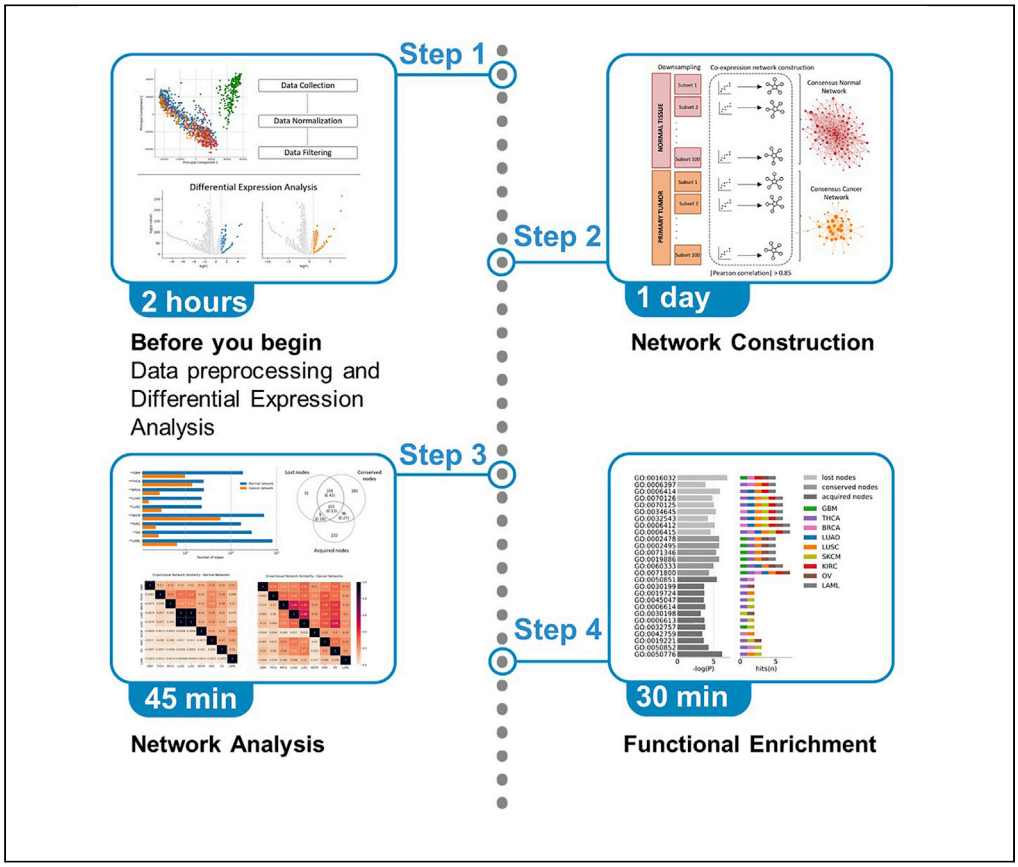
STAR Protocols

CellPress
OPEN ACCESS

Protocol

A computational approach to generate highly conserved gene co-expression networks with RNA-seq data



We describe a consensus approach for network construction based on fully conserved gene-gene interactions from randomly downsampled data subsets for an unbiased differential analysis of gene co-expression networks. The pipeline allows users to identify network nodes lost, conserved, and acquired in cancer as well as interpret the functional significance of these network changes. For proof of concept, the protocol is used to leverage RNA-seq data of tumor samples from TCGA and healthy tissue samples from the GTEx database.

Publisher's note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.

Zainab Arshad, John F. McDonald

john.mcdonald@biology.gatech.edu

Highlights

Constructing gene co-expression networks based on consensus gene-gene interactions

Differential network analysis identifies nodes lost, conserved, and acquired in cancer

Gene enrichment analysis determines the functional significance of network changes

Arshad & McDonald, STAR Protocols 3, 101432
June 17, 2022
https://doi.org/10.1016/j.xpro.2022.101432

STAR Protocols

CellPress
OPEN ACCESS

Protocol

A computational approach to generate highly conserved gene co-expression networks with RNA-seq data

Zainab Arshad[1] and John F. McDonald[1,2,3,*]

[1]Integrated Cancer Research Center, School of Biological Sciences, Petit Institute for Bioengineering and Bioscience, Georgia Institute of Technology, 315 Ferst Drive, Atlanta, GA 30619, USA

[2]Technical contact

[3]Lead contact

*Correspondence: john.mcdonald@biology.gatech.edu
https://doi.org/10.1016/j.xpro.2022.101432

SUMMARY

We describe a consensus approach for network construction based on fully conserved gene-gene interactions from randomly downsampled data subsets for an unbiased differential analysis of gene co-expression networks. The pipeline allows users to identify network nodes lost, conserved, and acquired in cancer as well as interpret the functional significance of these network changes. For proof of concept, the protocol is used to leverage RNA-seq data of tumor samples from TCGA and healthy tissue samples from the GTEx database.
For complete details on the use and execution of this protocol, please refer to Arshad and McDonald (2021).

