# Modelling the hydrological cycle in snow-dominated catchments

Guillaume THIREL
*Irstea*

Using R In Hydrology

EGU 2019

irstea.fr/en/thirel

G_Thirel

# Reminder of last year

Last year, I introduced rainfall-runoff modelling.

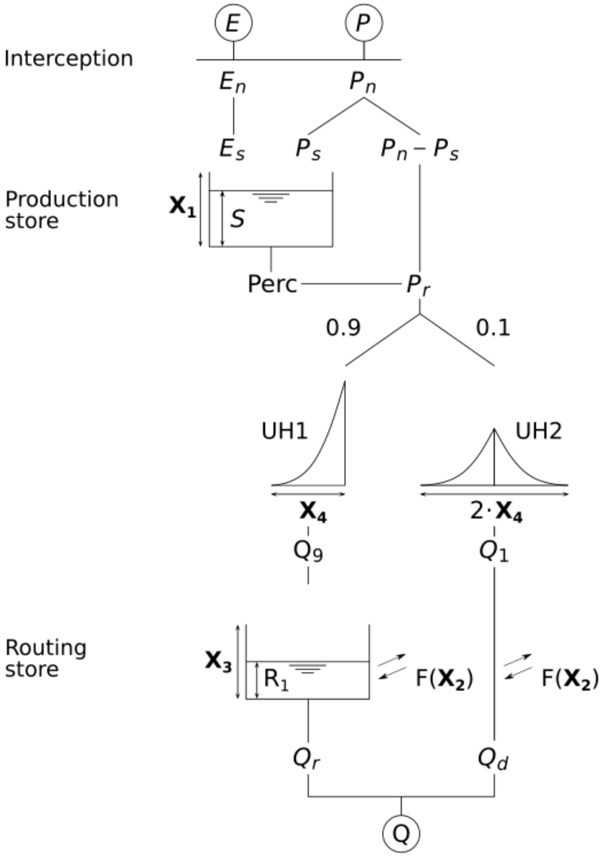
Updated list of hydrological modelling packages in R

- **airGR**
- **airGRteaching**
- **dynatopmodel**
- **Ecohydmod**
- **fuse**
- **hydromad** (not on CRAN)
- **sacsmaR** (not on CRAN)
- **topmodel**
- **TUWmodel**
- **WALRUS** (not on CRAN)
- etc.

# Example of a rainfall-runoff model

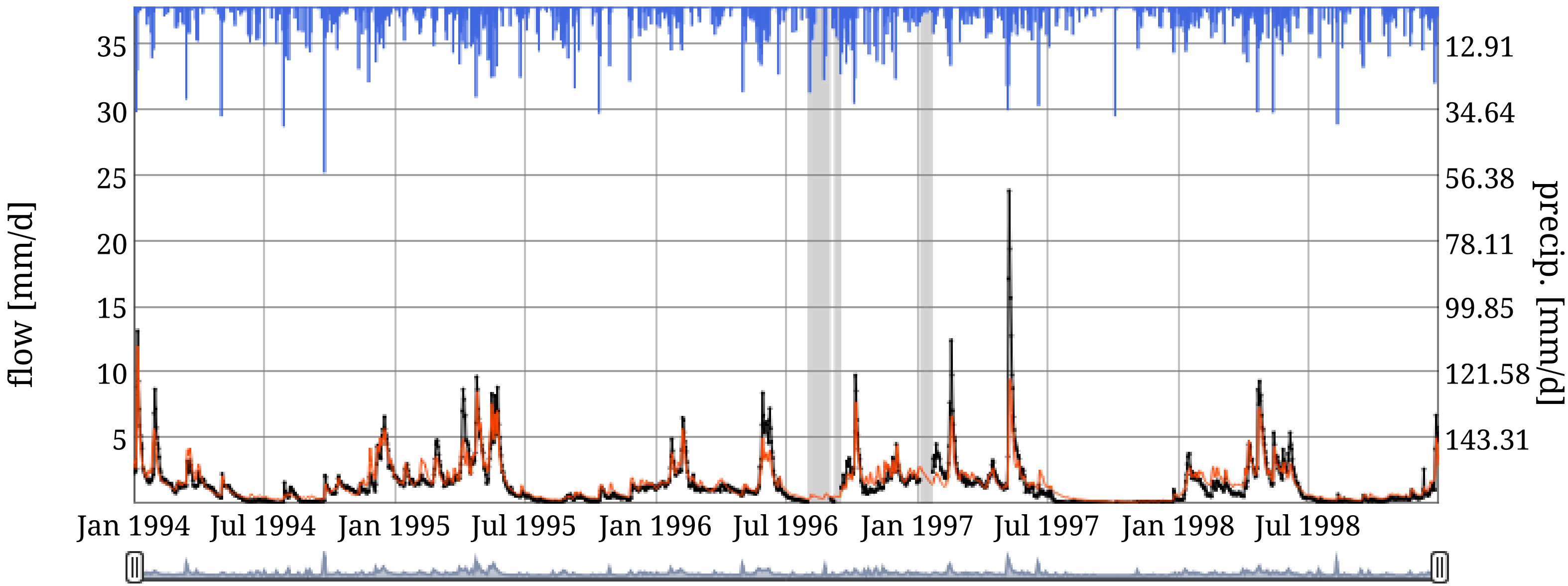
Strictly speaking, rainfall-runoff models transform rainfall inputs into discharge (runoff).

