

1    **Development of the Upgraded Tangent Linear and Adjoint of the**  
2    **Weather Research and Forecasting (WRF) Model**

3                    XIN ZHANG \*

*National Center for Atmospheric Research, Boulder, Colorado 80307, USA*

4                    XIANG-YU HUANG

*National Center for Atmospheric Research, Boulder, Colorado 80307, USA*

5                    NING PAN

*Fujian Meteorological Bureau, Fuzhou, Fujian, China*

---

\* *Corresponding author address:* Dr. Xin Zhang, NCAR, MMM, P.O. Box 3000, Boulder, CO 80307.

E-mail: xinzhang@ucar.edu

## ABSTRACT

The authors propose a new technique for parallelizations of tangent linear and adjoint codes, which were applied in the redevelopment for the Weather Research and Forecasting (WRF) Model with its Advanced Research WRF dynamic core using the automatic differentiation engine. The tangent linear and adjoint codes of the WRF model (WRFPLUS) now has the following improvements:

- A complete check interface ensures that developers write the tangent linear and adjoint codes in accuracy with ease and efficiency.
- To parallelize the WRFPLUS model, the authors adopted a new technique based on the nature of duality that existed among Message Passing Interface communication routines. The Registry in the WRF model was extended to automatically generate the tangent linear and adjoint codes of the required communication operations. This approach dramatically speeds up the software development cycle of the parallel tangent linear and adjoint codes and leads to improved parallel efficiency.
- Module interfaces were constructed for coupling tangent linear and adjoint codes of the WRF model with applications such as four-dimensional variational data assimilation, forecast sensitivity to observation and other.

# 1. Introduction

During the past two decades, the use of the adjoint technique in meteorology and oceanography rapidly increased. The adjoint model is a powerful tool in many applications, such as data assimilation, parameter estimation, and sensitivity analysis. (Errico and Vukicevic (1992); Errico (1997); Rabier et al. (1996); Langland et al. (1999); Li et al. (1999); Xiao et al. (2002); Xiao et al. (2008); Kleist and Morgan (2005a); Kleist and Morgan (2005b))

The Weather Research and Forecasting (WRF) Modeling System (Skamarock et al. (2005)) is a multi-agency effort to provide a next-generation mesoscale forecast model and data assimilation system that will advance both the understanding and the prediction of mesoscale weather as well as accelerate the transfer of research advances into operations. The WRF model was designed to be an efficient massively parallel computing code to take advantage of advanced high-performance computing systems. The code, which can be configured for both research and operations, also offers numerous physics options. The WRF model is maintained and supported as a community model to facilitate wide use internationally, for research, operations, and teaching. The model is suitable for a broad span of applications across scales ranging from large-eddy to global simulations. Such applications include real-time numerical weather prediction, data assimilation development and studies, parameterized-physics research, regional climate simulations, air-quality modeling, atmosphere-ocean coupling, and idealized simulations. As of this writing, the WRF model is in operational and research use around the world, and the number of registered WRF users exceeds 20000.

The first version of the adiabatic WRF tangent linear model (TLM) and adjoint model (ADM) system (WAMS) was developed by National Center for Atmospheric Research (NCAR) around 2007 (Xiao et al. (2008)). The WAMS is used in the adjoint sensitivity analysis (Xiao et al. (2008)) and Four-Dimensional Variational Data Assimilation (4D-Var) (Huang et al. (2009)). It took us more than one year to complete the parallelization of the WAMS by following the traditional incremental and iterative parallelization procedures, which include

algorithm analysis, identifying the time-consuming hotspot, data dependency analysis, locating the halo exchange and manually writing the communication routines for data exchanging. In the past few years, due to the tremendous time required for hand-coding parallelization, the WAMS has failed to repeat the rapid development pattern of the WRF model and the WRF Data Assimilation System (WRFDA)(Barker et al. (2012)). The growing gap between the WAMS and the WRF/WRFDA makes the WAMS inconvenient to use with other systems. One example was that the WRF 4D-Var system could not read the WRF boundary conditions of later versions because the WAMS was left behind and it does not include the upgrades of the boundary condition data structure in the WRF. Furthermore, because the WAMS uses the disk input/output (I/O) for storing/reading basic states trajectory and exchanging data among the FWM, TLM and ADM, the parallel efficiency is unsatisfactory on modern high-performance computers with distributed memory parallelization, especially for 4D-Var applications.

Encouraged by the rapid developments of 4D-Var, cloud analysis, forecast sensitivity to observations and chemistry data assimilation, we redeveloped the WRF TLM and ADM (called WRFPLUS) based on the latest repository WRF. Compared with the WAMS developed by Xiao et al. (2008) and Huang et al. (2009), the new system is an all-in-one system which includes the WRF full-physics forward model (FWM), TLM and ADM; This system also includes the tangent linear check and adjoint check procedures. A set of module interfaces was developed for easily coupling the WRFPLUS model with other systems such as data assimilation and adjoint sensitivity applications. An new approach was applied to develop the parallel code that dramatically reduces the software development cycle of the parallel TLM and ADM; and the derived parallel TLM and ADM have better parallel efficiency compared to the FWM.

The purpose of this paper is to describe the technical aspects of the newly developed WRFPLUS model. A brief introduction of the development of the WRFPLUS model is presented in section 2, followed by a demonstration of the linearity and adjoint tests in

section 3. A detailed description of parallelization strategy for tangent linear and adjoint codes, with the demonstrated parallel performance are in section 4. Section 5 introduces the module interfaces constructed in the WRFPLUS model for coupling purposes, such as in WRF 4D-Var. Concluding remarks appear in section 6.

