Supplementary material for

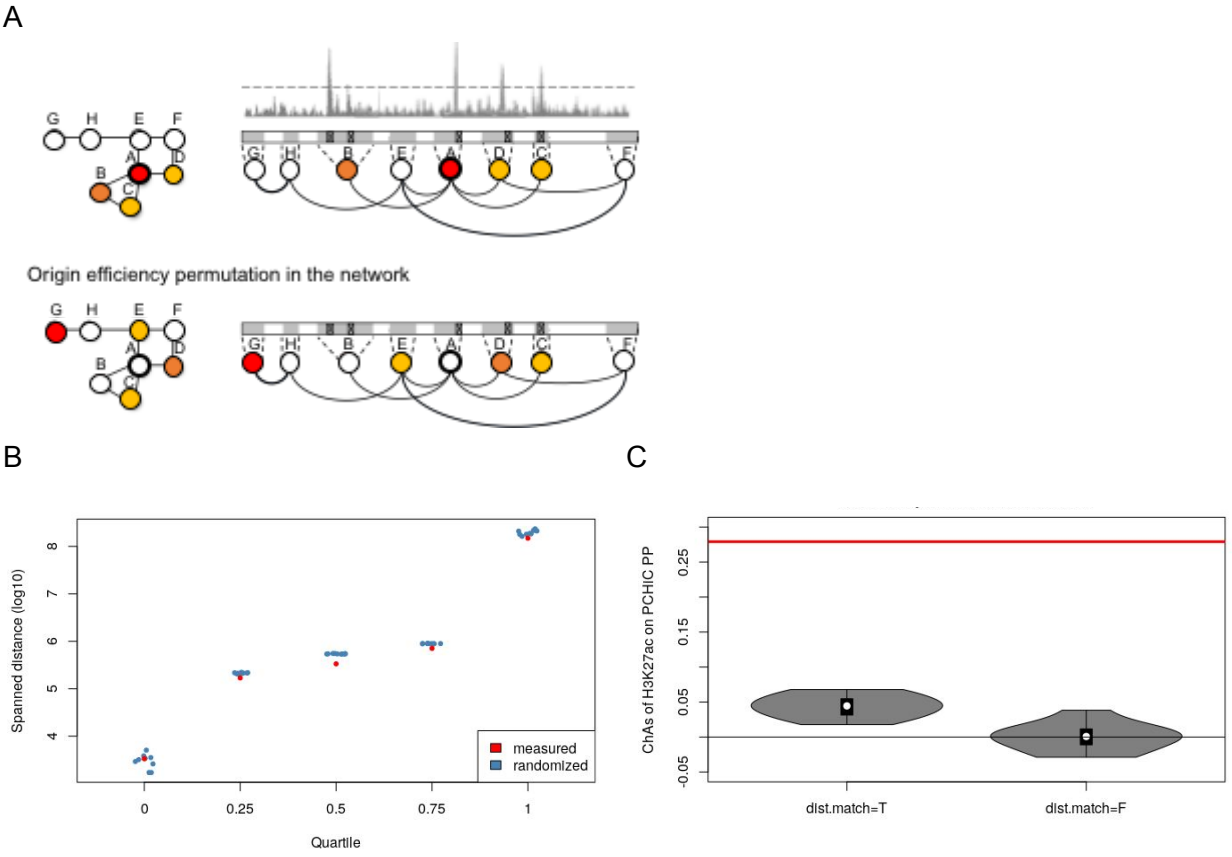
Using GARDEN-NET and ChAseR to explore human haematopoietic 3D chromatin interaction networks

Miguel Madrid-Mencía[1,2,3§], Emanuele Raineri[4§], Tran Bich Ngoc Cao[1,2,4] and Vera Pancaldi[1,2,3*]

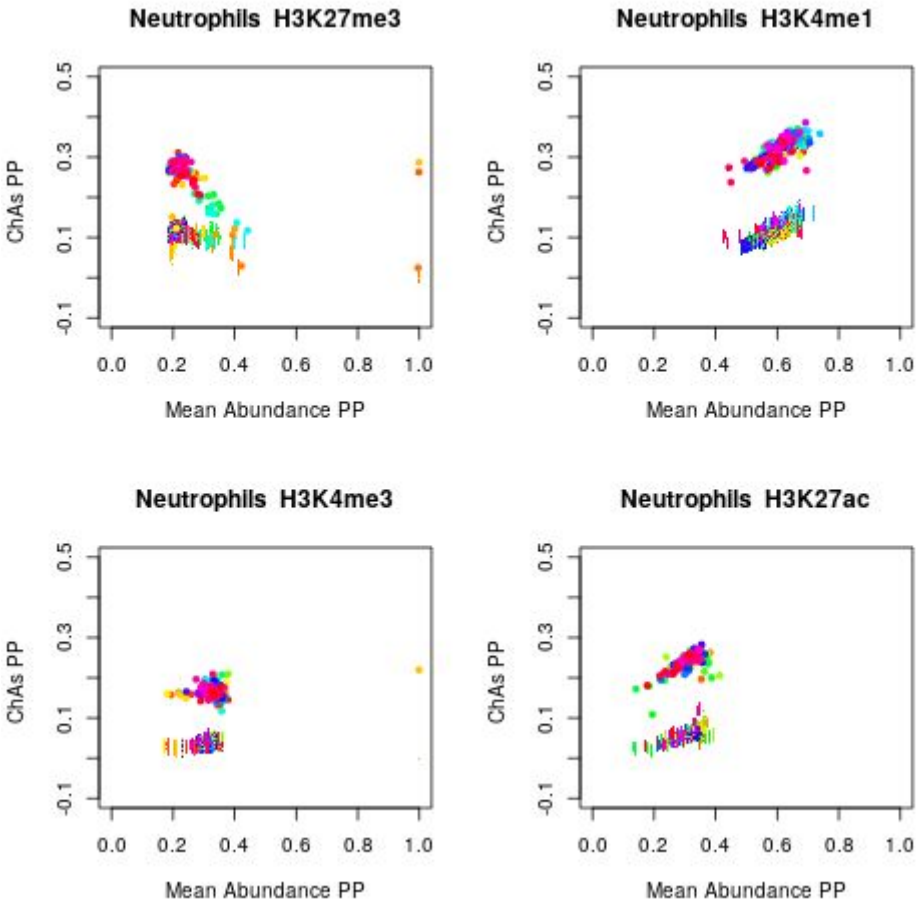
[1] Centre de Recherches en Cancérologie de Toulouse (CRCT), INSERM U1037, Toulouse, 31400, France [2] Université Paul Sabatier III, Toulouse, 31400, Toulouse, France
[3] Barcelona Supercomputing Center, Barcelona, 08034, Spain. [4]CNAG-CRG, Centre for Genomic Regulation (CRG), Barcelona Institute of Science and Technology (BIST), Barcelona, 08028, Spain
[4] Pharmacological, Medical and Agronomical Biotechnology Department, University of Science and Technology of Hanoi,100000, Vietnam.

* To whom correspondence should be addressed. Tel: +33 582741693; Email: vera.pancaldi@inserm.fr
§The authors wish it to be known that, in their opinion, the first 2 authors should be regarded as joint First Authors

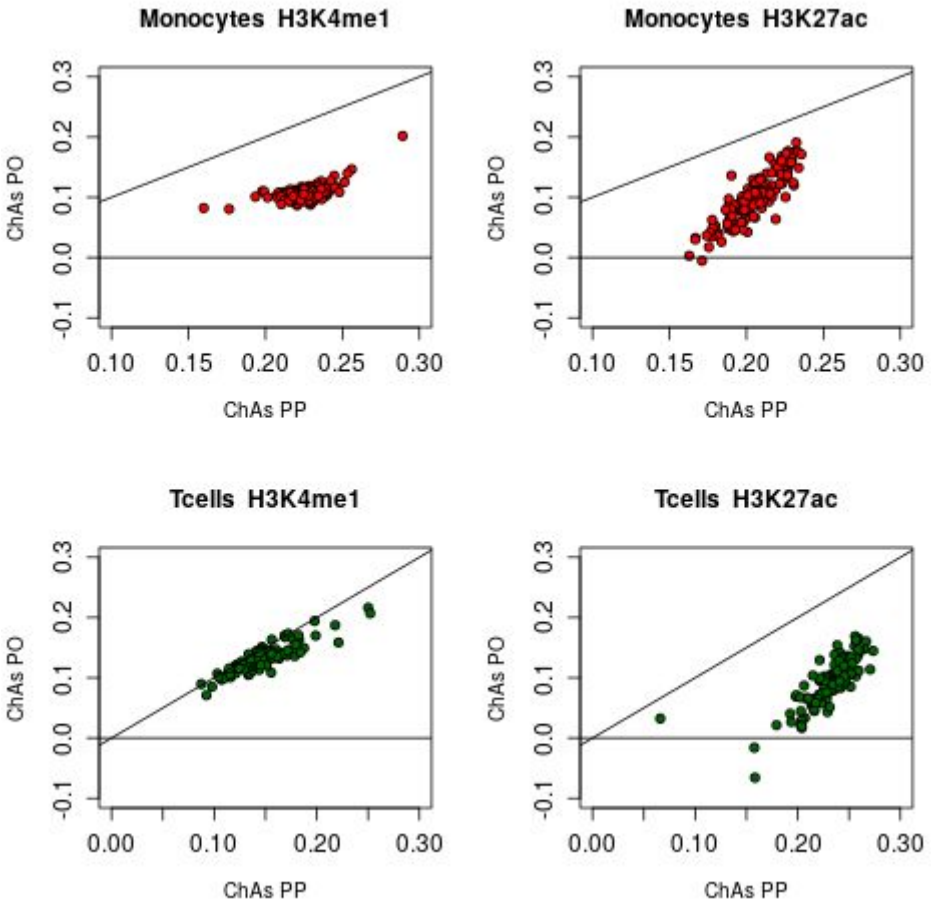
Supplementary Figures



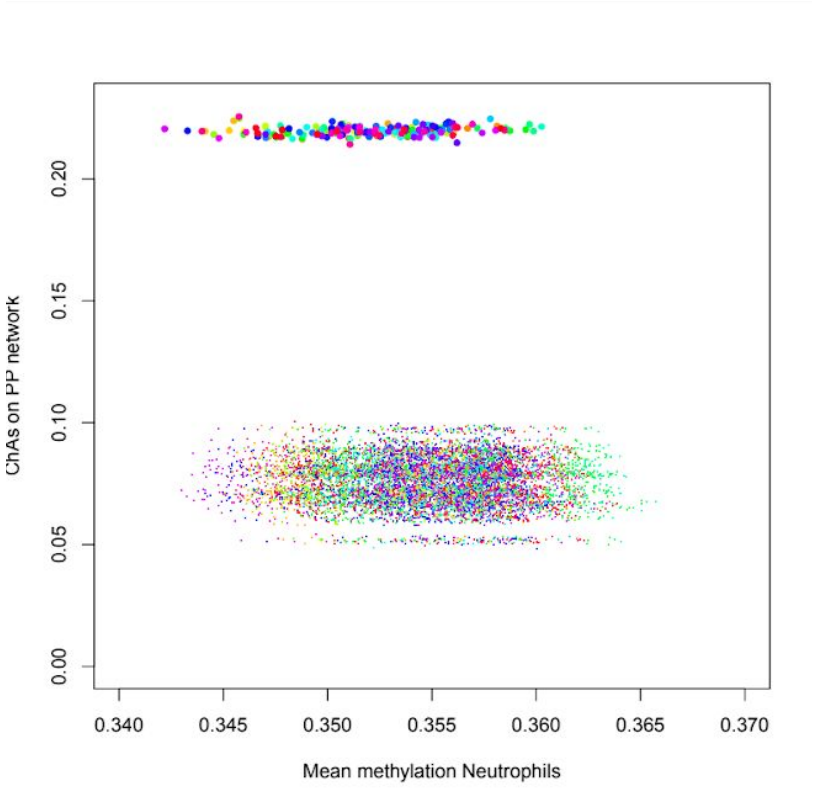
Supplementary Figure S1: Randomization strategies to estimate significance of ChAs values. **A**) Schematic of permutation of features on network. **B**) Randomization by network rewiring with preservation of distribution of distances. **C**) Example of ChAs calculations showing distribution of distance-preserving randomizations.



