airGRiwrn: an extension of the airGR R-package for handling Integrated Water Resources Management modeling

David Dorchies, Olivier Delaigue, Guillaume Thirel

To cite this version:

David Dorchies, Olivier Delaigue, Guillaume Thirel. airGRiwrn: an extension of the airGR R-package for handling Integrated Water Resources Management modeling. EGU General Assembly 2021 - vEGU21: Gather Online, Apr 2021, Vienna, Austria. pp.EGU21-2190, 10.5194/egusphere-egu21-2190. hal-03330881

HAL Id: hal-03330881
https://hal.inrae.fr/hal-03330881
Submitted on 20 Apr 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License
airGRiwrn R package
an extension of the airGR R-package for handling Integrated Water Resources Management modeling

David Dorchies\textsuperscript{1}, Olivier Delaigue\textsuperscript{2}, and Guillaume Thirel\textsuperscript{2}

1. G-EAU, Univ Montpellier, AgroParisTech, CIRAD, IRD, INRAE, Institut Agro, Montpellier, France
2. Université Paris-Saclay, INRAE, HYCAR research unit, Hydrology Research Group, Antony, France
EGU General Assembly 2021
The airGR packages constellation

**airGR** is an R-package for running GR rainfall-runoff models

The latest version has semi-distributed hydrological modelling capabilities that are exploited by the **airGRiwrn** package.

See [EGU21-1371](#) in the same session
Handle large network of airGR semi-distributed hydrological models with a minimum of complexity

Easily integrate withdrawal and release flows in the network

Calibration and simulation with both influenced and naturalised flows

Run user control algorithms for simulating flows with automatic regulation
Semi-distributed model in airGR

In airGR, GR hydrological model simulations are routed from upstream basins to downstream basins thanks to a lag model.
How airGRiwrn works

It extends airGR package functions for handling a network of semi-distributed GR models instead of a single global GR model.

```r
library(airGR)
library(airGRiwrn)
```

```r
## Attaching package: 'airGRiwrn'

## The following objects are masked from 'package:airGR':
## Calibration, CreateCalibOptions, CreateInputsCrit,
## CreateInputsModel, CreateRunOptions, RunModel
```
Description of the network

All required information for running a semi-distributed model network are set in a formatted `data.frame` called a `GRiwm` object which can be displayed as a scheme of the network.

Diagram

<table>
<thead>
<tr>
<th>id</th>
<th>down</th>
<th>length</th>
<th>model</th>
<th>area</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>STDIZ_04</td>
<td>85.570</td>
<td>RunModel_GR4J</td>
<td>2347.53</td>
</tr>
<tr>
<td>19</td>
<td>LOUVE_19</td>
<td>86.165</td>
<td>RunModel_GR4J</td>
<td>461.74</td>
</tr>
<tr>
<td>25</td>
<td>VITRY_25</td>
<td>38.047</td>
<td>RunModel_GR4J</td>
<td>2109.14</td>
</tr>
<tr>
<td>21</td>
<td>CHALO_21</td>
<td>NA</td>
<td>RunModel_GR4J</td>
<td>6291.55</td>
</tr>
</tbody>
</table>
Hydroclimatic input data handling

Inputs are represented by matrices with one named column by sub-basin instead of vectors as in `airGR`

```r
head(P, 10)
```

<table>
<thead>
<tr>
<th></th>
<th>STDIZ_04</th>
<th>LOUVE_19</th>
<th>VITRY_25</th>
<th>CHALO_21</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>0.7</td>
<td>0.4</td>
<td>0.9</td>
<td>0.6</td>
</tr>
<tr>
<td>[2,]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[3,]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[4,]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,]</td>
<td>0.0</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[6,]</td>
<td>6.1</td>
<td>4.7</td>
<td>1.6</td>
<td>3.8</td>
</tr>
<tr>
<td>[7,]</td>
<td>17.1</td>
<td>23.6</td>
<td>17.0</td>
<td>16.4</td>
</tr>
<tr>
<td>[8,]</td>
<td>2.2</td>
<td>3.1</td>
<td>6.1</td>
<td>3.2</td>
</tr>
<tr>
<td>[9,]</td>
<td>10.8</td>
<td>10.8</td>
<td>6.9</td>
<td>7.8</td>
</tr>
<tr>
<td>[10,]</td>
<td>8.6</td>
<td>9.7</td>
<td>4.1</td>
<td>5.5</td>
</tr>
</tbody>
</table>
Processing with a network of SD models

# Preparation of GRiwmInputsModel object
IMnat <- CreateInputsModel(
  griwrm,
  DatesR = DatesR,
  Precip = P,
  PotEvap = E,
  Qobs = Q
)

We use the classical functions and procedures of airGR to build objects embedding the whole network data...

