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

```
0001          ! Equations module for da
0002          ! allowing for perturbati
0003          ! by Antony Lewis (http:/
0004
0005          ! Dec 2003, fixed (fatal)
0006          ! Changes to tight coupli
0007          ! June 2004, fixed proble
0008          ! Generate vector modes o
0009          ! August 2004, fixed reio
0010          ! Nov 2004, change massiv
0011          ! Apr 2005, added DoLateR
0012          ! June 2006, added suppor
0013          ! Nov 2006, tweak to high
0014          ! June 2011, improved rad
0015          !             merged fderi
0016          !             optimized ne
0017          ! Feb 2012, updated PPF v
0018          ! Feb 2013: fixed various
0019          ! Oct 2013: fix PPF, cons
0020          ! Mar 2014: fixes for ten
0021
0022        module LambdaGeneral
0023        use precision
0024        use  ModelParams
0025        implicit none
0026
0027        real(dl)  :: w_lam = -1_d
0028        ! w_lam is now w0
0029        !comoving sound speed. Al
0030        !(otherwise assumed const
```

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

```
0001          ! Equations module for da
0002          ! allowing for perturbati
0003          ! by Antony Lewis (http:/
0004
0005          ! Dec 2003, fixed (fatal)
0006          ! Changes to tight coupli
0007          ! June 2004, fixed proble
0008          ! Generate vector modes o
0009          ! August 2004, fixed reio
0010          ! Nov 2004, change massiv
0011          ! Apr 2005, added DoLateR
0012          ! June 2006, added suppor
0013          ! Nov 2006, tweak to high
0014          ! June 2011, improved rad
0015          !             merged fderi
0016          !             optimized ne
0017          ! Feb 2012, updated PPF v
0018          ! Feb 2013: fixed various
0019          ! Oct 2013: fix PPF, cons
0020          ! Mar 2014: fixes for ten
0021
0022        module LambdaGeneral
0023        use precision
0024        use  ModelParams
0025        implicit none
0026
0027        real(dl)  :: w_lam = -1_d
0028        ! w_lam is now w0
0029        !comoving sound speed. Al
0030        !(otherwise assumed const
```

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

```
0031        real(dl) :: cs2_lam = 1_d
0032        !cs2_lam now is ce^2
0033
0034        logical :: use_tabulated_
0035        real(dl) :: wa_ppf = 0._d
0036        real(dl) :: c_Gamma_ppf =
0037        integer, parameter :: nwm
0038        integer :: nw_ppf
0039        real(dl) w_ppf(nwmax), a_
0040        real(dl) rde(nde),ade(nde
0041        real(dl), parameter :: am
0042        logical :: is_cosmologica
0043        private nde,ddw_ppf,rde,a
0044
0045        contains
0046
0047        subroutine DarkEnergy_Rea
0048        use IniFile
0049        Type(TIniFile) :: Ini
0050        character(LEN=Ini_max_str
0051        integer i
0052
0053        if (Ini_HasKey_File(Ini,'
0054            stop 'input variables
0055        end if
0056
0057        use_tabulated_w = Ini_Rea
0058        if(.not. use_tabulated_w)
0059            w_lam = Ini_Read_Doub
0060            wa_ppf = Ini_Read_Dou
```

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

```
0031        real(dl) :: cs2_lam = 1_d
0032        !cs2_lam now is ce^2
0033
0034        logical :: use_tabulated_
0035        real(dl) :: wa_ppf = 0._d
0036        real(dl) :: c_Gamma_ppf =
0037        integer, parameter :: nwm
0038        integer :: nw_ppf
0039        real(dl) w_ppf(nwmax), a_
0040        real(dl) rde(nde),ade(nde
0041        real(dl), parameter :: am
0042        logical :: is_cosmologica
0043        private nde,ddw_ppf,rde,a
0044
0045        contains
0046
0047        subroutine DarkEnergy_Rea
0048        use IniFile
0049        Type(TIniFile) :: Ini
0050        character(LEN=Ini_max_str
0051        integer i
0052
0053        if (Ini_HasKey_File(Ini,'
0054            stop 'input variables
0055        end if
0056
0057        use_tabulated_w = Ini_Rea
0058        if(.not. use_tabulated_w)
0059            w_lam = Ini_Read_Doub
0060            wa_ppf = Ini_Read_Dou
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 61

```
0061            if (Feedback >0) writ
0062        else
0063            wafile = Ini_Read_Str
0064            open(unit=10,file=waf
0065            nw_ppf=0
0066            do i=1,nwmax+1
0067                read(10,*,end=100
0068                a_ppf(i)=dlog(a_p
0069                nw_ppf=nw_ppf+1
0070            enddo
0071            write(*,'("Note: ", a
0072            write(*,*)'Increase n
0073            stop
0074    100     close(10)
0075            write(*,'("read in ",
0076            call setddwa
0077            call interpolrde
0078        endif
0079        cs2_lam = Ini_Read_Double
0080        call setcgammappf
0081
0082        end subroutine DarkEnergy
0083
0084
0085        subroutine setddwa
0086        real(dl), parameter :: wl
0087
0088        call spline(a_ppf,w_ppf,n
0089
0090        end subroutine setddwa
```

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

```
0061            if (Feedback >0) writ
0062        else
0063            wafile = Ini_Read_Str
0064            open(unit=10,file=waf
0065            nw_ppf=0
0066            do i=1,nwmax+1
0067                read(10,*,end=100
0068                a_ppf(i)=dlog(a_p
0069                nw_ppf=nw_ppf+1
0070            enddo
0071            write(*,'("Note: ", a
0072            write(*,*)'Increase n
0073            stop
0074    100     close(10)
0075            write(*,'("read in ",
0076            call setddwa
0077            call interpolrde
0078        endif
0079        cs2_lam = Ini_Read_Double
0080        call setcgammappf
0081
0082        end subroutine DarkEnergy
0083
0084
0085        subroutine setddwa
0086        real(dl), parameter :: wl
0087
0088        call spline(a_ppf,w_ppf,n
0089
0090        end subroutine setddwa
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 91

```
0091
0092
0093        function w_de(a)
0094        real(dl) :: w_de, al
0095        real(dl), intent(IN) :: a
0096
0097        if(.not. use_tabulated_w)
0098            w_de=w_lam+wa_ppf*(1.
0099        else
0100            al=dlog(a)
0101            if(al.lt.a_ppf(1)) th
0102                w_de=w_ppf(1)
0103            elseif(al.gt.a_ppf(nw
0104                w_de=w_ppf(nw_ppf
0105            else
0106                call cubicsplint(
0107            endif
0108        endif
0109        end function w_de  ! equa
0110
0111
0112        function drdlna_de(al)
0113        real(dl) :: drdlna_de, a
0114        real(dl), intent(IN) :: a
0115
0116        a=dexp(al)
0117        drdlna_de=3._dl*(1._dl+w_
0118
0119        end function drdlna_de
0120
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 91

```
0091
0092
0093        function w_de(a)
0094        real(dl) :: w_de, al
0095        real(dl), intent(IN) :: a
0096
0097        if(.not. use_tabulated_w)
0098            w_de=w_lam+wa_ppf*(1.
0099        else
0100            al=dlog(a)
0101            if(al.lt.a_ppf(1)) th
0102                w_de=w_ppf(1)
0103            elseif(al.gt.a_ppf(nw
0104                w_de=w_ppf(nw_ppf
0105            else
0106                call cubicsplint(
0107            endif
0108        endif
0109        end function w_de  ! equa
0110
0111
0112        function drdlna_de(al)
0113        real(dl) :: drdlna_de, a
0114        real(dl), intent(IN) :: a
0115
0116        a=dexp(al)
0117        drdlna_de=3._dl*(1._dl+w_
0118
0119        end function drdlna_de
0120
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 121

```
0121
0122        subroutine interpolrde
0123        real(dl), parameter :: rl
0124        real(dl) :: atol, almin,
0125        integer :: i
0126        external rombint
0127        atol=1.d-5
0128        almin=dlog(amin)
0129        do i=1,nde
0130            al=almin-almin/(nde-1
0131            fint=rombint(drdlna_d
0132            ade(i)=al
0133            rde(i)=dexp(fint) !rh
0134        enddo
0135        call spline(ade,rde,nde,r
0136        end subroutine interpolrd
0137
0138        function grho_de(a)  !8 p
0139        real(dl) :: grho_de, al,
0140        real(dl), intent(IN) :: a
0141        external rombint
0142
0143        if(.not. use_tabulated_w)
0144            grho_de=grhov*a**(1._
0145        else
0146            if(a.eq.0.d0)then
0147                grho_de=0.d0
0148            else
0149                al=dlog(a)
0150                if(al.lt.ade(1))t
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 121

```
0121
0122        subroutine interpolrde
0123        real(dl), parameter :: rl
0124        real(dl) :: atol, almin,
0125        integer :: i
0126        external rombint
0127        atol=1.d-5
0128        almin=dlog(amin)
0129        do i=1,nde
0130            al=almin-almin/(nde-1
0131            fint=rombint(drdlna_d
0132            ade(i)=al
0133            rde(i)=dexp(fint) !rh
0134        enddo
0135        call spline(ade,rde,nde,r
0136        end subroutine interpolrd
0137
0138        function grho_de(a)  !8 p
0139        real(dl) :: grho_de, al,
0140        real(dl), intent(IN) :: a
0141
0142        if(.not. use_tabulated_w)
0143            grho_de=grhov*a**(1._
0144        else
0145            if(a.eq.0.d0)then
0146                grho_de=0.d0
0147            else
0148                al=dlog(a)
0149                if(al.lt.ade(1))t
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 151

```
0151 |                     fint=rde(1)*(
0152 |                 else
0153 |                     call cubicspl
0154 |                 endif
0155 |                 grho_de=grhov*fin
0156 |             endif
0157 |         endif
0158 |         end function grho_de
0159 |
0160 |         !-------------------------
0161 |         SUBROUTINE cubicsplint(xa
0162 |         INTEGER n
0163 |         real(dl)x,y,xa(n),y2a(n),
0164 |         INTEGER k,khi,klo
0165 |         real(dl)a,b,h
0166 |         klo=1
0167 |         khi=n
0168 |   1     if (khi-klo.gt.1) then
0169 |             k=(khi+klo)/2
0170 |             if(xa(k).gt.x)then
0171 |                 khi=k
0172 |             else
0173 |                 klo=k
0174 |             endif
0175 |             goto 1
0176 |         endif
0177 |         h=xa(khi)-xa(klo)
0178 |         if (h.eq.0.) stop 'bad xa
0179 |         a=(xa(khi)-x)/h
0180 |         b=(x-xa(klo))/h
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 150

```
0150 |                     fint=rde(1)*(
0151 |                 else
0152 |                     call cubicspl
0153 |                 endif
0154 |                 grho_de=grhov*fin
0155 |             endif
0156 |         endif
0157 |         end function grho_de
0158 |
0159 |         !-------------------------
0160 |         SUBROUTINE cubicsplint(xa
0161 |         INTEGER n
0162 |         real(dl)x,y,xa(n),y2a(n),
0163 |         INTEGER k,khi,klo
0164 |         real(dl)a,b,h
0165 |         klo=1
0166 |         khi=n
0167 |   1     if (khi-klo.gt.1) then
0168 |             k=(khi+klo)/2
0169 |             if(xa(k).gt.x)then
0170 |                 khi=k
0171 |             else
0172 |                 klo=k
0173 |             endif
0174 |             goto 1
0175 |         endif
0176 |         h=xa(khi)-xa(klo)
0177 |         if (h.eq.0.) stop 'bad xa
0178 |         a=(xa(khi)-x)/h
0179 |         b=(x-xa(klo))/h
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 181

```
0181        y=a*ya(klo)+b*ya(khi)+&
0182        ((a**3-a)*y2a(klo)+(b**3-
0183        END SUBROUTINE cubicsplin
0184        !-----------------------
0185
0186
0187        subroutine setcgammappf
0188
0189        c_Gamma_ppf=0.4d0*sqrt(cs
0190
0191        end subroutine setcgammap
0192
0193
0194        end module LambdaGeneral
0195
0196        !ccccccccccccccccccccccc
0197
0198
0199        !Return OmegaK - modify t
0200        function GetOmegak()
0201        use precision
0202        use ModelParams
0203        real(dl)  GetOmegak
0204        GetOmegak = 1 - (CP%omega
0205
0206        end function GetOmegak
0207
0208
0209        subroutine init_backgroun
0210        use LambdaGeneral
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 180

```
0180        y=a*ya(klo)+b*ya(khi)+&
0181            ((a**3-a)*y2a(klo)+(b
0182        END SUBROUTINE cubicsplin
0183        !-----------------------
0184
0185
0186        subroutine setcgammappf
0187
0188        c_Gamma_ppf=0.4d0*sqrt(cs
0189
0190        end subroutine setcgammap
0191
0192
0193        end module LambdaGeneral
0194
0195        !ccccccccccccccccccccccc
0196
0197
0198        !Return OmegaK - modify t
0199        function GetOmegak()
0200        use precision
0201        use ModelParams
0202        real(dl)  GetOmegak
0203        GetOmegak = 1 - (CP%omega
0204
0205        end function GetOmegak
0206
0207
0208        subroutine init_backgroun
0209        use LambdaGeneral
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 211

```
0211        !This is only called once
0212        !It is called before firs
0213        !massive neutrinos are in
0214        is_cosmological_constant
0215        end  subroutine init_back
0216
0217
0218        !Background evolution
0219        function dtauda(a)
0220        !get d tau / d a
0221        use precision
0222        use ModelParams
0223        use MassiveNu
0224        use LambdaGeneral
0225        implicit none
0226        real(dl) dtauda
0227        real(dl), intent(IN) :: a
0228        real(dl) rhonu,grhoa2, a2
0229        integer nu_i
0230
0231        a2=a**2
0232
0233        !  8*pi*G*rho*a**4.
0234        grhoa2=grhok*a2+(grhoc+gr
0235        if (is_cosmological_const
0236            grhoa2=grhoa2+grhov*a
0237        else
0238            grhoa2=grhoa2+ grho_d
0239        end if
0240
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 210

```
0210        !This is only called once
0211        !It is called before firs
0212        !massive neutrinos are in
0213        is_cosmological_constant
0214        end  subroutine init_back
0215
0216
0217        !Background evolution
0218        function dtauda(a)
0219        !get d tau / d a
0220        use precision
0221        use ModelParams
0222        use MassiveNu
0223        use LambdaGeneral
0224        implicit none
0225        real(dl) dtauda
0226        real(dl), intent(IN) :: a
0227        real(dl) rhonu,grhoa2, a2
0228        integer nu_i
0229
0230        a2=a**2
0231
0232        !  8*pi*G*rho*a**4.
0233        grhoa2=grhok*a2+(grhoc+gr
0234        if (is_cosmological_const
0235            grhoa2=grhoa2+grhov*a
0236        else
0237            grhoa2=grhoa2+ grho_d
0238        end if
0239
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 241

```
0241        if (CP%Num_Nu_massive /=
0242            !Get massive neutrino
0243            do nu_i = 1, CP%nu_ma
0244                call Nu_rho(a*nu_
0245                grhoa2=grhoa2+rho
0246            end do
0247        end if
0248
0249        dtauda=sqrt(3/grhoa2)
0250
0251        end function dtauda
0252
0253        !ccccccccccccccccccccccc
0254
0255        !Gauge-dependent perturba
0256
0257        module GaugeInterface
0258        use precision
0259        use ModelParams
0260        use MassiveNu
0261        use LambdaGeneral
0262        use Errors
0263        use Transfer
0264        implicit none
0265        public
0266
0267        !Description of this file
0268        character(LEN=*), paramet
0269
0270        integer, parameter :: bas
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 240

```
0240        if (CP%Num_Nu_massive /=
0241            !Get massive neutrino
0242            do nu_i = 1, CP%nu_ma
0243                call Nu_rho(a*nu_
0244                grhoa2=grhoa2+rho
0245            end do
0246        end if
0247
0248        dtauda=sqrt(3/grhoa2)
0249
0250        end function dtauda
0251
0252        !ccccccccccccccccccccccc
0253
0254        !Gauge-dependent perturba
0255
0256        module GaugeInterface
0257        use precision
0258        use ModelParams
0259        use MassiveNu
0260        use LambdaGeneral
0261        use Errors
0262        use Transfer
0263        implicit none
0264        public
0265
0266        !Description of this file
0267        character(LEN=*), paramet
0268
0269        integer, parameter :: bas
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 271

```
0271
0272        logical :: DoTensorNeutri
0273
0274        logical :: DoLateRadTrunc
0275        !if true, use smooth appr
0276        !small scales, saving evo
0277
0278        logical, parameter :: sec
0279
0280        real(dl) :: Magnetic = 0.
0281        !Vector mode anisotropic
0282        real(dl) :: vec_sig0 = 1.
0283        !Vector mode shear
0284        integer, parameter :: max
0285        !Note higher values incre
0286
0287        !Supported scalar initial
0288        integer, parameter :: ini
0289        initial_iso_baryon=3,  in
0290        integer, parameter :: ini
0291
0292        type EvolutionVars
0293            real(dl) q, q2
0294            real(dl) k_buf,k2_buf
0295
0296            integer w_ix !Index o
0297            integer r_ix !Index o
0298            integer g_ix !Index o
0299
0300            integer q_ix !index i
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 270

```
0270
0271        logical :: DoTensorNeutri
0272
0273        logical :: DoLateRadTrunc
0274        !if true, use smooth appr
0275        !small scales, saving evo
0276
0277        logical, parameter :: sec
0278
0279        real(dl) :: Magnetic = 0.
0280        !Vector mode anisotropic
0281        real(dl) :: vec_sig0 = 1.
0282        !Vector mode shear
0283        integer, parameter :: max
0284        !Note higher values incre
0285
0286        !Supported scalar initial
0287        integer, parameter :: ini
0288            initial_iso_baryon=3,
0289        integer, parameter :: ini
0290
0291        type EvolutionVars
0292            real(dl) q, q2
0293            real(dl) k_buf,k2_buf
0294
0295            integer w_ix !Index o
0296            integer r_ix !Index o
0297            integer g_ix !Index o
0298
0299            integer q_ix !index i
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 301

```
0301 |             logical TransferOnly
0302 |
0303 |             !       nvar  - numbe
0304 |             integer nvar,nvart, n
0305 |
0306 |             !Max_l for the variou
0307 |             integer lmaxg,lmaxnr,
0308 |             integer lmaxnrt, lmax
0309 |             logical EvolveTensorM
0310 |             integer lmaxnrv, lmax
0311 |
0312 |             integer polind  !inde
0313 |
0314 |             !array indices for ma
0315 |             integer nu_ix(max_nu)
0316 |             integer nq(max_nu), l
0317 |             logical has_nu_relati
0318 |
0319 |             !Initial values for m
0320 |             !to non-relativistic
0321 |             real(dl) G11(max_nu),
0322 |             !True when using non-
0323 |             logical MassiveNuAppr
0324 |             real(dl) MassiveNuApp
0325 |
0326 |             !True when truncating
0327 |             logical high_ktau_neu
0328 |
0329 |             !Massive neutrino sch
0330 |             integer NuMethod
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 300

```
0300 |             logical TransferOnly
0301 |
0302 |             !       nvar  - numbe
0303 |             integer nvar,nvart, n
0304 |
0305 |             !Max_l for the variou
0306 |             integer lmaxg,lmaxnr,
0307 |             integer lmaxnrt, lmax
0308 |             logical EvolveTensorM
0309 |             integer lmaxnrv, lmax
0310 |
0311 |             integer polind  !inde
0312 |
0313 |             !array indices for ma
0314 |             integer nu_ix(max_nu)
0315 |             integer nq(max_nu), l
0316 |             logical has_nu_relati
0317 |
0318 |             !Initial values for m
0319 |             !to non-relativistic
0320 |             real(dl) G11(max_nu),
0321 |             !True when using non-
0322 |             logical MassiveNuAppr
0323 |             real(dl) MassiveNuApp
0324 |
0325 |             !True when truncating
0326 |             logical high_ktau_neu
0327 |
0328 |             !Massive neutrino sch
0329 |             integer NuMethod
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 331

```
0331
0332            !True when using tigh
0333            logical TightCoupling
0334            real(dl) TightSwitcho
0335
0336            !Numer of scalar equa
0337            integer ScalEqsToProp
0338            integer TensEqsToProp
0339            !beta > 1 for closed
0340            integer FirstZerolFor
0341            !Tensor vars
0342            real(dl) aux_buf
0343
0344            real(dl) pig, pigdot
0345            real(dl) poltruncfac
0346
0347            !PPF parameters
0348            real(dl) dgrho_e_ppf,
0349
0350            logical no_nu_multpol
0351            integer lmaxnu_tau(ma
0352            logical nu_nonrelativ
0353
0354            real(dl) denlk(max_l_
0355            real(dl) Kf(max_l_evo
0356
0357            integer E_ix, B_ix !t
0358            real(dl) denlkt(4,max
0359            real, pointer :: Outp
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 330

```
0330
0331            !True when using tigh
0332            logical TightCoupling
0333            real(dl) TightSwitcho
0334
0335            !Numer of scalar equa
0336            integer ScalEqsToProp
0337            integer TensEqsToProp
0338            !beta > 1 for closed
0339            integer FirstZerolFor
0340            !Tensor vars
0341            real(dl) aux_buf
0342
0343            real(dl) pig, pigdot
0344            real(dl) poltruncfac
0345
0346            !PPF parameters
0347            real(dl) dgrho_e_ppf,
0348
0349            logical no_nu_multpol
0350            integer lmaxnu_tau(ma
0351            logical nu_nonrelativ
0352
0353            real(dl) denlk(max_l_
0354            real(dl) Kf(max_l_evo
0355
0356            integer E_ix, B_ix !t
0357            real(dl) denlkt(4,max
0358            real, pointer :: Outp
0359
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 360

```
0360        end type EvolutionVars
0361
0362        !precalculated arrays
0363        real(dl) polfac(max_l_evo
0364
0365        real(dl), parameter :: ep
0366        integer, parameter :: lma
0367
0368        real(dl) epsw
0369        real(dl) nu_tau_notmassle
0370        contains
0371
0372
0373        subroutine GaugeInterface
0374        type(EvolutionVars) EV
0375        real(dl) c(24),w(EV%nvar,
0376        integer ind
0377
0378        call dverk(EV,EV%ScalEqsT
0379        if (ind==-3) then
0380            call GlobalError('Dve
0381            //'requirement  with
0382            //'equal to hmin, whi
0383            //'--- but most likel
0384            //'compiling with bou
0385        end if
0386        end subroutine GaugeInter
0387
0388        function next_nu_nq(nq) r
0389        integer, intent(in) :: nq
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 360

```
0360        end type EvolutionVars
0361
0362        !precalculated arrays
0363        real(dl) polfac(max_l_evo
0364
0365        real(dl), parameter :: ep
0366        integer, parameter :: lma
0367
0368        real(dl) epsw
0369        real(dl) nu_tau_notmassle
0370        contains
0371
0372
0373        subroutine GaugeInterface
0374        type(EvolutionVars) EV
0375        real(dl) c(24),w(EV%nvar,
0376        integer ind
0377
0378        call dverk(EV,EV%ScalEqsT
0379        if (ind==-3) then
0380            call GlobalError('Dve
0381                //'requirement  w
0382                //'equal to hmin,
0383                //'--- but most l
0384                //'compiling with
0385        end if
0386        end subroutine GaugeInter
0387
0388        function next_nu_nq(nq) r
0389        integer, intent(in) :: nq
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 390

```
0390        integer q, next_nq
0391
0392        if (nq==0) then
0393            next_nq=1
0394        else
0395            q = nu_q(nq)
0396            if (q>=10) then
0397                next_nq = nqmax
0398            else
0399                next_nq = nq+1
0400            end if
0401        end if
0402
0403        end function next_nu_nq
0404
0405        recursive subroutine Gaug
0406        use ThermoData
0407        type(EvolutionVars) EV, E
0408        real(dl) c(24),w(EV%nvar,
0409        integer ind, nu_i
0410        real(dl) cs2, opacity, do
0411        real(dl) tau_switch_ktau,
0412        real(dl) tau_switch_no_nu
0413        real(dl) noSwitch, smallT
0414
0415        noSwitch= CP%tau0+1
0416        smallTime =  min(tau, 1/E
0417
0418        tau_switch_ktau = noSwitc
0419        tau_switch_no_nu_multpole
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 390