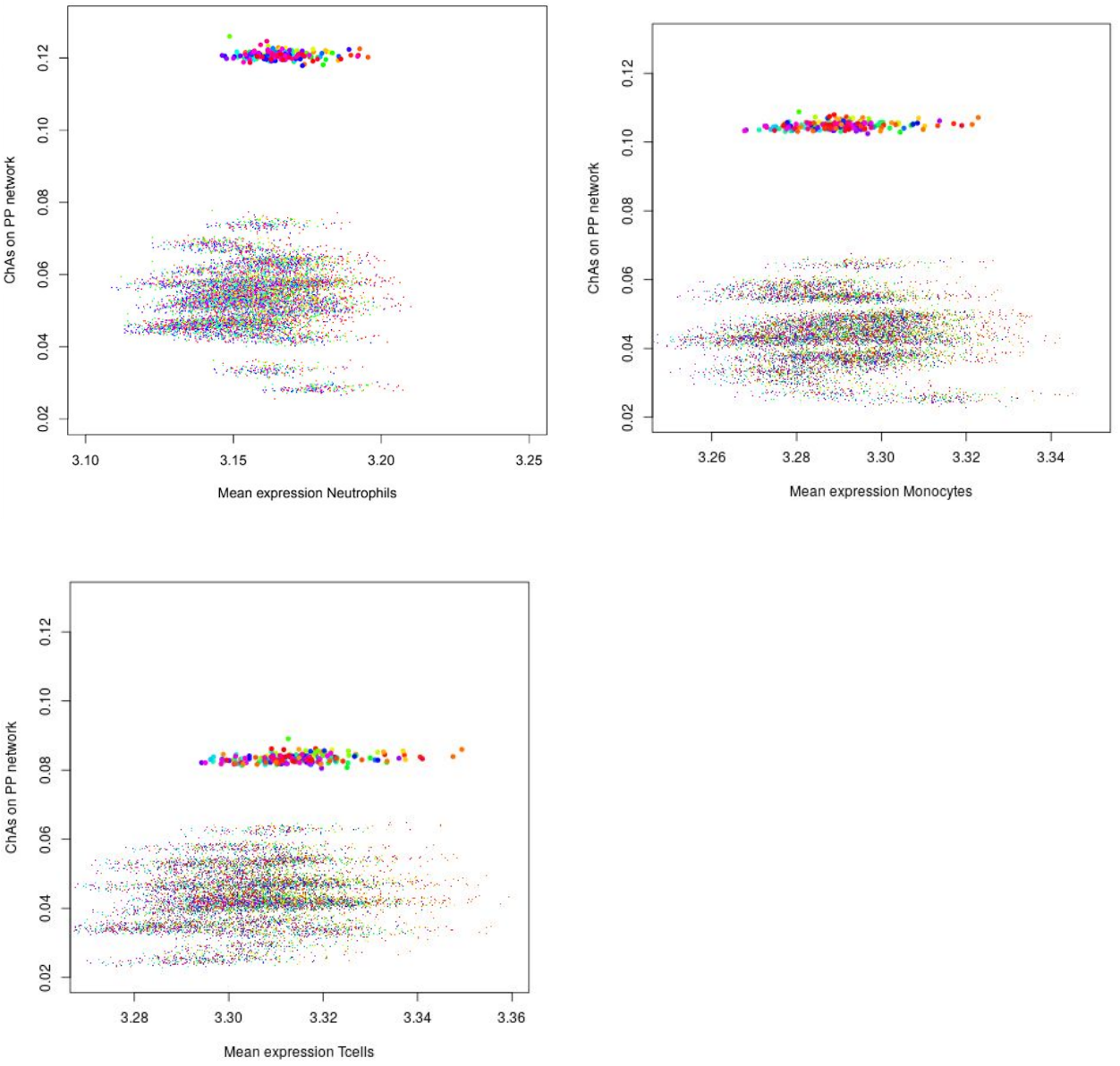
Supplementary Figure S2: ChAs vs Abundance plots in the PP PCHiC subnetwork for neutrophils for 4 histone modifications (large coloured circles) in 150 healthy individuals (Chen et al. 2016), showing 50 distance preserving randomizations (small coloured dots).



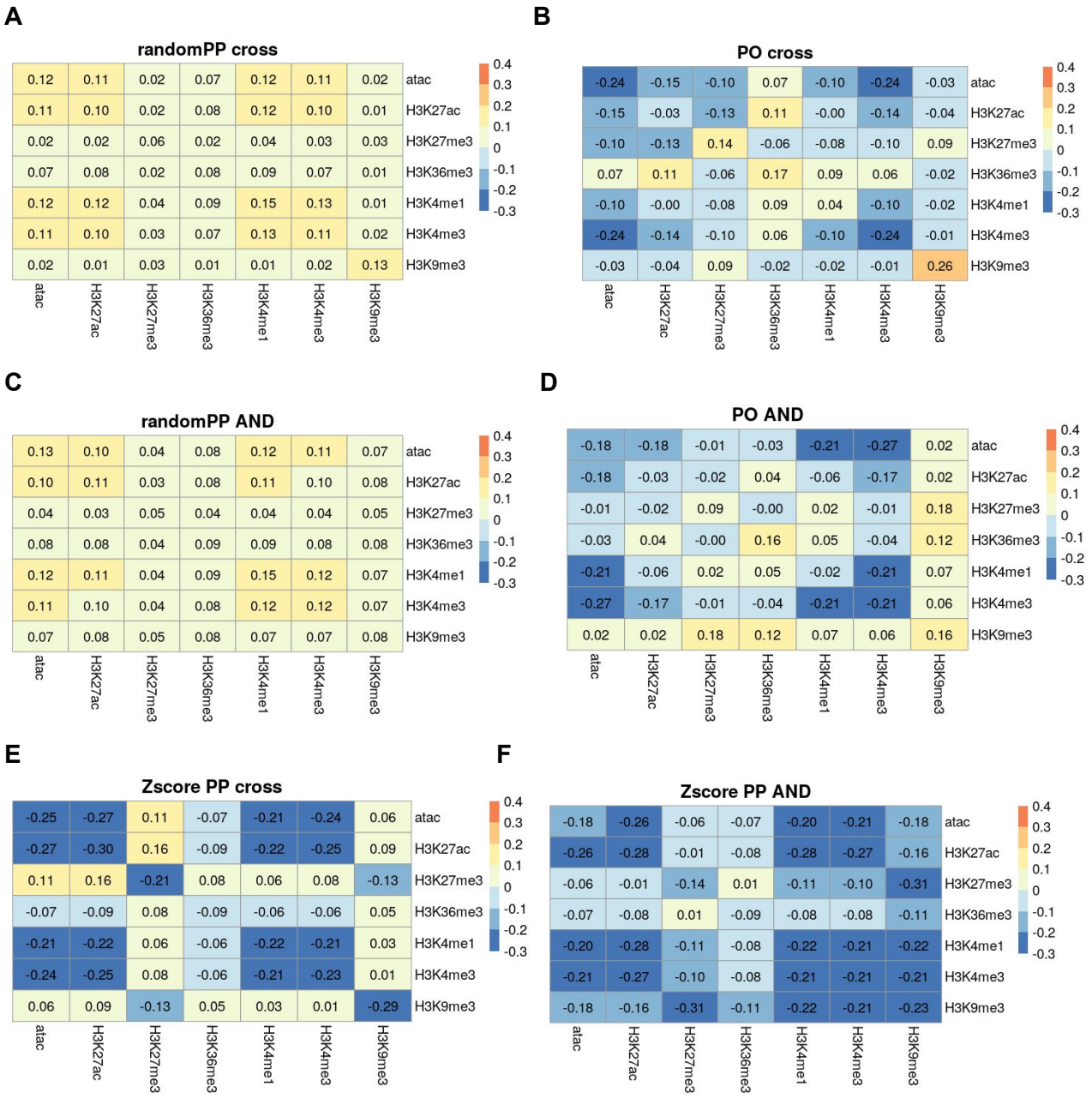
Supplementary Figure S3: Analysis of ChAs of histone modifications on purified samples from over 100 individuals on the PCHiC PP and PO networks for monocytes (top) and T cells (bottom).



Supplementary Figure S4: Analysis of ChAs versus mean value of DNA methylation. ChAs versus mean expression values in neutrophils, showing 50 distance-preserving randomizations shown as coloured dots.

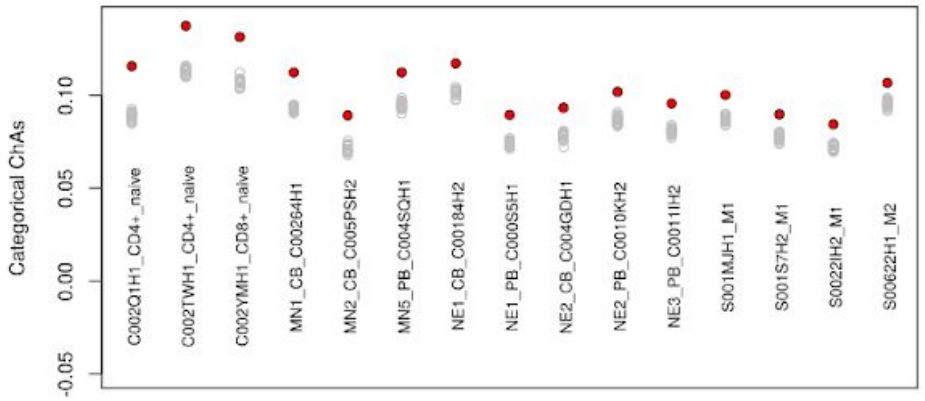


Supplementary Figure S5: Analysis of ChAs versus mean value of expression. ChAs versus mean expression values in neutrophils, monocytes and T cells, showing 50 distance-preserving randomizations shown as coloured dots.

