Fragment detection in simulated accretion disks

Studienarbeit
Sommersemester 2011

edited by
Jonathan Seyrich

supervised by
Tobias Müller and *Prof. Dr. Wilhelm Kley*

July 15, 2011

Contents

1 Abstract

The aim of this work is to develop an algorithm that is able to detect areas of fragmented mass, i. e. the existence of dense clumps of mass, within an accretion disk. The algorithm presented here, will take as input the surface density distribution of an accretion disk and then decide whether the distribution is such that mass must have been fragmented or not. Furthermore, it will identify all Hill spheres belonging to the fragmented masses.

2 Introduction

One of the most important recent questions in astrophysics concerns the formation of exosolar planets. When a star emerges through the collapse of a rotating molecular cloud, a fraction of roughly 0.1 % to 10 % the original mass composes a so called *accretion disk* or *protoplanetary disk* that rotates around the star to guarantee conservation of angular momentum. After about 10^6 y, most of this mass will be absorbed by the central star. Figure 1 shows observed images of such accretion disks.

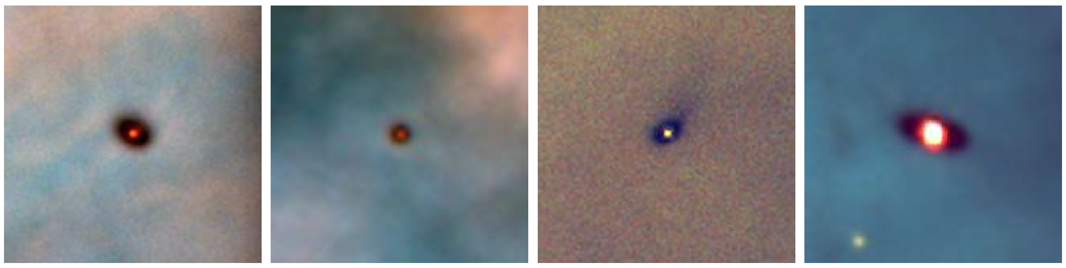
Figure 1: Hubble Space Telescope images of four protoplanetary disks around young stars in the Orion nebula, located 1.500 light-years away. Credit by Mark McCaughrean (Max-Planck-Institute for Astronomy), C. Robert O'Dell (Rice University), and NASA

Today, there exist two competing theories that try to explain the formation of planets out of such disks:

In the *Core Accretion Model*, small dust grains in the otherwise gaseous disk are the origin of the formation of planetesimals. If these grains with an initial extent of no more than 1 µm collide, they can accumulate more mass until they reach a size of up to 1 m. This theory leads to a problem still not solved so far: The grains are particles orbiting the central star at Keplerian velocities. The decay in the gas pressure with increasing distance to the central star represents a pressure gradient with outward direction so that, in contrast to the grains, the gas is rotating with sub-Keplerian velocities. This results in a drag on the solid particles which will, in consequence, spiral inwards as is shown in (Weidenschilling 1977). Weidenschilling (1977) demonstrate that this can happen within a time-scale much to small for the grains to aggregate enough mass to form planet-size objects.

The *Disk Instability Model* or *Gravitational Instability Model* states that planet formation is a direct consequence of a graviational instability caused by overdense regions within the disk. Assuming that the disk's mass is centered in a plane, some linear perturbation equations can be derived. Taking into account a potential describing the self-gravitation of the disk's mass, leads to a dispersion relation that links rotation, pressure and self-gravitation, with the self-gravitation acting as a destabilizing element. For further explanation see Toomre (1964). This theory can be applied to describe the formation of gas giants but were it to describe the formation of very small planets, the disk would have to cool down rapidly. There are legitimate doubts that disks can cool down fast enough to enable the formation for very small planets.

In this work, we limit ourselves to the gravitational instability model relating to which a whole bunch of numerical studies have been realized recently (for an example, see the diploma thesis of (Müller 2010)). As the examined disks are very flat, many of these use a simulation on a polar two-dimensional grid to propagate an initial (often axissymetric) distribution over time, e.g. with the *FARGO-algorithm* presented by (Masset 2000). Regardless of which time-integration algorithm is applied, all these simulations are stopped after given time and the density distribution is scanned for areas of fragmented mass. As, so far, there does not exist any satisfying algorithm or simple criterion, these scans generally have to be done by eye. If the density distribution resembles the examples shown in Fig. 2 on the left and in the middle, it is not difficult to check them for the presence of fragments. But if we have to examine the sort of distribution shown on the right of Fig. 2, it is almost impossible to draw a conclusion by eye.

Therefore, in order to cope with this problem, we want to develop an algorithm that is able to deliver reliable judgements over whether there are fragments or not. This algorithm should do so without the need of any further input than the distribution of the density over the grid.

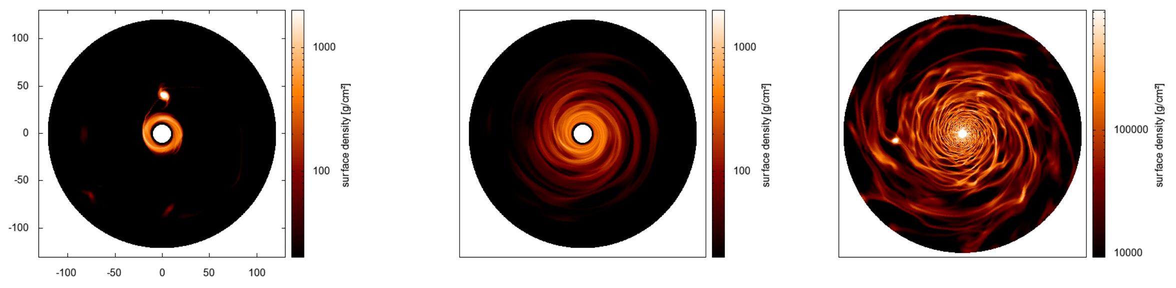

Figure 2: Three exemplary surface density distributions as they occur in our standard simulations. Obviously the example on the *left* is fragmented, whereas the one in the *middle* is not and the *right* one is not easy to decide!

2.1 Outline

In section 3 we will specify our problem. In section 4, we then will first turn the field of *Edge-Detection* in *Image-Processing* and analyse, in which way it resembles our problem. We will see that the questions handled in this field do not match our problem exactly.

In section 6, we will consider the simple idea of dividing the intervalls appearing on the grid in a given number of bins and than trying to draw conclusions on the state of the grid by analysing the distribution of density into the bins. It will become clear that such an approach does not lead to reasonable results.

Thereafter we will present the actual algorithm and expose step by step how it has been constructed. Not only will this algorithm show convincing hit rates in finding fragments but also it will offer the possibility to calculate the Hill spheres of all planetesimals as well as the possibility to fit a Gaussian function to the density distribution of the planetesimals.

3 Specification of our problem

If there are any fragments in a disk rotating a massive central star, some conditions must be satisfied:

As a fragment grows by accretion of mass, the density at the centre of the fragment increases. Consequently, the density inside the fragment exceeds the density of the neighborhood by far. Therefore, fragments can only be detected around grid points that have a local maximum in density. Further, as the density decreases sharply between the centre of a fragment and its surface, we can assume the density distribution to roughly resemble a Gaussian distribution of the form shown in Fig. 3. So, there have to be some grid points in the neighborhood of a local maxima where the gradient operator has a local maximum pointing towards the local maximum.

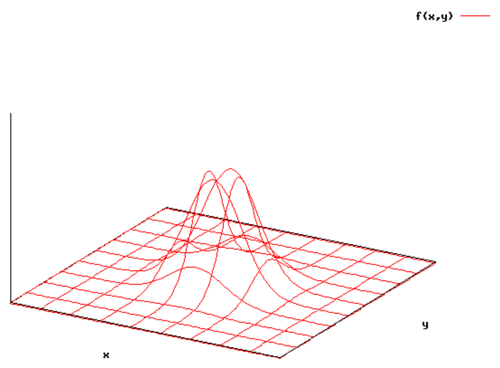

Figure 3: 2-dimensional Gaussian function.

In order to aggregate more mass, a small fragment has to attract this mass in any manner. For this, the attraction exerted on a mass by the fragment must be greater than the attraction exerted by the gravitation of the central star, i.e. we can assume the mass to be within the Hill sphere of the fragment. The Hill sphere will be introduced in section 7.2.

4 Parallels to Image-Processing

As a first step, we consider the analogies between the detection of high-density areas within a protoplanetary disk and the detection of edges in a grey-level image.

4.1 Edge Detection

The aim of edge detection is to spot discotinuities in the brightness of an images. More precisely, one wants to identify points at which the brightness changes drastically. Given an ideal (i.e. non-blurred) image, the whole information of the image is represented by the edges of sharply changing brightness. There are two common types of edge detectors:

Search-based methods calculate a measure of edge strength for every pixel, which usually is the magnitude of the gradient, and then are scanning the image for local maxima of this measure. As the calculation of the gradient can be distorted by outliers in the image-representation, this approach requieres a preprocessing step that smoothes the data. This is normally done by convolving the data with a gaussian kernel.

Depending on which smoothing operator is applied an as a function of the numerical approximations of the gradient operator, many different implementations of the search-based method can by constructed. One simple and long standing possibility is to approximate the gradient operator by the *Sobel operator*. In this case, with B denoting the brightness an $\boldsymbol{\nabla} = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})^T$ denoting the gradient operator, the partial derivatives are given as the result of 2-dimensional convulsions:

$$\frac{\partial B}{\partial x} = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} * B, \qquad \frac{\partial B}{\partial x} = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} * B$$

A still commonly used method is the *Canny edge detector*. Here, the direction of the gradient $\Phi = \arctan \frac{\partial_x B}{\partial_y B}$ is computed along with the magnitude. Then, the direction is rounded to $\Phi = \{0°, 45°, 90°, 135°\}$ and the magnitude $M = \sqrt{\left(\frac{\partial B}{\partial x}\right)^2 + \left(\frac{\partial B}{\partial y}\right)^2}$, in order to be identified as a local maximum, has to be the maximum of all magnitudes in direction Φ. For a detailed explanation of the Canny edge detector see Canny (1986).

In *zero-crossing-based methods*, after smoothing the image to remove outliers, a second-order derivative, e.g. the Laplace-operator, is calculated. Then, the zero-crossings of this expression are located and identified as edges. The *Marr-Hildreth*-algortithm for example convolves the image with the Laplacian of a Gaussian kernel, for the result of which zero-crossings are determined.

Some more recent approaches proceed as follows: First, a local coordinate-system (u, v) is attached to every grid point, with u pointing in the direction of the gradient. Then, with B again denoting the brightness, the directional derivative

$$\partial_u B,$$

which by design is the magnitude of the gradient, is calculated. Requiring the directional derivative of the magnitude to vanish for all points representing an edge, the edges are identified to be the points with

$$\partial_u(\partial_u B) \overset{!}{=} 0$$

and

$$\partial_u^2(\partial_u B) \overset{!}{\leq} 0$$

In both cases, the set of edges returned by the respective algorithm contains edges representing stark contrasts as well as more subtle ones. Consequently, at the next stage, all important edges have to be singled out of the complete set.