GR4J ->



# Example of a rainfall-runoff model

Example of running GR4J from **airGRteaching** on an Australian basin:



# Let it snow...

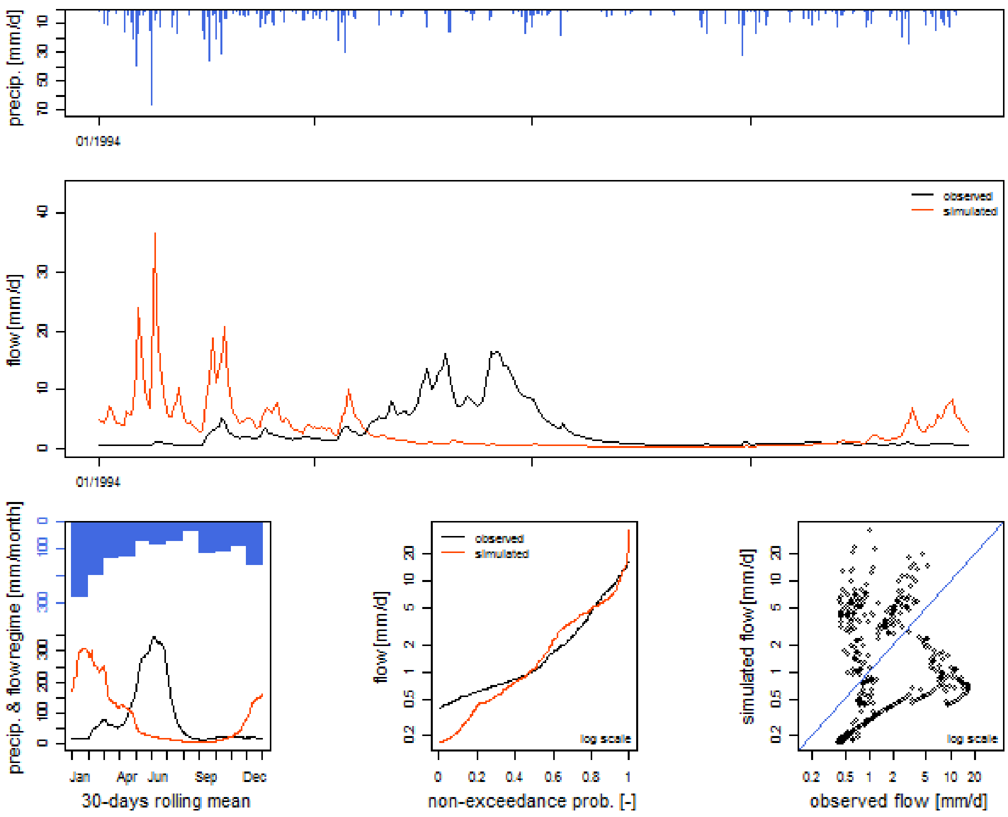
Snowfall happens when temperatures are close to 0 °C or lower.