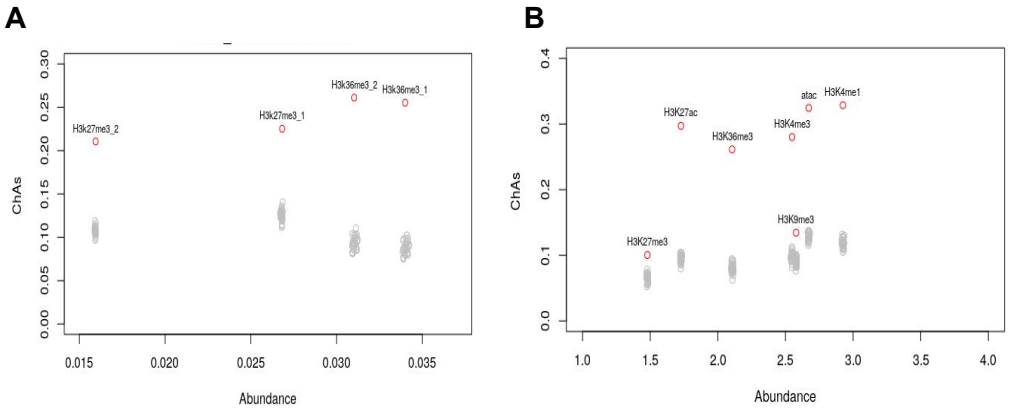


Supplementary Figure S6: Analysis of combined ChAs on PCHiC B cell networks, indicating preferential contacts between chromatin regions having two marks either on both or on either side of the contact (see Figure 4).

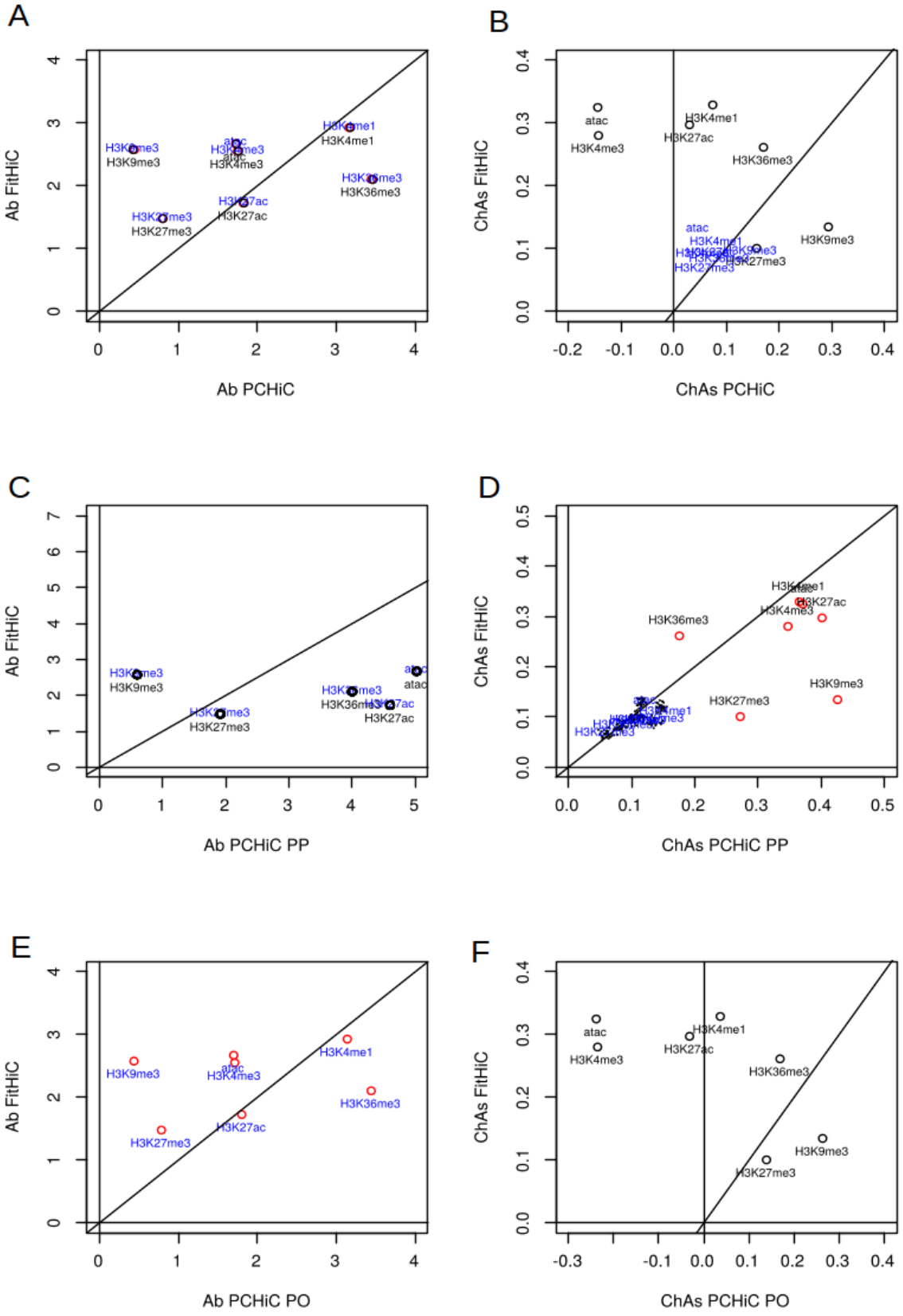
A) Cross-ChAs calculated on distance-preserving randomized PP subnetwork. **B**) Cross-ChAs calculated on PO subnetwork. **C**) AND-ChAs calculated on distance-preserving randomized PP subnetwork. **D**) AND-ChAs calculated on PO subnetwork. **E**) Z-score of Cross-ChAs on PP subnetwork (difference between ChAs and randomized ChAs). **F**) Z-score of And-ChAs on PP subnetwork (difference between ChAs and randomized ChAs).



Supplementary Figure S7: Analysis of ChAs of chromatin states as defined in Carrillo et al. 2017. 11 chromatin states defined using ChromHMM, ChAs calculated using the categorical option in ChAseR. Grey empty circles represent values obtained by distance-preserving randomizations.

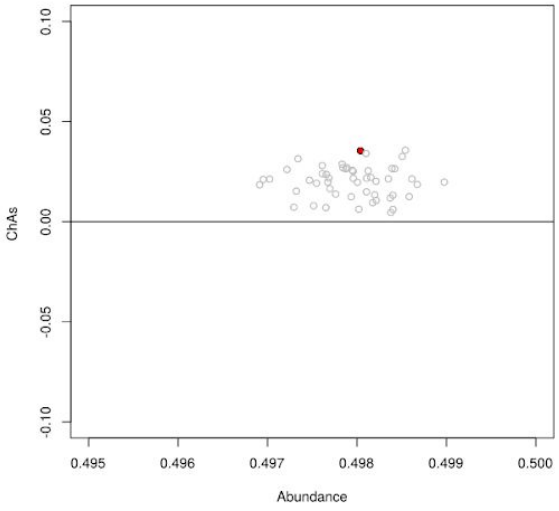


Supplementary Figure S8: Analysis of ChAs of histone modifications from Zhang et al. 2018 (A) and Beekman et al. (B) on the GM06990 lymphoblastoid cell line. Grey empty circles represent values obtained by distance-preserving randomizations.

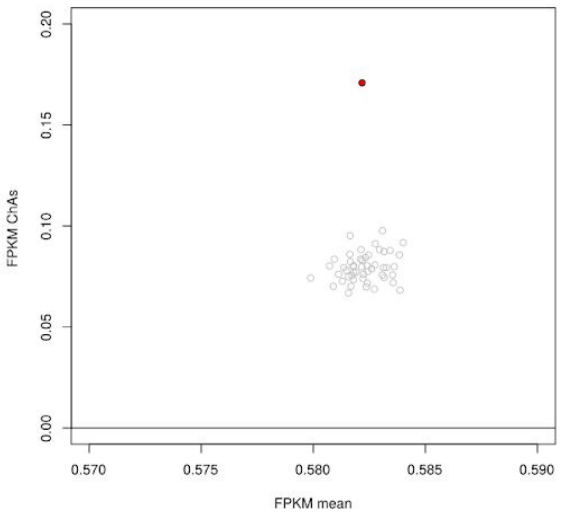


Supplementary Figure S9: Comparison of abundance and ChAs of histone modifications from Beekman et al. 2018 between the PCHiC B cell networks and the FitHiC GM06990 network. Abundance comparison between FitHic and PCHiC (A whole network,C PCHiC PP, E PCHiC PO) and corresponding comparison of ChAs (B whole network, D PCHiC PP, F PCHiC PO).

