

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1

0001	! Modules used by cmbmain
0002	
0003	! Code for Anisotropi
0004	! by Antony Lewis (ht
0005	! See readme.html for
0006	!
0007	! Based on CMBFAST b
0008	! on Boltzmann code w
0009	! Original CMBFAST co
0010	!
0011	! Copyright 1996 by H
0012	! the Massachusetts I
0013	!
0014	! THIS SOFTWARE IS PR
0015	! REPRESENTATIONS OR
0016	! By way of example,
0017	! M.I.T. AND C.f.A MA
0018	! MERCHANTABILITY OR
0019	! THE USE OF THE LICE
0020	! ANY THIRD PARTY PAT
0021	!
0022	! portions of this so
0023	! E. Bertschinger. S
0024	! for restrictions on
0025	
0026	
0027	module ModelParams
0028	use precision
0029	use Ranges
0030	use InitialPower

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1

0001	! Modules used by cmbmain
0002	
0003	! Code for Anisotropi
0004	! by Antony Lewis (ht
0005	! See readme.html for
0006	!
0007	! Based on CMBFAST b
0008	! on Boltzmann code w
0009	! Original CMBFAST co
0010	!
0011	! Copyright 1996 by H
0012	! the Massachusetts I
0013	!
0014	! THIS SOFTWARE IS PR
0015	! REPRESENTATIONS OR
0016	! By way of example,
0017	! M.I.T. AND C.f.A MA
0018	! MERCHANTABILITY OR
0019	! THE USE OF THE LICE
0020	! ANY THIRD PARTY PAT
0021	!
0022	! portions of this so
0023	! E. Bertschinger. S
0024	! for restrictions on
0025	
0026	
0027	module ModelParams
0028	use precision
0029	use Ranges
0030	use InitialPower

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 31

```
0031      use Reionization
0032      use Recombination
0033      use Errors
0034
0035      implicit none
0036      public
0037
0038      | character(LEN=*), parameter
0039
0040      integer :: FeedbackLevel
0041
0042
0042      logical, parameter :: Deb
0043
0044      logical, parameter :: Deb
0045
0046      real(dl) :: DebugParam =
0047
0048      logical :: do_bispectrum
0049      logical, parameter :: har
0050
0051      logical, parameter :: ful
0052
0053      integer, parameter :: Nu_
0054      !For CAMBparams%MassiveNu
0055      !Nu_int: always integrate
0056      !Nu_trunc: switch to expa
0057      !Nu_approx: approximate s
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 31

```
0031      use Reionization
0032      use Recombination
0033      use Errors
0034
0035      implicit none
0036      public
0037
0038      | character(LEN=*), parameter
0039
0040      integer :: FeedbackLevel
0041
0042      logical :: output_file_he
0043
0044
0045      logical, parameter :: Deb
0046
0047      logical, parameter :: Deb
0048
0049      real(dl) :: DebugParam =
0050
0051      logical :: do_bispectrum
0052      logical, parameter :: har
0053
0054      logical, parameter :: ful
0055
0056      integer, parameter :: Nu_
0057      !For CAMBparams%MassiveNu
0058      !Nu_int: always integrate
0059      !Nu_trunc: switch to expa
0060      !Nu_approx: approximate s
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 58

```
0058      !Nu_best: automatically u
0059
0060      integer, parameter :: max
0061      integer, parameter :: max
0062      integer, parameter :: fil
0063      integer, parameter :: out
0064
0065      !#SimDataAdd
0066      integer, parameter :: nTo
0067      integer, parameter :: cl_
0068      integer, parameter :: cl_
0069      !#SimDataAdd
0070
0071      integer :: max_bessels_l_
0072      real(dl) :: max_bessels_e
0073
0074      real(dl), parameter :: 0
0075      !When using outNone the o
0076
0077      Type(Regions) :: TimeStep
0078
0079      type TransferParams
0080          logical :: high_
0081
0082          integer :: num_r
0083          real(dl) :: kmax
0084          integer :: k_per
0085          real(dl) :: redsh
0086          !JD 08/13 Added so bo
0087          real(dl) :: PK_re
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 61

```
0061      !Nu_best: automatically u
0062
0063      integer, parameter :: max
0064      integer, parameter :: max
0065      integer, parameter :: fil
0066      integer, parameter :: out
0067
0068      integer :: max_bessels_l_
0069      real(dl) :: max_bessels_e
0070
0071      real(dl), parameter :: 0
0072      !When using outNone the o
0073
0074      Type(Regions) :: TimeStep
0075
0076      type TransferParams
0077          logical :: high_
0078          logical :: accur
0079          integer :: num_r
0080          real(dl) :: kmax
0081          integer :: k_per
0082          real(dl) :: redsh
0083          !JD 08/13 Added so bo
0084          real(dl) :: PK_re
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 87

```
0087      real(dl)      :: NLL_r
0088      integer       :: PK_re
0089      integer       :: NLL_r
0090      integer       :: PK_nu
0091      integer       :: NLL_n
0092
0093      end type TransferParams
0094
0095      !other variables, options
0096
0097      integer, parameter :: Non
0098      integer, parameter :: Non
0099
0100      ! Main parameters type
0101      type CAMBparams
0102
0103          logical      :: WantCls,
0104          logical      :: WantScal
0105          logical      :: DoLensing
0106          logical      :: want_zst
0107          logical      :: PK_WantT
0108          integer      :: NonLinea
0109          logical      :: Want_CMB
0110
0111      !#SimDataAdd
0112          logical      :: DoCS, DoGal,
0113          logical      :: OutputSimDat
0114          logical      :: sim_random_c
0115          integer      :: sim_random_s
0116          logical      :: DoShePowFoc
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 85

```
0085      real(dl)      :: NLL_r
0086      integer       :: PK_re
0087      integer       :: NLL_r
0088      integer       :: PK_nu
0089      integer       :: NLL_n
0090
0091      end type TransferParams
0092
0093      !other variables, options
0094
0095      integer, parameter :: Non
0096      integer, parameter :: Non
0097
0098      ! Main parameters type
0099      type CAMBparams
0100
0101          logical      :: WantCls,
0102          logical      :: WantScal
0103          logical      :: DoLensing
0104          logical      :: want_zst
0105          logical      :: PK_WantT
0106          integer      :: NonLinea
0107          logical      :: Want_CMB
0108
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 117

```
0117      logical      :: CSGalCosmoMC
0118      logical      :: doDvir = .fa
0119      !#SimDataAdd
0120
0121      integer       :: Max_l, M
0122      real(dl)      :: Max_eta_
0123      !_tensor settings on
0124      !Max_l and Max_eta_k
0125
0126      real(dl)      :: omegab,
0127      !Omega baryon, CDM, L
0128      real(dl)      :: H0,TCMB,
0129      integer       :: Num_Nu_m
0130      integer       :: Nu_mass_
0131      logical       :: share_de
0132      real(dl)      :: Nu_mass_
0133      real(dl)      :: Nu_mass_
0134      integer       :: Nu_mass_
0135
0136
0137      !#SimDataAdd
0138      integer       :: nTomoBin(1:2
0139      real(dl)      :: zph_low(1:nT
0140      real(dl)      :: zph_high(1:n
0141      integer       :: lmax_CS, lma
0142      real(dl)      :: fskycmb
0143      integer       :: sim_ncmbcls,
0144      real(dl)      :: photo_error
0145      real(dl)      :: nz_z0_lss,nz
0146      real(dl)      :: mean_int_ell
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 109

```
0109      integer       :: Max_l, M
0110      real(dl)      :: Max_eta_
0111      !_tensor settings on
0112      !Max_l and Max_eta_k
0113
0114      real(dl)      :: omegab,
0115      !Omega baryon, CDM, L
0116      real(dl)      :: H0,TCMB,
0117      integer       :: Num_Nu_m
0118      integer       :: Nu_mass_
0119      logical       :: share_de
0120      real(dl)      :: Nu_mass_
0121      real(dl)      :: Nu_mass_
0122      integer       :: Nu_mass_
0123
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 147

```
0147      real(dl)      :: cs2_de
0148      integer       :: cl_zbins,cl_
0149      real(dl)      :: ngal_cl,fsky
0150      real(dl)      :: nz_z0_cl,nz_
0151      real(dl)      :: c_nfw,theta_
0152      real(dl)      :: cl_taui = 2.
0153      real(dl)      :: cl_M = 1.d16
0154      real(dl)      :: cl_RedshiftB
0155      real(dl), dimension(:), a
0156      !#SimDataAdd
```

```
0157
0158      integer       :: Scalar_i 0124
0159      !must be one of the i 0125
0160      0126
0161      integer       :: OutputNo 0127
0162      !outNone, or C_Output 0128
0163      0129
0164      logical       :: Accurate 0130
0165      !Do you care about th 0131
0166      0132
0167      logical       :: Accurate 0133
0168      !Do you care about BB 0134
0169      0135
0170      !Reionization setting 0136
0171      logical       :: Accurate 0137
0172      !Do you care about pe 0138
0173      0139
0174      integer       :: MassiveN 0140
0175      0141
0176      type(InitialPowerPara 0142
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 124

```
integer       :: Scalar_i
!must be one of the i

integer       :: OutputNo
!outNone, or C_Output

logical       :: Accurate
!Do you care about th

logical       :: Accurate
!Do you care about BB

!Reionization setting
logical       :: Accurate
!Do you care about pe

integer       :: MassiveN

type(InitialPowerPara
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 177

```
0177      type(ReionizationPara
0178      type(RecombinationPar
0179      type(TransferParams)
0180
0181      real(dl) :: InitialC
0182      !ignored unless Scala
0183
0184      logical OnlyTransfers
0185      !If true, sigma_8 is
0186
0187      logical DerivedParame
0188
0189      !Derived parameters,
0190      type(ReionizationHist
0191
0192      logical flat,closed,o
0193      real(dl) omegak
0194      real(dl) curv,r, Ksig
0195      real(dl) tau0,chi0 !t
0196
0197      end type CAMBparams
0198
0199      type(CAMBparams), save ::
0200
0201      real(dl) scale !relative
0202
0203      logical ::call_again = .f
0204      !if being called again wi
0205
0206      !      grhom =kappa*a^2*rh
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 143

```
0143      type(ReionizationPara
0144      type(RecombinationPar
0145      type(TransferParams)
0146
0147      real(dl) :: InitialC
0148      !ignored unless Scala
0149
0150      logical OnlyTransfers
0151      !If true, sigma_8 is
0152
0153      logical DerivedParame
0154
0155      !Derived parameters,
0156      type(ReionizationHist
0157
0158      logical flat,closed,o
0159      real(dl) omegak
0160      real(dl) curv,r, Ksig
0161      real(dl) tau0,chi0 !t
0162
0163      end type CAMBparams
0164
0165      type(CAMBparams), save ::
0166
0167      real(dl) scale !relative
0168
0169      logical ::call_again = .f
0170      !if being called again wi
0171
0172      !      grhom =kappa*a^2*rh
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 207

```
0207      !      grhornomass=grhor*n
0208      !      taurst,taurend - ti
0209      !      dtaurec - dtau duri
0210      !      adotrad - a(tau) in
0211
0212      real(dl) grhom,grhog,grho
0213      real(dl) taurst,dtaurec,t
0214
0215      !Neutrinos
0216      real(dl) grhormass(max_nu
0217
0218      !      nu_masses=m_nu*c**2
0219      real(dl) :: nu_masses(max
0220
0221      real(dl) akthom !sigma_T
0222      real(dl) fHe !n_He_tot /
0223      real(dl) Nnow
0224
0225
0226      integer :: ThreadNum = 0
0227      !If zero assigned automat
0228
0229      !Parameters for checking/
0230      !If HighAccuracyDefault=.
0231      !If HighAccuracyDefault=.
0232      logical :: HighAccuracyDe
0233
0234      real(dl) :: lSampleBoost=
0235      !Increase lSampleBoost to
0236
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 173

```
0173      !      grhornomass=grhor*n
0174      !      taurst,taurend - ti
0175      !      dtaurec - dtau duri
0176      !      adotrad - a(tau) in
0177
0178      real(dl) grhom,grhog,grho
0179      real(dl) taurst,dtaurec,t
0180
0181      !Neutrinos
0182      real(dl) grhormass(max_nu
0183
0184      !      nu_masses=m_nu*c**2
0185      real(dl) :: nu_masses(max
0186
0187      real(dl) akthom !sigma_T
0188      real(dl) fHe !n_He_tot /
0189      real(dl) Nnow
0190
0191
0192      integer :: ThreadNum = 0
0193      !If zero assigned automat
0194
0195      !Parameters for checking/
0196      !If HighAccuracyDefault=.
0197      !If HighAccuracyDefault=.
0198      logical :: HighAccuracyDe
0199
0200      real(dl) :: lSampleBoost=
0201      !Increase lSampleBoost to
0202
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 237

```
0237      real(dl) :: AccuracyBoost
0238
0239      !Decrease step sizes, etc
0240      !Can also be used to impr
0241      !or improving accuracy fo
0242      !Note this does not incre
0243
0244      real(sp) :: lAccuracyBoos
0245      !Boost number of multipol
0246
0247      integer :: limber_phiphi
0248      integer :: num_redshiftwi
0249      integer :: num_extra_reds
0250
0251      integer, parameter :: lmi
0252      !must be either 1 or 2
0253
0254      real(dl), parameter :: Om
0255
0256      real(dl), parameter :: tol
0257
0258      !      used as parameter f
0259      real(dl), parameter :: sp
0260
0261      integer, parameter :: l0ma
0262
0263      !      lmax is max possibl
0264      integer, parameter :: lma
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 203

```
0203      real(dl) :: AccuracyBoost
0204
0205      !Decrease step sizes, etc
0206      !Can also be used to impr
0207      !or improving accuracy fo
0208      !Note this does not incre
0209
0210      real(sp) :: lAccuracyBoos
0211      !Boost number of multipol
0212
0213      integer :: limber_phiphi
0214      integer :: num_redshiftwi
0215      integer :: num_extra_reds
0216      integer :: num_custom_sou
0217      integer, allocatable :: c
0218
0219      integer, parameter :: lmi
0220      !must be either 1 or 2
0221
0222      real(dl), parameter :: Om
0223
0224      real(dl), parameter :: tol
0225
0226      !      used as parameter f
0227      real(dl), parameter :: sp
0228
0229      integer, parameter :: l0ma
0230
0231      !      lmax is max possibl
0232      integer, parameter :: lma
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 265

```
0265
0266      character(LEN=1024) :: hi
0267      !fiducial high-accuracy h
0268      !to lensing and C_L inter
0269      logical :: use_spline_tem
0270      integer, parameter :: lma
0271      real(dl), allocatable ::
0272
0273      integer, parameter :: der
0274          derived_zdrag=6, deri
0275          derived_thetaEQ=12, d
0276      integer, parameter :: nth
0277
0278      real(dl) ThermoDerivedPar
0279
0280      Type TBackgroundOutputs
0281          real(dl), pointer ::
0282          real(dl), allocatable
0283      end Type TBackgroundOutpu
0284
0285      Type(TBackgroundOutputs),
0286
0287      contains
0288
0289
0290      subroutine CAMBParams_Set
0291      use constants
0292      type(CAMBparams), intent(
0293      real(dl) GetOmegak, fract
0294      integer, optional :: erro
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 233

```
0233
0234      character(LEN=1024) :: hi
0235      !fiducial high-accuracy h
0236      !to lensing and C_L inter
0237      logical :: use_spline_tem
0238      integer, parameter :: lma
0239      real(dl), allocatable ::
0240
0241      integer, parameter :: der
0242          derived_zdrag=6, deri
0243          derived_thetaEQ=12, d
0244      integer, parameter :: nth
0245
0246      real(dl) ThermoDerivedPar
0247
0248      Type TBackgroundOutputs
0249          real(dl), pointer ::
0250          real(dl), allocatable
0251      end Type TBackgroundOutpu
0252
0253      Type(TBackgroundOutputs),
0254
0255      contains
0256
0257
0258      subroutine CAMBParams_Set
0259      use constants
0260      type(CAMBparams), intent(
0261      real(dl) GetOmegak, fract
0262      integer, optional :: erro
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 295

```
0295      logical, optional :: DoRe
0296      logical WantReion
0297      integer nu_i,actual_mass1
0298      real(dl) nu_massless_dege
0299      external GetOmegak
0300      real(dl), save :: last_ta
0301      !Constants in SI units
0302
0303      global_error_flag = 0
0304
0305      if ((P%WantTensors .or. P
0306          call GlobalError( 'Ca
0307      end if
0308
0309      if (present(error)) error
0310      if (global_error_flag/=0)
0311
0312      if (present(DoReion)) the
0313          WantReion = DoReion
0314      else
0315          WantReion = .true.
0316      end if
0317
0318      CP=P
0319      if (call_again) CP%Derive
0320
0321      CP%Max_eta_k = max(CP%Max
0322
0323      if (CP%WantTransfer) then
0324          CP%WantScalars=.true.
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 263

```
0263      logical, optional :: DoRe
0264      logical WantReion
0265      integer nu_i,actual_mass1
0266      real(dl) nu_massless_dege
0267      external GetOmegak
0268      real(dl), save :: last_ta
0269      !Constants in SI units
0270
0271      global_error_flag = 0
0272
0273      if ((P%WantTensors .or. P
0274          call GlobalError( 'Ca
0275      end if
0276
0277      if (present(error)) error
0278      if (global_error_flag/=0)
0279
0280      if (present(DoReion)) the
0281          WantReion = DoReion
0282      else
0283          WantReion = .true.
0284      end if
0285
0286      CP=P
0287      if (call_again) CP%Derive
0288
0289      CP%Max_eta_k = max(CP%Max
0290
0291      if (CP%WantTransfer) then
0292          CP%WantScalars=.true.
```

/Users/lplopa/Compare/camb_simdata/modules.f90, Top line: 325

```
0325         if (.not. CP%WantCls)
0326             CP%AccuratePolari
0327             CP%Reion%Reioniza
0328         end if
0329     else
0330         CP%transfer%num_redsh
0331     end if
0332
0333     if (CP%Num_Nu_Massive /=
0334         if (sum(CP%Nu_mass_nu
0335     end if
0336     if (CP%Omegan == 0 .and.
0337         if (CP%share_delta_ne
0338             CP%Num_Nu_Massles
0339         else
0340             CP%Num_Nu_Massles
0341         end if
0342         CP%Num_Nu_Massive =
0343         CP%Nu_mass_numbers =
0344     end if
0345
0346     nu_massless_degeneracy =
0347     if (CP%Num_nu_massive > 0
0348         if (CP%Nu_mass_eigens
0349         if (CP%Nu_mass_eigens
0350         if (all(CP%Nu_mass_nu
0351         if (CP%share_delta_ne
0352             !default case of
0353             fractional_number
0354             actual_massless =
```

/Users/lplopa/Compare/camb_des/modules.f90, Top line: 293

```
0293         if (.not. CP%WantCls)
0294             CP%AccuratePolari
0295             CP%Reion%Reioniza
0296         end if
0297     else
0298         CP%transfer%num_redsh
0299     end if
0300
0301     if (CP%Num_Nu_Massive /=
0302         if (sum(CP%Nu_mass_nu
0303     end if
0304     if (CP%Omegan == 0 .and.
0305         if (CP%share_delta_ne
0306             CP%Num_Nu_Massles
0307         else
0308             CP%Num_Nu_Massles
0309         end if
0310         CP%Num_Nu_Massive =
0311         CP%Nu_mass_numbers =
0312     end if
0313
0314     nu_massless_degeneracy =
0315     if (CP%Num_nu_massive > 0
0316         if (CP%Nu_mass_eigens
0317         if (CP%Nu_mass_eigens
0318         if (all(CP%Nu_mass_nu
0319         if (CP%share_delta_ne
0320             !default case of
0321             fractional_number
0322             actual_massless =
```

/Users/lplopa/Compare/camb_simdata/modules.f90, Top line: 355

```
0355         neff_i = fraction
0356         nu_massless_degen
0357         CP%Nu_mass_degene
0358     end if
0359     if (abs(sum(CP%Nu_mas
0360         stop 'Nu_mass_fra
0361 else
0362     CP%Nu_mass_eigenstate
0363 end if
0364
0365 if ((CP%WantTransfer).and
0366     CP%MassiveNuMethod =
0367 end if
0368
0369 CP%omegak = GetOmegak()
0370
0371 CP%flat = (abs(CP%omegak)
0372 CP%closed = CP%omegak < -
0373
0374 CP%open = .not.CP%flat.an
0375 if (CP%flat) then
0376     CP%curv=0
0377     CP%Ksign=0
0378     CP%r=1._dl !so we can
0379 else
0380     CP%curv=-CP%omegak/((
0381     CP%Ksign =sign(1._dl,
0382     CP%r=1._dl/sqrt(abs(C
0383 end if
0384 ! grho gives the contrib
```

/Users/lplopa/Compare/camb_des/modules.f90, Top line: 323

```
0323         neff_i = fraction
0324         nu_massless_degen
0325         CP%Nu_mass_degene
0326     end if
0327     if (abs(sum(CP%Nu_mas
0328         call MpiStop('Nu_
0329 else
0330     CP%Nu_mass_eigenstate
0331 end if
0332
0333 if ((CP%WantTransfer).and
0334     CP%MassiveNuMethod =
0335 end if
0336
0337 CP%omegak = GetOmegak()
0338
0339 CP%flat = (abs(CP%omegak)
0340 CP%closed = CP%omegak < -
0341
0342 CP%open = .not.CP%flat.an
0343 if (CP%flat) then
0344     CP%curv=0
0345     CP%Ksign=0
0346     CP%r=1._dl !so we can
0347 else
0348     CP%curv=-CP%omegak/((
0349     CP%Ksign =sign(1._dl,
0350     CP%r=1._dl/sqrt(abs(C
0351 end if
0352 ! grho gives the contrib
```

/Users/lplopa/Compare/camb_simdata/modules.f90, Top line: 385

```
0385      ! (r) one flavor of rela
0386      ! (m) nonrelativistic ma
0387      ! 8*pi*G*rho/c^2 at a=1,
0388      ! a=tau(Mpc)*adotrad, wi
0389      ! (Used only to set the
0390
0391      !H0 is in km/s/Mpc
0392
0393      grhom = 3*CP%h0**2/c**2*1
0394
0395      !grhom=3.3379d-11*h0*h0
0396      grhog = kappa/c**2*4*sigm
0397      ! grhog=1.4952d-13*tcmb**
0398      grhor = 7._dl/8*(4._dl/11
0399      !grhor=3.3957d-14*tcmb**4
0400
0401      !correction for fractiona
0402      !for massive Nu_mass_dege
0403
0404      grhornomass=grhor*nu_mass
0405      grhormass=0
0406      do nu_i = 1, CP%Nu_mass_e
0407          grhormass(nu_i)=grhor
0408      end do
0409      grhoc=grhom*CP%omegac
0410      grhob=grhom*CP%omegab
0411      grhov=grhom*CP%omegav
0412      grhok=grhom*CP%omegak
0413      ! adotrad gives the rela
0414      adotrad = sqrt((grhog+grh
```

/Users/lplopa/Compare/camb_des/modules.f90, Top line: 353

