Automatically detecting and tracking structures in protostellar outflows

DHRUVA KARKADA

Advised by Prof. Stella Offner

Dean's Scholars Honors Thesis, Spring 2021
Department of Astronomy
Department of Computer Science
The University of Texas at Austin

Abstract

Stars are one of the most fundamental astronomical objects to study. In particular, studying star formation yields insights into other astrophysical phenomena, such as how planetary systems develop and how the properties of galaxies change over time. However, star formation is complex. Simple models often fail to capture important details that affect measurable quantities such as the star's final mass.

One important phenomenon is the two jets of gas that are emitted in the polar directions of an accreting protostar. These jets, called protostellar outflows, carry mass away from the protostar and affect the kinematics of the surrounding gas. Furthermore, numerical simulations reveal that these outflows are typically not continuous streams, but rather discrete, fast-moving packets of gas. We would like to quantitatively understand how these bullets impact protostellar evolution.

In this work, I explain a novel algorithm I developed to detect and track bullets from star formation simulations. The presented approach solves a previously unaddressed problem in computer vision: unsupervised tracking of multiple fluid structures. Using the presented approach allows us to extract useful protostellar physics from numerical simulations.

Contents

Introduction **2**

Background **4**
- Magnetohydrodynamic Simulations 4
- Machine Learning in Astronomy 4
- Maximum Likelihood Estimation 6
- Optimization techniques 6
- Detection and Tracking 7

Approach **9**
- Segmentation via Dendrograms 9
- Tracking via Linking Matrices 11
- Deriving the Objective Function 13
- Initialization using Simulated Annealing 16
- Hyperparameter selection 17

Results and Future Work **18**

Acknowledgements **21**

Introduction

Many types of stars exist in the universe. Some are similar to our sun, many are more diminutive yet longer-lived, and few are massive powerhouse stars. These stellar properties directly correlate with important astrophysical phenomena, such as strong interstellar winds, astrochemistry, and the prevalence of planetary systems. Therefore, we would like to understand the mechanisms that control these stellar properties.

Unsurprisingly, most of these stellar properties are determined at the outset during the star formation process[1]. For example, a star's mass depends on how much available gas gravitationally collapses into a protostar. Or, the formation of planets depends on the presence of dust grains in the surrounding gas. This is why we're interested in understanding the mechanisms of star formation.

Roughly, the star formation process (Figure 1) proceeds as follows. Scattered throughout galaxies are giant clouds of cold gas called *molecular clouds*. They consist mainly of molecular hydrogen gas, some helium, and trace amounts of heavier elements. These molecular clouds are not perfectly uniform, and sometimes a region of relatively higher density exerts enough self-gravity to collapse inwards. As this collapse continues, the gas heats up, and eventually the central region becomes a dense protostar. The protostar continues to accrete gas until there is none left to accrete.

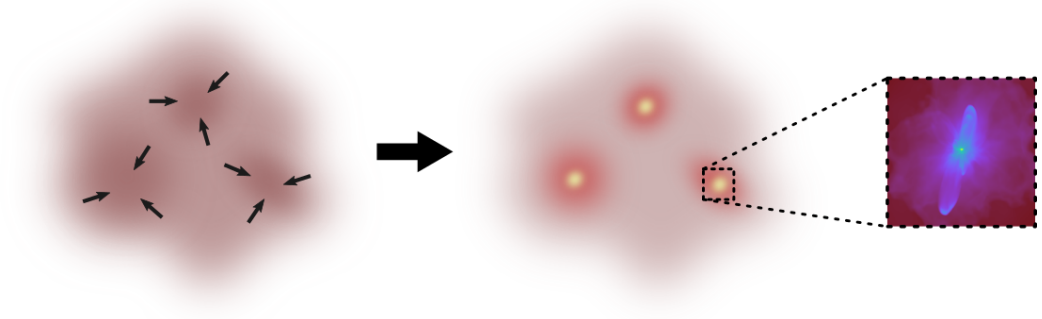

Figure 1: Dense regions of gas within a molecular cloud collapse under their own gravity and begin to form protostars. Each protostar accretes gas from its surroundings via an equatorial disk and may also eject gas via polar outflows.

A very simple model of star formation is the Jeans criterion for gravitational collapse. In this model, we only consider the gravitational instability of an isothermal, isotropic, homogeneous cloud of gas. Under these assumptions, one can derive the mass of the protostar[1]:

$$M_{\text{Jeans}} \propto \rho \left(\frac{kT}{Gm\rho} \right)^{3/2} \tag{1}$$

where T is the gas temperature, ρ is the gas density, m is the average particle mass, k is Boltzmann's constant, and G is the gravitational constant.

In reality, these assumptions are too strong. Real interstellar gas clouds are subject to highly nonlinear phenomena such as turbulence and magnetic fields[2,3]. Unfortunately, these nonlinearities immediately thwart analytical approaches, so we often rely on computational simulations to make headway.

Both numerical simulations and observations reveal much richer behavior than is suggested by the simple model discussed above. For example, real protostars accrete gas in an equatorial disk, and turbulent vortices can affect the kinematics of the infalling gas. One of the most prominent differences is the presence of polar outflows: jets of gas that are *ejected* at high speeds in the polar directions from the accreting region. Although outflows are frequently observed in both real data and simulations, simplistic models cannot explain them[3].

Furthermore, observations and numerical simulations reveal that these outflows are typically not continuous streams, but rather discrete, fast-moving packets of gas we call bullets[4–6]. These discrete bullets are caused by variable accretion rates; furthermore, they carry momentum and energy away from the protostar, so they significantly affect the kinematics of star formation[7]. This feedback phenomenon is not well understood; the goal of this work is to make progress on this front.

We would like to automatically detect and track the properties of protostellar bullets as a function of time. Unfortunately, unsupervised multi-object tracking remains a challenge in computer vision[8,9]. Existing tracking algorithms are reviewed in **Background**. However, our problem is further complicated by the bullets' ability to morph and coalesce over time (since they are fluid structures rather than solid bodies). Nonetheless, I applied principles from statistical inference and machine learning to develop an algorithm capable of distinguishing and tracking structure within a video of fluid matter. I explain this algorithm in detail in **Approach**. I report the algorithm's performance on test data in **Tests**. Finally, I use the algorithm to compute physical quantities of interest and analyze the time evolution of outflow bullets in **Results**.