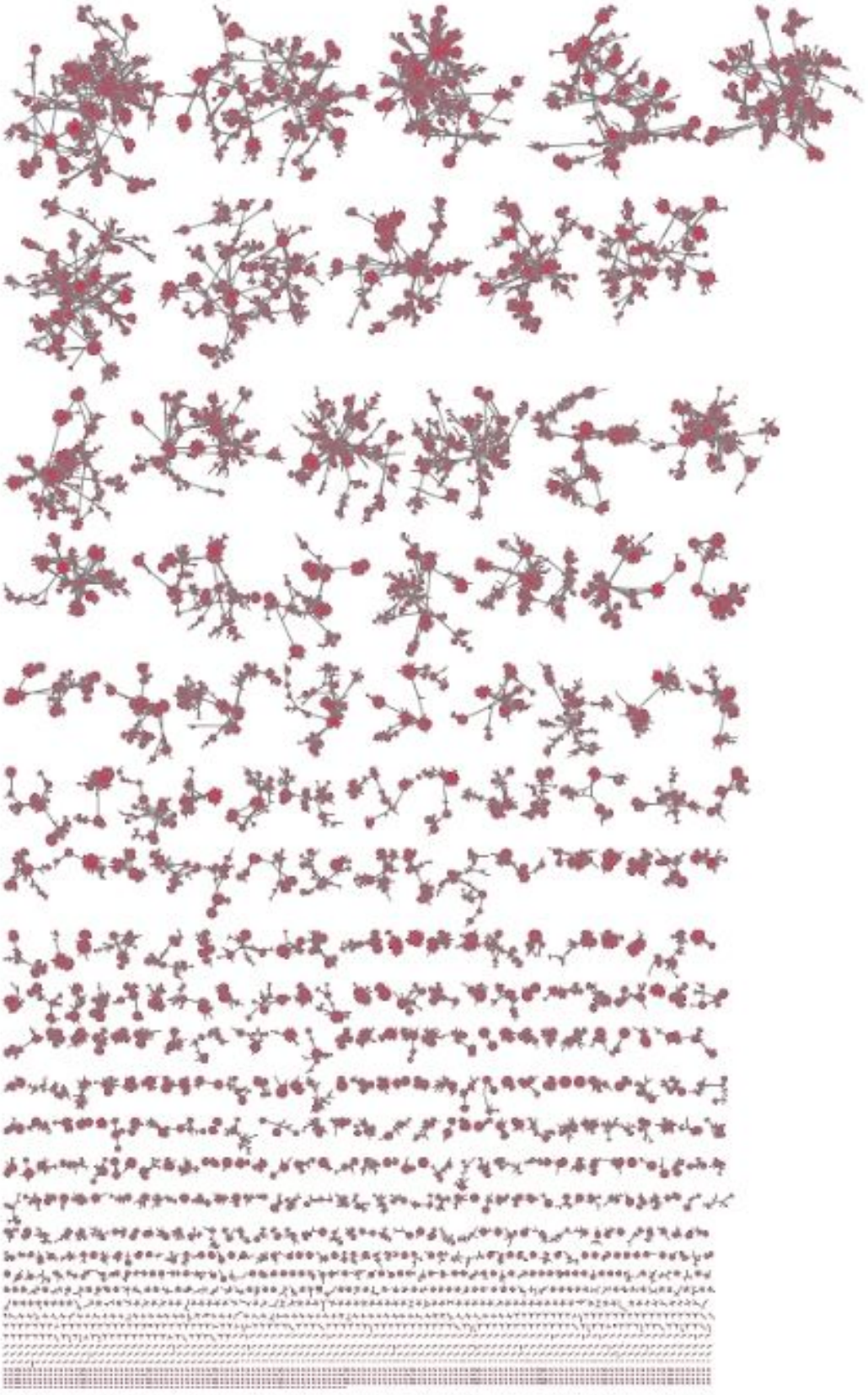
A



B



Supplementary Figure S10: FitHiC network for the GM06690 lymphoblastoid cell line (Zhang et al. 2018). **A**) ChAs analysis of methylation from ENCODE**. B)** ChAs analysis of expression from Zhang et al. 2018. Empty grey circles represent distance-preserving randomizations.

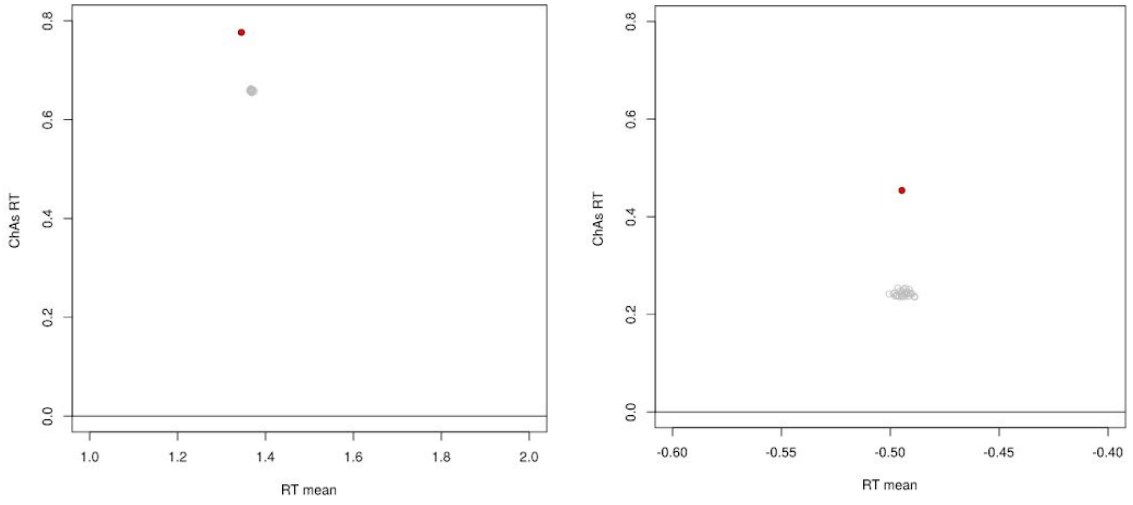


A

B

Supplementary Figure S11: Chromatin networks. **A**) PCHiC neutrophil network (Javierre et al. 2016). **B**) FitHiC network for GM06990 (Zhang et al. 2018)

A **B**



Supplementary Figure S12: Analysis of ChAs of RT in PCHiC and FitHiC networks. Empty grey circles represent distance-preserving randomizations. **A**) PCHiC networks for B cells. **B**) FitHiC networks for GM06990 lymphoblastoid cells.

Supplementary Text for figure 6:

Descriptions of the genes in Figure 6.

DDX23 neighbourhood genes (from String-db.org)

DDX23 - Probable ATP-dependent RNA helicase DDX23; Involved in pre-mRNA splicing and its phosphorylated form (by SRPK2) is required for spliceosomal B complex formation; DEAD-box helicases
RPAP3 - RNA polymerase II-associated protein 3; Forms an interface between the RNA polymerase II enzyme and chaperone/scaffolding protein, suggesting that it is required to connect RNA polymerase II to regulators of protein complex formation; Belongs to the RPAP3 family
'tmem106C':
TMEM106C - Transmembrane protein 106C
RAPGEF3 - Rap guanine nucleotide exchange factor 3; Guanine nucleotide exchange factor (GEF) for RAP1A and RAP2A small GTPases that is activated by binding cAMP. Through simultaneous binding of PDE3B to RAPGEF3 and PIK3R6 is assembled in a signaling complex in which it activates the PI3K gamma complex and which is involved in angiogenesis. Plays a role in the modulation of the cAMP-induced dynamic control of endothelial barrier function through a pathway that is independent on Rho- mediated signaling. Required for the actin rearrangement at cell- cell junctions, such as stress fibers and junctio [...]
HDAC7 - Histone deacetylase 7; Responsible for the deacetylation of lysine residues on the N-terminal part of the core histones (H2A, H2B, H3 and H4). Histone deacetylation gives a tag for epigenetic repression and plays an important role in transcriptional regulation, cell cycle progression and developmental events. Histone deacetylases act via the formation of large multiprotein

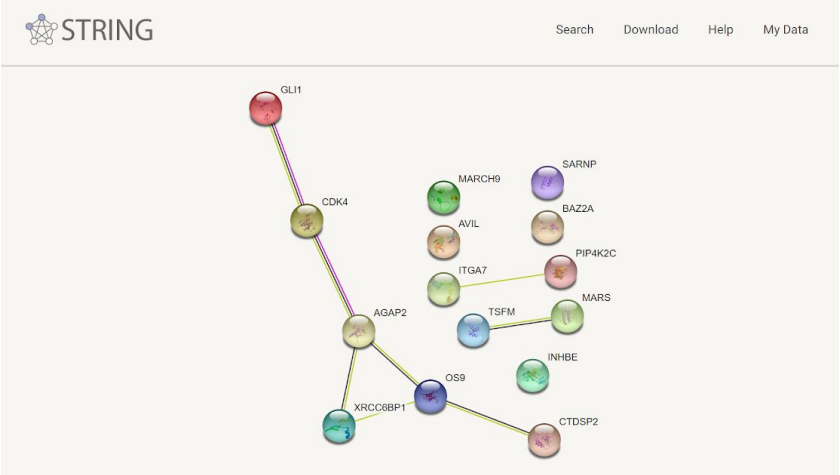
complexes. Involved in muscle maturation by repressing transcription of myocyte enhancer factors such as MEF2A, MEF2B and MEF2C. During muscle differentiation, it shuttles into the cytoplasm, allowing the expression [...]
CCDC184 - Coiled-coil domain containing 184
SENP1 - Sentrin-specific protease 1; Protease that catalyzes two essential functions in the SUMO pathway. The first is the hydrolysis of an alpha-linked peptide bond at the C-terminal end of the small ubiquitin-like modifier (SUMO) propeptides, SUMO1, SUMO2 and SUMO3 leading to the mature form of the proteins. The second is the deconjugation of SUMO1, SUMO2 and SUMO3 from targeted proteins, by cleaving an epsilon-linked peptide bond between the C-terminal glycine of the mature SUMO and the lysine epsilon-amino group of the target protein. Deconjugates SUMO1 from HIPK2. Deconjugates SUMO1 from [...]
ARID2 - AT-rich interactive domain-containing protein 2; Involved in transcriptional activation and repression of select genes by chromatin remodeling (alteration of DNA-nucleosome topology). Required for the stability of the SWI/SNF chromatin remodeling complex SWI/SNF-B (PBAF). May be involved in targeting the complex to different genes. May be involved in regulating transcriptional activation of cardiac genes; AT-rich interaction domain containing.

XRCC6BP1 Neighbourhood genes (from string-db.org)

XRCC6BP1 - Mitochondrial inner membrane protease ATP23 homolog; XRCC6 binding protein 1; Belongs to the peptidase M76 family [a.k.a. KUB3, ATP23, XRCC6BP1-002]
AGAP2 - Arf-GAP with GTPase, ANK repeat and PH domain-containing protein 2; GTPase-activating protein (GAP) for ARF1 and ARF5, which also shows strong GTPase activity. Isoform 1 participates in the prevention of neuronal apoptosis by enhancing PI3 kinase activity. It aids the coupling of metabotropic glutamate receptor 1 (GRM1) to cytoplasmic PI3 kinase by interacting with Homer scaffolding proteins, and also seems to mediate anti-apoptotic effects of NGF by activating nuclear PI3 kinase. Isoform 2 does not stimulate PI3 kinase but may protect cells from apoptosis by stimulating Akt. It also r [...]
CTDSP2 - Carboxy-terminal domain RNA polymerase II polypeptide A small phosphatase 2; Preferentially catalyzes the dephosphorylation of 'Ser- 5' within the tandem 7 residue repeats in the C-terminal domain (CTD) of the largest RNA polymerase II subunit POLR2A. Negatively regulates RNA polymerase II transcription, possibly by controlling the transition from initiation/capping to processive transcript elongation. Recruited by REST to neuronal genes that contain RE-1 elements, leading to neuronal gene silencing in non-neuronal cells. May contribute to the development of sarcomas; CTD family phosphatases
CDK4 - Cyclin-dependent kinase 4; Ser/Thr-kinase component of cyclin D-CDK4 (DC) complexes that phosphorylate and inhibit members of the retinoblastoma (RB) protein family including RB1 and regulate the cell-cycle during G(1)/S transition. Phosphorylation of RB1 allows dissociation of the transcription factor E2F from the RB/E2F complexes and the subsequent transcription of E2F target genes which are responsible for the progression through the G(1) phase. Hypophosphorylates RB1 in early G(1) phase. Cyclin D-CDK4 complexes are major integrators of various mitogenenic and antimitogenic signals. [...]
MARCH9 - E3 ubiquitin-protein ligase MARCH9; E3 ubiquitin-protein ligase that may mediate ubiquitination of MHC-I, CD4 and ICAM1, and promote their subsequent endocytosis and sorting to lysosomes via multivesicular bodies. E3 ubiquitin ligases accept ubiquitin from an E2