Snow is stored in the **snowpack**, to be released days to months later when temperature rise above 0 °C for a substancial period of time.

This creates a **shift between precipitation** (rainfall + snowfall) **and discharge**.

# Let it snow...

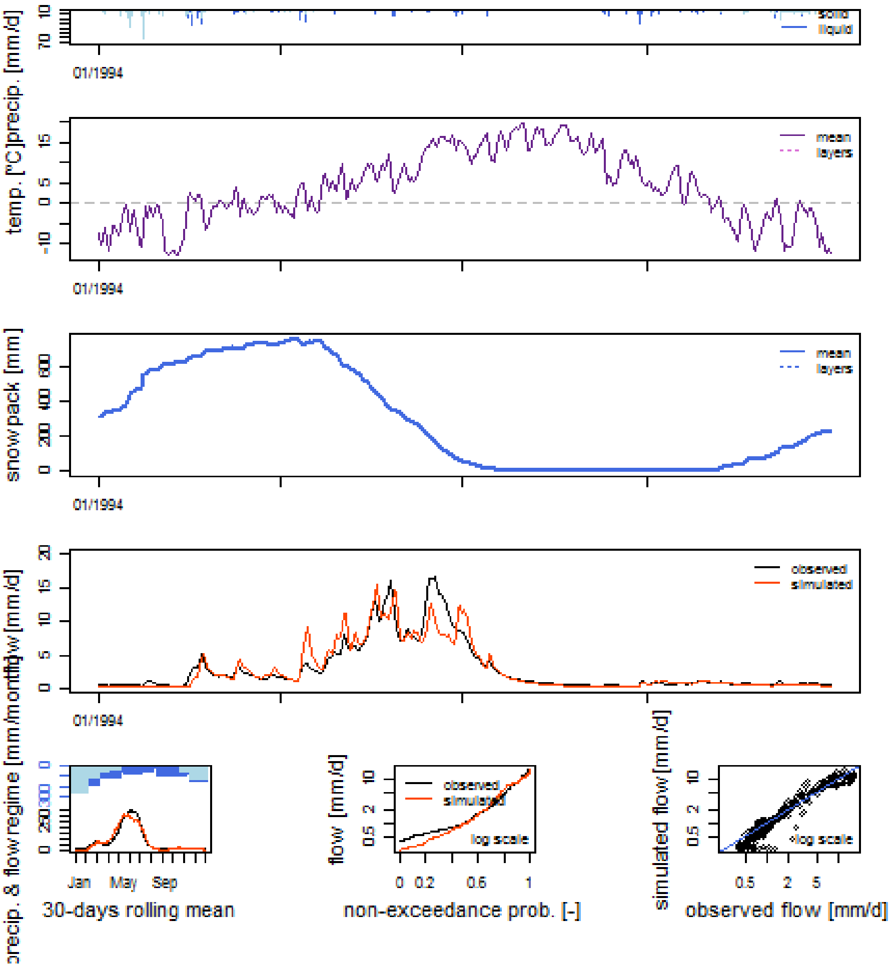
Example of simulating a snow-dominated basin with only a rainfall-runoff model:



# Let it snow...

Modelling the snow accumulation and melt improves a lot the simulations.

Same simulation with a snow model activated:



# Principles of snow models/modules

## Step 1: Accumulation

Snowpack = Snowpack + Snowfall

Depending on temperature, precipitation is either considered as rainfall or snowfall (or a mix of both).

