Bioinformatics 2024 July

Reverse network diffusion to remove indirect noise for better inference of gene regulatory networks

Jiating Yu, Jiacheng Leng, Fan Yuan, Duanchen Sun, Ling-Yun Wu

Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China

Presented by: Yijingxiu Lu

Reverse network diffusion to remove indirect noise for better inference of gene regulatory networks

Jiating Yu, Jiacheng Leng, Fan Yuan, Duanchen Sun, Ling-Yun Wu

Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China

$\neq$ Generative Diffusion

One-Sentence Summary:

- Denoise gene regulatory network with the **reversed** **diffusion process defined by random walk on graph**.
  - **diffusion process defined by random walk on graph:** proposed as ***Network Refinement*** (Yu et al. 2023)

$$h\left(f_m\left(g(W)\right)\right)$$

  - **reversed** **diffusion process defined by random walk on graph:**

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

Task: Denoise gene regulatory network

- **Input:** noisy observed network G_{obs}
- **Output:** direct network G_{dir}

Method: REverse Network Diffusion On Random walks (RENDOR) (**Network Diffusion $\neq$ Generative Diffusion**)

Benchmark: Dialogue on Reverse Engineering Assessment and Methods (DREAM)

- **DREAM** provides high-confidence networks for *E. coli* and *S. aureus*, each comprising ~1,700 transcriptional interactions at a precision of ~50%.
  - ***E.coli*:** experimentally validated interactions from a curated database (RegulonDB[16])
  - **ChIP-chip:** a high-confidence set of interactions supported by genome-wide transcription-factor binding data
  - ***S. cerevisiae**:* evolutionarily conserved binding motifs
  - ***in silico* data**

One-Sentence Summary:

- Denoise gene regulatory network with the **reversed diffusion process defined by random walk on graph**.
  - **diffusion process defined by random walk on graph:** proposed as ***Network Refinement*** (Yu et al. 2023)

$$h(f_m(g(W)))$$

  - **reversed diffusion process defined by random walk on graph:**

$$h(f_m^{-1}(g(W)))$$

Network Diffusion

- Describe the movement process of entities or states in the network

Generative Diffusion

- Uses diffusion and denoising processes to generate high-quality data

Task: Denoise gene regulatory network

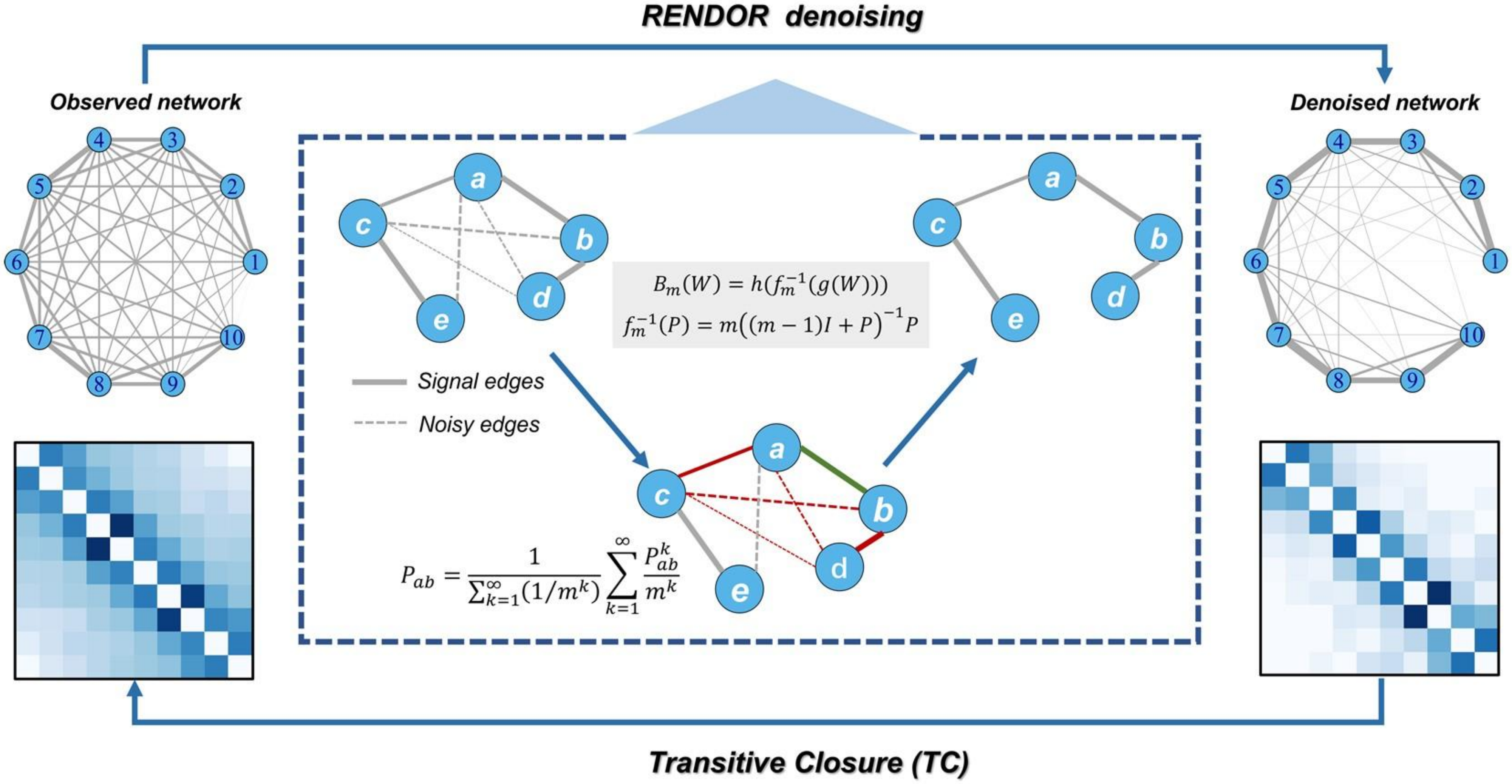
- **Input:** noisy observed network G_{obs}
- **Output:** direct network G_{dir}

Method: REverse Network Diffusion On Random walks (RENDOR) (**Network Diffusion $\neq$ Generative Diffusion**)

Benchmark: Dialogue on Reverse Engineering Assessment and Methods (DREAM)

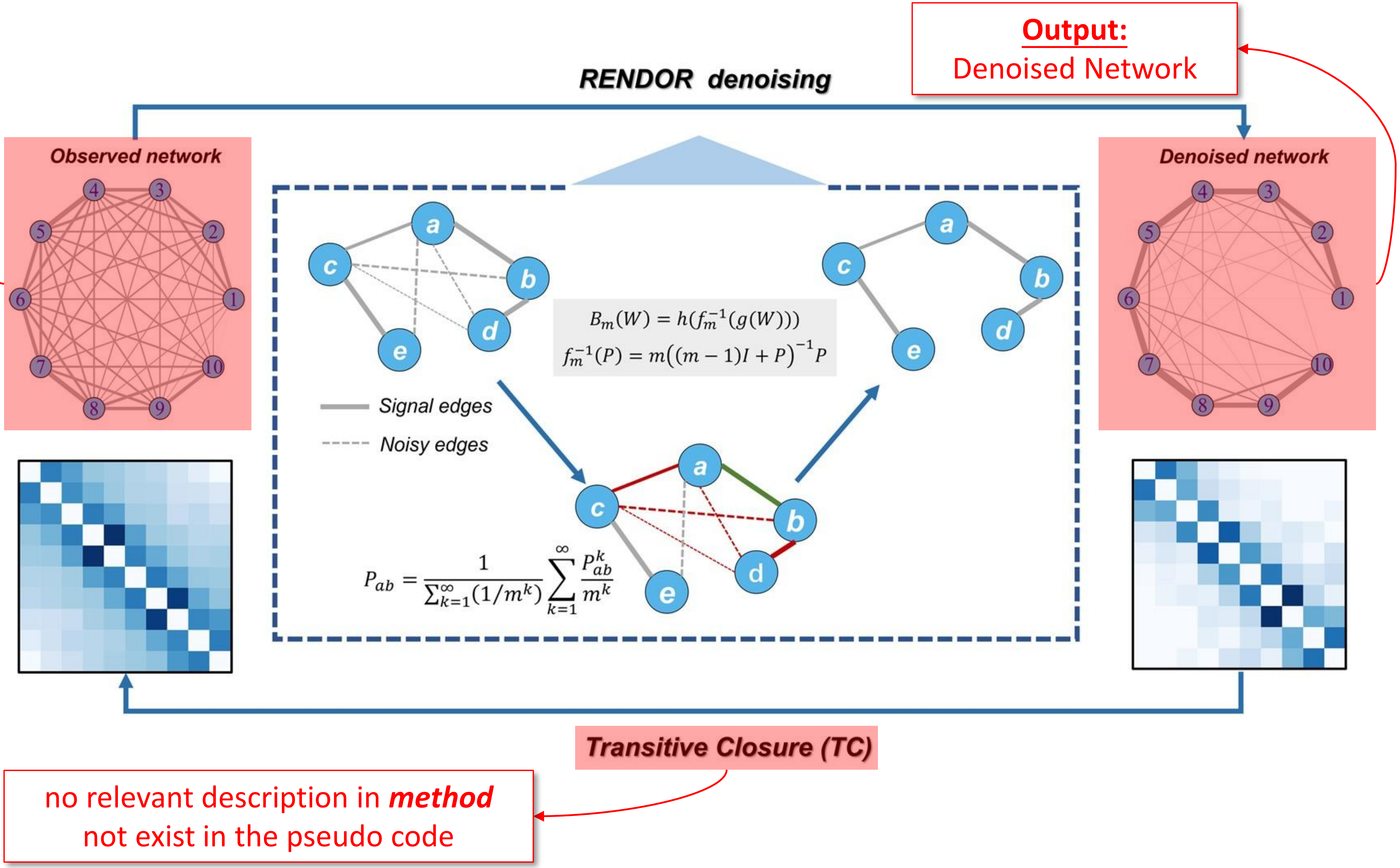
- **DREAM** provides high-confidence networks for *E. coli* and *S. aureus*, each comprising ~1,700 transcriptional interactions at a precision of ~50%.
  - ***E.coli*:** experimentally validated interactions from a curated database (RegulonDB[16])
  - **ChIP-chip:** a high-confidence set of interactions supported by genome-wide transcription-factor binding data
  - ***S. cerevisiae*:** evolutionarily conserved binding motifs
  - ***in silico* data**

