THE GALAXY SURVEY WITH THE SKA RESULTS FOR DIFFERENT SENSITIVITIES

SAHBA YAHYA

This results are obtained using these parameters, $H_0 = 73.0$, $c = 3 \times 10^5$, $\Omega_{m_0} = 0.24$, $\Omega_{K_0} = 0.0$, $\Omega_{\Lambda_0} = 0.76$, $w_0 = -1$, $w_a = 0$ This results marginalize the $\ln D_A(z)$ and $H(z)$ over $\{w_0, w_a, \Omega_m, \Omega_k, H_0\}$.

1. Results

- **Results for 0μJy sensitivity**

```
=== combined ===
 Vsur =   116.33371916142235       h^-3 Gpc^3
 Ngal =   1738478.9380518293       millions
 ngal =   14943.895463700861       10^-3 h^3 Mpc^-3
 Err[lnDa](%) =  0.13409
 Err[lnH](%)  =  0.29200
 r(lnDa,lnH)  =  0.40829
 Err[lnR](%)  =  0.09720

 *** Error on w***
 Err[w]            =  0.00426
 *** Error on w, with  Omega_m marginalized over ***
 Figure of Merit   2x2   =   25.759044210285058
 Err[w]            =  0.06983
      Err[wa]      =  0.42639
    Err[wa_w]      = -0.02448
     Err[Omega_m] %=      NaN
  Err[ln Omega_b] %=      NaN
     Err[Omega_m_w]= -0.00026
      Err[Omega_k]%=  4.39764
     Err[Omega_k_w]= -0.00217
```

Date: March 28, 2014.

```
      Err[H_0](%) =  1.33068

 << CMB Priors Added (Edit da_accuracy & om_accuracy in "add_cmb") >>
 Err[lnDa(z=1090)](%) =  0.20000
   Err[Omega_m](%)    =  2.00000
 *** Error on w***
 Err[w]            =  0.00415
 *** Error on w, with  Omega_m marginalized over ***
 Figure of Merit   2x2   =   26.795164050762256
 Err[w]            =  0.04433
      Err[wa]      =  0.39181
    Err[wa_w]      = -0.01540
     Err[Omega_m] %=  1.22075
  Err[ln Omega_b] %=  0.02377
     Err[Omega_m_w]=  0.00034
      Err[Omega_k]%=  0.70934
     Err[Omega_k_w]=  0.00017
       Err[H_0](%) =  0.47957
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$

ahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$ python FoM_Om
************************ Coe 2009 FoM DETF***********************************

FOM DETF (Coe2009) for OmuJy SKA =  9.56247594673
FoM DETF (Coe 2009) for OmuJy SKA + CMB =   20.1775166362

************************ SKA FoM ***********************************************

FoM of MOmJy 2x2 =  34.768519192
FoM of MOmJy_cmb  2x2 =  40.4546226351

*******************************************************************************
```

- **Results for 1μJy sensitivity**

```
=== combined ===
 Vsur =   116.33371916142235       h^-3 Gpc^3
 Ngal =   160925.53224535252       millions
 ngal =   1383.3094429144439       10^-3 h^3 Mpc^-3
 Err[lnDa](%) =  0.13424
 Err[lnH](%)  =  0.29223
 r(lnDa,lnH)  =  0.40830
```

```
 Err[lnR](%)  =  0.09730

 *** Error on w***
 Err[w]            =  0.00426
 *** Error on w, with  Omega_m marginalized over ***
 Figure of Merit   2x2   =   25.726334489912613
 Err[w]            =  0.06986
      Err[wa]      =  0.42663
    Err[wa_w]      = -0.02450
     Err[Omega_m] %=      NaN
  Err[ln Omega_b] %=      NaN
     Err[Omega_m_w]=  0.00348
      Err[Omega_k]%=  4.40178
     Err[Omega_k_w]= -0.00217
       Err[H_0](%) =  1.33135

 << CMB Priors Added (Edit da_accuracy & om_accuracy in "add_cmb") >>
 Err[lnDa(z=1090)](%) =  0.20000
   Err[Omega_m](%)    =  2.00000
 *** Error on w***
 Err[w]            =  0.00416
 *** Error on w, with  Omega_m marginalized over ***
 Figure of Merit   2x2   =   26.763623963042058
 Err[w]            =  0.04435
      Err[wa]      =  0.39205
    Err[wa_w]      = -0.01542
     Err[Omega_m] %=  1.22168
  Err[ln Omega_b] %=  0.02377
     Err[Omega_m_w]=  0.00034
      Err[Omega_k]%=  0.70988
     Err[Omega_k_w]=  0.00017
       Err[H_0](%) =  0.47990
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$ python FoM_1mJy_14
*********************** Coe 2009 FoM DETF**********************************

FOM DETF (Coe2009) for 1muJy SKA =  9.54949619831
FoM DETF (Coe 2009) for 1muJy SKA + CMB =   20.1734672793

************************ SKA FoM ***************************************************

FoM of 1 MOmJy 2x2 =  34.7222673388
FoM of 1 MOmJy_cmb  2x2 =  40.3992191386
```

```
*********************************************************************************
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$
```