BEFORE YOU BEGIN

Select case-control gene expression dataset for analysis

Timing: 1 day to 8 weeks (depending on availability of data)

Differential co-expression analysis is a pair-wise comparison of network structures that helps uncover molecular changes underlying various phenotypes such as disease stages, tissue types, molecular subtypes, or treatment responses (Yang et al., 2014; Hill and McDonald, 2015; Yu et al., 2017; Costa et al., 2018; Andonegui-Elguera et al., 2021; Paci et al., 2021).

This protocol describes the specific steps for differential co-expression analysis of lung squamous cell carcinoma (LUSC) samples (Hammerman et al., 2012) from TCGA (dbGaP:phs000178.v11.p8) (Weinstein et al., 2013) and healthy lung tissue samples from GTEx (dbGaP:phs000424.v8.p2) (Ardlie et al., 2015). We have also used this protocol to analyze eight other cancer types (Glioblastoma multiforme-GMB, thyroid carcinoma-THCA, breast-BRCA, lung adenocarcinoma-LUAD, skin cutaneous melanoma-SKCM, kidney renal clear cell carcinoma-KIRC, ovarian-OV, acute myeloid leukemia-LAML) published in (Arshad and McDonald, 2021).

RNA-seq data for this analysis can also be derived from human or non-human primary tissues or a relevant cell line for the phenotype of interest. Sequence alignment data from high-throughput sequencing (HTS) experiments can be used to generate gene expression profiles with analysis packages such as HTSeq (Anders et al., 2015). Information about processing and generating counts for RNA-Seq alignment with HTSeq can be found in the Tutorials section of the package.

Experimental variation between data from different sources can confound biological interpretation and therefore needs to be corrected before analysis. This can be performed in RNA-seq data using batch correction tools such as PEER (Stegle et al., 2012) or ComBat (Zhang et al., 2020). Conversely, publicly available batch-corrected data from resources such as the Recounts2 project (Mounir et al., 2019) can also be used.

In this protocol for LUSC, we implement our analysis on HT-seq counts data for 500 primary tumor samples and 374 healthy tissue samples from the Recounts2 project with TCGAbiolinks. Normalized counts (e.g., TPM and RPKM) can also be used here, in which case the data scaling step (step 3) under 'before you begin' can be skipped.

Make sure that your dataset of choice provides sufficient samples (n > 100) to generate robust networks for each class. Previous work by Liesecke et al. reported inconsistencies in performance of networks constructed with fewer than a hundred samples (Liesecke et al., 2019).

Data preparation for co-expression network analysis

Timing: 1 h

1. After the dataset is selected, download and preprocess before the network construction step. For this workflow, we utilize functions from the following packages:

```
> library(TCGAbiolinks)
> library(SummarizedExperiment)
> library(biomaRt)
> library(limma)
> library(TCGAutils)
> library(recount)
```

2. The following script queries data produced by the Recount2 project with *TCGAquery_recount2* and stores it in a *RangedSummarizedExperiment(rse)* object. The rse objects provide metadata on expression features and samples, as well as the count matrix.

```
> tissue = "lung"
> recount.gtex <- TCGAquery_recount2(project="gtex", tissue=tissue)
> recount.tcga <- TCGAquery_recount2(project="tcga", tissue=tissue)
```

3. Next, the coverage counts provided by Recounts2 are scaled by read length to get read counts. This rescaling is necessary because most RNA-seq analysis tools expect read count matrices as input. The following code rescales, and extracts counts data from the rse objects:

```
> eset.gtex <-assays(scale_counts(recount.gtex$gtex_lung, round = TRUE))$counts
> eset.tcga <-assays(scale_counts(recount.tcga$tcga_lung, round = TRUE))$counts
```

4. Next, the TCGA dataframe is filtered for the *sample type* and *molecular subtype* that need to be investigated. Here we set these options to 'Primary Tumor' and 'Lung Squamous Cell Carcinoma' respectively.

```
> eset.tcga.both <- eset.tcga[,which(colData(recount.tcga$tcga_lung)$gdc_cases.samples.sample_type=="Primary Tumor" & colData(recount.tcga$tcga_lung)$gdc_cases.project.name=="Lung Squamous Cell Carcinoma")]
```

