A COMPUTER PROGRAM FOR PRODUCING SYNTHETIC NON-LTE MOLECULAR SPECTRA FROM ASTROCHEMICAL MODELS

BY

JOSEPH EPSTEIN

A thesis submitted

in partial fulfillment of the requirements for the degree:

Bachelors of Science

Major Subject: Astronomy-Physics

Department: Astronomy

Supervisor: Prof. Eric Herbst

University of Virginia

Charlottesville, Virginia

May 2018

Astronomy & Astrophysics manuscript no. aanda
May 12, 2018

A program for producing synthetic non-LTE molecular transitions from astrochemical models

J. Epstein

Department of Astronomy, University of Virginia, Charlottesville, VA 22903, USA

May 15, 2018

ABSTRACT

Context. Molecular transitions can be observed and used to probe the physical and chemical conditions of molecular clouds, which are the building blocks for astrophysical structures.
Aims. To provide a tool for analyzing the variation in molecular transition lines throughout a time-dependent chemical model, and for making comparisons between theoretical and observed molecular transitions in non-LTE astrochemical environments.
Methods. Data from a time-dependent astrochemical model and its subsequent species abundances, calculated with the NAUTILUS code, are used to obtain the necessary input information for the RADEX radiative transfer code. This input data is run through RADEX to predict the strength of the spectral lines and their frequencies, which are then plotted in the form of a line spectrum.
Results. The analysis of the tool is sufficient for comparing different transition intensities at the same time in the model, as well as how the intensity of particular transition lines changes across all time steps in the model.

1. Introduction

Astrochemistry plays an important role in our understanding of the star- and planet-formation processes across the universe (e.g. Bergin et al. 2007; Kennicutt 1998; van Dishoeck & Blake 1998). Observations of molecular transitions at millimeter and infrared wavelengths can be supplemented by careful and detailed modeling to be a powerful tool for investigating the physical and chemical conditions of astrophysical objects (e.g., Genzel 1991; Black 2000). Different molecules have been developed as tracers for these differing physical and chemical conditions (van Dishoeck & Hogerheijde 1999).

Molecular clouds are large and dense enough t0 permit the formation of many types of molecules, in contrast to the predominantly ionized atomic gas in other areas of the interstellar medium (ISM). Higher density regions in the clouds, with lots of dust and gas cores, form clumps which contain more complex chemistry and are the beginning of star formation. At great depths within clouds, other molecules can be seen from their transitions, and more than 200 chemical species have been identified within constituent gas. When there are low densities and temperatures, the chemistry is very exotic, compared to laboratory experiments on Earth; some quite unstable species can exist in space because there is not enough energy to convert them to more-stable forms.

One of the major methods of studying these molecular clouds is analyzing their spectra, as molecules and their transition lines provide valuable insight into their local environment. This paper presents a tool which creates synthetic molecular spectra which can then be used to identify species within molecular clouds, as well as individual transition lines associated with specific physical and chemical conditions. It can also be used for analyzing the variation of transition line intensities for a species throughout the entire model. The transition lines are calculated taking the species' abundances, determined by the NAUTILUS astrochemical model of a time-dependent evolution with a large number of species connected by a vast network of chemical processes, and using them in the RADEX radiative transfer code to calculate the intensities of atomic and molecular transition lines.The synthetic spectra can be used to identify species within molecular clouds, as well as transition lines associated with specific physical and chemical conditions.

The paper is set up as follows. Section 2 describes the RADEX code while section 3 describes the NAUTILUS code, with each section covering some relevant scientific formalism and the inputs and outputs associated with each. Section 4 describes the tool developed as part of this work, while Section 5 demonstrates the tool being used via a case study of HCO^+, and Section 6 concludes the paper.

2. RADEX Program Model

RADEX is a computer program used to calculate the intensities of atomic and molecular lines produced in a uniform medium, based on statistical equilibrium calculations involving collision and radiative processes and including radiation from background sources (Van der Tak et al. 2007). It treats optical depth effects with an escape probability method and makes use of molecular data files maintained in the Leiden Atomic and Molecular Database (LAMDA) (Schöier et al. 2005). The program is available at: http://home.strw.leidenuniv.nl/ moldata/radex.html

2.1. Radex Formalism

2.1.1. Non-LTE

The assumption that the excitation temperature equals the kinetic temperature leads to a case known as Local Thermodynamic Equilibrium (LTE). LTE occurs when the radiant energy absorbed by a molecule is distributed via collisions before being emitted as radiation, and it is needed for Planck's law and Kirchhoff's law to apply. For many interstellar and circumstellar media, the density is too low to attain LTE, in which case the non-LTE analysis by RADEX can be used. In addition to low densities, physically dynamic areas such as shocks can disrupt environments and invalidate an assumption of LTE.

2.1.2. Radiative Transfer and Molecular Excitation

The specific intensity, I_ν, is defined as the amount of energy passing through a surface normal to the path, per unit time, surface, bandwidth (measured in frequency units), and solid angle. A source function is then defined by:

$$S_\nu = \frac{j_\nu}{\alpha_\nu} \tag{1}$$

Here j_ν and α_ν are the local emission and extinction coefficients, respectively. By defining the optical depth measured along the light path as $d\tau_\nu = \alpha_\nu ds$, the radiation transport equation can then be written in its integral form as:

$$I_\nu = I_\nu(0)e^{-\tau_\nu} + \int_0^{\tau_\nu} S_\nu(\tau'_\nu)e^{(\tau_\nu - \tau'_\nu)}d\tau'_\nu, \tag{2}$$

where I_ν is the radiation emerging from the medium and $I_\nu(0)$ is the "background" radiation entering the medium. The above equations hold for both the continuum radiation which is emitted over a large bandwidth, and for spectral lines, which arise when the local emission and absorption properties change drastically over a very small frequency interval, due to the presence of molecules.

In situations where LTE is unattainable, RADEX utilizes the assumption of statistical equilibrium (SE):

$$\frac{dn_i}{dt} = 0 = \sum_{j\neq i}^{N} n_j P_{ji} - n_i \sum_{j\neq i}^{N} P_{ji} = \mathcal{F}_i - n_i \mathcal{D}_i, \tag{3}$$

where P_{ij}, the destruction rate coefficient of level i, and its formation rate coefficient P_{ji} are given by

$$P_{ij} = \begin{cases} A_{ij} + B_{ij}\bar{J}_\nu + C_{ij} & (i > j) \\ B_{ij}\bar{J}_\nu + C_{ij} & (i < j) \end{cases} \tag{4}$$

In Eq. (4),

$$B_{ij}\bar{J}_\nu = B_{ij}\int_0^\infty J_\nu \phi(\nu) d\nu \tag{5}$$

is the number of induced radiative (de-)excitations from state i to state j per second per particle in state i, and

$$J_\nu = \frac{1}{4\pi}\int I_\nu d\Omega \tag{6}$$

is the specific intensity I_ν integrated over solid angle $d\Omega$ and averaged over all directions. A_{ij} is the spontaneous emission term, while C_{ij} is the collisional term. Both emission and absorption are equal unless there is degeneracy. The SE equations thus include the effects of non-local radiation. It is also assumed that the state-specific rates of formation $\mathcal{F}_i$ [$cm^3 s^{-1}$] and destruction, $\mathcal{D}_i$ [s^{-1}] are zero to ensure that the radiative transfer is solved independently of assumptions about chemical processes. The difficulty in solving radiative transfer problems is the interdependence of the molecular level populations and the local radiation field, requiring iterative solution methods. However, if only the global properties of an interstellar cloud are of interest, the calculation can be greatly simplified through the induction of the geometrically averaged escape probability β, the probability that a photon will escape the medium from where it was created. This probability depends only on the optical depth τ and is related to the intensity within the medium, ignoring background radiation and any local continuum, through

$$\bar{J}_{\nu_{ul}} = S_{\nu_{ul}}(1-\beta). \tag{7}$$

Several authors have developed detailed relations between β and τ for specific geometric assumptions, and the RADEX program offers the user a choice of three such expressions. The one used in my analysis is the expression derived for an expanding spherical shell, the so-called Sobolev or large velocity gradient (LVG) approximation (Sobolev 1960; Castor 1970; Elitzur 1992, pp. 42–44). This method is widely applied for moderate velocity gradients, to mimic turbulent motions. RADEX uses the formula by Mihalas (1978) and De Jong et al. (1980) for this geometry:

$$\beta_{LVG} = \frac{1}{\tau}\int_0^\tau e^{-\tau'}d\tau' = \frac{1-e^{-\tau}}{\tau} \tag{8}$$

