Adding non-trivial species in CLASS

Thomas Tram

Institute of Gravitation and Cosmology

October 10, 2014

Expanding CLASS

Adding new physics

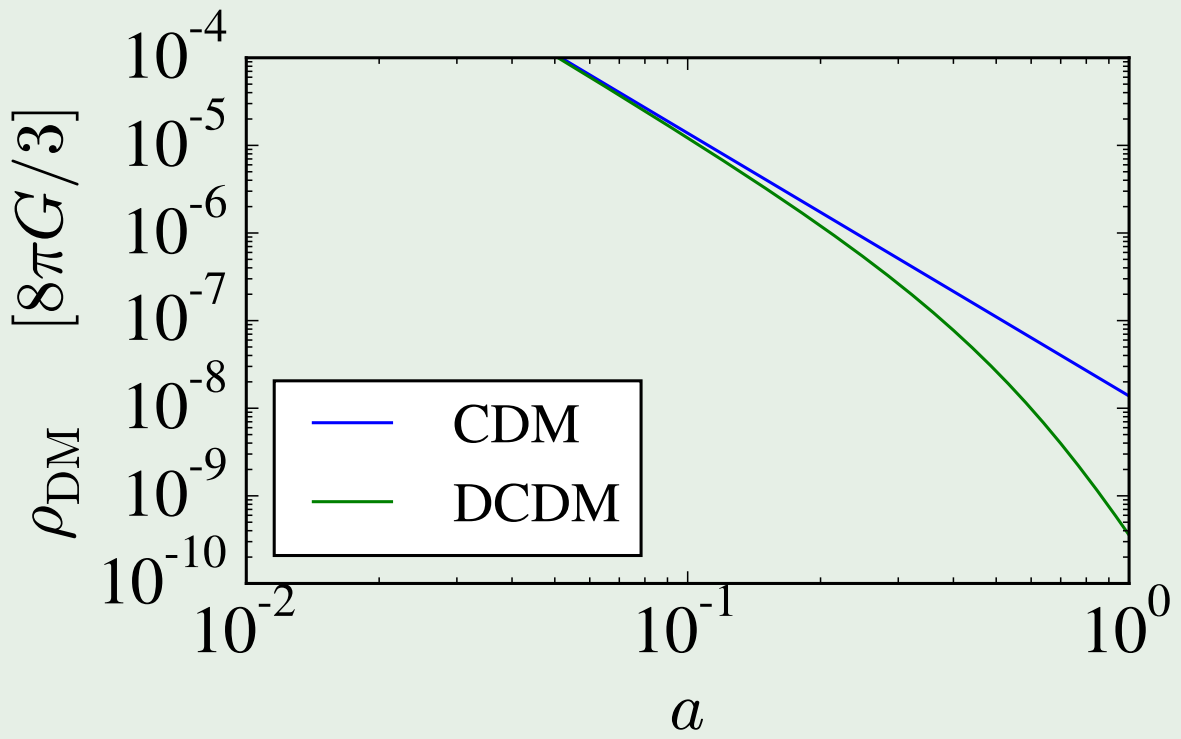
- Adding exotic physics to CLASS is **easy***.
- New features will be merged into the code and will be available in all future versions.
- This lecture will go through the implementation of decaying CDM, `dcdm`, step by step.

Footnote...

* If the rules of CLASS are carefully followed!