```
0390        integer q, next_nq
0391
0392        if (nq==0) then
0393            next_nq=1
0394        else
0395            q = nu_q(nq)
0396            if (q>=10) then
0397                next_nq = nqmax
0398            else
0399                next_nq = nq+1
0400            end if
0401        end if
0402
0403        end function next_nu_nq
0404
0405        recursive subroutine Gaug
0406        use ThermoData
0407        type(EvolutionVars) EV, E
0408        real(dl) c(24),w(EV%nvar,
0409        integer ind, nu_i
0410        real(dl) cs2, opacity, do
0411        real(dl) tau_switch_ktau,
0412        real(dl) tau_switch_no_nu
0413        real(dl) noSwitch, smallT
0414
0415        noSwitch= CP%tau0+1
0416        smallTime =  min(tau, 1/E
0417
0418        tau_switch_ktau = noSwitc
0419        tau_switch_no_nu_multpole
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 420

```
0420        tau_switch_no_phot_multpo
0421
0422        !Massive neutrino switche
0423        tau_switch_nu_massless =
0424        tau_switch_nu_nonrel = no
0425        tau_switch_nu_massive= no
0426
0427        !Evolve equations from ta
0428
0429        if (.not. EV%high_ktau_ne
0430            tau_switch_ktau=  max
0431        end if
0432
0433        if (CP%Num_Nu_massive /=
0434            do nu_i = 1, CP%Nu_ma
0435                if (EV%nq(nu_i) /
0436                    tau_switch_nu
0437                else if (.not. EV
0438                    tau_switch_nu
0439                else if (EV%NuMet
0440                    tau_switch_nu
0441                end if
0442            end do
0443        end if
0444
0445        if (DoLateRadTruncation)
0446            if (.not. EV%no_nu_mu
0447            tau_switch_no_nu_mult
0448
0449            if (.not. EV%no_phot_
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 420

```
0420        tau_switch_no_phot_multpo
0421
0422        !Massive neutrino switche
0423        tau_switch_nu_massless =
0424        tau_switch_nu_nonrel = no
0425        tau_switch_nu_massive= no
0426
0427        !Evolve equations from ta
0428
0429        if (.not. EV%high_ktau_ne
0430            tau_switch_ktau=  max
0431        end if
0432
0433        if (CP%Num_Nu_massive /=
0434            do nu_i = 1, CP%Nu_ma
0435                if (EV%nq(nu_i) /
0436                    tau_switch_nu
0437                else if (.not. EV
0438                    tau_switch_nu
0439                else if (EV%NuMet
0440                    tau_switch_nu
0441                end if
0442            end do
0443        end if
0444
0445        if (DoLateRadTruncation)
0446            if (.not. EV%no_nu_mu
0447                tau_switch_no_nu_
0448
0449            if (.not. EV%no_phot_
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 450

```
0450            tau_switch_no_phot_mu
0451        end if
0452
0453        next_switch = min(tau_swi
0454        tau_switch_no_nu_multpole
0455
0456        if (next_switch < tauend)
0457            if (next_switch > tau
0458                call GaugeInterfa
0459                if (global_error_
0460            end if
0461
0462            EVout=EV
0463
0464            if (next_switch == EV
0465                !TightCoupling
0466                EVout%TightCoupli
0467                EVout%TightSwitch
0468                call SetupScalarA
0469                call CopyScalarVa
0470                EV=EVout
0471                y=yout
0472                ind=1
0473                !Set up variables
0474                y(EV%g_ix+2) = EV
0475                call thermo(tau,c
0476
0477                if (second_order_
0478                    ! Francis-Yan
0479
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 450

```
0450            tau_switch_no_pho
0451        end if
0452
0453        next_switch = min(tau_swi
0454            tau_switch_no_nu_mult
0455
0456        if (next_switch < tauend)
0457            if (next_switch > tau
0458                call GaugeInterfa
0459                if (global_error_
0460            end if
0461
0462            EVout=EV
0463
0464            if (next_switch == EV
0465                !TightCoupling
0466                EVout%TightCoupli
0467                EVout%TightSwitch
0468                call SetupScalarA
0469                call CopyScalarVa
0470                EV=EVout
0471                y=yout
0472                ind=1
0473                !Set up variables
0474                y(EV%g_ix+2) = EV
0475                call thermo(tau,c
0476
0477                if (second_order_
0478                    ! Francis-Yan
0479
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 480

```
0480                        y(EV%g_ix+3)
0481                        (3._dl/7._dl)
0482
0483                        y(EV%polind+2
0484                        (25._dl/16._d
0485                        EV%pig*(EV%k_
0486                        y(EV%polind+3
0487                        dopacity/opac
0488                        (1._dl+(5._dl
0489                    else
0490                        y(EV%g_ix+3)
0491                        y(EV%polind+2
0492                        y(EV%polind+3
0493                    end if
0494                else if (next_switch=
0495                    !k tau >> 1, evol
0496                    EVout%high_ktau_n
0497                    EV%nq(1:CP%Nu_mas
0498                    call SetupScalarA
0499                    call CopyScalarVa
0500                    y=yout
0501                    EV=EVout
0502                else if (next_switch
0503                    !Mass starts to b
0504                    do nu_i = 1, CP%N
0505                        if (EV%nq(nu_
0506                        next_switch =
0507                            EVOut%nq(
0508                            call Setu
0509                            call Copy
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 480

```
0480                        y(EV%g_ix+3)
0481                            (3._dl/7.
0482
0483                        y(EV%polind+2
0484                            (25._dl/1
0485                            EV%pig*(E
0486                        y(EV%polind+3
0487                            dopacity/
0488                            (1._dl+(5
0489                    else
0490                        y(EV%g_ix+3)
0491                        y(EV%polind+2
0492                        y(EV%polind+3
0493                    end if
0494                else if (next_switch=
0495                    !k tau >> 1, evol
0496                    EVout%high_ktau_n
0497                    EV%nq(1:CP%Nu_mas
0498                    call SetupScalarA
0499                    call CopyScalarVa
0500                    y=yout
0501                    EV=EVout
0502                else if (next_switch
0503                    !Mass starts to b
0504                    do nu_i = 1, CP%N
0505                        if (EV%nq(nu_
0506                            next_swit
0507                        EVOut%nq(nu_i
0508                        call SetupSca
0509                        call CopyScal
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 510

```
0510                        EV=EVout
0511                        y=yout
0512                        exit
0513                    end if
0514                end do
0515            else if (next_switch
0516                !Neutrino becomes
0517                do nu_i = 1, CP%N
0518                    if (.not. EV%
0519                        EVout%nu_
0520                        call Setu
0521                        call Copy
0522                        EV=EVout
0523                        y=yout
0524                        exit
0525                    end if
0526                end do
0527            else if (next_switch
0528                !Very non-relativ
0529                do nu_i = 1, CP%N
0530                    if (.not. EV%
0531                        call Swit
0532                        exit
0533                    end if
0534                end do
0535            else if (next_switch=
0536                !Turn off neutrin
0537                ind=1
0538                EVout%no_nu_multp
0539                EVOut%nq(1:CP%Nu_
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 510

```
0510                        EV=EVout
0511                        y=yout
0512                        exit
0513                    end if
0514                end do
0515            else if (next_switch
0516                !Neutrino becomes
0517                do nu_i = 1, CP%N
0518                    if (.not. EV%
0519                        EVout%nu_
0520                        call Setu
0521                        call Copy
0522                        EV=EVout
0523                        y=yout
0524                        exit
0525                    end if
0526                end do
0527            else if (next_switch
0528                !Very non-relativ
0529                do nu_i = 1, CP%N
0530                    if (.not. EV%
0531                        call Swit
0532                        exit
0533                    end if
0534                end do
0535            else if (next_switch=
0536                !Turn off neutrin
0537                ind=1
0538                EVout%no_nu_multp
0539                EVOut%nq(1:CP%Nu_
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 540

```
0540 |                  call SetupScalarA
0541 |                  call CopyScalarVa
0542 |                  y=yout
0543 |                  EV=EVout
0544 |              else if (next_switch=
0545 |                  !Turn off photon
0546 |                  ind=1
0547 |                  EVout%no_phot_mul
0548 |                  call SetupScalarA
0549 |                  call CopyScalarVa
0550 |                  y=yout
0551 |                  EV=EVout
0552 |              end if
0553 |
0554 |              call GaugeInterface_E
0555 |              return
0556 |          end if
0557 |
0558 |          call GaugeInterface_ScalE
0559 |
0560 |          end subroutine GaugeInter
0561 |
0562 |          subroutine GaugeInterface
0563 |          use ThermoData
0564 |          type(EvolutionVars) EV, E
0565 |          real(dl) c(24),w(EV%nvart
0566 |          integer ind
0567 |          real(dl) opacity, cs2
0568 |
0569 |          if (EV%TensTightCoupling
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 540

```
0540 |                  call SetupScalarA
0541 |                  call CopyScalarVa
0542 |                  y=yout
0543 |                  EV=EVout
0544 |              else if (next_switch=
0545 |                  !Turn off photon
0546 |                  ind=1
0547 |                  EVout%no_phot_mul
0548 |                  call SetupScalarA
0549 |                  call CopyScalarVa
0550 |                  y=yout
0551 |                  EV=EVout
0552 |              end if
0553 |
0554 |              call GaugeInterface_E
0555 |              return
0556 |          end if
0557 |
0558 |          call GaugeInterface_ScalE
0559 |
0560 |          end subroutine GaugeInter
0561 |
0562 |          subroutine GaugeInterface
0563 |          use ThermoData
0564 |          type(EvolutionVars) EV, E
0565 |          real(dl) c(24),w(EV%nvart
0566 |          integer ind
0567 |          real(dl) opacity, cs2
0568 |
0569 |          if (EV%TensTightCoupling
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 570

```
0570            if (EV%TightSwitchoff
0571                call dverk(EV,EV%
0572            end if
0573            EVOut=EV
0574            EVOut%TensTightCoupli
0575            call SetupTensorArray
0576            call CopyTensorVariab
0577            Ev = EvOut
0578            y=yout
0579            call thermo(tau,cs2,o
0580            y(EV%g_ix+2)= 32._dl/
0581            y(EV%E_ix+2) = y(EV%g
0582        end if
0583
0584        call dverk(EV,EV%TensEqsT
0585
0586
0587        end subroutine GaugeInter
0588
0589        function DeltaTimeMaxed(a
0590        real(dl) a1,a2,t
0591        real(dl), optional :: tol
0592        if (a1>1._dl) then
0593            t=0
0594        elseif (a2 > 1._dl) then
0595            t = DeltaTime(a1,1.01
0596        else
0597            t= DeltaTime(a1,a2, t
0598        end if
0599        end function DeltaTimeMax
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 570

```
0570            if (EV%TightSwitchoff
0571                call dverk(EV,EV%
0572            end if
0573            EVOut=EV
0574            EVOut%TensTightCoupli
0575            call SetupTensorArray
0576            call CopyTensorVariab
0577            Ev = EvOut
0578            y=yout
0579            call thermo(tau,cs2,o
0580            y(EV%g_ix+2)= 32._dl/
0581            y(EV%E_ix+2) = y(EV%g
0582        end if
0583
0584        call dverk(EV,EV%TensEqsT
0585
0586
0587        end subroutine GaugeInter
0588
0589        function DeltaTimeMaxed(a
0590        real(dl) a1,a2,t
0591        real(dl), optional :: tol
0592        if (a1>1._dl) then
0593            t=0
0594        elseif (a2 > 1._dl) then
0595            t = DeltaTime(a1,1.01
0596        else
0597            t= DeltaTime(a1,a2, t
0598        end if
0599        end function DeltaTimeMax
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 600

```
0600
0601        subroutine GaugeInterface
0602        !Precompute various array
0603        integer j, nu_i
0604        real(dl) a_nonrel, a_mass
0605
0606        epsw = 100/CP%tau0
0607
0608        if (CP%WantScalars) then
0609            do j=2,max_l_evolve
0610                polfac(j)=real((j
0611            end do
0612        end if
0613
0614        if (CP%WantVectors) then
0615            do j=2,max_l_evolve
0616                vecfac(j)=real((j
0617                vecfacpol(j)=real
0618            end do
0619        end if
0620
0621        do j=1,max_l_evolve
0622            denl(j)=1._dl/(2*j+1)
0623        end do
0624
0625        do nu_i=1, CP%Nu_Mass_eig
0626            nu_mass = max(0.1_dl,
0627            a_mass =  1.e-1_dl/nu
0628            !if (HighAccuracyDefa
0629            time=DeltaTime(0._dl,
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 600

```
0600
0601        subroutine GaugeInterface
0602        !Precompute various array
0603        integer j, nu_i
0604        real(dl) a_nonrel, a_mass
0605
0606        epsw = 100/CP%tau0
0607
0608        if (CP%WantScalars) then
0609            do j=2,max_l_evolve
0610                polfac(j)=real((j
0611            end do
0612        end if
0613
0614        if (CP%WantVectors) then
0615            do j=2,max_l_evolve
0616                vecfac(j)=real((j
0617                vecfacpol(j)=real
0618            end do
0619        end if
0620
0621        do j=1,max_l_evolve
0622            denl(j)=1._dl/(2*j+1)
0623        end do
0624
0625        do nu_i=1, CP%Nu_Mass_eig
0626            nu_mass = max(0.1_dl,
0627            a_mass =  1.e-1_dl/nu
0628            !if (HighAccuracyDefa
0629            time=DeltaTime(0._dl,
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 630

```
0630            nu_tau_notmassless(1,
0631            do j=2,nqmax
0632                !times when each
0633                time= time + Delt
0634                nu_tau_notmassles
0635            end do
0636
0637            a_nonrel =  2.5d0/nu_
0638            nu_tau_nonrelativisti
0639            a_massive =  17.d0/nu
0640            nu_tau_massive(nu_i)
0641        end do
0642
0643        end subroutine GaugeInter
0644
0645
0646        subroutine SetupScalarArr
0647        !Set up array indices aft
0648        use MassiveNu
0649        !Set the numer of equatio
0650        type(EvolutionVars) EV
0651        integer, intent(out), opt
0652        integer neq, maxeq, nu_i
0653
0654        neq=basic_num_eqns
0655        maxeq=neq
0656        if (.not. EV%no_phot_mult
0657            !Photon multipoles
0658            EV%g_ix=basic_num_eqn
0659            if (EV%TightCoupling)
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 630

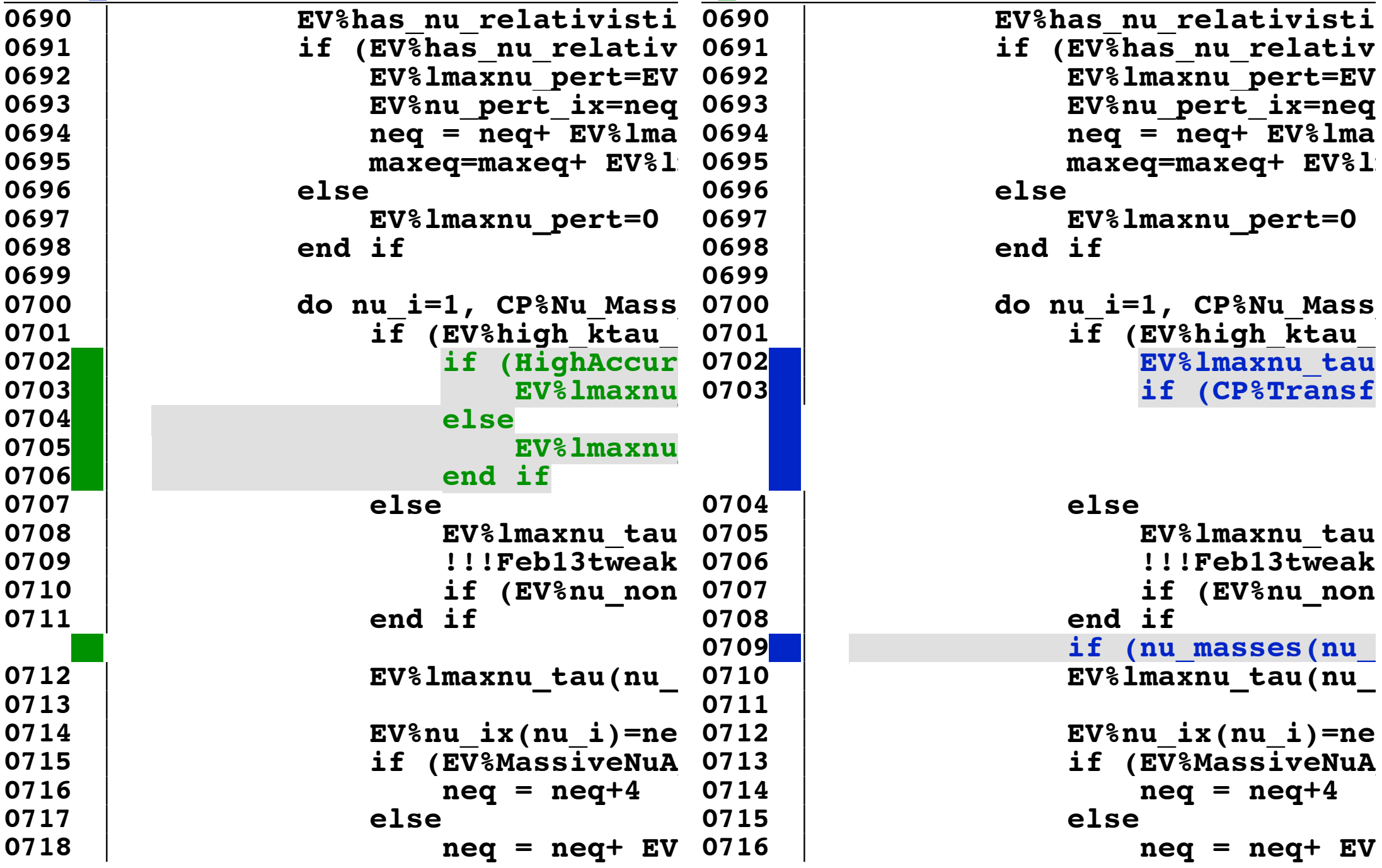
```
0630            nu_tau_notmassless(1,
0631            do j=2,nqmax
0632                !times when each
0633                time= time + Delt
0634                nu_tau_notmassles
0635            end do
0636
0637            a_nonrel =  2.5d0/nu_
0638            nu_tau_nonrelativisti
0639            a_massive =  17.d0/nu
0640            nu_tau_massive(nu_i)
0641        end do
0642
0643        end subroutine GaugeInter
0644
0645
0646        subroutine SetupScalarArr
0647        !Set up array indices aft
0648        use MassiveNu
0649        !Set the numer of equatio
0650        type(EvolutionVars) EV
0651        integer, intent(out), opt
0652        integer neq, maxeq, nu_i
0653
0654        neq=basic_num_eqns
0655        maxeq=neq
0656        if (.not. EV%no_phot_mult
0657            !Photon multipoles
0658            EV%g_ix=basic_num_eqn
0659            if (EV%TightCoupling)
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 660

```
0660 |                 neq=neq+2
0661 |             else
0662 |                 neq = neq+ (EV%lm
0663 |                 !Polarization mul
0664 |                 EV%polind = neq -
0665 |                 neq=neq + EV%lmax
0666 |             end if
0667 |         end if
0668 |         if (.not. EV%no_nu_multpo
0669 |             !Massless neutrino mu
0670 |             EV%r_ix= neq+1
0671 |             if (EV%high_ktau_neut
0672 |                 neq=neq + 3
0673 |             else
0674 |                 neq=neq + (EV%lma
0675 |             end if
0676 |         end if
0677 |         maxeq = maxeq +  (EV%lmax
0678 |
0679 |         !Dark energy
0680 |         if (.not. is_cosmological
0681 |             EV%w_ix = neq+1
0682 |             neq=neq+1 !ppf
0683 |             maxeq=maxeq+1
0684 |         else
0685 |             EV%w_ix=0
0686 |         end if
0687 |
0688 |         !Massive neutrinos
0689 |         if (CP%Num_Nu_massive /=
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 660

```
0660 |                 neq=neq+2
0661 |             else
0662 |                 neq = neq+ (EV%lm
0663 |                 !Polarization mul
0664 |                 EV%polind = neq -
0665 |                 neq=neq + EV%lmax
0666 |             end if
0667 |         end if
0668 |         if (.not. EV%no_nu_multpo
0669 |             !Massless neutrino mu
0670 |             EV%r_ix= neq+1
0671 |             if (EV%high_ktau_neut
0672 |                 neq=neq + 3
0673 |             else
0674 |                 neq=neq + (EV%lma
0675 |             end if
0676 |         end if
0677 |         maxeq = maxeq +  (EV%lmax
0678 |
0679 |         !Dark energy
0680 |         if (.not. is_cosmological
0681 |             EV%w_ix = neq+1
0682 |             neq=neq+1 !ppf
0683 |             maxeq=maxeq+1
0684 |         else
0685 |             EV%w_ix=0
0686 |         end if
0687 |
0688 |         !Massive neutrinos
0689 |         if (CP%Num_Nu_massive /=
```


/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 690
/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 690
0690 EV%has_nu_relativisti
0691 if (EV%has_nu_relativ
0692 EV%lmaxnu_pert=EV
0693 EV%nu_pert_ix=neq
0694 neq = neq+ EV%lma
0695 maxeq=maxeq+ EV%l
0696 else
0697 EV%lmaxnu_pert=0
0698 end if
0699
0700 do nu_i=1, CP%Nu_Mass
0701 if (EV%high_ktau_
0702 if (HighAccur
0703 EV%lmaxnu
0704 else
0705 EV%lmaxnu
0706 end if
0707 else
0708 EV%lmaxnu_tau
0709 !!!Feb13tweak
0710 if (EV%nu_non
0711 end if
0712 EV%lmaxnu_tau(nu_
0713
0714 EV%nu_ix(nu_i)=ne
0715 if (EV%MassiveNuA
0716 neq = neq+4
0717 else
0718 neq = neq+ EV
0690 EV%has_nu_relativisti
0691 if (EV%has_nu_relativ
0692 EV%lmaxnu_pert=EV
0693 EV%nu_pert_ix=neq
0694 neq = neq+ EV%lma
0695 maxeq=maxeq+ EV%l
0696 else
0697 EV%lmaxnu_pert=0
0698 end if
0699
0700 do nu_i=1, CP%Nu_Mass
0701 if (EV%high_ktau_
0702 EV%lmaxnu_tau
0703 if (CP%Transf
0704 else
0705 EV%lmaxnu_tau
0706 !!!Feb13tweak
0707 if (EV%nu_non
0708 end if
0709 if (nu_masses(nu_
0710 EV%lmaxnu_tau(nu_
0711
0712 EV%nu_ix(nu_i)=ne
0713 if (EV%MassiveNuA
0714 neq = neq+4
0715 else
0716 neq = neq+ EV

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 719

```
0719 |                endif
0720 |                maxeq = maxeq + n
0721 |            end do
0722 |        else
0723 |            EV%has_nu_relativisti
0724 |        end if
0725 |
0726 |        EV%ScalEqsToPropagate = n
0727 |        if (present(max_num_eqns)
0728 |            max_num_eqns=maxeq
0729 |        end if
0730 |
0731 |        end subroutine SetupScala
0732 |
0733 |        subroutine CopyScalarVari
0734 |        type(EvolutionVars) EV, E
0735 |        real(dl), intent(in) :: y
0736 |        real(dl), intent(out) ::
0737 |        integer lmax,i, nq
0738 |        integer nnueq,nu_i, ix_of
0739 |        real(dl) q, pert_scale
0740 |
0741 |        yout=0
0742 |        yout(1:basic_num_eqns) =
0743 |        if (.not. is_cosmological
0744 |            yout(EVout%w_ix)=y(EV
0745 |        end if
0746 |
0747 |        if (.not. EV%no_phot_mult
0748 |            if (EV%TightCoupling
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 717

```
0717 |                endif
0718 |                maxeq = maxeq + n
0719 |            end do
0720 |        else
0721 |            EV%has_nu_relativisti
0722 |        end if
0723 |
0724 |        EV%ScalEqsToPropagate = n
0725 |        if (present(max_num_eqns)
0726 |            max_num_eqns=maxeq
0727 |        end if
0728 |
0729 |        end subroutine SetupScala
0730 |
0731 |        subroutine CopyScalarVari
0732 |        type(EvolutionVars) EV, E
0733 |        real(dl), intent(in) :: y
0734 |        real(dl), intent(out) ::
0735 |        integer lmax,i, nq
0736 |        integer nnueq,nu_i, ix_of
0737 |        real(dl) q, pert_scale
0738 |
0739 |        yout=0
0740 |        yout(1:basic_num_eqns) =
0741 |        if (.not. is_cosmological
0742 |            yout(EVout%w_ix)=y(EV
0743 |        end if
0744 |
0745 |        if (.not. EV%no_phot_mult
0746 |            if (EV%TightCoupling
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 749