Frequently we find the following:

- Snowfall if T < 0 °C and rainfall if T >= 0 °C

or

- Snowfall if T < -1 °C, rainfall if T > +3 °C, a linear mix of both in between

# Principles of snow models/modules

## Step 2: Melting

Snowpack = Snowpack - Snow melt

**Degree-day models**

- Very common for catchment hydrology
- Simplest option
- Require only P and T

Principle: the rate at which snow melts is governed by the temperature above a threshold (e.g. 0 °C) and a parameter (expressed in mm/°C/day).

**Energy-balance models**

- Quite common for catchment hydrology
- More complex
- Require several variables sometimes difficult to obtain, such as wind speed, radiation, air humidity…

# Snow models in R packages

Not all hydrological modelling R packages contain a snow component.

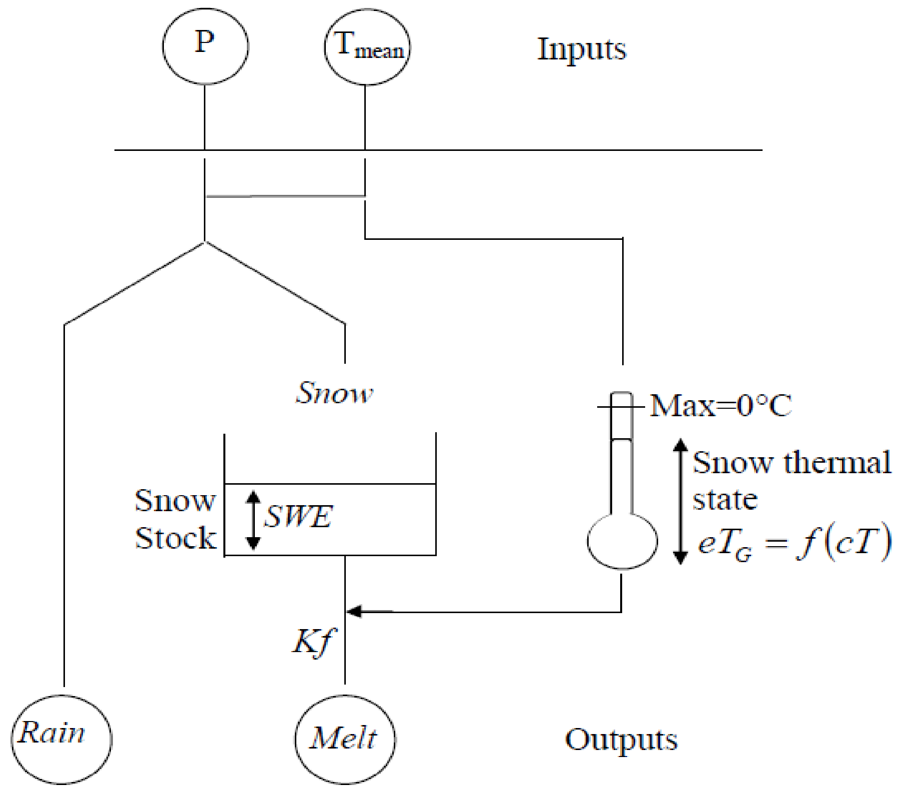
We could identify:

- CemaNeige in **airGR** and **airGRteaching**
- SNOW17 in **sacsmaR**
- a module in **SWATmodel**
- a module in **TUWmodel**
- a module in **WALRUS**
- etc.

