# Markov Time Sweeping and Personalized Community Detection

Ian Wood, Project Advisor: Yong-Yeol Ahn, Ph.D.

February 12, 2015

## 1 Abstract

Networks provide a useful framework to describe and investigate many complex systems, by providing an abstraction whereby a system's components and the relations between them are represented as nodes and links. To aid in the interpretation and use of these abstractions, community analysis aims at identifying meaningful modules that capture meso-scale organization of complex systems. Infomap and Modularity optimization are two common methods for community detection. Modularity finds communities defined by dense clusters of nodes, while Infomap finds communities by compressing descriptions of flow on the network. Both methods rely on an underlying discrete-time random walk process that implicitly assumes a fixed scale, resulting in limitations, including limits to their resolution and field-of-view. In this paper, I review the prior work on community detection algorithms, particularly Infomap, and on generalizing these methods to continuous-time Markov processes. I discuss how time sweeping allows community detection on multiple scales. Then I extend the continuous-time Infomap algorithm to selectively zoom into community structure based on pre-determined node importance. Initial investigations into potential applications of this method are presented, including applications to information retrieval and biological data.

## 2 Introduction

Community detection algorithms are important tools that aid scientists in understanding the meso-scale organization of complex networks by grouping nodes and links into modules according to some objective. Two algorithms widely used to determine both the number of communities in such a network and their structure are Modularity optimization and Infomap [15]. Modularity groups nodes according whether the density of links within a proposed community is greater than that expected by chance. However, this results in a significant resolution limit to the size of the communities that can be detected by the algorithm [6]. Infomap groups nodes according to how well a proposed community captures the flow on the graph, measured by the compression of an information-theoretic description of a

random walker's movement [15] [14]. While Infomap also has a resolution limit, as any such two-level partitioning taking global network properties into account must, in practice it is small enough to usually not be a problem [8]. More problematic for Infomap is its experimentally verified field-of-view limit, in which large non-clique-like communities like lattices are over-partitioned, and its equivalence of clique-like communities [16].

Some methods to avoid these problems include extensions to hierarchical community structures, or by modifying the underlying Markov dynamics. Hierarchical community structures produce multiple levels of communities, where the communities of one level form the nodes for the next, or successively aggregate communities according to some parameter. Alternatively, we can extend the Markov dynamics into the continuous case and examine the community structure found at different time scales in the dynamics. Both modularity and Infomap can be understood through a random-walk markov process, where at each discrete time step the random walker transitions to another node. Modularity optimizes the probability that such a step will stay in the same community versus (typically) the configuration null model [17] [10], while Infomap tries to reduce the average description length of this random walk [15]. One method to examine the community structure at finer scales is to add a self-loops to the nodes, increasing the chance that the random walker will stay within a community because it does not leave the node [3]. A more comprehensive method extends the entire process to a continuous time one, in which the random walker leaves nodes according to independent and identical Poisson processes (exponentially distributed time between jumps). A time scale can then be chosen to describe the random walker's path, with smaller times finding the walker at closer nodes, allowing for smaller communities, and longer times finding the random walker at farther distances, allowing for larger communities [17] [10]. Markov time-sweeping is then the resampling of communities at many time scales, finding partitions that remain stable over long time ranges [16].

A natural extension to a continuous-time Markov process is to manipulate the rates at which random walkers leave nodes according to some importance placed upon them. This would allow for a focusing effect, viewing chosen parts of the network in greater detail while grouping the less relevant parts of the network into large chunks. This method would have applications to different fields such as allowing information-retrieval applications to personalize citation communities to a researcher's query or allowing biological and medical researchers to find small, important groups within a molecular network highly relevant to a specific problem. We investigate such an extension in this paper, the rest of which is divided as follows: a discussion of Infomap, a discussion of Infomap's extension to a continuous-time Markov process, an extension to variable rates according to node importance, a discussion of the method on simple model networks, and applications to real-world data.

# 3 Infomap and Extensions

## 3.1 Basic Infomap

The core of the Infomap community detection algorithm is the map equation, an objective function that provides an information theoretic lower bound on the length on the description of a random walker's path. This is accomplished by describing the path with codes on two levels, one specifying the modules and one for each module specifying a node within the module. The basic map equation is written as follows [14]:

$$L_M = \sum_{\alpha=1}^{c} p^{\alpha}_{\circlearrowleft} H(\mathcal{P}^{\alpha}) + q_{\curvearrowright} H(\mathcal{Q}) \tag{1}$$

This describes the average number of bits required to specify the next node in the random walker's path, using the stationary distribution of the walker's visits across nodes. Here $c$ is the number of modules, $\mathcal{P}^{\alpha}$ is the set of stationary probabilities for a random walker visit in module $\alpha$, $\mathcal{Q}$ is the set of probabilities of transitions between modules at stationarity, and $H$ is the entropy function. In each module is also included a *pseudo-node* describing the probability of exiting the module, $q_{\alpha\curvearrowright}$, in addition to the probabilities $\pi_i$ of visiting node $i$, such that the weight for within-module movements in the overall sum is $p^{\alpha}_{\circlearrowleft} = q_{\alpha\curvearrowright} + \sum_{i\in\alpha} \pi_i$. At stationarity, the probability of exiting a module is the same as the probability of entering a module, so the set of between-module transition probabilities is $\mathcal{Q} = \{q_{\alpha\curvearrowright} | 1 < \alpha < c\}$ and $q_{\curvearrowright} = \sum_{\alpha=1}^{c} q_{\alpha\curvearrowright}$.

It should be noted here that the weights of this sum are greater than one, such that $\sum_{\alpha=1}^{c} p^{\alpha}_{\circlearrowleft} + q_{\curvearrowright} = 1 + 2q_{\curvearrowright}$ This is appropriate because we want to compare different numbers of proposed communities and each new community adds new pseudo-nodes. An interpretation is that each proposed community adds a pseudo-node to the within-community network, representing the exit of that community, and a pseudo-node to the network of communities, representing the entrance into the community, both of which have the same transition probability, relative to the stationary distribution on the nodes of the network. For accurate comparison across different numbers of proposed communities, the average code length to describe a node must include this additional penalty, as we do not want to consider the average node and psuedo-node code length, but only the average node code length.

The random walk process can be described as an update rule on the distribution of visit probabilities over the nodes as shown in matrix notation in Equation 2.

$$\mathbf{p_{r+1}} = \mathbf{p_r} D^{-1} A = \mathbf{p_r} M \tag{2}$$

Where $\mathbf{p_r}$ is the row vector of visit probabilities after $r$ updates, starting with a uniform distribution on the nodes, $D^{-1}$ is a diagonal matrix of the inverse weighted degrees of the nodes (the inverse of the sum of outgoing link weights for each node), and $A$ is the adjacency matrix where $A_{ij}$ is the weight of a link from node $i$ to node $j$; $M$ is then the transition matrix.

## 3.2 Teleportation

On undirected networks, Equation 2 can use the adjacency matrix of a network without further problem. On directed networks, however, sources and sinks can occur, where a source is a node with only out-going edges, while a sink is a node with only in-coming edges. Sinks are a problem as they absorb random walkers, so typically random walkers are allowed to teleport to other nodes with some probability $\alpha$. The adjacency matrix can be rewritten to include teleportation probabilities:

$$A_i = (1 - \alpha)A_i^{(0)} + \alpha \cdot \mathbf{v} \tag{3}$$

Here $A_i$ is row $i$ of the adjacency matrix, $A_i^{(0)}$ denotes row $i$ in the network's adjacency matrix without teleportation, and $\mathbf{v}$ is a preference vector over the nodes. Typically $\alpha$ is chosen to be 0.15, and the preference vector is uniform over the nodes. Lambiotte and Rosvall have shown that there are smarter ways to implement telelportation [11] in terms of ranking and clustering that is more robust to the choice of $\alpha$. Such an improvement can be implemented by teleporting to links instead of nodes. This is accomplished by setting the preference vector proportional to the weighted in-degree of the nodes. The authors also show that even more robust performance can be achieved through "unrecorded" teleportation, by taking one final step with the original transition matrix to get the final stationary distribution (and since we take an additional step, the link teleportation preference vector is set in proportion to out-degree). Due to slightly more robust performance over traditional uniform node teleportation and ease in implementation, I use recorded link teleportation when teleportation is required in the rest of this paper. This is implemented as an adjacency matrix computed from Equation 3 with a preference vector proportional to the node in-degree. Unrecorded link teleportation is reportedly far more robust to community detection and would be very beneficial here, however I have encountered difficulties in its implementation that will be discussed in the Practical Considerations section later.

## 3.3 Continuous Time Extensions

The central idea behind continuous-time Markov processes is that they do not transition at precise moments, but probabilistically transition, with no memory of how long they have waited. Each transition is a Poisson process, where the waiting time in a given state is exponentially distributed, since this distribution is the only continuous, memoryless distribution. In other words, for a random variable $T$ that is exponentially distributed, the following holds: $P\{T > t + s | T > s\} = P\{T > t\}$ for positive real values $t > s$. The change in the distribution of random walkers following such processes on a network can be derived as the forward Kolmogorov equations [2] [10] [17]

$$\dot{p_i} = \sum_j \frac{A_{ji}}{k_j} p_j - p_i \tag{4}$$

Where $A_{ji}$ is the network's adjacency matrix, $k_j$ is the out-degree of node $j$, and $p_j$ is

the current density of random walker probability at node $j$. This can be interpreted as at every moment a node $i$ receiving or losing the difference between the probability of a random walker coming into the node over an incoming edge and the probability of the random walker already occupying the node (and thus leaving). Implicit in this equation is a notion of rate. Here, all random walkers leave node $i$ with a waiting time distributed exponentially with rate parameter $\lambda_i$. This can be interpreted in terms of the average waiting time at each node, which is $\lambda_i^{-1}$. This rate is split among the rates of individual transitions $\lambda_{i,j}$, since $\lambda_i = \sum_{i \neq j} \lambda_{i,j}$ [2]. Therefore in this basic version, rates $\lambda_i = 1 = \sum_j \frac{A_{ji}}{k_j}$. This system of equations can be rewritten in matrix notation:

$$\dot{\mathbf{p}} = -\mathbf{p}D^{-1}L \tag{5}$$

Where $D$ is the diagonal, out-degree matrix of the network, and $L = D - A$ is the graph Laplacian. The solution of this differential equation is given as:

$$\mathbf{p}(t) = \mathbf{p}(0)[e^{-tD^{-1}L}] \tag{6}$$

The stationary distribution is then found as $t \to$ inf. This solution suggests a new transition matrix to use when computing Infomap, the matrix exponential $T = [e^{-tD^{-1}L}]$ [17]. Each row of this matrix can be interpreted as the probability of transitioning to another node, along any path, in time $t$. We lose the interpretation of Infomap as a complete description of the random walker's path, since we only observe where the random walker has transitioned to after a time $t$, but we gain insight into flows at greater or shorter distances. It should be noted that this extension also adds self-edges to the random walk much like [3], however it doesn't limit walkers to single step updates at regular intervals, and as a particular difference, the continuous time extension allows transitions to nodes not connected by a link even without teleportation, so long as there exists some path between the nodes.