```
0749 |                 lmax=1
0750 |             else
0751 |                 lmax = min(EV%lma
0752 |             end if
0753 |             yout(EVout%g_ix:EVout
0754 |             if (.not. EV%TightCou
0755 |                 lmax = min(EV%lma
0756 |                 yout(EVout%polind
0757 |             end if
0758 |         end if
0759 |
0760 |         if (.not. EV%no_nu_multpo
0761 |             if (EV%high_ktau_neut
0762 |                 lmax=2
0763 |             else
0764 |                 lmax = min(EV%lma
0765 |             end if
0766 |             yout(EVout%r_ix:EVout
0767 |         end if
0768 |
0769 |         if (CP%Num_Nu_massive /=
0770 |             do nu_i=1,CP%Nu_mass_
0771 |                 ix_off=EV%nu_ix(n
0772 |                 ix_off2=EVOut%nu_
0773 |                 if (EV%MassiveNuA
0774 |                     nnueq=4
0775 |                     yout(ix_off2:
0776 |                 else if (.not. EV
0777 |                     lmax=min(EV%l
0778 |                     nq = min(EV%n
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 747

```
0747 |                 lmax=1
0748 |             else
0749 |                 lmax = min(EV%lma
0750 |             end if
0751 |             yout(EVout%g_ix:EVout
0752 |             if (.not. EV%TightCou
0753 |                 lmax = min(EV%lma
0754 |                 yout(EVout%polind
0755 |             end if
0756 |         end if
0757 |
0758 |         if (.not. EV%no_nu_multpo
0759 |             if (EV%high_ktau_neut
0760 |                 lmax=2
0761 |             else
0762 |                 lmax = min(EV%lma
0763 |             end if
0764 |             yout(EVout%r_ix:EVout
0765 |         end if
0766 |
0767 |         if (CP%Num_Nu_massive /=
0768 |             do nu_i=1,CP%Nu_mass_
0769 |                 ix_off=EV%nu_ix(n
0770 |                 ix_off2=EVOut%nu_
0771 |                 if (EV%MassiveNuA
0772 |                     nnueq=4
0773 |                     yout(ix_off2:
0774 |                 else if (.not. EV
0775 |                     lmax=min(EV%l
0776 |                     nq = min(EV%n
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 779

```
0779                        do i=1,nq
0780                            ind= ix_o
0781                            ind2=ix_o
0782                            yout(ind2
0783                        end do
0784                        do i=nq+1, EV
0785                            lmax = mi
0786                            ind2=ix_o
0787                            yout(ind2
0788
0789                            !Add lead
0790                            q=nu_q(i)
0791                            pert_scal
0792                            lmax = mi
0793                            yout(ind2
0794                            + y(EV%nu
0795                        end do
0796                    end if
0797                end do
0798
0799                if (EVOut%has_nu_rela
0800                    lmax = min(EVOut%
0801                    yout(EVout%nu_per
0802                end if
0803            end if
0804
0805            end subroutine CopyScalar
0806
0807
0808            subroutine SetupTensorArr
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 777

```
0777                        do i=1,nq
0778                            ind= ix_o
0779                            ind2=ix_o
0780                            yout(ind2
0781                        end do
0782                        do i=nq+1, EV
0783                            lmax = mi
0784                            ind2=ix_o
0785                            yout(ind2
0786
0787                            !Add lead
0788                            q=nu_q(i)
0789                            pert_scal
0790                            lmax = mi
0791                            yout(ind2
0792                                + y(E
0793                        end do
0794                    end if
0795                end do
0796
0797                if (EVOut%has_nu_rela
0798                    lmax = min(EVOut%
0799                    yout(EVout%nu_per
0800                end if
0801            end if
0802
0803            end subroutine CopyScalar
0804
0805
0806            subroutine SetupTensorArr
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 809

```
0809        type(EvolutionVars) EV
0810        integer nu_i, neq
0811        integer, optional, intent
0812        neq=3
0813        EV%g_ix = neq-1 !EV%g_ix+
0814        if (.not. EV%TensTightCou
0815            EV%E_ix = EV%g_ix + (
0816            EV%B_ix = EV%E_ix + (
0817            neq = neq+ (EV%lmaxt-
0818        end if
0819        if (present(maxeq)) then
0820            maxeq =3 + (EV%lmaxt-
0821        end if
0822        EV%r_ix = neq -1
0823        if (DoTensorNeutrinos) th
0824            neq = neq + EV%lmaxnr
0825            if (present(maxeq)) m
0826            if (CP%Num_Nu_massive
0827                do nu_i=1, CP%nu_
0828                    EV%EvolveTens
0829                    if (EV%Evolve
0830                        EV%nu_ix(
0831                        neq = neq
0832                        if (prese
0833                    end if
0834                end do
0835            end if
0836        end if
0837
0838        EV%TensEqsToPropagate = n
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 807

```
0807        type(EvolutionVars) EV
0808        integer nu_i, neq
0809        integer, optional, intent
0810        neq=3
0811        EV%g_ix = neq-1 !EV%g_ix+
0812        if (.not. EV%TensTightCou
0813            EV%E_ix = EV%g_ix + (
0814            EV%B_ix = EV%E_ix + (
0815            neq = neq+ (EV%lmaxt-
0816        end if
0817        if (present(maxeq)) then
0818            maxeq =3 + (EV%lmaxt-
0819        end if
0820        EV%r_ix = neq -1
0821        if (DoTensorNeutrinos) th
0822            neq = neq + EV%lmaxnr
0823            if (present(maxeq)) m
0824            if (CP%Num_Nu_massive
0825                do nu_i=1, CP%nu_
0826                    EV%EvolveTens
0827                    if (EV%Evolve
0828                        EV%nu_ix(
0829                        neq = neq
0830                        if (prese
0831                    end if
0832                end do
0833            end if
0834        end if
0835
0836        EV%TensEqsToPropagate = n
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 839

```
0839 |
0840 |       end  subroutine SetupTens
0841 |
0842 |       subroutine CopyTensorVari
0843 |       type(EvolutionVars) EV, E
0844 |       real(dl), intent(in) :: y
0845 |       real(dl), intent(out) ::
0846 |       integer lmaxpolt, lmaxt,
0847 |
0848 |       yout=0
0849 |       yout(1:3) = y(1:3)
0850 |       if (.not. EVOut%TensTight
0851 |           lmaxt = min(EVOut%lma
0852 |           yout(EVout%g_ix+2:EVo
0853 |           lmaxpolt = min(EV%lma
0854 |           yout(EVout%E_ix+2:EVo
0855 |           yout(EVout%B_ix+2:EVo
0856 |       end if
0857 |       if (DoTensorNeutrinos) th
0858 |           lmaxt=min(EV%lmaxnrt,
0859 |           yout(EVout%r_ix+2:EVo
0860 |           do nu_i =1, CP%nu_mas
0861 |               if (EV%EvolveTens
0862 |                   lmaxt=min(EV%
0863 |                   do i=1,nqmax
0864 |                       ind= EV%n
0865 |                       ind2=EVOu
0866 |                       yout(ind2
0867 |                   end do
0868 |               end if
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 837

```
0837 |
0838 |       end  subroutine SetupTens
0839 |
0840 |       subroutine CopyTensorVari
0841 |       type(EvolutionVars) EV, E
0842 |       real(dl), intent(in) :: y
0843 |       real(dl), intent(out) ::
0844 |       integer lmaxpolt, lmaxt,
0845 |
0846 |       yout=0
0847 |       yout(1:3) = y(1:3)
0848 |       if (.not. EVOut%TensTight
0849 |           lmaxt = min(EVOut%lma
0850 |           yout(EVout%g_ix+2:EVo
0851 |           lmaxpolt = min(EV%lma
0852 |           yout(EVout%E_ix+2:EVo
0853 |           yout(EVout%B_ix+2:EVo
0854 |       end if
0855 |       if (DoTensorNeutrinos) th
0856 |           lmaxt=min(EV%lmaxnrt,
0857 |           yout(EVout%r_ix+2:EVo
0858 |           do nu_i =1, CP%nu_mas
0859 |               if (EV%EvolveTens
0860 |                   lmaxt=min(EV%
0861 |                   do i=1,nqmax
0862 |                       ind= EV%n
0863 |                       ind2=EVOu
0864 |                       yout(ind2
0865 |                   end do
0866 |               end if
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 869

```
0869            end do
0870         end if
0871
0872         end subroutine CopyTensor
0873
0874         subroutine GetNumEqns(EV)
0875         use MassiveNu
0876         !Set the numer of equatio
0877         type(EvolutionVars) EV
0878         real(dl) scal, max_nu_mas
0879         integer nu_i,q_rel,j
0880
0881         if (CP%Num_Nu_massive ==
0882             EV%lmaxnu=0
0883             max_nu_mass=0
0884         else
0885             max_nu_mass = maxval(
0886             do nu_i = 1, CP%Nu_ma
0887                 !Start with momen
0888                 q_rel=0
0889                 do j=1, nqmax
0890                     !two differen
0891                     if (nu_q(j) >
0892                     q_rel = q_rel
0893                 end do
0894
0895                 if (q_rel>= nqmax
0896                     EV%nq(nu_i)=n
0897                 else
0898                     EV%nq(nu_i)=q
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 867

```
0867            end do
0868         end if
0869
0870         end subroutine CopyTensor
0871
0872         subroutine GetNumEqns(EV)
0873         use MassiveNu
0874         !Set the numer of equatio
0875         type(EvolutionVars) EV
0876         real(dl) scal, max_nu_mas
0877         integer nu_i,q_rel,j
0878
0879         if (CP%Num_Nu_massive ==
0880             EV%lmaxnu=0
0881             max_nu_mass=0
0882         else
0883             max_nu_mass = maxval(
0884             do nu_i = 1, CP%Nu_ma
0885                 !Start with momen
0886                 q_rel=0
0887                 do j=1, nqmax
0888                     !two differen
0889                     if (nu_q(j) >
0890                     q_rel = q_rel
0891                 end do
0892
0893                 if (q_rel>= nqmax
0894                     EV%nq(nu_i)=n
0895                 else
0896                     EV%nq(nu_i)=q
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 899

```
0899 |                end if
0900 |                !q_rel = nint(nu_
0901 |                !EV%nq(nu_i)=max(
0902 |                EV%nu_nonrelativi
0903 |            end do
0904 |
0905 |            EV%NuMethod = CP%Mass
0906 |            if (EV%NuMethod == Nu
0907 |            !l_max for massive ne
0908 |            if (CP%Transfer%high_
0909 |                EV%lmaxnu=nint(25
0910 |            else
0911 |                EV%lmaxnu=max(3,n
0912 |                if (max_nu_mass>7
0913 |            endif
0914 |        end if
0915 |
0916 |        if (CP%closed) then
0917 |            EV%FirstZerolForBeta
0918 |        else
0919 |            EV%FirstZerolForBeta=
0920 |        end if
0921 |
0922 |        EV%high_ktau_neutrino_app
0923 |        if (CP%WantScalars) then
0924 |            EV%TightCoupling=.tru
0925 |            EV%no_phot_multpoles
0926 |            EV%no_nu_multpoles =.
0927 |            EV%MassiveNuApprox=.f
0928 |
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 897

```
0897 |                end if
0898 |                !q_rel = nint(nu_
0899 |                !EV%nq(nu_i)=max(
0900 |                EV%nu_nonrelativi
0901 |            end do
0902 |
0903 |            EV%NuMethod = CP%Mass
0904 |            if (EV%NuMethod == Nu
0905 |            !l_max for massive ne
0906 |            if (CP%Transfer%high_
0907 |                EV%lmaxnu=nint(25
0908 |            else
0909 |                EV%lmaxnu=max(3,n
0910 |                if (max_nu_mass>7
0911 |            endif
0912 |        end if
0913 |
0914 |        if (CP%closed) then
0915 |            EV%FirstZerolForBeta
0916 |        else
0917 |            EV%FirstZerolForBeta=
0918 |        end if
0919 |
0920 |        EV%high_ktau_neutrino_app
0921 |        if (CP%WantScalars) then
0922 |            EV%TightCoupling=.tru
0923 |            EV%no_phot_multpoles
0924 |            EV%no_nu_multpoles =.
0925 |            EV%MassiveNuApprox=.f
0926 |
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 929

```
0929 |             if (HighAccuracyDefau
0930 |                 EV%lmaxg  = max(n
0931 |             else
0932 |                 EV%lmaxg  = max(n
0933 |             end if
0934 |             EV%lmaxnr = max(nint(
0935 |             if (max_nu_mass>700 .
0936 |
0937 |             EV%lmaxgpol = EV%lmax
0938 |             if (.not.CP%AccurateP
0939 |
0940 |             if (EV%q < 0.05) then
0941 |                 !Large scales nee
0942 |                 scal  = 1
0943 |                 if (CP%AccuratePo
0944 |                 EV%lmaxgpol=max(3
0945 |                 EV%lmaxnr=max(3,n
0946 |                 EV%lmaxg=max(3,ni
0947 |                 if (CP%AccurateRe
0948 |                     EV%lmaxg=EV%l
0949 |                     EV%lmaxgpol=E
0950 |                 end if
0951 |             end if
0952 |
0953 |             if (EV%TransferOnly)
0954 |                 EV%lmaxgpol = min
0955 |                 EV%lmaxg = min(EV
0956 |             end if
0957 |             if (CP%Transfer%high_
0958 |                 if (HighAccuracyD
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 927

```
0927 |             if (HighAccuracyDefau
0928 |                 EV%lmaxg  = max(n
0929 |             else
0930 |                 EV%lmaxg  = max(n
0931 |             end if
0932 |             EV%lmaxnr = max(nint(
0933 |             if (max_nu_mass>700 .
0934 |
0935 |             EV%lmaxgpol = EV%lmax
0936 |             if (.not.CP%AccurateP
0937 |
0938 |             if (EV%q < 0.05) then
0939 |                 !Large scales nee
0940 |                 scal  = 1
0941 |                 if (CP%AccuratePo
0942 |                 EV%lmaxgpol=max(3
0943 |                 EV%lmaxnr=max(3,n
0944 |                 EV%lmaxg=max(3,ni
0945 |                 if (CP%AccurateRe
0946 |                     EV%lmaxg=EV%l
0947 |                     EV%lmaxgpol=E
0948 |                 end if
0949 |             end if
0950 |
0951 |             if (EV%TransferOnly)
0952 |                 EV%lmaxgpol = min
0953 |                 EV%lmaxg = min(EV
0954 |             end if
0955 |             if (CP%Transfer%high_
0956 |                 if (HighAccuracyD
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 959

```
0959 |                     EV%lmaxnr=max
0960 |                 else
0961 |                     EV%lmaxnr=max
0962 |                 endif
0963 |                 if (EV%q > 0.04 .
0964 |                     EV%lmaxg=max(
0965 |                 end if
0966 |             end if
0967 |
0968 |             if (CP%closed) then
0969 |                 EV%lmaxnu=min(EV%
0970 |                 EV%lmaxnr=min(EV%
0971 |                 EV%lmaxg=min(EV%l
0972 |                 EV%lmaxgpol=min(E
0973 |             end if
0974 |
0975 |             EV%poltruncfac=real(E
0976 |             EV%MaxlNeeded=max(EV%
0977 |             if (EV%MaxlNeeded > m
0978 |             call SetupScalarArray
0979 |             if (CP%closed) EV%nva
0980 |             EV%lmaxt=0
0981 |         else
0982 |             EV%nvar=0
0983 |         end if
0984 |
0985 |         if (CP%WantTensors) then
0986 |             EV%TensTightCoupling
0987 |             EV%lmaxt=max(3,nint(8
0988 |             EV%lmaxpolt = max(3,n
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 957

```
0957 |                     EV%lmaxnr=max
0958 |                 else
0959 |                     EV%lmaxnr=max
0960 |                 endif
0961 |                 if (EV%q > 0.04 .
0962 |                     EV%lmaxg=max(
0963 |                 end if
0964 |             end if
0965 |
0966 |             if (CP%closed) then
0967 |                 EV%lmaxnu=min(EV%
0968 |                 EV%lmaxnr=min(EV%
0969 |                 EV%lmaxg=min(EV%l
0970 |                 EV%lmaxgpol=min(E
0971 |             end if
0972 |
0973 |             EV%poltruncfac=real(E
0974 |             EV%MaxlNeeded=max(EV%
0975 |             if (EV%MaxlNeeded > m
0976 |             call SetupScalarArray
0977 |             if (CP%closed) EV%nva
0978 |             EV%lmaxt=0
0979 |         else
0980 |             EV%nvar=0
0981 |         end if
0982 |
0983 |         if (CP%WantTensors) then
0984 |             EV%TensTightCoupling
0985 |             EV%lmaxt=max(3,nint(8
0986 |             EV%lmaxpolt = max(3,n
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 989

```
0989 |            ! if (EV%q < 1e-3) EV
0990 |            if (DoTensorNeutrinos
0991 |                EV%lmaxnrt=nint(6
0992 |                EV%lmaxnut=EV%lma
0993 |            else
0994 |                EV%lmaxnut=0
0995 |                EV%lmaxnrt=0
0996 |            end if
0997 |            if (CP%closed) then
0998 |                EV%lmaxt=min(EV%F
0999 |                EV%lmaxpolt=min(E
1000 |                EV%lmaxnrt=min(EV
1001 |                EV%lmaxnut=min(EV
1002 |            end if
1003 |            EV%MaxlNeededt=max(EV
1004 |            if (EV%MaxlNeededt >
1005 |            call SetupTensorArray
1006 |        else
1007 |            EV%nvart=0
1008 |        end if
1009 |
1010 |
1011 |        if (CP%WantVectors) then
1012 |            EV%lmaxv=max(10,nint(
1013 |            EV%lmaxpolv = max(5,n
1014 |
1015 |            EV%nvarv=(EV%lmaxv)+(
1016 |
1017 |            EV%lmaxnrv=nint(30*lA
1018 |
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 987

```
0987 |            ! if (EV%q < 1e-3) EV
0988 |            if (DoTensorNeutrinos
0989 |                EV%lmaxnrt=nint(6
0990 |                EV%lmaxnut=EV%lma
0991 |            else
0992 |                EV%lmaxnut=0
0993 |                EV%lmaxnrt=0
0994 |            end if
0995 |            if (CP%closed) then
0996 |                EV%lmaxt=min(EV%F
0997 |                EV%lmaxpolt=min(E
0998 |                EV%lmaxnrt=min(EV
0999 |                EV%lmaxnut=min(EV
1000 |            end if
1001 |            EV%MaxlNeededt=max(EV
1002 |            if (EV%MaxlNeededt >
1003 |            call SetupTensorArray
1004 |        else
1005 |            EV%nvart=0
1006 |        end if
1007 |
1008 |
1009 |        if (CP%WantVectors) then
1010 |            EV%lmaxv=max(10,nint(
1011 |            EV%lmaxpolv = max(5,n
1012 |
1013 |            EV%nvarv=(EV%lmaxv)+(
1014 |
1015 |            EV%lmaxnrv=nint(30*lA
1016 |
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1019

```
1019 |           EV%nvarv=EV%nvarv+EV%
1020 |           if (CP%Num_Nu_massive
1021 |               stop 'massive neu
1022 |           end if
1023 |       else
1024 |           EV%nvarv=0
1025 |       end if
1026 |
1027 |       end subroutine GetNumEqns
1028 |
1029 |       !ccccccccccccccccccccccc
1030 |       subroutine SwitchToMassiv
1031 |       !When the neutrinos are n
1032 |       !energy-integrated hierar
1033 |       type(EvolutionVars) EV, E
1034 |       integer, intent(in) :: nu
1035 |
1036 |       real(dl) a,a2,pnu,clxnu,d
1037 |       real(dl) qnu
1038 |       real(dl) y(EV%nvar), yout
1039 |
1040 |       a=y(1)
1041 |       a2=a*a
1042 |       EVout=EV
1043 |       EVout%MassiveNuApprox(nu_
1044 |       call SetupScalarArrayIndi
1045 |       call CopyScalarVariableAr
1046 |
1047 |       !Get density and pressure
1048 |       call Nu_background(a*nu_m
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1017

```
1017 |           EV%nvarv=EV%nvarv+EV%
1018 |           if (CP%Num_Nu_massive
1019 |               stop 'massive neu
1020 |           end if
1021 |       else
1022 |           EV%nvarv=0
1023 |       end if
1024 |
1025 |       end subroutine GetNumEqns
1026 |
1027 |       !ccccccccccccccccccccccc
1028 |       subroutine SwitchToMassiv
1029 |       !When the neutrinos are n
1030 |       !energy-integrated hierar
1031 |       type(EvolutionVars) EV, E
1032 |       integer, intent(in) :: nu
1033 |
1034 |       real(dl) a,a2,pnu,clxnu,d
1035 |       real(dl) qnu
1036 |       real(dl) y(EV%nvar), yout
1037 |
1038 |       a=y(1)
1039 |       a2=a*a
1040 |       EVout=EV
1041 |       EVout%MassiveNuApprox(nu_
1042 |       call SetupScalarArrayIndi
1043 |       call CopyScalarVariableAr
1044 |
1045 |       !Get density and pressure
1046 |       call Nu_background(a*nu_m
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1049

```
1049
1050        !Integrate over q
1051        call Nu_Integrate_L012(EV
1052        !clxnu_here  = rhonu*clxn
1053        dpnu=dpnu/rhonu
1054        qnu=qnu/rhonu
1055        clxnu = clxnu/rhonu
1056        pinu=pinu/rhonu
1057
1058        yout(EVout%nu_ix(nu_i))=c
1059        yout(EVout%nu_ix(nu_i)+1)
1060        yout(EVout%nu_ix(nu_i)+2)
1061        yout(EVout%nu_ix(nu_i)+3)
1062
1063        call Nu_Intvsq(EV,y, a, n
1064        !Analytic solution for hi
1065        EVout%G11(nu_i)=EVout%G11
1066        EVout%G30(nu_i)=EVout%G30
1067
1068        EV=EVout
1069        y=yout
1070
1071        end subroutine SwitchToMa
1072
1073        subroutine MassiveNuVarsO
1074        implicit none
1075        type(EvolutionVars) EV
1076        real(dl) :: y(EV%nvar), y
1077        real(dl), optional :: grh
1078        !grho = a^2 kappa rho
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1047

```
1047
1048        !Integrate over q
1049        call Nu_Integrate_L012(EV
1050        !clxnu_here  = rhonu*clxn
1051        dpnu=dpnu/rhonu
1052        qnu=qnu/rhonu
1053        clxnu = clxnu/rhonu
1054        pinu=pinu/rhonu
1055
1056        yout(EVout%nu_ix(nu_i))=c
1057        yout(EVout%nu_ix(nu_i)+1)
1058        yout(EVout%nu_ix(nu_i)+2)
1059        yout(EVout%nu_ix(nu_i)+3)
1060
1061        call Nu_Intvsq(EV,y, a, n
1062        !Analytic solution for hi
1063        EVout%G11(nu_i)=EVout%G11
1064        EVout%G30(nu_i)=EVout%G30
1065
1066        EV=EVout
1067        y=yout
1068
1069        end subroutine SwitchToMa
1070
1071        subroutine MassiveNuVarsO
1072        implicit none
1073        type(EvolutionVars) EV
1074        real(dl) :: y(EV%nvar), y
1075        real(dl), optional :: grh
1076        !grho = a^2 kappa rho
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1079

```
1079 |         !gpres = a^2 kappa p
1080 |         !dgrho = a^2 kappa \delta
1081 |         !dgp =  a^2 kappa \delta
1082 |         !dgq = a^2 kappa q (heat
1083 |         !dgpi = a^2 kappa pi (ani
1084 |         !dgpi_diff = a^2 kappa (3
1085 |
1086 |         integer nu_i
1087 |         real(dl) pinudot,grhormas
1088 |         real(dl) adotoa, grhonu_t
1089 |         real(dl) clxnu, qnu, pinu
1090 |         real(dl) dtauda
1091 |
1092 |         grhonu=0
1093 |         dgrhonu=0
1094 |         do nu_i = 1, CP%Nu_mass_e
1095 |             grhormass_t=grhormass
1096 |
1097 |             !Get density and pres
1098 |             call Nu_background(a*
1099 |
1100 |             if (EV%MassiveNuAppro
1101 |                 clxnu=y(EV%nu_ix(
1102 |                 !dpnu = y(EV%iq0+
1103 |                 qnu=y(EV%nu_ix(nu
1104 |                 pinu=y(EV%nu_ix(n
1105 |                 pinudot=yprime(EV
1106 |             else
1107 |                 !Integrate over q
1108 |                 call Nu_Integrate
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1077

```
1077 |         !gpres = a^2 kappa p
1078 |         !dgrho = a^2 kappa \delta
1079 |         !dgp =  a^2 kappa \delta
1080 |         !dgq = a^2 kappa q (heat
1081 |         !dgpi = a^2 kappa pi (ani
1082 |         !dgpi_diff = a^2 kappa (3
1083 |
1084 |         integer nu_i
1085 |         real(dl) pinudot,grhormas
1086 |         real(dl) adotoa, grhonu_t
1087 |         real(dl) clxnu, qnu, pinu
1088 |         real(dl) dtauda
1089 |
1090 |         grhonu=0
1091 |         dgrhonu=0
1092 |         do nu_i = 1, CP%Nu_mass_e
1093 |             grhormass_t=grhormass
1094 |
1095 |             !Get density and pres
1096 |             call Nu_background(a*
1097 |
1098 |             if (EV%MassiveNuAppro
1099 |                 clxnu=y(EV%nu_ix(
1100 |                 !dpnu = y(EV%iq0+
1101 |                 qnu=y(EV%nu_ix(nu
1102 |                 pinu=y(EV%nu_ix(n
1103 |                 pinudot=yprime(EV
1104 |             else
1105 |                 !Integrate over q
1106 |                 call Nu_Integrate
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1109

