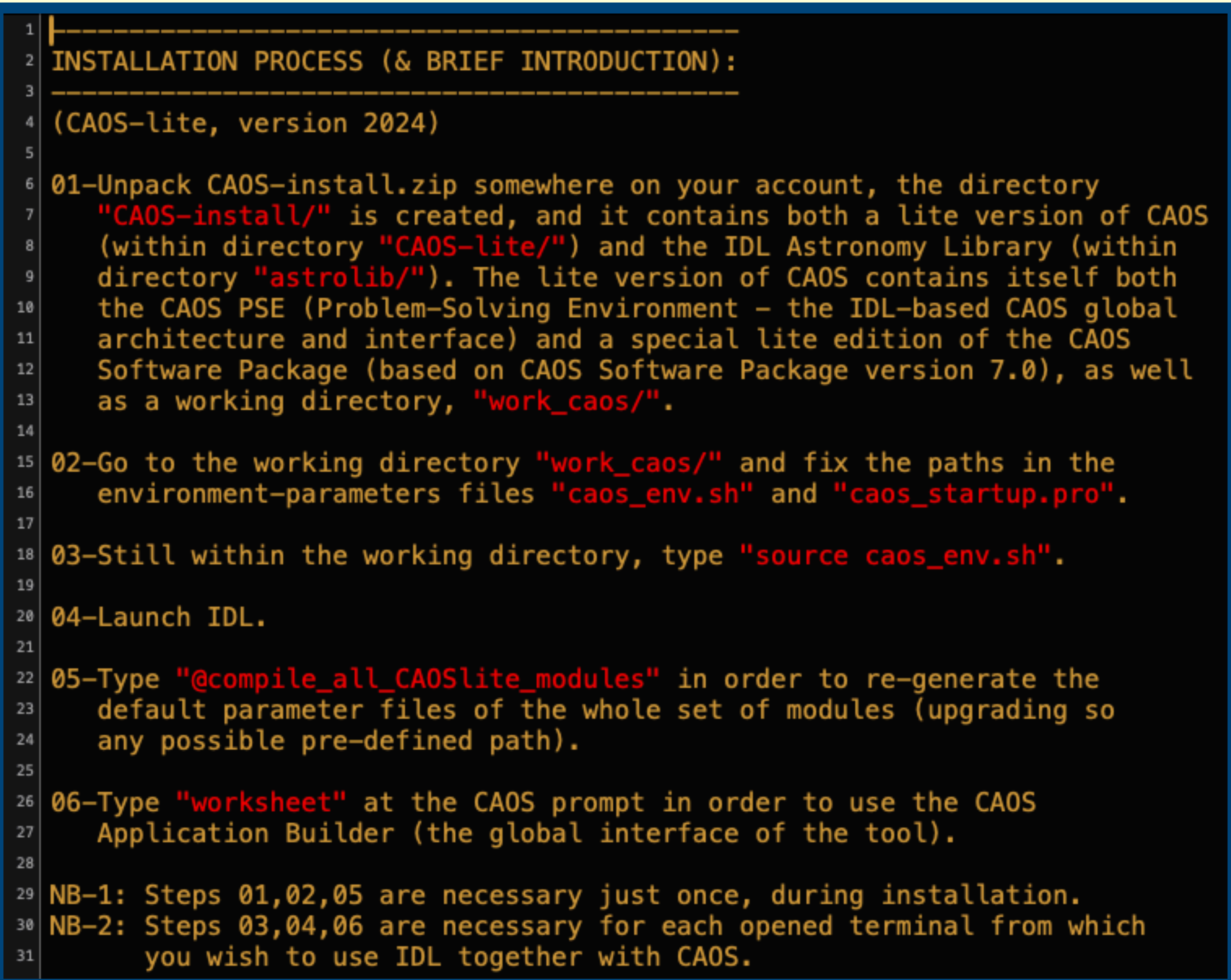
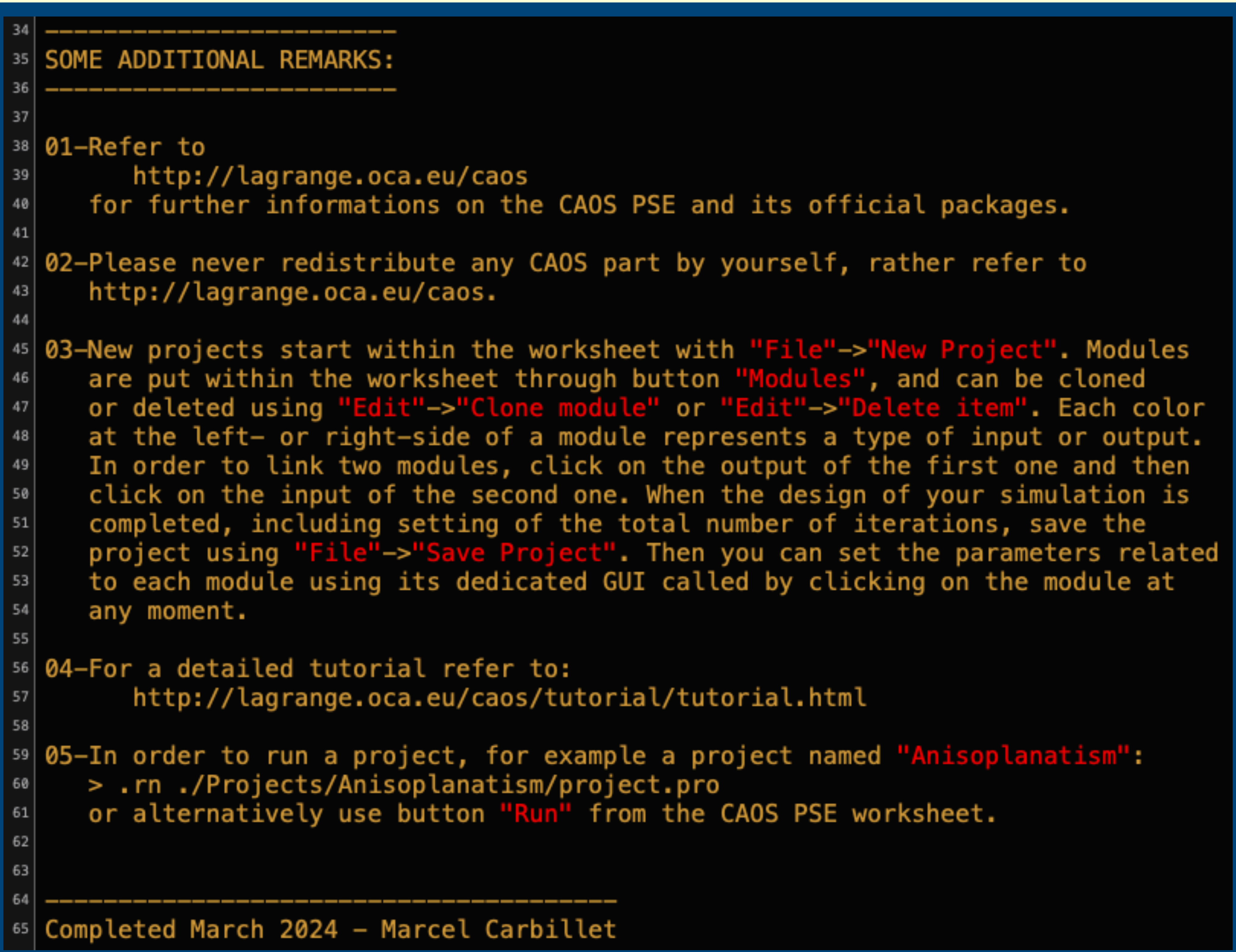
Installation of CAOS-lite — 1
(CAOS PSE + lite version of Soft.Pack.CAOS)



```
--------------------------------------------
INSTALLATION PROCESS (& BRIEF INTRODUCTION):
--------------------------------------------
(CAOS-lite, version 2024)

01-Unpack CAOS-install.zip somewhere on your account, the directory
   "CAOS-install/" is created, and it contains both a lite version of CAOS
   (within directory "CAOS-lite/") and the IDL Astronomy Library (within
   directory "astrolib/"). The lite version of CAOS contains itself both
   the CAOS PSE (Problem-Solving Environment - the IDL-based CAOS global
   architecture and interface) and a special lite edition of the CAOS
   Software Package (based on CAOS Software Package version 7.0), as well
   as a working directory, "work_caos/".

02-Go to the working directory "work_caos/" and fix the paths in the
   environment-parameters files "caos_env.sh" and "caos_startup.pro".

03-Still within the working directory, type "source caos_env.sh".

04-Launch IDL.

05-Type "@compile_all_CAOSlite_modules" in order to re-generate the
   default parameter files of the whole set of modules (upgrading so
   any possible pre-defined path).

06-Type "worksheet" at the CAOS prompt in order to use the CAOS
   Application Builder (the global interface of the tool).

NB-1: Steps 01,02,05 are necessary just once, during installation.
NB-2: Steps 03,04,06 are necessary for each opened terminal from which
      you wish to use IDL together with CAOS.
```