ubiquitin-conjugating enzyme in the form of a thioester and then directly transfer the ubiquitin to targeted substrates; Membrane associated ring-CH-type fingers
INHBE - Inhibin beta E chain; Inhibins and activins inhibit and activate, respectively, the secretion of follitropin by the pituitary gland. Inhibins/activins are involved in regulating a number of diverse functions such as hypothalamic and pituitary hormone secretion, gonadal hormone secretion, germ cell development and maturation, erythroid differentiation, insulin secretion, nerve cell survival, embryonic axial development or bone growth, depending on their subunit composition. Inhibins appear to oppose the functions of activins
GLI1 - Zinc finger protein GLI1; Acts as a transcriptional activator. Binds to the DNA consensus sequence 5'- GACCACCCA-3'. May regulate the transcription of specific genes during normal development. May play a role in craniofacial development and digital development, as well as development of the central nervous system and gastrointestinal tract. Mediates SHH signaling. Plays a role in cell proliferation and differentiation via its role in SHH signaling (Probable); Zinc fingers C2H2-type
PIP4K2C - Phosphatidylinositol 5-phosphate 4-kinase type-2 gamma; May play an important role in the production of Phosphatidylinositol bisphosphate (PIP2), in the endoplasmic reticulum
OS9 - Protein OS-9; Lectin which functions in endoplasmic reticulum (ER) quality control and ER-associated degradation (ERAD). May bind terminally misfolded non-glycosylated proteins as well as improperly folded glycoproteins, retain them in the ER, and possibly transfer them to the ubiquitination machinery and promote their degradation. Possible targets include TRPV4; MRH domain containing
AVIL - Advillin; Ca(2+)-regulated actin-binding protein. May have a unique function in the morphogenesis of neuronal cells which form ganglia. Required for SREC1-mediated regulation of neurite-like outgrowth. Plays a role in regenerative sensory axon outgrowth and remodeling processes after peripheral injury in neonates. Involved in the formation of long fine actin-containing filopodia-like structures in fibroblast. Plays a role in ciliogenesis; Gelsolin/villins
SARNP - SAP domain-containing ribonucleoprotein; Binds both single-stranded and double-stranded DNA with higher affinity for the single-stranded form. Specifically binds to scaffold/matrix attachment region DNA. Also binds single- stranded RNA. Enhances RNA unwinding activity of DDX39A. May participate in important transcriptional or translational control of cell growth, metabolism and carcinogenesis. Component of the TREX complex which is thought to couple mRNA transcription, processing and nuclear export, and specifically associates with spliced mRNA and not with unspliced pre-mRNA. TREX is [...]
ITGA7 - Integrin alpha-7; Integrin alpha-7/beta-1 is the primary laminin receptor on skeletal myoblasts and adult myofibers. During myogenic differentiation, it may induce changes in the shape and mobility of myoblasts, and facilitate their localization at laminin-rich sites of secondary fiber formation. It is involved in the maintenance of the myofibers cytoarchitecture as well as for their anchorage, viability and functional integrity. Isoform Alpha-7X2B and isoform Alpha-7X1B promote myoblast migration on laminin 1 and laminin 2/4, but isoform Alpha-7X1B is less active on laminin 1 (In vitr [...]
TSFM - Elongation factor Ts, mitochondrial; Associates with the EF-Tu.GDP complex and induces the exchange of GDP to GTP. It remains bound to the aminoacyl-tRNA.EF- Tu.GTP complex up to the GTP hydrolysis stage on the ribosome; Belongs to the EF-Ts family
BAZ2A - Bromodomain adjacent to zinc finger domain protein 2A; Essential component of the NoRC (nucleolar remodeling complex) complex, a complex that mediates silencing of a fraction of rDNA by recruiting histone-modifying enzymes and DNA methyltransferases, leading to heterochromatin

formation and transcriptional silencing. In the complex, it plays a central role by being recruited to rDNA and by targeting chromatin modifying enzymes such as HDAC1, leading to repress RNA polymerase I transcription. Recruited to rDNA via its interaction with TTF1 and its ability to recognize and bind histone H [...]
ORMDL2 - ORM1-like protein 2; Negative regulator of sphingolipid synthesis



Supplementary Figure S13: STRING (www.string.embl.de) interactions between promoter neighbouring the XRCC6BP1 gene on the PCHiC PP neutrophil network .

References:

Beekman, R. et al. (2018). The reference epigenome and regulatory chromatin landscape of chronic lymphocytic leukemia. Nature Medicine ,24(6):868–880

Chen, L.et al. (2016). Genetic Drivers of Epigenetic and Transcriptional Variation in Human Immune Cells. Cell, 167(5).

Carrillo-de Santa-Pau, et al.(2017). Automatic identification of informative regions with epigenomic changesassociated to hematopoiesis. Nucleic Acids Research, 45(16)

Javierre, B. M. et al. (2016) Lineage-Specific Genome Architecture Links Enhancers and Non-coding Disease Variants to Target Gene Promoters.Cell, 167(5):1369–1384.e19

Zhang, H. et al. (2018). Targeting CDK9 Reactivates Epigenetically Silenced Genes in Cancer. Cell, 175(5):1244–1258.e26.

GARDEN-NET technical details

Frontend

GARDEN-NET[1] web (frontend) is done entirely on the client side. A server is needed (backend) only to send requests to search gene names, Ensembl ID or genomic ranges and to upload features files.
GARDEN-NET is written using the TypeScript[2] language whose transpilation process to JavaScript[3] code is managed by Webpack[4] which also manages all related bundler tasks to distribute the code with all its libraries in only one JavaScript file. A large set of libraries are used for making the web as interactive as possible and especially for the visualization of chromatin networks. To add, remove and update them we use the Yarn[5] package manager to orchestrate all dependencies.
We want to highlight the following libraries:

- Cytoscape.js[6]: the entire visualization and interaction of networks was implemented thanks to this library
- React[7] with Redux[8] and React Router[9]: technologies which allowed us to build webs by isolated orchestrated components

More information is available at https://github.com/VeraPancaldiLab/GARDEN-NET

Backend

GARDEN-NET server (backend) is composed mainly by the HTTP server Nginx[10] which manages all requests and especially the redirection from the web to a lightweight WSGI web application framework called Flask[11] written in Python[12]. This WSGI application communicates web petitions of search and user feature files to their respective R[13] scripts.
To facilitate the management, deployment and isolation of the scripts and the queue job system for the user feature files, the backend is built in three split Docker[14] containers orchestrated by docker-compose[15] which controls the execution order and relaunches them in case of error or server restart. The first Docker container contains the WSGI application with its Python dependencies which listens to the API request along with the R script with their libraries, especially ChAseR[16]. The second Docker container contains the Celery[17] queue job system written in Python and the last Docker container contains Redis[18] coordinated with the Celery container as its memory database.
More information is available at ttps://github.com/VeraPancaldiLab/GARDEN-NET_backend.

[1] https://pancaldi.bsc.es/garden-net
[2] https://www.typescriptlang.org/
[3] https://developer.mozilla.org/en-US/docs/Web/JavaScript
[4] https://webpack.js.org/
[5] https://yarnpkg.com/lang/en/
[6] https://doi.org/10.1093/bioinformatics/btv557
[7] https://reactjs.org/
[8] https://redux.js.org/
[9] https://reacttraining.com/react-router/
[10] https://www.nginx.com/

[11] https://palletsprojects.com/p/flask/
[12] https://www.python.org/
[13] https://www.r-project.org/
[14] https://www.docker.com/
[15] https://docs.docker.com/compose/
[16] https://bitbucket.org/eraineri/chaser/src
[17] http://www.celeryproject.org/
[18] https://redis.io/

ChAseR: computing correlations in chromatin networks

Emanuele Raineri

2020-01-17

version `0.0.0.9`

Installation

```
library(devtools)
devtools::install_bitbucket("eraineri/chaser", build_vignettes=TRUE)
```

Introduction

Chromatin networks are a way of representing contacts between regions of the genome which are not necessarily adjacent but are found to be close to each other in 3D. Such networks may capture, for example, the interaction between a promoter and an enhancer, or between two promoters of genes that are being co-transcribed (see eg (Pancaldi et al. 2016), (Lundberg et al. 2016)). One can associate to any node in the graph other experimental data measured along the genome; for example, chromatin binding peaks as measured by ChipSeq. One useful indicator of the relation between the three dimensional structure of the DNA and such other features is chromatin assortativity (Pancaldi et al. 2016) i.e. Pearson correlation computed across the edges defined by the 3D contacts.

`ChAseR` is an R package which helps in computing chromatin assortativity efficiently. It tackles four aspects of the computation :

- how to represent the edges of the network starting from an existing dataset (e.g. the result of a promoter-capture experiment (Schoenfelder et al. 2018))
- how to associate to each node in the network one or more genomic features
- how to compute the correlation itself.
- how to evaluate the statistical significance of the correlation.

To this purposes `ChAseR` defines a class called `chromnet` and an handful of functions. Objects of class `chromnet` contain a network and optionally features associated to each node of the network. There is no need for the user to access the chromnet object in any way different from using the functions described below.

There are 6 functions which operate on `chromnet` objects, namely:

- `make_chromnet`
- `load_features`
- `subset_chromnet`
- `chas`
- `export`
- `randomize`

This guide is a quick introduction to `ChAseR` and does not cover all the possible options of every function. See the online help for details.

Creating the network

`make_chromnet` maps a properly formatted `data.frame` or file to a `chromnet` object. The data frame must have 6 columns: the first three describe the position (chromosome, start, end) of the first node (a genomic region, for example a bait), the remaining three are the coordinates of an other node which interacts with it.

One can also give a file name as argument to `make_chromnet` as in the example below, in which case the `data.frame` is read from disk. Optionally, it is possible to specify a features matrix.

```
library(chaser)
chromnetfn  <- system.file("extdata",
                           "mESC_wt_and_KO-barebone-format.txt.gz",
                           package = "chaser", mustWork=TRUE)
net  <- chaser::make_chromnet(chromnetfn)
```

`ChAseR` defines a `summary`, `print` and `plot` functions for the `chromnet` class (the `plot` function requires a running instance of Cytoscape).

```
print(net)
```

```
## nodes: 55855
## edges: 139974
## no features
```

The nodes in the network created by chromnet have names which contain the coordinates of the corresponding locus (eg `chr1:10500-11000`).

