BMC Bioinformatics

RESEARCH ARTICLE Open Access

Denoising large-scale biological data using network filters

Andrew J. Kavran[1,2] and Aaron Clauset[2,3,4*]

*Correspondence:
aaron.clauset@colorado.edu
[2] BioFrontiers Institute, University of Colorado, Boulder, CO, USA
Full list of author information is available at the end of the article

Abstract

Background: Large-scale biological data sets are often contaminated by noise, which can impede accurate inferences about underlying processes. Such measurement noise can arise from endogenous biological factors like cell cycle and life history variation, and from exogenous technical factors like sample preparation and instrument variation.

Results: We describe a general method for automatically reducing noise in large-scale biological data sets. This method uses an interaction network to identify groups of correlated or anti-correlated measurements that can be combined or "filtered" to better recover an underlying biological signal. Similar to the process of denoising an image, a single network filter may be applied to an entire system, or the system may be first decomposed into distinct modules and a different filter applied to each. Applied to synthetic data with known network structure and signal, network filters accurately reduce noise across a wide range of noise levels and structures. Applied to a machine learning task of predicting changes in human protein expression in healthy and cancerous tissues, network filtering prior to training increases accuracy up to 43% compared to using unfiltered data.

Conclusions: Network filters are a general way to denoise biological data and can account for both correlation and anti-correlation between different measurements. Furthermore, we find that partitioning a network prior to filtering can significantly reduce errors in networks with heterogenous data and correlation patterns, and this approach outperforms existing diffusion based methods. Our results on proteomics data indicate the broad potential utility of network filters to applications in systems biology.

Keywords: Networks, Denoising, Machine learning

Background

System-wide molecular profiling data are often contaminated by noise, which can obscure biological signals of interest. Such noise can arise from both endogenous biological factors and exogenous technical factors. These factors include reagent and protocol variability, researcher technique, passage number effects, stochastic gene expression, and cell cycle asynchronicity. This variability can mask underlying biological signals when measuring cell state and how it changes under different conditions, e.g., in development

© The Author(s) 2021. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

[1, 2], cancer progression [3], and adaptive drug resistance [4, 5]. Noise has also been implicated in the appearance of false signals and in the non-replicability of some studies [6, 7]. Identifying and correcting noisy measurements before analysis is likely to improve the detection of subtle biological signals and enable more accurate predictions in systems biology.

If correlations between related molecular signals are stronger than correlations among sources of noise, then distinct but related signals can be combined to denoise biological measurements, at the expense of a smaller effective sample size. There are three common approaches to identifying related signals: gene sets, subspace embedding, and networks. In the first category, methods like GSEA [8, 9] use the enrichment of genes within curated sets to project the data onto biologically relevant features. While gene sets can increase the power to identify differentially regulated processes, they are inherently coarse, and can themselves be noisy, incomplete, or biased, and thus may not generalize to novel processes. Subspace embedding techniques include PCA [10], clustering [11], and neural network autoencoders [12, 13]. These methods can capture novel gene-gene correlations, but they rarely incorporate biological information into the feature extraction, which can limit both interpretability and generalizability.

Molecular profiling data alone does not directly inform which measurements should be more or less related to each other. Networks that represent a molecular system's functional structure can provide this missing information. For example, protein-protein interaction, metabolic reaction, and gene regulation networks each encode precise and biologically meaningful information about which groups of measured protein expression levels, metabolite concentrations, or transcript levels are functionally related, and hence which measurements should be combined to filter out independent noise. Current network approaches use computationally intensive methods to identify which entities are most related, which can limit their utility for large networks and general usability [14, 15]

Among neighboring elements in the network, the underlying signals may be correlated (assortative) or anti-correlated (disassortative) [16]. For example, differential expression tends to correlate between neighboring genes in a regulatory network [17]. In contrast, inhibitory or compensatory interactions [18, 19] will tend to produce a disassortative relationship. Beyond pairs of measurements, networks can also exhibit large-scale mixing patterns among these interactions, such that a network may be more or less assortative in some regions and disassortative in others [20]. Existing network-based methods typically do not exploit this variability, and instead assume globally assortative mixing by applying a single filter to the whole network [14, 15, 21]. Mismatching the filter and the relationship type, e.g., an assortative filter with anti-correlated measurements, can further obscure the underlying biological signals. Here, we describe a general network-based method that can automatically detect large-scale mixing patterns and account for both assortative and disassortative relationships.

These network filters are closely related to kernel-based methods in image processing [22], in which groups of related pixels are transformed together to improve their underlying visual signal. Most such techniques leverage an image's underlying grid geometry to choose which pixels have related signals for denoising. Networks lack this geometry because a node's interactions are inherently unordered, whereas the

left- and right-hand neighbors of a pixel are clearly defined. This connection between network filters and image processing is rich with potentially useful ideas that could be adapted to process large-scale biological data. For instance, community detection in networks is a clear analog of the common "segmentation" step in image analysis, in which pixels are first partitioned into groups that represent the large-scale structure of an image, e.g., to separate foreground and background, or a car from the street, and then different filters are applied to each segment (module).

We first describe two classes of network filters, which combine measurement values from neighboring nodes to calculate an assortative or disassortative denoised value, and we describe a general algorithm that decomposes the network into structural modules and then automatically applies the most appropriate filter to the nodes and connections within each module. When applied to synthetic data where the true values and network structure are known, these filters substantially reduce errors relative to a baseline. In addition, we show how applying the wrong filter with respect to the underlying biological relationship can lead to increased errors. Finally, to test the practical utility of these methods in a more realistic setting, we investigate the impact of network filtering on a machine learning task in which we predict changes in human protein expression data when a healthy tissue becomes cancerous. Using the network filters to denoise the expression data before model training increases the subsequent prediction accuracy up to 43% compared to training on unfiltered data.

Results

Network filters

A network filter is specified by a function $f[i, \mathbf{x}, G]$, which takes as input the index of the measurement (node) to be denoised, the list of all measurements $\mathbf{x}$, and the network structure G among those measurements. The output is the denoised value $\hat{x}_i$. Here, we consider only local network filters, which use the measurement values of i's immediate neighbors in G, denoted by the node set v_i, which are likely to be the most biologically relevant for denoising. Each filter is applied synchronously, so that all denoised values are obtained simultaneously to prevent feedback within the denoising process.

We note that the idea of a network filter can naturally generalize to exploit information, if available, about the sign or strength of interactions in G. This information can be encoded by an edge weight w_{ij}, which can capture inhibitory or excitatory interactions that are strong or weak. Below, we focus on the case in which this information is not available.

When a measurement x_i correlates with the values of its neighbors x_{v_i} in the network (assortativity), a network filter should adjust x_i to be more similar to the measured values of its neighbors (Fig. 1a). Among the many choices of functions with this qualitative behavior, the mean and median have useful mathematical properties, and connect with past work [21]. This setting is analogous to a smoothing operation in image processing, in which a pixel's value is replaced by the mean or median of its value and its neighbors' values. In the context of a network, the mean and median "smoothing" filters have the forms:

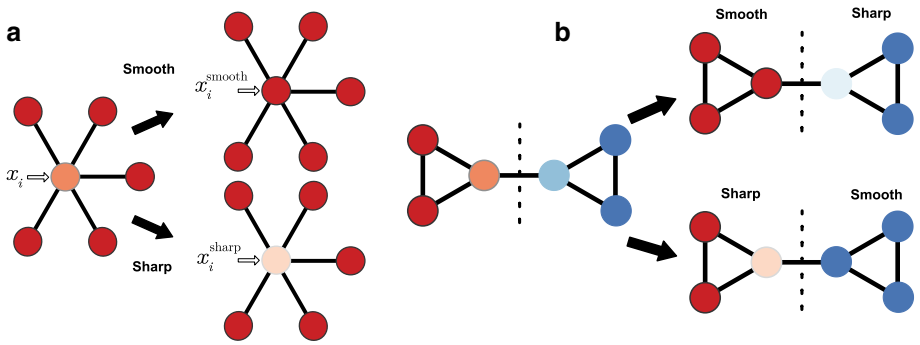

Fig. 1 Schematics of network filters. Network filters are tools that denoise real-valued biological data using a biologically meaningful network to exploit the correlation ("smoothing") or anti-correlation ("sharpening") among neighboring measurements. **a** A measurement x_i and its neighboring values in network, where the color intensity is proportional to the measured value. In applying the smooth filter, x_i is adjusted to be more similar to its neighbors; in applying the sharp filter, x_i is adjusted to be more distant from its neighbors. **b** Measurements can also first be partitioned into groups (dashed line) by detecting structural modules within the network, and then different filters applied to different modules, ignoring between-module edges, e.g., if the signals are assortative in some communities and disassortative in others

$$f_{\bullet,1}[i,\mathbf{x},G] = \frac{1}{1+k_i}\left(x_i + \sum_{j\in\nu_i} x_j w_{ij}\right), \tag{1}$$

where $w_{ij}=1$ and k_i is the degree of node i, reflecting unweighted interactions, and

$$f_{\bullet,2}[i,\mathbf{x},G] = \text{median}[\{\mathrm{x_i},\ \mathrm{x}_{\nu_i}\}]. \tag{2}$$

When a measurement x_i anti-correlates with the values of its neighboring nodes, a network filter should adjust x_i to be more distant from its neighbors (Fig. 1a). This setting is analogous to enhancing the contrast in an image, e.g., when using the technique of unsharp masking to enhance the high frequency signal in an image to make it sharper. In the context of a network, this "sharpening" filter has the form:

$$f_{\circ}[i,\mathbf{x},G] = \alpha(x_i - f_{\bullet,1}[i,\mathbf{x},G]) + \bar{\mathbf{x}} \tag{3}$$

where α is a constant scaling factor, and $\bar{\mathbf{x}} = n^{-1}\sum_i x_i$ is the global mean. Because α is a free parameter, its value should be determined de novo for each data set. For the data sets in this study, we empirically determined the optimal $\alpha = 0.8$ using cross validation.

When a system exhibits large-scale mixing patterns of assortative and disassortative relationships, a network should first be partitioned into structural modules using a community detection algorithm, so that relationships within each module are more homogeneous. Let $\vec{s} = \mathcal{A}(G)$ denote the result of applying a community detection algorithm $\mathcal{A}$ to network G, and say that G_{s_i} denotes the subgraph of nodes and connections within the module s_i that contains node i. Given such a modular decomposition $\vec{s}$, a filter can then be applied to only the subgraph G_{s_i} that contains measurement i. As a result, relationships that span the boundary between two modules will have no influence on the filtered values (Fig. 1b).

After partitioning, the same filter can be applied to every community, or sharp and smooth filters can be applied to communities with more or less assortative values, respectively. We define such a "patchwork filter" as:

$$f[i, \mathbf{x}, G_{s_i}] = \begin{cases} f_{\circ}[i, \mathbf{x}, G_{s_i}], & \text{if } r_{s_i} < 0 \\ f_{\bullet,1}[i, \mathbf{x}, G_{s_i}], & \text{if } r_{s_i} \geq 0 \end{cases}, \tag{4}$$

where r_{s_i} is the standard assortativity coefficient calculated over observed values within community s_i [16]. While any community detection algorithm can be used for $\mathcal{A}$, here we use methods from three classes of algorithms: modularity maximization [23], spectral partitioning [24], and statistical inference. For community detection by statistical inference, we use the degree-corrected stochastic block model or DC-SBM [25] or the "metadata-aware" version of DC-SBM [26], which are considered state-of-the-art methods [27].

Tests using synthetic data

We evaluated the performance of these network filters in two controlled experiments with either non-modular or modular synthetic networks, and varying structures and levels of noise. Also, we compare the performance of network filters to other network-based denoising methods that combine values of nodes weighted by a diffusion matrix [14, 15].

In the first experiment, we generated simple random graphs with heavy-tailed degree distributions (see "Methods" section) and assigned each node a value drawn from a Normal distribution with mean $\mu = 100$ and standard deviation $\sigma = 10$. These values were drawn in such a way that the assortativity coefficient of the network ranged from $r \in [-0.8, 0.8]$ (see "Methods" section). As a result, connected values ranged from being highly anticorrelated to highly correlated. To simulate independent measurement noise, we permuted the values among a uniformly random 25% of nodes, and then denoised these "corrupted" values. We find qualitatively similar results for other choices of the fraction permuted. Results report the mean absolute error (MAE) of a denoised value, averaged over 5000 replications.

Without a filter, the average error of a "denoised" value is independent of the underlying correlation (assortativity) among connected values, because this nearby information is left unexploited (Fig. 2a). In contrast, applying a network filter to denoise the corrupted values can substantially improve their accuracy, depending on how strongly coupled a measurement's true value is with its neighbors', and what filter is applied to recover that information. For the particular parameters of this experiment, filtering can reduce the error by 37–50% over no filter, and by roughly 20% even in the case of uncorrelated signals ($r = 0$), due to a regression to the mean effect. Error reductions are largest when a network "smoothing" filter is applied to strongly assortative signals, and when a network "sharpening" filter is applied to strongly disassortative signals. That is, denoising works best when the underlying signal structure is matched with the assumptions of the filter.

When the wrong filter is applied, however, error rates can increase relative to not filtering. In such a case, the filter creates more errors in the data than it corrects. On the other hand, this "mismatch" penalty only degrades the overall accuracy at very high