```
1109 |                 !clxnu_here  = rh
1110 |                 !dpnu=dpnu/rhonu
1111 |                 qnu=qnu/rhonu
1112 |                 clxnu = clxnu/rho
1113 |                 pinu=pinu/rhonu
1114 |                 adotoa = 1/(a*dta
1115 |                 rhonudot = Nu_drh
1116 |
1117 |                 call Nu_pinudot(E
1118 |                 pinudot=pinudot/r
1119 |             endif
1120 |
1121 |             grhonu_t=grhormass_t*
1122 |             gpnu_t=grhormass_t*pn
1123 |
1124 |             grhonu = grhonu  + gr
1125 |             if (present(gpres)) g
1126 |
1127 |             dgrhonu= dgrhonu + gr
1128 |             if (present(dgq)) dgq
1129 |             if (present(dgpi)) dg
1130 |             if (present(gdpi_diff
1131 |             if (present(pidot_sum
1132 |         end do
1133 |         if (present(grho)) grho =
1134 |         if (present(dgrho)) dgrho
1135 |         if (present(clxnu_all)) c
1136 |
1137 |         end subroutine MassiveNuV
1138 |
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1107

```
1107 |                 !clxnu_here  = rh
1108 |                 !dpnu=dpnu/rhonu
1109 |                 qnu=qnu/rhonu
1110 |                 clxnu = clxnu/rho
1111 |                 pinu=pinu/rhonu
1112 |                 adotoa = 1/(a*dta
1113 |                 rhonudot = Nu_drh
1114 |
1115 |                 call Nu_pinudot(E
1116 |                 pinudot=pinudot/r
1117 |             endif
1118 |
1119 |             grhonu_t=grhormass_t*
1120 |             gpnu_t=grhormass_t*pn
1121 |
1122 |             grhonu = grhonu  + gr
1123 |             if (present(gpres)) g
1124 |
1125 |             dgrhonu= dgrhonu + gr
1126 |             if (present(dgq)) dgq
1127 |             if (present(dgpi)) dg
1128 |             if (present(gdpi_diff
1129 |             if (present(pidot_sum
1130 |         end do
1131 |         if (present(grho)) grho =
1132 |         if (present(dgrho)) dgrho
1133 |         if (present(clxnu_all)) c
1134 |
1135 |         end subroutine MassiveNuV
1136 |
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1139

```
1139        subroutine Nu_Integrate_L
1140        type(EvolutionVars) EV
1141        !  Compute the perturbati
1142        !  of one eigenstate of m
1143        !  density of one eigenst
1144        !  momentum.
1145        integer, intent(in) :: nu
1146        real(dl), intent(in) :: a
1147        real(dl), intent(OUT) ::
1148        real(dl), optional, inten
1149        real(dl) tmp, am, aq,v, p
1150        integer iq, ind
1151
1152        !  q is the comoving mome
1153
1154        drhonu=0
1155        fnu=0
1156        if (present(dpnu)) then
1157            dpnu=0
1158            pinu=0
1159        end if
1160        am=a*nu_masses(nu_i)
1161        ind=EV%nu_ix(nu_i)
1162        do iq=1,EV%nq(nu_i)
1163            aq=am/nu_q(iq)
1164            v=1._dl/sqrt(1._dl+aq
1165            drhonu=drhonu+ nu_int
1166            fnu=fnu+nu_int_kernel
1167            if (present(dpnu)) th
1168                dpnu=dpnu+  nu_in
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1137

```
1137        subroutine Nu_Integrate_L
1138        type(EvolutionVars) EV
1139        !  Compute the perturbati
1140        !  of one eigenstate of m
1141        !  density of one eigenst
1142        !  momentum.
1143        integer, intent(in) :: nu
1144        real(dl), intent(in) :: a
1145        real(dl), intent(OUT) ::
1146        real(dl), optional, inten
1147        real(dl) tmp, am, aq,v, p
1148        integer iq, ind
1149
1150        !  q is the comoving mome
1151
1152        drhonu=0
1153        fnu=0
1154        if (present(dpnu)) then
1155            dpnu=0
1156            pinu=0
1157        end if
1158        am=a*nu_masses(nu_i)
1159        ind=EV%nu_ix(nu_i)
1160        do iq=1,EV%nq(nu_i)
1161            aq=am/nu_q(iq)
1162            v=1._dl/sqrt(1._dl+aq
1163            drhonu=drhonu+ nu_int
1164            fnu=fnu+nu_int_kernel
1165            if (present(dpnu)) th
1166                dpnu=dpnu+  nu_in
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1169

```
1169 |                 pinu=pinu+ nu_int
1170 |             end if
1171 |             ind=ind+EV%lmaxnu_tau
1172 |         end do
1173 |         ind = EV%nu_pert_ix
1174 |         do iq=EV%nq(nu_i)+1,nqmax
1175 |             !Get the rest from pe
1176 |             aq=am/nu_q(iq)
1177 |             v=1._dl/sqrt(1._dl+aq
1178 |             pert_scale=(nu_masses
1179 |             tmp = nu_int_kernel(i
1180 |             drhonu=drhonu+ tmp/v
1181 |             fnu=fnu+nu_int_kernel
1182 |             if (present(dpnu)) th
1183 |                 dpnu=dpnu+ tmp*v
1184 |                 pinu = pinu+ nu_i
1185 |             end if
1186 |         end do
1187 |
1188 |         if (present(dpnu)) then
1189 |             dpnu = dpnu/3
1190 |         end if
1191 |
1192 |         end subroutine Nu_Integra
1193 |
1194 |         subroutine Nu_pinudot(EV,
1195 |         type(EvolutionVars) EV
1196 |         integer, intent(in) :: nu
1197 |         real(dl), intent(in) :: a
1198 |
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1167

```
1167 |                 pinu=pinu+ nu_int
1168 |             end if
1169 |             ind=ind+EV%lmaxnu_tau
1170 |         end do
1171 |         ind = EV%nu_pert_ix
1172 |         do iq=EV%nq(nu_i)+1,nqmax
1173 |             !Get the rest from pe
1174 |             aq=am/nu_q(iq)
1175 |             v=1._dl/sqrt(1._dl+aq
1176 |             pert_scale=(nu_masses
1177 |             tmp = nu_int_kernel(i
1178 |             drhonu=drhonu+ tmp/v
1179 |             fnu=fnu+nu_int_kernel
1180 |             if (present(dpnu)) th
1181 |                 dpnu=dpnu+ tmp*v
1182 |                 pinu = pinu+ nu_i
1183 |             end if
1184 |         end do
1185 |
1186 |         if (present(dpnu)) then
1187 |             dpnu = dpnu/3
1188 |         end if
1189 |
1190 |         end subroutine Nu_Integra
1191 |
1192 |         subroutine Nu_pinudot(EV,
1193 |         type(EvolutionVars) EV
1194 |         integer, intent(in) :: nu
1195 |         real(dl), intent(in) :: a
1196 |
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1199

```
1199 |         !  Compute the time deriv
1200 |         !  and the shear perturba
1201 |         real(dl) pinudot
1202 |         real(dl) aq,q,v,aqdot,vdo
1203 |         real(dl) psi2,psi2dot
1204 |         real(dl) am, pert_scale
1205 |         integer iq,ind
1206 |
1207 |         !  q is the comoving mome
1208 |         pinudot=0._dl
1209 |         ind=EV%nu_ix(nu_i)+2
1210 |         am=a*nu_masses(nu_i)
1211 |         do iq=1,EV%nq(nu_i)
1212 |             q=nu_q(iq)
1213 |             aq=am/q
1214 |             aqdot=aq*adotoa
1215 |             v=1._dl/sqrt(1._dl+aq
1216 |             vdot=-aq*aqdot/(1._dl
1217 |             pinudot=pinudot+nu_in
1218 |             ind=ind+EV%lmaxnu_tau
1219 |         end do
1220 |         ind = EV%nu_pert_ix+2
1221 |         do iq=EV%nq(nu_i)+1,nqmax
1222 |             q=nu_q(iq)
1223 |             aq=am/q
1224 |             aqdot=aq*adotoa
1225 |             pert_scale=(nu_masses
1226 |             v=1._dl/sqrt(1._dl+aq
1227 |             vdot=-aq*aqdot/(1._dl
1228 |             psi2dot=ydot(EV%r_ix+
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1197

```
1197 |         !  Compute the time deriv
1198 |         !  and the shear perturba
1199 |         real(dl) pinudot
1200 |         real(dl) aq,q,v,aqdot,vdo
1201 |         real(dl) psi2,psi2dot
1202 |         real(dl) am, pert_scale
1203 |         integer iq,ind
1204 |
1205 |         !  q is the comoving mome
1206 |         pinudot=0._dl
1207 |         ind=EV%nu_ix(nu_i)+2
1208 |         am=a*nu_masses(nu_i)
1209 |         do iq=1,EV%nq(nu_i)
1210 |             q=nu_q(iq)
1211 |             aq=am/q
1212 |             aqdot=aq*adotoa
1213 |             v=1._dl/sqrt(1._dl+aq
1214 |             vdot=-aq*aqdot/(1._dl
1215 |             pinudot=pinudot+nu_in
1216 |             ind=ind+EV%lmaxnu_tau
1217 |         end do
1218 |         ind = EV%nu_pert_ix+2
1219 |         do iq=EV%nq(nu_i)+1,nqmax
1220 |             q=nu_q(iq)
1221 |             aq=am/q
1222 |             aqdot=aq*adotoa
1223 |             pert_scale=(nu_masses
1224 |             v=1._dl/sqrt(1._dl+aq
1225 |             vdot=-aq*aqdot/(1._dl
1226 |             psi2dot=ydot(EV%r_ix+
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1229

```
1229          psi2=y(EV%r_ix+2)  +
1230          pinudot=pinudot+nu_in
1231      end do
1232
1233      end subroutine Nu_pinudot
1234
1235      !ccccccccccccccccccccccc
1236      function Nu_pi(EV, y, a,
1237      type(EvolutionVars) EV
1238      integer, intent(in) :: nu
1239      real(dl), intent(in) :: a
1240      real(dl) :: am
1241      real(dl) pinu,q,aq,v
1242      integer iq, ind
1243
1244      if (EV%nq(nu_i)/=nqmax) s
1245      pinu=0
1246      ind=EV%nu_ix(nu_i)+2
1247      am=a*nu_masses(nu_i)
1248      do iq=1, EV%nq(nu_i)
1249          q=nu_q(iq)
1250          aq=am/q
1251          v=1._dl/sqrt(1._dl+aq
1252          pinu=pinu+nu_int_kern
1253          ind =ind+EV%lmaxnut+1
1254      end do
1255
1256      end function Nu_pi
1257
1258      !ccccccccccccccccccccccc
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1227

```
1227          psi2=y(EV%r_ix+2)  +
1228          pinudot=pinudot+nu_in
1229      end do
1230
1231      end subroutine Nu_pinudot
1232
1233      !ccccccccccccccccccccccc
1234      function Nu_pi(EV, y, a,
1235      type(EvolutionVars) EV
1236      integer, intent(in) :: nu
1237      real(dl), intent(in) :: a
1238      real(dl) :: am
1239      real(dl) pinu,q,aq,v
1240      integer iq, ind
1241
1242      if (EV%nq(nu_i)/=nqmax) s
1243      pinu=0
1244      ind=EV%nu_ix(nu_i)+2
1245      am=a*nu_masses(nu_i)
1246      do iq=1, EV%nq(nu_i)
1247          q=nu_q(iq)
1248          aq=am/q
1249          v=1._dl/sqrt(1._dl+aq
1250          pinu=pinu+nu_int_kern
1251          ind =ind+EV%lmaxnut+1
1252      end do
1253
1254      end function Nu_pi
1255
1256      !ccccccccccccccccccccccc
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1259

```
1259        subroutine Nu_Intvsq(EV,y
1260        type(EvolutionVars) EV
1261        integer, intent(in) :: nu
1262        real(dl), intent(in) :: a
1263        real(dl), intent(OUT) ::
1264
1265        !  Compute the third orde
1266        !by integrating over mome
1267        real(dl) aq,q,v, am
1268        integer iq, ind
1269
1270        !  q is the comoving mome
1271        am=a*nu_masses(nu_i)
1272        ind=EV%nu_ix(nu_i)
1273        G11=0._dl
1274        G30=0._dl
1275        if (EV%nq(nu_i)/=nqmax) s
1276        do iq=1, EV%nq(nu_i)
1277            q=nu_q(iq)
1278            aq=am/q
1279            v=1._dl/sqrt(1._dl+aq
1280            G11=G11+nu_int_kernel
1281            if (EV%lmaxnu_tau(nu_
1282                G30=G30+nu_int_ke
1283            end if
1284            ind = ind+EV%lmaxnu_t
1285        end do
1286
1287        end subroutine Nu_Intvsq
1288
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1257

```
1257        subroutine Nu_Intvsq(EV,y
1258        type(EvolutionVars) EV
1259        integer, intent(in) :: nu
1260        real(dl), intent(in) :: a
1261        real(dl), intent(OUT) ::
1262
1263        !  Compute the third orde
1264        !by integrating over mome
1265        real(dl) aq,q,v, am
1266        integer iq, ind
1267
1268        !  q is the comoving mome
1269        am=a*nu_masses(nu_i)
1270        ind=EV%nu_ix(nu_i)
1271        G11=0._dl
1272        G30=0._dl
1273        if (EV%nq(nu_i)/=nqmax) s
1274        do iq=1, EV%nq(nu_i)
1275            q=nu_q(iq)
1276            aq=am/q
1277            v=1._dl/sqrt(1._dl+aq
1278            G11=G11+nu_int_kernel
1279            if (EV%lmaxnu_tau(nu_
1280                G30=G30+nu_int_ke
1281            end if
1282            ind = ind+EV%lmaxnu_t
1283        end do
1284
1285        end subroutine Nu_Intvsq
1286
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1289

```
1289 |
1290 |       subroutine MassiveNuVars(
1291 |       implicit none
1292 |       type(EvolutionVars) EV
1293 |       real(dl) :: y(EV%nvar), a
1294 |       real(dl), intent(out), op
1295 |       !grho = a^2 kappa rho
1296 |       !gpres = a^2 kappa p
1297 |       !dgrho = a^2 kappa \delta
1298 |       !dgp =  a^2 kappa \delta
1299 |       !dgq = a^2 kappa q (heat
1300 |       integer nu_i
1301 |       real(dl) grhormass_t, rho
1302 |
1303 |       do nu_i = 1, CP%Nu_mass_e
1304 |           grhormass_t=grhormass
1305 |
1306 |           !Get density and pres
1307 |           call Nu_background(a*
1308 |
1309 |           if (EV%MassiveNuAppro
1310 |               clxnu=y(EV%nu_ix(
1311 |               qnu=y(EV%nu_ix(nu
1312 |           else
1313 |               !Integrate over q
1314 |               call Nu_Integrate
1315 |               !clxnu_here  = rh
1316 |               qnu=qnu/rhonu
1317 |               clxnu = clxnu/rho
1318 |           endif
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1287

```
1287 |
1288 |       subroutine MassiveNuVars(
1289 |       implicit none
1290 |       type(EvolutionVars) EV
1291 |       real(dl) :: y(EV%nvar), a
1292 |       real(dl), intent(out), op
1293 |       !grho = a^2 kappa rho
1294 |       !gpres = a^2 kappa p
1295 |       !dgrho = a^2 kappa \delta
1296 |       !dgp =  a^2 kappa \delta
1297 |       !dgq = a^2 kappa q (heat
1298 |       integer nu_i
1299 |       real(dl) grhormass_t, rho
1300 |
1301 |       do nu_i = 1, CP%Nu_mass_e
1302 |           grhormass_t=grhormass
1303 |
1304 |           !Get density and pres
1305 |           call Nu_background(a*
1306 |
1307 |           if (EV%MassiveNuAppro
1308 |               clxnu=y(EV%nu_ix(
1309 |               qnu=y(EV%nu_ix(nu
1310 |           else
1311 |               !Integrate over q
1312 |               call Nu_Integrate
1313 |               !clxnu_here  = rh
1314 |               qnu=qnu/rhonu
1315 |               clxnu = clxnu/rho
1316 |           endif
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1319

```
1319
1320            grhonu_t=grhormass_t*
1321            gpnu_t=grhormass_t*pn
1322
1323            grho = grho  + grhonu
1324            gpres= gpres + gpnu_t
1325            dgrho= dgrho + grhonu
1326            dgq  = dgq   + grhonu
1327
1328            if (present(wnu_arr))
1329                wnu_arr(nu_i) =pn
1330            end if
1331        end do
1332
1333        end subroutine MassiveNuV
1334
1335        !cccccccccccccccccccccccc
1336        subroutine output(EV,y, j
1337        use ThermoData
1338        use lvalues
1339        use ModelData
1340        implicit none
1341        integer j
1342        type(EvolutionVars) EV
1343        real(dl), target :: y(EV%
1344        real(dl), dimension(:),po
1345
1346        real(dl) dgq,grhob_t,grho
1347        real(dl) qgdot,pigdot,pir
1348        real(dl) a,a2,dz,z,clxc,c
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1317

```
1317
1318            grhonu_t=grhormass_t*
1319            gpnu_t=grhormass_t*pn
1320
1321            grho = grho  + grhonu
1322            gpres= gpres + gpnu_t
1323            dgrho= dgrho + grhonu
1324            dgq  = dgq   + grhonu
1325
1326            if (present(wnu_arr))
1327                wnu_arr(nu_i) =pn
1328            end if
1329        end do
1330
1331        end subroutine MassiveNuV
1332
1333        !cccccccccccccccccccccccc
1334        subroutine output(EV,y, t
1335        use ThermoData
1336        use lvalues
1337        use ModelData
1338        implicit none
1339        type(EvolutionVars) EV
1340        real(dl), target :: y(EV%
1341        real(dl), dimension(:),po
1342
1343        real(dl) dgq,grhob_t,grho
1344        real(dl) qgdot,pigdot,pir
1345        real(dl) a,a2,dz,z,clxc,c
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1349

```
1349
1350        real(dl) tau,x,divfac
1351        real(dl) dgpi_diff, pidot
1352        real(dl), target :: pol(3
1353        !dgpi_diff = sum (3*p_nu
1354
1355        real(dl) k,k2  ,adotoa, g
1356        real(dl)  diff_rhopi, oct
1357        real(dl) sources(CTransSc
1358        !        real(dl) t4,t92
1359        real(dl) ISW
1360        real(dl) w_eff
1361        real(dl) hdotoh,ppiedot
1362
1363        yprime = 0
1364        call derivs(EV,EV%ScalEqs
1365
1366        if (EV%TightCoupling .or.
1367            pol=0
1368            polprime=0
1369            ypolprime => polprime
1370            ypol => pol
1371        else
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1346

```
1346
1347        real(dl) tau,x,divfac
1348        real(dl) dgpi_diff, pidot
1349        real(dl), target :: pol(3
1350        !dgpi_diff = sum (3*p_nu
1351
1352        real(dl) k,k2  ,adotoa, g
1353        real(dl)  diff_rhopi, oct
1354        real(dl) sources(CTransSc
1355        !        real(dl) t4,t92
1356        real(dl) ISW
1357        real(dl) w_eff
1358        real(dl) hdotoh,ppiedot
1359        integer, intent(in) :: no
1360        real(dl) opacity, dopacit
1361
1362
1363        call IonizationFunctionsA
1364            visibility, dvisibili
1365
1366
1367        yprime = 0
1368        call derivs(EV,EV%ScalEqs
1369
1370        if (EV%TightCoupling .or.
1371            pol=0
1372            polprime=0
1373            ypolprime => polprime
1374            ypol => pol
1375        else
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1372

```
1372 |            ypolprime => yprime(E
1373 |            ypol => y(EV%polind+1
1374 |        end if
1375 |
1376 |        k=EV%k_buf
1377 |        k2=EV%k2_buf
1378 |
1379 |        a   =y(1)
1380 |        a2  =a*a
1381 |        etak=y(2)
1382 |        clxc=y(3)
1383 |        clxb=y(4)
1384 |        vb  =y(5)
1385 |        vbdot =yprime(5)
1386 |
1387 |        !  Compute expansion rate
1388 |
1389 |        grhob_t=grhob/a
1390 |        grhoc_t=grhoc/a
1391 |        grhor_t=grhornomass/a2
1392 |        grhog_t=grhog/a2
1393 |
1394 |        !  8*pi*a*a*SUM[rho_i*clx
1395 |        dgrho=grhob_t*clxb+grhoc_
1396 |
1397 |        !  8*pi*a*a*SUM[(rho_i+p_
1398 |        dgq=grhob_t*vb
1399 |
1400 |        if (is_cosmological_const
1401 |            w_eff = -1_dl
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1376

```
1376 |            ypolprime => yprime(E
1377 |            ypol => y(EV%polind+1
1378 |        end if
1379 |
1380 |        k=EV%k_buf
1381 |        k2=EV%k2_buf
1382 |
1383 |        a   =y(1)
1384 |        a2  =a*a
1385 |        etak=y(2)
1386 |        clxc=y(3)
1387 |        clxb=y(4)
1388 |        vb  =y(5)
1389 |        vbdot =yprime(5)
1390 |
1391 |        !  Compute expansion rate
1392 |
1393 |        grhob_t=grhob/a
1394 |        grhoc_t=grhoc/a
1395 |        grhor_t=grhornomass/a2
1396 |        grhog_t=grhog/a2
1397 |
1398 |        !  8*pi*a*a*SUM[rho_i*clx
1399 |        dgrho=grhob_t*clxb+grhoc_
1400 |
1401 |        !  8*pi*a*a*SUM[(rho_i+p_
1402 |        dgq=grhob_t*vb
1403 |
1404 |        if (is_cosmological_const
1405 |            w_eff = -1_dl
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1402

```
1402 |             grhov_t=grhov*a2
1403 |         else
1404 |             !ppf
1405 |             w_eff=w_de(a)   !effe
1406 |             grhov_t=grho_de(a)/a2
1407 |             dgrho=dgrho+EV%dgrho_
1408 |             dgq=dgq+EV%dgq_e_ppf
1409 |         end if
1410 |         grho=grhob_t+grhoc_t+grho
1411 |         gpres=(grhog_t+grhor_t)/3
1412 |
1413 |
1414 |         dgpi= 0
1415 |         dgpi_diff = 0
1416 |         pidot_sum = 0
1417 |
1418 |         if (CP%Num_Nu_Massive /=
1419 |             call MassiveNuVarsOut
1420 |         end if
1421 |
1422 |         adotoa=sqrt((grho+grhok)/
1423 |
1424 |         if (EV%no_nu_multpoles) t
1425 |             z=(0.5_dl*dgrho/k + e
1426 |             dz= -adotoa*z - 0.5_d
1427 |             clxr=-4*dz/k
1428 |             qr=-4._dl/3*z
1429 |             pir=0
1430 |             pirdot=0
1431 |         else
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1406

```
1406 |             grhov_t=grhov*a2
1407 |         else
1408 |             !ppf
1409 |             w_eff=w_de(a)   !effe
1410 |             grhov_t=grho_de(a)/a2
1411 |             dgrho=dgrho+EV%dgrho_
1412 |             dgq=dgq+EV%dgq_e_ppf
1413 |         end if
1414 |         grho=grhob_t+grhoc_t+grho
1415 |         gpres=(grhog_t+grhor_t)/3
1416 |
1417 |
1418 |         dgpi= 0
1419 |         dgpi_diff = 0
1420 |         pidot_sum = 0
1421 |
1422 |         if (CP%Num_Nu_Massive /=
1423 |             call MassiveNuVarsOut
1424 |         end if
1425 |
1426 |         adotoa=sqrt((grho+grhok)/
1427 |
1428 |         if (EV%no_nu_multpoles) t
1429 |             z=(0.5_dl*dgrho/k + e
1430 |             dz= -adotoa*z - 0.5_d
1431 |             clxr=-4*dz/k
1432 |             qr=-4._dl/3*z
1433 |             pir=0
1434 |             pirdot=0
1435 |         else
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1432