5. The filtered count matrix for tumor is then merged with the normal tissue matrix into one dataframe for normalization.

```
> dataPrep <-merge(as.data.frame(eset.gtex), as.data.frame(eset.tcga.both), by=0, all=TRUE)
```

6. Next, RNA-seq read counts are normalized for GC-content with *TCGAanalyze_Normalization* from the TCGAbiolinks package. This normalization step is necessary to address the sequence bias against GC-rich and GC-poor fragments in RNA-Seq data. EDASeq package needs to be installed prior to running the function.

```
> BiocManager::install("EDASeq")
```

The *geneInfo* argument in TCGAanalyze_Normalization should be set to geneInfoHT when using gene expression data aligned against hg38, otherwise if your data are aligned against hg19, *geneInfo* should be set to geneinfo.

```
> rownames(dataPrep)<-dataPrep$Row.names

> dataPrep$Row.names<-NULL

> dataNorm <- TCGAanalyze_Normalization(tabDF = dataPrep,

                      geneInfo = geneInfoHT,

                      method = "gcContent")
```

7. Quantile filtering is then applied to normalized counts with a cutoff of 0.25 with TCGAanalyze_Filtering to remove genes with expression levels lower than the 25th percentile of the data.

```
> dataFilt <- TCGAanalyze_Filtering(tabDF = dataNorm,

                    method = "quantile",

                    qnt.cut = 0.25)
```

8. Finally, expression matrices for normal and cancer samples are saved into csv files with samples for rows and genes for columns, as follows:

```
> write.csv(dataFilt[,colnames(eset.gtex)], paste0("GTEx-",tissue,".csv"), quote = FALSE)

>  write.csv(dataFilt[,colnames(eset.tcga)], paste0("TCGA-cancer-",tissue,".csv"), quote = FALSE)
```

Differential expression analysis

Timing: 30 min

To determine if genes associated with co-expression changes in cancer exhibit any significant expression changes in cancer relative to normal tissue, differential gene expression levels for these genes need to be determined in LUSC samples.

There is a wide variety of tools available for gene expression analysis (DEA) of RNA-seq data (Costa-Silva et al., 2017) that can be employed in this step.

In this protocol we use the *TCGAanalyze_DEA* function from TCGAbiolinks that in turn utilizes the edgeR package (Robinson et al., 2010). We choose to use this function as it is available as a part of the TCGAbiolinks package. However, other independent DEA tools such as DESeq2 may also be used in this step.

9. The *method* argument is set to 'glmLRT', which fits a negative binomial generalized log-linear model to the read counts for each gene in the two conditions (normal and tumor) and then computes gene-wise exact tests for differences in the means of these distributions to determine differential expression.

```
> dataDEGs <- TCGAanalyze_DEA(mat1 = group1, mat2 = group2,
                      Cond1type = "Tumor",
                      Cond2type = "Normal",
                      pipeline = "edgeR",
                      method = "glmLRT")
```

10. In our work, the thresholds of p-value <0.05 and $|\log_2(\text{fold change})| \geq 1$ are set to define differentially expressed genes (DEGs). Alternatively, adjusted p-values using the False Discovery Rate correction may also be used to identify significant DEGs here.
11. DEGs are further divided into up-regulated DEGs with log2 FC$\geq$1 and down-regulated DEGs with log2 FC ≤ -1.

Cancer driver genes

Timing: 30 min

12. To compare differential expression and co-expression patterns of cancer driver genes (oncogenes and tumor suppressor genes) in the LUSC dataset, a comprehensive list of cancer driver genes is downloaded from Cancer Gene Census, COSMIC (Sondka et al., 2018); https://cancer.sanger.ac.uk/census.

This dataset includes 315 oncogenes and 315 tumor suppressors. Out of these, 72 genes are labeled both oncogene and tumor suppressor gene in COSMIC.