Framework


RENDOR denoising
Observed network
Denoised network
Signal edges
Noisy edges
$B_m(W) = h(f_m^{-1}(g(W)))$
$f_m^{-1}(P) = m\big((m-1)I + P\big)^{-1}P$
$P_{ab} = \frac{1}{\sum_{k=1}^{\infty}(1/m^k)} \sum_{k=1}^{\infty} \frac{P_{ab}^k}{m^k}$
Transitive Closure (TC)

Framework

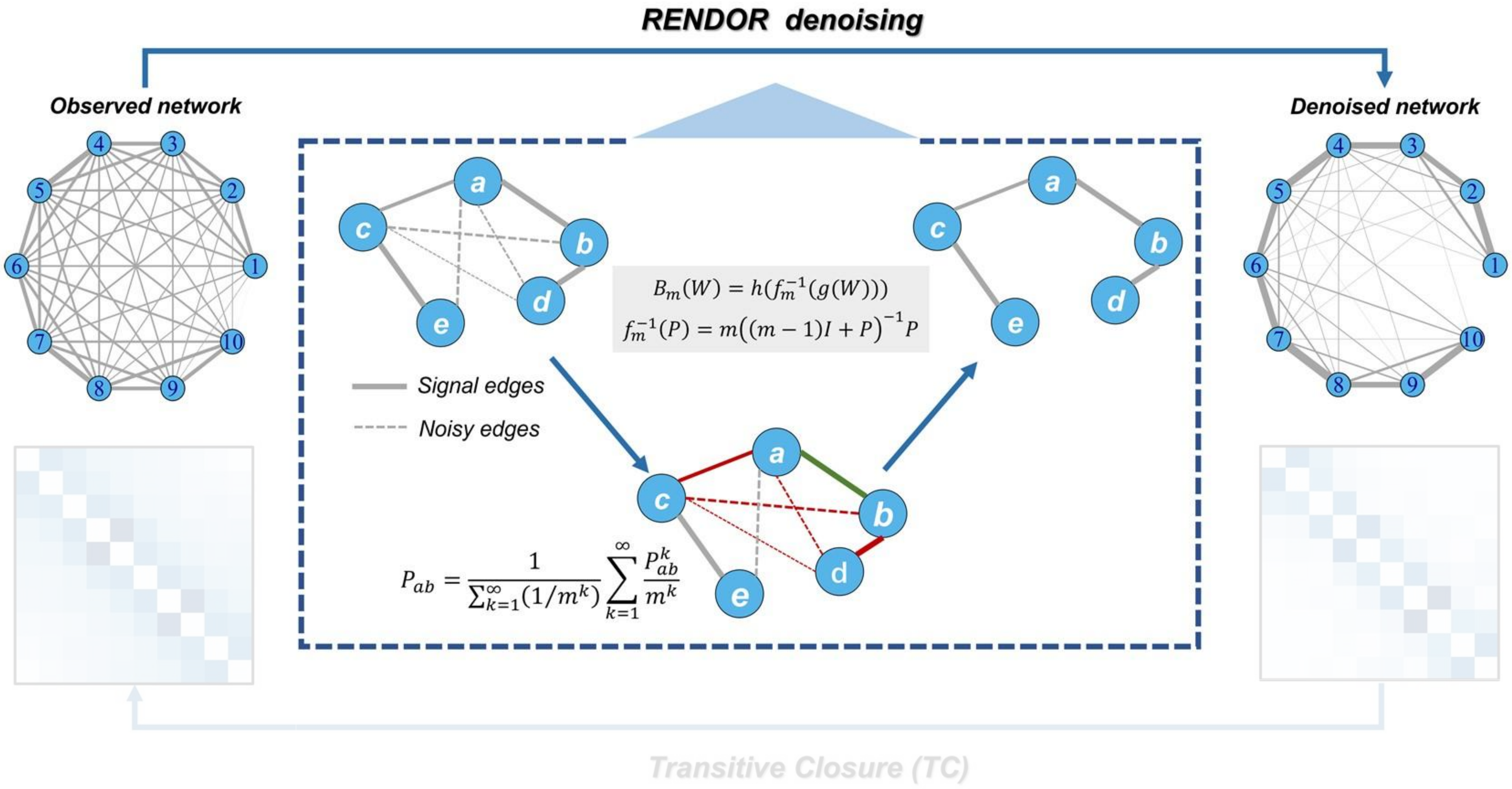
Input:
Noisy Network



Framework

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

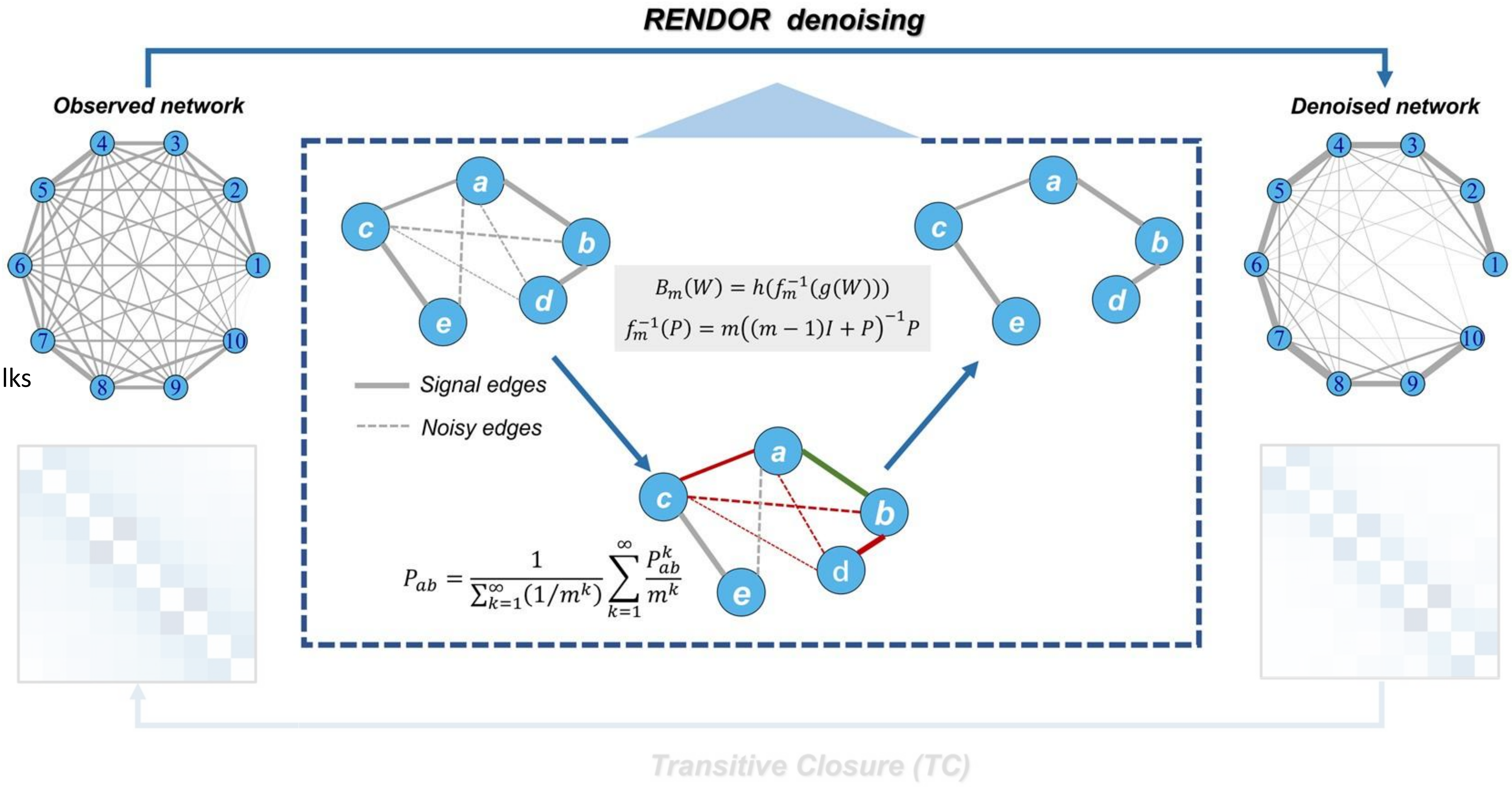


Framework

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