## 2. Description of the WRF tangent linear and adjoint models

After the release of the WRF model version 3.2, we started to use TAPENADE (Hascoët and Pascual (2004)) to redevelop the TLM and the ADM of the WRF Advanced Research WRF (ARW) core based on the latest WRF model. The development of WRFPLUS model follows the same three phases proposed by Xiao et al. (2008). First, numerical experiments were conducted to make sure that the adiabatic version of the WRF model with simplified physics parameterization routines can produce the major features that the full-physics model does. Second, the TLM and its ADM were generated by TAPENADE and modified manually whenever necessary. The third step was to verify the correctness of the TLM and the ADM.

TAPENADE is a source-to-source automatic differentiation (AD) tool for programs written in Fortran 77-95, i.e., TAPENADE generates a TLM or an ADM from the source code of a given model. Like other AD tools, TAPENADE struggles with such complicated codes as WRFs third-order Runge-Kutta large timesteps and small acoustic timesteps (Xiao et al. (2008)). Checking and improving TAPENADE-generated code can necessitate manual intervention (i) to represent first-order physical effects on the model evolution, (ii) to minimize code length so that developing its adjoint is simple, and (iii) to allow the code to run quickly in the iterations required to lower costs. We also adopted three simplified physics packages that have maximum impact on a forecast compared to a no-physics model: surface friction, cumulus parameterization, and large-scale condensation. These were developed by Jimmy Dudhia (personal communication) and are similar to the physics packages in the WAMS (Xiao

102 et al. (2008)).

### 103 3. Linearity test and adjoint test

104 Testing TLM consistency with FWM as well as ADM consistency with TLM before either  
 105 is used in any real application is essential (Vukicevic (1991); Errico and Vukicevic (1992);  
 106 Gilmour et al. (2001)). We developed the tangent linear and adjoint check procedures  
 107 following Navon et al. (1992).

108 Let  $\mathbf{f}(\mathbf{x})$ ,  $g\mathbf{f}(\mathbf{x}, g\mathbf{x})$  and  $a\mathbf{f}(\mathbf{x}, a\mathbf{x})$  denote a FWM, a TLM and an ADM, respectively,  
 109 where  $\mathbf{x}$ ,  $g\mathbf{x}$  and  $a\mathbf{x}$  is the column vector of model state variables, perturbations of state  
 110 variables and adjoint of state variables, respectively. The correctness of the TLM can be  
 111 tested as:

$$\Phi(\lambda) = \frac{\|\mathbf{f}(\mathbf{x} + \lambda g\mathbf{x}) - \mathbf{f}(\mathbf{x})\|}{\|g\mathbf{f}(\mathbf{x}, \lambda g\mathbf{x})\|}, \lim_{\lambda \rightarrow 0} \Phi(\lambda) = 1 \quad (1)$$

112 where  $\|\cdot\|$  denotes the norm of the vector. The adjoint relation is tested by :

$$\langle g\mathbf{f}(\mathbf{x}, g\mathbf{x}), g\mathbf{f}(\mathbf{x}, g\mathbf{x}) \rangle = \langle a\mathbf{f}(\mathbf{x}, g\mathbf{f}(\mathbf{x}, g\mathbf{x})), g\mathbf{x} \rangle \quad (2)$$

113 If the tangent linear and adjoint codes are correct, the above two relations should hold up to  
 114 the machine accuracy. Because different model variables have different magnitudes, we also  
 115 designed the capability to perform checks on individual variables separately. We performed  
 116 the tangent linear and adjoint checks with the test case being integrated up to 24h. We  
 117 sequentially reduced the initial perturbations by a factor ( $\lambda$ ) of 10 and repeatedly calculated  
 118  $\Phi(\lambda)$  in Eqs. (1) on NCAR's IBM machine with 64-bit precision. Table 1 shows the value of  
 119  $\Phi(\lambda)$  from the tangent linear forecasts and the differences between two nonlinear forecasts  
 120 over the whole domain. This indicated that the tangent linear forecast approximates the  
 121 difference between two nonlinear forecasts as the initial perturbations decrease and approach  
 122 zero.

123 In the adjoint relationship, the left-hand side (lhs) involves only the tangent linear code,

124 while the right-hand side (rhs) involves the adjoint code. If lhs and rhs have the same  
 125 value with machine accuracy, the adjoint code is correct compared with the tangent linear  
 126 code. Using the same test case with 24h integration, lhs and rhs for the test case are  
 127 0.14182720729878E+14 and 0.14182720729883E+14, respectively. This indicates that the  
 128 adjoint code is correct.

## 129 4. Parallelization of the WRFPLUS model

130 TAPENADE has few problems in generating tangent linear and adjoint codes from the  
 131 sequential forward codes. However, it cannot derive such codes of parallel communication  
 132 routines inside a parallel forward model. In most of the atmospheric and oceanic models, the  
 133 communication routines to parallelize finite difference algorithms are linear operators; hence  
 134 as matrices, the adjoint is simply the transpose, which is the dual operator, and the tangent  
 135 linear is the original linear operator acting on the perturbations (see Cheng (2006) and  
 136 Utke et al. (2009)). With the proper coding structure and available parallel communication  
 137 routines in forward codes, it is straightforward to write communication routines for tangent  
 138 linear models. We need only use the same parallel communication templates in forward codes  
 139 and add the corresponding perturbation variables. In the adjoint code, the data flow of the  
 140 original program needs to be reversed and any communication needs to be reversed as well.  
 141 Due to the duality between *MPI\_SEND* and *MPI\_RECV* calls, in transforming FWM to  
 142 ADM, we replace *MPI\_SEND* calls with *MPI\_RECV*, and vice versa. Please note that  
 143 it is the adjoint variable which is subject to being exchanged in the ADM. Figures 1(a) and  
 144 1(b) could be helpful to understand the data flow of communication in the FWM and the  
 145 ADM, respectively. In the FWM, the variable  $U$  in the ghost region of processor P1 will be  
 146 overwritten by the value of  $U$  received from processor P0. In the ADM, the communication  
 147 needs to be reversed. The adjoint variable  $a.U$  in the ghost region of P1 will be sent to P0  
 148 and added to the value of  $a.U$  of P0; then the  $a.U$  in the ghost region of P1 will be set to

149 zero.