Since probability density is conserved for ergodic dynamics (without sinks in the network), each row of $T$ sums to one, and we have that $D_T^{-1}T = T$, where $D_T$ is the diagonal out-degree matrix for $T$. Thus, using the $T$ as the network itself preserves the same transition matrix in the traditional Infomap algorithm, allowing for the use of previous tools with the new network. However, $T$ is now directional, even on undirected graphs. In cases where this is important, we can make use of the fact that at stationarity the flow of probability out of a node is equal to the flow into a node, thus the matrix $X_{ij} = \pi_i T_{ij}$, is symmetric (i.e. undirected), and since the out-degree of each node $i$ is just the constant $\pi_i$ multiplied by one, $D_X^{-1}X = T$, and this undirected network is similarly an equivalent problem for existing Infomap tools. Since the results will be the same, the renormalization to an undirected graph is not performed, unless noted.

The extension to this continuous time method we will examine in this report is the manipulation of the rates of individual nodes based on the interest of the application. Thus we set $\lambda_i$ according to some mapping between some information we have about the nodes and rate values. The corresponding equations then become:

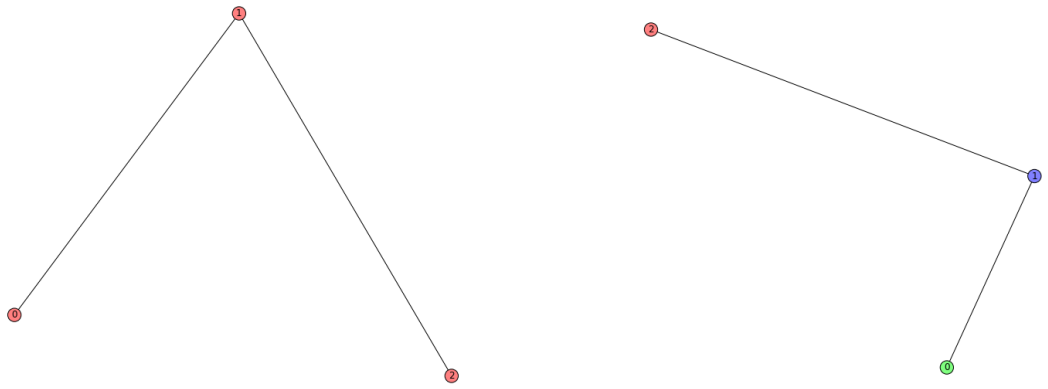


Figure 1: Found Markov Rate Infomap communities. Left: The simple network with a low rate on the middle node and high rates on the ends. Right: The simple network with a high rate on the middle node and low rates on the ends.

$$\dot{p_i} = \sum_j \frac{A_{ji}}{k_j} \lambda_j p_j - \lambda_i p_i \tag{7}$$

$$\dot{\mathbf{p}} = -\mathbf{p}\Lambda D^{-1} L \tag{8}$$

$$\mathbf{p}(t) = \mathbf{p}(0)[e^{-t\Lambda D^{-1} L}] \tag{9}$$

Where $\lambda_i$ is the leaving rate of random walkers on node $i$ as described before, and $\Lambda$ is the diagonal matrix of these rates. We refer to the Infomap algorithm based on this diffusion process as the *Markov Rate Infomap*.

# 4 Artificially Created Examples

## 4.1 A Simple Example

To develop a better understanding of how setting different rates on different nodes can affect the map equation, we start by examining in detail a very simple undirected network of three nodes and two links. We start by holding time constant at one time unit while changing the rate parameters for the nodes. The units here are abstract, such that the rates and the time chosen have meaning only in reference to each other. We choose two rates, 100 and 0.1, and assign them to different nodes. Because this rate is the parameter for the exponential distribution on waiting times at each node, a higher rate denotes a faster movement from the node, such that, intuitively, low rate nodes can be understood as attracting random walker

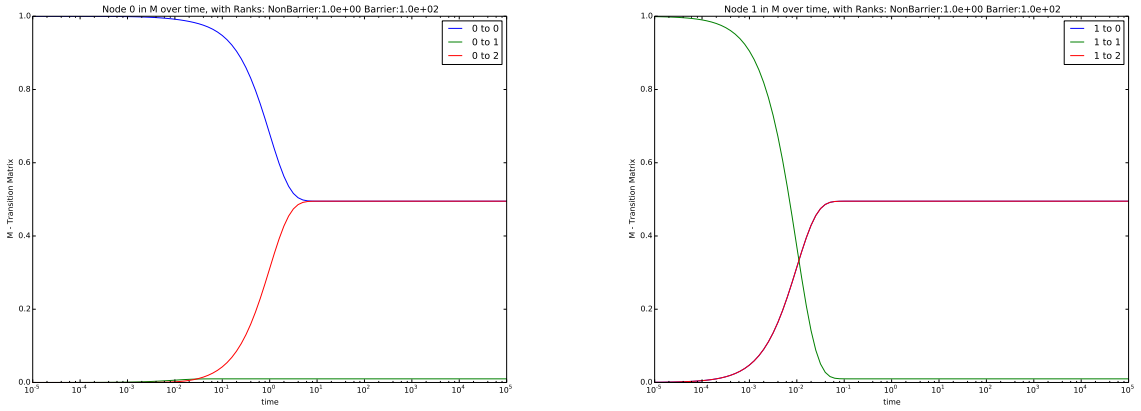



Figure 2: Transitions for different time intervals, given that the outer nodes have rate 1 and the center has rate 100. Left: Transitions from node 0 (outer) to the other nodes, Right: Transitions from node 1 (center) to the other nodes

density from higher rate nodes. Shown in Figure 1 are the resulting Infomap communities if the center node has a low rate or a high rate, while the outer nodes have the opposite.

If the middle node has a low rate, all nodes are grouped together, since the overhead of assigning the rarely visited and quickly left outer nodes to their own communities is very high. If the middle node is given a high rate however, each node is assigned its own community, since the likelihood of remaining in an outer node having visited it during the previous time step is very high, while the walker is rarely found on the center node. We will call the middle node a *barrier* since it separates the two outer nodes with a different rate.

To see how the transition matrix changes with time for set rates, we can look at the transitions from outer node 0 to the other nodes (which by symmetry is the same for node 2) and the transitions from the middle node 1 in Figure 2. Plotted are the values for the rows of the transition matrix. It can be seen that for short time intervals, the outer nodes tend to keep their random walkers, while at longer intervals a walker is equally likely to move from one outer node to the other as it is to remain. The barrier node is rarely transitioned to, and walkers are only likely to stay on it during brief time intervals, quickly overcome by transitions to the outer nodes.

Since rates are only in terms of the time unit, if we keep time constant at 1 and likewise keep a ratio of 100 to 1 between barrier and non-barrier rates, the graphs in Figure 2 can be exactly reproduced but with the x-axis labeled as the non-barrier rank. If instead we keep the barrier rate constant at 100 and the time constant at 1, but vary the non-barrier rates we can graph the transitions in Figure 3. For very low leaving rates, the outer nodes keep their random walkers, while transitioning to the other outer node as the rate approaches the time interval, 1. As the non-barrier rate approaches the barrier rate of 100, the transition to the center barrier node exceeds the transition to either outer node. This occurs before the rates are equivalent because the barrier also has a higher degree, and the walker can

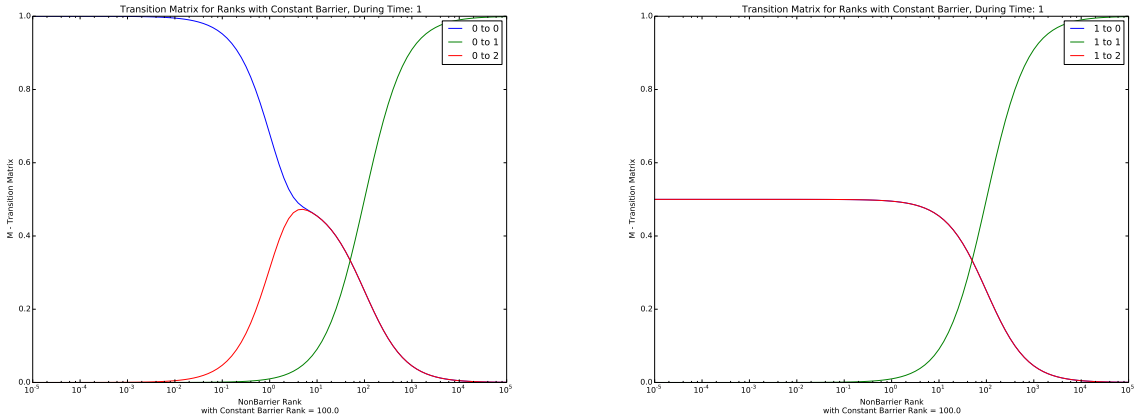



Figure 3: Transitions for different non-barrier rates, given a time-step of 1 and barrier rate of 100, Left: Transitions from node 0 (outer) to the other nodes, Right: Transitions from node 1 (center, barrier) to the other nodes

transition to the other outer node before returning to the barrier node, or making any other finite number of round trips.

## 4.2 Adding a node

We can add a second barrier node in order to better understand where the Infomap algorithm chooses to make cuts between communities. In Figure 4 we can see the found community structure once again using a time-step of 1 and rates of 100 and 0.1. On the left, the barrier is given a lower rate, and two communities are found, split between the two lower rate barriers. On the right, the barrier is given a higher rate, and three communities are found, one for each of the low-rate outer nodes and one for the barrier nodes. We can infer from these splits that the tendency is to separate low-rate nodes into separate communities. This make intuitive sense, since the walker is less likely to transition out of the low-rate nodes, each coded step would most often record staying in the lowest rate node in the community, rarely transitioning out of communities composed mainly of low-rate nodes.

