DCMIP-based Initialization of Multiple Tropical Cyclones in an Idealized Framework with HAFS-FV3

Kyle Ahern

October 29, 2020

Version 0.0 - 13 October 2020
Version 0.1 - 28 October 2020
Version 0.2 - 29 October 2020

1 Introduction

This document provides a description of an idealized initialization in FV3, wherein the initial state of the model includes one or multiple warm-core vortices. Certain initial characteristics of each vortex (e.g., vortex position, size, intensity) are specified by the user in an easy-to-use namelist. At the time of this writing, the immediate idealized case has been successfully tested on uniform, stretched, and single static-nested grids in FV3, using a *dynamics-only* configuration. While it is now possible to attempt running the idealized simulation *with* physics packages, it may not function properly (and indeed, *will not* function properly if the user intends to initialize the model state as an aquaplanet).

The motivation for this work is to test the forthcoming implementation of multiple moving nests in HAFS-FV3. The general goal of the multiple moving nest effort is to provide an effective and efficient way to forecast multiple tropical cyclones (TCs) for research and operations—effective in that the high-resolution nests allow for representation of convective-scale features important in TC forecasting, and efficient in that the area over which high-resolution computations are performed can be restricted to relevant features that may be separated by thousands of kilometers (such as several TCs across the Atlantic and Pacific basins). Straightforward alternatives to this approach include using a very large nest that encapsulates all relevant features, or limiting the nest to only one or a few closely neighboring features. These alternatives are undesirable as the former necessitates excessive resources, and the latter may be unable to represent relevant features outside the specified nest.

Strictly speaking, testing the multiple moving nest framework does not require an idealized configuration. However, given the goal of that effort, the nests will have to track and move with a TC over time. A trial case with multiple idealized TCs can be used to test TC-tracking nests. A *real* case could also be chosen to test the new multiple moving nest framework, but simulating real cases can come with a gamut of complicating factors specific to the scenario, including those related to topography, initial conditions, and physics options. The present philosophy is that it behooves the developer to test new features with the simplest trials first. Naturally, we expect programming bugs and problems to emerge as we test the multiple moving nest framework, and narrowing the scope of potential causes for these issues using simple tests will facilitate understanding the problems at hand.

Additionally, this idealized testing capability has potential beyond testing of the multiple moving nest framework in HAFS-FV3. At this time, the test can be used to simulate the interaction between multiple and variable vortices, which can be useful for research and educational purposes. The original source code is modular and documented, which should aid understanding and customization (if, for instance, one wanted to adjust or add their own features or options).

2 Methodology

The initialization of a TC in this idealized case is based on the initialization specified for the Dynamical Core Model Intercomparison Project's test cases in 2016 (DCMIP, [Ulr+08]). Although this case is slightly altered from DCMIP2016's TC test case, most of the assumptions and formulations are the same in principle, but covered here in the context of HAFS-FV3 for the sake of completeness and convenience. The original case, as noted in the DCMIP2016 documentation, is based off works from [RJ12; RJ11b; RJ11a; RJ11c].

In this test, initial vortices are superimposed onto a background environment at rest and in hydrostatic balance. In the default case with one initial vortex (identical to DCMIP2016), the initialization is favorable for TC rapid intensification over a 10-day simulation period. Profiles for specific humidity q, virtual temperature T_v, and pressure p are prescribed. The initial state assumes that the motion is in approximate gradient wind balance. By default, the vertical structure is meant to closely represent the [Jor58] observational sounding. From these specifications, the required prognostic and diagnostic variables for FV3 are specified. The prognostic variables are the zonal wind u, meridional wind v, vertical wind w, potential temperature θ, pressure thickness δp, and q. Various auxiliary pressure and height fields constitute the required diagnostic fields, which will be indicated later.

