WRF 中加入空间格点人为热思路及流程

By. 赵漾

思路

将自己的格点人为热数据命名为 grid_AH，插值到 WRF 网格后作为一个新变量放到 wrfinput 里，通过 Registry 加变量读取，然后一步一步调用，把这个变量传递到需要的模块里。

WRF主要模块调用顺序：

读初始场：input_input

物理过程初始化：phy_init

输出第一个时次的history文件

读入其它auxinput文件如wrflowinp

读入侧边界：input_boundary

开始积分

微物理过程调用在solve_em.F

其它物理过程调用在module_first_rk_step_part1.F

物理过程初始化在start_em.F

参考网址：
https://wenku.baidu.com/view/3cc7f2381b5f312b3169a45177232f60dccce798.html

一、
WRF 默认的获取人为热（AH）的方式是通过子例程 read_param()来读取查找表：**module_sf_urban.F**

SUBROUTINE urban(…)　　该子例行计算的是一个格点某一时刻的值，可以看到后面调用该子例行时都是在 I,J 循环里，通过输入太阳时角 OMG 来计算某一时刻

　CALL read_param(…)　　通过查找表获得该格点的 AH，然后用 AH 进行相关变量的计算

　SOUBROUTINE read_param(…)
　END SOUBROUTINE read_param

END SUBROUTINE urban

现在获取 AH 就不使用子例程 read_param()，而是直接把我们自己的格点人为热值 one_grid_AH 输入 SUBROUTINE urban(), 然后在计算时把原本的 AH 用现在的 one_grid_AH 替换掉。

二、
一个格点的 AH 及与 AH 相关的变量是由 SUBROUTINE urban()获取的，下面就来查看哪些程序调用了 SUBROUTINE urban():

grep -r "CALL urban"

得到在 **module_sf_clm.F, module_sf_noahdrv.F, module_sf_noahmpdrv.F** 中有调用到子例行 urban，并且 module_sf_noahdrv.F 中就有三处.

```
module_sf_clm.F:            CALL urban(one_grid_AH,LSOLAR_URB,
module_sf_noahdrv.F:            CALL urban(one_grid_AH,LSOLAR_URB,
module_sf_noahdrv.F:                CALL urban(one_grid_AH,LSOLAR_URB,
module_sf_noahdrv.F:                CALL urban(one_grid_AH,LSOLAR_URB,
module_sf_noahmpdrv.F:    CALL urban(one_grid_AH,LSOLAR_URB,
```

在这些模块中的 CALL urban()里一一加入 one_grid_AH；

one_grid_AH 从哪里来？需要在 CALL urban()之前对其进行定义及赋值；
这三个模块的 CALL urban()分别在子例行 subroutine clmdrv(), SUBROUTINE lsm() 和 SUBROUTINE lsm_mosaic(), SUBROUTINE noahmp_urban 中；

因此要将我们的二维 grid_AH 加入这 4 个 subroutine，然后在 I,J 循环里取出某一格点的值 one_grid_AH 后赋给 CALL urban();

三、
在 subroutine clmdrv(), SUBROUTINE lsm() 和 SUBROUTINE lsm_mosaic(), SUBROUTINE noahmp_urban 中加入了新的形参 grid_AH,因此也要在 CALL 这 4 个子例行中加入对应的实参：

grep 后发现，这 4 个子例行都在 **module_surface_driver.F** 这个模块里，并且都在 SUBROUTINE surface_driver()里进行 CALL lsm(), CALL lsm_mosaic(), call noahmp_urban, CALL clmdrv();

因此也要在 SUBROUTINE surface_driver()加入 grid_AH,以此 CALL 三个子例行时能够得到输入的 grid_AH;

grep CALL surface_driver()的位置, 得到 **dyn_em/module_first_rk_step_part1.F**

注：
微物理过程调用在 solve_em.F
其他物理过程调用在 module_first_rk_step_part1.F

具体操作流程

Registry.EM_COMMON

Line 1691:
state real grid_AH ij dyn_em 1 - i0rh "grid_AH" "Gridded Anthropogenic Heat" "W m-2"

