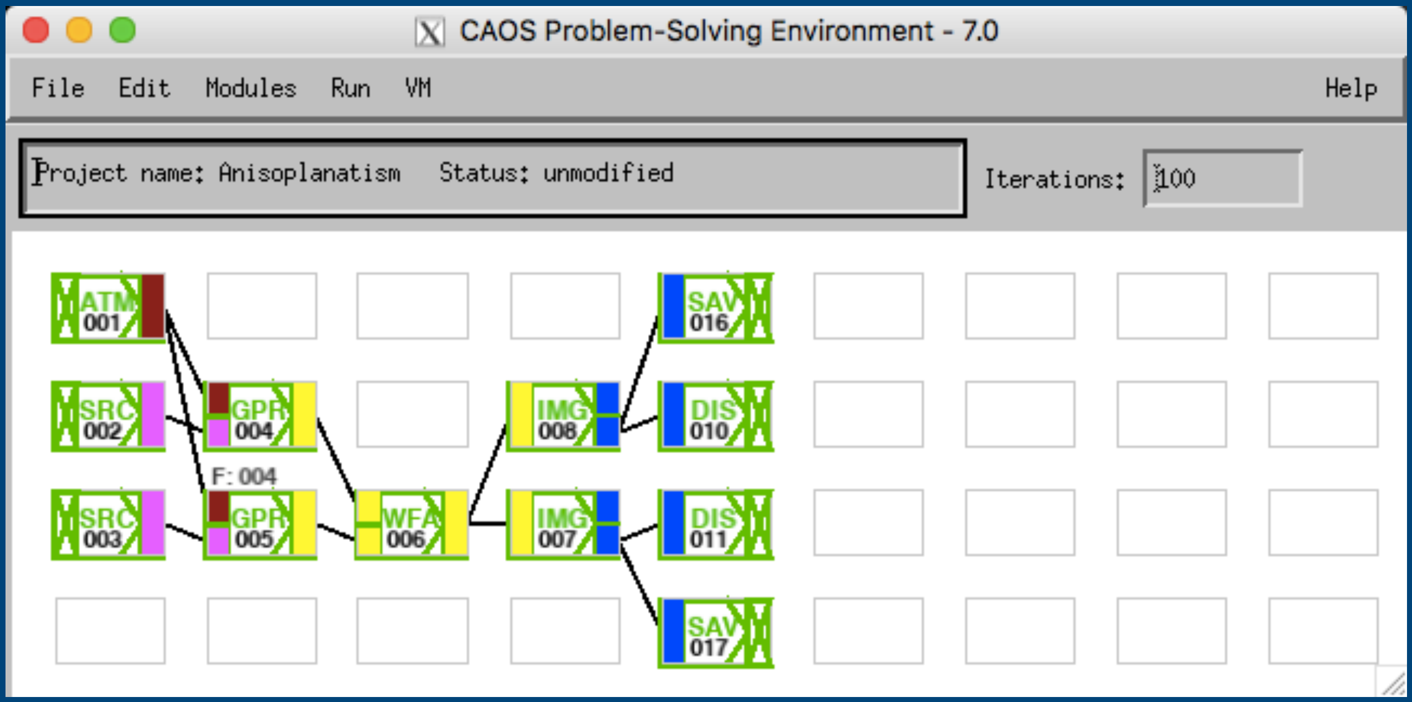
Build project for anisoplanatism…

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".