150 In the WRF model, hundreds of thousands of lines of code are automatically generated  
151 from a user-edited table called the Registry (Michalakes and Schaffer (2004)). The Registry  
152 provides a high-level single point of control over the fundamental structure of the model  
153 data. It contains lists describing state data fields and their attributes: dimensionality,  
154 binding to particular solvers, association with WRF I/O streams, communication operations,  
155 and run time configuration options (namelist elements and their bindings to model control  
156 structures). Adding or modifying a state variable to WRF involves modifying a single line  
157 of a single file; this single change is then automatically propagated to scores of locations in  
158 the source code the next time the code is compiled.

159 Halo entries in the Registry define communication operations in the model. Halo entries  
160 specify halo updates around a patch horizontally. A typical halo entry is :

halo	HALO_EM_C	dyn_em	4:u_2,v_2
------	-----------	--------	-----------

161 The first field is the keyword "halo". The second entry –"HALO\_EM\_C", which is the given  
162 name of the halo exchange template, is used in the model to refer to the communication  
163 operation being defined. The third entry denotes the associated solver, and the fourth entry  
164 is a list of information about the operation. This example specifies that 4 points (one cell each  
165 in north, south, east, and west directions, respectively) of the stencil are used in updating  
166 the state arrays for fields  $u_2$  and  $v_2$  across the processors. During compilation, the WRF  
167 Registry automatically generates a code segment based on this halo entry:

...
CALL HALO_EM_C_sub ( grid, local_communicator, ...)
...

168 The code snippet is included in the places where the communications are needed. At the  
169 same time, the Registry also generates the subroutine "HALO\_EM\_C\_sub":



```

...
! for Y direction
CALL RSL_LITE_PACK ( local_communicator, grid%u_2, ...,0,...
CALL RSL_LITE_PACK ( local_communicator, grid%v_2, ...,0,...
CALL RSL_LITE_EXCH_Y
CALL RSL_LITE_PACK ( local_communicator, grid%u_2, ...,1,...
CALL RSL_LITE_PACK ( local_communicator, grid%v_2, ...,1,...
...
! for X direction
CALL RSL_LITE_PACK ( local_communicator, grid%u_2, ...,0,...
CALL RSL_LITE_PACK ( local_communicator, grid%v_2, ...,0,...
CALL RSL_LITE_EXCH_X
CALL RSL_LITE_PACK ( local_communicator, grid%u_2, ...,1,...
CALL RSL_LITE_PACK ( local_communicator, grid%v_2, ...,1,...
...

```

170 In the above code segment, the outgoing slices of  $u_2$  and  $v_2$  for Y direction (south-north)  
171 exchanging are packed by "RSL\_LITE\_PACK" into a local contiguous memory region. There-  
172 fore, one call of "RSL\_LITE\_EXCH\_Y" is able to complete the data exchanges in the south-  
173 north direction. Once every processor receives the incoming data, "RSL\_LITE\_PACK" will  
174 be called again with different arguments to unpack the data to the ghost area position. The  
175 similar operations are performed in the X direction (east-west) as well.

176 The forward communication codes show an efficient chain of established relationships.  
177 Therefore, perturbing and adjointing the code is simple. However, it is an error-prone and  
178 time-consuming task to manually write all tangent linear and adjoint codes of communication  
179 subroutines. Since the WRF Registry is able to generate halo exchange routines, there is  
180 the possibility of letting the Registry generate the tangent linear and adjoint codes of halo

181 exchanges too. To enable the WRF Registry to generate the corresponding tangent linear  
 182 and adjoint communication codes automatically, we modified the Registry and added a new  
 183 entry "halo\_anta" as:

halo_anta	HALO_EM_C	dyn_em	4:u_2,v_2
-----------	-----------	--------	-----------

184 With this new entry, the Registry will not only generate "HALO\_EM\_C\_sub", but also  
 185 generate "HALO\_EM\_C\_TL\_sub" for tangent linear codes and "HALO\_EM\_C\_AD\_sub" for  
 186 adjoint codes. The subroutine "HALO\_EM\_C\_TL\_sub" looks like:

```

...
! for Y direction
CALL RSL_LITE_PACK ( local_communicator, grid%u_2, ...,0,...
CALL RSL_LITE_PACK ( local_communicator, grid%v_2, ...,0,...
CALL RSL_LITE_PACK ( local_communicator, grid%g_u_2, ...,0,...
CALL RSL_LITE_PACK ( local_communicator, grid%g_v_2, ...,0,...
CALL RSL_LITE_EXCH_Y
CALL RSL_LITE_PACK ( local_communicator, grid%u_2, ...,1,...
CALL RSL_LITE_PACK ( local_communicator, grid%v_2, ...,1,...
CALL RSL_LITE_PACK ( local_communicator, grid%g_u_2, ...,1,...
CALL RSL_LITE_PACK ( local_communicator, grid%g_v_2, ...,1,...
...
! for X direction

```

187 The TLM has exactly the same exchange stencil and data flow as the FWM; however, in  
 188 addition to the basic state fields ( $u_2$  and  $v_2$ ), the perturbation fields ( $g_{u_2}$  and  $g_{v_2}$ )  
 189 exchanged as well, which is required by the TLM.

