Cell Reports, Volume *26*

Supplemental Information

PyMINEr Finds Gene and Autocrine-Paracrine Networks from Human Islet scRNA-Seq

Scott R. Tyler, Pavana G. Rotti, Xingshen Sun, Yaling Yi, Weiliang Xie, Michael C. Winter, Miles J. Flamme-Wiese, Budd A. Tucker, Robert F. Mullins, Andrew W. Norris, and John F. Engelhardt

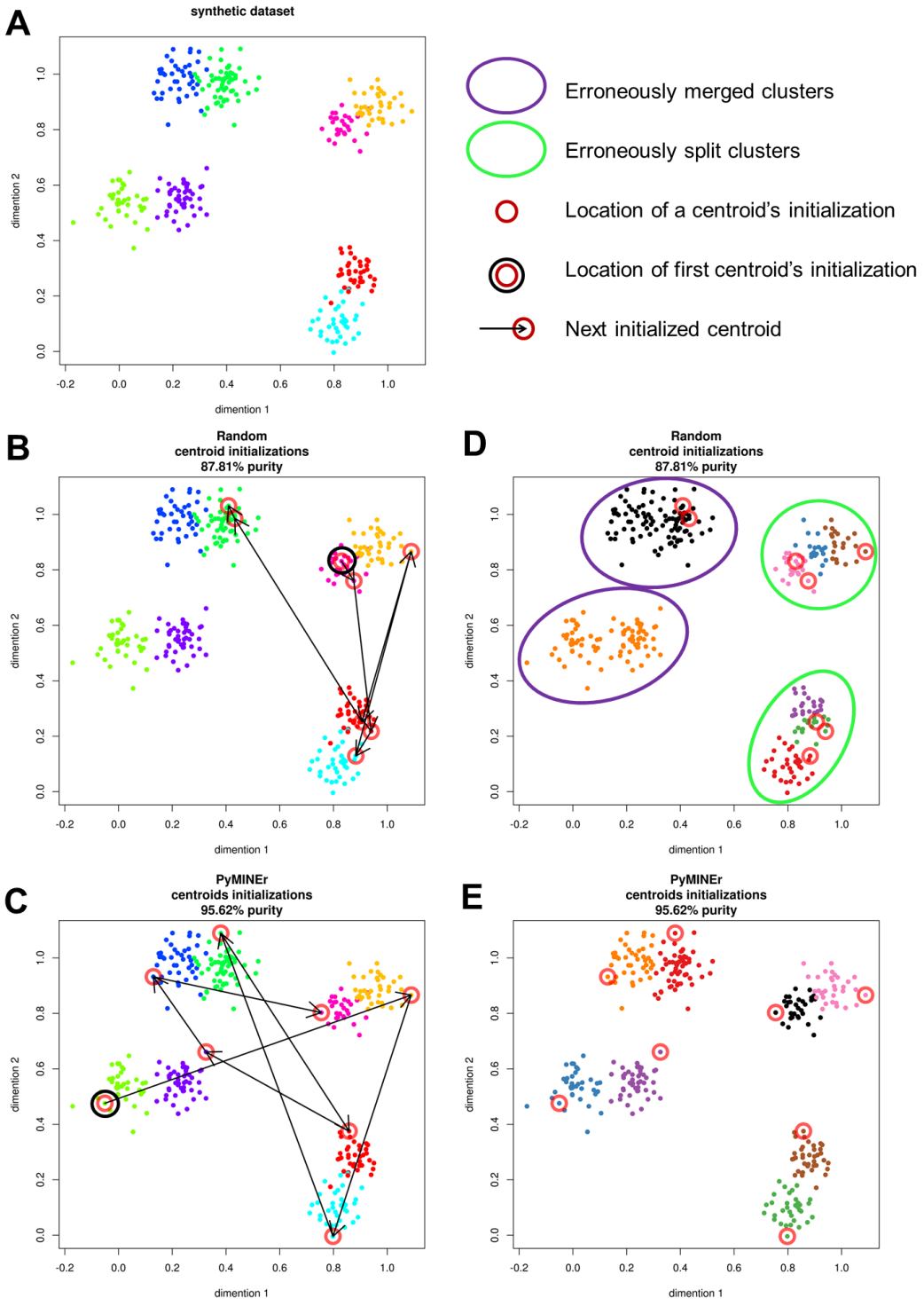

Figure S1.
PyMINEr centroid seeding algorithm applied to a synthetic dataset
Related to Figure 1
(A) A 2-dimensional synthetic dataset comprised of eight groups with Gaussian noise was used to illustrate the comparative effectiveness of the centroid seeding algorithm used in PyMINEr to traditional, random centroid seeding. Group identity is shown by color. **(B-C)** The initialization locations for each centroid seeded by (**B**) randomly, or (**C**)