- ***g*:** define random walks (RW) on graph

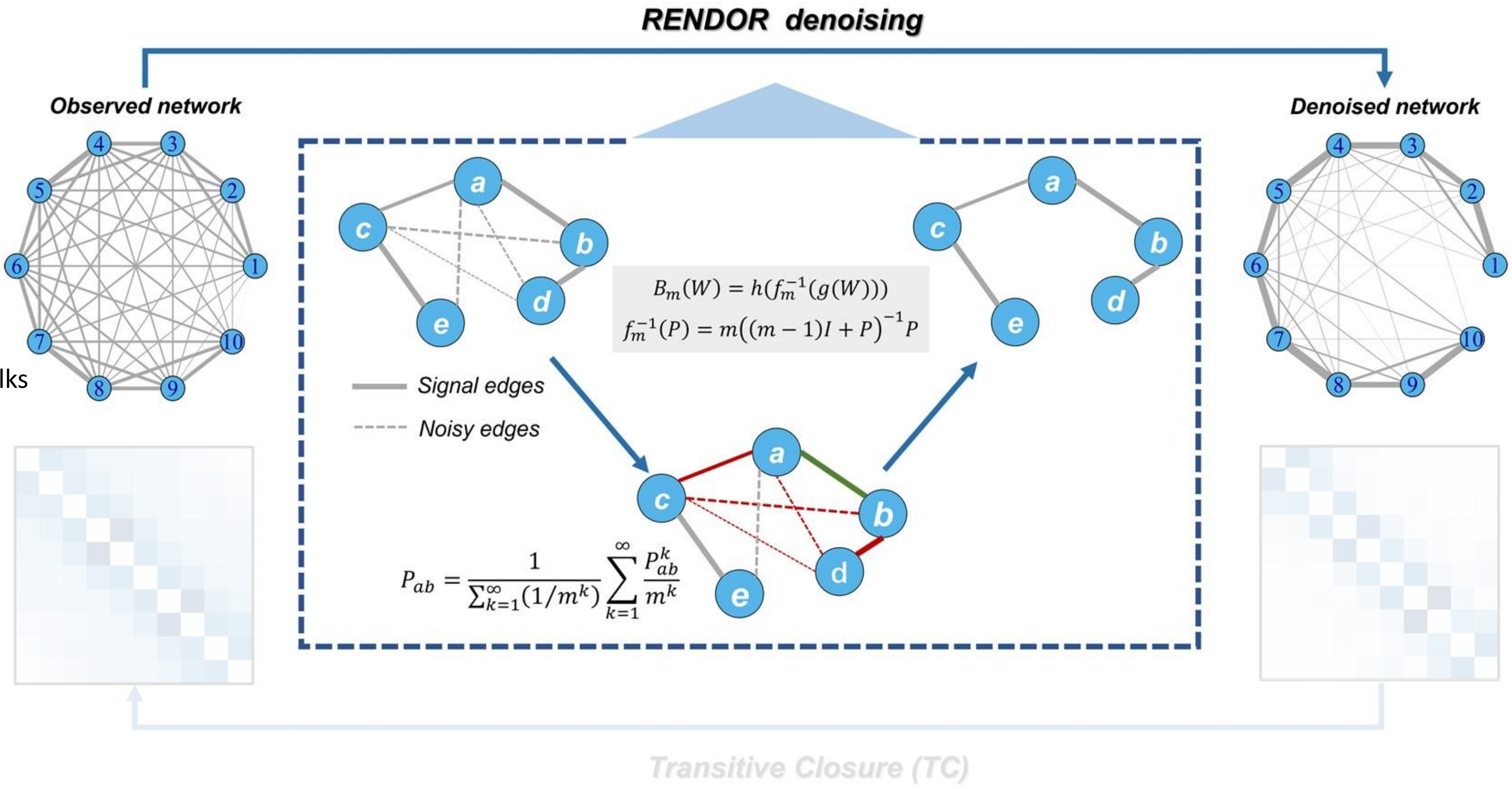


Framework

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

- ***g*:** define random walks (RW) on graph
- ***h*:** map from RW to denoised graph

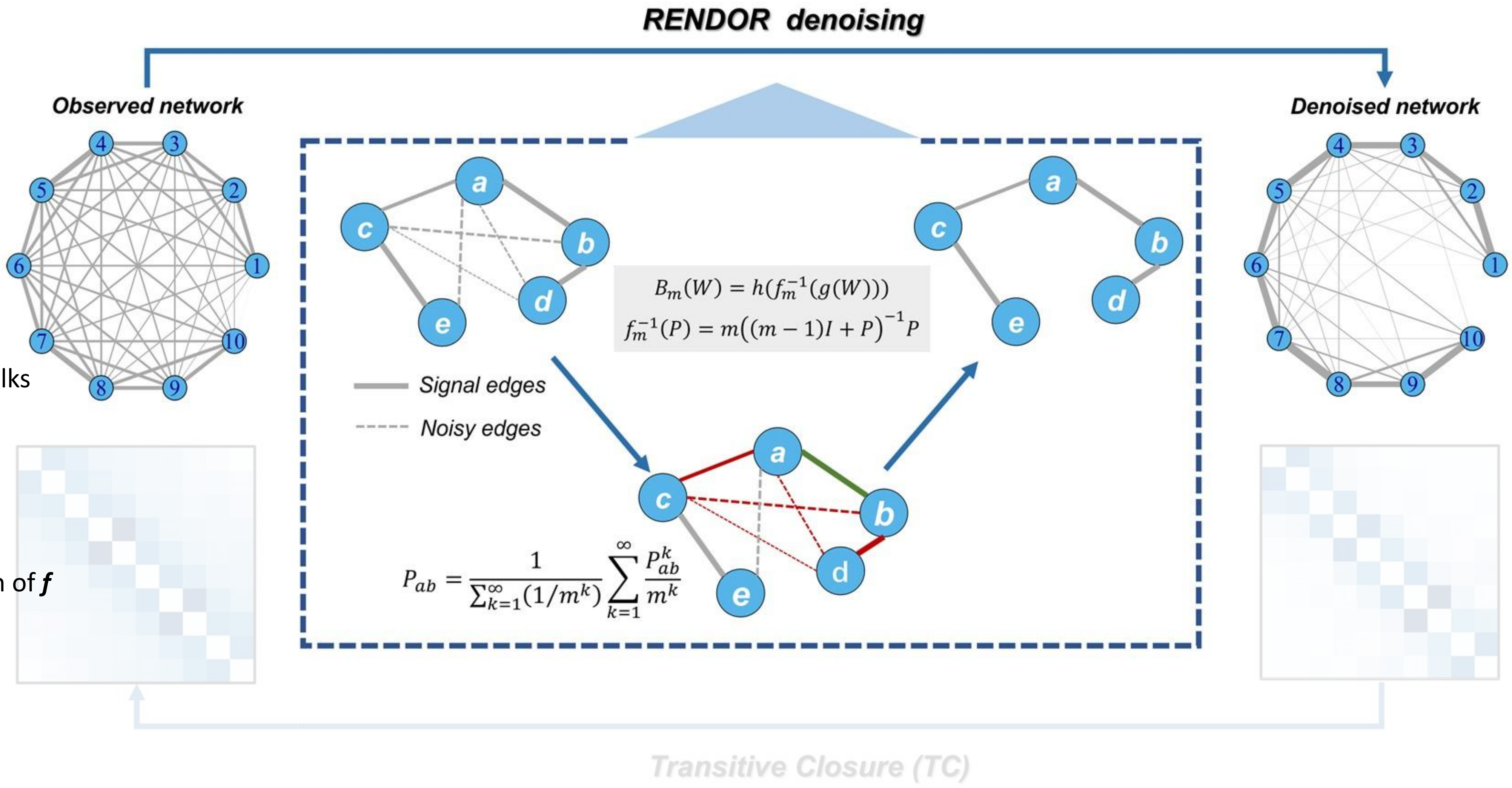


Framework

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

- ***g*:** define random walks (RW) on graph
- ***h*:** map from RW to denoised graph
- ***f^{-1}*:** inverse function of ***f***