As before we look at the effect of different time steps on the transition matrix for node 0 (outer, non-barrier) and node 1 (center, barrier) which symmetrically describe the other two nodes in the network. If we hold the barrier rate at 100 and the non-barrier rates at 1 we can see in Figure 5 transitions similar to that of the simple network. The outer nodes tend to retain random walker density until the time step is large enough to expect a walker to transition to either low-rate node equally. The transitions from the barrier nodes are a little more interesting, but tell the same story. For very short time steps the walkers transition to barrier nodes, but at longer time-steps the walkers tend to transition to the low-rate nodes. The proximity of the node being transitioned to is noticeable for certain time domains, but eventually vanishes until only the rate is important.

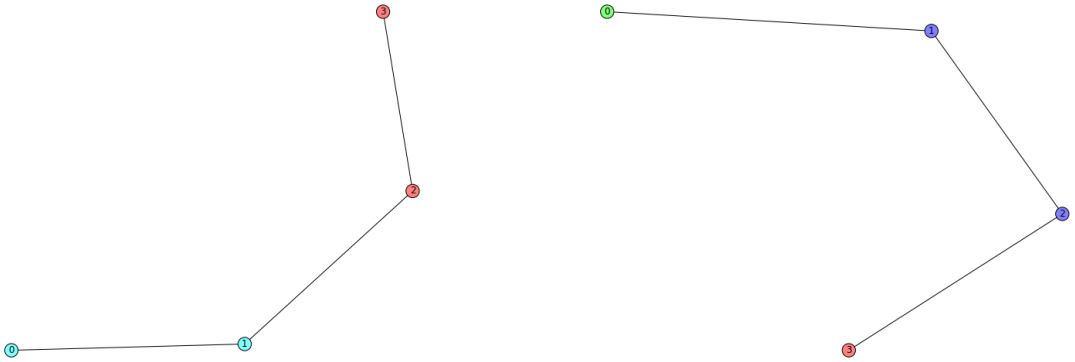



Figure 4: Found Markov Rate Infomap communities. Left: The double-barrier simple network with a low rate on the middle nodes and high rates on the ends. Right: The double-barrier simple network with a high rates on the middle nodes and low rates on the ends.

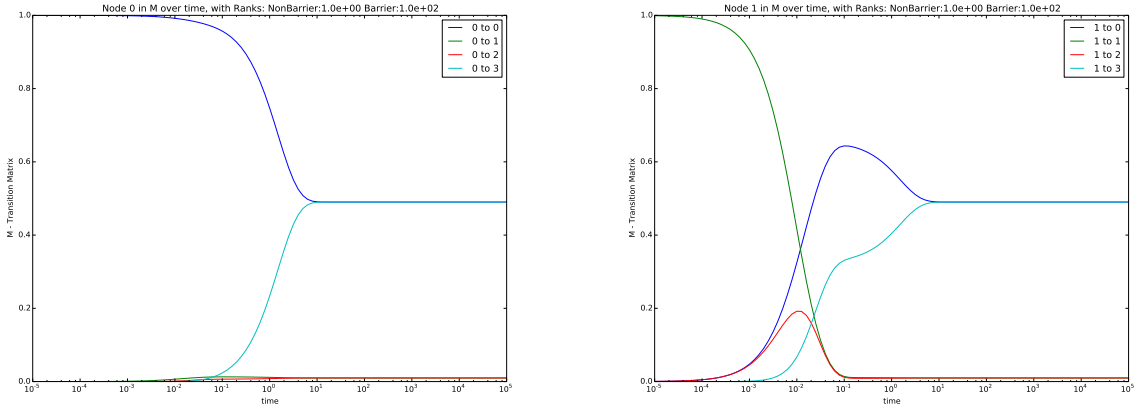



Figure 5: Transitions for different time intervals, given that the outer nodes have rate 1 and the center has rate 100. Left: Transitions from node 0 (outer) to the other nodes, Right: Transitions from node 1 (center) to the other nodes

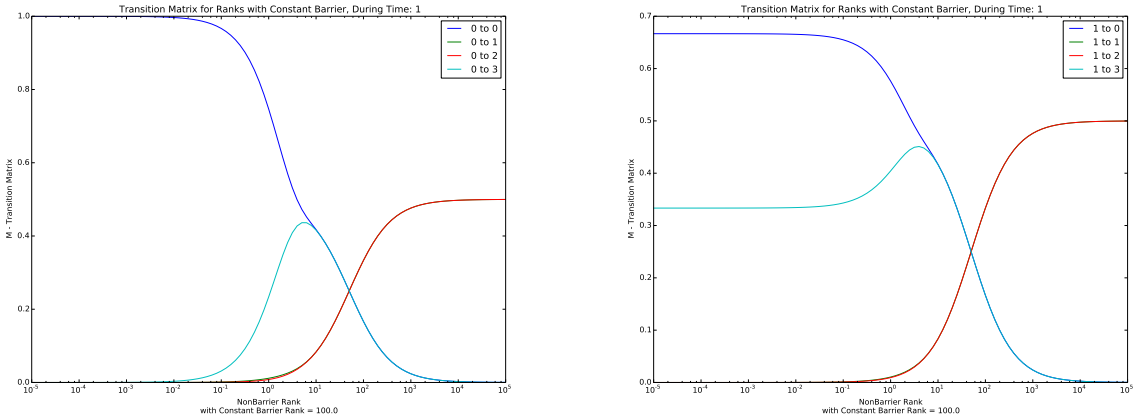



Figure 6: Transitions for different non-barrier rates, given a time-step of 1 and barrier rate of 100, Left: Transitions from node 0 (outer) to the other nodes, Right: Transitions from node 1 (center, barrier) to the other nodes

We can also look at what happens to transitions when we hold the barrier rate constant at 100, the time at 1 and vary the non-barrier rate in Figure 6. The separation between transitions to the nearer low-rate node is maintained until the non-barrier rate approaches the barrier rate, for both low-rate and high-rate nodes.

## 4.3 Rates on a Grid

To investigate a more interesting, but regular, example, we create a toroidal grid, where the network is composed of 120 nodes, in the shape of a 3 by 40 toroid, drawn in the left of Figure 7. Placed at equal intervals, $m$, along the long axis, we can denote barriers that are given a higher rate than the other nodes. For this example, the barrier rates are 100, while the non-barrier rates are 1. For example, if we place the barriers as groups of 3 nodes at intervals of 4 nodes along the long axis, we can draw the graph in the circular layout shown on the right of Figure 7 where the barriers are shown in pink. We can then examine the communities found by Infomap with and without the rate information.

As we can see in the left of Figure 8, the basic Infomap algorithm, without any rate information, over-partitions the lattice structure of the toroid. There is no real community structure, but there are advantages found in the compression of the random walk description, so 14 communities of different sizes are cut from the graph. In the Markov Rate Infomap, however, there are 11 found communities, mostly corresponding to the 10 barriers. Most communities are found approximately the same size, with one barrier set (of three nodes). Nearly every boundary between communities occurs between a high-rate barrier node and a low-rate non-barrier node. However, perhaps due to the heuristic optimization method, one community has two barriers and two have none. Perhaps because of this miscounting an especially small community was also created containing one barrier, but including the only

Figure 7: The toroidal grid, Left: Spring Layout, Right: Circular layout, pink denotes barriers

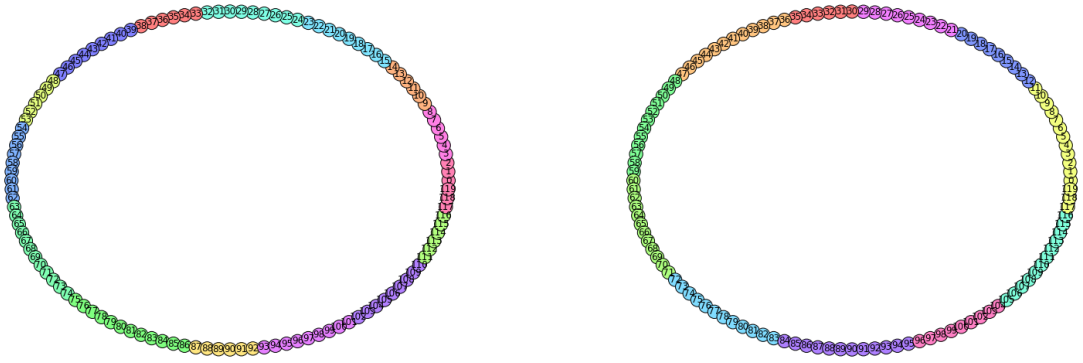


Figure 8: The toroidal grid, different colors denote different communities. Left: Basic Infomap found communities, Right: Markov Rate based Infomap

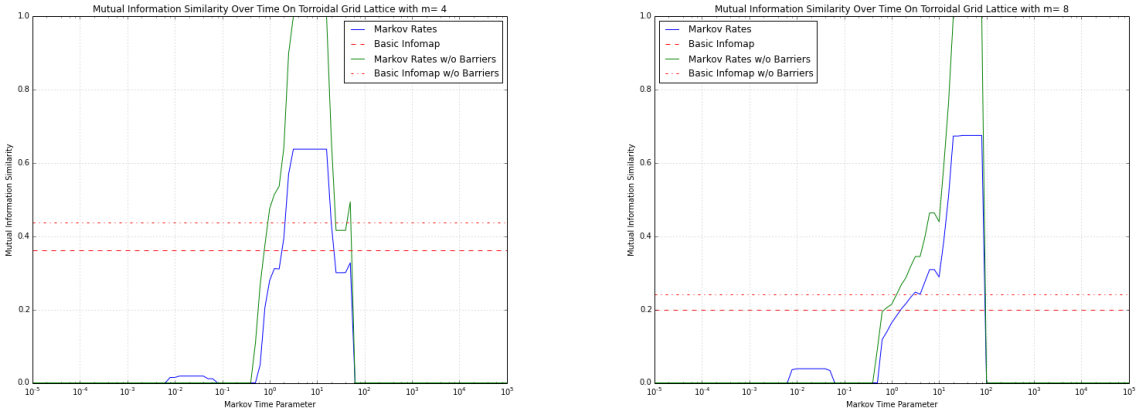



Figure 9: Mutual Information between the Markov Rate InfoMap found communities and the hypothetical communities for intervals between barriers: Left: m=4 and Right: m=8

community boundary not between a barrier and non-barrier node.