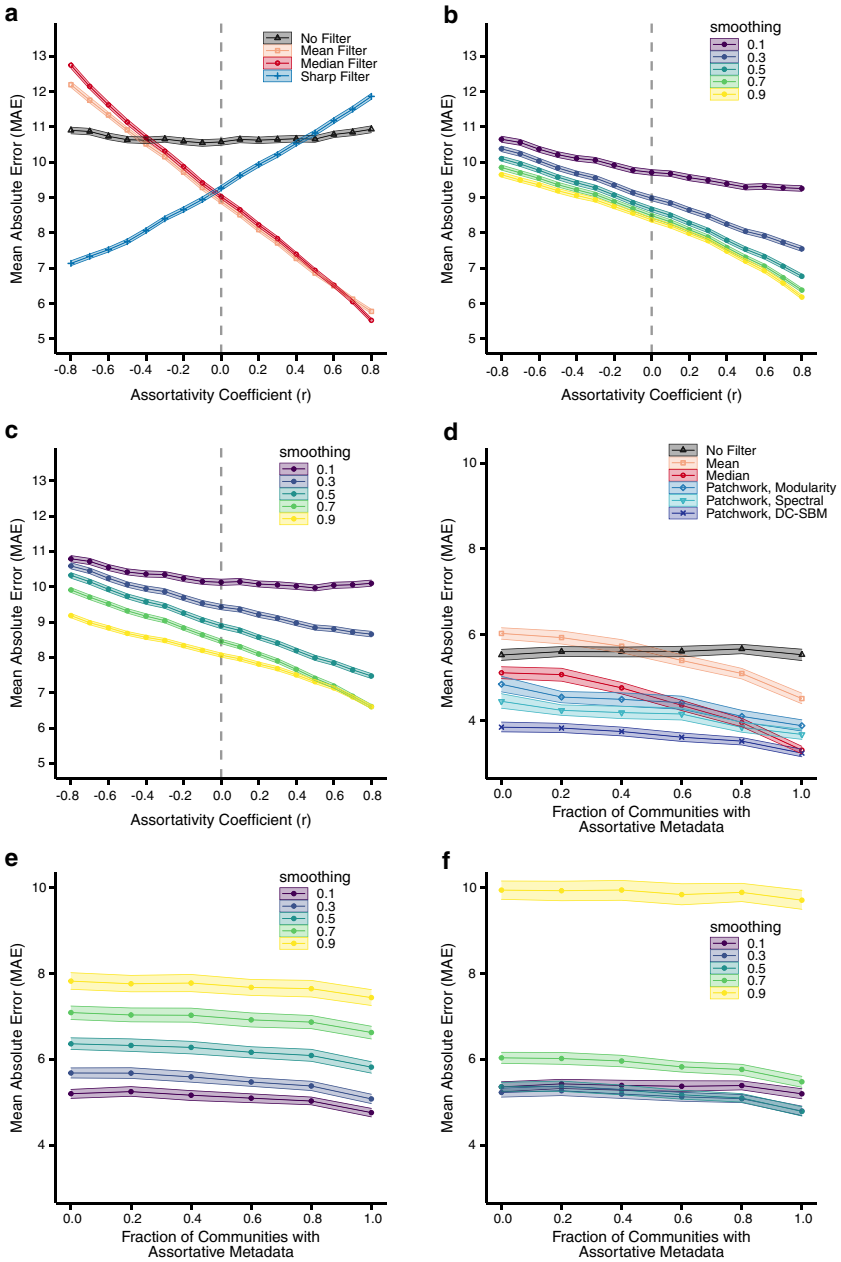

Fig. 2 Filter performance on synthetic networks. Network filter tests on synthetic graphs with varying structures and known noise. The Mean Absolute Error (MAE) of **a** network filters, **b** Laplacian exponential diffusion kernel, and **c** netSmooth on the permuted nodes as a function of the assortativty coefficient of 5000 instances of noisy non-modular graphs. The smooth filters (mean and median) perform best on assortative data ($r > 0$), while the sharp filter is optimal for disassortative data ($r < 0$). When data are neither assortative nor disassortative ($r \approx 0$), netSmooth and Laplacian exponential kernels perform best. The MAE of **d** network filters, **e** Laplacian exponential diffusion kernel, and **f** netSmooth on the permuted nodes as a function of the fraction of communities with assortative data values for 100 instances of noisy modular graphs. Each network instance has 5 communities and we vary how many communities have assortative versus disassortative data values with a moderate assortativity coefficient $|r| \in [0.4, 0.7]$. The shaded areas indicate 99% bootstrapped confidence intervals

levels of correlation (anti-correlation) among the signals, where its magnitude exceeds the natural benefits of filtering at all (Fig. 2a). When the underlying correlations are moderate ($|r| < 0.4$), the average benefits of network filtering will tend to outweigh the average error induced applying the wrong filter.

We also applied two other network methods to these non-modular synthetic graphs. These methods denoise data by combining node values weighted by a diffusion kernel. In the method called netSmooth [14], every node is weighted by a personalized PageRank random walk vector [28], which are linearly combined to create a denoised value. The second method is conceptually similar, but uses a graph Laplacian exponential diffusion kernel to weight nodes before linearly combining them to create the new denoised value [15]. Both methods have an adjustable parameter that determines the smoothness of the resulting denoised values. For greater values of this smoothing parameter, the methods place less weight on the node's original noisy value and more weight on distant nodes.

We applied both methods to the same synthetic random graphs as the network filters, while varying the smoothing parameters between low smoothing (parameter = 0.1), and high smoothing (parameter = 0.9). The Laplacian exponential kernel (Fig. 2b) and netSmooth (Fig. 2c) decrease the error of the noisy data as the assortativity increases. Furthermore, both methods show lower error as the smoothing parameter increases. These diffusion-based methods perform better than the smoothing network filters at either highly dissassortative or weakly assortative values. Since these methods will typically use a larger number of node's values to denoise, their regression to the mean effect tends to be more accurate than the more localized smoothing network filters. However, when a node's value becomes more correlated with its neighbors' values, the smoothing network filters decrease the noise more than the diffusion-based methods. And while the diffusion based methods work better than the smoothing filters on disassortative data, the sharp filter is the best performing method for weakly to strongly disassortative data.

These tests assume that the network structure itself is not noisy. However, in real biological networks, there can be both missing and spurious edges [29]. We tested the robustness of network filters to noise in network structure. After creating a synthetic graph and assigning data to nodes, we add different levels of noise by replacing true edges with new edges between nodes chosen uniformly at random [30]. Thus, this process simulates both cases where the network is missing edges and contains false edges. We find that a noisy network decreases the performance of the mean filter (Additional file 1: Fig. S1A) and median filter (Additional file 1: Fig. S1B) on graphs with assortative data ($r > 0$), and the sharp filter on graphs with disassortative data ($r < 0$) (Additional file 1: Fig. S1C). However, the network filters still substantially reduce the error compared to the no-filter baseline. When the network is very noisy (90% rewired edges), applying a filter reduces the error compared to the no-filter baseline. This pattern is due to a regression to the mean effect, since rewiring the network effectively shrinks the assortativity coefficient closer to zero.

In the second experiment, we again generated simple random graphs with heavy-tailed degree distributions, but now also with modular structure, which better captures the structure of empirical biological networks (see "Methods" section). These modules denote groups of nodes that connect to other groups in statistically similar ways. For instance, protein interaction networks can be decomposed into groups with similar

biological function, and these groups can have distinct types or levels of signal assortativity [20]. In this situation, applying a single filter to all parts of the network could introduce bias in the denoised values, by pooling nearby measurements indiscriminately, compared to filtering modules independently.

Here, we plant $\kappa = 5$ modules in the same kind of synthetic network as our first experiment, set each module to have a different mean value, and then vary the fraction of modules that have a positive assortativity coefficient $|r| \in [0.4, 0.7]$ versus a negative coefficient (see "Methods" section). This kind of signal heterogeneity across modules mitigates the denoising benefits of a simple regression to the mean, and provides a harder test for denoising methods. Given these choices, we generated values within a module, and simulated measurement noise as in the previous experiment (see "Methods" section). In addition to the previous filters, we also apply the "patchwork" filter in this experiment.

As before, the average error of a denoised value with no filter provides a consistent baseline against which we may assess improvements from filtering (Fig. 2d). And similarly, the error for both the smooth and median filters falls steadily as the fraction of modules with assortative signals increases. For the particular parameters of this experiment, the median filter performs roughly 20% better than the mean filter, reflecting the median's well-known robustness to outliers, which arise here from the planted signal heterogeneity.

The global sharp filter works poorly for all ratios when applied uniformly across the whole network (Additional file 1: Fig. S2). Because each module has a distinct mean value, the global sharp filter generates errors by assuming the global mean is a good representation of the whole network.

In contrast, the patchwork filter with different community detection algorithms exhibits less dynamic range in its error (Fig. 2d). When paired with the DC-SBM, it is substantially more accurate than any other filter across different degrees of modular assortativity. For the particular parameters of this experiment, the patchwork filter paired with the DC-SBM reduces the mean error by 30–41% compared to no filtering and by 3–36% compared to median or mean filtering. Only when all of the modules are assortative does the median filter come close to the DC-SBM patchwork filter's accuracy. This advantage arises because the patchwork filter avoids applying the same filter to different types of underlying signals, if the structure of those signals correlates with the structure of the network (as it does here). That is, applying a single filter to a modular network can introduce errors when denoising, if the local mixing patterns across modules are heterogeneous. Pairing a community detection algorithm with network filters can avoid this problem by identifying large groups of nodes that should be filtered together, in much the same way that different image filters can be applied after first segmenting an image into distinct regions.

However, for the modularity maximization and spectral partitioning algorithms, the patchwork filter does not perform as well as when paired with the DC-SBM because the algorithms do not partition the network as closely to the true community structure. Thus, the patchwork filter uses measurements from outside a single community more often with these algorithms. Despite imperfect partitioning, the patchwork filter paired with modularity and spectral partitioning algorithms performs 14–28% better than the

mean filter across all levels of modular assortativity. The median filter outperforms the spectral patchwork (9%) and modularity patchwork (15%) at the highest level of modular assortativity, but the patchwork filter still outperforms, or matches the median filter across the rest of the levels of modular assortativity.

