Using MOMA, a package for inference of Master Regulators proteins using multi-omic data

Evan O. Paull[1], Sunny Jones[1], Mariano J. Alvarez[2], Federico M. Giorgi[2], and Andrea Califano[1]

[1]Department of Systems Biology, Columbia University, 1130 St. Nicholas Ave., New York
[2]DarwinHealth Inc, New York, NY, USA
[3]Department of Pharmacy and Biotechnology, University of Bologna, Via Selmi 3, Bologna Italy

November 12, 2019

1 Overview of MOMA

2 Citation

3 Installation of *moma* package

MOMA requires the R-system (`http://www.r-project.org`), and the following R packges for full functionality, including graphics and plotting functions.

- dplyr
- grDevices
- graphics
- MKmisc
- methods
- parallel
- readr
- reshape2
- RColorBrewer
- survival
- stats
- tibble
- utils

4 Getting started

First load the library into the R session:

```
> library(moma)
```

Explore the test data, and confirm that we have 100 test samples and 2506 VIPER [?] inferred proteins in the test viper matrix.

```
> names(gbm.example)
```

[1] "vipermat" "cindy" "pval.map" "rawsnp" [5] "fusions" "gene.loc.mapping" "clinical" "fCNV" [9] "mutSig" "rawcnv"

```
> dim(gbm.example$vipermat)
```

[1] 2506 100

```
> dim(gbm.example$rawsnp)
```

[1] 8327 100

Next, we determine the biological pathways we'd like to use to aid in our inference task. In this case, we're using the CINDy algorithm as well as protein-protein interactions predicted by the PrePPI structure-based algorithm.

```
> pathways <- list()
> pathways[['cindy']] = gbm.example$cindy
> pathways[['preppi']] = gbm.example$pval.map
```

5 Generating the *moma* object

Generate the MOMA object from the input data we have. Note that we need:

- VIPER matrix, continuous values
- SNP matrix, binary 0/1 values
- CNV matrix, continuous values
- Fusion matrix, binary 0/1 values
- pathway list
- Gene blacklist
- output folder

```
> momaObj <- moma.constructor(gbm.example$vipermat, gbm.example$rawsnp,
+         gbm.example$rawcnv, gbm.example$fusions, pathways,
+         gene.blacklist=gbm.example$mutSig,
+         output.folder='gbm-test/',
+         gene.loc.mapping=gbm.example$gene.loc.mapping)
```

6 MOMA Analysis on GBM Data

The first step, 'runDIGGIT()' will run the DIGGIT inference algorithm [?] to find statistical interactions between VIPER-inferred proteins and genomic events.

The 'makeInteractions()' function will infer robust computational predictions using all the provided data, including the Conditional Inference of Network Dynamics (CINDy) algorithm [?]. The option 'cindy.only=FALSE' will allow interactions without evidence from CINDy, but with strong evidence from other sources, to be used.

The 'Rank()' function will create a final ranking of candidate Master Regulators for this cohort of patient samples.

```
> momaObj$runDIGGIT(fCNV=gbm.example$fCNV)
> momaObj$makeInteractions(cindy.only=FALSE)
> momaObj$Rank(use.cindy=TRUE)
```

Clustering of the samples, using the protein ranks computed in the last step, can then be performed. Multiple cluster solutions will be calculated, ranging from 2 to 10 clusters by default. The silhouette or reliability score of each can be assessed to determine an optimal 'k'; here, we are selecting the k=2 solution. Genomic saturation analysis is then performed on each cluster with the 'saturationPlots()' function, allowing us to find the key proteins that explain the majority of genomic events supplied in the SNP/CNV and Fusion matrices.

```
> clustering.solutions <- momaObj$Cluster()
```

