Der Einfluss der Ultravioletten Hintergrundstrahlung auf das Kühlverhalten von Gasen

Stefan Lüders

Munich 2018

The Effect of the Ultraviolet Background on Gas Cooling

Stefan Lüders

Munich 2018

The Effect of the Ultraviolet Background on Gas Cooling

Bachelor Thesis

at the

Ludwig–Maximilians–Universität (LMU) München

submitted by

Stefan Lüders

(Matr. Nr.: 11587082)

born on 1996-10-26 in Aschaffenburg

supervised by
Klaus Dolag

and
Aura Obreja

Munich, July 24, 2018

Evaluator: Klaus Dolag

Contents

1 Introduction **6**

2 Physical Background **7**

3 Models of the UVB **10**
3.1 Haardt & Madau 2005 (HM05) 10
3.2 Haardt & Madau 2012 (HM12) 10
3.3 Faucher-Giguère 2009 (FG09) and Faucher-Giguère 2011 (FG11) 11
3.4 Khaire & Srianand 2018 (KS18) 12
3.5 Differences between the four UVB models 12

4 The photoionization software Cloudy **12**

5 Method **14**
5.1 Generating the data 14
5.2 Analysing the data 16

6 Evaluation of results **17**

7 Summary **23**

A Cloudy input generation algorithm **24**

B Data visualization script **26**

C Table of Abbreviations and important symbols **31**

1 Introduction

The universe began in an extremely hot and dense state, approximately 13.8 billion years ago. 10^{-4}s after this "big bang", quarks and gluons finish condensing into protons and neutrons, at the end of the so-called Hadron Era. At temperatures of 10^{12}K, neutrons and protons were in thermal equilibrium, but as the universe expanded and cooled further, neutrons left this equilibrium in a process called "freeze out". Three minutes after the big bang, temperatures dropped below 10^{9}K and protons and neutrons begun forming bonds. This started a short lived chain of fusion reactions, creating large amounts of Helium and trace amounts of elements up to Beryllium in a process called "primordial nucleosynthesis". At roughly 75% Hydrogen and 25% Helium, the metallicity Z of this early universe, measured relatively to the metallicity of the solar system $Z_{\odot}$, is almost zero. The metallicity rose later on, as heavier elements were formed in stars. The ever increasing heavy element content of the Universe plays an important role in many astrophysical processes (e.g. Spatschek 2018).

In these early stages, all matter in the universe was still ionized, and remained so until temperatures dropped below 3000K almost 380000 years after the big bang. The afterglow of this event named "recombination" is visible to this day as the Cosmic Microwave Background (CMB), now redshifted to an equivalent temperature of 3K. The CMB is an important part of the Cosmic Background Radiation. In particular it is an important factor in cooling gas through the Inverse Compton Effect (e.g Wiersma et al. 2009).

After recombination, small irregularities in the still extremely homogeneous universe lead to the formation of the first stars (grouped in galaxies) and quasars, which reionized most of the Hydrogen in the Universe by redshift $z \sim 7$ (e.g. Barkana & Loeb 2001). The radiation emitted by the galaxies and quasars during their lifetimes defines the Ultraviolet Background (UVB) of the Universe, which plays an important role in gas photoionization on astrophysical scales. The rate of star formation continues to rise up to roughly $z = 2$. This coincides with the peak of quasar activity across cosmic history (see Fig. 1) and directly shapes the strength of the UVB throughout time.

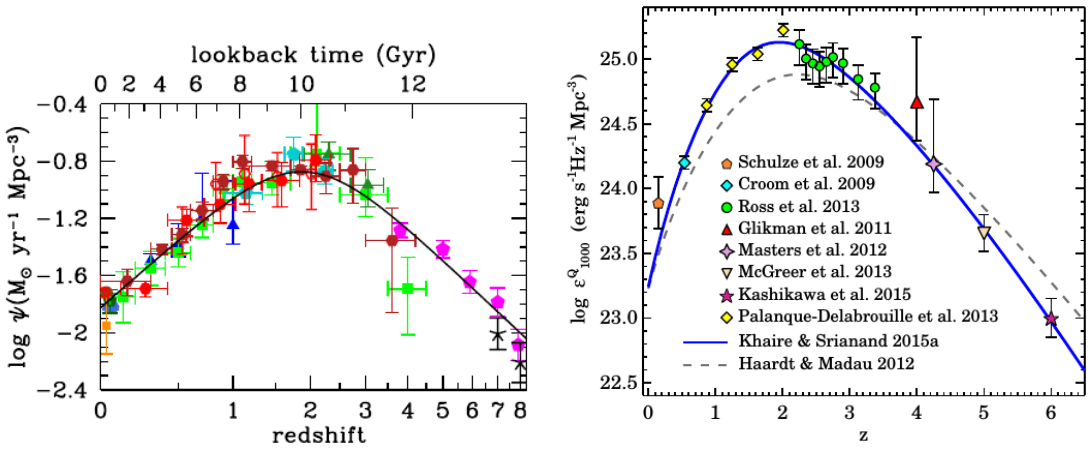

Figure 1: **Left:** The history of the cosmic star formation rate density, reproduced from Fig. 9 of Madau & Dickinson (2014). **Right:** The history of quasar emissivity at 1000 Å, reproduced from Fig. 1 of Khaire & Srianand (2018). Both peak at $z = 2$ and directly correlate to the strength of the UVB.

As time progresses, the small inhomogeneities in the initial (considered at CMB or $z \sim 1100$) density field are enhanced under the effect of gravity. Thus, gravity causes the dark and baryonic matter in the Universe to form a filamentary structure, the so-called "Cosmic Web" (Bond et al. 1996). According to the currently accepted cosmological model, Λ Cold Dark Matter (ΛCDM), the universe consists of roughly 25% Dark Matter (DM) and 70% dark energy (Planck Collaboration et al. 2016), represented mathematically by a cosmological constant Λ. Only $\sim$5% of the energy density of the universe is predicted to be "ordinary" baryonic matter and leptons.

Clumps of dark matter form "halos", acting as deep gravitational wells into which baryonic matter falls, therefore directing the formation of galaxies and supergalactic structures (e.g Somerville & Davé 2015). For gas to collapse into these dense DM halos against its own pressure, it needs to

radiate away thermal energy. The efficiency of this process depends on factors such as the density and metallicity of the gas and on the UVB photoionizing it, and indirectly on the redshift as the UVB changes over time.

In the past, many astrophysical simulations assumed Collisional Ionization Equilibrium (CIE) as basis for their cooling calculations (e.g. Wiersma et al. 2009; Gnedin & Hollon 2012; Gnat & Ferland 2012). In CIE, the equilibrium is given by the balance of collisional ionization from the ground states of the various gas particles and the recombination of higher ionization stages (e.g. Dopita & Sutherland (2003)). Assuming all ions are in the ground state in CIE significantly reduces the complexity of the cooling function by making it independent of the gas Hydrogen density n_{H}, at the cost of neglecting photoionization effects. In the fully ionized limit, the CIE approximation can thus be off by up to two orders of magnitude (Gnedin & Hollon 2012).

Thanks to advances in simulation software and hardware capabilities, it is now possible to simulate gas clouds in photoionization equilibrium, with background radiation, and use this data as basis for better, more accurate cooling functions.

The UVB is of particular importance to the cooling processes in the temperature range of 10^4 - 10^8 K, which is the typically interesting range for cosmological simulations. This work takes a look at how different radiation fields published in the literature affect the cooling functions, and examines whether the choice of UVB data could noticeably affect large scale simulations using these cooling functions.

I use for this purpose the photoionization package Cloudy (Ferland et al. 2013), which already comes coupled with several options for the UVBs. Of these, two UVB models are of interest for this work: Haardt & Madau 05 (see Section 3) and Haardt & Madau (2012). The other two models which I use and which are not provided with Cloudy are Faucher-Giguere 11[1] and Khaire & Srianand (2018).

Section 2 takes a look at the background physics involved, describing both the processes important in the cooling of cosmic gas clouds as well as some of the math behind the cooling function itself. Section 4 provides a quick introduction to the photoionization software Cloudy. Section 5 describes the process used to generate the data for this paper, while Section 6 explains it and provides its interpretation. Section 7 provides a summary of the results.

All the raw data created for this paper as well as the programs written to evaluate it are available upon request.

2 Physical Background

The overall cooling Λ of a gas cloud, defined as the loss of thermal energy per unit volume and unit time, is given by a balance of several factors:

- photoionization heating Γ_{ph}
- recombination cooling Λ_{rec}
- metal line cooling Λ_{CE}
- Bremsstrahlung cooling/free free emission Λ_{ff}
- Compton cooling/heating Λ_{comp}

There is an important difference between the *loss of energy* Λ (also referred to as the "cooling rate") and the *cooling function*

$$\frac{\Lambda}{n_e n_{\mathrm{H}}},$$

where n_e is the electron density of the cloud and n_{H} is the Hydrogen density of the cloud. For the remainder of this paper, it will be assumed that most of the free electrons stem from the ionization of Hydrogen, and thus

$$n_e \approx n_{\mathrm{H}},$$

which gives the cooling function

[1] http://galaxies.northwestern.edu/uvb/, last accessed 2018-06-19

$$\frac{\Lambda}{n_{\mathrm{H}}^2}.$$

The advantage of using the cooling function over the cooling rate is that it does not depend on n_{H} in the CIE approximation, which assumes that the effects of external radiation, like photoionization and the Compton Effect, are negligible. This greatly reduces the complexity and storage requirements of the tables the cooling function is usually interpolated from, and was thus popular in early simulations (e.g Gnat & Ferland 2012).

Another important thing to keep in mind is that the usage of symbols for cooling rate and function is not consistent in literature. Dopita & Sutherland (2003) refer to the cooling rate as $\dot{Q}$ and the cooling function itself as Λ, whereas Draine (2011) and papers such as Wiersma et al. (2009) refer to the cooling rate as Λ and the cooling function as Λ/n_{H}^2. This paper strictly adheres to the latter convention.

Unless explicitly stated otherwise, this paper will only deal with cooling effects. If the *net cooling* is used instead, it will be referred to as

$$\frac{|\Lambda - H|}{n_{\mathrm{H}}^2},$$

where H is a catch-all term for the heating effects.

The total cooling is then given by

$$\frac{\Lambda_{\mathrm{total}}}{n_{\mathrm{H}}^2} = \frac{\Lambda_{\mathrm{CE}}}{n_{\mathrm{H}}^2} + \frac{\Lambda_{\mathrm{rec}}}{n_{\mathrm{H}}^2} + \frac{\Lambda_{\mathrm{ff}}}{n_{\mathrm{H}}^2} \pm \frac{\Lambda_{\mathrm{comp}}}{n_{\mathrm{H}}^2}.$$

All of the aforementioned effects shall now be described in more detail.