2.2. The RADEX Program

RADEX is a non-LTE radiative transfer code that uses the escape probability formulation assuming an isothermal and homogeneous medium without large-scale velocity fields. The program iteratively solves the statistical equilibrium equations starting from optically thin statistical equilibrium for the initial level populations.

The program can handle up to seven collision partners simultaneously, however for most species in dense molecular clouds, H_2 is the main collision partner. The output of the program is the background-subtracted line intensity in units of the equivalent radiation temperature in the Rayleigh-Jeans limit. The radiation peak temperature T_R can be directly compared to the observed antenna temperature corrected for the optical efficiency of the telescope. However, it should be emphasized that RADEX contains no information about the geometry or length scale and that it is assumed that the source fills the antenna beam.

Calculations with RADEX proceed as follows. A first guess of the populations of the molecular energy levels is produced by solving statistical equilibrium in the optically thin case. The only radiation taken into account is the unshielded background radiation field; internally produced radiation is not yet available. The solution for the level populations allows calculation of the optical depths of all the lines, which are then used to re-calculate the molecular excitation. The new calculation treats the background radiation in the same manner as the internally produced radiation. The program iteratively finds a consistent solution for the level populations and the radiation field. When the optical depths of the lines with $\tau > 10^{-2}$ are stable from one iteration to the next to a given tolerance (default 10^{-6}), the program writes output and stops.

There are several ways in which RADEX can be used to analyze molecular line observations. In most of these applications, the modeled quantity is the velocity-integrated line intensity, as the excitation is assumed to be independent of velocity. If the H_2 column density is known from other observations, the ratio of the two column densities gives the molecular abundance, averaged over the source.

The limited number of free parameters makes RADEX very useful to rapidly analyze large datasets. However, a limitation of the program is that only one molecule is treated at a time, so the effects of line overlap are not taken into account. Such overlaps may occur both at radio and at infrared wavelengths (e.g., Expósito et al. 2006) and can be very common in chemically-rich regions (Belloche et al 2015). Ratios of other lines and other

molecules may be easily computed using the Python scripts included in the RADEX distribution, and the program may also be used to create synthetic spectra.

2.3. RADEX Input and Output

The input parameters to RADEX are the following:

1. The name of the molecular data file containing line data of the species, including energy levels, statistical weights, Einstein A-coefficients and collisional rate coefficients.
2. The name of the file to write the output to.
3. The frequency range for the output file [GHz]. All transitions from the molecular data file are always taken into account in the calculation, but often it is practical to write only a limited set of lines to the output.
4. The kinetic temperature of the cloud [K].
5. The number of collision partners to be used. Most users will want H_2 as only collision partner.
6. The name and density [cm^{-3}] of each collision partner.
7. The temperature of the background radiation field [K].
8. The column density of the molecule [cm^{-2}].
9. The full-width half max FWHM line width [km s^{-1}].

The output file written by RADEX first replicates the input parameters, and then lists the following quantities for each spectral line within the specified frequency range.

1. Quantum numbers, upper state energy [K], frequency [GHz], and wavelength [μm]. These numbers are just copied from the molecular data file, which usually comes from the LAMDA database referenced in the beginning of Section 2.
2. The excitation temperature [K]. In general, different lines have different excitation temperatures.
3. The line optical depth, defined as the optical depth of the equivalent rectangular line shape ($\phi_\nu = \frac{1}{\Delta\nu}$).
4. The line intensity, defined as the Rayleigh-Jeans equivalent temperature T_R [K].
5. The line flux, defined as the velocity-integrated intensity, both in units of $Kkm\ s^{-1}$ (common in radio astronomy) and of $erg\ cm^{-2}\ s^{-1}$ (common in infrared astronomy).

3. Nautilus Program Model

NAUTILUS is optimized to model time-dependent evolution of a large set of gas-phase and surface species linked through thousands of gas-phase, gas-grain, and surface processes. In this model, the abundance of each species is computed by solving a set of rate equations for the aforementioned processes, with gas-phase, gas-grain, and grain-surface chemistries connected via accretion, diffusion, and desorption.

The code is written in Fortran 90 and uses the LSODES (Livermore Solver for Ordinary Differential Equations with general Sparse Jacobian matrix) solver, part of ODEPACK (Hindmarsh 1983). It is adapted from the original gas-grain model of Hasegawa & Herbst (1993) and its subsequent evolutions made over the years at Ohio State University. NAUTILUS has been intensively utilized in various studies of molecular cloud, hot molecular core (HMC), and protoplanetary disk chemistry, e.g. predictions for ALMA (Semenov et al. 2008).

The 3-phase model (gas, grain surface, and grain mantle) assumes both accretion and desorption only occur between the gas and the outermost surface layer, as in Hasegawa & Herbst (1993b). However, it also assumes the chemical reactions can occur in the mantle just as on the surface, but with a smaller diffusion rate for the species. As opposed to previous models, it also assumes the bulk diffusion is driven by the diffusion of water molecules in the ice, as suggested by molecular dynamics simulations and experiments (Ghesqui'ere et al. 2015).

The low diffusion-to-binding energy ratio, used for surface, makes reactions involving light atoms or molecules other than H more competitive in front of hydrogenation re- actions. This leads to a lower abundance of species formed by hydrogenation chains on the grains as well as in the gas phase (Ruaud et al. 2016). The model predictions are confronted to ice observations in the envelope of low-mass and massive young stellar objects as well as toward background stars. After 10^6 yrs, they found a dramatic fall down of the gas phase abundances due to the large accretion on dust which is not counterbalanced by the desorption. This strongly constrains the chemical-age of these clouds to be of the order of a few 10^5 yrs and strongly rejects the largest ages of few 10^6 yrs. An extensive description of the NAUTILUS model can be found in Ruaud et al. (2016).