[1] "using pearson correlation with weighted vipermat" [1] "testing clustering options, k = 2..15" [1] "2 0.618073073073074" [1] "3 0.818053053053053" [1] "4 0.75460960960961" [1] "5 0.79466966966967" [1] "6 0.765690690690691" [1] "7 0.781246246246247" [1] "8 0.759434434434434" [1] "9 0.739864864864865" [1] "10 0.652727727727728" [1] "11 0.702447447447448" [1] "12 0.697382382382383" [1] "13 0.626146146146146" [1] "14 0.565650650650651" [1] "15 0.569354354354355"

```
> # Pick the k=5 cluster and save this to the moma Object
> momaObj$sample.clustering <- clustering.solutions[[1]]$clustering
> # generate genomic saturation statistics and plot data
> momaObj$saturationPlots()
```

[1] "Analyzing cluster 1" [1] "Top : 100 regulators" [1] "Warning: could not map entrez IDs to Cytoband for IDS, skipping..." character(0) [1] "Warning: could not map entrez IDs to Cytoband for IDS, skipping..." character(0) [1] "Warning: could not map entrez IDs to Cytoband for IDS, skipping..." character(0) [1] "Warning: could not map entrez IDs to Cytoband for IDS, skipping..." [1] "84059" [1] "Computing coverage for sample TCGA-14-1825-01" [1] "Computing coverage for sample TCGA-16-0846-01" [1] "Computing coverage for sample TCGA-06-0171-02" [1] "Computing coverage for sample TCGA-32-2634-01" [1] "Computing coverage for sample TCGA-26-1442-01" [1] "Computing coverage for sample TCGA-32-1980-01" [1] "Computing coverage for sample TCGA-28-5216-01" [1] "Computing coverage for sample TCGA-06-0178-01" [1] "Computing coverage for sample TCGA-15-1444-01" [1] "Computing coverage for sample TCGA-02-0055-01" [1] "Computing coverage for sample TCGA-19-2629-01" [1] "Computing coverage for sample TCGA-06-0132-01" [1] "Computing coverage for sample TCGA-06-2570-01" [1] "Computing coverage for sample TCGA-02-2483-01" [1] "Computing coverage for sample TCGA-06-0174-01" [1] "Computing coverage for sample TCGA-28-5215-01" [1] "Computing coverage for sample TCGA-02-0047-01" [1] "Computing coverage for sample TCGA-41-3915-01" [1] "Computing coverage for sample TCGA-26-5134-01" [1] "Computing coverage for sample TCGA-41-2571-01" [1] "Computing coverage for sample TCGA-27-2521-01" [1] "Computing coverage for sample TCGA-06-0649-01" [1] "Computing coverage for sample TCGA-27-1830-01" [1] "Computing coverage for sample TCGA-06-5411-01" [1] "Computing coverage for sample TCGA-12-0618-01" [1] "Computing coverage for sample TCGA-06-0129-01" [1] "Computing coverage for sample TCGA-19-1390-01" [1] "Computing coverage