First, the model state is initialized isothermally, at rest, and in hydrostatic balance by

setting

$$\begin{aligned} u(i,j,k) &= 0, \\ v(i,j,k) &= 0, \\ w(i,j,k) &= 0, \\ Q(i,j,k,n_q) &= q(i,j,k) = 3 \times 10^{-6}, \end{aligned}$$

and then calling the **hydro_eq** subroutine in **init_hydro.F90** (which sets θ and δp). In the above, i is the abscissal coordinate, j is the ordinate coordinate, k is the vertical coordinate, Q is the set of atmospheric water fields in FV3, and n_q is the index for water species. The details of this procedure are not imperative as these fields will be overwritten later, but the purpose of this step is to guarantee that the model state is set regardless of the user's specifications.

2.1 Combining Multiple Vortex Structures

Most of the initial fields will have horizontal structure based on distances from the centers of vortices. For each vortex specified by the user, the radius r is calculated:

$$\begin{aligned} r_n(i,j) &= a \arccos[\sin\phi_{n,c}\sin\phi + \cos\phi_{n,c}\cos\phi\cos(\lambda - \lambda_{n,c})], \\ a &= 6.3712 \times 10^6 \text{ m}, \\ \phi &= \phi(i,j), \\ \lambda &= \lambda(i,j), \end{aligned} \tag{1}$$

where the subscript n denotes the vortex index (e.g., vortex #1, vortex #2), a is Earth's radius, ϕ is latitude, λ is longitude, and the subscript c denotes a value at the vortex center.

Because we want to be able to initialize multiple TCs at the same time, we must determine how the initial fields of the vortices will interact, if they will at all. As a first guess, one could assume that the fields associated with the initialized TCs are essentially waves that can interfere with other waves. If it is further assumed that the system is initially linear, then the superposition principle can be applied: The responses from each vortex initialization alone can be summed to yield the fields that would result from all of the vortices initialized at the same time. Here, we take a similar but altered approach, where the waves resulting from each individual vortex are modified using inverse distance weighting:

$$\begin{aligned} c_n(i,j) &= \left(\frac{1}{r_n(i,j)}\right)^2, \\ c_0(i,j) &= \sum_{n=1}^{N}\left(\frac{1}{r_n(i,j)}\right)^2, \\ W_n(i,j) &= \frac{c_n(i,j)}{c_0(i,j)}. \end{aligned}$$

In the expressions above, N is the number of vortices to initialize, c_n is a vortex-dependent weighting parameter, c_0 is the sum of all c_n, and W_n is an inverse distance-based weight for vortex n. The sum of all weights at (i,j) is unity. For an initial field X that is dependent on the number of vortices chosen (such as pressure), we assume that the weighted sum of initial fields from each vortex alone yields the field with all vortices present:

$$X(i,j,k) = \sum_{n=1}^{N} W_n(i,j) X_n(i,j,k), \tag{2}$$

where X_n is the field X that would be used if only vortex n was to be initialized. If one wanted to employ the superposition principle explicitly, then they could decompose each field into base-state (X_0) and perturbation (X') components, assume all perturbations are associated with the vortices, calculate each such perturbation from each vortex, and then sum them:

$$X_n(i,j,k) = X_0(i,j,k) + X'_n(i,j,k), \tag{3}$$

$$X(i,j,k) = X_0(i,j,k) + \sum_{n=1}^{N} X'_n(i,j,k). \tag{4}$$

Note that using equations (2) and (3), one can easily find,

$$X(i,j,k) = X_0(i,j,k) + \sum_{n=1}^{N} W_n(i,j) X'_n(i,j,k), \tag{5}$$

which when compared with equation (4) represents the difference between the two methods described here. The distance-weighted method will be used by default, but removing the weighting would be straightforward.

2.2 Pressure Fields

The pressure thickness δp is the first calculated prognostic field. For the purpose of calculating δp in FV3, the pressure at any point in space (except the explicitly set model top) is calculated as

$$p(i,j,k) = A(k) + B(k) p_s(i,j), \tag{6}$$

where p_s is the surface pressure, and $A(k)$ and $B(k)$ are given vertical profiles associated with the number of eta-levels chosen for the simulation. Essentially, all that is required to solve for δp is the surface pressure field.