Associating features to nodes

Features can be mapped on the nodes using `chaser::load_features`. This function accepts the following formats:

- A tsv file which contains the node names in the first column and the features in the remaining columns. This is useful when features have been measured exactly at the coordinates corresponding to the nodes, so that there is no neeed for `ChAseR` to perform an overlap. This format correspond to `type="features_on_nodes"` in the arguments of `chaser::load_features`
- A tsv file (with header) where the first three columns specify a genomic coordinate and the remaining columns are features. One can specify an arbitrary function to take care of the cases when multiple values of the same feature are assigned to the same node (eg one might want to average all the measurements which are assigned to the same node, or take the minimum value, or count how many measurements overlap with a specific node). (`type="features_table"`).
- bed6 files. Again, one can specify an arbitrary function to process features before assigning them.
- bed3 files. In this case what gets mapped on the node is the fraction of the node occupied by the interval defined by each bed3 line.
- MACS2 output files; again what gets mapped to each node is the fraction of the node occupied by a peak.
- chromhmm files: they produce a feature for each state (what gets mapped to each node is the fraction of the node occupied by the state).

```
featfn <- system.file("extdata", "ftable-agchic.txt.gz",
       package = "chaser", mustWork=TRUE)
net <- chaser::load_features(net, featfn, type="features_on_nodes",
     missingv=0)
print(net)
```

```
## nodes: 55855
```

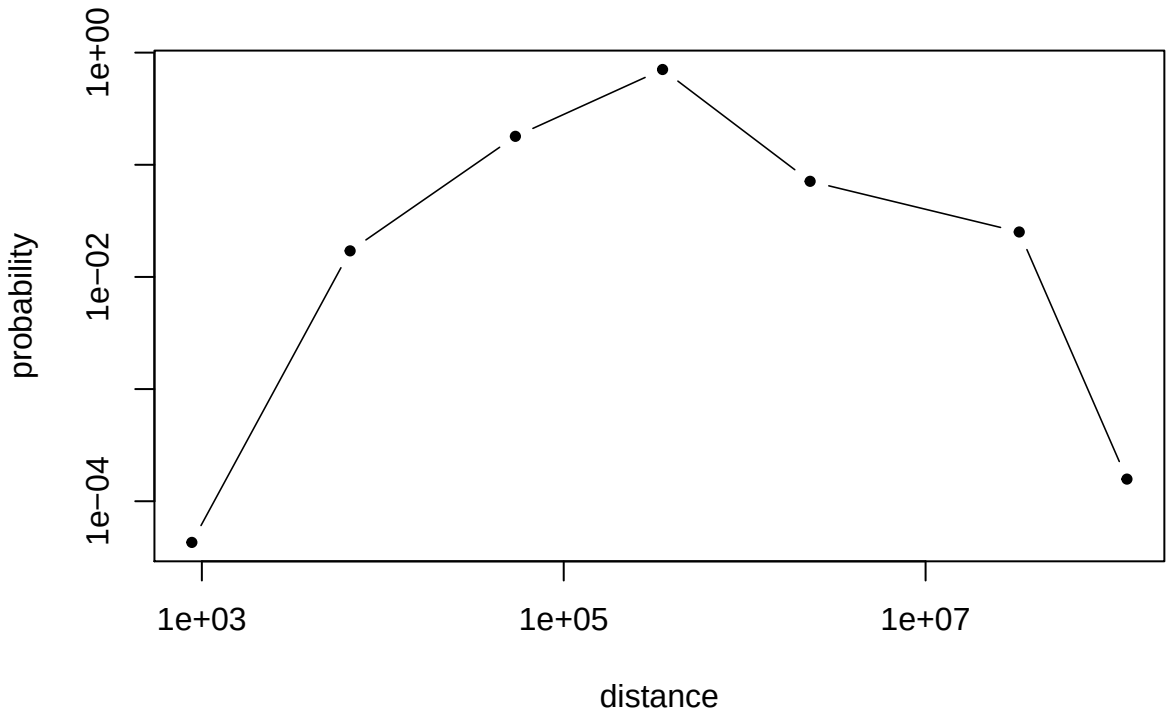
```
## edges: 139974
## 78  feature(s)
```

exporting information from chromnet objects

The function `chaser::export` computes or extracts information from a `chromnet` object. Possible export types are:

- `features` (the default)
- `scatterplot` (to plot a scatterplot of a feature, where each edge is a dot and the values of the features on the two corresponding nodes appear on the x and y axes).
- `nodes` returns a data.frame containing all the nodes in the network
- `igraph` exports the net as an `igraph` object
- `edges` returns a `data.frame` containing all the edges in the network.
- `baits` returns the node names of the baits of a promoter-capture experiments, assuming they were stored in the first 3 columns of the `data.frame` which defined the network.
- `log-binned-distance-density` returns the density of the edge distances (the genomic distance from one end of the edge to the other) estimated in logarithmic bins. The first bin has size 9 and contains edges with distances from 1 to 9, the second bin has size 89 and contains distances from 10 to 99, the n-th bin has size $10^n - 10^{n-1} - 1$ and corresponds to edges with distances from 10^{n-1} to $10^n - 1$. One can plot the average distance of all the edges in a bin and the fraction of edges contained therein, as in the example below:

```
lbd <- chaser::export(net, "log-binned-distance-density")
plot(lbd$mean_distance, lbd$p, type='b',
     pch=20, lwd=0.8, log="xy", xlab="distance",
     ylab="probability")
```



extracting subnetworks

Through `subset_chromnet` one can create subnetworks for example by chromosome

```
net1 <- chaser::subset_chromnet(net, chrom="chr1")
print(net1)
```

```
## nodes: 3120
## edges: 6724
## 78  feature(s)
```

or one can select a special set of nodes and only consider edges joining nodes which belong to the set as in the following snippet:

```
baits <- chaser::export(net, "baits")

# Once we have the baits we can consider a network
# where the edges always connect a bait to another bait.
# This is called the promoter-promoter (PP) network
# in the case of Promoter Capture HiC (PCHiC)::

netbb <- chaser::subset_chromnet(net, method="nodes", nodes1=baits)
print(netbb)
```

```
## nodes: 13099
## edges: 37590
## 78  feature(s)
```

Similarly one can construct a network where edges exist only between a bait and an *other end* , which corresponds to the PO network in the case of PCHiC.

```
# extract other ends
tmp <- chaser::export(net, "nodes")$name
oes <- tmp[!(tmp %in% baits)]
netbo <- chaser::subset_chromnet(net, method="nodes", nodes1=baits, nodes2=oes)
print(netbo)
```

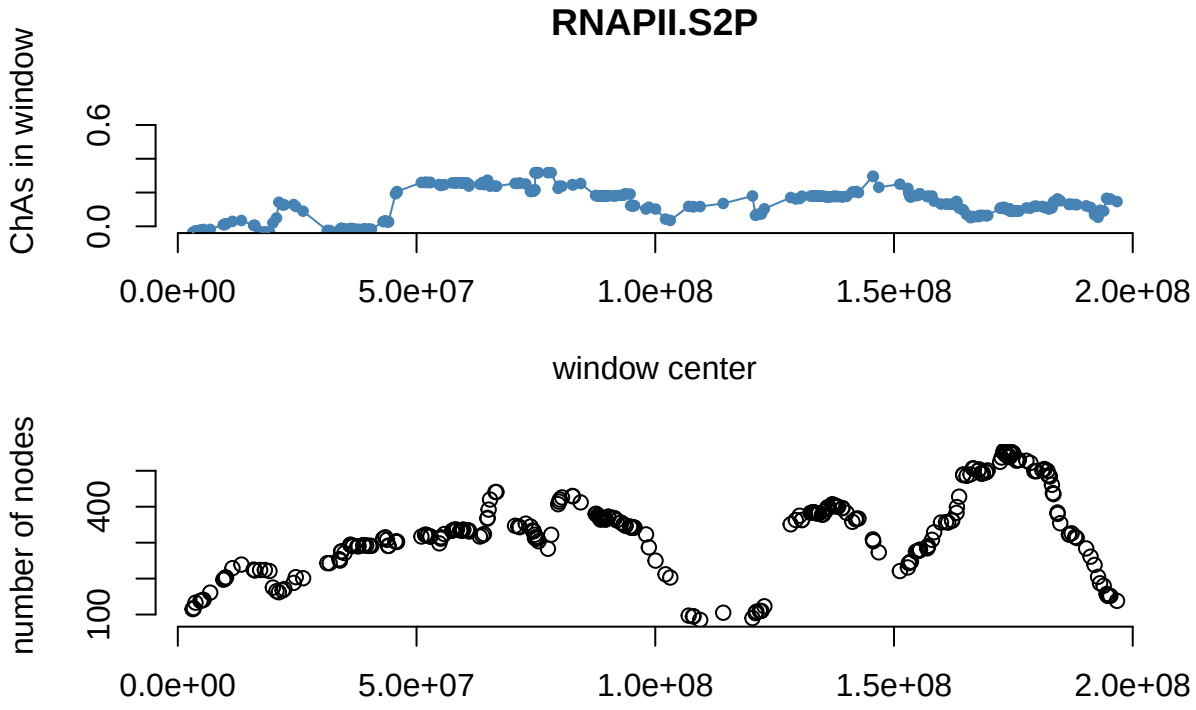
```
## nodes: 54123
## edges: 102384
## 78  feature(s)
```

It is also possible to select a node and consider the subnetwork formed by all the nodes close (along the sequence) to it, uaing the method `dist1d`.