Installation of CAOS-lite — 2
(CAOS PSE + lite version of Soft.Pack.CAOS)



```
------------------------
SOME ADDITIONAL REMARKS:
------------------------


01-Refer to
      http://lagrange.oca.eu/caos
   for further informations on the CAOS PSE and its official packages.

02-Please never redistribute any CAOS part by yourself, rather refer to
   http://lagrange.oca.eu/caos.

03-New projects start within the worksheet with "File"->"New Project". Modules
   are put within the worksheet through button "Modules", and can be cloned
   or deleted using "Edit"->"Clone module" or "Edit"->"Delete item". Each color
   at the left- or right-side of a module represents a type of input or output.
   In order to link two modules, click on the output of the first one and then
   click on the input of the second one. When the design of your simulation is
   completed, including setting of the total number of iterations, save the
   project using "File"->"Save Project". Then you can set the parameters related
   to each module using its dedicated GUI called by clicking on the module at
   any moment.

04-For a detailed tutorial refer to:
      http://lagrange.oca.eu/caos/tutorial/tutorial.html

05-In order to run a project, for example a project named "Anisoplanatism":
   > .rn ./Projects/Anisoplanatism/project.pro
   or alternatively use button "Run" from the CAOS PSE worksheet.



---------------------------------------
Completed March 2024 - Marcel Carbillet
```

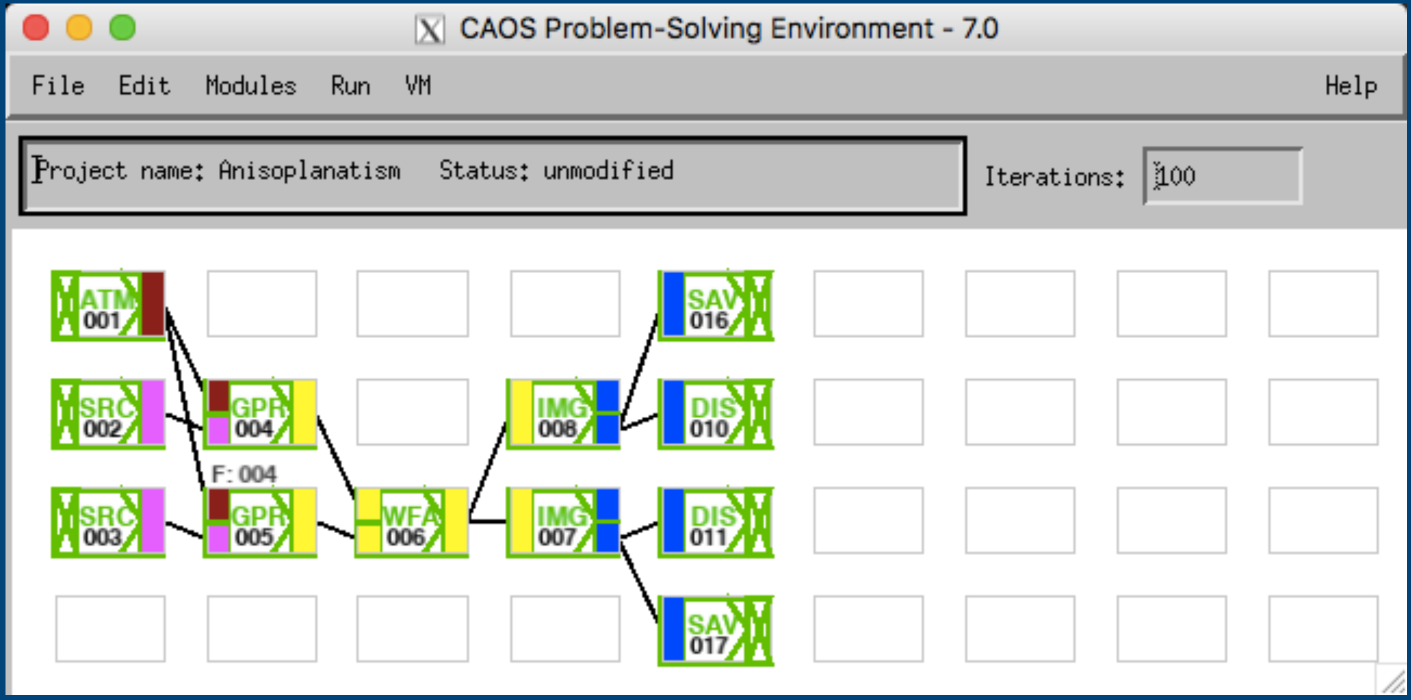
Build project for anisoplanatism…

0- COMPLETELY FINALISE INSTALLATION OF THE CAOS PSE AND THE SOFTWARE PACKAGE CAOS (POSSIBLY LITE VERSION OF IT) BEFORE GOING ON !!

Then, within the CAOS interface…

1- Reproduce the project ``Anisoplanatism” here beside.

2- Click on the ATM module, its graphical user interface (GUI) opens, then change its parameters into your own ones (r_0, L_0, altitude of the layers, mainly), and finally save them with button ``Save”.