3.1. NAUTILUS Formalism

3.1.1. General Equations

Abundances of each species are obtained by solving rate equations for gas-phase, grain-surface, and grain-mantle chemistries. The differential equations governing gaseous, surface, and mantle concentrations of a species i, denoted as $n(i), n_s(i)$ and $n_m(i)$ [cm^{-3}] respectively, are:

$$\frac{dn(i)}{dt}\Big|_{tot} = \sum_l \sum_j k_{lj} n(l) n(j) + k_{diss}(j) n(j) + k_{des}(i) n_s(i) - k_{acc}(i) n(i) - k_{diss}(i) n(i) - n(i) \sum_l k_{ij} n(j) \quad (9)$$

$$\frac{dn_s(i)}{dt}\Big|_{tot} = \sum_l \sum_j k^s_{lj} n_s(l) n_s(j) + k^s_{diss}(j) n_s(j) + k_{acc}(i) n(i) + k^m_{swap}(i) n_m(i) + \frac{dn_m(i)}{dt}\Big|_{m\to s} - n_s(i) \sum_j k^s_{ij} n_s(j) - k_{des}(i) n_s(i) - k^s_{diss}(i) n_s(i) - k^s_{swap}(i) n_s(i) - \frac{dn_s(i)}{dt}\Big|_{s\to m} \quad (10)$$

$$\frac{dn_m(i)}{dt}\Big|_{tot} = \sum_l \sum_j k^m_{lj} n_m(l) n_m(j) + k^m_{diss}(j) n_m(j) + k^m_{swap}(i) n_s(i) + \frac{dn_s(i)}{dt}\Big|_{s\to m} - n_m(i) \sum_j k^m_{ij} n_m(j) - k^m_{diss}(i) n_m(i) - k^m_{swap}(i) n_m(i) - \frac{dn_m(i)}{dt}\Big|_{m\to s} \quad (11)$$

where k_{ij}, k^s_{ij}, and $k^m_{ij}[cm^3 s^{-1}]$ represent the rate coefficients for reactions between species i and j in the gas-phase, on the grain surface, and in the mantle, respectively. k_{acc} and $k_{des}[s^{-1}]$ are respectively the accretion/evaporation rate coefficient of individual species onto/from the grain surface. In the model k_{des} can be thermal or non-thermal. $k_{diss}[s^{-1}]$ gathers dissociation rate coefficients by (1) direct cosmic-ray processes, (2) secondary UV photons induced by cosmic-rays and (3) the standard interstellar UV photons (Wakelam et al. 2012). k^s_{swap} and $k^m_{swap}[s^{-1}]$ are the surface-mantle and mantle-surface swapping rates, respectively. $\frac{dn_s(i)}{dt}\Big|_{s\to m}$ and $\frac{dn_m(i)}{dt}\Big|_{m\to s}$ are related to the individual transfer of the species *i* from the surface to the mantle and *vice versa*.

3.1.2. Accretion, Diffusion, and Desorption

Accretion, diffusion and thermal desorption rate coefficients are computed based on Hasegawa et al. (1992). NAUTILUS assumes a sticking probability of 1.0 for all neutral species, except for H and H_2. For these, it uses the temperature dependent expressions of Matar et al. (2010) and Chaabouni et al. (2012) derived from laboratory experiments. The non-thermal desoprtion mechanisms considered are (1) the cosmic-rays desorption mechanism (see Hasegawa & Herbst 1993a), (2) the chemical desorption mechanism (desorption due to the exothermicity of surface reactions, set to 1% efficiency) (see Garrod et al. 2007), and (3) the photo-desorption. The non-thermal mechanisms covered in other studies such as Roberts et al. (2007) are not included.

In regards to the photo-desorption, Ruaud et al. (2016) provides a summary and insight on the emerging science from recent experiments involving the process. Following these new insights and considering the lack of knowledge on the photo-desorption process, the NAUTILUS code follows the recommendations of Bertin et al. (2012, 2013) and uses a simplistic approach which considers a single photo-desorption yield for all the molecules rather than individual ones determined from experiments.

3.1.3. Gas-Phase Chemistry

The gas-phase bimolecular reaction rates are given by the modified Arrhenius equation as a function of temperature T (in Kelvin)

$$k(T) = \alpha \left(\frac{T}{300}\right)^{\beta} exp\left(-\frac{\gamma}{T}\right). \quad (12)$$

where α is the value of the reaction rate, β characterizes the temperature dependence of the rate, and γ is the activation barrier (in Kelvin) for exothermic and endothermic reactions with activation energies.

Ionization and dissociation rates by (1) direct impact of cosmic ray particles, (2) secondary UV photons induced by cosmic ray/H_2 interactions, and (3) interstellar FUV photons are also calculated by the model according to the following equations (Wakelam et al. 2012):

$$k_{CR} = A_i \zeta_{CR}, \quad (13)$$

$$k_{FUV} = A\,(exp - CA_V)\chi, \quad (14)$$

where ζ_{CR} is the cosmic ray ionization rate (typically 1.3 x 10^{-17} s^{-1}), A_V is the visual extinction, χ the FUV flux, A_i and A are parameters which represent the rate coefficients and exp(-C A_V) takes into account the continuum attentuation from the dust.

3.2. NAUTILUS Input and Output

The inputs for NAUTILUS include a datafile containing the evolutionary structure of a chemical model via the physical parameters of interest for each timestep, as well as the initial abundances. It also requires the relevant chemical processes which are adopted from various networks, such as the gas-phase network *kida.uva.2011* used by Ruaud et al. (2015). NAUTILUS derives a block containing the ordinary differential equations for the reactions and then solves each of them for each timestep. The solutions describe how the physical conditions and abundances change throughout the evolution, with the output file containing the abundances at each timestep (relative to H) [number ratio] for several spatial positions of a specific species.

4. The Program

This tool uses molecular data files, as well as kinetic temperatures and the calculated abundances of a specific species throughout an astrochemical model, to create a synthetic spectrum displaying the expected intensity of the species' transition lines throughout the evolution. It is particularly useful in situations of non-LTE, and with increasingly complex chemistry and environments being observed throughout the universe, this tool was developed to assist in probing regions for specific species or physical conditions by providing synthetic spectra to be compared with observations.

4.1. Basic Capabilities and Limitations

The tool is limited by the capabilities of both RADEX and NAUTILUS. Therefore, any species which is incompatible with either RADEX or NAUTILUS is beyond the scope of this tool. For a given time-dependent model, it can either provide a spectrum showing the transitions for a species at a singular specified timestep in order to analyze specific transition lines, or create an animation showing the transitions for each timestep in order to analyze how the line intensities vary throughout the entire evolution.

The code is written to be run using a MAC OSX shell, however commands should be easily modified for use with Unix Linux shells. Currently, the tool can only analyze one species at a time. It also runs RADEX calculations with H_2 as its only collision partner.

4.2. Method of Use

Using the tool proceeds as follows. Upon running the python script, there is a prompt asking the user if they already have a RADEX input file prepared for the species. If yes, the tool then asks the user for the name of the input file. If no, the tool prompts the user for the necessary RADEX inputs. It first asks what you wish to name your RADEX input and output files, then to provide the name of your molecular data file. Next, it asks for the upper and lower bounds of the frequency band of interest [GHz], the background temperature [K], and the line width [km s^{-1}]. Finally, the user is prompted to enter the name of their abundance file (the output from NAUTILUS), as well as the associated structural evolution file used for calculating the abundances. After all of the necessary initial inputs have been obtained by the tool, it then allows the user to choose whether they want a spectrum for a singular specific time, or for all of the timesteps in the model. The first option then requires the use to input which year in the model they would like to analyze before running, while the second option immediately begins running upon being chosen.

4.2.1. Singular Timestep Analysis

Once the user chooses the first option and enters the year they would like to analyze, the tool finds the timestep closest to the given year. For that time, it reads the kinetic temperature and calculates the H_2 density from the structure evolution file, and calculates the column density of the species using the abundance

at the time. Using all this information, it writes and saves a properly formatted RADEX input file to the current directory. Next, the input file is run through RADEX and data from the output file is used to plot the peak intensity [K] of all the transition lines on a single plot. It also plots each individual line in order to show the width of each line's Gaussian profile resulting from Doppler broadening, which is the dominant line broadening mechanism in the interstellar medium (van der Tak et al. 2007). These plots are then displayed in a grid to allow for the individual line intensities and widths at one timestep to be compared with one each other.

4.2.2. Entire Evolution Analysis

Once the user chooses the second option, the tool then makes a subdirectory named after the user's choice of input file name. The process is then the same for obtaining the necessary RADEX inputs and writing the RADEX input file, however it is now done for every timestep within the structure evolution file. For each of the timesteps, the tool also plots the peak intensities [K] of all the transition lines on a single plot. Rather than comparing different transition lines for the same timestep, this option turns all the single timestep plots into an animation visualizing the varying peak intensities of the transition lines throughout the entire structure evolution. This allows for values of an individual transition line to be compared across timesteps.

4.3. Tool Input and Output

For a structural evolution file to be used as an input for this tool, the column with the timesteps should be labeled 'time', the abundance values should be in the second column, the density values should be in the third column, and the kinetic temperatures should be in the fourth column. The abundance file used should have exactly one line at the top which is not data. The tool outputs the aforementioned plots and/or animation, saving all plots as pdf files. All of these formating requirements are exemplified in Figures 1 and 2.

```
# time      log(Av)      log(n)      log(Tg)      log(Td)      Vd
# (yr)      (mag)        (cm-3)      (K)          (K)          (cms-1)
1.0000000e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
1.1497570e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
1.3219411e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
1.5199111e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
1.7475284e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
2.0092330e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
2.3101297e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
2.6560878e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
3.0538555e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
3.5111917e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
4.0370173e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
4.6415888e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
5.3366992e+00 0.0000000e+00 4.6989700e+00 1.0000000e+00 1.0000000e+00 0.0000000e+00
```

Fig. 1. Example structural evolution file format.

```
# time [year]; Each column is the abundance\
 (relative to H) [number ratio] for several\
 spatial positions

 1.0000000E+00 1.211255E-13
 1.1497570E+00 1.239354E-13
 1.3219411E+00 1.264916E-13
 1.5199111E+00 1.287868E-13
 1.7475284E+00 1.308189E-13
 2.0092330E+00 1.325931E-13
 2.3101297E+00 1.341249E-13
 2.6560878E+00 1.354422E-13
 3.0538555E+00 1.365864E-13
 3.5111917E+00 1.376098E-13
```

Fig. 2. Example abundance file format.

5. HCO^+ Case Study

Here an example of the tool is provided for the species HCO^+ in a model which experiences a shock starting 1 million years into the model. The shock reaches a peak dust velocity of approximately 1.14e+06 [cm s^{-1}]. An astrophysical shock brings about significant thermal changes in an environment, thus it will most certainly be in non-LTE and consequently relevant for RADEX. An example of each of the two analysis options is presented below.

5.1. Singular Time

The first option allows the user to specify a specific year for which they would like to analyze the HCO^+ transition lines. When running the script, the dialogue is as shown in Figure 3 below. When inserting the name of the various files, the tool will add the proper file extension to the end of the name if the user's input does not already contain it. The background temperature and line width are set to be equal to 2.73 and 5.0, respectively, if the user leaves the inputs blank. After giving the name of the abundance and structural evolution file, the user is prompted to choose an analysis option. From Figure 3 below, it can be seen the user specifies the timestep of interest before the code runs.

```
Do you have a Radex input file? (y/n): n

Please insert data manually.

What would you like to name your input file(s)?: hco+

Output file name?: hco+

Molecular data file?: hco+

Frequency range max? (GHz): 10000

Frequency range min? (GHz): 1

Background temperature? (K): 2.73

Line width?(km/s-1): 5.0

Abundance file?: HCO+

Structural Evolution file?: structure_evolution

Enter 1 if you want a spectrum at 1 specific time.
Enter 2 if you want spectra for all times.
1

For what year would you like the abundance?: 1003500
```

Fig. 3. Example display when running the tool with the first option.

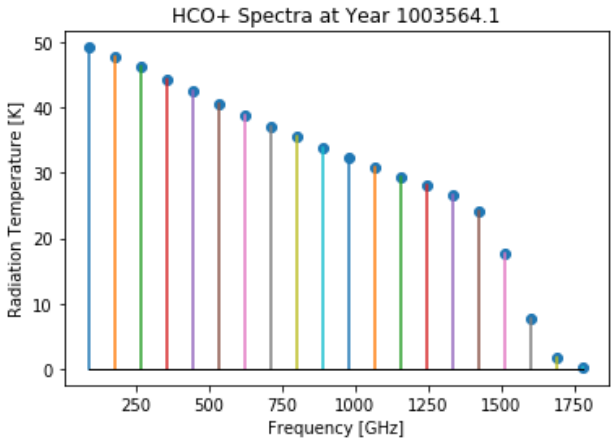

Fig. 4. Plot showing all HCO^+ transition lines (from 1-10000 GHz) and their intensities at the specified year.

In Figure 4, all of the transition lines are plotted together to allow for the analysis of their relative intensities. The first line is the 0-1 transition, the second is the 1-2 transition, and so on. Furthermore, the y-axis displays the line intensity, defined as the Rayleigh-Jeans equivalent temperature T_R [K]. Each of these lines is also individually plotted in a graph as shown in Figure 5. The title at the top of the graph gives the transition the line represents, as well as the frequency value of the line peak.

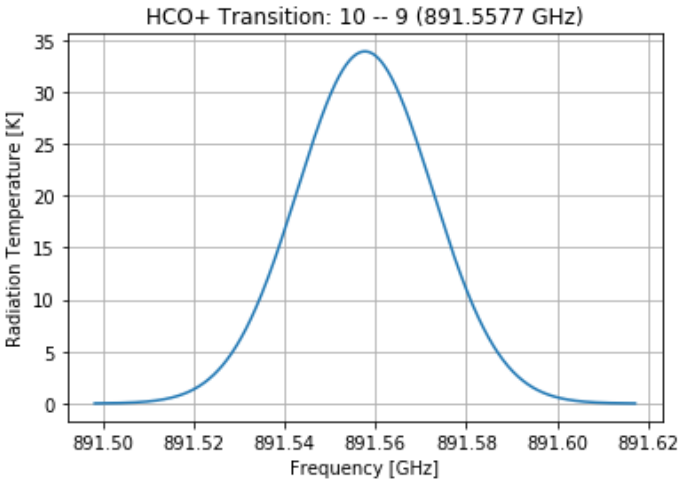

Fig. 5. Example plot of an individual transition line.

While the axis of the individual plots of the grid shown in Figure 6 are too small to be read, that is not the case for the actual program output. I have included the grid in order to demonstrate the ability to more accurately compare intensities of the different transition lines for a species at a specified timestep. Be aware the scale of the y axis is the same for each plot in the same row, however each row has a different y-axis scale than the one above or below it. The y-axis values vary by row because the plot with all the transition lines already displays a line's intensity relative to all the other lines.

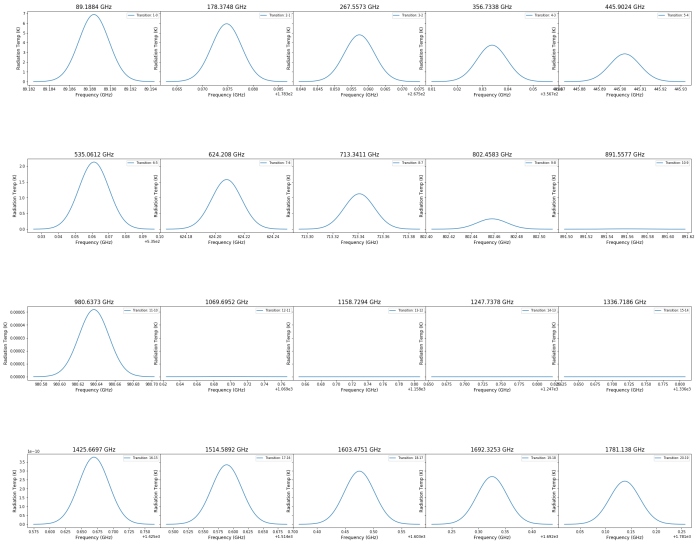

Fig. 6. Grid of individual transition line plots.