---

dyn_em/module_first_rk_step_part1.F

Line 10
SUBROUTINE first_rk_step_part1(

Line 450:
CALL surface_driver(
add grid_AH=grid%grid_AH,

Line 1511
END SUBROUTINE first_rk_step_part1

---

module_sf_urban.F

Line 288
SUBROUTINE urban(
add one_grid_AH
Line 327
REAL, INTENT(INOUT) :: one_grid_AH

Line759
AH=one_grid_AH*ahdiuprf(tloc)

Line 1585
END SUBROUTINE urban

---

module_sf_clm.F

Line 3812
subroutine clmdrv(
add grid_AH,

Line 3941
real,dimension(ims:ime,jms:jme),intent(inout)::grid_AH

Line 4300
REAL:: one_grid_AH（注意：需要加在声明部分，不要写到执行部分去了）

I,J loop stats from the line 4379

Line 5226
one_grid_AH=grid_AH(I,J)

Line 5228
CALL urban(
add one_grid_AH,

Line 5354
end subroutine clmdrv
---
module_sf_noahdrv.F

Line 31
SUBROUTINE lsm(
add grid_AH,

Line 247
REAL, DIMENSION(ims:ime, jms:jme), INTENT(INOUT) :: grid_AH

Line 652
REAL :: one_grid_AH

Line 1326
one_grid_AH=grid_AH(I,J)

Line 1328
CALL urban(
add one_grid_AH,

Line 1674
END SUBROUTINE lsm

Line 2194
SUBROUTINE lsm_mosaic(
add grid_AH,

Line 2430
REAL, DIMENSION(ims:ime, jms:jme), INTENT(INOUT) :: grid_AH

Line 2876
REAL :: one_grid_AH

Line 3630
one_grid_AH=grid_AH(I,J)

Line 3632
CALL urban(
add one_grid_AH,

Line 4486
CALL urban
add one_grid_AH,

Line 4656
END SUBROUTINE lsm_mosaic

---

module_sf_noahmpdrv.F

Line 2457
SUBROUTINE noahmp_urban (
add grid_AH

Line 2515
REAL, DIMENSION(ims:ime, jms:jme), INTENT(INOUT) :: grid_AH

Line 2689
REAL :: one_grid_AH

Line 2909
one_grid_AH=grid_AH(I,J)