```
0353      ! (r) one flavor of rela
0354      ! (m) nonrelativistic ma
0355      ! 8*pi*G*rho/c^2 at a=1,
0356      ! a=tau(Mpc)*adotrad, wi
0357      ! (Used only to set the
0358
0359      !H0 is in km/s/Mpc
0360
0361      grhom = 3*CP%h0**2/c**2*1
0362
0363      !grhom=3.3379d-11*h0*h0
0364      grhog = kappa/c**2*4*sigm
0365      ! grhog=1.4952d-13*tcmb**
0366      grhor = 7._dl/8*(4._dl/11
0367      !grhor=3.3957d-14*tcmb**4
0368
0369      !correction for fractiona
0370      !for massive Nu_mass_dege
0371
0372      grhornomass=grhor*nu_mass
0373      grhormass=0
0374      do nu_i = 1, CP%Nu_mass_e
0375          grhormass(nu_i)=grhor
0376      end do
0377      grhoc=grhom*CP%omegac
0378      grhob=grhom*CP%omegab
0379      grhov=grhom*CP%omegav
0380      grhok=grhom*CP%omegak
0381      ! adotrad gives the rela
0382      adotrad = sqrt((grhog+grh
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 415

```
0415
0416
0417      Nnow = CP%omegab*(1-CP%yh
0418
0419      akthom = sigma_thomson*Nn
0420      !sigma_T * (number densit
0421
0422      fHe = CP%YHe/(mass_ratio_
0423
0424      if (.not.call_again) then
0425          call init_massive_nu(
0426          call init_background
0427          if (global_error_flag
0428              CP%tau0=TimeOfz(0
0429              ! print *, 'chi =
0430              last_tau0=CP%tau0
0431              if (WantReion) ca
0432          end if
0433      else
0434          CP%tau0=last_tau0
0435      end if
0436
0437      !JD 08/13 Changes for non
0438      !if ( CP%NonLinear==NonLi
0439      if (CP%NonLinear==NonLine
0440          CP%Transfer%kmax = ma
0441          if (FeedbackLevel > 0
0442              write (*,*) 'max_
0443      end if
0444
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 383

```
0383
0384
0385      Nnow = CP%omegab*(1-CP%yh
0386
0387      akthom = sigma_thomson*Nn
0388      !sigma_T * (number densit
0389
0390      fHe = CP%YHe/(mass_ratio_
0391
0392      if (.not.call_again) then
0393          call init_massive_nu(
0394          call init_background
0395          if (global_error_flag
0396              CP%tau0=TimeOfz(0
0397              ! print *, 'chi =
0398              last_tau0=CP%tau0
0399              if (WantReion) ca
0400          end if
0401      else
0402          CP%tau0=last_tau0
0403      end if
0404
0405      !JD 08/13 Changes for non
0406      !if ( CP%NonLinear==NonLi
0407      if (CP%NonLinear==NonLine
0408          CP%Transfer%kmax = ma
0409          if (FeedbackLevel > 0
0410              write (*,*) 'max_
0411      end if
0412
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 445

```
0445      if (CP%closed .and. CP%ta
0446          call GlobalError('chi
0447      end if
0448
0449      if (global_error_flag/=0)
0450          if (present(error)) e
0451          return
0452      end if
0453
0454      if (present(error)) then
0455          error = 0
0456      else if (FeedbackLevel >
0457          write(*, '("Om_b h^2
0458          write(*, '("Om_c h^2
0459          write(*, '("Om_nu h^2
0460          write(*, '("Om_Lambda
0461          write(*, '("Om_K
0462          write(*, '("Om_m (1-Om
0463          write(*, '("100 theta
0464          if (CP%Num_Nu_Massive
0465              write(*, '("N_eff
0466                  sum(CP%Nu_mas
0467              do nu_i=1, CP%Nu_
0468                  conv = k_B*(8
0469                      (CP%nu_ma
0470                  write(*, '(I2,
0471                      CP%nu_mas
0472              end do
0473          end if
0474      end if
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 413

```
0413      if (CP%closed .and. CP%ta
0414          call GlobalError('chi
0415      end if
0416
0417      if (global_error_flag/=0)
0418          if (present(error)) e
0419          return
0420      end if
0421
0422      if (present(error)) then
0423          error = 0
0424      else if (FeedbackLevel >
0425          write(*, '("Om_b h^2
0426          write(*, '("Om_c h^2
0427          write(*, '("Om_nu h^2
0428          write(*, '("Om_Lambda
0429          write(*, '("Om_K
0430          write(*, '("Om_m (1-Om
0431          write(*, '("100 theta
0432          if (CP%Num_Nu_Massive
0433              write(*, '("N_eff
0434                  sum(CP%Nu_mas
0435              do nu_i=1, CP%Nu_
0436                  conv = k_B*(8
0437                      (CP%nu_ma
0438                  write(*, '(I2,
0439                      CP%nu_mas
0440              end do
0441          end if
0442      end if
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 475

```
0475      CP%chi0=rofChi(CP%tau0/CP
0476      scale= CP%chi0*CP%r/CP%ta
0477
0478      end subroutine CAMBParams
0479
0480
0481      function GetTestTime()
0482      real(sp) GetTestTime
0483      real(sp) atime
0484
0485      !              GetTestTime =
0486      !Can replace this if etim
0487      !Or just comment out - on
0488      call cpu_time(atime)
0489      GetTestTime = atime
0490
0491      end function GetTestTime
0492
0493
0494      function rofChi(Chi) !sin
0495      real(dl) Chi,rofChi
0496
0497      if (CP%closed) then
0498          rofChi=sin(chi)
0499      else if (CP%open) then
0500          rofChi=sinh(chi)
0501      else
0502          rofChi=chi
0503      endif
0504      end function rofChi
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 443

```
0443      CP%chi0=rofChi(CP%tau0/CP
0444      scale= CP%chi0*CP%r/CP%ta
0445
0446      end subroutine CAMBParams
0447
0448
0449      function GetTestTime()
0450      real(sp) GetTestTime
0451      real(sp) atime
0452
0453      !              GetTestTime =
0454      !Can replace this if etim
0455      !Or just comment out - on
0456      call cpu_time(atime)
0457      GetTestTime = atime
0458
0459      end function GetTestTime
0460
0461
0462      function rofChi(Chi) !sin
0463      real(dl) Chi,rofChi
0464
0465      if (CP%closed) then
0466          rofChi=sin(chi)
0467      else if (CP%open) then
0468          rofChi=sinh(chi)
0469      else
0470          rofChi=chi
0471      endif
0472      end function rofChi
```

```
0505
0506
0507      function cosfunc (Chi)
0508      real(dl) Chi,cosfunc
0509
0510      if (CP%closed) then
0511          cosfunc= cos(chi)
0512      else if (CP%open) then
0513          cosfunc=cosh(chi)
0514      else
0515          cosfunc = 1._dl
0516      endif
0517      end function cosfunc
0518
0519      function tanfunc(Chi)
0520      real(dl) Chi,tanfunc
0521      if (CP%closed) then
0522          tanfunc=tan(Chi)
0523      else if (CP%open) then
0524          tanfunc=tanh(Chi)
0525      else
0526          tanfunc=Chi
0527      end if
0528
0529      end function tanfunc
0530
0531      function invsinfunc(x)
0532      real(dl) invsinfunc,x
0533
0534      if (CP%closed) then
```

```
0473
0474
0475      function cosfunc (Chi)
0476      real(dl) Chi,cosfunc
0477
0478      if (CP%closed) then
0479          cosfunc= cos(chi)
0480      else if (CP%open) then
0481          cosfunc=cosh(chi)
0482      else
0483          cosfunc = 1._dl
0484      endif
0485      end function cosfunc
0486
0487      function tanfunc(Chi)
0488      real(dl) Chi,tanfunc
0489      if (CP%closed) then
0490          tanfunc=tan(Chi)
0491      else if (CP%open) then
0492          tanfunc=tanh(Chi)
0493      else
0494          tanfunc=Chi
0495      end if
0496
0497      end function tanfunc
0498
0499      function invsinfunc(x)
0500      real(dl) invsinfunc,x
0501
0502      if (CP%closed) then
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 535

```
0535      invsinfunc=asin(x)
0536      else if (CP%open) then
0537          invsinfunc=log((x+sqr
0538      else
0539          invsinfunc = x
0540      endif
0541      end function invsinfunc
0542
0543      function f_K(x)
0544      real(dl) :: f_K
0545      real(dl), intent(in) :: x
0546      f_K = CP%r*rofChi(x/CP%r)
0547
0548      end function f_K
0549
0550
0551      function DeltaTime(a1,a2,
0552      implicit none
0553      real(dl) DeltaTime, atol
0554      real(dl), intent(IN) :: a
0555      real(dl), optional, inten
0556      real(dl) dtauda, rombint
0557      external dtauda, rombint
0558
0559      if (present(in_tol)) then
0560          atol = in_tol
0561      else
0562          atol = tol/1000/exp(A
0563      end if
0564      DeltaTime=rombint(dtauda,
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 503

```
0503      invsinfunc=asin(x)
0504      else if (CP%open) then
0505          invsinfunc=log((x+sqr
0506      else
0507          invsinfunc = x
0508      endif
0509      end function invsinfunc
0510
0511      function f_K(x)
0512      real(dl) :: f_K
0513      real(dl), intent(in) :: x
0514      f_K = CP%r*rofChi(x/CP%r)
0515
0516      end function f_K
0517
0518
0519      function DeltaTime(a1,a2,
0520      implicit none
0521      real(dl) DeltaTime, atol
0522      real(dl), intent(IN) :: a
0523      real(dl), optional, inten
0524      real(dl) dtauda, rombint
0525      external dtauda, rombint
0526
0527      if (present(in_tol)) then
0528          atol = in_tol
0529      else
0530          atol = tol/1000/exp(A
0531      end if
0532      DeltaTime=rombint(dtauda,
```

```
0565
0566      end function DeltaTime
0567
0568      function TimeOfz(z)
0569      implicit none
0570      real(dl) TimeOfz
0571      real(dl), intent(IN) :: z
0572
0573      TimeOfz=DeltaTime(0._dl,1
0574      end function TimeOfz
0575
```

```
0576      function DeltaPhysicalTim
0577      use constants
0578      real(dl), intent(in) :: a
0579      real(dl), optional, inten
```

```
0533
0534      end function DeltaTime
0535
0536      function TimeOfz(z)
0537      implicit none
0538      real(dl) TimeOfz
0539      real(dl), intent(IN) :: z
0540
0541      TimeOfz=DeltaTime(0._dl,1
0542      end function TimeOfz
0543
```

```
subroutine TimeOfzArr(nz,
integer, intent(in) :: nz
real(dl), intent(in) :: r
real(dl), intent(out) ::
integer i
```

```
!Dumb slow version
!$OMP PARALLEL DO DEFAULT
do i=1, nz
    outputs(i) = timeOfz(
end do
!$OMP END PARALLEL DO
```

```
end subroutine TimeOfzArr
```

```
0558
0559      function DeltaPhysicalTim
0560      use constants
0561      real(dl), intent(in) :: a
0562      real(dl), optional, inten
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 580

```
0580      real(dl) rombint,DeltaPhy
0581      external rombint
0582
0583      if (present(in_tol)) then
0584          atol = in_tol
0585      else
0586          atol = 1d-4/exp(Accur
0587      end if
0588      DeltaPhysicalTimeGyr = ro
0589      end function DeltaPhysica
0590
0591      function AngularDiameterD
0592      !This is the physical (no
0593      real(dl) AngularDiameterD
0594      real(dl), intent(in) :: z
0595
0596      AngularDiameterDistance =
0597
0598      end function AngularDiamete
0599
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 563

```
0563      real(dl) rombint,DeltaPhy
0564      external rombint
0565
0566      if (present(in_tol)) then
0567          atol = in_tol
0568      else
0569          atol = 1d-4/exp(Accur
0570      end if
0571      DeltaPhysicalTimeGyr = ro
0572      end function DeltaPhysica
0573
0574      function AngularDiameterD
0575      !This is the physical (no
0576      real(dl) AngularDiameterD
0577      real(dl), intent(in) :: z
0578
0579      AngularDiameterDistance =
0580
0581      end function AngularDiamete
0582
0583      subroutine AngularDiamete
0584      !This is the physical (no
0585      !z array must be monotoni
0586      integer,intent(in) :: n
0587      real(dl), intent(out) ::
0588      real(dl), intent(in) :: z
0589      integer i
0590
0591      call ComovingRadialDistan
0592      if (CP%flat) then
```

```
0600      function AngularDiameterD
0601      !From http://www.slac.sta
0602      real(dl) AngularDiameterD
0603      real(dl), intent(in) :: z
0604
0605      AngularDiameterDistance2
0606
0607      end function AngularDiamete
0608
0609      function LuminosityDistan
0610      real(dl) LuminosityDistan
0611      real(dl), intent(in) :: z
0612
0613      LuminosityDistance = Angu
0614
0615      end function LuminosityDi
0616
0617      function ComovingRadialDi
0618      real(dl) ComovingRadialDi
0619      real(dl), intent(in) :: z
```

```
0593      arr = arr/(1+z)
0594      else
0595          do i=1, n
0596              arr(i) = CP%r/(1
0597          end do
0598      end if
0599
0600      end subroutine AngularDia
0601
0602
0603      function AngularDiameterD
0604      !From http://www.slac.sta
0605      real(dl) AngularDiameterD
0606      real(dl), intent(in) :: z
0607
0608      AngularDiameterDistance2
0609
0610      end function AngularDiamete
0611
0612      function LuminosityDistan
0613      real(dl) LuminosityDistan
0614      real(dl), intent(in) :: z
0615
0616      LuminosityDistance = Angu
0617
0618      end function LuminosityDi
0619
0620      function ComovingRadialDi
0621      real(dl) ComovingRadialDi
0622      real(dl), intent(in) :: z
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 620

```
0620
0621      ComovingRadialDistance =
0622
0623      end function ComovingRadi
0624
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 623

```
0623      ComovingRadialDistance =
0624
0625      end function ComovingRadi
0626
0627
0628      subroutine ComovingRadial
0629      !z array must be monotoni
0630      integer, intent(in) :: n
0631      real(dl), intent(out) ::
0632      real(dl), intent(in) :: z
0633      real(dl), intent(in) :: t
0634      integer i
0635
0636      !$OMP PARALLEL DO DEFAULT
0637      do i = 1, n
0638          if (i==1) then
0639              if (z(i) < 1e-6_d
0640                  arr(i) = 0
0641              else
0642                  arr(i) = Delt
0643              end if
0644          else
0645              if (z(i) < z(i-1)
0646                  arr(i) = DeltaTim
0647              end if
0648          end do
0649      !$OMP END PARALLEL DO
0650      do i = 2, n
0651          arr(i) = arr(i) + ar
0652      end do
```

```
0625      function Hofz(z)
0626      !!non-comoving Hubble in
0627
0628      real(dl) Hofz, dtauda, a
0629      real(dl), intent(in) :: z
0630      external dtauda
0631
0632      a = 1/(1+z)
0633      Hofz = 1/(a**2*dtauda(a))
0634
0635      end function Hofz
```

```
0636      real(dl) function BAO_D_v
```

```
0653
0654      end subroutine ComovingRa
0655
0656      function Hofz(z)
0657      !non-comoving Hubble in M
0658      !multiply by c/1e3 to get
0659      real(dl) Hofz, dtauda, a
0660      real(dl), intent(in) :: z
0661      external dtauda
0662
0663      a = 1/(1+z)
0664      Hofz = 1/(a**2*dtauda(a))
0665
0666      end function Hofz
0667
0668      subroutine HofzArr(arr, z
0669      !non-comoving Hubble in M
0670      !multiply by c/1e3 to get
0671      integer, intent(in) :: n
0672      real(dl), intent(out) ::
0673      real(dl), intent(in) :: z
0674      integer i
0675
0676      do i=1, n
0677          arr(i) = Hofz(z(i))
0678      end do
0679
0680      end subroutine HofzArr
0681
0682      real(dl) function BAO_D_v
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 637

```
0637      real(dl), intent(in) :: z
0638      real(dl) ADD
0639
0640      ADD = DA*(1.d0+z)
0641      BAO_D_v_from_DA_H = ((ADD
0642
0643      end function BAO_D_v_from
0644
0645      real(dl) function BAO_D_v
0646      real(dl), intent(IN) :: z
0647
0648      BAO_D_v = BAO_D_v_from_DA
0649
0650      end function BAO_D_v
0651
0652      function dsound_da_exact(
0653      implicit none
0654      real(dl) dsound_da_exact,
0655      external dtauda
0656
0657      R = 3*grhob*a / (4*grhog)
0658      cs=1.0d0/sqrt(3*(1+R))
0659      dsound_da_exact=dtauda(a)
0660
0661      end function dsound_da_ex
0662
0663
0664      function dsound_da(a)
0665      !approximate form used e.
0666      implicit none
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 683

```
0683      real(dl), intent(in) :: z
0684      real(dl) ADD
0685
0686      ADD = DA*(1.d0+z)
0687      BAO_D_v_from_DA_H = ((ADD
0688
0689      end function BAO_D_v_from
0690
0691      real(dl) function BAO_D_v
0692      real(dl), intent(IN) :: z
0693
0694      BAO_D_v = BAO_D_v_from_DA
0695
0696      end function BAO_D_v
0697
0698      function dsound_da_exact(
0699      implicit none
0700      real(dl) dsound_da_exact,
0701      external dtauda
0702
0703      R = 3*grhob*a / (4*grhog)
0704      cs=1.0d0/sqrt(3*(1+R))
0705      dsound_da_exact=dtauda(a)
0706
0707      end function dsound_da_ex
0708
0709
0710      function dsound_da(a)
0711      !approximate form used e.
0712      implicit none
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 667

```
0667      real(dl) dsound_da, dtauda
0668      external dtauda
0669
0670      R=3.0d4*a*CP%omegab*(CP%h
0671      !           R = 3*grhob*a
0672      cs=1.0d0/sqrt(3*(1+R))
0673      dsound_da=dtauda(a)*cs
0674
0675      end function dsound_da
0676
0677      function dtda(a)
0678      real(dl) dtda, dtauda, a
0679      external dtauda
0680      dtda= dtauda(a)*a
0681      end function
0682
0683      function CosmomcTheta()
0684      real(dl) zstar, astar, at
0685      real(dl) CosmomcTheta
0686      real(dl) ombh2, omdmh2
0687      real(dl) rombint
0688      external rombint
0689
0690      ombh2 = CP%omegab*(CP%h0/
0691      omdmh2 = (CP%omegac+CP%om
0692
0693      !!From Hu & Sugiyama
0694      zstar = 1048*(1+0.00124*
0695      (0.0783*ombh2**(-0.23
0696      (omdmh2+ombh2)**(0.56
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 713

```
0713      real(dl) dsound_da, dtauda
0714      external dtauda
0715
0716      R=3.0d4*a*CP%omegab*(CP%h
0717      !           R = 3*grhob*a
0718      cs=1.0d0/sqrt(3*(1+R))
0719      dsound_da=dtauda(a)*cs
0720
0721      end function dsound_da
0722
0723      function dtda(a)
0724      real(dl) dtda, dtauda, a
0725      external dtauda
0726      dtda= dtauda(a)*a
0727      end function
0728
0729      function CosmomcTheta()
0730      real(dl) zstar, astar, at
0731      real(dl) CosmomcTheta
0732      real(dl) ombh2, omdmh2
0733      real(dl) rombint
0734      external rombint
0735
0736      ombh2 = CP%omegab*(CP%h0/
0737      omdmh2 = (CP%omegac+CP%om
0738
0739      !!From Hu & Sugiyama
0740      zstar = 1048*(1+0.00124*
0741      (0.0783*ombh2**(-0.23
0742      (omdmh2+ombh2)**(0.56
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 697

```
0697
0698      astar = 1/(1+zstar)
0699      atol = 1e-6
0700      rs = rombint(dsound_da,ld
0701      DA = AngularDiameterDista
0702      CosmomcTheta = rs/DA
0703      !          print *, 'z* = ', z
0704
0705      end function CosmomcTheta
0706
0707      end module ModelParams
0708
0709
0710
0711      !cccccccccccccccccccccccccccccccccc
0712
0713      module lvalues
0714      use precision
0715      use ModelParams
0716      implicit none
0717      public
0718
0719      Type lSamples
0720          integer l0
0721          integer l(lmax_arr)
0722      end Type lSamples
0723
0724      Type(lSamples) :: lSamp
0725      !Sources
0726      logical :: Log_lvalues =
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 743

```
0743
0744      astar = 1/(1+zstar)
0745      atol = 1e-6
0746      rs = rombint(dsound_da,ld
0747      DA = AngularDiameterDista
0748      CosmomcTheta = rs/DA
0749      !          print *, 'z* = ', z
0750
0751      end function CosmomcTheta
0752
0753      end module ModelParams
0754
0755
0756
0757      !cccccccccccccccccccccccccccccccccc
0758
0759      module lvalues
0760      use precision
0761      use ModelParams
0762      implicit none
0763      public
0764
0765      Type lSamples
0766          integer l0
0767          integer l(lmax_arr)
0768      end Type lSamples
0769
0770      Type(lSamples) :: lSamp
0771      !Sources
0772      logical :: Log_lvalues =
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 727

```
0727
0728      contains
0729
0730      function lvalues_indexOf(
0731      type(lSamples) :: lSet
0732      integer, intent(in) :: l
0733      integer lvalues_indexOf,
0734
0735      do i=2,lSet%l0
0736          if (l < lSet%l(i)) th
0737              lvalues_indexOf =
0738                  return
0739          end if
0740      end do
0741      lvalues_indexOf = lSet%l0
0742
0743      end function lvalues_ind
0744
0745      subroutine initlval(lSet,
0746
0747      ! This subroutines initia
0748
0749      implicit none
0750      type(lSamples) :: lSet
0751
0752      integer, intent(IN) :: ma
0753      integer lind, lvar, step,
0754      real(dl) AScale
0755
0756      Ascale=scale/lSampleBoost
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 773

```
0773
0774      contains
0775
0776      function lvalues_indexOf(
0777      type(lSamples) :: lSet
0778      integer, intent(in) :: l
0779      integer lvalues_indexOf,
0780
0781      do i=2,lSet%l0
0782          if (l < lSet%l(i)) th
0783              lvalues_indexOf =
0784                  return
0785          end if
0786      end do
0787      lvalues_indexOf = lSet%l0
0788
0789      end function lvalues_ind
0790
0791      subroutine initlval(lSet,
0792
0793      ! This subroutines initia
0794
0795      implicit none
0796      type(lSamples) :: lSet
0797
0798      integer, intent(IN) :: ma
0799      integer lind, lvar, step,
0800      real(dl) AScale
0801
0802      Ascale=scale/lSampleBoost
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 757

```
0757
0758     if (lSampleBoost >=50) th
0759         !just do all of them
0760         lind=0
0761         do lvar=lmin, max_l
0762             lind=lind+1
0763             ls(lind)=lvar
0764         end do
0765         lSet%l0=lind
0766         lSet%l(1:lind) = ls(1
0767         return
0768     end if
0769
0770     lind=0
0771     do lvar=lmin, 10
0772         lind=lind+1
0773         ls(lind)=lvar
0774     end do
0775
0776     if (CP%AccurateReionizati
0777         if (lSampleBoost > 1)
0778             do lvar=11, 37,1
0779                 lind=lind+1
0780                 ls(lind)=lvar
0781             end do
0782         else
0783             do lvar=11, 37,2
0784                 lind=lind+1
0785                 ls(lind)=lvar
0786             end do
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 803