Background

Magnetohydrodynamic Simulations

The timescale of star formation is long compared to human timescales. Observations indicate that molecular clouds have a lifespan on the order of 10^7 years[1]. This gives an estimate for the length of the star formation process. Since we are interested in dynamic quantities, i.e. the movement of bullets within the polar outflow, it's generically not possible to use real observational data to perform this analysis.

Thankfully, powerful computational tools exist to address this problem. Instead of observing a real protostellar system, we may instead simulate the system using magnetohydrodynamic (MHD) codes. MHD is effective because it can account for magnetic fields and turbulence, both of which make the problem of star formation analytically intractable. Using simulation, we can analyze the long-term astrophysics of relatively slow-evolving systems.

In this work, I analyze the protostellar simulations produced by Offner and Chaban[4]. These simulate a single dense core surrounded by a warm, turbulent medium embedded in an initially uniform magnetic field. The simulation encompasses a cube of side length $0.26\,\mathrm{pc}$ for a duration of about $5 \times 10^5\,\mathrm{yr}$. The simulation code also explicitly tags the high-velocity outflow gas as *tracer gas*, which allows us to distinguish the outflow from the environmental gas.

The MHD simulation produces 3D maps of gas density, which can be projected along the line of sight to produce 2D "column density" images (Figure 2). I preprocess the simulation outputs by producing logarithmically-scaled column density maps of only the tracer gas. Since the environmental gas serves as a sort of noise, focusing on the tracer greatly simplifies the problem.

Machine Learning in Astronomy

Rapid strides in computer science in the past decade have allowed machine learning (ML) techniques to become accessible to all scientists. Supervised ML techniques (such as neural networks and deep learning) have been especially successful in learning patterns from data. However, supervised ML approaches typically

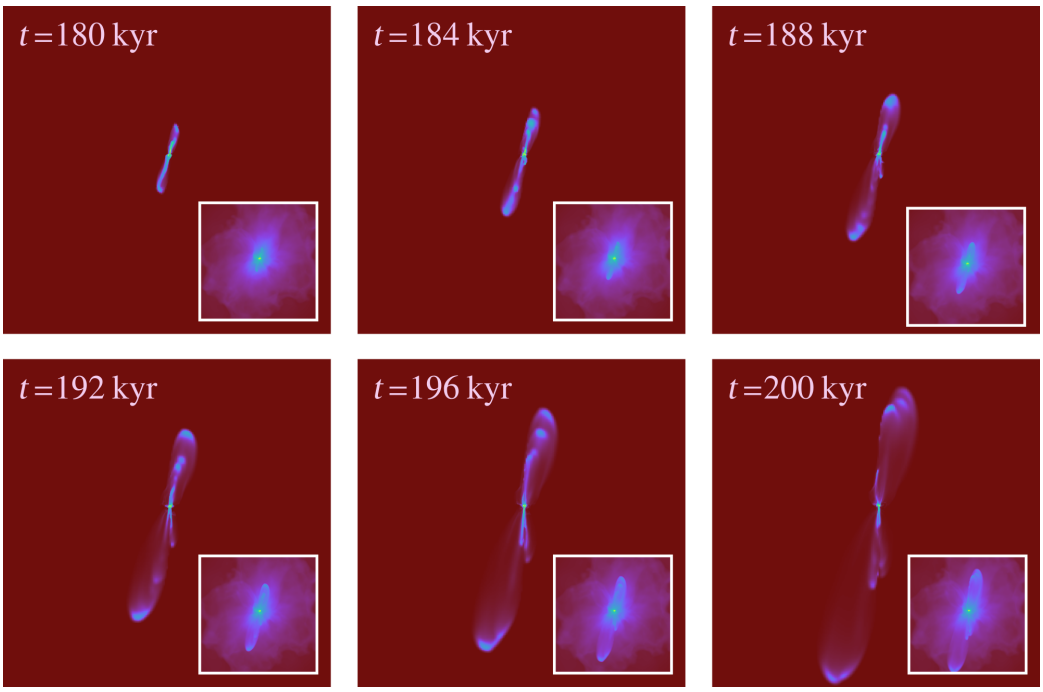

Figure 2: Column density of tracer gas at different times in protostellar evolution. The tracer gas consists of the outflow and any environmental gas which is carried away (entrained) by the outflow. Bullets (dense packets of outflow gas) are clearly visible. Insets: the corresponding column density of all gas (including the tracer).

require large, labelled data sets. In contrast, unsupervised learning aims to extract patterns from unlabelled data. In this sense, unsupervised learning is more flexible. However, this also means that unsupervised learning is dependent on heuristics or prior assumptions on the data.

Astronomy has greatly benefitted from these advances since observational astronomy is data-rich and supervised machine learning algorithms are very effective in extracting features from rich data sets[10,11]. However, since running MHD codes is computationally expensive even on supercomputing clusters, we don't have the quantity of data required for supervised learning. Furthermore, labelling the data for a fluid tracking problem introduces the possibility of human bias, since there are no concrete boundaries between different regions of gas. For these reasons, we worked to develop an unsupervised algorithm capable of detecting and tracking fluid structures in videos of protostellar outflows.

Maximum Likelihood Estimation

In machine learning, the dominant paradigm for solving an optimization problem is to frame it as an *objective function*: a real-valued function over the high-dimensional parameter space whose extremum represents the optimal solution[12]. The objective function depends on both the model parameters and the available data, but this dependence may be highly nonlinear. As a result, it's typically not possible to analytically solve for the optimal parameters. In addition, because there may be very many parameters, it's usually impractical to perform any sort of brute search (this is the well-known curse of dimensionality)[13]. Because of these two setbacks, we usually rely on optimization algorithms such as gradient descent and its variants[12].

One way construct a plausible objective function is to use a statistical technique called *maximum likelihood estimation* (MLE). In the MLE framework, we try to find the probabilistic model that is most likely to reproduce the data we see[14]. To do this, we formulate the probability of our model reproducing the observed data conditioned on the model parameters:

$$\Pr\left(\mathcal{D}|\vec{\theta}\right) = \prod_{i=0}^{N} \Pr\left(\mathcal{D}_i|\vec{\theta}\right) \tag{2}$$

where $\mathcal{D}$ is the observed data and $\vec{\theta}$ is the parameter vector. Notice that we've assumed that the data can be partitioned into N statistically independent parts, which allows us to express the conditional probability as a product over the partition.