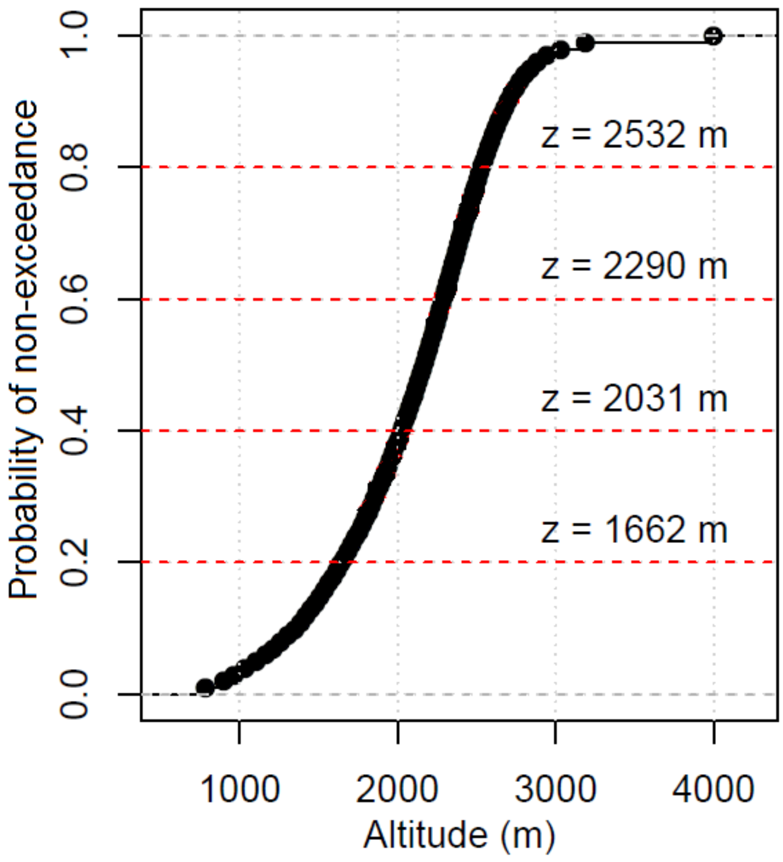
# The CemaNeige example in airGR

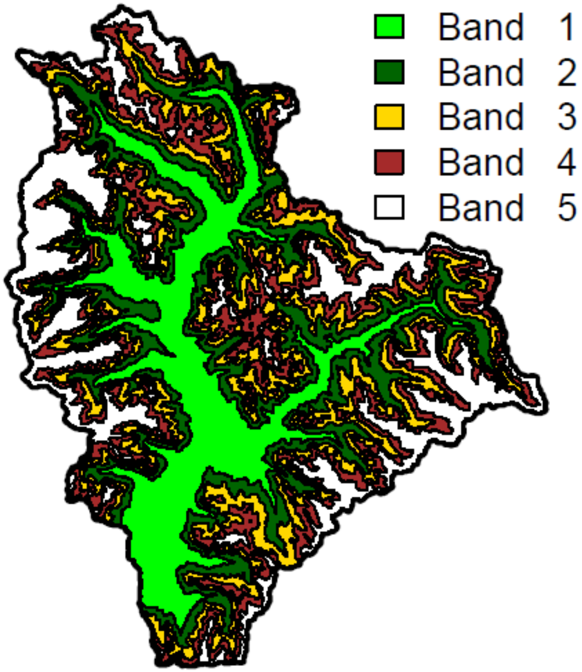
CemaNeige is a degree-day model.

It was conceived on the principle of having a model "as simple as possible but not simpler"...



# The importance of taking into account altitude gradients



# How to extrapolate lumped input in airGR

Initially, we have lumped input:

```
head(BasinObs, n = 2)
```

```
##         DatesR   P    T   E       Qmm
## 1 1999-01-01 0.2 -3.9 0.1 0.6422962
## 2 1999-01-02 4.0 -3.3 0.1 0.6418041
```

Let's extrapolate data (we often do that on 5 elevation bands):

```
## preparation of the InputsModel object
BasinObs <- BasinObs2
InputsModel <-
  CreateInputsModel(FUN_MOD = RunModel_CemaNeigeGR4J,
                    DatesR = BasinObs$DatesR,
                    Precip = BasinObs$P,
                    PotEvap = BasinObs$E,
                    TempMean = BasinObs$T,
                    ZInputs = median(BasinInfo$HypsoData),
                    HypsoData = BasinInfo$HypsoData,
                    NLayers = 5)
```