- **Results for 7.3μJy senstivity**

```
=== combined ===
 Vsur =   116.33371916142235       h^-3 Gpc^3
 Ngal =   11788.925144398419       millions
 ngal =   101.33712933255700       10^-3 h^3 Mpc^-3
 Err[lnDa](%) =  0.13499
 Err[lnH](%)  =  0.29358
 r(lnDa,lnH)  =  0.40828
 Err[lnR](%)  =  0.09782

 *** Error on w***
 Err[w]            =  0.00428
 *** Error on w, with  Omega_m marginalized over ***
 Figure of Merit   2x2   =   25.529367943450627
 Err[w]            =  0.07002
      Err[wa]      =  0.42799
    Err[wa_w]      = -0.02462
     Err[Omega_m] %=      NaN
  Err[ln Omega_b] %=      NaN
     Err[Omega_m_w]= -0.00706
      Err[Omega_k]%=  4.42037
     Err[Omega_k_w]= -0.00219
       Err[H_0](%) =  1.33466

 << CMB Priors Added (Edit da_accuracy & om_accuracy in "add_cmb") >>
 Err[lnDa(z=1090)](%) =  0.20000
   Err[Omega_m](%)    =  2.00000
 *** Error on w***
 Err[w]            =  0.00418
 *** Error on w, with  Omega_m marginalized over ***
 Figure of Merit   2x2   =   26.572443203439267
 Err[w]            =  0.04444
      Err[wa]      =  0.39337
    Err[wa_w]      = -0.01550
     Err[Omega_m] %=  1.22674
  Err[ln Omega_b] %=  0.02377
     Err[Omega_m_w]=  0.00034
      Err[Omega_k]%=  0.71279
```

```
     Err[Omega_k_w]=  0.00017
       Err[H_0](%) =  0.48155
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$ python FoM_7point3
*********************** Coe 2009 FoM DETF**********************************

FOM DETF (Coe2009) for 7point3muJy SKA =  9.48586185491
FoM DETF (Coe 2009) for 7point3muJy SKA + CMB =   20.049268645

************************ SKA FoM ***************************************************

FoM of 7.3 MOmJy 2x2 =  34.4681421735
FoM of 7.3 MOmJy_cmb  2x2 =  40.0991553131

*************************************************************************************
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$
```