The (total) pressure field associated with vortex n has radial and vertical structure:

$$\begin{aligned} p_n(i,j,k) &= p_0(k) + p'_n(i,j,k) \\ p'_n(i,j,k) &= \begin{cases} -\Delta p_n \exp\left[-\left(\frac{r_n}{S_n}\right)^{3/2} - \left(\frac{z}{z_p}\right)^2 \right] \left(\frac{T_{v,0} - \Gamma z}{T_{v,0}} \right)^{\frac{gR_d}{\Gamma}}, & \text{if } 0 \le z \le z_t \\ 0, & \text{if } z > z_t. \end{cases} \end{aligned} \tag{7}$$

In (7), p_0 is the background vertical profile of pressure, Δp_n is the surface pressure depression between the center of vortex n and the background, S_n indicates the pressure change in the radial direction (a half-width for the wave-like pressure perturbation associated with vortex n, a "size" parameter), z is height, z_p indicates vortex pressure decay with height, $T_{v,0}$ is the background virtual temperature at the surface, $\Gamma \equiv -dT_v/dz$ is the environmental lapse rate, g is gravitational acceleration (9.8 m s^{-2} in FV3), R_d is the gas constant for dry air (287.04 J kg^{-1} K^{-1}), and z_t is the tropopause height. Several of these parameters can be specified explicitly by the user: Δp_n, S_n, z_p, z_t, and Γ. The height field z is solved for iteratively and described in Appendix A. The $T_{v,0}$ term,

$$T_{v,0} = T_0(1 + 0.608 q_0),$$

is set implicitly by the user through setting the background surface temperature T_0 and background surface humidity q_0. The background profile of pressure is separated by the tropopause, and is calculated using the hydrostatic assumption, the ideal gas law, and an isothermal stratosphere:

$$p_0(k) = \begin{cases} p_0(k_s) \left(\frac{T_{v,0} - \Gamma z}{T_{v,0}} \right)^{\frac{g}{R_d \Gamma}}, & \text{if } 0 \le z \le z_t; \\ p_t \exp\left(\frac{g(z_t - z)}{R_d T_{v,t}} \right), & \text{if } z > z_t; \end{cases}$$

where k_s is the lowest atmospheric layer in FV3, p_t is the tropopause pressure, and $T_{v,t}$ is the background virtual temperature at the tropopause. The user can specify the surface background pressure $p_0(k_s)$. The tropopause pressure is given by

$$p_t = p_0(k_s) \left(\frac{T_{v,t}}{T_{v,0}} \right)^{\frac{g}{R_d \Gamma}},$$

and $T_{v,t}$ is given by

$$T_{v,t} = T_{v,0} - \Gamma z_t.$$

The surface pressure can be determined from (7) by setting $z = 0$:

$$p'_n(i,j,k_s) = -\Delta p_n \exp\left[-\left(\frac{r_n}{S_n}\right)^{3/2}\right],$$
$$p(i,j,k_s) = p_s(i,j) = p_0(k_s) + \sum_{n=1}^{N} W_n(i,j) p'_n(i,j,k_s). \tag{8}$$

Using equations (6) and (8), we can determine the layer thickness in pressure for FV3:

$$\delta p(i,j,k) = A(k+1) - A(k) + p_s(i,j)[B(k+1) - B(k)]. \tag{9}$$

Equations (6) and (8) are also used to find various auxiliary pressure fields in FV3, which include an assortment of transformations of the actual pressure field (such as the log-pressure field).