Project name: Anisoplanatism Status: unmodified

Iterations: 100

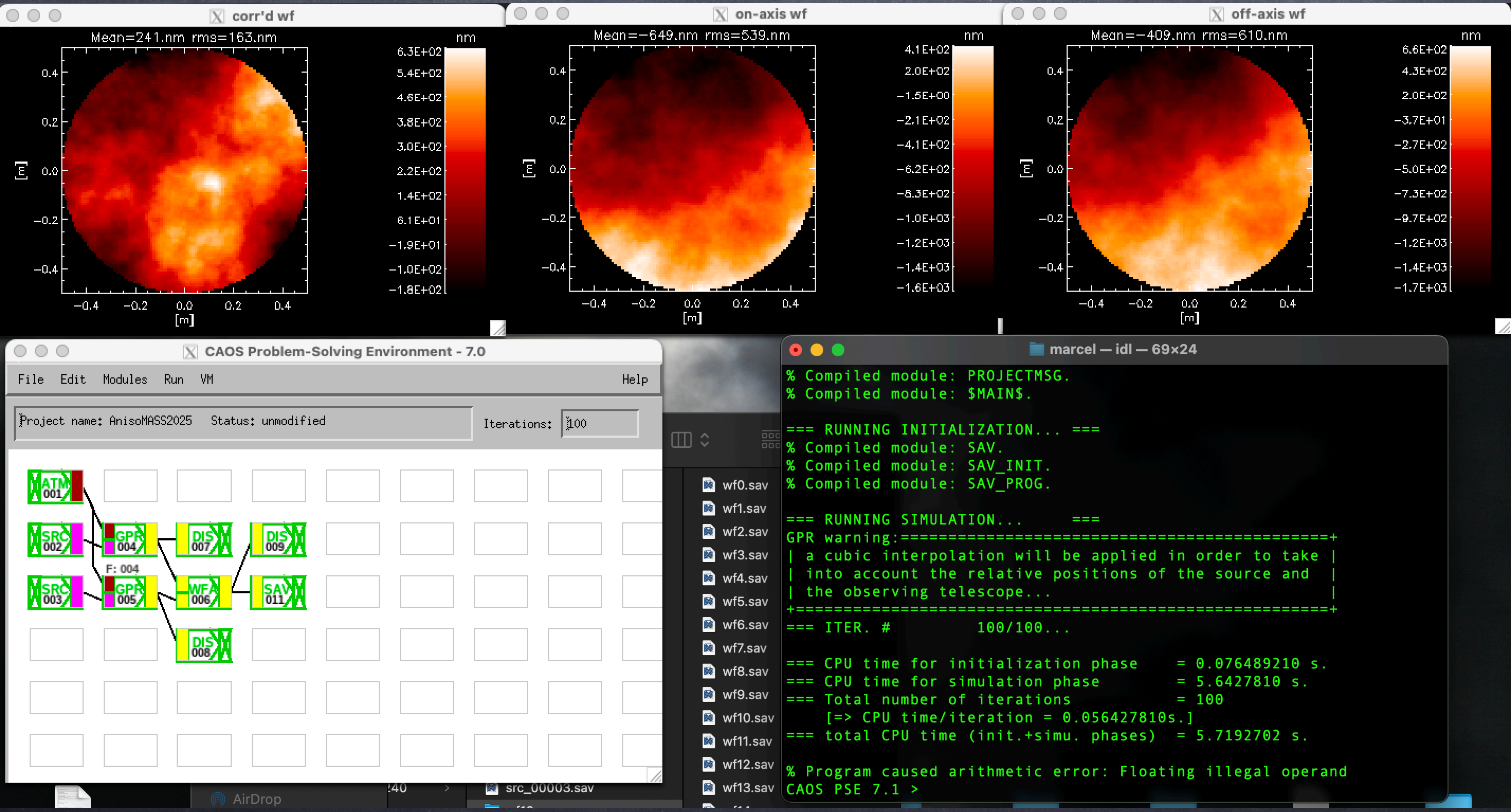
3- Choose a value for the off-axis angle (typically in between 0” and 60”) within second occurence of module SRC and, as a consequence, adapt the name of the saved PSFs within the two modules SAV (one for each module IMG, i.e. one for each considered wavelength: for example 500nm and 1650nm).