A prevalent question in image-detecion is the problem of scale. As a picture usually displays many planes, objects that are more distant to the camera appear much smaller than objects of similar physical size at the front. So, depending on which object on an image is of most interest, different scales are important. On way to single out objects of the desired scale is to conduct the smoothing step using Gaussian kernels with corresponding variance σ. Given a situation where many different scales are of interest, this will take many successive calculations. What is more, this approach is not feasible in fully automated pocessings for the lack of computers with intuition for which object is most relevant. (Lindeberg 2008) tried to do away with this problem introducing *scale-space theory*. Here, the brightness $B = B(x, y, \sigma)$ is considered to be a function of space *and* scale. The scale of most interest can then be singled out by nomalizing the deployed derivatives and maximizing them over scale.

An examplary application of edge detectors is shown in Fig. 4, and a survey of commonly used edge detectors is given by Ziou & Tabbone (1998).

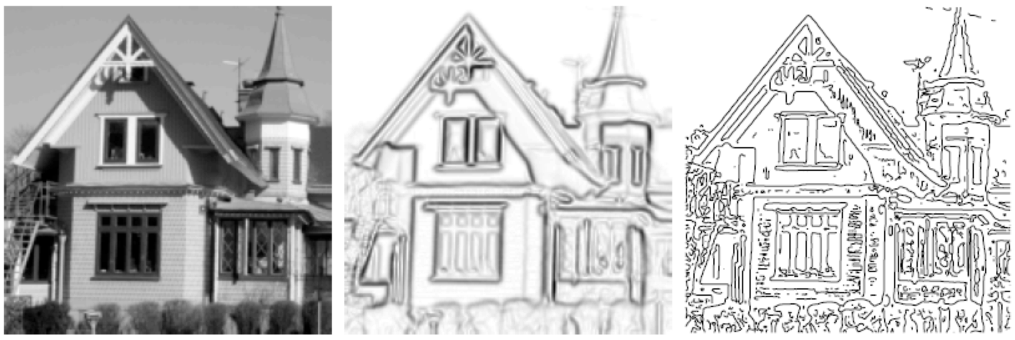

Figure 4: Left: a grey-level image of a house. Middle: the negative value of the gradient magnitude M with $\sigma = 1$. Right: Differential geometric edges as described in the text, with $\sigma = 1$. Credit by Tony Lindeberg (Royal Institute of Technology)

4.2 Comparisons

At first glance, our problem resembles that of finding edges of an image. In either case, areas of interest, i.e. areas which represent a local maximum of a quantity (magnitude of gradient in one case, density in the other), have to be singled out. Then a threshold is applied to make sure, that only the most interesting grid points remain. But taking a closer look, significant differences can be noticed:

At first, we observe that, unlike in edge detection, we do not have the problem of different scales, because we look at a 2-dimensional representation of a 2-dimensional grid whereas an image is a 2-dimensional representation of a 3-dimensional system. Further, it wouldn't be effective to apply an initial smoothing step, as in our case we consider the interesting grid points to be precisely the outliers in density. Another advantageous result of the specifications exposed in **??** is the requirement for the fragments to have a Hill sphere containing the aggregated mass. This yields a reliable threshold for the local density maxima as we will see in section 7.2.

5 Overview of considered simulations

Throughout this work we have considered various simulations, which for one reason or another (as will become clear in the process) are interesting examples to proof the effectiveness of our algorithm. These simulations are listed in Table 1. Plots with the corresponding surface density are presented in the following sections. The denotations used in this work are summarized in Table 2.

Name	Σ profile	M_0 $[M_\odot]$	$r_{\mathrm{min}} - r_{\mathrm{max}}$ [AU]	$N_{\mathrm{rad}} \times N_{\mathrm{az}}$	t [a]	Credit	See
SimA15	$\propto r^{-2}$	1.0	10 – 120	1024×2560	15	Mueller, T.	Fig. 9
SimA20	$\propto r^{-2}$	1.0	10 – 120	1024×2560	20	Mueller, T.	Fig. 9
SimA25	$\propto r^{-2}$	1.0	10 – 120	1024×2560	25	Mueller, T.	Fig. 7
SimB	$\propto r^{-2}$	1.0	10 – 120	512×1280	100	Mueller, T.	Fig. 8
SimC	$\propto r^{-2}$	1.0	1.0 – 2.5	512×1536	13	Meru, F.	Fig. 11
SimD	$\propto r^{-2}$	1.0	1.0 – 2.5	512×1536	40	Meru, F.	Fig. 12
SimE	$\propto r^{-2}$	1.0	10 – 120	512×1280	5	Mueller, T.	Fig. 10

Table 1: List of the simulations considered in this work

denotation	quantity
Σ	surface density
M_0	mass of the central star
M_{tot}	total mass of the disk
r_{min}	inner radius of the disk
r_{max}	outer radius of the disk
N_{r}	number of grid points in radial direction
N_φ	number of grid points in azimuthal direction
N_{grid}	total number of grid points
ρ_{min}	minimum of surface density
ρ_{max}	maximum of surface density
n_φ	azimuthal index ($n_\varphi = 1..N_\varphi$)
n_r	radial index ($n_r = 1..N_{rad}$)
$r(n_r, n_\varphi)$	distance of grid point with index (n_r, n_φ) to central star
$\phi(n_r, n_\varphi)$	angle of grid point with index (n_r, n_φ)

Table 2: List of the denotations used in this work

In the next section we will present a first simple idea to determine whether an accretion disk is fragmented or not and which, for the lack of better term, shall be called *Bin-Accumulation* here.

6 Bin-Accumulation

Given a density distribution as in Fig. 5 we can draw the assumption that at most grid points, the density is very low and only some grid points have medium or high density.
Therefore, if we divide the intervall $[\rho_{\text{min}}, \rho_{\text{max}}]$ into a given number N of bins and assign each grid point to a bin according to its density and plot the mass m_{bin} represented by the grid points of a bin against the bin number n_{bin}, we should get a plot, that resembles the right panel of Fig. 5. Most of the mass is concentrated on grid points with low density, as these grid points constitute the large majority of grid points. Some of the mass is ditributed over grid points with medium density and a bit more of the mass should be represented by the grid points with high density.

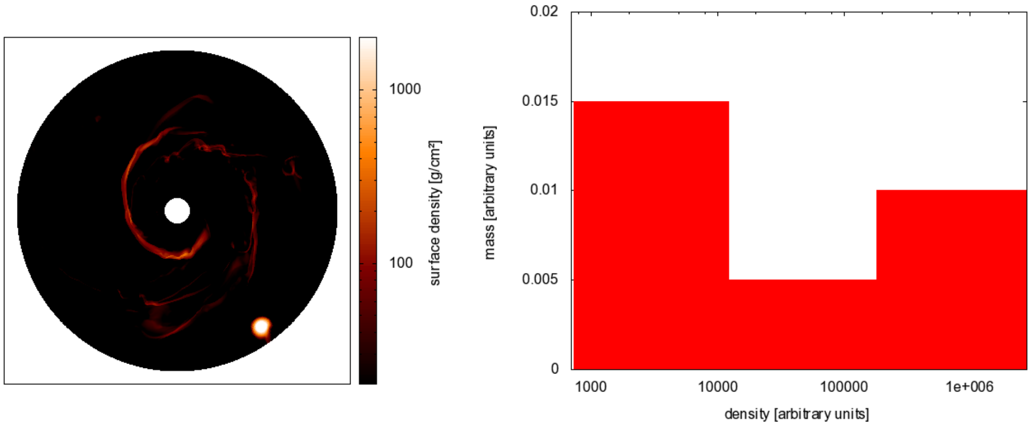

Figure 5: Left: Examplary density distribution of a fragmented disk. Right: Strongly Simplified supposition of a plot of $\frac{m_{\text{bin}}}{M_{\text{disk}}}$ against n_{bin} for $N = 100$.

In contrast, given the sort of distribution shown in Fig. 6, there are no grid points with high density but many grid points with medium density. Thus, much of the mass will be representated by grid points with low density, but an evenly fair share of it will be concentrated on grid points with medium density. Plotting the same quantities as for the fragmented example, we should obtain a diagram similar to Fig. 6 on the right.

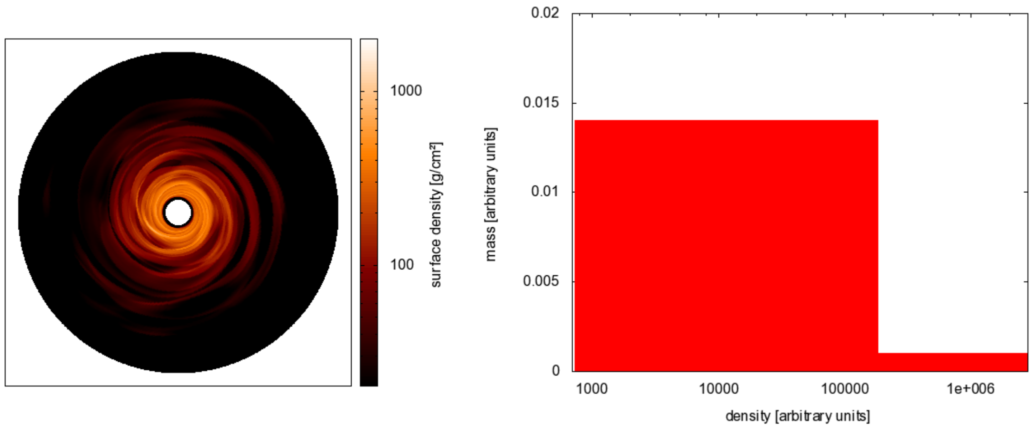

Figure 6: Left: Examplary density distribution of a non-fragmented disk. Right: Strongly simplified supposition of a plot of $\frac{m_{\text{bin}}}{M_{\text{disk}}}$ against n_{bin} for $N = 100$.