5.2. All Times

The second analysis option does the same calculations as the first option, but for every timestep. It does not plot any individual lines, only a plot of all the transition lines for each timestep. By analyzing the plots at different times throughout the structural evolution, the user can better understand when certain physical and chemical conditions or processes are present throughout the model. A sample of the line intensities for HCO^+ throughout the model are shown in Figures 7-15.

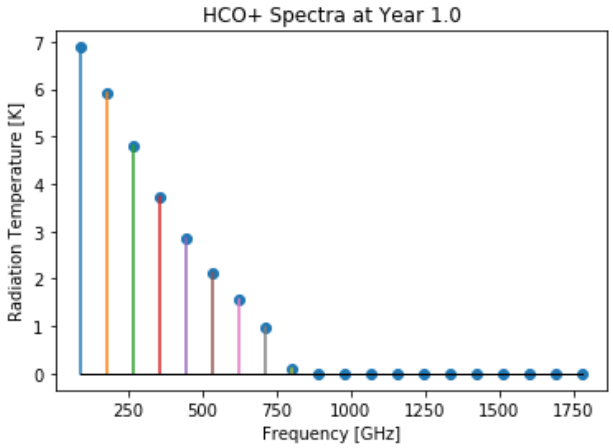

Fig. 7. Initial transition line intensities at year 1.

The initial intensities of the transition lines at year 1 are seen to be quite low, with the majority of them almost relatively non-existent. For the first one million years of the model, there are no chemistry-driving processes. Comparing the initial intensities from Figure 7 and to the intensities just before the shock begins from Figure 8, it is clear there was no noticeable intensity changes in the first 10^6 years of the model. This suggests any future chemistry or intensity changes can be attributed to the shock.

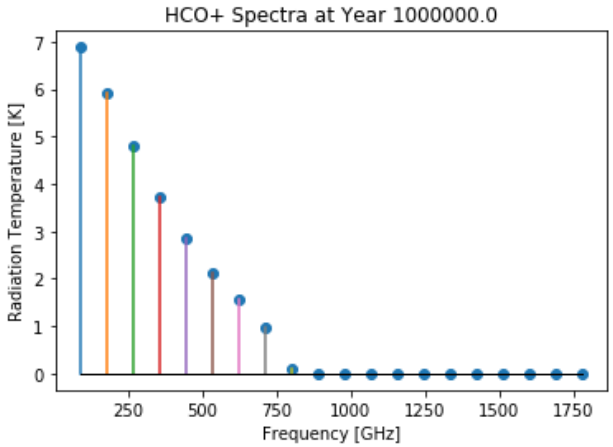

Fig. 8. Transition line intensities when the shock begins.

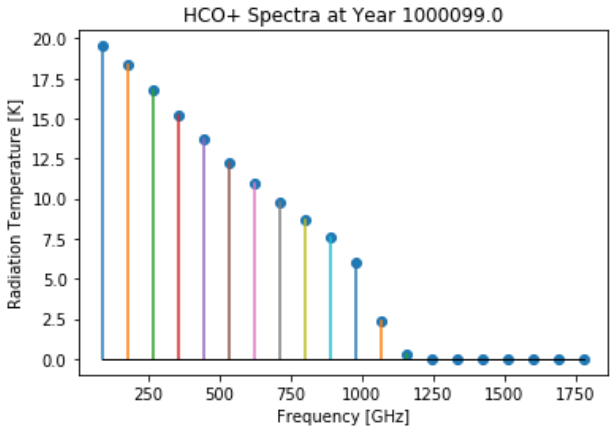

Fig. 9. Approximately 100 years after shock begins.

After a million years of dormancy, it takes only 100 years after the start of the shock for transition line intensities to more than double. Also, the relative line intensities differ with the higher frequency lines reaching values comparable values while lower lines become optically thick.

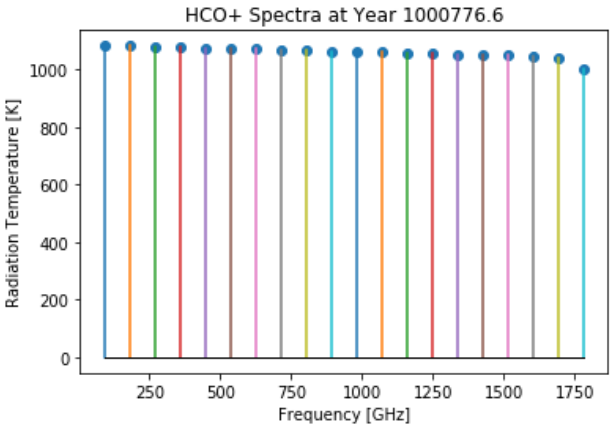

Fig. 10. Approximately 775 years after shock begins. Around this time, the velocity of the shock peaks.

The peak values of the line intensities are plotted in figure 10, with every single transition line above, or nearly at 1,000 K. This value is exponentially higher than the previous maximum values of around 7K before the shock and around 20K just 100 years after the shock starts. This exponential increase has reached its peak in just around 775 years, leading to the conclusion that the shock actually only lasts a short time, yet drastically affects the chemical and physical conditions of molecular species. Looking at the plot in Figure 11, which is around 1500 years after the shock began, one sees that the peak intensities have declined to nearly half the intensity they were just 777 years after the shock start. This demonstrates that the decrease in transition line intensity occurs at a much slower rate than the increase due to the shock.

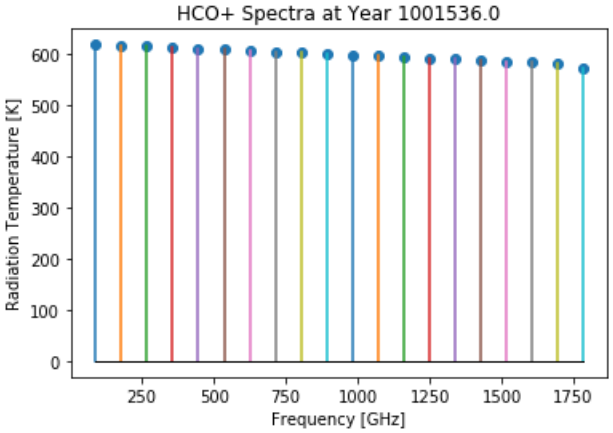

Fig. 11. Approximately 1,500 years after shock begins.

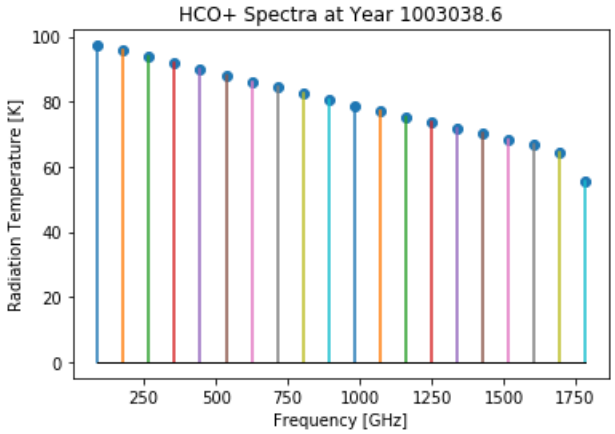

Fig. 12. Approximately 3,000 years after shock begins.

