EFFECTIVE FIELD THEORY OF DARK ENERGY

Computer Programming Project

Lucía Fonseca de la Bella

supervised by

H. Steigerwald (CPT)

C. Marinoni (CPT)

21/02/2014

Aix*Marseille université

Outline

- This project is focused on dark energy models, more concretely on the Effective Field Theory of Dark Energy (EFT of DE) which is used to explore the space of modified gravity models able to explain the current accelerating expansion phase of the Universe.

- F. Piazza, H. Steigerwald and C. Marinoni, "Phenomenology of dark energy : exploring the space of theories with future redshift surveys", arXiv: 1312.6111v1 [astro-ph.CO]

- Interactive Data Language (IDL) has been used to implement functions and routines.

Table of Contents

EFT of DE

In the framework of a modification of gravity, DE causes effects as much to the **background evolution** (e.g. the *Hubble rate* $H(t)$) as to the **cosmological perturbations level** (e.g. the *growth rate* $f(t)$ of the large structures).

- Advantages of EFT of DE
  - EFT of DE parametrizes theories themselves in terms of *structural functions* in terms of characteristic coefficients which can be effectively constrained by future cosmological observations.
  - It allows us to work not only with one theory but with a *space of theories*.
  - It provides a clean separation between the background and the perturbation sectors.
- Drawback of EFT of DE : observations should have enough power to fix continuous functions of time.

Task

In the studied paper, the authors proposed a specific parametrization of the structural functions.

However, most of the conclusions of the mentioned paper are based on this choice.

The proposed method is desired to become a standard method of DE phenomenology.

Generalization of the parametrization

To compute,for the most general EFT parametrization, the growth of structure functions $f(z)\sigma_8(z)$, where $f(z)$ is the growth rate as a function of the redshift and $\sigma_8(z)$ is the variance.

Table of Contents

Interactive Data Language

- IDL is a computing environment for the interactive analysis and visualization of data.
- Programming in IDL is a time-saving alternative to programming in FORTRAN or C++.
- We can find implemented functions and programs (ready to use) but also we can create our own functions and programs.

	FUNCTION	PROGRAM

CODE	function test_func, b
	;core of the function
	c=exp(b)
	return, c
	end
		pro test_pro, a
		f=test_func(a)
		if f gt 1000 then begin
		print, 'the # is larger than 1000'
		endif else if a lt 10 then begin
		print, 'the # is lower than 1000'
		t=f*100
		print, t
		endif
		end

HOW TO CALL (IDL shell / code)	IDL>.r test_func.pro % Compiled module: TEST_FUNC. IDL>f=test_func(5) IDL>print, f 148.413	IDL>.r test_pro.pro % Compiled module: TEST_PRO. IDL>test_pro,5 the # is lower than 1000 14841.3

Table of Contents

General parametrization

The structural functions are reconstructed via a set of randomly generated polynomials.

Time dependent couplings in action	Dimensionless free functions	Parameterization
$\lambda(t)$, $C(t)$, $\mu(t)$	$\bar{\omega}(x)$	$= \bar{\omega} \equiv const.$
	$\eta(x)$	Solution of differential equation for ω
$\mu_3(t)$	$\eta_3(x)$	$= \sum_{i=0}^{n-1} \eta_{3i}(1-x)^i/i!$
$\varepsilon_4(t)$	$\eta_4(x)$	$= \sum_{i=0}^{n-1} \eta_{4i}(1-x)^i/i!$
	ω	$= \bar{\omega} + \sum_{i=1}^{n-1} \omega_i(x-xo)^i/i!$

where x denotes the physical matter density parameter of the fiducial model, $\overline{\Omega}_m$, $x_0 = \Omega_{m,0}$ at the present time, and ω and $\overline{\omega}$ are the state equation parameter and the one of the fiducial model, respectively.

Taylor expansion

```
function eft_eta3,
om,omegam,wo,alpha=alpha,beta=beta,eta3=eta3,model=model,derivative=derivative

if not keyword_set(model) then model='wabeta2eta3eta4'

if not keyword_set(derivative) then derivative=0

case model of

'wbarcst': begin

  common eft_random,eta3ran,eta4ran,wran

  case derivative of

  0:res=taylor(eta3ran,1.d0-om)

  1:if n_elements(eta3ran) eq 1 then res=dblarr(n_elements(om)) else res=-
taylor(eta3ran[1:n_elements(eta3ran)-1],1.d0-om)

  else: stop

  endcase

return,res

end
```