Here we compute the assortativity on chr1 on windows of 10Mb along the genome. The plot on the bottom is the number of nodes present in each window.

```
net1n <- chaser::export(net1, "nodes")
nidx <- as.integer(seq.int(1, 3120, length.out=312)) # pick one every 10
# network for each genomic window
netw <- lapply(nidx, function(e){
    chaser::subset_chromnet(net1, method="dist1d",
                            center=net1n[e, "name"], radius=1e7)})
nnodes <-sapply(netw, function(e){nrow(chaser::export(e, "nodes"))})
chasw <- lapply(netw, chaser::chas)
chaswrna <- sapply(chasw, function(e){e[["RNAPII.S2P"]]})
net1npos <- as.integer((net1n$start + net1n$end)/2)[nidx]
chaswrna <- chaswrna[order(net1npos)]
nnodes <- nnodes[order(net1npos)]
```

```
net1npos <-net1npos[order(net1npos)]
op <- par(mfrow=c(2,1),
          mai=c(0.8, 0.8, 0.3, 0.5),
          omi=c(0,0,0.4,0))
plot(net1npos, chaswrna, type='p', col="steelblue",
     pch=20, axes=FALSE, ylim=c(0,1), xlab="window center", ylab="ChAs in window",
     main="RNAPII.S2P")
axis(1)
axis(2, at=seq(0,0.6, by=0.2))
lines(net1npos, chaswrna, col="steelblue")
plot( net1npos, nnodes, xlab="", ylab="number of nodes",
     axes=FALSE)
axis(1)
axis(2)
```




```
par(op)
```

another option of `subset_chromnet` (`dist3d`) allows one to select a subnetwork based on three dimensional distance (measured in number of edge hops from one node to another on the contact network).

computing correlations

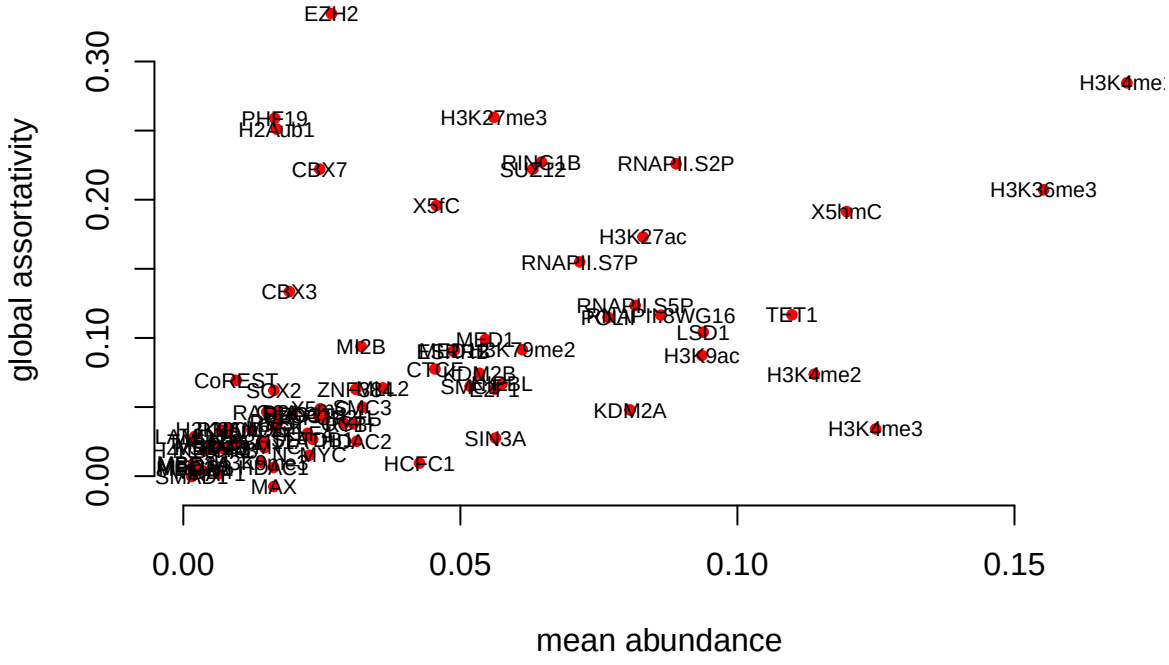
The `chaser::chas` function computes different forms of network correlations, depending on the method specified by the user:

- `chas` (default): a vector of assortativities, one for each feature mapped onto the network

- `crosschas` a matrix $k \times k$ containing cross-assortativities between all possible pairs of features. This includes the assortativities of each feature with itself, which lie on the diagonal of the matrix.
- `corrfun` a matrix $n \times k$ which associates to each node the mean value of each feature computed on all its neighbours.
- `categorical` when the features are unordered categories (eg gene functions).

One useful plot is assortativity versus abundance:

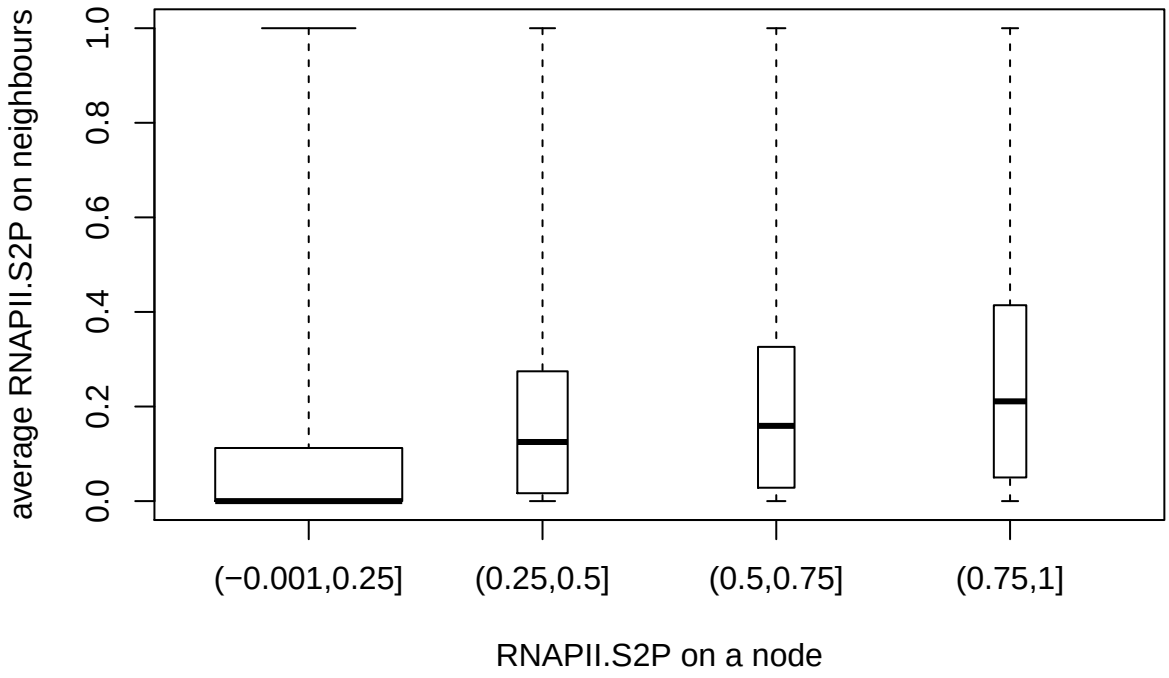
```
# compute chas
net_chas <- chaser::chas(net)
# extract features
featm <- chaser::export(net)
# plot chas vs feature abundance
plot(colMeans(featm), net_chas, xlab="mean abundance",
     ylab="global assortativity", axes=FALSE, pch=20, col="red")
text(colMeans(featm), net_chas, names(net_chas), cex=0.7)
axis(1)
axis(2)
```



Here we give an example of how to use the `corr_fun` option of `chas` to look at correlations in a more granular way than by giving a single number for the whole network: we plot for each node the abundance at that node and the average abundance of the neighbours.

```
## create a factor by cutting the abundance
cf          <- cut(featm[,"RNAPII.S2P"], breaks=4)
net_corrfun <- chaser::chas(net, "corr_fun")
boxplot(net_corrfun[, "RNAPII.S2P" ] ~ cf, range=0,
```

```
        varwidth=TRUE, xlab= "RNAPII.S2P on a node",
        ylab= "average RNAPII.S2P on neighbours" )
```



A note on assortativity

In what follows, let's assume that the matrix F containing the features has n rows (corresponding to n nodes) and k columns. Let A be a symmetric $n \times n$ matrix, the adjacency matrix of the network. We can safely assume A (although not F) to be sparse.

As said before, chromatin assortativity is correlation computed across the edges; i.e. the random variables being paired are the nodes at the end of the same edge. Let $x_i(x_j)$ be the value of a feature at node i (j). The Pearson correlation is given by

$\frac{E[(x_i-\mu)(x_j-\mu)]}{\sigma_{x_i}\sigma_{x_j}}$

where μ is the mean of the feature across all nodes, weighted by the number of edges they take part in

$$\mu = \frac{1}{2m}\sum_{i=1}^{n} d_i x_i$$

where d_i is the degree of node i and m is the number of edges. Similarly

$$\sigma_x = \frac{1}{2m}\sqrt{\sum_{i=1}^{n} d_i (x_i - \mu)^2}$$

If we normalize each column of F by subtracting its mean and dividing by its σ we obtain a matrix $\tilde{F}$ such that the assortativity (as a matrix $k \times k$) is given by:

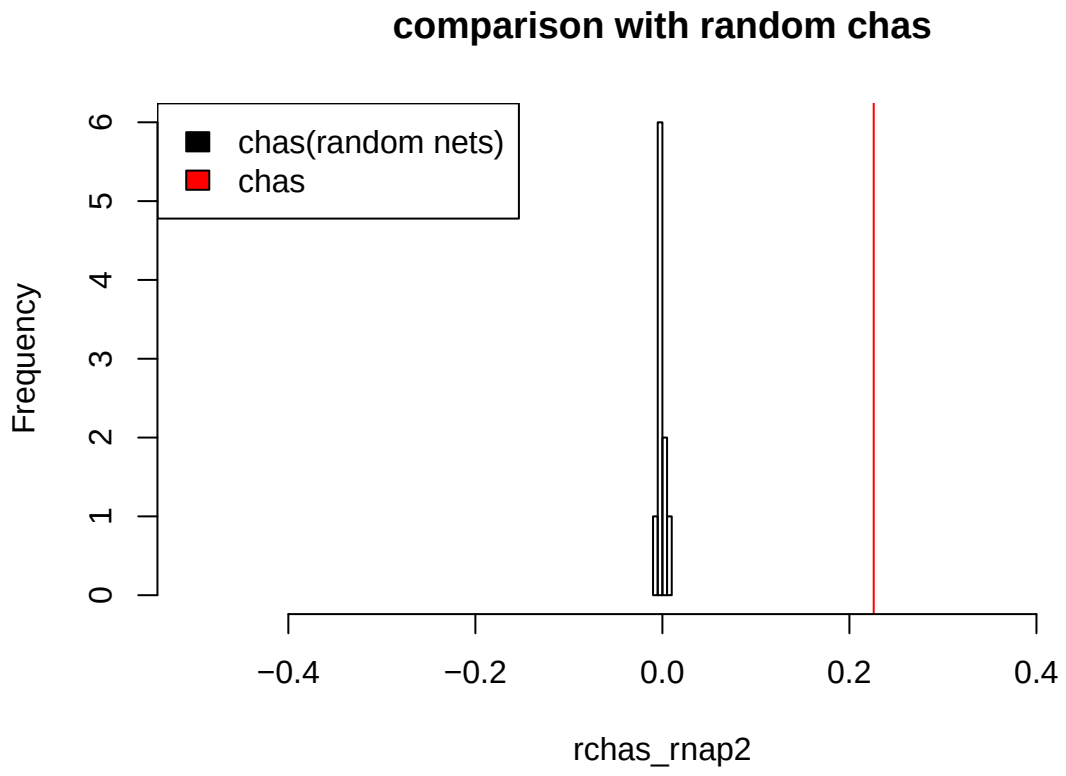
$\frac{1}{2m}\tilde{F}^T A\tilde{F}$

randomization

simple redistribution of features

One can randomize the network by redistributing the features in the nodes at random before computing the correlation as in the following example

```
## rnets is a list of 10 randomized chromnet objects.
rnets <- chaser::randomize(net, nrandom=10)
rchas <- lapply(rnets, chaser::chas)
rchas_rnap2 <- unlist(Map(function(e){e[["RNAPII.S2P"]]}, rchas))
hist(rchas_rnap2, axes=FALSE, xlim=c(-0.5, 0.5),
     main="comparison with random chas")
abline(v=net_chas["RNAPII.S2P"], col="red")
axis(1)
axis(2)
legend("topleft", legend=c("chas(random nets)", "chas"),
              fill=c("black", "red"))
```



randomization with a set of invariant nodes

The randomizaton can be more subtle, for example one can divide nodes into two non overlapping groups and features will be redistributed only inside each group.