4- Fix the parameters of the other modules.

5- Run the simulation project by using button ``Run” within the CAOS interface (or with the IDL-CAOS command ``.rn ./Projects/Anisoplanatism/project.pro” for a project called ``Anisoplanatism”).

6- Repeat steps 3 to 5 for each chosen value of the off-axis angle.

7- Compute the rms of the corrected wavefront and the FWHM for each resulting PSFs (two for each off-axis angle value) with routine ``dataprocessing.pro”.

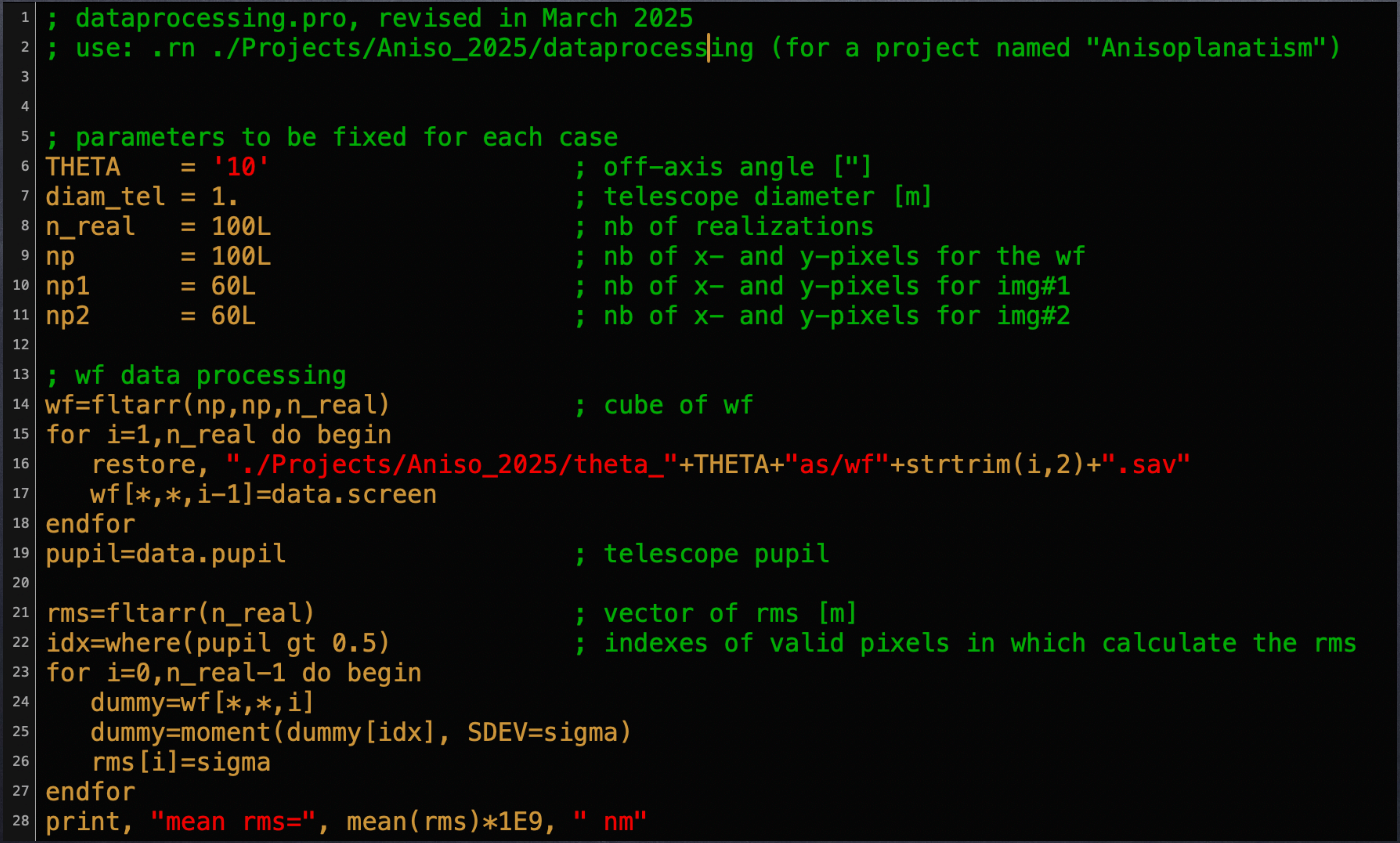


Project name: AnisoMASS2025   Status: unmodified

Iterations: 100

```
% Compiled module: PROJECTMSG.
% Compiled module: $MAIN$.

=== RUNNING INITIALIZATION... ===
% Compiled module: SAV.
% Compiled module: SAV_INIT.
% Compiled module: SAV_PROG.

=== RUNNING SIMULATION...     ===
GPR warning:=============================================+
| a cubic interpolation will be applied in order to take |
| into account the relative positions of the source and  |
| the observing telescope...                             |
+========================================================+
=== ITER. #          100/100...

=== CPU time for initialization phase    = 0.076489210 s.
=== CPU time for simulation phase        = 5.6427810 s.
=== Total number of iterations           = 100
    [=> CPU time/iteration = 0.056427810s.]
=== total CPU time (init.+simu. phases)  = 5.7192702 s.

% Program caused arithmetic error: Floating illegal operand
CAOS PSE 7.1 >
```

(routine dataprocessing.pro — 1)



```
; dataprocessing.pro, revised in March 2025
; use: .rn ./Projects/Aniso_2025/dataprocessing (for a project named "Anisoplanatism")


; parameters to be fixed for each case
THETA    = '10'                    ; off-axis angle ["]
diam_tel = 1.                      ; telescope diameter [m]
n_real   = 100L                    ; nb of realizations
np       = 100L                    ; nb of x- and y-pixels for the wf
np1      = 60L                     ; nb of x- and y-pixels for img#1
np2      = 60L                     ; nb of x- and y-pixels for img#2

; wf data processing
wf=fltarr(np,np,n_real)            ; cube of wf
for i=1,n_real do begin
   restore, "./Projects/Aniso_2025/theta_"+THETA+"as/wf"+strtrim(i,2)+".sav"
   wf[*,*,i-1]=data.screen
endfor
pupil=data.pupil                   ; telescope pupil

rms=fltarr(n_real)                 ; vector of rms [m]
idx=where(pupil gt 0.5)            ; indexes of valid pixels in which calculate the rms
for i=0,n_real-1 do begin
   dummy=wf[*,*,i]
   dummy=moment(dummy[idx], SDEV=sigma)
   rms[i]=sigma
endfor
print, "mean rms=", mean(rms)*1E9, " nm"
```