```
0803
0804     if (lSampleBoost >=50) th
0805         !just do all of them
0806         lind=0
0807         do lvar=lmin, max_l
0808             lind=lind+1
0809             ls(lind)=lvar
0810         end do
0811         lSet%l0=lind
0812         lSet%l(1:lind) = ls(1
0813         return
0814     end if
0815
0816     lind=0
0817     do lvar=lmin, 10
0818         lind=lind+1
0819         ls(lind)=lvar
0820     end do
0821
0822     if (CP%AccurateReionizati
0823         if (lSampleBoost > 1)
0824             do lvar=11, 37,1
0825                 lind=lind+1
0826                 ls(lind)=lvar
0827             end do
0828         else
0829             do lvar=11, 37,2
0830                 lind=lind+1
0831                 ls(lind)=lvar
0832             end do
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 787

```
0787         end if
0788
0789         step = max(nint(5*Asc
0790         bot=40
0791         top=bot + step*10
0792     else
0793         if (lSampleBoost >1)
0794             do lvar=11, 15
0795                 lind=lind+1
0796                 ls(lind)=lvar
0797             end do
0798         else
0799             lind=lind+1
0800             ls(lind)=12
0801             lind=lind+1
0802             ls(lind)=15
0803         end if
0804         step = max(nint(10*As
0805         bot=15+max(step/2,2)
0806         top=bot + step*7
0807     end if
0808
0809     do lvar=bot, top, step
0810         lind=lind+1
0811         ls(lind)=lvar
0812     end do
0813
0814     !Sources
0815     if (Log_lvalues) then
0816         !Useful for generatin
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 833

```
0833         end if
0834
0835         step = max(nint(5*Asc
0836         bot=40
0837         top=bot + step*10
0838     else
0839         if (lSampleBoost >1)
0840             do lvar=11, 15
0841                 lind=lind+1
0842                 ls(lind)=lvar
0843             end do
0844         else
0845             lind=lind+1
0846             ls(lind)=12
0847             lind=lind+1
0848             ls(lind)=15
0849         end if
0850         step = max(nint(10*As
0851         bot=15+max(step/2,2)
0852         top=bot + step*7
0853     end if
0854
0855     do lvar=bot, top, step
0856         lind=lind+1
0857         ls(lind)=lvar
0858     end do
0859
0860     !Sources
0861     if (Log_lvalues) then
0862         !Useful for generatin
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 817

```
0817      step=max(nint(20*Asca
0818      do
0819          lvar = lvar + ste
0820          if (lvar > max_l)
0821              lind=lind+1
0822              ls(lind)=lvar
0823              step = nint(step*
0824      end do
0825  else
0826      step=max(nint(20*Asca
0827      bot=ls(lind)+step
0828      top=bot+step*2
0829
0830      do lvar = bot,top,ste
0831          lind=lind+1
0832          ls(lind)=lvar
0833      end do
0834
0835      if (ls(lind)>=max_l)
0836          do lvar=lind,1,-1
0837              if (ls(lvar)<
0838          end do
0839          lind=lvar
0840          if (ls(lind)<max_
0841              lind=lind+1
0842              ls(lind)=max_
0843          end if
0844      else
0845          step=max(nint(25*
0846          !Get EE right aro
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 863

```
0863      step=max(nint(20*Asca
0864      do
0865          lvar = lvar + ste
0866          if (lvar > max_l)
0867              lind=lind+1
0868              ls(lind)=lvar
0869              step = nint(step*
0870      end do
0871  else
0872      step=max(nint(20*Asca
0873      bot=ls(lind)+step
0874      top=bot+step*2
0875
0876      do lvar = bot,top,ste
0877          lind=lind+1
0878          ls(lind)=lvar
0879      end do
0880
0881      if (ls(lind)>=max_l)
0882          do lvar=lind,1,-1
0883              if (ls(lvar)<
0884          end do
0885          lind=lvar
0886          if (ls(lind)<max_
0887              lind=lind+1
0888              ls(lind)=max_
0889          end if
0890      else
0891          step=max(nint(25*
0892          !Get EE right aro
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 847

```
0847      bot=ls(lind)+step
0848      top=bot+step
0849
0850      do lvar = bot,top
0851          lind=lind+1
0852          ls(lind)=lvar
0853      end do
0854
0855      if (ls(lind)>=max
0856          do lvar=lind,
0857              if (ls(lv
0858          end do
0859          lind=lvar
0860          if (ls(lind)<
0861              lind=lind
0862              ls(lind)=
0863          end if
0864      else
0865          if (HighAccur
0866              step=max(
0867          else
0868              step=max(
0869          end if
0870          bot=ls(lind)+
0871          top=min(5000,
0872
0873          do lvar = bot
0874              lind=lind
0875              ls(lind)=
0876          end do
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 893

```
0893      bot=ls(lind)+step
0894      top=bot+step
0895
0896      do lvar = bot,top
0897          lind=lind+1
0898          ls(lind)=lvar
0899      end do
0900
0901      if (ls(lind)>=max
0902          do lvar=lind,
0903              if (ls(lv
0904          end do
0905          lind=lvar
0906          if (ls(lind)<
0907              lind=lind
0908              ls(lind)=
0909          end if
0910      else
0911          if (HighAccur
0912              step=max(
0913          else
0914              step=max(
0915          end if
0916          bot=ls(lind)+
0917          top=min(5000,
0918
0919          do lvar = bot
0920              lind=lind
0921              ls(lind)=
0922          end do
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 877

```
0877
0878         if (max_l > 5
0879             !Should b
0880             step=max(
0881             lvar = ls
0882             do
0883                 lvar
0884                 if (l
0885                 lind=
0886                 ls(li
0887                 step
0888             end do
0889         end if
0890         !Sources
0891     end if !log_lvalu
0892
0893         if (ls(lind) /=ma
0894             lind=lind+1
0895             ls(lind)=max_
0896         end if
0897         if (.not. CP%flat
0898             !Not in CP%flat c
0899         end if
0900     end if
0901     lSet%l0=lind
0902     lSet%l(1:lind) = ls(1:lin
0903
0904     end subroutine initlval
0905
0906     subroutine InterpolateCla
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 923

```
0923
0924         if (max_l > 5
0925             !Should b
0926             step=max(
0927             lvar = ls
0928             do
0929                 lvar
0930                 if (l
0931                 lind=
0932                 ls(li
0933                 step
0934             end do
0935         end if
0936         !Sources
0937     end if !log_lvalu
0938
0939         if (ls(lind) /=ma
0940             lind=lind+1
0941             ls(lind)=max_
0942         end if
0943         if (.not. CP%flat
0944             !Not in CP%flat c
0945         end if
0946     end if
0947     lSet%l0=lind
0948     lSet%l(1:lind) = ls(1:lin
0949
0950     end subroutine initlval
0951
0952     subroutine InterpolateCla
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 907

```
0907      type (lSamples), intent(i
0908      real(dl), intent(in) :: i
0909      real(dl), intent(out):: a
0910      integer, intent(in) :: ma
0911      integer il,llo,lhi, xi
0912      real(dl) ddCl(lSet%l0)
0913      real(dl) xl(lSet%l0)
0914
0915      real(dl) a0,b0,ho
0916      real(dl), parameter :: cl
0917
0918      if (max_ind > lSet%l0) st
0919
0920      xl = real(lSet%l(1:lSet%l
0921      call spline(xl,iCL(1),max
0922
0923      llo=1
0924      do il=lmin,lSet%l(max_ind
0925          xi=il
0926          if ((xi > lSet%l(llo+
0927              llo=llo+1
0928          end if
0929          lhi=llo+1
0930          ho=lSet%l(lhi)-lSet%l
0931          a0=(lSet%l(lhi)-xi)/h
0932          b0=(xi-lSet%l(llo))/h
0933
0934          all_Cl(il) = a0*iCl(1
0935                  +(b0**3-b0)*ddCl(
0936      end do
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 953

```
0953      type (lSamples), intent(i
0954      real(dl), intent(in) :: i
0955      real(dl), intent(out):: a
0956      integer, intent(in) :: ma
0957      integer il,llo,lhi, xi
0958      real(dl) ddCl(lSet%l0)
0959      real(dl) xl(lSet%l0)
0960
0961      real(dl) a0,b0,ho
0962      real(dl), parameter :: cl
0963
0964      if (max_ind > lSet%l0) ca
0965
0966      xl = real(lSet%l(1:lSet%l
0967      call spline(xl,iCL(1),max
0968
0969      llo=1
0970      do il=lmin,lSet%l(max_ind
0971          xi=il
0972          if ((xi > lSet%l(llo+
0973              llo=llo+1
0974          end if
0975          lhi=llo+1
0976          ho=lSet%l(lhi)-lSet%l
0977          a0=(lSet%l(lhi)-xi)/h
0978          b0=(xi-lSet%l(llo))/h
0979
0980          all_Cl(il) = a0*iCl(1
0981                  +(b0**3-b0)*ddCl(
0982      end do
```

```
0937
0938     end subroutine Interpolat
0939
0940     subroutine InterpolateCLA
0941     type (lSamples), intent(i
0942     real(dl), intent(in) :: i
0943     real(dl), intent(out):: a
0944     integer, intent(in) :: ma
0945     integer, intent(in), opti
0946     integer maxdelta, il
0947     real(dl) DeltaCL(lSet%l0)
0948     real(dl), allocatable ::
0949
0950     if (max_ind > lSet%l0) st
0951
0952     if (use_spline_template .
0953         if (template_index<=3
0954         !interpolate only
0955         !Using unlensed f
0956         maxdelta=max_ind
0957         do while (lSet%l(
0958             maxdelta=maxd
0959         end do
0960         DeltaCL(1:maxdelt
0961
0962         call InterpolateC
0963
0964         do il=lmin,lSet%l
0965             all_Cl(il) =
0966         end do
```

```
0983
0984     end subroutine Interpolat
0985
0986     subroutine InterpolateCLA
0987     type (lSamples), intent(i
0988     real(dl), intent(in) :: i
0989     real(dl), intent(out):: a
0990     integer, intent(in) :: ma
0991     integer, intent(in), opti
0992     integer maxdelta, il
0993     real(dl) DeltaCL(lSet%l0)
0994     real(dl), allocatable ::
0995
0996     if (max_ind > lSet%l0) ca
0997
0998     if (use_spline_template .
0999         if (template_index<=3
1000         !interpolate only
1001         !Using unlensed f
1002         maxdelta=max_ind
1003         do while (lSet%l(
1004             maxdelta=maxd
1005         end do
1006         DeltaCL(1:maxdelt
1007
1008         call InterpolateC
1009
1010         do il=lmin,lSet%l
1011             all_Cl(il) =
1012         end do
```

```
/Users/lplopa/Compare/camb_des/modules.  
f90, Top line: 1013
```

```
0967  
0968         if (maxdelta < ma  
0969             !directly int  
0970             allocate(tmpa  
0971             call Interpol  
0972             !overlap to r  
0973             all_cl(lSet%l  
0974             deallocate(tm  
0975         end if  
0976         return  
0977     end if  
0978 end if  
0979  
0980 call InterpolateClArr(lSe  
0981  
0982  
0983 end subroutine Interpolat  
0984  
0985 end module lvalues  
0986  
0987  
0988 !ccccccccccccccccccccccccccccc  
0989  
0990 module ModelData  
0991 use precision  
0992 use ModelParams  
0993 use InitialPower  
0994 use lValues  
0995 use Ranges  
0996 use AMUtils
```

```

1013
1014         if (maxdelta < ma
1015             !directly int
1016             allocate(tmpa
1017             call Interpol
1018             !overlap to r
1019             all_cl(1Set%l
1020             deallocate(tm
1021         end if
1022         return
1023     end if
1024 end if
1025
1026 call InterpolateClArr(1Se
1027
1028
1029 end subroutine Interpolat
1030
1031 end module lvalues
1032
1033
1034 !ccccccccccccccccccccccccccccc
1035
1036 module ModelData
1037 use precision
1038 use ModelParams
1039 use InitialPower
1040 use lValues
1041 use Ranges
1042 use AMUtils

```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 997

```
0997      implicit none
0998      public
0999
1000      Type LimberRec
1001          integer n1,n2 !corres
1002          real(dl), dimension(:
1003          real(dl), dimension(:
1004      end Type LimberRec
1005
1006      Type ClTransferData
1007          !Cl transfer function
1008          !values of q for inte
1009      Type (lSamples) :: ls
1010      integer :: NumSources
1011      !Changes -scalars: 2
1012      !- tensors: T and E a
1013
1014      Type (Regions) :: q
1015      real(dl), dimension(:
1016
1017      !The L index of the l
1018      integer, dimension(:)
1019      !For each l, the set
1020      !indices LimberWindow
1021      Type(LimberRec), dime
1022
1023      !The maximum L needed
1024      integer max_index_non
1025
1026      end Type ClTransferData
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1043

```
1043      implicit none
1044      public
1045
1046      Type LimberRec
1047          integer n1,n2 !corres
1048          real(dl), dimension(:
1049          real(dl), dimension(:
1050      end Type LimberRec
1051
1052      Type ClTransferData
1053          !Cl transfer function
1054          !values of q for inte
1055      Type (lSamples) :: ls
1056      integer :: NumSources
1057      !Changes -scalars: 2
1058      !- tensors: T and E a
1059
1060      Type (Regions) :: q
1061      real(dl), dimension(:
1062
1063      !The L index of the l
1064      integer, dimension(:)
1065      !For each l, the set
1066      !indices LimberWindow
1067      Type(LimberRec), dime
1068
1069      !The maximum L needed
1070      integer max_index_non
1071
1072      end Type ClTransferData
```

```
1027
1028      Type(ClTransferData), sav
1029
1030      !Computed output power sp
1031
1032      integer, parameter :: C_T
1033      integer :: C_last = C_Phi
1034      integer, parameter :: CT_
```

```
1035
1036      logical :: has_cl_2D_arra
1037
1038      real(dl), dimension (:,:,
1039      !Indices are Cl_xxx( 1 ,
1040      !where Cl_type is one of
1041
1042      real(dl), dimension (:,:,
1043      !Indices are Cl_xxx( 1 ,
1044      !where ordering of fields
1045
1046      !The following are set on
1047      integer lmax_lensed !Only
1048      real(dl) , dimension (:,:,
1049      !Cl_lensed(1, power_index
1050
1051      contains
```

```
1073
1074      Type(ClTransferData), sav
1075
1076      !Computed output power sp
1077
1078      integer, parameter :: C_T
1079      integer :: C_last = C_Phi
1080      integer, parameter :: CT_
```

```
1081      integer, parameter :: nam
1082      character(LEN=name_tag_le
1083      character(LEN=name_tag_le
1084      character(LEN=name_tag_le
```

```
1085
1086
1087      logical :: has_cl_2D_arra
1088
1089      real(dl), dimension (:,:,
1090      !Indices are Cl_xxx( 1 ,
1091      !where Cl_type is one of
1092
1093      real(dl), dimension (:,:,
1094      !Indices are Cl_xxx( 1 ,
1095      !where ordering of fields
1096
1097      !The following are set on
1098      integer lmax_lensed !Only
1099      real(dl) , dimension (:,:,
1100      !Cl_lensed(1, power_index
1101
1102      contains
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1052

```
1052
1053      subroutine Init_ClTransfe
1054      !Need to set the Ranges a
1055      Type(ClTransferData) :: C
1056      integer st
1057
1058      deallocate(CTrans%Delta_p
1059      call Ranges_getArray(CTra
1060
1061      allocate(CTrans%Delta_p_l
1062              min(CTrans%max_index
1063      if (st /= 0) stop 'Init_C
1064      CTrans%Delta_p_l_k = 0
1065
1066      end subroutine Init_ClTra
1067
1068      subroutine Init_Limber(CT
1069      Type(ClTransferData) :: C
1070
1071      allocate(CTrans%Limber_l_
1072      CTrans%Limber_l_min = 0
1073      if (num_redshiftwindows>0
1074          allocate(CTrans%Limbe
1075      end if
1076
1077      end subroutine Init_Limbe
1078
1079      subroutine Free_ClTransfe
1080      Type(ClTransferData) :: C
1081      integer st
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1103

```
1103
1104      subroutine Init_ClTransfe
1105      !Need to set the Ranges a
1106      Type(ClTransferData) :: C
1107      integer st
1108
1109      deallocate(CTrans%Delta_p
1110      call Ranges_getArray(CTra
1111
1112      allocate(CTrans%Delta_p_l
1113              min(CTrans%max_index
1114      if (st /= 0) call MpiStop
1115      CTrans%Delta_p_l_k = 0
1116
1117      end subroutine Init_ClTra
1118
1119      subroutine Init_Limber(CT
1120      Type(ClTransferData) :: C
1121
1122      allocate(CTrans%Limber_l_
1123      CTrans%Limber_l_min = 0
1124      if (num_redshiftwindows>0
1125          allocate(CTrans%Limbe
1126      end if
1127
1128      end subroutine Init_Limbe
1129
1130      subroutine Free_ClTransfe
1131      Type(ClTransferData) :: C
1132      integer st
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1082

```
1082
1083      deallocate(CTrans%Delta_p
1084      nullify(CTrans%Delta_p_l
1085      call Ranges_Free(CTrans%q
1086      call Free_Limber(CTrans)
1087
1088      end subroutine Free_ClTra
1089
1090      subroutine Free_Limber(CT
1091      Type(ClTransferData) :: C
1092      integer st,i,j
1093
1094      if (associated(CTrans%Lim
1095          do i=1, CTrans%NumSou
1096              if (CTrans%Limber
1097                  do j=CTrans%L
1098                      deallocat
1099                      deallocat
1100                  end do
1101              end if
1102          end do
1103          deallocate(CTrans%Lim
1104      end if
1105      deallocate(CTrans%Limber_
1106      nullify(CTrans%Limber_l_m
1107      nullify(CTrans%Limber_win
1108
1109      end subroutine Free_Limbe
1110
1111      subroutine CheckLoadedHig
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1133

```
1133
1134      deallocate(CTrans%Delta_p
1135      nullify(CTrans%Delta_p_l
1136      call Ranges_Free(CTrans%q
1137      call Free_Limber(CTrans)
1138
1139      end subroutine Free_ClTra
1140
1141      subroutine Free_Limber(CT
1142      Type(ClTransferData) :: C
1143      integer st,i,j
1144
1145      if (associated(CTrans%Lim
1146          do i=1, CTrans%NumSou
1147              if (CTrans%Limber
1148                  do j=CTrans%L
1149                      deallocat
1150                      deallocat
1151                  end do
1152              end if
1153          end do
1154          deallocate(CTrans%Lim
1155      end if
1156      deallocate(CTrans%Limber_
1157      nullify(CTrans%Limber_l_m
1158      nullify(CTrans%Limber_win
1159
1160      end subroutine Free_Limbe
1161
1162      subroutine CheckLoadedHig
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1112

```
1112      integer L
1113      real(dl) array(7)
1114
1115      if (.not. allocated(highL
1116          allocate(highL_CL_tem
1117
1118      call OpenTxtFile(high
1119      !cafea
1120      ! print*, '1:', highL_un
1121      if (lmin==1) highL_CL
1122      do
1123          read(fileio_unit,
1124          if (L>lmax_extrap
1125          ! array = array
1126          highL_CL_template
1127          highL_CL_template
1128          highL_CL_template
1129      end do
1130
1131      500      if (L< lmax_extrap_hi
1132          stop 'CheckLoaded
1133          close(fileio_unit)
1134      end if
1135
1136      end subroutine CheckLoade
1137
1138
1139      subroutine Init_Cls
1140
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1163

```
1163      integer L
1164      real(dl) array(7)
1165
1166      if (.not. allocated(highL
1167          allocate(highL_CL_tem
1168
1169      call OpenTxtFile(high
1170
1171      if (lmin==1) highL_CL
1172      do
1173          read(fileio_unit,
1174          if (L>lmax_extrap
1175          ! array = array
1176          highL_CL_template
1177          highL_CL_template
1178          highL_CL_template
1179      end do
1180
1181      500      if (L< lmax_extrap_hi
1182          call MpiStop('Che
1183          close(fileio_unit)
1184      end if
1185
1186      end subroutine CheckLoade
1187
1188
1189      subroutine Init_Cls
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1141

```
1141      call CheckLoadedHighLTemp
1142      if (CP%WantScalars) then
1143          if (allocated(Cl_scal
1144              allocate(Cl_scalar(lm
1145              Cl_scalar = 0
1146              if (has_cl_2D_array)
1147                  if (allocated(Cl_
1148                      allocate(Cl_scala
1149                  Cl_scalar_array =
1150              end if
1151      end if
1152
1153      if (CP%WantVectors) then
1154          if (allocated(Cl_vect
1155              allocate(Cl_vector(lm
1156              Cl_vector = 0
1157      end if
1158
1159
1160      if (CP%WantTensors) then
1161          if (allocated(Cl_tens
1162              allocate(Cl_tensor(lm
1163              Cl_tensor = 0
1164      end if
1165
1166      end subroutine Init_Cls
1167
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1190

```
1190      call CheckLoadedHighLTemp
1191      if (CP%WantScalars) then
1192          if (allocated(Cl_scal
1193              allocate(Cl_scalar(lm
1194              Cl_scalar = 0
1195              if (has_cl_2D_array)
1196                  if (allocated(Cl_
1197                      allocate(Cl_scala
1198                          3+num_redshif
1199                  Cl_scalar_array =
1200              end if
1201      end if
1202
1203      if (CP%WantVectors) then
1204          if (allocated(Cl_vect
1205              allocate(Cl_vector(lm
1206              Cl_vector = 0
1207      end if
1208
1209
1210      if (CP%WantTensors) then
1211          if (allocated(Cl_tens
1212              allocate(Cl_tensor(lm
1213              Cl_tensor = 0
1214      end if
1215
1216      end subroutine Init_Cls
1217
1218      function open_file_header
1219      character(LEN=*) , intent(
```

```
1220 character(LEN=*), intent(  
1221 character(LEN=name_tag_le  
1222 integer, intent(in), opti  
1223 integer :: unit, nn  
1224  
1225 if (present(n)) then  
1226     nn = n  
1227 else  
1228     nn = 6  
1229 end if  
1230 open(newunit=unit, file=fi  
1231 if (output_file_headers)  
1232     write(unit, '("#",1A' /  
1233 end if  
1234  
1235 end function open_file_he  
1236  
1237 function scalar_fieldname  
1238 integer, intent(in) :: i  
1239 character(LEN=5) :: scala  
1240 character(LEN=3), paramet  
1241  
1242 if (i<=3) then  
1243     scalar_fieldname = sc  
1244 else  
1245     scalar_fieldname = 'W  
1246 end if  
1247  
1248 end function scalar_field  
1249
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1168

```
1168      subroutine output_cl_file
1169      implicit none
1170      integer in,il
1171      character(LEN=*) ScalFile
1172      real(dl), intent(in), opt
1173      real(dl) fact
1174      integer last_C
1175      real(dl), allocatable ::

1176
1177
1178      if (present(factor)) then
1179          fact = factor
1180      else
1181          fact =1
1182      end if