We also applied the diffusion-based methods to these synthetic modular networks. The error for both the Laplacian exponential kernel (Fig. 2e) and netSmooth (Fig. 2f) only slightly decreases as the fraction of modules with assortative signals increases. In contrast to the non-modular case, increasing the smoothing parameter for both methods increases the error across all settings. This loss of accuracy occurs because increasing the smoothing parameter places greater weight on more distant nodes which are more likely to be drawn form a different distribution. Hence, the diffusion kernels are more likely to combine values from nodes from different communities leading to a higher error rate.

Denoising protein expression levels in cancer

To evaluate the utility of network filters for denoising biological data in realistic settings, we construct a machine learning task in which we predict the precise changes in human protein expression levels when a healthy tissue becomes cancerous (see "Methods" section). This task has potential applications to detecting pre-cancerous lesions [31, 32]. We then quantify the improvement in out-of-sample prediction accuracy when using a network filter to denoise the input expression data before model training, compared to training on unfiltered data.

For this experiment, protein expression data are drawn from the Human Protein Atlas (HPA) [33], which provides large-scale immunohistochemistry (IHC) measurements for over 12,000 human proteins in 20 tissues, each in both healthy and cancerous states. Antibody based methods like IHC are known to be noisy and prone to variation from uncontrolled experimental parameters [34], which makes this data set a realistic example of noisy molecular profiling data. A standard principal component analysis (PCA) of the raw HPA expression data reveals that the first component correlates with variations in tissue type, while the second correlates with differences between tissue state (healthy vs. cancerous) (Fig. 3a). Some tissues, however, change more than others, and the changes are not always in the same direction. Hence, predicting the precise changes represents a useful and non-trivial machine learning task that network filtering may improve.

For the network filters and diffusion-based methods, we use a comprehensive map of the human protein-protein interaction network (PPIN) [35], which combines data from several interactome databases and is curated for biological interactions with high levels of evidence. While this network represents a broad collection of authoritative interactome data, the completeness of the human PPIN is still uncertain [29], and we do not regard this network as itself noise-free. Taking the intersection of proteins contained in both expression data and interaction network (see "Methods" section) yields data on $n = 8199$ proteins in a network with $m = 37{,}607$ edges.

In the machine learning task, we perform a K-nearest neighbor regression on an embedded representation of the protein expression data to learn how expression levels change with tissue state (see "Methods" section). We evaluate the trained model via the MAE between the predicted and the actual changes in protein expression under leave-one-out cross validation (in which we train on 19 tissue pairs, and

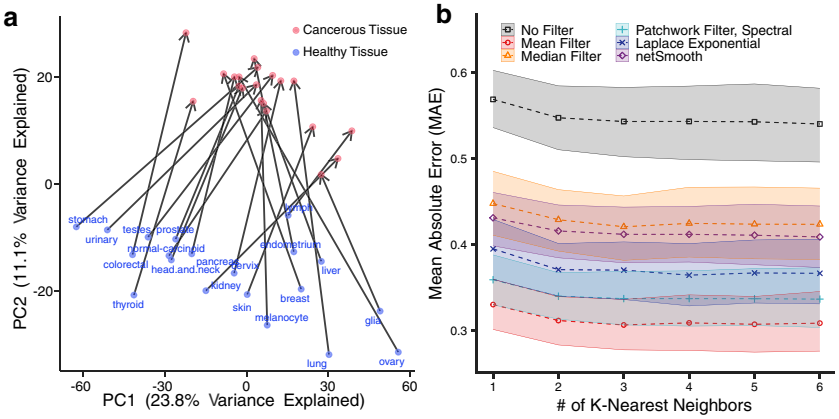

Fig. 3 Denoising to predict protein expression changes in healthy and cancerous tissues. Tests of the network filters on a cancer protein expression prediction task. In this test, we predict the protein expression changes that occur when a healthy tissue becomes cancerous, quantified by the out-of-sample prediction accuracy with and without using network filters to preprocess the data before training. **a** The first two principal components of immunohistochemistry data of healthy and cancerous tissues in the Human Protein Atlas. Arrows connect a healthy tissue (blue) to the corresponding cancer (red). The first component captures variations across tissues, while the second captures variation in state (healthy vs. cancerous). Predicting the precise changes between healthy and cancerous tissues is a non-trivial task. Therefore, we perform a K-Nearest Neighbors regression on the HPA data, with and without preprocessing with network filters. We evaluate the model by leave-one-out cross validation, and calculating the MAE of the predicted and actual data values for the left out healthy-cancerous pair. **b** All network filters and diffusion methods improve the MAE compared to the no-filter baseline. We compare this across different choices of K, as it is a free parameter. The shaded areas represent 95% bootstrapped confidence intervals

predict on the 20th) with or without denoising the expression data with a network filter or diffusion-based method prior to model training. Because the number K is a free parameter that controls the complexity of the learned model, we evaluate the robustness of our results by systematically varying K. For the patchwork filter, we partitioned the graph into 10 modules using the DC-SBM [25] or spectral algorithm [24], while the modularity maximization algorithm [23] automatically chooses the number of modules. Then, we apply the mean filter within each module. In this data, most measured values are weakly assortative across protein interaction edges, and only a few detected modules exhibit any disassortative signal, and even then their internal r is relatively close to zero (Additional file 1: Fig. S3). In this situation, the smooth filter typically outperforms the sharp filter (Fig. 2a).

We used the method proposed by Ronen and Akalin to optimize the smoothing parameter for the diffusion based methods by maximizing the entropy of points embedded in a 2-dimensional PCA space [14]. Since the distributions of the healthy tissue and delta vector data are quite different, we optimized the smoothing parameters of each individually.

Across model complexities, we find that denoising before model training using any type of network filter or diffusion-based method provides a substantial reduction in prediction error relative to training on unfiltered data (Fig. 3b, Additional file 1: Fig. S4). The median filter and netSmooth have very similar performance with around 22% improvement in MAE from no filter. The Laplacian exponential diffusion kernel, patchwork filter

paired with spectral community detection, and mean network filters have the lowest MAE, improving upon the raw data by 32%, 37%, and 43%, respectively.

Error rates tend to decrease with greater model complexity K, suggesting that more complex models are better able to capture variations in the precise expression level changes between tissue states. This decrease in error also occurs without first filtering the expression data. However, the improvement in prediction accuracy from increasing the model complexity without filtering is modest (5.2% at $K = 6$) compared to the improvement from first applying the best network filter (42% at $K = 1$, and 43% at $K = 6$).

We note that in this real-world setting, the patchwork filter, which first partitions the protein interaction network into protein groups, performs better with the spectral or modularity maximization algorithms than with the DC-SBM. The patchwork filter paired with these algorithms performed very well, but the mean filter still performed better than them. This behavior suggests that the partitions produced by the community detection algorithms did not correlate sufficiently strongly with the underlying variation in biological signals to correctly localize the most relevant adjacent measurements, in contrast to our controlled experiments (Fig. 2d). Developing community detection algorithms that choose more biologically relevant partitions may be a useful direction of future work.

Discussion

Large data sets of biological signals, such as system-wide measurements of molecular concentrations, are often noisy. However, these measurements are not fully independent because they reflect the dynamics of a single interconnected system. Using a network to represent the underlying biological relationships among a set of measurements, we can leverage the size of these data sets to systematically denoise many measurements at once, improving the data's utility for understanding the structure and dynamics of complex biological systems or making accurate predictions in systems biology.