190 The adjoint code "HALO\_EM\_C\_AD\_sub" looks like:

...
-----

```

! for Y direction
CALL RSL_LITE_PACK_AD ( local_communicator, grid%a_u_2, ...,0,...
CALL RSL_LITE_PACK_AD ( local_communicator, grid%a_v_2, ...,0,...
CALL RSL_LITE_EXCH_Y
CALL RSL_LITE_PACK_AD ( local_communicator, grid%a_u_2, ...,1,...
CALL RSL_LITE_PACK_AD ( local_communicator, grid%a_v_2, ...,1,...
...
! for X direction

```

191 The ADM also has the same exchange stencil as the FWM and the TLM, but the data flow  
 192 is reversed. On each processor, the entire patch of basic state variables (including halo) is  
 193 stored in the memory stack during the forward recomputation stage and will be restored from  
 194 the memory stack during the adjoint calculation. Therefore, in adjoint communication codes,  
 195 only the adjoint variables ( $a\_u\_2$  and  $a\_v\_2$ ) need to be exchanged and the new subroutines  
 196 "RSL\_LITE\_PACK\_AD" will pack the data slices from where we receive data in the FWM  
 197 and the TLM for sending and unpack the received data to the slices where we send out data  
 198 in the FWM and the TLM.

199 With the upgraded WRF Registry, all tangent linear and adjoint communication routines  
 200 are generated automatically during compilation. To manually insert the tangent linear and  
 201 adjoint communication interfaces into the TLM and the ADM is straightforward. In the  
 202 TLM, the tangent linear communication routines are inserted into the same locations where  
 203 the forward communication routines reside in the FWM. In the ADM, the calling sequence of  
 204 the adjoint communication routines is the reverse of that in the FWM and the TLM. From  
 205 the code snippets presented above, we found that because both basic state variables and  
 206 perturbation variables are packed together, although the amount of data to communicate  
 207 is doubled in the TLM, the communication overhead is kept the same as in the FWM.  
 208 This is highly desirable for a modern distributed memory parallel computer system with

high bandwidth. In the ADM, in general, there are two stages: the first is the forward  
recomputation part, which propagates the basic states within one timestep and has the same  
communication overhead and amount of data to communicate with the FWM; The second  
is the adjoint backward part, which has the same communication overhead and amount as  
the first. Therefore, the total communication overhead and amount of data to communicate  
are doubled in the ADM. However, due to the fact that the adjoint code may has three to  
four times the amount of computation code than the forward code and the computations  
to communications ratio is still large, the communication overhead is not substantial and  
should not impact the parallel scalability of the ADM.

## 5. Parallel performance

To demonstrate the parallel efficiency of the WRFPLUS model, we prepared the initial  
condition and boundary conditions for a 15km-resolution domain (not shown) with domain  
center at  $32.6^{\circ}N$ ,  $110^{\circ}E$ . There are  $665 \times 383$  grid points in the horizontal direction and  
45 levels in the vertical direction; the timestep is 90s. We tested this case on NCAR's two  
supercomputers: Lynx and Bluefire. Lynx is a single-cabinet Cray XT5m supercomputer,  
comprised of 76 compute nodes, each with 12 processors on 2 hex-core AMD 2.2 GHz Opteron  
chips, with a total of 912 processors. Each Lynx computer node has 16 gigabytes of memory,  
for 1.33 gigabytes per processor, and totaling 1.216 terabytes of memory in the system.  
Bluefire is an IBM clustered Symmetric MultiProcessing (SMP) system comprised of 4,096  
Power 6 processors. The 4,096 processors are deployed in 128 nodes, each containing 32  
processors. Nodes are interconnected with an InfiniBand switch for parallel processing using  
MPI. We used the default compilation options and the default processors topology calculated  
by the WRF model. We did not do any further optimization to get the best performance.  
Therefore, the following results do not reflect the best performance possible on a specific  
supercomputer.

We ran the WRFPLUS model on 16, 32, 64, 128, 256, 512, 1024 and 2048 processors of Bluefire and measured the parallel performance. Figure 2 shows the results for the average wallclock time for one-timestep integration (Fig.2(a)) and speedup for the FWM, TLM and ADM, respectively (Fig.2(b)). Please note that the timing results for the FWM are different from the standard WRF run, which is compiled with 4 bytes-long real size. In the WRFPLUS, the FWM is compiled with 8 bytes long real size. In general, due to the higher precision requirement in the WRFPLUS, the actual computational performance of the FWM is slower than the standard WRF. Speedup for  $N$  processors was calculated as the wallclock time using 16 processors divided by the wallclock time using  $N$  processors. The computing times for all models are considerably reduced with an increased number of processors up to 2048. In general, the TLM has a better speedup than the FWM and the ADM has a slightly better or comparable speedup than the FWM. The results confirm the successful implementation of the new parallel approach.

We also ran the WRFPLUS model on 16, 32, 64, 128, 256 and 512 processors of Lynx and measured the parallel performance. Figure 3 shows the results for the average wallclock time for one-timestep integration (Fig.3(a)) and speedup (Fig.3(b)). We drew similar conclusions and again confirmed the high efficiency of the parallelization strategy of the WRFPLUS.

## 6. Implementation of an inline coupling interface in the WRFPLUS model

One of the motivations to upgrade the WRF TLM and ADM is to improve the computational performance of the WRF 4D-Var. The old WRF 4D-Var system contains a two-way coupling between the WRFDA and the WAMS (Huang et al. (2009)) exchanges information via disk files and is a multiple program multiple data (MPMD) system. During the coupling, the exchanged data are written to disk and a signal file is prepared to inform the other component that data is ready to be read. Exchange of a field between the WRFDA

and the WAMS consists of gathering and scattering operations across the processors via disk files, which is very inefficient on modern distributed-memory supercomputers with large number of processor. This limits the number of processors that can be used for high-resolution modeling.

Since the WRFDA and the WRFPLUS share the same software infrastructure including parallelization, field definition, I/O, Registry etc., it is straightforward to couple the WRFDA and the WRFPLUS to a single executable 4D-Var system, in which all information (basic states, perturbation and adjoint forcing) from the WRFPLUS is passed as arguments to the coupling interfaces and the WRFDA fetches the data from the coupling interfaces instead of disk files.