13. Save these gene sets in comma separated value (csv) files for use during the protocol.

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Software and algorithms		
R (3.6.1)	(The R Foundation, 2018)	https://www.r-project.org/
RStudio (2020)	(RStudio, 2020)	https://rstudio.com/
Python (3.8.8)	(Python Software Foundation, 2016)	https://www.python.org/
TCGAbiolinks (2.12.6)	(Mounir et al., 2019)	https://bioconductor.org/packages/release/bioc/html/TCGAbiolinks.html
Recount (1.10.13)	(Collado-Torres et al., 2017)	https://bioconductor.org/packages/release/bioc/html/recount.html
Limma (3.42.2)	(Smyth, 2005)	https://bioconductor.org/packages/release/bioc/html/limma.html
biomaRt (2.40.5)	(Durinck et al., 2005)	https://bioconductor.org/packages/release/bioc/html/biomaRt.html
SummarizedExperiment (1.14.1)	(Morgan et al., 2020)	https://bioconductor.org/packages/release/bioc/html/SummarizedExperiment.html
edgeR (3.26.8)	(Robinson et al., 2010)	https://bioconductor.org/packages/release/bioc/html/edgeR.html
Devtools (2.4.3)	(Wickham et al., 2021)	https://cran.r-project.org/web/packages/devtools/index.html
Pandas (1.2.4)	(McKinney, 2010)	https://pandas.pydata.org/

STEP-BY-STEP METHOD DETAILS

Step 1: Construction of co-expression networks

Timing: 1 day

In this step, consensus co-expression networks will be constructed using interactions (significant gene-gene correlations) found conserved across all iterations of the network construction pipeline.

1. For next steps in the pipeline (network construction, analysis, and visualization) we switched to Python. Alternately, packages from R may also be used here.
2. We will utilize functions from the following python packages:

```
> import pandas as pd

> import networkx as nx
```

3. In this step we use the gene expression matrix to calculate pair-wise correlation between all gene pairs using *pandas.DataFrame.corr* with the *method* argument set to 'pearson'. The returned object is an adjacency matrix (n by n) where each cell represents the correlation coefficient between a gene pair, with an associated p-value.

```
> from scipy.stats import pearsonr

> corr = data.corr()

> corr_pval = data.corr(method=lambda x, y: pearsonr(x, y)[1]) - np.eye(*corr.shape)
```

4. Next, we select a threshold(r) for the correlation coefficient to define significant gene relationships to be included in our networks. Previous work from the McDonald lab found, after evaluating different values of correlation thresholds (0.70–0.99), that networks of random signals could

appear to be connected for values of r < 0.85 (Hill and McDonald, 2015). Hence, we select gene pairs satisfying |r| > 0.85, p-value < 0.05.

```
> corr = corr.mask(corr_pval >= 0.05)

> links = corr.stack().reset_index()

> links = links[(links[0] >= 0.85)|(links[0] <= -0.85)]
```

5. Save the network as an adjacency list of edges with three columns (1) node 1, (2) node 2 and (3) correlation strength.

```
> links.columns = ['node 1', 'node 2','correlation strength']

> links.to_csv('adjacency_list.csv', index=False)
```

6. The number of edges in a co-expression matrix is highly dependent on the number of samples in small datasets. Therefore, to ensure fair comparison between datasets with variable sample sizes, we used a consensus approach to define our signal networks. For this:
   a. Both normal and cancer datasets are randomly down sampled to a hundred subsets each consisting of a hundred samples.

```
> normal_data = normal_data.sample(n = 100)

> cancer_data = cancer_data.sample(n = 100)
```

   b. Next, we generate a network for each data subset following steps 3–5. `Links` from each of the 100 networks are combined into a single list called `links_100_iterations`, for normal and cancer data independently.
   c. A frequency is assigned to each unique interaction from all hundred networks, based on their occurrence across the hundred iterations, for cancer and normal individually. Save each of these interactions along with their frequencies in a csv file for use during the protocol.

```
> link_frequency = links_100_iterations.value_counts()
```

   d. Finally, network edges found in each of the hundred iterations are used to define the final networks. Save the final networks as dataframes with two columns (1) node 1 and (2) node 2, for normal (df_normal) and cancer (df_cancer) individually.

```
> consensus_edges = list(link_frequency [link_frequency >= 100].keys())

> df_normal = pd.DataFrame([[i.split(' ')[0],i.split(' ')[1]] for i in consensus_edges],
columns=['node 1', 'node 2'])

# Repeat these steps (6 b to d) for cancer_data and store the corresponding consensus links in
df_cancer
```

7. All relationships (correlations) are treated as unsigned in all downstream analyses because nearly 100% of the correlations in the dataset are positive values.

Step 2: Network structural analysis

Timing: 30 min

In this step, consensus co-expression networks from the previous step will be analyzed to identify and compare network connectivity and network similarity between normal and cancer networks. Additionally, network hub nodes will be defined for use later in the protocol.