It is sometimes difficult to optimize this product. Instead, we'll take the logarithm of this function, resulting in a new function which is monotonic with the original. So, minimum of the negative log-likelihood coincides with the maximum of the likelihood function. But the log-likelihood may be easier to optimize numerically, since the factors in the product turn into a sum of independent terms. We're interested in finding the $\vec{\theta}$ which minimizes the negative log-likelihood.

Optimization techniques

Given an objective function, how can we find an optimal solution (i.e. a global minimum)? If the objective function is convex, it suffices to initialize the parameters at any point and perform gradient descent (i.e. following the gradient of the

objective function down to its minimum). However, gradient descent relies on two fundamental assumptions: that the objective function is easily differentiable, and that the objective function is convex[12,15].

The former assumption is not difficult to address in practice. Software packages exist to automatically differentiate arbitrary objective functions. However, objective functions corresponding to interesting problems are rarely convex. In other words, these high-dimensional manifolds often contain many local minima[15]. In these cases, the result of gradient descent depends strongly on initialization. Specifically, gradient descent will only find a viable solution if the state is initialized within the corresponding "basin."

One technique to address this concern is a physics-inspired Monte Carlo method known as *simulated annealing*[16]. This method allows the parameter estimate to make discrete jumps in state space. The jump is accepted with probability 1 if the resulting cost is lower than the cost of the previous state. If the resulting cost is higher, the jump might still be accepted with probability $P = \exp(-\Delta E/T)$ where ΔE is the change in cost and T is the current "temperature" of the algorithm. Clearly, at high temperature, nearly all jumps are accepted, even for large cost increases. As $T \to 0$, large cost increases become less and less acceptable. In the annealing algorithm, we repeatedly allow the solution to jump while gradually lowering the temperature to zero. Because the solution may explore many different basins early in the annealing process, this algorithm typically succeeds at approximating a global minimum[12,16].

Detection and Tracking

Multiple object tracking (MOT) is a challenging problem in machine vision which has been addressed using a variety of techniques in recent years[8]. In this section, I will review some of the popular approaches.

In the machine learning literature, the tracking problem is usually divided into two sub-problems: detection and association[8]. Detection refers to the problem of identifying the relevant objects in each frame of the video. For example, if we are interested in tracking cars in a dashcam video, the detection stage would consist of first identifying all the cars in each frame of the video. In machine vision, the task of localizing objects in an image is called *segmentation*. In practice, segmentation is typically done by using a neural network which has been trained to look for the object in question. Note that to use a neural network for segmentation, one must have sufficient labelled data to train the network to find

the objects in question[9,17].

Association is then the problem of constructing trajectories by connecting the detected objects across frames. The main challenge is that different objects may have different speeds and directions. In special cases, optical flow methods may ameliorate this problem, but these methods typically require high frame rates[18]. If the frame rate is low compared to the average object velocity, it may be difficult to infer the correct trajectories. Another issue is occlusion: if an object passes behind another object, it will not be detected for some time, potentially confusing the tracking algorithm[19].

The dominant paradigm, *tracking-by-detection*, treats the two steps as separate and sequential. First the objects are detected, and then trajectories are constructed. This approach is convenient because one can use existing detectors and then perform tracking in the low-dimensional feature space. However, this approach suffers from one main drawback: since association is dependent on detection, errors in detection can cause the association to fail catastrophically[8,19].

Recent works have attempted to address the issue of dependence on detection by performing learning association jointly with detection[20]. In other words, the detection and trajectory construction are codependent and thus put on equal footing. However, it is still unclear how to extend these techniques to the unsupervised setting, where deep neural networks cannot serve as object detectors.

Furthermore, in the case of protostellar outflows, additional problems arise: the fluid structures may merge, bifurcate, and dramatically change morphology over the course of their trajectory. This introduces additional challenges to the association problem. Furthermore, since the 3D simulation data is very large (on the order of 1 gigabyte per frame), we cannot afford to save frames at a high frame rate. Therefore, optical flow methods are not feasible.

For these reasons, I decided to use the tracking-by-detection approach. I use a classical machine vision algorithm to detect structures in the outflows. Using this reduced-dimension data structure, I perform association using a continuous optimization scheme. The association algorithm only considers temporally local correlations; in other words, trajectories are constructed only looking at the previous frame. However, tracking is *not* real-time, which means that the trajectories at each frame can be optimized simultaneously. Both stages of the algorithm is described in detail in the next section.

Approach

Segmentation via Dendrograms

To capture the spatial structure of the outflows in each frame, I use an enhanced version of a data structure known as a *dendrogram*. Dendrograms are tree-like structures which are commonly used in the star formation literature to represent the structural relationships between dense regions of gas[11]. Because they are fundamentally trees, dendrograms are ideal for representing hierarchical, nested clumps of gas. I construct a dendrogram for each frame in the simulation and then perform tracking directly on the time series of dendrograms. Because of this, I use the term "frame" to refer to the corresponding dendrogram.

To understand how dendrograms are constructed, consider a 2D one-channel image representing the projected gas density of a protostellar region. Since this is a one-channel image, we can visualize the intensity as a topographic map, as shown in Figure 3. Then, by starting at each basin and working upwards, connecting branches at the basin boundaries, we can construct the equivalent dendrogram. The leaves of the dendrogram represent dense cores of gas, while internal branches represent the enveloping structures that connect the aforementioned cores. I'll use the term *branch* to collectively refer to both leaves and internal branches.