A bit about the physics

Background evolution



A bit about the physics

Background evolution

$$\rho_{\text{dcdm}}{}' = -3\frac{a'}{a}\rho_{\text{dcdm}} - a\,\Gamma_{\text{dcdm}}\,\rho_{\text{dcdm}}\ ,$$

$$\rho_{\text{dr}}{}' = -4\frac{a'}{a}\rho_{\text{dr}} + a\,\Gamma_{\text{dcdm}}\,\rho_{\text{dcdm}}\ .$$

Input parameters

`Omega_dcdmdr` $\equiv \Omega_{\text{dcdm}} + \Omega_{\text{dr}}$,

`omega_dcdmdr` $\equiv \omega_{\text{dcdm}} + \omega_{\text{dr}}$,

`Omega_ini_dcdm` $\equiv \Omega_{\text{dcdm, initial}}$,

`omega_ini_dcdm` $\equiv \omega_{\text{dcdm, initial}}$.

input.c modifications

input_init()

```
/** These two arrays must contain the strings of
    names to be searched
    for and the coresponding new parameter */
char * const target_namestrings[] = {"100*theta_s",
     "Omega_dcdmdr", "omega_dcdmdr", "Omega_scf", "
    Omega_ini_dcdm", "omega_ini_dcdm"};
char * const unknown_namestrings[] = {"h", "
    Omega_ini_dcdm", "Omega_ini_dcdm","
    scf_shooting_parameter", "Omega_dcdmdr", "
    omega_dcdmdr"};
enum computation_stage target_cs[] = {
    cs_thermodynamics, cs_background, cs_background
    , cs_background, cs_background, cs_background};
```

input.c modifications

input_init()

```
/** These two arrays must contain the strings of
    names to be searched
    for and the coresponding new parameter */
char * const target_namestrings[] = {"100*theta_s",
     "Omega_dcdmdr", "omega_dcdmdr", "Omega_scf", "
    Omega_ini_dcdm", "omega_ini_dcdm"};
char * const unknown_namestrings[] = {"h", "
    Omega_ini_dcdm", "Omega_ini_dcdm","
    scf_shooting_parameter", "Omega_dcdmdr", "
    omega_dcdmdr"};
enum computation_stage target_cs[] = {
    cs_thermodynamics, cs_background, cs_background
    , cs_background, cs_background, cs_background};
```

input_read_arguments()

I am skipping this part, since you have seen it many times already!

input.c modifications

input_try_unknown_parameters()

```
for (i=0; i < pfzw->target_size; i++) {
  switch (pfzw->target_name[i]) {
  case theta_s:
    output[i] = 100.*th.rs_rec/th.ra_rec-pfzw->
        target_value[i];
    break;
  case Omega_dcdmdr:
    rho_dcdm_today = ba.background_table[(ba.
        bt_size-1)*ba.bg_size+ba.index_bg_rho_dcdm
        ];
    if (ba.has_dr == _TRUE_)
      rho_dr_today = ba.background_table[(ba.
          bt_size-1)*ba.bg_size+ba.index_bg_rho_dr
          ];
    else
      rho_dr_today = 0.;
    output[i] = (rho_dcdm_today+rho_dr_today)/(ba.
        H0*ba.H0)-pfzw->target_value[i];
    break;
```

input.c modifications

input_get_guess()

```
/** Here we should right reasonable guesses for the unknown
    parameters. Also estimate dxdy, i.e. how the unknown
    parameter responds to the known. This can simply be
    estimated as the derivative of the guess formula.*/

for (index_guess=0; index_guess < pfzw->target_size;
    index_guess++) {
  switch (pfzw->target_name[index_guess]) {

  case Omega_dcdmdr:
    Omega_M = ba.Omega0_cdm+ba.Omega0_dcdmdr+ba.Omega0_b;

    if (gamma < 1)
      a_decay = 1.0;
    else
      a_decay = pow(1+(gamma*gamma-1.)/Omega_M,-1./3.);
    xguess[index_guess] = pfzw->target_value[index_guess]/
        a_decay;
    dxdy[index_guess] = 1./a_decay;
```