by PyMINEr initializations. The progression of centroid initialization is indicated by arrows moving from the first to the second and subsequent centroids. **(D)** Random centroid seeding can often result in clusters that are either merged or split, and thus performance is poor. **(E)** Performance with PyMINEr centroid seeding was better, as evident from a lack of cluster mergers and splitting. In (**B-E**), the location of initialized centroids are denoted by a red ring, and in (**B,C**), arrows indicate the progression of centroid selection, with the first centroid denoted by a larger black ring.

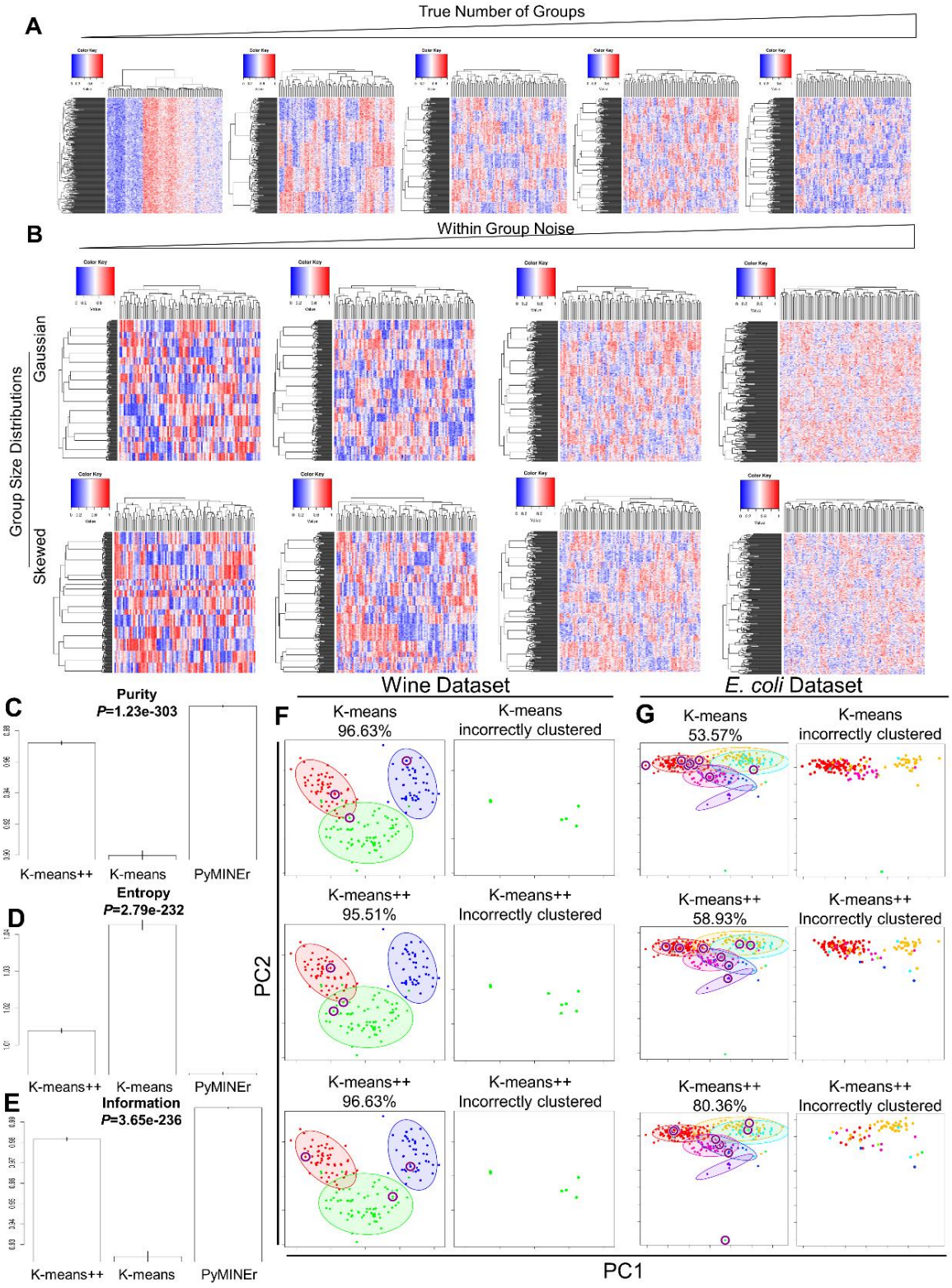

Figure S2.

PyMINEr clustering is more accurate than competing techniques

Related to Figure 1

(A) Heatmaps of synthetic datasets consisting of 300 data points being clustered (rows), with 100 measurements per sample (columns). The number of clusters present in simulated datasets varied over a range between 1 through 20

clusters; here we only show examples from every fifth group for illustrative purposes. **(B)** Points were assigned to a group either uniformly random (leading to a Gaussian distribution in group sizes), or in a skewed manner (generation of several larger, and some very small, groups). These variables were tested across all group numbers, although we display only example datasets where k=15. **(C-E)** Effectiveness of clustering via PyMINEr vs. k-means and k-means++, using synthetic datasets (**Figure S2A-B**) with clusters over a range of sample group numbers, skewness in cluster size, and noise. **(C)** Purity of clustering, with higher purity indicative of fewer cluster splitting events. **(D)** Relative entropy, with low entropy indicative of fewer cluster splitting events. **(E)** Relative mutual information, with higher levels indicative of fewer cluster mergers. Bar graphs show means with s.d. error bars. **(F-G)** Accuracy of PyMINEr clustering for real-world datasets. (**F**) Ability to discriminate types of wines based on their characteristics. (**G**) Subcellular localization of proteins in *E. coli*. The first two principal components of each dataset are shown, along with the location to which the centroids were initialized by each method, noted with purple rings. Groups were each assigned a color, and all the points belonging to that group were plotted using the designated color. Also noted for each clustering method is the percentage of cluster purity.