The dendrogram is more than just a tree: it also contains information about

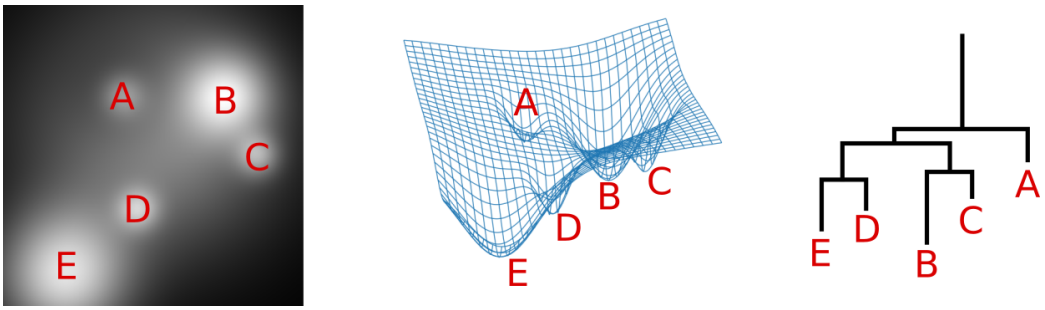

Figure 3: A 2D one-channel image, its topographic representation, and resulting dendrogram. The dendrogram has 9 branches, of which 5 are leaves (labelled for clarity).

the elevations at which basins meet, which captures more information about the spatial structure of the gas. But not all the spatial information is captured; we lose information about how the mass is distributed within each branch. In addition, the dendrogram only roughly captures the relative spatial organization of dense cores in the gas; the exact positions are lost. This more detailed information may be useful during the tracking stage of the algorithm, so in the segmentation stage I use, I also compute the following properties for each branch:

- Mass (m). The depth of the branch is insufficient and physically irrelevant. A better quantity to consider is the mass of the gas within the branch.

$$m = \sum_{r,c} m_{rc} \tag{3}$$

  where m_{rc} is the mass of the gas at pixel (r, c) belonging to the branch.

- Center of mass ($\vec{\mu}$). We want to know where in space this branch is located, not just its relative location to other branches.

$$\vec{\mu} = \frac{1}{m} \sum_{r,c} m_{rc} \vec{x}_{rc} \tag{4}$$

  where $\vec{x}_{rc}$ is the spatial coordinate of pixel (r, c).

- Covariance matrix ($\mathbf{S}$). If we approximate the gas in the branch as being spatially distributed as a 2D Gaussian, then the corresponding covariance matrix expresses the size and shape of the gas structure. In this view, $\vec{\mu}$ corresponds to the mean of the Gaussian approximation.

$$\mathbf{S} = \frac{1}{m} \sum_{r,c} m_{rc} (\vec{x}_{rc} - \vec{\mu})(\vec{x}_{rc} - \vec{\mu})^T \tag{5}$$

To construct the dendrogram, I implemented a recursive variant of Meyer's watershed algorithm[21]. This 1993 algorithm was proposed to perform "morphological image segmentation," which refers to image segmentation of a topographic map. Conceptually, this algorithm involves finding a local minimum of the topographic map (i.e. a dense core of gas) and then "flooding" the basin until it overflows into a neighboring basin. Once another basin has been identified, we can repeat the flooding procedure and join the two branches. By performing this recursively, we construct a dendrogram bottom-up, starting from the leaves

and ending at the root. Conveniently, during the algorithmic flooding procedure, we can compute the quantities of interest mentioned above.

Viewing dendrograms in the lens of the outflow problem, it's tempting to claim that dendrogram leaves correspond to the bullets we're interested in. However, this is not strictly true. When two bullets are physically nearby in space, the two leaves may in fact contain little mass; instead, the branch that joins the two leaves contains much of the mass. This effect is important when bullets merge or there is occlusion. For this reason, it's not sufficient to perform tracking on the leaves alone. The tracking algorithm must account for the entire dendrogram.

The dendrogram provides a view into how mass is spatially organized in the outflow. The input images are high-dimensional ($d \approx 10^4$), since each pixel corresponds to a dimension in the vector space of possible images. In contrast, dendrograms have significantly lower dimension: each dendrogram has on the order of 10^1 branches, each with a handful of features we have computed. In this view, dendrogram construction is a form a feature extraction. However, we have paid a small price: the data is slightly less structured than before. Whereas each image is always a square grid of pixel values, dendrograms can take any shape. The trees need not even be binary. This introduces challenges in running efficient optimization algorithms for the tracking stage.

Tracking via Linking Matrices

Broadly speaking, we are interested in tracking how protostellar outflow mass moves over time. After segmentation, the mass is organized in a dendrogram, so the tracking stage must associate branches from time t with the branches at time $t + 1$.

Previous work has shown that it is possible to construct a cost function whose minimum corresponds to the true associations of the object detections between frames[8]. However, there are two fundamental issues with this approach. One is that object association is a combinatorial problem. Although combinatorial optimization techniques exist, they are typically not as robust as continuous techniques such as gradient descent[22]. Secondly, the fluid structures are not self-contained. Mass can flow from one to multiple branches, so object association may not capture the underlying dynamics of the fluid structures over time.

To address both problems simultaneously, I relax the assumption that all fluid in a branch in frame t will flow into a single branch in frame $t+1$. Instead, I use a framework in which mass can flow from one branch into multiple branches (and

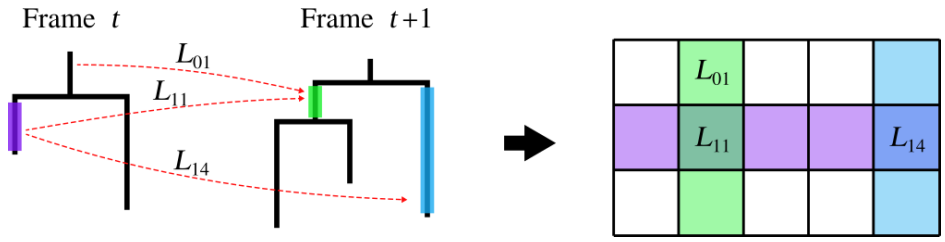

Figure 4: Linking matrices (right) describe how mass flows between branches in consecutive frames (left). The matrix elements L_{ij} specify the proportion of mass from branch i (at time t) that flows into branch j (at time $t+1$). Three such elements are explicitly labelled here.

conversely, a branch can "accept" mass from multiple branches in the previous frame). In graph theoretic terms, I arrange the branches in frames t and $t+1$ into a complete bipartite graph, where the edge weights describe mass flow from the branches of dendrogram t into the branches of dendrogram $t+1$. The nonzero quadrant of the adjacency matrix then fully encapsulates how the mass flows; I call this data structure the *linking matrix* since it describes how dendrogram t is linked to dendrogram $t+1$.