1183
1184      if (CP%WantScalars .and.
1185          last_C=min(C_PhiTemp,
1186                    open(unit=fileio_unit
1187!cafea
1188!      print*, '2:', fileio_uni
1189      do in=1,CP%InitPower%nn
1190          do il=lmin,min(10
1191              write(fileio_
1192          end do
1193          do il=10100,CP%Ma
1194              write(fileio_
1195                  fact*Cl_s
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1250

```
1250      subroutine output_cl_file
1251      implicit none
1252      integer in,il, i, j
1253      character(LEN=*) ScalFile
1254      real(dl), intent(in), opt
1255      real(dl) fact
1256      integer last_C
1257      real(dl), allocatable ::
1258      integer unit
1259      character(LEN=name_tag_le

1260
1261
1262      if (present(factor)) then
1263          fact = factor
1264      else
1265          fact =1
1266      end if

1267
1268      if (CP%WantScalars .and.
1269          last_C=min(C_PhiTemp,
1270                    unit = open_file_head

1271
1272      do in=1,CP%InitPower%
1273          do il=lmin,min(10
1274              write(unit,tr
1275          end do
1276          do il=10100,CP%Ma
1277              write(unit,tr
                  fact*Cl_s
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1196

```
1196         end do
1197     end do
1198     close(fileio_unit)
1199 end if
1200
1201 if (CP%WantScalars .and.
1202     allocate(outarr(1:3+n
1203     open(unit=fileio_unit
1204     !cafea
1205     !print*, '3:', fileio_
1206
1207     do in=1,CP%InitPower
1208     do il=lmin,min(10
1209         outarr=C1_sca
1210         outarr(1:2,:)
1211         outarr(:,1:2)
1212         write(fileio_
1213     end do
1214     do il=10100,CP%Ma
1215         outarr=C1_sca
1216         outarr(1:2,:)
1217         outarr(:,1:2)
1218         write(fileio_
1219     end do
1220     close(fileio_unit)
1221     deallocate(outarr)
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1278

```
1278         end do
1279     end do
1280     close(unit)
1281 end if
1282
1283 if (CP%WantScalars .and.
1284     allocate(outarr(1:3+n
1285     do i=1, 3+num_redshif
1286     do j=1, 3+num_red
1287         cov_names(j +
1288     end do
1289     end do
1290     unit = open_file_head
1291
1292     do in=1,CP%InitPower%
1293     do il=lmin,min(10
1294         outarr=C1_sca
1295         outarr(1:2,:)
1296         outarr(:,1:2)
1297         write(unit,tr
1298     end do
1299     do il=10100,CP%Ma
1300         outarr=C1_sca
1301         outarr(1:2,:)
1302         outarr(:,1:2)
1303         write(unit,tr
1304     end do
1305     end do
1306     close(unit)
1307     deallocate(outarr)
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1222

```
1222      end if
1223
1224      if (CP%WantTensors .and.
1225         open(unit=fileio_unit
1226            !cafea
1227            ! print*, '4:', fileio_uni
1228      do in=1,CP%InitPower%nn
1229         do il=lmin,CP%Max
1230            write(fileio_
1231               end do
1232         end do
1233         close(fileio_unit)
1234      end if
1235
1236      if (CP%WantTensors .and.
1237         open(unit=fileio_unit
1238            !cafea
1239            ! print*, '2:', fileio
1240      do in=1,CP%InitPower
1241         do il=lmin,CP%Max
1242            write(fileio
1243               fact*Cl_t
1244         end do
1245         do il=CP%Max_l_te
1246            write(fileio_
1247         end do
1248      end do
1249      close(fileio_unit)
1250      end if
1251
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1308

```
1308      end if
1309
1310      if (CP%WantTensors .and.
1311         unit = open_file_head
1312
1313      do in=1,CP%InitPower%
1314         do il=lmin,CP%Max
1315            write(unit, '(
1316         end do
1317      end do
1318      close(unit)
1319      end if
1320
1321      if (CP%WantTensors .and.
1322         unit = open_file_head
1323
1324      do in=1,CP%InitPower%
1325         do il=lmin,CP%Max
1326            write(unit, '(
1327               fact*Cl_t
1328         end do
1329         do il=CP%Max_l_te
1330            write(unit, '(
1331         end do
1332      end do
1333      close(unit)
1334      end if
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1252

```
1252      if (CP%WantScalars .and.  
1253          open(unit=fileio_unit  
1254      !cafea  
1255      ! print*, '5:', fileio_unit  
1256      do in=1,CP%InitPower%nn  
1257          do il=lmin, lmax  
1258              write(fileio_  
1259          end do  
1260      end do  
1261      close(fileio_unit)  
1262  end if  
1263  
1264  
1265      if (CP%WantScalars .and.  
1266          open(unit=fileio_unit  
1267      !cafea  
1268      ! print*, '6:', fileio_  
1269      do in=1,CP%InitPower%  
1270          do il=lmin,min(CP  
1271              write(fileio_  
1272          end do  
1273          do il=min(CP%Max_  
1274              write(fileio_  
1275          end do  
1276      end do  
1277  end if  
1278  end subroutine output_cl_  
1279  
1280  subroutine output_lens_po
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1334

```
1334      if (CP%WantScalars .and.  
1335          unit = open_file_head  
1336  
1337      do in=1,CP%InitPower%  
1338          do il=lmin, lmax  
1339              write(unit, '  
1340          end do  
1341      end do  
1342      close(unit)  
1343  end if  
1344  
1345      if (CP%WantScalars .and.  
1346          unit = open_file_head  
1347  
1348      do in=1,CP%InitPower%  
1349          do il=lmin,min(CP  
1350              write(unit, '  
1351          do il=min(CP%Max_  
1352              write(unit, '  
1353          end do  
1354      end do  
1355      close(unit)  
1356  end if  
1357  end subroutine output_cl_  
1358  
1359  subroutine output_lens_po
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1281

```
1281      !Write out L TT EE BB TE
1282      !This input supported by
1283      implicit none
1284      integer in,il
1285      real(dl), intent(in), opt
1286      real(dl) fact, scale, BB,
1287      character(LEN=*) LensPotF

1288      !output file of dimension
1289      !This is the format used

1290
1291      !(Cl_scalar and scalar_ou
1292      ! - for historical reason

1293
1294      if (present(factor)) then
1295          fact = factor
1296      else
1297          fact =1
1298      end if

1299
1300      if (CP%WantScalars .and.
1301          open(unit=fileio_unit
1302
1303          !cafea
1304          !print*, '10:', filei
1305          do in=1,CP%InitPower%n
1306              do il=lmin,min(10
1307                  TT = Cl_scala
1308                  EE = Cl_scala
1309                  TE = Cl_scala
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1360

```
1360      !Write out L TT EE BB TE
1361      !This input supported by
1362      implicit none
1363      integer in,il
1364      real(dl), intent(in), opt
1365      real(dl) fact, scale, BB,
1366      character(LEN=*) LensPotF
1367      integer unit

1368      !output file of dimension
1369      !This is the format used

1370
1371      !(Cl_scalar and scalar_ou
1372      ! - for historical reason

1373
1374      if (present(factor)) then
1375          fact = factor
1376      else
1377          fact =1
1378      end if

1379
1380      if (CP%WantScalars .and.
1381          unit = open_file_he

1382
1383          do in=1,CP%InitPower%
1384              do il=lmin,min(10
1385                  TT = Cl_scala
1386                  EE = Cl_scala
1387                  TE = Cl_scala
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1310

```
1310      if (CP%WantTe
1311          TT= TT+C1
1312          EE= EE+C1
1313          TE= TE+C1
1314          BB= C1_te
1315      else
1316          BB=0
1317      end if
1318      scale = (real
1319
1320      write(fileio_
1321          (real(il+
1322      end do
1323      do il=10100,CP%Ma
1324          scale = (real
1325          write(fileio_
1326              scale*C1_
1327              (real(il+
1328      end do
1329      end do
1330      close(fileio_unit)
1331  end if
1332  end subroutine output_len
1333
1334  !#SimDataAdd
1335  !      subroutine OutputSimData
1336  !          implicit none
1337
1338  !          type(CAMBParams) ::
1339
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1387

```
1387      if (CP%WantTe
1388          TT= TT+C1
1389          EE= EE+C1
1390          TE= TE+C1
1391          BB= C1_te
1392      else
1393          BB=0
1394      end if
1395      scale = (real
1396
1397      write(unit,'(
1398          (real(il+
1399      end do
1400      do il=10100,CP%Ma
1401          scale = (real
1402          write(unit,'(
1403              scale*C1_
1404              (real(il+
1405      end do
1406      end do
1407      close(unit)
1408  end if
1409  end subroutine output_len
1410
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1340

```
1340 ! integer :: nc,ll
1341 ! real(dl) :: NoiseCl(
1342 ! real(dl) :: sigma2(P
1343 ! real(dl) :: TT,EE,TE
1344 ! character(LEN=100) :
1345
1346 ! cmboutnate = trim(da
1347 ! setoutname = trim(da
1348
1349 ! xlc = sqrt(8.d0*log(
1350 ! fwhm_rad = P%fwhm_ar
1351 ! sigma2 = (fwhm_rad/x
1352
1353 ! NoiseCl = 0.d0
1354
1355 ! do ll = P%lmin_cmb:P
1356 ! do nc = 1,P%
1357 ! Nois
1358 !
1359 !
1360 ! Nois
1361 !
1362 !
1363 ! end do
1364 ! end do
1365
1366 ! NoiseCl(1,:) = 1.d0/
1367 ! NoiseCl(2,:) = 1.d0/
1368
1369 ! call CreateTxtFile(c
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1411

```


```

```
1370
1371      !      do ll = P%lmin_cmb:P
1372      !          TT = Cl_scal
1373      !          EE = Cl_scal
1374      !          TE = Cl_scal
1375      !          if(P%WantTen
1376      !              TT =
1377      !              EE =
1378      !              TE =
1379      !              BB =
1380      !          else
1381      !              BB =
1382      !          end if
1383
1384      !          TT = TT*2.d0
1385      !          Nois
1386      !          EE = EE*2.d0
1387      !          Nois
1388      !          TE = TE*2.d0
1389      !          if(P%WantTen
1390      !              BB =
1391      !
1392      !          end if
1393
1394      !          write(34, '(1
1395      !
1396      !      end do
1397
1398      !      call CreateTxtFile(s
1399      !      write (37, '(A,A)')
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1400

```
1400 !       write (37,'(A,L1)')
1401 !       write (37,'(A,L1)')
1402 !       write (37,'(A,I5)')
1403 !       write (37,'(A,A)')
1404 !       if (num_cls == 5) wr
1405 !       write (37,'(A,I5)')
1406 !       write (37,'(A,I5)')
1407 !       write (37,'(A,A)')
1408 !       close(37)
1409
1410 !       end subroutine OutputSim
1411 !#SimDataAdd
1412
1413       subroutine output_veccl_f
1414       implicit none
1415       integer in,il
1416       character(LEN=*) VecFile
1417       real(dl), intent(in), opt
1418       real(dl) fact
1419
1420
1421       if (present(factor)) then
1422         fact = factor
1423       else
1424         fact =1
1425       end if
1426
1427
1428       if (CP%WantVectors .and.
1429         open(unit=fileio_unit
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1411

```
1411
1412       subroutine output_veccl_f
1413       implicit none
1414       integer in,il
1415       character(LEN=*) VecFile
1416       real(dl), intent(in), opt
1417       real(dl) fact
1418       integer unit
1419
1420       if (present(factor)) then
1421         fact = factor
1422       else
1423         fact =1
1424       end if
1425
1426
1427       if (CP%WantVectors .and.
1428         unit = open_file_he
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1430

```
1430      do in=1,CP%InitPower%
1431          do il=lmin,CP%Max
1432              write(fileio_
1433                  end do
1434          end do
1435
1436      close(fileio_unit)
1437  end if
1438
1439  end subroutine output_vec
1440
1441  subroutine NormalizeClsAt
1442  implicit none
1443  integer, intent(IN) :: ln
1444  integer in
1445  real(dl) Norm
1446
1447  do in=1,CP%InitPower%nn
1448      if (CP%WantScalars) t
1449          Norm=1/Cl_scalar(
1450          Cl_scalar(lmin:CP
1451      end if
1452
1453      if (CP%WantTensors) t
1454          if (.not.CP%WantS
1455              !Otherwise Norm a
1456              Cl_tensor(lmin:CP
1457              Cl_tensor(lmi
1458          end if
1459  end do
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1429

```
1429      do in=1,CP%InitPower%
1430          do il=lmin,CP%Max
1431              write(unit,'(
1432          end do
1433      end do
1434      close(unit)
1435
1436  end if
1437
1438  end subroutine output_vec
1439
1440  subroutine NormalizeClsAt
1441  implicit none
1442  integer, intent(IN) :: ln
1443  integer in
1444  real(dl) Norm
1445
1446  do in=1,CP%InitPower%nn
1447      if (CP%WantScalars) t
1448          Norm=1/Cl_scalar(
1449          Cl_scalar(lmin:CP
1450      end if
1451
1452      if (CP%WantTensors) t
1453          if (.not.CP%WantS
1454              !Otherwise Norm a
1455              Cl_tensor(lmin:CP
1456              Cl_tensor(lmi
1457  end if
1458  end do
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1460

```
1460
1461     end subroutine Normalize
1462
1463     subroutine ModelData_Free
1464
1465     call Free_ClTransfer(CTra
1466     call Free_ClTransfer(CTra
1467     call Free_ClTransfer(CTra
1468     if (allocated(Cl_vector))
1469     if (allocated(Cl_tensor))
1470     if (allocated(Cl_scalar))
1471     if (allocated(Cl_lensed))
1472     if (allocated(Cl_scalar_a
1473
1474     end subroutine ModelData_
1475
1476     end module ModelData
1477
1478
1479     !cccccccccccccccccccccccccccccccccc
1480     module MassiveNu
1481     use precision
1482     use ModelParams
1483     implicit none
1484     private
1485
1486     real(dl), parameter :: c
1487     !const = int q^3 F(q) dq
1488     real(dl), parameter :: c
1489     real(dl), parameter :: z
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1458

```
1458
1459     end subroutine Normalize
1460
1461     subroutine ModelData_Free
1462
1463     call Free_ClTransfer(CTra
1464     call Free_ClTransfer(CTra
1465     call Free_ClTransfer(CTra
1466     if (allocated(Cl_vector))
1467     if (allocated(Cl_tensor))
1468     if (allocated(Cl_scalar))
1469     if (allocated(Cl_lensed))
1470     if (allocated(Cl_scalar_a
1471
1472     end subroutine ModelData_
1473
1474     end module ModelData
1475
1476
1477     !cccccccccccccccccccccccccccccccccc
1478     module MassiveNu
1479     use precision
1480     use ModelParams
1481     implicit none
1482     private
1483
1484     real(dl), parameter :: c
1485     !const = int q^3 F(q) dq
1486     real(dl), parameter :: c
1487     real(dl), parameter :: z
```



```
1517
1518      subroutine Nu_init
1519
1520      ! Initialize interpolati
1521      ! Use cubic splines inte
1522
1523      integer i
1524      real(dl) dq,dlfdlq, q, am
1525      real(dl) spline_data(nrho
1526
1527      ! nu_masses=m_nu(i)*c**2
1528      ! Get number density n o
1529      ! rho_massless/n = int q
1530      ! then m = Omega_nu/N_nu
1531      ! Error due to velocity
1532
1533      do i=1, CP%Nu_mass_eigens
1534          nu_masses(i)=const/(1
```

```
1518
1519      subroutine sum_mnu_for_m1
1520      use constants
1521      real(dl), intent(in) :: m
1522      real(dl), intent(out) ::
1523      real(dl) :: m2,m3
1524
1525      m2 = sqrt(m1**2 + delta_m
1526      m3 = sqrt(m1**2 + sgn*del
1527      summnu = m1 + m2 + m3 - t
1528      dsummnu = m1/m2+m1/m3 + 1
1529
1530      end subroutine sum_mnu_fo
1531
1532      subroutine Nu_init
1533      ! Initialize interpolati
1534      ! Use cubic splines inte
1535
1536      integer i
1537      real(dl) dq,dlfdlq, q, am
1538      real(dl) spline_data(nrho
1539
1540      ! nu_masses=m_nu(i)*c**2
1541      ! Get number density n o
1542      ! rho_massless/n = int q
1543      ! then m = Omega_nu/N_nu
1544      ! Error due to velocity
1545
1546      do i=1, CP%Nu_mass_eigens
1547          nu_masses(i)=const/(1
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1535

```
1535      /CP%Nu_mass_degen
1536      end do
1537
1538      if (allocated(r1)) return
1539      allocate(r1(nrhopn),p1(nr
1540
1541
1542      nqmax=3
1543      if (AccuracyBoost >1) nqm
1544      if (AccuracyBoost >2) nqm
1545      if (AccuracyBoost >3) nqm
1546      !note this may well be wo
1547
1548      if (nqmax > nqmax0) call
1549
1550      !We evolve evolve 4F_1/dl
1551      !Integration scheme gets
1552      !see CAMB notes
1553      if (nqmax==3) then
1554          !Accurate at 2e-4 lev
1555          nu_q(1:3) = (/0.91320
1556          nu_int_kernel(1:3) =
1557      else if (nqmax==4) then
1558          !This seems to be ver
1559          nu_q(1:4) = (/0.7, 2.
1560          nu_int_kernel(1:4) =
1561      else if (nqmax==5) then
1562          !exact for n=-4,-2..3
1563          !This seems to be ver
1564          nu_q(1:5) = (/0.58316
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1547

```
1547      /CP%Nu_mass_degen
1548      end do
1549
1550      if (allocated(r1)) return
1551      allocate(r1(nrhopn),p1(nr
1552
1553
1554      nqmax=3
1555      if (AccuracyBoost >1) nqm
1556      if (AccuracyBoost >2) nqm
1557      if (AccuracyBoost >3) nqm
1558      !note this may well be wo
1559
1560      if (nqmax > nqmax0) call
1561
1562      !We evolve evolve 4F_1/dl
1563      !Integration scheme gets
1564      !see CAMB notes
1565      if (nqmax==3) then
1566          !Accurate at 2e-4 lev
1567          nu_q(1:3) = (/0.91320
1568          nu_int_kernel(1:3) =
1569      else if (nqmax==4) then
1570          !This seems to be ver
1571          nu_q(1:4) = (/0.7, 2.
1572          nu_int_kernel(1:4) =
1573      else if (nqmax==5) then
1574          !exact for n=-4,-2..3
1575          !This seems to be ver
1576          nu_q(1:5) = (/0.58316
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1565

```
1565      nu_int_kernel(1:5) =
1566      else
1567          dq = (12 + nqmax/5)/r
1568          do i=1,nqmax
1569              q=(i-0.5d0)*dq
1570              nu_q(i) = q
1571              dlfdlq=-q/(1._dl+
1572              nu_int_kernel(i)=
1573          end do
1574      end if
1575      nu_int_kernel=nu_int_kern
1576
1577      dlnam=-(log(am_min/am_max
1578
1579
1580      !$OMP PARALLEL DO DEFAULT
1581      !$OMP & PRIVATE(am, rhonu
1582      do i=1,nrhopn
1583          am=am_min*exp((i-1)*dlnam
1584          call nuRhoPres(am,rhonu,p
1585          r1(i)=log(rhonu)
1586          p1(i)=log(pnu)
1587      end do
1588      !$OMP END PARALLEL DO
1589
1590
1591      call splini(spline_data,n
1592      call splder(r1,dr1,nrhopn
1593      call splder(p1,dp1,nrhopn
1594      call splder(dr1,ddr1,nrho
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1577

```
1577      nu_int_kernel(1:5) =
1578      else
1579          dq = (12 + nqmax/5)/r
1580          do i=1,nqmax
1581              q=(i-0.5d0)*dq
1582              nu_q(i) = q
1583              dlfdlq=-q/(1._dl+
1584              nu_int_kernel(i)=
1585          end do
1586      end if
1587      nu_int_kernel=nu_int_kern
1588
1589      dlnam=-(log(am_min/am_max
1590
1591
1592      !$OMP PARALLEL DO DEFAULT
1593      !$OMP & PRIVATE(am, rhonu
1594      do i=1,nrhopn
1595          am=am_min*exp((i-1)*d
1596          call nuRhoPres(am,rho
1597          r1(i)=log(rhonu)
1598          p1(i)=log(pnu)
1599      end do
1600      !$OMP END PARALLEL DO
1601
1602
1603      call splini(spline_data,n
1604      call splder(r1,dr1,nrhopn
1605      call splder(p1,dp1,nrhopn
1606      call splder(dr1,ddr1,nrho
```

```
1595
1596
1597     end subroutine Nu_init
1598
1599     !cccccccccccccccccccccccccccccccccccccccc
1600     subroutine nuRhoPres(am,r
1601     !   Compute the density an
1602     !   in units of the mean d
1603
1604     real(dl), parameter :: q
1605     integer, parameter :: nq=
1606     real(dl) dum1(nq+1),dum2(
1607     real(dl), intent(in) :: a
1608     real(dl), intent(out) ::
1609     integer i
1610     real(dl) q,aq,v,aqdn,adq
1611
1612
1613     !   q is the comoving mome
1614     !   Integrate up to qmax a
1615     adq=qmax/nq
1616     dum1(1)=0._dl
1617     dum2(1)=0._dl
1618     do i=1,nq
1619         q=i*adq
1620         aq=am/q
1621         v=1._dl/sqrt(1._dl+aq
1622         aqdn=adq*q*q*q/(exp(q
1623         dum1(i+1)=aqdn/v
1624         dum2(i+1)=aqdn*v
```

```
1607
1608
1609     end subroutine Nu_init
1610
1611     !cccccccccccccccccccccccccccccccccccccccc
1612     subroutine nuRhoPres(am,r
1613     !   Compute the density an
1614     !   in units of the mean d
1615
1616     real(dl), parameter :: q
1617     integer, parameter :: nq=
1618     real(dl) dum1(nq+1),dum2(
1619     real(dl), intent(in) :: a
1620     real(dl), intent(out) ::
1621     integer i
1622     real(dl) q,aq,v,aqdn,adq
1623
1624
1625     !   q is the comoving mome
1626     !   Integrate up to qmax a
1627     adq=qmax/nq
1628     dum1(1)=0._dl
1629     dum2(1)=0._dl
1630     do i=1,nq
1631         q=i*adq
1632         aq=am/q
1633         v=1._dl/sqrt(1._dl+aq
1634         aqdn=adq*q*q*q/(exp(q
1635         dum1(i+1)=aqdn/v
1636         dum2(i+1)=aqdn*v
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1625

```
1625      end do
1626      call splint(dum1,rhonu,nq
1627      call splint(dum2,pnu,nq+1
1628      ! Apply asymptotic corrr
1629      ! energy density.
1630      rhonu=(rhonu+dum1(nq+1)/a
1631      pnu=(pnu+dum2(nq+1)/adq)/
1632
1633      end subroutine nuRhoPres
1634
1635      !cccccccccccccccccccccccccccccccc
1636      subroutine Nu_background(
1637      use precision
1638      use ModelParams
1639      real(dl), intent(in) :: a
1640      real(dl), intent(out) ::
1641
1642      ! Compute massive neutri
1643      ! density of one eigenst
1644      ! interpolate from a tab
1645
1646      real(dl) d
1647      integer i
1648
1649      if (am <= am_minp) then
1650          rhonu=1._dl + const2*
1651          pnu=(2-rhonu)/3._dl
1652          return
1653      else if (am >= am_maxp) t
1654          rhonu = 3/(2*const)*(
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1637

```
1637      end do
1638      call splint(dum1,rhonu,nq
1639      call splint(dum2,pnu,nq+1
1640      ! Apply asymptotic corrr
1641      ! energy density.
1642      rhonu=(rhonu+dum1(nq+1)/a
1643      pnu=(pnu+dum2(nq+1)/adq)/
1644
1645      end subroutine nuRhoPres
1646
1647      !cccccccccccccccccccccccccccccccc
1648      subroutine Nu_background(
1649      use precision
1650      use ModelParams
1651      real(dl), intent(in) :: a
1652      real(dl), intent(out) ::
1653
1654      ! Compute massive neutri
1655      ! density of one eigenst
1656      ! interpolate from a tab
1657
1658      real(dl) d
1659      integer i
1660
1661      if (am <= am_minp) then
1662          rhonu=1._dl + const2*
1663          pnu=(2-rhonu)/3._dl
1664          return
1665      else if (am >= am_maxp) t
1666          rhonu = 3/(2*const)*(
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1655