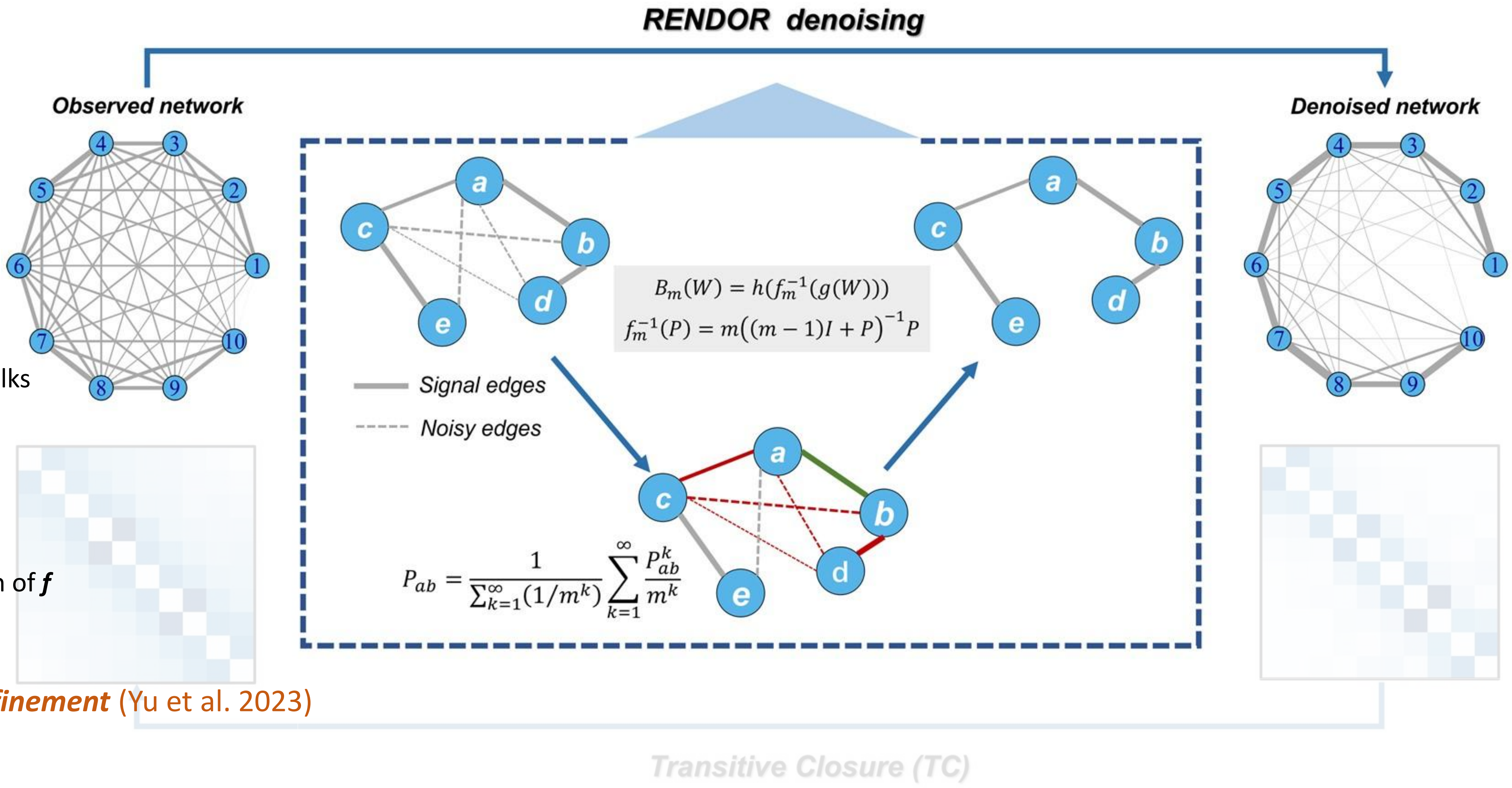


Framework

RENDOR = $h\left(f_m^{-1}\left(g(W)\right)\right)$

- ***g*:** define random walks (RW) on graph
- ***h*:** map from RW to denoised graph
- ***f*⁻¹:** inverse function of ***f***
- ***f*:**

From ***Network Refinement*** (Yu et al. 2023)

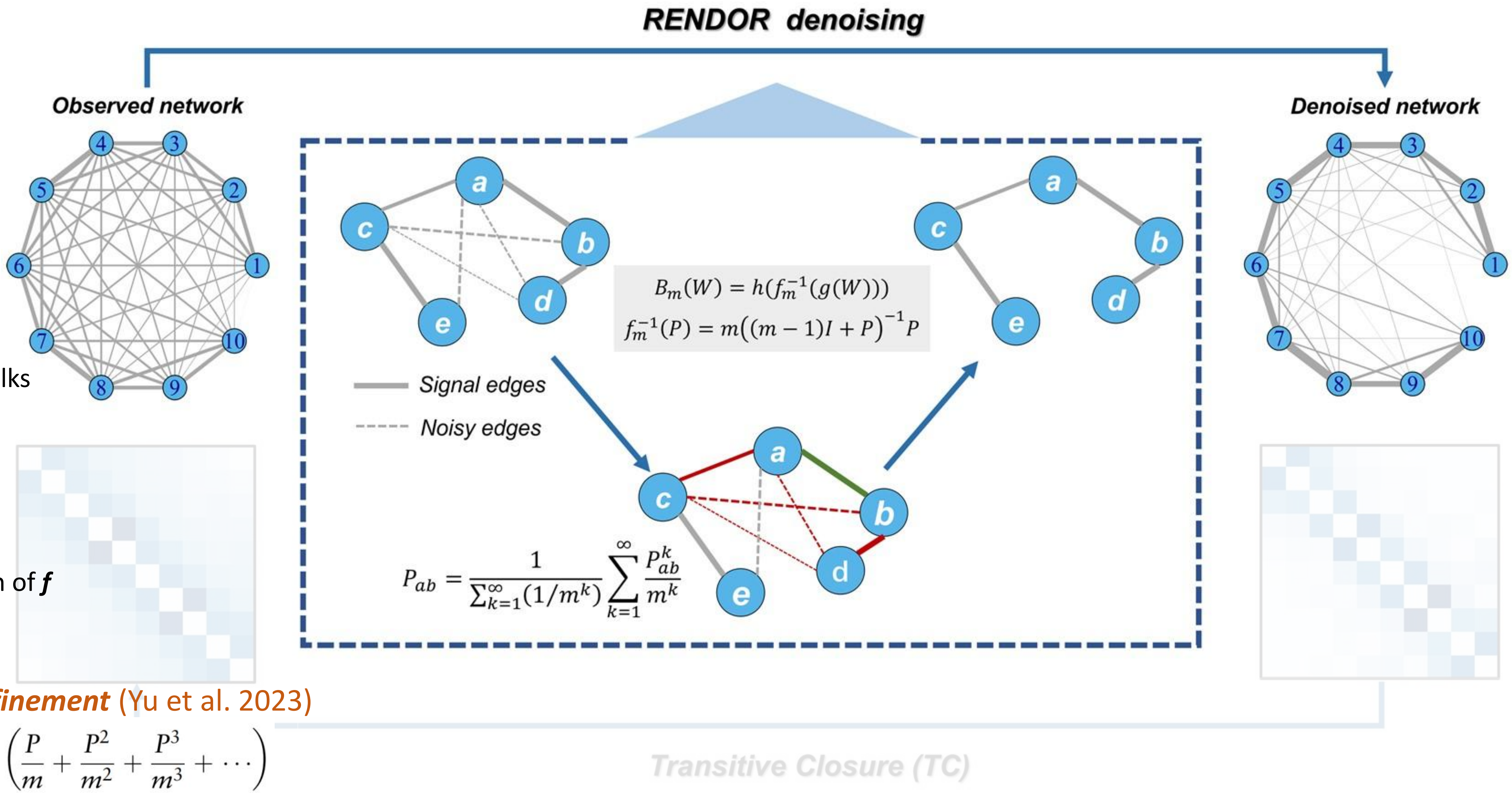


Framework

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

- ***g*:** define random walks (RW) on graph
- ***h*:** map from RW to denoised graph
- ***f*$^{-1}$:** inverse function of ***f***
- ***f*:**