Experiments using synthetic data with realistic biological network structures and a variety of underlying signals indicates that network filters can substantially reduce noise in large biological data sets across a broad range of circumstances (Fig. 2a, d, Additional file 1: Fig. S1). The greatest benefit is obtained when the type of filter is matched to the underlying relationship among the signals, e.g., smoothing for assortative signals (correlation) and sharpening for disassortative signals (anti-correlation). However, for modest levels of correlation, even the wrong kind of filter yields some benefit because of a regression to the mean effect, in which combining several related signals filters out more noise than it introduces through bias. When signal types are heterogeneous across the network, so that the strength or direction of the correlation differs in different parts of the network, a "patchwork" filter often performs better. In this approach, we first partition the network into smaller, more homogeneous modules (groups of interrelated measurements) and then apply filters independently to the measurements now localized within each module (Fig. 2d).

In a more realistic setting, in which we train a machine learning algorithm to predict changes in human protein expression levels when healthy tissue becomes cancerous, applying a network filter based on a high-quality protein interaction network before

model training substantially improves prediction accuracy, compared to training on unfiltered data (Fig. 3b). In this experiment, the protein interaction network itself is not noise-free [29], indicating that filtering using an imperfect network can be better than not filtering at all. Our experiment on rewiring network edges further supports that network filters still work well on noisy network structures (Additional file 1: Fig. S1).

In each experiment, we compared our network filters to techniques relying on network diffusion algorithms to weight the nodes before combining them. Both netSmooth and the Laplacian exponential diffusion kernel have similar characteristics to the smoothing network filters. In the non-modular synthetic graphs, they perform better with more assortative underlying data. However, on modular graphs with heterogeneous data values, the performance only slightly increases as more communities have assortative data values, and decreases when communities have disassortative values.

We find an apparent trade-off between the size of the local area of nodes used to denoise a value and the range of values that can be recovered. The diffusion-based techniques outperform local smoothing network filters when there is no correlation or anti-correlation between neighboring values. This improvement is caused by a larger regression to the mean effect from using many more neighbors to denoise any given value. While this effect is beneficial in the experiment with non-modular synthetic graphs, it strongly hinders their performance on modular graphs with heterogenous data values because the diffusion-based techniques tend to use values outside their community, which are drawn from different underlying distributions. Furthermore, increasing the smoothing parameter increases the weight of values outside of the community, strongly deteriorating their performance. Thus, regression to the mean is not beneficial in this experiment since each community has a different distribution of data values. On the other hand, the mean and median filters are more localized and hence make fewer errors due to combining neighbors' values from different communities.

Network filters are ultimately systems-level tools applied to a group of related biological measurements to reduce the overall noise in the system of measurements. On balance, applying network filters reduces the noise in a system of measurements, as evidenced by our tests on synthetic and real datasets. However, there is no guarantee that every individual node's measurement is less noisy after applying a network filter. Furthermore, network filters increase the correlation (or anti-correlation) between the set of denoised values, which reduces the effective sample size. Thus, narrowing the focus to individual nodes after filtering the whole dataset is not the intended use case of network filters. Network filters have the greatest potential for answering questions that take the dataset as a whole, like our machine learning example, rather than considering data of single nodes, such as differential gene expression analysis. Such problems will require more specialized tools specifically suited to the input data.

Network filters could be useful for datasets beyond the ones we describe here, as they only require that a network explains the causal structure of a system of measurements. But some input data may not benefit in its raw form by using network filters particularly if it does not make sense to average together a set of the raw values. For example, in the IHC data from the HPA, each protein is on the same scale of none, low, medium, or high expression level which we converted to a numeric value between zero to three. Since the values of each protein are on the same scale, averaging them together in the

smooth filter is reasonable and will produce a value that has shifted to look more like its neighbor nodes. However, other types of data such as intensities from mass spectrometry based proteomics or raw read counts from RNA-seq can have wildly different scales between proteins and transcripts, which is inherent in the measurement platform. Taking the average of these measurements could create nonsensical values that are largely different from the raw values, and thus may not denoise the data very well. In these cases, transformations of the raw data, like z-score standardization of a node's values across different samples, may be more appropriate.

There are a number of potentially valuable directions for future work on network filters, which may improve their error rates or adapt them to more complicated settings or tasks. Techniques from image processing, both simple and advanced, represent a particularly promising direction to explore [36–38]. For instance, here, we only considered the network filters combine measurements associated with directly adjacent nodes. As a result, the denoised values associated with low degree nodes in the network derive from a relatively smaller number of measurements, and hence are likely to have larger residual noise than will higher degree nodes. Modifying the network filter for low degree nodes to look beyond nearest neighbors, e.g., to ensure a minimum number of pooled measurements per node, may provide better guarantees on the accuracy of the denoised value. An example of this type of technique in image processing include the Gaussian filter [39].

Image segmentation, in which an image is first partitioned into visually distinct pieces, e.g., separating the foreground from the background, is a common preprocessing step in image analysis. The patchwork filter considered here is a simple adaptation of this idea, but it relies on off-the-shelf community detection algorithms to partition the nodes, considers different modules independently, and ignores connections that run between modules. While this approach should retain the most informative relationships among the measurements it also serves to reduce the degrees of many nodes, which may lessen the benefits of filtering, as described above. Furthermore, the patchwork filter will not work well on networks with disassortative community structure where nodes in the same community tend to not form edges between each other. In such cases, the patchwork filter would significantly reduce the degree of all nodes and limit the potential for network filters to denoise their data. Thus, the patchwork filter may perform best with community detection algorithms that return assortative community structures and sever the least number of edges within communities.

Developing filters that utilize the edges between modules could mitigate the induced low-degree effects that come from applying a patchwork filter to account for signal heterogeneity in the system. Such between-module edges should likely be considered separately from within-module edges, e.g., by adjusting their weights w_{ij} to more accurately capture the character of the particular signal relationship between the modules containing nodes i and j.

The benefits of a patchwork filter necessarily depends on how closely the network partition correlates with the underlying biological structure of the system. Off-the-shelf community detection algorithms may not always provide such partitions [40]. While the DC-SBM was able to recover partitions that were good for denoising in the synthetic data task, it did not perform as well as the modularity maximization and spectral

algorithms on the real world data example. Since the assortativity coefficients for the Human Protein Atlas range from 0 to 0.1, the benefit is dominated by the regression to the mean effect, which does better on higher degree nodes to reduce the noise. Thus, the community detection method that finds partitions optimal for denoising may differ network to network [27]. Trying a few different community detection methods like we did here should aid in finding network partitions that best correlate with the system's underlying structure. In some settings, developing application-specific partitioning algorithms, or algorithms that can exploit biologically meaningful node attributes [26], may improve the behavior of a patchwork filter. For data sets where the data is relatively homogenous, a smoothing or sharpening filter applied to the network as a whole may provide more benefits than the patchwork filter.

Finally, the network filters defined here make few specific assumptions about the underlying noise-generating process itself. In specific applications, much more may be known about the direction, magnitude, and clustering of errors across large-scale measurements. For instance, in molecular profiling data, endogenous biological factors like cell cycle effects likely induce distinct noise patterns compared to exogenous technical factors like sample preparation or instrument variation. Developing more application specific error models that could be combined with network filters may provide more powerful denoising techniques than the general filters described here.

Conclusion

Network filters are a flexible tool and can exploit a variety of network data, including networks of molecular binding interactions. Network filters can be extended to exploit information about the sign or strength of interactions or to allow the type of interaction to vary across different modules within the network. These filters can also be applied to networks of any size, ranging from local signaling pathways to entire protein interaction networks. In fact, any network that correlates with the underlying causal structure of a set of measured variables could potentially be used as a filter. By exploiting these underlying relationships, a network filter pools correlated information, which mitigates independent noise, in much the same way that image processing techniques use information from nearby pixels to denoise an image. Overall, our study demonstrates that network filters have the potential to improve the analysis of system-level biological data.