for sample TCGA-06-2558-01” [1] ”Analyzing cluster 2” [1] ”Top : 100 regulators” [1] ”Computing coverage for sample TCGA-76-4931-01” [1] ”Computing coverage for sample TCGA-06-5418-01” [1] ”Computing coverage for sample TCGA-06-0747-01” [1] ”Computing coverage for sample TCGA-32-2632-01” [1] ”Computing coverage for sample TCGA-06-0743-01” [1] ”Computing coverage for sample TCGA-06-5856-01” [1] ”Computing coverage for sample TCGA-28-5213-01” [1] ”Computing coverage for sample TCGA-06-5859-01” [1] ”Computing coverage for sample TCGA-12-0616-01” [1] ”Computing coverage for sample TCGA-06-0645-01” [1] ”Computing coverage for sample TCGA-15-0742-01” [1] ”Computing coverage for sample TCGA-14-0790-01” [1] ”Computing coverage for sample TCGA-06-2565-01” [1] ”Computing coverage for sample TCGA-06-5858-01” [1] ”Computing coverage for sample TCGA-14-0789-01” [1] ”Computing coverage for sample TCGA-06-0157-01” [1] ”Computing coverage for sample TCGA-12-3653-01” [1] ”Computing coverage for sample TCGA-27-2519-01” [1] ”Computing coverage for sample TCGA-12-3652-01” [1] ”Computing coverage for sample TCGA-06-0749-01” [1] ”Computing coverage for sample TCGA-12-5295-01” [1] ”Computing coverage for sample TCGA-76-4926-01” [1] ”Computing coverage for sample TCGA-02-2486-01” [1] ”Computing coverage for sample TCGA-32-2615-01” [1] ”Computing coverage for sample TCGA-28-1747-01” [1] ”Computing coverage for sample TCGA-27-1835-01” [1] ”Computing coverage for sample TCGA-14-2554-01” [1] ”Computing coverage for sample TCGA-28-5209-01” [1] ”Computing coverage for sample TCGA-06-0745-01” [1] ”Computing coverage for sample TCGA-28-5208-01” [1] ”Computing coverage for sample TCGA-12-0619-01” [1] ”Computing coverage for sample TCGA-27-1832-01” [1] ”Computing coverage for sample TCGA-28-1753-01” [1] ”Computing coverage for sample TCGA-14-0817-01” [1] ”Computing coverage for sample TCGA-06-2563-01” [1] ”Computing coverage for sample TCGA-06-0882-01” [1] ”Computing coverage for sample TCGA-06-0744-01” [1] ”Computing coverage for sample TCGA-06-0168-01” [1] ”Computing coverage for sample TCGA-19-2620-01” [1] ”Computing coverage for sample TCGA-08-0386-01” [1] ”Computing coverage for sample TCGA-06-1804-01” [1] ”Computing coverage for sample TCGA-06-2559-01” [1] ”Computing coverage for sample TCGA-06-0158-01” [1] ”Computing coverage for sample TCGA-02-2485-01” [1] ”Computing coverage for sample TCGA-06-5408-01” [1] ”Computing coverage for sample TCGA-27-1837-01” [1] ”Computing coverage for sample TCGA-28-2509-01” [1] ”Computing coverage for sample TCGA-19-5960-01” [1] ”Computing coverage for sample TCGA-28-2513-01” [1] ”Computing coverage for sample TCGA-41-2572-01” [1] ”Computing coverage for sample TCGA-14-1034-02” [1] ”Computing coverage for sample TCGA-06-0184-01” [1] ”Computing coverage for sample TCGA-26-5135-01” [1] ”Computing coverage for sample TCGA-06-0646-01” [1] ”Computing coverage for sample TCGA-76-4925-01” [1] ”Computing coverage for sample TCGA-12-3650-01” [1] ”Computing coverage for sample TCGA-06-0219-01” [1] ”Computing coverage for sample TCGA-41-4097-01” [1] ”Computing coverage for sample TCGA-06-0750-01” [1] ”Computing coverage for sample TCGA-16-1045-01” [1] ”Computing coverage for sample TCGA-06-5414-01” [1] ”Computing coverage for sample TCGA-32-1970-01” [1] ”Computing coverage for sample TCGA-06-0130-01” [1] ”Computing coverage for sample TCGA-06-0210-02” [1] ”Computing coverage for sample TCGA-28-2514-01” [1] ”Computing coverage for sample TCGA-06-0644-01” [1] ”Computing coverage for sample TCGA-27-2528-01” [1] ”Computing coverage for sample TCGA-14-1823-01” [1] ”Computing coverage for sample TCGA-06-2567-01” [1] ”Computing coverage for sample TCGA-14-0781-01” [1] ”Computing coverage for sample TCGA-19-2624-01” [1] ”Computing coverage for sample TCGA-27-1831-01”

```
> cluster1.checkpoint <- momaObj$checkpoints[[1]]
> print (cluster1.checkpoint[1:10])
```

[1] ”148103” ”9817” ”10224” ”6662” ”57106” ”57615” ”3174” ”80095” [9] ”55900” ”148198”

References

[1] Giorgi, Federico M., et al. ”Inferring protein modulation from gene expression data using conditional mutual information.” PloS one 9.10 (2014): e109569.

[2] Chen, James C., et al. ”Identification of causal genetic drivers of human disease through systems-level analysis of regulatory networks.” Cell 159.2 (2014): 402-414.

[3] Alvarez, Mariano J., et al. ”Functional characterization of somatic mutations in cancer using network-based inference of protein activity.” Nature genetics 48.8 (2016): 838.