```
1432            clxr=y(EV%r_ix)
1433            qr  =y(EV%r_ix+1)
1434            pir =y(EV%r_ix+2)
1435            pirdot=yprime(EV%r_ix
1436        end if
1437
1438        if (EV%no_phot_multpoles)
1439            z=(0.5_dl*dgrho/k + e
1440            dz= -adotoa*z - 0.5_d
1441            clxg=-4*dz/k -4/k*opa
1442            qg=-4._dl/3*z
1443            pig=0
1444            pigdot=0
1445            octg=0
1446            octgprime=0
1447            qgdot = -4*dz/3
1448        else
1449            if (EV%TightCoupling)
1450                pig = EV%pig
1451                !pigdot=EV%pigdot
1452                if (second_order_
1453                    octg = (3._dl
1454                    ypol(2) = EV%
1455                    ypol(3) = (3.
1456                else
1457                    ypol(2) = EV%
1458                    octg=0
1459                end if
1460                octgprime=0
1461            else
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1436

```
1436            clxr=y(EV%r_ix)
1437            qr  =y(EV%r_ix+1)
1438            pir =y(EV%r_ix+2)
1439            pirdot=yprime(EV%r_ix
1440        end if
1441
1442        if (EV%no_phot_multpoles)
1443            z=(0.5_dl*dgrho/k + e
1444            dz= -adotoa*z - 0.5_d
1445            clxg=-4*dz/k -4/k*opa
1446            qg=-4._dl/3*z
1447            pig=0
1448            pigdot=0
1449            octg=0
1450            octgprime=0
1451            qgdot = -4*dz/3
1452        else
1453            if (EV%TightCoupling)
1454                pig = EV%pig
1455                !pigdot=EV%pigdot
1456                if (second_order_
1457                    octg = (3._dl
1458                    ypol(2) = EV%
1459                    ypol(3) = (3.
1460                else
1461                    ypol(2) = EV%
1462                    octg=0
1463                end if
1464                octgprime=0
1465            else
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1462

```
1462                pig =y(EV%g_ix+2)
1463                pigdot=yprime(EV%
1464                octg=y(EV%g_ix+3)
1465                octgprime=yprime(
1466            end if
1467            clxg=y(EV%g_ix)
1468            qg  =y(EV%g_ix+1)
1469            qgdot =yprime(EV%g_ix
1470        end if
1471
1472        dgrho = dgrho + grhog_t*c
1473        dgq   = dgq   + grhog_t*q
1474        dgpi  = dgpi  + grhor_t*p
1475
1476
1477        !  Get sigma (shear) and
1478        !  have to get z from eta
1479        z=(0.5_dl*dgrho/k + etak)
1480        sigma=(z+1.5_dl*dgq/k2)/E
1481
1482        if (is_cosmological_const
1483            ppiedot=0
1484        else
1485            hdotoh=(-3._dl*grho-3
1486            ppiedot=3._dl*EV%dgrh
1487            grhov_t*(1+w_eff)*k*z
1488            ppiedot=ppiedot*adoto
1489        end if
1490
1491        polter = 0.1_dl*pig+9._dl
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1466

```
1466                pig =y(EV%g_ix+2)
1467                pigdot=yprime(EV%
1468                octg=y(EV%g_ix+3)
1469                octgprime=yprime(
1470            end if
1471            clxg=y(EV%g_ix)
1472            qg  =y(EV%g_ix+1)
1473            qgdot =yprime(EV%g_ix
1474        end if
1475
1476        dgrho = dgrho + grhog_t*c
1477        dgq   = dgq   + grhog_t*q
1478        dgpi  = dgpi  + grhor_t*p
1479
1480
1481        !  Get sigma (shear) and
1482        !  have to get z from eta
1483        z=(0.5_dl*dgrho/k + etak)
1484        sigma=(z+1.5_dl*dgq/k2)/E
1485
1486        if (is_cosmological_const
1487            ppiedot=0
1488        else
1489            hdotoh=(-3._dl*grho-3
1490            ppiedot=3._dl*EV%dgrh
1491                grhov_t*(1+w_eff)
1492            ppiedot=ppiedot*adoto
1493        end if
1494
1495        polter = 0.1_dl*pig+9._dl
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1492

```
1492
1493          if (CP%flat) then
1494              x=k*(CP%tau0-tau)
1495              divfac=x*x
1496          else
1497              x=(CP%tau0-tau)/CP%r
1498              divfac=(CP%r*rofChi(x
1499          end if
1500
1501
1502          if (EV%TightCoupling) the
1503              if (second_order_tigh
1504                  pigdot = EV%pigdo
1505                  ypolprime(2)= (pi
1506              else
1507                  pigdot = -dopac(j
1508                  +etak/EV%Kf(1)-
1509                  ypolprime(2)= pig
1510              end if
1511          end if
1512
1513          pidot_sum =  pidot_sum +
1514          diff_rhopi = pidot_sum -
1515
1516
1517          !Maple's fortran output -
1518          !2phi' term (\phi' + \psi
1519          ISW = (4.D0/3.D0*k*EV%Kf(
1520          -diff_rhopi/k**2-1.D0/ado
1521          -2.D0/k*adotoa/EV%Kf(1)*e
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1496

```
1496
1497          if (CP%flat) then
1498              x=k*(CP%tau0-tau)
1499              divfac=x*x
1500          else
1501              x=(CP%tau0-tau)/CP%r
1502              divfac=(CP%r*rofChi(x
1503          end if
1504
1505
1506          if (EV%TightCoupling) the
1507              if (second_order_tigh
1508                  pigdot = EV%pigdo
1509                  ypolprime(2)= (pi
1510              else
1511                  pigdot = -dopacit
1512                      +etak/EV%Kf(1
1513                  ypolprime(2)= pig
1514              end if
1515          end if
1516
1517          pidot_sum =  pidot_sum +
1518          diff_rhopi = pidot_sum -
1519
1520
1521          !Maple's fortran output -
1522          !2phi' term (\phi' + \psi
1523          ISW = (4.D0/3.D0*k*EV%Kf(
1524              -diff_rhopi/k**2-1.D0
1525              -2.D0/k*adotoa/EV%Kf(
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1522

```
1522
1523          !e.g. to get only late-ti
1524          !  if (1/a-1 < 30) ISW=0
1525
1526          !The rest, note y(9)->oct
1527          sources(1)= ISW +  ((-9.D
1528          (11.D0/10.D0*sigma- 3.D0/
1529          (-180.D0*ypolprime(2)-30.
1530          (-(9.D0*pigdot+ 54.D0*ypo
1531          (-21.D0/5.D0*adotoa*sigma
1532          vbdot+3.D0/40.D0*qgdot- 9
1533          (-9.D0/160.D0*dopac(j)*pi
1534          (3.D0/16.D0*ddvis(j)*pig+
1535
1536          ! Doppler term
1537          !   sources(1)=  (sigma+v
1538          !         +1.D0/k/EV%Kf(1
1539
1540          !Equivalent full result
1541          !    t4 = 1.D0/adotoa
1542          !    t92 = k**2
1543          !    sources(1) = (4.D0/3
1544          !        (3.D0/8.D0*ypol(
1545          !    sources(1) = sources
1546          !         3.D0/8.D0*EV%Kf
1547          !        gpres)*sigma*exp
1548          !        EV%Kf(1)+(vbdot-
1549          !        5.D0*sigma*adoto
1550          !        27.D0/80.D0*ypol
1551          !        -9.D0/160.D0*dop
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1526

```
1526
1527          !e.g. to get only late-ti
1528          !  if (1/a-1 < 30) ISW=0
1529
1530          !The rest, note y(9)->oct
1531          sources(1)= ISW +  ((-9.D
1532              (11.D0/10.D0*sigma- 3
1533              (-180.D0*ypolprime(2)
1534              (-(9.D0*pigdot+ 54.D0
1535              (-21.D0/5.D0*adotoa*s
1536              vbdot+3.D0/40.D0*qgdo
1537              (-9.D0/160.D0*dopacit
1538              (3.D0/16.D0*ddvisibil
1539
1540          ! Doppler term
1541          !   sources(1)=  (sigma+v
1542          !         +1.D0/k/EV%Kf(1
1543
1544          !Equivalent full result
1545          !    t4 = 1.D0/adotoa
1546          !    t92 = k**2
1547          !    sources(1) = (4.D0/3
1548          !        (3.D0/8.D0*ypol(
1549          !    sources(1) = sources
1550          !         3.D0/8.D0*EV%Kf
1551          !        gpres)*sigma*exp
1552          !        EV%Kf(1)+(vbdot-
1553          !        5.D0*sigma*adoto
1554          !        27.D0/80.D0*ypol
1555          !        -9.D0/160.D0*dop
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1552

```
1552        !         160.D0*pig)*opac
1553        !         8.D0*ddvis(j)*yp
1554
1555
1556        if (x > 0._dl) then
1557            !E polarization sourc
1558            sources(2)=vis(j)*pol
1559            !factor of four becau
1560        else
1561            sources(2)=0
1562        end if
1563
1564        if (CTransScal%NumSources
1565            !Get lensing sources
1566            !Can modify this here
1567            if (tau > tau_maxvis
1568                !phi_lens = Phi -
1569                phi = -(dgrho +3*
1570
1571                sources(3) = -2*p
1572                !We include the l
1573            else
1574                sources(3) = 0
1575            end if
1576        end if
1577
1578        end subroutine output
1579
1580
1581        !ccccccccccccccccccccccc
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1556

```
1556        !         160.D0*pig)*opac
1557        !         8.D0*ddvisibilit
1558
1559
1560        if (x > 0._dl) then
1561            !E polarization sourc
1562            sources(2)=visibility
1563            !factor of four becau
1564        else
1565            sources(2)=0
1566        end if
1567
1568        if (CTransScal%NumSources
1569            !Get lensing sources
1570            !Can modify this here
1571            if (tau > tau_maxvis
1572                !phi_lens = Phi -
1573                phi = -(dgrho +3*
1574
1575                sources(3) = -2*p
1576                !We include the l
1577            else
1578                sources(3) = 0
1579            end if
1580        end if
1581
1582        end subroutine output
1583
1584
1585        !ccccccccccccccccccccccc
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1582

```
1582        subroutine outputt(EV,yt,
1583        !calculate the tensor sou
1584        use ThermoData
1585
1586        implicit none
1587        integer j,n
1588        type(EvolutionVars) :: EV
1589        real(dl), target :: yt(n)
1590        real(dl) tau,dt,dte,dtb,x
1591        real(dl) pig, pigdot, oct
1592        real(dl) sinhxr,cothxor
1593        real(dl) k,k2
1594        real(dl), dimension(:),po
1595        real(dl), target :: pol(3
1596        real(dl) dtauda
1597
1598        call derivst(EV,EV%nvart,
1599
1600        k2=EV%k2_buf
1601        k=EV%k_buf
1602        aux=EV%aux_buf
1603        shear = yt(3)
1604
1605        x=(CP%tau0-tau)/CP%r
1606
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1586

```
1586        subroutine outputt(EV,yt,
1587        !calculate the tensor sou
1588        use ThermoData
1589
1590        implicit none
1591        integer n
1592        type(EvolutionVars) :: EV
1593        real(dl), target :: yt(n)
1594        real(dl) tau,dt,dte,dtb,x
1595        real(dl) pig, pigdot, oct
1596        real(dl) sinhxr,cothxor
1597        real(dl) k,k2
1598        real(dl), dimension(:),po
1599        real(dl), target :: pol(3
1600        real(dl) dtauda
1601        real(dl) opacity, dopacit
1602
1603
1604        call IonizationFunctionsA
1605            visibility, dvisibili
1606
1607        call derivst(EV,EV%nvart,
1608
1609        k2=EV%k2_buf
1610        k=EV%k_buf
1611        aux=EV%aux_buf
1612        shear = yt(3)
1613
1614        x=(CP%tau0-tau)/CP%r
1615
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1607

```
1607        !  And the electric part
1608        if (.not. EV%TensTightCou
1609            !  Use the full expre
1610            pig=yt(EV%g_ix+2)
1611            pigdot=ytprime(EV%g_i
1612            E => yt(EV%E_ix+1:)
1613            Eprime=> ytprime(EV%E
1614            Bprime => ytprime(EV%
1615            octg=ytprime(EV%g_ix+
1616        else
1617            !  Use the tight-coup
1618            a =yt(1)
1619            adotoa = 1/(a*dtauda(
1620            pigdot=32._dl/45._dl*
1621            pig = 32._dl/45._dl*k
1622            pol=0
1623            polEprime=0
1624            polBprime=0
1625            E=>pol
1626            EPrime=>polEPrime
1627            BPrime=>polBPrime
1628            E(2)=pig/4._dl
1629            EPrime(2)=pigdot/4
1630            octg=0
1631        endif
1632
1633        sinhxr=rofChi(x)*CP%r
1634
1635        if (EV%q*sinhxr > 1.e-8_d
1636            prefac=sqrt(EV%q2*CP%
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1616

```
1616        !  And the electric part
1617        if (.not. EV%TensTightCou
1618            !  Use the full expre
1619            pig=yt(EV%g_ix+2)
1620            pigdot=ytprime(EV%g_i
1621            E => yt(EV%E_ix+1:)
1622            Eprime=> ytprime(EV%E
1623            Bprime => ytprime(EV%
1624            octg=ytprime(EV%g_ix+
1625        else
1626            !  Use the tight-coup
1627            a =yt(1)
1628            adotoa = 1/(a*dtauda(
1629            pigdot=32._dl/45._dl*
1630            pig = 32._dl/45._dl*k
1631            pol=0
1632            polEprime=0
1633            polBprime=0
1634            E=>pol
1635            EPrime=>polEPrime
1636            BPrime=>polBPrime
1637            E(2)=pig/4._dl
1638            EPrime(2)=pigdot/4
1639            octg=0
1640        endif
1641
1642        sinhxr=rofChi(x)*CP%r
1643
1644        if (EV%q*sinhxr > 1.e-8_d
1645            prefac=sqrt(EV%q2*CP%
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1637

```
1637            cothxor=cosfunc(x)/si
1638
1639            polter = 0.1_dl*pig +
1640            polterdot=9._dl/15._d
1641            polterddot = 9._dl/15
1642            Eprime(2)-polterdot)
1643            +0.1_dl*(k*(-octg*EV%
1644            dopac(j)*(pig - polte
1645
1646            dt=(shear*expmmu(j) +
1647
1648            dte=CP%r*15._dl/8._dl
1649            ((ddvis(j)*polter + 2
1650            + 4._dl*cothxor*(dvis
1651            vis(j)*polter*(k2 -6*
1652
1653            dtb=15._dl/4._dl*EV%q
1654        else
1655            dt=0._dl
1656            dte=0._dl
1657            dtb=0._dl
1658        end if
1659
1660        end subroutine outputt
1661
1662        !ccccccccccccccccccccccc
1663        subroutine outputv(EV,yv,
1664        !calculate the vector sou
1665        use ThermoData
1666
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1646

```
1646            cothxor=cosfunc(x)/si
1647
1648            polter = 0.1_dl*pig +
1649            polterdot=9._dl/15._d
1650            polterddot = 9._dl/15
1651                Eprime(2)-polterd
1652                +0.1_dl*(k*(-octg
1653                dopacity*(pig - p
1654
1655            dt=(shear*exptau + (1
1656
1657            dte=CP%r*15._dl/8._dl
1658                ((ddvisibility*po
1659                + 4._dl*cothxor*(
1660                visibility*polter
1661
1662            dtb=15._dl/4._dl*EV%q
1663        else
1664            dt=0._dl
1665            dte=0._dl
1666            dtb=0._dl
1667        end if
1668
1669        end subroutine outputt
1670
1671        !ccccccccccccccccccccccc
1672        subroutine outputv(EV,yv,
1673        !calculate the vector sou
1674        use ThermoData
1675
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1667

```
1667          implicit none
1668          integer j,n
1669          type(EvolutionVars) :: EV
1670          real(dl), target :: yv(n)
1671          real(dl) tau,dt,dte,dtb,x
1672          real(dl) vb,qg, pig, polt
1673          real(dl) k,k2
1674          real(dl), dimension(:),po
1675
1676          call derivsv(EV,EV%nvarv,
1677
1678          k2=EV%k2_buf
1679          k=EV%k_buf
1680          sigma = yv(2)
1681          vb  = yv(3)
1682          qg  = yv(4)
1683          pig = yv(5)
1684
1685
1686          x=(CP%tau0-tau)*k
1687
1688          if (x > 1.e-8_dl) then
1689              E => yv(EV%lmaxv+3:)
1690              Eprime=> yvprime(EV%l
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1676

```
1676          implicit none
1677          integer n
1678          type(EvolutionVars) :: EV
1679          real(dl), target :: yv(n)
1680          real(dl) tau,dt,dte,dtb,x
1681          real(dl) vb,qg, pig, polt
1682          real(dl) k,k2
1683          real(dl), dimension(:),po
1684          real(dl) opacity, dopacit
1685
1686
1687          call IonizationFunctionsA
1688              visibility, dvisibili
1689
1690
1691          call derivsv(EV,EV%nvarv,
1692
1693          k2=EV%k2_buf
1694          k=EV%k_buf
1695          sigma = yv(2)
1696          vb  = yv(3)
1697          qg  = yv(4)
1698          pig = yv(5)
1699
1700
1701          x=(CP%tau0-tau)*k
1702
1703          if (x > 1.e-8_dl) then
1704              E => yv(EV%lmaxv+3:)
1705              Eprime=> yvprime(EV%l
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1691

```
1691
1692            polter = 0.1_dl*pig +
1693            polterdot=9._dl/15._d
1694
1695            if (yv(1) < 1e-3) the
1696                dt = 1
1697            else
1698                dt =0
1699            end if
1700            dt= (4*(vb+sigma)*vis
1701            + 4*(expmmu(j)*yvprim
1702
1703            dte= 15._dl/2*2*polte
1704
1705            dtb= -15._dl/2*polter
1706        else
1707            dt=0
1708            dte=0
1709            dtb=0
1710        end if
1711
1712        end subroutine outputv
1713
1714
1715        !cccccccccccccccccccccccc
1716        subroutine initial(EV,y,
1717        !  Initial conditions.
1718        use ThermoData
1719        implicit none
1720
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1706

```
1706
1707            polter = 0.1_dl*pig +
1708            polterdot=9._dl/15._d
1709
1710            if (yv(1) < 1e-3) the
1711                dt = 1
1712            else
1713                dt =0
1714            end if
1715            dt= (4*(vb+sigma)*vis
1716                + 4*(exptau*yvpri
1717
1718            dte= 15._dl/2*2*polte
1719
1720            dtb= -15._dl/2*polter
1721        else
1722            dt=0
1723            dte=0
1724            dtb=0
1725        end if
1726
1727        end subroutine outputv
1728
1729
1730        !cccccccccccccccccccccccc
1731        subroutine initial(EV,y,
1732        !  Initial conditions.
1733        use ThermoData
1734        implicit none
1735
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1721

```
1721        type(EvolutionVars) EV
1722        real(dl) y(EV%nvar)
1723        real(dl) Rp15,tau,x,x2,x3
1724        Rc,Rb,Rv,Rg,grhonu,chi
1725        real(dl) k,k2
1726        real(dl) a,a2, iqg, rhoma
1727        integer l,i, nu_i, j, ind
1728        integer, parameter :: i_c
1729        i_qg=5,i_qr=6,i_vb=7,i_pi
1730        integer, parameter :: i_m
1731        real(dl) initv(6,1:i_max)
1732
1733        nullify(EV%OutputTransfer
1734
1735        if (CP%flat) then
1736            EV%k_buf=EV%q
1737            EV%k2_buf=EV%q2
1738            EV%Kf(1:EV%MaxlNeeded
1739        else
1740            EV%k2_buf=EV%q2-CP%cu
1741            EV%k_buf=sqrt(EV%k2_b
1742
1743            do l=1,EV%MaxlNeeded
1744                EV%Kf(l)=1._dl-CP
1745            end do
1746        end if
1747
1748        k=EV%k_buf
1749        k2=EV%k2_buf
1750
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1736

```
1736        type(EvolutionVars) EV
1737        real(dl) y(EV%nvar)
1738        real(dl) Rp15,tau,x,x2,x3
1739            Rc,Rb,Rv,Rg,grhonu,ch
1740        real(dl) k,k2
1741        real(dl) a,a2, iqg, rhoma
1742        integer l,i, nu_i, j, ind
1743        integer, parameter :: i_c
1744            i_qg=5,i_qr=6,i_vb=7,
1745        integer, parameter :: i_m
1746        real(dl) initv(6,1:i_max)
1747
1748        nullify(EV%OutputTransfer
1749
1750        if (CP%flat) then
1751            EV%k_buf=EV%q
1752            EV%k2_buf=EV%q2
1753            EV%Kf(1:EV%MaxlNeeded
1754        else
1755            EV%k2_buf=EV%q2-CP%cu
1756            EV%k_buf=sqrt(EV%k2_b
1757
1758            do l=1,EV%MaxlNeeded
1759                EV%Kf(l)=1._dl-CP
1760            end do
1761        end if
1762
1763        k=EV%k_buf
1764        k2=EV%k2_buf
1765
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1751

```
1751        do j=1,EV%MaxlNeeded
1752            EV%denlk(j)=denl(j)*k
1753            EV%denlk2(j)=denl(j)*
1754            EV%polfack(j)=polfac(
1755        end do
1756
1757        !Get time to switch off t
1758        !The numbers here are a b
1759        !The high k increase save
1760        !The lower k ones are mor
1761        !as ensuring tight coupli
1762        if (EV%k_buf > epsw) then
1763            if (EV%k_buf > epsw*5
1764                ep=ep0*5/Accuracy
1765                if (HighAccuracyD
1766            else
1767                ep=ep0
1768            end if
1769        else
1770            ep=ep0
1771        end if
1772        if (second_order_tightcou
1773        EV%TightSwitchoffTime = m
1774
1775
1776        y=0
1777
1778        !  k*tau, (k*tau)**2, (k*
1779        x=k*tau
1780        x2=x*x
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1766

```
1766        do j=1,EV%MaxlNeeded
1767            EV%denlk(j)=denl(j)*k
1768            EV%denlk2(j)=denl(j)*
1769            EV%polfack(j)=polfac(
1770        end do
1771
1772        !Get time to switch off t
1773        !The numbers here are a b
1774        !The high k increase save
1775        !The lower k ones are mor
1776        !as ensuring tight coupli
1777        if (EV%k_buf > epsw) then
1778            if (EV%k_buf > epsw*5
1779                ep=ep0*5/Accuracy
1780                if (HighAccuracyD
1781            else
1782                ep=ep0
1783            end if
1784        else
1785            ep=ep0
1786        end if
1787        if (second_order_tightcou
1788        EV%TightSwitchoffTime = m
1789
1790
1791        y=0
1792
1793        !  k*tau, (k*tau)**2, (k*
1794        x=k*tau
1795        x2=x*x
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1781

```
1781          x3=x2*x
1782          rhomass =  sum(grhormass(
1783          grhonu=rhomass+grhornomas
1784
1785          om = (grhob+grhoc)/sqrt(3
1786          omtau=om*tau
1787          Rv=grhonu/(grhonu+grhog)
1788
1789          Rg = 1-Rv
1790          Rc=CP%omegac/(CP%omegac+C
1791          Rb=1-Rc
1792          Rp15=4*Rv+15
1793
1794          if (CP%Scalar_initial_con
1795          stop 'Invalid initial con
1796
1797          a=tau*adotrad*(1+omtau/4)
1798          a2=a*a
1799
1800          initv=0
1801
1802          !  Set adiabatic initial
1803
1804          chi=1  !Get transfer func
1805          initv(1,i_clxg)=-chi*EV%K
1806          initv(1,i_clxr)= initv(1,
1807          initv(1,i_clxb)=0.75_dl*i
1808          initv(1,i_clxc)=initv(1,i
1809          initv(1,i_qg)=initv(1,i_c
1810          initv(1,i_qr)=-chi*EV%Kf(
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1796

```
1796          x3=x2*x
1797          rhomass =  sum(grhormass(
1798          grhonu=rhomass+grhornomas
1799
1800          om = (grhob+grhoc)/sqrt(3
1801          omtau=om*tau
1802          Rv=grhonu/(grhonu+grhog)
1803
1804          Rg = 1-Rv
1805          Rc=CP%omegac/(CP%omegac+C
1806          Rb=1-Rc
1807          Rp15=4*Rv+15
1808
1809          if (CP%Scalar_initial_con
1810              stop 'Invalid initial
1811
1812          a=tau*adotrad*(1+omtau/4)
1813          a2=a*a
1814
1815          initv=0
1816
1817          !  Set adiabatic initial
1818
1819          chi=1  !Get transfer func
1820          initv(1,i_clxg)=-chi*EV%K
1821          initv(1,i_clxr)= initv(1,
1822          initv(1,i_clxb)=0.75_dl*i
1823          initv(1,i_clxc)=initv(1,i
1824          initv(1,i_qg)=initv(1,i_c
1825          initv(1,i_qr)=-chi*EV%Kf(
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1811

