A recipe on how to implement UniR-WRF wind farm algorthm

The recipe on how to implement the algorithm is linked to version WRFV3.8 (ARW- core). It was tested and found OK using a Cray machine. In the following, the tilde symbol “~” indicates [your-WRF directory]. Steps to take:

1) Download a fresh WRF ARW-core V3.8

2) Check that it installes OK on your computer.
[Our option: 42. (dmpar) on PGI (ftn/gcc): Cray XC CLE]

3) Clean out what you just compiled by the command: clean -a

4) Copy file: module_wind_unir.F in the UniR_wind.tar file to ~/phys/.

5) Go to: “~Registry/Registry.EM_COMMON” :

a) After the line that starts with: “package fitchscheme”, add

```
   package    unirscheme     windfarm_opt==2     -      state:power
```

b) After the line starting with “rconfig integer windfarm_ij...”, add the following two lines:

```
rconfig   integer    windfarm_ind    namelist,physics    1    1  rh
"windfarm_ind"          "use flow induction factor" ""

rconfig   real   windfarm_tiplossf   namelist,physics    1    1. rh
"windfarm_tiploss"      "Prandtl's tip loss factor" ""
```

6) Go to your namelist.wrf

a) Add two lines in the physics section:

```
   windfarm_ind                        = 1,    ! induction factor
   windfarm_tiplossf                   = 0.9,  ! tiplossfactor
```

7) Go to the file: ~/phys/module_physics_init.F:

a) After the line: `USE module_wind_fitch`, add a new line:

```
  USE module_wind_unir
```

b) After “IF (config_flags%windfarm_opt .EQ. 1) THEN” add a new option:

```
  ELSEIF ( config_flags%windfarm_opt .EQ. 2 ) THEN
   CALL init_module_wind_unir(                           &
    id,config_flags,xlong,xlat,windfarm_initialized,  &
    ims,ime,jms,jme,its,ite,jts,jte,ids,ide,jds,jde   )
```

8) Go to the file: ~/phys/module_pbl_driver.F

a) After the line: `USE module_wind_fitch`, add a new line:

```
   USE module_wind_unir
```

b) Add a new section after the section: “CASE (fitchscheme)”:

```
     CASE (unirscheme)
```

```
       IF (PRESENT(id) .AND.  &
         PRESENT(z_at_w) ) THEN
        CALL wrf_debug(100,'in phys/module_wind_unir.F')
         CALL dragforceunir(   &
         & ID=id                &
         &,Z_AT_W=z_at_w,u=u_phy,v=v_phy &
         &,DX=dx,DZ=dz8w,DT=dt           &
         &,QKE=qke                       &
         &,DU=rublten,DV=rvblten         &
         &,WINDFARM_OPT=windfarm_opt,POWER=power              &
         &,IDS=ids,IDE=ide,JDS=jds,JDE=jde,KDS=kds,KDE=kde    &
         &,IMS=ims,IME=ime,JMS=jms,JME=jme,KMS=kms,KME=kme    &
         &,ITS=its,ITE=ite,JTS=jts,JTE=jte,KTS=kts,KTE=kte    )
                ELSE
                     WRITE ( message , FMT = '(A,6(L1,1X))' )
&                     'present: '//
&                    'ID, '//
&                    'z_at_w, '//
&                    'xlat_u, '//
&                    'xlong_u, '//
&                    'xlat_v, '//
&                    'xlong_v = ' ,
&                     PRESENT( id ) ,
&                     PRESENT( z_at_w )
                    CALL wrf_debug(0,message)
                     CALL wrf_error_fatal('Lack arguments to call
turbine_drag')
                  ENDIF
```

c) In the section “USE module_state_description, ONLY :”, add:

```
UNIRSCHEME
```

after the “FITCHSCHEME”. That is at 2 places (under each core).

9) Go to the file: ~/phys/Makefile.

a) After the line “module_wind_fitch.o \”, add:

```
module_wind_unir.o \
```

10) Compile the model by redoing what you did in step 2).