```
1655      pnu = 900._dl/120._dl
1656      return
1657  end if
1658
1659
1660      d=log(am/am_min)/dlnam+1.
1661      i=int(d)
1662      d=d-i
1663
1664      ! Cubic spline interpola
1665      rhonu=r1(i)+d*(dr1(i)+d*(
1666          -dr1(i+1)+d*(dr1(i)+d
1667      pnu=p1(i)+d*(dp1(i)+d*(3.
1668          -dp1(i+1)+d*(dp1(i)+d
1669      rhonu=exp(rhonu)
1670      pnu=exp(pnu)
1671
1672  end subroutine Nu_backgro
1673
1674  !cccccccccccccccccccccccccccccc
1675  subroutine Nu_rho(am,rhon
1676  use precision
1677  use ModelParams
1678  real(dl), intent(in) :: a
1679  real(dl), intent(out) ::
1680
1681      ! Compute massive neutri
1682      ! density of one eigenst
1683      ! interpolate from a tab
1684
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1667

```
1667      pnu = 900._dl/120._dl
1668      return
1669  end if
1670
1671
1672      d=log(am/am_min)/dlnam+1.
1673      i=int(d)
1674      d=d-i
1675
1676      ! Cubic spline interpola
1677      rhonu=r1(i)+d*(dr1(i)+d*(
1678          -dr1(i+1)+d*(dr1(i)+d
1679      pnu=p1(i)+d*(dp1(i)+d*(3.
1680          -dp1(i+1)+d*(dp1(i)+d
1681      rhonu=exp(rhonu)
1682      pnu=exp(pnu)
1683
1684  end subroutine Nu_backgro
1685
1686  !cccccccccccccccccccccccccccccc
1687  subroutine Nu_rho(am,rhon
1688  use precision
1689  use ModelParams
1690  real(dl), intent(in) :: a
1691  real(dl), intent(out) ::
1692
1693      ! Compute massive neutri
1694      ! density of one eigenst
1695      ! interpolate from a tab
1696
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1715

```
1715      real(dl) d
1716      real(dl), intent(IN) :: a
1717      integer i
1718
1719      if (am< am_minp) then
1720          rhonudot = 2*const2*a
1721      else if (am>am_maxp) then
1722          rhonudot = 3/(2*const
1723      else
1724          d=log(am/am_min)/dlna
1725          i=int(d)
1726          d=d-i
1727          ! Cubic spline inter
1728          rhonudot=dr1(i)+d*(dd
1729              -2._dl*ddr1(i)-dd
1730              +2._dl*(dr1(i)-dr
1731
1732          rhonudot=rhonu*adotoa
1733      end if
1734
1735      end function Nu_drho
1736
1737      end module MassiveNu
1738
1739      ! wrapper function to avo
1740      subroutine init_massive_n
1741      use MassiveNu
1742      use ModelParams
1743      implicit none
1744      logical, intent(IN) :: ha
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1727

```
1727      real(dl) d
1728      real(dl), intent(IN) :: a
1729      integer i
1730
1731      if (am< am_minp) then
1732          rhonudot = 2*const2*a
1733      else if (am>am_maxp) then
1734          rhonudot = 3/(2*const
1735      else
1736          d=log(am/am_min)/dlna
1737          i=int(d)
1738          d=d-i
1739          ! Cubic spline inter
1740          rhonudot=dr1(i)+d*(dd
1741              -2._dl*ddr1(i)-dd
1742              +2._dl*(dr1(i)-dr
1743
1744          rhonudot=rhonu*adotoa
1745      end if
1746
1747      end function Nu_drho
1748
1749      end module MassiveNu
1750
1751      ! wrapper function to avo
1752      subroutine init_massive_n
1753      use MassiveNu
1754      use ModelParams
1755      implicit none
1756      logical, intent(IN) :: ha
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1745

```
1745
1746     if (has_massive_nu) then
1747         call Nu_Init
1748     else
1749         nu_masses = 0
1750     end if
1751 end subroutine init_massi
1752
1753
1754     !cccccccccccccccccccccccccccccccccccccccc
1755
1756     module Transfer
1757     use ModelData
1758     use Errors
1759     implicit none
1760     public
1761     !#SimDataReplace
1762     integer, parameter :: Tra
1763         Transfer_r=5, Transfe
1764     Transfer_tot=7, Transfer_
1765         ! total perturbations
1766         Transfer_Weyl = 10, &
1767     Transfer_Newt_vel_cdm=11,
1768     Transfer_vel_baryon_cdm =
1769     Transfer_Psi=14
1770     !ana simdata : Aici e modific
1771
1772     ! integer, parameter :: Tr
1773
1774     integer, parameter :: Tr
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1757

```
1757
1758     if (has_massive_nu) then
1759         call Nu_Init
1760     else
1761         nu_masses = 0
1762     end if
1763 end subroutine init_massi
1764
1765
1766     !cccccccccccccccccccccccccccccccccccccccc
1767
1768     module Transfer
1769     use ModelData
1770     use Errors
1771     implicit none
1772     public
1773
1774     integer, parameter :: Tra
1775         Transfer_r=5, Transfe
1776     Transfer_tot=7, Trans
1777         ! total perturbations
1778         Transfer_Weyl = 10, &
1779     Transfer_Newt_vel_cdm =
1780
1781     integer, parameter :: Tra
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1775

```
1775 !#SimDataReplace
1776
1777     logical :: transfer_inter
1778     !set to false to output c
1779
1780     integer :: transfer_power
1781     !What to use to calculate
1782     !Transfer_tot uses total
1783
1784     logical :: get_growth_sig
1785     !gets sigma_vdelta, like
1786     !in late LCDM f*sigma8 =
1787
1788     Type MatterTransferData
1789         !Computed data
1790         integer :: num_q_t
1791         real(dl), dimension (
1792         real(dl), dimension (
1793         real(dl), dimension (
1794         real, dimension(:, :, :
1795         !TransferData(entry, k
1796     end Type MatterTransferData
1797
1798     Type MatterPowerData
1799     !everything is a func
1800     integer :: num_k,
1801     real(dl), dimension (
1802     !matpower is log(P_k)
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1782

```
1782     character(len=name_tag_le
1783         [ 'CDM      ', 'baryon
1784         'no_nu     ', 'total_de
1785
1786     logical :: transfer_inter
1787     !set to false to output c
1788
1789     integer :: transfer_power
1790     !What to use to calculate
1791     !Transfer_tot uses total
1792
1793     logical :: get_growth_sig
1794     !gets sigma_vdelta, like
1795     !in late LCDM f*sigma8 =
1796
1797     Type MatterTransferData
1798         !Computed data
1799         integer :: num_q_t
1800         real(dl), dimension (
1801         real(dl), dimension (
1802         real(dl), dimension (
1803         real, dimension(:, :, :
1804         !TransferData(entry, k
1805     end Type MatterTransferData
1806
1807     Type MatterPowerData
1808     !everything is a func
1809     integer :: num_k,
1810     real(dl), dimension (
1811     !matpower is log(P_k)
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1803

```
1803      real(dl), dimension(:
1804      !if NonLinear, nonlin
1805      !function of k and re
1806      real(dl), dimension(:
1807  end Type MatterPowerData
1808
1809  Type (MatterTransferData)
1810
1811  interface Transfer_GetMat
1812  module procedure Transfer
1813  end interface
1814
1815  contains
1816
1817  subroutine Transfer_GetUn
1818  !Get  $2\pi^2/k^3$  T_1 T_2 P_
1819  Type(MatterTransferData)
1820  real(dl), intent(inout)::
1821  integer, optional, intent
1822  integer, optional, intent
1823  logical, optional, intent
1824  real(dl) h, k
1825  integer nz, nk, zix, ik
1826  integer s1, s2
1827  logical hnorm
1828
1829
1830  s1 = transfer_power_var
1831  if (present(var1)) s1 =
1832  s2 = transfer_power_var
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1812

```
1812      real(dl), dimension(:
1813      !if NonLinear, nonlin
1814      !function of k and re
1815      real(dl), dimension(:
1816  end Type MatterPowerData
1817
1818  Type (MatterTransferData)
1819
1820  interface Transfer_GetMat
1821  module procedure Transfer
1822  end interface
1823
1824  contains
1825
1826  subroutine Transfer_GetUn
1827  !Get  $2\pi^2/k^3$  T_1 T_2 P_
1828  Type(MatterTransferData)
1829  real(dl), intent(inout)::
1830  integer, optional, intent
1831  integer, optional, intent
1832  logical, optional, intent
1833  real(dl) h, k
1834  integer nz, nk, zix, ik
1835  integer s1, s2
1836  logical hnorm
1837
1838
1839  s1 = transfer_power_var
1840  if (present(var1)) s1 =
1841  s2 = transfer_power_var
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1833

```
1833      if (present(var2)) s2 =
1834      hnorm = .true.
1835      if (present(hubble_units)
1836
1837      nk=M%num_q_trans
1838      nz=CP%Transfer%PK_num_red
1839      if (nk/= size(PK,1) .or.
1840
1841      h = CP%H0/100
1842
1843      do ik=1,nk
1844          k = M%TransferData(Tr
1845          do zix=1,nz
1846              PK(ik,zix) = M%Tr
1847                  M%TransferDat
1848          end do
1849      end do
1850      if (hnorm) PK= PK * h**3
1851      !print*, 'sinozaur:', PK
1852      !print*, 'sinozaur:',tra
1853      end subroutine Transfer_G
1854
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1841

```
1841      if (present(var2)) s2 =
1842      hnorm = .true.
1843      if (present(hubble_units)
1844
1845      nk=M%num_q_trans
1846      nz=CP%Transfer%PK_num_red
1847      if (nk/= size(PK,1) .or.
1848
1849      h = CP%H0/100
1850
1851      do ik=1,nk
1852          k = M%TransferData(Tr
1853          do zix=1,nz
1854              PK(ik,zix) = M%Tr
1855                  M%TransferDat
1856          end do
1857      end do
1858      if (hnorm) PK= PK * h**3
1859
1860      end subroutine Transfer_G
1861
1862      subroutine Transfer_GetUn
1863      !Get  $2\pi^2/k^3 T_1 T_2 P$ 
1864      Type(MatterTransferData),
1865      real(dl), intent(inout)::
1866      integer, optional, intent
1867      integer, optional, intent
1868      logical, optional, intent
1869      Type(MatterPowerData) ::
```

```
1855      subroutine Transfer_GetMa
1856      !Does *NOT* include non-l
1857      !Get total matter power s
1858      !Here there definition is
1859      !We are assuming that Cls
1860      !sepctrum is generated to
1861      Type(MatterTransferData),
1862      Type(MatterPowerData) ::
1863      integer, intent(in), opti
1864      integer, intent(in), opti
1865      integer, intent(in), opti
1866      real(dl) h, kh, k, power
1867      integer ik
1868      integer nz,itf, itf_start
1869      integer :: s1,s2, p_ix
1870
1871
```

```
1870      integer zix
1871
1872      call Transfer_GetUnspline
1873      do zix=1, CP%Transfer%PK_
1874          call Transfer_GetMatt
1875              CP%Transfer%PK_re
1876          call NonLinear_GetRat
1877          PK(:,zix) = PK(:,zix
1878          call MatterPowerdata_
1879      end do
1880
1881      end subroutine Transfer_G
1882
1883      subroutine Transfer_GetMa
1884      !Does *NOT* include non-l
1885      !Get total matter power s
1886      !Here there definition is
1887      !We are assuming that Cls
1888      !sppectrum is generated to
1889      Type(MatterTransferData),
1890      Type(MatterPowerData) ::
1891      integer, intent(in), opti
1892      integer, intent(in), opti
1893      integer, intent(in), opti
1894      real(dl) h, kh, k, power
1895      integer ik
1896      integer nz,itf, itf_start
1897      integer :: s1,s2, p_ix
1898
1899
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1872

```
1872      s1 = transfer_power_var
1873      if (present(var1)) s1 =
1874      s2 = transfer_power_var
1875      if (present(var2)) s2 =
1876      p_ix = 1
1877      if (present(power_ix)) p_
1878
1879      if (present(itf_only)) th
1880          itf_start=itf_only
1881          itf_end = itf_only
1882          nz = 1
1883      else
1884          itf_start=1
1885          nz= size(MTrans%Trans
1886          itf_end = nz
1887      end if
1888      PK_data%num_k = MTrans%nu
1889      PK_Data%num_z = nz
1890
1891      allocate(PK_data%matpower
1892      allocate(PK_data%ddmat(PK
1893      allocate(PK_data%nonlin_r
1894      allocate(PK_data%log_kh(P
1895      allocate(PK_data%redshift
1896      PK_data%redshifts = CP%Tr
1897
1898      h = CP%H0/100
1899
1900      do ik=1,MTrans%num_q_tran
1901          kh = MTrans%TransferD
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1900

```
1900      s1 = transfer_power_var
1901      if (present(var1)) s1 =
1902      s2 = transfer_power_var
1903      if (present(var2)) s2 =
1904      p_ix = 1
1905      if (present(power_ix)) p_
1906
1907      if (present(itf_only)) th
1908          itf_start=itf_only
1909          itf_end = itf_only
1910          nz = 1
1911      else
1912          itf_start=1
1913          nz= size(MTrans%Trans
1914          itf_end = nz
1915      end if
1916      PK_data%num_k = MTrans%nu
1917      PK_Data%num_z = nz
1918
1919      allocate(PK_data%matpower
1920      allocate(PK_data%ddmat(PK
1921      allocate(PK_data%nonlin_r
1922      allocate(PK_data%log_kh(P
1923      allocate(PK_data%redshift
1924      PK_data%redshifts = CP%Tr
1925
1926      h = CP%H0/100
1927
1928      do ik=1,MTrans%num_q_tran
1929          kh = MTrans%TransferD
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1902

```
1902      k = kh*h
1903      PK_data%log_kh(ik) =
1904      power = ScalarPower(k
1905      if (global_error_flag
1906          call MatterPowerd
1907      return
1908      end if
1909      do itf = 1, nz
1910          PK_data%matpower(
1911              log(MTrans%Tr
1912              MTrans%Transf
1913              *pi*twopi*h**
1914
1915      ! print*, 'bbb: ', ik, itf, P
1916
1917      end do
1918      end do
1919
1920      ! print*, 'Modules: transf
1921      call MatterPowerdata_gets
1922
1923      end subroutine Transfer_G
1924
1925      subroutine MatterPowerDat
1926      !Loads in kh, P_k from fi
1927      !Not redshift is not stor
1928      !Also note that output _m
1929
1930      !Get total matter power s
1931      !Here there definition is
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1930

```
1930      k = kh*h
1931      PK_data%log_kh(ik) =
1932      power = ScalarPower(k
1933      if (global_error_flag
1934          call MatterPowerd
1935      return
1936      end if
1937      do itf = 1, nz
1938          PK_data%matpower(
1939              log(MTrans%Tr
1940              MTrans%Transf
1941              *pi*twopi*h**
1942
1943      end do
1944
1945      call MatterPowerdata_gets
1946
1947      end subroutine Transfer_G
1948
1949      subroutine MatterPowerDat
1950      !Loads in kh, P_k from fi
1951      !Not redshift is not stor
1952      !Also note that output _m
1953
1954      !Get total matter power s
1955      !Here there definition is
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1932

```
1932      use AmlUtils
1933      character(LEN=*) :: fname
1934      Type(MatterPowerData) ::
1935      real(dl)kh, Pk
1936      integer ik
1937      integer nz
1938
1939
1940      nz = 1
1941      call openTxtFile(fname, f
1942      !cafea
1943      !print*, '15:', fileio_unit,
1944      PK_data%num_k = FileLines
1945      PK_Data%num_z = 1
1946
1947      allocate(PK_data%matpower
1948      allocate(PK_data%ddmat(PK
1949      allocate(PK_data%nonlin_r
1950      allocate(PK_data%log_kh(P
1951
1952      allocate(PK_data%redshift
1953      PK_data%redshifts = 0
1954
1955      do ik=1,PK_data%num_k
1956          read (fileio_unit,*)
1957          PK_data%matpower(ik,1
1958          PK_data%log_kh(ik) =
1959      end do
1960
1961      call MatterPowerdata_gets
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1956

```
1956      use AmlUtils
1957      character(LEN=*) :: fname
1958      Type(MatterPowerData) ::
1959      real(dl)kh, Pk
1960      integer ik
1961      integer nz
1962
1963
1964      nz = 1
1965      call openTxtFile(fname, f
1966
1967      PK_data%num_k = FileLines
1968      PK_Data%num_z = 1
1969
1970      allocate(PK_data%matpower
1971      allocate(PK_data%ddmat(PK
1972      allocate(PK_data%nonlin_r
1973      allocate(PK_data%log_kh(P
1974
1975      allocate(PK_data%redshift
1976      PK_data%redshifts = 0
1977
1978      do ik=1,PK_data%num_k
1979          read (fileio_unit,*)
1980          PK_data%matpower(ik,1
1981          PK_data%log_kh(ik) =
1982      end do
1983
1984      call MatterPowerdata_gets
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1962

```
1962
1963         end subroutine MatterPowe
1964
1965
1966         subroutine MatterPowerdat
1967         Type(MatterPowerData) ::
1968         integer i
1969         real(dl), parameter :: cl
1970
1971         do i = 1,PK_Data%num_z
1972             call spline(PK_data%l
1973                 cllo,clhi,PK_data
1974         end do
1975
1976         end subroutine MatterPowe
1977
1978         subroutine MatterPowerdat
1979         Type(MatterPowerData) ::
1980
1981         call NonLinear_GetRatios(
1982         PK_data%matpower = PK_dat
1983         call MatterPowerdata_gets
1984
1985         end subroutine MatterPowe
1986
1987         subroutine MatterPowerdat
1988         Type(MatterPowerData) ::
1989         integer i
1990
1991         deallocate(PK_data%log_kh
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 1985

```
1985
1986         end subroutine MatterPowe
1987
1988
1989         subroutine MatterPowerdat
1990         Type(MatterPowerData) ::
1991         integer i
1992         real(dl), parameter :: cl
1993
1994         do i = 1,PK_Data%num_z
1995             call spline(PK_data%l
1996                 cllo,clhi,PK_data
1997         end do
1998
1999         end subroutine MatterPowe
2000
2001         subroutine MatterPowerdat
2002         Type(MatterPowerData) ::
2003
2004         call NonLinear_GetRatios(
2005         PK_data%matpower = PK_dat
2006         call MatterPowerdata_gets
2007
2008         end subroutine MatterPowe
2009
2010         subroutine MatterPowerdat
2011         Type(MatterPowerData) ::
2012         integer i
2013
2014         deallocate(PK_data%log_kh
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 1992

```

1992      deallocate(PK_data%matpow
1993      deallocate(PK_data%ddmat,
1994      deallocate(PK_data%nonlin
1995      deallocate(PK_data%redshi
1996      call MatterPowerdata_Null
1997
1998      end subroutine MatterPowe
1999
2000      subroutine MatterPowerdat
2001      Type(MatterPowerData) ::
2002
2003      nullify(PK_data%log_kh)
2004      nullify(PK_data%nonlin_ra
2005      nullify(PK_data%redshifts
2006
2007      end subroutine MatterPowe
2008
2009      function MatterPowerData_
2010      !Get matter power spectru
2011      Type(MatterPowerData) ::
2012      integer, intent(in) :: it
2013      real (dl), intent(in) ::
2014      real(dl) :: logk
2015      integer llo,lhi
2016      real(dl) outpower, dp
2017      real(dl) ho,a0,b0
2018      integer, save :: i_last =
2019
2020      logk = log(kh)
2021      if (logk < PK%log_kh(1))

```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2015

```

2015      deallocate(PK_data%matpow
2016      deallocate(PK_data%ddmat,
2017      deallocate(PK_data%nonlin
2018      deallocate(PK_data%redshi
2019      call MatterPowerdata_Null
2020
2021      end subroutine MatterPowe
2022
2023      subroutine MatterPowerdat
2024      Type(MatterPowerData) ::
2025
2026      nullify(PK_data%log_kh)
2027      nullify(PK_data%nonlin_ra
2028      nullify(PK_data%redshifts
2029
2030      end subroutine MatterPowe
2031
2032      function MatterPowerData_
2033      !Get matter power spectru
2034      Type(MatterPowerData) ::
2035      integer, intent(in) :: it
2036      real (dl), intent(in) ::
2037      real(dl) :: logk
2038      integer llo,lhi
2039      real(dl) outpower, dp
2040      real(dl) ho,a0,b0
2041      integer, save :: i_last =
2042
2043      logk = log(kh)
2044      if (logk < PK%log_kh(1))

```

/Users/lplopa/Compare/camb_simdata/modules.f90, Top line: 2022

```
2022      dp = (PK%matpower(2,i
2023          ( PK%log_kh(2)-PK
2024      outpower = PK%matpowe
2025      else if (logk > PK%log_kh
2026          !Do dodgy linear extr
2027
2028      dp = (PK%matpower(PK%
2029          ( PK%log_kh(PK%nu
2030      outpower = PK%matpowe
2031      else
2032          llo=min(i_last,PK%num
2033          do while (PK%log_kh(1
2034              llo=llo-1
2035          end do
2036          do while (PK%log_kh(1
2037              llo=llo+1
2038          end do
2039          i_last =llo
2040          lhi=llo+1
2041          ho=PK%log_kh(lhi)-PK%
2042          a0=(PK%log_kh(lhi)-lo
2043          b0=1-a0
2044
2045      outpower = a0*PK%matp
2046          ((a0**3-a0)* PK%d
2047          +(b0**3-b0)*PK%dd
2048      end if
2049
2050      outpower = exp(max(-30._d
2051
```

/Users/lplopa/Compare/camb_des/modules.f90, Top line: 2045