8. For a quantitative comparison of overall network connectivity and to explore relative changes in network connectivity between normal and cancer, record and compare:
   a. total number of gene correlations (network vertices) found in any of the hundred iterations of network construction from step 6b, for normal and cancer samples.
   b. fully conserved network connections found in all hundred iterations of the pipeline from step 6d, for normal and cancer samples.
9. Besides the quantitative comparison of network structures described above, a more literal way to calculate network similarity is a node by node and edge by edge comparison. Jaccard similarity can be used as a measure of network similarity between normal and cancer networks using *sklearn.metrics.jaccard_score* based on shared network nodes and edges.
   a. Read the networks saved in step 5 into dataframes each with three columns: node 1, node 2 and correlation strength. Next, load these co-expression networks into Networkx using the following code:

```
> G_normal = nx.from_pandas_edgelist(df_normal,'node 1','node 2')

> G_cancer = nx.from_pandas_edgelist(df_cancer,'node 1','node 2')
```

   b. Get the set of unique (1) nodes and (2) edges from normal and cancer networks and calculate their Jaccard similarities. The following code implements network similarity between cancer and normal networks based on node overlap:

```
> list1 = G_cancer.nodes()

> list2 = G_normal.nodes()

> intersection = len(list(set(list1).intersection(list2)))

> union = (len(list1) + len(list2)) - intersection

> Jaccard_similarity = return float(intersection) / union
```

10. We defined network hub nodes as the top 2% most connected nodes in the network. In the literature, hub nodes have been defined anywhere from the top 1%–10% of network nodes with highest connectivity. For this protocol we choose to restrict hub nodes to a conservative cutoff of 2%.
11. Sort network nodes based on their degree and select the top 0.02*n genes as hubs, where n is the total number of genes:

```
> n_genes = sorted(G_normal.degree, key=lambda x: x[1], reverse=True)

> threshold = int(len(n_genes)*0.02)

> n_hubnodes = pd.DataFrame(n_genes,columns=['GeneId','Degree'])[0:threshold]['GeneId']
```

Step 3: Identify differentially co-expressed gene lists

Timing: 15 min

Comparing network connectivity in normal and cancer datasets, we found that while a large percentage of network nodes are lost in cancer, some of these interactions are also conserved in disease state. Additionally, cancer networks also acquire some unique interactions not previously found in normal tissue. In this step, we will identify the proportions of normal nodes lost and conserved in cancer, as well as the proportion of acquired cancer nodes.

12. For comparative analysis, network nodes were divided into three differential cases of interest (1) lost nodes, (2) conserved nodes and (3) acquired nodes, where:
    a. Lost nodes represent genes displaying significant correlations across normal samples that are not co-expressed in cancer samples. This is calculated as:

```
> lost_nodes = G_normal.nodes() - G_cancer.nodes()
```

    b. Conserved nodes are genes with connections in both normal and cancer samples that are calculated as:

```
> conserved_nodes = G_normal.nodes().intersection(G_cancer.nodes())
```

    c. Acquired nodes are genes that display gene-gene correlations in cancer samples only and are calculated as:

```
> Acquired_nodes = G_cancer.nodes() - G_normal.nodes()
```

13. The proportion of network nodes in each of these categories is recorded.

```
> lost_normal_nodes = len(lost_nodes)/len(G_normal.nodes())
> conserved_normal_nodes = len(conserved_nodes)/len(G_normal.nodes())
> acquired_cancer_nodes = len(acquired_nodes)/len(G_cancer.nodes())
```

14. In addition to this, we also record network nodes that are differentially expressed and network nodes that are cancer drivers (COSMIC).

Step 4: Gene enrichment analysis

Timing: 30 min

In this step, we use functional enrichment analysis to determine if genes (nodes) that are lost, conserved, or acquired in cancer networks are differentially enriched for biological functions.

15. There are a number of tools available for conducting gene ontology analysis that can be used here. For this project we used GSEApy, which is an Enrichr pathway analysis package within the Python wrapper. GSEApy is a popular tool for the gene enrichment analysis of gene lists, based on Fisher's exact test, that includes more than a hundred gene set databases with over 180,000 gene sets in multiple categories.
16. Submit these gene lists (lost, conserved and acquired nodes) for genome ontology (GO) enrichment analysis using the function *gseapy.enrichr* as follows:

```
> import gseapy
> enr = gseapy.enrichr(gene_list = nodes,
            description = 'pathway',
```

```
            gene_sets = ['GO_Biological_Process'], organism='Human',
            cutoff=0.5)
```