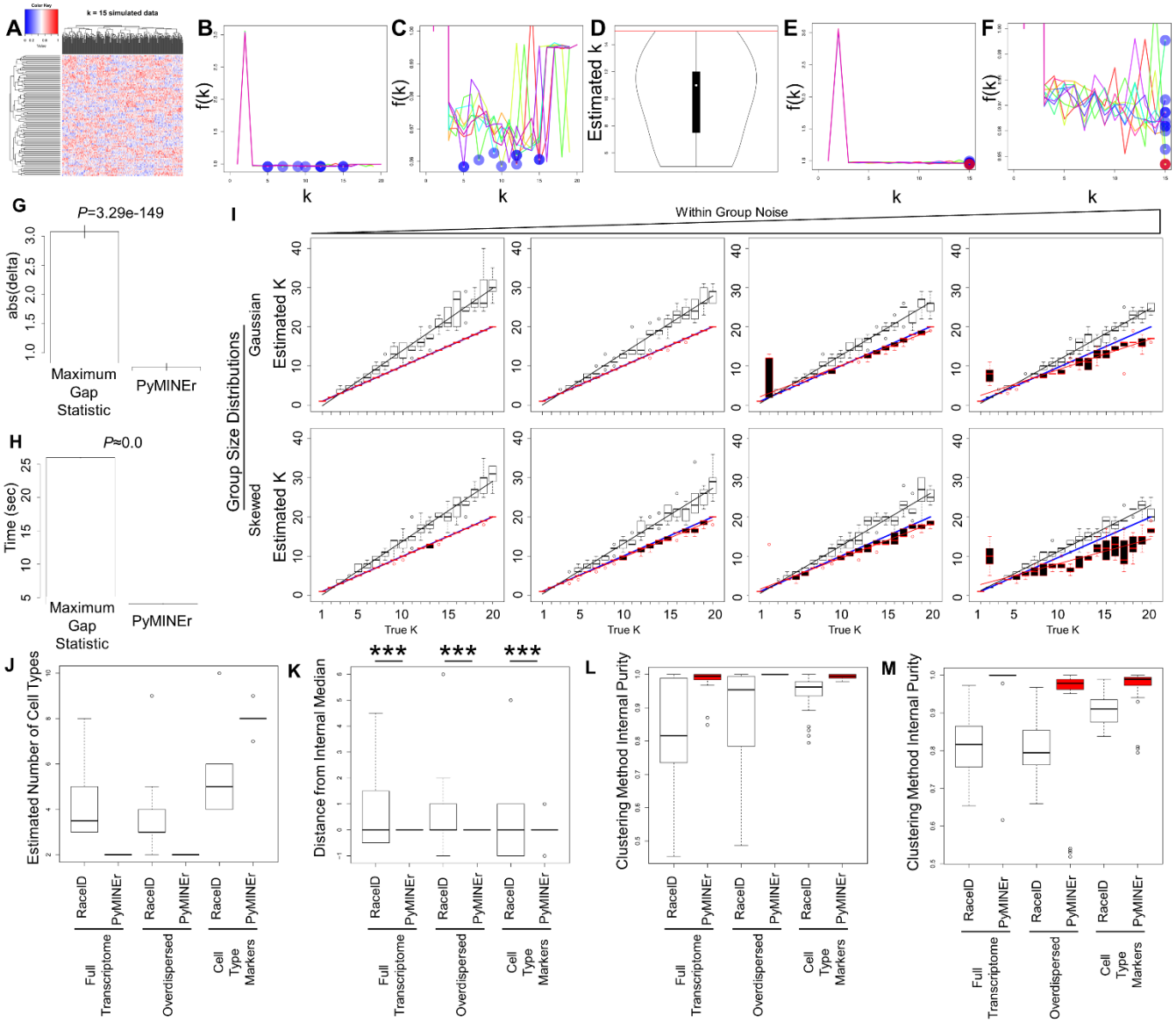

Figure S3.
PyMINErr k selection algorithm and clustering outperforms the maximum gap statistic in most scenarios and PyMINErr clustering algorithms show greater internal consistency compared to RaceID
Related to Figure 1
(A) A heatmap of synthetic simulated dataset with 15 groups containing Gaussian noise was used to demonstrate the process of k selection and clustering results (points being clustered in columns of the heatmap, whereas rows indicate features). **(B-C)** Initial PyMINErr clustering to determine the number of groups in the dataset (k), logged as the f(k) results for 10 iterations, which are shown by different colored lines. **(B)** The minimum f(k) result for each iteration, shown as blue rings. The y-axis scale in **(C)** was adjusted to see the small change in (**B**). **(D)** The 90th percentile estimate from the minimum f(k) results, indicated by a red line, is the correct result of k = 15. **(E-F)** Once the results from (**D**) were obtained, giving the final estimate of k, 10 iterations of clustering were performed by PyMINErr for this estimate of k. Each iteration (noted in different colored lines) logs the f(k) value at the final estimate of k, in this case where k=15 (noted in blue rings). The iteration of clustering with the lowest f(k) value was then used for final group assignments (denoted by a red ring). The y-axis scale in **(F)** was adjusted to see the small change in (**E**). **(G)** For determining the number of groups within a dataset, we used the synthetic datasets exemplified in **Figure S2A-B**,