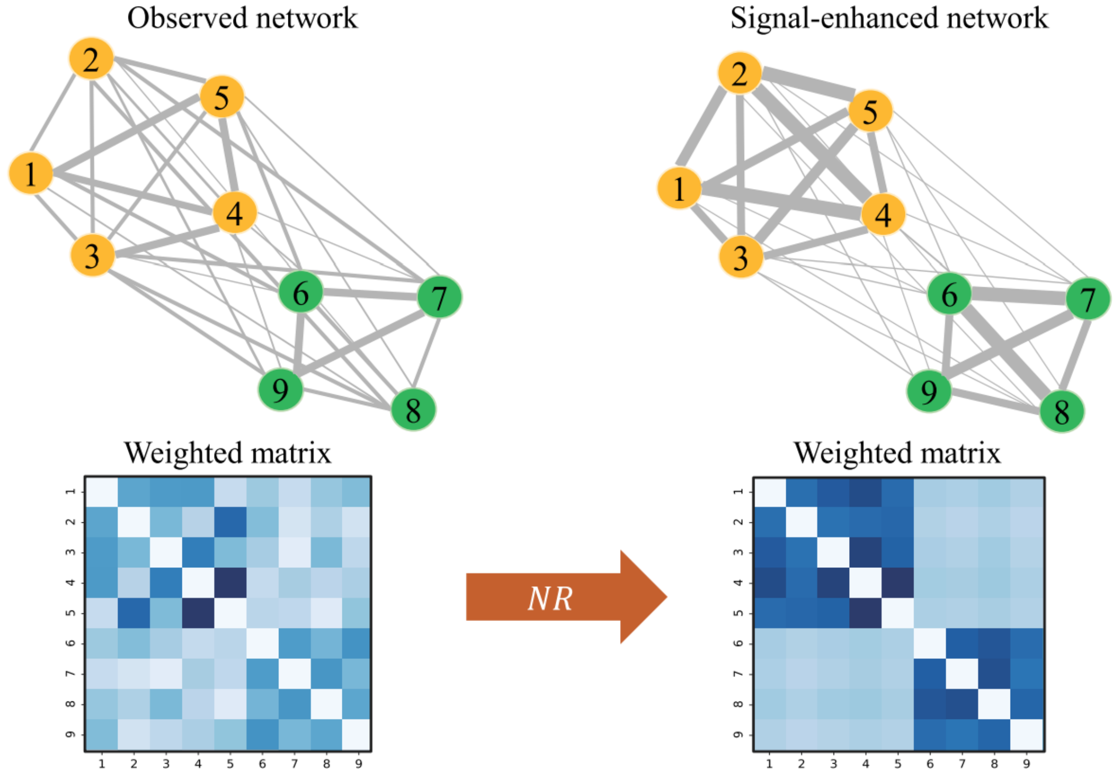
From ***Network Refinement*** (Yu et al. 2023)

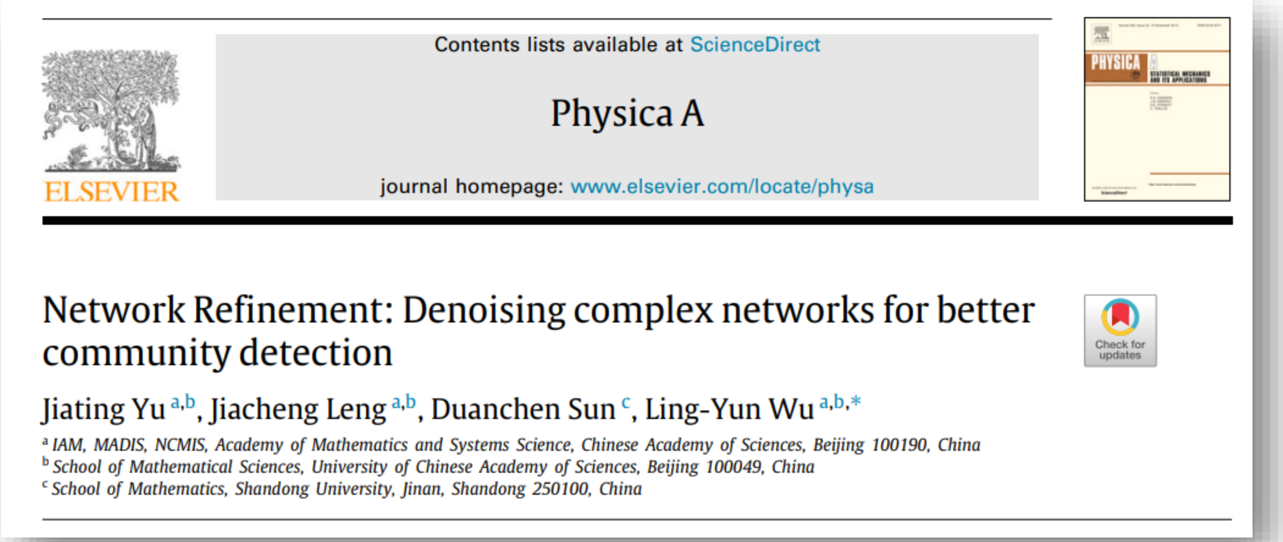
$$\begin{aligned} f_m(P) &= \frac{1}{\sum_{k=1}^{\infty}(1/m^k)}\left(\frac{P}{m} + \frac{P^2}{m^2} + \frac{P^3}{m^3} + \cdots\right) \\ &= \frac{1}{\sum_{k=1}^{\infty}(1/m^k)} \sum_{k=1}^{\infty} \frac{P^k}{m^k} \\ &= (m-1)P(mI-P)^{-1} \end{aligned}$$

$$f_m^{-1}(P) = m((m-1)I+P)^{-1}P$$

Network Refinement (NR) Yu et al. 2023

Goal: Enhance signals in network



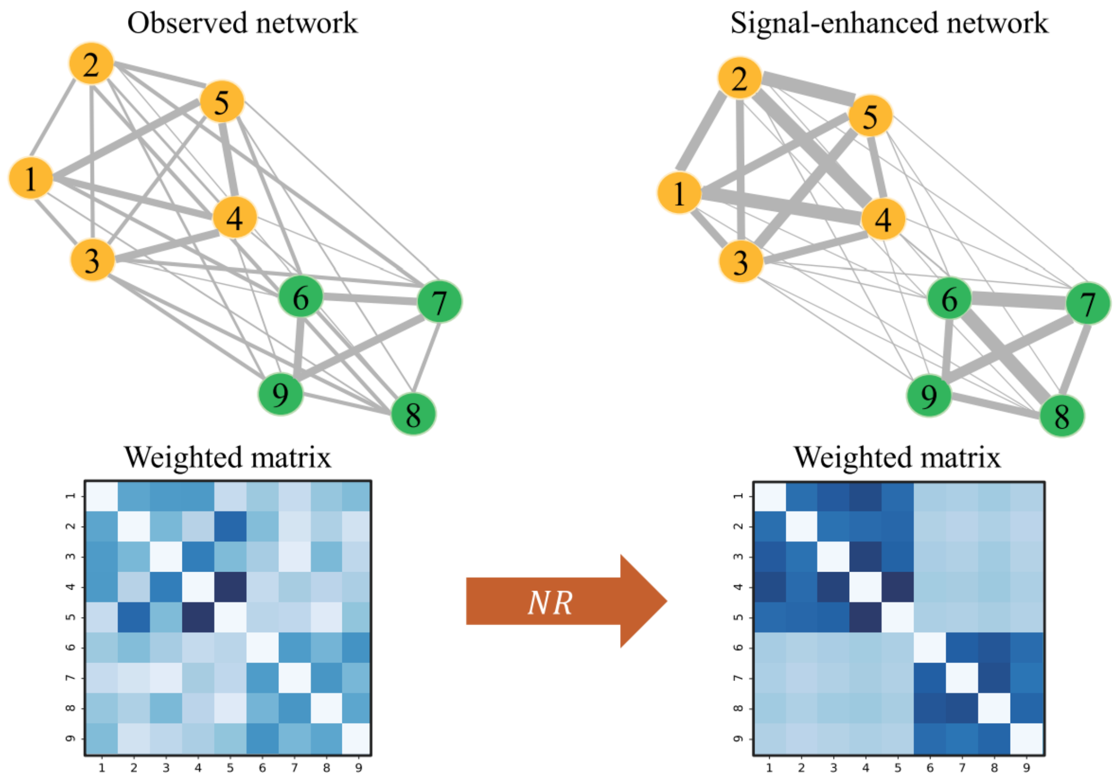

Contents lists available at ScienceDirect
Physica A
journal homepage: www.elsevier.com/locate/physa