Heating due to photoionization

The radiation hitting the cloud ionizes its atoms and molecules. The electrons that are set free in this process carry the excess photon energy as kinetic energy. This means there is an increase in total kinetic energy of the cloud as particles are ionized, and the temperature rises.

According to Draine (2011), the heating rate per unit volume from this process is

$$\Gamma_{\mathrm{ph}} = n(X^{+r}) \int_{\nu_0}^{\infty} \sigma_{\mathrm{ph}}(\nu) c \left[\frac{u_\nu}{h\nu}\right] (h\nu - h\nu_0) d\nu$$

where $h\nu_0$ is the threshold energy for photoionization, σ_{pe} is the photoionization cross section, $n(X)$ is the density of the species being ionized and u_ν is the spectrum.

The UVB provides the ionizing radiation for this process.

Cooling due to recombination

Ions capture free electrons in the cloud, removing the kinetic energy of the electrons from the gas. This process works directly against photoionization heating, removing the same electrons from the cloud that were previously freed by photoionization. For a constant radiation background, an equilibrium will eventually be reached, where the recombination cancel the photoionization and vice versa. This is known as "photoionization equilibrium". This paper deals exclusively with photoionization equilibrium, and does not consider CIE or non-equilibrium cooling.

Note that the process of capturing free electrons favors electrons with below-average energies. This makes recombination a *heating* effect locally (e.g Dopita & Sutherland 2003). However, in photoionization equilibrium, this can safely be ignored, as the recombination process counteracts the heating through photoionization, turning it into a cooling effect overall.

According to Draine (2011), the rate at which thermal energy per unit volume is lost by recombination cooling is

$$\Lambda_{\mathrm{rec}} = \alpha_B n_e n(H^+) \langle E_{rr} \rangle$$

where α_B is the rate coefficient for radiative recombination and E_{rr} is the mean kinetic energy of the recombining electrons.

Metal line cooling

Free electrons and ions may scatter off each other inelastically. This transfers kinetic energy from the electron to the ion, thus exciting the ion by raising one or more of the ion's electrons to higher energy levels. The ion will then radiate away this excitation energy as the electrons drop back to the unexcited level, and the radiation will escape the gas if it is transparent. This conversion of kinetic to radiation energy lowers the overall energy of the gas as well as its temperature. This is the primary cooling mechanisms of gases at medium temperatures (10^4 - 10^6 K).

The simulations in this paper assume completely transparent gas for this important process to work. See Section 5 on how this was achieved.

Collisional excitation cooling strongly depends on the composition of the gas. Atomic metals (understood as any element heavier than Helium), as well as molecules, provide many possible line transitions that can be collisionally excited and thus increase cooling noticeably. Hydrogen and Helium are less important; in HII regions, where Hydrogen is completely ionized and Helium is typically already excited, there is very little opportunity for additional collisional excitation. Metals such as Oxygen and Carbon are much more important.

The rate of energy loss of the gas by collisional excitation is

$$\Lambda_{\mathrm{CE}} = \sum_X \sum_i n(X,i) \sum_{j<i} A_{ij}(E_i - E_j)$$

where the sum is over species X and excited states i (Draine 2011).

Elements can only be considered on their own if they do not significantly contribute to the electron density (e.g. Wiersma et al. 2009; Shen et al. 2010). This is usually true for elements heavier than Helium at metallicities up to solar metallicity. The cooling rate is then given by

$$\Lambda = \Lambda_{\mathrm{H,He}} + \sum_{i>\mathrm{He}} \Lambda_i$$

where $\Lambda_{\mathrm{H,He}}$ is the combined cooling due to Hydrogen and Helium and Λ_i is the overall cooling by the metal i.

Note that at large temperatures (and low densities) heavy elements may actually suppress cooling in the presence of an ionizing radiation field. The radiation field ionizes the heavy elements stronger than pure collisional excitation would, reducing the opportunity for collisional excitation processes to convert kinetic into radiation energy (e.g. Wiersma et al. 2009).

Cooling due to free-free emissions (Bremsstrahlung)

Electrons scattering elastically with ions will give off Bremsstrahlung, as they are accelerated in the electric field of the ions. This effect is very important at high temperatures and high densities, since both of these circumstances increase the chance of such collisions happening.

According to Draine (2011), in a pure Hydrogen plasma near 10^4K the free free emission energy per recombining electron is

$$\frac{\Lambda_{\mathrm{ff}}}{n_e n(H^+)\alpha_B} = 0.54 T_4^{0.37} kT$$

where $T_4 \equiv \frac{T}{10^4 K}$.

Compton cooling

The Compton effect describes the scattering of photons on free or loosely bound electrons. The "regular" Compton effect occurs when photons scatter off these electrons inelastically, transferring energy from the photon to the electrons. In the context of gas clouds this is a heating effect. The *inverse* Compton effect occurs when photons scatter superelastically, transferring energy from the electron to the photon. This is a cooling effect instead. Which effect occurs depends on the ratio of the electron's thermal energy to the energy of the photon being scattered (e.g Dopita & Sutherland 2003).

According to Dopita & Sutherland (2003), the Compton term is given by:

$$\Lambda_{\mathrm{comp}} = \int_\nu F_\nu \sigma_{\nu,\gamma} \frac{n_e}{m_e c^2} (4kT_e - h\nu) d\nu,$$

where F_ν is the flux of radiation at frequency ν and $\sigma_{\nu,\gamma}$ is the scattering cross section, given by the Klein-Nishina formula:

$$\sigma_{\nu,\gamma} = \frac{3\sigma_T}{4}\left\{\frac{1+q}{q^3}\left[\frac{2q(1+q)}{1+2q} - \ln(1+2q)\right] + \frac{1}{2q}\ln(1+2q) - \frac{1+3q}{(1+2q)^2}\right\}$$

where

$$q = \frac{\gamma h\nu}{m_e c^2}$$

and

$$\sigma_T = \frac{8\pi}{3}\frac{e^2}{m_e c^2} = 6.65 \times 10^{-25}\mathrm{cm}^2.$$

Furthermore, for nonrelativistic thermal plasmas

$$\langle\gamma\rangle = \frac{3kT_e}{2m_e c^2}.$$

Therefore, the scattering cross section then takes the asymptotic forms

$$\sigma_{\nu,\gamma} = \sigma_T, \quad q \ll 1$$

and

$$\sigma_{\nu,\gamma} = \frac{3\sigma_T}{8q}\left[\frac{1}{2} + \ln(2q)\right]$$

Since the difference of the thermal and photon energies appears explicitly in Λ_{comp}, the Compton term changes sign when $q = 8\gamma^2/3$, which means the difference between cooling and heating.

3 Models of the UVB

The shape of the Ultraviolet Background (UVB) is predominantly defined by the emissions of quasars and hot and massive young stars. However, secondary radiation due to absorption and re-emission in the gas of and around galaxies also plays an important role. The "clumpy" Intergalactic Medium (IGM) is ionized by the primary radiation, but then "reradiates" the energy it gained through recombination processes. The work of Haardt & Madau (1996) was the first to describe this process in detail by considering the ionization state of the IGM in its calculations, prior to which the IGM was only modeled as a purely photoelectrically absorbing material. Much of the later work on UVBs by Haardt & Madau, as well as the UVBs considered here are ultimately based on this work.

3.1 Haardt & Madau 2005 (HM05)

This model is a successor to the original 1996 model, as well as the intervening 2001 model. It was created using the original version of the cosmological radiative transfer simulation code CUBA, described in Haardt & Madau (2001). While there was no published paper associated with it, it has been released to collaborators upon request. Files containing the HM05 tables are part of the current release of Cloudy and can be easily integrated into simulations run with it. (See Section 4 for details on how to include HM05 in a Cloudy simulation.)

3.2 Haardt & Madau 2012 (HM12)

Haardt & Madau (2012) is the latest and most popular iteration of the Haardt & Madau series of UVB data sets and has been used in numerous papers and publications. Major improvements over the previous versions include[2]:

- modeling of the resonance scattering of Lyman series photons

[2] http://www.ucolick.org/~pmadau/CUBA/DOWNLOADS.html, last accessed 2018-06-19

- inclusion of the X-ray background, produced by quasars
- adjustment of the effect of the IGM due to new data about the mean free path of 1 Ryd photons
- better treatment of the photoionization structure of absorbers, allowing more accurate predictions on e.g. the effects of photon Helium interaction
- consideration of the UV emissions due to young, active galaxies

The tables containing HM12 can be found here[3]. Like HM05, this UVB is available with the current release of CLOUDY.

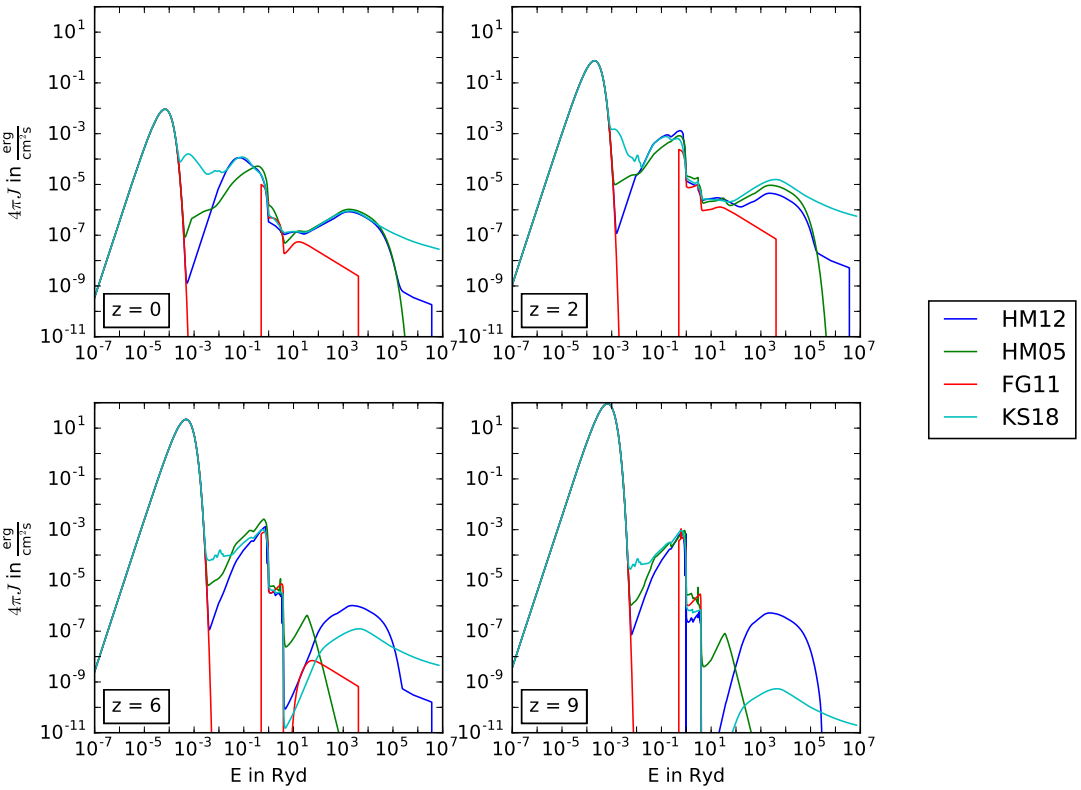

Figure 2: The comparison among the four UVB models (colored curves) at four different redshifts. The first peak between 10^{-7} and 10^{-2} Ryd is the corresponding CMB at each redshift, which is independent of the UVB models.