Taylor expansion

```
function eft_eta3,
om,omegam,wo,alpha=a

if not keyword_set(mode

if not keyword_set(deriv

case model of

'wbarcst': begin

  common eft_random,e

  case derivative of

  0:res=taylor(eta3ran,1.

  1:if n_elements(eta3ran) eq 1 then res=dblarr(n_elements(om)) else res=-
taylor(eta3ran[1:n_elements(eta3ran)-1],1.d0-om)

  else: stop

  endcase

return,res

end
```

```
pro set_eft, wo=wo

common eft_random,eta3ran,eta4ran,wran

  n=2

  eta3ran=2.d0*randomu(systime,n,/double)-1.d0

  eta4ran=2.d0*randomu(systime,n,/double)-1.d0

  wo=-1.d0

  wran=[wo,(0.5d0+0.5d0)*randomu(systime,n,/double)-
0.5d0]

end
```

Taylor expansion

```
function eft_eta3,
om,omegam,wo,alpha=alpha,beta=beta,eta3=eta3,model=model,derivative=derivative

if not keyword_set(model) then model='wabeta2eta3eta4'

if not keyword_set(derivative) then derivative=0

case model of

'wbarcst': begin

  common eft_random,eta3ran,eta4ran,wran

  case derivative of

  0:res=taylor(eta3ran,1.d0-om)

  1:if n_elements(eta3ran) eq 1 then res=dblarr(n_elements(om)) else res=-
taylor(eta3ran[1:n_elements(eta3ran)-1],1.d0-om)

  else: stop

  endcase

return,res

end
```

Taylor expansion

```
function eft_eta3,
om,omegam,wo,alpha=al
if not keyword_set(model
if not keyword_set(deriva
case model of
'wbarcst': begin
  common eft_random,et
  case derivative of
  0:res=taylor(eta3ran,1.c
  1:if n_elements(eta3ran
taylor(eta3ran[1:n_eleme
  else: stop
  endcase
return,res
end
```

```
function taylor, n=n,c,x
; c coefficient vector,  x basis vector,  j vector for indices
if not keyword_set(n) then n=n_elements(c)
m=n_elements(x)
j=dindgen(n)
k=transpose(rebin(j,n,m))
z=rebin(x,m,n)
v=c/factorial(j)
u=transpose(rebin(v,n,m))
;vector whose elements are the taylor terms
y=u*z^k
;taylor sum
if size(y,/n_dimensions) eq 1 then res=y else res=total(y,2)
return,res
end
```

Taylor expansion

```
function eft_eta3,
om,omegam,wo,alpha=alpha,beta=beta,eta3=eta3,model=model,derivative=derivative

if not keyword_set(model) then model='wabeta2eta3eta4'

if not keyword_set(derivative) then derivative=0

case model of

'wbarcst': begin

  common eft_random,eta3ran,eta4ran,wran

  case derivative of

  0:res=taylor(eta3ran,1.d0-om)

  1:if n_elements(eta3ran) eq 1 then res=dblarr(n_elements(om)) else res=-
taylor(eta3ran[1:n_elements(eta3ran)-1],1.d0-om)

  else: stop

  endcase

return,res

end
```

Table of Contents

The main program is devoted to the plot of the growth of structure functions. All the functions and programs created before are essential here.

```
pro plot_fsigma8, "arguments & keywords"

;only the main part of the code is explained here

;          T H E O R Y

if plottheory eq 1 then begin

zmin=0.d0

zmax=1.d0

n=100

z=dindgen(n)*(zmax-zmin)/double(n-1)+zmin

;          EFT

if size(where(codename_model eq 'wbarcst'),/dimensions) eq 1 then begin

set_params,'wbarcst',params=params,radiation=radiation,name=name,theory=theory,model=model,om
egam=omegam,omegal=omegal,omegar=omegar,ho=ho,sigma8=sigma8,omegab=omegab,wa=wa,ns=ns

 om=dindgen(100)*(0.9999d0-omegam)/99.d0+omegam

f=growth_rate(z,theory=theory,model=model,wo=wo,wa=wa,omegam=omegam,omegal=omegal,omega
r=omegar,method='numerical')
```

```
m=20
;create a table to be filled by the stable cases, n is the lenght of the redshift table
  fs8eft=dblarr(m,n)
  For i=0,m-1 do begin
  set_eft,wo=wo
  stable=eft_stability(om, omegam,wo,alpha=alpha,beta=beta,model=model,eta2=eta2, eta3=eta3, eta4=eta4)
  while stable eq 0 do begin
  set_eft,wo=wo
  stable=eft_stability(om, omegam,wo,alpha=alpha,beta=beta,model=model,eta2=eta2, eta3=eta3, eta4=eta4)
  endwhile
f=growth_rate(z,theory=theory,model=model,wo=wo,wa=wa,omegam=omegam,omegal=omegal,omegar=omegar,
method='numerical')
gf=growth_factor(z,ff=f,theory=theory,model=model,wo=wo,wa=wa,omegam=omegam,omegal=omegal,omegar=o
megar)
  fs8eft[i,*]=f*sigma8*gf
endfor
endif ; eft
```