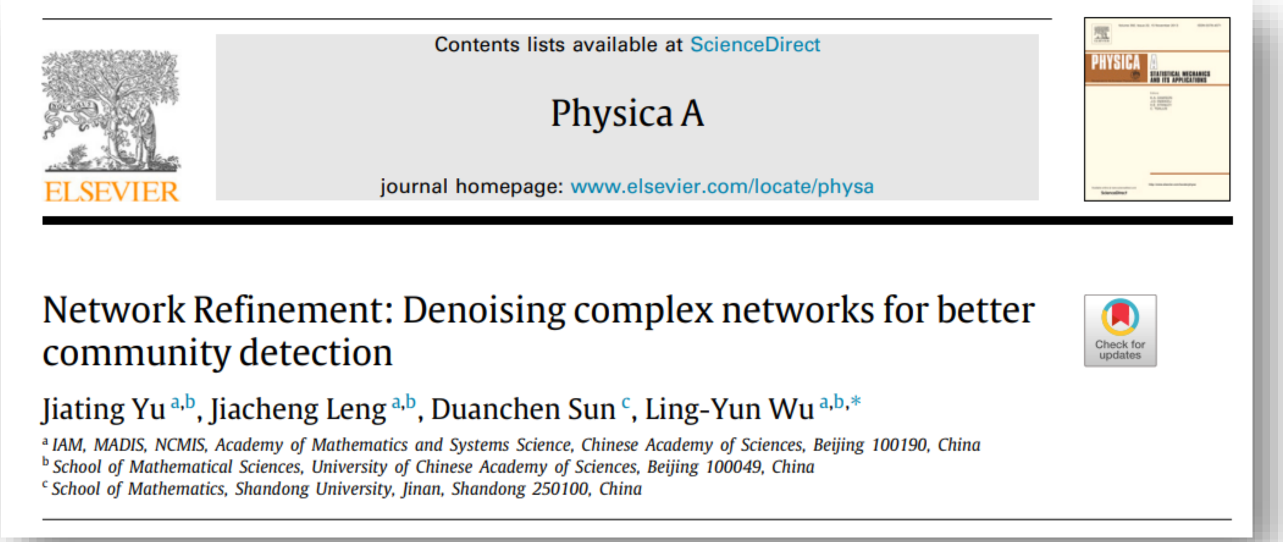
Network Refinement: Denoising complex networks for better community detection

Jiating Yu [a,b], Jiacheng Leng [a,b], Duanchen Sun [c], Ling-Yun Wu [a,b,*]
[a] IAM, MADIS, NCMIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
[b] School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
[c] School of Mathematics, Shandong University, Jinan, Shandong 250100, China

Network Refinement (NR) Yu et al. 2023

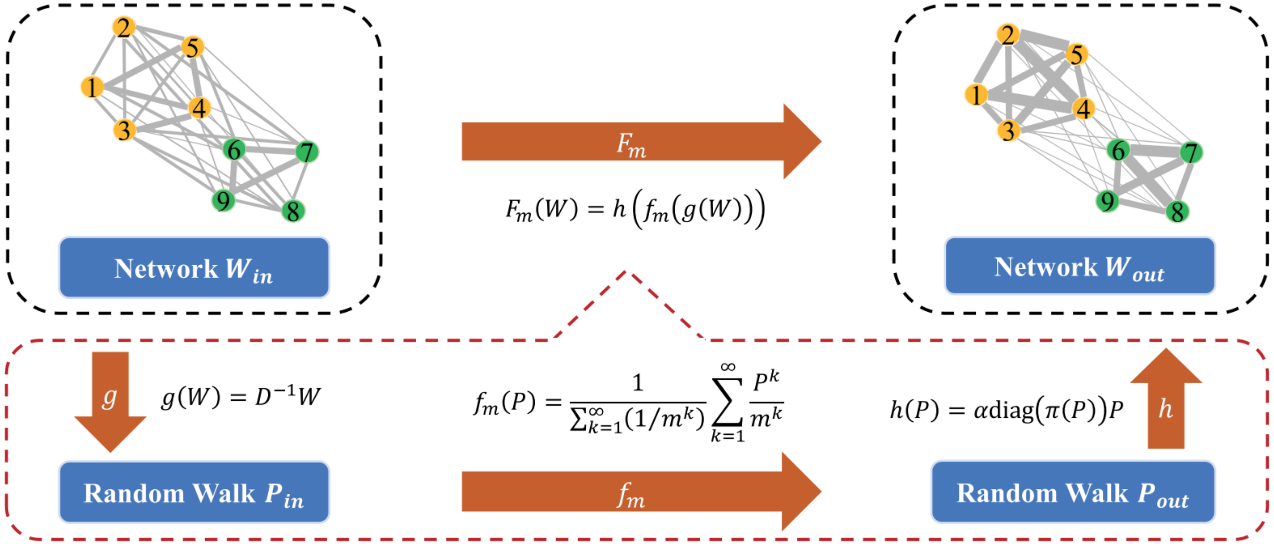
Goal: Enhance signals in network
Method: global network diffusion process defined by random walk on graph




Contents lists available at ScienceDirect
Physica A
journal homepage: www.elsevier.com/locate/physa

Network Refinement: Denoising complex networks for better community detection

Jiating Yu [a,b], Jiacheng Leng [a,b], Duanchen Sun [c], Ling-Yun Wu [a,b,*]
[a] IAM, MADIS, NCMIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
[b] School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
[c] School of Mathematics, Shandong University, Jinan, Shandong 250100, China



Methodology (Pseudo code)

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

Pseudocode for RENDOR

Input: $\boldsymbol{W}_{\text{obs}}$: weighted adjacency matrix of observed network;

$\boldsymbol{m}$: diffusion intensity parameter;

$\varepsilon_1, \varepsilon_2$: preprocessing parameters.

Output: $\boldsymbol{W}_{\text{dir}}$: denoised adjacency matrix of direct network.

1. $\tilde{\boldsymbol{W}}_{\text{obs}} = \boldsymbol{W}_{\text{obs}} + \varepsilon_1 \boldsymbol{J} + \varepsilon_2 \boldsymbol{I}$
2. $\boldsymbol{P}_{\text{obs}} = \boldsymbol{g}\left(\tilde{\boldsymbol{W}}_{\text{obs}}\right) = \left(\operatorname{diag}\left\{\tilde{\boldsymbol{W}}_{\text{obs}} \mathbf{1}\right\}\right)^{-1} \tilde{\boldsymbol{W}}_{\text{obs}}$
3. $\boldsymbol{P}_{\text{dir}} = \boldsymbol{f}_{\boldsymbol{m}}^{-1}(\boldsymbol{P}_{\text{obs}}) = \boldsymbol{m}\left((\boldsymbol{m}-1)\boldsymbol{I} + \boldsymbol{P}_{\text{obs}}\right)^{-1} \boldsymbol{P}_{\text{obs}}$
4. for $\boldsymbol{i} = 1, \ldots, \boldsymbol{n}$:
   if $\min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\} \geq 0$: $\boldsymbol{\beta}_i = 0$
   else: $\boldsymbol{\beta}_i = \min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\}$
5. $\tilde{\boldsymbol{P}}_{\text{dir}} = \boldsymbol{P}_{\text{dir}} - (\boldsymbol{\beta}_1 \mathbf{1}, \ldots, \boldsymbol{\beta}_n \mathbf{1})^T$
6. $\boldsymbol{W}_{\text{dir}} = \boldsymbol{h}\left(\tilde{\boldsymbol{P}}_{\text{dir}}\right) = \operatorname{diag}\left\{\boldsymbol{\pi}\left(\tilde{\boldsymbol{P}}_{\text{dir}}\right)\right\} \tilde{\boldsymbol{P}}_{\text{dir}}$

Methodology (Pseudo code)

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

Pseudocode for RENDOR

Input: W_{obs}: weighted adjacency matrix of observed network;

m: diffusion intensity parameter;

$\varepsilon_1, \varepsilon_2$: preprocessing parameters.

Output: W_{dir}: denoised adjacency matrix of direct network.

1. $\tilde{W}_{\mathrm{obs}} = W_{\mathrm{obs}} + \varepsilon_1 J + \varepsilon_2 I$ — Ensure that the input matrix mat is symmetric and normalized
2. $P_{\mathrm{obs}} = g(\tilde{W}_{\mathrm{obs}}) = \left(\mathrm{diag}\{\tilde{W}_{\mathrm{obs}} 1\}\right)^{-1} \tilde{W}_{\mathrm{obs}}$
3. $P_{\mathrm{dir}} = f_m^{-1}(P_{\mathrm{obs}}) = m\left((m-1)I + P_{\mathrm{obs}}\right)^{-1} P_{\mathrm{obs}}$
4. for $i = 1, \ldots, n$:
   if $\min_j\{(P_{\mathrm{dir}})_{ij}\} \geq 0$: $\beta_i = 0$
   else: $\beta_i = \min_j\{(P_{\mathrm{dir}})_{ij}\}$
5. $\tilde{P}_{\mathrm{dir}} = P_{\mathrm{dir}} - (\beta_1 1, \ldots, \beta_n 1)^T$
6. $W_{\mathrm{dir}} = h(\tilde{P}_{\mathrm{dir}}) = \mathrm{diag}\{\pi(\tilde{P}_{\mathrm{dir}})\}\tilde{P}_{\mathrm{dir}}$

Methodology (Pseudo code)

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

Pseudocode for RENDOR

Input: $\boldsymbol{W}_{\text{obs}}$: weighted adjacency matrix of observed network;

$\boldsymbol{m}$: diffusion intensity parameter;

$\varepsilon_1, \varepsilon_2$: preprocessing parameters.