17. Biological process terms with adjusted p < .05 were considered significant and recorded.

```
> enr[enr.results['Adjusted p-value']<0.05]
```

EXPECTED OUTCOMES

Network connectivity in LUSC cancer relative to healthy normal tissue

Integrating all unique edges from the 100 iterations of network construction step (Step by step 1) resulted in 15717 and 23191 significantly correlated changes in expression (edges) between pairs of genes (nodes) in cancer and normal respectively. Of these, 297 edges in cancer and 2261 edges in normal were consistently found in each model iteration and these conserved edge lists were used to define our final consensus networks. Figure 1 in the study by (Arshad and McDonald, 2021) presents results for this overall reduction in network complexity observed in nine different cancer types, including LUSC.

Dividing the edges into lost, conserved and acquired categories, we observed a dramatic loss of 2175 (96.2%) network connections (edges) relative to those present in their respective normal precursor tissues. Only 86 (3.8%) of the connections present in the precursor normal tissues were conserved in the cancers, whereas cancer exhibits 211 (71.04%) newly acquired connections. These results provide quantitative evidence for the dynamic rewiring of normal biological pathways in cancer, marked by significant reduction in normal edges as well as acquisition of novel cancer gene interactions.

Comparison between differentially connected network nodes and DEGs

26 (4.53%) of lost network nodes and 12 (5.59%) of acquired cancer nodes displayed a significant change in expression between cancer and normal. Remarkably, none of the network hub genes lost or acquired in cancer displayed any significant changes in expression in the cancer. Further, network nodes lost in cancer included 12 COSMIC genes while 6 of the acquired cancer genes were cancer driver genes. Of this only one lost network COSMIC gene (nodes) and none of acquired network COSMIC genes (nodes) displayed a significant change in expression between cancer and normal.

Overall, we found that genes displaying significant changes in gene-gene connections in the transition from normal to cancer cells were often not differentially expressed. This observation was consistent across different cancer types [Figure 2 in (Arshad and McDonald, 2021)].

Functional enrichment results

Functional enrichment analysis of genes lost, conserved, and acquired in cancer resulted in a total of 5, 8 and 6 hits, respectively. Translation, macromolecule biosynthesis and protein transport were the biological process most significantly enriched for lost nodes. Conserved nodes in cancer networks were predominantly enriched for antigen processing and presentation pathways, along with Ras and ARF protein signaling pathways. While acquired cancer nodes were enriched for regulation of immune processes including negative regulation of interferon-beta and interleukin-12 production.

LIMITATIONS

There are some limitations of co-expression analysis that need to be noted. For example, co-expression networks are sensitive to sample sizes with small datasets potentially giving rise to false positives. To minimize this possibility, we limited our networks to include conserved edges from multiple iterations of down-sampled equally (>100) sized datasets. In addition, it is important to keep in mind

that while correlated changes in the expression of genes associated with cancer are reliable system-level indicators of changes in network structure, this does not imply that each correlation is necessarily indicative of a direct (transcription factor - target gene) regulatory change on the molecular level. Indeed, it is likely that a number of the significantly correlated changes computationally identified between genes (represented as edges between nodes in the network) are the result of indirect regulatory interactions on the molecular level. For example, gene A could directly regulate gene B, and gene B could directly regulate gene C, in this case a connection identified between gene A and gene C would be an example of an indirect regulatory interaction.

TROUBLESHOOTING

Problem 1

Errors while using Recounts2 package (in step 2, before you begin). These could include errors related to function arguments as well as connection errors.

Potential solution

TCGA and GTEx expression data used in this protocol were downloaded from the Recount2 project, with TCGAbiolinks. This resource provides batch-corrected data from 2041 different studies (including TCGA and GTEx projects), processed under one single pipeline to avoid any technical variabilities that may affect downstream integrative analysis. For errors related to querying and downloading and preprocessing Recounts2 data, refer to this workflow (http://bioconductor.org/packages/release/workflows/html/recountWorkflow.html).

Problem 2

"Error in names(y) <- 1:length(y) : 'names' attribute [2] must be the same length as the vector [0]" while using the TCGAanalyze_Normalization function (in step 6, before you begin).

Potential solution

This error indicates that the data set includes Ensemble IDs that are not included in the geneInfo data set. To avoid this, the *geneInfo* argument in TCGAanalyze_Normalization should be set to geneInfoHT when using gene expression data aligned against hg38, otherwise if your data are aligned against hg19, *geneInfo* should be set to geneinfo.

Problem 3

Error "Segmentation fault (core dumped)" with *pandas.DataFrame.corr* (in step 3, step-by-step method details).

