# Kalman Filter Driven Estimation of Community Structure in Time Varying Graphs

Lisa J.K. Durbeck and Peter Athanas
*Virginia Tech*
*Bradley Department of ECE*
*Blacksburg, Virginia USA*
{ldurbeck,athanas} at vt.edu

***Abstract*—Community detection is an NP-hard graph problem that has been the subject of decades of research. Moreover, efficient methods are needed for time-varying graphs. In this paper we propose and evaluate a method of approximating the latent block structure within a time-varying graph using a Kalman filter. The method described breaks a stream of graph updates into samples of sufficient size, each one forming a graph $G_t$, and has the desirable feature that it accurately updates its representation of the latent block structure using a relatively small amount of information: the prior $t-1$ predicted block structure and the current datastream sample $G_t$. This paper details the underlying system of linear equations, used here to represent community detection, that achieves 97% accuracy estimating the latent block representation as the community structure changes. This is demonstrated for synthetic graphs generated by a hybrid mixed-model stochastic block model from the DARPA/MIT Graph Challenge with time-varying block structure.**

***Index Terms*—Computation (stat.CO); Graph partition; graph sparsification; stochastic block models; community detection; time series data; streaming graphs; Kalman filter**

## I. Introduction

Graphs are often used to represent connections between elements of a set, such as similarity or kinship between elements; a family tree or social network are examples of sets that can be represented as the nodes and edges of a graph. It is possible to identify groups of nodes that are heavily connected among themselves, but sparsely connected to the rest of the graph; these groups are often characterised as communities, modules or partitions. They occur in a wide variety of networks.

Detecting communities is a fundamental and highly relevant problem in network science with multiple applications. It reveals non-trivial internal network organisation at a coarser grain and uncovers relationships between nodes that may not be easily accessible from direct empirical tests. Detecting communities also helps to better understand the properties of dynamic processes taking place over a network; for instance, the diffusion of ideas within and between communities is considerably affected by the structure of the network [1].

A *graph partition* generated by a community detection algorithm is a more compact encoding of the graph that groups nodes along boundaries that are present in the data, preserving closely knit communities by some measure. In the usual definition of community detection, and in our case, the boundaries are chosen that result in more intramural than extramural connectivity. The resulting *partition* captures the community structure of the graph using a several-fold reduction in the number of edges. The optimal grouping is NP-hard; thus, heuristic approximations are utilized. It is desirable to have a simple, fast and accurate graph partitioner; given the ongoing increase in data collected and the growing size of problems being solved, there is continual need for improvement in graph partitioning, coming not only from algorithm development and fast implementations, but also from computer architecture and custom hardware.

This paper pertains to community detection within *time-varying graphs*. These are graphs that possess an additional dimension of temporal variation in set elements—nodes or edges—corresponding to some form of dynamics that introduce temporal variation to the graph over time, such as elements dropping out, or new connections being formed. In such a graph there is generally not a single partition that holds for all time, but rather a time series of partitions that; each represents the community structure for a period of time and then gives way to a better representation of the changed graph.

The problem is how to generate a time-varying partition corresponding with the time-varying community structure of the graph, and to do so with only the information present in the current time window, which we treat as an adjacency matrix containing nodes and edges that is relatively sparse in its informational content. Changes in the adjacency information are used to identify the latent or hidden community structure. Further, the method used should give insights into the temporal dynamics of the graph and be more efficient than the DARPA/MIT Graph Challenge (DGC) baseline. This paper seeks effective means of approximating dynamic community structure.

We investigate the use of Kalman filtering (KF) [28] to track temporal variation in the graph partition and update community labels. KF is an efficient recursive method of estimating the internal state of a linear dynamic system from a series of noisy measurements [12]. We use KF to produce time-varying graph partitions; this method takes advantage of the information from all prior partitions along with the current graph information (edge and node insertions and deletions) to accurately update its representation of the latent block structure of the graph.

The main contributions of this paper are: 1) formulating time-varying graph partition to allow Kalman filtering of graph node and edge updates; 2) infrequent but effective updates of previous node label (group assignment) decisions during the KF process; and 3) comparative evaluation of the approach to a known baseline in Streaming Stochastic Block Models (DGC SSBM) to contextualize the results.

The paper is organized as follows. The Approach section describes linear dynamic system equations, how the latent information is represented, and the parts of the KF model unique to this work. The Methods section outlines the KF setup and parameters. The Experiments and Results section describes the time-varying graph datasets and how they were generated, defines the evaluation criteria, and presents results. The Related Work section provides the larger context for this work. The Discussion and Future Work section summarizes the findings and contributions of this work and how it could be further developed.

## II. Approach

For graphs in which the node and connection information is dynamic, the community structure can change over time, requiring a method of update of the communities. The envisioned datastream partitioner observes incoming graph updates and maintains the community structure of the graph while minimizing resource utilization and latency. The hypothesis tested here is that KF can be used to observe the latent community structure from observed node and edge updates, given an appropriate observation and state transition model. This hypothesis is tested by constructing a KF model appropriate for community detection and comparing its performance to the DGC baseline algorithm for streaming SBM graphs.

The complementary goals motivating this work in general are maximizing the graph analysis benefit while minimizing the cost, for time-varying graphs. Minimizing the cost implies minimizing the number of graph partition calls or invocations performed on the datastream, because partitioning is computationally expensive, introduces latency into real-time observation of the graph, and can require access to the entire graph, depending on the algorithm used. The KF computation is itself computationally expensive; however, there are linear-time approximations for KF that do nearly equally well [23]. In the present work, the focus is on the most straightforward implementation of KF rather than the most performant.

### A. Baseline Algorithm (DGC SSBM)

The baseline algorithm for DGC by Kao [13] does not make use of any information gleaned from the time series to trigger invocation of the partitioner. In the *emergingEdges* example, the partitioner is simply run with each incoming data stage; each stage consists of a 10% addition of edges sampled at random from a larger graph $G$ without replacement.

This approach to streaming incurs no cost in *assessing* the time series for changes relevant to community structure but at the expense of unnecessary costs incurred partitioning when it is the case that the changes to the community structure are insignificant—as is true for nearly every stage of the *emergingEdges* graphs [6]. Also, at timestep $k$ the baseline algorithm partitions using a graph that combines timesteps $G_0$ through $G_k$, which is suitable for the stationary examples in the *emergingEdges* dataset, but results in an algorithm that averages any temporal dynamics together. This approach provides little sensitivity with which to capture dynamics, and too little control over the weighting of "old" data, to be highly useful for data that exhibits nonstationarity.

### B. Kalman Filter-based Algorithm (KF)

Kalman filtering is a form of linear quadratic estimation commonly used in signal processing and controls. It uses a series of measurements observed over time, containing statistical noise, measurement-process noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single (noisy) measurement alone, by estimating a joint probability distribution over the variables in each timestep. When used with its *optimal gain* update mechanism, it produces a minimum mean-square error estimate [12].

In its application here, the measurements, or direct observations, are the graph updates present in the time series. These can be node or edge insertions and deletions. The KF *observation model* and *state transition model* and other aspects of the algorithm use these observations or measurements to estimate the latent variables. In its application here, the unknown latent variables are the community memberships of nodes and the overall community structure. Given a time series of graph updates, we use KF to update the nodal community membership probabilities. The goal is to construct a KF filter that will estimate group membership accurately from the measured observations, even as the graph and its community structure change in the time series data, for hybrid mixed-membership stochastic block model time-varying graphs, and evaluate the results.

The challenge is to devise an appropriate representation of the problem dynamics, which is not an insignificant challenge. With a suitable representation of the dynamics, the KF method learns a *gain* that is adapted to and improved upon from repeated recursive applications on the time series data that minimize mean squared error.

*1) Representation of the Latent Information:* While the aim is to discover and track the community membership probabilities for each node, the representation of the latent block information used here is not, for instance, an $N \times m$ matrix with one entry per node $n$ and community $m$, which has the downside of presuming knowledge of the number of communities. Rather than presume a certain number (or maximum number) of communities, we instead assess the probability of co-membership for each pair of nodes, from which the block structure can be recovered. For a graph $G$ with $N$ nodes, at each update cycle we update the $N \times N$ matrix of pairwise probabilities $p^{\{u,v\}}$ that nodes $u$ and $v$ are members of the same group by using the prior probability and the relevant subset of edge information. For our purposes, this relevant set contains only current edge information so as to evolve the pairwise probabilities along with the time-varying graph; however, the choice of sample can be modified to suit the analytic objective. Recursion within the KF algorithm utilizes all prior pairwise probabilities in its estimation, so they do factor into the present time period, but they are not explicitly retained.

The precedents for representing the latent block structure as pairwise co-membership probabilities are prior work by Reichardt and Bornholdt [24], who used Monte Carlo methods to approximate the pairwise co-memberships; and by Ferry [7], who formulated an efficient Bayesian method of approximating the pairwise group co-membership probability and produced a successful demonstration of its use partitioning static graphs within Lancichinetti's LFR benchmark [16], [17] using the co-membership probabilities as a starting point for an agglomerative clustering technique. The KF observation model we constructed was inspired by the pairwise probabilities update formulated by Ferry [7], although the formulation is fundamentally different on account of the constraints of the KF algorithm. The benefit of establishing a method of assessing co-membership probabilities within a KF-based framework is that this leverages KF methods of estimation optimization to allow for monitoring time-varying graphs and their changing community structure.

Computing $p^{\{u,v\}}$ exactly is expensive; instead, Ferry's work suggests this can be conveniently approximated using only the small subset of graph information $\epsilon_{uv}(G)$ that is most relevant to the question of co-membership, forming an estimate based on the presence or absence of edges only on the vertex pairs that intersect $u, v$. From this they derive the likelihood ratio under the competing hypotheses that $u$ and $v$ are in the same group and in different groups, producing an approximation $\hat{p}^{\{u,v\}}$ of the posterior probability $p^{\{u,v\}}$ given the prior probability, and considering the contributions from edges in $\epsilon_{uv}(G)$.

This formulation allows update to the co-membership probabilities matrix. Additional work is required to derive a suitable clustering from this information and assign group memberships, using the co-membership probabilities as a starting point for something such as an agglomerative clustering technique. To construct partitions from these co-membership probabilities, Ferry used average-linkage clustering, defining the distance between clusters to be the average over $p^{\{u,v\}}$. This approach gave highly accurate clusters based on NMI as compared with other commonly used approaches on Lancichinetti's LFR benchmark graphs—which have power-law distribution of group size and node degree [7], [8], [16]. Their results provide an important basis for the validity of this approach for updating co-membership probabilities given a sequence of edge updates.

*2) Linear dynamic system:* The key piece of the algorithm that is unique to the problem of community detection is not the representation of noise or the method of linear estimation, but rather the method of update to the hidden state given observations. In a two-stage KF, the update is provided during the *prediction* phase by the state transition model (and process model) and within the *update* phase by the observation model. Taking the term names from Wikipedia [28], the KF prediction model assumes the true state at time $k$ is evolved from the state at $(k-1)$ according to

$$\mathbf{X}_k = \mathbf{F}_k \mathbf{X}_{k-1} + \mathbf{w}_k$$

where $\mathbf{F}_k$ is the state transition model and $\mathbf{w}_k$ is the process noise drawn from a zero-mean multivariate normal distribution with covariance $\mathbf{Q}_k : \mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)$ and at time $k$ an observation (or measurement) $\mathbf{Z}_k$ of the true state $\mathbf{X}_k$ is made according to

$$\mathbf{Z}_k = \mathbf{H}_k \mathbf{X}_k + \mathbf{v}_k$$

where $\mathbf{H}_k$ is the observation model that maps the true state space during the prediction phase, $\mathbf{X}_k$, into the observed space, and $\mathbf{v}_k$ is the observation noise, which is also assumed to be zero-mean Gaussian white noise with covariance $\mathbf{R}_k : \mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$.

In order to use KF, one must model the process in accordance with its framework. This includes specifying the following $N \times N$ matrices, for each timestep $k$:

- $\mathbf{Z}_k$, the current sample of measurements or observations;
- $\mathbf{X}_0$, the initial state or its mean and covariance;
- $\mathbf{F}_k$, the state-transition model;
- $\mathbf{H}_k$, the observation model;
- $\mathbf{Q}_k$, the covariance of the process noise;
- $\mathbf{R}_k$, the covariance of the observation noise;

The current observations $\mathbf{Z}_k$ are an $N \times N$ matrix that is the adjacency matrix of the graph formed in the current timestep, $G_k$. It contains ones where there is an edge between nodes $u$ and $v$ and zeros elsewhere, with the exception that the diagonal entries are set to the relative incidence of extramural edges, $s$ which is the average ratio of between-group edges to within-group edges, calculated from $\mathbf{X}_0$ and $\mathbf{B}_0$. The diagonal entry value was chosen in order to pull out and appropriately scale specific terms in the $\mathbf{H}_k$ observation model when $p^{\{u,v\}} = 0$.

The initial state $\mathbf{X}_0$ can either be provided exactly or assumed to be $\mathbf{X}_0 \cong (\mu_0, \sum_0)$, with initial state mean $\mu_0$ and initial state covariance $\sum_0$. We provide the state mean of the co-membership probabilities exactly, from the known block partition $\mathbf{B_0}$ at the first graph stage $\mathbf{X}_0$. This approach assumes that $\mathbf{B_0}$ is generated by invocation of a partitioner.

$\mathbf{X}_0$ is an $N \times N$ matrix that contains $p^{\{u,v\}} = 1.0$ for every node pair that are in the same group, and $p^{\{u,v\}} = 0.0$ elsewhere. The diagonal entries of $\mathbf{X}_0$ are set to the relative probability of extramural edges $s$, which is estimated from $\mathbf{B_0}$ and $\mathbf{Z}_0$. We set $p^{\{u,u\}}$ to a medium-to-low value; this lets $\mathbf{Z}_1$, the adjacency matrix, begin to have ones when the observation model has them, while also letting the observation model then influence other values $\mathbf{Z}_{uv}$ where $u <> v$. This allows some cause $\mathbf{H}_{uv}$ other than noise to create ones in the adjacency matrix for $\mathbf{Z}_{uv}$. The covariance or standard deviation for the initial state mean is provided in $N \times N$ matrix form as well as the value for each entry.

For the state-transition model for each timestep, $\mathbf{F}_k$, we use the identity matrix $\mathbf{I}$ with zero offsets. This is somewhat analagous to saying that the initial prediction for state $\mathbf{X}_{k+1}$ is the prior state plus Gaussian white noise, which is then fitted to the observations and residuals.

The observation model for each timestep $\mathbf{H}_k$ is constructed with most entries set to the probability of an edge in general, which is defined as $\rho$, times the probability that $\{u, v\}$ are in the same group. The likelihood of any edge $\{u, v\}$ existing is the sparsity of the graph $G_k$ at timestep $k$ which is defined for an undirected graph as $\rho = E/(N \times (N-1))$, if all positions $u, v$ are equally likely. The diagonal entries are set to $\rho$ and the off-diagonal entries are reduced by their relative role in contributing to a one or zero in $Z_k$, which they have mainly by being co-members with either $u$ or $v$, akin to the neighborhood assessment $\epsilon_{uv}(G)$ performed by Ferry [7]. Given that the main contributors to the values in $Z_k$ should be dominated by $p^{\{u,v\}}$ and could be overpowered by a lot of other nonzero probabilities along the $u$ or $v$ column of $X_k$, we scale the off-diagonal entries by a factor of $m$, the number of blocks in the graph, since the probability of $\{u, v\}$ being in the same group is roughly $1/m$, if all groups were of equal size and therefore equally likely.

$\mathbf{Q}_k$, the covariance of the process noise, and $\mathbf{R}_k$, the covariance of the observation noise, are fitted to the set of observations for each time-varying graph using the expectation likelihood maximization function within the `pykalman` python package [5].

## III. Experiments & Results

Experiments utilize hybrid mixed membership stochastic block model (HMMB) graph generation from the DGC graph generator by Kao [13] combined with Wanye's parameterization interface [27]. The DGC graph generator is based on the work of Kao, Gleyzer, Chung, Karrer and Newman, Peixoto, and others [11], [13]–[15], [21]. These generative models produce graphs that statistically correspond with real-world graphs and the corresponding community membership information. The second-generation DGC graph generator based on Peixoto [21] implements Karrer and Newman's classic stochastic blockmodel [15] broadened to represent a greater range of graphs using a hybrid mixed-membership model (HMMB) of Kao [14]. HMMB graphs are a composite of three identifiable network models as described in Kao [14] that allow time-varying connections between members. An edge is drawn from a Poisson distribution with rate $\lambda_{ij}$,

$$a_{ij} \sim \text{Poisson}\left(\lambda_{ij} T\right),$$

over some time duration and frequency $T$. This is similar to Karrer and Newman's model [15]. HMMB models the rate $\lambda_{i,j}$ of interaction from each node $i$ to $j$ as

$$\lambda_{ij} = (\lambda_i \lambda_j) \times \left(\pi_i^{\mathrm{T}} B \pi_j\right) \times I_{ij},$$

by combining three canonical network models in this rate—the power-law degree distribution and small-world network model of Chung $(\lambda_i \lambda_j)$, the mixed-membership stochastic block model of Karrer and Newman $\left(\pi_i^{\mathrm{T}} B \pi_j\right)$, and the Erdôs-Rényi sparsity model $I_{ij}$—HMMB models properties of real-world networks.

The DGC dataset for SSBM contains various sets of graph structures that are self-similar along that range in other respects such as sparseness and degree distribution, and probability of inter- and intra-group edges [4], [13] and that possess known ground truth as to the group membership for each node.

The first set of experiments uses graphs we constructed to be similar to the *emergingEdges* graph dataset for DGC [4]—which gradually reveals a graph in stages of edges picked at random. These graphs are more or less stationary in community structure over time. We construct a sample set by repeated graph generation while conserving the block membership vector

The second set of experiments uses another synthetic dataset we similarly constructed that has time-varying block membership that exhibits nonstationarity. It has the same basic characteristics as the DGC datasets and was constructed using the core elements of the DGC graph generation process, but here the edges are drawn from a dynamic model within which the block structure is changing over time. This allows for a truer test of the ability to monitor change in community structure than possible with the original DGC graphs. The partition of such a graph into nonoverlapping communities is less straightforward, as it can change over time. We constructed an undirected binary graph representation containing a series of files, one for each timestep $0 - T$.

We modified the DGC graph generator to return the block membership for each node and ran it within a loop that evolved the block structure to perform operations like block merges or splits. For a block merge such as the one pictured in Figure 1, we selected two groups based on their size to draw members from, along with a hidden group with initially no members, and randomly chose elements from either to migrate to the hidden group at each timestep.

The general experimental goals are to establish the quality of the results and the computational efficiency of the approach. We evaluate the computational efficiency of our approach in terms of its speedup over the baseline. Speedup is typically defined in terms of latency as $t/W$ of the new algorithm over $t/W$ of the old, where $t$ is the total execution time and $W$ is the workload. Here we report $t$ in the number of cycles, where a cycle is defined as a call to the partitioner $f$. For simplicity in this calculation we assume the same partitioner is used for either. Since our approach is used here to prevent calls to the partitioner, cycle time speedup captures these performance gains. However, because speedup is so intertwined with the graph properties, dependent on the degree of community structure dynamics within the problem, there is no general trend. We report speedup times in terms of the graph dynamics for the sample set.

Experiments were carried out using Python implementations of the algorithms on a server-class `CentOS` machine with 512 GB of memory. We evaluated the accuracy of our approach by

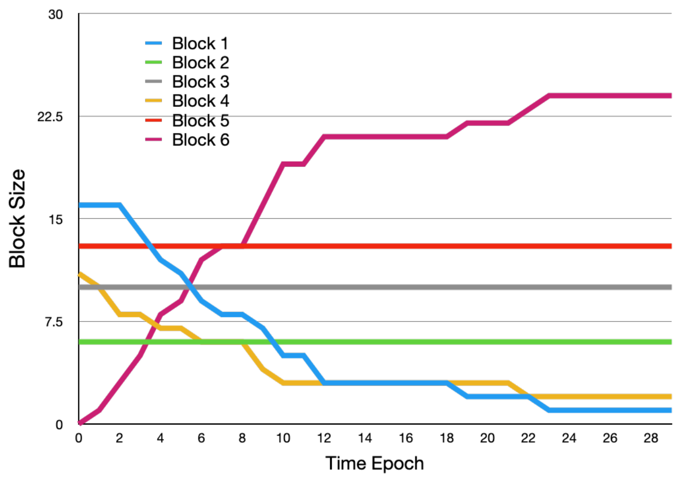

Fig. 1. A time-varying graph from the dynamic graph generator, showing growth and reduction in group size. The size of each group is represented as a colored line over x-axis time. Nodes with block membership in Block 1 and 4 are gradually added to a new group, Block 6.

comparing the true and estimated co-memberships produced within the KF predicted state. The KF filter update provides $\mathbf{X}_k$ for each time period. We construct the corresponding hidden true state under the assumption of non-overlapping groups using the group memberships known at graph generation time by setting $T_k$ element $\{u, v\} = 1$ where nodes $u$ and $v$ are in the same block $b$ of the partition-set $B$ and zero elsewhere. Accuracy is defined as the fraction of nodes correctly identified as co-members, which is the number matched of the algorithm's results with the true labels over the total number of ones in the true co-membership probability matrix constructed for each timestep $k$ from the block partition $\mathbf{B}_K$. The mean accuracy observed across all experiments was 97.4% $(n = 163072)$.

Speedup for stationary graphs similar to the *emergingEdges* dataset is observed as roughly the number of time periods $T$ over the settling time of the KF estimation – at which point the pairwise Euclidean distance of $\mathbf{X}_k - \mathbf{X}_{k+1}$ from timestep to timestep is well below our initial setting for the distance metric that indicates a need to partition, $\kappa$. For a thirty-timestep example, this results in a speedup of roughly 10x.

We note that speedup is highly dependent on the dynamics of community structure, and that for the example in Figure 1, there is virtually no speedup depending on the setting for $\kappa$, the sensitivity of which is driven by the analytic problem. In general, speedup is expected to be $T$ divided by the ratio of two sums: the number of triggering $\sum_t \Delta B < \kappa$ versus nontriggering $\sum_t \Delta B \geq \kappa$ timeperiods $t$.

## IV. Related Work

A summary of approaches to community detection or clustering can be found in Fortunato [9]. There are many methods, often with different objectives. Global objective functions for clustering are not yet well characterized, and many studies are still needed to understand the underlying mechanisms that are responsible for the interactions of nodes in different graphs. See for example [3], [9], [20], [22].

Tracking changes in community structure in time-varying graphs, or dynamic graphs, has been a subject of study for some time. A number of time-varying properties of dynamic graphs such as shrinking diameter have been identified [2], [18]. The problem of minimizing the effort of maintaining accurate community structure for dynamic graphs was first addressed at least twenty years ago in the domain of adaptive finite element mesh generation, by Ou and Ranka [19], who observed that, as the graph mesh is updated, the number of actual updates to the partition assignment of the mesh elements at any given time is small relative to the size of the graph.

In prior work, we showed that, for the stationary DGC *emergingEdges* dataset, accuracy could be maintained with a partition update rate at a frequency based on the logarithm of the current graph sizes [6]. This, however, does not hold for true time-varying dynamic graphs. This also leaves open the graph update scenarios such as call or email logs, in which edge and node updates are not random, and the pattern of updates, or even a small set of changes, can vary the community structure in ways that may be of importance in understanding the data.

The KF approach has been successfully applied to tracking node degree changes in opportunistic mobile transmission networks by Soelistijanto [25]; exploring disease spread over a social network given an epidemiological model of transmission dynamics by Wang [26]; and dynamic community detection by Fu [10] and Zhang [29], who both track an evolving block membership. Fu showed improvement over a dynamic model of node's funtional roles in existing groups using KF with logistic normal priors [10].

Zhang showed improvement over periodic application of static methods for a community detector based on non-negative matrix factorization (NMF) [29]. For an approximation of an adjacency matrix $X$ as the multiplication of two lower-rank matrices $X \simeq WH$, they used KF to update the latent factors of matrix $H$, which gives the cluster assignments for nodes, and interleaved this with a joint alternating least squares algorithm that learned the dynamics of $W$, which gives the centroids of the groups.

Our approach differs from that of Fu or Zhang in that we are observing a different phenomenon in order to simplify a different class of algorithms. That is, we are attempting to represent not $H$ in NMF but the dynamics of the changes in co-membership probabilities across time. This results in a different model upon which we are basing our dynamics of observation and state transition. At the algorithmic level, our approach has the benefit over NMF of not setting the number of clusters *a priori* as is necessary in NMF, but rather discovering the number of clusters from the data.

An approximation of the co-membership probability matrix $p^{\{v,w\}}$ was sought by Reichardt and Bornholdt [24], who estimated its value by a Monte Carlo sampling of the partition space, and by the faster Bayesian approximations of Ferry [7]. Their demonstrations of these concepts, however, were limited to the construction of a working partitioner for static graphs [7]. We base the filter's hidden state representation on estimating the pairwise block co-membership probability that was demonstrated successfully for partitioning static graphs by Ferry [7]. Here, we adapt the approach to better fit the context of monitoring time-varying graphs and their evolving communities. Doing so within a KF-based framework leverages its methods of estimation optimization.

For their work with static graphs, Ferry used a simple log-uniform prior for the prior probability estimate, and the complete edge information $E$ from the full static graph $G$. In our formulation for KF, the prior probability estimate is not a log-uniform prior but instead given by the previous cycle, i.e. $\hat{p}_{t-1}^{\{u,v\}}$, and we compute the posterior probability on the edge counts $n_1$, $n_2$ and so on in the current timestep sample $G_k$. It is possible for the KF state space to be initialized based upon equal probability of membership in any community, using a log-uniform prior, if it is not known. Here, for simplicity, we assume that we partition the first timestep $G_0$ to provide the initial state and covariance, and subsequently track updates using KF.

## V. Discussion & Future Work

The problem of maintaining an accurate community-structure assessment of relationships in a temporal dataset with time-varying characteristics was examined for dynamic graphs. Use of a technique common in signal processing was explored, namely, the application of Kalman filtering in an online mode to graph updates. While KF at its most basic is a computationally expensive process, linear-time approximations exist that appear to be amenable to this use case [23]. KF was applied to the problem of maintaining a model of node group co-membership probabilities, with an appropriate update model of community structure dynamics.

Invoking a partitioner on stages of a streaming dynamic graph is an unneeded expense if the co-membership probabilities have not changed significantly. Ideally repartition occurs only when the current time window of the time series suggests sufficient change to the community structure that the previous partition no longer captures the characteristics of the data.

Our approach was tested for synthetic graphs with real-world characteristics generated from a hybrid mixed-membership streaming stochastic block model. A baseline dataset of streaming HMM-SBM graphs was augmented by the construction of graphs with stationary and time-varying community structure using the same basic generation techniques but preserving the block membership state and evolving it. These were used to demonstrate effectiveness in maintaining an accurate assessment as compared with the true community structure over time.

Experimental results demonstrate the accuracy and time-performance of this approach to time-varying communities. Cycle-time performance as measured by number of repartitions over the datastream is close to optimal due to the extra work of updating the graph state. This suppresses unnecessary partition cycles done within the DGC baseline spectral method, which constructs a new partition at every time period.

Future work includes expanding the dataset to which this approach is applied to assess how well the model holds, along with an investigation of the fit of linear-time KF approximations to the problem domain and model. Whether this dynamic system model or approach applies to the wider classes of time-varying graphs in general is a harder question than the one addressed here. By careful choice of sample problems we demonstrate whether the approach holds promise. We leave the search for an optimal repartition metric and trigger for this approach as future work requiring a representative sample of graphs as well as a motivating analysis problem.

## VI. Acknowledgements

The authors would like to thank Edward Kao, Frank Wanye and Nicholas J. Macias for helpful discussions and other assistance, and the anonymous reviewers for their helpful comments. This research was supported in part by an appointment to the Intelligence Community Postdoctoral Research Fellowship Program at the National Institute of Standards and Technology (NIST), administered by Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy and the Office of the Director of National Intelligence (ODNI).

## References

[1] Andrea Apolloni, Karthik Channakeshava, Lisa Durbeck, Maleq Khan, Chris Kuhlman, Bryan Lewis, and Samarth Swarup. A study of information diffusion over a realistic social network model. In *2009 International Conference on Computational Science and Engineering*, volume 4, pages 675–682, 2009.

[2] Lars Backstrom, Dan Huttenlocher, Jon Kleinberg, and Xiangyang Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54, 2006.

[3] Vinh Loc Dao, Cécile Bothorel, and Philippe Lenca. Community structure: A comparative evaluation of community detection methods. *Network Science*, 8(1):1–41, 2020.

[4] DARPA/MIT. Data sets | graphchallenge. *Graph Challenge Website https://graphchallenge.mit.edu/data-sets*, 2016.

[5] Daniel Duckworth and Pierre Haessig. Welcome to pykalman, the dead-simple kalman filter, kalman smoother, and em library for python, 2022. [Online; accessed 21-August-2022].

[6] L.J.K. Durbeck and P. Athanas. Incremental streaming graph partitioning. In *2020 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–8, 2020.

[7] James P Ferry, J Oren Bumgarner, and Stephen T Ahearn. Probabilistic community detection in networks. In *14th International Conference on Information Fusion*, pages 1–8. IEEE, 2011.

[8] James P Ferry, J Oren Bumgarner, and Stephen T Ahearn. Probabilistic community detection in networks. In *14th International Conference on Information Fusion*, pages 1–8. IEEE, 2011.

[9] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

[10] Wenjie Fu, Le Song, and Eric P Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of the 26th annual international conference on machine learning*, pages 329–336, 2009.

[11] Vitaliy Gleyzer, Andrew J Soszynski, and Edward K Kao. Leveraging linear algebra to count and enumerate simple subgraphs. In *2020 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–8. IEEE, 2020.

[12] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

[13] Edward Kao, Vijay Gadepally, Michael Hurley, Michael Jones, Jeremy Kepner, Sanjeev Mohindra, Paul Monticciolo, Albert Reuther, Siddharth Samsi, William Song, et al. Streaming graph challenge: Stochastic block partition. In *2017 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–12. IEEE, 2017.

[14] Edward K Kao, Steven Thomas Smith, and Edoardo M Airoldi. Hybrid mixed-membership blockmodel for inference on realistic network interactions. *IEEE Transactions on Network Science and Engineering*, 6(3):336–350, 2018.

[15] Brian Karrer and Mark EJ Newman. Stochastic blockmodels and community structure in networks. *Physical review E*, 83(1):016107, 2011.

[16] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80(1):016118, 2009.

[17] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008.

[18] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187, 2005.

[19] Chao-Wei Ou and Sanjay Ranka. Parallel incremental graph partitioning. *IEEE Transactions on Parallel and Distributed Systems*, 8(8):884–896, 1997.

[20] Leto Peel, Daniel B Larremore, and Aaron Clauset. The ground truth about metadata and community detection in networks. *Science advances*, 3(5):e1602548, 2017.

[21] Tiago P Peixoto. Efficient monte carlo and greedy heuristic for the inference of stochastic block models. *Physical Review E*, 89(1):012804, 2014.

[22] Carey E Priebe, Youngser Park, Joshua T Vogelstein, John M Conroy, Vince Lyzinski, Minh Tang, Avanti Athreya, Joshua Cape, and Eric Bridgeford. On a two-truths phenomenon in spectral graph clustering. *Proceedings of the National Academy of Sciences*, 116(13):5995–6000, 2019.

[23] Matti Raitoharju and Robert Piché. On computational complexity reduction methods for kalman filter extensions. *IEEE Aerospace and Electronic Systems Magazine*, 34(10):2–19, 2019.

[24] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical review E*, 74(1):016110, 2006.

[25] Bambang Soelistijanto. Improving node popularity calculation using kalman filter in opportunistic mobile social networks. In *2018 6th International Conference on Information and Communication Technology (ICoICT)*, pages 119–124. IEEE, 2018.

[26] Wanli Wang, K Tse Chi, and Shiyuan Wang. Performance comparison of nonlinear kalman filters in epidemic tracking on networks. *IEEE Systems Journal*, 14(4):5475–5485, 2020.

[27] Frank Wanye, Vitaliy Gleyzer, Edward Kao, and Wu-chun Feng. Topology-guided sampling for fast and accurate community detection. pages 1–8, 2021.

[28] Wikipedia contributors. Kalman filter — Wikipedia, the free encyclopedia, 2022. [Online; accessed 21-August-2022].

[29] Xiao Ying Zhang and Ye Yuan. Dynamic community detection via kalman filter-incorporated non-negative matrix factorization. In *2021 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, volume 1, pages 1–6. IEEE, 2021.