The evolving velocity field around protostars

Brinch, C.

Chapter 7

Advances in line radiation transfer modeling

7.1 Introduction

When interpreting astronomical emission line data originating from molecules that are not in thermodynamical equilibrium, it is essential for understanding the source to be able to successfully model the signal. In order to do so, the equations of radiation transport and the molecular excitation need to be solved. There are no general solutions to this problem, and therefore computational expensive numerical methods must be applied. Throughout this thesis, our main modeling tool has been the radiation transfer code *RATRAN* (Hogerheijde & van der Tak 2000), which we have used to simulate the radiation field in protostellar envelopes and disks. As a two-dimensional axis-symmetric code, *RATRAN* was originally developed to model envelopes with rotation and gaseous protoplanetary disks. At the time when *RATRAN* was written, the computers could converge on a solution to a relatively coarse model in hours. With present day computers, the same model converges in a matter of minutes. Hence, the level of detail in these models has increased accordingly. During the same period, observations of far higher resolution have become available, in particular with the first light of the Submillimeter Array in late 2003. In order to follow this advancement, our models are now reaching a level of complexity where CPU time is not the main bottle neck, but rather the basic design of the code itself. In this thesis we have encountered these limitation in Chapters 4 and 6.

There are three main short-comings of the current code: (i) It assumes cylindrical symmetry, effectively confining all structure to two dimensions. This became a problem in Chapter 4 where we found the need for two different rotation axes. (ii) Convergence becomes rapidly slower as the grid resolution is increased which severely limits the effective range of scales of the models. In Chapter 6 we had the need for a model with a radius of 12000 AU, resolved down to scales of

10 AU. This is a huge dynamic range of scales, which in Chapter 6 resulted in a very large number of grid cells and consequently a very slow convergence. (iii) Even with a low number of grid cells, molecules with a large number of transitions converge very slowly. In this thesis we have not dealt with computationally complex molecules, and therefore this has not been a serious issue here. The points (i) and (ii) are of course related in the sense that gridding in three dimensions requires n times more cells (where n is the average number of resolution elements per spatial direction) than the corresponding two-dimensional grid and since it follows trivially that more cells require more CPU time, point (ii) makes three-dimensional grids unfeasible. However, the loss of performance with increasing cell number is more severe than what this effect suggest for codes which use Accelerated Lambda Iteration (such as *RATRAN*). Above a certain number of grid cells, the acceleration is lost and thus convergence takes many more iterations than otherwise.

The third point becomes a problem because for molecules with a large number of transitions, e.g. H_2O, the Monte Carlo approach becomes ineffective even for a small number of cells and therefore high resolution grids are not feasible for these species. However, molecules with many possible transitions are typical very far from LTE and it is desirable not to have gradients in excitation conditions that are too large over the cells which calls for a finer grid (depending on the model). If we furthermore have depletion in certain regions, like the abundance distribution that we showed in Chapter 5, very strong density gradients are present in the model, which calls for higher grid resolution in the boundary regions. Finer grids means more cells and therefore the level population of these molecules are extremely hard to calculate using codes like *RATRAN*.

In this chapter we present a new code which improves on all of these three points. The code is based on *RATRAN*, but a whole new photon propagation algorithm and gridding method is implemented making the code significantly different from *RATRAN*. The code is highly optimized for speed and the user interface is completely reworked for a very flexible model setup.

7.2 Solution method

7.2.1 The molecular level population equilibrium

The problem of radiation transport is described by the equation,

$$\frac{dI_\nu}{ds} = -\alpha_\nu I_\nu + j_\nu. \tag{7.1}$$

For line radiation transfer, the emission and absorption coefficients j_ν and α_ν are determined from the Einstein coefficients, for transition between any two adjacent energy levels E_2 and E_1, through the relations,

$$j_{21} = \frac{\hbar\nu}{2} n_2 A_{21} \phi(\nu), \tag{7.2}$$

$$\alpha_{21} = \frac{\hbar\nu}{2} \left(n_1 B_{12} - n_2 B_{21}\right) \phi(\nu), \tag{7.3}$$

where $h\nu = E_2 - E_1$. The frequency dependent velocity profile function ϕ is given by a Gaussian with a width determined by the local microturbulence.

Solving Eq. 7.1 for the intensity I_ν gives the local mean radiation field through an integral over all solid angles,

$$J_\nu = \frac{1}{4\pi} \int I_\nu d\Omega, \tag{7.4}$$

whereas the population of the l'th level n_l is given by,

$$n_l = \frac{\Sigma_{k>l} n_k A_{kl} + \Sigma_{k\neq l} n_k (B_{kl} J_\nu + C_{kl})}{\Sigma_{k<l} A_{lk} + \Sigma_{k\neq l} (B_{lk} J_\nu + C_{lk})}, \tag{7.5}$$

in which we have included the molecule dependent collision rates C_{lk}. The level populations depends on J_ν which again depends on the level populations through j_ν and κ_ν, and thus a solution is obtained by iteratively solving Eq. 7.4 and 7.5 until an equilibrium between the radiation field and the level populations is reached.