background.c modifications

background_functions()

```
/* dcdm */
if (pba->has_dcdm == _TRUE_) {
  /* Pass value of rho_dcdm to output */
  pvecback[pba->index_bg_rho_dcdm] = pvecback_B[pba->
      index_bi_rho_dcdm];
  rho_tot += pvecback[pba->index_bg_rho_dcdm];
  p_tot += 0.;
  rho_m += pvecback[pba->index_bg_rho_dcdm];
}

/* dr */
if (pba->has_dr == _TRUE_) {
  /* Pass value of rho_dr to output */
  pvecback[pba->index_bg_rho_dr] = pvecback_B[pba->
      index_bi_rho_dr];
  rho_tot += pvecback[pba->index_bg_rho_dr];
  p_tot += (1./3.)*pvecback[pba->index_bg_rho_dr];
  rho_r += pvecback[pba->index_bg_rho_dr];
}
```

background.c modifications

background_indices()

```
/* - index for dcdm */
class_define_index(pba->index_bg_rho_dcdm,pba->has_dcdm,index_bg
    ,1);

/* - index for dr */
class_define_index(pba->index_bg_rho_dr,pba->has_dr,index_bg,1);

/* - now, indices in vector of variables to integrate.
   First {B} variables, then {C} variables. */

index_bi=0;

/* -> energy density in DCDM */
class_define_index(pba->index_bi_rho_dcdm,pba->has_dcdm,index_bi
    ,1);

/* -> energy density in DR */
class_define_index(pba->index_bi_rho_dr,pba->has_dr,index_bi,1);
```

background.c modifications

background_initial_conditions()

```
  /* Set initial values of {B} variables: */

  if (pba->has_dcdm == _TRUE_){
    /* Remember that the critical density today in CLASS
        conventions is H0^2 */
    pvecback_integration[pba->index_bi_rho_dcdm] =
      pba->Omega_ini_dcdm*pba->H0*pba->H0*pow(pba->a_today/a,3);

  if (pba->has_dr == _TRUE_){
    if (pba->has_dcdm == _TRUE_){

      f = 1./3.*pow(a/pba->a_today,6)*pvecback_integration[pba->
          index_bi_rho_dcdm]*pba->Gamma_dcdm/pow(pba->H0,3)/sqrt(
          Omega_rad);
      pvecback_integration[pba->index_bi_rho_dr] = f*pba->H0*pba
          ->H0/pow(a/pba->a_today,4);
    }
    else{
```

background.c modifications

background_output_titles()

```
class_store_columntitle(titles,"(.)rho_dcdm",pba->has_dcdm);
class_store_columntitle(titles,"(.)rho_dr",pba->has_dr);
```

background_output_data()

```
class_store_double(dataptr, pvecback[pba->index_bg_rho_dcdm
    ], pba->has_dcdm, storeidx);
class_store_double(dataptr, pvecback[pba->index_bg_rho_dr],
     pba->has_dr, storeidx);
```

background.c modifications

background_derivs()

```
if (pba->has_dcdm == _TRUE_){
  /** compute dcdm density rho' = -3aH rho - a Gamma rho*/
  dy[pba->index_bi_rho_dcdm] = -3.*y[pba->index_bi_a]*
      pvecback[pba->index_bg_H]*y[pba->index_bi_rho_dcdm]
      - y[pba->index_bi_a]*pba->Gamma_dcdm*y[pba->
      index_bi_rho_dcdm];
}

if ((pba->has_dcdm == _TRUE_) && (pba->has_dr == _TRUE_)){
  /** Compute dr density rho' = -4aH rho - a Gamma rho*/
  dy[pba->index_bi_rho_dr] = -4.*y[pba->index_bi_a]*
      pvecback[pba->index_bg_H]*y[pba->index_bi_rho_dr] +
      y[pba->index_bi_a]*pba->Gamma_dcdm*y[pba->
      index_bi_rho_dcdm];
}
```

perturbations.c modifications

perturb_indices_of_perturbs()

```
      if (ppt->has_density_transfers == _TRUE_) {

        if (pba->has_dcdm == _TRUE_)
          ppt->has_source_delta_dcdm = _TRUE_;

        if (pba->has_dr == _TRUE_)
          ppt->has_source_delta_dr = _TRUE_;

      if (ppt->has_velocity_transfers == _TRUE_) {

          ppt->has_source_theta_dcdm = _TRUE_;
        if (pba->has_fld == _TRUE_)

        if (pba->has_dr == _TRUE_)
          ppt->has_source_theta_dr = _TRUE_;
```