- **Results of 23μJy sensitivity**

```
=== combined ===
Vsur =   116.33371916142235       h^-3 Gpc^3
Ngal =   1106.0338963935974       millions
ngal =   9.5074231647222316       10^-3 h^3 Mpc^-3
Err[lnDa](%) =  0.14512
Err[lnH](%)  =  0.31470
r(lnDa,lnH)  =  0.40802
Err[lnR](%)  =  0.10511

*** Error on w***
Err[w]            =  0.00458
*** Error on w, with  Omega_m marginalized over ***
Figure of Merit   2x2   =   22.862327309799355
Err[w]            =  0.07236
     Err[wa]      =  0.45010
   Err[wa_w]      = -0.02649
    Err[Omega_m] %=*********
 Err[ln Omega_b] %=*********
    Err[Omega_m_w]=  0.00101
     Err[Omega_k]%=  4.72216
    Err[Omega_k_w]= -0.00243
      Err[H_0](%) =  1.38215

<< CMB Priors Added (Edit da_accuracy & om_accuracy in "add_cmb") >>
```

```
 Err[lnDa(z=1090)](%) =  0.20000
   Err[Omega_m](%)    =  2.00000
 *** Error on w***
 Err[w]             =  0.00445
 *** Error on w, with  Omega_m marginalized over ***
 Figure of Merit   2x2   =   24.038217192902810
 Err[w]             =  0.04566
      Err[wa]       =  0.41482
    Err[wa_w]       = -0.01668
     Err[Omega_m] %=  1.31689
  Err[ln Omega_b] %=  0.02377
     Err[Omega_m_w]=  0.00037
      Err[Omega_k]%=  0.76525
     Err[Omega_k_w]=  0.00019
       Err[H_0](%) =  0.50566
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$ python FoM_2
************************ Coe 2009 FoM DETF**********************************

FOM DETF (Coe2009) for 23 muJy SKA =  9.48586185491
FoM DETF (Coe 2009) for 23 muJy SKA + CMB =   20.049268645

************************* SKA FoM ***********************************************

FoM of 23 MOmJy 2x2 =  31.0642918259
FoM of 23 MOmJy_cmb  2x2 =  36.0851026084

*********************************************************************************
Sahbas-MacBook-Pro:w0_wa_marginlized_over_omega_K_Omega_m_h_Ob sahba$
```

2. Summary

The main subroutines that been used to produce this results are

```
!####################################################
!
! SUBROUTINES
!
!###################################################
```

TABLE 1. Different values of the $S_{\rm rms}$ and the corresponded values of σ_{w_a}, σ_{w_0} and FoM, marginalized over $\{\Omega_m, \Omega_K, H_0\}$