with each combination of group size, skewness, and noise simulated 5 times each. In total, 800 simulations were performed for each method, including 20 conditions for group size (k=1-20), 2 conditions for skewness (skewed or Gaussian), and 4 levels of within-group noise. The average and standard deviation of the absolute distance of the estimated number of groups, and the true number of groups simulated. **(H)** The amount of time (in seconds) taken by each algorithm to determine the number of groups present. **(I)** Boxplots for multiple replicates in estimating the number of groups in a dataset. Black-lined white boxes are estimates of the number of groups (k) as determined by the maximum gap statistic, while red-lined black boxes are estimates of the group number determined by PyMINEr. The blue line indicates perfect prediction, where the number of groups estimated is equal to the true number of groups in the synthetic dataset, indicated along the x-axis (n=5 for each box). **(J)** The number of cell types as estimated by either RaceID or PyMINEr with three subsets of genes from the human scRNAseq dataset. The full transcriptome boxes represent the number of cell types estimated when the entire transcriptome was used. Cell type markers or overdispersed genes (which show higher than expected variance in expression) are sometimes used in scRNAseq to select genes which may contribute to cell identity. We therefore also show the results of cell type estimates for all three of these datasets (n=10 iterations for each algorithm and dataset). **(K)** To compare internal consistency, each dataset from (**J**) was centered around its median, then an F-test for equal variance was performed comparing each algorithm's performance on each of the three given datasets. PyMINEr results were more self-consistent than those obtained using RaceID as indicated by lower internal variance in cell type number estimates (n=10 iterations; F-test; ***: $P<0.001$). **(L)** Self-consistency as assessed by purity for clustering results between iterations. Clustering by PyMINEr was more internally consistent with respect to cell type labeling, both when the number of cell types was automatically determined using each algorithm (2-Way ANOVA, Factor1 = clustering method, Factor2 = input dataset; Factor1 *P*-value = 3.4e-31). **(M)** PyMINEr also showed greater cluster purity when the number cell types was manually set to 8, rather than automatically determined by each method (2-Way ANOVA, Factor1 = clustering method, Factor2 = input dataset; Factor1 *P*-value = 7.8e-38). Boxplots for PyMINEr results are shaded red.

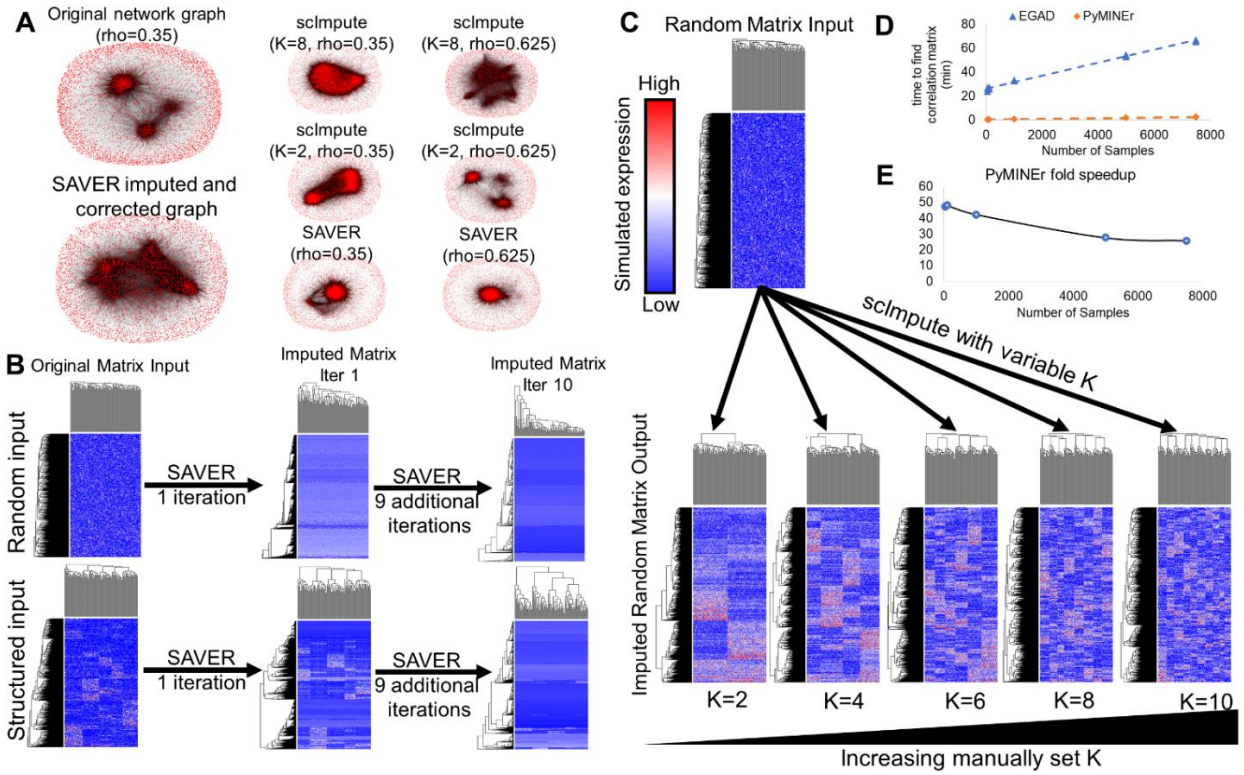

Figure S4.