Methods

Synthetic data with known noise and structure

In the first experiment, we generate simple non-modular random graphs using the Chung-Lu (CL) model [41–43] with $n = 100$ nodes and a degree distribution that, in expectation, follows a power law distribution $\Pr(k) \propto k^{-\alpha}$ with parameter $\alpha = 3$ for $k \geq 1$. If the generated degree sequence included a node with degree $k > 17$, a new degree sequence was sampled. This choice ensured that no star-like subgraphs were created. In our analysis, only nodes in the largest connected component were included. This choice mitigates the bias experienced by low degree nodes, which are the most likely nodes to exist outside the largest component.

For each CL synthetic network, we generate node values using the procedure described below. We vary the assortativity coefficient $r \in [-0.8, 0.8]$ while drawing values from a

Normal distribution with mean and variance $\mu = \sigma^2 = 100$. We simulate measurement noise by taking a random permutation of a uniformly random 25% of the node values. We then apply each of the networks filters (mean, median, sharp) to these noisy values, and calculate the mean absolute error (MAE) of the original and denoised values. We also apply netSmooth and Laplacian exponential kernel methods varying smoothing parameter values to this data, and calculate the MAE of original and denoised values. Results are averaged over 5000 repetitions of this process.

To create noisy non-modular networks, we performed a random rewiring procedure previously described [30]. After generating a non-modular random graph using the CL model and generating metadata, we select a given proportion of edges to remove from the graph. Then, we placed the same number of new edges between any two nodes chosen uniformly at random, while ensuring that there were no multi-edges in the graph. Then the filters were applied to the noisy network as normal.

In the second experiment, we generate simple modular random graphs using the degree-corrected stochastic block model (DC-SBM) [25], with $\kappa = 5$ communities of $n_r = 100$ nodes each ($n = 500$ nodes total), and the same degree distribution as the non-modular case. The network's modular structure is specified using the standard "planted partition" model [25], in which the community mixing matrix ω_{rs} is given by a linear combination of a perfectly modular graph and a random graph, and has the form $\omega_{rs} = \lambda\omega_{rs}^{\text{planted}} + (1-\lambda)\omega_{rs}^{\text{random}}$, with $\lambda = 0.85$.

For each DC-SBM network, we generate node values with the following properties: (i) the distribution of values within each module are drawn from a module-specific Normal distribution with mean $\mu = \{110, 80, 60, 40, 20\}$ and variance $\sigma^2 = 25$, (ii) $\kappa' \in [0, 5]$ communities are assigned to have negative assortativity coefficients, and (iii) the within-community assortativity coefficients are chosen uniformly at random on the interval $|r| \in [0.4, 0.7]$. These choices construct a hard test in which a filter's accuracy is effectively penalized if it uses nodes outside a given community to denoise a particular value. For the patchwork filter, we partition the network using three different classes of community detection algorithms. The "metadata-aware" DC-SBM [26] and spectral algorithm [24] partitioned the graph in $\hat{\kappa} = 5$ communities. Modularity maximization partitioned the graph into the number of clusters that maximizes the modularity function [23]. Noise is induced and accuracy is assessed as in the non-modular case, except that the nodes are randomly permuted within each module rather than the whole network.

Generating synthetic correlated measurements

We generate node values with a specified assortativity coefficient r_*, for a specified adjacency matrix A, using Markov chain Monte Carlo (MCMC). The assortativity coefficient r is defined as

$$r = \frac{\sum_{ij}\left(A_{ij} - k_i k_j/2m\right)x_i x_j}{\sum_{ij}\left(k_i\delta_{ij} - k_i k_j/2m\right)x_i x_j}$$

where k_i is the degree of node i, x_i is the value associated with node i, $2m = \sum_{ij} A_{ij}$ is twice the number of edges in the network, A_{ij} is the entry in the adjacency matrix for nodes i and j, and δ_{ij} is the Kronecker delta function.

Given a network A, a desired assortativity coefficient r_*, and a node value distribution $\Pr(x)$, we generate a set of node values as follows.

1 Assign each node a value drawn iid from $\Pr(x)$.
2 Calculate the current assortativity coefficient r_0.
3 Set $t = 1$.
4 While the difference between the desired and current assortativity coefficient $\Delta = |r_t - r_*| > \beta$, a specified tolerance, do:

- Pick a node i uniformly at random and assign it a new value x_i' drawn iid from $\Pr(x)$.
- Calculate the corresponding assortativity coefficient r_t and difference $\Delta' = |r_t - r_*|$.
- If the new value does not improve the assortativity, i.e., $\Delta' > \Delta$ restore x_i. Otherwise, increment t.

5 Return the node values $\mathbf{x}$ with the desired assortativity coefficient, r_*.

In our experiments, we set $\beta = 0.009$.

Diffusion-based denoising methods

We benchmark the network filters against two comparison methods that weight nodes based on different diffusion kernels. The Laplacian exponential diffusion kernel [15] $\mathbf{S}_\beta$ is defined as

$$\mathbf{S}_\beta = \mathrm{e}^{-\beta \mathbf{L}}$$

where β is a real valued smoothing parameter, and $\mathbf{L}$ is the graph Laplacian. The denoised data vector is found by multiplying the matrix and noisy data vector

$$\mathbf{x}_{\text{Laplacian},\beta} = \mathbf{S}_\beta \mathbf{x}.$$

The netSmooth method [14] uses the personalized Pagerank [28] vector to weight each node. This kernel $\mathbf{K}$ is defined as

$$\mathbf{K}_\alpha = (1-\alpha)(\mathbf{I} - \alpha \mathbf{B})^{-1}$$

where $\mathbf{B}$ is a adjacency matrix that is degree normalized by column such that $\mathbf{B}_{ij} = \frac{1}{k_j}$ if there exists and edge between i and j, α is the smoothing parameter (also known as the restart probability), and $\mathbf{I}$ is the identity matrix. The denoised data vector is found by multiplying this kernel and the noisy kernel as such

$$\mathbf{x}_{\text{netSmooth},\alpha} = \mathbf{K}_\alpha \mathbf{x}.$$

For simplicity, we call both β and α "smoothing parameter" throughout as they have a similar function for their respective methods.

Human protein expression and interaction

Protein expression data were drawn from the Human Protein Atlas (HPA) version 16 [33], which details protein expression in human tissues by large scale immunohistochemistry (IHC), for over 12,000 proteins in 20 tissue types, each in a healthy and cancerous state. We represented the IHC scores of not detected, low, medium, and high as numerical values of 0, 1, 2, and 3, respectively. In cases where a protein had scores from multiple patients, the numerical values were averaged together. Human protein interaction (PPIN) data were drawn from the HINT database [35], which combines data from several interactome databases and is curated for biological interactions with high levels of evidence. The HINT network contains $n = 12{,}864$ proteins and $m = 62{,}435$ undirected, unweighted edges.

To construct the network filter, we first map the data from the HPA to the PPIN. HPA proteins are indexed by their Ensembl IDs, while HINT proteins are indexed by their Uniprot IDs. A map from Ensembl IDs to Uniprot IDs was constructed using the HGNC BioMart tool. If a node had multiple mapped expression values, we averaged them. We allow protein expression values from HPA to map to multiple nodes if the Ensembl ID maps to multiple nodes in the PPIN. If the gene expression value does not map to any nodes in the PPIN, we discard these as they cannot be denoised by the network filters. There is one protein in the cancer dataset and 283 proteins in the healthy tissue dataset missing protein expression values in no more than 2 cancers or healthy tissues. For these cases, we impute the missing data from the same protein in another cancer or healthy tissue uniformly at random (impute healthy from healthy, and cancer from cancer).

After keeping the largest connected component of nodes with associated HPA data values, these preprocessing steps produce a network with $n = 8199$ proteins with IHC expression information across all 20 tissue types and both healthy and cancerous states, and $m = 37{,}607$ edges. The included healthy-cancerous tissue pairs are: breast, glioma, cervix, colorectal, endometrial, testicular, thyroid, renal, liver, lung, lymphoma, pancreas, prostate, skin, stomach, melanocyte, urinary, head and neck, ovary, carcinoid. For the healthy tissues, the protein expression values of specific cells types that can give rise to the corresponding cancer were averaged together to form one vector (Additional file 1: Table S1).