perturbations.c modifications

perturb_indices_of_perturbs()

```
class_define_index(ppt->index_tp_delta_dcdm, ppt->
    has_source_delta_dcdm,index_type,1);

class_define_index(ppt->index_tp_delta_dr,   ppt->
    has_source_delta_dr, index_type,1);

class_define_index(ppt->index_tp_theta_dcdm, ppt->
    has_source_theta_dcdm,index_type,1);

class_define_index(ppt->index_tp_theta_dr,   ppt->
    has_source_theta_dr,  index_type,1);
```

perturbations.c modifications

perturb_prepare_output()

```
/* Decaying cold dark matter */
class_store_columntitle(ppt->scalar_titles, "
    delta_dcdm", pba->has_dcdm);
class_store_columntitle(ppt->scalar_titles, "
    theta_dcdm", pba->has_dcdm);
/* Decay radiation */
class_store_columntitle(ppt->scalar_titles, "delta_dr"
    , pba->has_dr);
class_store_columntitle(ppt->scalar_titles, "theta_dr"
    , pba->has_dr);
class_store_columntitle(ppt->scalar_titles, "shear_dr"
    , pba->has_dr);
```

perturbations.c modifications

perturb_vector_init()

```
/* dcdm */

class_define_index(ppv->index_pt_delta_dcdm,pba->
    has_dcdm,index_pt,1); /* dcdm density */
class_define_index(ppv->index_pt_theta_dcdm,pba->
    has_dcdm,index_pt,1); /* dcdm velocity */

/* ultra relativistic decay radiation */
if (pba->has_dr==_TRUE_){
  ppv->l_max_dr = ppr->l_max_dr;
  class_define_index(ppv->index_pt_F0_dr,_TRUE_,index_pt
      ,ppv->l_max_dr+1); /* all momenta in Boltzmann
      hierarchy  */
}
```

perturbations.c modifications

perturb_vector_init()

```
  /** - case of switching approximation while a wavenumber
      is being integrated */

    if (_scalars_) {

      if (pba->has_dcdm == _TRUE_) {

        ppv->y[ppv->index_pt_delta_dcdm] =
          ppw->pv->y[ppw->pv->index_pt_delta_dcdm];

        ppv->y[ppv->index_pt_theta_dcdm] =
          ppw->pv->y[ppw->pv->index_pt_theta_dcdm];
      }

      if (pba->has_dr == _TRUE_){
        for (l=0; l <= ppv->l_max_dr; l++)
          ppv->y[ppv->index_pt_F0_dr+l] =
            ppw->pv->y[ppw->pv->index_pt_F0_dr+l];
      }
```

perturbations.c modifications

perturb_initial_conditions()

```
      if (pba->has_dcdm == _TRUE_) {
        ppw->pv->y[ppw->pv->index_pt_delta_dcdm] = 3./4.*ppw->pv->y
            [ppw->pv->index_pt_delta_g]; /* dcdm density */

    if (ppt->gauge == newtonian) {

      if (pba->has_dcdm == _TRUE_) {
        ppw->pv->y[ppw->pv->index_pt_delta_dcdm] += (-3.*
            a_prime_over_a - a*pba->Gamma_dcdm)*alpha;
        ppw->pv->y[ppw->pv->index_pt_theta_dcdm] = k*k*alpha;
      }

        if (pba->has_dr == _TRUE_)
          delta_dr += (-4.*a_prime_over_a + a*pba->Gamma_dcdm*ppw->
              pvecback[pba->index_bg_rho_dcdm]/ppw->pvecback[pba->
              index_bg_rho_dr])*alpha;

    } /* end of gauge transformation to newtonian gauge */
```