Related to Figure 3

(A) Graph networks from PyMINEr are shown for our original dataset (rho=0.35), and networks constructed based on scImpute or SAVER with, rho=0.35 or 0.625 to adjust for power gains after imputation). Overall, graph structure from our scRNAseq dataset (**Table 1**) was notably altered by imputation, where the structure from scImpute was largely dependent on the manually set hyperparameter K. When K was set to 8 (i.e., 8 predicted cell types present), the graph structure shared little similarity to the original structure; interestingly, however, with k set to 2, graph structure appeared to be relatively conserved. These results indicate that constructing network graphs from imputed datasets using scImpute are highly dependent on the selection of the hyperparameter K. SAVER on the other hand condensed all nodes into a single hub of co-regulated genes. The SAVER algorithm has an optional correction for this over-correlation and this procedure indeed appears somewhat efficacious, but still blurs modules together as shown below. **(B)** Imputing synthetic and random or synthetic and structured datasets by SAVER decrease variance globally, collapsing many expression values to a non-variant single value after one iteration. However after additional iterations, most of the transcriptome has converged to single values. **(C)** A matrix of simulated Gaussian random data with the lower 50% converted to zero (to coarsely simulate dropout) was imputed using scImpute with variable K (i.e.: the user selecting the number of cell types). scImpute artificially creates clusters that perfectly mirrors the manually input K – all from purely random data. **(D)** A plot depicting the number of samples used in a synthetic dataset (n=50-7,500, x-axis) and the time taken to find the Spearman correlation matrix by either PyMINEr (orange) or EGAD (blue) (Ballouz et al., 2017). We found that PyMINEr's Spearman correlation graph building function to be substantially faster than EGAD's (*P*=2.7e-36). **(E)** A plot of the fold speed-up for using PyMINEr as a function of the number of samples in a dataset. With small datasets, PyMINEr performs ~50 fold faster than EGAD in finding the full Spearman correlation matrix, however, this advantage drops to ~20 fold faster with larger datasets.