For this purpose, three major developments were needed:

- i. Enable the WRFPLUS model to be callable from the WRFDA with a simple application programming interface consisting of:
  - (a) Initialize the WRFPLUS model.
  - (b) Advance one of the WRFPLUS model components (FWM, TLM and ADM) .
  - (c) Finalize the WRFPLUS model.
- ii. Develop a set of regridding routines that can interpolate data on the WRFPLUS grid to the WRFDA grid (and vice-versa), which can be called by the WRFDA in full MPI parallel mode.
- iii. Modify the WRFDA to allow it to call the WRFPLUS with forcing data from the WRFDA and retrieve field data from the WRFPLUS (e.g. gradients).

In this paper we discuss only the first development, the other two developments will be introduced in a separate paper.

The WRF model already has a well-defined routine that advances the integration, which makes calling it from an external model straightforward. New initialization and finalization

284 routines had to be coded mostly to deal with the TLM and the ADM. A namelist option  
 285 *dyn\_opt* was borrowed to allow the WRFPLUS to decide which model will be advanced.  
 286 *dyn\_opt* = 2 will activate the FWM, *dyn\_opt* = 202 will activate the TLM and *dyn\_opt* = 302  
 287 will activate the ADM.

288 The fully implemented interfaces as seen from the WRFDA point of view looks like this:

- **Components routines** : The interfaces to activate forward, tangent linear and adjoint model
  - **wrf\_run** : Interface to run forward (nonlinear) model
  - **wrf\_run\_tl** : Interface to run tangent linear model
  - **wrf\_run\_ad** : Interface to run adjoint model
- **Data exchange routines** : The interfaces to exchange data between the WRFDA and the WRFPLUS
  - **read\_xtraj** : Read the trajectories from the FWM integration
  - **save\_xtraj** : Save the trajectories from the FWM integration
  - **read\_tl\_pert** : Read initial perturbation for the TLM integration
  - **save\_tl\_pert** : Save trajectories of perturbation from the TLM integration.
  - **read\_ad\_forcing** : Read adjoint forcing for the ADM integration
  - **save\_ad\_forcing** : Save initial adjoint forcing from the ADM integration

290 These interfaces are written not only for the coupling with the WRFDA but are designed for  
 291 general coupling purposes. In addition to the coupling of the WRFDA and the WRFPLUS  
 292 to construct the WRF 4D-Var, we successfully coupled the WRFPLUS with the community  
 293 GSI to construct a GSI-based WRF 4D-Var (Zhang and Huang (2013)). Figure 4 shows the  
 294 parallel performance of WRF 4D-Var run with 15km-resolution Continental U.S. (CONUS)  
 295 domain (not shown), This domain has 450X450 horizontal grids and 51 vertical levels. The

assimilation window is 6 hours and the integration timestep is 90s. Only GTS conventional data is assimilated. The WRF 4D-Var shows impressive scalability, i.e. with the addition of more processors, the total performance of the WRF 4D-Var increased accordingly. The implementation and coupling work quite well; it has already replaced the old WRF 4D-Var system since the V3.4 release of WRF 4D-Var. Beside the performance benefit, there are other advantages compared to the old WRF 4D-Var system. The execution of the new 4D-Var is simpler than before because it is unnecessary to launch a multiple program multiple data (MPMD) collection of different executables.

## 7. Conclusion

In this paper we describe the implemented technique of the new tangent linear and adjoint codes of the WRF ARW core and demonstrated how, using TAPANADE (a free AD tool), the WRFPLUS was developed and carried forward into later versions. Compared to the WAMS, the new WRFPLUS has the following improvements: 1) The tangent linear and adjoint codes are maintained to be consistent with the latest WRF changes; 2) Parallelization strategy and efficiency are enhanced; 3) Complete tangent-linear and adjoint checks ensure the accuracy of the existing codes and newly developed codes; and 4) Module interfaces for coupling were constructed in the WRFPLUS that led to a single executable WRF 4D-Var system with efficient parallel performance and scalability.



## 314 8. Figures and tables

315 *a. Figures*

316 *b. Tables*

317 *Acknowledgments.*

318 The National Center for Atmospheric Research is sponsored by the National Science  
319 Foundation. This work is supported by the Air Force Weather Agency. The WRFPLUS  
320 system benefitted greatly from close collaboration with Dr. Qiang Chen and his Adjoint  
321 Code Generator. We thank Dong-Kyou Lee and Gyu-Ho Lim of Seoul National University  
322 for their comments on the manuscript and generous support through Korea-USA Weather  
323 and Climate Center. We thank Fuqing Zhang and Jon Poterjoy of Penn State University for  
324 their valuable suggestions and comments on the manuscript. Mary Golden helped to edit  
325 the manuscript.

## REFERENCES

- 328 Barker, D., et al., 2012: The weather research and forecasting (wrf) model’s community  
 329 variational/ensemble data assimilation system: Wrfda. *Bull. Amer. Meteor. Soc.*, **93**, 831–  
 330 843.
- 331 Cheng, B., 2006: A duality between forward and adjoint MPI communication routines.  
 332 *Computational Methods in Science and Technology*, Polish Academy of Sciences, 23–24.
- 333 Errico, R., 1997: What is an adjoint model? *BAMS*, **78**, 2577–2591.
- 334 Errico, R. M. and T. Vukicevic, 1992: Sensitivity analysis using of the psu-near mesoscale  
 335 model. *Mon. Wea. Rev.*, **120**, 1644–1660.
- 336 Gilmour, I., L. Smith, and R. Buizza, 2001: Linear regime duration: Is 24 hours a long time  
 337 in synoptic weather forecasting? *J. Atmos. Sci.*, **58**, 3525–3539.
- 338 Hascoët, L. and V. Pascual, 2004: Tapenade 2.1 user’s guide. Technical Report 0300, INRIA.  
 339 URL <http://www.inria.fr/rrrt/rt-0300.html>.
- 340 Huang, X.-Y., et al., 2009: Four-dimensional variational data assimilation for wrf: Formu-  
 341 lation and preliminary results. *Mon. Wea. Rev.*, **137**, 299–314.
- 342 Kleist, D. T. and M. C. Morgan, 2005a: Interpretation of the stucture and evolution of  
 343 adjoint-derived forecast sensitivity gradients. *Mon. Wea. Rev.*, **133**, 466–484.
- 344 Kleist, D. T. and M. C. Morgan, 2005b: Application of adjoint-derived forecast sensitivities  
 345 to the 2425 january 2000 u.s. east coast snowstorm. *Mon. Wea. Rev.*, **133**, 31483175.
- 346 Langland, R. H., R. Gelaro, G. D. Rohaly, and M. A. Shapiro, 1999: Target observations  
 347 in fastex: Adjoint based targeting procedure and data impact experiments in iop17 and  
 348 iop18. *Quart. J. Roy. Meteor. Soc.*, **125**, 3241–3270.

