EnsScores

Ensemble Scores

User's guide

Jean-Michel Brankart

`http://pp.ige-grenoble.fr/pageperso/brankarj/`

Institut des Géosciences de l'Environnement
Université Grenoble Alpes, CNRS, France

The purpose of EnsScores is to provide tools to compute probabilistic scores of ensemble simulations. The scores evaluates the reliability and the resolution of the simulation by comparison to verification data.

The tools are provided as a library of modules, which can be easily plugged in any existing software. This library includes:

- the computation of the Continuous Rank Probability Score (CRPS);
- the computation of the Reduced Centered Random variable (RCRV) score;
- the computation of scores based on the relative entropy of user-defined events.

1 Description of the modules

In this section, the modules are described one by one, giving for each of them: the method that has been implemented, the list of public variables and public routines (with a description of input and output data), the MPI parallelization, and an estimation of the computational cost as a function of the size of the problem.

1.1 Module: score_crps

The purpose of this module is to compute the CRPS of an ensemble simulation by comparison to verification data.

Method

Public variables

mpi_comm_score_crps MPI communicator to use (default=mpi_comm_world).

crps_missing_value missing value to use where no valid data is available (default=-9999.).

Public routines

crps_score: compute CRPS score (with option to partition the data):

- `crps` **(output)** : CRPS score for each subset of data;
- `reliability` **(output)** : reliability part of CRPS;
- `resolution` **(output)** : resolution part of CRPS;
- `ens` **(input)** : ensemble to evaluate (model equivalent to verification data);
- `verif` **(input)** : verification data;
- `partition` **(input, optional)** : partition of verification data.

crps_cumul: accumulate data to prepare the final computation of the score (for advanced use, if the full ensemble is only made progressively available).

crps_final: compute final score from accumulated data (for advanced use, if the full ensemble is only made progressively available).

MPI parallelization

Computational cost

1.2 Module: score_rcrv

The purpose of this module is to compute the RCRV of an ensemble simulation by comparison to verification data.

Method

Public variables

mpi_comm_score_rcrv MPI communicator to use (default=mpi_comm_world).

rcrv_missing_value missing value to use where no valid data is available (default=-9999.).

rcrv_with_anamorphosis use anamorphosis to compute reduced variable (default=.FALSE.).

rcrv_number_of_quantiles number of quantiles to perform anamorphosis (default=11).

Public routines

rcrv_score: compute RCRV score (with option to partition the data):

`ens_bias` **(output)** : bias component of RCRV (should be 0);

`ens_spread` **(output)** : spread component of RCRV (should be 1);

`ens` **(input)** : ensemble to evaluate (model equivalent to verification data);

`verif` **(input)** : verification data;

`partition` **(input, optional)** : partition of verification data.

rcrv_cumul: accumulate data to prepare the final computation of the score (for advanced use, if the full ensemble is only made progressively available).

MPI parallelization

Computational cost

1.3 Module: score_entropy

The purpose of this module is to compute scores based on the relative entropy of user-defined events.

Method

Public variables

mpi_comm_score_entropy MPI communicator to use (default=mpi_comm_world).

score_entropy_base base to use in the computation of logarithms (default=2.).

Public routines

events_score: compute entropy based score:

`score` **(output)** : ensemble score for each event;

`ens` **(input)** : ensemble to evaluate;

`pref` **(input)** : reference probability distribution for each event;

`events_outcome` **(input)** : callback routine providing the outcome of the events for a given member.

events_relative_entropy: compute relative entropy:

`relative_entropy` **(output)** : relative entropy (with respect to reference distribution);

`ens` **(input)** : ensemble to evaluate;

`pref` **(input)** : reference probability distribution for each event;

`events_outcome` **(input)** : callback routine providing the outcome of the events for a given member.

events_cross_entropy: compute cross entropy:

`cross_entropy` **(output)** : cross entropy (with reference distribution);

`ens` **(input)** : ensemble to evaluate;

`pref` **(input)** : reference probability distribution for each event;

`events_outcome` **(input)** : callback routine providing the outcome of the events for a given member.

events_entropy: compute ensemble entropy:

`entropy` **(output)** : ensemble entropy;

`number_outcome` **(input)** : number of possible outcomes for the events;

`ens` **(input)** : ensemble to evaluate;

`events_outcome` **(input)** : callback routine providing the outcome of the events for a given member.

events_probability: compute events marginal probability distributions from the ensemble:

`pens` **(output)** : ensemble probability distribution for each event;

`ens` **(input)** : ensemble to evaluate;

`events_outcome` **(input)** : callback routine providing the outcome of the events for a given member.

MPI parallelization

Computational cost