Predicting expression changes in human cancer

The machine learning task is to predict the changes in protein expression levels when a human tissue changes types from healthy to cancerous. We use K-nearest neighbors regression to learn a model that can predict these changes when given the expression levels of a healthy tissue (Fig. 4). We train and evaluate the model using leave-one-out cross validation, in which the model is trained on the observed changes in 19 healthy-cancerous tissue pairs, and is tested on one unobserved pair. We first train and evaluate the model on unfiltered data, and then compare its accuracy to a model where we apply a network filter to the expression data prior to training.

First, we applied principal component analysis (PCA) on the training set of 19 healthy tissue protein vectors as a feature extraction method. Next, using the embedded PCA space learned from the training set, we project the held-out healthy sample into the same PCA space. We then determine the K-nearest neighbors of the held-out healthy tissue by

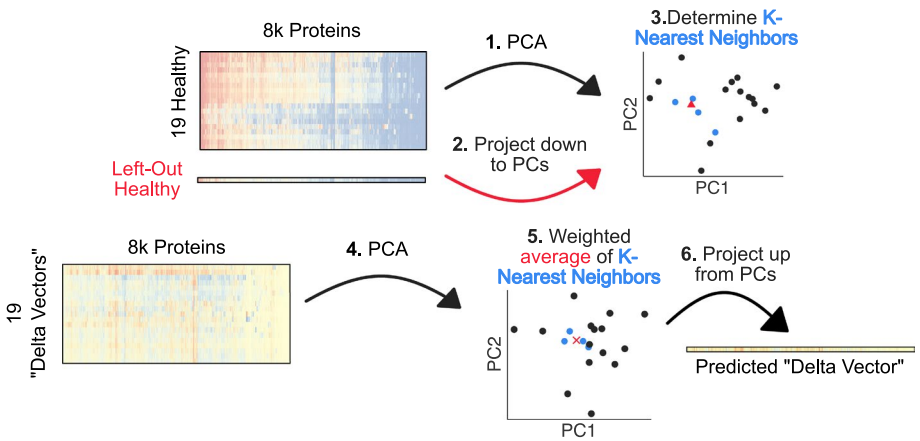

Fig. 4 Schematic of K-nearest neighbors regression framework. We designed a weighted K-nearest neighbors regression framework to predict the protein expression changes a healthy tissue would undergo when becoming cancerous, given a vector of protein expression profile of a healthy tissue. First, we extract features from the training set of 19 healthy tissue protein expression vectors by PCA. Second, we project the left out healthy vector down to the same PCA space, and third, determine K-nearest neighbors to use for the prediction task. Fourth, we extract the features from the 19 delta vectors by PCA, and fifth, predict the delta vector for the left-out healthy sample by taking the weighted average of the K-nearest neighbors' delta vectors. Finally, sixth, we project the predicted delta vector from PCA space back to a vector of protein expression values to calculate the error

calculating the Euclidean distance of the first four principal components between this point and all other healthy tissues.

Given this identification of which healthy tissues are most similar to the left-out healthy tissue, we predict the protein expression changes for the held-out observation. We calculate the expression changes between cancerous and healthy tissues, which we call a "delta" vector. Then, we perform PCA on the 19 delta vectors to extract features. The weighted average of the delta vectors corresponding to the K-nearest neighbors learned from the healthy tissues are averaged together, where the weight is proportional to the inverse of the Euclidean distance to the held-out healthy tissue. Finally, we project the predicted delta vector from four principal components back to the $n = 8199$ proteins and calculate the mean absolute error (MAE) of this vector and the actual delta vector.

The basic networks filters evaluated in this task have the form given in the main text. For the patchwork filter, the DC-SBM or spectral algorithms partition the PPIN into $\kappa = 10$ communities, and modularity maximization automatically chooses the number of communities that maximizes the modularity function. Then, we apply the mean filter within each community.

For the diffusion-based methods, we choose optimized smoothing parameters for the human protein expression data set using a method described by the netSmooth authors [14] to maximize the entropy of a 2D embedding of the data. As the healthy data and delta vectors have different data distributions, we choose optimized smoothing parameters for each data set separately. Briefly, the healthy tissue protein expression or delta vectors were embedded in a PCA space with the first two principal components. This space was discreteized into a four by four grid, equally spaced from the data points at the minimum and maximum of each PC. We calculated the Shannon entropy, $H(x) = -\sum_i P(x_i) \log P(x_i)$, of this discretized embedding, and chose the

smallest smoothing parameter that maximized the entropy. For netSmooth, the smoothing parameter was 0.2 for the healthy tissues and 0.3 for the delta vectors. And for the Laplacian exponential diffusion kernel, it was 0.2 and 0.1 for healthy tissues and delta vectors, respectively.

Abbreviations

CL: Chung-Lu; DC-SBM: Degree-corrected stochastic block model; GSEA: Gene set enrichment analysis; HGNC: HUGO Gene Nomenclature Committee; HINT: High-quality interactomes; HPA: Human Protein Atlas; IHC: Immunohistochemistry; MAE: Mean absolute error; MCMC: Markov chain Monte Carlo; PCA: Principal component analysis; PPIN: Protein–protein interaction network.

Supplementary Information

The online version supplementary material available at https://doi.org/10.1186/s12859-021-04075-x.

Additional file 1: Figures and Tables. Figure S1. Filter performance on rewired synthetic networks. Figure S2. Filter performance on modular synthetic networks, including the sharp filter. Figure S3. Distribution of assortativity coefficients of network modules with Human Protein Atlas Data. Figure S4. KNN regression of Human Protein Atlas data with all network filters. Table S1. Cell types from the Human Protein Atlas dataset averaged together to form a single healthy tissue vector.

Acknowledgements

We would like to thank Natalie Ahn and Mark Newman for helpful conversations.

Authors' contributions

Conceptualization: AC, AJK. Data Curation: AC, AJK. Formal Analysis: AJK. Funding Acquisition: AC. Investigation: AC, AJK. Software: AJK. Visualization: AJK. Writing—original draft: AC, AJK. Writing—review and editing: AC, AJK. Both authors read and approved the manuscript.

Funding

AJK was supported in part by the Interdisciplinary Quantitative Biology (IQ Biology) Program (NSF IGERT Grant No. 1144807) at the BioFrontiers Institute, University of Colorado, Boulder. AC was supported in part by Grant No. IIS-1452718 from the National Science Foundation. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Availability of data and materials

The datasets and code supporting the conclusions of the article are available in github repository https://github.com/andykavran/Network_Filters.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author details

[1] Department of Biochemistry, University of Colorado, Boulder, CO, USA. [2] BioFrontiers Institute, University of Colorado, Boulder, CO, USA. [3] Department of Computer Science, University of Colorado, Boulder, CO, USA. [4] Santa Fe Institute, Santa Fe, NM, USA.

Received: 25 August 2020 Accepted: 15 March 2021
Published online: 25 March 2021

References

1. Woodworth MB, Girskis KM, Walsh CA. Building a lineage from single cells: genetic techniques for cell lineage tracking. Nat Rev Genet. 2017;18(4):230.
2. McKenna A, Gagnon JA. Recording development with single cell dynamic lineage tracing. Development. 2019;146(12):169730.
3. Pastushenko I, Blanpain C. Emt transition states during tumor progression and metastasis. Trends Cell Biol. 2018;29:212–26.