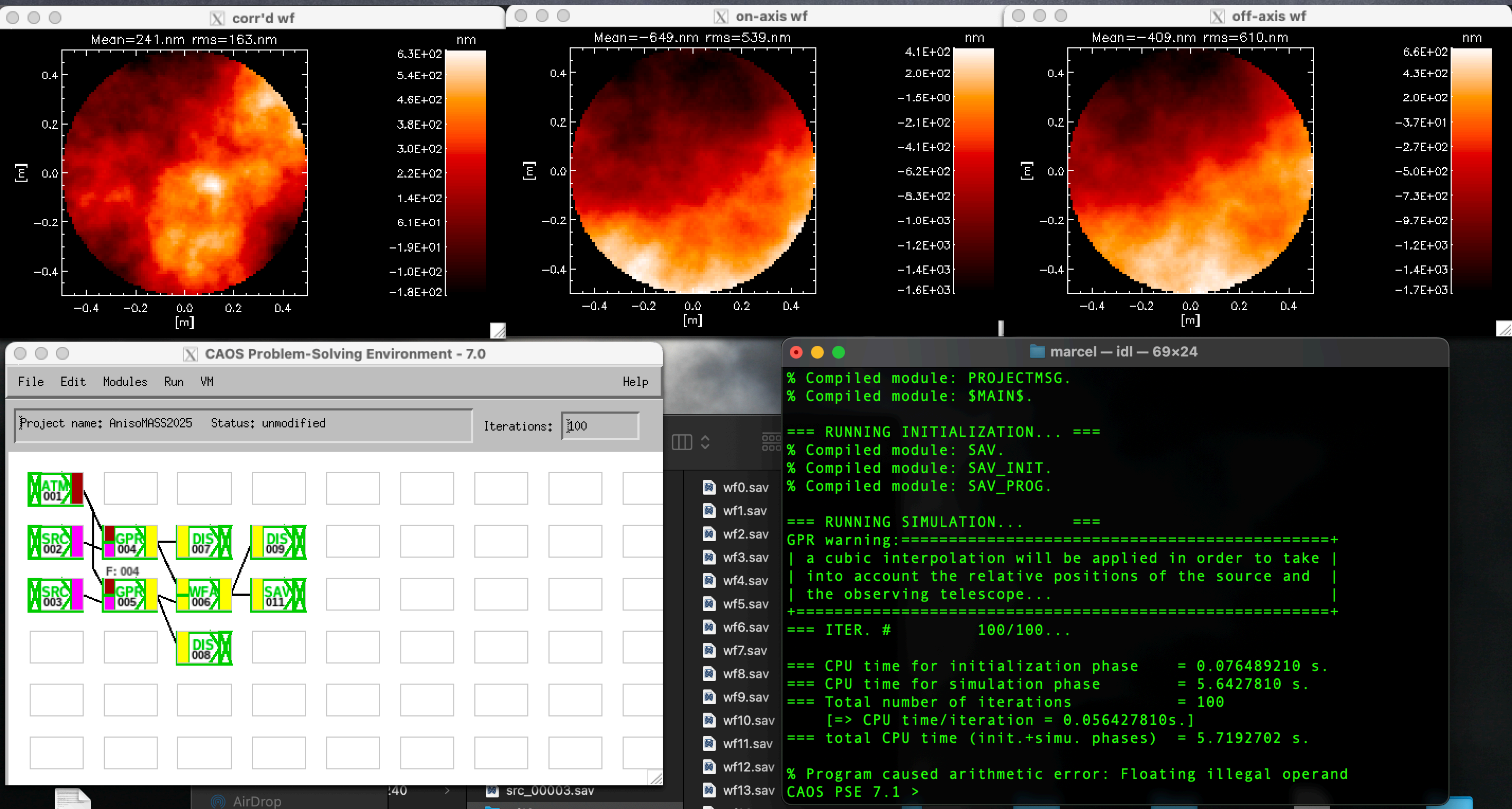
3- Choose a value for the off-axis angle (typically in between 0" and 60") within second occurence of module SRC and, as a consequence, adapt the name of the saved PSFs within the two modules SAV (one for each module IMG, i.e. one for each considered wavelength: for example 500nm and 1650nm).

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".



corr'd wf
Mean=241.nm rms=163.nm
nm
6.3E+02
5.4E+02
4.6E+02
3.8E+02
3.0E+02
2.2E+02
1.4E+02
6.1E+01
−1.9E+01
−1.0E+02
−1.8E+02
[m]
on-axis wf
Mean=−649.nm rms=539.nm
nm
4.1E+02
2.0E+02
−1.5E+00
−2.1E+02
−4.1E+02
−6.2E+02
−8.3E+02
−1.0E+03
−1.2E+03
−1.4E+03
−1.6E+03
off-axis wf
Mean=−409.nm rms=610.nm
nm
6.6E+02
4.3E+02
2.0E+02
−3.7E+01
−2.7E+02
−5.0E+02
−7.3E+02
−9.7E+02
−1.2E+03
−1.4E+03
−1.7E+03
CAOS Problem-Solving Environment - 7.0
File Edit Modules Run VM Help
Project name: AnisoMASS2025 Status: unmodified
Iterations: 100
ATM 001
SRC 002
GPR 004
DIS 007
DIS 009
F: 004
SRC 003
GPR 005
WFA 006
SAV 011
DIS 008
wf0.sav
wf1.sav
wf2.sav
wf3.sav
wf4.sav
wf5.sav
wf6.sav
wf7.sav
wf8.sav
wf9.sav
wf10.sav
wf11.sav
wf12.sav
wf13.sav
src_00003.sav
AirDrop
marcel — idl — 69×24
% 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 >

