```
function wfgeneration, dim, length, L0, r0, lambda, SEED=seed
;
; wave-front (wf) generation following Kolmogorov or von Karman model
;
; dim    = wf linear dimension [px],
; length = wf physical length [m],
; L0     = wf outer-scale [m],
; seed   = random generation seed (OPTIONAL),
; r0     = Fried parameter at wavelength 'lambda' [m],
; lambda = wavelength at which r0 is defined.
;
; Marcel Carbillet [marcel.carbillet@unice.fr],
; UMR 7293 Lagrange (UNS/CNRS/OCA), February 2013.
;
; Last modification: Feb. 2014
;
phase = (randomu(seed,dim,dim)-.5) * 2*!PI    ; rnd uniformly distributed phase
                                              ; (between -PI and +PI)
rr = dist(dim)
if L0 eq !VALUES.F_INFINITY then rr[0,0] = 1.; avoid 1/infinity afterwards
                                              ; for Kolmogorov model
modul = (rr^2+(length/L0)^2)^(-11/12.)        ; vonKarman/Kolmogorov model
if L0 eq !VALUES.F_INFINITY then modul[0,0] = 0.
```

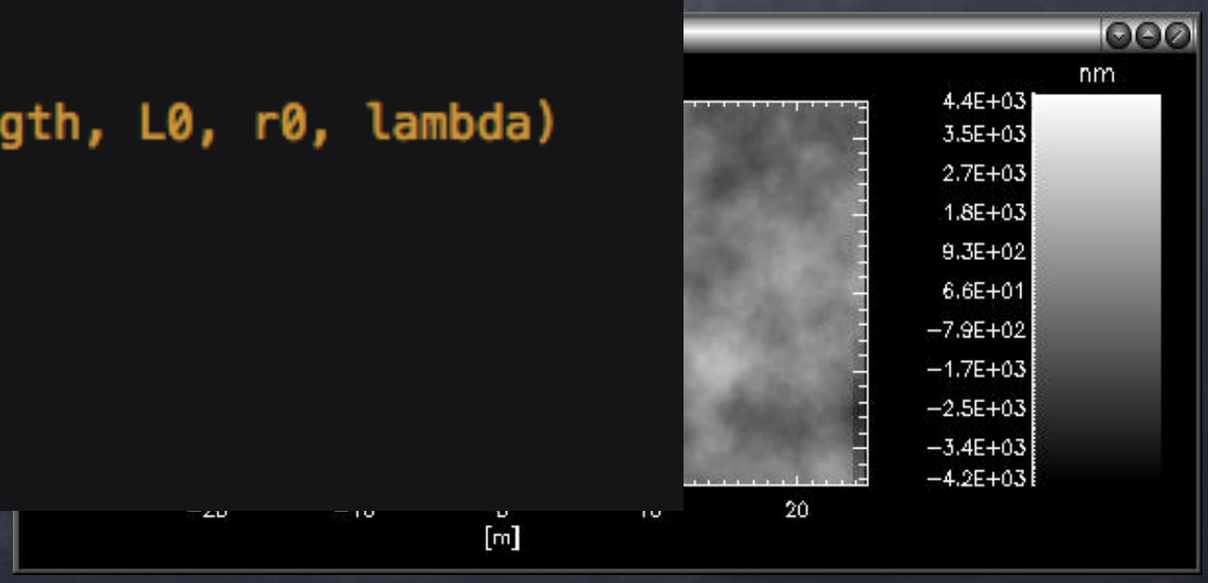
```
function wf_cube, dim, length, L0, r0, lambda, n_wf

cube_wf = fltarr(dim, dim, n_wf)

for i=0, n_wf/2-1L do begin
   cube_wf[*,*,i]=wfgeneration(dim, length, L0, r0, lambda)
endfor

save, cube_wf, FILE='cube_wf.sav'

return, cube_wf
end
```



« wf generation »
générer un cube de fronts d'onde statistiquement indépendants (une centaine)...
=> calculer l'écart-type moyen pour différents $[r_0, L_0]$