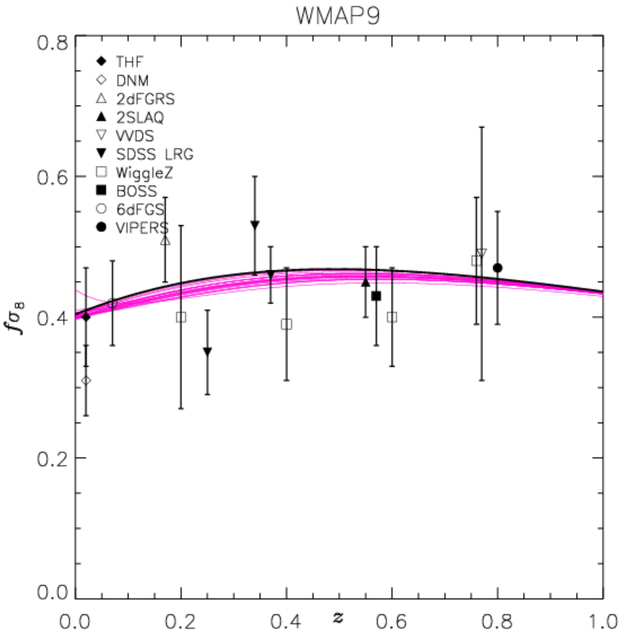
```
m=20

 ;create a table to be

  fs8eft=dblarr(m,n)

  For i=0,m-1 do beg

  set_eft,wo=wo

  stable=eft_stability

  while stable eq 0 d

  set_eft,wo=wo

  stable=eft_stability

  endwhile
f=growth_rate(z,the
method='numerical')

gf=growth_factor(z,ft=f,theory=theory,model=model,wo=wo,wa=wa,omegam=omegam,omegal=omegal,omegar=omegar)

  fs8eft[i,*]=f*sigma8*gf

endfor

endif ; eft
```

```
function eft_stability, om,
omegam,wo,alpha=alpha,beta=beta,model=model,eta2=eta2, eta3=eta3,
eta4=eta4

kinetic=eft_kinetic(om,omegam,wo,alpha=alpha,beta=beta,eta2=eta2,eta3=eta3,eta4=eta4,model=model)

gradient=eft_gradient(om,omegam,wo,alpha=alpha,beta=beta,eta3=eta3,eta4=eta4,model=model)

stable=where((kinetic ge 0) and (gradient ge 0),nstable,complement=instable)

if nstable eq n_elements(om) then res=1 else res=0

;stable=1, unstable=0

return, res

end
```

```
m=20
;create a table to be filled by the stable cases, n is the lenght of the redshift table
  fs8eft=dblarr(m,n)
  For i=0,m-1 do begin
  set_eft,wo=wo
  stable=eft_stability(om, omegam,wo,alpha=alpha,beta=beta,model=model,eta2=eta2, eta3=eta3, eta4=eta4)
  while stable eq 0 do begin
  set_eft,wo=wo
  stable=eft_stability(om, omegam,wo,alpha=alpha,beta=beta,model=model,eta2=eta2, eta3=eta3, eta4=eta4)
  endwhile
f=growth_rate(z,theory=theory,model=model,wo=wo,wa=wa,omegam=omegam,omegal=omegal,omegar=omegar,
method='numerical')
gf=growth_factor(z,ff=f,theory=theory,model=model,wo=wo,wa=wa,omegam=omegam,omegal=omegal,omegar=o
megar)
  fs8eft[i,*]=f*sigma8*gf
endfor
endif ; eft
```

```
;---------------------------------------------;



;      plot theory

if plottheory eq 1 then begin

; eft

if size(where(codename_model eq 'wbarcst'),/dimensions) eq 1 then begin

 For i=0,n_elements(fs8eft[*,0])-1 do begin

 oplot,z,fs8eft[i,*],color=6 ;thick=thick+1.

 endfor

  endif ;eft

  tek_color

  endif

endif ;plottheory
```

Table of Contents


WMAP9
THF
DNM
2dFGRS
2SLAQ
VVDS
SDSS LRG
WiggleZ
BOSS
6dFGS
VIPERS
$f\sigma_8$
z
0.0
0.2
0.4
0.6
0.8
1.0

THANK YOU FOR YOUR ATTENTION !

References :

[1] F. Piazza, H. Steigerwald and C. Marinoni, "Phenomenology of dark energy : exploring the space of theories with future redshift surveys", arXiv : 1312.6111v1 [astro-ph.CO].

[2] V. Mukhanov, "Physical foundations of cosmology", Cambridge University Press.

[3] E. Majerotto, L. Guzzo, J. Peacock et al., arXiv :1205.6215.