349 Li, Z., A. Barcilon, and I. M. Navon, 1999: Study of block onset using sensitivity perturba-  
350 tions in climatological flows. *Mon. Wea. Rev.*, **127**, 879–900.

351 Michalakes, J. and D. Schaffer, 2004: Wrf tiger team documentation: The registry. Techni-  
352 cal report, INRIA. URL [http://www.mmm.ucar.edu/wrf/WG2/software\\_2.0/registry\\_](http://www.mmm.ucar.edu/wrf/WG2/software_2.0/registry_schaffer.pdf)  
353 [schaffer.pdf](http://www.mmm.ucar.edu/wrf/WG2/software_2.0/registry_schaffer.pdf).

354 Navon, I. M., X. Zou, J. Derber, and J. Sela, 1992: Variational data assimilation with an  
355 adiabatic version of the nmc spectral model. *Mon. Wea. Rev.*, **120**, 1433–1446.

356 Rabier, F., E. Klinker, P. Courtier, and A. Hollingsworth, 1996: Sensitivity of forecast errors  
357 to initial conditions. *Quart. J. Roy. Meteor. Soc.*, **122**, 121–150.

358 Skamarock, W. C., J. B. Klemp, J. Dudhia, D. O. Gill, D. M. Barker, W. Wang, and J. G.  
359 Powers, 2005: A description of the advanced research wrf version 2. Technical report,  
360 NCAR Tech. Note NCAR/TN-468+STR, 100 pp. [Available from UCAR communications,  
361 P. O. Box 3000, Boulder, Co 80307-3000.].

362 Utke, J., L. Hascoët, P. Heimbach, C. Hill, P. Hovland, and U. Naumann, 2009: Toward  
363 adjointable mpi. *Proceedings of the 10th IEEE International Workshop on Parallel and*  
364 *Distributed Scientific and Engineering, PDSEC’09*.

365 Vukicevic, T., 1991: Nonlinear and linear evolution of initial forecast errors. *Mon. Wea.*  
366 *Rev.*, **119**, 1602–1611.

367 Xiao, Q., X. Zou, M. Ponceta, M. A. Shapiro, and C. S. Velden, 2002: Impact of gms-5 and  
368 goes-9 satellite-derived winds on the prediction of a norpex extratropical cyclone. *Mon.*  
369 *Wea. Rev.*, **130**, 507–528.

370 Xiao, Q., et al., 2008: Application of an adiabatic wrf adjoint to the investigation of the  
371 may 2004 mcmurdo, antarctica, severe wind event. *Mon. Wea. Rev.*, **136**, 3696–3713.

372 Zhang, X. and X.-Y. Huang, 2013: Development of gsi-based wrf 4d-var system. *93rd Amer-*  
373 *ican Meteorological Society Annual Meeting*, Austin, TX, USA, 000–000.

## List of Tables

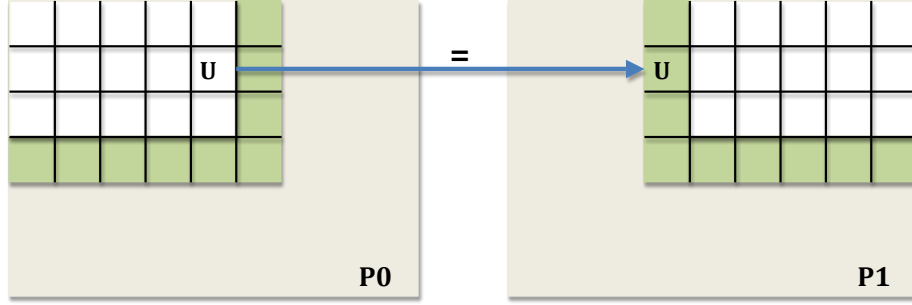
- |   |  |    |
|---|--|----|
| 1 | Ratio of norms between the tangent linear forecasts and the differences of the       |    |
|   | two nonlinear model forecasts at 24 h. The norm is defined as the summation          |    |
|   | of the squares of all variables (perturbations of tangent linear model and           |    |
|   | difference of two nonlinear models) over the whole domain at 24 h. Here $\lambda$ is |    |
|   | the perturbation scaling factors of the initial perturbation.                        | 21 |

TABLE 1. Ratio of norms between the tangent linear forecasts and the differences of the two nonlinear model forecasts at 24 h. The norm is defined as the summation of the squares of all variables (perturbations of tangent linear model and difference of two nonlinear models) over the whole domain at 24 h. Here  $\lambda$  is the perturbation scaling factors of the initial perturbation.

$\lambda$	<i>Ratio</i>
.1000E+0	0.10114281198481E+01
.1000E-01	0.10008545240448E+01
.1000E-02	0.10000832484054E+01
.1000E-03	0.10000095806229E+01
.1000E-04	0.10000007503957E+01
.1000E-05	0.10000001743469E+01
.1000E-06	0.10000000344215E+01
.1000E-07	0.99999998551913E+00
.1000E-08	0.10000001453468E+01
.1000E-09	0.10000007302081E+01
.1000E-10	0.10000775631370E+01

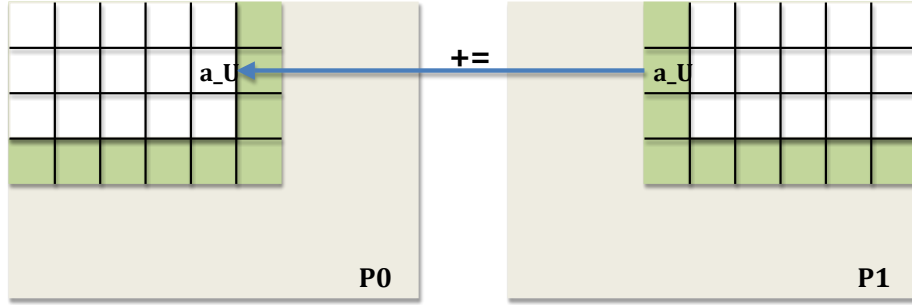
## List of Figures

1	Schematic diagram of the halo communication in the FWM and the ADM. Gray area denotes the entire model domain; Green zone denotes the ghost area; white zone is the computational patch for each processor; $U$ denotes the basic state variable in the FWM and $a.U$ denotes the adjoint variable in the ADM.	23
2	Parallel performance for one-timestep integration on Bluefire	24
3	Parallel performance for one-timestep integration on Lynx	25
4	Parallel performance for 4D-Var with 5 iterations on Bluefire	26



$$U(P1) = U(P0)$$

(a) Halo exchange between two neighbor processors in forward model

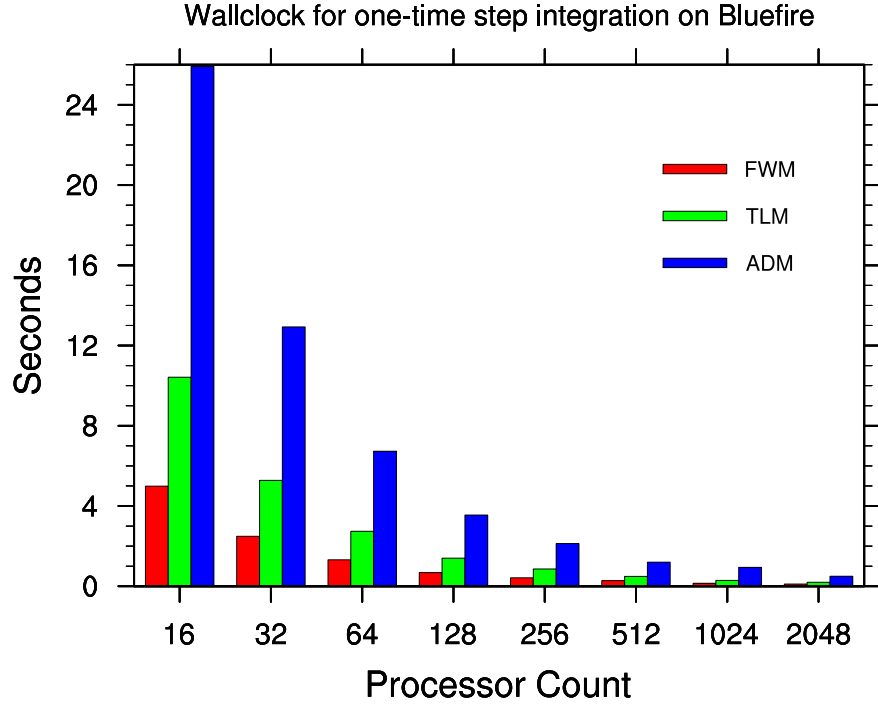


$$\begin{aligned} a\_U(P0) &= a\_U(P0) + a\_U(P1) \\ a\_U(P1) &= 0.0 \end{aligned}$$

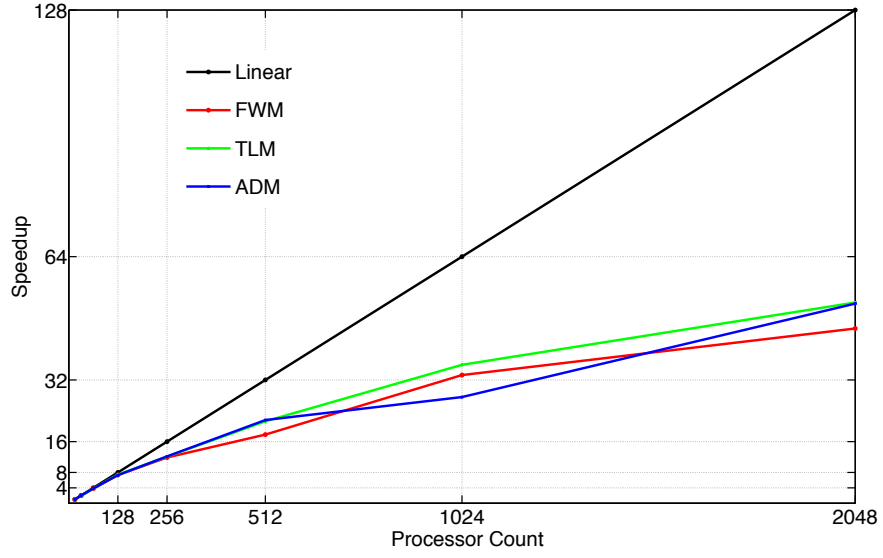
(b) Adjoint of halo exchange between two neighbor processors in adjoint model

FIG. 1. Schematic diagram of the halo communication in the FWM and the ADM. Gray area denotes the entire model domain; Green zone denotes the ghost area; white zone is the computational patch for each processor;  $U$  denotes the basic state variable in the FWM and  $a\_U$  denotes the adjoint variable in the ADM.