By looking at the mutual information similarity between the found communities and the *a priori* communities drawn between barriers, with an additional community representing the barriers, we can plot the effects of time on the community detection in Figure 9. Since the barriers themselves do not particularly belong to either community we plot the mutual information similarity both considering the barriers as their own community and without considering the barriers at all. There are significant portions of the time parameter during which the Markov Rate Infomap separates the *a priori* communities significantly better than the basic Infomap baseline, and perfectly for certain time ranges (ignoring the barrier nodes).

The plateau of best performance occurs when the time range is on the order of the community size, since the low-rate nodes have a rate of 1. For reference, the size of the *a priori* communities can be calculated as $(m-1)\times 3$ and the x axis of the plots in Figure 9 is in logarithmic scale. There is a sudden drop off when the time-step is on the order of the whole network, due to our setting the non-barrier rates to 1. Since the average leaving time for such a node is $1^{-1}=1$ then the average time to make 61 such jumps (the farthest any node on the toroid is from any other) is only 61 time units. Of course, we must take into account alternative paths, but any difference in transition probability will quickly collapse around this time. If we consider the transition graphs of the simple examples we can see that our best performance occurs during the period when transitions to low-rate nodes occur with easily distinguishable probabilities.

## 4.4 A Double-Barrier Toroid Grid

As we did with the simple example, we can add a double barrier to the toroid. The overall structure of the network remains the same, but now the each barrier is composed of six nodes, as shown in Figure 10. For barriers spaced $m$-nodes apart on the long axis we then

Figure 10: The toroidal grid, Left: Circular layout, pink denotes barriers, Right: Found communities using Markov Rate Infomap

calculate the *a priori* community sizes as $(m-1)\times 3$. The Markov Rate Infomap community detection is better behaved with these double barriers, as we can see on the right of Figure 10. Community boundaries split the double barriers. The clusters of low-rate nodes keep a density of random walkers, and are more likely to transition to the nearby low-rate nodes given some time interval. If the walker does reach the barrier, it has a greater chance to return. We may suppose that the distinctions between transitions to nearby low-rate nodes seen in the double-barrier simple network are again apparent here, with the closer low-rate nodes more attractive. The mutual information similarity between the communities found through the Markov Rate Infomap and the *a priori* communities are shown in Figure 11. The plateaus of good performance are visibly wider and the slopes less jagged.

## 4.5 Simple Random Graph

We can extend the simple network to a random network to see how the results of the simpler examples extend to less structured networks. For this purpose we use an Erdos-Renyi binomial random graph, or a stochastic block model, in which the existence of every edge between nodes within predefined communities is created with a certain probability, while those between nodes of different communities are created with a smaller probability. In the examples that follow, there will be a 10% chance for any intra-community edge and a 1% chance for any inter-community edge. Increasing the parallels with the simple and double-barrier simple networks we add an additional constraint, creating only three communities of twenty nodes each and not allowing connections between two communities, such that one community forms the only path between the other two, like the barrier of the simple network as shown in Figure 12. This middle barrier can be split in two such that each half of this community can form connections to only one other community, which we will refer to as a split middle, much like the double-barrier simple network. We will call these underlying

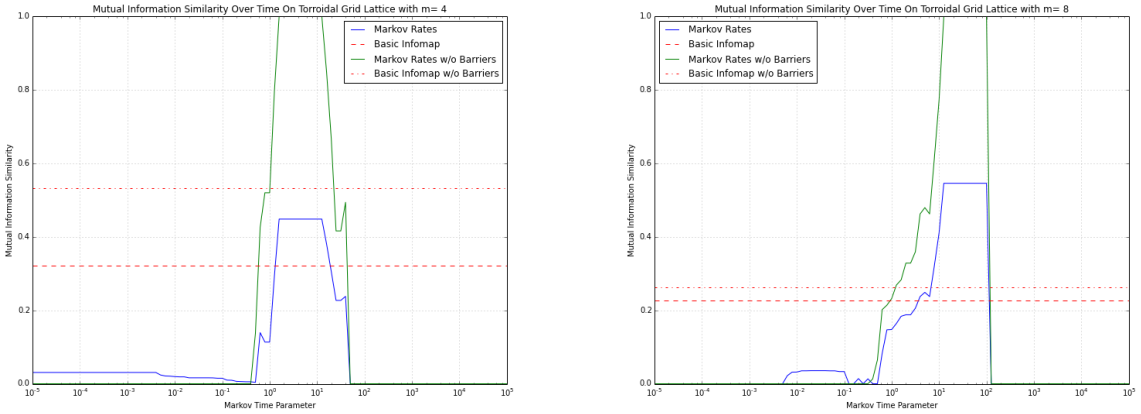



Figure 11: Mutual Information between the Markov Rate InfoMap found communities and the hypothetical communities for intervals between barriers: Left: m=4 and Right: m=8

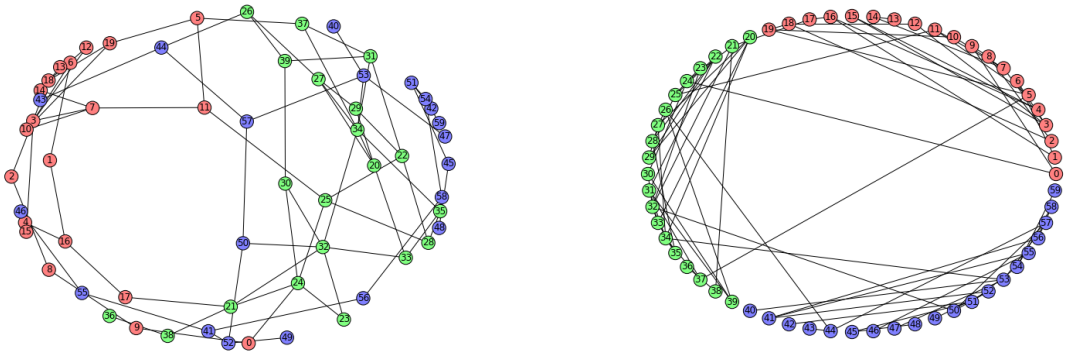


Figure 12: The simple undirected random network. The colors represent the predefined communities. The probability of the creation of an intra-community edge is 10% while the probability of an inter-community edge is 1%. Left:Spring layout, Right: Circular Layout

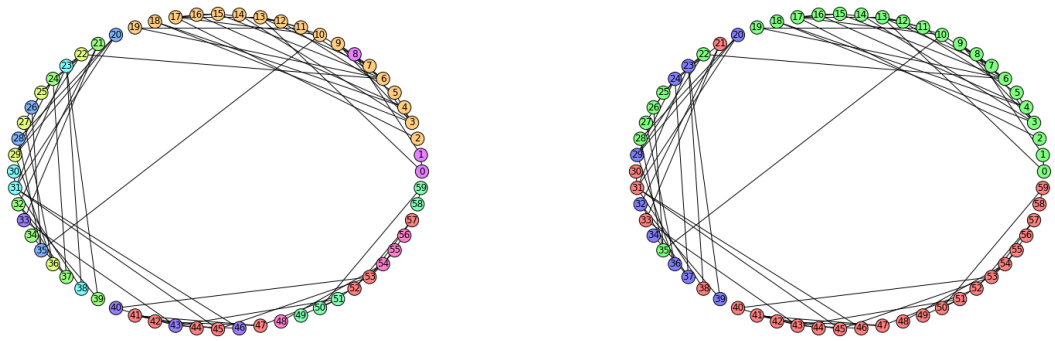


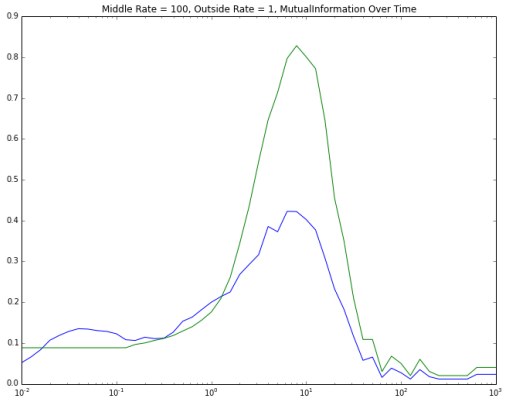
Figure 13: The found communities of the simple undirect random network. Left: Basic Infomap, Right: Markov Rate Infomap, barrier rate 100, nonbarrier rate 1, time 9

tolopological communities *groups* to avoid confusion with the community structure found by Infomap.

We can use this model to observe how the basic Infomap and Markov Rate Infomap split the network into communities. In Figure 13 we see that the basic Infomap tends to over-partition the network beyond the predefined structure, while the Markov Rate Infomap, with a higher rate in the barrier group and a time step of 9, partitions the network into the two non-barrier communities with the barrier nodes partitioned to either non-barrier community or a third barrier-exclusive community. As could be expected, for certain parameters, the middle group of higher rates between two low rate groups is where the community boundaries are formed.

To observe the effects of the time parameter on the community split we average the mutual information between found community structure at various times for 100 random graphs with the *a priori* structure given by the generation of the network, as drawn in Figure 14. When the barrier group is given a high rate we see that the performance peaks around the size of the group, as in the case of the grid network. Like the double-barrier grid network, the additional constraint that barrier nodes each may be connected to only one low-rate group increases the plateau width and raises similarity even higher.

If we reverse the rates such that the middle barrier has a low rate and the outer communities have higher rates, the similarity reaches peaks at short time steps, but these peaks are not as high as the previous rate distributions, although the plateau of good performance lasts over a wider domain. This is explained by the behavior of the fast rate and low rates during a small time step. The high-rate outer communities have a long enough time that the walker distribution is spread within them without leaving, while the low-rate barrier community has no time to spread its distribution, thus the outer communities are grouped correctly, while the center barrier group is over-partitioned. As before, a split-middle improves performance.



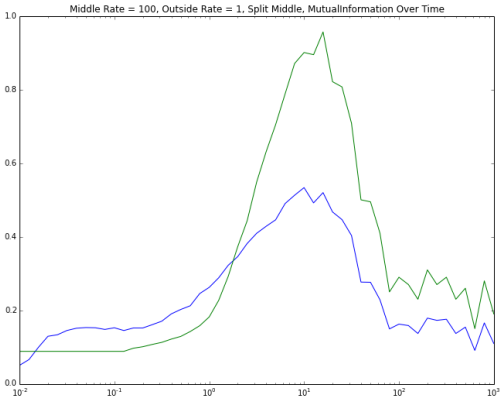
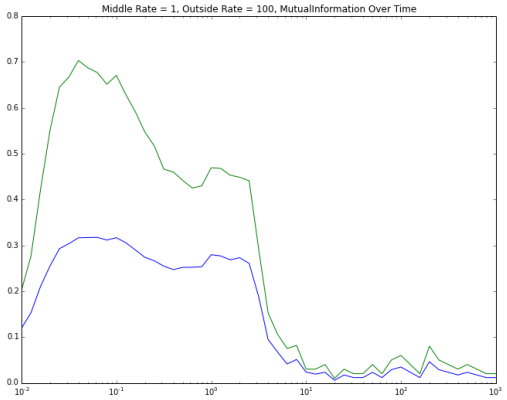




Figure 14: Mutual information similarity between the *a priori* community structure of the networks vs the Markov Rate Infomap found communities at different times given the barrier community has a rate of 100 and the non-barrier communities have rates of 1. Each point is the average mutual information similarity of communities detected on 100 generated graphs. Blue: The mutual information similarity between found structure and *a priori* communities, Green: The same, but with the barrier community ignored, Left: connections to both other communities allowed for every node of the barrier community, Right: split middle



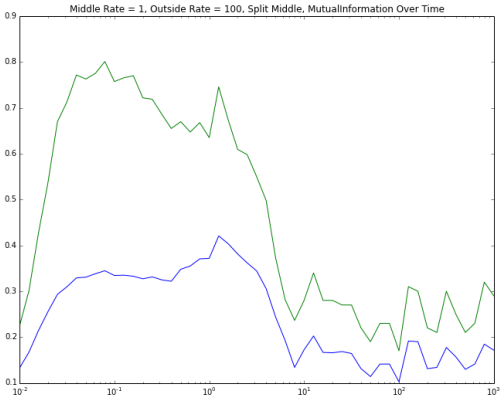




Figure 15: Mutual information similarity between the *a priori* community structure of the networks vs the Markov Rate Infomap found communities at different times given the barrier community has a rate of 1 and the non-barrier communities have rates of 100. Each point is the average mutual information similarity of communities detected on 100 generated graphs. Blue: The mutual information similarity between found structure and *a priori* communities, Green: The same, but with the barrier community ignored, Left: connections to both other communities allowed for every node of the barrier community, Right: split middle

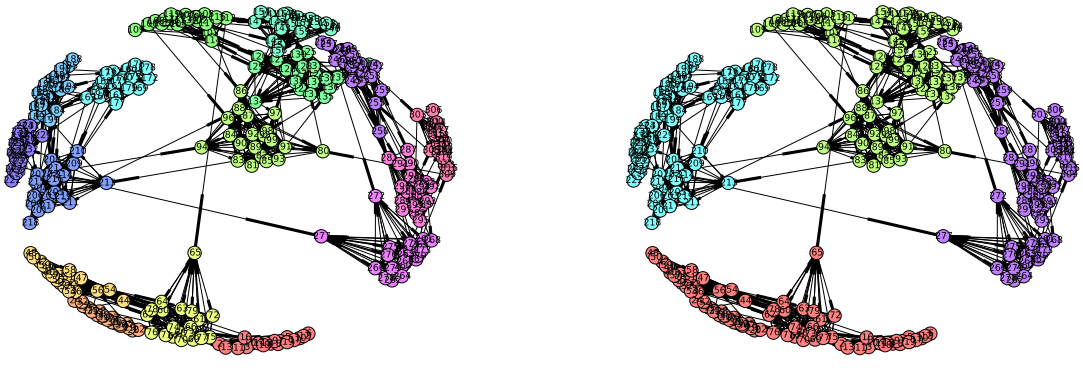


Figure 16: A two-level, hierarchical, directed random network. Left: low-level community coloring, Right: high-level community coloring

## 4.6 Partial Summary

Now, we can step back and note our main observations about the behavior of the Markov Rate Infomap community detection.

1. Nodes with low-rates tend to split into different communities
2. Nodes with high-rates tend to be grouped together, except when near low-rate nodes and observed at a sufficiently long time interval.
3. If high-rate nodes are near low-rate nodes, community boundaries tend to occur along their edges.

Intuitively, one can imagine that walkers are attracted to nodes of lower rate, and thus higher rate nodes are attracted into communities of nodes of lower rates than themselves. This attraction reduces the inter-community terms in the map equation. However, all nodes are as well repulsed by each other, with a strength inversely correlated with the nodes' rates. This repulsion reduces the intra-community entropy terms of the map equation. Only when the structure of the network or time step is sufficiently long can this repulsion be overcome to group nodes together.

## 4.7 Two Level Hierarchical Random Network

Now we would like to see how the Markov Rate Infomap performs on a more complicated example with structure closer to a real application. If we have a network with some nodes of more interest to a researcher than others, we would like the more important nodes to form a finer-grained community structure, while the less important nodes compose broader communities. In both cases we would still like our method to form communities based on

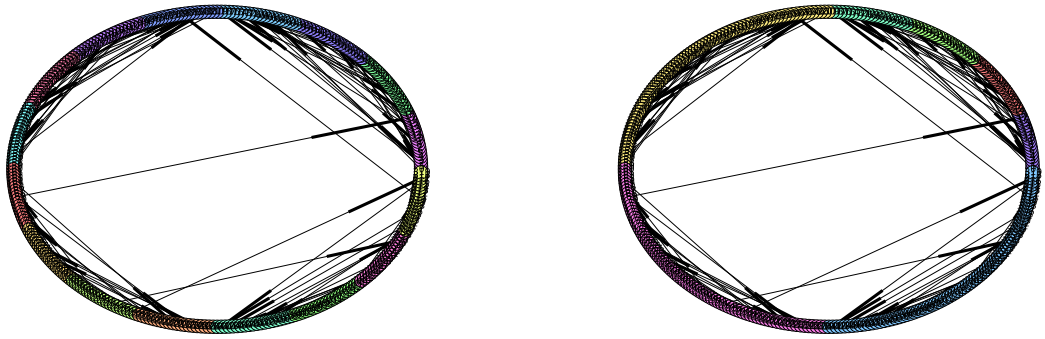


Figure 17: A two-level, hierarchical, found communities. Left: Basic Infomap detected communities, Right: Markov Rate Informap detected communities

the underlying structure of the network. A simple model for investigating such a problem is a two-level hierarchical network, in which two sets of community structure are apparent. To this end we create a simple two level random graph using a stochastic block model with three probabilities. By construction there are 16 low-level communities of 20 nodes with a probabiliy of 40% for the creation of each possible within-community edge. There are 4 non-overlapping high-level communities composed of the nodes of 4 low-level communities with a probability of .6% for each within-high-level-community edge. Finally there is a .006% chance for each inter-high-level-community edge. To make the example more general, all edges are directed. Such a graph is colored by both low and high leveled communities in Figure 16.

Suppose all the nodes of one high-level community are more interesting to a researcher than those of the other communities. Can we determine the lower-level communities within that particular high-level community by giving the nodes within that high-level community a lower rate than all the rest? Let us call the lower rate the important rate, and the higher rate, the unimportant rate. The parameters are set as follows: the time step is $e$, the important rate is 1, the unimportant rate is 10, and the teleporation rate to 0.001. We can see in Figure 17 that basic Infomap (with no rate information) finds the lowest level communities, while the Markov Rate Infomap splits the one important rate high-level community into the low level community structure while observing the high-level community structure for the remainder of the network. Unfortunately this partitioning is very sensitive to the values of the stochastic block model and the teleportation rate. Using unrecorded teleportation may help reduce this dependence.

To investigate the effects of changes in the time-parameter, we hold all other parameters constant as described above, and change the time parameter, plotting the mutual information between the detected community structure and the defined community structures at both the low and high levels. A covering that splits one high level community into low level

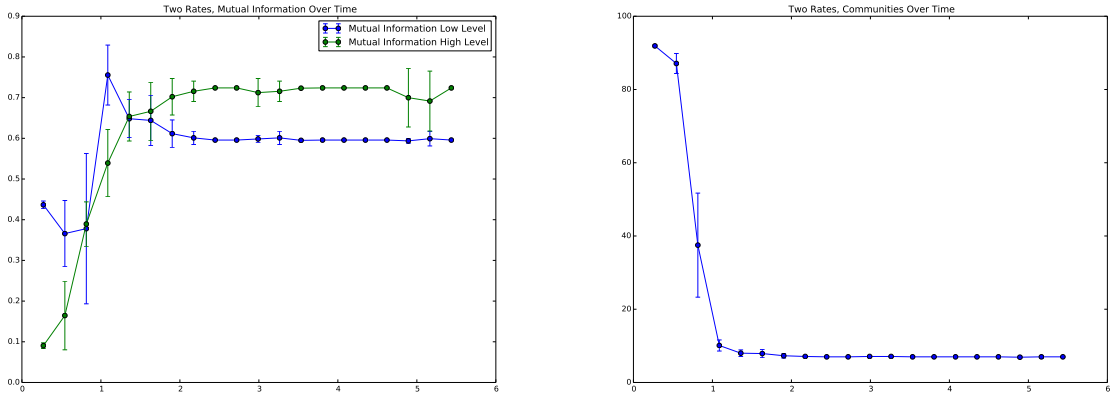



Figure 18: Left: Mutual Infomation between found communities and defined communities. Right: Number of found communities over time

community while keeping the rest of the high level communities should have a high mutual information similarity to both *a priori* defined community structures. Indeed, as seen in Figure 18 after about a time of 2 the performance plateaus for a local domain (linear scale rather than the log scales of previous examples), while the number of found communities falls to 7 and remains. Error bars are the standard deviations over ten randomly generated networks.

