Integrate the latest version of new driver into ORCHIDEE trunk

Objective:

The new driver was initially developed by Jan in the branch Routing, and then it was included in ORCHIDEE by Josefine (and others ?). Since then, the code has been probably updated by Josefine (and others ?) in the trunk, by following the evolution of ORCHIDEE. This can explain some differences between the trunk and the branch Routing. It is not yet clear if some tests have been applied to the new driver in the trunk after those updates.

In 2019-2020, more changes were applied to the new driver by Jan, realized in the branch Routing. These changes fixed some bugs in the previous version. Now we need integrate these new changes into the trunk.

What is done:

1) Simulation test with existing FG3nd in the trunk

In order to understand the functionality of the new driver, I did simulation FG3nd by using the trunk (rev 6816), with the orchidee driver already integrated. This is done with the server of meso center.

The job script was created by using libIGCM. The forcing file is WFDEI. It can run without error message. However it seems quite slow (almost 1 day for 1-year simulation), and it uses large memory. For example the maximal virtual memory of 100 G is not enough, and 200G is OK. It demands 64 processors, rather than 32 (using the older driver with the default configuration of ORCHIDEE).

WFDEI forcing data has 0.5 degree resolution in space and 3 hour in time. By default, the variable slab_size_max is 80 in ORCHIDEE, which means reading/processing 10 days of data. In order to test if the large value of slab_size_max may be responsible for the slowness, I did a simple simulation by reducing slab_size_max from 80 to 20, and PeriodLength from 1Y to 1M (config.card). However, the simulation need the max virtual memory of about 200G. Thus, using the forcing files of high spatial resolution, the meso center server may not be the good choice for the global simulation, and the servers with higher performance will be better.

2) Comparisons of new driver codes between the routing and the trunk

In order to integrate the latest version of new driver into the trunk, I downloaded the codes from the branch Routing. Then I compared the new driver between trunk (rev 6816) and the Routing. There are some important changes in orchideedriver.f90, forcing_tools.f90 for example. It is important to understand their differences. The detailed comparisons can be found in the appendix A.

3) Modification of new driver in the trunk and test with WFDEI

I discussed with Jan about the differences of drivers between the branch Routing and the trunk. Then I modified the new driver in the trunk (rev 6816) according to the discussion.

The compilation is OK, and we can create a job by using libIGCM. The simulation FG3nd is running with WFDEI for the global simulation, though it is slow and takes large memory on the server of meso center.

4) Compare FG3 and FG3nd by using the same forcing data and the same configurations

I tried to make simulations of FG3 and FG3nd by using CRU NCEP 2-degree forcing data and the same configurations. This is for a better comparison between the new and the old drivers, as suggested by Philippe Peylin.

It seems that FG3/FG3nd experiment was derived from a very early version of FG2. There are several differences between FG3/FG3nd and the recent version of FG2 (see below). It looks like that the configurations of FG3/FG3nd did not follow the evolution of FG2 of ORCHIDEE.

a) *stomate.card:*
FG3 and FG3nd does not need a global_forest_mangagement file FMmap.nc, while FG2 need it.

b) o*rchidee.def:*
FG3/FG3nd has FOREST_MANAGED_FORCED=2, while FG2 uses a flag OK_READ_FM_MAP=y.

c) *sechiba.card:*
FG3nd need a background albedo file by default, while FG2 does not.

d) *restart file:*
By default FG3nd configuration does not include restart file, while FG2/FG3 can. It seems that the default format of restart file of FG3nd (which was derived from an old version of trunk) may not be the same as that of FG1trans of recent trunk version.

e) *SPRED_PRES:*
The default value is different between FG2 and FG3nd. FG2 uses the half of forcing step by default, while FG3nd prefers shorter duration (1 hour for example). We can change them to the same values.

f) *number of forcing file as input:*
The new driver need 3 files of forcing data, while FG2 with old driver need only the current year.

Therefore, it looks necessary to update FG3nd/FG3 configurations to be consistent with that of FG2, so that the comparisons between FG3 and FG3nd make more sense.

Even if setting c), d), e), f) to be the same for FG3 and FG3nd, it is not very clear if the other configurations are the same for the two simulations.

5) Some other concern in the simulation
I ran a simulation of FG3nd by using CRU NCEP forcing file (2 degree), which is the default forcing file in current FG simulations. The simulation can run for a small region. But there is a segmentation error when using the global dataset.

I noticed that the number of land points calculated from the domain and from the ‘contfrac’ is not the same. Suggested by Fabienne, I made a quick check of the forcing file. In fact the ‘contfrac’ parameter in the forcing file has a value of 0 in the last pixel (-89, 179), while the landsea mask sets this point as ‘land’. This small issue may cause memory problem and is likely responsible for the segmentation error. We probably need new conditions (as Jan suggested), or a few other modification to the driver, such that we may solve this issue.

Summary and questions:

1) The recent version of the new driver is integrated into the trunk version of ORCHIDEE
2) Simulation of FG3nd with WFDEI runs, but it only works for regional simulations when using CRU NCEP
3) It seems necessary to modify a bit the new driver, in order to be better deal with possible

inconsistency of the land numbers (ie., from the domain and from the forcing file)
4) It seems necessary to update FG3/FG3nd configurations to the recent ORCHIDEE trunk
4) There are some important differences in the organization of subroutines/modules the new driver between the routing and the trunk
6) The simulation on meso center machines seems relative slow for global simulation by using high-spatial resolution forcing data. High-performance servers will be better for such a simulation.

Appendix :

A: compare the new driver from trunk (6816) and branch Routing

forcing_tools.f90: some differences between routing and trunk.

