The validation of the TELEMAC system links from Python for TELEMAC

Tools for the validation of the TELEMAC system : validateTELEMAC.py

validateTELEMAC.py

Arguments available under validateTELEMAC.py are listed with validateTELEMAC.py -help.
They are :

- -c CONFIGNAME (alternatively –configname=CONFIGNAME)
- -f CONFIGFILE (alternatively –configfile=CONFIGFILE)
- -r ROOTDIR (alternatively –rootdir=ROOTDIR)
- -v VERSION (alternatively –version=VERSION)
- -m MODULES (alternatively –modules=MODULES)
- -s (alternatively –screen)
- -w WDIR (alternatively –workdirectory=WDIR)
- –jobname=JOBNAME
- –queue=HPC_QUEUE
- –walltime=WALLTIME
- –email=EMAIL
- –hosts=HOSTS
- –ncsize=NCSIZE
- –nctile=NCTILE
- –ncnode=NCNODE
- –mpi
- –split
- –merge
- –run
- -b (alternatively –bypass)
- -k RANK (alternatively –rank=RANK)
- –act=ACTION
- –draw=DRAWING
- –save=SAVING
- –cast=CASTING
- –report=REPORT
- –clean
- –use-link

-c allows the user to specify the configuration to be used from the configurations given in the configuration file. Failing that, if the configuration file does not refer to a specific configuration, one will be selected arbitrarily.

-f allows the user to specify a different configuration file (./configs/systel.cfg or the file given by the environment variable SYSTELCFG are considered by default). The full name of the file (i.e. including path) is then required.

-r allows the user to specify a different root directory (generally corresponding to a different version of the TELEMAC-MASCARET system).

-v specifies the report version number for the validation.

-m gives the list of modules to be validated. Default list is taken from the configuration file modules entry.

-s indicates whether to display on screen or save silently.

-w allows the user to specify the temporary directory to run into. This saves splitting the input files again when the same number of processors is used in a parallel run.

–jobname assigns a jobname to help track the run on HPC.

–queue specifies the HPC queue where the job is to be run.

–walltime specifies a walltime (real time, not CPU time) for the job corresponding to the maximum time it should take. If this limit is exceeded, the HPC queue manager will stop the job.

–email defines the list of users (email addresses) to which the HPC queue manager has to send mail, if any, about the job.

–hosts if specified, the job will only run on one or several of the hosts. ';' delimited

–ncsize gives the number of processors to be used for the job.

–nctile gives the number of nodes to be used for the job.

–ncnode specifies the number of cores to be used on each node (cannot be greater than the number of cores in a node). This can be useful if memory requirements for the job are high.

–mpi (**internal command**).

–split if specified, will only prepare and split the input files but will not run the job.

–merge if specified, will merge the different contributions to the job output into one result file. –merge usually follows a –run command.

–run if specified, will only run the multi-processor job but will not recombine the output into one result file at the end of the job. –run usually follows a –split command.

-b bypasses failures and try to carry out validation to the end. A final report will highlight problem areas that need addressing.

-k 4 prime numbers joined by a '.' identifying the rank of a test case for compilation/running/validation/other. Only test cases with a certain rank will be run through the automatic validation. Default is 0.0.0.0, meaning all test cases.

–act targets specific actions from the XML file (e.g. translate, run, compare princi) and will only perform these.

–draw targets specific plots from the XML file and will only produce these.

–save targets specific data extractions from the XML file and will only perform these.

–cast filters specific casting actions from the XML file and will only perform these.

–report creates a report summary of the validation exercise in csv format.

–clean removes all object files and executables, and result files from the subforlders corresponding to the specified configurations and modules.

–use-link allows the user to setup symbolic links to the input files rather than making copies of the files in the temporary working folder. Available on linux platforms only.