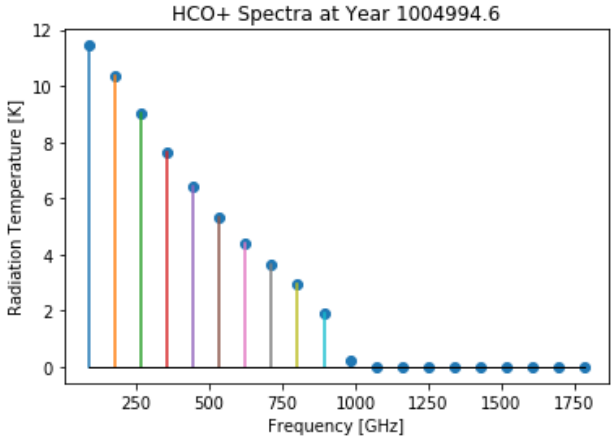

Fig. 13. Approximately 5,000 years after shock begins.

Looking forward to 3,000 and 5,000 years after the shock start, as shown in Figures 12 and 13, respectively, the rate of line intensity decrease still follows the trend of occurring more slowly than the increase. After over 4,000 years since the peak intensities occurred, the line intensities are still greater than what they initially were, at 777 years before reaching the peak intensities.

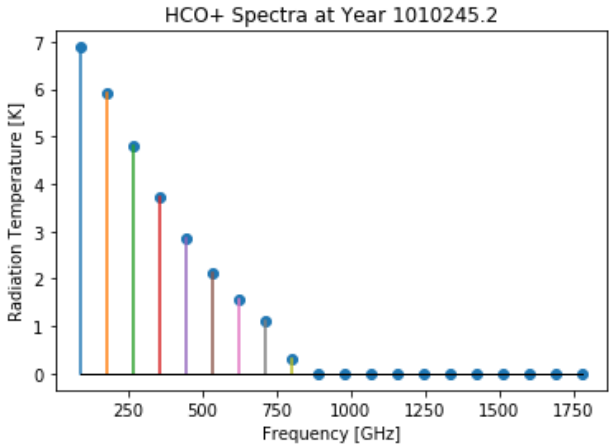

Fig. 14. Approximately 10,000 years after shock begins.

From Figure 14, which is 10,000 years after the initial shock, it seems the line intensities have settled near their initial value. By analyzing the line intensities an entire 1.25e6 years post-shock, it is apparent not much has changed since 10,000 following the start of the shock, suggesting the model had returned to a steady state approximately 10,000 years after the shock began. All of the transition lines in figure 15 are very slightly higher than those in Figures 7 and 8, which is clearly seen by the height of the first transition line with a frequency larger than 750 GHz. This leads to a conclusion that the shock has a lasting effect on the chemical/physical conditions of the cloud, with the HCO^+ molecules having a slightly higher overall energy than before the shock.

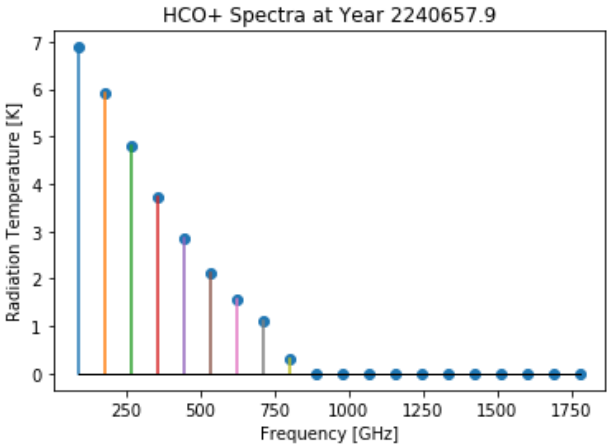

Fig. 15. Approximately 1.25e6 years after shock begins.

6. Conclusion

In this paper, I present a tool to create theoretical molecular spectra to be used in identifying species within molecular clouds, as well as the individual transition lines associated with specific physical and chemical conditions. It can also be used for analyzing the variation of transition line intensities for a species as a model evolves. An example trial run of the tool for HCO^+ in a molecular cloud experiencing a shock is provided. The tool uses the NAUTILUS chemical model to determine abundances of species which are then used in RADEX calculations to obtain the transition line intensities to be graphed on a spectrum. While the specific case study for the tool focused on a shock, this tool can be used for any astrochemical model where it is believed the emission from a species will not be in LTE.

There are many possibilities to improve the applicability and robustness of the tool far beyond its current use. First, the axis displaying the peak intensities of the transition lines throughout the evolution should be made to be constant and logarithmic in order to avoid confusion regarding to the differing y-axis scales when analyzing the results. Another useful addition to the tool would be to allow for the analysis of multiple species at once. The tool would only need to be adjusted by running the analysis multiple times for different species and simply incorporating all of the transition lines into a single spectrum. Also, the number of collision partners could be improved, as the tool only allows for H_2 to be the single collision partner and RADEX allows for up to seven different collision partners.

References

[1] Belloche, A., Garrod, R. T., Muller, H. S. P., Menten, K. M., (ASP), 499, 181

