UFS Coastal Regression Tests (RT)

Updated Oct 2024

tests/ directory

```
compile.sh
default_vars.sh
detect_machine.sh
fv3_conf
logs
module-setup.sh
parm
rt_35d.conf
rt_coastal.conf
rt.sh
rt_utils.sh
run_compile.sh
run_test.sh
tests
```

Example command

compile.sh frontera "-DAPP=CSTLS –DBUILD_TOOLS=ON" coastal intel YES NO

```
Usage: ./rt.sh -a <account> | -b <file> | -c | -d | -e | -h | -k | -l <file> | -m | -n <name> | -o | -r | -v | -w

  -a  <account> to use on for HPC queue
  -b  create new baselines only for tests listed in <file>
  -c  create new baseline results
  -d  delete run direcotries that are not used by other tests
  -e  use ecFlow workflow manager
  -h  display this help
  -k  keep run directory after rt.sh is completed
  -l  runs test specified in <file>
  -m  compare against new baseline results
  -n  run single test <name>
  -o  compile only, skip tests
  -r  use Rocoto workflow manager
  -v  verbose output
  -w  for weekly_test, skip comparing baseline results
```

Example command

rt.sh –l rt_coastal.conf –a coastal –k -n "coastal_sandy_shinnecock_atm2sch intel"

Component name abbreviations used in the testsuite:

atm	ATMESH data component
pam	PAHM model component
adc	ADCIRC model component
sch	SCHISM model component
fvc	FVCOM model component
ww3	WaveWatch III model component
ww3data	WW3DATA data component

rt_coastal.conf file

Application Based Configurations

CSTLA CSTLAW CSTLPA CSTLPAW
CSTLF
CSTLR
CSTLS CSTLSW CSTLPS CSTLPSW
CSTLW

- ADCIRC
- FVCOM
- ROMS
- SCHISM
- PAHM
- WW3

NOTE: The number of pre-configured applications (left panels) will be reduced significantly. The purpose of the applications is to compile specific model components without user intervention. More flexibility will be added as the UFS-Coastal development progresses.

```
### RT.CONF FORMATTING ###
# COMPILE Line ( Items separated by a | )
# Item 1: COMPILE - This tells rt.conf the following information is to be used in setting up a compile job
# Item 2: Compile number - must be sequential in rt.conf - use as a reference for compile failures
# Item 3: Compiler to use in build (intel or gnu)
# Item 4: CMAKE Options - Provide all CMAKE options for the build
# Item 5: Machines to run on (- is used to ignore specified machines, + is used to only run on specified machines)
## -> EX: + hera orion gaea = compile will only run on hera orion and gaea machines
## -> EX: - wcoss2 acorn = compile will NOT be run on wcoss2 or acorn
# Item 6: [set as fv3]. Used to control the compile job only if FV3 was present, previously used to run a test w/o compiling code
#
# RUN Line ( Items separated by a | )
## NOTE: The build resulting from the COMPILE line above the RUN line will be used to run the test
# Item 1: RUN - This tells rt.conf the following information is to be used in setting up a model run
# Item 2: Test name. (Which test in the tests/tests directory should be sourced)
# Item 3: Machines to run on (- is used to ignore specified machines, + is used to only run on specified machines).
## reference example above
# Item 4: Controls whether the run creates its own baseline or it uses the baseline from a different (control) test.
# Item 5: Test name to compare baselines with if not itself.
```