7.2.2 Photon transport

In *RATRAN* and similar codes (e.g., *URANIA*, Pavlyuchenkov & Shustov 2004), the source model is laid out onto a grid of rectangular cells. Model properties such as density, temperature, molecular abundance, as well as the radiation field J_ν and the level populations n_i are taken to be constant over each cell. Photon packages are traced backwards in random directions from random points of absorption within a cell and Eq. 7.1 is integrated along the paths. Summing over all photons, the mean radiation field J_ν is obtained for the cell.

In our implementation, the source model is no longer mapped onto a regular grid of cells. Instead we use a set of points which represents the local environment (density, temperature, populations, etc.). The points are distributed in three-dimensional space rather than in a two-dimensional slice, and thus we are able to use three-dimensional source models whereas in *RATRAN*, even though

photons propagate in three-dimensional space, the source model needs to be rotational symmetric around the second axis and mirror symmetric around the first axis. Our points are placed at random throughout the entire computational domain, however with a probability that is weighed by a source model dependent function. In particular we adopt the approach of Ritzerveld & Icke (2006) which is briefly described in the following.

The Poisson point process gives the probability of finding k points within a subset V of the domain S of our source model,

$$P(k) = \frac{(\lambda|V|)^k e^{-\lambda|V|}}{k!}, \quad k = 0, 1, 2, \ldots \tag{7.6}$$

λ is called the point intensity and can in general be a function defined on S. It is desirable to obtain a point distribution for which the gradient of the level populations between any two neighboring points becomes as small as possible. This is assured if the mean distance between neighboring points is inversely proportional to the local opacity which is true if $\lambda = c\, n^d$, where n is the density and d is the dimensionality of the source model, in our case 3. The proportionality factor c is a global constant, determined by the total number of points used to describe the source model.

The source model is randomly Nyquist sampled, using Eq. 7.6 to determine the local point density, in order to make sure that all relevant scales of the source model is appropriately described by the point distribution. Besides using this resulting set of points as basis for our computational grid, the obvious advantage of not having to construct a grid by hand, is that points can be added (or removed) dynamically by changing c and resample the source model. Figure 7.1 shows that the neighbor distance is proportional to the density over four orders of magnitude. The (arbitrary) offset between the density graph and the histogram scales with the total number of grid points.

The points are connected by Delaunay triangulation, using the public available *QHull* library (Barber et al. 1996). The transport itself goes along Delaunay lines only from one point to another, which makes integration of Eq. 7.1 particularly simple and very fast. In the three-dimensional Delaunay triangulation, the expectation value for the number of lines attached to a grid point is approximately 16 (Ritzerveld & Icke 2006) and the spatial sampling of J_ν is thus limited to this number of directions.

Figure 7.2 shows an example of a Delaunay grid based on a simple disk model. The grid shown here is the two-dimensional equivalent to the three-dimensional Delaunay grid used in our code. The point distribution is seen to follow the density contours closely and above the disk where density approaches zero, there are no

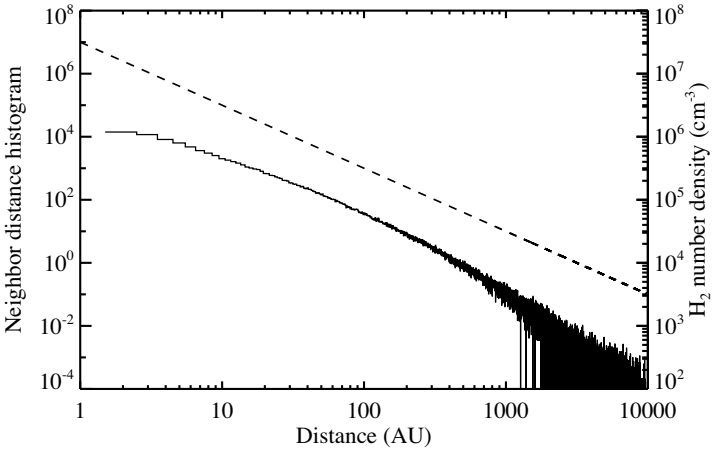

Figure 7.1: A histogram showing the distribution of neighbor distances from 1 AU to 10000 AU. This distribution scales very well with the number density, shown with a dashed line. To make this plot, the density model was sampled by 20000 points.