Out of these observations we can now construct a simple algorithm that (using the same variable names as in the text above) would be as follows:

```
function Bin_Accumulation(grid)
        array $m_{\text{bin}}[N]$;
        $\rho_{\text{min}}$=calculate_rho_min(grid);
        $\rho_{\text{max}}$=calculate_rho_max(grid);
        divide_intervall_in_bins($\rho_{\text{min}}$,$\rho_{\text{max}}$);
        for all gridpoints do
                $n_{\text{bin}}$=assign_correct_bin(gridpoint);
                $m_{\text{bin}}[n_{\text{bin}}] = m_{\text{bin}}[n_{\text{bin}}]$+gridpoint.mass;
        od
end
```

At first, ρ_{min} and ρ_{max} are calculated. To allow for a better representation of the data (as most of the grid points only have a low density), we choose a logarithmic division of the intervall $[\rho_{\text{min}}, \rho_{\text{max}}]$. With $\rho_{n_{\text{bin}}}$ denoting the lower density limit in bin n_{bin} we choose

$$\rho_{n_{\text{bin}}} = \rho_{\text{min}} \left(\frac{\rho_{\text{max}}}{\rho_{\text{min}}}\right)^{\frac{n_{\text{bin}}}{N}} .$$

To facilitate the analysis of the plots, we substitute the relative mass for the accumulated relative mass, i.e.

$$\frac{m_{\text{bin}}[n_{\text{bin}}]}{M_{\text{disk}}} \to \frac{\sum_{i=1}^{n_{\text{bin}}} m_{\text{bin}}[i]}{M_{\text{disk}}}$$

Such an algorithm is easy to implement and surely is not too expansive, as it only calculates ρ_{min} and ρ_{max}, going over each of the grid points once and than runs through the grid one more time to calculate the masses of the bins, it solely conducts $\mathcal{O}(N_{\text{grid}})$ operations for a disk with N_{grid} cells.

It now remains to be seen if this algorithm can help to detect fragments with much more accuracy than the analysis by eye. We have applied the algorithm to several distributions with $N = 100$. All of them represent snapshots of real simulations of accretion disks. At first, in Fig. 7 and Fig. 8 we once again consider the examples from the beginning of this section:

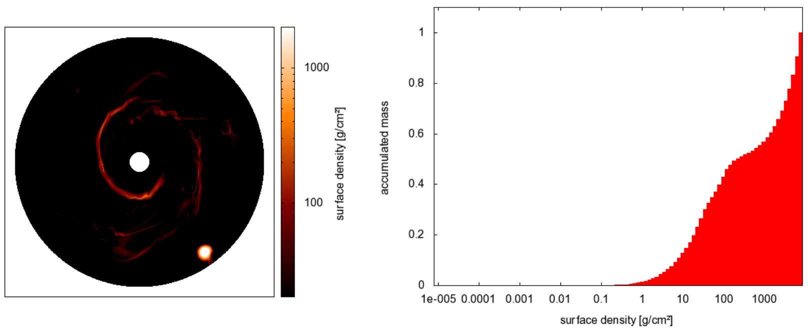

Figure 7: Surface density distribution (left) and accumulated mass (right) of SimA25.

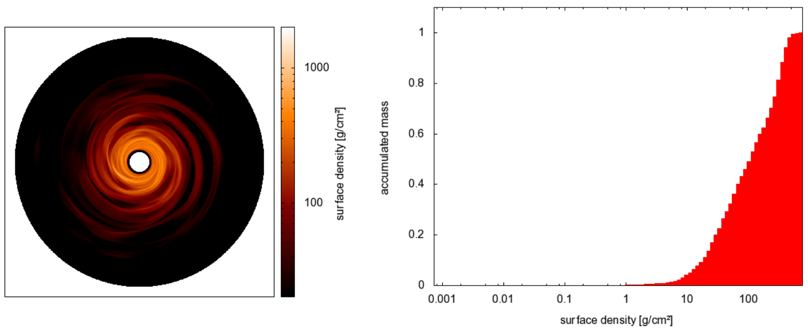

Figure 8: Surface density distribution *(left)* and accumulated mass *(right)* of SimB.

As we see, the plots obtained by the algorithm confirm our expectations stated above. We could easily identify the first example to be fragmented and the second not to be without having a look at the 2-dimensional plots. Another example out of the same simulation as the one of Fig. 7 is shown in Fig. 9 together with the according plot. Here too, the plot shows the characteristic sharp increase in accumulated mass for large n_{bin} and we would consider the disk to have fragments without regarding the 2-dimensional density distribution.

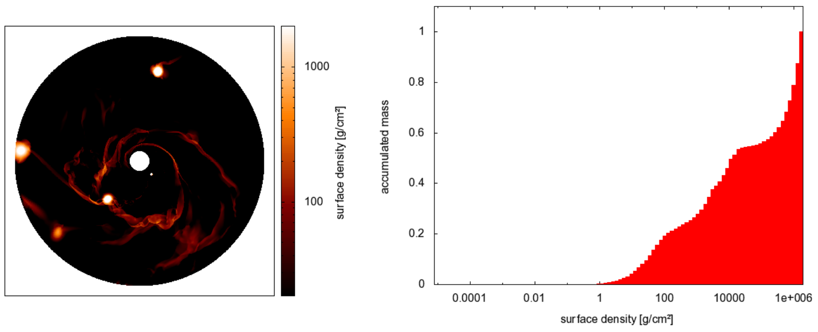

Figure 9: Surface density distribution *(left)* and accumulated mass *(right)* of SimA20.

But this so far does not bring ous any further as it would be no more difficult to figure out the nature of these two examples by looking at the 2-dim. plot (as we have already mentioned in Section 2) To be a real help, the programm would have to deliver reliable results for distributions as at the right of Fig. 2 and the like. Even regarding the very first example at the left of Fig. 2, we obtain Fig. 10 and comparing this with the 2-dim representation, it would be more easy to judge by the left plot.

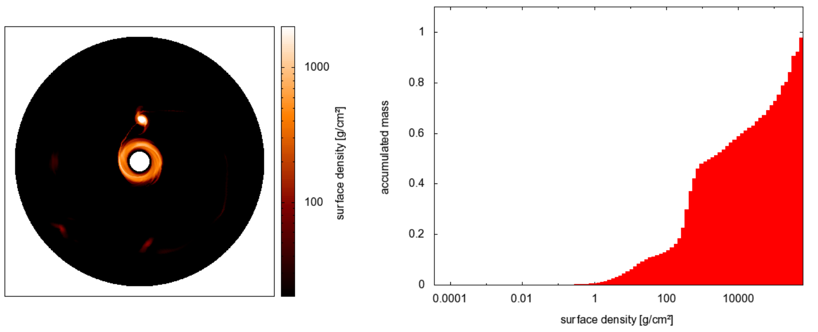

Figure 10: Surface density distribution *(left)* and accumulated mass *(right)* of SimE.

Up to this point we could expand our algorithm to include a function to process the masses in the respective bins automatically further and so even Fig. 10 could be fully analyzed by a fast algorithm. However, if we now finally remember Fig. 2 (shown again in Fig. 11) and consider Fig. 12 simultaneously, the algorithm returns very similar plots. Already at first glance, we can detect differences between Fig. 11 and Fig. 12. Nevertheless, Bin-Accumulation leads to almost identical results.

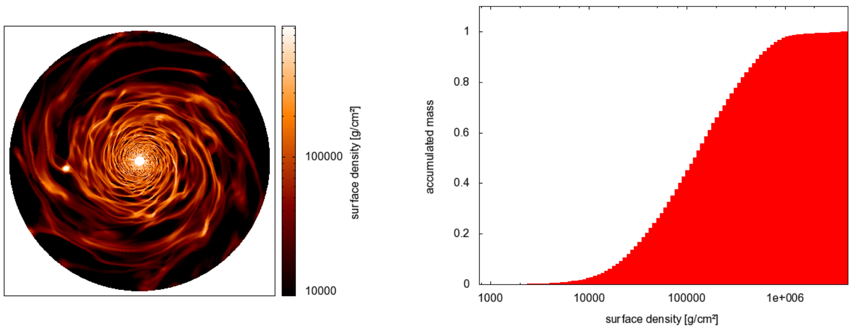

Figure 11: Surface density distribution *(left)* and accumulated mass *(right)* of SimC.

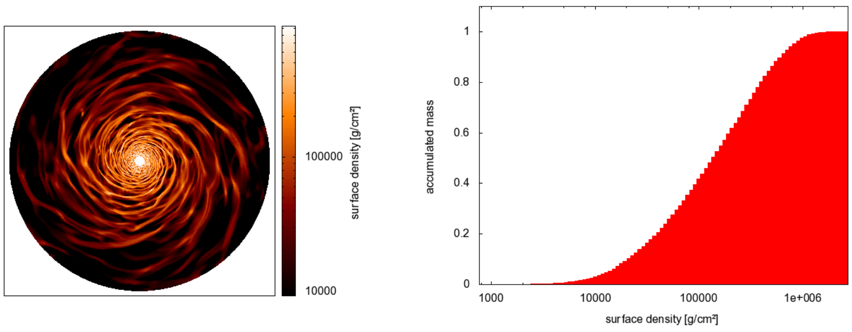

Figure 12: Surface density distribution *(left)* and accumulated mass *(right)* of SimC.

This could have two reasons: Either, the differences between the distributions in Fig. 11 and Fig. 12 observed by eye are just marginal and the two disks really have the same properties, or Bin-Accumulation does not cope with disks which resemble these two elements. This would be a fatal blow to Bin-Accumulation, because the latter are exactly the sort of disks that do not allow for analysis by eye. But Bin-Accumulation returning the same output for different disks is no surprise as we can easily demonstrate:

Let $D(r,\phi)$ be a polar grid with coordinates r,ϕ, radius R and exactly one fragment around a local maximum at $r = 0.9R, \phi = \pi$(without loss of generality) that satisfies the conditions, stated in section 3 just by a whisker. Now, consider a path $\vec{r}(t) : [0,1] \to \mathbb{R}^2, t \mapsto (\alpha t, \pi)$ with $\vec{r}(0) = (r_{\text{min}}, \pi)$ and $\vec{r}(1) = (r_{\text{max}}, \pi)$. Along this path, the density distribution $d(r,\phi)$ shall be given as illustrated in the first plot of Fig. 13. Then, consider a disk $\tilde{D}(r,\phi)$ that is exactly the same as $D(r,\phi)$ except for the points along the path $\vec{r}(t)$, where the distribution $\tilde{d}(\vec{r}(t))$ shall be given as illustrated on the right hand side of Fig. 13. As D satisfies the conditions to be fragmented extremely close, its mass is just big enough to attract the surrounding particles. Because the attraction of the central star increases with decreasing radius, the mass of the local maximum in $\tilde{D}$ is too small to attract any mass and so $\tilde{D}$ does not satisfy the conditions to be able to contain dense clumps. But nevertheless, Bin-Accumulation would return exactly the same output.

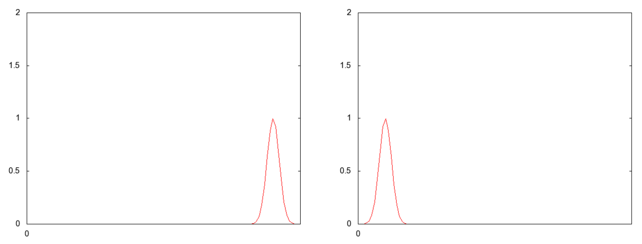

Figure 13: Left: Qualitative density distribution d of disk D along path $\vec{r}(t)$. Right: Qualitative density distribution $\tilde{d}$ of disk $\tilde{D}$ along path $\vec{r}(t)$.