```
2045      dp = (PK%matpower(2,i
2046          ( PK%log_kh(2)-PK
2047      outpower = PK%matpowe
2048      else if (logk > PK%log_kh
2049          !Do dodgy linear extr
2050
2051      dp = (PK%matpower(PK%
2052          ( PK%log_kh(PK%nu
2053      outpower = PK%matpowe
2054      else
2055          llo=min(i_last,PK%num
2056          do while (PK%log_kh(1
2057              llo=llo-1
2058          end do
2059          do while (PK%log_kh(1
2060              llo=llo+1
2061          end do
2062          i_last =llo
2063          lhi=llo+1
2064          ho=PK%log_kh(lhi)-PK%
2065          a0=(PK%log_kh(lhi)-lo
2066          b0=1-a0
2067
2068      outpower = a0*PK%matp
2069          ((a0**3-a0)* PK%d
2070          +(b0**3-b0)*PK%dd
2071      end if
2072
2073      outpower = exp(outpower)
2074
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2052

```
2052      end function MatterPowerD
2053
2054      subroutine Transfer_GetMa
2055      Type(MatterTransferData),
2056      integer, intent(in) :: it
2057      integer, intent(in), opti
2058      real, intent(out) :: outp
2059      real, intent(in) :: minkh
2060      real(dl) :: outpowerd(npo
2061      real(dl):: minkhd, dlnkhd
2062
2063      minkhd = minkh; dlnkhd =
2064      call Transfer_GetMatterPo
2065      outpower(1:npoints) = out
2066
2067      end subroutine Transfer_G
2068
2069      !JD 08/13 for nonlinear l
2070      !Changed input variable f
2071      !redshift in the PK_redsh
2072      !array, itf, is given by
2073      !Also changed (CP%NonLine
2074      !CP%NonLinear/=NonLinear_
2075      subroutine Transfer_GetMa
2076      !Allows for non-smooth pr
2077      !if CP%Nonlinear/ = NonLi
2078      !Get total matter power s
2079      !in units of (h Mpc^{-1})
2080      !Here there definition is
2081      !We are assuming that Cls
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2075

```
2075      end function MatterPowerD
2076
2077      subroutine Transfer_GetMa
2078      Type(MatterTransferData),
2079      integer, intent(in) :: it
2080      integer, intent(in), opti
2081      real, intent(out) :: outp
2082      real, intent(in) :: minkh
2083      real(dl) :: outpowerd(npo
2084      real(dl):: minkhd, dlnkhd
2085
2086      minkhd = minkh; dlnkhd =
2087      call Transfer_GetMatterPo
2088      outpower(1:npoints) = out
2089
2090      end subroutine Transfer_G
2091
2092      !JD 08/13 for nonlinear l
2093      !Changed input variable f
2094      !redshift in the PK_redsh
2095      !array, itf, is given by
2096      !Also changed (CP%NonLine
2097      !CP%NonLinear/=NonLinear_
2098      subroutine Transfer_GetMa
2099      !Allows for non-smooth pr
2100      !if CP%Nonlinear/ = NonLi
2101      !Get total matter power s
2102      !in units of (h Mpc^{-1})
2103      !Here there definition is
2104      !We are assuming that Cls
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2082

```
2082      !sepctrum is generated to
2083      Type(MatterTransferData),
2084      Type(MatterPowerData) ::
2085
2086      integer, intent(in) :: it
2087      real(dl), intent(out) ::
2088      real(dl), intent(in) :: m
2089      integer, intent(in), opti
2090
2091      real(dl), parameter :: cl
2092      integer ik, llo,il,lhi,la
2093      real(dl) matpower(MTrans%
2094      real(dl) atransfer,xi, a0
2095      integer itf
2096      integer :: s1,s2
2097
2098      s1 = transfer_power_var
2099      if (present(var1)) s1 =
2100      s2 = transfer_power_var
2101      if (present(var2)) s2 =
2102
2103      itf = CP%Transfer%PK_reds
2104
2105      if (npoints < 2) stop 'Ne
2106
2107      !           if (minkh < MTr
2108      !           stop 'Transf
2109      !           end if
2110      if (minkh*exp((npoints-1)
2111          .and. FeedbackLevel >
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2105

```
2105      !sepctrum is generated to
2106      Type(MatterTransferData),
2107      Type(MatterPowerData) ::
2108
2109      integer, intent(in) :: it
2110      real(dl), intent(out) ::
2111      real(dl), intent(in) :: m
2112      integer, intent(in), opti
2113
2114      real(dl), parameter :: cl
2115      integer ik, llo,il,lhi,la
2116      real(dl) matpower(MTrans%
2117      real(dl) atransfer,xi, a0
2118      integer itf
2119      integer :: s1,s2
2120
2121      s1 = transfer_power_var
2122      if (present(var1)) s1 =
2123      s2 = transfer_power_var
2124      if (present(var2)) s2 =
2125
2126      itf = CP%Transfer%PK_reds
2127
2128      if (npoints < 2) call Mpi
2129
2130      !           if (minkh < MTr
2131      !           stop 'Transf
2132      !           end if
2133      if (minkh*exp((npoints-1)
2134          .and. FeedbackLevel >
```

/Users/lplopa/Compare/camb_simdata/modules.f90, Top line: 2112

```
2112      write(*,*) 'Warning:
2113
2114
2115      if (CP%NonLinear/=NonLine
2116      call Transfer_GetMatt
2117      call NonLinear_GetRat
2118  end if
2119
2120      h = CP%H0/100
2121      logmink = log(minkh)
2122      do ik=1,MTrans%num_q_tran
2123          kh = MTrans%TransferD
2124          k = kh*h
2125          kvals(ik) = log(kh)
2126          atransfer=MTrans%Tran
2127          if (CP%NonLinear/=Non
2128              atransfer = atran
2129          matpower(ik) = log(at
2130              !Put in power spectru
2131  end do
2132
2133      call spline(kvals,matpowe
2134
2135      llo=1
2136      lastix = npoints + 1
2137      do il=1, npoints
2138          xi=logmink + dlnkh*(i
2139          if (xi < kvals(1)) th
2140              outpower(il)=-30.
2141          cycle
```

/Users/lplopa/Compare/camb_des/modules.f90, Top line: 2135

```
2135      write(*,*) 'Warning:
2136
2137
2138      if (CP%NonLinear/=NonLine
2139      call Transfer_GetMatt
2140      call NonLinear_GetRat
2141  end if
2142
2143      h = CP%H0/100
2144      logmink = log(minkh)
2145      do ik=1,MTrans%num_q_tran
2146          kh = MTrans%TransferD
2147          k = kh*h
2148          kvals(ik) = log(kh)
2149          atransfer=MTrans%Tran
2150          if (CP%NonLinear/=Non
2151              atransfer = atran
2152          matpower(ik) = log(at
2153              !Put in power spectru
2154  end do
2155
2156      call spline(kvals,matpowe
2157
2158      llo=1
2159      lastix = npoints + 1
2160      do il=1, npoints
2161          xi=logmink + dlnkh*(i
2162          if (xi < kvals(1)) th
2163              outpower(il)=-30.
2164          cycle
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2142

```
2142      end if
2143      do while ((xi > kvals
2144              llo=llo+1
2145              if (llo >= MTrans
2146      end do
2147      if (llo == MTrans%num
2148          lastix = il
2149          exit
2150      end if
2151      lhi=llo+1
2152      ho=kvals(lhi)-kvals(l
2153      a0=(kvals(lhi)-xi)/ho
2154      b0=(xi-kvals(llo))/ho
2155
2156      outpower(il) = a0*mat
2157          +(b0**3-b0)*ddmat
2158      end do
2159
2160      do while (lastix <= npoin
2161          !Do linear extrapolat
2162          !Obviously inaccurate,
2163          outpower(lastix) = 2*
2164          lastix = lastix+1
2165      end do
2166
2167      outpower = exp(max(-30.d0
2168
2169      do il = 1, npoints
2170          k = exp(logmink + dln
2171          outpower(il) = outpow
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2165

```
2165      end if
2166      do while ((xi > kvals
2167              llo=llo+1
2168              if (llo >= MTrans
2169      end do
2170      if (llo == MTrans%num
2171          lastix = il
2172          exit
2173      end if
2174      lhi=llo+1
2175      ho=kvals(lhi)-kvals(l
2176      a0=(kvals(lhi)-xi)/ho
2177      b0=(xi-kvals(llo))/ho
2178
2179      outpower(il) = a0*mat
2180          +(b0**3-b0)*ddmat
2181      end do
2182
2183      do while (lastix <= npoin
2184          !Do linear extrapolat
2185          !Obviously inaccurate,
2186          outpower(lastix) = 2*
2187          lastix = lastix+1
2188      end do
2189
2190      outpower = exp(max(-30.d0
2191
2192      do il = 1, npoints
2193          k = exp(logmink + dln
2194          outpower(il) = outpow
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2172

```
2172         if (global_error_flag
2173     end do
2174
2175     if (CP%NonLinear/=NonLine
2176
2177     end subroutine Transfer_G
2178
2179     subroutine Transfer_Get_S
2180     !Calculate MTrans%sigma_8
2181     !of radius R h^{-1} Mpc,
2182     !set val, var2 e.g. to ge
2183     Type(MatterTransferData)
2184     real(dl), intent(in) :: R
2185     integer, intent(in), opti
2186     integer, intent(in), opti
2187     logical, intent(in), opti
2188     real(dl), intent(out) ::
2189     integer ik
2190     real(dl) kh, k, h, x, win
2191     real(dl) lnk, dlnk, lnko
2192     real(dl), dimension(CP%Tr
2193     real(dl) powers
2194     integer s1,s2
2195     integer :: ix = 1
2196
2197     s1 = transfer_power_var
2198     if (present(var1)) s1 =
2199     s2 = transfer_power_var
2200     if (present(var2)) s2 =
2201     if (present(power_ix)) ix
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2195

```
2195         if (global_error_flag
2196     end do
2197
2198     if (CP%NonLinear/=NonLine
2199
2200     end subroutine Transfer_G
2201
2202     subroutine Transfer_Get_S
2203     !Calculate MTrans%sigma_8
2204     !of radius R h^{-1} Mpc,
2205     !set val, var2 e.g. to ge
2206     Type(MatterTransferData)
2207     real(dl), intent(in) :: R
2208     integer, intent(in), opti
2209     integer, intent(in), opti
2210     logical, intent(in), opti
2211     real(dl), intent(out) ::
2212     integer ik
2213     real(dl) kh, k, h, x, win
2214     real(dl) lnk, dlnk, lnko
2215     real(dl), dimension(CP%Tr
2216     real(dl) powers
2217     integer s1,s2
2218     integer :: ix = 1
2219
2220     s1 = transfer_power_var
2221     if (present(var1)) s1 =
2222     s2 = transfer_power_var
2223     if (present(var2)) s2 =
2224     if (present(power_ix)) ix
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2202

```
2202
2203      H=CP%h0/100._dl
2204      lnko=0
2205      dsig8o=0
2206      sig8=0
2207      sig8o=0
2208      do ik=1, MTrans%num_q_tra
2209          kh = MTrans%TransferD
2210          if (kh==0) cycle
2211          k = kh*H
2212
2213          dsig8 = MTrans%Transf
2214              CP%Transfer%PK_re
2215          if (s1==s2) then
2216              dsig8 = dsig8**2
2217          else
2218              dsig8 = dsig8*MTr
2219                  CP%Transfer%P
2220          end if
2221          x= kh *R
2222          win =3*(sin(x)-x*cos(
2223          lnk=log(k)
2224          if (ik==1) then
2225              dlnk=0.5_dl
2226              !Approx for 2._dl
2227              !Contribution sho
2228          else
2229              dlnk=lnk-lnko
2230          end if
2231          powers = ScalarPower(
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2225

```
2225
2226      H=CP%h0/100._dl
2227      lnko=0
2228      dsig8o=0
2229      sig8=0
2230      sig8o=0
2231      do ik=1, MTrans%num_q_tra
2232          kh = MTrans%TransferD
2233          if (kh==0) cycle
2234          k = kh*H
2235
2236          dsig8 = MTrans%Transf
2237              CP%Transfer%PK_re
2238          if (s1==s2) then
2239              dsig8 = dsig8**2
2240          else
2241              dsig8 = dsig8*MTr
2242                  CP%Transfer%P
2243          end if
2244          x= kh *R
2245          win =3*(sin(x)-x*cos(
2246          lnk=log(k)
2247          if (ik==1) then
2248              dlnk=0.5_dl
2249              !Approx for 2._dl
2250              !Contribution sho
2251          else
2252              dlnk=lnk-lnko
2253          end if
2254          powers = ScalarPower(
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2232

```
2232      dsig8=(win*k**2)**2*p
2233      sig8=sig8+(dsig8+dsig
2234      dsig8o=dsig8
2235      lnko=lnk
2236    end do
2237
2238    if (present(root)) then
2239      if (root) sig8 =sqrt(
2240    else
2241      sig8 =sqrt(sig8)
2242    end if
2243    outvals(1:CP%Transfer%PK_
2244
2245  end subroutine Transfer_G
2246
2247  subroutine Transfer_GetSi
2248    !Get array of SigmaR at (
2249    Type(MatterTransferData)
2250    real(dl), intent(in) :: R
2251    real(dl), intent(out) ::
2252    integer, intent(in), opti
2253    integer i, red_ix, ik, su
2254    real(dl) kh, k, h, dkh
2255    real(dl) lnk, dlnk, lnko,
2256    real(dl), dimension(size(
2257    type(MatterPowerData) ::
2258    integer, parameter :: nsu
2259
2260    minR = minval(R)
2261    red_ix = CP%Transfer%PK_
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2255

```
2255      dsig8=(win*k**2)**2*p
2256      sig8=sig8+(dsig8+dsig
2257      dsig8o=dsig8
2258      lnko=lnk
2259    end do
2260
2261    if (present(root)) then
2262      if (root) sig8 =sqrt(
2263    else
2264      sig8 =sqrt(sig8)
2265    end if
2266    outvals(1:CP%Transfer%PK_
2267
2268  end subroutine Transfer_G
2269
2270  subroutine Transfer_GetSi
2271    !Get array of SigmaR at (
2272    Type(MatterTransferData)
2273    real(dl), intent(in) :: R
2274    real(dl), intent(out) ::
2275    integer, intent(in), opti
2276    integer red_ix, ik, subk
2277    real(dl) kh, k, h, dkh
2278    real(dl) lnk, dlnk, lnko,
2279    real(dl), dimension(size(
2280    type(MatterPowerData) ::
2281    integer, parameter :: nsu
2282
2283    minR = minval(R)
2284    red_ix = CP%Transfer%PK_
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2262

```
2262      if (present(redshift_ix))
2263
2264      call Transfer_GetMatterPo
2265
2266      H=CP%h0/100._dl
2267      lnko=0
2268      dsig8o=0
2269      sig8=0
2270      sig8o=0
2271      if (MTrans%TransferData(T
2272      do ik=1, MTrans%num_q_tra
2273          if (ik < MTrans%num_q
2274              dkh = (MTrans%Tra
2275              !after last step
2276          end if
2277          if (ik <= MTrans%num_
2278              do subk = 1, nsub
2279                  k = kh*H
2280                  lnk=log(k)
2281
2282                  x= kh *R
2283                  win =3*(sin(x)-x*
2284                  if (ik==1 .and. s
2285                      dlnk=0.5_dl
2286                      !Approx for 2
2287                      !Contribution
2288                  else
2289                      dlnk=lnk-lnko
2290                  end if
2291                  dsig8=win**2*(Mat
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2285

```
2285      if (present(redshift_ix))
2286
2287      call Transfer_GetMatterPo
2288
2289      H=CP%h0/100._dl
2290      lnko=0
2291      dsig8o=0
2292      sig8=0
2293      sig8o=0
2294      if (MTrans%TransferData(T
2295      do ik=1, MTrans%num_q_tra
2296          if (ik < MTrans%num_q
2297              dkh = (MTrans%Tra
2298              !after last step
2299          end if
2300          if (ik <= MTrans%num_
2301              do subk = 1, nsub
2302                  k = kh*H
2303                  lnk=log(k)
2304
2305                  x= kh *R
2306                  win =3*(sin(x)-x*
2307                  if (ik==1 .and. s
2308                      dlnk=0.5_dl
2309                      !Approx for 2
2310                      !Contribution
2311                  else
2312                      dlnk=lnk-lnko
2313                  end if
2314                  dsig8=win**2*(Mat
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2292

```
2292          sig8=sig8+(dsig8+
2293          dsig8o=dsig8
2294          lnko=lnk
2295          kh = kh + dkh
2296          end do
2297      end do
2298      call MatterPowerdata_Free
2299
2300      SigmaR=sqrt(sig8/(pi*twop
2301
2302      end subroutine Transfer_G
2303
2304      subroutine Transfer_Get_s
2305      !Calculate MTrans%sigma_8
2306      !of radius R h^{-1} Mpc
2307      ! set val, var2 e.g. to g
2308      Type(MatterTransferData)
2309      real(dl), intent(in), opt
2310      integer, intent(in), opti
2311      integer ix
2312      real(dl) :: radius = 8._d
2313
2314      if (global_error_flag /=
2315
2316      if (present(R)) radius =
2317
2318      do ix = 1, CP%InitPower%n
2319          call Transfer_Get_Sig
2320      end do
2321
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2315

```
2315          sig8=sig8+(dsig8+
2316          dsig8o=dsig8
2317          lnko=lnk
2318          kh = kh + dkh
2319          end do
2320      end do
2321      call MatterPowerdata_Free
2322
2323      SigmaR=sqrt(sig8/(pi*twop
2324
2325      end subroutine Transfer_G
2326
2327      subroutine Transfer_Get_s
2328      !Calculate MTrans%sigma_8
2329      !of radius R h^{-1} Mpc
2330      ! set val, var2 e.g. to g
2331      Type(MatterTransferData)
2332      real(dl), intent(in), opt
2333      integer, intent(in), opti
2334      integer ix
2335      real(dl) :: radius = 8._d
2336
2337      if (global_error_flag /=
2338
2339      if (present(R)) radius =
2340
2341      do ix = 1, CP%InitPower%n
2342          call Transfer_Get_Sig
2343      end do
2344
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2322

```
2322      end subroutine Transfer_G
2323
2324      subroutine Transfer_Get_s
2325      !Get sigma8 and sigma_{de
2326      Type(MatterTransferData)
2327      real(dl), intent(in), opt
2328      integer, intent(in), opti
2329      real(dl) :: radius = 8._d
2330      integer s1, s2, ix
2331
2332      if (global_error_flag /=
2333
2334      if (present(R)) radius =
2335      s1 = transfer_power_var
2336      if (present(var_delta))
2337      s2 = Transfer_Newt_vel_cd
2338      if (present(var_v)) s2 =
2339
2340      do ix = 1, CP%InitPower%n
2341          call Transfer_Get_Sig
2342          if (get_growth_sigma8
2343              MTrans%sigma2_vde
2344      end do
2345
2346      end subroutine Transfer_G
2347
2348      subroutine Transfer_output
2349      Type(MatterTransferData),
2350      integer in, j
2351      !JD 08/13 Changes in here
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2345

```
2345      end subroutine Transfer_G
2346
2347      subroutine Transfer_Get_s
2348      !Get sigma8 and sigma_{de
2349      Type(MatterTransferData)
2350      real(dl), intent(in), opt
2351      integer, intent(in), opti
2352      real(dl) :: radius = 8._d
2353      integer s1, s2, ix
2354
2355      if (global_error_flag /=
2356
2357      if (present(R)) radius =
2358      s1 = transfer_power_var
2359      if (present(var_delta))
2360      s2 = Transfer_Newt_vel_cd
2361      if (present(var_v)) s2 =
2362
2363      do ix = 1, CP%InitPower%n
2364          call Transfer_Get_Sig
2365          if (get_growth_sigma8
2366              MTrans%sigma2_vde
2367      end do
2368
2369      end subroutine Transfer_G
2370
2371      subroutine Transfer_output
2372      Type(MatterTransferData),
2373      integer in, j
2374      !JD 08/13 Changes in here
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2352

```
2352      integer j_PK
2353
2354      do in=1, CP%InitPower%nn
2355          if (CP%InitPower%nn>1
2356              do j_PK=1, CP%Transfe
2357                  j = CP%Transfer%P
2358                  write(*, '("at z =
2359                      CP%Transfer%r
2360              end do
2361              if (get_growth_sigma8
2362                  do j_PK=1, CP%Tra
2363                      j = CP%Transf
2364                      write(*, '("at
2365                          CP%Transf
2366                  end do
2367              end if
2368      end do
2369
2370      end subroutine Transfer_o
2371
2372      subroutine Transfer_Alloc
2373      Type(MatterTransferData)
2374      integer st
2375
2376      deallocate(MTrans%q_trans
2377      deallocate(MTrans%Transfe
2378      deallocate(MTrans%sigma_8
2379      if (get_growth_sigma8) de
2380      allocate(MTrans%q_trans(M
2381      allocate(MTrans%TransferD
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2375

```
2375      integer j_PK
2376
2377      do in=1, CP%InitPower%nn
2378          if (CP%InitPower%nn>1
2379              do j_PK=1, CP%Transfe
2380                  j = CP%Transfer%P
2381                  write(*, '("at z =
2382                      CP%Transfer%r
2383              end do
2384              if (get_growth_sigma8
2385                  do j_PK=1, CP%Tra
2386                      j = CP%Transf
2387                      write(*, '("at
2388                          CP%Transf
2389                  end do
2390              end if
2391      end do
2392
2393      end subroutine Transfer_o
2394
2395      subroutine Transfer_Alloc
2396      Type(MatterTransferData)
2397      integer st
2398
2399      deallocate(MTrans%q_trans
2400      deallocate(MTrans%Transfe
2401      deallocate(MTrans%sigma_8
2402      if (get_growth_sigma8) de
2403      allocate(MTrans%q_trans(M
2404      allocate(MTrans%TransferD
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2382

```
2382      !JD 08/13 Changes in here
2383      allocate(MTrans%sigma_8(C
2384      if (get_growth_sigma8) al
2385
2386      end  subroutine Transfer_
2387
```

```
2388      subroutine Transfer_Free(
2389      Type(MatterTransferData):
2390      integer st
2391
2392      deallocate(MTrans%q_trans
2393      deallocate(MTrans%Transfe
2394      deallocate(MTrans%sigma_8
2395      if (get_growth_sigma8) de
2396      nullify(MTrans%q_trans)
2397      nullify(MTrans%TransferDa
2398      nullify(MTrans%sigma_8)
2399      nullify(MTrans%sigma2_vde
2400
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2405

```
2405      !JD 08/13 Changes in here
2406      allocate(MTrans%sigma_8(C
2407      if (get_growth_sigma8) al
2408
2409      end  subroutine Transfer_
2410
```

```
2411      subroutine Transfer_Nulli
2412      Type(MatterTransferData):
2413
```

```
2414      Mtrans%num_q_trans = 0
2415      nullify(MTrans%q_trans)
2416      nullify(MTrans%TransferDa
2417      nullify(MTrans%sigma_8)
2418      nullify(MTrans%sigma2_vde
2419
```

```
2420      end subroutine Transfer_N
2421
```

```
2422      subroutine Transfer_Free(
2423      Type(MatterTransferData):
2424      integer st
2425
```

```
2426      deallocate(MTrans%q_trans
2427      deallocate(MTrans%Transfe
2428      deallocate(MTrans%sigma_8
2429      if (get_growth_sigma8) de
2430      call Transfer_Nullify(MTr
```

```
2431
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2401

```
2401      end subroutine Transfer_F
2402
2403      !JD 08/13 Changes for non
2404      !Changed function below t
2405      subroutine Transfer_SetFo
2406      Type(TransferParams) :: P
2407      integer i
2408      real maxRedshift
2409
2410      P%kmax = max(P%kmax,5*Acc
2411      P%k_per_logint = 0
2412      maxRedshift = 10
2413      P%NLL_num_redshifts = ni
2414      if (HighAccuracyDefault .
2415          !only notionally more
2416          maxRedshift =15
2417      end if
2418      if (P%NLL_num_redshifts >
2419          stop 'Transfer_SetFor
2420      do i=1,P%NLL_num_redshift
2421          P%NLL_redshifts(i) =
2422      end do
2423
2424      end subroutine Transfer_S
2425
2426
2427
2428      subroutine Transfer_SaveT
2429      use IniFile
2430      Type(MatterTransferData),
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2432