Potential solution

This error indicates that python has crashed while processing a very high dimensional dataset and the allocated RAM is full. To work around this error, you can increase the stack that your operating system allocates for the python process or breakdown the counts matrix into smaller chunks, calculate correlations for each subset and merge the results into one matrix. A code snippet for the latter is included below:

```
> n = len(data.columns)

# breakdown the counts matrix into two

> subset_1 = data.iloc[:,list(range(0,n//2))]

> subset_2 = data.iloc[:,list(range(n//2,n))]

# calculate correlations for each subset

> _1st = corr(subset_1, subset_1) #1st quadrant

> _2nd = corr(subset_1, subset_2) #2nd quadrant
```

```
> _3rd = corr(subset_2, subset_1) #3rd quadrant

> _4th = corr(subset_2, subset_2) #4th quadrant

# merge results into one matrix

> correlation_matrix = pd.concat([pd.concat([_1st, _2nd]), pd.concat([_3rd, _4th])],
axis=1)
```

RESOURCE AVAILABILITY

Lead contact

Request for further information should be directed to and will be fulfilled by the lead contact, John F. McDonald (john.mcdonald@biology.gatech.edu).

Materials availability

This study did not generate new unique reagents.

Data and code availability

This paper analyzes existing, publicly available data. Details of these datasets are listed in the key resources table. This paper does not report original code.

ACKNOWLEDGMENTS

This research was supported by the Mark Light Integrated Cancer Research Center Student Fellowship, the Deborah Nash Endowment Fund, and the Ovarian Cancer Institute (Atlanta). The results shown here are based upon data generated by the TCGA Research Network: http://cancergenome.nih.gov/.

AUTHOR CONTRIBUTIONS

Z.A. and J.F.M. conceived the project; Z.A. conducted the computational analyses; Z.A. and J.F.M. wrote the paper.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

Anders, S., Pyl, P.T., and Huber, W. (2015). HTSeq—a Python framework to work with high-throughput sequencing data. Bioinformatics *31*, 166–169. https://doi.org/10.1093/BIOINFORMATICS/BTU638.

Andonegui-Elguera, S.D., Zamora-Fuentes, J.M., Espinal-Enríquez, J., and Hernández-Lemus, E. (2021). Loss of long distance co-expression in lung cancer. Front. Genet. *12*, 625741. https://doi.org/10.3389/fgene.2021.625741.

Ardlie, K.G., DeLuca, D.S., Segrè, A.V., Sullivan, T.J., Young, T.R., Gelfand, E.T., Trowbridge, C.A., Maller, J.B., Tukiainen, T., Lek, M., et al. (2015). The genotype-tissue expression (GTEx) pilot analysis: multitissue gene regulation in humans. Science *348*, 648–660. https://doi.org/10.1126/science.1262110.

Arshad, Z., and McDonald, J.F. (2021). Changes in gene-gene interactions associated with cancer onset and progression are largely independent of changes in gene expression. iScience *24*, 103522. https://doi.org/10.1016/j.isci.2021.103522.

Collado-Torres, L., Nellore, A., Kammers, K., Ellis, S.E., Taub, M.A., Hansen, K.D., Jaffe, A.E., Langmead, B., and Leek, J.T. (2017). Reproducible RNA-seq analysis using recount2. Nat. Biotechnol. *35*, 319–321. https://doi.org/10.1038/nbt.3838.

Costa-Silva, J., Domingues, D., and Lopes, F.M. (2017). RNA-seq differential expression analysis: an extended review and a software tool. PLoS ONE *12*, e0190152. https://doi.org/10.1371/journal.pone.0190152.

Costa, R.L., Boroni, M., and Soares, M.A. (2018). Distinct co-expression networks using multi-omic data reveal novel interventional targets in HPV-positive and negative head-and-neck squamous cell cancer. Sci. Rep. *8*, 1–13. https://doi.org/10.1038/s41598-018-33498-5.

Durinck, S., Moreau, Y., Kasprzyk, A., Davis, S., De Moor, B., Brazma, A., and Huber, W. (2005). BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. Bioinformatics *21*, 3439–3440. https://doi.org/10.1093/bioinformatics/bti525.

Hammerman, P.S., Voet, D., Lawrence, M.S., Voet, D., Jing, R., Cibulskis, K., Sivachenko, A., Stojanov, P., McKenna, A., Lander, E.S., et al. (2012). Comprehensive genomic characterization of squamous cell lung cancers. Nature *489*, 519–525. https://doi.org/10.1038/nature11404.