```
1811          initv(1,i_vb)=0.75_dl*ini
1812          initv(1,i_pir)=chi*4._dl/
1813          initv(1,i_aj3r)=chi*4/21.
1814          initv(1,i_eta)=-chi*2*EV%
1815
1816          if (CP%Scalar_initial_con
1817              !CDM isocurvature
1818
1819              initv(2,i_clxg)= Rc*o
1820              initv(2,i_clxr)=initv
1821              initv(2,i_clxb)=initv
1822              initv(2,i_clxc)=1+ini
1823              initv(2,i_qg)=-Rc/9*o
1824              initv(2,i_qr)=initv(2
1825              initv(2,i_vb)=0.75_dl
1826              initv(2,i_pir)=-Rc*om
1827              initv(2,i_eta)= Rc*om
1828              initv(2,i_aj3r)=0
1829              !Baryon isocurvature
1830              if (Rc==0) stop 'Isoc
1831
1832              initv(3,:) = initv(2,
1833              initv(3,i_clxc) = ini
1834              initv(3,i_clxb)= init
1835
1836              !neutrino isocurvatur
1837
1838              initv(4,i_clxg)=Rv/Rg
1839              initv(4,i_clxr)=1-x2/
1840              initv(4,i_clxc)=-omta
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1826

```
1826          initv(1,i_vb)=0.75_dl*ini
1827          initv(1,i_pir)=chi*4._dl/
1828          initv(1,i_aj3r)=chi*4/21.
1829          initv(1,i_eta)=-chi*2*EV%
1830
1831          if (CP%Scalar_initial_con
1832              !CDM isocurvature
1833
1834              initv(2,i_clxg)= Rc*o
1835              initv(2,i_clxr)=initv
1836              initv(2,i_clxb)=initv
1837              initv(2,i_clxc)=1+ini
1838              initv(2,i_qg)=-Rc/9*o
1839              initv(2,i_qr)=initv(2
1840              initv(2,i_vb)=0.75_dl
1841              initv(2,i_pir)=-Rc*om
1842              initv(2,i_eta)= Rc*om
1843              initv(2,i_aj3r)=0
1844              !Baryon isocurvature
1845              if (Rc==0) stop 'Isoc
1846
1847              initv(3,:) = initv(2,
1848              initv(3,i_clxc) = ini
1849              initv(3,i_clxb)= init
1850
1851              !neutrino isocurvatur
1852
1853              initv(4,i_clxg)=Rv/Rg
1854              initv(4,i_clxr)=1-x2/
1855              initv(4,i_clxc)=-omta
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1841

```
1841 |            initv(4,i_clxb)= Rv/R
1842 |            iqg = - Rv/Rg*(x/3 -
1843 |            initv(4,i_qg) =iqg
1844 |            initv(4,i_qr) = x/3
1845 |            initv(4,i_vb)=0.75_dl
1846 |            initv(4,i_pir)=x2/Rp1
1847 |            initv(4,i_eta)=EV%Kf(
1848 |
1849 |            !neutrino isocurvatur
1850 |
1851 |            initv(5,i_clxg)=Rv/Rg
1852 |            initv(5,i_clxr)=-x -3
1853 |            initv(5,i_clxc)=-9*om
1854 |            initv(5,i_clxb)= 3*Rv
1855 |            iqg = Rv/Rg*(-1 + 3*R
1856 |            initv(5,i_qg) =iqg
1857 |            initv(5,i_qr) = 1 - x
1858 |            initv(5,i_vb)=0.75_dl
1859 |            initv(5,i_pir)=2*x/(4
1860 |            initv(5,i_eta)=2*EV%K
1861 |            initv(5,i_aj3r) = 3._
1862 |
1863 |            !quintessence isocurv
1864 |        end if
1865 |
1866 |        if (CP%Scalar_initial_con
1867 |            InitVec = 0
1868 |            do i=1,initial_nummod
1869 |                InitVec = InitVec
1870 |            end do
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1856

```
1856 |            initv(4,i_clxb)= Rv/R
1857 |            iqg = - Rv/Rg*(x/3 -
1858 |            initv(4,i_qg) =iqg
1859 |            initv(4,i_qr) = x/3
1860 |            initv(4,i_vb)=0.75_dl
1861 |            initv(4,i_pir)=x2/Rp1
1862 |            initv(4,i_eta)=EV%Kf(
1863 |
1864 |            !neutrino isocurvatur
1865 |
1866 |            initv(5,i_clxg)=Rv/Rg
1867 |            initv(5,i_clxr)=-x -3
1868 |            initv(5,i_clxc)=-9*om
1869 |            initv(5,i_clxb)= 3*Rv
1870 |            iqg = Rv/Rg*(-1 + 3*R
1871 |            initv(5,i_qg) =iqg
1872 |            initv(5,i_qr) = 1 - x
1873 |            initv(5,i_vb)=0.75_dl
1874 |            initv(5,i_pir)=2*x/(4
1875 |            initv(5,i_eta)=2*EV%K
1876 |            initv(5,i_aj3r) = 3._
1877 |
1878 |            !quintessence isocurv
1879 |        end if
1880 |
1881 |        if (CP%Scalar_initial_con
1882 |            InitVec = 0
1883 |            do i=1,initial_nummod
1884 |                InitVec = InitVec
1885 |            end do
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1871

```
1871        else
1872            InitVec = initv(CP%Sc
1873            if (CP%Scalar_initial
1874            !So we start with chi
1875        end if
1876
1877        y(1)=a
1878        y(2)= -InitVec(i_eta)*k/2
1879        !get eta_s*k, where eta_s
1880
1881        !  CDM
1882        y(3)=InitVec(i_clxc)
1883
1884        !  Baryons
1885        y(4)=InitVec(i_clxb)
1886        y(5)=InitVec(i_vb)
1887
1888        !  Photons
1889        y(EV%g_ix)=InitVec(i_clxg
1890        y(EV%g_ix+1)=InitVec(i_qg
1891
1892        if (.not. is_cosmological
1893            y(EV%w_ix) = InitVec(
1894        end if
1895
1896        !  Neutrinos
1897        y(EV%r_ix)=InitVec(i_clxr
1898        y(EV%r_ix+1)=InitVec(i_qr
1899        y(EV%r_ix+2)=InitVec(i_pi
1900
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1886

```
1886        else
1887            InitVec = initv(CP%Sc
1888            if (CP%Scalar_initial
1889            !So we start with chi
1890        end if
1891
1892        y(1)=a
1893        y(2)= -InitVec(i_eta)*k/2
1894        !get eta_s*k, where eta_s
1895
1896        !  CDM
1897        y(3)=InitVec(i_clxc)
1898
1899        !  Baryons
1900        y(4)=InitVec(i_clxb)
1901        y(5)=InitVec(i_vb)
1902
1903        !  Photons
1904        y(EV%g_ix)=InitVec(i_clxg
1905        y(EV%g_ix+1)=InitVec(i_qg
1906
1907        if (.not. is_cosmological
1908            y(EV%w_ix) = InitVec(
1909        end if
1910
1911        !  Neutrinos
1912        y(EV%r_ix)=InitVec(i_clxr
1913        y(EV%r_ix+1)=InitVec(i_qr
1914        y(EV%r_ix+2)=InitVec(i_pi
1915
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1901

```
1901        if (EV%lmaxnr>2) then
1902            y(EV%r_ix+3)=InitVec(
1903        endif
1904
1905        if (CP%Num_Nu_massive ==
1906
1907        do nu_i = 1, CP%Nu_mass_e
1908            EV%MassiveNuApproxTim
1909            a_massive =  20000*k/
1910            if (a_massive >=0.99)
1911                EV%MassiveNuAppro
1912            else if (a_massive >
1913                EV%MassiveNuAppro
1914            end if
1915            ind = EV%nu_ix(nu_i)
1916            do  i=1,EV%nq(nu_i)
1917                y(ind:ind+2)=y(EV
1918                if (EV%lmaxnu_tau
1919                ind = ind + EV%lm
1920            end do
1921        end do
1922
1923        end subroutine initial
1924
1925
1926        !ccccccccccccccccccccccc
1927        subroutine initialt(EV,yt
1928        !  Initial conditions for
1929        use ThermoData
1930        implicit none
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1916

```
1916        if (EV%lmaxnr>2) then
1917            y(EV%r_ix+3)=InitVec(
1918        endif
1919
1920        if (CP%Num_Nu_massive ==
1921
1922        do nu_i = 1, CP%Nu_mass_e
1923            EV%MassiveNuApproxTim
1924            a_massive =  20000*k/
1925            if (a_massive >=0.99)
1926                EV%MassiveNuAppro
1927            else if (a_massive >
1928                EV%MassiveNuAppro
1929            end if
1930            ind = EV%nu_ix(nu_i)
1931            do  i=1,EV%nq(nu_i)
1932                y(ind:ind+2)=y(EV
1933                if (EV%lmaxnu_tau
1934                ind = ind + EV%lm
1935            end do
1936        end do
1937
1938        end subroutine initial
1939
1940
1941        !ccccccccccccccccccccccc
1942        subroutine initialt(EV,yt
1943        !  Initial conditions for
1944        use ThermoData
1945        implicit none
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1931

```
1931        real(dl) bigR,tau,x,aj3r,
1932        integer l
1933        type(EvolutionVars) EV
1934        real(dl) k,k2 ,a, omtau
1935        real(dl) yt(EV%nvart)
1936        real(dl) tens0, ep, tensf
1937
1938        if (CP%flat) then
1939            EV%aux_buf=1._dl
1940            EV%k2_buf=EV%q2
1941            EV%k_buf=EV%q
1942            EV%Kft(1:EV%MaxlNeede
1943        else
1944            EV%k2_buf=EV%q2-3*CP%
1945            EV%k_buf=sqrt(EV%k2_b
1946            EV%aux_buf=sqrt(1._dl
1947        endif
1948
1949        k=EV%k_buf
1950        k2=EV%k2_buf
1951
1952        do l=1,EV%MaxlNeededt
1953            if (.not. CP%flat) EV
1954            EV%denlkt(1,l)=k*denl
1955            tensfac=real((l+3)*(l
1956            EV%denlkt(2,l)=k*denl
1957            EV%denlkt(3,l)=k*denl
1958            EV%denlkt(4,l)=k*4._d
1959        end do
1960
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1946

```
1946        real(dl) bigR,tau,x,aj3r,
1947        integer l
1948        type(EvolutionVars) EV
1949        real(dl) k,k2 ,a, omtau
1950        real(dl) yt(EV%nvart)
1951        real(dl) tens0, ep, tensf
1952
1953        if (CP%flat) then
1954            EV%aux_buf=1._dl
1955            EV%k2_buf=EV%q2
1956            EV%k_buf=EV%q
1957            EV%Kft(1:EV%MaxlNeede
1958        else
1959            EV%k2_buf=EV%q2-3*CP%
1960            EV%k_buf=sqrt(EV%k2_b
1961            EV%aux_buf=sqrt(1._dl
1962        endif
1963
1964        k=EV%k_buf
1965        k2=EV%k2_buf
1966
1967        do l=1,EV%MaxlNeededt
1968            if (.not. CP%flat) EV
1969            EV%denlkt(1,l)=k*denl
1970            tensfac=real((l+3)*(l
1971            EV%denlkt(2,l)=k*denl
1972            EV%denlkt(3,l)=k*denl
1973            EV%denlkt(4,l)=k*4._d
1974        end do
1975
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1961

```
1961        if (k > 0.06_dl*epsw) the
1962            ep=ep0
1963        else
1964            ep=0.2_dl*ep0
1965        end if
1966
1967        !    finished_tightcoupli
1968        EV%TightSwitchoffTime = m
1969
1970        a=tau*adotrad
1971        rhomass =  sum(grhormass(
1972        omtau = tau*(grhob+grhoc)
1973
1974        if (DoTensorNeutrinos) th
1975            bigR = (rhomass+grhor
1976        else
1977            bigR = 0._dl
1978        end if
1979
1980        x=k*tau
1981
1982        yt(1)=a
1983        tens0 = 1
1984
1985        yt(2)= tens0
1986        !commented things are for
1987        !-15/28._dl*x**2*(bigR-1)
1988
1989        elec=-tens0*(1+2*CP%curv/
1990
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 1976

```
1976        if (k > 0.06_dl*epsw) the
1977            ep=ep0
1978        else
1979            ep=0.2_dl*ep0
1980        end if
1981
1982        !    finished_tightcoupli
1983        EV%TightSwitchoffTime = m
1984
1985        a=tau*adotrad
1986        rhomass =  sum(grhormass(
1987        omtau = tau*(grhob+grhoc)
1988
1989        if (DoTensorNeutrinos) th
1990            bigR = (rhomass+grhor
1991        else
1992            bigR = 0._dl
1993        end if
1994
1995        x=k*tau
1996
1997        yt(1)=a
1998        tens0 = 1
1999
2000        yt(2)= tens0
2001        !commented things are for
2002        !-15/28._dl*x**2*(bigR-1)
2003
2004        elec=-tens0*(1+2*CP%curv/
2005
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 1991

```
1991        !shear
1992        yt(3)=-5._dl/2/(bigR+5)*x
1993        !           + 15._dl/14*x*
1994
1995        yt(4:EV%nvart)=0._dl
1996
1997        !  Neutrinos
1998        if (DoTensorNeutrinos) th
1999            pir=-2._dl/3._dl/(big
2000            !             + (bigR-1
2001            aj3r=  -2._dl/21._dl/
2002            !             + 3._dl/7
2003            yt(EV%r_ix+2)=pir
2004            yt(EV%r_ix+3)=aj3r
2005            !Should set up massiv
2006        end if
2007
2008        end subroutine initialt
2009
2010        !cccccccccccccccccccccccc
2011        subroutine initialv(EV,yv
2012        !  Initial conditions for
2013
2014        implicit none
2015        real(dl) bigR,Rc,tau,x,pi
2016        type(EvolutionVars) EV
2017        real(dl) k,k2 ,a, omtau
2018        real(dl) yv(EV%nvarv)
2019
2020        if (CP%flat) then
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2006

```
2006        !shear
2007        yt(3)=-5._dl/2/(bigR+5)*x
2008        !           + 15._dl/14*x*
2009
2010        yt(4:EV%nvart)=0._dl
2011
2012        !  Neutrinos
2013        if (DoTensorNeutrinos) th
2014            pir=-2._dl/3._dl/(big
2015            !             + (bigR-1
2016            aj3r=  -2._dl/21._dl/
2017            !             + 3._
2018            yt(EV%r_ix+2)=pir
2019            yt(EV%r_ix+3)=aj3r
2020            !Should set up massiv
2021        end if
2022
2023        end subroutine initialt
2024
2025        !cccccccccccccccccccccccc
2026        subroutine initialv(EV,yv
2027        !  Initial conditions for
2028
2029        implicit none
2030        real(dl) bigR,Rc,tau,x,pi
2031        type(EvolutionVars) EV
2032        real(dl) k,k2 ,a, omtau
2033        real(dl) yv(EV%nvarv)
2034
2035        if (CP%flat) then
```

```
2021            EV%k2_buf=EV%q2
2022            EV%k_buf=EV%q
2023        else
2024            stop 'Vectors not sup
2025        endif
2026
2027        k=EV%k_buf
2028        k2=EV%k2_buf
2029
2030        omtau = tau*(grhob+grhoc)
2031
2032        a=tau*adotrad*(1+omtau/4)
2033
2034        x=k*tau
2035
2036        bigR = (grhornomass)/(grh
2037        Rc=CP%omegac/(CP%omegac+C
2038
2039        yv(1)=a
2040
2041
2042        yv(2)= vec_sig0*(1- 15._d
2043        !qg
2044        yv(4)= vec_sig0/3* (4*big
2045        (1 - 0.25_dl*omtau*(3*Rc-
2046        -x/2*Magnetic
2047        yv(3)= 3._dl/4*yv(4)
2048
2049        yv(5:EV%nvarv) = 0
2050
```



```
2036            EV%k2_buf=EV%q2
2037            EV%k_buf=EV%q
2038        else
2039            stop 'Vectors not sup
2040        endif
2041
2042        k=EV%k_buf
2043        k2=EV%k2_buf
2044
2045        omtau = tau*(grhob+grhoc)
2046
2047        a=tau*adotrad*(1+omtau/4)
2048
2049        x=k*tau
2050
2051        bigR = (grhornomass)/(grh
2052        Rc=CP%omegac/(CP%omegac+C
2053
2054        yv(1)=a
2055
2056
2057        yv(2)= vec_sig0*(1- 15._d
2058        !qg
2059        yv(4)= vec_sig0/3* (4*big
2060             (1 - 0.25_dl*omtau*(3
2061             -x/2*Magnetic
2062        yv(3)= 3._dl/4*yv(4)
2063
2064        yv(5:EV%nvarv) = 0
2065
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2051

```
2051          !         if (.false.) the
2052          !          yv((EV%lmaxv-1+
2053          !          yv((EV%lmaxv-1+
2054          !          yv((EV%lmaxv-1+
2055          !          yv((EV%lmaxv-1+
2056          !          yv(4) = 0
2057          !          yv(3)= 3._dl/4*
2058          !           return
2059          !         end if
2060
2061          !  Neutrinos
2062          !q_r
2063          yv((EV%lmaxv-1+1)+(EV%lma
2064          + x**2*vec_sig0/6/BigR +0
2065          !pi_r
2066          pir=-2._dl/3._dl*x*vec_si
2067          yv((EV%lmaxv-1+1)+(EV%lma
2068          yv((EV%lmaxv-1+1)+(EV%lma
2069
2070          end subroutine initialv
2071
2072
2073          subroutine outtransf(EV,
2074          !write out clxc, clxb, cl
2075          implicit none
2076          type(EvolutionVars) EV
2077          real(dl), intent(in) :: t
2078          real(dl) clxc, clxb, clxg
2079          real(dl) grho,gpres,dgrho
2080          real, target :: Arr(:)
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2066

```
2066          !         if (.false.) the
2067          !          yv((EV%lmaxv-1+
2068          !          yv((EV%lmaxv-1+
2069          !          yv((EV%lmaxv-1+
2070          !          yv((EV%lmaxv-1+
2071          !          yv(4) = 0
2072          !          yv(3)= 3._dl/4*
2073          !           return
2074          !         end if
2075
2076          !  Neutrinos
2077          !q_r
2078          yv((EV%lmaxv-1+1)+(EV%lma
2079          + x**2*vec_sig0/6/Big
2080          !pi_r
2081          pir=-2._dl/3._dl*x*vec_si
2082          yv((EV%lmaxv-1+1)+(EV%lma
2083          yv((EV%lmaxv-1+1)+(EV%lma
2084
2085          end subroutine initialv
2086
2087
2088          subroutine outtransf(EV,
2089          !write out clxc, clxb, cl
2090          implicit none
2091          type(EvolutionVars) EV
2092          real(dl), intent(in) :: t
2093          real(dl) clxc, clxb, clxg
2094          real(dl) grho,gpres,dgrho
2095          real, target :: Arr(:)
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2081

```
2081        real(dl) y(EV%nvar),yprim
2082
2083        yprime = 0
2084        EV%OutputTransfer =>  Arr
2085        call derivs(EV,EV%ScalEqs
2086        nullify(EV%OutputTransfer
2087
2088        Arr(Transfer_kh+1:Transfe
2089
2090        end subroutine outtransf
2091
2092        !ccccccccccccccccccccccc
2093        subroutine derivs(EV,n,ta
2094        !  Evaluate the time deri
2095        !  ayprime is not necessa
2096        use ThermoData
2097        use MassiveNu
2098        implicit none
2099        type(EvolutionVars) EV
2100
2101        integer n,nu_i
2102        real(dl) ay(n),ayprime(n)
2103        real(dl) tau,w
2104        real(dl) k,k2
2105
2106        !  Internal variables.
2107
2108        real(dl) opacity
2109        real(dl) photbar,cs2,pb43
2110        clxcdot,clxbdot,adotdota,
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2096

```
2096        real(dl) y(EV%nvar),yprim
2097
2098        yprime = 0
2099        EV%OutputTransfer =>  Arr
2100        call derivs(EV,EV%ScalEqs
2101        nullify(EV%OutputTransfer
2102
2103        Arr(Transfer_kh+1:Transfe
2104
2105        end subroutine outtransf
2106
2107        !ccccccccccccccccccccccc
2108        subroutine derivs(EV,n,ta
2109        !  Evaluate the time deri
2110        !  ayprime is not necessa
2111        use ThermoData
2112        use MassiveNu
2113        implicit none
2114        type(EvolutionVars) EV
2115
2116        integer n,nu_i
2117        real(dl) ay(n),ayprime(n)
2118        real(dl) tau,w
2119        real(dl) k,k2
2120
2121        !  Internal variables.
2122
2123        real(dl) opacity
2124        real(dl) photbar,cs2,pb43
2125            clxcdot,clxbdot,adotd
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2111

```
2111 |       real(dl) q,aq,v
2112 |       real(dl) G11_t,G30_t, wnu
2113 |
2114 |       real(dl) dgq,grhob_t,grho
2115 |       real(dl) qgdot,qrdot,pigd
2116 |       real(dl) a,a2,z,clxc,clxb
2117 |       real(dl) clxq, vq,  E2, d
2118 |       integer l,i,ind, ind2, of
2119 |       real(dl) dgs,sigmadot,dz
2120 |       real(dl) dgpi,dgrho_matte
2121 |       !non-flat vars
2122 |       real(dl) cothxor !1/tau i
2123 |       !ppf
2124 |       real(dl) Gamma,S_Gamma,ck
2125 |       real(dl) w_eff, grhoT
2126 |
2127 |       k=EV%k_buf
2128 |       k2=EV%k2_buf
2129 |
2130 |       a=ay(1)
2131 |       a2=a*a
2132 |
2133 |       etak=ay(2)
2134 |
2135 |       !  CDM variables
2136 |       clxc=ay(3)
2137 |
2138 |       !  Baryon variables
2139 |       clxb=ay(4)
2140 |       vb=ay(5)
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2126

```
2126 |       real(dl) q,aq,v
2127 |       real(dl) G11_t,G30_t, wnu
2128 |
2129 |       real(dl) dgq,grhob_t,grho
2130 |       real(dl) qgdot,qrdot,pigd
2131 |       real(dl) a,a2,z,clxc,clxb
2132 |       real(dl) clxq, vq,  E2, d
2133 |       integer l,i,ind, ind2, of
2134 |       real(dl) dgs,sigmadot,dz
2135 |       real(dl) dgpi,dgrho_matte
2136 |       !non-flat vars
2137 |       real(dl) cothxor !1/tau i
2138 |       !ppf
2139 |       real(dl) Gamma,S_Gamma,ck
2140 |       real(dl) w_eff, grhoT
2141 |
2142 |       k=EV%k_buf
2143 |       k2=EV%k2_buf
2144 |
2145 |       a=ay(1)
2146 |       a2=a*a
2147 |
2148 |       etak=ay(2)
2149 |
2150 |       !  CDM variables
2151 |       clxc=ay(3)
2152 |
2153 |       !  Baryon variables
2154 |       clxb=ay(4)
2155 |       vb=ay(5)
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2141