It is possible to use combinations of the above. For example:

validateTELEMAC.py : actions

- translate: to translate the steering file from French to English and vice versa
- compile: to compile the project source code
- run: to submit the project run
- cas: to compare the project steering file with the default settings in the dictionary
- princi: to compare the project source code (princi file) to the standard TELEMAC source code

===== Example

```
<action  xref= “1” rank= “3”
 do= “translate;compile;run;cas;princi”
 code= “telemac2d” target= “t2d_bumpflu.cas”
 title= “bumpflu scalar mode”
 />
```

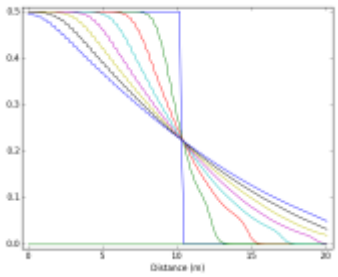
In this example, an action called “1” with rank 3 consists in translating, compiling, running and checking the steering and princi files for the test case t2d_bumpflu.cas of TELEMAC2D.

validateTELEMAC.py : plots

- plot1d
- plot2d

===== Example1

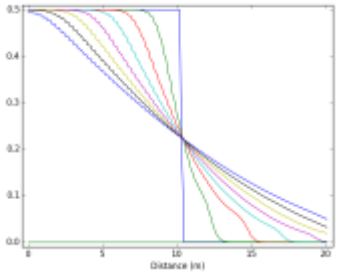
```
<plot1d  xref= “Figure1” type= “history” config= “together” time= ”[0:-1]”
size= ”[12;3]” >
 <layer vars= “surface libre:line” extract= “703;(0;1)” target= “1:T2DRES” />
 <plot1d />
```



In this example, the variable 'surface libre*' from the 2D result file of action “1” (target) is extracted at node 703 and at point with coordinates (0;1) for all the available times between start (0) and end (-1). The 2 time histories are plotted in the png file “Figure1.png”.
The appearance of the plot cannot be changed as of yet. This should be considered as a preliminary plots. The user should extract the data with the save1d command to produce a more polished version.
config=“together” refers to the fact that the results of all the configurations under action “1” (e.g. ubuntu, gfortran, intel as defined by the user in systel.cfg) are extracted. This allows, for example, the comparison of different compilers and/or optimisation options.

===== Example2

```
<plot1d  xref= “Figure2” type= “v-section” extract= ”(0;0.93):(21;1.07)”
config= “oneofall” time= ”[-1]” size= ”[12;5]” >
 <layer vars= “surface libre:line” target= “1:T2DRES” />
 <layer vars= “fond:line” target= “1:T2DRES” />
 <plot1d />
```

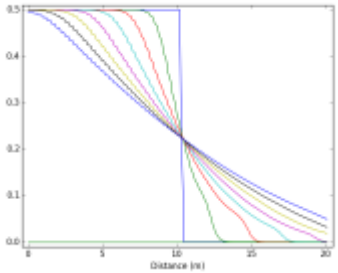


In this example, the variables 'surface libre*' and 'fond*' from the 2D result file of action “1” (target) are plotted along the line with end points (0;0.93) and (21;1.07) for the last timestep only (-1) in the png file “Figure2.csv”.
config=“oneofall” will extract the results for only one of the configurations under action “1” (e.g. ubuntu, gfortran, intel as defined by the user in systel.cfg). It is the user's responsibility to make sure that all configurations agree.
It is noted that config can be assigned the name of a particular configuration as well as is the case in the example below.

===== Example3

```
<plot2d  xref= “Figure3” config= “one” time= ”[-1]” size= ”[12;3]” >
 <layer vars= “VITESSE:map” target= “1:T2DRES” />
 <layer vars= “VITESSE:angle” target= “1:T2DRES” sample=
”(7;0.25)(13.5;1.75){10;4}” />
 <plot2d />
```

In this example,



validateTELEMAC.py : save

- save1d

===== Example1

```
<save1d  xref= “Figure1” type= “history” config= “together” time= ”[0:-1]” >
 <layer vars= “surface libre:line” extract= “703;(0;1)” target= “1:T2DRES” />
 <save1d/>
```

In this example, the variable 'surface libre*' from the 2D result file of action “1” (target) is extracted at node 703 and at point with coordinates (0;1) for all the available times between start (0) and end (-1). The 2 time histories are saved into the CSV file “Figure1.csv”.

===== Example2

```
<save1d  xref= “Figure2” type= “v-section” config= “oneofall” time= ”[-1]” >
 <layer vars= “surface libre:line;fond:line” extract= ”(0;0.93):(21;1.07)”
target= “1:T2DRES” />
 <save1d/>
```

In this example, the variables 'surface libre*' and 'fond*' from the 2D result file of action “1” (target) are extracted along the line with end points (0;0.93) and (21;1.07) for the last timestep only (-1). The 2 cross-sections are saved into the CSV file “Figure2.csv”.

validateTELEMAC.py : cast