points. However, we still need to trace a number of photons along each Delaunay line, since we cannot conserve momentum stringently on this grid. In principle, a photon passing a grid point from a certain direction should continue to travel in the exact same direction. This is in general not possible due to the random orientation of the Delaunay lines, so instead we choose one of the two Delaunay lines that make the smallest angle with the incoming line. The outgoing line is picked at random, but weighed by the ratio of the two angles. The same procedure is used at all subsequent grid points (using the original momentum vector to determine the outgoing direction) until the photons escape the cloud. By sending a number of photons along each initial Delaunay line, we thus probe, not a single line of sight, but rather a cone, while still conserving momentum on average. The transport method is illustrated in Fig. 7.3 where the thickness of the connecting Delaunay lines are proportional to the probability of a photon traveling along that line if emitted in the direction indicated by the dashed line.

During gridding of our source model, we also distribute a number of points randomly on the surface of a sphere surrounding our model. These points are also Delaunay triangulated and connected to the grid points, but they do not represent

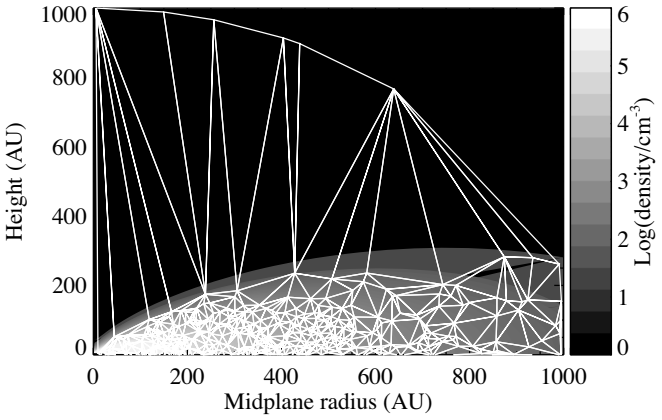

Figure 7.2: An example of the gridding used in our code. The underlying source model is a standard flat disk model and the grid consists of 200 points. For visualization purposes we show here a 2D version of the grid.

anything except the surface of our computational domain. Whenever a photon reaches one of these sink points, it is considered to have escaped the model and we add the CMB contribution before going on to trace the next photon.

Because of the relative low number of photons needed to probe the spatial directions, we can allow ourself to increase the number of photons used to sample the different frequencies, while we still maintain a low (initial) number of photons per grid point. When J_ν has been obtained for a grid point, the level populations are solved for, using the same Λ-iteration method as in *RATRAN* (see Hogerheijde & van der Tak 2000, for details on (Accelerated) Lambda Iteration). Three subsequent iterations of finding J_ν and solving for n_l are done, after which the variance of the three solutions is calculated. If the change in level populations between these iterations is smaller than some preset limit, the grid point is said to be converged. By looping through all grid points until all have converged, and increasing the number of photons in the unconverged points, we end up with a global solution with all grid points converged.

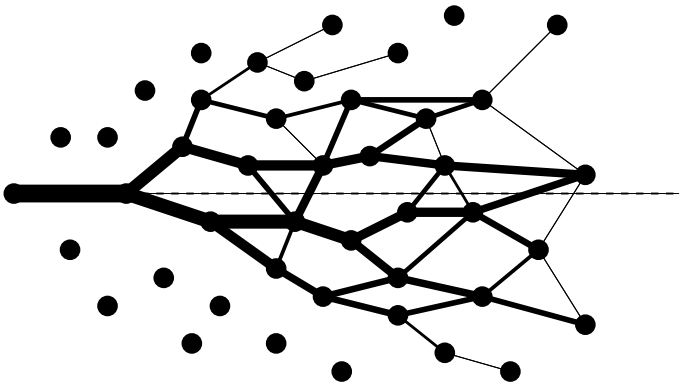

Figure 7.3: Illustration of the photon propagation. When a photon reaches a neighboring grid point, it chooses its new direction with a propability that is proportional to the angle between the new direction and the original direction (dashed line). In this figure, the lines represent possible photon paths and their thickness is proportional to the probability of a photon traveling along that particular Delaunay line. In this figure, the lines disappear below a certain probability threshold. The photons travel from left to right.

7.2.3 Raytracing

After population equilibrium has been reached, we raytrace lines of sight through the model in order to obtain an image cube of the radiation that escapes from the surface. For the raytracing we let the photons move in straight lines, rather than jumping from grid point to grid point. We therefore make use of the Voronoi diagram which corresponds to our Delaunay tessellation. The cells of the Voronoi diagram are connected to the vertex points of the Delaunay tessellation by definition: Given the i'th grid point P_i, the i'th Voronoi cell consists of all points $q \in (x, y, z)$ for which $|P_i - q| < |P_{j\neq i} - q|$ is true. The entire volume of the Voronoi cell is represented by the populations of the corresponding Delaunay point and so integration of Eq. 7.1 becomes a matter of stepping through the source model and figuring out in which Voronoi cell the photon is, which, from an algorithmic point of view, comes down to a simple sorting, not of thousands of cells, but only of, on average, 16 neighboring cells.