rt_coastal.conf file

The full list of regression tests (RTs) and their status can be found in the ufs-coastal repository wiki page. Trying to keep the page up-to-date as much as possible.

```
.
.
.
##########
### SCHISM tests: 11-15
COMPILE | 11 | intel | -DAPP=CSTLS -DUSE_ATMOS=ON -DNO_PARMETIS=OFF -DOLDIO=ON | | fv3 |
RUN | coastal_florence_hsofs.atm2sch    | | baseline |
RUN | coastal_sandy_shinnecock_atm2sch | | baseline |

COMPILE | 12 | intel | -DAPP=CSTLSW -DUSE_ATMOS=ON -DUSE_WW3=ON -DNO_PARMETIS=OFF -DOLDIO=ON -
DPDLIB=ON | | fv3 |
RUN | coastal_florence_hsofs.atm2sch2ww3 | | baseline |
RUN | coastal_sandy_shinnecock_atm2sch2ww3 | | baseline |

COMPILE | 13 | intel | -DAPP=CSTLS -DUSE_PAHM=ON -DNO_PARMETIS=OFF -DOLDIO=ON | | fv3 |
RUN | coastal_florence_hsofs_sch_pam     | | baseline |
RUN | coastal_sandy_shinnecock.sch_pam   | | baseline |

COMPILE | 14 | intel | -DAPP=CSTLPS -DUSE_ATMOS=ON -DNO_PARMETIS=OFF -DOLDIO=ON | | fv3 |
RUN | coastal_florence_hsofs.pam2sch    | | baseline |
RUN | coastal_sandy_shinnecock_pam2sch | | baseline |

COMPILE | 15 | intel | -DAPP=CSTLPSW -DUSE_ATMOS=ON -DUSE_WW3=ON -DNO_PARMETIS=OFF -DOLDIO=ON | |
fv3 |
RUN | coastal_florence_hsofs.pam2sch2ww3   | | baseline |
RUN | coastal_sandy_shinnecock_pam2sch2ww3 | | baseline |
.
.
.
```

Compile & Run using rt.sh

rt.sh (found in ufs-coastal/tests folder)

- Enter directory tests: **cd ufs-coastal/tests**
- Make sure that the script is actually executable (use: **chmod +x rt.sh** if it is not)
- Run the script as:

rt.sh orion –l rt_coastal.conf -a <account> –k –n “coastal_ike_Shinnecock_atm2sch intel”

- platform to use
- application configuration file
- account for job submission
- Regression test to use from the rt_coastal.conf file
- compiler

```
Usage: ./rt.sh -a <account> | -b <file> | -c | -d | -e | -h | -k | -l <file> | -m | -n <name> | -o | -r | -v | -w

  -a  <account> to use on for HPC queue
  -b  create new baselines only for tests listed in <file>
  -c  create new baseline results
  -d  delete run direcotries that are not used by other tests
  -e  use ecFlow workflow manager
  -h  display this help
  -k  keep run directory after rt.sh is completed
  -l  runs test specified in <file>
  -m  compare against new baseline results
  -n  run single test <name>
  -o  compile only, skip tests
  -r  use Rocoto workflow manager
  -v  verbose output
  -w  for weekly_test, skip comparing baseline results
```

Steps to run an existing RT case

Step 1: Setup Work Environment

- Log into one of the pre-configured supported platform (Tier 1) such as Orion, Hercules etc.
- Note that the input files to run UFS Coastal specific RTs are not the part of the input files used by UFS Weather Model. See here for their locations on the currently supported platforms.

Step 2: Download the UFS Coastal Code

```
>> git clone --recursive https://github.com/oceanmodeling/ufs-weather-model.git
```

Step 3: Point to the correct input directory (RT) in the *rt.sh* script

```
>>  cd ufs-weather-model/tests/

>>  edit rt.sh
```

Example for MSU's Hercules:

- Open *rt.sh* and find Hercules section
- Edit DISKNM variable as following

```
elif [[ $MACHINE_ID = hercules ]]; then

  module load contrib rocoto
  ROCOTORUN=$(which rocotorun)
  ROCOTOSTAT=$(which rocotostat)
  ROCOTOCOMPLETE=$(which rocotocomplete)

  module use /work/noaa/epic/role-epic/spack-stack/hercules/modulefiles
  module load ecflow/5.8.4
  ECFLOW_START=/work/noaa/epic/role-epic/spack-stack/hercules/ecflow-5.8.4/bin/ecflow_start.sh
  ECF_PORT=$(( $(id -u) + 1500 ))

  QUEUE=windfall
  COMPILE_QUEUE=windfall
  PARTITION=hercules
  dprefix=/work2/noaa/stmp/${USER}
  DISKNM=/work/noaa/epic/hercules/UFS_WM_RT
  DISKNM=/work2/noaa/nems/tufuk/RT
  STMP=$dprefix/stmp
  PTMP=$dprefix/stmp

  SCHEDULER=slurm
  cp fv3_conf/fv3_slurm.IN_hercules fv3_conf/fv3_slurm.IN
  cp fv3_conf/compile_slurm.IN_hercules fv3_conf/compile_slurm.IN
```

Demo Cases: SCHISM + DATM (atmospheric forcing only)

Step 4: Run the Regression Test "DATM+SCHISM"

```
>> cd ufs-weather-model/tests/

>> ./rt.sh -l rt_coastal.conf -a <account> -k -n "coastal_ike_shinnecock_atm2sch intel"
```

Replace <account> with your HPC allocation name.

```
+ compiles[$COMPILE_NR]=11_intel
+ echo 'COMPILING 11_intel'
COMPILING 11_intel
+ [[ false == true ]]
+ [[ '' != '' ]]
+ [[ false == true ]]
+ create_or_run_compile_task
+ cat
+ [[ false == true ]]
+ [[ false == true ]]
+ ./run_compile.sh /work2/noaa/nems/tufuk/COASTAL/ufs-coastal/tests /work2/noaa/stmp/tufuk/stmp/tufuk/FV3_RT/rt_2546607 '-DAPP=CSTLS -DUSE_ATMOS=ON -DNO_PARMETIS=OFF -DOLDIO=ON' 11_intel
```

it compiles the executable

run directory

```
+ ((  NODES * TPN < TASKS  ))
+ NODES=1
+ PPN=6
+ ((  TASKS - ( PPN * NODES ) > 0  ))
+ cat
+ [[ hercules = jet ]]
+ [[ false == true ]]
+ [[ false == true ]]
+ ./run_test.sh /work2/noaa/nems/tufuk/COASTAL/ufs-coastal/tests /work2/noaa/stmp/tufuk/stmp/tufuk/FV3_RT/rt_2546607 coastal_ike_shinnecock_atm2sch 001 11_intel
```

prepares run directory and submits job

Demo Cases: DATM+SCHISM (atmospheric forcing only)

Step 4: Run the Regression Test "DATM+SCHISM"

```
+ continue
+ read -r line
+ '[' '' ']'
+ [[ false == true ]]
+ [[ false == true ]]
+ set +e
+ cat /work2/noaa/nems/tufuk/COASTAL/ufs-coastal/tests/logs/log_hercules/compile_11_intel_time.log
+ cat /work2/noaa/nems/tufuk/COASTAL/ufs-coastal/tests/logs/log_hercules/rt_001_coastal_ike_shinnecock_atm2sch_intel.log
+ FILES='fail_test_* fail_compile_*'
+ for f in $FILES
+ [[ -f fail_test_* ]]
+ for f in $FILES
+ [[ -f fail_compile_* ]]
+ [[ -e fail_test ]]
+ echo

+ echo REGRESSION TEST WAS SUCCESSFUL
REGRESSION TEST WAS SUCCESSFUL
+ echo
+ echo REGRESSION TEST WAS SUCCESSFUL
+ rm -f 'fv3_*.x' fv3_11_intel.exe modules.fv3_11_intel.lua 'modulefiles/modules.fv3_*' keep_tests.tmp
+ [[ true == false ]]
+ [[ false == true ]]
+ [[ false == true ]]
+ [[ coastal_ike_shinnecock_atm2sch != '' ]]
+ rm -f rt_single.conf
+ date
++ printf '%02dh:%02dm:%02ds\n' 2 53 46
+ elapsed_time=02h:53m:46s
+ echo 'Elapsed time: 02h:53m:46s. Have a nice day!'
+ echo 'Elapsed time: 02h:53m:46s. Have a nice day!'
Elapsed time: 02h:53m:46s. Have a nice day!
```

checks against the baseline

log file for baseline check

test runs without any issue

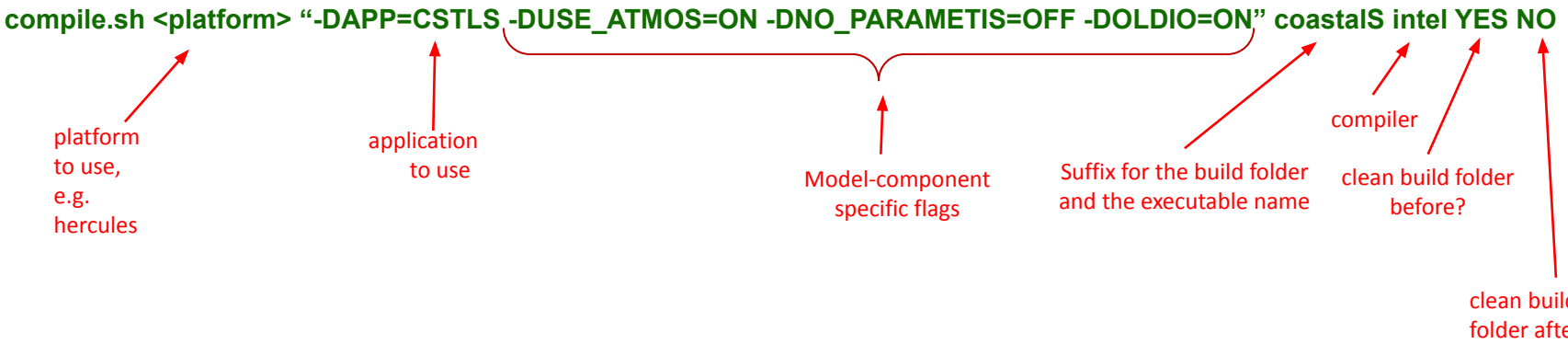
NOTE: Baselines are both platform and compiler dependent (Intel vs. GNU)

UFS Coastal specific baselines are currently only available on Orion, Hercules, and Frontera

General steps to compile and configure a new application

Compile using compile.sh

- Enter tests directory : **cd ufs-coastal/tests**
- Make sure that the script is actually executable (use: **chmod +x compile.sh** if it is not)
- Run the script as:



The above command will build SCHISM with the supplied options in the "build_fv3_coastalS" directory. The final UFS executable "fv3_coastalS.exe" will be located in the tests directory.

The next step will be to copy the UFS executable to the "work" directory where all model and UFS configuration files are located.

General steps for configuring an application

- **Load** the required modules to compile/run UFS-Coastal:
  - module use ufs-weather-model/modulefiles and then module load ufs_frontera.intel
- **Compile** the UFS executable using one of the predefined application cases
  compile.sh <platform> "-DAPP=CSTLS -DUSE_ATMOS=ON -DNO_PARAMETIS=OFF -DOLDIO=ON" coastalS intel YES NO
- **Copy** the executable to a new "work" directory where you'll put all model and UFS config files
- **Configure** each model component as usual for the application (as a standalone run)
- **Generate** ESMFmesh files for: (a) atmospheric forcing (required by CDEPS) and (b) WW3 (if it is in your application, required by WW3)
  - For atmospheric forcing we use: (a) a ncl script to generate the corresponding SCRIP file and (b) the ESMF_Scrip2Unstruct program to generate the ESMFmesh file
  - For WW3 we run: (a) the ww3_grid program to generate the SCRIP file from *.msh and (b) the ESMF_Scrip2Unstruct program to generate the ESMFmesh file
- **Collect** all model configuration and input files into the work folder where the UFS executable is located
- **Submit** the job using your SLURM, PBS or any other scheduler / job submission script
- Make sure that the modules and libraries used to compile the UFS executable are properly loaded from within the job submission script