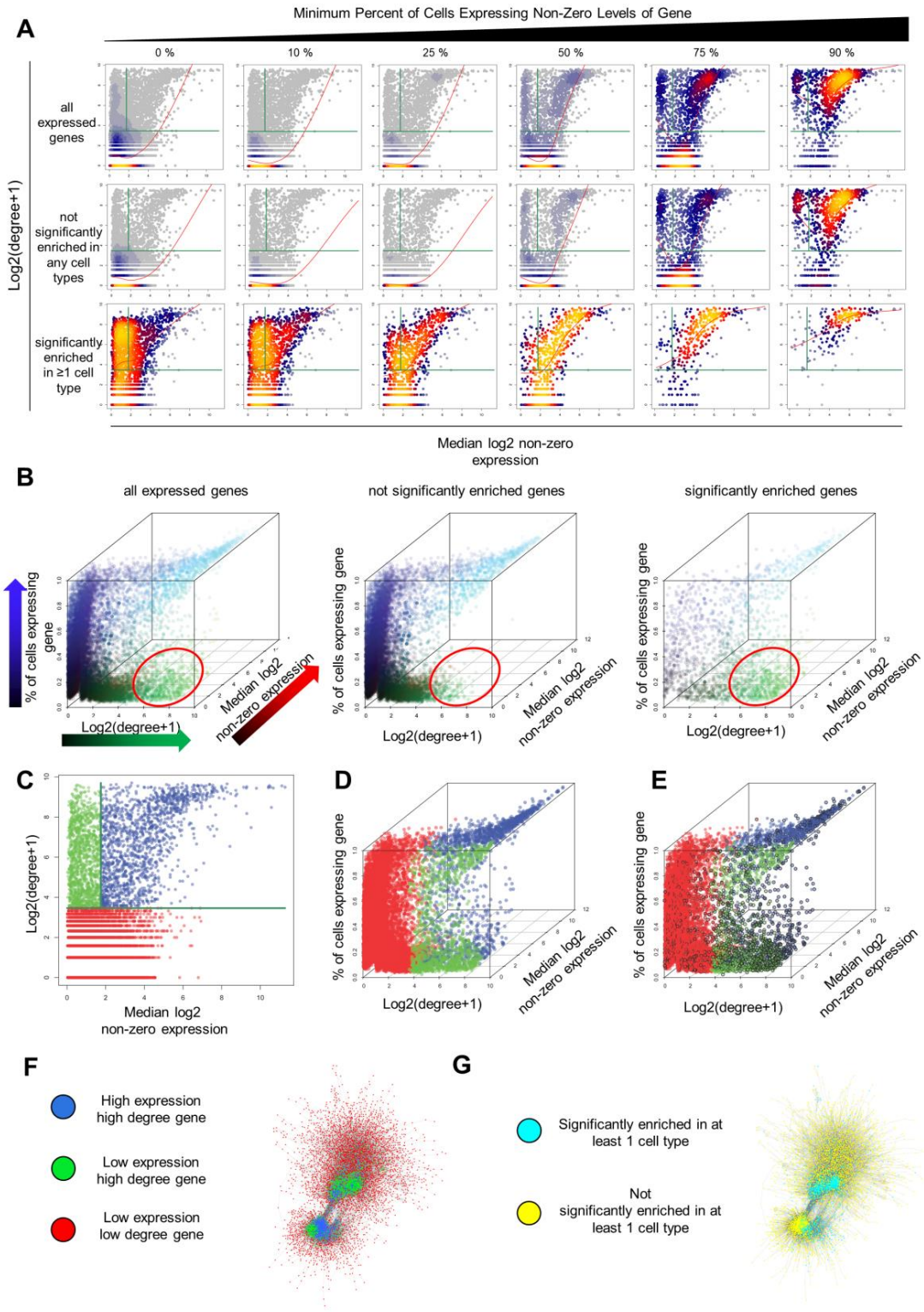

Figure S5.
The relationship between median active transcription and detectable correlations (*degree*)
Related to Figure 4
(A) The relationship between a gene's expression level, number of network-connections, and the percentage of cells expressing the gene was investigated here. We calculated the median log2 expression for each gene, after removing

cells that did not express a given transcript (i.e., median log2 non-zero expression). We expected to find that genes with greater expression would show a greater level of connectivity because the effects of noise and stochasticity at the cellular level would be minimized. Most genes follow the expected pattern of correlation between the median non-zero expression level, and its network-connectivity (i.e., *degree*) (Spearman correlation $P<1e\text{-}18$ in all gene subsets, loess smoothed regression shown as a red line). Here point color indicates relative density (grey: low density, yellow: high density); however, several discrete populations are apparent at various thresholds for percent cells expressing each gene (segregated by green lines). **(B)** Shown is a colorized 3D-scatter plot demonstrating the relationships between the percent of cells expressing a gene, the gene's connectivity [log2(*degree*+1)), and the expression level (median log2(non-zero expression)]. Each gene's location within the 3 dimensions is also color-coded in RGB, where red is expression level, green is connectivity, and blue is the percent of cells expressing the transcript. Note that due to the overall correlation between connectivity and expression, there is no red only population of cells (i.e., genes with high expression and low connectivity). Additionally, the low expression and high connectivity population of genes do not fit the more common correlative pattern between connectivity (i.e., *degree*) and median non-zero expression. This population is shown in green and noted with a red ellipse. 