3.3 Faucher-Giguère 2009 (FG09) and Faucher-Giguère 2011 (FG11)

The model of Faucher-Giguère et al. (2009) (FG09) was developed with the purpose of providing an alternative to the popular but aging Haardt & Madau UVB ones, as well as accounting for more recent (at the time) observational data. To develop this model, Faucher-Giguère et al. (2009) used their own, independent radiative transfer/photoionization code and observational constraints.

Faucher-Giguère et al. (2009) found that their model matches the Haardt & Madau model quite well at lower z, but diverges increasingly towards higher redshift. FG11[4] is an updated version of FG09 from December 2011. It remedies some of the divergence issues of FG09 from HM05 at high redshifts, and covers a larger range of redshifts and frequencies, along other minor updates.

[3] http://www.ucolick.org/~pmadau/CUBA/DOWNLOADS.html
[4] http://galaxies.northwestern.edu/uvb/, last accessed 2018-06-19

The cooling function derived from FG09 was intensively used for the FIRE simulations (Hopkins et al. 2014), as well as for some a first implementation of the local photoionization field in the AREPO code (Kannan et al. 2016). The usage of FG09 in these widely known, highly publicized simulations is the primary reason for including Faucher-Giguère in this comparison; however, since FG09 was not locatable to the author in preparation of this paper, FG11 was used instead.

3.4 Khaire & Srianand 2018 (KS18)

Compared to HM12, KS18[5] extends the frequency range to cover from TeV γ-ray to far-infrared (FIR), and corrects the amount of secondary Lyα photons escaping from galaxies for dust attenuation. Apart from that, it uses the same basic model as HM12. The major difference between these models is the available observational data. Being the newest UVB data set by far, KS18 incorporates the most recent findings and constraints, including the HI column density distribution of Inoue et al. (2014), the quasar emissivity from the compilation of Khaire & Srianand (2015a), the cosmic star formation rate density from the compilation of Khaire & Srianand (2015b), and galaxy UV escape fraction of Khaire et al. (2016) (Khaire & Srianand 2018).

KS18 is not yet available with Cloudy, but will likely become its next default UVB. In the meantime, the UVB data can be found here[6].

3.5 Differences between the four UVB models

Evidently, the different UVB models are similar in many ways, with subtle differences in calculations and predictions. These differences stem from the different assumptions and parameters used in modeling, such as the quasar and galaxy emissivities. The measurements of the Hydrogen photoionization rate Γ_{HI} used to calibrate the models vary as well, with KS18 predicting consistently higher HI ionization and FG09 predicting consistently lower HI ionization than HM12.

While KS18 and HM12 largely agree on HeII ionization except for high redshifts, FG09 predicts ionization rates of up to an order of magnitude lower at $z < 3$.

FG09 in particular differs from the other models as it did not attempt to match X-Ray observational data, contrary to HM12 and KS18 (Khaire & Srianand 2018).

Fig. 2 shows the different UVBs in comparison. Note that this comparison also includes the CMB, and represent the total input radiation fields used in the Cloudy simulations of this work.

4 The photoionization software Cloudy

The following section provides a quick introduction to Cloudy, the software used to generate the data examined in this paper. It is essentially an abridged version of the first few chapters of Hazy, Cloudy's official documentation. Please refer to Hazy for an in-depth guide[7].

Cloudy is a program designed to predict the spectral emissions of clouds of matter in space. It has been developed by Gary J. Ferland, along with numerous contributors, over the past 40 years. At the time of writing, the latest version is Cloudy 17.01, released on 2018-04-18. This version was used to create all data examined in this paper.

Cloudy accurately simulates a wide variety of physical effects, such as

- the radiation background
- cooling, heating and chemical processes of the lightest 30 elements and many molecules
- effects of small dust particles ("grains")

Despite the prediction of spectral emissions being the primary focus of Cloudy, in this paper it was mostly used for its cooling simulation.

Detailed instructions on how to download and compile Cloudycan be found here[8]. The code runs on all major operating systems, however for the easiest installation process it is recommended to use Linux.

[5] Also "KS18Q18", after their Q18 fiducial model, named for the exponent of their QSO SED power law $\alpha = -1.8$

[6] ftp://ftp.iucaa.in/in.coming/KS18EBL/

[7] Both Cloudy and Hazy can be found at https://www.nublado.org/

[8] https://www.nublado.org/wiki/StepByStep

Once installed, the program can be used by providing it with short scripts through the standard input. Note that CLOUDY input scripts will need to end in the file extension `.in`. A script called `input.in` located in the `/cloudy/source` folder can be run like this:

```
./cloudy.exe -r input
```

For an example of a typical CLOUDY input script, see section 5.1, which discusses one of the scripts used to create the data examined in this paper.

In order for the simulation to work, the script needs to specify the Hydrogen density of the cloud, as well as the intensity and shape of the radiation field, called the "incident radiation field". Apart from these, a wide variety of parameters can be specified, including the metallicity and composition of the cloud, the presence of grains and various debug- and output commands.

The Hydrogen density can be set with the `hden X` command, where `X` is the decadic logarithm of the Hydrogen density in units of of cm^{-3}. Using the logarithm of a value is typical for CLOUDY commands; frequently it is context sensitive whether an argument is interpreted linearly or logarithmically. For example, commands that take temperatures as arguments will interpret inputs up to 10 as a decadic logarithm, but inputs greater than 10 as linear. Other commands may make this distinction at 0. This behavior can be manually overwritten using the keywords `linear` or `logarithmic`. Similarly, commands usually interpret input in cgs units. Depending on the command, this may be overwritten with contextual keywords as well.

Many incident radiation fields can be given as input by setting up their shape and intensity. There are multiple ways of setting up the shape, including:

- specifying the shape as a power law.
- specifying the shape as the blackbody radiation spectrum of a certain temperature.
- interpolating the shape from a table in a file.
- using CLOUDY output files as input for the spectrum.

In contrast, there are only two distinct ways of defining the intensity: Providing the luminosity of a central object, or providing the intensity at the surface of the cloud. These are referred to as the "Luminosity Case" and "Intensity Case".

Note that "Intensity" in the context of CLOUDY does not refer to the specific intensity I or the mean intensity J, but rather the angle-independent mean intensity $4\pi J$. Osterbrock & Ferland (2006) define J as:

$$J = \frac{1}{4\pi} \int I(\theta, \phi) d\Omega \ [\mathrm{erg\ s^{-1}\ cm^{-2}\ sr^{-1}}]$$

where the specific intensity $I(\theta, \phi)$ is given by the flow of radiation past a unit area, per unit time, per solid angle, in a given direction. For a ray at an angle θ to the normal of a surface area dA the energy passing this surface area is given by:

$$dE = I(\theta, \phi) \cos(\theta) d\Omega dA dt.$$

Care must be taken to not confuse J for the mean intensity per unit frequency J_ν. It is recommended to always double check all intensity units when working with the intensity case.

In the Luminosity Case, a radius must be chosen so CLOUDY can calculate the intensity at the surface of the cloud. Like with many CLOUDY commands, a default is chosen if no value is provided. This default radius is 10^{30} cm. In the Intensity Case, the intensity at an arbitrary energy (commonly 1 Ryd) needs to be supplied using the `f(nu)` command, which will then automatically be used to normalize the given shape of the field.

On top of this, some commands specify both the shape and the intensity of the radiation field, such as the `table HM12 X` command, which loads the Haardt & Madau 12 UVB for redshift `X`. Similarly, it is possible to load the HM05 UVB by using `table HM05 X`.

It is possible to overlay up to 100 radiation fields on top of each other to customize the total radiation field. In this case, consecutive shape and intensity/luminosity defining commands are assumed to belong together. Because of this, and because some commands define both shape *and* intensity, the order of the commands is important and it is good practice to keep matching commands close together to avoid side effects. It is furthermore recommended to use `save continuum` to save the incident radiation field. This allows one to later verify that it looks as expected.

No matter how the radiation field is specified, it should be defined between the lowest frequency Cloudy simulates, 10 MHz, and the highest energy at 7.354×10^6Ryd $\approx$ 100MeV. If this condition is not met, Cloudy will still perform the simulation but output a warning.

Lastly, Cloudy needs an exit condition that stops the simulation once it has been reached. Cloudy divides the clouds it simulates into layers or "zones" of equal physical properties, and by default the simulation stops once it reaches either

- the 1400th zone or
- a zone with a temperature below 4000K.

At 4000K, the simulation typically has passed the ionization front of the cloud, which means that most of the ionizing radiation has been absorbed and subsequent Hydrogen is in an electronically neutral state.

These exit conditions are usually sufficient, but custom ones can be set; for example, the simulation can be exited once the temperature drops below a custom minimum with the `stop temperature` command.

5 Method

All calculations were performed with version 17.01 of Cloudy, last described by Ferland et al. (2013). Gas clouds with varying properties were simulated in Cloudy. The data produced was then visualized using matplotlib and systematically examined for unusual behavior. Section 5.1 describes how the data was generated while Section 5.2 describes how it was visualized. See Section 6 for analysis of the results.

5.1 Generating the data

All 384 permutations of the following parameters were simulated in Cloudy:

Parameters	Values
redshift z	0, 1, 2, 3, 6, 9
Hydrogen density n_{H} (in cm^{-3})	1, 10^{-2}, 10^{-4}, 10^{-6}
metallicity Z (in $Z_{\odot}$)	1, 10^{-1}, 10^{-2}
UVB	HM05, HM12, FG11, KS18

Table 1: Parameters used in the Cloudy simulations.