More concretely, at time t, the linking matrix $L_{(ij)}$ is an $n \times m$ matrix, where n is the number of branches in dendrogram t and m is the number of branches at $t+1$. The matrix elements L_{ij} specify the fraction of mass from branch i (at time t) that flows into branch j (at time $t+1$). So, the L_{ij} must satisfy $0 \leq L_{ij} \leq 1$ and $\sum_j L_{ij} = 1$ for all i. In other words, each row specifies a probability mass function. In practice, these constraints can be enforced by reparametrizing using the softmax function. The linking matrix is schematically explained in Figure 4.

It's useful here to define some notational conventions I'll use hereforth. I'll use the indices i and j to refer to branches from frame t and $t+1$ respectively, in the context of a linking matrix $L^{(t)}$. I'll usually drop the (t) superscript, since the method is symmetric with respect to time translation (i.e. no part depends explicitly on t). I'll also define

$$m_{ij} = m_i L_{ij} \tag{6}$$

to be the amount of mass from branch i that flows into branch j. This shorthand will be used extensively in the next section.

To summarize, the set of linking matrices $\{L^{(t)}\}$ fully specifies how mass from each dendrogram is distributed to the branches of the next dendrogram. Each row of a linking matrix describes how the mass in a given dendrogram branch is distributed among the branches in the next frame. Conversely, each

column of a linking matrix indirectly describes how the mass in a given branch of the next frame receives contributions from branches in the current frame.

In this scheme, the matrix elements of the linking matrices are the optimization parameters, and since they are real-valued parameters, we can construct a cost function and perform gradient descent. This way, we sidestep the challenges posed by combinatorial optimization while simultaneously allowing the flexibility required for tracking fluid structures.

Deriving the Objective Function

In many standard machine learning problems, there are best practices for choosing objective functions. For example, regression problems typically utilize mean-squared-error as an objective function, whereas classifiers usually use cross-entropy loss. However, for unsupervised problems such as the present tracking problem, there is far more flexibility in choosing an objective function. This is both a strength and a weakness: the flexibility allows us to incorporate heuristics, but it may be difficult to define an objective function which allows for fast convergence.

Because of this, it's important to take a principled approach in constructing the objective function. I used maximum likelihood estimation (MLE) to derive the objective function. In this section, I'll use physics to motivate different factors in the likelihood function. Each factor contributes a term in the objective function (which is the negative log likelihood).

Conservation of mass. The total mass from frame t flowing into each branch in frame $t+1$ should match the branch's true mass. Let's define the fractional mass increase of branch j as

$$\tilde{m}_j = \frac{m_j}{\sum_i m_{ij}} \tag{7}$$

where the numerator is the true mass of branch j and the denominator is the total mass contribution from branches in the previous frame. We define the relative error as

$$\chi_j = \frac{(\tilde{m}_j - 1)^2}{\tilde{m}_j}. \tag{8}$$

This functional form has the following nice properties: a) $\chi_j \geq 0$ with equality only when $\tilde{m}_j = 1$ (i.e. the linking matrix exactly conserves mass), and b) it is symmetric with respect to the reciprocal of the argument ($\tilde{m}_j \leftarrow 1/\tilde{m}_j$). This

reciprocal symmetry reflects time-reversal symmetry, since $\tilde{m}_j$ represents the fractional increase in mass over time.

With these definitions, we can assign a Gaussian distribution to the probability of seeing the data $\mathcal{D}$ (i.e. the dendrograms at t and $t+1$) given a linking matrix $L_{(ij)}$:

$$\Pr\Big(\mathcal{D}|L_{(ij)}\Big) \propto \prod_j \exp\left(-\frac{\chi_j^2}{2\sigma_m^2}\right) \tag{9}$$

where σ_m is a hyperparameter which describes the sensitivity to large deviations from mass conservation. I found that $\sigma_m = .01$ works well.

Locality and Morphology. The spatial distribution of gas in any branch in frame $t+1$ (i.e. its location and shape) should roughly match those of the branches that contribute to it from frame t. First, I'll define the contribution factor $c_{ij} = m_{ij}/\sum_i m_{ij}$ which is the fraction of total mass flowing into branch j that comes from branch i.

Then, we define the inflow center-of-mass as

$$\vec{\mu}_{\text{in},j} = \sum_i c_{ij}\vec{\mu}_i \tag{10}$$

and the inflow covariance matrix as

$$\mathbf{S}_{\text{in},j} = \sum_i c_{ij}\left(\mathbf{S}_i + (\vec{\mu}_i - \vec{\mu}_{\text{in},j})(\vec{\mu}_i - \vec{\mu}_{\text{in},j})^T\right). \tag{11}$$

These formulae follow from the mean and covariance matrix of a Gaussian mixture. Recall that the covariance matrix here describes the spatial distribution of gas in a branch by approximating the gas as being normally distributed in space. In other words, we're modelling the inflow gas as a mixture of Gaussian branches (weighted by c_{ij}) and computing the overall mean and covariance matrix of the mixture. Then, we'll compare this with the mean and covariance matrix of the final branch j:

$$\Pr\Big(\mathcal{D}|L_{(ij)}\Big) \propto \prod_j \exp\left(-\frac{\|\vec{\mu}_j - \vec{\mu}_{\text{in},j}\|^2}{2\sigma_\mu^2}\right) \tag{12}$$

and

$$\Pr\Big(\mathcal{D}|L_{(ij)}\Big) \propto \prod_j \exp\left(-\frac{\|\mathbf{S}_j - \mathbf{S}_{\text{in},j}\|^2}{2\sigma_{\mathbf{S}}^2}\right) \tag{13}$$

so that large mismatches in location and morphology are exponentially unlikely. I use the hyperparameters $\sigma_\mu = 10$ and $\sigma_{\mathbf{S}} = 1$.

Structure. The linking matrix should preserve the hierarchical structure between dendrograms t and $t+1$. Specifically, if two branches i_1 and i_2 have a parent-child relation (the closest relation in dendrogram space), their mass should flow into branches $\{j_n\}$ that are also nearby in dendrogram space.