```
2141 |
2142 |       !  Compute expansion rate
2143 |
2144 |       grhob_t=grhob/a
2145 |       grhoc_t=grhoc/a
2146 |       grhor_t=grhornomass/a2
2147 |       grhog_t=grhog/a2
2148 |       if (is_cosmological_const
2149 |           grhov_t=grhov*a2
2150 |           w_eff = -1_dl
2151 |       else
2152 |           !ppf
2153 |           w_eff=w_de(a)   !effe
2154 |           grhov_t=grho_de(a)/a2
2155 |       end if
2156 |
2157 |       !  Get sound speed and io
2158 |       if (EV%TightCoupling) the
2159 |           call thermo(tau,cs2,o
2160 |       else
2161 |           call thermo(tau,cs2,o
2162 |       end if
2163 |
2164 |       gpres=(grhor_t+grhog_t)/3
2165 |       grho_matter=grhob_t+grhoc
2166 |
2167 |       !total perturbations: mat
2168 |       !  8*pi*a*a*SUM[rho_i*clx
2169 |       dgrho_matter=grhob_t*clxb
2170 |       !  8*pi*a*a*SUM[(rho_i+p_
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2156

```
2156 |
2157 |       !  Compute expansion rate
2158 |
2159 |       grhob_t=grhob/a
2160 |       grhoc_t=grhoc/a
2161 |       grhor_t=grhornomass/a2
2162 |       grhog_t=grhog/a2
2163 |       if (is_cosmological_const
2164 |           grhov_t=grhov*a2
2165 |           w_eff = -1_dl
2166 |       else
2167 |           !ppf
2168 |           w_eff=w_de(a)   !effe
2169 |           grhov_t=grho_de(a)/a2
2170 |       end if
2171 |
2172 |       !  Get sound speed and io
2173 |       if (EV%TightCoupling) the
2174 |           call thermo(tau,cs2,o
2175 |       else
2176 |           call thermo(tau,cs2,o
2177 |       end if
2178 |
2179 |       gpres=(grhor_t+grhog_t)/3
2180 |       grho_matter=grhob_t+grhoc
2181 |
2182 |       !total perturbations: mat
2183 |       !  8*pi*a*a*SUM[rho_i*clx
2184 |       dgrho_matter=grhob_t*clxb
2185 |       !  8*pi*a*a*SUM[(rho_i+p_
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2171

```
2171        dgq=grhob_t*vb
2172
2173        if (CP%Num_Nu_Massive > 0
2174            call MassiveNuVars(EV
2175        end if
2176
2177        grho = grho_matter+grhor_
2178
2179        if (CP%flat) then
2180            adotoa=sqrt(grho/3)
2181            cothxor=1._dl/tau
2182        else
2183            adotoa=sqrt((grho+grh
2184            cothxor=1._dl/tanfunc
2185        end if
2186
2187        dgrho = dgrho_matter
2188
2189        ! if (w_lam /= -1 .and. w
2190        !    clxq=ay(EV%w_ix)
2191        !    vq=ay(EV%w_ix+1)
2192        !    dgrho=dgrho + clxq*g
2193        !    dgq = dgq + vq*grhov
2194        !end if
2195
2196        if (EV%no_nu_multpoles) t
2197            !RSA approximation of
2198            !Approximate total de
2199            z=(0.5_dl*dgrho/k + e
2200            dz= -adotoa*z - 0.5_d
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2186

```
2186        dgq=grhob_t*vb
2187
2188        if (CP%Num_Nu_Massive > 0
2189            call MassiveNuVars(EV
2190        end if
2191
2192        grho = grho_matter+grhor_
2193
2194        if (CP%flat) then
2195            adotoa=sqrt(grho/3)
2196            cothxor=1._dl/tau
2197        else
2198            adotoa=sqrt((grho+grh
2199            cothxor=1._dl/tanfunc
2200        end if
2201
2202        dgrho = dgrho_matter
2203
2204        ! if (w_lam /= -1 .and. w
2205        !    clxq=ay(EV%w_ix)
2206        !    vq=ay(EV%w_ix+1)
2207        !    dgrho=dgrho + clxq*g
2208        !    dgq = dgq + vq*grhov
2209        !end if
2210
2211        if (EV%no_nu_multpoles) t
2212            !RSA approximation of
2213            !Approximate total de
2214            z=(0.5_dl*dgrho/k + e
2215            dz= -adotoa*z - 0.5_d
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2201

```
2201            clxr=-4*dz/k
2202            qr=-4._dl/3*z
2203            pir=0
2204        else
2205            !  Massless neutrinos
2206            clxr=ay(EV%r_ix)
2207            qr  =ay(EV%r_ix+1)
2208            pir =ay(EV%r_ix+2)
2209        endif
2210
2211        if (EV%no_phot_multpoles)
2212            if (.not. EV%no_nu_mu
2213                z=(0.5_dl*dgrho/k
2214                dz= -adotoa*z - 0
2215                clxg=-4*dz/k-4/k*
2216                qg=-4._dl/3*z
2217            else
2218                clxg=clxr-4/k*opa
2219                qg=qr
2220            end if
2221            pig=0
2222        else
2223            !  Photons
2224            clxg=ay(EV%g_ix)
2225            qg=ay(EV%g_ix+1)
2226            if (.not. EV%TightCou
2227        end if
2228
2229        !  8*pi*a*a*SUM[rho_i*clx
2230        dgrho=dgrho + grhog_t*clx
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2216

```
2216            clxr=-4*dz/k
2217            qr=-4._dl/3*z
2218            pir=0
2219        else
2220            !  Massless neutrinos
2221            clxr=ay(EV%r_ix)
2222            qr  =ay(EV%r_ix+1)
2223            pir =ay(EV%r_ix+2)
2224        endif
2225
2226        if (EV%no_phot_multpoles)
2227            if (.not. EV%no_nu_mu
2228                z=(0.5_dl*dgrho/k
2229                dz= -adotoa*z - 0
2230                clxg=-4*dz/k-4/k*
2231                qg=-4._dl/3*z
2232            else
2233                clxg=clxr-4/k*opa
2234                qg=qr
2235            end if
2236            pig=0
2237        else
2238            !  Photons
2239            clxg=ay(EV%g_ix)
2240            qg=ay(EV%g_ix+1)
2241            if (.not. EV%TightCou
2242        end if
2243
2244        !  8*pi*a*a*SUM[rho_i*clx
2245        dgrho=dgrho + grhog_t*clx
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2231

```
2231 |
2232 |        !  8*pi*a*a*SUM[(rho_i+p_
2233 |        dgq=dgq + grhog_t*qg+grho
2234 |
2235 |        !  Photon mass density ov
2236 |        photbar=grhog_t/grhob_t
2237 |        pb43=4._dl/3*photbar
2238 |
2239 |        ayprime(1)=adotoa*a
2240 |
2241 |        if (.not. is_cosmological
2242 |            !ppf
2243 |            grhoT = grho - grhov_
2244 |            vT= dgq/(grhoT+gpres)
2245 |            Gamma=ay(EV%w_ix)
2246 |
2247 |            !sigma for ppf
2248 |            sigma = (etak + (dgrh
2249 |            sigma = sigma/adotoa
2250 |
2251 |            S_Gamma=grhov_t*(1+w_
2252 |            ckH=c_Gamma_ppf*k/ado
2253 |            Gammadot=S_Gamma/(1+c
2254 |            Gammadot=Gammadot*ado
2255 |            ayprime(EV%w_ix)=Gamm
2256 |
2257 |            if(ckH*ckH.gt.3.d1)th
2258 |                Gamma=0
2259 |                Gammadot=0.d0
2260 |                ayprime(EV%w_ix)=
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2246

```
2246 |
2247 |        !  8*pi*a*a*SUM[(rho_i+p_
2248 |        dgq=dgq + grhog_t*qg+grho
2249 |
2250 |        !  Photon mass density ov
2251 |        photbar=grhog_t/grhob_t
2252 |        pb43=4._dl/3*photbar
2253 |
2254 |        ayprime(1)=adotoa*a
2255 |
2256 |        if (.not. is_cosmological
2257 |            !ppf
2258 |            grhoT = grho - grhov_
2259 |            vT= dgq/(grhoT+gpres)
2260 |            Gamma=ay(EV%w_ix)
2261 |
2262 |            !sigma for ppf
2263 |            sigma = (etak + (dgrh
2264 |            sigma = sigma/adotoa
2265 |
2266 |            S_Gamma=grhov_t*(1+w_
2267 |            ckH=c_Gamma_ppf*k/ado
2268 |            Gammadot=S_Gamma/(1+c
2269 |            Gammadot=Gammadot*ado
2270 |            ayprime(EV%w_ix)=Gamm
2271 |
2272 |            if(ckH*ckH.gt.3.d1)th
2273 |                Gamma=0
2274 |                Gammadot=0.d0
2275 |                ayprime(EV%w_ix)=
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2261

```
2261            endif
2262
2263            Fa=1+3*(grhoT+gpres)/
2264            dgqe=S_Gamma - Gammad
2265            dgqe=-dgqe/Fa*2._dl*k
2266            dgrhoe=-2*k2*EV%kf(1)
2267            dgrho=dgrho+dgrhoe
2268            dgq=dgq+dgqe
2269
2270            EV%dgrho_e_ppf=dgrhoe
2271            EV%dgq_e_ppf=dgqe
2272        end if
2273
2274        !  Get sigma (shear) and
2275        ! have to get z from eta
2276        z=(0.5_dl*dgrho/k + etak)
2277        if (CP%flat) then
2278            !eta*k equation
2279            sigma=(z+1.5_dl*dgq/k
2280            ayprime(2)=0.5_dl*dgq
2281        else
2282            sigma=(z+1.5_dl*dgq/k
2283            ayprime(2)=0.5_dl*dgq
2284        end if
2285
2286        !if (w_lam /= -1 .and. w_
2287        !
2288        !   ayprime(EV%w_ix)= -3*
2289        !       -(1+w_lam)*k*vq -
2290        !
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2276

```
2276            endif
2277
2278            Fa=1+3*(grhoT+gpres)/
2279            dgqe=S_Gamma - Gammad
2280            dgqe=-dgqe/Fa*2._dl*k
2281            dgrhoe=-2*k2*EV%kf(1)
2282            dgrho=dgrho+dgrhoe
2283            dgq=dgq+dgqe
2284
2285            EV%dgrho_e_ppf=dgrhoe
2286            EV%dgq_e_ppf=dgqe
2287        end if
2288
2289        !  Get sigma (shear) and
2290        ! have to get z from eta
2291        z=(0.5_dl*dgrho/k + etak)
2292        if (CP%flat) then
2293            !eta*k equation
2294            sigma=(z+1.5_dl*dgq/k
2295            ayprime(2)=0.5_dl*dgq
2296        else
2297            sigma=(z+1.5_dl*dgq/k
2298            ayprime(2)=0.5_dl*dgq
2299        end if
2300
2301        !if (w_lam /= -1 .and. w_
2302        !
2303        !   ayprime(EV%w_ix)= -3*
2304        !       -(1+w_lam)*k*vq -
2305        !
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2291

```
2291          !   ayprime(EV%w_ix+1) =
2292          !
2293          !end if
2294          !
2295
2296          if (associated(EV%OutputT
2297              EV%OutputTransfer(Tra
2298              EV%OutputTransfer(Tra
2299              EV%OutputTransfer(Tra
2300              EV%OutputTransfer(Tra
2301              EV%OutputTransfer(Tra
2302              clxnu_all=0
2303              dgpi  = grhor_t*pir +
2304              if (CP%Num_Nu_Massive
2305                  call MassiveNuVar
2306              end if
2307              EV%OutputTransfer(Tra
2308              EV%OutputTransfer(Tra
2309              EV%OutputTransfer(Tra
2310              EV%OutputTransfer(Tra
2311              !Transfer_Weyl is k^2
2312              EV%OutputTransfer(Tra
2313              EV%OutputTransfer(Tra
2314              EV%OutputTransfer(Tra
2315              EV%OutputTransfer(Tra
2316          end if
2317
2318          !  CDM equation of motion
2319          clxcdot=-k*z
2320          ayprime(3)=clxcdot
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2306

```
2306          !   ayprime(EV%w_ix+1) =
2307          !
2308          !end if
2309          !
2310
2311          if (associated(EV%OutputT
2312              EV%OutputTransfer(Tra
2313              EV%OutputTransfer(Tra
2314              EV%OutputTransfer(Tra
2315              EV%OutputTransfer(Tra
2316              EV%OutputTransfer(Tra
2317              clxnu_all=0
2318              dgpi  = grhor_t*pir +
2319              if (CP%Num_Nu_Massive
2320                  call MassiveNuVar
2321              end if
2322              EV%OutputTransfer(Tra
2323              EV%OutputTransfer(Tra
2324              EV%OutputTransfer(Tra
2325              EV%OutputTransfer(Tra
2326              !Transfer_Weyl is k^2
2327              EV%OutputTransfer(Tra
2328              EV%OutputTransfer(Tra
2329              EV%OutputTransfer(Tra
2330              EV%OutputTransfer(Tra
2331          end if
2332
2333          !  CDM equation of motion
2334          clxcdot=-k*z
2335          ayprime(3)=clxcdot
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2321

```
2321 |
2322 |       !  Baryon equation of mot
2323 |       clxbdot=-k*(z+vb)
2324 |       ayprime(4)=clxbdot
2325 |       !  Photon equation of mot
2326 |       clxgdot=-k*(4._dl/3._dl*z
2327 |
2328 |       ! old comment:Small k: po
2329 |       ! Easy to see instability
2330 |
2331 |       !  Use explicit equation
2332 |
2333 |       if (EV%TightCoupling) the
2334 |           !  ddota/a
2335 |           gpres=gpres + grhov_t
2336 |           adotdota=(adotoa*adot
2337 |
2338 |           pig = 32._dl/45/opaci
2339 |
2340 |           !  First-order approx
2341 |           slip = - (2*adotoa/(1
2342 |           +(-adotdota*vb-k/2*ad
2343 |
2344 |           if (second_order_tigh
2345 |               ! by Francis-Yan
2346 |               !AL: First order
2347 |
2348 |               !  8*pi*G*a*a*SUM
2349 |               dgs = grhog_t*pig
2350 |
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2336

```
2336 |
2337 |       !  Baryon equation of mot
2338 |       clxbdot=-k*(z+vb)
2339 |       ayprime(4)=clxbdot
2340 |       !  Photon equation of mot
2341 |       clxgdot=-k*(4._dl/3._dl*z
2342 |
2343 |       ! old comment:Small k: po
2344 |       ! Easy to see instability
2345 |
2346 |       !  Use explicit equation
2347 |
2348 |       if (EV%TightCoupling) the
2349 |           !  ddota/a
2350 |           gpres=gpres + grhov_t
2351 |           adotdota=(adotoa*adot
2352 |
2353 |           pig = 32._dl/45/opaci
2354 |
2355 |           !  First-order approx
2356 |           slip = - (2*adotoa/(1
2357 |               +(-adotdota*vb-k/
2358 |
2359 |           if (second_order_tigh
2360 |               ! by Francis-Yan
2361 |               !AL: First order
2362 |
2363 |               !  8*pi*G*a*a*SUM
2364 |               dgs = grhog_t*pig
2365 |
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2351

```
2351                ! Define shear de
2352                sigmadot = -2*ado
2353
2354                !Once know slip,
2355                qgdot = k*(clxg/4
2356
2357                pig = 32._dl/45/o
2358                + (32._dl/45._dl/
2359
2360                pigdot = -(32._dl
2361                dopacity*11._dl/6
2362                + (32._dl/45._dl/
2363                *(dopacity/opacit
2364
2365                EV%pigdot = pigdo
2366            end if
2367
2368            !  Use tight-coupling
2369            !  zeroth order appro
2370            vbdot=(-adotoa*vb+cs2
2371            +k/4*pb43*(clxg-2*EV%
2372
2373            vbdot=vbdot+pb43/(1+p
2374
2375            EV%pig = pig
2376        else
2377            vbdot=-adotoa*vb+cs2*
2378        end if
2379
2380        ayprime(5)=vbdot
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2366

```
2366                ! Define shear de
2367                sigmadot = -2*ado
2368
2369                !Once know slip,
2370                qgdot = k*(clxg/4
2371
2372                pig = 32._dl/45/o
2373                    + (32._dl/45.
2374
2375                pigdot = -(32._dl
2376                    dopacity*11._
2377                    + (32._dl/45.
2378                    *(dopacity/op
2379
2380                EV%pigdot = pigdo
2381            end if
2382
2383            !  Use tight-coupling
2384            !  zeroth order appro
2385            vbdot=(-adotoa*vb+cs2
2386                +k/4*pb43*(clxg-2
2387
2388            vbdot=vbdot+pb43/(1+p
2389
2390            EV%pig = pig
2391        else
2392            vbdot=-adotoa*vb+cs2*
2393        end if
2394
2395        ayprime(5)=vbdot
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2381

```
2381 |
2382 |       if (.not. EV%no_phot_mult
2383 |           !  Photon equations o
2384 |           ayprime(EV%g_ix)=clxg
2385 |           qgdot=4._dl/3*(-vbdot
2386 |           +EV%denlk(1)*clxg-EV%
2387 |           ayprime(EV%g_ix+1)=qg
2388 |
2389 |           !  Use explicit equat
2390 |           if (.not. EV%tightcou
2391 |               E2=ay(EV%polind+2
2392 |               polter = pig/10+9
2393 |               ix= EV%g_ix+2
2394 |               if (EV%lmaxg>2) t
2395 |                   pigdot=EV%den
2396 |                   +8._dl/15._dl
2397 |                   ayprime(ix)=p
2398 |                   do  l=3,EV%lm
2399 |                       ix=ix+1
2400 |                       ayprime(i
2401 |                   end do
2402 |                   ix=ix+1
2403 |                   !  Truncate t
2404 |                   ayprime(ix)=k
2405 |               else !closed case
2406 |                   pigdot=EV%den
2407 |                   ayprime(ix)=p
2408 |               endif
2409 |               !  Polarization
2410 |               !l=2
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2396

```
2396 |
2397 |       if (.not. EV%no_phot_mult
2398 |           !  Photon equations o
2399 |           ayprime(EV%g_ix)=clxg
2400 |           qgdot=4._dl/3*(-vbdot
2401 |               +EV%denlk(1)*clxg
2402 |           ayprime(EV%g_ix+1)=qg
2403 |
2404 |           !  Use explicit equat
2405 |           if (.not. EV%tightcou
2406 |               E2=ay(EV%polind+2
2407 |               polter = pig/10+9
2408 |               ix= EV%g_ix+2
2409 |               if (EV%lmaxg>2) t
2410 |                   pigdot=EV%den
2411 |                       +8._dl/15
2412 |                   ayprime(ix)=p
2413 |                   do  l=3,EV%lm
2414 |                       ix=ix+1
2415 |                       ayprime(i
2416 |                   end do
2417 |                   ix=ix+1
2418 |                   !  Truncate t
2419 |                   ayprime(ix)=k
2420 |               else !closed case
2421 |                   pigdot=EV%den
2422 |                   ayprime(ix)=p
2423 |               endif
2424 |               !  Polarization
2425 |               !l=2
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2411

```
2411                ix=EV%polind+2
2412                if (EV%lmaxgpol>2
2413                    ayprime(ix) =
2414                    do l=3,EV%lma
2415                        ix=ix+1
2416                        ayprime(i
2417                    end do
2418                    ix=ix+1
2419                    !truncate
2420                    ayprime(ix)=-
2421                    k*EV%poltrunc
2422                else !closed case
2423                    ayprime(ix) =
2424                endif
2425            end if
2426        end if
2427
2428        if (.not. EV%no_nu_multpo
2429            !  Massless neutrino
2430            clxrdot=-k*(4._dl/3._
2431            ayprime(EV%r_ix)=clxr
2432            qrdot=EV%denlk(1)*clx
2433            ayprime(EV%r_ix+1)=qr
2434            if (EV%high_ktau_neut
2435                !ufa approximatio
2436                !Method from arXi
2437                !
2438                !
2439                !
2440                !
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2426

```
2426                ix=EV%polind+2
2427                if (EV%lmaxgpol>2
2428                    ayprime(ix) =
2429                    do l=3,EV%lma
2430                        ix=ix+1
2431                        ayprime(i
2432                    end do
2433                    ix=ix+1
2434                    !truncate
2435                    ayprime(ix)=-
2436                        k*EV%polt
2437                else !closed case
2438                    ayprime(ix) =
2439                endif
2440            end if
2441        end if
2442
2443        if (.not. EV%no_nu_multpo
2444            !  Massless neutrino
2445            clxrdot=-k*(4._dl/3._
2446            ayprime(EV%r_ix)=clxr
2447            qrdot=EV%denlk(1)*clx
2448            ayprime(EV%r_ix+1)=qr
2449            if (EV%high_ktau_neut
2450                !ufa approximatio
2451                !Method from arXi
2452                !
2453                !
2454                !
2455                !
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2441

```
2441 |                    !
2442 |                    !
2443 |                    !
2444 |                    !
2445 |                    !
2446 |                    pirdot= -3*pir*co
2447 |                    ayprime(EV%r_ix+2
2448 |
2449 |                    !
2450 |                    !
2451 |                    !
2452 |                    !
2453 |                    !                  a
2454 |                else
2455 |                    ix=EV%r_ix+2
2456 |                    if (EV%lmaxnr>2)
2457 |                        pirdot=EV%den
2458 |                        ayprime(ix)=p
2459 |                        do l=3,EV%lma
2460 |                            ix=ix+1
2461 |                            ayprime(i
2462 |                        end do
2463 |                        !  Truncate t
2464 |                        ix=ix+1
2465 |                        ayprime(ix)=k
2466 |                    else
2467 |                        pirdot=EV%den
2468 |                        ayprime(ix)=p
2469 |                    end if
2470 |                end if
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2456

```
2456 |                    !
2457 |                    !
2458 |                    !
2459 |                    !
2460 |                    !
2461 |                    pirdot= -3*pir*co
2462 |                    ayprime(EV%r_ix+2
2463 |
2464 |                    !
2465 |                    !
2466 |                    !
2467 |                    !
2468 |                    !                  a
2469 |                else
2470 |                    ix=EV%r_ix+2
2471 |                    if (EV%lmaxnr>2)
2472 |                        pirdot=EV%den
2473 |                        ayprime(ix)=p
2474 |                        do l=3,EV%lma
2475 |                            ix=ix+1
2476 |                            ayprime(i
2477 |                        end do
2478 |                        !  Truncate t
2479 |                        ix=ix+1
2480 |                        ayprime(ix)=k
2481 |                    else
2482 |                        pirdot=EV%den
2483 |                        ayprime(ix)=p
2484 |                    end if
2485 |                end if
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2471

```
2471        end if ! no_nu_multpoles
2472
2473        !  Massive neutrino equat
2474        if (CP%Num_Nu_massive ==
2475
2476        do nu_i = 1, CP%Nu_mass_e
2477            if (EV%MassiveNuAppro
2478                !Now EV%iq0 = clx
2479                !see astro-ph/020
2480                G11_t=EV%G11(nu_i
2481                G30_t=EV%G30(nu_i
2482                off_ix = EV%nu_ix
2483                w=wnu_arr(nu_i)
2484                ayprime(off_ix)=-
2485                ayprime(off_ix+1)
2486                ayprime(off_ix+2)
2487                ayprime(off_ix+3)
2488            else
2489                ind=EV%nu_ix(nu_i
2490                do i=1,EV%nq(nu_i
2491                    q=nu_q(i)
2492                    aq=a*nu_masse
2493                    v=1._dl/sqrt(
2494
2495                    ayprime(ind)=
2496                    ind=ind+1
2497                    ayprime(ind)=
2498                    ind=ind+1
```

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

```
2486        end if ! no_nu_multpoles
2487
2488        !  Massive neutrino equat
2489        if (CP%Num_Nu_massive ==
2490
2491        !DIR$ LOOP COUNT MIN(1),
2492        do nu_i = 1, CP%Nu_mass_e
2493            if (EV%MassiveNuAppro
2494                !Now EV%iq0 = clx
2495                !see astro-ph/020
2496                G11_t=EV%G11(nu_i
2497                G30_t=EV%G30(nu_i
2498                off_ix = EV%nu_ix
2499                w=wnu_arr(nu_i)
2500                ayprime(off_ix)=-
2501                ayprime(off_ix+1)
2502                ayprime(off_ix+2)
2503                ayprime(off_ix+3)
2504            else
2505                ind=EV%nu_ix(nu_i
2506                !DIR$ LOOP COUNT
2507                do i=1,EV%nq(nu_i
2508                    q=nu_q(i)
2509                    aq=a*nu_masse
2510                    v=1._dl/sqrt(
2511
2512                    ayprime(ind)=
2513                    ind=ind+1
2514                    ayprime(ind)=
2515                    ind=ind+1
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2499

```
2499                    if (EV%lmaxnu
2500                        ayprime(i
2501                    else
2502                        ayprime(i
2503                        +k*8._dl/
2504                        do l=3,EV
2505                            ind=i
2506                            aypri
2507                        end do
2508                        !  Trunca
2509                        ind = ind
2510                        ayprime(i
2511                    end if
2512                    ind = ind+1
2513                end do
2514            end if
2515        end do
2516
2517        if (EV%has_nu_relativisti
2518            ind=EV%nu_pert_ix
2519            ayprime(ind)=+k*a2*qr
2520            ind2= EV%r_ix
2521            do l=1,EV%lmaxnu_pert
2522                ind=ind+1
2523                ind2=ind2+1
2524                ayprime(ind)= -a2
2525                +   (EV%denlk(l)*
2526            end do
2527            ind=ind+1
2528            ind2=ind2+1
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2516