Output: $\boldsymbol{W}_{\text{dir}}$: denoised adjacency matrix of direct network.

1. $\tilde{\boldsymbol{W}}_{\text{obs}} = \boldsymbol{W}_{\text{obs}} + \varepsilon_1 \boldsymbol{J} + \varepsilon_2 \boldsymbol{I}$

Ensure that the input matrix mat is symmetric and normalized

2. $\boldsymbol{P}_{\text{obs}} = \boldsymbol{g}(\tilde{\boldsymbol{W}}_{\text{obs}}) = \left(\text{diag}\{\tilde{\boldsymbol{W}}_{\text{obs}} 1\}\right)^{-1} \tilde{\boldsymbol{W}}_{\text{obs}}$

$\boxed{g(W) = D^{-1} W}$

3. $\boldsymbol{P}_{\text{dir}} = \boldsymbol{f}_m^{-1}(\boldsymbol{P}_{\text{obs}}) = \boldsymbol{m}((\boldsymbol{m}-1)\boldsymbol{I} + \boldsymbol{P}_{\text{obs}})^{-1} \boldsymbol{P}_{\text{obs}}$
4. for $\boldsymbol{i} = 1, \ldots, \boldsymbol{n}$:
   if $\min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\} \geq 0$: $\boldsymbol{\beta}_i = 0$
   else: $\boldsymbol{\beta}_i = \min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\}$
5. $\tilde{\boldsymbol{P}}_{\text{dir}} = \boldsymbol{P}_{\text{dir}} - (\boldsymbol{\beta}_1 1, \ldots, \boldsymbol{\beta}_n 1)^T$
6. $\boldsymbol{W}_{\text{dir}} = \boldsymbol{h}(\tilde{\boldsymbol{P}}_{\text{dir}}) = \text{diag}\{\boldsymbol{\pi}(\tilde{\boldsymbol{P}}_{\text{dir}})\} \tilde{\boldsymbol{P}}_{\text{dir}}$

Methodology (Pseudo code)

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

Pseudocode for RENDOR

Input: $\boldsymbol{W}_{\text{obs}}$: weighted adjacency matrix of observed network;

$\boldsymbol{m}$: diffusion intensity parameter;

$\boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2$: preprocessing parameters.

Output: $\boldsymbol{W}_{\text{dir}}$: denoised adjacency matrix of direct network.

1. $\tilde{\boldsymbol{W}}_{\text{obs}} = \boldsymbol{W}_{\text{obs}} + \boldsymbol{\varepsilon}_1\boldsymbol{J} + \boldsymbol{\varepsilon}_2\boldsymbol{I}$
2. $\boldsymbol{P}_{\text{obs}} = \boldsymbol{g}(\tilde{\boldsymbol{W}}_{\text{obs}}) = \left(\text{diag}\{\tilde{\boldsymbol{W}}_{\text{obs}}\boldsymbol{1}\}\right)^{-1}\tilde{\boldsymbol{W}}_{\text{obs}}$
3. $\boldsymbol{P}_{\text{dir}} = \boldsymbol{f}_{\boldsymbol{m}}^{-1}(\boldsymbol{P}_{\text{obs}}) = \boldsymbol{m}((\boldsymbol{m}-1)\boldsymbol{I} + \boldsymbol{P}_{\text{obs}})^{-1}\boldsymbol{P}_{\text{obs}}$
4. for $\boldsymbol{i} = 1, \ldots, \boldsymbol{n}$:
   if $\min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\} \geq 0$: $\boldsymbol{\beta}_i = 0$
   else: $\boldsymbol{\beta}_i = \min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\}$
5. $\tilde{\boldsymbol{P}}_{\text{dir}} = \boldsymbol{P}_{\text{dir}} - (\boldsymbol{\beta}_1\boldsymbol{1}, \ldots, \boldsymbol{\beta}_n\boldsymbol{1})^T$
6. $\boldsymbol{W}_{\text{dir}} = \boldsymbol{h}(\tilde{\boldsymbol{P}}_{\text{dir}}) = \text{diag}\{\boldsymbol{\pi}(\tilde{\boldsymbol{P}}_{\text{dir}})\}\tilde{\boldsymbol{P}}_{\text{dir}}$

Ensure that the input matrix mat is symmetric and normalized

$$\boxed{g(W) = D^{-1}W}$$

$$\boxed{f_m^{-1}(P) = m((m-1)I + P)^{-1}P}$$

Methodology (Pseudo code)

RENDOR =

$$h\left(f_m^{-1}\left(g(W)\right)\right)$$

Pseudocode for RENDOR

Input: $\boldsymbol{W}_{\text{obs}}$: weighted adjacency matrix of observed network;

$\boldsymbol{m}$: diffusion intensity parameter;

$\boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2$: preprocessing parameters.

Output: $\boldsymbol{W}_{\text{dir}}$: denoised adjacency matrix of direct network.

1. $\tilde{\boldsymbol{W}}_{\text{obs}} = \boldsymbol{W}_{\text{obs}} + \boldsymbol{\varepsilon}_1 \boldsymbol{J} + \boldsymbol{\varepsilon}_2 \boldsymbol{I}$ Ensure that the input matrix mat is symmetric and normalized
2. $\boldsymbol{P}_{\text{obs}} = \boldsymbol{g}\left(\tilde{\boldsymbol{W}}_{\text{obs}}\right) = \left(\text{diag}\left\{\tilde{\boldsymbol{W}}_{\text{obs}} \boldsymbol{1}\right\}\right)^{-1} \tilde{\boldsymbol{W}}_{\text{obs}}$ $\boxed{g(W) = D^{-1} W}$
3. $\boldsymbol{P}_{\text{dir}} = \boldsymbol{f}_{\boldsymbol{m}}^{-1}(\boldsymbol{P}_{\text{obs}}) = \boldsymbol{m}\left((\boldsymbol{m}-1)\boldsymbol{I} + \boldsymbol{P}_{\text{obs}}\right)^{-1} \boldsymbol{P}_{\text{obs}}$ $\boxed{f_m^{-1}(P) = m((m-1)I + P)^{-1} P}$
4. for $\boldsymbol{i} = 1, \ldots, \boldsymbol{n}$:
   if $\min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\} \geq 0$: $\boldsymbol{\beta}_i = 0$
   else: $\boldsymbol{\beta}_i = \min_j\{(\boldsymbol{P}_{\text{dir}})_{ij}\}$
5. $\tilde{\boldsymbol{P}}_{\text{dir}} = \boldsymbol{P}_{\text{dir}} - (\boldsymbol{\beta}_1 \boldsymbol{1}, \ldots, \boldsymbol{\beta}_n \boldsymbol{1})^T$
6. $\boldsymbol{W}_{\text{dir}} = \boldsymbol{h}\left(\tilde{\boldsymbol{P}}_{\text{dir}}\right) = \text{diag}\left\{\boldsymbol{\pi}\left(\tilde{\boldsymbol{P}}_{\text{dir}}\right)\right\} \tilde{\boldsymbol{P}}_{\text{dir}}$ $\boxed{h(P) = \alpha \cdot \text{diag}(\pi(P))}$

Methodology (code from github)

