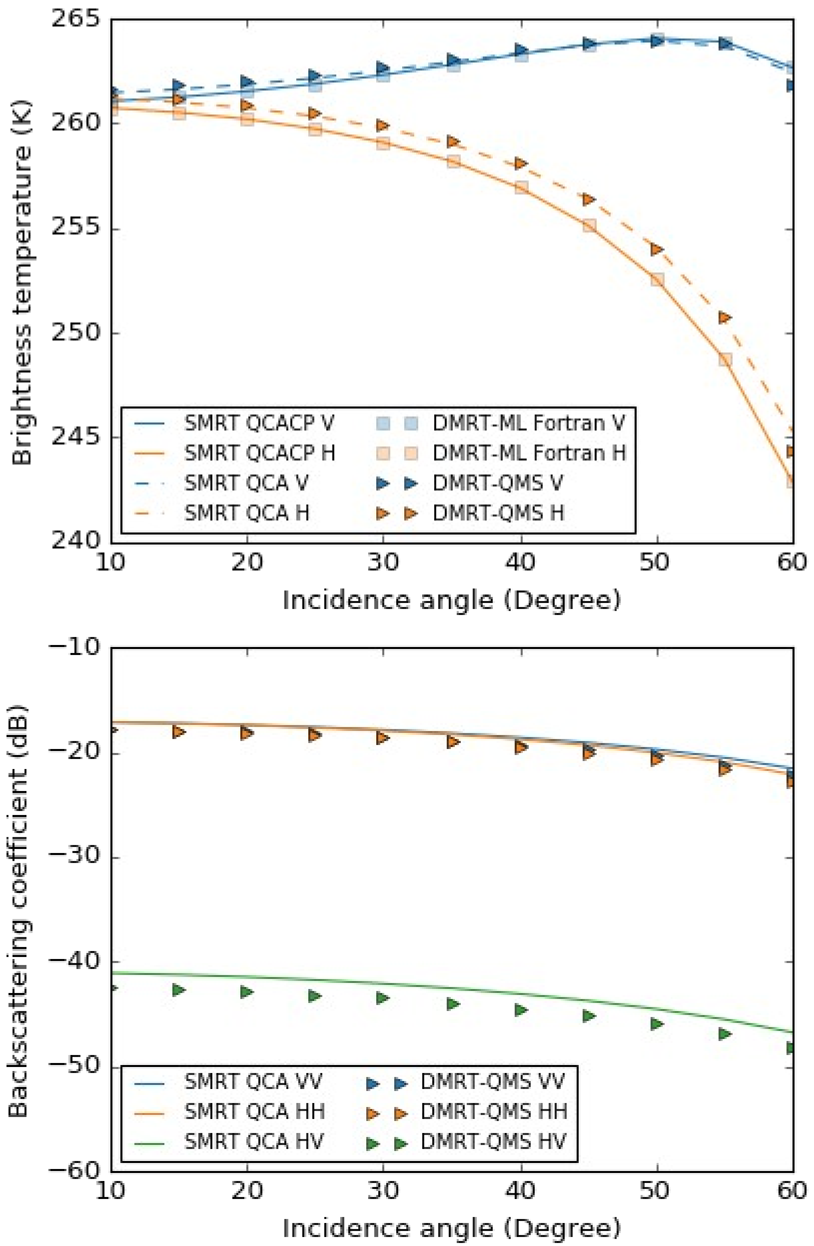
SMRT intents to be an unification of existing models, a repository of community knowledge.
→ thin wrappers for DMRT-QMS, MEMLS and HUT have been written to call these legacy models in their original form.
→ convenient comparison and cross-check opportunities

```
# general import for smrt
from smrt import make_snowpack, make_model, sensor

# import for memls
from smrt.utils import memls_legacy

# prepare snowpack
pc = 0.2e-3
snowpack = make_snowpack(thickness=[10], microstructure_model="exponential",
                         density=[300], temperature=[265], corr_length=pc)

# create the sensor
theta = range(10, 80, 5)
radiometer = sensor.passive(37e9, theta)

# create the EM Model
m = make_model("iba", "dort")

# run the model
sresult = m.run(radiometer, snowpack)

# run MEMLS matlab code
mresult = memls_legacy.run(radiometer, snowpack)


# outputs
plt.plot(theta, sresult.TbV(), 'r-', label='SMRT V')
plt.plot(theta, sresult.TbH(), 'r--', label='SMRT H')
```