## CreateInputsModel.GRiwm: Treating sub-basin STDIZ_04...
## CreateInputsModel.GRiwm: Treating sub-basin LOUVE_19...
## CreateInputsModel.GRiwm: Treating sub-basin VITRY_25...
## CreateInputsModel.GRiwm: Treating sub-basin CHALO_21...
# Calibration period selection

I_Run <- 366:length(DatesR)

# Set up GRiwrmRunOptions object

RO <- CreateRunOptions(
  InputsModel = IM,
  IndPeriod_Run = I_Run
)

# Calibration criterion:
# the GRiwrmInputsCrit object

IC <- CreateInputsCrit(
  InputsModel = IM,
  FUN_CRIT = airGR::ErrorCrit_NSE,
  RunOptions = RO,
  Obs = Q[I_Run,
)

# Set up GRiwrmCalibOptions object

CO <- CreateCalibOptions(IM)

# Calibration of the network of
# models from upstream to downstream

OC <- Calibration(
  InputsModel = IM,
  RunOptions = RO,
  InputsCrit = IC,
  CalibOptions = CO,
  useUpstreamQsim = TRUE
)

Calibration.GRiwrmInputsModel: Treating sub-basin STD12_04...
Grid-Screening in progress (0% 20% 40% 60% 80% 100%)
Screening completed (81 runs)
  Param = 247.151,  -0.020,   83.096,    2.384
  Crit. NSE[Q]       = 0.8732
Steepest-descent local search in progress
Calibration completed (26 iterations, 273 runs)
  Param = 208.513,  -0.130,   74.440,    3.506
  Crit. NSE[Q]       = 0.9236

Processing with a network of SD models
Simulation run and outputs

```r
# Make a list with sub-basin calibrated parameters
Param <- sapply(names(OC),
function(x) {OC[[x]]$Param})

# Run the network of models
OM <- RunModel(IM,
   RunOptions = RO,
   Param = Param)

# Plot downstream sub-basin
# with airGR::plot.OutputsModel
plot(OM[[4]], Q[I_Run, 4])
```
The GRIwrmOutputsModel has an attribute "Qm3s" containing a data.frame with simulated flows for all the nodes in m³/s.

\[
Qm3s \leftarrow \text{attr(OM, "Qm3s")}
\]

A dedicated plot function displays simulated flow time series for all the nodes.

\[
\text{plot(Qm3s[1:365,])}
\]
airGRiwrn easily integrates direct flow injections or withdrawals in the network by using the matrix of observed flows instead of an hydrological model in network nodes.

Network representation in **airGRiwrn** →
Integrating human withdrawals and releases in the network allow to calibrate the model with influenced observed flows and then to run the model without the influences in order to compute naturalised flows by simulation.

Regime of observed influenced flows (black) and naturalised simulated flows (orange)
A supervisor provides simulation outputs during simulation to controllers that execute a logic of control which apply regulated flows in the network.

The user can write control logic representing withdrawal restriction, reservoir operations... with a complex algorithm.

Results of the simulation of regulated withdrawal restrictions
For complete documentation

https://airgriwrm.g-eau.fr

airGRiwrn: airGR based Integrated Water Resource Management R package

airGRiwrn is an extension of the airGR R package for managing semi-distributive hydrological model on an anthropized catchment.

This package is developed as part of the IN-WOP project (http://www.waterpi.eu/joint-calljoint-call-2018-waterworks-2017-booklet-in-wop) by the mixed research unit G-EAU (https://g-eau.fr) and the HYDRO team of the INRAE HYGAR research unit (https://www.elyon.inrae.fr/hydim/Equipes-de-recherche/HYGAR).

Installation

We need the package \texttt{remotes} to install the package from the Iristea Gitlab repository:

\begin{verbatim}
install.packages("remotes")
\end{verbatim}

The package \texttt{airGRiwrn} is under development and is only available on Gitlab:

\begin{verbatim}
remotes::install_github("in-wop/airGRiwrn", host = "gitlab.irstea.fr", dependencies = TRUE, build = TRUE)
\end{verbatim}

\texttt{dependencies = TRUE} and \texttt{build_vignettes = TRUE} are optional and respectively trigger the installation of extracted packages, if in the vignettes and the compilation and the installation of the vignettes.
Source code, bug tracker...

https://gitlab.irstea.fr/in-wop/airGRiwrm
Package installation

The package will be submitted on CRAN soon.

In the meantime, to install the package*:

```r
install.packages("devtools")  ## required
remotes::install_gitlab("in-wop/airgriwrm", 
  host = "gitlab.irstea.fr")
```

* It also requires the priori installation of 'Rtools' on windows platforms for compiling airGR Fortran's code