Uncertainties/ setups	$S_{\rm rms}$ (Jy)	σ_{w_0}	σ_{w_a}	$\sigma_{\Omega_m}\%$	σ_{Ω_b}	$\sigma_{\Omega_K}\%$	$\sigma_{H_0}\%$	FoM	DETF FoM
SKA	0 μ	0.06983	0.42639	-	-	4.39764	1.33068	127	99
	1 μ	0.06986	0.42663	-	-	4.40178	1.33135	124	96
	7.3 μ	0.07002	0.42799	-	-	4.42037	1.33466	112	85
	23 μ	0.07236	0.45010	-	-	4.72216	1.38215	55	35
SKA + CMB	0 μ	0.04433	0.39181	1.22075	2.01440	0.70934	0.47957	319	306
	1 μ	0.04435	0.39205	1.22168	2.01457	0.70988	0.47990	316	299
	7.3 μ	0.04444	0.39337	1.22674	2.01652	0.71279	0.48155	299	279
	23 μ	0.04566	0.41482	1.31689	2.1980	0.76525	0.50566	213	182

```
SUBROUTINE report_result(z,bias,npara,fis)
  IMPLICIT none
  integer, intent(IN) :: npara
  double precision, intent(IN) :: fis(npara,npara), z,bias
  double precision, allocatable, dimension(:,:) :: cov
  double precision, allocatable, dimension(:)   :: work
  double precision :: r12,err_lnda,err_lnh,err_lnR,beta, linear_pk,dgdlna,g
  integer :: i,j
  external linear_pk,dgdlna,g
  ALLOCATE(cov(2,2),work(2))
  cov=fis
  CALL DVERT(cov,2,2,work)
  beta=(1d0+dgdlna(z)/g(z))/bias
  r12=cov(1,2)/sqrt(cov(1,1)*cov(2,2))
  err_lnda=sqrt(cov(1,1))
  err_lnh=sqrt(cov(2,2))
  err_lnR=err_lnda*sqrt((1d0-r12**2d0) &
       /(1d0+2d0*r12*err_lnda/err_lnh+(err_lnda/err_lnh)**2d0))
  open(12,file='Fisher_1mJy_diff_14bins.txt' ,status='unknown')
  write(12,'(4F18.5)') err_lnda,err_lnh,err_lnR
  open(13,file='output_1mJy_diff_14bins.txt', status='unknown')
  write(13,'(5F18.5)') z, err_lnda*1d2,err_lnh*1d2,err_lnR*1d2, beta

  print'(1A15,1F9.5)','Err[lnDa](%) =',err_lnda*1d2
  print'(1A15,1F9.5)','Err[lnH](%)  =',err_lnh*1d2
  print'(1A15,1F9.5)','r(lnDa,lnH)  =',r12
  print'(1A15,1F9.5)','Err[lnR](%)  =',err_lnR*1d2
  DEALLOCATE(cov,work)
  return
```

```
END SUBROUTINE report_result

!###########################################


SUBROUTINE report_result3x3(fis)
  IMPLICIT none
  double precision, intent(IN) :: fis(5,5)
  double precision :: work(5),cov(5,5), A(5,5), M55DET, DET5x5,DET2x2,A05(5,5)
  integer :: i, j
  print*,'*** Error on w, ignoring Omega_k and Omega_m ***'
  print'(1A20,1F9.5)','Err[w]            =',1d0/dsqrt(fis(1,1))
  cov=fis
  A05=SQRT(fis)
  A = fis
  !###################### Calculate DET 5x5 #######################
  DET5x5 =  M55DET(SQRT(fis))
  DET2x2 =  A05(1,1)*A05(2,2) - A05(1,2)*A05(2,1)
  !open(14,file='Fis5x5_OmJy_diff_14bins.txt' ,status='unknown')
  !write(14,'(4F18.5)')  A
  !#######################################################


  CALL DVERT(cov,5,5,work)
  print*,'*** Error on w, with  Omega_m marginalized over ***'
 !print*,'The Fisher Matrix = ', '[', A(1,1) ,    A(1,2),  A(1,3) , A(1,4),  A(1,5), ';'
 !                                            , A(2,1) , A(2,2) ,  A(2,3) , A(2,4) ,  A(
 !                                            , A(3,1), A(3,2), A(3,3),A(3,4) ,A(3,5), '
 !                                             A(4,1), A(4,2),A(4,3),A(4,4),A(4,5) , ';'
 !                                           , A(5,1), A(5,2), A(5,3), A(5,4), A(5,5), '
! print*,'The Fisher Matrix **0.5 = ', '([[ ', A05(1,1) ,  ',',  A05(1,2), ',', A05(1,3)
 !                                            ',','[', A05(2,1) , ',', A05(2,2) ,  ',',A
  !                                           '[', A05(3,1),  ',',A05(3,2), ',',A05(3,3)
    !                                         '[', A05(4,1), ',',A05(4,2), ',',A05(4,3),
   !                                         '[', A05(5,1), ',',A05(5,2), ',',A05(5,3),

  print*, 'Figure of Merit   2x2   =' , abs(DET2x2)
! print*, 'Figure of Merit   5x5   =' , abs(DET5x5)
  print'(1A20,1F9.5)','Err[w]            =',dsqrt(cov(1,1))
  print'(1A20,1F9.5)','Err[wa]      =',dsqrt(cov(2,2))
  print'(1A20,1F9.5)','Err[wa_w]      =',(cov(1,2))
  !print'(1A20,1F9.5)','Err[wa_w]      =',(cov(2,1))
  print'(1A20,1F9.5)','Err[Omega_m](%) =',dsqrt(cov(5,5))*1d2
```

```
  print'(1A20,1F9.5)','Err[Omega_k](%) =',dsqrt(cov(3,3))*1d2
   print'(1A20,1F9.5)','Err[H_0](%) =',dsqrt(cov(4,4))*1d2
  return
END SUBROUTINE report_result3x3

!##########################################

SUBROUTINE transform_fisher(z,fisDH,fis3x3)
  USE cosmo
  ! transform 2x2 Fisher matrix for (Da,H) to 3x3 matrix for
  ! (w,Omega_k,ln(Omega_m)), and accumulate it [e.g., Eq.(30) of Shoji et al.]
  IMPLICIT none
  integer :: a,b,i,j
  double precision, intent(IN)    :: z,fisDH(2,2)
  double precision, intent(INOUT) :: fis3x3(5,5)
  double precision :: dpdq(5,2)
  double precision :: chi,h2,func0,func1,func2,func3,func4,rombint,fz
  external h2,func0,func1,func2,func3,func4,rombint
  chi=rombint(func0,0d0,z,1d-7)
  ! ORIGINAL CODE
  !dpdq(1,1)=-1.5d0*ode0*rombint(func1,0d0,z,1d-7)/chi         ! dlnDa/dw
  !dpdq(1,2)= 1.5d0*ode0*dlog(1d0+z)/h2(z)                     ! dlnH/dw
   !dpdq(2,1)=-1.5d0*ode0*rombint(func4,0d0,z,1d-7)/chi        ! dlnDa/dwa
  !dpdq(2,2)= 1.5d0*ode0*(dlog(1d0+z) - z/(1d0+ z))/h2(z)                         ! dlnH/dwa
  !dpdq(3,1)=-0.5d0*rombint(func2,0d0,z,1d-7)/chi+chi**2d0/6d0 ! dlnDa/dOmega_k
  !dpdq(3,2)= 0.5d0*(1d0+z)**2d0/h2(z)                         ! dlnH/dOmega_k
  !dpdq(5,1)=-0.5d0*om0*rombint(func3,0d0,z,1d-7)/chi          ! dlnDa/dln(Omega_m)
  !dpdq(5,2)= 0.5d0*om0*(1d0+z)**3d0/h2(z)                     ! dlnH/dln(Omega_m)
  ! MODYFIED CODE
   fz =(1d0 + z)**(3d0*(1d0 + w0+ w_a)) * exp(-3d0 * w_a *(z/(1d0+ z)))
  dpdq(1,1)=-1.5d0*ode0*rombint(func1,0d0,z,1d-7)/chi         ! dlnDa/dw
  dpdq(1,2)= 1.5d0*ode0*dlog(1d0+z)*fz/h2(z)                     ! dlnH/dw
  dpdq(2,1)=-1.5d0*ode0*rombint(func4,0d0,z,1d-7)/chi        ! dlnDa/dwa
  dpdq(2,2)= 1.5d0*ode0*(dlog(1d0+z) - z/(1d0+ z))*fz/h2(z)                        ! dlnH/dwa
  dpdq(3,1)=-0.5d0*rombint(func2,0d0,z,1d-7)/chi+chi**2d0/6d0! dlnDa/dOmega_k
  dpdq(3,2)= 0.5d0*((1d0+z)**2d0 - fz)/h2(z)                          ! dlnH/dOmega_k
  dpdq(4,1)= -1d0/H_0  !dlnDa/dH_0
  dpdq(4,2)= c/(2998d0*H_0)! dlnH/dH_0
  dpdq(5,1)=-0.5d0*rombint(func3,0d0,z,1d-7)/chi     ! dlnDa/d(Omega_m)
  dpdq(5,2)= 0.5d0*((1d0+z)**3d0 - fz)/h2(z)  ! dlnH/dOmega_m
  do a=1,5
     do b=1,5
        do i=1,2
```

```
          do j=1,2
             ! transform and accumulate fis3x3
             fis3x3(a,b)=fis3x3(a,b)+dpdq(a,i)*dpdq(b,j)*fisDH(i,j)
          enddo
       enddo
    enddo
  enddo
  return
END SUBROUTINE transform_fisher
!########################################################
SUBROUTINE add_cmb(fis3x3)
  USE cosmo
  ! Add the distance information from CMB [e.g., Eq.(38) of Shoji et al.]
  IMPLICIT none
  integer :: a,b,i,j
  double precision :: da_accuracy=0.2d0 ! Percent error in Da(zcmb)
  double precision :: om_accuracy=2d0   ! Percent error in Omega_m
  double precision, intent(INOUT) :: fis3x3(5,5)
  double precision :: dpdq(5),zcmb=1090d0