2.3 Temperature Fields

Next, the field for potential temperature θ is established. For this purpose, we compute the virtual temperature T_v, which can be converted to θ. From hydrostatic balance and the ideal gas law,

$$T_v(i,j,k) = -\frac{g}{R_d}\left(\frac{\partial \ln p}{\partial z}\right)^{-1} \approx -\frac{g}{R_d}\left(\frac{\delta z}{\delta(\ln p)}\right), \tag{10}$$

where δz is the thickness of layer k and $\delta(\ln p)$ is the layer's log-pressure thickness (similar to δp). The virtual temperature field can also be decomposed into a background state ($\bar{T}_v$) and a perturbation/vortex component (T'_v):

$$T_v(i,j,k) = \bar{T}_v(k) + T'_v(i,j,k). \tag{11}$$

The background virtual temperature profile is defined as

$$\bar{T}_v(k) = \begin{cases} T_{v,0} - \Gamma z, & \text{if } 0 \le z \le z_t \\ T_{v,t} = T_{v,0} - \Gamma z_t, & \text{if } z > z_t. \end{cases}$$

The perturbation virtual temperature field associated with vortex n is

$$T'_{v,n}(i,j,k) = \begin{cases} \bar{T}_v \left[\left(1 + \dfrac{2R_d \bar{T}_v z}{g z_p^2 \left[1 - \dfrac{p_0(k_s)}{\Delta p_n} \exp\left(\left(\dfrac{r_n}{S_n}\right)^{3/2} + \left(\dfrac{z}{z_p}\right)^2 \right) \right]} \right)^{-1} - 1 \right], & \text{if } 0 \le z \le z_t \\ 0, & \text{if } z > z_t. \end{cases} \tag{12}$$

Although equation (10) is used to calculate θ in FV3, equation (12) is included in the method as it is used to iteratively solve for z (see Appendix A).

2.4 Momentum Fields

After the temperature field is handled, the u and v fields are initialized. Note that FV3's momentum fields are on a grid staggered horizontally from the mass and thermal fields, where u is aligned along the abscissal sides of each grid cell and v is aligned along the ordinate sides of each grid cell. All of these momentum fields' dependencies (r_n, W_n, and all required fields therefrom) are recomputed on the staggered grid in question.

We calculate the horizontal winds using the quadratic expression for tangential wind V assuming gradient-wind balance: A balance of the Coriolis, centrifugal, and pressure gradient forces. For this, we require pressure and virtual temperature fields, which we have already computed. Above the tropopause ($z > z_t$), the wind is set to zero. Elsewhere, the expression for the tangential wind associated with vortex n is

$$V_n(i,j,k) = -\frac{f_{n,c} r_n}{2} + \mathrm{sgn}(\phi_{n,c}) \sqrt{\frac{f_{n,c}^2 r_n^2}{4} + \frac{r_n R_d T_{v,n}}{p_n}\left(\frac{\partial p}{\partial r}\right)_n}, \tag{13}$$

where $f_{n,c} = 2\Omega \sin(\phi_{n,c})$ is the Coriolis parameter at the central latitude $\phi_{n,c}$ of vortex n, $\mathrm{sgn}(\phi_{n,c})$ is the sign of $\phi_{n,c}$ (i.e., positive in the Northern Hemisphere, negative in the Southern Hemisphere), and $T_{v,n}$ is the virtual temperature field associated with vortex n alone. The variable pressure and virtual temperature fields can be eliminated from equation (13) using equations (7), (11), and (12):

$$V_n(i,j,k) = -\frac{f_{n,c} r_n}{2} + \mathrm{sgn}(\phi_{n,c}) \sqrt{\frac{f_{n,c}^2 r_n^2}{4} - \frac{\frac{3}{2}\bar{T}_v R_d \left(\frac{r_n}{S_n}\right)^{3/2}}{1 + \frac{2\bar{T}_v R_d z}{g z_p^2} - \frac{p_0(k_s)}{\Delta p_n} \exp\left(\left(\frac{r_n}{S_n}\right)^{3/2} + \left(\frac{z}{z_p}\right)^2\right)}}. \tag{14}$$

The zonal and meridional components are calculated from the tangential wind field of vortex n similar to [NJ08]:

$$\begin{aligned} d_{n,1}(i,j) &= \sin\phi_{n,c}\cos\phi - \cos\phi_{n,c}\sin\phi\cos(\lambda - \lambda_{n,c}), \\ d_{n,2}(i,j) &= \cos\phi_{n,c}\sin(\lambda - \lambda_{n,c}), \\ d_n(i,j) &= \max\left(\varepsilon, \sqrt{d_{n,1}^2 + d_{n,2}^2}\right), \\ u'_n(i,j,k) &= \frac{V_n d_{n,1}}{d_n}, \qquad (15) \\ v'_n(i,j,k) &= \frac{V_n d_{n,2}}{d_n}. \qquad (16) \end{aligned}$$