1) use mask(1:slab_size) in the routing, instead of mask(:) in the trunk :
we accept that from the routing, suggested by Jan, in order to avoid a bug.

2) if forcing_tstep_ave >= one_day/4.0 in the routing (instead 3.0 in trunk):
we accept 3.0 from the trunk, suggested by Jan, in order that it works also for 6-hourly forcing data.

3) subroutine forcing_buildindex:
In the routing, there is a new condition if MAXVAL(var2d) > var_missing when dealing with the missing values in vard2D. This is included in my test within the trunk.

4) subroutine forcing_contfract2d:
In the routing, there is a new condition if MAXVAL(contfrac) >= contfrac_missing when counting the number of land points ‘nbland’. This is accepted in my test.

5) subroutine forcing_zoomgrid: The computation of nbland_loc is slightly different.

In the routing: nbland_loc = SUM(contfrac_loc)
in the trunk:

```
nbland_loc = 0
Do ik = 1, SIZE(contfrac_loc)
    IF (contfrac_loc(ik) > 0.0) THEN
        nbland_loc = nbland_loc  +1.0
    ENDIF
 ENDDO
```

In my test, I accept the lines from trunk.

6) subroutine forcing_givegrid:
In the trunk, there is a few new lines to compare nbpoint_loc and nbland_loc in the subroutine forcing_givegrid. This condition is not included in the routing.
I understand that this condition is useful to notify the inconsistency problem between the two numbers. This condition can be be kept in the new driver. Maybe more importantly, we need take into account the inconsistency problem in the new driver.

forcingdaily_tools.f90: identical

globgrd.f90: slight difference.
There is a new line CALL getin('WRF_CALENDAR', calendar) in the routing. This defines the calendar used by the WRF simulation.

If we do not do the WRF simulation, I guess that we do not need this line ? In my current test, it is not yet included.

orchidee_drive.f90: important changes between trunk and routing.
Trunk:

1) use time module (which is not used in routing)

2) use xios_orchidee, constants, constants_soil (which seem to have been reorganized differently from the routing)

3) new parameters (co2 and vege related) added in the trunk compared to the routing

4) call grid_init : minor change.
CALL grid_init (nbp_loc, nbseg, "RegLonLat", "ForcingGrid") in the routing
CALL grid_init (nbp_loc, nbseg, regular_lonlat, "ForcingGrid") in the trunk

5) when transferring global grid variables to the orchidee version of the root proc:
A condition 'is_root_proc' is applied in the routing, but not in the trunk. This condition is not applied in my current test (also agreed to Jan).

6) call time_initialise in order to set the starting date in IOISPL and initialize the calendar (realized differently in the routing, see below)

7) before going into the time loop for itau = 1, nbdt:
a) calling xios_orchidee_init:
this subroutine need an input variable julian_diff, which is now a global variable in the src_global/time.f90 in the trunk. (it is only a local variable in the routing).
b) Then calling sechiba_xios_initialize (which is not used in the routing)
c) Then calling xios_orchidee_close_definition (not used in the routing)

8) within the time loop:
a) call time_nextstep in order to update the time
b) Julian = data0+itau *(dt/one_day), which is switched by 0.5 from routing.
I modified to julian = (julian_start + julian_end) /2.0 in the trunk to be consistent with the routing.
c) after getting forcing data, call xios_orchidee_update_calendar without any condition, (while a condition of ok_calendar is applied in the routing)
d) if itau == 1:
call sechiba_initialize to set up orchidee before doing an call for getting actual fluxes.
Else:
call sechiba_main
There is no update of calendar in these steps in the trunk, while there are several calls of xios_orchidee_update_calendar in the routing with some condition applied. These updates of calendar are not necessary in the trunk.

Routing:

1) Use tool_para module (not used in trunk) : this difference seems not important

2) use xios (not xios_orchidee as the trunk): the subroutine xios_orchidee_init from xios has less input arguments.

3) use thermosoilc (soilth_lev) : not necessary for the version in the trunk

4) define a variable julian0 in the routing, but the same variable is now defined in src_global/solar.f90 in trunk

5) get the vertical soil levels soilth_lev for the thermo scheme, which is to be used in xios_orchidee_init:

This is not necessary in the trunk, because it is already included in xios_orchidee_init.

6) initialize the calendar:

The routing initialize julian = date0 + 0.5*dt/one_day, this line is not used in the trunk. This seems not necessary to be taken into account (according to Jan).

Then call xios_orchidee_init with a condition of NOT ok_calendar, which is not included in the trunk. This call is not necessary for trunk.

7**)** Within the time loop:

a) the definition of variable 'julian' is different from that in the trunk.

julian = date0 + (itau-0.5)*(dt/one_day), while in trunk julian=data0+itau *(dt/one_day).

The one from the routing is accepted.

b) xios_orchidee_update_calendar is called in the routing for several times with the conditions of ok_calendar, not ok_largest, not ok_grdc etc.

This is not necessary in the trunk according to Jan.

8) when close everything: call xios_orchidee_context_finalize. This line is not used in the trunk.

B: Rev 6816 .vs. the current version of trunk

The reason to use the revision 6816, instead of the latest version of trunk, is explained below. I made a test of FG1trans with the latest version of trunk. The compilation is OK. But when running a simulation, I found an error message: MPI_ABORT was invoked on rank 0 in communicator MPI_COMM_WORLD with errorcode 1. Looking at this with Fabienne, it is related to an issue of mass balance, that is., the mass balance is not closed in stomate_lpj for carbon.

Recently Matthew has investigated this issue and has made good progress. It seems that this is solved.