```
2432      end subroutine Transfer_F
2433
2434      !JD 08/13 Changes for non
2435      !Changed function below t
2436      subroutine Transfer_SetFo
2437      Type(TransferParams) :: P
2438      integer i
2439      real maxRedshift
2440
2441      P%kmax = max(P%kmax,5*Acc
2442      P%k_per_logint = 0
2443      maxRedshift = 10
2444      P%NLL_num_redshifts = ni
2445      if (HighAccuracyDefault .
2446          !only notionally more
2447          maxRedshift =15
2448      end if
2449      if (P%NLL_num_redshifts >
2450          call MpiStop('Transfe
2451      do i=1,P%NLL_num_redshift
2452          P%NLL_redshifts(i) =
2453      end do
2454
2455      end subroutine Transfer_S
2456
2457
2458
2459      subroutine Transfer_SaveT
2460      use IniFile
2461      Type(MatterTransferData),
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2431

```
2431      integer i,ik
2432      character(LEN=Ini_max_str
2433      !JD 08/13 Changes in here
2434      integer i_PK
2435      character(len=20) fmt
2436
2437
2438      write (fmt,*) Transfer_ma
2439      fmt = '('//trim(adjustl(f
2440
2441      do i_PK=1, CP%Transfer%PK
2442          if (FileNames(i_PK) /
2443              i = CP%Transfer%P
2444              open(unit=fileio_
2445                  !cafea
2446                  !print*, '20:', fi
2447                  do ik=1,MTrans%nu
2448                      if (MTrans%Tr
2449                          write(fil
2450                      end if
2451                  end do
2452                  close(fileio_unit
2453              end if
2454      end do
2455
2456      end subroutine Transfer_S
2457
2458      subroutine Transfer_SaveM
2459      use IniFile
2460      !Export files of total m
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2462

```
2462      integer i,ik
2463      character(LEN=Ini_max_str
2464      !JD 08/13 Changes in here
2465      integer i_PK
2466      integer unit
2467
2468
2469      do i_PK=1, CP%Transfer%PK
2470          if (FileNames(i_PK) /
2471              i = CP%Transfer%P
2472              unit = open_file_
2473
2474          do ik=1,MTrans%nu
2475              if (MTrans%Tr
2476                  write(uni
2477              end if
2478          end do
2479          close(unit)
2480      end if
2481      end do
2482
2483      end subroutine Transfer_S
2484
2485      subroutine Transfer_SaveM
2486      use IniFile
2487      !Export files of total m
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2461

```
2461      Type(MatterTransferData),
2462      character(LEN=Ini_max_str
2463      integer itf,in,i
2464      integer points
2465      real, dimension(:,:,:), a
2466      character(LEN=80) fmt
2467      real minkh,dlnkh
2468      Type(MatterPowerData) ::
2469      integer ncol
2470      !JD 08/13 Changes in here
2471      integer itf_PK
2472
2473      ncol=1
2474
2475      write (fmt,*) CP%InitPowe
2476      fmt = '('//trim(adjustl(f
2477      do itf=1, CP%Transfer%PK_
2478          if (FileNames(itf) /=
2479              if (.not. transfe
2480                  itf_PK = CP%T
2481
2482          points = MTra
2483          allocate(outp
2484
2485          do in = 1, CP
2486              call Tran
2487              !JD 08/13
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2486

```
2486      Type(MatterTransferData),
2487      character(LEN=Ini_max_str
2488      integer itf,in,i
2489      integer points
2490      real, dimension(:,:,:), a
2491
2492      real minkh,dlnkh
2493      Type(MatterPowerData) ::
2494      integer ncol
2495      !JD 08/13 Changes in here
2496      integer itf_PK
2497      integer unit
2498      character(name_tag_len) :
2499
2500      ncol=1
2501      if (CP%InitPower%nn>1 .an
2502
2503      do itf=1, CP%Transfer%PK_
2504          if (FileNames(itf) /=
2505              if (.not. transfe
2506                  itf_PK = CP%T
2507
2508          points = MTra
2509          allocate(outp
2510
2511          do in = 1, CP
2512              call Tran
2513              !JD 08/13
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2488

```
2488      !Changed
2489      if(CP%Non
2490          call
2491
2492      outpower(
2493      call Matt
2494      end do
2495
2496      open(unit=fil
2497      !cafea
2498      ! print*, '28:
2499      do i=1,points
2500          write (fi
2501      end do
2502      close(fileio_
2503      else
2504          minkh = 1e-4
2505          dlnkh = 0.02
2506          points = log(
2507          !
2508          allocate(outp
2509          do in = 1, CP
2510              call Tran
2511          end do
2512
2513      open(unit=fil
2514
2515      !cafea
2516      !print*, '34:
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2513

```
2513      !Changed
2514      if(CP%Non
2515          call
2516
2517      outpower(
2518      call Matt
2519      end do
2520      columns = ['P
2521      unit = open_f
2522
2523      do i=1,points
2524          write (un
2525      end do
2526      close(unit)
2527      else
2528          minkh = 1e-4
2529          dlnkh = 0.02
2530          points = log(
2531          !
2532          allocate(outp
2533          do in = 1, CP
2534              call Tran
2535          end do
2536      columns(1) =
2537      unit = open_f
2538
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2517

```
2517      do i=1,point
2518          write (fi
2519      end do
2520      close(fileio_
2521      end if
2522
2523          deallocate(outpow
2524      end if
2525  end do
2526
2527  end subroutine Transfer_S
2528
2529  !JD 08/13 New function fo
2530  !Build master redshift ar
2531  !redshifts and an array o
2532  !At the same time fill ar
2533  !of their desired redshif
2534  !Finally define number of
2535  !P%num_redshifts = P%PK_n
2536  !from the fact that z=0 i
2537  subroutine Transfer_SortA
2538  Type(TransferParams) :: P
2539  integer i, iPK, iNLL
2540  real(dl), parameter :: to
2541
2542      i=0
2543      iPK=1
2544      iNLL=1
2545      do while (iPk<=P%PK_num_r
2546          !JD write the next li
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2539

```
2539      do i=1,points
2540          write (un
2541      end do
2542      close(unit)
2543      end if
2544
2545          deallocate(outpow
2546      end if
2547  end do
2548
2549  end subroutine Transfer_S
2550
2551  !JD 08/13 New function fo
2552  !Build master redshift ar
2553  !redshifts and an array o
2554  !At the same time fill ar
2555  !of their desired redshif
2556  !Finally define number of
2557  !P%num_redshifts = P%PK_n
2558  !from the fact that z=0 i
2559  subroutine Transfer_SortA
2560  Type(TransferParams) :: P
2561  integer i, iPK, iNLL
2562  real(dl), parameter :: to
2563
2564      i=0
2565      iPK=1
2566      iNLL=1
2567      do while (iPk<=P%PK_num_r
2568          !JD write the next li
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2547

```
2547      i=i+1
2548      if (i > max_transfer_
2549          call Mpistop('Tra
2550
2551      if(iNLL>P%NLL_num_red
2552          P%redshifts(i)=P%
2553          P%PK_redshifts_in
2554          iPK=iPK+1
2555      else if(iPK>P%PK_num
2556          P%redshifts(i)=P%
2557          P%NLL_redshifts_i
2558          iNLL=iNLL+1
2559      else
2560          P%redshifts(i)=P%
2561          P%PK_redshifts_in
2562          P%NLL_redshifts_i
2563          iPK=iPK+1
2564          iNLL=iNLL+1
2565      end if
2566  end do
2567  P%num_redshifts=i
2568
2569  end subroutine Transfer_S
2570
2571  end module Transfer
2572
2573
2574      !cccccccccccccccccccccccccccccccccccc
2575
2576  module ThermoData
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2569

```
2569      i=i+1
2570      if (i > max_transfer_
2571          call Mpistop('Tra
2572
2573      if(iNLL>P%NLL_num_red
2574          P%redshifts(i)=P%
2575          P%PK_redshifts_in
2576          iPK=iPK+1
2577      else if(iPK>P%PK_num
2578          P%redshifts(i)=P%
2579          P%NLL_redshifts_i
2580          iNLL=iNLL+1
2581      else
2582          P%redshifts(i)=P%
2583          P%PK_redshifts_in
2584          P%NLL_redshifts_i
2585          iPK=iPK+1
2586          iNLL=iNLL+1
2587      end if
2588  end do
2589  P%num_redshifts=i
2590
2591  end subroutine Transfer_S
2592
2593  end module Transfer
2594
2595
2596      !cccccccccccccccccccccccccccccccccccc
2597
2598  module ThermoData
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2577

```
2577      use ModelData
2578      implicit none
2579      private
2580      integer,parameter :: nthelens
2581
2582      real(dl) tb(nthermo),cs2(
2583      real(dl) dcs2(nthermo)
2584      real(dl) dotmu(nthermo),
2585      real(dl) sdotmu(nthermo),
2586      real(dl) demmu(nthermo)
2587      real(dl) dddotmu(nthermo)
2588      real(dl) winlens(nthermo)
2589      real(dl) tauminn,dlntau,M
2590      real(dl), dimension(:), a
2591      logical, parameter :: dow
2592
2593      real(dl) :: tight_tau, ac
2594      !Times when 1/(opacity*ta
2595      real(dl) :: matter_verydo
2596      real(dl) :: r_drag0, z_st
2597
2598      public thermo,inithermo,v
2599      Thermo_OpacityToTime,
2600      z_star, z_drag !!JH
2601      contains
2602
2603      subroutine thermo(tau,cs2
2604      !Compute unperturbed soun
2605      !and ionization fraction
2606      !If requested also get ti
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2599

```
2599      use ModelData
2600      implicit none
2601      private
2602      integer,parameter :: nthelens
2603
2604      real(dl) tb(nthermo),cs2(
2605      real(dl) dcs2(nthermo)
2606      real(dl) dotmu(nthermo),
2607      real(dl) sdotmu(nthermo),
2608      real(dl) demmu(nthermo)
2609      real(dl) dddotmu(nthermo)
2610      real(dl) winlens(nthermo)
2611      real(dl) tauminn,dlntau,M
2612
2613
2614      real(dl) :: tight_tau, ac
2615      !Times when 1/(opacity*ta
2616      real(dl) :: matter_verydo
2617      real(dl) :: r_drag0, z_st
2618
2619      public thermo,inithermo,
2620      Thermo_OpacityToTime,
2621      z_star, z_drag, GetBa
2622      contains
2623
2624      subroutine thermo(tau,cs2
2625      !Compute unperturbed soun
2626      !and ionization fraction
2627      !If requested also get ti
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2607

```
2607      implicit none
2608      real(dl) tau,cs2b,opacity
2609      real(dl), intent(out), op
2610
2611      integer i
2612      real(dl) d
2613
2614      d=log(tau/tauminn)/dlntau
2615      i=int(d)
2616      d=d-i
2617      if (i < 1) then
2618          !Linear interpolation
2619          cs2b=cs2(1)+(d+i-1)*d
2620          opacity=dotmu(1)+(d-1
2621      stop 'thermo out of b
2622      else if (i >= nthermo) th
2623          cs2b=cs2(nthermo)+(d+
2624          opacity=dotmu(nthermo
2625          if (present(dopacity)
2626              dopacity = 0
2627      stop 'thermo: sho
2628      end if
2629      else
2630          !Cubic spline interpo
2631          cs2b=cs2(i)+d*(dcs2(i
2632              -2*dcs2(i)-dcs2(i
2633              +2*(cs2(i)-cs2(i+
2634          opacity=dotmu(i)+d*(d
2635              -2*ddotmu(i)-ddot
2636              +2*(dotmu(i)-dotm
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2628

```
2628      implicit none
2629      real(dl) tau,cs2b,opacity
2630      real(dl), intent(out), op
2631
2632      integer i
2633      real(dl) d
2634
2635      d=log(tau/tauminn)/dlntau
2636      i=int(d)
2637      d=d-i
2638      if (i < 1) then
2639          !Linear interpolation
2640          cs2b=cs2(1)+(d+i-1)*d
2641          opacity=dotmu(1)+(d-1
2642      call MpiStop('thermo
2643      else if (i >= nthermo) th
2644          cs2b=cs2(nthermo)+(d+
2645          opacity=dotmu(nthermo
2646          if (present(dopacity)
2647              dopacity = 0
2648      call MpiStop('the
2649      end if
2650      else
2651          !Cubic spline interpo
2652          cs2b=cs2(i)+d*(dcs2(i
2653              -2*dcs2(i)-dcs2(i
2654              +2*(cs2(i)-cs2(i+
2655          opacity=dotmu(i)+d*(d
2656              -2*ddotmu(i)-ddot
2657              +2*(dotmu(i)-dotm
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2637

```
2637
2638         if (present(dopacity)
2639             dopacity=(ddotmu(
2640                 -ddotmu(i))-2
2641                 +dddotmu(i+1)
2642             end if
2643         end if
2644     end subroutine thermo
2645
2646
2647     function Thermo_OpacityTo
2648     real(dl), intent(in) :: o
2649     integer j
2650     real(dl) Thermo_OpacityTo
2651     !Do this the bad slow way
2652     !The answer is approximat
2653     j =1
2654     do while(dotmu(j)> opacit
2655         j=j+1
2656     end do
2657
2658     Thermo_OpacityToTime = ex
2659
2660     end function Thermo_Opaci
2661
2662     subroutine inithermo(taum
2663     ! Compute and save unper
2664     ! as a function of time.
2665     ! accuracy (numerical int
2666     use constants
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2658

```
2658
2659         if (present(dopacity)
2660             dopacity=(ddotmu(
2661                 -ddotmu(i))-2
2662                 +dddotmu(i+1)
2663             end if
2664         end if
2665     end subroutine thermo
2666
2667
2668     function Thermo_OpacityTo
2669     real(dl), intent(in) :: o
2670     integer j
2671     real(dl) Thermo_OpacityTo
2672     !Do this the bad slow way
2673     !The answer is approximat
2674     j =1
2675     do while(dotmu(j)> opacit
2676         j=j+1
2677     end do
2678
2679     Thermo_OpacityToTime = ex
2680
2681     end function Thermo_Opaci
2682
2683     subroutine inithermo(taum
2684     ! Compute and save unper
2685     ! as a function of time.
2686     ! accuracy (numerical int
2687     use constants
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2667

```
2667      use precision
2668      use ModelParams
2669      use MassiveNu
2670      real(dl) taumin,taumax
2671
2672
2673      real(dl) tau01,adot0,a0,a
2674      real(dl) xe0,tau,a,a2
2675      real(dl) adot,tg0,ahalf,a
2676      real(dl) dtbdla,vfi,cfl,m
2677      integer ncount,i,j1,j2,iv
2678      real(dl) spline_data(nthe
2679      real(dl) last_dotmu
2680      real(dl) dtauda !diff of
2681      external dtauda
2682      real(dl) a_verydom
2683      real(dl) awin_lens1p,awin
2684      real(dl) z_eq, a_eq
2685      real(dl) rombint
2686      integer noutput
2687      external rombint
2688
2689      call Recombination_Init(C
2690          CP%h0,CP%tcmb,CP%yhe,
2691          !almost all the time spen
2692          if (global_error_flag/=0)
2693          Maxtau=taumax
2694          tight_tau = 0
2695          actual_opt_depth = 0
2696          ncount=0
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2687

```
2687      use precision
2688      use ModelParams
2689      use MassiveNu
2690      real(dl) taumin,taumax
2691
2692
2693      real(dl) tau01,adot0,a0,a
2694      real(dl) xe0,tau,a,a2
2695      real(dl) adot,tg0,ahalf,a
2696      real(dl) dtbdla,vfi,cfl,m
2697      integer ncount,i,j1,iv,ns
2698      real(dl) spline_data(nthe
2699      real(dl) last_dotmu
2700      real(dl) dtauda !diff of
2701      external dtauda
2702      real(dl) a_verydom
2703      real(dl) awin_lens1p,awin
2704      real(dl) z_eq, a_eq
2705      real(dl) rombint
2706      integer noutput
2707      external rombint
2708
2709      call Recombination_Init(C
2710          CP%h0,CP%tcmb,CP%yhe,
2711          !almost all the time spen
2712          if (global_error_flag/=0)
2713          Maxtau=taumax
2714          tight_tau = 0
2715          actual_opt_depth = 0
2716          ncount=0
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2697

```
2697      z_star=0.d0
2698      z_drag=0.d0
2699      thomc0= Compton_CT * CP%t
2700      r_drag0 = 3.d0/4.d0*CP%om
2701      !thomc0=5.0577d-8*CP%tcmb
2702
2703      tauminn=0.05d0*taumin
2704      dlntau=log(CP%tau0/taumin
2705      last_dotmu = 0
2706
2707      matter_verydom_tau = 0
2708      a_verydom = AccuracyBoost
2709
2710      ! Initial conditions: as
2711      tau01=tauminn
2712      adot0=adotrad
2713      a0=adotrad*tauminn
2714      a02=a0*a0
2715      ! Assume that any entrop
2716      ! This gives wrong tempe
2717      ! the error is harmless.
2718      tb(1)=CP%tcmb/a0
2719      xe0=1._dl
2720      x1=0._dl
2721      x2=1._dl
2722      xe(1)=xe0+0.25d0*CP%yhe/(
2723      barssc=barssc0*(1._dl-0.7
2724      cs2(1)=4._dl/3._dl*barssc
2725      dotmu(1)=xe(1)*akthom/a02
2726      sdotmu(1)=0
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2717

```
2717      z_star=0.d0
2718      z_drag=0.d0
2719      thomc0= Compton_CT * CP%t
2720      r_drag0 = 3.d0/4.d0*CP%om
2721      !thomc0=5.0577d-8*CP%tcmb
2722
2723      tauminn=0.05d0*taumin
2724      dlntau=log(CP%tau0/taumin
2725      last_dotmu = 0
2726
2727      matter_verydom_tau = 0
2728      a_verydom = AccuracyBoost
2729
2730      ! Initial conditions: as
2731      tau01=tauminn
2732      adot0=adotrad
2733      a0=adotrad*tauminn
2734      a02=a0*a0
2735      ! Assume that any entrop
2736      ! This gives wrong tempe
2737      ! the error is harmless.
2738      tb(1)=CP%tcmb/a0
2739      xe0=1._dl
2740      x1=0._dl
2741      x2=1._dl
2742      xe(1)=xe0+0.25d0*CP%yhe/(
2743      barssc=barssc0*(1._dl-0.7
2744      cs2(1)=4._dl/3._dl*barssc
2745      dotmu(1)=xe(1)*akthom/a02
2746      sdotmu(1)=0
```

```
2727
2728       do i=2,nthermo
2729           tau=tauminn*exp((i-1)
2730           dtau=tau-tau01
2731           !   Integrate Friedman
2732
2733           a=a0+adot0*dtau
2734           scaleFactor(i)=a
2735           a2=a*a
2736
2737           adot=1/dtauda(a)
2738
2739           if (matter_verydom_ta
2740               matter_verydom_ta
2741           end if
2742
2743           a=a0+2._dl*dtau/(1._d
2744           !   Baryon temperature
2745           !   Use quadrature so
2746           !   This is redundant a
2747           tg0=CP%tcmb/a0
2748           ahalf=0.5d0*(a0+a)
2749           adothalf=0.5d0*(adot0
2750           !   fe=number of free
2751           !   particles (e+p+H+H
2752           !   more accuracy is r
2753           !   the solution of th
2754           fe=(1._dl-CP%yhe)*xe(
2755           thomc=thomc0*fe/adoth
2756           etc=exp(-thomc*(a-a0))
```

```
2747
2748       do i=2,nthermo
2749           tau=tauminn*exp((i-1)
2750           dtau=tau-tau01
2751           !   Integrate Friedman
2752
2753           a=a0+adot0*dtau
2754           scaleFactor(i)=a
2755           a2=a*a
2756
2757           adot=1/dtauda(a)
2758
2759           if (matter_verydom_ta
2760               matter_verydom_ta
2761           end if
2762
2763           a=a0+2._dl*dtau/(1._d
2764           !   Baryon temperature
2765           !   Use quadrature so
2766           !   This is redundant a
2767           tg0=CP%tcmb/a0
2768           ahalf=0.5d0*(a0+a)
2769           adothalf=0.5d0*(adot0
2770           !   fe=number of free
2771           !   particles (e+p+H+H
2772           !   more accuracy is r
2773           !   the solution of th
2774           fe=(1._dl-CP%yhe)*xe(
2775           thomc=thomc0*fe/adoth
2776           etc=exp(-thomc*(a-a0))
```

/Users/lplopa/Compare/camb_simdata/modules.f90, Top line: 2757

```
2757      a2t=a0*a0*(tb(i-1)-tg
2758      tb(i)=CP%tcmb/a+a2t/(
2759
2760      ! If there is re-ioni
2761      ! requested value.
2762      if (CP%Reion%Reioniza
2763          if(ncount == 0) t
2764              ncount=i-1
2765          end if
2766          xe(i) = Reionizat
2767          !print *,1/a-1,xe
2768          if (CP%AccurateRe
2769              dotmu(i)=(Rec
2770
2771              if (last_dotm
2772                  actual_op
2773              end if
2774              last_dotmu =
2775          end if
2776      else
2777          xe(i)=Recombinati
2778      end if
2779
2780      ! Baryon sound speed
2781      dtbdla=-2._dl*tb(i)-t
2782      barssc=barssc0*(1._dl
2783      cs2(i)=barssc*tb(i)* (
2784
2785
2786      ! Calculation of the
```

/Users/lplopa/Compare/camb_des/modules.f90, Top line: 2777

```
2777      a2t=a0*a0*(tb(i-1)-tg
2778      tb(i)=CP%tcmb/a+a2t/(
2779
2780      ! If there is re-ioni
2781      ! requested value.
2782      if (CP%Reion%Reioniza
2783          if(ncount == 0) t
2784              ncount=i-1
2785          end if
2786          xe(i) = Reionizat
2787          !print *,1/a-1,xe
2788          if (CP%AccurateRe
2789              dotmu(i)=(Rec
2790
2791              if (last_dotm
2792                  actual_op
2793              end if
2794              last_dotmu =
2795          end if
2796      else
2797          xe(i)=Recombinati
2798      end if
2799
2800      ! Baryon sound speed
2801      dtbdla=-2._dl*tb(i)-t
2802      barssc=barssc0*(1._dl
2803      cs2(i)=barssc*tb(i)* (
2804
2805
2806      ! Calculation of the
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2787

```
2787      dotmu(i)=xe(i)*akthom
2788
2789      if (tight_tau==0 .and
2790      !Tight coupling switc
2791
2792      if (tau < 0.001) then
2793          sdotmu(i)=0
2794      else
2795          sdotmu(i)=sdotmu(
2796      end if
2797
2798      a0=a
2799      tau01=tau
2800      adot0=adot
2801  end do !i
2802
2803      if (CP%Reion%Reionization
2804          write(*,*) 'Warning: x
2805          write(*,*) 'Check inp
2806          write(*,*) 'function
2807      end if
2808
2809      do j1=1,nthermo
2810          if (sdotmu(j1) - sdot
2811              emmu(j1)=1.d-30
2812          else
2813              emmu(j1)=exp(sdot
2814              if (.not. CP%Accu
2815                  actual_opt_de
2816          actual_opt_depth
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2807