4. Hugo W, Shi H, Sun L, Piva M, Song C, Kong X, Moriceau G, Hong A, Dahlman KB, Johnson DB, et al. Non-genomic and immune evolution of melanoma acquiring mapki resistance. Cell. 2015;162(6):1271–85.
5. Muranen T, Selfors LM, Worster DT, Iwanicki MP, Song L, Morales FC, Gao S, Mills GB, Brugge JS. Inhibition of pi3k/mtor leads to adaptive resistance in matrix-attached cancer cells. Cancer Cell. 2012;21(2):227–39.
6. Michiels S, Koscielny S, Hill C. Prediction of cancer outcome with microarrays: a multiple random validation strategy. The Lancet. 2005;365(9458):488–92.
7. Button KS, Ioannidis JP, Mokrysz C, Nosek BA, Flint J, Robinson ES, Munafò MR. Power failure: why small sample size undermines the reliability of neuroscience. Nat Rev Neurosci. 2013;14(5):365.
8. Mootha VK, Lindgren CM, Eriksson KF, Subramanian A, Sihag S, Lehar J, Puigserver P, Carlsson E, Ridderstrale M, Laurila E, Houstis N, Daly MJ, Patterson N, Mesirov JP, Golub TR, Tamayo P, Spiegelman B, Lander ES, Hirschhorn JN, Altshuler D, Groop LC. PGC-1α-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes. Nat Genet. 2003;34(3):267–73.
9. Barbie DA, Tamayo P, Boehm JS, Kim SY, Moody SE, Dunn IF, Schinzel AC, Sandy P, Meylan E, Scholl C, et al. Systematic rna interference reveals that oncogenic kras-driven cancers require tbk1. Nature. 2009;462(7269):108.
10. Ronan T, Qi Z, Naegle KM. Avoiding common pitfalls when clustering biological data. Sci Signal. 2016;9(432):6.
11. Sørlie T, Perou CM, Tibshirani R, Aas T, Geisler S, Johnsen H, Hastie T, Eisen MB, Van De Rijn M, Jeffrey SS, et al. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. Proc Natl Acad Sci USA. 2001;98(19):10869–74.
12. Gupta A, Wang H, Ganapathiraju M. Learning structure in gene expression data using deep architectures, with an application to gene clustering. In: 2015 IEEE international conference on bioinformatics and biomedicine (BIBM), 2015; p. 1328–35. IEEE.
13. Tan J, Ung M, Cheng C, Greene CS. Unsupervised feature construction and knowledge extraction from genome-wide assays of breast cancer with denoising autoencoders. In: Pacific Symposium on Biocomputing, 2015; p. 132–43.
14. Ronen J, Akalin A. netsmooth: Network-smoothing based imputation for single cell rna-seq. F1000Research 2018;7.
15. Dørum G, Snipen L, Solheim M, Saebo S. Smoothing gene expression data with network information improves consistency of regulated genes. Stat Appl Genet Mol Biol 2011;10(1):37.
16. Newman MEJ. Mixing patterns in networks. Phys Rev E. 2003;67(2):026126.
17. Ideker T, Ozier O, Schwikowski B, Siegel AF. Discovering regulatory and signalling circuits in molecular interaction networks. Bioinformatics. 2002;18(suppl-1):233–40.
18. Goncalves A, Leigh-Brown S, Thybert D, Stefflova K, Turro E, Flicek P, Brazma A, Odom DT, Marioni JC. Extensive compensatory cis–trans regulation in the evolution of mouse gene expression. Genome Res. 2012;22(12):2376–84.
19. Bauer PM, Fulton D, Bo YC, Sorescu GP, Kemp BE, Jo H, Sessa WC. Compensatory phosphorylation and protein–protein interactions revealed by loss of function and gain of function mutants of multiple serine phosphorylation sites in endothelial nitric-oxide synthase. J Biol Chem. 2003;278(17):14841–9.
20. Peel L, Delvenne J-C, Lambiotte R. Multiscale mixing patterns in networks. Proc Natl Acad Sci USA. 2018;115(16):4057–62.
21. Rudolph JD, de Graauw M, van de Water B, Geiger T, Sharan R. Elucidation of signaling pathways from large-scale phosphoproteomic data using protein interaction networks. Cell Syst. 2016;3(6):585–93.
22. Mansourpour M, Rajabi M, Blais J. Effects and performance of speckle noise reduction filters on active radar and sar images. In: Proceeding of ISPRS, 2006; vol. 36, p. 41.
23. Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. J Stat Mech Theory Exp. 2008;2008(10):10008.
24. Ng A, Jordan M, Weiss Y. On spectral clustering: analysis and an algorithm. Adv Neural Inf Process Syst. 2001;14:849–56.
25. Karrer B, Newman MEJ. Stochastic blockmodels and community structure in networks. Phys Rev E. 2011;83(1):016107.
26. Newman MEJ, Clauset A. Structure and inference in annotated networks. Nat Commun. 2016;7:11863.
27. Ghasemian A, Hosseinmardi H, Clauset A. Evaluating overfit and underfit in models of network community structure. IEEE Trans Knowl Data Eng. 2019;32(9):1722–35.
28. Jeh G, Widom J. Scaling personalized web search. In: Proceedings of the 12th international conference on world wide web, 2003; p. 271–9. ACM.
29. Hart GT, Ramani AK, Marcotte EM. How complete are current yeast and human protein-interaction networks? Genome Biol. 2006;7(11):120.
30. Middendorf M, Ziv E, Wiggins CH. Inferring network mechanisms: The *Drosophila melanogaster* protein interaction network. Proc Natl Acad Sci. 2005;102(9):3192–7.
31. Campbell JD, Mazzilli SA, Reid ME, Dhillon SS, Platero S, Beane J, Spira AE. The case for a pre-cancer genome atlas (pcga). Cancer Prevent Res. 2016;9(2):119–24.
32. Spira A, Yurgelun MB, Alexandrov L, Rao A, Bejar R, Polyak K, Giannakis M, Shilatifard A, Finn OJ, Dhodapkar M, et al. Precancer atlas to drive precision prevention trials. Cancer Res. 2017;77(7):1510–41.
33. Uhlén M, Fagerberg L, Hallström BM, Lindskog C, Oksvold P, Mardinoglu A, Sivertsson Å, Kampf C, Sjöstedt E, Asplund A, et al. Tissue-based map of the human proteome. Science. 2015;347(6220):1260419.
34. Vyberg M, Nielsen S. Proficiency testing in immunohistochemistry—experiences from nordic immunohistochemical quality control (nordiqc). Virchows Archiv. 2016;468(1):19–29.
35. Das J, Yu H. Hint: High-quality protein interactomes and their applications in understanding human disease. BMC Syst Biol. 2012;6(1):92.
36. Motwani MC, Gadiya MC, Motwani RC, Harris FC. Survey of image denoising techniques. In: Proceedings of GSPX, 2004; p. 27–30.

37. Agostinelli F, Anderson MR, Lee H. Adaptive multi-column deep neural networks with application to robust image denoising. In: Advances in neural information processing systems, 2013; p. 1493–501.
38. Öktem R, Egiazarian K, Lukin VV, Ponomarenko NN, Tsymbal OV. Locally adaptive dct filtering for signal-dependent noise removal. EURASIP J Adv Signal Process. 2007;2007(1):042472.
39. Deng G, Cahill L. An adaptive Gaussian filter for noise reduction and edge detection. In: 1993 IEEE conference record nuclear science symposium and medical imaging conference, 1993; p. 1615–9. IEEE.
40. Peel L, Larremore DB, Clauset A. The ground truth about metadata and community detection in networks. Sci Adv. 2017;3(5):1602548.
41. Aiello W, Chung F, Lu L. A random graph model for power law graphs. Exp Math. 2001;10(1):53–66.
42. Chung F, Lu L. Connected components in random graphs with given expected degree sequences. Ann Combin. 2002;6:125–45.
43. Alam M, Khan M, Vullikanti A, Marathe M. An efficient and scalable algorithmic method for generating large-scale random graphs. In: SC'16: Proceedings of the international conference for high performance computing, networking, storage and analysis, 2016; p. 372–83. IEEE.

Publisher's note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions