CLOBAL SYSTEMS LABORATORY

An MPAS App with Unified Workflow Tools Janet Derrico, Emily Carpenter, Christina Holt CIRES/NOAA GSL



NOAA

The Rapid Refresh Forecast System (RRFS) development teams recommend transitioning to the MPAS dynamical core to address biases in storm intensity and rainfall associated with the FV3 dynamical core. NOAA GSL scientists are now rigorously testing the MPAS model, aided by a bespoke MPAS application developed in collaboration with the UFS Unified Workflow (UW) team. Leveraging expertise of those who came before us from NCAR, NSSL, and GSL, this application features a portable build system that utilizes spack-stack-installed modules for libraries and packages used in building the developmental codes. The MPAS App installs Miniconda to support the Python environments needed for codes that configure and run a workflow with Rocoto.

Opaque Clouds

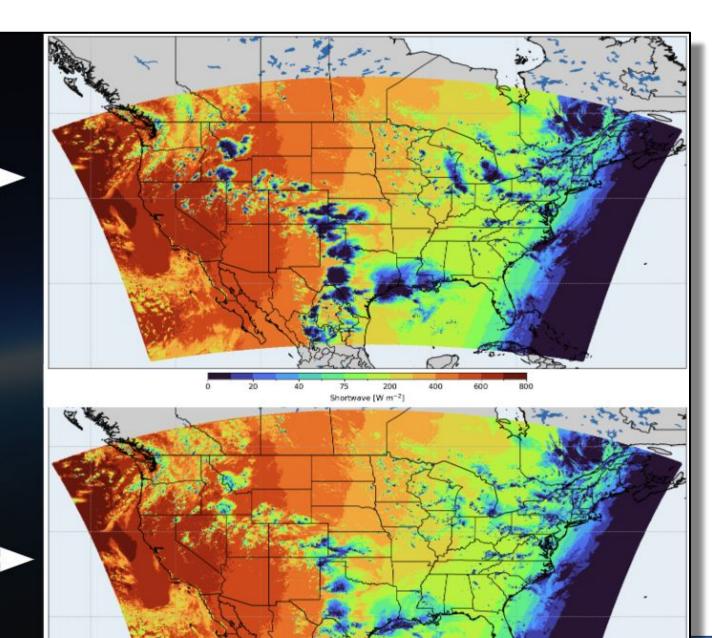
Original Configuration was not configured to use the effective radii from the microphysics; defaulted to very poorly set values, making clouds too opaque.

Update to use effective radii from microphysics Changed the default value of

config microp re to true

Slide credit: Anders Jensen, GSL

 Currently working on a revision of the macrophysics tailored for MPAS.



Benefits of the MPAS App:

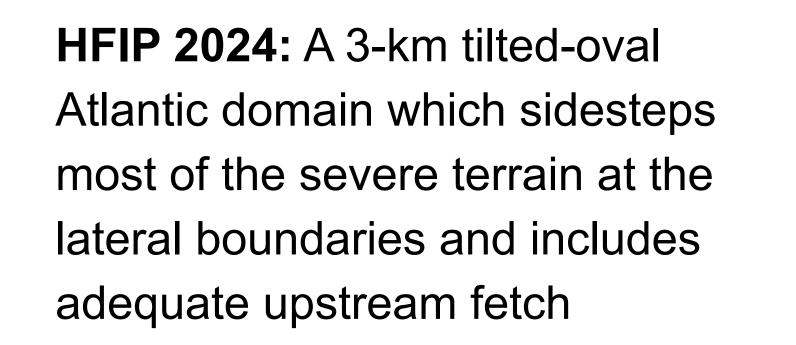
• All scientists start with the same codes

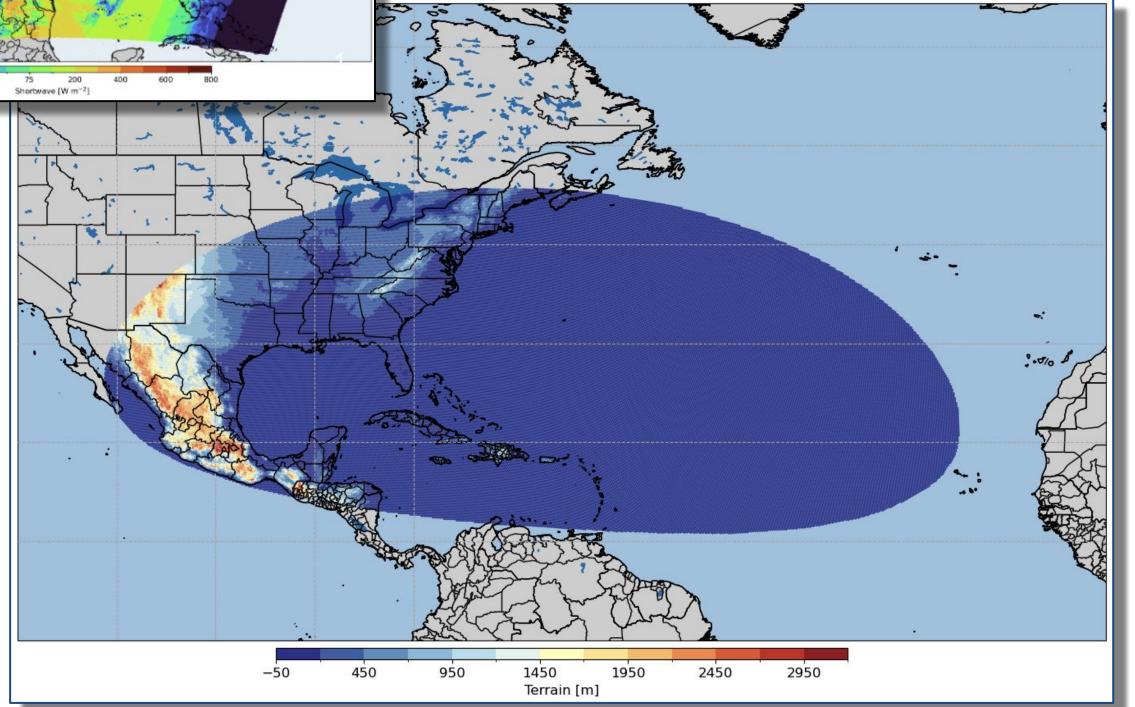
CIRES

- Changes to code are tracked via Git and changes to experiments via YAML configs
- Drastically reduces lines of code we're responsible for-one config file handles input for all steps of model • Easily portable to other platforms

This solution aims to provide NOAA GSL researchers with a user-friendly environment for advancing MPAS-related research while broader UFS integration plans are finalized. Ultimately, the goal is to incorporate the MPAS dycore into the ufs-weather-model and include MPAS as a supported choice in the UFS Short Range Weather App. The experience gained from the GSL MPAS App will directly inform the development of the UFS SRW App, ensuring seamless integration and shared functionality across both platforms. The MPAS App will be used for GSL's contributed experiments in the 2024 Hurricane Forecast Improvement Program (HFIP).

The MPAS App contains a build system and the Unified Workflow Framework for configuring and running experiments.





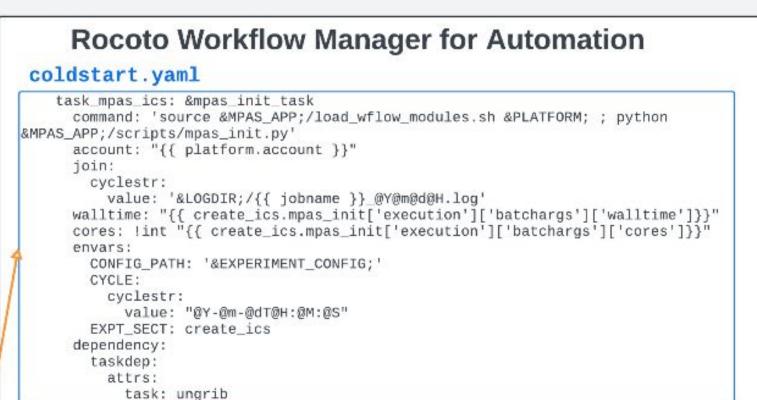
Unified Workflow (UW)

UW Tools is a versatile Python package that provides a command-line interface and a Python API containing two types of utilities: generic tools that streamline the configuration process and **UW Drivers** responsible for running of computational codes common to workflow tasks within the UFS.

UW defines a consistent **YAML-based control language** that

defaults.yaml experiment_dir: mpas_app: user.yam prepare_grib experiment_dir: /scratch2/place/for/outpu ungrib: platform: hera create_ics mpas_init: account: my accourt boundary_conditions interval_hours: 6 length: 6 offset: 0 UW Config Tool path: "{{ prepare_grib.ungrib['run_dir'] }}" execution batchargs uw config realize -i defaults.yaml --update-file user.yaml cores: 4 walltime: 01:30:00 --output-file experiment.yaml module use {{ user.mpas_app }}/modulefiles

Unified Workflow Framework



- module load build // user platform \\ intel			
<pre>- module load build_{{ user.platform }}_intel executable: "{{ user.mpas_app }}/exec/init_atmosphere_model"</pre>			LIW Doo
mpiargs:			UW Roce
- "ntasks={{ create_ics.mpas_init['execution']['batchargs']['cores'] }}"			
mpicmd: srun			un reacte realize a coldetart ward a react
files_to_copy: &mpas_init_files_to_copy			uw rocoto realize -c coldstart.yaml -o rocoto
<pre>stream_list.atmosphere.diagnostics: "{{ user.mpas_app }}/src/MPAS-Model/stream</pre>			
<pre>stream_list.atmosphere.output: "{{ user.mpas_app }}/src/MPAS-Model/stream_list</pre>	.atmosphere.output"		
<pre>stream_list.atmosphere.surface: "{{ user.mpas_app }}/src/MPAS-Model/stream_lis</pre>	t.atmosphere.surface" experiment.yaml	1	<task name="mpas ics"></task>
	K		<account>nrtrr</account>
files_to_link: &mpas_init_files_to_link	experiment_dir: /scratch2/place/for/output		<pre><account>in crr</account> <cores>4</cores></pre>
CAM_ABS_DATA.DBL: "{{ user.mpas_app }}/src/MPAS-Model/CAM_ABS_DATA.DBL" CAM_AEROPT_DATA.DBL: "{{ user.mpas_app }}/src/MPAS-Model/CAM_AEROPT_DATA.DBL"	mpas_app: "/path/to/clone"		<walltime>01:30:00</walltime>
	mpus_upp: /puch/co/orono		<pre><command/>source &MPAS_APP;/load_wflow_modules.sh &PLATFORM ; pyt</pre>
namelist:	prepare_grib:		&MPAS_APP;/scripts/mpas_init.py
<pre>base_file: "{{ user.mpas_app }}/src/MPAS-Model/namelist.init_atmosphere"</pre>	ungrib:		<jobname>mpas_ics</jobname>
update_values: &mpas_init_update_values			<join></join>
nhyd_model:	create_ics:		<cyclestr>&LOGDIR/mpas_ics_@Y@m@d@H.log</cyclestr>
config_init_case: 7	<pre>mpas_init: &mpas_init_config boundary conditions:</pre>		
	interval_hours: 6		<envar></envar>
run_dir: '{{ user.experiment_dir }}/{{ cycle.strftime("%Y%m%d%H") }}/mpas_ics'	length: 6		<name>CONFIG_PATH</name>
streams:	offset: 0		<value>&EXPERIMENT_CONFIG;</value>
platform	path: "/path/to/ungrib/run/dir"		
<pre>platform: account: '{{ platform.account }}'</pre>	execution:		<dependency></dependency>
scheduler: '{{ platform.scheduler }}'	batchargs:		<taskdep task="ungrib"></taskdep>
reate_lbcs:	cores: 4 walltime: 01:30:00		
mpas_init:	envcmds:		
	- module use /path/to/clone/modulefiles		
orecast:	- module load build_hera_intel		
mpas:	executable: "/path/to/clone/exec/init_atmosphere_model"		UW Drivers
	mpiargs:		
	- "ntasks=4" mpicmd: srun		uw ungrib runcycle 2024-05-15T18
	files_to_copy:		
	stream_list.atmosphere.diagnostics:		
	"/path/to/clone/src/MPAS-Model/stream_list.atmosphere.diagnostics"		key-path prepare_grib
	stream_list.atmosphere.output:		
	"/path/to/clone/src/MPAS-Model/stream_list.atmosphere.output" stream_list.atmosphere.surface:		
	"/path/to/clone/src/MPAS-Model/stream_list.atmosphere.surface"		uw mpas_init runcycle 2024-05-15T1
	files_to_link: &mpas_init_files_to_link		-c experiment.yaml
MDAS Ann Configuration	CAM_ABS_DATA.DBL: "/path/to/clone/src/MPAS-Model/CAM_ABS_DATA.DBL"		
MPAS App Configuration	CAM_AEROPT_DATA.DBL: "/path/to/clone/src/MPAS-Model/CAM_AEROPT_DATA.DBL"		key-path create_ics
Lovor	namelist:		
Layer	<pre>namelist: base_file: "/path/to/clone/src/MPAS-Model/namelist.init_atmosphere"</pre>		which which which and a solution of the
	update_values: &mpas_init_update_values		uw mpas_init runcycle 2024-05-15T1
	nhyd_model:		-c experiment.yaml
	config_init_case: 7		
			key-path create_lbcs
	run_dir: '{{ user.experiment_dir }}/{{ cycle.strftime("%Y%m%d%H") }}/mpas_ics' streams:		
	a chi e dilla a		
	platform:		uw mpas runcycle 2024-05-15T18
	account: 'my_account'		
	scheduler: 'slurm'		-c experiment.yaml
	create_lbcs:		key-path forecast
	mpas_init:		Key-path Torecast
	forecast:		
	mpas:		MPAS APP Run Scripts

configures tools including to compare unordered namelist files with mixed-case keys, render templates, and even create the Rocoto workflow manager's XML workflow definition file.

Drivers are designed for a specific program and are responsible for:

- Preparing run directories by staging data files
- Defining a component's resource requirements
- Executing a program, either directly or by submitting a job to a batch system
- Validating their config file against a schema

Historically, each NWP application has created bespoke scripts to drive its workflows, leading to duplication of effort and incompatibility across UFS applications. The UW vision is to unify these disparate configuration layers into a single, cohesive framework that interfaces with existing workflows and workflow managers. The aim is to alleviate common pain points, provide the tools necessary to create a concise and efficient runtime environment, flatten the learning curve to get what you need out of an app, and empower scientists to realize their experiments exactly as they envision them.