```
2516                    if (EV%lmaxnu
2517                        ayprime(i
2518                    else
2519                        ayprime(i
2520                            +k*8.
2521                        do l=3,EV
2522                            ind=i
2523                            aypri
2524                        end do
2525                        !  Trunca
2526                        ind = ind
2527                        ayprime(i
2528                    end if
2529                    ind = ind+1
2530                end do
2531            end if
2532        end do
2533
2534        if (EV%has_nu_relativisti
2535            ind=EV%nu_pert_ix
2536            ayprime(ind)=+k*a2*qr
2537            ind2= EV%r_ix
2538            do l=1,EV%lmaxnu_pert
2539                ind=ind+1
2540                ind2=ind2+1
2541                ayprime(ind)= -a2
2542                    +   (EV%denlk
2543            end do
2544            ind=ind+1
2545            ind2=ind2+1
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2529

```
2529 |            ayprime(ind)= k*(ay(i
2530 |        end if
2531 |
2532 |        end subroutine derivs
2533 |
2534 |
2535 |
2536 |        subroutine derivsv(EV,n,t
2537 |        !  Evaluate the time deri
2538 |        use ThermoData
2539 |        use MassiveNu
2540 |        implicit none
2541 |        type(EvolutionVars) EV
2542 |        integer n,l
2543 |        real(dl), target ::  yv(n
2544 |        real(dl) ep,tau,grho,rhop
2545 |        logical finished_tightcou
2546 |        real(dl), dimension(:),po
2547 |        real(dl)  grhob_t,grhor_t
2548 |        real(dl) sigma, qg,pig, q
2549 |        real(dl) k,k2,a,a2, adotd
2550 |        real(dl) pir,adotoa
2551 |
2552 |        stop 'ppf not implemented
2553 |
2554 |        k2=EV%k2_buf
2555 |        k=EV%k_buf
2556 |
2557 |        !E and B start at l=2. Se
2558 |        E => yv(EV%lmaxv+3:)
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2546

```
2546 |            ayprime(ind)= k*(ay(i
2547 |        end if
2548 |
2549 |        end subroutine derivs
2550 |
2551 |
2552 |
2553 |        subroutine derivsv(EV,n,t
2554 |        !  Evaluate the time deri
2555 |        use ThermoData
2556 |        use MassiveNu
2557 |        implicit none
2558 |        type(EvolutionVars) EV
2559 |        integer n,l
2560 |        real(dl), target ::  yv(n
2561 |        real(dl) ep,tau,grho,rhop
2562 |        logical finished_tightcou
2563 |        real(dl), dimension(:),po
2564 |        real(dl)  grhob_t,grhor_t
2565 |        real(dl) sigma, qg,pig, q
2566 |        real(dl) k,k2,a,a2, adotd
2567 |        real(dl) pir,adotoa
2568 |
2569 |        stop 'ppf not implemented
2570 |
2571 |        k2=EV%k2_buf
2572 |        k=EV%k_buf
2573 |
2574 |        !E and B start at l=2. Se
2575 |        E => yv(EV%lmaxv+3:)
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2559

```
2559        Eprime=> yvprime(EV%lmaxv
2560        B => E(EV%lmaxpolv:)
2561        Bprime => Eprime(EV%lmaxp
2562        neutprime => Bprime(EV%lm
2563        neut => B(EV%lmaxpolv+1:)
2564
2565        a=yv(1)
2566
2567        sigma=yv(2)
2568
2569        a2=a*a
2570
2571        !  Get sound speed and op
2572
2573        call thermo(tau,cs2,opaci
2574        if (k > 0.06_dl*epsw) the
2575            ep=ep0
2576        else
2577            ep=0.2_dl*ep0
2578        end if
2579
2580        finished_tightcoupling =
2581        ((k/opacity > ep).or.(1._
2582
2583
2584        ! Compute expansion rate
2585        ! Also calculate gpres: 8
2586        grhob_t=grhob/a
2587        grhoc_t=grhoc/a
2588        grhor_t=grhornomass/a2
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2576

```
2576        Eprime=> yvprime(EV%lmaxv
2577        B => E(EV%lmaxpolv:)
2578        Bprime => Eprime(EV%lmaxp
2579        neutprime => Bprime(EV%lm
2580        neut => B(EV%lmaxpolv+1:)
2581
2582        a=yv(1)
2583
2584        sigma=yv(2)
2585
2586        a2=a*a
2587
2588        !  Get sound speed and op
2589
2590        call thermo(tau,cs2,opaci
2591        if (k > 0.06_dl*epsw) the
2592            ep=ep0
2593        else
2594            ep=0.2_dl*ep0
2595        end if
2596
2597        finished_tightcoupling =
2598            ((k/opacity > ep).or.
2599
2600
2601        ! Compute expansion rate
2602        ! Also calculate gpres: 8
2603        grhob_t=grhob/a
2604        grhoc_t=grhoc/a
2605        grhor_t=grhornomass/a2
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2589

```
2589        grhog_t=grhog/a2
2590        grhov_t=grhov*a**(-1-3*w_
2591
2592        grho=grhob_t+grhoc_t+grho
2593        gpres=(grhog_t+grhor_t)/3
2594
2595        adotoa=sqrt(grho/3._dl)
2596        adotdota=(adotoa*adotoa-g
2597
2598        photbar=grhog_t/grhob_t
2599        pb43=4._dl/3*photbar
2600
2601        yvprime(1)=adotoa*a
2602
2603        vb = yv(3)
2604        qg = yv(4)
2605        qr = neut(1)
2606
2607        !  8*pi*a*a*SUM[(rho_i+p_
2608        rhoq=grhob_t*vb+grhog_t*q
2609        !  sigma = 2*rhoq/k**2
2610        !for non-large k this exp
2611        !so propagate sigma equat
2612        ! print *,yv(2),2*rhoq/k*
2613
2614        if (finished_tightcouplin
2615            !  Use explicit equat
2616
2617            pig = yv(5)
2618
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2606

```
2606        grhog_t=grhog/a2
2607        grhov_t=grhov*a**(-1-3*w_
2608
2609        grho=grhob_t+grhoc_t+grho
2610        gpres=(grhog_t+grhor_t)/3
2611
2612        adotoa=sqrt(grho/3._dl)
2613        adotdota=(adotoa*adotoa-g
2614
2615        photbar=grhog_t/grhob_t
2616        pb43=4._dl/3*photbar
2617
2618        yvprime(1)=adotoa*a
2619
2620        vb = yv(3)
2621        qg = yv(4)
2622        qr = neut(1)
2623
2624        !  8*pi*a*a*SUM[(rho_i+p_
2625        rhoq=grhob_t*vb+grhog_t*q
2626        !  sigma = 2*rhoq/k**2
2627        !for non-large k this exp
2628        !so propagate sigma equat
2629        ! print *,yv(2),2*rhoq/k*
2630
2631        if (finished_tightcouplin
2632            !  Use explicit equat
2633
2634            pig = yv(5)
2635
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2619

```
2619            polter = 0.1_dl*pig +
2620
2621            vbdot = -adotoa*vb-ph
2622
2623            !  Equation for the p
2624
2625            yvprime(4)=-0.5_dl*k*
2626
2627            !  Equation for the p
2628            yvprime(5)=k*(2._dl/5
2629            -opacity*(pig - polte
2630            ! And for the moments
2631            do  l=3,EV%lmaxv-1
2632                yvprime(l+3)=k*de
2633                vecfac(l)*yv(l+4)
2634            end do
2635            !  Truncate the hiera
2636            yvprime(EV%lmaxv+3)=k
2637            (EV%lmaxv+2._dl)*yv(E
2638
2639            !E equations
2640
2641            Eprime(2) = - opacity
2642            do l=3,EV%lmaxpolv-1
2643                Eprime(l) =-opaci
2644                vecfacpol(l)*E(l+
2645            end do
2646            !truncate
2647            Eprime(EV%lmaxpolv)=0
2648
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2636

```
2636            polter = 0.1_dl*pig +
2637
2638            vbdot = -adotoa*vb-ph
2639
2640            !  Equation for the p
2641
2642            yvprime(4)=-0.5_dl*k*
2643
2644            !  Equation for the p
2645            yvprime(5)=k*(2._dl/5
2646                -opacity*(pig - p
2647            ! And for the moments
2648            do  l=3,EV%lmaxv-1
2649                yvprime(l+3)=k*de
2650                    vecfac(l)*yv(
2651            end do
2652            !  Truncate the hiera
2653            yvprime(EV%lmaxv+3)=k
2654                (EV%lmaxv+2._dl)*
2655
2656            !E equations
2657
2658            Eprime(2) = - opacity
2659            do l=3,EV%lmaxpolv-1
2660                Eprime(l) =-opaci
2661                    vecfacpol(l)*
2662            end do
2663            !truncate
2664            Eprime(EV%lmaxpolv)=0
2665
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2649

```
2649            !B-bar equations
2650
2651            do l=2,EV%lmaxpolv-1
2652                Bprime(l) =-opaci
2653                vecfacpol(l)*B(l+
2654            end do
2655            !truncate
2656            Bprime(EV%lmaxpolv)=0
2657        else
2658            !Tight coupling expan
2659
2660            pig = 32._dl/45._dl*k
2661
2662            EV%pig = pig
2663
2664            vbdot=(-adotoa*vb  -3
2665            - pb43/(1+pb43)/opaci
2666            ( 2*pb43*adotoa**2/(1
2667            )/(1+pb43)
2668
2669            !  Equation for the p
2670            ! Get drag from vbdot
2671            yvprime(4)=-0.5_dl*k*
2672            (vbdot+adotoa*vb)/pho
2673
2674            !  Set the derivative
2675            yvprime(5:n)=0._dl
2676            yv(5)=pig
2677            E(2)=  pig/4
2678        endif
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2666

```
2666            !B-bar equations
2667
2668            do l=2,EV%lmaxpolv-1
2669                Bprime(l) =-opaci
2670                    vecfacpol(l)*
2671            end do
2672            !truncate
2673            Bprime(EV%lmaxpolv)=0
2674        else
2675            !Tight coupling expan
2676
2677            pig = 32._dl/45._dl*k
2678
2679            EV%pig = pig
2680
2681            vbdot=(-adotoa*vb  -3
2682                - pb43/(1+pb43)/o
2683                ( 2*pb43*adotoa**
2684                )/(1+pb43)
2685
2686            !  Equation for the p
2687            ! Get drag from vbdot
2688            yvprime(4)=-0.5_dl*k*
2689                (vbdot+adotoa*vb)
2690
2691            !  Set the derivative
2692            yvprime(5:n)=0._dl
2693            yv(5)=pig
2694            E(2)=  pig/4
2695        endif
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2679

```
2679
2680        yvprime(3) = vbdot
2681
2682        !  Neutrino equations:
2683
2684        !  Massless neutrino anis
2685        pir=neut(2)
2686        neutprime(1)= -0.5_dl*k*p
2687        neutprime(2)=2._dl/5*k*qr
2688        !  And for the moments
2689        do  l=3,EV%lmaxnrv-1
2690            neutprime(l)=k*denl(l
2691        end do
2692
2693        !  Truncate the hierarchy
2694        neutprime(EV%lmaxnrv)=k*E
2695        (EV%lmaxnrv+2._dl)*neut(E
2696
2697
2698        !  Get the propagation eq
2699
2700        rhopi=grhog_t*pig+grhor_t
2701
2702        yvprime(2)=-2*adotoa*sigm
2703
2704        end subroutine derivsv
2705
2706
2707
2708        !ccccccccccccccccccccccc
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2696

```
2696
2697        yvprime(3) = vbdot
2698
2699        !  Neutrino equations:
2700
2701        !  Massless neutrino anis
2702        pir=neut(2)
2703        neutprime(1)= -0.5_dl*k*p
2704        neutprime(2)=2._dl/5*k*qr
2705        !  And for the moments
2706        do  l=3,EV%lmaxnrv-1
2707            neutprime(l)=k*denl(l
2708        end do
2709
2710        !  Truncate the hierarchy
2711        neutprime(EV%lmaxnrv)=k*E
2712            (EV%lmaxnrv+2._dl)*ne
2713
2714
2715        !  Get the propagation eq
2716
2717        rhopi=grhog_t*pig+grhor_t
2718
2719        yvprime(2)=-2*adotoa*sigm
2720
2721        end subroutine derivsv
2722
2723
2724
2725        !ccccccccccccccccccccccc
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2709

```
2709        subroutine derivst(EV,n,t
2710        !  Evaluate the time deri
2711        use ThermoData
2712        use MassiveNu
2713        implicit none
2714        type(EvolutionVars) EV
2715        integer n,l,i,ind, nu_i
2716        real(dl), target ::  ayt(
2717        real(dl) tau,grho,rhopi,c
2718        real(dl), dimension(:),po
2719        real(dl) q,aq,v
2720        real(dl)  grhob_t,grhor_t
2721        real(dl) Hchi,pinu, pig
2722        real(dl) k,k2,a,a2
2723        real(dl) pir, adotoa, rho
2724
2725        real(dl) cothxor
2726
2727        k2=EV%k2_buf
2728        k= EV%k_buf
2729
2730        a=ayt(1)
2731
2732        Hchi=ayt(2)
2733
2734        shear=ayt(3)
2735
2736        a2=a*a
2737
2738        ! Compute expansion rate
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2726

```
2726        subroutine derivst(EV,n,t
2727        !  Evaluate the time deri
2728        use ThermoData
2729        use MassiveNu
2730        implicit none
2731        type(EvolutionVars) EV
2732        integer n,l,i,ind, nu_i
2733        real(dl), target ::  ayt(
2734        real(dl) tau,grho,rhopi,c
2735        real(dl), dimension(:),po
2736        real(dl) q,aq,v
2737        real(dl)  grhob_t,grhor_t
2738        real(dl) Hchi,pinu, pig
2739        real(dl) k,k2,a,a2
2740        real(dl) pir, adotoa, rho
2741
2742        real(dl) cothxor
2743
2744        k2=EV%k2_buf
2745        k= EV%k_buf
2746
2747        a=ayt(1)
2748
2749        Hchi=ayt(2)
2750
2751        shear=ayt(3)
2752
2753        a2=a*a
2754
2755        ! Compute expansion rate
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2739

```
2739        ! Also calculate gpres: 8
2740        grhob_t=grhob/a
2741        grhoc_t=grhoc/a
2742        grhor_t=grhornomass/a2
2743        grhog_t=grhog/a2
2744        if (is_cosmological_const
2745            grhov_t=grhov*a2
2746        else
2747            grhov_t=grho_de(a)/a2
2748        end if
2749
2750        grho=grhob_t+grhoc_t+grho
2751
2752        !Do massive neutrinos
2753        if (CP%Num_Nu_Massive >0)
2754            do nu_i=1,CP%Nu_mass_
2755                call Nu_rho(a*nu_
2756                grho=grho+grhorma
2757            end do
2758        end if
2759
2760        if (CP%flat) then
2761            cothxor=1._dl/tau
2762            adotoa=sqrt(grho/3._d
2763        else
2764            cothxor=1._dl/tanfunc
2765            adotoa=sqrt((grho+grh
2766        end if
2767
2768        aytprime(1)=adotoa*a
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2756

```
2756        ! Also calculate gpres: 8
2757        grhob_t=grhob/a
2758        grhoc_t=grhoc/a
2759        grhor_t=grhornomass/a2
2760        grhog_t=grhog/a2
2761        if (is_cosmological_const
2762            grhov_t=grhov*a2
2763        else
2764            grhov_t=grho_de(a)/a2
2765        end if
2766
2767        grho=grhob_t+grhoc_t+grho
2768
2769        !Do massive neutrinos
2770        if (CP%Num_Nu_Massive >0)
2771            do nu_i=1,CP%Nu_mass_
2772                call Nu_rho(a*nu_
2773                grho=grho+grhorma
2774            end do
2775        end if
2776
2777        if (CP%flat) then
2778            cothxor=1._dl/tau
2779            adotoa=sqrt(grho/3._d
2780        else
2781            cothxor=1._dl/tanfunc
2782            adotoa=sqrt((grho+grh
2783        end if
2784
2785        aytprime(1)=adotoa*a
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2769

```
2769 |
2770 |       call thermo(tau,cs2,opaci
2771 |
2772 |       if (.not. EV%TensTightCou
2773 |           !  Don't use tight co
2774 |           !  Equation for the p
2775 |
2776 |
2777 |           !E and B start at l=2
2778 |           E => ayt(EV%E_ix+1:)
2779 |           B => ayt(EV%B_ix+1:)
2780 |           Eprime=> aytprime(EV%
2781 |           Bprime => aytprime(EV
2782 |
2783 |           ind = EV%g_ix+2
2784 |
2785 |           !  Photon anisotropic
2786 |           pig=ayt(ind)
2787 |           polter = 0.1_dl*pig +
2788 |
2789 |           if (EV%lmaxt > 2) the
2790 |               aytprime(ind)=-EV
2791 |               -opacity*(pig - p
2792 |
2793 |               do l=3, EV%lmaxt
2794 |                   ind = ind+1
2795 |                   aytprime(ind)
2796 |               end do
2797 |
2798 |               !Truncate the hie
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2786

```
2786 |
2787 |       call thermo(tau,cs2,opaci
2788 |
2789 |       if (.not. EV%TensTightCou
2790 |           !  Don't use tight co
2791 |           !  Equation for the p
2792 |
2793 |
2794 |           !E and B start at l=2
2795 |           E => ayt(EV%E_ix+1:)
2796 |           B => ayt(EV%B_ix+1:)
2797 |           Eprime=> aytprime(EV%
2798 |           Bprime => aytprime(EV
2799 |
2800 |           ind = EV%g_ix+2
2801 |
2802 |           !  Photon anisotropic
2803 |           pig=ayt(ind)
2804 |           polter = 0.1_dl*pig +
2805 |
2806 |           if (EV%lmaxt > 2) the
2807 |               aytprime(ind)=-EV
2808 |                   -opacity*(pig
2809 |
2810 |               do l=3, EV%lmaxt
2811 |                   ind = ind+1
2812 |                   aytprime(ind)
2813 |               end do
2814 |
2815 |               !Truncate the hie
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2799

```
2799                    ind=ind+1
2800                    aytprime(ind)=k*E
2801                    (EV%lmaxt+3._dl)*
2802
2803                    !E and B-bar equa
2804
2805                    Eprime(2) = - opa
2806                    EV%denlkt(3,2)*E(
2807
2808                    do l=3, EV%lmaxpo
2809                        Eprime(l) =(E
2810                        -opacity*E(l)
2811                    end do
2812                    l= EV%lmaxpolt
2813                    !truncate: diffic
2814                    Eprime(l) = (EV%d
2815
2816                    Bprime(2) =-EV%de
2817                    do l=3, EV%lmaxpo
2818                        Bprime(l) =(E
2819                        -opacity*B(l)
2820                    end do
2821                    l=EV%lmaxpolt
2822                    !truncate
2823                    Bprime(l) =(EV%de
2824
2825                else !lmax=2
2826
2827                aytprime(ind)=k*8._dl
2828                Eprime(2) = - opacity
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2816

```
2816                    ind=ind+1
2817                    aytprime(ind)=k*E
2818                        (EV%lmaxt+3._
2819
2820                    !E and B-bar equa
2821
2822                    Eprime(2) = - opa
2823                        EV%denlkt(3,2
2824
2825                    do l=3, EV%lmaxpo
2826                        Eprime(l) =(E
2827                            -opacity*
2828                    end do
2829                    l= EV%lmaxpolt
2830                    !truncate: diffic
2831                    Eprime(l) = (EV%d
2832
2833                    Bprime(2) =-EV%de
2834                    do l=3, EV%lmaxpo
2835                        Bprime(l) =(E
2836                            -opacity*
2837                    end do
2838                    l=EV%lmaxpolt
2839                    !truncate
2840                    Bprime(l) =(EV%de
2841
2842                else !lmax=2
2843
2844                    aytprime(ind)=k*8
2845                    Eprime(2) = - opa
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2829

```
2829            Bprime(2) = - EV%denl
2830            end if
2831
2832        else  !Tight coupling
2833            pig = 32._dl/45._dl*k
2834        endif
2835
2836        rhopi=grhog_t*pig
2837
2838
2839        !  Neutrino equations:
2840        !  Anisotropic stress
2841        if (DoTensorNeutrinos) th
2842            neutprime => aytprime
2843            neut => ayt(EV%r_ix+1
2844
2845            !  Massless neutrino
2846            pir=neut(2)
2847
2848            rhopi=rhopi+grhor_t*p
2849
2850            if (EV%lmaxnrt>2) the
2851                pirdt=-EV%denlkt(
2852                neutprime(2)=pird
2853                !  And for the mo
2854                do  l=3, EV%lmaxn
2855                    neutprime(l)=
2856                end do
2857
2858                !  Truncate the h
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2846

```
2846            Bprime(2) = - EV%
2847            end if
2848
2849        else  !Tight coupling
2850            pig = 32._dl/45._dl*k
2851        endif
2852
2853        rhopi=grhog_t*pig
2854
2855
2856        !  Neutrino equations:
2857        !  Anisotropic stress
2858        if (DoTensorNeutrinos) th
2859            neutprime => aytprime
2860            neut => ayt(EV%r_ix+1
2861
2862            !  Massless neutrino
2863            pir=neut(2)
2864
2865            rhopi=rhopi+grhor_t*p
2866
2867            if (EV%lmaxnrt>2) the
2868                pirdt=-EV%denlkt(
2869                neutprime(2)=pird
2870                !  And for the mo
2871                do  l=3, EV%lmaxn
2872                    neutprime(l)=
2873                end do
2874
2875                !  Truncate the h
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2859

```
2859                    neutprime(EV%lmax
2860                    (EV%lmaxnrt+3._dl
2861                else
2862                    pirdt= 8._dl/15._
2863                    neutprime(2)=pird
2864                end if
2865
2866                !  Massive neutrino e
2867                if (CP%Num_Nu_massive
2868                    do nu_i=1,CP%Nu_m
2869                        if (.not. EV%
2870                            rhopi=rho
2871                        else
2872                            ind=EV%nu
2873
2874                            pinu= Nu_
2875                            rhopi=rho
2876
2877                            do i=1,nq
2878                                q=nu_
2879                                aq=a*
2880                                v=1._
2881                                if (E
2882                                    a
2883                                    d
2884
2885
2886                                    e
2887                                    i
2888                                    !
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2876

```
2876                    neutprime(EV%lmax
2877                        (EV%lmaxnrt+3
2878                else
2879                    pirdt= 8._dl/15._
2880                    neutprime(2)=pird
2881                end if
2882
2883                !  Massive neutrino e
2884                if (CP%Num_Nu_massive
2885                    do nu_i=1,CP%Nu_m
2886                        if (.not. EV%
2887                            rhopi=rho
2888                        else
2889                            ind=EV%nu
2890
2891                            pinu= Nu_
2892                            rhopi=rho
2893
2894                            do i=1,nq
2895                                q=nu_
2896                                aq=a*
2897                                v=1._
2898                                if (E
2899                                    a
2900                                    d
2901
2902
2903                                    e
2904                                    i
2905                                    !
```

/Users/lp1opa/Compare/camb_simdata/equations_ppf.f90, Top line: 2889

```
2889 |                                    a
2890 |                                 else
2891 |                                    a
2892 |                                 end i
2893 |                                 ind=i
2894 |                              end do
2895 |                           end if
2896 |                        end do
2897 |                     end if
2898 |                  end if
2899 |
2900 |                  !  Get the propagation eq
2901 |
2902 |                  if (CP%flat) then
2903 |                   aytprime(3)=-2*adotoa
2904 |                  else
2905 |                   aytprime(3)=-2*adotoa
2906 |                  endif
2907 |
2908 |                  aytprime(2)=-k*shear
2909 |
2910 |                  end subroutine derivst
2911 |
2912 |
2913 |
2914 |                  !ccccccccccccccccccccccc
2915 |
2916 |                  end module GaugeInterface
2917 |
```

/Users/lp1opa/Compare/camb_des/equations_ppf.f90, Top line: 2906

```
2906 |                                    a
2907 |                                 else
2908 |                                    a
2909 |                                 end i
2910 |                                 ind=i
2911 |                              end do
2912 |                           end if
2913 |                        end do
2914 |                     end if
2915 |                  end if
2916 |
2917 |                  !  Get the propagation eq
2918 |
2919 |                  if (CP%flat) then
2920 |                   aytprime(3)=-2*adotoa
2921 |                  else
2922 |                   aytprime(3)=-2*adotoa
2923 |                  endif
2924 |
2925 |                  aytprime(2)=-k*shear
2926 |
2927 |                  end subroutine derivst
2928 |
2929 |
2930 |
2931 |                  !ccccccccccccccccccccccc
2932 |
2933 |                  end module GaugeInterface
2934 |
```