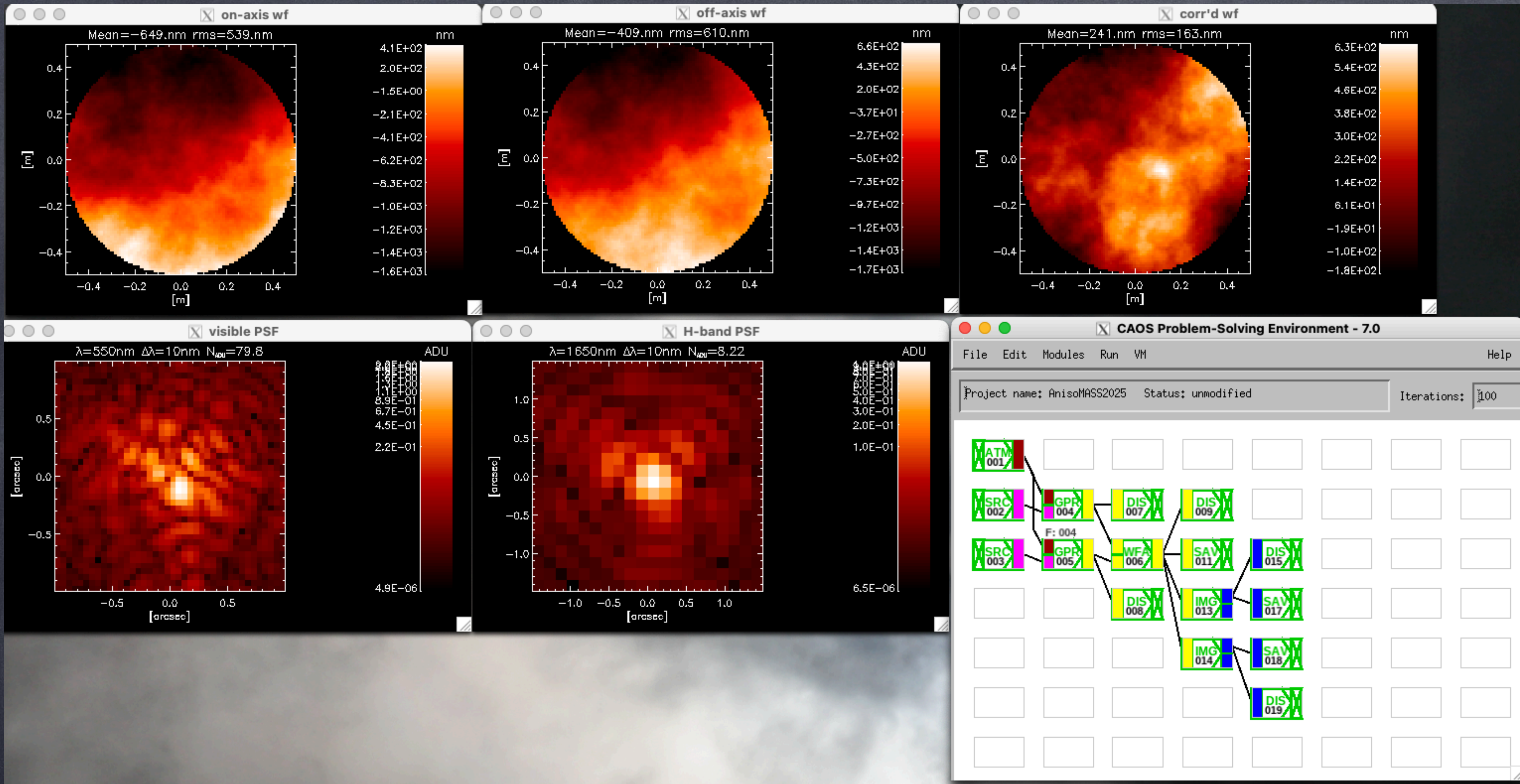
(routine dataprocessing.pro — 1)

```
; 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"
```



- addition of IMG module occurrences for two wavelengths

- => Δx for each band ?

  => FoV=N.Δx for each band ?

(routine dataprocessing.pro — 2)

```
; 500-nm images processing
img500nm=fltarr(np1,np1,n_real)      ; cube of 500-nm PSFs
for i=1,n_real do begin
   restore, "./Projects/Aniso_2025/theta_"+THETA+"as/PSF500nm"+strtrim(i,2)+".sav"
   img500nm[*,*,i-1]=data.image
endfor

PSF_LE = total(img500nm,3)           ; long-exposure PSF
LAMBDA = data.lambda                 ; wavelength [m]
RES    = data.resolution             ; pixel size ["]
dummy  = gauss2dfit(PSF_LE,a) & sig = (a[3]+a[2])/2.
fwhm   = 2*sig*sqrt(2*alog(2))*RES ; FWHM ["]
print, "FWHM = ", fwhm, '" = ', fwhm/(LAMBDA/diam_tel*!RADEG*3600), " lambda/D"

; H-band images processing
imgHband=fltarr(np2,np2,n_real)      ; cube of H-band PSFs
for i=1,n_real do begin
   restore, "./Projects/Aniso_2025/theta_"+THETA+"as/PSF1650nm"+strtrim(i,2)+".sav"
   imgHband[*,*,i-1]=data.image
endfor

PSF_LE = total(imgHband,3)           ; long-exposure PSF
LAMBDA = data.lambda                 ; wavelength [m]
RES    = data.resolution             ; pixel size ["]
dummy  = gauss2dfit(PSF_LE,a) & sig = (a[3]+a[2])/2.
fwhm   = 2*sig*sqrt(2*alog(2))*RES ; FWHM ["]
print, "FWHM = ", fwhm, '" = ', fwhm/(LAMBDA/diam_tel*!RADEG*3600), " lambda/D"
```