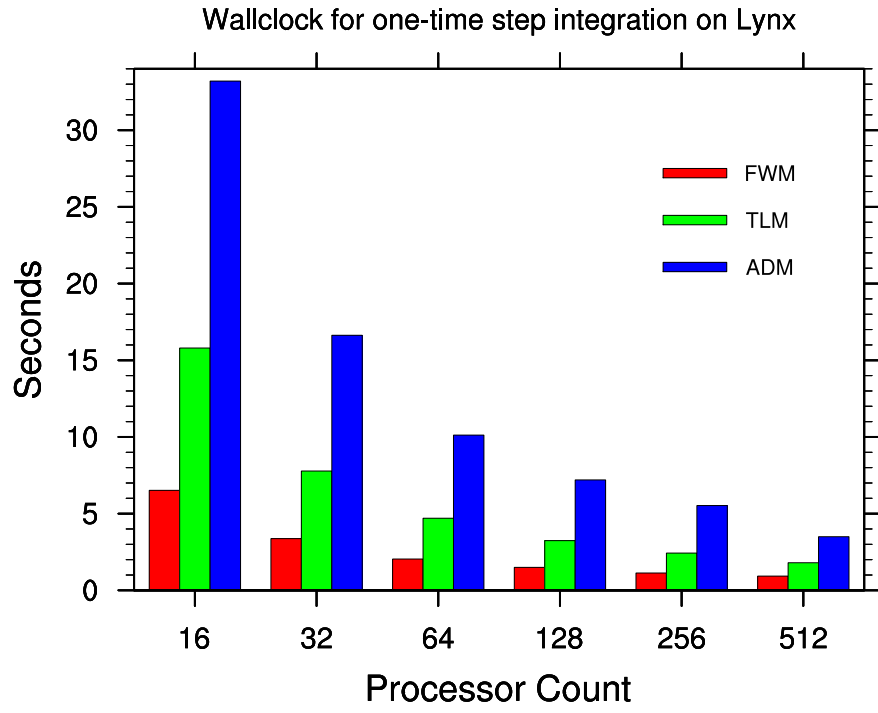


(a) Wallclock time for one-timestep integration

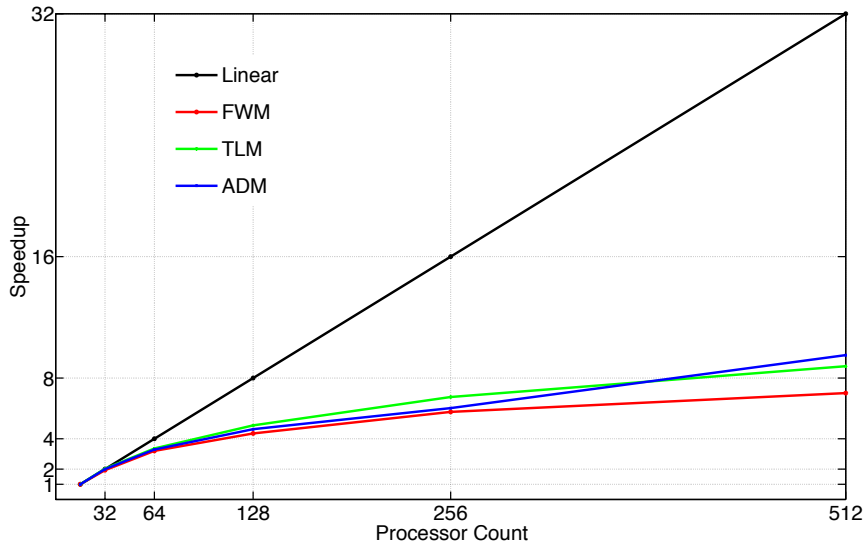


(b) Parallel speedup

FIG. 2. Parallel performance for one-timestep integration on Bluefire



(a) Wallclock time for one-timestep integration



(b) Parallel speedup

FIG. 3. Parallel performance for one-timestep integration on Lynx

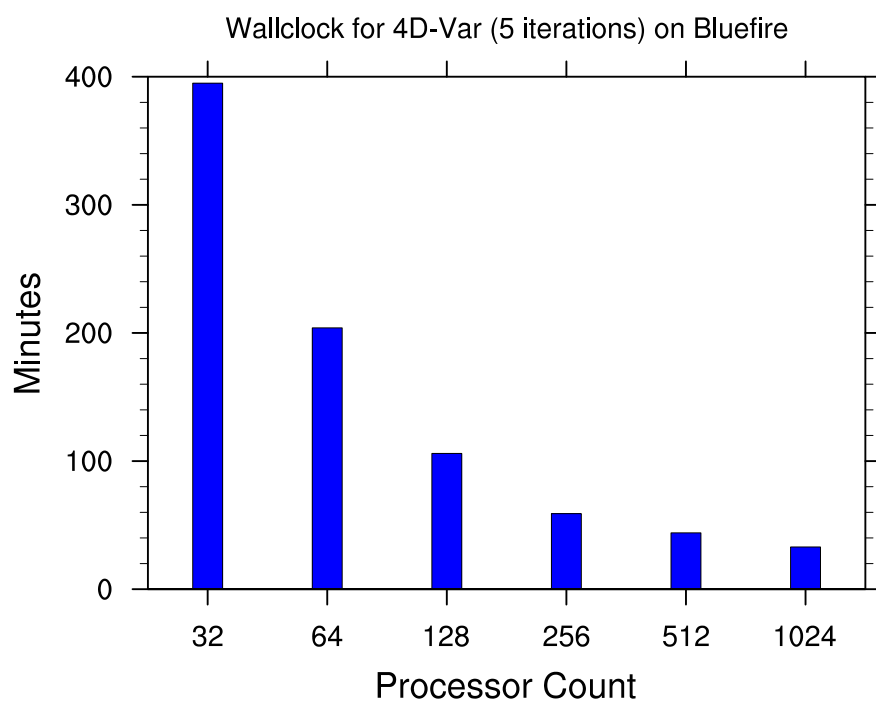


FIG. 4. Parallel performance for 4D-Var with 5 iterations on Bluefire