If instead we set the time constant at $e$, after the beginning of the above found performance plateau, we can observe the effect of changing the size of the lowest level community. Surprisingly, as seen in Figure 19, the community detection is remarkably robust for community sizes greater than 20, at least in a local range, nearly always finding the 7 communities we expect. In this case the high probability for the existence of every within-community link must be high enough that low level communities are still found despite much larger sizes.

# 5 Practical Considerations

Before moving on to real data, we must address some difficulties implementing continuous time Markov processes in practice. The first difficulty arises in implementing teleportation in combination with the continuous time process, while the second involves the computational complexity of a matrix exponential. Both problems require more exploration than I give here in order to implement this method as a useful tool in either user-based applications or further scientific investigations, the first because it affects the sensitivity of the method on a user-defined parameter, and the second because it affects the run-time of the method.

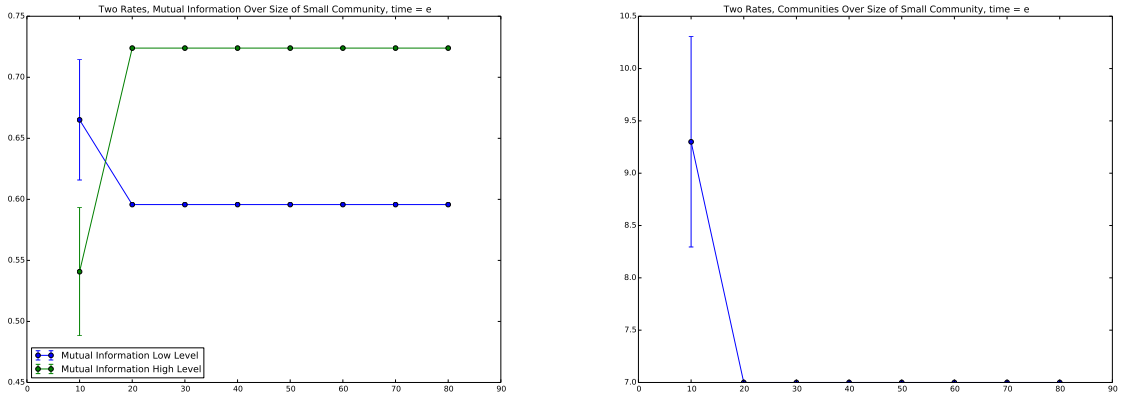



Figure 19: Left: Mutual Infomation between found communities and defined communities. Right: Number of found communities over the size of the lowest level community

## 5.1 Teleportation Difficulties

As mentioned before, smarter, unrecorded link teleportation allows more robust performance of a community detection algorithm to the rate of the teleportation parameter [11]. Indeed, the communities found in the two-level hierarchical random network in particular are very sensitive to the setting of the teleportation rate, which must be set quite low. In other work the problem is avoided by observing only undirected networks [17] [16], or by altering the adjacency matrix to include teleportation transitions in order to ensure ergodic dynamics [10], as we have. With ergodic dynamics, and an original transition matrix normalized such that all rows (if rows represent source nodes) sum to one, we can ensure that the rows of the matrix exponential also sum to one, and thus can be interpreted as a probability of transitioning from one node to any other, along any path, in an amount of time. If instead we have sinks, or nodes without outgoing edges, and no teleportation, the rows of the matrix exponential no longer sum to one, thus losing that interpretation.

By applying recorded link teleportation, in proportion to node in-degree, to the adjacency matrix before the matrix exponential is computed, we ensure ergodic dynamics. However, unrecorded-link teleportation, in proportion to node out-degree, requires an additional step without teleportation. If we take the matrix exponential and then apply teleportation, this can be accomplished, but a new series of questions arise. In-degree and out-degree will not be preserved by the matrix exponential, so should the original adjacency matrix or the matrix exponential be used to determine the appropiate teleporation preference vector? Should we normalize the adjacency matrix such that each row sums to 1 before the exponential? If we do the out-degree of all nodes in the exponential will be equal or less than 1, but if we don't, the interpretation of the exponential becomes even more vague as random walker density could increase by traveling through some nodes. Can we ensure in any case, that recorded and unrecorded link teleportation have the same stationary distribution?

Applying teleportation after the exponential of the matrix also creates a difficulty in finding the equivalent problem to pass to existing tools. The out-degree can be preserved if all rows sum to 1 after the matrix exponential by multiplying each row by its corresponding element in the out-degree preference vector. However, since changing in-degree would change the relative outgoing weights there is no corresponding simple solution for recorded link teleportation. In addition, since the matrix without teleportation must be used to calculate the stationary distribution, we must rely on the teleporation implemented in the Infomap optimization code. In this report we use Rosvall's existing code [14], for which my calculations of the value of the map equation on arbitrary partitions agree exactly for recorded link teleportation. Once unrecorded link teleportation is used, my calculations of the stationary distribution exactly match the existing code, but my calculations for the value of the map equation do not. I have tried calculating the map equation with many of the variations mentioned in this section and with possible unstated assumptions in [11]. Regardless, my calculations of the map equation do not match the code, so I am not confident using it with unrecorded link teleportation.

Despite these difficulties, there may be a number of advantages to applying teleporation after the matrix exponential. The adjacency matrix of a network, despite possibly large numbers of nodes, is often sparse, which can lead to considerable savings in computation time. However, if the network is strongly connected, the matrix exponential will be a dense matrix, and if we include teleportation in the adjacency matrix, it will likely be dense as well. More work is also required to know which method would better condition the matrix for the computation of the exponential.

As the simplest case, all teleportation in this report is recorded-link teleportation applied to the adjacency matrix before the matrix exponential.

## 5.2 Dubious Ways to Compute the Matrix Exponential

Efficient and accurate computation of the exponential of a matrix is itself a difficult problem with a lot of literature, reviewed in Moler and Van Loan's paper "Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later" [13]. They note that the matrix exponential can be formally defined by the convergent power series from its Taylor expansion:

$$e^{tA} = I + tA + \frac{t^2A^2}{2!} + ... \tag{10}$$

Where $A$ is an $n \times n$ real or complex matrix, and $t$ is a constant. However, considerations of efficiency and numerical roundoff and cancellation errors lead to different approximations whose algorithmic complexity is upwards of $O(n^3)$, which is considerable. The scipy library uses a method involving the scaling and squaring of a matrix and the Padé approximation of the exponential of the matrix [1] [7]. This method takes advantage of the fact that $e^A = (e^{A/m})^m$ to scale the matrix to reduce the roundoff error and computational costs of the Padé approximation, which increase with $t||A||$. This is an effective method to compute the whole matrix exponential at one given time $t$ [13].

However, for the purposes of large matrices, or repeated evaluations at different values $t$, decomposition methods may be more efficient. These methods take advantage of the implication from the power series that if $A = SBS^{-1}$ then $e^{tA} = Se^{tB}S^{-1}$. An $S$ is found such that $e^{tB}$ is easy to compute. For example, in an eigenvector or singular value decomposition, the matrix $B$ is diagonal, and diagonal matrices have an exponential easy to calculate: $e^{tB} = diag(e^{B_{1,1}t}, \ldots, e^{B_{n,n}t})$. This algorithm fails however when $A$ does not have full rank. When applicable, it could allow for fast Markov time-sweeping, as the complexity of recomputing the exponential at different values of $t$ would only require two matrix multiplications, rather than the entire computation of the exponential.

Another family of methods that may be of interest for use with large networks are Krylov space method, such as the one implemented in the expokit software package [19] [13]. These methods estimate the solution, not to the exponential of the matrix itself, but of the multiplication of a matrix by a vector. Through Arnoldi iteration, a stabilized repeated multiplication of a matrix to a vector, the original matrix $A$ can be partially reduced to a matrix $V_m$ of $m$ orthonormal columns and an $m \times m$ upper Hessenberg matrix $H_m$. Then $e^A\mathbf{v} \approx V_m e^{H_m} e_1$ where $e_1$ is the first column of identity matrix $I_m$. This reduces a large sparse problem to a small dense one [13]. Because this algorithm avoids matrix multiplications, there are considerable time savings. By setting $\vec{v}$ as a mask over nodes in a community, the values of $q_{\alpha\curvearrowright}$ could be quickly calculated, if this method were integrated into the optimization algorithm for Infomap. However, without further investigation it is hard to tell if the advantages of this algorithm would be dissipated through the repeated calculations of the Louvain method. It would certainly be necessary to keep the sparseness of the adjacency matrix by avoiding teleportation until after the exponential of the matrix is computed, requiring a solution to the problems discussed in the previous subsection.

# 6 Real World Applications

Despite the above difficulties, we have some preliminary results applying this method to large networks, including real-world data. In this section I will discuss the application of Markov Rate Infomap to a number of problems. The main benefit of this method is in finding different communities depending on the preferences of the user or the details of a problem, and we will try to highlight this in applications. First, as an intuitive and visual example, the method's use on image segmentation problems will be discussed. Second, initial results using the method for an information retrieval task will be presented. Finally, I will discuss its possible use with disease networks.

## 6.1 Image Segmentation

Image segmentation is a process by which an image is automatically broken into sections in order to detect the objects in an image. The objects of interest may be different depending on user and application [5]. Although it is an expensive approach, the image segmentation problem can be turned into a community detection problem through the creation of an

Figure 20: 78 × 78px image of balloons

undirected network based on the image. This approach is expensive because each pixel in the image becomes a node in the network, with links to adjacent nodes weighted according to some measure of similarity between nodes. A simple 100 × 100 image then has 10,000 nodes A traditional way to weight edges is given by Equation 11 [5] [18]:

$$w_{ij} = \begin{cases} e^{-(\frac{d(i,j)^2}{\sigma_x^2} + \frac{|F(i)-F(j)|^2}{\sigma_i^2})} & \text{if } d(i,j) < d_{max} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Where $d(i,j)$ is a distance function between nodes $i$ and $j$ in the orignal image, the Chebyshev distance in the following examples; $F(i)$ is the feature vector of pixel $i$, the grayscale value in the following examples; and $\sigma_x$, $\sigma_i$, and $d_{max}$ are user defined parameters. The addition of these extra parameters complicates analysis of the use of continuous-time Markov process method, but some differences between rate mappings and time are clear.

We will limit our analysis to the image presented in Figure 20[1] and used in [16]. It is a simple 78px×78px image of balloons. The grayscale version of this image is used in the following analysis. If we set $\sigma_x = 1.2$, $\sigma_i = 10$, and $d_{max} = 1$ we find that the normal Infomap algorithm segments the picture as shown in Figure 21. Each found community is given a different color in the figure, but the colors mean nothing else. The figure is over-segmented and ignores the boundaries of many objects.

Markov Rate Infomap was tested on this network by giving node rates according to two mapping function. The first mapping function we will refer to as *basic rates* and simply assigns a rate of 1 to all pixels containing a grayscale greater than 100 and 100 to all other pixels. The second mapping function we will refer to as *inverse grayscale rates* and assigns a rate of $1/(.1 + F(i))$, thus darker nodes are given a slower rate and more importance in both mappings.

Using basic rates, we can pick out many details at small times, finding larger groups as time increases, including some of the small patterns on the balloons. The background is

[1]from the website: http://www.iconarchive.com/show/christmas-icons-by-mohsenfakharian/balloons-icon.html

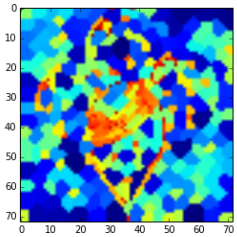


Figure 21: Regular Infomap segmentation of the balloons image with $\sigma_x = 1.2$, $\sigma_i = 10$, and $d_{max} = 1$

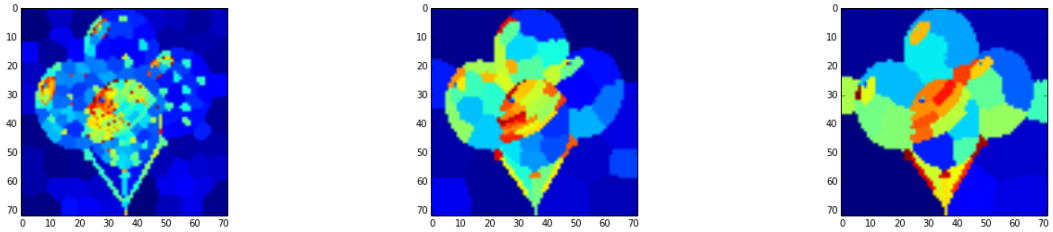


Figure 22: Continuous-time Infomap segmentation with basic rates and $\sigma_x = 1.2$, $\sigma_i = 10$, and $d_{max} = 1$. From Left-to-Right: time=0.2, 1, and 2

still over-partitioned, but it seems that even for a simple division in the importance given to pixels we can find good partitions with finer details in areas assigned lower rates.

For different values of the parameters and rates we can find more detailed, but highly over-partitioned images from the regular Infomap, as shown in Figure 23. Using inverse grayscale rates we can pick out many details even at large times, resulting in some over partitioning in areas with lower rates, but little or no over-partitioning in the high-rate background. The middle row shows extreme over-partitioning, into groups of only one pixel, that we should count against the quality of the overall partitioning. The middle row was created using the 1-pixel partitions of the bottom row, but still has some relevance to the top

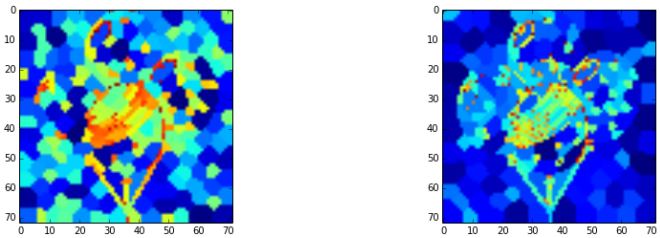


Figure 23: Regular Infomap segmentation of the balloons image with Left:$\sigma_x = 1.2$, $\sigma_i = 10$, and $d_{max} = 2$. Right: $\sigma_x = 2$, $\sigma_i = 3$, and $d_{max} = 3$

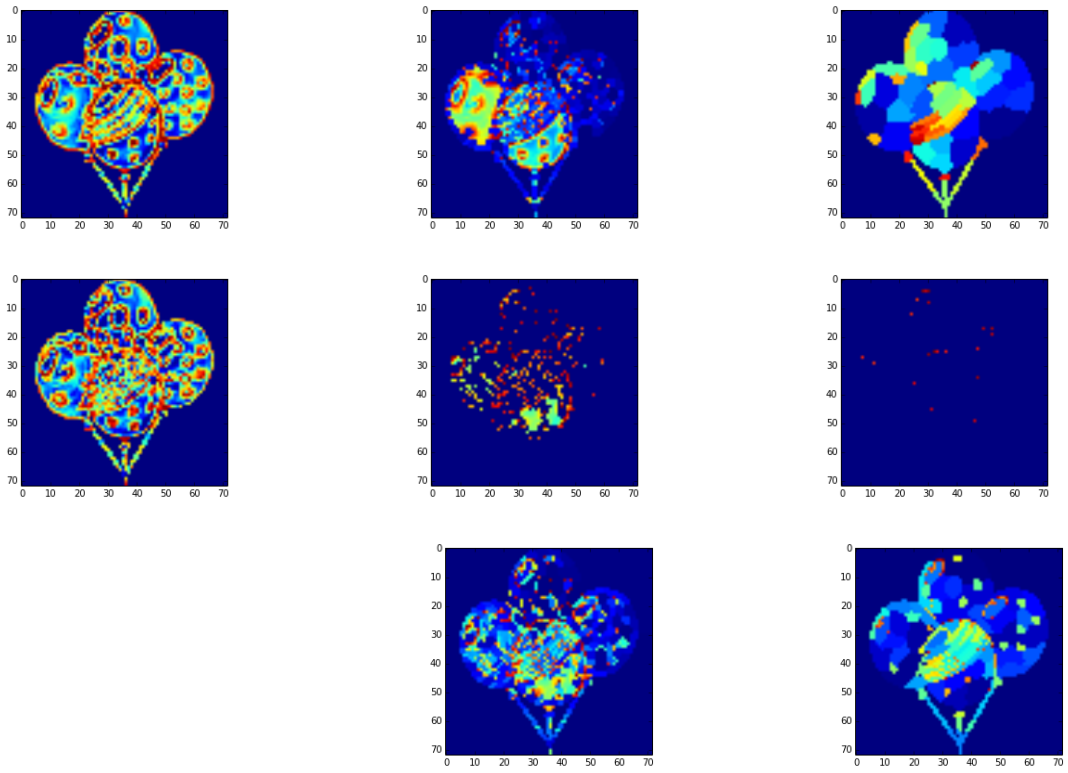


Figure 24: Continuous-time Infomap segmentation with inverse grayscale rates. Top: $\sigma_x = 1.2$, $\sigma_i = 10$, and $d_{max} = 2$. Middle: $\sigma_x = 2$, $\sigma_i = 3$, and $d_{max} = 3$ - only communities of size 1. Bottom: $\sigma_x = 2$, $\sigma_i = 3$, and $d_{max} = 3$. From Left-to-Right: time= 10, 100, 1000 (Bottom only 100 and 1000)

row (please forgive the missing figure on the bottom which was simply forgotten during the production of these figures). At a large time step of 1000, we have few single-pixel partitions, although some over-partitioning of the balloons remains.

Note, some of the balloon partitions have similar color to the background, but are in fact distinct. It is interesting to note that most of the small communities occur along boundaries in the gray-scale of the image. This is a useful property, as it means that nodes forming bridges between two areas of different rates are separated into boundary communities using this method, rather than encouraging the grouping of separate groups of rates together. We may understand this because random walkers from the lower-rate community are unlikely to visit a higher-rate boundary node, while random walkers from the higher-rate community are more likely to traverse the boundary node into the lower-rate community rather than stay within the higher-rate community. This would increase the within-community and leaving entropies respectively if the boundary node were included in either community. A more systematic investigation of various parameters and more examples are needed to draw firm conclusions, and is underway, although it will take some time, given that each continuous-time partition requires about an hour to compute on a modern server given the current implementation of the matrix exponential and Infomap algorithm.

## 6.2 Information Retrieval

In information retrieval tasks where a user is recommended papers in response to a query, the documents are typically presented in a ranked list. Could we instead return groups of related documents individually tailored to a user's query? By tailoring the rates on nodes in citation network we can tailor the found communities in response to a user's query and profile. For example, a scientist specializing in information retrieval researching a subject related to the fields of information retrieval, network analysis, and economics might desire a finer grained view of information retrieval into sub-topics regarding ranking, evaluation, feedback, etc., while economics and network analysis remain as large communities. An economics researcher might instead desire a finer grained view of the economics field while grouping all documents in the information retrieval and network analysis fields into a larger computer science field. Since lower-rate nodes tend to be broken into smaller communities, we might accomplish this focusing of citation network communities by assigning highly ranked nodes with lower rates.

As an initial investigation we use a citation network of 41,370 publications from 111 journals and 1,442 conference proceedings or workshops on computer science mainly from the ACM Digital Library, provided by Xiaozhong Liu, Ph.D. To rank nodes we use a probability distribution over all publications in response to a query, given through a method involving PageRank with topic modeled priors over vertices and edges, described in Liu, Zhang, and Guo's paper [12]. The distribution assigns higher probabilities to more relevant papers, and lower probabilities to less relevant papers. Because of memory limitations, the network we examine is the subgraph induced by the 10,000 most relevant publications. We have only had time to examine a single example query.

Two mappings from probabilities to rates were examined. *Basic citation rates* simply

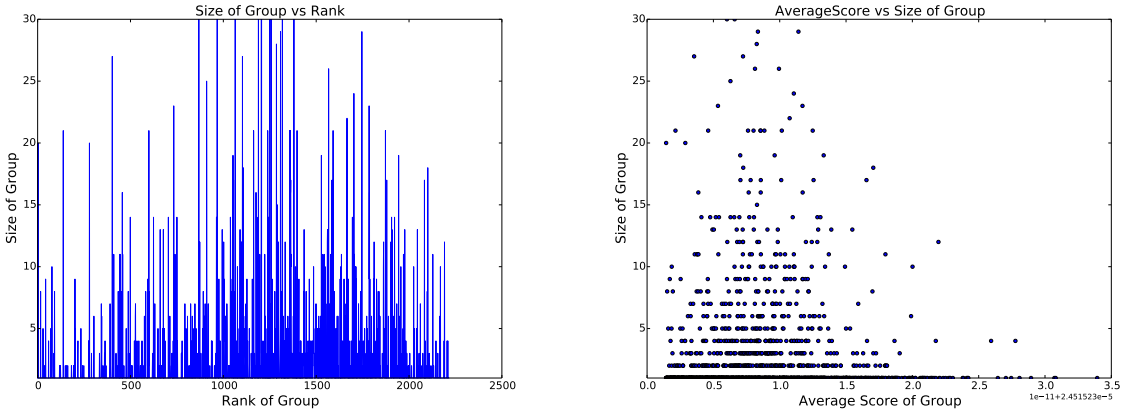



Figure 25: Regular Infomap: Left: Size of publication community vs rank, Right: Size of publication community vs average probability score

assign a rate of 1 to the 1,000 most relevant publications, and a rate of 100 to the rest. *Exponential rates* first assign a z-score to each node according to the probabilities assigned to them, where for publication $i$, $z(i) = \frac{p(i)-\mu_{10000p}}{\sigma_{10000p}}$, where $p(i)$ is the probability distribution over publications, and $\mu_{10000p}$ and $\sigma_{10000p}$ are the mean the standard deviation of the probabilities of the top 10000 publications respectively. The exponential rates are then given by $e^{-z(i)}$. In all cases groups are ranked according to the average probability assigned to nodes in the group, such that the higher the average probability of the group the higher the rank.

The size of communities found by the regular Infomap algorithm is shown in Figure 25 against rank and average probability score. The size of the communities is not well ordered by the rank of the community, although communities with high average probability scores tend to be smaller (lower scoring nodes reduce the average score of larger communities). Figure 26 shows the same comparisons of community ranks using basic citation rates. The size of communities is better ordered according to probability score rank, although very low ranks also create small communities. The ordering is much more impressive given exponential rates, as shown in Figure 27. In all cases however, there is an over-partitioning of many communities into single publications, while at least one community in all cases (outside of figure views) reaches a size of hundreds or over a thousand publications. The over-partitioning of the network may be ameliorated at larger times yet to be examined. Although the size of very large communities should not be easy to reduce, it occurs at a rank low enough not to be too important. Investigations of more queries and different times are underway, however improvements to the computation of the matrix exponential would be very beneficial to this analysis. In addition, a good means to judge the quality of the found publication communities is needed to properly evaluate this method.

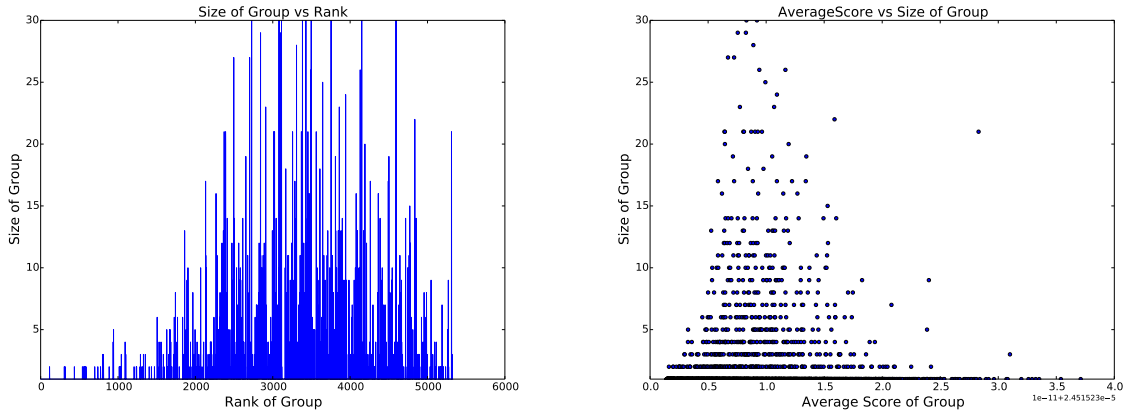



Figure 26: Markov Rate Infomap basic citation rates: Left: Size of publication community vs rank, Right: Size of publication community vs average probability score

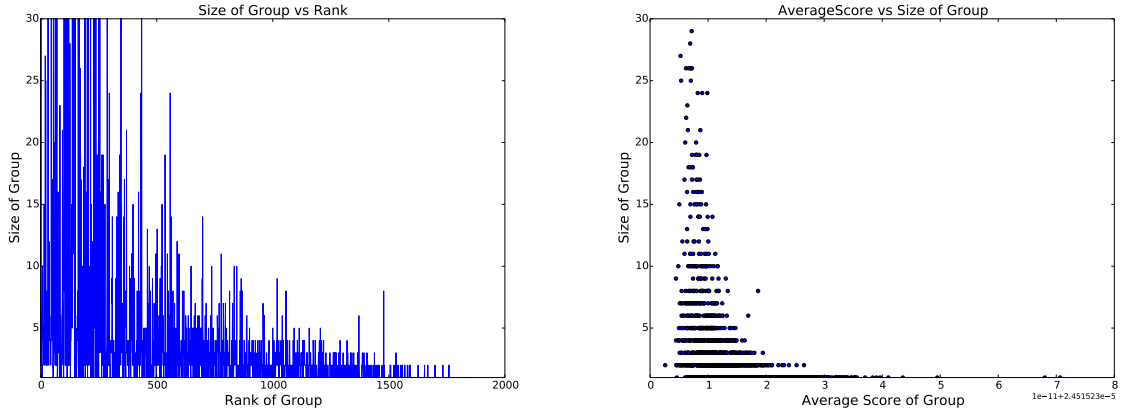



Figure 27: Markov Rate Infomap exponential rates: Left: Size of publication community vs rank, Right: Size of publication community vs average probability score

### 6.3 Future Directions: Disease Networks

Network medicine is an emerging field in the study of diseases. A variety of data has been compiled on molecular interaction networks within the human body, including protein-protein interaction networks, metabolic networks, and gene regulatory networks, and the literature on how diseases behave on these networks is growing. The networks can be broken up into three types of modules assumed to be related and overlapping: topological modules found through network community detection, functional modules of nodes with similar biological functions in network neighborhoods, and disease modules of nodes whose disruption results in particular disease phenotypes [4]. Diffusion models involving random walks or continuous time kernels have already been used to successfully predict disease genes [9] [20]. As a topological diffusion method that can incorporate external data, the Markov Rate Infomap may be useful in identifying disease modules or detecting disease genes, thus helping to direct hypotheses to test experimentally.

Towards the end of testing this method we have begun a collaboration with Amitabh Sharma, Ph.D., an Associate Scientist at the Channing Division of Network Medicine, Brigham & Women's Hospital and associated with BarabasiLab. He has provided us with a protein-protein interaction network of 14,215 as well as statistics and corresponding p-values for expression differences in certain diseases, such as asthma. However, this work is still in early stages and there are no results to report yet.

## 7 Conclusion

Markov time sweeping is an extension of community detection algorithms based on discrete random walks into a continuous time model. This has been applied to the popular community detection algorithms of Modularity and Infomap by replacing their discrete transition matrices with matrix exponentials. By extending this method with variable rates on nodes, we have shown that a Markov Rate Infomap can be focused to regard areas of the network with finer resolution. A number of practical concerns arise in the implementation of the exponential of a matrix, but possible solutions have been outlined, and if these difficulties can be overcome the method holds some promise for application to large networks. In particular, the method holds promise for applications in which incorporation of node information with the topology of the network is desirable, whether it is user preference in information retrieval or experimental results in disease module detection.

# References


[1] Awad H Al-Mohy and Nicholas J Higham. A new scaling and squaring algorithm for the matrix exponential. *SIAM Journal on Matrix Analysis and Applications*, 31(3):970–989, 2009.

[2] David F Anderson. Introduction to stochastic processes with applications in the biosciences, 2011.

[3] Alex Arenas, Alberto Fernandez, and Sergio Gomez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, 2008.

[4] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nature Reviews Genetics*, 12(1):56–68, 2011.

[5] Arnaud Browet, P-A Absil, and Paul Van Dooren. Community detection for hierarchical image segmentation. In *Combinatorial Image Analysis*, pages 358–371. Springer, 2011.

[6] Santo Fortunato and Marc Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[7] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed 2014-08-28].

[8] Tatsuro Kawamoto and Martin Rosvall. The map equation and the resolution limit in community detection. *arXiv preprint arXiv:1402.4385*, 2014.

[9] Sebastian Köhler, Sebastian Bauer, Denise Horn, and Peter N Robinson. Walking the interactome for prioritization of candidate disease genes. *The American Journal of Human Genetics*, 82(4):949–958, 2008.

[10] Renaud Lambiotte, J-C Delvenne, and Mauricio Barahona. Laplacian dynamics and multiscale modular structure in networks. *arXiv preprint arXiv:0812.1770*, 2008.

[11] Renaud Lambiotte and Martin Rosvall. Ranking and clustering of nodes in networks with smart teleportation. *Physical Review E*, 85(5):056107, 2012.

[12] Xiaozhong Liu, Jinsong Zhang, and Chun Guo. Full-text citation analysis: A new method to enhance scholarly networks. *Journal of the American Society for Information Science and Technology*, 64(9):1852–1863, 2013.

[13] Cleve Moler and Charles Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM review*, 45(1):3–49, 2003.

[14] Martin Rosvall, Daniel Axelsson, and Carl T Bergstrom. The map equation. *The European Physical Journal-Special Topics*, 178(1):13–23, 2009.

[15] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[16] Michael T Schaub, Jean-Charles Delvenne, Sophia N Yaliraki, and Mauricio Barahona. Markov dynamics as a zooming lens for multiscale community detection: non clique-like communities and the field-of-view limit. *PloS one*, 7(2):e32210, 2012.

[17] Michael T Schaub, Renaud Lambiotte, and Mauricio Barahona. Encoding dynamics for multiscale community detection: Markov time sweeping for the map equation. *Physical Review E*, 86(2):026112, 2012.

[18] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[19] Roger B Sidje. Expokit: a software package for computing matrix exponentials. *ACM Transactions on Mathematical Software (TOMS)*, 24(1):130–156, 1998.

[20] Oron Vanunu, Oded Magger, Eytan Ruppin, Tomer Shlomi, and Roded Sharan. Associating genes and protein complexes with disease via network propagation. *PLoS computational biology*, 6(1):e1000641, 2010.