In the above, $\varepsilon = 10^{-25}$ is used to prevent divisions by zero. As the background state is at rest, the total horizontal wind field with all n vortices is given by

$$u(i,j,k) = \sum_{n=1}^{N} W_n u'_n, \tag{17}$$

$$v(i,j,k) = \sum_{n=1}^{N} W_n v'_n. \tag{18}$$

The total vertical motion w is zero.

2.5 Moisture Field

In a system with height as its vertical coordinate, the initial specific humidity field q has only vertical structure, with no perturbations specific to any given vortex. However, because FV3's height field is variable across horizontal space, the model's specific humidity field can have three-dimensional structure. The specific humidity field is set by

$$q(i,j,k) = \begin{cases} \max\left(q_t, q_0 \exp\left(-\frac{z}{z_{q1}}\right) \exp\left[-\left(\frac{z}{z_{q2}}\right)^2\right]\right), & \text{if } 0 \le z \le z_t \\ q_t, & \text{if } z > z_t. \end{cases} \tag{19}$$

In (19), q_t is a the specific humidity above the tropopause; and z_{q1} and z_{q2} are shape parameters describing the linear and quadratic decreases of tropospheric q with height, respectively. All of these parameters can be specified by the user.

3 Running the Case

To allow for easy modificiations to the test case without repeated recompilation of the model, some code of the FV3 core was modified (namely, **fv_control.F90** and **fv_restart.F90**). Furthermore, the original module to handle the vast arrangement of test cases in **test_cases.F90** was replaced with a system of smaller modules, which was done to facilitate reading, debugging, and customizing code from a specific test case such as this one. The details of these modifications are outlined in Appendix B.

To run the idealized case, one must set-up and run all steps of the modeling system leading up to the forecast step (e.g., grid generation, orography), using the grid specifications desired.

The forecast input namelists (**input.nml** and others) must be adjusted to accomodate an idealized case. Thus, if using the HAFS workflow, then one may have to stop the process at the forecast step to modify the input namelists manually, unless the workflow is modified to streamline idealized simulations (which may be beyond the intended scope of that workflow). Table 1 outlines the namelist options to set for the aquaplanet configuration with no physics packages.

The user can customize their case settings using the new **&test_case_nml** namelist block, which contains all the options pertinent to the case specified by the **selected_case** option in the **&fv_core_nml** block. The options are outlined in Table 2.

Forecast Namelist Variables		
Namelist Block	Option	Value
&atmos_model_nml	dycore_only	.true.
&fv_core_nml	external_ic	.false.
	external_eta	.false.
	nggps_ic	.false.
	ncep_ic	.false.
	ecmwf_ic	.false.
	mountain	.false.
	warm_start	.false.
	selected_case[*]	99
&test_case_nml	aqua[*]	.true.

Table 1: Forecast namelist (e.g., **input.nml**) options to set for the aquaplanet configuration of the multiple vortex test case. Options with an asterisk are newly implemented in FV3 (see Appendix B).

Customization options in &test_case_nml			
Option	Symbol	Code name	Default setting
Number of vortices	N	num_vortex	0 (16 maximum)
Use aquaplanet	N/A	aqua	.true.
Central longitude	$\lambda_{n,c}$	lon(N)	0.0, 0.0, 0.0, ... °E
Central latitude	$\phi_{n,c}$	lat(N)	0.0, 0.0, 0.0, ... °N
Central surface pressure depression	Δp_n	dp(N)	1115.0, 1115.0, ... Pa
Half-width of depression	S_n	rsize(N)	282000.0, ... m
Background surface pressure	$p_0(k_s)$	p_0	101500.0 Pa
Background surface temperature	T_0	T_0	302.15 K
Background surface humidity	q_0	q_0	0.021 kg kg^{-1}
Atmospheric lapse rate	Γ	lapse_rate	7.0e-3 K m^{-1}
Tropopause height	z_t	z_trop	15000.0 m
Stratospheric humidity	q_t	q_trop	1.0e-11 kg kg^{-1}
Vertical scale for p decay	z_p	z_p	7000.0 m
Linear vertical scale for q decay	z_{q1}	z_q1	3000.0 m
Quadratic vertical scale for q decay	z_{q2}	z_q2	8000.0 m
Convergence threshold for z	z_c	z_conv	1.0e-6 m