perturbations.c modifications

perturb_initial_conditions()

```
if (pba->has_dr == _TRUE_) {

  f_dr = pow(pow(a/pba->a_today,2)/pba->H0,2)*ppw->
      pvecback[pba->index_bg_rho_dr];

  ppw->pv->y[ppw->pv->index_pt_F0_dr] = delta_dr*f_dr;

  ppw->pv->y[ppw->pv->index_pt_F0_dr+1] = 4./(3.*k)*
      theta_ur*f_dr;

  ppw->pv->y[ppw->pv->index_pt_F0_dr+2] = 2.*shear_ur*
      f_dr;

  ppw->pv->y[ppw->pv->index_pt_F0_dr+3] = l3_ur*f_dr;

}
```

perturbations.c modifications

perturb_total_stress_energy()

```
/* dcdm contribution */
if (pba->has_dcdm == _TRUE_) {
  ppw->delta_rho += ppw->pvecback[pba->index_bg_rho_dcdm]*y
      [ppw->pv->index_pt_delta_dcdm];
  ppw->rho_plus_p_theta += ppw->pvecback[pba->
      index_bg_rho_dcdm]*y[ppw->pv->index_pt_theta_dcdm];
}

if (pba->has_dr == _TRUE_) {
  rho_dr_over_f = pow(pba->H0/a2,2);
  ppw->delta_rho += rho_dr_over_f*y[ppw->pv->index_pt_F0_dr
      ];
  ppw->rho_plus_p_theta += 4./3.*3./4*k*rho_dr_over_f*y[ppw
      ->pv->index_pt_F0_dr+1];
  ppw->rho_plus_p_shear += 2./3.*rho_dr_over_f*y[ppw->pv->
      index_pt_F0_dr+2];
  ppw->delta_p += 1./3.*rho_dr_over_f*y[ppw->pv->
      index_pt_F0_dr];
}
```

perturbations.c modifications

perturb_sources()

```
/* delta_dcdm */
if (ppt->has_source_delta_dcdm == _TRUE_) {
  _set_source_(ppt->index_tp_delta_dcdm) = y[ppw->pv->
      index_pt_delta_dcdm];
}

/* delta_dr */
if (ppt->has_source_delta_dr == _TRUE_) {
  f_dr = pow(a2_rel/pba->H0,2)*pvecback[pba->
      index_bg_rho_dr];
  _set_source_(ppt->index_tp_delta_dr) = y[ppw->pv->
      index_pt_F0_dr]/f_dr;
}
```

perturbations.c modifications

perturb_sources()

```
/* theta_dcdm */
if (ppt->has_source_theta_dcdm == _TRUE_) {
  _set_source_(ppt->index_tp_theta_dcdm) = y[ppw->pv->
      index_pt_theta_dcdm];
}
```

```
/* theta_dr */
if (ppt->has_source_theta_dr == _TRUE_) {
  f_dr = pow(a2_rel/pba->H0,2)*pvecback[pba->
      index_bg_rho_dr];
  _set_source_(ppt->index_tp_theta_dr) = 3./4.*k*y[ppw->pv
      ->index_pt_F0_dr+1]/f_dr;
}
```

perturbations.c modifications

perturb_print_variables()

```
if (pba->has_dcdm == _TRUE_) {
  delta_dcdm = y[ppw->pv->index_pt_delta_dcdm];
  theta_dcdm = y[ppw->pv->index_pt_theta_dcdm];
}

if (pba->has_dr == _TRUE_) {
  f_dr = pow(pvecback[pba->index_bg_a]*pvecback[pba->
      index_bg_a]/pba->H0,2)*pvecback[pba->index_bg_rho_dr
      ];
  delta_dr = y[ppw->pv->index_pt_F0_dr]/f_dr;
  theta_dr = y[ppw->pv->index_pt_F0_dr+1]*3./4.*k/f_dr;
  shear_dr = y[ppw->pv->index_pt_F0_dr+2]*0.5/f_dr;
}
```