This is a very fast process compared to moving through a rectangular grid, but not as fast as moving along the Delaunay lines, which is why this transport method is not used when determining the level populations.

7.2.4 Interface

An important improvement to the code is a redesign of the user interface, for a flexible and very adaptable model input method. In a grid based code, the model needs to be mapped onto the regular grid, and this requires a lot of interpolation, with the added risk of introducing error by, for instance, not strictly conserving mass. Furthermore, since the model may originate from many different sources, e.g., analytical functions for the velocity, tabulated data for the temperature, output from another code for the density etc., several custom schemes need to be employed in order to map a single model onto the grid.

In our implementation, the model is supplied through a number of subroutines. These subroutines are compiled and linked each time the code is invoked. The user has complete freedom to formulate the model within these subroutines as long as the physical property, e.g., the density, as a function of x, y, and z is returned. The simplest possibility is to give a functional expression, but more elaborate solutions, such as interpolating in a look-up table, are also possible. In principle, a function call to an external program can be given, so that, for example, the temperature could be calculated self-consistently on demand by a separate program. This last option could become interesting for future model work in the sense that it allows the user to link our code directly to another code (typically a hydrodynamical or chemistry code) and run the two simultaneously, sharing their output.

7.3 Performance

One of the main purposes with this code is to improve performance with respect to previously available codes. The aim is to make a code which is faster than *RATRAN*, while still providing consistent results, and which is fully three-dimensional.

Figure 7.4 shows the performance of our code as well as the one-dimensional and two-dimensional versions of *RATRAN*. The model which is used in this test is a very simple, spherical power-law envelope in free fall

$$n(r) = 1 \times 10^6 \text{ cm}^{-3} (r/1000 \text{ AU})^{-1.5}, \tag{7.7}$$

$$t(r) = 20 \text{ K } (r/1000 \text{ AU})^{-0.4}, \tag{7.8}$$

$$v(r) = -\sqrt{\frac{2G\ 1.0\ \text{M}_\odot}{r}}. \tag{7.9}$$

The test in Fig. 7.4 is for CO which is an computationally "easy" molecule (i.e., it

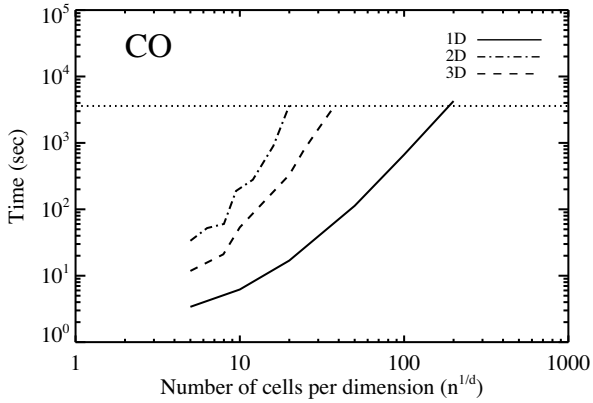

Figure 7.4: The performance of the code described in this paper along with that of 1D and 2D *RATRAN*, for a simple model with increasing grid resolution. The horizontal dotted line marks 1 hour of running time.

has a small number of transitions per level). The figure shows the number of grid cells used to describe the source model per dimension (grid points in the case of our code) versus the time it takes to converge on a solution. All tests shown here have been made on the same standard desktop computer. Our code is seen to be faster than two-dimensional *RATRAN* for the entire range of grid resolutions that was tested. We increased the number of grid cells/points until the model would run for more than 1 hour, marked by the dotted line at 3600 seconds. We cannot compete with the one-dimensional code which is seen to be much faster than both the two-dimensional code and our code.

More interesting is the case of H_2O, which is a notorious "difficult" molecule to calculate because of the many possible and closely spaced transitions between its levels. The result of this benchmark is shown in Fig. 7.5. In this figure it is seen that the convergence takes much longer than for CO for all three codes. Especially the two-dimensional version of *RATRAN* shows a much steeper relation between grid resolution and convergence time. The three-dimensional code also shows slow-down with respect to the CO calculations, but the slope of the relation is much more shallow than for the two-dimensional code, which means that water models of a reasonable resolution can be calculated in minutes with our code, as opposed to hours with the two-dimensional code.

It should be noted though that if the source model is intrinsically one- dimen-

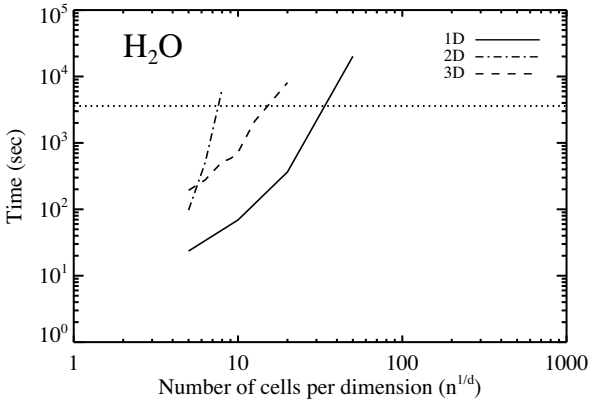

Figure 7.5: Similar to Fig. 7.4 but for the water molecule.

sional, it should certainly be calculated with the one-dimensional code rather than by our three-dimensional code. However, if any asymmetries exists in the source model, either in the density distribution or the velocity field, it will be faster to use our code.

7.4 Examples

Here we present three examples that illustrate the capabilities of our code. These examples are not necessarily physical, but are meant as demonstrations of the code. In the following we assume a source distance of 140 pc when we talk about the resolution in the images.

7.4.1 Comparison to *RATRAN*

In this first example we show that our code produces a result which is similar to the solution of *RATRAN* for a model which can also be calculated by a two-dimensional code. We use the same simple setup as in our benchmark tests (Sect. 7.3) to calculate a CO image. In Fig. 7.6 a single spectrum is shown, taken toward the center of the model at three different resolutions as calculated by *RATRAN* and by our code. The resolutions used are 10″, 1″, and 0.1″. In all three cases the spectra are seen to match well, except for the fact that in the unconvolved (pixel resolution) case, the spectra begin to show pixelization. The discrepancy

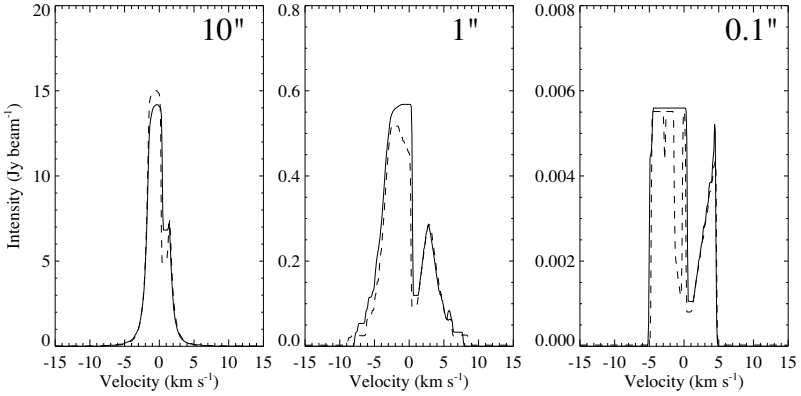

Figure 7.6: CO J=1–0 spectra of a simple infalling envelope model. In each panel the corresponding *RATRAN* spectrum is plotted on top of the spectrum made with the code presented in this chapter. The result from our code is shown in dashed linestyle.

which is seen in a few of the channels is mainly due to the difficulties in conserving mass properly when mapping a spherical profile onto a regular grid.

A more robust way to compare the solutions of the two codes is to compare the actual level populations. This is best done by plotting the radial excitation temperature, defined as

$$\frac{n_u}{n_l} = \frac{g_u}{g_l} \exp\left(-\frac{\Delta E}{k_B T_{ex}}\right) \tag{7.10}$$

where n_u and n_l are the upper and lower level populations of two adjacent levels, g_u and g_l the corresponding statistical weights and ΔE the energy difference between the levels. The excitation temperature as a function of radius of the first four transitions of CO is plotted in Fig. 7.7. In this figure we used the one-dimensional version of *RATRAN* for increased resolution, but still the thin curve is seen to be more step-like than the thick line, calculated with our code. Apart from that, the two curves are perfectly similar.

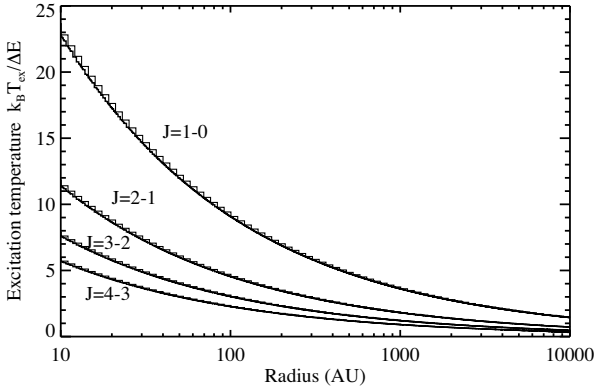

Figure 7.7: The excitation temperature of the first 4 transitions of CO (from the top curve and down). The thick line is the excitation temperature calculated by the code presented here whereas the thin, slightly boxy curve is calculated by the one-dimensional version of *RATRAN*. Apart from the lower resolution of *RATRAN*, the curves are indistinguishable.

7.4.2 Misaligned disk/envelope system

The second example is a model which is intrinsically three dimensional. The model used here is similar to the model of L1489 IRS presented in Chapter 4, where a disk is rotating around a different axis than the rotation axis of the envelope. This example demonstrates that such geometries can be distinguished from axi-symmetric models.

The model has two components, a disk and a shell structured envelope. The envelope is rotating and infalling while the disk is in pure Keplerian motion. The rotation axis of the two components is misaligned by 45°. The envelope is described by a power-law and is spherically symmetric. The H_2 number density is parameterized by

$$n(r) = 10^6 \text{ cm}^{-3} \left(\frac{r}{1000 \text{ AU}} \right)^{-2}, \tag{7.11}$$

with an inner radius of 400 AU and an outer radius of 3000 AU, resulting in a mass of 0.25 $M_\odot$. The temperature is defined similarly,

$$T(r) = 20 \text{ K} \left(\frac{r}{1000 \text{ AU}} \right)^{-0.4}. \tag{7.12}$$

This temperature description also applies within the 400 AU radius. Within the radius of 400 AU the density profile is described by a parameterized disk,

$$n(r,\theta) = 2.5 \times 10^{24}\ \mathrm{cm}^{-2} \left(\frac{r}{1\ \mathrm{AU}}\right)^{-1} \frac{1}{\sqrt{2\pi}H_0} e^{-\frac{1}{2}\left(\frac{r\ \cos(\theta)}{H_0}\right)^2}, \tag{7.13}$$

with a scale height $H_0 = 0.25r$ resulting in a disk mass of 0.01 $M_\odot$. The velocity field is parameterized as

$$v_\phi \cos(\alpha) = -\frac{v_r}{\sqrt{2}} \sin(\alpha) = \sqrt{\frac{GM}{r}}, \tag{7.14}$$

where M is the central mass for which we use 1 $M_\odot$ and α is set to 0.2 in the envelope and 0.0 in the disk. Tilting the disk with respect to the envelope will result in a discontinuity in the density and velocity fields, but we will assume that the effects of this is negligible. Figure 7.8 shows four different realizations of this model. The left column shows a plot of the density profile cross section with an arrow indicating the axes of rotation. The right column shows the corresponding integrated emission maps as calculated by our code. These maps represent emission of HCO^+ J =1–0 and they have been convolved by an 0.7″ Gaussian in order to reproduce the typical submillimeter interferometer resolution.

In the two situations shown in Fig. 7.8 (A and B), the rotation axis of the disk and of the envelope is aligned. In panel B, this axis is inclined by 45° towards the observer. These two situations are axi-symmetric, and can therefore also be reproduced by *RATRAN*. In panels C and D (Fig. 7.8), the rotation axis of the disk is offset from that of the envelope by 45°. In panel C, the system is viewed from the side, whereas in panel D, the disk is inclined towards the observer. These two situations are not reproduceable by two-dimensional codes, since the geometry of this model is intrinsically three-dimensional. The changing disk geometry is clearly observable in the image maps. Each of the models were computed, including raytracing, within an hour on a standard laptop computer, using 10^4 grids points.

7.4.3 Complex structured disks

In the third example we consider a gaseous protoplanetary disk with a density perturbation in the azimuthal direction, in particular, a spiral density wave. This example is constructed specifically to demonstrate the code and the particular setup that we use is very artificial. However, the young stellar object AB Auriga shows sign of a spiral pattern in its disk (Fukagawa et al. 2004, and others). In

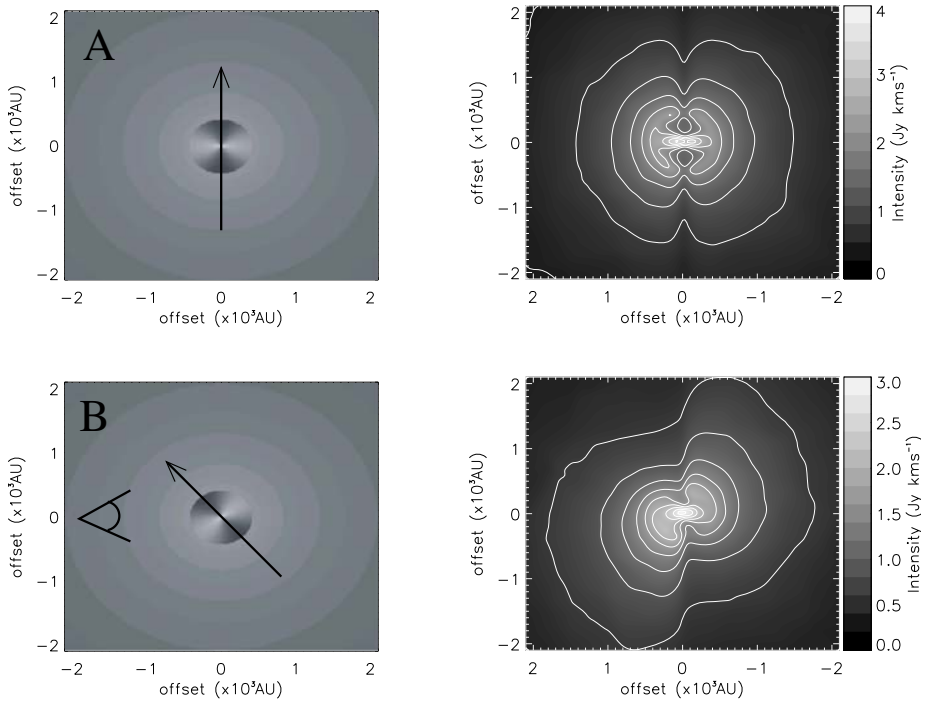

Figure 7.8: The model described in Sect. 7.4.2. The left column shows the density model as well as the rotation axis and viewing angle. The right column shows the intensity distribution of HCO^+ J =1–0 line emission as calculated by our code. This figure is continued on the following page.

general, protoplanetary disks are known to have density structure, such as holes and gaps and with the increased resolution of ALMA, it might be possible to image such structure in the future. The radial and vertical structure of the disk used here is similar to the disk described in Eq. 7.13. The velocity field is kept fully Keplerian, with no components in the radial and vertical directions. For the density spiral wave we modify Eq. 7.13, using the analytic description of Cox & Gómez (2002) for a logarithmic spiral with a pitch angle δ,

$$n(r, \theta, \phi) = n_{disk}(r, \theta) D \cos(\gamma), \tag{7.15}$$

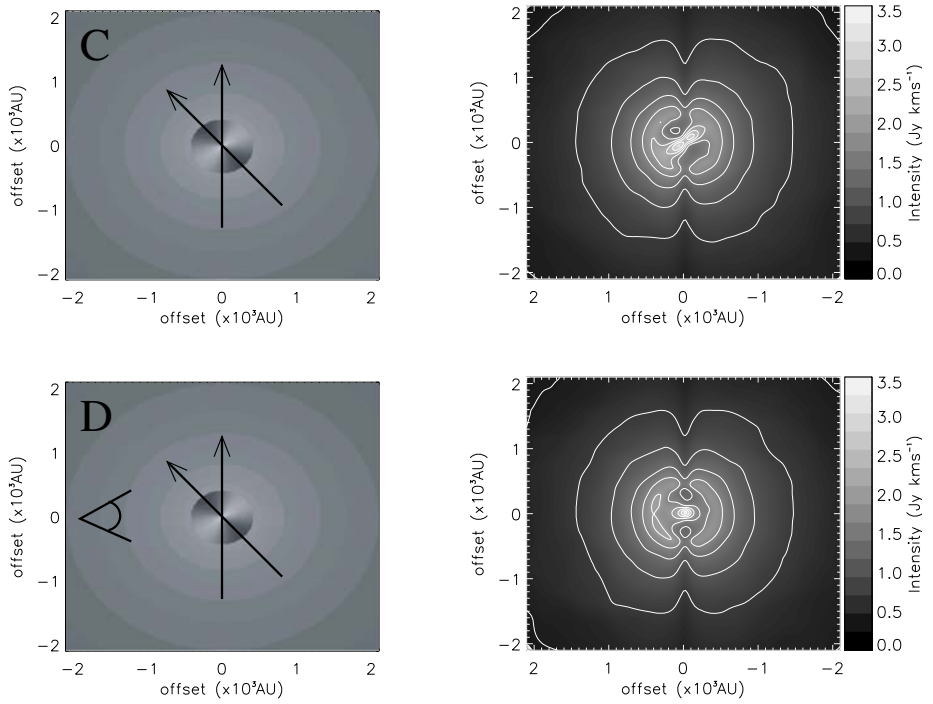

Figure 7.8: Continued from previous page, now for models with different disk and envelope rotation axes.

where

$$\gamma = N\left(\phi - \frac{\ln(r/r_0)}{\tan(\delta)}\right). \tag{7.16}$$

We have chosen a pitch angle $\delta = 0.3$ and the number of spiral arms $N = 2$. The coefficient D is chosen so that the overall disk mass is 0.25 $M_\odot$. In a real science case, a molecular transition with a critical density that falls right in between the peaks of the density perturbation would be needed in order to detect it observationally. Here we have adjusted the density contrast so that the maximum and minimum values for the density wave falls on either side of the critical density for HCO^+ J =1–0 in order to make the spiral pattern visible in the figure. The contrast is several orders of magnitude.