# How to extrapolate lumped input in airGR

What we get as inputs:

```
str(InputsModel)
```

```
## List of 6
##  $ DatesR        : POSIXlt[1:4230], format: "1999-01-01" "1999-01-02" ...
##  $ Precip        : num [1:4230] 0.2 4 1.2 0 0 0 0 0.7 3.1 8.7 ...
##  $ PotEvap       : num [1:4230] 0.1 0.1 0.1 0.3 0.4 0.5 0.4 0.2 0 0 ...
##  $ LayerPrecip   :List of 5
##   ..$ L1: num [1:4230] 0.145 2.906 0.872 0 0 ...
##   ..$ L2: num [1:4230] 0.179 3.573 1.072 0 0 ...
##   ..$ L3: num [1:4230] 0.202 4.049 1.215 0 0 ...
##   ..$ L4: num [1:4230] 0.223 4.462 1.339 0 0 ...
##   ..$ L5: num [1:4230] 0.251 5.011 1.503 0 0 ...
##  $ LayerTempMean:List of 5
##   ..$ L1: num [1:4230] -0.333 0.267 0.376 5.784 8.092 ...
##   ..$ L2: num [1:4230] -2.52 -1.92 -1.82 3.59 5.89 ...
##   ..$ L3: num [1:4230] -3.84 -3.24 -3.14 2.26 4.56 ...
##   ..$ L4: num [1:4230] -4.87 -4.27 -4.17 1.22 3.52 ...
##   ..$ L5: num [1:4230] -6.1004 -5.5004 -5.4055 -0.0105 2.2844 ...
##  $ ZLayers       : num [1:5] 1348 1852 2157 2394 2677
##  - attr(*, "class")= chr [1:4] "InputsModel" "daily" "GR" "CemaNeige"
```

# How to extrapolate lumped input in airGR

Simulations:

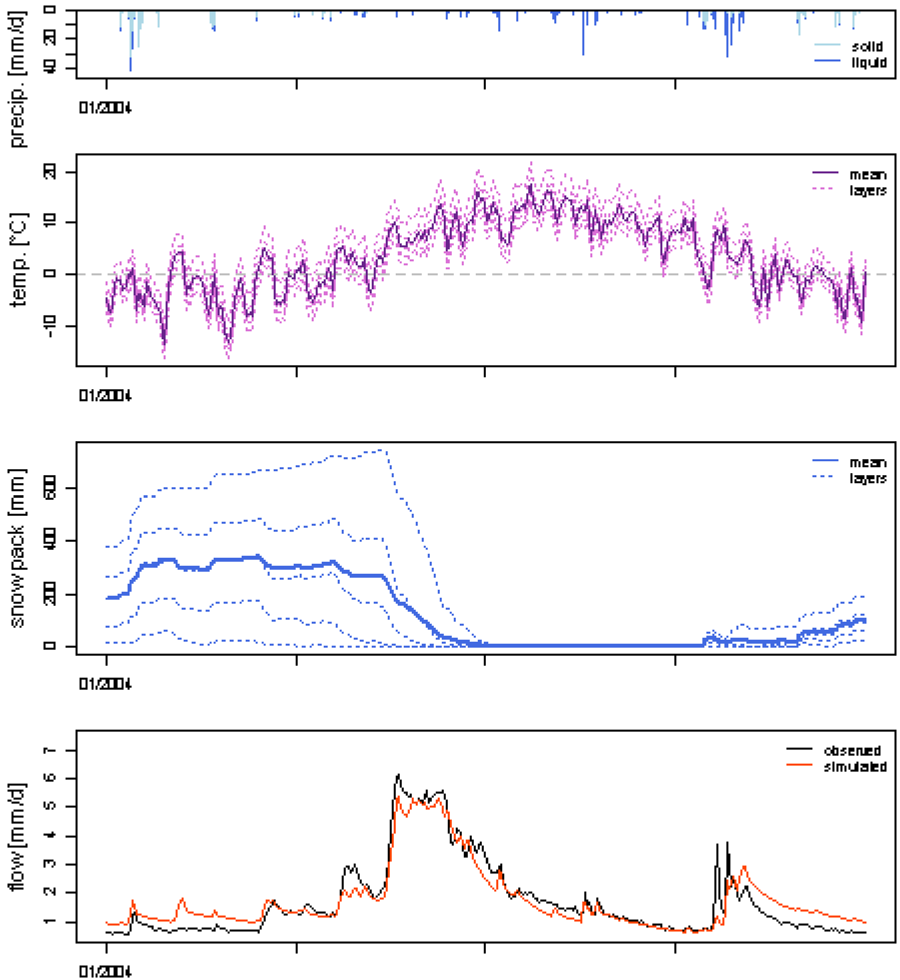
```
## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_CemaNeigeGR4J,
                               InputsModel = InputsModel,
                               IndPeriod_Run = Ind)

## Parameters setting
Param <- c(X1 = 310, X2 = 3.6, X3 = 340, X4 = 1.,
           CNX1 = 0.7, CNX2 = 5)

## simulation
OutputsModel <- RunModel_CemaNeigeGR4J(InputsModel = InputsModel,
                                       RunOptions = RunOptions,
                                       Param = Param)
```

```
# plot of selected variables
plot(OutputsModel, Qobs = BasinObs$Qmm[Ind],
     which = c("Precip", "Temp", "SnowPack", "Flows"))
```