perturbations.c modifications

perturb_print_variables()

```
    /* converting synchronous variables to newtonian ones */
    if (ppt->gauge == synchronous) {

      if (pba->has_dr == _TRUE_) {
        delta_dr += (-4.*a*H+a*pba->Gamma_dcdm*pvecback[pba->
            index_bg_rho_dcdm]/pvecback[pba->index_bg_rho_dr])*
            alpha;
        theta_dr += k*k*alpha;
      }

      if (pba->has_dcdm == _TRUE_) {
        delta_dcdm += alpha*(-a*pba->Gamma_dcdm-3.*a*H);
        theta_dcdm += k*k*alpha;
      }

    }
```

perturbations.c modifications

perturb_print_variables()

```
    /* Decaying cold dark matter */
    class_store_double(dataptr, delta_dcdm, pba->has_dcdm,
        storeidx);
    class_store_double(dataptr, theta_dcdm, pba->has_dcdm,
        storeidx);
    /* Decay radiation */
    class_store_double(dataptr, delta_dr, pba->has_dr, storeidx
        );
    class_store_double(dataptr, theta_dr, pba->has_dr, storeidx
        );
    class_store_double(dataptr, shear_dr, pba->has_dr, storeidx
        );
```

perturbations.c modifications

perturb_derivs()

```
/** -> dcdm and dr */
if (pba->has_dcdm == _TRUE_) {
  dy[pv->index_pt_delta_dcdm] = -(y[pv->index_pt_theta_dcdm
      ]+metric_continuity) - a * pba->Gamma_dcdm / k2 *
      metric_euler;
  dy[pv->index_pt_theta_dcdm] = - a_prime_over_a*y[pv->
      index_pt_theta_dcdm] + metric_euler;
}
/** -> dr */
if ((pba->has_dcdm == _TRUE_)&&(pba->has_dr == _TRUE_)) {
  f_dr = pow(pow(a/pba->a_today,2)/pba->H0,2)*pvecback[pba
      ->index_bg_rho_dr];
  fprime_dr = pba->Gamma_dcdm*pvecback[pba->
      index_bg_rho_dcdm]*pow(a,5)/pow(pba->H0,2);
  dy[pv->index_pt_F0_dr] = -k*y[pv->index_pt_F0_dr
      +1]-4./3.*metric_continuity*f_dr+fprime_dr*(y[pv->
      index_pt_delta_dcdm]+metric_euler/k2);
```

spectra.c modifications

spectra_indices

```
class_define_index(psp->index_tr_delta_dcdm,ppt->
    has_source_delta_dcdm,index_tr,1);

class_define_index(psp->index_tr_delta_dr,ppt->
    has_source_delta_dr,index_tr,1);

class_define_index(psp->index_tr_theta_dcdm,ppt->
    has_source_theta_dcdm,index_tr,1);

class_define_index(psp->index_tr_theta_dr,ppt->
    has_source_theta_ur,index_tr,1);
```

spectra.c modifications

spectra_output_tk_titles()

```
class_store_columntitle(titles,"d_dcdm",pba->has_dcdm);
class_store_columntitle(titles,"d_dr",pba->has_dr);

class_store_columntitle(titles,"t_dcdm",pba->has_dcdm);
class_store_columntitle(titles,"t_dr",pba->has_dr);
```

spectra.c modifications

spectra_output_tk_data()

```
class_store_double(dataptr,tk[psp->
    index_tr_delta_dcdm],ppt->has_source_delta_dcdm
    ,storeidx);
class_store_double(dataptr,tk[psp->
    index_tr_delta_dr],ppt->has_source_delta_dr,
    storeidx);
```

```
class_store_double(dataptr,tk[psp->
    index_tr_theta_dcdm],ppt->has_source_theta_dcdm
    ,storeidx);
class_store_double(dataptr,tk[psp->
    index_tr_theta_dr],ppt->has_source_theta_dr,
    storeidx);
```