After we made sure that Bin-Accumulation is no useful tool to check for fragmented disks, as it just takes account of the densities regardless of their positions, we now want to describe the implementation of a suitable algorithm in detail.

7 Implementation of the algorithm

To design an algorithm for the detection of dense clumps, we have to bear in mind the specifications of a fragmented disk, stated in section 3. By applying these conditions to the grid points, we will successively filter out all the grid points that do not present fragmented areas until only the actually fragmented areas are left behind.

7.1 Local maxima

As outlined in section 3, dense clumps can only be found around grid points with a local maximum of surface density. Therefore, it would be reasonable to have a list of all local maxima, as these represent the interesting areas of the disk. The first task of the algorithm should be to calculate the local maxima. Understandably, we consider a grid point P to be a lokal maximum, if non of its surrounding grid points P_j has a greater density ρ. The first step ouf our algorithm then is as follows:

```
list $localmax$
function calculate_localmax(grid,$localmax$)
    for all gridpoints $P$ do
        if no_neighbor_has_greater_density($P$) then
            $localmax$.add($P$);
        fi
    od
end
```

Clearly, the list *localmax* now contains way too much grid points, as for example even the highlighted grid points in Fig. 14 would be part of the list but, if anything, the mass represented by these points could perhaps be attracted by the much denser grid points in their approximity.

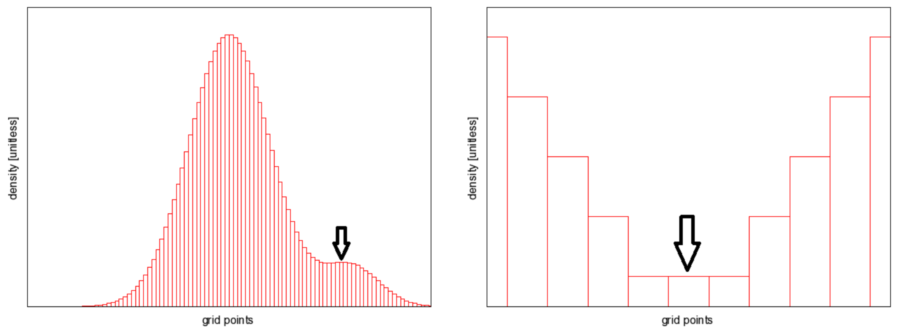

Figure 14: Two schematical examples of grid points (highlighted) that are lokal maxima but clearly would not be at the centre of fragments.

As a realistic example, we can again consider SimA25 and calculate the local maxima. Fig. 15 shows all maxima found by the algorithm.

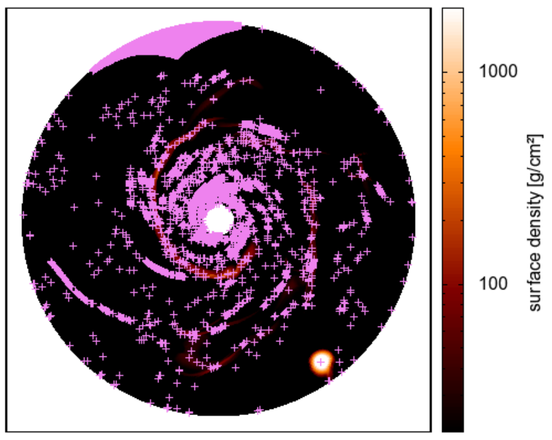

Figure 15: SimA25 with all local maxima (number of local maxima > 17000). Every cross corresponds to one maximum.

To get rid of the redundant maxima, we now utilize the further conditions stated in section 3.

7.2 Hill sphere

As mentioned briefly in section 3, we can assume a local max to be part of a fragment if it is heavy enough to attract surrounding particles stronger than the central star. Before we think about how to apply this condition in the algorithm, we want to give some short theoretical background on the Hill sphere.

Given a astonomical body P that orbits a massive central body, the *Hill sphere* or *Roche lobe* of the astronomical body is the area within which objects O are more attracted by the P than by the central body M_0. The potential Φ of the O within the gravitational field of M_0 and P, denoted as *Roche potential*, is composed of three terms: The potentials of the gravitational forces of M_0 and P and a zentrifugal term caused by the movement relative to the centre of mass C of M_0 and P. With r_{M_0} denoting the distance between O and M_0, r_P denoting the distance between O and P, r_C for the distance between O and C and the masses M_0 and m_P, the potential can be written as

$$\Phi(O) = -\frac{GM_0}{r_{M_0}} - \frac{Gm_P}{r_P} - \frac{\omega^2 r_C^2}{2},$$

where G is the gravitational constant and ω is the azimuthal velocity. Given a coordinate system with M_0 at the origin and P on the x-axis at distance $r(P)$ to M_0 (see Fig. 16), we can transform the general expression for Φ.

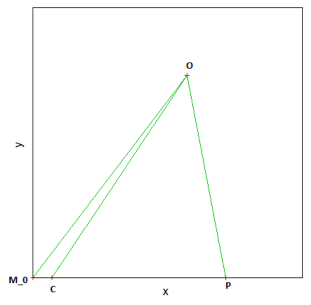

Figure 16: Object O in the gravivational field of M_0 and P

The orbits of objects in a protoplanetary disk are Keplerian, i.e. with $M = M_0 + m_P$, it is

$$\omega^2 = \frac{GM}{r(P)^3}.$$

Using this relation and $q := \frac{m_P}{M_0}$, the potential as a function of the coordiantes (x, y) of the illustrated coordinate system is

$$\Phi(x,y) = \frac{GM}{r(P)} \left[\frac{1}{1+q}(x^2+y^2)^{-1/2} + \frac{q}{1+q}\left((x-1)^2+y^2\right)^{-1/2} + \frac{1}{2}\left(\left(x - \frac{1}{1+q}\right)^2 + y^2\right)\right]$$

The two points, where the boundary of the Hill sphere crosses the line connecting M_0 and P are denoted as L_1 (between M_0 and P) and as L_2 (on the connecting line but with greater distance to M_0 than P). L_1 is the point where the distance between P and the boundary of its Hill sphere is maximal. At L_2, the distance is minimal. L_1 is the point where the attraction by M_0 is exactly compensated by the sum of the attraction by P and the zentrifugal force which at the point L_1 both point into the direction of M_0. The boundary of the Hill sphere is an equipotential line, i.e. all points on that line have the same potential as L_1.

How can we use this information to construct a filter for the local maxima? As a suitable simulation of a protoplanetary disk uses a high-resolution grid, there will be many grid points within the Roche lobe of a fragment. W we can assume a local maximum at the centre of a dense clump to be dense enough for its Roche lobe to contain the nearest grid points. More precisely, with *initial_roche_range* denoting a integer factor, we can assume that starting at a local max P at the centre of a fragment, the grid points *initial_roche_range* steps in left, right, outer and inner direction still are completely within the Roche lobe of P. In section 8, we will test which *initial_roche_range* would be most effective but alternatively it could be a parameter for the algorithm depending on how strict a search for fragments should be conducted. As an illustration, see Fig. 17 where $initial\_roche\_range = 3$.

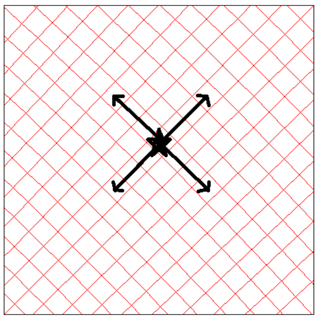

Figure 17: Starting at the grid point ★, the grid points 3 steps to the right/left/out/in are checked for if they are within the Roche radius of ★.

As we have seen above, the point on the Roche radius L_2 with minimal distance to P lies in outer direction on the line connection the central star and P. It therefore is sufficient, to just check the grid point *initial_roche_range* steps in outer direction. To do so, the algorithm at first has to calculate the point L_1, i.e. we have to search the roots of the directional derivative of the gravitational potential Φ in direction of the line $\overline{M_0 P}$. With u denoting the direction of the line $\overline{M_0 P}$ we seek L_1 with

$$\partial_u \Phi(L_1) \overset{!}{=} 0.$$

As L_1 is the only zero-crossing of $\partial_u \Phi$ between M_0 and P, we can use a bisection algorithm. With the knowledge of L_1, we now can calculate the Roche radius for every given angle ϕ relative to $\overline{M_0 P}$: We start at a point $\vec{r}_1$ which has an angle ϕ relative to $\overline{M_0 P}$ an distance $\|L_1 - P\|$ to P. Then, again with the use of a bisection algorithm, we scan the line $\overline{\vec{r}_1 P}$ for the point $\vec{r}_2$ where

$$\Phi(\vec{r}_2) - \Phi(L_1) \overset{!}{=} 0$$

We can do so, because L_1 is the point on the Roche radius with maximal distance to P. For illustration see Fig. 18.

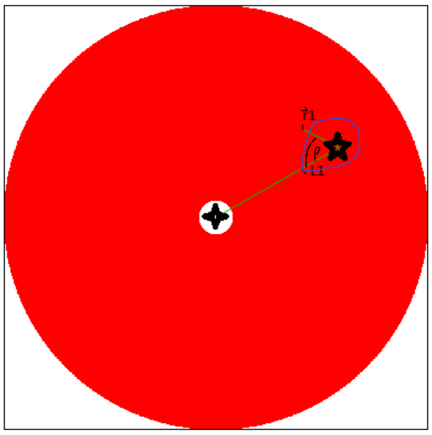

Figure 18: Illustration of L_1, $\vec{r}_1$ and ϕ as mentioned in the text, with ★ hightlighting the position of P. The Roche lobe is represented by the blue curve.

With denotations as in the text, the function that controls if the Hill sphere of a local maximum is too small, reads as follows:

```
int initial_roche_range;
function hillsphere_too_small(localmax)
        for all $P \in localmax$ do
                calculate_$L_1$($P$);
                calculate_roche_radius($P$,$\phi = \pi$);
                if is_not_in_hillsphere($P$,initial_roche_range) then
                        localmax.remove($P$);
                fi
        od
end
```

First, L_1 is computed for all points P in the list *localmax*. Then, the Roche radius at relative angle $\phi = \pi$ is identified and finally, the algorithm checks if the point *initial_roche_range* steps in outer direction is part of the Hill sphere.

If we test the function on SimA20, we obtain the local maxima marked by crosses in the left plot of Fig. 19. As we can observe, there are only local max within areas of relativly great density. But, as the density around the localmax denoted in the plot as P_3 is indeed much greater as the density of most other grid points but still so low as it should not be identified as a fragment, discarding all local maxima with too small a Hill sphere still is not enough. Further, if we not only want to check a disk whether it is fragmented but also want the algorithm to return the exact number of fragments, there cannot be two local max like P_1 and P_2 of Fig. 19. To illustrate this, the left panel of Fig. 19 shows the qualitative 1-dim. shape of the density ditribution around P_1 and P_2. Even if both grid points have a sufficiently large Hill sphere, they surly would be part of the same fragment. Another example is shown in the right panel of Fig. 19. The encircled localmax have enough density for their attraction to be greater as the attraction of the central star. But clearly being part of a so called spiral arm, they are not part of a fragment.

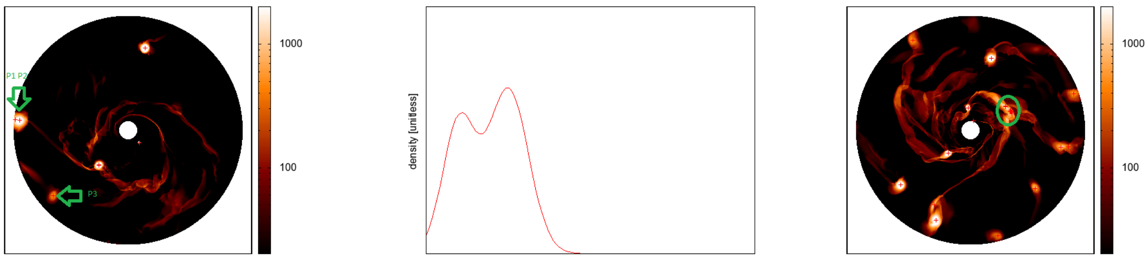

Figure 19: Surface density and local max that with *initial_roche_range* = 3 have a large enough Hill sphere for SimA20 (left) and SimA15 (right). The middle plane shows a 1-dim. qualitative shape of the surface density around the two local max P_1 and P_2 of the left plot.

Another problem arising by applying the function *hillsphere_too_small*() to the list of all local maxima lies in the runtime: For every local maxima P that is tested for a large enough Roche lobe, there first has to be calculated the point L_1 and then the Roche radius for relative angle $\phi = \pi$. Both this operations requiere an iterative numerical algorithm to find zero-crossings. Having to apply such an algorithm to more than 17000 local maxima as would be the case in SimA25 (see Fig. 15) therefore leads to an excessive runtime.

As we have just seen, it would be appropriate to let *hillsphere_too_small*() be preceded by other functions. On the one hand, these functions should filter out most of the local maxima so that the numerically expansive function *hillsphere_too_small*() just has to check a small rest of candidates. On the other hand, these functions should also discard all local maxima that resemble the left maximum of the middle-plane plot of Fig. 19 (i.e. P_1 in the corresponding left density distribution) or maxima similar to the ones qualitatively plottet in Fig. 20, which (strongly simplified) represent maxima within a spiral arm.

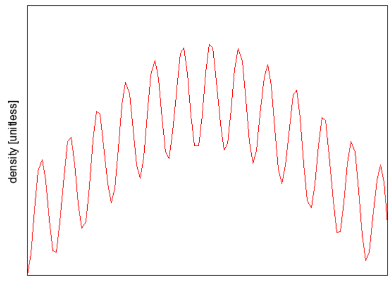

Figure 20: 1-dim. qualitative shape of the surface density within a spiral arm.

To find adequate functions, we remember another specification of our problem mentioned in section 3. There, we have discussed that the surface density around a fragment resembles a Gaussian function shown in Fig. 3, i.e. within a small range around a local maximum P the density is extremly high but than decreases sharply with increasing distance to P and stays flat within a larger range around P.

Out of this observation, two usefull criterions can be derived. These, together with a function that uses the respective criterion, shall be presented in the following to subsections.

7.3 Slope criterion

Given the qualitative shape of the density within and around a fragment, we can derive that there exists a grid point in the approximity of the corresponding local max P where the magnitude of the gradient $M := \|(\partial_x, \partial_y)^T\|$ is much greater as the average of M over all grid points within a wider range around P. Further, it is clear that for all grid points around P the gradient must point to P. Therefore, with *slope_multiplyer* denoting an integer factor, there has to be a grid point P_1 around P where

$$M(P_1) \geq \mathit{slope\_multiplyer} \cdot \langle M \rangle$$

with $\langle M \rangle$ denoting the azimuthally averaged gradient magnitude. We can implement this criterion as follows:

As we are working with a polar grid, the gradient is given as

$$\boldsymbol{\nabla} = \hat{e}_r \partial_r + \hat{e}_\phi \frac{1}{r} \partial_\phi$$

which we calculate via the familiar central differences approzimation

$$(\boldsymbol{\nabla}\rho(n_r, n_\varphi), \hat{e}_r) = \frac{\rho(n_r+1, n_\varphi) - \rho(n_r-1, n_\varphi)}{r(n_r+1, n_\varphi) - r(n_r-1, n_\varphi)}$$
$$(\boldsymbol{\nabla}\rho(n_r, n_\varphi), \hat{e}_\phi) = \frac{\rho(n_r, n_\varphi+1) - \rho(n_r, n_\varphi-1)}{\frac{2\pi}{N_\varphi} r(n_r, n_\varphi)}.$$

Because M decreases with increasing radial distance r, grid points with small radial index can have a greater M than grid points with greater distance to the central star. Keeping this in mind, we calculate an average expression $\langle M \rangle$ for all n_r

$$\langle M \rangle(n_r) = \frac{1}{N_\varphi} \sum_{i=1}^{N_\varphi} M(n_r, i)$$

and implement the following function that tests if the slope around a local max is too small:

```
int $slope\_multiplyer$;
bool $too\_small = true$;
funtion slope_too_small($localmax$)
        for all radial indizes $n_r$ do
```

```
                    calculate_$\langle M\rangle(n_r)$;
          od
          for all $P \in localmax$ do
                    repeat
                              go_to_next_point_$(n_r, n_\varphi)$();
                              if $M(n_r, n_\varphi) \geq slope\_multiplyer \cdot \langle M\rangle(n_r)$ then
                                        $too\_small$=false;
                              fi
                    until does_not_point_to_$P(\nabla(n_r, n_\varphi))$
                    if $too\_small$ then
                              $localmax$.remove($P$)
                    fi
          od
end
```

Here, the function *go_to_next_point_*$(n_r, n_\varphi)()$ is constructed that, given a grid as in Fig. 21 where ★ shall denote P and the circle shall mark the points where the gradient changes direction, all points within the circle are returned onces.

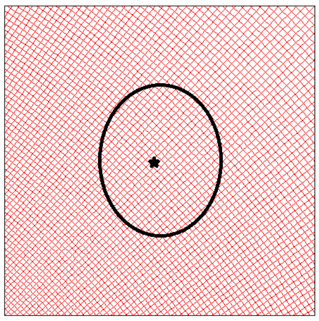
Figure 21: Illustration of the functionality of *go_to_next_point_*$(n_r, n_\varphi)()$. Every point within the circle is returned onces.

To find a suitable value for *slope_multiplyer*, we proceeded as follows:

Regarding SimE, we see that this simulation obviously has one concise fragment (see Fig. 10). Therefore, we fitted a Gaussian function to the density distribution around the local maximum P at the center of the fragment. This yielded a Gaussian with $\sigma = 0.167$. We then plottet a 1-dim. Gaussian function with $\sigma = 0.167$, centred in an intervall $[-\pi r(P), +\pi r(P)]$ which we divided in $N_{\text{az,SimE}}$ bins. Averaging the according Gaussian over the bins, we got Fig. 22 and found out, that there is a bin whose gradient magnitude M exceeds the average of all bins by a factor 400. Therefore, taking the fragment of SimE as a benchmark fragment, and leaving space for less sharp fragments, we concluded that *slope_multiplyer* $= 100$ is an adequate value.

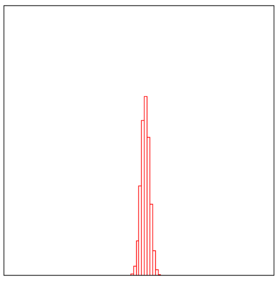

Figure 22: A Gaussian function with $\sigma = 0.167$ on an intervall $[-125, 125]$ divided in 1280 bins, plotted for the subinterval $[-5.5, 5.5]$.

As an example of the functionality of *slope_too_small*(), choosing $slope\_multiplyer = 100$ we have applied the function to SimA15. In Fig. 23, we see that the number of local max is not extremely high. Further, the encircled areas, which out of experience are no fragments, are discarded. Nevertheless there are still many points within the highlighted spiral arm because in this arm, density changes sharply whereas for the other grid points with identical n_rad the density hardly changes at all.

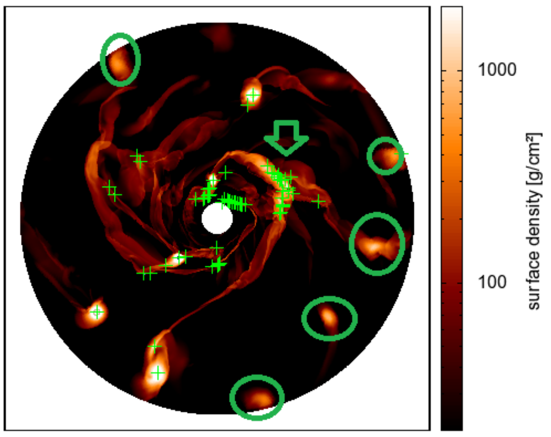

Figure 23: Local maxima of SimA15 after the function *slope_too_small*() has been applied.

7.4 Density criterion

As we could see, applying *slope_to_small*() and *hillsphere_too_small*() to all local maxima P of SimA15 still would return some P within the spiral arm. So, we now interpret the according specification in another way. As the density of P at the centre of a fragment exceeds the density of all grid points within a wider range around P by far, we compare $\rho(P)$ with the average of ρ over all grid points within that range. As adjacent max with extremely high density appear mostly within spiral arms that extend in azimuthal direction, we take the average only in azimuthal direction,, i.e. considering the wider range to consist of all (n_r, n_φ) with $|\phi(n_r, n_\varphi) - \phi(P)| \leq \frac{\pi}{8}$ we calculate:

$$\langle\rho\rangle(P) = \frac{8}{N_\varphi} \sum_{i=-\frac{N_\varphi}{16}}^{i=+\frac{N_\varphi}{16}} \rho(n_{rad,P}, n_{az,P} + i)$$

With a integer multiplyer *multiplyer* we consider a localmax to satisfy criterion $(*)$ if

$$\rho(P) \geq multiplyer \cdot \langle\rho\rangle(P).$$

With that, we can implement a function *density_too_small*() that sorts out all local maxima whose density is too small for them to satisfy the according condition:

```
int $multiplyer$;
function density_too_small($localmax$)
        for all $P \in localmax$ do
                calculate_$\langle\rho\rangle(P)$;
                if $\rho(P) \leq multiplyer \cdot \langle\rho\rangle(P)$ then
                        $localmax$.remove($P$);
                fi
        od
end
```

To find a suitable value for the factor *multiplyer* we proceeded in an analoguous way to finding the factor *slope_multiplyer*:

Again, we assumed that every fragment on a protoplanetary disk has to resemble the benchmark fragment of SimE to which we had already fitted a Gaussian function with $\sigma = 0.167$ for the determination of *slope_multiplyer*. As, in order to calculate $\langle\rho\rangle(P)$, we this time only consider the azimuthal intervall $\phi_P \pm \frac{\pi}{8}$, we consider the Gaussian function with $\sigma = 0.167$ on the intervall $[-\frac{\pi}{8}r(P), +\frac{\pi}{8}r(P)]$ which we divide into $\frac{N_{az,SimE}}{8}$ bins. In the grids used to simulate accretion disks, the density value at every grid point represents the average of the density within the area of that grid point. Therefore, we assign to each bin the average of the Gaussian within the subinterval represented by the bin. That is, if a bin b represents the subintervall $[c, d]$ its density value is given as

$$\rho(b) = \frac{1}{d-c} \int_c^d e^{-\frac{x^2}{2\sigma^2}} dx.$$

With b_0 denoting the bin containing the maximum of the Gaussian, we now can calculate the quotient q of the density in b_0 and the average of all bins b_i:

$$q = \frac{\rho(b_0)}{\frac{8}{N_{\text{az,SimE}}} \sum_{i=1}^{N_{\text{az,SimE}}} \rho(b_i)}$$

We now assume that every fragments resembles SimE and that the variance σ of the corresponding fitted Gaussian only depends on the radial distance $r(P)$ between the maximum at the centre of the fragment P and the central star (we can assume this because fragments in simulations tend to have a larger extend the further away they are from the central star), i.e. with a constant factor a we have $\sigma = a \cdot r(P)$. Given that this assumption has to be valid for SimE, too, a has to be $a = \frac{r(P_{\text{SimE}})}{\sigma_{\text{SimE}}}$. With this two assumptions we can now show, that the quotient q is a function of only the azimuthal resolution of a simulation:

With $x_{0,i}$ and $x_{1,i}$ denoting the lower and the upper limit of the subintervalls b_i plus L denoting the length of a subinterval, we have

$$q = \frac{\rho(b_0)}{\frac{8}{N_\varphi}\sum_{i=1}^{N_\varphi}\rho(b_i)} = \frac{\frac{1}{L}\int_{x_{0,0}}^{x_{1,0}} e^{-\frac{x^2}{2\sigma^2}}dx}{\frac{8}{N_\varphi}\sum_{i=1}^{N_\varphi}\frac{1}{L}\int_{x_{0,i}}^{x_{1,i}} e^{-\frac{x^2}{2\sigma^2}}dx} = \frac{\frac{1}{L}\int_{L/2}^{L/2} e^{-\frac{x^2}{2\sigma^2}}dx}{\frac{8}{N_\varphi}\frac{1}{L}\int_{-\frac{\pi}{8}r(P)}^{+\frac{\pi}{8}r(P)} e^{-\frac{x^2}{2\sigma^2}}dx}$$

As $\frac{\pi}{8}r(P) \gg \sigma$, we get

$$\int_{-\frac{\pi}{8}r(P)}^{+\frac{\pi}{8}r(P)} e^{-\frac{x^2}{2\sigma^2}}dx = \int_{-\infty}^{+\infty} e^{-\frac{x^2}{2\sigma^2}}dx = \sqrt{2\pi}\sigma$$

and with the errorfunction

$$\text{erf}(x) = \frac{2}{\pi}\int_0^x e^{-t^2}dt$$

we can write (based on the symmetry of the Gaussian and the errorfunction)

$$q = \frac{\frac{1}{L}\sqrt{2\pi}\sigma\,\text{erf}(\frac{L}{2\sqrt{2}\sigma})}{\frac{1}{L}\frac{8}{N_\varphi}\sqrt{2\pi}\sigma}$$

Substituting

$$\sigma = a \cdot r(P); \qquad L = \frac{\frac{\pi}{4}r(P)}{\frac{N_\varphi}{8}},$$

after some transformations, we arrive at

$$q = \frac{a\sqrt{2\pi}N_\varphi\,\text{erf}(\frac{1}{2a\sqrt{2}N_\varphi})}{8a\sqrt{2\pi}}$$

and as we see, q only depends on N_φ.

To allow for a more realistic estimate, we add a constant c to the Gaussian fit that represents the average density within the Roche lobe around P. The quotient then changes to

$$q \to \tilde{q} = \frac{\rho(b_0) + c}{\frac{8}{N_\varphi} \sum_{i=1}^{N_\varphi} \rho(b_i) + \rho(b_0) + c}$$

where, for the ease of implementation in the algorithm, we have added another term $\rho(b_0)$ in the denominator. Assuming every fragment to have a qualitative shape similar to SimE, we can now set

$$multiplyer = 0.2 \cdot \tilde{q} = 0.2 \frac{a\sqrt{2\pi} N_\varphi \operatorname{erf}(\frac{1}{2a\sqrt{2}N_\varphi}) + c}{8a\sqrt{2\pi} + 8a\sqrt{2\pi}\operatorname{erf}(\frac{1}{2a\sqrt{2}N_\varphi}) + c},$$

where we inserted a factor of 0.2 to comprise fragments that are not as sharp as SimE.

Having found a way to choose the right factor *multiplyer* for the function *density_too_small*(), we can now, as an example, apply this function to all maxima of SimA15 whereupon only the maxima in Fig. 24 are left behind. In opposition to *slope_too_small*() and *hillsphere_too_small*(), the function *density_too_small*() discarded all local maxima within the encircled spiral arm. In exchange, the points marked by arrows, correctly discarded by *density_too_small*(), are not sorted out by the density criterion.

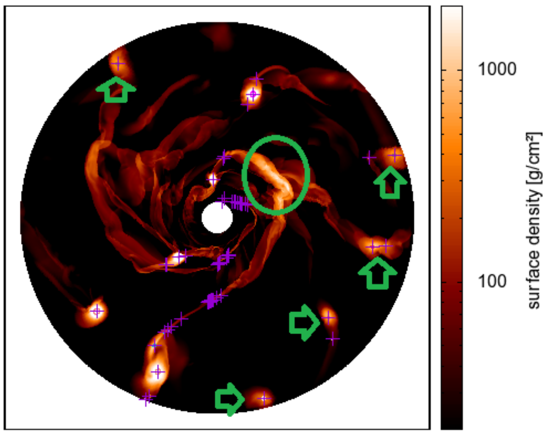

Figure 24: Local maxima of SimA15 after the function *density_too_small*() has been applied.

7.5 Composition of criterions

As we have seen in the preceeding subsections, although non of the functions *slope_too_small*(), *hillsphere_too_small*() and *density_too_small*() can discard all redundant maxima by itself, each function can sort out some type of bad candidates. Therefore, it only would be rational to use a composition of all three of them. With regard to the sequence, one clearly should apply the functions in ascending order of their runtime. As *hillsphere_too_small*() uses two bisectional algorithms over the runtime of which we have no control, we should first execute the other two functions. Given that in *slope_too_small*() we calculate averages of N_φ grid points whereas in *density_too_small*() the averaging is over just $\frac{N_\varphi}{8}$ grid points, the latter function shall be executed first. Having stipulated the right sequence we can now apply all functions to SimA15 (choosing $initial\_roche\_range = 3$) whereafter only the maxima in Fig. 25 remain. Now, all undesired candidates have been discarded and we can assume the disk to contain 8 fragments.

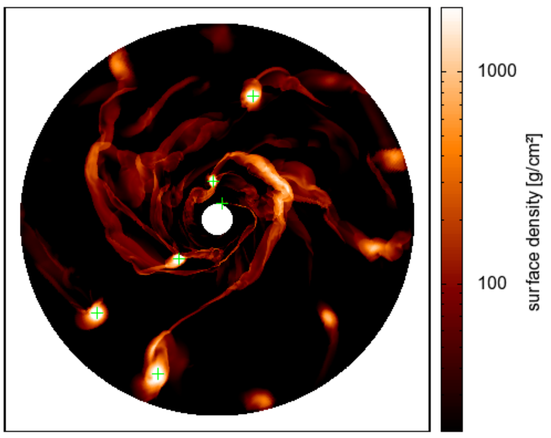

Figure 25: Local maxima of SimA15 after *slope_too_small*(), *hillsphere_too_small*() and *density_too_small*() have been applied.

As all remaining maxima are part of a actual fragment, we now can calculate the Hill sphere for this points. This could be of great use for simulations because, having a list of all grid points that belong to a Hill sphere, its mass can be determined and the corresponding density distribution can be replaced by a point mass at the position of the fragment's center of mass.

7.6 Calculating the Hill sphere

We first explain how to calculate the Hill sphere for one fragment. Afterwards we will discuss what has to be changed if there are more fragments that could possibly interfere with one another.

7.6.1 Hill sphere for one fragment

As exposed in Fig. 7.2, the Hill sphere denotes the area around a fragment within which particles are more attracted by the fragment than by the central star. If we consider the fragment to have been developed by gradually aggregating more mass, we can emulate this process to calculate the Hill sphere

H.

We first start with the initial fragment $F^0 = P$, i.e. the corresponding local maximum. Then we calculate the Roche radius of H^0 (which has already be done in *hillsphere_too_small*()) and consider all grid points that are completely inside the Roche radius to be part of the fragment F^1. We then calculate the centre of the mass CM^1 of the fragment. With this, we can calculate the Roche radius H^1 and consider all grid points inside H^1 to be part of the fragment F^2. As the density of the grid points decreases rapidly with increasing distance to P, the mass aggregated in the n-th step will decrease with increasing n so that the process will terminate quickly because the mass aggregated in the preceding step was not great enough to expand the Roche radius by one whole more grid point. As an illustration of the described process see Fig. 26.

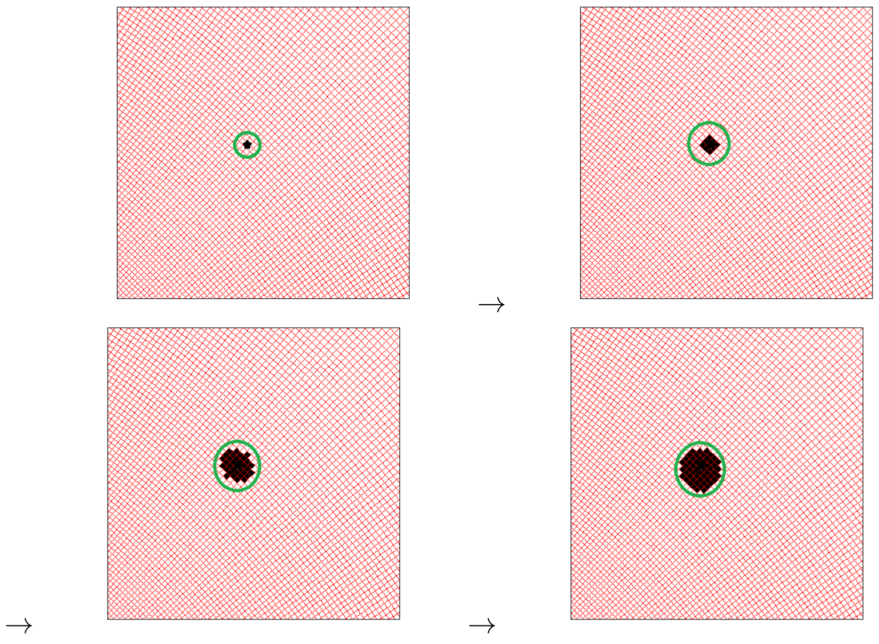

Figure 26: Illustration of the process of a fragment evolving by gradually aggregating more mass. The black grid points are part of F^i the green circle represents the Roche lobe H^i(which is not a circle as exposed in Fig 7.2). After step 4, no uncolored grid point is fully inside H^4.

7.6.2 Hill sphere for more fragments

If there is more than one fragment we have to be careful. If two fragments are close together, we cannot just calculate the Hill spheres one after another. A problem that could arise in this case is illustrated in Fig. 27: If we first calculate the Hill sphere of P_1 it can happen that grid points that are inside H^0 of P_2 are aggregated by the Hill sphere around P_1 and so, when the Hill sphere of P_2 is calculated, could not be aggregated by P_2 as a particle cannot be part of two fragments.

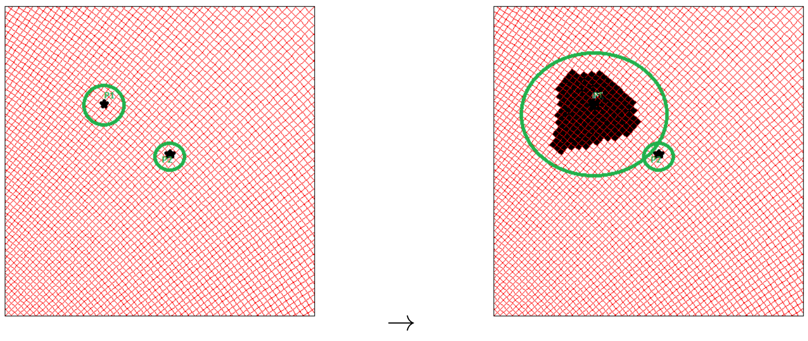

Figure 27: Illustration of the problem arising if the Hill spheres are calculated one after another.

To avoid such possible problems, we have to calculate all Hill spheres at once, i.e. for every local max P_i we calculate H_i^0, than F_i^0 and so forth. This is illustrated in Fig. 28. If at one point some grid points are inside two Roche radii (as illustrated in the right panel of Fig. 28) they are added to the fragment which exerts a stronger attraction on them. The process now is iterated until non of the fragments can aggregate more mass. Then, if n is the index of the last step, F_i^n contains all grid points that are completely inside the Hill sphere of fragment i.

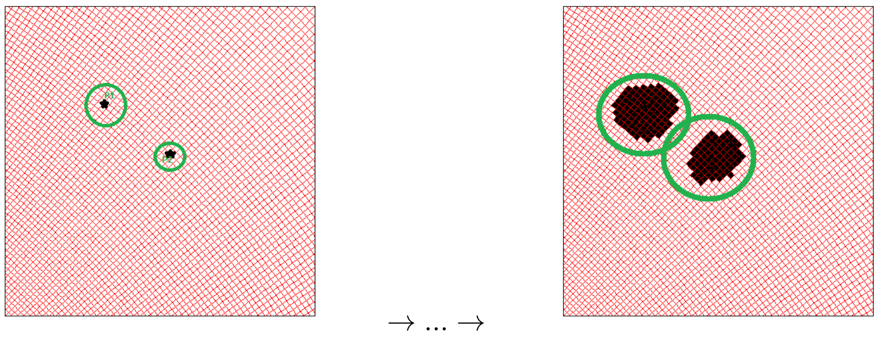

Figure 28: Illustration of the process of the fragments aggregating mass simultaneously.

Emulating this process of simultaneous aggregation of mass by the fragments leads to the following algorithm:

```
function calculate_Hill_spheres($localmax$)
    int j=0;
    repeat
        for all $P \in localmax$ do
            calculate_new_roche_radius_$H^j(P)$;
            calculate_new_fragment_$F^j(P)$;
            calculate_new_CM_$CM^j(P)$;
        od
        j++;
    until no_$P$_can_aggregate_more_mass();
    end
```

As a first test, after having executed *slope_too_small*(), *hillsphere_too_small*() and *density_too_small*(), we apply this algorithm to SimA15. The result can be seen in Fig. 29. Here, the crosses highlight the centers of mass of the respective fragments.

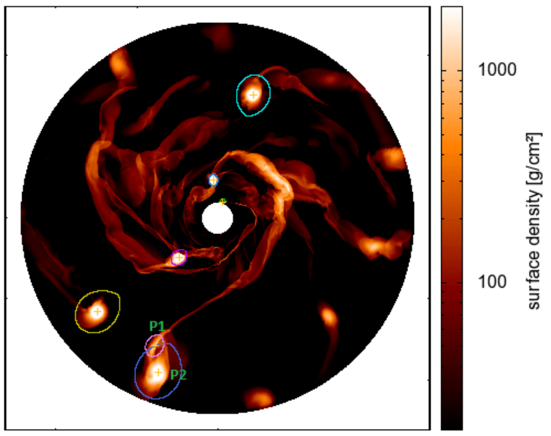

Figure 29: Hill spheres of SimA15 after *slope_too_small*(), *hillsphere_too_small*() and *density_too_small*() have been applied.

Obviously the Hill spheres of the local maxima P_1 and P_2 are overlapping. We can interpret this situation in such a way that in fact, these two very dense areas are one fragment that is evolving out of two overdense clumpse. If we do so, we can write another function that checks if some fragments are overlapping and, this being the case, merges the involved candidates:

```
function merge_fragments($localmax$)
    for all $P \in localmax$ do
        for all $P_j \neq P \in localmax$ do
            if overlapping($P_j$.hillsphere ,$P$.hillsphere)
                $P$.hillsphere.add($P_j$.hillsphere);
                $localmax$.remove($P_j$);
            fi
        od
    od
end
```

Applying *merge_fragments*(*localmax*) to SimA15, we observe, that the fragments around P_1 and P_2 have indeed been merged (see Fig. 30).

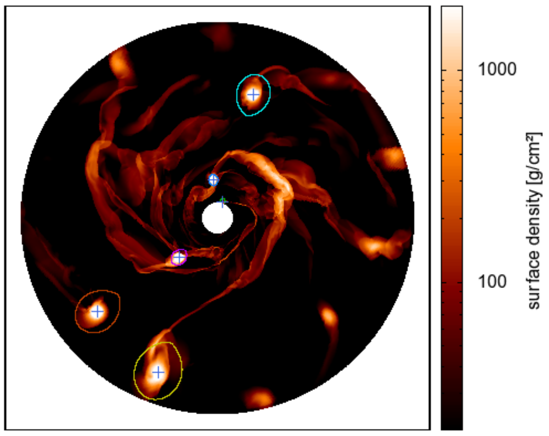

Figure 30: Hill spheres of SimA15 after overlapping fragments have been merged.

7.7 Fitting a Gaussian function

After we have calculated all Hill spheres and merged all overlapping candidates, we could, as another feature, fit a 2 dimensional Gaussian function to the density ditribution of a sphere.

$$g(\vec{r}) \approx \rho_0 e^{\left(-\frac{\|\vec{r}-\vec{r}_0\|^2}{2\sigma^2}\right)}$$

We choose r_0 to be the local max P corresponding to the sphere -$r_0 = P$ and $\rho_0 = \rho(P)$. We than choose σ such that the mass represented by the Gaussian equals the mass of the Hill sphere m_{hs}, i.e.

$$m_{hs} \overset{!}{=} \rho_0 \int_{\infty}^{\infty} e^{\left(-\frac{\|\vec{r}-\vec{r}_0\|^2}{2\sigma^2}\right)} dx = \sqrt{2\pi}\sigma$$

With that, we have $\sigma = \frac{m_{hs}}{\sqrt{2\pi}}$.

Instead of cutting out all the mass represented by the Hill sphere and replacing it by a point mass as mentioned at the end of section 7.5, we can cut out the mass represented by the value of the Gaussian $g(\vec{r})$, i.e. at every grid point (n_r, n_φ) that is part of the Hill sphere, we set

$$m_{hs} = m_{hs} - \min\left(m_{hs}, g(n_r, n_\varphi)\right).$$

Doing so would avoid perturbations of the simulation which would occur if suddenly the density of all grid points of a Hill sphere was set to zero.

7.8 Kepler orbits

In the last subsection we regarded the fragments as points of mass orbiting the central star, we can extend our algorithm to one last feature: Given the azimuthal and the radial velocities of the mass in all gridpoints and assuming the fragments to orbit the central star on Keplerian orbits, we first calculate the azimuthal an radial velocity of the corresponding centre of mass. Wee can now infer the fragments' orbital parameters ϵ and $\mathcal{P}$. With this, we can compute the orbits

$$r(\phi) = \frac{\mathcal{P}}{1+\epsilon\cos\phi}.$$

Plotting this along with the surface density ditribution (as we have examplarily done for SimE in Fig. 31) one can forecast the position of a fragment at a given later time of the simulation and compare this to the actual result of the simulation at this timestep.

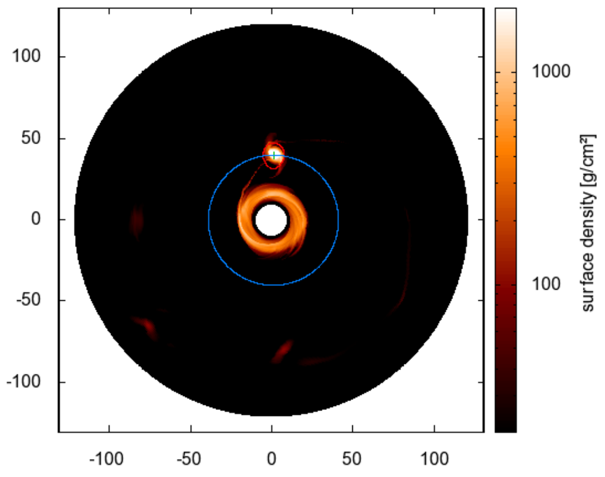

Figure 31: The Hillsphere around the only fragment (red) forecast Kepler orbit of the fragment (blue) of SimE.

Having presented the whole algorithm, in the next section we want to apply it to all the simulations considered in this work.

8 Application of the algorithm

We now apply the complete algorithm, i.e. we include all functions exposed in the last section, and test different factors *initial_roche_range*. To do so, we first apply the algorithm to the presented simulations with density distributions that can by judged by eye as well. In doing so, we can determine the best value for *initial_roche_range* and then apply the algorithm with this *initial_roche_range* to the distributions that cannot be checked for fragments without automatic help. We start with *initial_roche_range* $= 1$ and get the plots in Fig. 32. Then we choose *initial_roche_range* $= 3$ (see Fig. 33), *initial_roche_range* $= 5$ (see Fig. 34) and finally *initial_roche_range* $= 7$ (shown in Fig. 35).

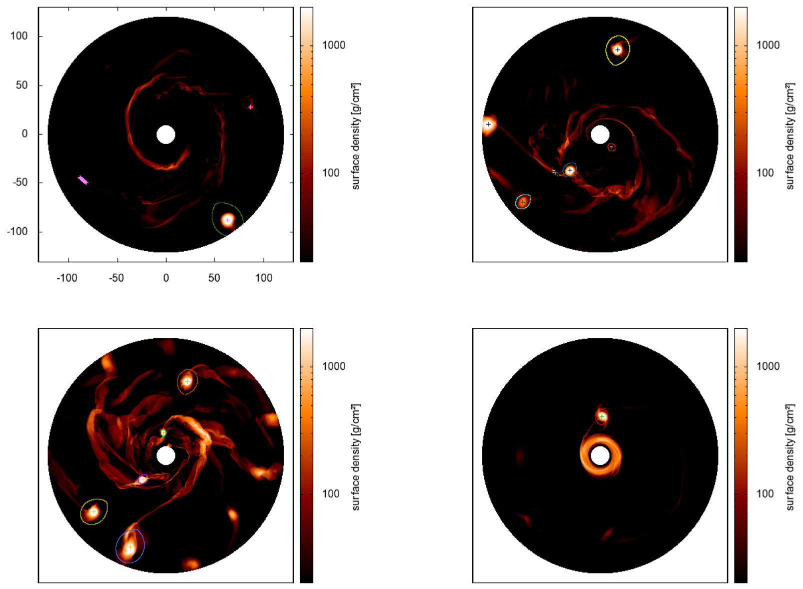

Figure 32: Plots for *initial_roche_range* = 1. Top left: SimA25. Top center: SimA20. Top right: SimA15. Bottom: SimE.

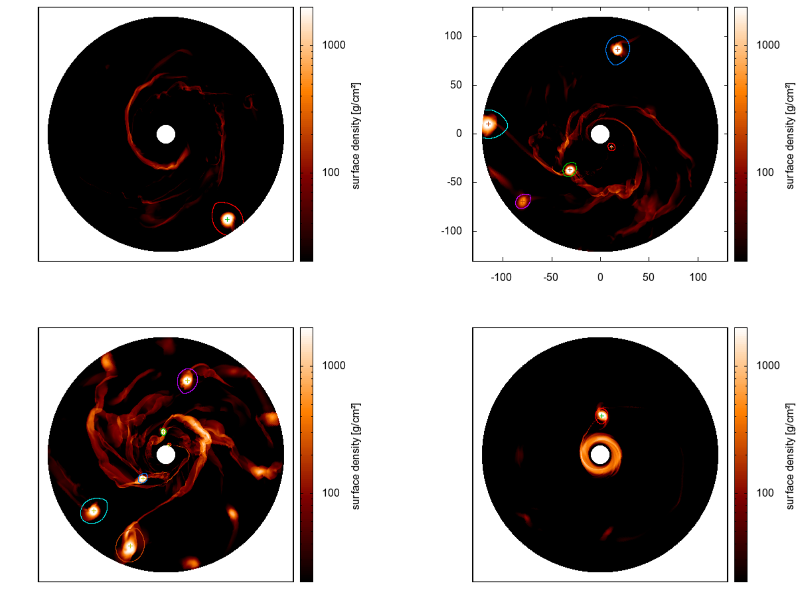

Figure 33: Plots for *initial_roche_range* = 3. Top left: SimA25. Top center: SimA20. Top right: SimA15. Bottom: SimE.

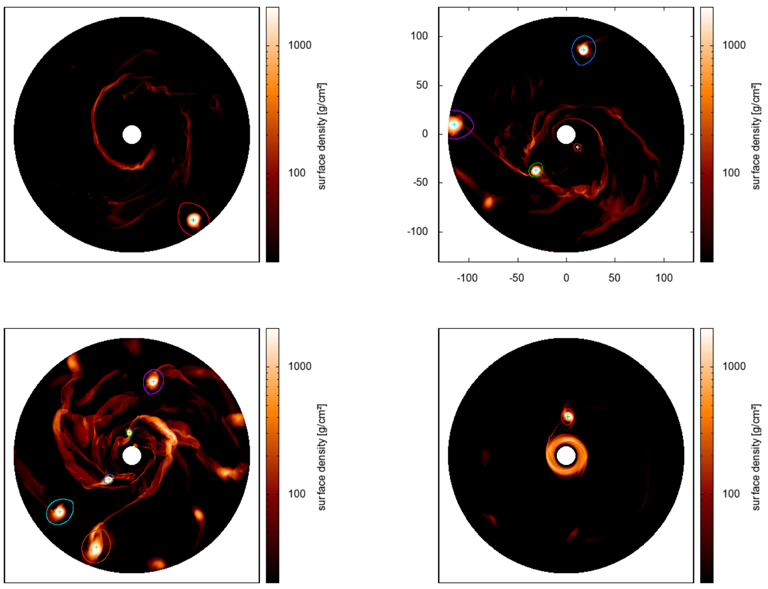

Figure 34: Plots for *initial_roche_range* = 5. Top left: SimA25. Top center: SimA20. Top right: SimA15. Bottom: SimE.

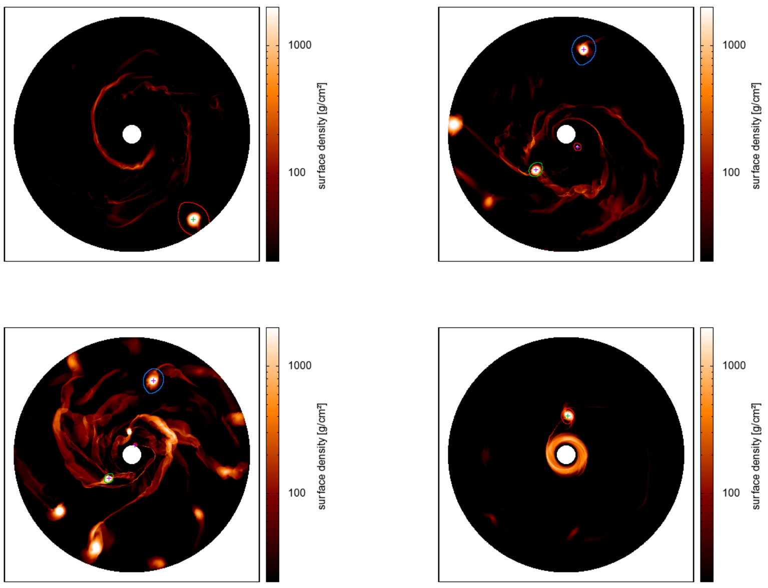

Figure 35: Plots for *initial_roche_range* = 7. Top left: SimA25. Top center: SimA20. Top right: SimA15. Bottom: SimE.

As we see, the plots with $initial\_roche\_range = 1$ contain too much maxima. choosing $initial\_roche\_range = 3$, SimA20 still contains one fragment that out of experience should not be there. $initial\_roche\_range = 7$ discards too much maxima but with $initial\_roche\_range = 5$, every significant dense clump is return as a fragment. Applying this value to SimC and SimD we can see that SimC indeed is fragmented whereas SimD is not (Fig. 36).

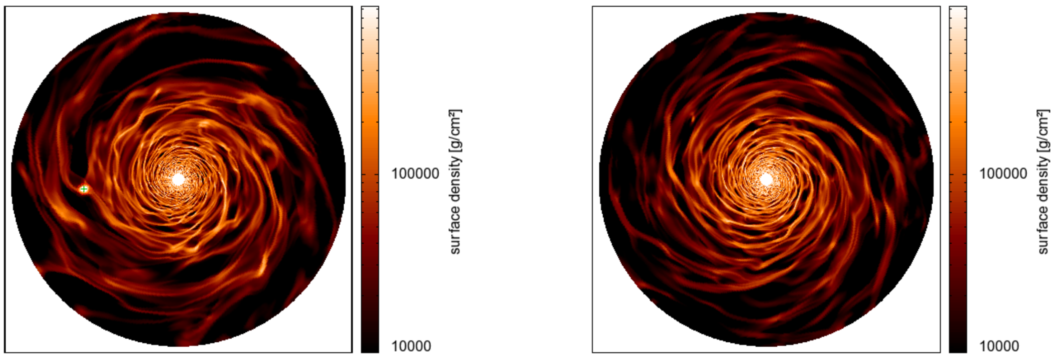

Figure 36: SimC (left) and SimD (right) for $initial\_roche\_range = 5$.

9 Summary

In this work we considered the problem of detecting areas of dense clumps within simulations of accretion disks. As we have demonstrated, neither edge detection algorithms of image processing nor simple algorithm which disregard the spacial distribution of surface density could satisfatorily be applied to this problem. But we were able to construct an algorithm that not only can assess with high accuracy whether a disk is fragmented or not, but that also can calculate the Hill spheres of the respective fragments. With this, simulations can cut out part of the mass of a fragment and replace it by a point mass that can easily be progagated in time. Likewise, a forecast of the fragment's Kepler orbit can be calculated.

We hope that the algorithm exposed in this work will held to ease the work of all the scientists that conduct simulations in the field of accretion disks.

References

Canny, J. 1986, IEEE Trans. Pattern Analysis and Machine Intelligence, 8, 679

Lindeberg, T. 2008, Encyclopedia of Computer Science and Engineering, 4, 2495

Masset, F. 2000, Astronomy and Astrophysics Supplement, 141, 165

Müller, T. 2010, Diploma thesis, Universität Tübingen

Toomre, A. 1964, The Astrophysical Journal, 139, 1217

Weidenschilling, S. J. 1977, Royal Astronomical Society, Monthly Notices, 180, 57

Ziou, D. & Tabbone, S. 1998, International Journal of Pattern Recognition and Image Analysis, 8, 537