```
2807      dotmu(i)=xe(i)*akthom
2808
2809      if (tight_tau==0 .and
2810      !Tight coupling switc
2811
2812      if (tau < 0.001) then
2813          sdotmu(i)=0
2814      else
2815          sdotmu(i)=sdotmu(
2816      end if
2817
2818      a0=a
2819      tau01=tau
2820      adot0=adot
2821  end do !i
2822
2823      if (CP%Reion%Reionization
2824          write(*,*) 'Warning: x
2825          write(*,*) 'Check inp
2826          write(*,*) 'function
2827      end if
2828
2829      do j1=1,nthermo
2830          if (sdotmu(j1) - sdot
2831              emmu(j1)=1.d-30
2832          else
2833              emmu(j1)=exp(sdot
2834              if (.not. CP%Accu
2835                  actual_opt_de
2836          actual_opt_de
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2817

```
2817         end if
2818         if (CP%AccurateRe
2819             if (sdotmu(nt
2820                 tau01=1-(
2821                 tau01=tau
2822                 z_star =
2823             end if
2824         end if
2825     end if
2826 end do
2827
2828     if (CP%AccurateReionizati
2829         write(*, '("Reion opt
2830     end if
2831
2832
2833     iv=0
2834     vfi=0._dl
2835     ! Getting the starting an
2836     if (ncount == 0) then
2837         cf1=1._dl
2838         ns=nthermo
2839     else
2840         cf1=exp(sdotmu(ntherm
2841         ns=ncount
2842     end if
2843     maxvis = 0
2844     do j1=1,ns
2845         vis = emmu(j1)*dotmu(
2846         tau = tauminn*exp((j1
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2837

```
2837         end if
2838         if (CP%AccurateRe
2839             if (sdotmu(nt
2840                 tau01=1-(
2841                 tau01=tau
2842                 z_star =
2843             end if
2844         end if
2845     end if
2846 end do
2847
2848     if (CP%AccurateReionizati
2849         write(*, '("Reion opt
2850     end if
2851
2852
2853     iv=0
2854     vfi=0._dl
2855     ! Getting the starting an
2856     if (ncount == 0) then
2857         cf1=1._dl
2858         ns=nthermo
2859     else
2860         cf1=exp(sdotmu(ntherm
2861         ns=ncount
2862     end if
2863     maxvis = 0
2864     do j1=1,ns
2865         vis = emmu(j1)*dotmu(
2866         tau = tauminn*exp((j1
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2847

```
2847      vfi=vfi+vis*cf1*dlnta
2848      if ((iv == 0).and.(vf
2849          taurst=9._dl/10._
2850          iv=1
2851      elseif (iv == 1) then
2852          if (vis > maxvis)
2853              maxvis=vis
2854              tau_maxvis =
2855          end if
2856          if (vfi > 0.995)
2857              taurend=tau
2858              iv=2
2859              exit
2860          end if
2861      end if
2862  end do
2863
2864      if (iv /= 2) then
2865          call GlobalError('ini
2866          return
2867      end if
2868
2869      if (dowinlens) then
2870          vfi=0
2871          awin_lens1p=0
2872          awin_lens2p=0
2873          winlens=0
2874          do j1=1,nthermo-1
2875              vis = emmu(j1)*do
2876              tau = tauminn*exp
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2867

```
2867      vfi=vfi+vis*cf1*dlnta
2868      if ((iv == 0).and.(vf
2869          taurst=9._dl/10._
2870          iv=1
2871      elseif (iv == 1) then
2872          if (vis > maxvis)
2873              maxvis=vis
2874              tau_maxvis =
2875          end if
2876          if (vfi > 0.995)
2877              taurend=tau
2878              iv=2
2879              exit
2880          end if
2881      end if
2882  end do
2883
2884      if (iv /= 2) then
2885          call GlobalError('ini
2886          return
2887      end if
2888
2889      if (dowinlens) then
2890          vfi=0
2891          awin_lens1p=0
2892          awin_lens2p=0
2893          winlens=0
2894          do j1=1,nthermo-1
2895              vis = emmu(j1)*do
2896              tau = tauminn*exp
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2877

```
2877          vfi=vfi+vis*cf1*d
2878          if (vfi < 0.995)
2879              dwing_lens =
2880
2881              awin_lens1p =
2882              awin_lens2p =
2883          end if
2884          winlens(j1)= awin
2885      end do
2886  end if
2887
2888      ! Calculating the timeste
2889
2890      if (CP%WantTensors) then
2891          dtaurec=min(dtaurec,t
2892      else
2893          dtaurec=min(dtaurec,t
2894          if (do_bispectrum .an
2895      end if
2896
2897      if (CP%Reion%Reionization
2898
2899      if (DebugMsgs) then
2900          write (*,*) 'taurst,
2901      end if
2902
2903      call splini(spline_data,n
2904      call splder(cs2,dcs2,nthe
2905      call splder(dotmu,ddotmu,
2906      call splder(ddotmu,dddotm
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2897

```
2897          vfi=vfi+vis*cf1*d
2898          if (vfi < 0.995)
2899              dwing_lens =
2900
2901              awin_lens1p =
2902              awin_lens2p =
2903          end if
2904          winlens(j1)= awin
2905      end do
2906  end if
2907
2908      ! Calculating the timeste
2909
2910      if (CP%WantTensors) then
2911          dtaurec=min(dtaurec,t
2912      else
2913          dtaurec=min(dtaurec,t
2914          if (do_bispectrum .an
2915      end if
2916
2917      if (CP%Reion%Reionization
2918
2919      if (DebugMsgs) then
2920          write (*,*) 'taurst,
2921      end if
2922
2923      call splini(spline_data,n
2924      call splder(cs2,dcs2,nthe
2925      call splder(dotmu,ddotmu,
2926      call splder(ddotmu,dddotm
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2907

```
2907      call splder(dddotmu, ddddo
2908      call splder(emmu, demmu, nt
2909      if (downlens) call splde
2910
2911      call SetTimeSteps
2912
2913      !$OMP PARALLEL DO DEFAULT
2914      do j2=1, TimeSteps%npoints
2915          call DoThermoSpline(j
2916      end do
2917      !$OMP END PARALLEL DO
2918
2919
2920      if ((CP%want_zstar .or. C
2921      if (CP%want_zdrag .or. CP
2922
2923      if (CP%DerivedParameters)
2924          rs = rombint(dsound_da
2925          DA = AngularDiameterD
2926
2927          ThermoDerivedParams(
2928          ThermoDerivedParams(
2929          ThermoDerivedParams(
2930          ThermoDerivedParams(
2931          ThermoDerivedParams(
2932          ThermoDerivedParams(
2933          rs = rombint(dsound_da
2934          ThermoDerivedParams(
2935          ThermoDerivedParams(
2936          ThermoDerivedParams(
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2927

```
2927      call splder(dddotmu, ddddo
2928      call splder(emmu, demmu, nt
2929      if (downlens) call splde
2930
2931      call SetTimeSteps
2932
2933
2934      if ((CP%want_zstar .or. C
2935      if (CP%want_zdrag .or. CP
2936
2937      if (CP%DerivedParameters)
2938          rs = rombint(dsound_da
2939          DA = AngularDiameterD
2940
2941          ThermoDerivedParams(
2942          ThermoDerivedParams(
2943          ThermoDerivedParams(
2944          ThermoDerivedParams(
2945          ThermoDerivedParams(
2946          rs = rombint(dsound_da
2947          ThermoDerivedParams(
2948          ThermoDerivedParams(
2949          ThermoDerivedParams(
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2937

```
2937      z_eq = (grhob+grhoc)/
2938      ThermoDerivedParams(
2939      a_eq = 1/(1+z_eq)
2940      ThermoDerivedParams(
2941      ThermoDerivedParams(
2942      ThermoDerivedParams(
2943
2944      if (associated(Backgr
2945          if (allocated(Bac
2946              deallocate(Ba
2947              noutput = size(Ba
2948              allocate(Backgrou
2949              do i=1,noutput
2950                  BackgroundOut
2951                  BackgroundOut
2952                  BackgroundOut
2953                  Backgroun
2954              end do
2955      end if
2956
2957      if (FeedbackLevel > 0
2958          write(*,'("Age of
2959          write(*,'("zstar
2960          write(*,'("r_s(zs
2961          write(*,'("100*th
2962          write(*,'("DA(zst
2963
2964          write(*,'("zdrag
2965          write(*,'("r_s(zd
2966
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2950

```
2950      z_eq = (grhob+grhoc)/
2951      ThermoDerivedParams(
2952      a_eq = 1/(1+z_eq)
2953      ThermoDerivedParams(
2954      ThermoDerivedParams(
2955      ThermoDerivedParams(
2956
2957      if (associated(Backgr
2958          if (allocated(Bac
2959              deallocate(Ba
2960              noutput = size(Ba
2961              allocate(Backgrou
2962              do i=1,noutput
2963                  BackgroundOut
2964                  BackgroundOut
2965                  BackgroundOut
2966                  Backgroun
2967              end do
2968      end if
2969
2970      if (FeedbackLevel > 0
2971          write(*,'("Age of
2972          write(*,'("zstar
2973          write(*,'("r_s(zs
2974          write(*,'("100*th
2975          write(*,'("DA(zst
2976
2977          write(*,'("zdrag
2978          write(*,'("r_s(zd
2979
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2967

```
2967      write(*,'("k_D(zs
2968      write(*,'("100*th
2969
2970      write(*,'("z_EQ (
2971      write(*,'("k_EQ M
2972      write(*,'("100*th
2973      write(*,'("100*th
2974
2975          end if
2976      end if
2977
2978      end subroutine inithermo
2979
2980
2981      subroutine SetTimeSteps
2982      real(dl) dtau0
2983      integer nri0, nstep
2984
2985      call Ranges_Init(TimeStep
2986
2987      call Ranges_Add_delta(Tim
2988
2989      ! Calculating the timeste
2990      if (CP%WantTensors) then
2991          dtau0=max(taurst/40,M
2992      else
2993          dtau0=Maxtau/500._dl/
2994          if (do_bispectrum) dt
2995          !Don't need this sinc
2996          ! if (CP%DoLensing)
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 2980

```
2980      write(*,'("k_D(zs
2981      write(*,'("100*th
2982
2983      write(*,'("z_EQ (
2984      write(*,'("k_EQ M
2985      write(*,'("100*th
2986      write(*,'("100*th
2987
2988          end if
2989      end if
2990
2991      end subroutine inithermo
2992
2993
2994      subroutine SetTimeSteps
2995      real(dl) dtau0
2996      integer nri0, nstep
2997
2998      call Ranges_Init(TimeStep
2999
3000      call Ranges_Add_delta(Tim
3001
3002      ! Calculating the timeste
3003      if (CP%WantTensors) then
3004          dtau0=max(taurst/40,M
3005      else
3006          dtau0=Maxtau/500._dl/
3007          if (do_bispectrum) dt
3008          !Don't need this sinc
3009          ! if (CP%DoLensing)
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 2997

```
2997      ! if (CP%AccurateBB)
2998      end if
2999
3000      call Ranges_Add_delta(Tim
3001
3002      if (CP%Reion%Reionization
3003          nri0=int(Reionization
3004          !Steps while reioniza
3005          call Ranges_Add(Times
3006      end if
3007
3008      !Create arrays out of the
3009      call Ranges_GetArray(Time
3010      nstep = TimeSteps%npoints
3011
3012      if (allocated(vis)) then
3013          deallocate(vis,dvis,d
3014          if (dowinlens) deallo
3015      end if
3016      allocate(vis(nstep),dvis(
3017      if (dowinlens) allocate(l
3018
3019      if (DebugMsgs .and. Feedb
3020
3021      end subroutine SetTimeSte
3022
3023
3024      subroutine ThermoData_Fre
3025      if (allocated(vis)) then
3026          deallocate(vis,dvis,d
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 3010

```
3010      ! if (CP%AccurateBB)
3011      end if
3012
3013      call Ranges_Add_delta(Tim
3014
3015      if (CP%Reion%Reionization
3016          nri0=int(Reionization
3017          !Steps while reioniza
3018          call Ranges_Add(Times
3019      end if
3020
3021      !Create arrays out of the
3022      call Ranges_GetArray(Time
3023      nstep = TimeSteps%npoints
3024
3025      if (DebugMsgs .and. Feedb
3026
3027      end subroutine SetTimeSte
3028
3029
3030      subroutine ThermoData_Fre
3031
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 3027

```
3027      if (downlens) deallo
3028      end if
3029      call Ranges_Free(TimeStep
3030
3031      end subroutine ThermoData
3032
3033      !cccccccccccccc
3034      subroutine DoThermoSpline
3035      integer j2,i
3036      real(dl) d,ddopac,tau
3037
3038      !      Cubic-spline interp
3039
3039      d=log(tau/tauminn)/dlntau
3040      i=int(d)
3041
3042      d=d-i
3043
3043      if (i < nthermo) then
3044      opac(j2)=dotmu(i)+d*(
3045          -2._dl*ddotmu(i)-
3046          +2._dl*(dotmu(i)-
3047      dopac(j2)=(ddotmu(i)+
3048          -ddotmu(i))-2._dl
3049          +dddotmu(i+1)+2._
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 3032

```
3032      call Ranges_Free(TimeStep
3033
3034      end subroutine ThermoData
3035
3036
3037      subroutine IonizationFunc
3038      vis, dvis, ddvis, exp
3039      real(dl), intent(in) :: t
3040      real(dl), intent(out):: o
3041      real(dl) d
3042      integer i
3043
3044      d=log(tau/tauminn)/dlntau
3045      i=int(d)
3046
3047      d=d-i
3048
3048      if (i < nthermo) then
3049      opac=dotmu(i)+d*(ddot
3050          -2._dl*ddotmu(i)-
3051          +2._dl*(dotmu(i)-
3052      dopac=(ddotmu(i)+d*(d
3053          -ddotmu(i))-2._dl
3054          +dddotmu(i+1)+2._
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 3050

```
3050      *dlntau)
3051      ddopac=(dddotmu(i)+d*
3052      -dddotmu(i))-2._d
3053      +d*(dddotmu(i)+d
3054      -dddotmu(i+1))))
3055      /(tau*dlntau)**2
3056      expmmu(j2)=emmu(i)+d*
3057      -2._dl*demmu(i)-d
3058      +2._dl*(emmu(i)-e
3059
3060      if (dowinlens) then
3061          lenswin(j2)=winle
3062          -2._dl*dwinle
3063          +2._dl*(winle
3064
3065      vis(j2)=opac(j2)*expm
3066      dvis(j2)=expmmu(j2)*(
3067      ddvis(j2)=expmmu(j2)*
3068      else
3069      opac(j2)=dotmu(ntherm
3070      dopac(j2)=ddotmu(nthe
3071      ddopac=dddotmu(ntherm
3072      expmmu(j2)=emmu(nther
3073      vis(j2)=opac(j2)*expm
3074      dvis(j2)=expmmu(j2)*(
3075      ddvis(j2)=expmmu(j2)*
3076      end if
3077      end subroutine DoThermoSp
3078
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 3055

```
3055      *dlntau)
3056      ddopac=(dddotmu(i)+d*
3057      -dddotmu(i))-2._d
3058      +d*(dddotmu(i)+d
3059      -dddotmu(i+1))))
3060      /(tau*dlntau)**2
3061      expmmu=emmu(i)+d*(dem
3062      -2._dl*demmu(i)-d
3063      +2._dl*(emmu(i)-e
3064
3065      if (dowinlens) then
3066          lenswin=winlens(i
3067          -2._dl*dwinle
3068          +2._dl*(winle
3069
3070      vis=opac*expmmu
3071      dvis=expmmu*(opac**2+
3072      ddvis=expmmu*(opac**3
3073      else
3074      opac=dotmu(nthermo)
3075      dopac=ddotmu(nthermo)
3076      ddopac=dddotmu(ntherm
3077      expmmu=emmu(nthermo)
3078      vis=opac*expmmu
3079      dvis=expmmu*(opac**2+
3080      ddvis=expmmu*(opac**3
3081      end if
3082      end subroutine Ionization
3083
3084
```

```
3079
3080     function ddamping_da(a)
3081     real(dl) :: ddamping_da
3082     real(dl), intent(in) :: a
3083     real(dl) :: R
3084     real(dl) :: dtauda
3085     external dtauda
3086
3087     R=r_drag0*a
3088     !ignoring reionisation, n
3089     ddamping_da = (R**2 + 16*
3090
3091     end function ddamping_da
3092
3093
3094     !!!!!!!!!!!!!!!!!!!!!!!
3095     !JH: functions and subrou
3096
3097     function doptdepth_dz(z)
3098     real(dl) :: doptdepth_dz
3099     real(dl), intent(in) :: z
3100     real(dl) :: a
3101     real(dl) :: dtauda
3102     external dtauda
3103
3104     a = 1._dl/(1._dl+z)
3105
3106     !ignoring reionisation, n
3107     doptdepth_dz = Recombinat
3108
```

```
3085
3086     function ddamping_da(a)
3087     real(dl) :: ddamping_da
3088     real(dl), intent(in) :: a
3089     real(dl) :: R
3090     real(dl) :: dtauda
3091     external dtauda
3092
3093     R=r_drag0*a
3094     !ignoring reionisation, n
3095     ddamping_da = (R**2 + 16*
3096
3097     end function ddamping_da
3098
3099
3100     !!!!!!!!!!!!!!!!!!!!!!!
3101     !JH: functions and subrou
3102
3103     function doptdepth_dz(z)
3104     real(dl) :: doptdepth_dz
3105     real(dl), intent(in) :: z
3106     real(dl) :: a
3107     real(dl) :: dtauda
3108     external dtauda
3109
3110     a = 1._dl/(1._dl+z)
3111
3112     !ignoring reionisation, n
3113     doptdepth_dz = Recombinat
3114
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 3109

```
3109      end function doptdepth_dz
3110
3111      function optdepth(z)
3112      real(dl) :: rombint2
3113      external rombint2
3114      real(dl) optdepth
3115      real(dl),intent(in) :: z
3116
3117      optdepth = rombint2(doptd
3118
3119      end function optdepth
3120
3121
3122      function ddragoptdepth_dz
3123      real(dl) :: ddragoptdepth
3124      real(dl), intent(in) :: z
3125      real(dl) :: a
3126      real(dl) :: dtauda
3127      external dtauda
3128
3129      a = 1._dl/(1._dl+z)
3130      ddragoptdepth_dz = doptde
3131
3132      end function ddragoptdept
3133
3134
3135      function dragoptdepth(z)
3136      real(dl) :: rombint2
3137      external rombint2
3138      real(dl) dragoptdepth
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 3115

```
3115      end function doptdepth_dz
3116
3117      function optdepth(z)
3118      real(dl) :: rombint2
3119      external rombint2
3120      real(dl) optdepth
3121      real(dl),intent(in) :: z
3122
3123      optdepth = rombint2(doptd
3124
3125      end function optdepth
3126
3127
3128      function ddragoptdepth_dz
3129      real(dl) :: ddragoptdepth
3130      real(dl), intent(in) :: z
3131      real(dl) :: a
3132      real(dl) :: dtauda
3133      external dtauda
3134
3135      a = 1._dl/(1._dl+z)
3136      ddragoptdepth_dz = doptde
3137
3138      end function ddragoptdept
3139
3140
3141      function dragoptdepth(z)
3142      real(dl) :: rombint2
3143      external rombint2
3144      real(dl) dragoptdepth
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 3139

```
3139      real(dl),intent(in) :: z
3140
3141      dragoptdepth = rombint2(
3142
3143      end function dragoptdepth
3144
3145
3146      subroutine find_z(func,zo
3147      real(dl), external :: fun
3148      real(dl), intent(out) ::
3149      real(dl) :: try1,try2,dif
3150      integer :: i
3151
3152      try1 = 0.d0
3153      try2 = 10000.d0
3154
3155      i=0
3156      diff = 10.d0
3157      do while (diff .gt. 1d-3)
3158          i=i+1
3159          if (i .eq. 100) then
3160              call GlobalError(
3161                  zout=0
3162                  return
3163          end if
3164
3165          diff = func(try2)-fun
3166          avg = 0.5d0*(try2+try
3167          if (func(avg) .gt. 1.
3168              try2 = avg
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 3145

```
3145      real(dl),intent(in) :: z
3146
3147      dragoptdepth = rombint2(
3148
3149      end function dragoptdepth
3150
3151
3152      subroutine find_z(func,zo
3153      real(dl), external :: fun
3154      real(dl), intent(out) ::
3155      real(dl) :: try1,try2,dif
3156      integer :: i
3157
3158      try1 = 0.d0
3159      try2 = 10000.d0
3160
3161      i=0
3162      diff = 10.d0
3163      do while (diff .gt. 1d-3)
3164          i=i+1
3165          if (i .eq. 100) then
3166              call GlobalError(
3167                  zout=0
3168                  return
3169          end if
3170
3171          diff = func(try2)-fun
3172          avg = 0.5d0*(try2+try
3173          if (func(avg) .gt. 1.
3174              try2 = avg
```


/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 3169

```
3169         else
3170             try1 = avg
3171         end if
3172     end do
3173
3174     zout = avg
3175
3176     end subroutine find_z
3177
3178     !!!!!!!!!!!!!!!!!!!!!!!!!!!!! end J
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 3175

```
3175         else
3176             try1 = avg
3177         end if
3178     end do
3179
3180     zout = avg
3181
3182     end subroutine find_z
3183
3184     !!!!!!!!!!!!!!!!!!!!!!!!!!!!! end J
```

```
subroutine GetBackgroundE
integer, intent(in) :: nt
real(dl), intent(in) :: t
real(dl) :: outputs(5, nt)
real(dl) spline_data(nthe
real(dl) :: d, tau, cs2b,
integer i, ix
```

```
call splini(spline_data,n
call splder(xe,ddxe,nther
call splder(Tb,ddTb,nther
```

```
outputs = 0
do ix = 1, ntimes
    tau = times(ix)
    if (tau < tauminn) cy
d=log(tau/tauminn)/dl
i=int(d)
d=d-i
```

/Users/lp1opa/Compare/camb_simdata/modules.f90, Top line: 3179

```
3179
3180
3181
end module ThermoData
```

/Users/lp1opa/Compare/camb_des/modules.f90, Top line: 3205

```
3205 call thermo(tau,cs2b,
3206
3207     if (i < nthermo) then
3208         outputs(1,ix)=xe(
3209             -2._dl*ddxe(i
3210             +2._dl*(xe(i)
3211             vis=emmu(i)+d*(de
3212             -2._dl*demmu(
3213             +2._dl*(emmu(
3214             Tbaryon = tb(i)+d
3215             -2._dl*ddtb(i
3216             +2._dl*(tb(i)
3217     else
3218         outputs(1,ix)=xe(
3219         vis = emmu(ntherm
3220         Tbaryon = Tb(nthe
3221     end if
3222
3223     outputs(2, ix) = opac
3224     outputs(3, ix) = opac
3225     outputs(4, ix) = cs2b
3226     outputs(5, ix) = Tbar
3227 end do
3228
3229 end subroutine GetBackgro
3230
3231 end module ThermoData
3232
```