Table 2: All options currently customizable through the new **&test_case_nml** namelist block (which would be placed in forecast namelists like **input.nml**). The "symbol" field denotes the name of the variable as it appears in mathematically in this documentation. The "code name" field indicates the name paired to the value in the test source code. Default settings are also listed (in Fortran convention), along with units.

3.1 Example Namelists

Namelist snippit for a single vortex just north of the Equator with default settings:

```
&fv_core_nml
       {long list of settings...}
       selected_case = 99
/

&test_case_nml
       num_vortex = 1
       aqua = .true.
       lon = 30.
       lat = 10.
/
```

Namelist block for two vortices in proximity, with one vortex smaller and more intense:

```
&test_case_nml
       num_vortex = 2
       aqua = .true.
       lon = 33.0, 47.0
       lat = 15.0, 5.0
       dp  = 2230., 1100.
       rsize = 177000., 212000.
/
```

Namelist block for four different vortices in an initially "dry" atmosphere:

```
&test_case_nml
       num_vortex = 4
       aqua = .true.
       lon = 10.0, 100.0, 190.0, 280.0
       lat = 10.0, 10.0, 10.0, 10.0
       dp  = 550., 1100., 2200., 2750.
       rsize = 212000., 212000., 212000., 212000.
       q_0 = 1.e-11
/
```

Namelist similar to the previous namelist, except initializing just the first vortex:

```
&test_case_nml
       num_vortex = 1
       aqua = .true.
       lon = 10.0, 100.0, 190.0, 280.0
       lat = 10.0, 10.0, 10.0, 10.0
       dp  = 550., 1100., 2200., 2750.
       rsize = 212000., 212000., 212000., 212000.
       q_0 = 1.e-11
/
```

A Solving for Height

The height field z is required to calculate various necessary fields for this test case. FV3's vertical coordinate is not z, however, and must be computed for the case. Still, with equations (6) and (8), the pressure field can be calculated *without* dependence on z. From the hydrostatic assumption and ideal gas law,

$$\begin{aligned} \frac{\partial p}{\partial z} &= -\frac{gp}{R_d T_v}, \\ \frac{\Delta p}{\Delta z} &\approx -\frac{gp}{R_d T_v}, \end{aligned} \tag{A1}$$

we can determine how the pressure field varies with height (or vice versa), given p and T_v. The Δp and Δz terms are point-differences between p and z.

We can guess the height at some level k, and calculate p and T_v at the guessed height. We can compare the guess of p at level k to the *known* p at that level to form Δp in equation (A1), which becomes minimized when the guessed z is approximately correct. The Δz term represents error in the guess at z, which is also minimized when the guess at z is nearly correct. Mathematically, at some given point in the horizontal (i, j),

$$\Delta z = -\Delta p \frac{R_d T_{v,g}}{g p_g},$$
$$z(k) = z_g + (p_g - p(k)) \frac{R_d T_{v,g}}{g p_g}, \tag{A2}$$

where p_g and $T_{v,g}$ are the guesses of pressure and virtual temperature at $z = z_g$, using equations (7), (11), and (12). The height is considered found in the event that $|\Delta z| < z_c$, where z_c is a small convergence threshold (which can be user-specified, though this is not recommended). Otherwise, another guess is made with z_g set equal to the resulting height from the previous iteration.