Line 2911
CALL urban(
add one_grid_AH

Line 3189
END SUBROUTINE noahmp_urban

---

module_surface_driver.F

Line 7
SUBROUTINE surface_driver(
add grid_AH

Line 658
REAL, DIMENSION(ims:ime, jms:jme) , INTENT(INOUT):: grid_AH

Line 2547
CALL lsm_mosaic(
add grid_AH,

Line 2673
CALL lsm(
add grid_AH,

Line 3002
call noahmp_urban(
add grid_AH,

Line 3564
CALL clmdrv(
add grid_AH

Line 4076
END SUBROUTINE surface_driver

---

module_physics_init.F

Line 27
SUBROUTINE phy_init(
add grid_AH,

Line 258
REAL, DIMENSION(ims:ime, jms:jme), INTENT(INOUT) :: grid_AH

Line 1454
END SUBROUTINE phy_init

dym_em/start_em.F

Line 965
CALL phy_init (
add grid%grid_AH,

遇到的问题

①重新编译后报错提示：
module_first_rk_step_part1.f90(356): error #6460: This is not a field name that is defined in the encompassing structure. [GRID_AH]

```
254 module_first_rk_step_part1.f90(356): error #6460: This is not a field name that is defined in the encompassing struct
    ure.   [GRID_AH]
255 CALL surface_driver(HYDRO_dt=HYDRO_dt,sfcheadrt=grid%sfcheadrt,INFXSRT=grid%INFXSRT,soldrain=grid%soldrain,grid_AH=gr
    id%grid_AH,AC&
256 ----------------------------------------------------------------------------------------------------------------
    ---^
257 module_first_rk_step_part1.f90(356): error #8284: If the actual argument is scalar, the dummy argument shall be scala
    r unless the actual argument is of type character or is an element of an array that is not assumed shape, pointer, or
     polymorphic.   [GRID_AH]
258 CALL surface_driver(HYDRO_dt=HYDRO_dt,sfcheadrt=grid%sfcheadrt,INFXSRT=grid%INFXSRT,soldrain=grid%soldrain,grid_AH=gr
    id%grid_AH,AC&
259 -----^
260 compilation aborted for module_first_rk_step_part1.f90 (code 1)
261
262 real    0m1.541s
263 user    0m0.902s
264 sys     0m0.653s
265 make[2]: [module_first_rk_step_part1.o] Error 1 (ignored)
266 rm -f solve_em.o
```

不改动 Registry 后也是同样的报错，因此判定应当是对 Registry 的改动有问题

- Remember that ALL Registry changes require that the WRF code be cleaned and rebuilt

```
./clean -a
./configure
./compile em_real
```

修改 Registry 后，不仅要编译，还要重新 clean and configure.

②重新编译后运行，模型总是会到第二天中午的时候断掉，并且没有报错：
Timing for main: time 2019-06-24_12:15:30 on domain 2: 0.81247 elapsed seconds
看是不是有异常值：
将传递到子例行 urban 的 one_grid_AH 输出到 rsl（用函数 **wrf_message**），在 module_sf_urban.F 里写：

```
write(mesg,*) "here is my grid_AH(I,J)", one_grid_AH
call wrf_message(mesg)
```

发现输出的 AH 和我们输入的不一样，并且存在异常大值和负值（我们的格点 AH 没有负值）；
将子例程里 INTENT(IN) :: grid_AH 修改为 INTENT(INOUT) :: grid_AH 后异常值消失；
但是此时输出的 one_grid_AH 全为 0；去掉 wrfinput 里添加的 AH 后运行，为相同的情况，说明模式在从初始场读取或变量传递的过程中出现了问题；

从模式运行最开始的地方对 AH 进行输出，从而检查数据有没有被读进去，

module_surface_driver.F：

```
Line 4
CHARACTER (LEN=256) , PRIVATE      :: mesg

Line 2544
DO I=ims,ime
  DO J=jms,jme
    WRITE( mesg,* ) 'here is my input grid_AH',    grid_AH(I,J)
    CALL wrf_message(mesg)
  ENDDO
ENDDO
```

发现输出的 AH 依旧为全为 0，说明 dyn_em/module_first_rk_step_part1.F 在 CALL surface_driver 时，grid_AH 参数就没有正常输入进去；

后来发现在./real.exe 生成的 wrfinput 文件里，就存在一个名为 GRID_AH 的变量，并且里面的值全为 0；因此不要自己另外创建一个变量!!!（因为我之前以为 wrfinput 里没有 grid_AH，自己又手动在里面加了一个 grid_AH 变量）,直接把这个原本存在的 GRID_AH 变量里的值进行修改就行!! 最后再运行时，我们的格点 AH 就被成功读入啦!!

Registry中input和output文件：

```
i0: wrfinput
i1: met_em
i2, 3: used
i4: wrflowinp
i5-8, 12-15: chem
i9: wrfsfdda-grid nudging - surface
i10: wrffdda-grid nudging - 3d
i11: obs nudging
i16: IC:CG
17-24: 空闲
h0: wrfout
h1, 5: used
h2: afwa rainfall, afwa_diag_opt=1
h3: wrfxtrm, output_diagnostics = 1
h6: cam...
h22: &diag: z_lev_diag=1
h23: &diag: p_lev_diag=1
```