Import memls_legacy wrapper

Call MEMLS matlab code from Python using radiometer and snowpack

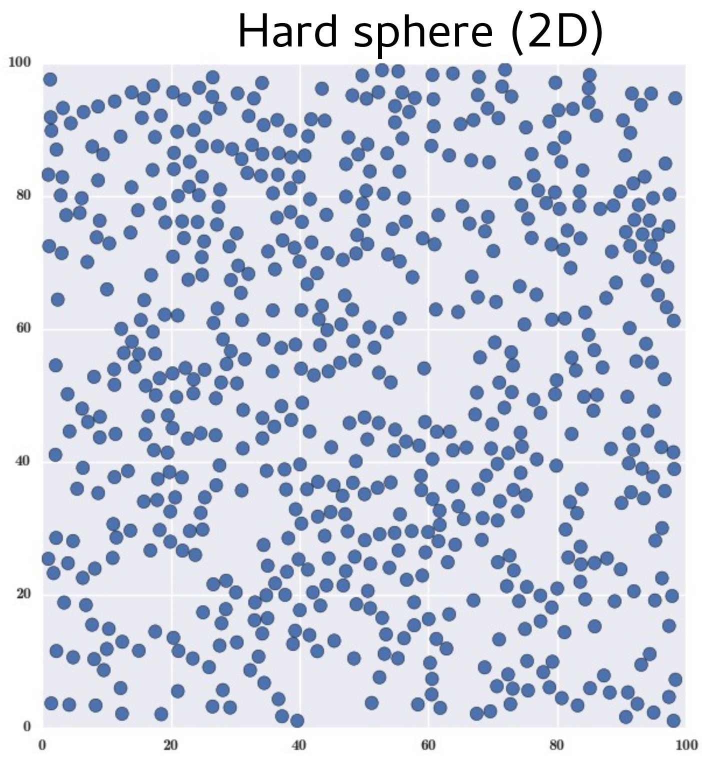
Example of intercomparison:

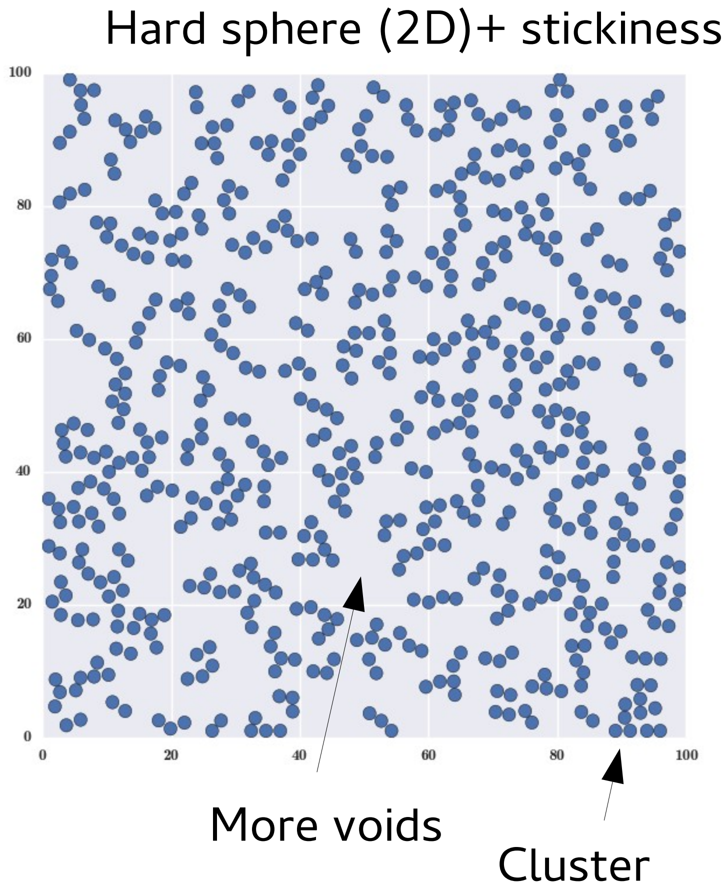


<u>Comparisons of 4 models:</u>

- DMRT-ML (original) - SMRT DMRT QCA-CP
- DMRT-QMS (original) - SMRT DMRT QCA

Exploration of microstructure representation:



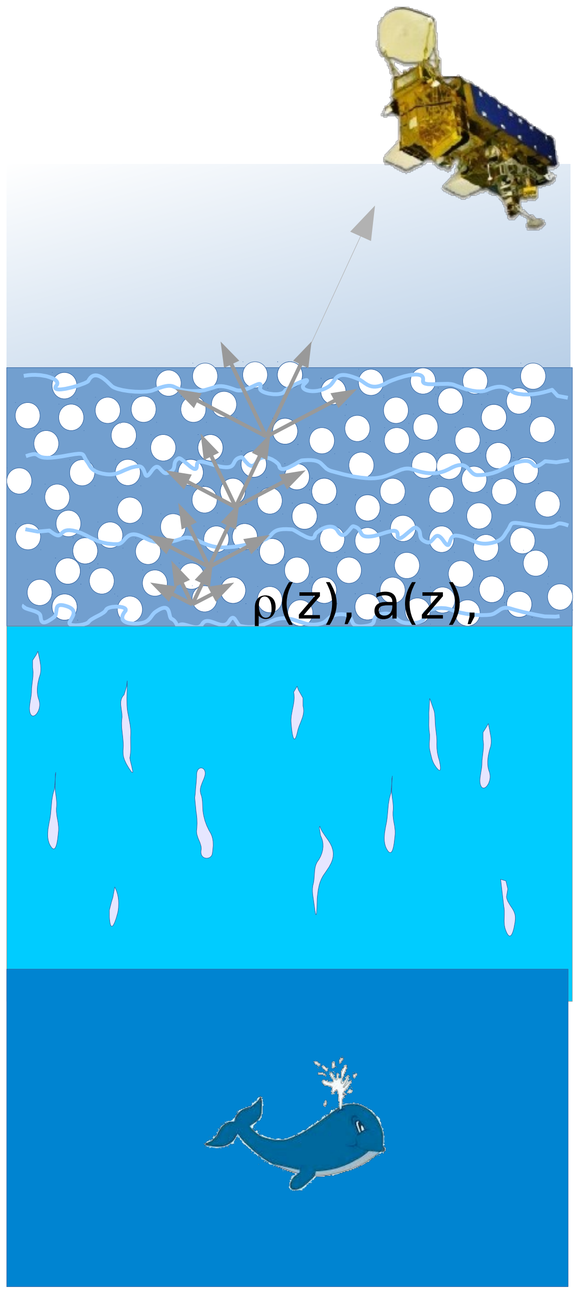


Both have the same SSA, but the medium on the right is **more heterogeneous** than on the left ("higher coarseness").

Exploration of microstructure representation:

- Independent random spheres → useless for snow. Good for clouds
- Sticky Hard Sphere → used in the DMRT models
- Exponential autocorrelation function → used in the MEMLS model
- Teubner Strey autocorrelation function
- Gaussian random field autocorrelation function
- Measured microstructure using tomography (work in progress)
- Inferred autocorrelation function from SnowMicroPen (??)

Future works: snow & sea-ice



esa

MOSAiC
The International Arctic
Drift Expedition

SMRT training

February 9-11, Col du Lautaret, France

(before the Snow Science Winter School)

Ghislain Picard, Melody Sandells, Henning Löwe