46% of the genes in this population were significantly enriched in at least one cell type ($\chi^2 = 3143.7$; $P<2.22e\text{-}16$). The three panels show distributions for (left) all genes, (middle) genes that were not significantly enriched for expression in at least one cell type, and (right) only genes that were significantly enriched in at least one cell type. The light blue population of genes denote high expression and high degree; green points correspond to highly connected low level expressing genes found in a subset of cells. **(C)** To more clearly denote the populations of genes, we segregated genes into three groups with low non-zero expression and low connectivity (red), low non-zero expression and high connectivity (green), and high expression and high connectivity (blue). **(D)** Addition of a z-axis to (**C**) corresponding to the percent of cells expressing the given gene. **(E)** Addition of a black ring around the genes in (**D**) that are significantly enriched in at least one cell type. Overall, these results indicate that this form of network analysis can overcome the inherent noise of cell type-specific genes with weak transcription when cells are sequenced with sufficient depth. **(F)** Genes with low-level expression and low connectivity are shown in red, and those with low-level expression and high connectivity are shown in green; genes with high-level expression and high network connectivity are shown in blue. **(G)** Alternative representation of the graph network in (**F**) showing the subset of genes that are significantly enriched in at least one cell type (labeled in cyan), with all other genes in the network shown in yellow. 46% of the low expression high degree genes (green) from (**F**) are also contained in the cell type-specific cyan population of (**G**).