For this iterative process, we require a reasonable first guess. Our first guess for $z(i, j, k)$ chosen to be $z(i, j, k-1)$, or the height of the level immediately below k. We are given the height at the lowest level in the atmosphere (set to zero in the case of no terrain), which can be used to begin the process of finding z for the rest of the atmosphere (e.g., layer $k = 1$ uses $z(k = 0)$ to find $z(k = 1)$, which is then used to find $z(k = 2)$).

Lastly, we consider that the height field can be affected by the presence of multiple vortices. To accomodate this, the height field resulting from the effects of all vortices together (the "total" height field) is solved level-by-level, starting at the bottom layer of the atmosphere and iterating upward to the model top. We find the total height field similar to other fields in this idealized case:

$$z(i, j, k) = \sum_{n=1}^{N} W_n(i, j) z_n(i, j, k), \tag{A3}$$

where z_n is the final result of (A2) using p_g and $T_{v,g}$ calculated from vortex n alone. To be clear, the effects of all vortices are included in the known p field and the first guess height for each model level.

B Redesign of FV3's Test Process

Prior to this endeavor, FV3's testing module in the **test_cases.F90** source file was used to run the model with idealized configurations. The original module handles over 40 different test cases of varying model-state involvement, complexity, and purpose. To accomodate all of these cases, the module necessitates a dizzying flood of members (variables and/or functions) specific to each case or group of cases. Some of these cases work only in specific FV3 configurations (such as the choice of grid), including the original DCMIP2016 TC test case that cannot run on global grids due to an overflow/underflow bug. Furthermore, the end user is largely unable to customize test cases without touching the code directly (and forcing recompilation), as the original **&test_case_nml** namelist options provided in **fv_control.F90** are severely limited.

We feel that the process of running idealized tests in FV3 could be streamlined for end users, and modularized for developers to improve interpretation and new case implementation. Our approach to this redesign consists of two basic ideas:

1. The original test module can be split into many seperate modules, each specific to a certain kind of test or group of tests (e.g., a module for a baroclinic wave test seperate from a module for simulating convection).

2. Take advantage of namelist I/O to allow users to customize key test case options *without* touching code or recompiling the model.

To realize this design, the testing process is handled by three "kinds" of module. First, we have the **test_base_mod** module, which acts like a plain data structure with members relevant to the model *in general*, regardless of the test case specified. For instance, **test_base_mod** includes information about the working processor's grid in a parallel environment (as is often the case with FV3), whether the model is hydrostatic and/or adiabatic, and the number of model levels. It also has a member indicating the type of test to run, which is compared

against an enumerated list of cases (each case is given a name-value pair). This module is designed so that all case-specific modules can use it in its entirety.

Second, we have the **test_init_mod** module, which uses **test_base_mod** to direct the control flow during namelist I/O and model initialization to the intended case-specific module(s), so that they can read the namelist and set the model state. This simplifies this redesign's implementation in the rest of FV3—it prevents existing modules in FV3 from having to include a large number of new modules for each test case. Additionally, methods that must be carried out *regardless* of the test are also done in this module.

Lastly, the test-specific module itself, which includes routines and members to initialize (or perhaps even provide forcing for) the model state. In the immediate work, that module is called **multi_vtx_mod**, and its I/O and initialization subroutines are used by **test_init_mod**.

Using this testing design, a developer can easily add and manage their own test case by:

1. Writing their own module code for initializing the model and reading a user namelist for configuration (if desired),
2. Adding a name-value pair to the enumerated list of cases in **test_base_mod**,
3. Including the new case module's I/O and initialization subroutines in **test_init_mod**,
4. Adding calls to these subroutines in the **test_init_mod** module's switch statements, and
5. Adding the new module's source path to the **atmos_cubed_sphere**'s **makefile** and recompiling.

C Installation

Because this code includes a redesign of a complicated module, we recommend making a new directory with its own build of HAFS specifically for idealized cases. This will prevent any mishaps from affecting your existing workflow(s). For our setup, we start by cloning the HAFS_nogsi branch on github from June 2020:

```
mkdir <path_to_HAFS>
cd <path_to_HAFS>
git clone -b feature/hafs_nogsi https://github.com/hafs-community/HAFS.git .
git checkout 2addfe9
git submodule update --init --recursive
```

Three Fortran source codes need to be retrieved:

1. test_control.F90 ,
2. test_init.F90 ,
3. multi_vortex.F90 .

You should not *need* to modify these source files at all for the installation to work. We recommend putting these files in a new "tests" directory:

```
cd <path_to_HAFS>/sorc/hafs_forecast.fd/FV3/atmos_cubed_sphere/tools
mkdir tests
cd tests
cp <path_to_new_source>/test_control.F90 .
cp <path_to_new_source>/test_init.F90 .
cp <path_to_new_source>/multi_vortex.F90 .
```

Next, navigate back to the **atmos_cubed_sphere** directory:

```
cd <path_to_HAFS>/sorc/hafs_forecast.fd/FV3/atmos_cubed_sphere
```

Open the **makefile** . Look for the **SRCS_F90** block near line 34:

```
SRCS_F90 = \
        ./model/a2b_edge.F90                                    \
        ./model/multi_gases.F90                                 \
        ./model/boundary.F90                                    \
        <...>
```

Add paths to the three new source files after all source files from the **tools** subdirectory have been listed in the **SRCS_F90** block:

```
SRCS_F90 = \
        <...>
        ./tools/sim_nc_mod.F90                          \
        ./tools/sorted_index.F90                        \
        ./tools/tests/test_control.F90                  \
        ./tools/tests/test_init.F90                     \
        ./tools/tests/multi_vortex.F90                  \
        ./driver/fvGFS/DYCORE_typedefs.F90              \
        <...>
```

Next, the new modules have to be incorporated to the existing FV3 codebase. To do this, you need to change some lines in **fv_control.F90** and **fv_restart.F90**. If you are using the HAFS version recommended at the start of this appendix, then you can just replace the existing files with the edited files provided:

```
cd model
cp <path_to_new_source>/fv_control.F90 .
cd ../tools
cp <path_to_new_source>/fv_restart.F90 .
```

These modifications disable the old **test_cases_mod**, redirect reading of the **&test_case_nml** namelist to the testing modules, and replace the initialization subroutine call in **fv_restart.F90**. If you are starting from a different version of HAFS than the one recommended at the start of this appendix, the modifications to make to the code should accomplish these changes to the model. After these replacements or modifications have been made, the model can be built. From there, an idealized case can be run manually from the forecast step in HAFS.

References

[Jor58] Charles L Jordan. "Mean soundings for the West Indies area". In: *Journal of Meteorology* 15.1 (1958), pp. 91–97.

[NJ08] Ramachandran D Nair and Christiane Jablonowski. "Moving vortices on the sphere: A test case for horizontal advection problems". In: *Monthly Weather Review* 136.2 (2008), pp. 699–711.

[Ulr+08] Paul A Ulrich et al. *Dynamical Core Model Intercomparison Project (DCMIP2016) Test Case Document*. 2008. URL: `https://github.com/ClimateGlobalChange/DCMIP2016`.

[RJ11a] KA Reed and C Jablonowski. "Impact of physical parameterizations on idealized tropical cyclones in the Community Atmosphere Model". In: *Geophysical Research Letters* 38.4 (2011).

[RJ11b] Kevin A Reed and Christiane Jablonowski. "An analytic vortex initialization technique for idealized tropical cyclone studies in AGCMs". In: *Monthly Weather Review* 139.2 (2011), pp. 689–710.

[RJ11c] Kevin A Reed and Christiane Jablonowski. "Assessing the uncertainty in tropical cyclone simulations in NCAR's Community Atmosphere Model". In: *Journal of Advances in Modeling Earth Systems* 3.3 (2011).

[RJ12] Kevin A Reed and Christiane Jablonowski. "Idealized tropical cyclone simulations of intermediate complexity: A test case for AGCMs". In: *Journal of Advances in Modeling Earth Systems* 4.2 (2012).