[2] Bergin, E. A., Aikawa, Y., Blake, G. A., & van Dishoeck, E. F. 2007, in Protostars and Planets V, ed. B. Reipurth, D. Jewitt, & K. Keil, 751
[3] Bertin M., et al., 2013, ApJ, 779, 120
[4] Black, J. H. 2000, in Astrochemistry: From Molecular Clouds to Planetary Systems, ed. Y. C. Minh, & E. F. van Dishoeck, (ASP), 197, 81
[5] Castor, J. I. 1970, MNRAS, 149, 111
[6] Chaabouni H., Bergeron H., Baouche S., Dulieu F., Matar E., Congiu E., Gavilan L., Lemaire J. L., 2012, A&A, 538, A128
[7] De Jong, T., Boland, W., & Dalgarno, A. 1980, A&A, 91, 68
[8] Elitzur, M., & Asensio Ramos, A. 2006, MNRAS, 365, 779
[9] Expósito, J. P. F., Agúndez, M., Tercero, B., Pardo, J. R., & Cernicharo, J. 2006, ApJ, 646, L127
[10] Garrod R. T., Wakelam V., Herbst E., 2007, A&A, 467, 1103
[11] Genzel, R. 1991, in Stellar NATO ASIC Proc. 342: The Physics of Star Formation and Early Stellar Evolution, 155
[12] Ghesquiére P., Mineva T., Talbi D., Theulé P., Noble J. A., Chiavassa T., 2015, Physical Chemistry Chemical Physics (Incorporating Faraday Transactions), 17, 114
[13] Hasegawa T. I., Herbst E., Leung C. M., 1992, ApJS, 82, 167
[14] Hasegawa T. I., Herbst E., 1993a, MNRAS, 261, 83
[15] Hasegawa T. I., Herbst E., 1993b, MNRAS, 263, 5
[16] Hindmarsh, A. C. 1983, in IMACS Transactions on Scientific Computation, vol. 1, Scientific Computing, ed. R. S. Stepleman, 55
[17] Kennicutt, Jr., R. C. 1998, ARA&A, 36, 189
[18] Matar E., Bergeron H., Dulieu F., Chaabouni H., Accolla M., Lemaire J. L., 2010, J. Chem. Phys., 133, 104507
[19] Mihalas, D. 1978, Stellar atmospheres, 2nd edition (San Francisco: W. H. Freeman and Co.)
[20] Roberts J. F., Rawlings J. M. C., Viti S., Williams D. A., 2007, MNRAS, 382, 733
[21] Ruaud M., Wakelam V., & Hersant F., 2016, MNRAS, 459, 3756
[22] Schoier, F. L., van der Tak, F. F. S., van Dishoeck, & E. F., Black, J. H. 2005, A&A, 432, 369
[23] Semenov, D., Pavlyuchenkov, Y., Henning, T., Wolf, S., & Launhardt, R. 2008, ApJ, 673, L195
[24] Sobolev, V. 1960, Moving envelopes of stars (Harvard University Press)
[25] van der Tak, F. F. S., Black, J. H., Schöier, F. L., Jansen, D. J., & van Dishoeck, E. F. 2007, A&A, 468, 627
[26] van Dishoeck, E. F., & Blake, G. A. 1998, ARA&A, 36, 317
[27] van Dishoeck, E.F., & Hogerheijde, M. R. 1999, in NATO ASIC Proc. 540:The Origin of Stars and Planetary Systems, 97
[28] Wakelam V. et al., 2012, ApJS, 199, 21

```python
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
"""
Created on Thu Jan 25 23:46:23 2018

@author: josephepstein
"""

import numpy as np
import matplotlib.pyplot as plt
import os
import os.path


# Global Variables

rad_input = ''
output_file = ''
line_1_arr = ''
line_2_arr = ''
e_up_arr = '' # K
freq_arr = '' # GHz
wavel_arr = '' # um
t_ex_arr = '' # K
tau_arr = ''
t_r_arr = '' # K
pop_up_arr = ''
pop_low_arr = ''
flux1_arr = '' # K*km/s
flux2_arr = '' # erg/cm2/2
output_arr = []
line_width = '' # km/s
timestamp = '' # year
user_setting = ''
structure_arr = []


# Function 1: Taking Input Data from User
def make_rad_input(): # set default parameter, also use optional
parameters

    global rad_input
    global output_file
    global line_width
    global timestamp
    global user_setting
```

```
    global structure_arr

    # 1) Prompt user to upload an input Radex data file

    # Check to see if they already have a Radex input file
    while True:
        file_check = raw_input('Do you have a Radex input file?
(y/n): ')
        if file_check == 'y':
            rad_input = raw_input('Enter the name of your data
file: ')
            return
        if file_check == 'n':
            print '\n'
            print 'Please insert data manually.'
            break

    # 2) Prompt user to input all info needed to get Radex inputs

    rad_input=raw_input('What would you like to name your input
file(s)?: ')

    if rad_input.lower().endswith('.inp'):
        rad_input = rad_input
    else:
        rad_input = rad_input+'.inp'

    output_file = raw_input('Output file name?: ')
    if output_file == '':
        output_file = 'radex.out'

    moldat = raw_input('Molecular data file?: ')
    if moldat.lower().endswith('.dat'):
       moldat = moldat
    else:
        moldat = moldat+'.dat'

    freq_range_high = raw_input('Frequency range max? (GHz): ')
    if freq_range_high == '':
        freq_range_high = 0

    freq_range_low = raw_input('Frequency range min? (GHz): ')
    if freq_range_low == '':
        freq_range_low = 0

    partners = 1 # 1 - H2
```

```
    partner1 = 'H2'

    bkg_temp = raw_input('Background temperature? (K): ')
    if bkg_temp == '':
        bkg_temp= 2.73

    line_width = raw_input('Line width?(km/s-1): ')
    if line_width == '':
        line_width = 5.0
    else:
        line_width = float(line_width)

    abundance_file = raw_input('Abundance file?: ')
    if abundance_file.lower().endswith('.ab'):
       abundance_file = abundance_file
    else:
        abundance_file = abundance_file+'.ab'

    structure = raw_input('Structural Evolution file?: ')
    if structure.lower().endswith('.dat'):
       structure = structure
    else:
        structure = structure+'.dat'

    if structure != '':
        structure_arr = np.genfromtxt(structure, skip_header=2)
        abundance_arr = np.genfromtxt(abundance_file,
skip_header=1)
        user_setting = raw_input('Enter 1 if you want a spectrum
at 1'\
                         'specific time.'+'\n'+'Enter 2 if you
want'\
                         'spectra for all times.'+'\n')

  # This allows the user to pick a specific time to create a
spectrum for
        if user_setting == str(1):
            while True:
                try:
                    timestamp=float(raw_input('For what year would
you'\
                                              'like the spectrum?:
'))
                    break
                except:
                    print "\n"
```

```
                print "Please insert a number."
            # Get abundance for year closest to input year

            #First get column position of the time
            lowest = 10000000
            for i in range(len(structure_arr['time'])):
                difference = abs(timestamp-structure_arr['time']
[i])
                if difference < lowest:
                    lowest = difference
                    position = i

            # get abundance from .ab file from given time
            log_abundance = abundance_arr[position][1]
            abundance = 10**log_abundance
            N_H2 = (1e21) # H column density cm-2
            col_dens = (abundance*N_H2) # abundance from model *
N_H2

            # Get kinetic temp from structure_evolution file too:
log(Tg)
            log_kin_temp = structure_arr[position][3]
            kin_temp = 10**log_kin_temp

            log_density = structure_arr[position][2]
            density = 10**log_density # cm-3
            partner1_dens = density

            # Write the Radex input file
            subdr_name = rad_input[:-4]
            os.system('mkdir '+rad_input[:-4])
            file = open(subdr_name+'/'+rad_input[:-4]+'.inp', 'w')
            file.write(moldat+'\n') # molecular data file
            file.write(output_file+'.rdx'+'\n') # output file name
            file.write(str(freq_range_low)+'
'+str(freq_range_high)+'\n')
            file.write(str(kin_temp)+'\n') # kinetic temp (K)
            file.write(str(partners)+'\n') # number of collision
partners
            file.write(partner1+'\n') # first collision partner
            file.write(str(partner1_dens)+'\n') #dens of coll.
partner (cm-3)
            file.write(str(bkg_temp)+'\n') # Background radiation
temp (K)
            file.write(str(col_dens)+'\n') # molecular column
density (cm-2)
```

```
            file.write(str(line_width)+'\n') # line width (km/s)
            file.write('0\n') # run another calc (1 yes, 0 no)
            file.close()

    # This makes a subdirectory containing radex input files for
    # every time in the structure evolution file
        if user_setting == str(2):
# make subdirectory to save all the files in, and go into the
subdirectory
            subdr_name = rad_input[:-4]
            os.system('mkdir '+rad_input[:-4])
            file = open('test_file.txt', 'w')
            file.write('This is a test')
            file.close()
            # make list of input and output file names
            rad_input_list = []
            output_file_list = []
            for i in range(len(structure_arr['time'])):
                rad_input_list.append(rad_input+str(i))
                output_file_list.append(output_file+str(i))
# for each time/position in the structure file, calculate the
necessary values
            for i in range(len(structure_arr['time'])):
                # get abundance from .ab file from given time
                log_abundance = abundance_arr[i][1]
                abundance = 10**log_abundance
                N_H2 = (1e21) # H column density cm-2
                col_dens = (abundance*N_H2) # abundance from model
* N_H2

                #Get kinetic temp from structure_evolution file
too: log(Tg)
                log_kin_temp = structure_arr[i][3]
                kin_temp = 10**log_kin_temp


                log_density = structure_arr[i][2]
                density = 10**log_density # cm-3
                partner1_dens = density

                # Write the Radex input file
                file = open(subdr_name+'/'+rad_input[:-4]+str(i)
+'.inp', 'w')
                file.write(moldat+'\n') # molecular data file
                file.write(output_file+str(i)+'.rdx'+'\n') #
output file name
```

```
                file.write(str(freq_range_low)+'
'+str(freq_range_high)+'\n')
                file.write(str(kin_temp)+'\n') # kinetic temp (K)
                file.write(str(partners)+'\n') # number of
collision partners
                file.write(partner1+'\n') # first collision
partner
                file.write(str(partner1_dens)+'\n') #dens coll
partner (cm-3)
                file.write(str(bkg_temp)+'\n') #Background
radiation temp (K)
                file.write(str(col_dens)+'\n') # mol column
density (cm-2)
                file.write(str(line_width)+'\n') # line width (km/
s)
                file.write('0\n') # run another calc (1 yes, 0 no)
                file.close()
    return



#Function 2: Running Input(s) File Through Radex
def run_radex():
    if user_setting == str(1):
        command_name = 'radex < '+rad_input[:-4]
+'/'+rad_input[:-4]+'.inp'
        os.system(command_name)

    if user_setting == str(2):
        for i in range(len(structure_arr['time'])):
            radex_command = 'radex < '+rad_input[:-4]
+'/'+rad_input[:-4] \
                                                              +str(i
)+'.inp'
            os.system(radex_command)
            move_output='mv '+output_file+str(i)+'.rdx
'+rad_input[:-4]+'/'
            os.system(move_output)
    return



# Function 3: Getting all of the information from the radex output
file.
def read_radex_1():
    #take output radex file made from previous command
```

```
    global output_file
    global output_arr
    output_arr = np.genfromtxt(output_file+'.rdx', skip_header=11)

    global line_1_arr
    global line_2_arr
    global e_up_arr # K
    global freq_arr  # GHz
    global wavel_arr # um
    global t_ex_arr  # K
    global tau_arr
    global t_r_arr # K
    global pop_up_arr
    global pop_low_arr
    global flux1_arr # K*km/s
    global flux2_arr # erg/cm2/2

    line_1_arr = output_arr[:,0]
    line_2_arr = output_arr[:,1]
    e_up_arr = output_arr[:,3] # K
    freq_arr = output_arr[:,4] # GHz
    wavel_arr = output_arr[:,5] # um
    t_ex_arr = output_arr[:,6] # K
    tau_arr = output_arr[:,7]
    t_r_arr = output_arr[:,8] # K
    pop_up_arr = output_arr[:,9]
    pop_low_arr = output_arr[:,10]
    flux1_arr = output_arr[:,11] # K*km/s
    flux2_arr = output_arr[:,12] # erg/cm2/2

    return


def make_spectra_1():

    a_list = []
    b_list = []
    c_list = []
    j_list = []
    k_list = []

    markerline, stemlines, baseline = plt.stem(freq_arr[:],
t_r_arr[:],'-')
    plt.setp(baseline, color='black', linewidth=1)
    plt.title('HCO+ Spectra at Year '+str(timestamp))
    plt.xlabel('Frequency [GHz]')
```

```
    plt.ylabel('Radiation Temperature [K]')
    plt.show()


 # Makes array of plots each showing an individual line
    for i in range(len(output_arr)):
        a = output_arr[i][8] # t_r
        a_list.append(a)

        b = output_arr[i][4] # line_center
        b_list.append(b)

        c = (line_width/299792)*b
        c_list.append(c)

        j = np.linspace(b - 4*c,b + 4*c,1000)
        k = a*np.exp(-((j-b)**2)/(2*c**2))
        plt.plot(j,k)
        plt.xlabel('Frequency [GHz]')
        plt.ylabel('Radiation Temperature [K]')
        plt.title('HCO+ Transition: '+str(int(output_arr[i][0]))+'
-- ' \
            +str(int(output_arr[i][2]))+' ('+str(output_arr[i][4])
+' GHz)')
        plt.grid()
        plt.show()


        j = np.linspace(b_list[i] - 4*c_list[i],b_list[i] +
4*c_list[i],1000)
        j_list.append(j)

        k = a_list[i]*np.exp(-((j-b_list[i])**2)/(2*c_list[i]**2))
# gaussian
        k_list.append(k)

    #get size of side for 2D array of plots
    subplots_size = int(np.sqrt(len(output_arr))+1)
    f, axarr = plt.subplots(subplots_size,subplots_size,
sharey='row')
    f.set_figheight(5)
    f.set_figwidth(5)
    plt.subplots_adjust(top=subplots_size, bottom=0, left=0, \
        right=subplots_size, hspace=subplots_size/4,
wspace=subplots_size/8)
```

```
    m = 0 # row
    n = 0 # column
    for l in range(len(output_arr)):

        j = np.linspace(b_list[l] - 4*c_list[l],b_list[l] +
4*c_list[l],1000)
        k = a_list[l]*np.exp(-((j-b_list[l])**2)/(2*c_list[l]**2))
# gaussian
        if n < subplots_size:
            axarr[m,n].plot(j,k)
            axarr[m,n].set_title(str(output_arr[l][4])+' GHz',
fontsize=15)
            axarr[m,n].set_ylabel('Radiation Temp (K)',
fontsize=12)
            axarr[m,n].set_xlabel('Frequency (GHz)', fontsize=12)
            axarr[m,n].legend(['Transition:
'+str(int(output_arr[l][0]))+'-'\
                 +str(int(output_arr[l][2]))],loc='upper right',
fontsize=8)
            n += 1
            continue
        elif n == subplots_size:
            n = 0
            m += 1
            axarr[m,n].plot(j,k)
            axarr[m,n].set_title(str(output_arr[l][4])+' GHz',
fontsize=15)
            axarr[m,n].set_ylabel('Radiation Temp (K)',
fontsize=12)
            axarr[m,n].set_xlabel('Frequency (GHz)', fontsize=12)
            axarr[m,n].legend(['Transition:
'+str(int(output_arr[l][0]))+'-'\
                 +str(int(output_arr[l][2]))],loc='upper
right',fontsize=8)
            n += 1
            continue

    plt.show()

    return
```

```
def read_make_2():
    for i in range(len(structure_arr['time'])):
        #take output radex file made from previous command
        global output_file
        global output_arr
        output_arr = np.genfromtxt(rad_input[:-4]+'/'+output_file
+str(i)+\
                                   '.rdx', skip_header=11)

        global line_1_arr
        global line_2_arr
        global e_up_arr # K
        global freq_arr  # GHz
        global wavel_arr # um
        global t_ex_arr  # K
        global tau_arr
        global t_r_arr # K
        global pop_up_arr
        global pop_low_arr
        global flux1_arr # K*km/s
        global flux2_arr # erg/cm2/2

        line_1_arr = output_arr[:,0]
        line_2_arr = output_arr[:,1]
        e_up_arr = output_arr[:,3] # K
        freq_arr = output_arr[:,4] # GHz
        wavel_arr = output_arr[:,5] # um
        t_ex_arr = output_arr[:,6] # K
        tau_arr = output_arr[:,7]
        t_r_arr = output_arr[:,8] # K
        pop_up_arr = output_arr[:,9]
        pop_low_arr = output_arr[:,10]
        flux1_arr = output_arr[:,11] # K*km/s
        flux2_arr = output_arr[:,12] # erg/cm2/2

        fig = plt.figure()
        markerline, stemlines,
baseline=plt.stem(freq_arr[:],t_r_arr[:],'-')
        plt.setp(baseline, color='black', linewidth=1)
        plt.title('HCO+ Spectra at Year
'+str(structure_arr['time'][i]))
        plt.xlabel('Frequency [GHz]')
        plt.ylabel('Radiation Temperature [K]')
        plt.savefig(rad_input[:-4]+'/'+rad_input[:-4]+str(i)
+'.pdf')
        plt.show()
```

```
        plt.close(fig)

    return


make_rad_input()
run_radex()

if user_setting == str(1):
    read_radex_1()
    make_spectra_1()

if user_setting == str(2):
    read_make_2()
```