Here, I use the term *dendrogram space* to refer to the metric space that can be constructed on any tree: the metric $g(i_1, i_2)$ is the length of the shortest path between vertex i_1 and i_2. This metric space describes whether two branches are "hierarchically closeby." I claim that linking matrices that don't preserve this structure are physically unlikely.

We'll consider two branches i_1 and i_2 which have a parent-child relation in dendrogram t. The linking matrix defines how the mass from i_1 and i_2 is distributed over the branches $\{j_n\}$ in dendrogram $t+1$. Let's define this distribution as

$$P_{i_1,i_2}(j) = \frac{m_{i_1 j} + m_{i_2 j}}{m_{i_1} + m_{i_2}} \tag{14}$$

or in other words, the fraction of mass from branches i_1 and i_2 that flow into branch j. This is a probability distribution over the branches of dendrogram $t+1$.

Although this distribution isn't a distribution in a Euclidean space, it's a distribution on a metric space, so we can still define the variance of the distribution using the Frechet variance[23]:

$$V_F \equiv \frac{1}{2}\mathbb{E}\left[g(X_1, X_2)^2\right] \tag{15}$$

which in this case is

$$V_F(i_1, i_2) = \frac{1}{2}\sum_{j_1,j_2} g(j_1, j_2)^2 P_{i_1,i_2}(j_1) P_{i_1,i_2}(j_2). \tag{16}$$

Note that the initial variance of the two branches i_1 and i_2 in dendrogram t is

$$V_{F,\text{in}}(i_1, i_2) = \frac{m_{i_1} m_{i_2}}{(m_{i_1} + m_{i_2})^2}. \tag{17}$$

Physically realistic linking matrices preserve dendrogram structure, so for any dendrogram-local pairs of branches from frame t, their mass distribution over branches in frame $t+1$ should match their initial variance. If the distribution instead has a high variance, then that means that the mass from branches i_1 and i_2 diffuses throughout dendrogram $t+1$. This violates locality and implies

that hierarchical structure isn't preserved by the linking matrix. I model this constraint with

$$\Pr\Big(\mathcal{D}|L_{(ij)}\Big) \propto \prod_{i_1,i_2} \exp\left(-\frac{\Big(V_F(i_1,i_2) - V_{F,\text{in}}(i_1,i_2)\Big)^2}{2\sigma_V^2}\right) \tag{18}$$

where the product is only over pairs of branches i_1 and i_2 that have a parent-child relationship. This selects only the branch pairs that are dendrogram-local. I set the hyperparameter $\sigma_V = 0.3$.

These considerations (mass conservation, locality, morphology, and hierarchical structure) each contribute the total conditional probability $\Pr\Big(\mathcal{D}|L_{(ij)}\Big)$, which is proportional to the product of equations 9, 12, 13, and 18. By Bayes' theorem,

$$\Pr\Big(L_{(ij)}|\mathcal{D}\Big) \propto \Pr\Big(\mathcal{D}|L_{(ij)}\Big)\Pr\Big(L_{(ij)}\Big). \tag{19}$$

Since I assert no prior about the likelihood of linking matrices, $\Pr\Big(L_{(ij)}\Big)$ assumes the maximum-entropy distribution, which is the uniform distribution. Therefore, we can directly take the negative log-likelihood of $\Pr\Big(\mathcal{D}|L_{(ij)}\Big)$, obtaining a sum of terms which we can attempt to optimize using gradient descent. This will give us a maximum likelihood estimation for the model parameters $L_{(ij)}$.

Initialization using Simulated Annealing

The objective function detailed above is not convex. As a result, naive initialization schemes often yielded poor results after gradient descent, since local optimization cannot guarantee global convergence without good initial parameters. Following the strategy utilized by other tracking algorithms[8,22], I use an existing optimization algorithm to initialize the parameters.

Specifically, I used simulated annealing on a simpler problem with fewer degrees of freedom, then used the result of the annealing procedure to initialize the parameters of the linking matrices. I considered the problem where each dendrogram branch at time t is associated with a single dendrogram branch at time $t+1$. In terms of the linking matrix, this means that each row is an indicator function. This is clearly a special case of the generic linking problem described above, where each row of a linking matrix specifies an arbitrary probability mass function. Since the restricted problem has a smaller state space than the general

problem, it's feasible to try a global optimization method (i.e. simulated annealing). In addition, annealing conveniently handles the discreteness of the smaller state space.

The annealer optimizes the function

$$C(s) = \sum_{t=1}^{t_{\max}} \sum_{j \in d_t} \left\| \vec{\mu}_{\text{in},j} - \vec{\mu}_j \right\|^2 + \beta \chi_j \tag{20}$$

where $\vec{\mu}_{\text{in},j}$ is the inflow center-of-mass defined in equation 10, $\vec{\mu}_j$ is the center-of-mass of branch j, and χ_j is the relative mass error defined in equation 8. I set the hyperparameter $\beta = 100$.

Hyperparameter selection

The hyperparameters that describe the relative importance of terms in the likelihood function were determined manually by examining the units and scale of the quantities being measured in each likelihood factor. The hyperparameters are summarized in the table below.

Hyperparameter	**Justification**
$\sigma_m = .01$	We would like to strongly penalize deviations from $\chi_j = 1$. Therefore, our likelihood function should have a small error interval (standard deviation) around the mean.
$\sigma_\mu = 10$	Since the distances are measured in pixels, a location error of 10 pixels is a reasonable standard deviation.
$\sigma_{\mathbf{S}} = 1$	The L_2-distance between two 2×2 covariance matrices of the bullets we see is on the order of 10, so an error interval of 1 is reasonable.
$\sigma_V = 0.3$	Because the dendrograms have depth on the order of 10, the Frechet variance should be on the order of 1. So I chose an error interval of comparable magnitude.
$\beta = 100$	Since we want to strongly penalize $\chi_j \neq 1$, and the scale of the location term is being measured in pixels, we boost the mass term in the annealing function.

Results and Future Work

The tracking algorithm is designed to work on 2D videos of protostellar evolution. However, the simulation outputs are 3D, so I first compute the column density along a line of sight to produce 2D projection images. This produces a video containing the dynamics of all protostellar gas. I also produce a video that only shows the tracer gas (recall that the tracer gas is the outflow gas along with any gas that is picked up by the outflow).