# How to extrapolate lumped input in airGR



# A step forward: using (satellite) snow data

Most of the time, snow modules are calibrated together with rainfall-runoff models with discharge data only.

Using snow data in addition to discharge can better constrain the model and therefore improve the simulations.

## Packages to retrieve MODIS sensor SCA data

- **MODIS**: Acquisition and Processing of MODIS Products
- **MODISSnow**: Provides a Function to Download MODIS Snow Cover

## Other snow data sources

- **snotelr**: Calculate and Visualize 'SNOTEL' Snow Data and Seasonality

# Calibrating CemaNeige with MODIS SCA

Initially, we have lumped input:

```
head(BasinObs, n = 2)
```

```
##        DatesR P     T E   Qls   Qmm  SCA1  SCA2  SCA3  SCA4  SCA5
## 1 2002-01-01 0 -10.0 0 14612 0.553 0.150 0.426 0.686    NA    NA
## 2 2002-01-02 0  -4.8 0 14354 0.543 0.156 0.392 0.654 0.844 0.885
```

Here we define the optimisation criterion:

```
## criterion: 75 % KGE'(Q) + 5 % KGE'(SCA) on each of the 5 layers
inCrit <-
  CreateInputsCrit(FUN_CRIT = rep("ErrorCrit_KGE2", 6),
                   InputsModel = InputsModel,
                   RunOptions = RunOptions,
                   Obs = BasinObs[Ind, c("Qmm", "SCA1", "SCA2",
                                         "SCA3", "SCA4", "SCA5")],
                   VarObs = list("Q", "SCA", "SCA", "SCA",
                                 "SCA", "SCA"),
                   Weights = list(0.75, 0.05, 0.05,
                                  0.05, 0.05, 0.05))
```

# Calibrating CemaNeige with MODIS SCA

Let's see how GR4J + CemaNeige calibrates with the in-built **airGR** optimisation tool:

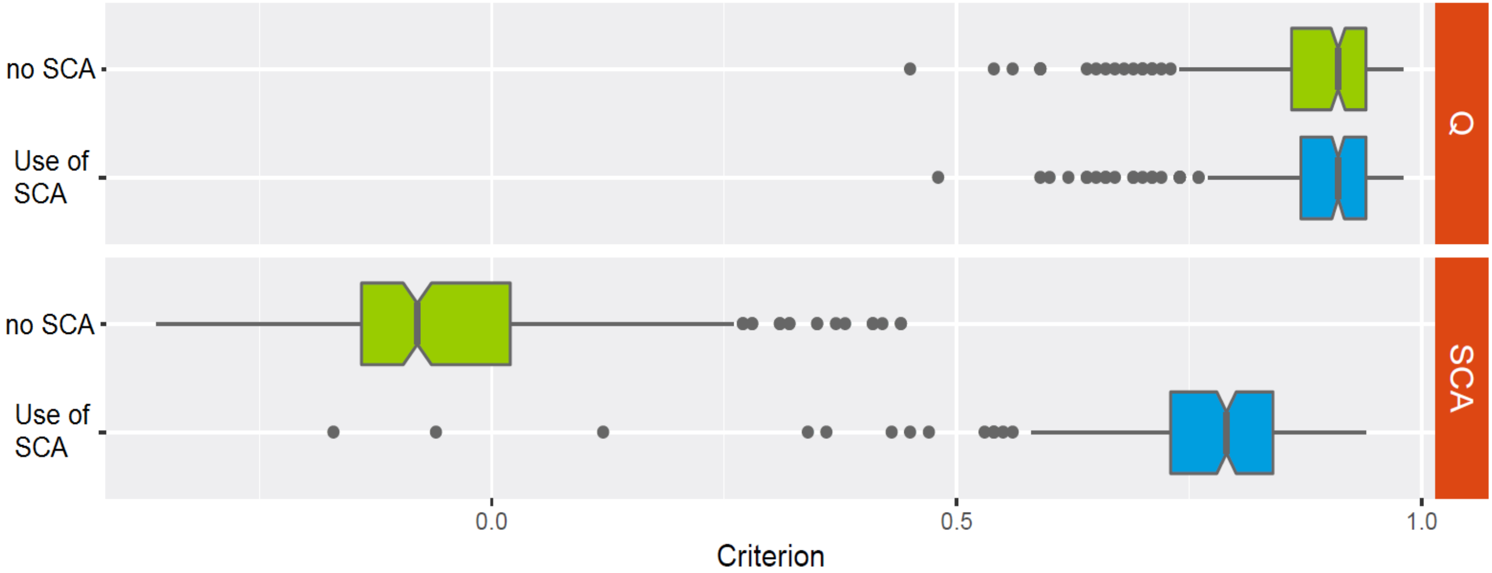
```
OutputsCalib <- Calibration_Michel(InputsModel = InputsModel,
                                   RunOptions = RunOptions,
                                   InputsCrit = inCrit,
                                   CalibOptions = CalibOptions,
                                   FUN_MOD = RunModel_CemaNeigeGR4J)
```

```
## Grid-Screening in progress (0% 20% 40% 60% 80% 100%)
##      Screening completed (6561 runs)
##          Param =  432.681 ,   -0.020 ,   83.096 ,    1.417 ,    0.705 ,
##          Crit. Composite    = 0.7506
## Steepest-descent local search in progress
##      Calibration completed (95 iterations, 8050 runs)
##          Param =  361.405 ,    2.575 ,  254.680 ,    0.997 ,    0.840 ,
##          Crit. Composite    = 0.8998
##
##     Formula: sum(0.75 * KGE'[Q], 0.05 * KGE'[SCA], 0.05 * KGE'[SCA],
##                  0.05 * KGE'[SCA], 0.05 * KGE'[SCA], 0.05 * KGE'[SCA])
```

# Calibrating CemaNeige with MODIS SCA

Validation over 277 basins: SCA is much better and Q is similar.

We could show a better transferability of the model.



# Final words

We saw how to model snow-dominated basins in R with **airGR**.

Several snow modules are available in R.

Several snow datasets can be retrieved with dedicated packages.

Contact: airGR@irstea.fr

Meet Olivier Delaigue at the **airGR** poster
**A.39 from 8:30 am on Friday**