Hill, C.G., and McDonald, J.F. (2015). Evidence and potential clinical significance of changes in gene network interactions in ovarian cancer. J. Biomed. Eng. Inform. *2*, 1. https://doi.org/10.5430/jbei.v2n1p1.

Liesecke, F., De Craene, J.O., Besseau, S., Courdavault, V., Clastre, M., Vergès, V., Papon, N., Giglioli-Guivarc'h, N., Glévarec, G., Pichon, O., and Duge de Bernonville, T. (2019). Improved gene co-expression network quality through expression dataset down-sampling and network aggregation. Sci. Rep. *9*, 1–16. https://doi.org/10.1038/s41598-019-50885-8.

McKinney, W. (2010). Data structures for statistical computing in Python. Proc. 9th Python Sci. Conf. *445*, 56–61. https://doi.org/10.25080/majora-92bf1922-00a.

Morgan, M., Obenchain, V., Hester, J., and Pagès, H. (2020). Bioconductor - SummarizedExperiment. https://bioconductor.org/packages/3.11/bioc/html/SummarizedExperiment.html.

Mounir, M., Lucchetta, M., Silva, T.C., Olsen, C., Bontempi, G., Chen, X., Noushmehr, H., Colaprico, A., and Papaleo, E. (2019). New functionalities in the TCGAbiolinks package for the study and integration of cancer data from GDC and GTEX. PLoS Comput. Biol. *15*, e1006701. https://doi.org/10.1371/journal.pcbi.1006701.

Paci, P., Fiscon, G., Conte, F., Wang, R.S., Farina, L., and Loscalzo, J. (2021). Gene co-expression in the interactome: moving from correlation toward causation via an integrated approach to disease module discovery. NPJ Syst. Biol. Appl. *7*, 1–11. https://doi.org/10.1038/s41540-020-00168-0.

Python Software Foundation (2016). Welcome to Python.org. https://www.python.org/.

Robinson, M.D., McCarthy, D.J., and Smyth, G.K. (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics *26*, 139–140. https://doi.org/10.1093/BIOINFORMATICS/BTP616.

RStudio (2020). RStudio | open source & professional software for data science teams. https://www.rstudio.com/.

Smyth, G.K. (2005). Limma: linear models for microarray data. In Bioinformatics and Computational Biology Solutions Using R and Bioconductor (Springer), pp. 397–420. https://doi.org/10.1007/0-387-29362-0_23.

Sondka, Z., Bamford, S., Cole, C.G., Ward, S.A., Dunham, I., and Forbes, S.A. (2018). The COSMIC Cancer Gene Census: describing genetic dysfunction across all human cancers. Nat. Rev. Cancer *18*, 696–705. https://doi.org/10.1038/s41568-018-0060-1.

Stegle, O., Parts, L., Piipari, M., Winn, J., and Durbin, R. (2012). Using probabilistic estimation of expression residuals (PEER) to obtain increased power and interpretability of gene expression analyses. Nat. Protoc. *7*, 500–507. https://doi.org/10.1038/nprot.2011.457.

The R Foundation (2018). R: the R project for statistical computing. https://www.r-project.org/.

Weinstein, J.N., Collisson, E.A., Mills, G.B., Shaw, K.R.M., Ozenberger, B.A., Ellrott, K., Sander, C., Stuart, J.M., Chang, K., Creighton, C.J., et al. (2013). The cancer genome atlas pan-cancer analysis project. Nat. Genet. *45*, 1113–1120. https://doi.org/10.1038/ng.2764.

Wickham, H., Hester, J., and Chang, W. (2021). Tools to Make Developing R Packages Easier - Package "Devtools".

Yang, Y., Han, L., Yuan, Y., Li, J., Hei, N., and Liang, H. (2014). Gene co-expression network analysis reveals common system-level properties of prognostic genes across cancer types. Nat. Commun. *5*, 3231. https://doi.org/10.1038/ncomms4231.

Yu, W., Zhao, S., Wang, Y., Zhao, B.N., Zhao, W., and Zhou, X. (2017). Identification of cancer prognosis-associated functional modules using differential co-expression networks. Oncotarget *8*, 112928–112941. https://doi.org/10.18632/oncotarget.22878.

Zhang, Y., Parmigiani, G., and Johnson, W.E. (2020). ComBat-seq: batch effect adjustment for RNA-seq count data. NAR Genom. Bioinform. *2*, lqaa078. https://doi.org/10.1093/nargab/lqaa078.