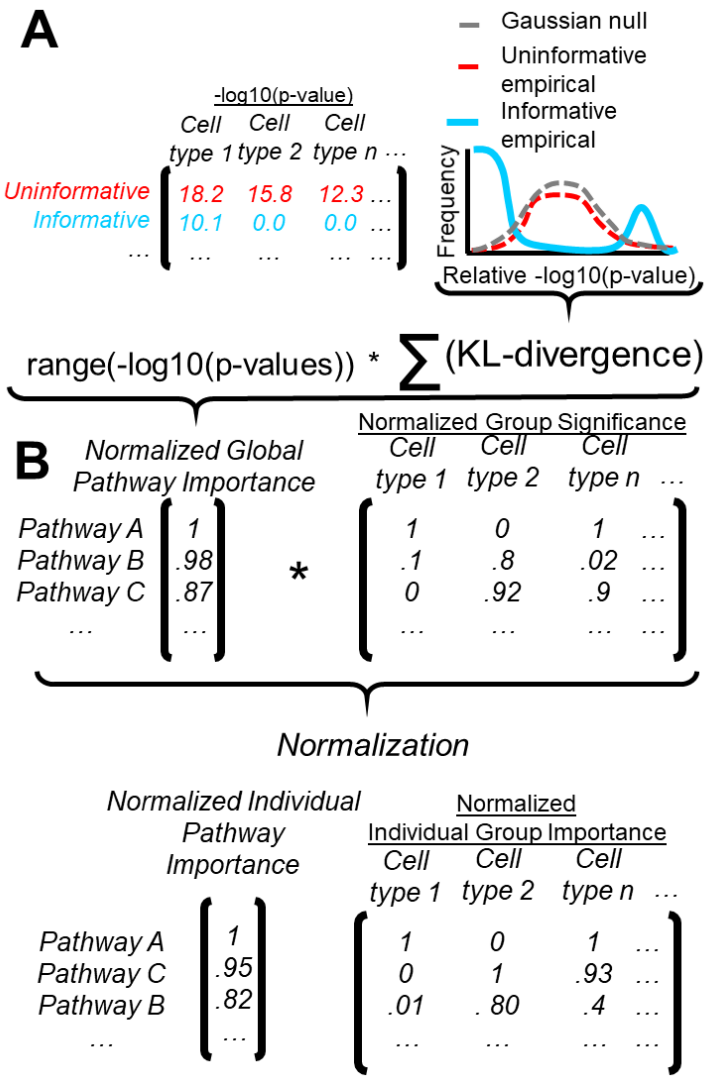

Figure S6.
Novel KL-divergence based pathway ranking metrics
Related to Figure 6
(A) A metric for combining pathway analyses of different cell types or groups that prioritizes based on entropy and overall significance. Pathway analysis frequently yields highly significant p-values for several groups being compared. While using the appropriate background gene set diminishes this, it can still be problematic. For example, observing high significance in a single pathway for all cell types, does not provide any useful information in how these gene sets are different for each other. More informative pathways will be those that are highly significant in some cell types, and non-significant in other cell types. We therefore devised a metric to identify pathways with high information/low entropy across groups – or in the case of scRNAseq, cell types. The uninformative pathways typically share a Gaussian distribution of -log10(p-values), while informative pathways show a bimodal, high information distribution. We therefore calculate the sum KL-divergence from the uninformative null Gaussian distribution. To renormalize for high levels of significance, we additionally multiply by the range of significance across groups (i.e., range within rows). After normalizing again, we obtain the final Normalized Global Pathway Importance. **(B)** A metric for ranking pathway importance within a cell type or group. It is often desirable to have a sorted list of informative pathways for each group being compared. To create this metric, we use the Normalized Global Pathway Importance, multiplied by the normalized -log10(p-values) within each group; in this way, the two metrics are scaled equally. This resulting metric is then scaled within groups (columns), and the range-KL-divergence calculation is performed again and re-normalized yielding the Normalized Individual Pathway Importance and the Normalized Individual Group Importance metrics.