Because the data is not labelled, there are no direct metrics for evaluating the accuracy of the algorithm. However, we can examine the quality of the algorithmic tracking results by eye to understand its strengths and weaknesses. To do this, we first identify the structures that we classify as bullets, and then use the linking matrices to extract feasible trajectories for these bullets.

First, let's define the total mass $m_{i,\mathrm{tot}}$ of a dendrogram branch to be the mass enclosed by that branch and all its dendrogram descendants. Then I define the mass fraction of a bullet to be $f_i = m_i/m_{i,\mathrm{tot}}$ (such that $f_i \in (0, 1]$). Finally, I define a bullet as any branch with $f_i > 0.8$. (This threshold was determined by inspection, although the results are not sensitive to this choice.) If a bullet's dendrogram parent is also a bullet, then they are grouped as part of the same bullet. This definition segments the dendrogram into bullets (which consist of branches near the leaves) and inter-bullet outflow gas (the upper branches in the dendrogram).

In Figure 5, we see that bullets are identified and tracked faithfully. Large bullets are especially easy to identify and track. The tracking algorithm is able to handle bullet merger events due to the robustness of the linking matrices. However, visually verifying the tracking accuracy can be sensitive to the way in which we define bullets and extract trajectories from the linking matrix. In the results shown, we use a simple scheme where we define bullets by a threshold (as described above) and assign trajectories to bullets using a modified stable-marriage algorithm (where the linking matrix encodes the "preferences" of bullets to be assigned to the same trajectory). However, a more sophisticated scheme may be able to better capture the time evolution of bullets.

Despite these promising results, the algorithm has weaknesses which can cause tracking failure. For example, the algorithm sometimes struggles to track nascent bullets that are close to the inner region (roughly within $1 \times 10^4\,\mathrm{AU}$ of

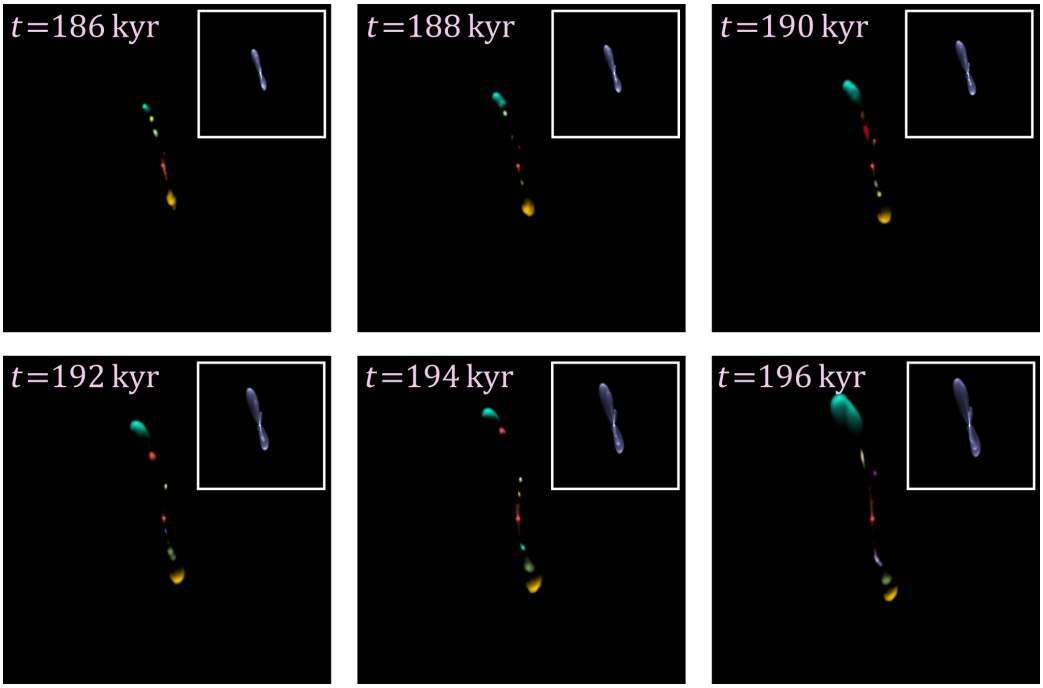

Figure 5: Over 6 frames, bullets are faithfully tracked, as shown by their colors. Merging events retain the color of the larger-mass bullet, which is seen in frames 2 and 6 (teal) and frame 4 (moss green). Note: mergers increase the spatial extent of the bullet since the merged bullet includes both original bullets and the branch that joined them. Insets: the original outflows.

the protostar). This may be because the protostar is am outflow mass source, so mass conservation does not necessarily apply near the protostar. In addition, the algorithm also does not account for boundary conditions, so it struggles to track bullets that are leaving the frame.

Several possible improvements may address this issues. First, the likelihood function includes no motion model. This is because motion is difficult to model when we are considering clusters of gas which may diffuse and split between frames. Simple motion models, such as linear velocity fields[8], need to be modified to account for the diffusion of fluid.

Secondly, it's not clear how logarithmic mass affects the mass conservation term. Recall that in the simulation output frames, the pixel intensity represents column density on a logarithmic scale. This is to account for the large dynamic range of outflow structures; indeed, dendrograms constructed from linear-scale images of column density fail to capture important features. However, we only expect physical mass to be conserved, not logarithmic mass. Further investiga-

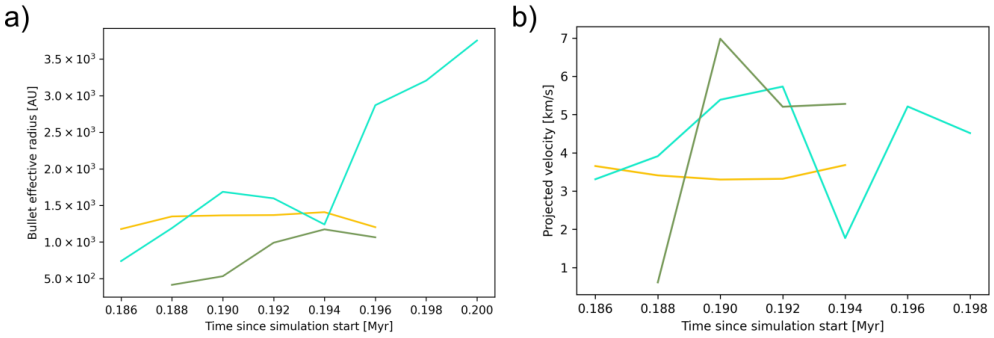

Figure 6: Time evolution plots for a few select bullets. The colors correspond to the colors shown in Figure 5. a) The cross-sectional area of bullets over time. Initially, bullets increase in size; some bullets seem to decrease in size later, but this is simply an artifact of the merging process (the dendrogram leaf is getting smaller since it's starting to overlap with another leaf). The teal curve shows that the size recovers after the merger is complete. b) The apparent velocity of the bullets, determined by finite difference between frames. The inconsistencies in velocity is likely due to merger events.

tions are required to elucidate the role that logarithmic mass plays.