A python script was used to generate the required Cloudy input files (See Appendix A). Each of the 384 generated files was then run on the USM's server cluster DORC[9].

A typical Cloudy input file looks like this:

```
hden -2
CMB 3
metallicity -1
stop zone 1
iterate to convergence
constant temperature 5 vary
grid 4, 8, 0.1 ncpus 16
save continuum "arrayjob213_z3_nH2_Z1_HM05_cont"
save cooling each last "arrayjob213_z3_nH2_Z1_HM05_cool"
table HM05 redshift 3
```

In this example, the simulation is run with Hydrogen density $n_{\mathrm{H}} = 10^{-2}$ cm^{-3}, redshift $z = 3$ and metallicity $Z = 10^{-1} \times Z_{\odot}$ for the HM05 background. The purpose of the individual lines is as follows:

```
hden -2
```

[9] http://www.usm.uni-muenchen.de/

This command sets the Hydrogen density to 10^{-2} cm^{-3}, which is a medium density in the context of this simulation. This is one of the commands required for the simulation to run successfully (see Section 4). The Hydrogen density determines the overall density of the gas since the abundances of other elements are usually specified relative to Hydrogen.

```
CMB 3
```

This command tells CLOUDY to add the Cosmic Microwave Background (CMB) at redshift $z = 3$ to the incident radiation field. This is necessary as the UVB data does not include the CMB. The CMB is particularly important at low densities and high redshifts.

```
metallicity -1
```

This line sets the metallicity to 10^{-1} times the solar metallicity, using the default CLOUDY solar abundances. In the case of primordial metallicity, this would instead be `abundances "primordial.abn"`. While the metallicity command scales the solar abundances as a whole by a set factor (for example, 10^{-1}), the abundances command changes the actual abundances of elements relative to each other. In this case, the predefined abundances found in the file "primordial.abn" that comes with CLOUDY are loaded, which accurately reflect the state of the universe at its earliest stages.

```
stop zone 1
```

CLOUDY divides the cloud it simulates into layers of constant physical characteristics, called "zones" (see Section 4), where "higher" zones are deeper into the cloud. Each zone will then be simulated in succession until an exit condition is met. Typically, this is after the ionization front of the gas cloud where the gas temperature falls below 4000K or after 1400 zones.

Using the `stop zone 1` command, CLOUDY always stops at the first zone. This simulates a transparent gas. Due to having an open geometry (the CLOUDY default) and the radiation not penetrating or being scattered into deeper zones, incoming radiation interacts with the radiation only once, and the gas cloud does not self-interact through outgoing radiation. This is equivalent to the gas cloud being transparent.

The gas is assumed to be transparent since the simulated densities are quite low. At higher densities, self-shielding can become important. If the radiation can not escape the gas and carry energy with it, radiation cooling is suppressed.

```
iterate to convergence
```

This command is used to make sure the optical depth of the cloud is calculated correctly. It is necessary if line transfer or radiation pressure are important to the results at the simulation. Simulations run for this paper generally took three iterations to converge.

```
constant temperature 5 vary
grid 4, 8, 0.1 ncpus 16
```

These two lines tell CLOUDY to simulate temperatures from 10^4K to 10^8K. The `constant temperature` sets the temperature of the gas cloud to a given value. It does not actually allow the temperature of the gas cloud to change during the simulation and as such must be used with caution. Since only a "snapshot" of the gas cloud is taken immediately upon exposure this does not pose a problem for this simulation.

To set the temperature, but not force it to a fixed value during the entire simulation, one can use the `corona` command. Testing revealed that for the purpose of this paper, the results of these commands were identical, so for the final simulations `constant temperature` was chosen for readability.

Usually, these commands are used to set a single initial temperature for the gas cloud. However, to get the temperature dependent cooling function $\Lambda(T)$ for a set of parameters, multiple data points are required. To acquire these, the `vary` keyword is used. In combination with the `grid` command on the next line, it tells CLOUDY to run a series of simulations for variations of a given variable - in this case temperature T - instead of a single one. Here, temperatures are varied from 10^4K to 10^8K at 0.1 dex increments. This means for each of the 384 input files, 40 simulations will actually be run, for a total of 384×40. The total computing time rises accordingly.

Note that the `vary` keyword overwrites the input parameter of its "parent command", in this case `5`. However, for CLOUDY to run the file, the parameter is still required, and as such can be chosen arbitrarily in the desired range.

```
save continuum last "arrayjob213_z3_nH2_Z1_HM05_cont"
save cooling each last "arrayjob213_z3_nH2_Z1_HM05_cool"
```

These instruct CLOUDY to save data in addition to the regular output file. `save continuum` saves the continuum of the incident radiation field. In this case this is later used to verify that the shape of the radiation field is as one would expect. The continuum files are quite large due to saving the continuum for each iteration and each grid-run. They can easily exceed 100MB each. The `last` keyword tells CLOUDY to only save the continuum of the last iteration, saving some space.

The cooling function(s) are saved with the command `save cooling` . By default, the total cooling and total heating are saved. Using `each`, the cooling of each element is saved in addition to the total cooling function, but heating is omitted. The keyword `last` tells CLOUDY to only save the cooling of the last iteration.

Note that these instructions differ slightly from the lines that will be generated by the script in Appendix A. This is because of an oversight during the creation of the initial version of the script, which only used `save cooling each` as seen above. However, both the cooling due to the elements and the heating were required for the analysis of the results. Another set of input files were generated without the `each` command to generate the missing data. The script in the appendix reflects this by including two `save cooling` commands to save all the data required at once.

Note that the filenames, including the extensions, are completely arbitrary. Further note that in this case the filenames contain information about the individual simulation in question, with all minus signs omitted for parsing reasons (i.e. `nH2` instead of `nH-2`).

```
table HM05 redshift 3
```

Lastly, this very important commands tells CLOUDY which incident radiation field to use. Of the UVBs packaged with CLOUDY, this paper takes a look at HM05 and HM12, loaded with `table HM05` and `table HM12`, respectively. See Section 3 for more information on these UVBs. These table commands take a number as argument, providing the redshift z of the UVB. Due to the way CLOUDY interprets lines - in general, only the first four letters and any following numbers are parsed, with minor exceptions -, it is possible to include arbitrary text before the argument. Here, `redshift` is added for readability.

The other two radiation fields, FG11 and KS18Q18, were taken from here[10] and here[11], respectively (see also Section 3). UVBs not available with CLOUDY are supposed to be imported with the `table SED` command. However, when trying to normalize the UVB using `f(nu)` after loading it with `table SED`, an error was thrown for unknown reasons. To get around this issue, I used instead the legacy command `interpolate` .

5.2 Analysing the data

The simulations produced over 26GB of data, which had to be cut down considerably to be processable. The continuum data alone was 14.1GB, with an additional 12.0GB of regular output files. The cooling data itself was comparatively small with just 3.3MB in total.

As the regular CLOUDY output files are not important to the subject of this paper, they were ignored beyond making sure there were no unforeseen critical errors. Additionally, CLOUDY saved the input continuum for each grid-run separately (384×40 times instead of just 384). As these are identical within a grid, only the very first spectrum was used for plotting across the entire temperature range for a set of parameters. This reduced the size of the continuum files from 36.5MB each to just under 1MB each.

Due to the shape of $\Lambda(T)$ being dependent on four input parameters (z, Z, n_{H}, UVB), visualizing the entire data set at once proved infeasible. In the end, an interactive solution was found.

[10] http://galaxies.northwestern.edu/uvb/
[11] ftp://ftp.iucaa.in/in.coming/KS18EBL/

Therefore, I wrote a python script using matplotlib which allows browsing through the parameter space. The script shows three graphs. On the left, it shows the cooling function

$$\frac{\Lambda}{n_{\mathrm{H}}^2}$$

for different Hydrogen densities. In the middle, it shows the net cooling function,

$$\frac{|\Lambda - H|}{n_{\mathrm{H}}^2},$$

which accounts for heating H. See Section 2 for more information on the cooling function.

Lastly, on the right it shows the input continuum, which is a combination of the chosen UVB and the CMB at the chosen redshift.

For the source code, see Appendix B. The data created by the simulation and the scripts found in the appendices are also available digitally upon request.

The script allows one to browse through all of the data using the keys W, A, S, D, Q and E. Press W to increase redshift and S to decrease readshift. Likewise, A reduces metallicity whereas D increases it. Q and E switch between UVBs.

It also features a separate viewing mode that makes it easier to find differences between the UVBs. To switch between modes, press the spacebar. In alternate mode, the cooling functions are plotted relative to HM12, which equals unity in this mode. This way it is directly visible by what factor the cooling functions differ between backgrounds. HM12 was chosen as baseline since it was the newest available version of the established Haardt & Madau series of UVBs and has been used in a lot of publications. Note that the alternate plot of the net cooling functions is using a symlog plot, where the range $0-1$ is plotted linearly and values greater than 1 are plotted logarithmically. This is due to some large positive factors occurring in edge cases when comparing the net cooling functions. See the next section (6) for the information gained from the use of this tool.

6 Evaluation of results

This section describes and explains the results of the simulations presented above (see Section 5 and Appendix A). The effect of each of the input parameters on the UVBs and the cooling functions are described in detail. The conclusions were drawn using the interactive script described in the previous section. Some of the figures in this section, namely the ones breaking down the cooling functions by metals, have been taken from a different set of data (See also Section 5 and Appendix A). Care has been taken to make sure the data matches up.

The effect of redshift z

The redshift is an important factor in the cooling of a gas cloud because it greatly affects the shape and strength of the UVB. Stellar formation rate density and quasar activity peak for $z \approx 2$, flooding the universe with radiation (See also Section 2). Thus the UVB is at its most intense at this redshift (Compare Fig. 2). As the redshift increases, all UVB models discussed start to develop a significant minimum at ∼10 Ryd.

As one can see in Fig. 2, the CMB (visible in the energy range up to 10^{-2} Ryd) steadily decreases in intensity with decreasing redshift. This is due to the radiation density declining as the universe expands over time.

Lastly, one can see that FG11 is very focused on the energy range 10^0 - 10^4 with an intensity of zero at many energies. There appears to be data missing in this model, particularly at $z = 9$.

The cooling functions themselves are dependent on z through the UVB. At high Hydrogen densities, redshift barely changes the shapes of the cooling functions (green and red curves in Fig. 3). However, at the low densities typical of the intergalactic medium (IGM), the z-dependent UVB influences a lot the shape of the cooling functions (blue and cyan curves in Fig. 3, corresponding to $n_{\mathrm{H}} = 10^{-4}$ cm^{-3} and 10^{-6} cm^{-3}).

This is due to the fact that at lower densities the inverse Compton effect plays a much larger role, overpowering other sources of cooling. It is also mainly driven by interaction with the CMB, which is much more intense at higher redshifts. On the other hand, the cross section for collisional recombination is very low at these lower densities.

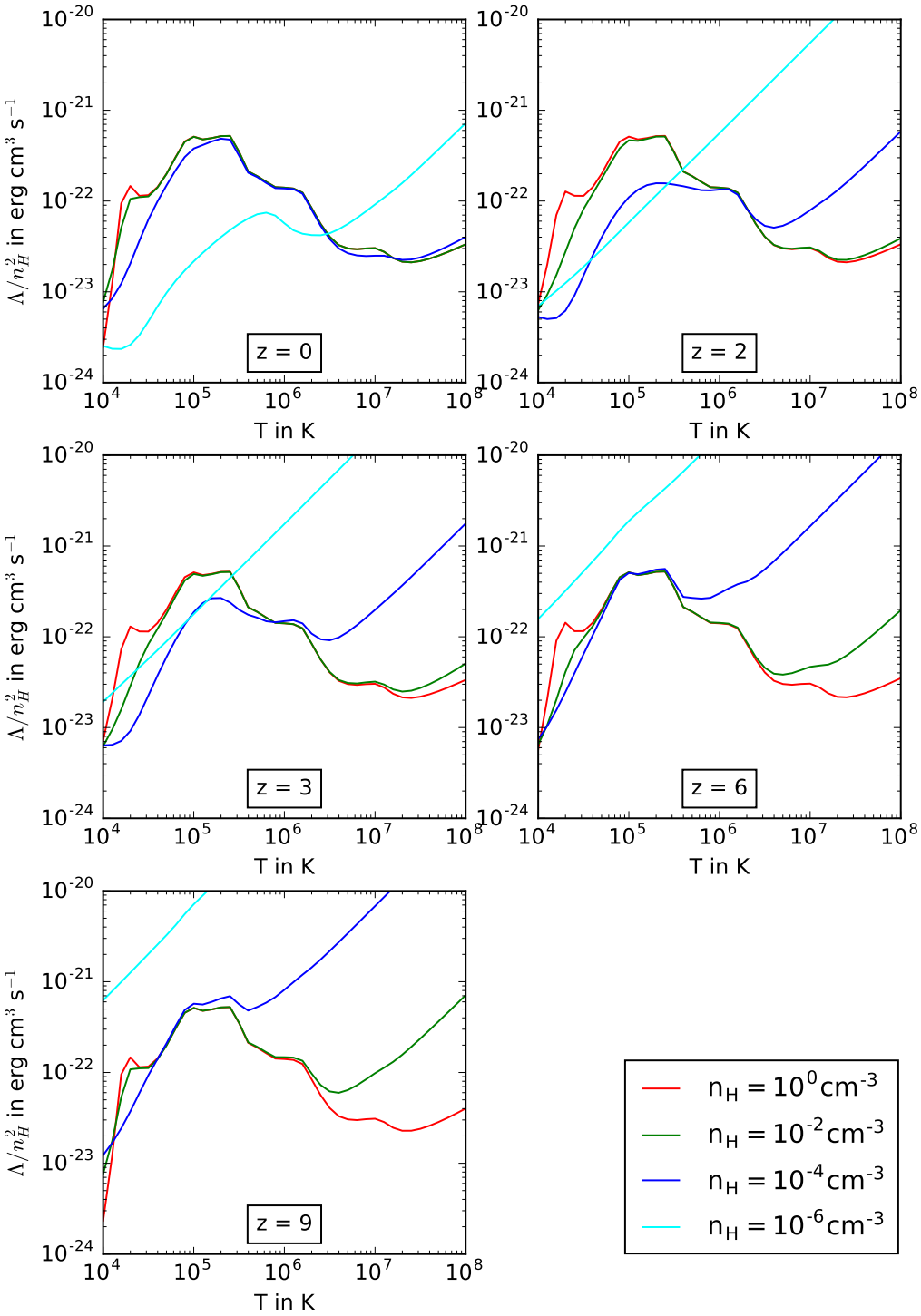

Figure 3: Comparison of cooling functions at different redshifts and densities. The functions in this figure were calculated for solar metallicity using the HM12 UVB. Note how Compton cooling takes over at $n_{\rm H} = 10^{-6}$ cm^{-3} by $z = 2$, but $n_{\rm H} = 10^{0}$ cm^{-3} keeps its distinct shape up to $z = 9$.

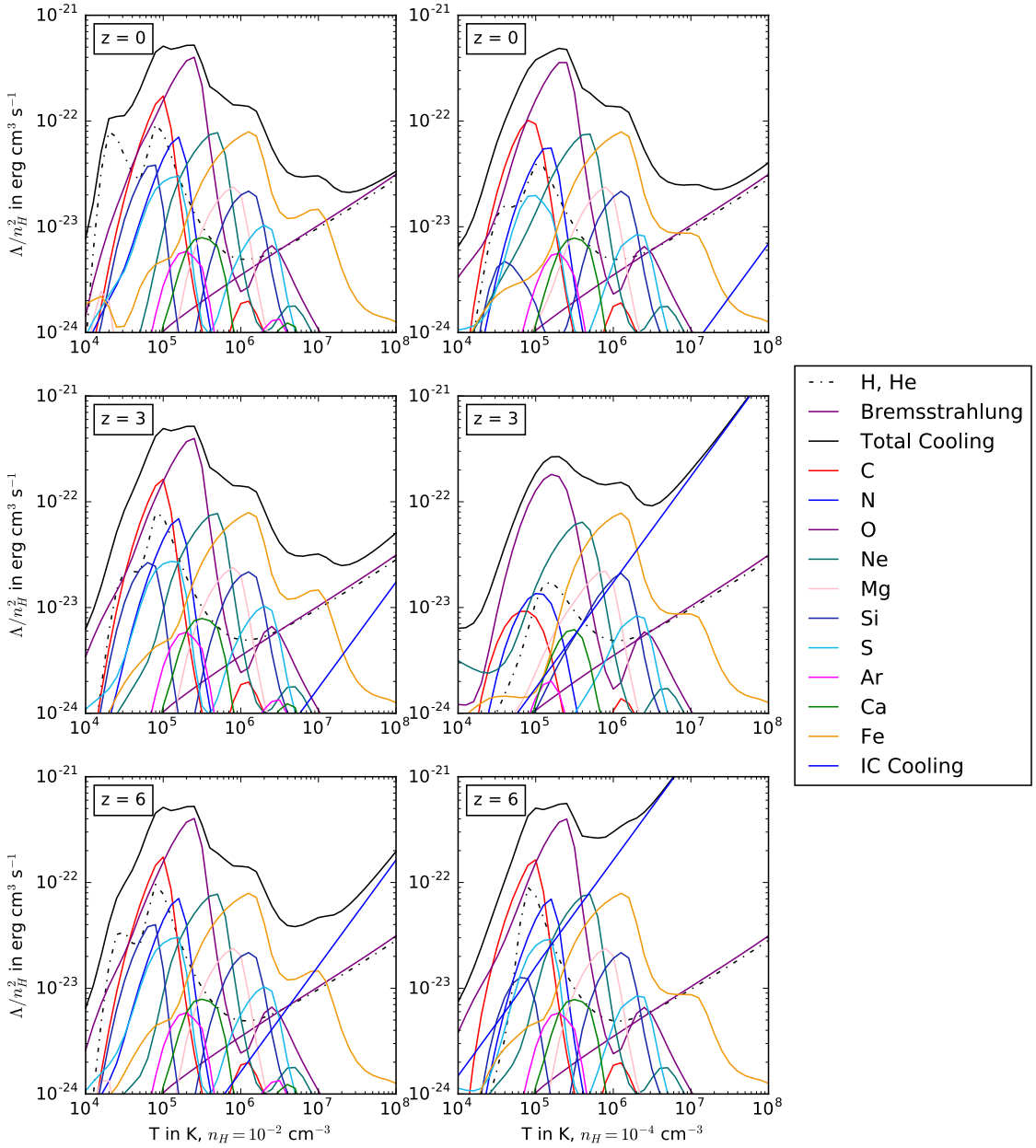

Figure 4: Cooling functions broken down by constituents for different redshifts. The left column shows $n_{\mathrm{H}} = 10^{-2}$ cm^{-3}, the right one 10^{-6}cm^{-3}. The rows show redshifts 0, 3 and 6, from top to bottom. Only the most important constituents of the cooling functions are shown. Notice how the combined cooling of Hydrogen and Helium is dominated by the Bremsstrahlung interaction of Hydrogen and electrons at larger temperatures. For large temperatures, large redshifts or low densities inverse Compton cooling dominates all other factors.

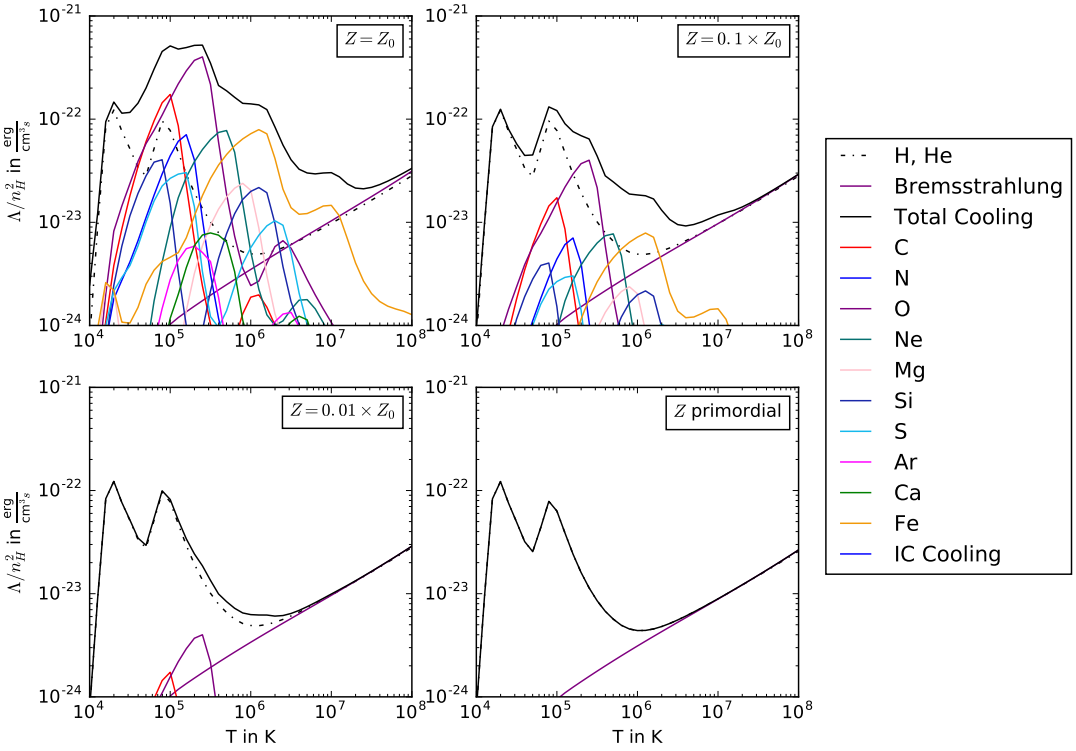

Figure 5: Cooling functions and constituents at $z = 0$ using the HM12 UVB for different metallicities, with $Z_\odot$ being solar metallicity and primordial metallicity being taken from CLOUDY's primordial abundances. At low metallicities there can obviously not be as much metal cooling as with higher metallicities. This leads to a "compression" of the shape of the function. The lower the metallicity, the more important Hydrogen and Helium cooling becomes, particularly due to Hydrogen-electron Bremsstrahlung at high temperatures. Note how the total Bremsstrahlung cooling matches the H+He cooling almost perfectly in the low metallicity cases. This is due to the metal-electron Bremsstrahlung missing. In the primordial case, metal cooling is completely gone, and the total cooling function is defined by the Hydrogen and Helium cooling.

Fig. 4 shows the cooling functions for different densities $n_\text{H} = 10^{-2}$ cm^{-3} and $n_\text{H} = 10^{-4}$ cm^{-3} broken down to their constituents, at $z = 0$, 3 and 6 from top to bottom. As one can see, neither at different densities, nor at different redshifts does the metal cooling vary by a lot. For large temperatures, Hydrogen becomes increasingly important. This is due to the Hydrogen-electron-Bremsstrahlung interaction that dominates the overall Bremsstrahlung cooling as well. Electron-electron-Bremsstrahlung as well as metal-electron-Bremsstrahlung only play a minor role and leads to a small difference between total Bremsstrahlung cooling and Hydrogen cooling at large temperatures.

For very large temperatures Compton cooling rapidly becomes the most important cooling factor. This is magnified at higher redshifts and lower densities.

The effect of metallicity Z

Metal cooling is quite important at low to medium temperatures (10^4K - 10^6K). Accordingly, there is a severe dip in cooling at these temperatures if metallicity is reduced, and Compton cooling takes over much sooner. This leads to a "compression" of the cooling function, as seen in Fig. 5. Once metals start being negligible ($Z \approx 10^{-2} \times Z_0$) only Hydrogen and Helium cooling remain, along with Bremsstrahlung and the inverse Compton effect.

In the primordial case, where there *are* no metals, the cooling function obviously takes into account only Hydrogen and Helium.

The effect of Hydrogen density n_{H}

As seen in the previous figures, the Hydrogen density is important in determining which cooling process dominates and how strong the overall cooling is. At high densities, metal cooling through electron ion collisions is very effective (since collisions can occur often). Compton cooling however can not keep up, since the same intensity of radiation has to cool more matter. This allows high density gas to cool comparatively efficiently at low redshifts, where Compton cooling is not as important (Fig. 3).

At lower densities, metal cooling is far less important. The gas particles need to collide to excite ions, which can then radiate this energy away. These collisions are improbable at low densities. However, due to there being less particles, Compton cooling becomes far more effective, since there is now less matter to be cooled with the same amount of radiation.

Fig. 3 shows an interesting edge case of this for $z = 2$ and $z = 3$. For $n_{\mathrm{H}} = 10^{-4}$ cm^{-3}, the density is still too high for metal cooling to be negligible and Compton cooling completely takes over. However, this density is also low enough for metal cooling to be inefficient. For 10^4K to 10^6K it is thus the smallest of all considered cooling functions. For $T > 10^6$K it is more efficient than higher densities due to Compton cooling, but still orders of magnitude lower than $n_{\mathrm{H}} = 10^{-6}$ cm^{-3}.

The effects of the different UVBs

As the different UVBs provide similar input data the cooling functions do not vary by a lot. In most cases they do not differ by more than a factor of 2, and agree particularly well at higher temperatures, where the CMB dominates cooling via the Compton effect. FG11 matches the other UVBs surprisingly well, especially at high temperatures. HM05 generally leads to less cooling, particularly for $n_{\mathrm{H}} = 10^{-4}$ cm^{-3}. KS18's cooling functions agree with HM12's except for minor variations.

Looking at the net cooling, one notices particularly strong minima in the $10^4\mathrm{K} - 10^5\mathrm{K}$ range. These denote the equilibrium temperature, where heating and cooling cancel each other (e.g. Wiersma et al. 2009)). Changes in the equilibrium temperature is one of the bigger effects of changing the UVB. However, due to the low resolution of the simulation (0.1 dex), an in-depth look at the equilibrium temperature is not possible in this paper (see Fig. 6).

The following comparison was made using the script described in Section 5. HM05, FG11 and KS18 will be compared to HM12, which is chosen as "normal" due to being the most established, commonly accepted model.

The cooling functions created using HM05, predecessor of HM12, generally match those generated using HM12 quite well. Still, there are important differences. Out of all the UVBs compared, HM05 has a significantly higher equilibrium temperature than the other UVBs. This effect appears to be of the order of up to half a magnitude and occurs particularly at low redshifts, and across all metallicities and Hydrogen densities.

Furthermore, in the low temperature range of $10^4\mathrm{K} - 10^5\mathrm{K}$ the HM05 cooling functions are up to a factor 2 smaller than their HM12 counterparts. This is especially pronounced at lower densities and metallicities.

For higher redshifts the HM05 cooling functions match HM12 much better. A study of the equilibrium temperature is out of the scope of these simulations. The cooling functions match well, however, particularly at higher densities and low temperatures.

HM05's effects are particularly strong at $n_{\mathrm{H}} = 10^{-4}$ cm^{-3}. As such one must keep in mind that in simulations using HM05 concerned with the temperature of low density gas such as the ISM and IGM results my vary strongly from simulations using more modern data. The overall cooling behavior, however, is still modeled correctly.

FG11 is the "odd one out" of the UVBs compared here, focusing on a much narrower energy range than the other data sets, and containing particularly little data at large redshifts ($z = 9$). (See also Section 3 for the differences in approach to the calculation of FG11 compared to the other UVBs.) Despite its differences in shape, the cooling functions it generated were consistent with the cooling functions using other UVBs.

The largest differences in the FG11 cooling compared to HM12 occur at low redshifts. Particularly at solar metallicity and $z = 0$, where the cooling function for $n_{\mathrm{H}} = 10^{-6}$ cm^{-3} is locally over

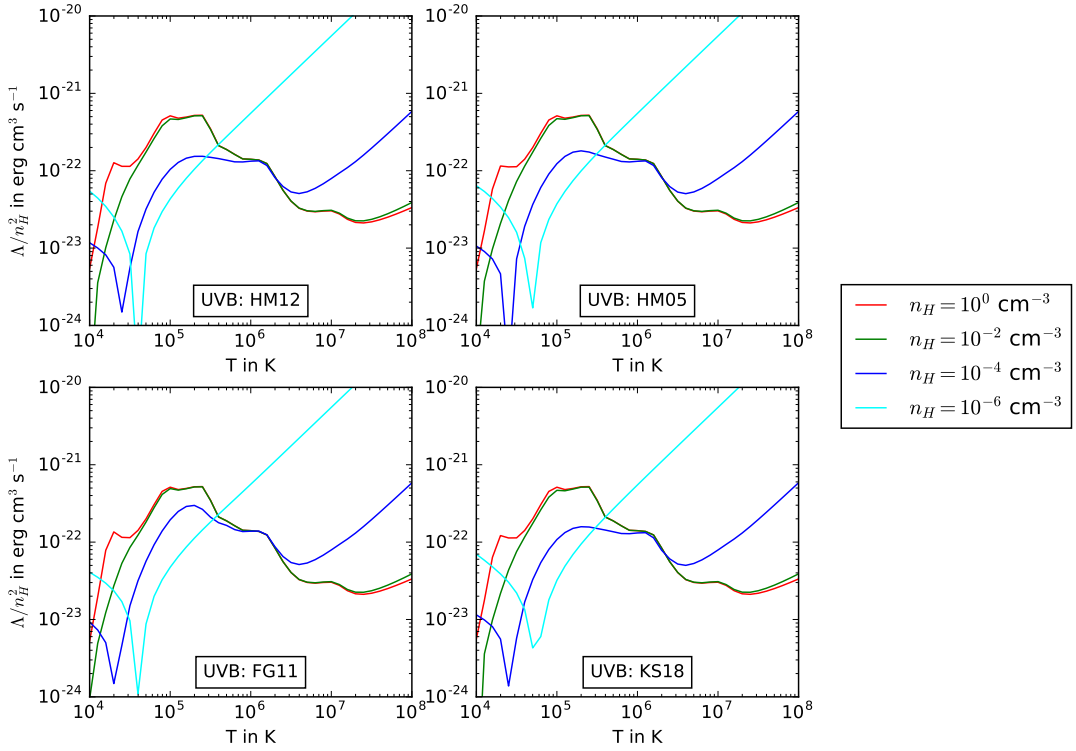

Figure 6: Cooling functions at different UVBs for $z = 2$ and $Z = Z_\odot$. The equilibrium temperatures are visible as minima in the 10^4K - 10^5K temperature range. The overall shape of the cooling functions are very similar for different UVBs.

3 times larger than the HM12 equivalent. At low redshifts and low densities, FG11 produces consistently larger cooling than HM12 at almost all temperatures, especially at low temperatures and metallicites. The difference frequently exceeds a factor of 2. However, the equilibrium temperatures are largely unaffected.

For larger redshifts, this effect shifts to slightly higher densities. While it is strongest in the $n_\mathrm{H} = 10^{-6}$ cm^{-3} curve for $z = 0$, $n_\mathrm{H} = 10^{-4}$ cm^{-3} shows the largest discrepancy for $z = 2$ and $n_\mathrm{H} = 10^{-2}$ cm^{-3} begins diverging at $z = 3$. This effect disappears at the largest simulated redshifts ($z = 6$ and $z = 9$), where the cooling functions match HM12's cooling functions exceedingly well.

Despite the severe lack of data at higher redshifts, FG11 provides consistent results particularly here. This suggests that only a small band of energy is actually important to the cooling processes. At low redshifts the generated cooling functions are consistently larger than those generated with other UVBs. While the equilibrium temperature matches that of other models, cooling rate or time will only be comparable between simulations in the early universe.

As a further refinement of HM12, KS18 only leads to minor adjustment to the cooling functions, with differences generally $< 30\%$. The equilibrium temperatures are indistinguishable from HM12 at the resolution of this simulation. The largest differences in the cooling functions are generally found in the $10^4\mathrm{K} - 10^5\mathrm{K}$ temperature range, particularly at $n_\mathrm{H} = 10^{-2}$ cm^{-3} and large redshifts. Here, KS18's cooling functions are up to 40% smaller than HM12's.

KS18 predicts larger cooling than HM12 only for $n_\mathrm{H} = 10^{-2}$ cm^{-3} at $z = 3$, across all metallicities. This goes up to 25% at $Z = Z_\odot$ and $T \approx 10^5$K.

KS18 is generally consistent with the results of HM12. Assuming that the cooling functions of KS18 are more accurate than HM12's due to access to newer and better data, HM12 consistently overestimates cooling in the $10^4\mathrm{K} - 10^5\mathrm{K}$ temperature range. This suggests that HM05 was more accurate here. Either way, cooling due to the KS18 background is directly comparable to HM12, since the differences should be negligible for most applications.

7 Summary

Using CLOUDY it was possible to reproduce a great variety of cooling functions across the chosen parameter space (see Section 5) and observe a variety of predicted physical effects, such as metal line cooling, the suppression of cooling through heavy metals, Bremsstrahlung cooling and inverse Compton cooling. It was seen that for the low densities typically found in the ISM and IGM, inverse Compton Cooling plays a much larger role than Bremsstrahlung.

The comparison of UVBs showed that the choice of UVB as basis for the cooling function should not greatly affect the outcome of a simulation it is embedded in, provided this simulation does not specifically concern itself with the equilibrium temperature of low density gas, or the cooling of this gas in the $10^4\text{K} - 10^5\text{K}$ temperature range. In particular it was shown that the choice of UVB does not play a major role in the cooling at higher temperatures ($> 10^5\text{K}$).

Further it was shown that while the general cooling behavior of low density gases does not depend on the UVB as the overall shape of the cooling function stays the same, the UVBs should still be considered and their effects understood to properly evaluate the results of a simulation. Specifically, cooling due to FG11 is in excellent agreement with other UVBs at higher redshifts, but is significantly higher at lower redshifts (frequently to a factor of 2). At high redshifts, the lack of data did not seem to inhibit this UVB's effect on the cooling functions. This suggests it might be possible to save space and/or computing power by omitting unimportant parts of the spectrum. Simulations using HM05 are expected to produce equilibrium temperatures that are up to half a magnitude too high at low redshifts. HM12 may lead to too much cooling at lower temperatures of $10^4\text{K} - 10^5\text{K}$. This could have a noticeable effect on equilibrium temperatures below 10^4K.

Lastly, KS18 is largely in agreement with HM12, making these two comparable, but leads to less cooling at low temperatures. Since KS18 is the newest and based on the most recent data and observations, it should be the optimal choice for new simulations, especially given that comparability to other simulations is not an issue.

A Cloudy input generation algorithm

A digital version of this script is available upon request.

```
import os
import numpy as np
import math

# iterable lists of parameters
redshift = (0, 1, 2, 3, 6, 9)
nH = (0, 2, 4, 6)                    # sign omitted for spacing
metallicity = (0, 1, 2, "X")        # as above, X = primordial
UVB = ("HM05", "HM12", "KS18", "FG11")
arr_nr = 1                           # unique identifier for input files


# Because the table SED command didn't work, this converts the KS18 and
# FG11 source files into interpolate(...) ... continue(...) commands
# Based on a script by Aura Obreja, used with permission
def convertUVB(file, UVB, redshift):
    en_Ryd = np.genfromtxt(UVB + '/z_' + str(redshift) + ".sed",
                           comments='#', usecols=(0), dtype=(float))
    Jnu = np.genfromtxt(UVB + '/z_' + str(redshift) + ".sed",
                           comments='#', usecols=(1), dtype=(float))
    cont_x = 4.*np.pi*Jnu

    # Calculate f(nu) line
    # Find value reasonably close to 1 (<1% difference),
    # use it for normalization
    for i in range(len(en_Ryd)):
        if math.fabs(en_Ryd[i] - 1) < 0.01:
            file.write("f(nu) " + str(math.log(4*math.pi*Jnu[i], 10)) +\
                       " at " + str(en_Ryd[i]) + "\n")
            break

    bigstr = ""
    if UVB == "KS18":
        for en,cont in zip(en_Ryd[::-1],cont_x[::-1]):
            bigstr = bigstr + \
                     ('continue (%.5e  %.5e)\n'%(en,np.log10(cont)))
    elif UVB == "FG11":
        for en,cont in zip(en_Ryd,cont_x):
            bigstr = bigstr + \
                     ('continue (%.5e  %.5e)\n'%(en,np.log10(cont)))

    bigstr = "interpolate " + bigstr[9:]
    file.write(bigstr)

# Iterate through all 384 possible variations of input parameters,
# assemble a cloudy input file for each
for i in range(len(redshift)):
    for j in range(len(nH)):
        for k in range(len(metallicity)):
            for l in range(len(UVB)):
                # Create input file;
                # filename contains data about parameters
                file = open("arrayjob" + str(arr_nr).zfill(3) + "_z" + \
                            str(redshift[i]) + "_nH" + str(nH[j]) + \
                            "_Z" + str(metallicity[k]) + "_" + UVB[l] +\
```

```
                ".in", "w")

            arr_nr += 1

            # Set hydrogen density and microwave background
            file.write("hden -" + str(nH[j]) + "\n")
            file.write("CMB " + str(redshift[i]) + "\n")

            # Set metallicity or primordial abundances
            if metallicity[k] == "X":
                file.write("abundances \"primordial.abn\" \n")
            else:
                file.write("metallicity -" + \
                           str(metallicity[k]) + "\n")

            # One zone to simulate transparency,
            # convergence to make sure results make sense
            file.write("stop zone 1 \n")
            file.write("iterate to convergence \n")

            # Vary temperature, save results
            file.write("constant temperature 5 vary \n")
            file.write("grid 4, 8, 0.05 ncpus 16 \n")
            file.write("save continuum \"" + file.name[:-3] + \
                       "_cont\" \n")
            # saves Ctot and Htot:
            file.write("save cooling last \"" + file.name[:-3] + \
                       "_cool\" \n")
            # saves Ctot and components:
            file.write("save cooling each last \"" + \
                       file.name[:-3] + "_cool\" \n")

            # Add UVB
            if UVB[l] == "HM05" or UVB[l] == "HM12":
                file.write("table " + str(UVB[l]) + " redshift " + \
                           str(redshift[i]) + "\n")
            else:
                convertUVB(file, UVB[l], redshift[i])

            file.close()

```

B Data visualization script

A digital version of this script is available upon request.

```
import matplotlib.pyplot as plt
import numpy as np
import os
import math

# The "getch" classes were taken in accordance with licensing condition
# from
# https://github.com/ActiveState/code/tree/master/recipes/Python/134892 \
#    _getchlike_unbuffered_character_reading_stdboth
class _Getch:
    """Gets a single character from standard input.  Does not echo to the
screen."""
    def __init__(self):
        try:
            self.impl = _GetchWindows()
        except ImportError:
            self.impl = _GetchUnix()

    def __call__(self): return self.impl()


class _GetchUnix:
    def __init__(self):
        import tty, sys

    def __call__(self):
        import sys, tty, termios
        fd = sys.stdin.fileno()
        old_settings = termios.tcgetattr(fd)
        try:
            tty.setraw(sys.stdin.fileno())
            ch = sys.stdin.read(1)
        finally:
            termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
        return ch


class _GetchWindows:
    def __init__(self):
        import msvcrt

    def __call__(self):
        import msvcrt
        return msvcrt.getch()


getch = _Getch()

# shape of the "cooling" array:
# cooling = [z][Z][nH][UVB][T,Lambda(T)/nH^2]
cooling = np.zeros((10,4,7,5,41,2))
heating = np.zeros((10,4,7,5,41,2))

# for conversion between UVB names and indices
UVBs = {"HM12" : 0, "HM05" : 1, "FG11" : 2, "KS18" : 3}
```

```
56  UVBsReverse = ["HM12", "HM05", "FG11", "KS18"]
57
58  # iterable lists of parameters
59  nHs = (0, -2, -4, -6)
60  zs = (0, 2, 3, 6, 9)
61  Zs = (0, 1, 2)
62
63  # colors for plotting
64  colors = range(10)
65  colors[0] = "r"
66  colors[-2] = "g"
67  colors[-4] = "b"
68  colors[-6] = "cyan"
69
70  # read in all cooling data
71  for file in os.listdir("cooling"):
72      # parse filename
73      z = int(file[13])
74      nH = int(file[17])
75      if file[20] == "X":
76          Z = 3
77      else:
78          Z = int(file[20])
79
80      UVB = file[22:26]
81
82      # Extract relevant cooling data and attach to big array
83      tempcooling = np.genfromtxt("cooling/" + file, usecols=(1,3),
84                                  comments="#")
85      tempheating = np.genfromtxt("cooling/" + file, usecols=(1,2),
86                                  comments="#")
87      cooling[z][Z][nH][UVBs[UVB]] = tempcooling
88      heating[z][Z][nH][UVBs[UVB]] = tempheating
89
90  # starting values of plotbrowser
91  z = 0
92  Z = 0
93  UVB = "HM12"
94  HM12isUnity = False
95  key = ""
96
97  # Enter matplotlib interactive mode and prepare figure
98  plt.ion()
99  # Historically grown Axes names:
100 # Axes = left most graph
101 # Axes3 = middle graph
102 # Axes2 = right most graph
103 fig, (Axes, Axes3, Axes2) = plt.subplots(1, 3, figsize=(30,10))
104
105 # Initial instructions
106 print("Focus this terminal. Press W/S to change redshift. Press A/D to \
107         change metallicity. Press Q/E to change UVB. \nPress Space to \
108         view differences of cooling functions relative to HM12.\nPress \
109         R to quit.\n")
110 # initial state
111 print("redshift: " + str(z) + "\nmetallicity: -" + \
112       str(Z) + "\nUVB: " + UVB)
113
```

```
# Display plots until user presses "r"
while key != "r":
    # Clear Axes for new plotting
    Axes.cla()
    Axes2.cla()
    Axes3.cla()

    # Actually plot cooling functions

    # Case one: Regular plotting
    if HM12isUnity == False:
        # Prepare Axes

        # regular cooling (left most graph)
        Axes.set_yscale("log")
        Axes.set_xscale("log")
        Axes.set_ylim(1e-24, 1e-21)
        Axes.set_ylabel(r"Normalized Cooling $\Lambda / n_{H}^{2}$ in " + \
        r"$\mathregular{erg}\ \mathregular{cm}^3\ \mathregular{s}^{-1}$")
        Axes.set_xlabel("T in K")

        # abs of cooling - heating/nH^2 (middle graph)
        Axes3.set_yscale("log")
        Axes3.set_xscale("log")
        Axes3.set_ylim(1e-24, 1e-21)
        Axes3.set_ylabel(r"Normalized $(\Lambda -H) / n_{H}^{2}$ in " + \
        r"$\mathregular{erg}\ \mathregular{cm}^3\ \mathregular{s}^{-1}$")
        Axes3.set_xlabel("T in K")

        # Plot functions into prepared Axes
        for nH in nHs:
            #regular
            Axes.plot(cooling[z][Z][-1*nH][UVBs[UVB]][...,0],
                      cooling[z][Z][-1*nH][UVBs[UVB]][...,1] / \
                        math.pow(10, 2*nH), colors[nH],
                      label="n_H: 10^" + str(nH) )
            #with heating
            Axes3.plot(cooling[z][Z][-1*nH][UVBs[UVB]][...,0],
                       abs(cooling[z][Z][-1*nH][UVBs[UVB]][...,1] - \
                        heating[z][Z][-1*nH][UVBs[UVB]][...,1]) / \
                        math.pow(10, 2*nH), colors[nH],
                       label="n_H: 10^" + str(nH) )

    # Case two: Plot relative to HM12/alternate viewing mode
    else:
        # Prepare Axes

        # regular cooling (left most graph)
        Axes.set_yscale("linear")
        Axes.set_xscale("log")
        Axes.set_ylabel("Cooling relative to HM12 cooling")
        Axes.set_xlabel("T in K")

        #accounting for heat (middle graph)
        Axes3.set_yscale("symlog")    # symlog!
        Axes3.set_xscale("log")
        Axes3.set_ylabel("Cooling relative to HM12, accounting for heating")
        Axes3.set_xlabel("T in K")
```

```
172
173         # Plot functions. Divide by HM12's function each time
174         for nH in nHs:
175             #regular
176             Axes.plot(cooling[z][Z][-1*nH][UVBs[UVB]][...,0],
177                       ( cooling[z][Z][-1*nH][UVBs[UVB]][...,1] / \
178                         math.pow(10, 2*nH) ) / \
179                       ( cooling[z][Z][-1*nH][UVBs["HM12"]][...,1] / \
180                         math.pow(10, 2*nH) ),
181                       colors[nH], label="n_H: 10^" + str(nH) )
182             #with heating
183             Axes3.plot(cooling[z][Z][-1*nH][UVBs[UVB]][...,0],
184                        ( abs(cooling[z][Z][-1*nH][UVBs[UVB]][...,1] - \
185                          heating[z][Z][-1*nH][UVBs[UVB]][...,1]) / \
186                          math.pow(10, 2*nH) ) / \
187                        ( abs(cooling[z][Z][-1*nH][UVBs["HM12"]][...,1] - \
188                          heating[z][Z][-1*nH][UVBs["HM12"]][...,1]) / \
189                          math.pow(10, 2*nH) ),
190                        colors[nH], label="n_H: 10^" + str(nH) )
191
192     Axes.legend()
193     Axes3.legend()
194
195
196     # continuum data is big, reload from file each time instead of
197     # keeping it in memory all at once
198     # smallcontinuums assumes that the continuum looks the same across
199     # grid runs
200     if Z < 3:
201         UVBdata = np.genfromtxt("smallcontinuums/z" + str(z) + "_nH" + \
202                                 str(-1*nH) + "_Z" + str(Z) + "_" + UVB + \
203                                 "_cont", skip_header=1,usecols=(0,1))
204     elif Z == 3:
205         UVBdata = np.genfromtxt("smallcontinuums/z" + str(z) + "_nH" + \
206                                 str(-1*nH) + "_ZX_" + UVB + "_cont",
207                                 skip_header=1,usecols=(0,1))
208
209     # Turn it into plottable format
210     UVBdatax = np.empty(UVBdata.shape[0])
211     UVBdatay = np.empty(UVBdata.shape[0])
212     for i in range(0, UVBdata.shape[0]):
213         UVBdatax[i] = UVBdata[i][0]
214         UVBdatay[i] = UVBdata[i][1]
215
216     # Actually plot continuum data (right most graph)
217     # prepare Axes
218     Axes2.set_yscale("log")
219     Axes2.set_xscale("log")
220     Axes2.set_ylim(1e-11, 1e0)
221     Axes2.set_xlim(1e-7, 1e7)
222
223     Axes2.set_ylabel("4*pi*J in erg/cm^2/s")
224     Axes2.set_xlabel("E in Ryd")
225     # plot
226     Axes2.plot(UVBdatax, UVBdatay, "r", label="UVB: " + UVB)
227     Axes2.legend()
228
229     fig.canvas.draw()
```

```
230
231
232     # Handle user input
233     key = getch()
234     if key == "w":
235         if z < 9:
236             z = zs[zs.index(z)+1]
237     elif key == "s":
238         if z > 0:
239             z = zs[zs.index(z)-1]
240     elif key == "d":
241         if Z > 0:
242             Z = Z-1
243     elif key == "a":
244         if Z < 3:
245             Z = Z+1
246     elif key == "q":
247         if UVBs[UVB] > 0:
248             # cycling through UVBs by transforming to index and back
249             UVB = UVBsReverse[UVBs[UVB]-1]
250     elif key == "e":
251         if UVBs[UVB] < 3:
252             # cycling through UVBs by transforming to index and back
253             UVB = UVBsReverse[UVBs[UVB]+1]
254     elif key == " ":
255         HM12isUnity = not HM12isUnity
256
257     # Provide instructions and updated status information
258     os.system("clear")
259     print("Focus this terminal. Press W/S to change redshift. Press A/D\
260           to change metallicity. Press Q/E to change UVB. \nPress Space\
261           to view differences of cooling functions relative to HM12.\n \
262           Press R to quit.\n")
263     if Z == 3:
264         print("redshift: " + str(z) + "\nmetallicity: primordial" + \
265               "\nUVB: " + UVB)
266     else:
267         print("redshift: " + str(z) + "\nmetallicity: -" + str(Z) + \
268               "\nUVB: " + UVB)
269
270
271 exit()
272
273
```

C Table of Abbreviations and important symbols

This table lists all of the abbreviations used in this document.

Abbreviation	Complete name/Reference
UVB	UltraViolet Background
CMB	Cosmic Microwave Background
CIE	Collisional Ionization Equilibrium
HM05	Haardt & Madau 2005
HM12	Haardt & Madau (2012)
FG09	Faucher-Giguère et al. (2009)
KS18	Khaire & Srianand (2018)
$Z_\odot$	Solar metallicity
z	Redshift
$n_{\rm H}$	Number density of Hydrogen atoms
$\Gamma_{\rm HI}$	Photoionization rate of HI
$\Gamma_{\rm HeII}$	Photoionization rate of HeII
T	Temperature
Λ	Cooling rate
QSO	Quasi Stellar Object
SED	Spectral Energy Distribution

References

Barkana R., Loeb A., 2001, Physics Reports, 349, 125

Bond J. R., Kofman L., Pogosyan D., 1996, Nature, 380, 603

Dopita M. A., Sutherland R. S., 2003, Astrophysics of the diffuse universe

Draine B. T., 2011, Physics of the Interstellar and Intergalactic Medium

Faucher-Giguère C.-A., Lidz A., Zaldarriaga M., Hernquist L., 2009, ApJ, 703, 1416

Ferland G. J., et al., 2013, Rev. Mexicana Astron. Astrofis., 49, 137

Gnat O., Ferland G. J., 2012, ApJS, 199, 20

Gnedin N. Y., Hollon N., 2012, ApJS, 202, 13

Haardt F., Madau P., 1996, ApJ, 461, 20

Haardt F., Madau P., 2001, in Neumann D. M., Tran J. T. V., eds, Clusters of Galaxies and the High Redshift Universe Observed in X-rays. p. 64 (arXiv:astro-ph/0106018)

Haardt F., Madau P., 2012, ApJ, 746, 125

Hopkins P. F., Kereš D., Oñorbe J., Faucher-Giguère C.-A., Quataert E., Murray N., Bullock J. S., 2014, MNRAS, 445, 581

Inoue A. K., Shimizu I., Iwata I., Tanaka M., 2014, MNRAS, 442, 1805

Kannan R., Vogelsberger M., Stinson G. S., Hennawi J. F., Marinacci F., Springel V., Macciò A. V., 2016, MNRAS, 458, 2516

Khaire V., Srianand R., 2015a, MNRAS, 451, L30

Khaire V., Srianand R., 2015b, ApJ, 805, 33

Khaire V., Srianand R., 2018, preprint, (arXiv:1801.09693)

Khaire V., Srianand R., Choudhury T. R., Gaikwad P., 2016, MNRAS, 457, 4051

Madau P., Dickinson M., 2014, ARA&A, 52, 415

Osterbrock D. E., Ferland G. J., 2006, Astrophysics of gaseous nebulae and active galactic nuclei

Planck Collaboration et al., 2016, A&A, 594, A13

Shen S., Wadsley J., Stinson G., 2010, MNRAS, 407, 1581

Somerville R. S., Davé R., 2015, ARA&A, 53, 51

Spatschek K.-H., 2018, Astrophysik, doi:10.1007/978-3-662-55467-8.

Wiersma R. P. C., Schaye J., Smith B. D., 2009, MNRAS, 393, 99

Selbständigkeitserklärung

Hiermit erkläre ich, die vorliegende Arbeit selbständig verfasst zu haben und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel benutzt zu haben.

Stefan Lüders
München, 2018-06-27