In the following examples we use the group baits (and implicitly the group *other ends*) to this purpose.

```
rnets <- chaser::randomize(net, nrandom=10, preserve.nodes=baits)
rchas_baits <- lapply(rnets, chaser::chas)
```

randomization which preserves distances

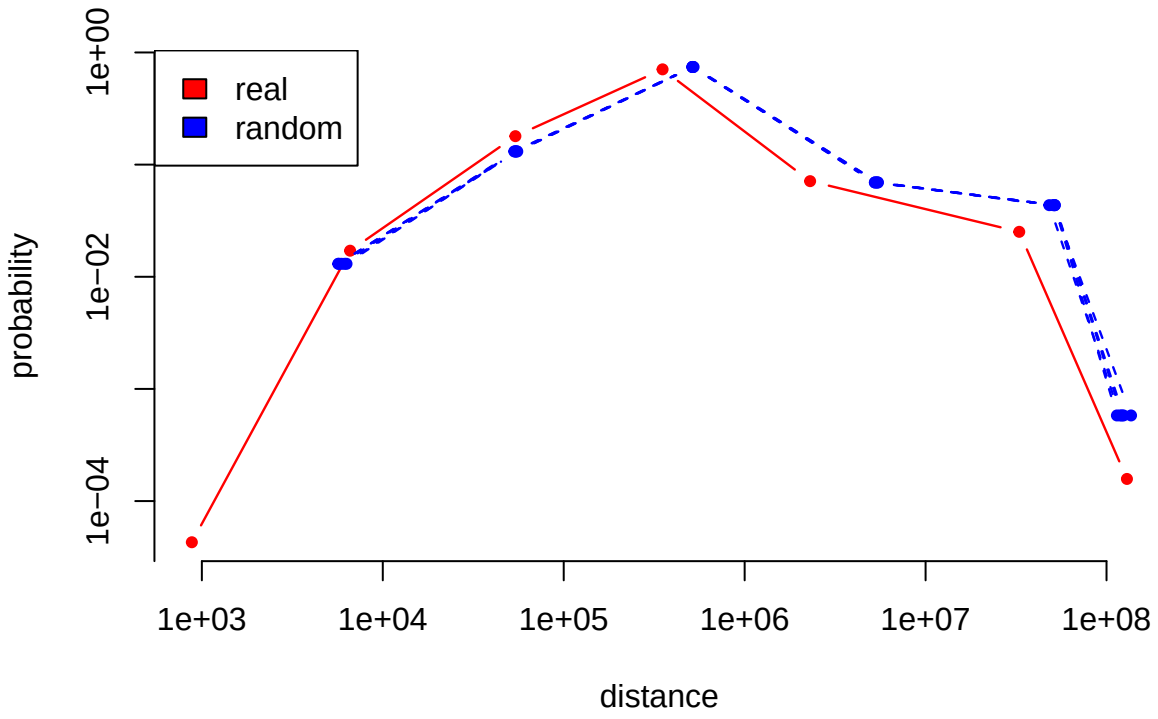
Finally the randomization can preserve (approximately) the distribution of the genomic distances spanned by the connected nodes, when the user selects the option `dist.match=TRUE`. Note that the algorithm randomizes chromosomes one at the time; hence the randomized network will not contain interchromosomal contacts.

```
netr <- chaser::randomize(net1, nrandom=5, dist.match=TRUE)
rchas_dist <- lapply(netr, chaser::chas)
```

The code above creates a list `netr` of length 2 (in order not to slow down the compilation of this vignette); each element of the list `rchas_dist` is an array of 78 assortativities computed on randomized networks.

We can compare the distribution of distances in the true network with the distribution of distances in the randomized network:

```
lbd <- chaser::export(net, "log-binned-distance-density")
plot(lbd$mean_distance, lbd$p, type='b',
     pch=20, lwd=1.2, log="xy", xlab="distance",
     ylab="probability", col="red", axes=FALSE)
axis(1, 10^seq(3,8))
axis(2, at=10^seq(-5,0,by=1))
legend("topleft", fill=c("red","blue"), legend=c("real", "random"))
for (i in seq(1, 5)){
    lbdr <- chaser::export(netr[[i]], "log-binned-distance-density")
    points(lbdr$mean_distance, lbdr$p, type='b',
     pch=20, lwd=1.2, col="blue", lty="dashed")
}
```



The algorithm for distance preserving random networks works chromosome by chromosome (hence it will not generate interchromosomal contacts). First it counts the number of edges in each logarithmic decade in the real network. Then, for each decade, it generates a matching number of random edges with distances in that decade.

Discussion

In the diagram below we illustrate a typical `ChAseR` workflow. A file containing a list of edges is made into a chromnet object by `make_chromnet`. After that, the user can load a file containing genomic features (in the figure we hint at H3K27ac peaks) and use `ChAseR` to assign those features to the relevant nodes in the network. This is done automatically by intersecting the coordinates of the nodes with the coordinates of the features. One can then compute various measures of correlation with `chas` and check if they are significant with `randomize`.

References

Lundberg, Scott M, William B Tu, Brian Raught, Linda Z Penn, Michael M Hoffman, and Su-In Lee. 2016. "ChromNet: Learning the Human Chromatin Network from All Encode Chip-Seq Data." *Genome Biology* 17 (1): 82.

Pancaldi, Vera, Enrique Carrillo-de-Santa-Pau, Biola Maria Javierre, David Juan, Peter Fraser, Mikhail Spivakov, Alfonso Valencia, and Daniel Rico. 2016. "Integrating Epigenomic Data and 3D Genomic Structure with a New Measure of Chromatin Assortativity." *Genome Biology* 17 (1): 152.

Schoenfelder, Stefan, Biola-Maria Javierre, Mayra Furlan-Magaril, Steven W Wingett, and Peter Fraser. 2018. "Promoter Capture Hi-c: High-Resolution, Genome-Wide Profiling of Promoter Interactions." *JoVE (Journal of Visualized Experiments)*, no. 136: e57320.

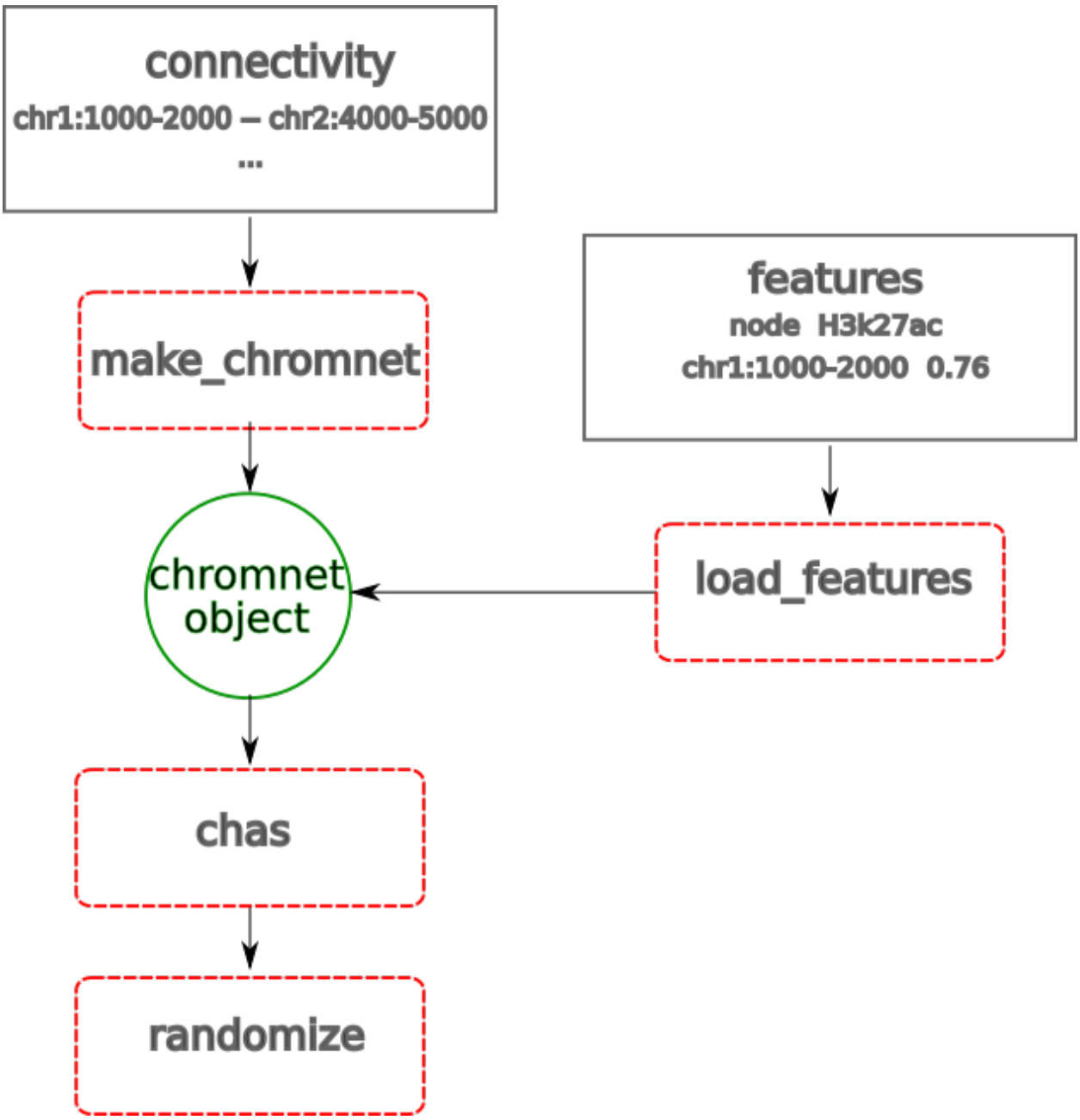


Figure 1: A ChAseR workflow