Figure 7.9 shows a face-on view of the integrated intensity of this model. The

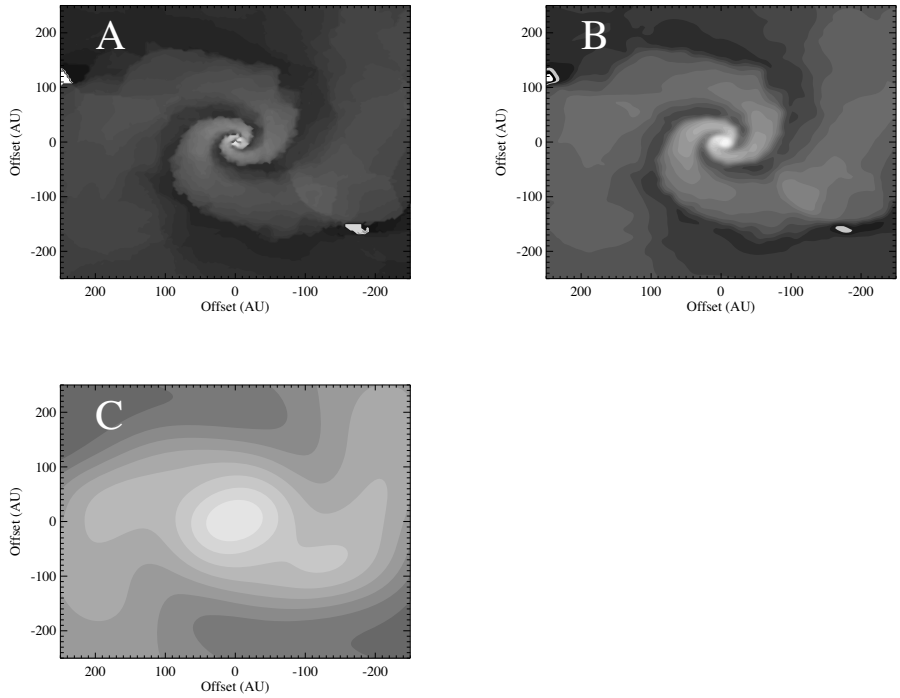

Figure 7.9: The emission signature of a spiral density wave in a face-on disk. The resolution of the three images is **a)** 0.02″, **b)** 0.07″, and **c)** 0.7″.

three panels correspond to three different resolutions, unconvolved (pixel limited, 0.02″), highest ALMA resolution (0.07″) and highest SMA resolution (0.7″).

7.5 Outlook

While the code presented in this chapter is already capable of producing images and spectra which are almost indistinguishable from the corresponding results of *RATRAN*, our code still needs proper testing before it can be applied to real science problems. Some features still needs to be implemented, such as secondary collision partners, separate gas and dust temperatures, proper treatment of lines with hyperfine structure, etc., and additional code optimization needs to be done in order gain full advantage of our method. However, as we have shown in this

chapter, the main code framework is in place and it is working well. With the coming of the Herschel Space Observatory which will observe many H_2O and OH transitions in the far-infrared, and ALMA, which will reach an unprecedented resolution in the submillimeter wavelength regime, codes like the one presented here are certainly going to be needed in order to interpret the observations which these instruments will provide. The main advantage of this code compared to other line radiation transfer codes is faster convergence and a more flexible model input method for high dimensional models. While this code is not the first to use Delaunay lines for photon transport, it is the first one to solve molecular excitation and level population on random triangulated grids.

References

Barber, C. B., Dobkin, D. P., & Huhdanpaa, H. 1996, ACM Trans. Math. Softw., 22, 469

Cox, D. P. & Gómez, G. C. 2002, ApJS, 142, 261

Fukagawa, M., Hayashi, M., Tamura, M., et al. 2004, ApJ, 605, L53

Hogerheijde, M. R. & van der Tak, F. F. S. 2000, A&A, 362, 697

Pavlyuchenkov, Y. N. & Shustov, B. M. 2004, Astronomy Reports, 48, 315

Ritzerveld, J. & Icke, V. 2006, Phys. Rev. E, 74, 026704