From the bullet trajectories, we can compute the time evolution of bullet properties (Figure 6). These preliminary results indicate that bullets grow over time, as expected. During some periods, the bullet size seems to decrease, but this is an artifact of the merging process. This is because, as two bullets merge, their dendrogram leaves become smaller and their parent branch grows larger. We also see that the bullet velocities are on the order of $10\,\mathrm{km\,s^{-1}}$, but are distinguishably different from bullet to bullet. However, the velocities are computed via finite difference between the bullets' centers of mass; in the future, we intend to use the momentum and energy (which are encoded in the simulation outputs) to directly probe the kinematics of the bullets over time.

In the future, we would like to incorporate a motion model to improve accuracy and robustness. We would also like to extend the algorithm to directly operate on the 3D simulation outputs. We expect that dendrogram construction will be computationally more expensive; however, this will allow us to avoid the challenges associated with occlusion. In addition, by tracking on the simulation outputs, we can directly extract the mass evolution and kinematics of the bullets from the simulation data, rather than inferring them from projective images. Finally, we hope to apply this algorithm to simulations of larger star-forming regions to understand the kinematics of dense cores of gas in a stellar nursery.

Acknowledgements

I'd like to thank Prof. Stella Offner for her patience and guidance over the years. I would not be where I am today without her mentorship. I'd also like to thank the Dean's Scholars program for the opportunities and community they've provided me over the course of my undergraduate education. The memories I've made through DS will stay with me for life. In addition, I can't understate how much I appreciate all my friends for their support over the years. I'd like to specifically thank Karthik for being an extremely humorous and supportive roommate. Finally, I'd like to thank my family for their unconditional love and encouragement.

References

[1]D. Ward-Thompson and A. P. Whitworth, *An introduction to star formation* (Cambridge University Press, 2011).

[2]P. Girichidis, S. S. Offner, A. G. Kritsuk, R. S. Klessen, P. Hennebelle, J. D. Kruijssen, M. G. Krause, S. C. Glover, and M. Padovani, "Physical processes in star formation", Space Science Reviews **216**, 1–67 (2020).

[3]A. L. Rosen, S. S. Offner, S. I. Sadavoy, A. Bhandare, E. Vázquez-Semadeni, and A. Ginsburg, "Zooming in on individual star formation: low-and high-mass stars", Space Science Reviews **216**, 1–42 (2020).

[4]S. S. Offner and J. Chaban, "Impact of protostellar outflows on turbulence and star formation efficiency in magnetized dense cores", The Astrophysical Journal **847**, 104 (2017).

[5]S. S. Offner and H. G. Arce, "Investigations of protostellar outflow launching and gas entrainment: hydrodynamic simulations and molecular emission", The Astrophysical Journal **784**, 61 (2014).

[6]Y. Zhang, H. G. Arce, D. Mardones, S. Cabrit, M. M. Dunham, G. Garay, A. Noriega-Crespo, S. S. Offner, A. C. Raga, and S. A. Corder, "An episodic wide-angle outflow in hh 46/47", The Astrophysical Journal **883**, 1 (2019).

[7]J. Bally, "Protostellar outflows", Annual Review of Astronomy and Astrophysics **54**, 491–528 (2016).

[8]W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, "Multiple object tracking: a literature review", Artificial Intelligence, 103448 (2020).

[9]G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, "Deep learning in video multi-object tracking: a survey", Neurocomputing **381**, 61–88 (2020).

[10]C. J. Fluke and C. Jacobs, "Surveying the reach and maturity of machine learning and artificial intelligence in astronomy", Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **10**, e1349 (2020).

[11]D. Baron, "Machine learning in astronomy: a practical overview", arXiv preprint arXiv:1904.07248 (2019).

[12]A. Törn and A. Zilinskas, "Global optimization", (1989).

[13]C. C. Aggarwal, *Data mining: the textbook* (Springer, 2015).

[14]I. J. Myung, "Tutorial on maximum likelihood estimation", Journal of mathematical Psychology **47**, 90–100 (2003).

[15]J. Nocedal and S. Wright, *Numerical optimization* (Springer Science & Business Media, 2006).

[16]S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing", science **220**, 671–680 (1983).

[17]Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", nature **521**, 436–444 (2015).

[18]M. J. Black and P. Anandan, "A framework for the robust estimation of optical flow", in 1993 (4th) international conference on computer vision (IEEE, 1993), pp. 231–236.

[19]Y. Xiang, A. Alahi, and S. Savarese, "Learning to track: online multi-object tracking by decision making", in Proceedings of the ieee international conference on computer vision (2015), pp. 4705–4713.

[20]Z. Lu, V. Rathod, R. Votel, and J. Huang, "Retinatrack: online single stage joint detection and tracking", in Proceedings of the ieee/cvf conference on computer vision and pattern recognition (2020), pp. 14668–14678.

[21]S. Beucher and F. Meyer, "The morphological approach to segmentation: the watershed transformation", Optical Engineering **34**, 433–433 (1992).

[22]A. Milan, S. Roth, and K. Schindler, "Continuous energy minimization for multitarget tracking", IEEE transactions on pattern analysis and machine intelligence **36**, 58–72 (2013).

[23]P. Dubey and H.-G. Müller, "Fréchet analysis of variance for random objects", Biometrika **106**, 803–821 (2019).