  double precision :: chi,h2,func0,func1,func2,func3,func4,rombint
  external h2,func0,func1,func2,func3,func4,rombint
  chi=rombint(func0,0d0,zcmb,1d-7)
  dpdq(1)=-1.5d0*ode0*rombint(func1,0d0,zcmb,1d-7)/chi         ! dlnDa/dw
  dpdq(2)=-1.5d0*ode0*rombint(func4,0d0,zcmb,1d-7)/chi         ! dlnDa/dwa
  dpdq(3)=-0.5d0*rombint(func2,0d0,zcmb,1d-7)/chi+chi**2d0/6d0 ! dlnDa/dOmega_k
  dpdq(4)= -1d0/H_0      ! dlnDa/dH_0
  dpdq(5)=-0.5d0*rombint(func3,0d0,zcmb,1d-7)/chi       ! dlnDa/dOmega_m



  !dpdq(5)=-0.5d0*rombint(func2,0d0,zcmb,1d-7)/chi+chi**2d0/6d0 !dlnDa/dh
   do a=1,5
    do b=1,5
       ! add Da(z=1090)
       fis3x3(a,b)=fis3x3(a,b)+dpdq(a)*dpdq(b)*1d4/da_accuracy**2d0
    enddo
  enddo
  ! add Omega_matter
  fis3x3(5,5)=fis3x3(5,5)+1d4/(om_accuracy*om0)**2d0
  print*,'<< CMB Priors Added (Edit da_accuracy & om_accuracy in "add_cmb") >>'
  print'(1A23,1F9.5)','Err[lnDa(z=1090)](%) =',da_accuracy
  print'(1A23,1F9.5)','Err[Omega_m](%)    =',om_accuracy
  return
```

```
END SUBROUTINE add_cmb
!###############################################################
!####################################################
! Functions
!###################################################
DOUBLE PRECISION FUNCTION h2(redshift)
  USE cosmo
  ! h2(z) = Omega_matter(1+z)^3+Omega_lambda
  IMPLICIT none
  DOUBLE PRECISION, intent(IN) :: redshift
  DOUBLE PRECISION :: fz,ok0
    fz =(1d0 + redshift)**(3d0*(1d0 + w0+ w_a)) * exp(-3d0 * w_a *(redshift/(1d0+ redshift)))
  h2 = om0*(1d0+redshift)**3d0+ ok0 * (1d0+redshift)**2d0+ode0* fz
  return
END FUNCTION h2
DOUBLE PRECISION FUNCTION func0(redshift)
  USE cosmo
  ! func0(z) = 1/[h2(z)]^0.5
  IMPLICIT none
  DOUBLE PRECISION, intent(IN) :: redshift
  DOUBLE PRECISION :: h2
  external :: h2
  func0 = 1d0/dsqrt(h2(redshift))
  return
END FUNCTION func0
DOUBLE PRECISION FUNCTION func1(redshift)
  USE cosmo
  ! func1(z) = ln(1+z)/[h2(z)]^1.5
  IMPLICIT none
  DOUBLE PRECISION, intent(IN) :: redshift
  DOUBLE PRECISION :: h2, fz
  external :: h2
  fz =(1d0 + redshift)**(3d0*(1d0 + w0+ w_a)) * exp(-3d0 * w_a *(redshift/(1d0+ redshift)))
  func1 = dlog(1d0+redshift) * fz /h2(redshift)**1.5d0
  return
END FUNCTION func1
DOUBLE PRECISION FUNCTION func2(redshift)
  USE cosmo
  ! func2(z) = (1+z)^2/[h2(z)]^1.5
  IMPLICIT none
  DOUBLE PRECISION, intent(IN) :: redshift
  DOUBLE PRECISION :: h2,fz
  external :: h2
```

```
   fz =(1d0 + redshift)**(3d0*(1d0 + w0+ w_a)) * exp(-3d0 * w_a *(redshift/(1d0+ redshif
  func2 =( (1d0+redshift)**2d0 - fz)/h2(redshift)**1.5d0
  return
END FUNCTION func2
DOUBLE PRECISION FUNCTION func3(redshift)
  USE cosmo
  ! func3(z) = (1+z)^3/[h2(z)]^1.5
  IMPLICIT none
  DOUBLE PRECISION, intent(IN) :: redshift
  DOUBLE PRECISION :: h2, fz
  external :: h2
   fz =(1d0 + redshift)**(3d0*(1d0 + w0+ w_a)) * exp(-3d0 * w_a *(redshift/(1d0+ redshif
  func3 =( (1d0+redshift)**3d0 - fz) /h2(redshift)**1.5d0
  return
END FUNCTION func3

DOUBLE PRECISION FUNCTION func4(redshift)
  USE cosmo
  ! func4(z) = ln(1+z) - z/1+z /[h2(z)]^1.5
  IMPLICIT none
  DOUBLE PRECISION, intent(IN) :: redshift
  DOUBLE PRECISION :: h2, fz
  external :: h2
  fz =(1d0 + redshift)**(3d0*(1d0 + w0+ w_a)) * exp(-3d0 * w_a *(redshift/(1d0+ redshift
  func4 = fz * (dlog(1d0+redshift) - (redshift/(1d0+ redshift)))/h2(redshift)**1.5d0
  return
END FUNCTION func4
```