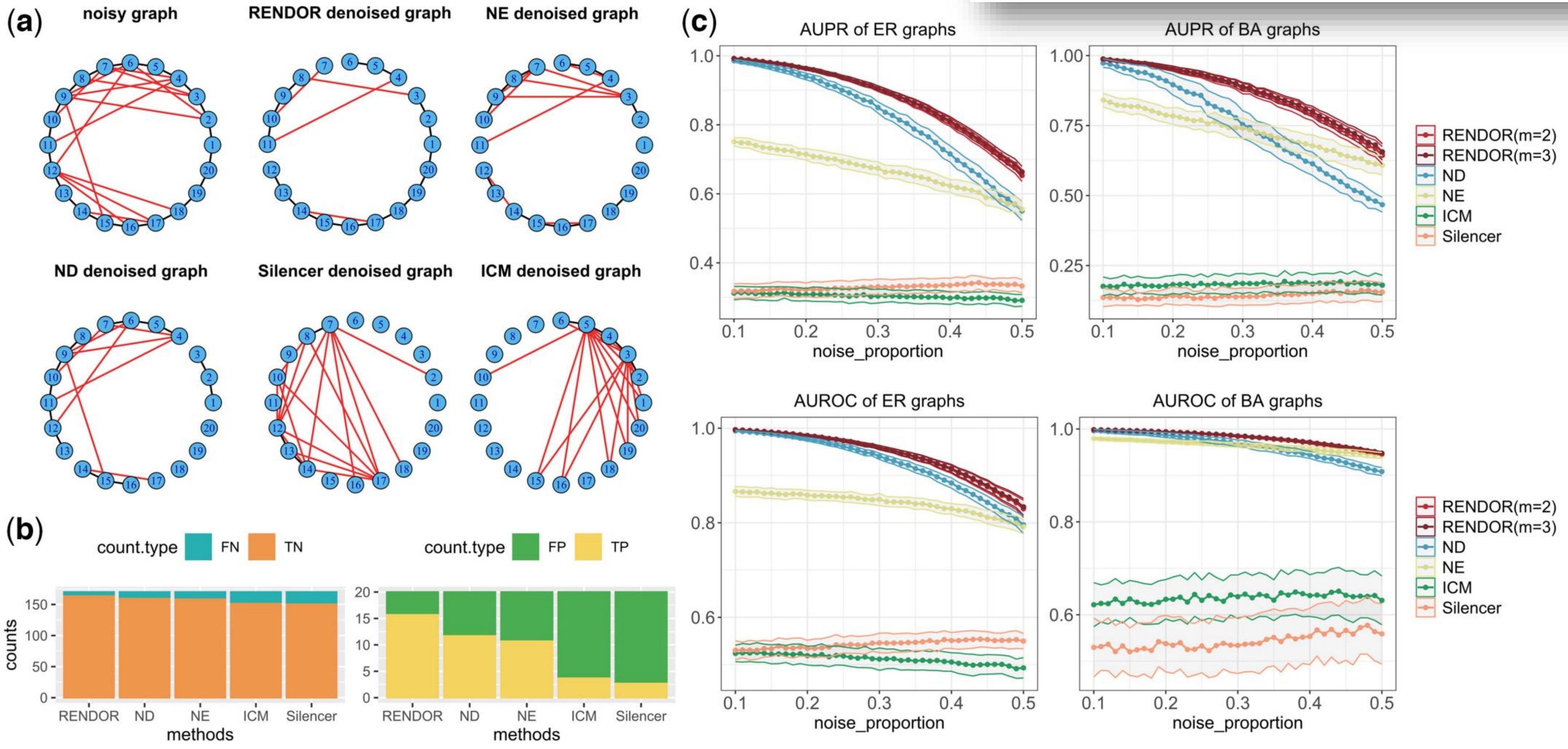
```
function [output_network]=RNDRW(mat, m)

[n_tf,n]=size(mat);
for i=1:n_tf
    mat(i,i)=0;
end


%% **************** input matrix imputation ****************
mat(1:n_tf,1:n_tf)=(mat(1:n_tf,1:n_tf)+mat(1:n_tf,1:n_tf)')/2;
mat1=[mat;[zeros(n-n_tf,n_tf),eye(n-n_tf,n-n_tf)]];
mat1=(mat1+mat1')/2;
mat1=(mat1-min(mat1(:)))/(max(mat1(:))-min(mat1(:)));
mat1=mat1+min(mat1(mat1>0))+min(mat1(mat1>0))*eye(n);


% mat1=[mat;[mat(:,(n_tf+1):end)',eye(n-n_tf,n-n_tf)]];
% mat1=(mat1+mat1')/2;
% mat1=(mat1-min(mat1(:)))./(max(mat1(:))-min(mat1(:)));
% mat1=mat1+min(mat1(mat1>0));
```

```


%% *********************** RNDRW ****************************
P1 = mat1./sum(mat1,2);
P2 = m * P1 /((m-1)*eye(n) + P1);
P2 = P2 - min(min(transpose(P2)),0)';
P2 = P2 ./ sum(P2,2);
stat_d = abs(null((P2-eye(n))'));
net_new = diag(stat_d)*P2;


%% ******************************************************
net_new = net_new + net_new';
output_network = net_new(1:n_tf, :);
```

Experiments

on the simulated networks

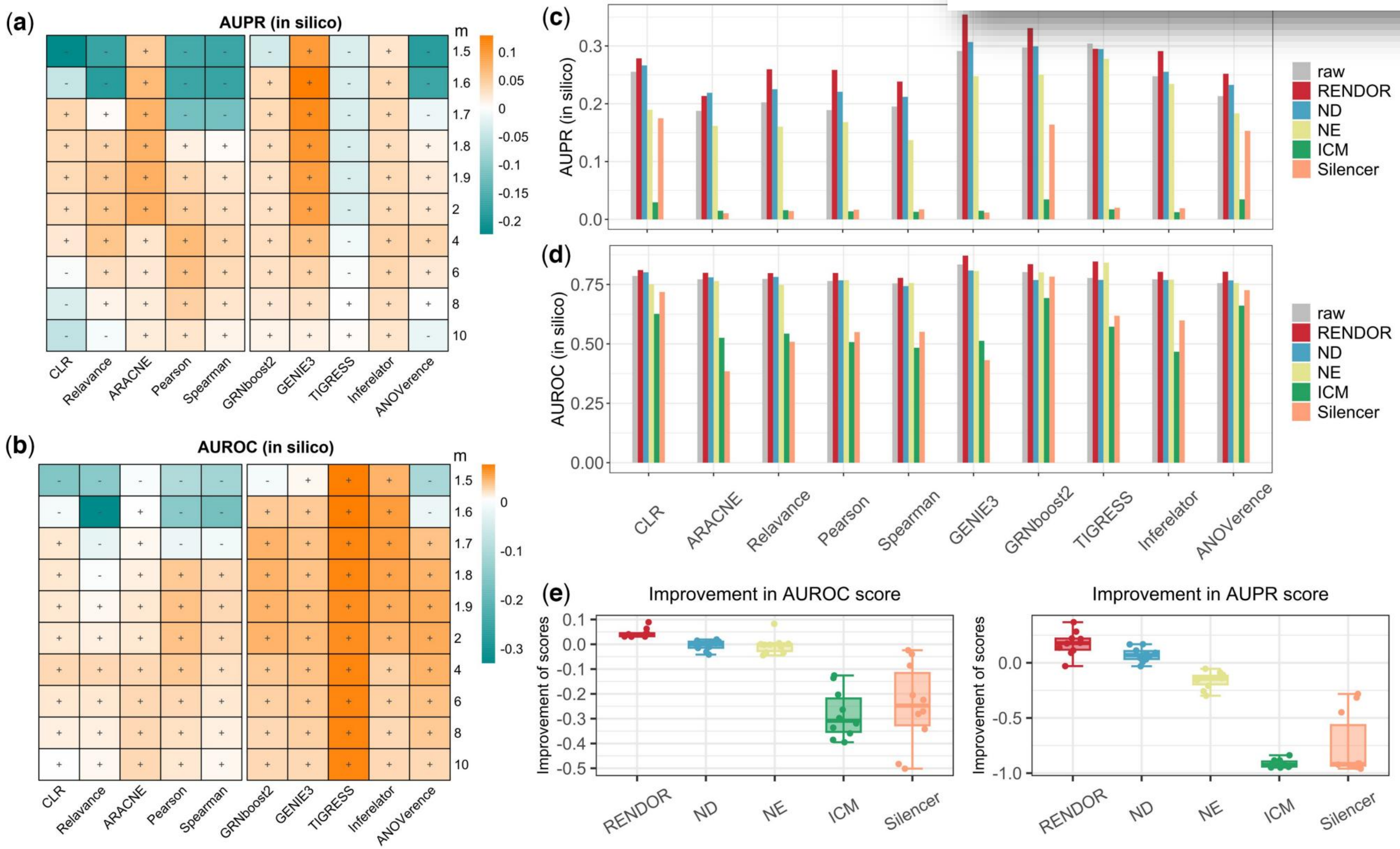
We compared the denoising performance of RENDOR with four other state-of-the-art GRN denoising methods: ND (Feizi *et al.* 2013), NE (Wang *et al.* 2018), Silencer (Barzel and Barabási 2013), and inverse correlation matrix (ICM) (Alipanahi and Frey 2013).



Experiments

on DREAM5 dataset

We compared the denoising performance of RENDOR with four other state-of-the-art GRN denoising methods: ND (Feizi *et al.* 2013), NE (Wang *et al.* 2018), Silencer (Barzel and Barabási 2013), and inverse correlation matrix (ICM) (Alipanahi and Frey 2013).



Conclusion

- In this work, authors propose **RENDOR**, a novel denoising approach for improving the accuracy of **network inference**.

- RENDOR is designed to handle noisy networks affected by indirect effects.
    - effectively models higher-order indirect interactions between nodes through network diffusion, employs reverse network diffusion to **eliminate indirect effects**, and outputs refined networks containing only direct signal edges.

- Through comprehensive evaluations on both **simulated noisy networks** and **real GRNs**, the authors demonstrated that RENDOR consistently outperforms alternative denoising methods for GRN inference, enhancing the inference accuracy by effectively mitigating the impact of indirect noise.