Snow Microwave Radiative Transfer (SMRT): A new model framework to simulate snow-microwave interactions for active and passive remote sensing applications

G. Picard[1], M. Sandells[2], **H. Löwe**[3], C. Mätzler[4], A. Kontu[5], M. Dumont[6], W. Maslanka[7], S. Morin[6], R. Essery[8], J. Lemmetyinen[5], A. Wiesmann[4], N. Floury[9], M. Kern[9]

[1]LGGE (F), [2]CORES (UK), [3]WSL-SLF (CH), [4]Gamma-RS (CH), [5]FMI (FI), [6]MeteoF-CEN (F), [7]U-Reading (UK), [8]U-Edinburgh (UK), [9]ESA-Estec (NL)

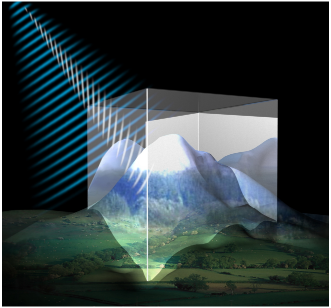
AGU Fall Meeting, San Francisco, 15. Dec 2016

Laboratoire de Glaciologie et Géophysique de l'Environnement

ILMATIETEEN LAITOS
METEOROLOGISKA INSTITUTET
FINNISH METEOROLOGICAL INSTITUTE

GAMMA REMOTE SENSING

Introduction: Context



(pic: CoRH2O, ESA)

Still on the wishlist:

- Retrieve global snow mass from (microwave) remote sensing

Still challenging:

- Snow-microwave forward modeling

Available models:

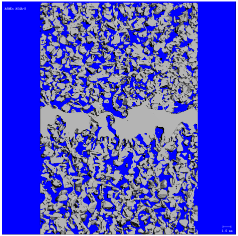
- HUT (passive, Pulliainen, FMI)
- MEMLS (active/passive, Mätzler, UBern)
- DMRT-QMS (active/passive, Tsang, UMich)
- DMRT-ML (passive, Picard, LGGE)

Plenty of choices, why another model?

Why another model:

Plenty of model comparison exercises:

- (Tedesco et al. 2006), (Tian et al. 2010), (Brucker et al. 2011), (Roy et al. 2013), (Löwe et al. 2015), (Pan et al. 2015), (Tan et al. 2016), ...



Main problem: Concurrent differences

- Electromagnetic theories
- RT solvers
- Microstructure models (aka “grain size”)

ESA-ITT (2014):

Microstructural origin of electromagnetic signatures
in microwave remote sensing of snow

Goal:

- Develop a new model (unifying, extensible) to address some of the difficulties

SMRT Model – Outline

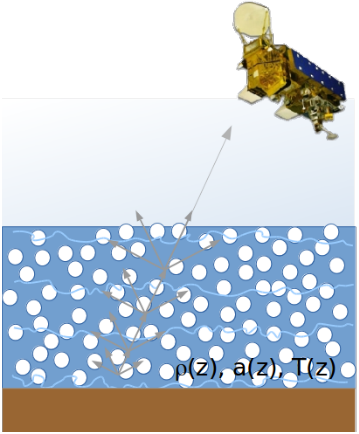
SMRT Model – Outline

SMRT: Overview

Main physics:

- Active/passive
- Multi-layer
- Vector RTE
  (presently: isotropy)
- Solver: Discrete ordinate
- Phase function: IBA
- Layer microstructure:
  Correlation functions
- Interfaces
  (presently: Fresnel)
- Options for substrate,
  atmosphere, permittivity

$$\mu \frac{\partial \mathbf{I}^{(l)}(\mu,\phi,z)}{\partial z} = -\boldsymbol{\kappa}_{\boldsymbol{e}}^{(l)}(\mu)\cdot \mathbf{I}^{(l)}(\mu,\phi,z)$$
$$+\frac{1}{4\pi}\iint_{4\pi} \mathbf{P}^{(l)}(\mu,\mu',\phi-\phi')\cdot \mathbf{I}^{(l)}(\mu',\phi',z)\, d\Omega' + \boldsymbol{\kappa}_{\boldsymbol{a}}^{(l)}(\mu)\, T^{(l)}$$



SMRT: Microstructure representation

Layer microstructure:

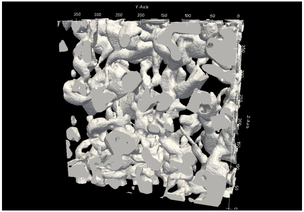
- represented by a two-point correlation *function*

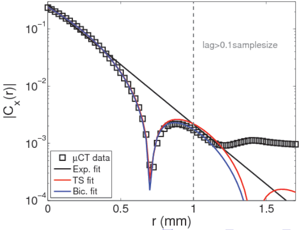
Implemented models: (and reasons for them)

- Exponential model (MEMLS)
- Sticky hard spheres (DMRT-ML, DMRT-QMS)
- Independent sphere (classic)
- Teubner–Strey (empirical evidence)
- (Level-cut) Gaussian random fields (full-field methods)

Why different choices?

- To be flexible with non-obvious correlation functions:



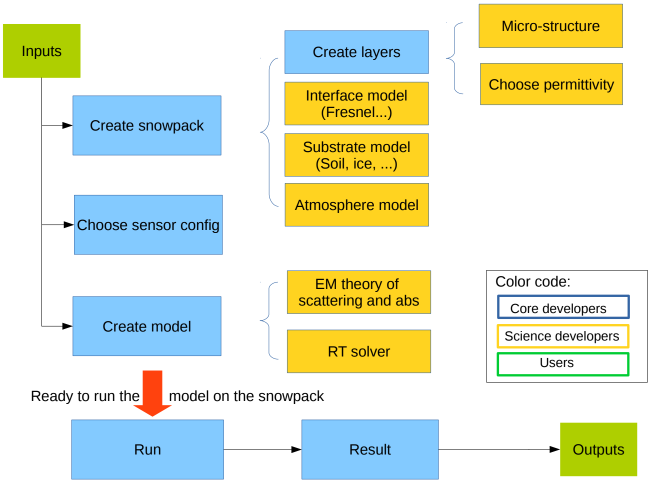


SMRT Model – Outline

SMRT: Overview

Key goals:

- modular, extensible, easy to use



SMRT: Implementation

Why python?

- open, object oriented (to realize “easy to use”)

Example of a model run:

Inputs

Create snowpack

Choose sensor config

Create model

Run

Outputs

```python
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())
```

SMRT: Legacy support

To facilitate model inter-comparison:

- Shallow wrappers for MEMLS, HUT, DMRT-QMS (no code)

```
# 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')
```

SMRT: Documentation

Online: Docstring/Sphinx:

Table Of Contents

Quick search

Go

smrt.substrate package

This directory contains different options to represent the substrate, that is the lower boundary conditions of the radiation transfer equation. This is usually the soil or ice or water but can be an aluminium plate or an absorber.

To create a substrate, use/implement an helper function such as **make_soil()**. This function is able to automatically load a specific soil model .

Examples:

```
from smrt import make_soil
soil = make_soil("soil_wegmuller", "dobson85", moisture=0.2, sa
```

It is recommand to first read the documentation of **make_soil()** and then explore the different types of soil models.

For developers:

To develop a new substrate formulation, you must add a file in the smrt/substrate directory. The name of the file is used by make_soil to build the substrate object.

Submodules

- smrt.substrate.flat module
- smrt.substrate.reflector module
- smrt.substrate.soil_qnh module
- smrt.substrate.soil_wegmuller module

Tutorials: Notebooks

Table of Contents

How to run a sensitivity study on snowpack properties ?

SMRT is able to run the model for a sensor that have a list of frequencies, angles or polarization automatically. However, for snowpack properties setting for instance the corr_length in a layer to a list (or array) does not work (yet). It is necessary to do a little more programming, but it is not so complex.

First import the various module we always need:

In [1]:
```
import numpy as np
import smrt
import matplotlib.pyplot as plt
%matplotlib inline
```

Build a list of snowpack

We want to test the sensitivity of TB's to the corr_length. We first build a list of snowpack with different corr_length

In [2]:
```
# prepare the snowpack
density = 300.0
corr_length = np.arange(0, 0.5, 0.01) *1e-3  # from 0 to 0.5mm

# the naive way: without list comprehension

snowpack = list()
for x in corr_length:
    sp = smrt.make_snowpack([10], "exponential", density=[density], temperature=[265], corr_length=x)
    snowpack.append(sp)
```

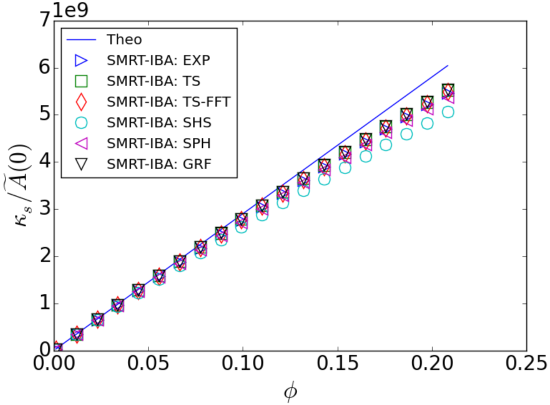
A better way is to use list comprehension. To achieve the same as these latter 4 last lines:

SMRT Model – Outline

Against closed-form asymptotics...

Low-density, low-frequency limit of the scattering coefficient:

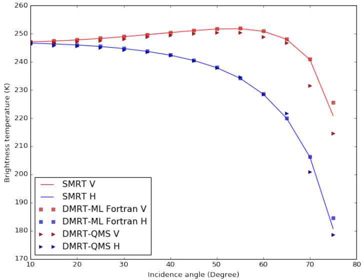
$$\kappa_s^{\mathrm{IBA}} = \phi \left[\frac{2}{3} k_0^4 \frac{1}{4\pi} (\epsilon_2 - \epsilon_1)^2 \left| \frac{3\epsilon_1}{2\epsilon_1 + \epsilon_2} \right|^2 \right] \widetilde{A}(0)$$



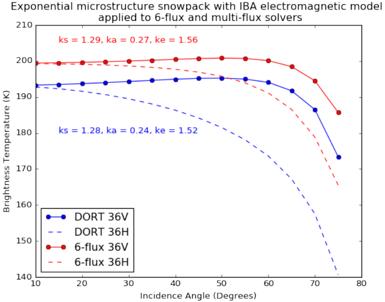
Against existing models...

Using the “legacy” wrapper:

vs DMRT-QMS/DMRT-ML:



vs MEMLS:

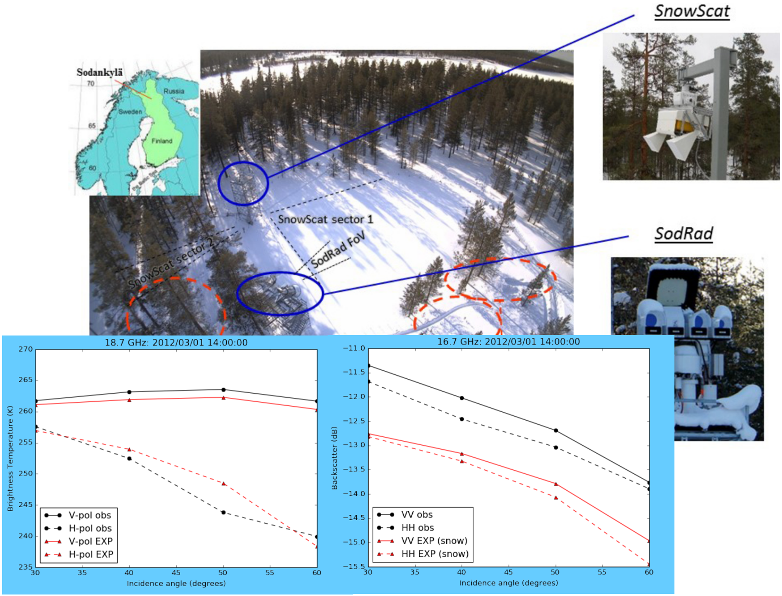


To be certain:

- Implementation of MEMLS’ 6-Flux solver in SMRT (Todo)

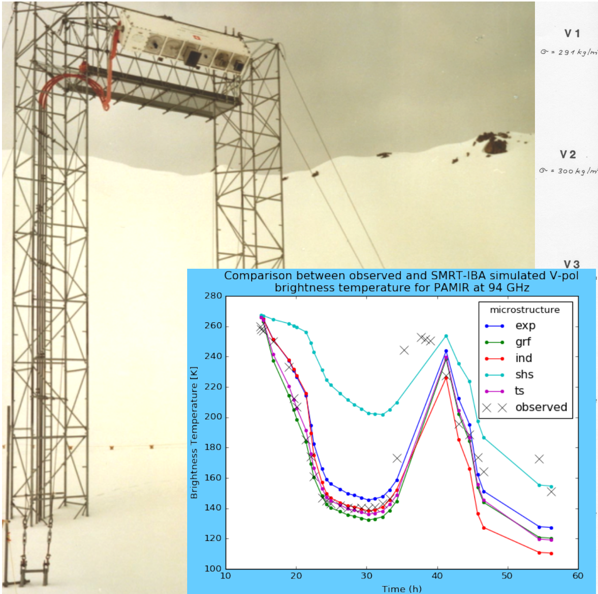
Against NoSREx...

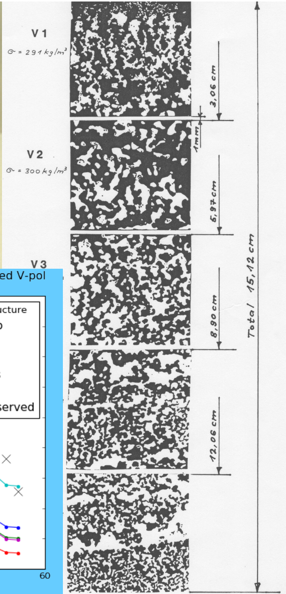
Nordic **S**now **R**adar **Ex**periment (Lemmetyinen et al. 2016):



Against PAMIR...

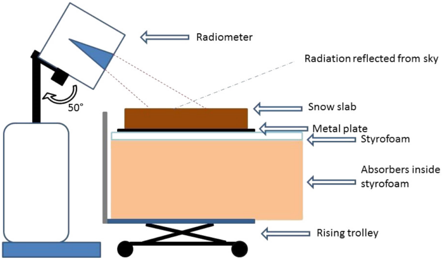
Passive and **A**ctive **M**icrowave and **I**nfrared **R**adiometer (Mätzler, 1987):

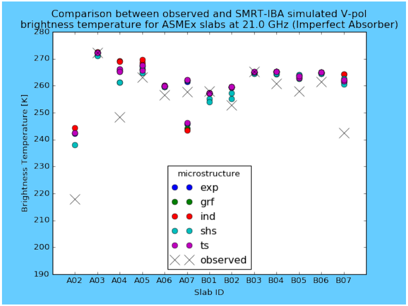


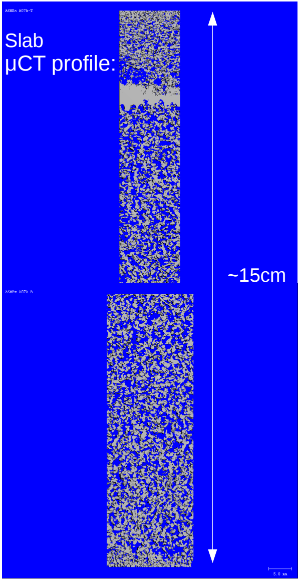


Against ASMEx...

Arctic **S**now **M**icrostructure **Ex**periment (Maslanka et al. 2016):







Conclusions

Snow microwave radiative transfer (SMRT) model:

- Modular, extensible framework
- Getting into the data has started only now
- Among other things not shown here: SMRT+CROCUS (SSA)

Next steps:

- Code release: Spring 2017
- Website: `http://www.smrt-model.science/`
- Model paper: (Picard et al., in prep)
- Validation paper: (Sandells et al., in prep)
- SMRT user workshop: Fall 2017 (COST)

Support will be given to maximize community adoption!

Acknowledgements: