Boolean Network Models of Human Preimplantation Development

Mathieu Bolteau, Lokmane Chebouba, Laurent David, Jérémie Bourdon, Carito Guziolowski

Boolean Network Models of Human Preimplantation Development

Mathieu Bolteau[1*], Lokmane Chebouba[2,3], Laurent David[4],
Jérémie Bourdon[1], Carito Guziolowski[1*]
[1]Nantes Université, École Centrale Nantes,
CNRS, LS2N, UMR 6004, F-44000, Nantes, France
[2]University of Science and Technology Houari Boumediene (USTHB),
El-Alia BP 32 Bab-Ezzouar, Algiers, 16111, Algeria
[3]University of Constantine 1
[4]Nantes Université, CHU Nantes, INSERM, Center for
Research in Transplantation and Translational Immunology,
UMR 1064, F-44000, Nantes, France
*To whom correspondence should be addressed;
E-mail: {firstname.lastname}@ls2n.fr.

January 31, 2024

Keywords: Boolean Networks, Human Embryonic Development, scRNAseq, Systems Biology, Answer Set Programming

Abstract: Single-cell transcriptomic studies of differentiating systems allow meaningful understanding, especially in human

embryonic development and cell fate determination. We present an innovative method aimed at modeling these intricate processes by leveraging scRNAseq data from various human developmental stages. Our implemented method identifies pseudo-perturbations, since actual perturbations are unavailable due to ethical and technical constraints. By integrating these pseudo-perturbations with prior knowledge gene interactions, our framework generates stage-specific Boolean networks (BNs). We apply our method to medium and late trophectoderm developmental stages and identify 20 pseudo-perturbations required to infer BNs. The resulting BN families delineate distinct regulatory mechanisms, enabling the differentiation between these developmental stages. We show that our program outperforms existing pseudo-perturbation identification tool. Our framework contributes to comprehending human developmental processes and holds potential applicability to diverse developmental stages and other research scenarios.

1 Introduction

Current assisted reproductive technologies, specifically in vitro fertilization (IVF), struggles with a 25% success rate, necessitating innovative approaches. Advanced technologies like transcriptomics and proteomics offer a deeper understanding of embryo development. Deciphering this process aims to refine and establish robust embryo quality assessment techniques. Our field's future lies in computational models for preimplantation development, starting with transcription factor networks, invaluable for predicting perturbation impacts.

Understanding the sequence of events that regulate human preimplantation development remains a central question. In our work (Meistermann et al.,

2021), single-cell transcriptomic (scRNAseq) analysis delineated transcription factor hierarchies. In cancer, Chevalier et al. (2020) inferred Boolean networks from scRNAseq using pseudo-time distributions, focusing on cell fate dynamics through averaged cell expressions at each stage. Dunn et al. (2019) utilized knockout data on mouse stem cells, revealing insights via perturbation-based approaches.

Our study, an extension of our previous research (Bolteau et al., 2023), introduces a framework to derive Boolean networks (BNs) illustrating human preimplantation development's progression. Leveraging prior knowledge network (PKN) and scRNAseq data mapping, our approach identifies stage-specific pseudo-perturbations. This step is essential due to the limited availability of perturbation data, prompting us to extract pseudo-perturbations from scRNAseq data, leveraging its redundancy and sparsity. These identified pseudo-perturbations play a crucial role in constructing stage-specific Boolean Network (BN) models. In particular, our approach models each state in human development with a pool of 20 cells expressing different gene expression behavior.

2 Background

In this section we define the concepts needed to understand our method.

Prior Knowledge Network A Prior Knowledge Network (PKN) is a signed and oriented graph, where nodes correspond to biological entities (*e.g.*, genes, proteins), and edges represent causal or functional relationships between these entities (Radulescu et al., 2006). Nodes in the PKN are categorized into four types principally based on the graph's topology: *(i) protein complexes*, explicitly denoting protein complexes; *(ii) inputs*,

representing nodes without predecessors; *(iii) readouts*, denoting nodes without successors; and *(iv) intermediates*, encompassing other nodes.

Pseudo-perturbation A pseudo-perturbation represents a Boolean vector that encodes the expression status of a set of k genes (of type *input* and *intermediate*) within a particular cell. In the context of comparing cells across various classes or developmental stages, a *match* refers to a pair of 2 pseudo-perturbations from different classes that exhibit an exact gene expression vector, signifying similarity in genetic activity.

Experimental Design An experimental design is composed of different pseudo-perturbation combinations in specific cells along with the associated values for the readout genes, encompassing Boolean values for pseudo-perturbations and normalized values for the readout genes. An experimental design can be seen as multiple entry-output values describing different states a biological system can take. Such information can be used to infer a model which explains globally the system's behavior.

ASP Introduction Answer Set Programming (ASP) (Baral, 2003) stands as a declarative programming paradigm utilized to delineate knowledge bases using rules. Atoms, composed of predicates with associated terms (integers, constants, or variables), form the fundamental units of these programs. Atoms assert basic knowledge in a straightforward manner. ASP's rules consist of a *head* and a *body* where the head, a set of atoms, denotes the goal, while the body delineates the conditions for its deduction. Within the body, *literals*, representing propositions, can be either positive or negative. Two specific rules exist in ASP: *facts* and *integrity constraints*. Facts, presented as rules with only a head and no body, serve to represent essential knowledge. Conversely, integrity constraints define limitations on the admissible interpretations of a knowledge base, ensuring that

only valid models are considered. We present in Supplementary Note 1, a logic program example to better understand the explained concepts. ASP provides a powerful way of specifying complex rules and constraints, proving instrumental in solving diverse problems across domains like artificial intelligence, natural language processing, and planning.

3 Materials and Methods

3.1 Data

In this study, we exploit scRNAseq data from stage-matched human embryos, leveraging the dataset initially compiled by Petropoulos et al. (2016) and subsequently refined in Meistermann et al. (2021). This dataset comprises the expression profiles of $34,054$ genes across $1,496$ cells derived from 88 stage-matched human embryos. Our analysis is based on the count matrix. We acknowledge and address the zero-inflation concern present in scRNAseq datasets (Jiang et al., 2022), noting that our dataset contains 63% zero values, a phenomenon we consider crucial in our methodology. Following up on the annotation provided by Meistermann et al. (2021), we focus on investigating the trophectoderm (TE) cell fate, specifically targeting the transition between the medium TE and late TE developmental stages, encompassing 248 and 332 cells, respectively. Of note, the TE cell fate maturation is necessary to promote embryo implantation, a key step in embryo viability and pregnancy success.

For preprocessing, we filtered the scRNAseq count matrix to retain genes present in the Prior Knowledge Network (see Section 3.3), binarizing input and intermediate genes. Genes were marked "expressed" (1) if at least 2 reads were observed, otherwise "absent" (0). Readout genes underwent "Min-Max"

Table 1: Case studies description.

Dataset	Source	Class name (C1;C2)	Genes[1]	Cells[2]	#C1's cells[2]	#C2's cells[2]
A	artificial	C1;C2	10	10	5	5
B	subset of single-cell data	E^{TE} ; M^{TE}	30	24	12	12
C	subset of single-cell data	E^{TE} ; M^{TE}	100	50	25	25
D	subset of single-cell data	E^{TE} ; M^{TE}	120	200	100	100
P	phosphoproteomics data[3]	CR ; PR	79	191	136	55
SC	single-cell data	M^{TE} ; L^{TE}	111	680	348	332

E^{TE} = early TE ; M^{TE} = medium TE ; L^{TE} = late TE ; CR = Complete Remission ; PR = Primary Resistant (see (Chebouba et al., 2018)). [1] For dataset P, proteins are studied (not genes). [2] For dataset P, patients are studied (not cells). [3] From Chebouba et al. (2018).

normalization, scaling values to $[0, 1]$. We curated diverse dataset subsets for testing various case studies, detailed in Table 1. Additionally, we included an extra case study (P in Table 1) from an alternate dataset with phosphoproteomics data sourced from Chebouba et al. (2018).

3.2 Method overview

Our method involves three steps aimed at constructing stage-specific Boolean Networks (BNs), as depicted in Figure 1. Initially, we reconstruct a Prior Knowledge Network (PKN) by querying a biological knowledge database to establish gene-gene interactions. Subsequently, we establish an experimental design tailored to each developmental stage, outlining the necessary entries and outputs essential for BN inference. Finally, we integrate the PKN with the previously created experimental designs to derive stage-specific BNs. Further elaboration on these steps is provided in the following sections. Our implemented framework, along with the complete set of data and results, can be accessed at the following link: https://doi.org/10.5281/zenodo.10580801 .

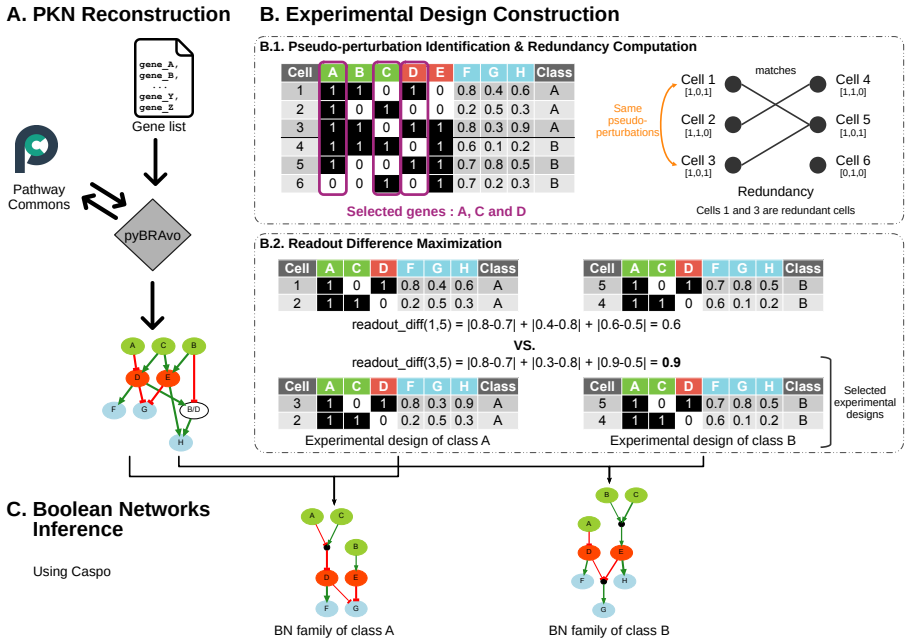

Figure 1: Developed framework comprising three main steps. **A.** The PKN was reconstructed using the pyBRAvo tool that queries the Pathway Commons database with an input gene list. **B.** The ASP program selects a set of k genes (here, $k = 3$ with A, C and D) to maximize pseudo-perturbation identification. In this example, 2 (optimal) pseudo-perturbations are identified: (cell 1, cell 5) and (cell 2, cell 4). Redundancies in scRNAseq data are observed, with cell 3 sharing the same Boolean vector as cell 1, representing two equivalent solutions. The second sub-step identifies pseudo-perturbations maximizing readout difference between the two classes, leading to the selection of pairs (cell 3, cell 5) and (cell 2, cell 4), forming the experimental design. **C.** Boolean networks (BNs) are inferred using the Caspo tool, combining the reconstructed PKN and both experimental designs. Each BN is compatible with the PKN topology and minimizes the gene expression error in the (entry-output) experimental designs.

3.3 PKN Reconstruction

In our framework, the reconstruction of the PKN relies on the utilization of the pyBRAvo tool (Lefebvre et al., 2021). Starting from a list of genes, pyBRAvo employs queries on the Pathways Commons resource to identify the predecessors of the initial genes (Figure 1A). This iterative process continues until a specified reconstruction depth is reached. Through pyBRAvo, we generate a gene interaction graph, serving as the foundational knowledge base for our methodology. Pathway Commons v.13 (Rodchenkov et al., 2019) is leveraged, excluding miRTarBase, MSigDB, and CTD databases to eliminate miRNA and toxicogenomics interactions.

Subsequently, this reconstructed PKN undergoes a reduction process to align with the genes present in both the PKN and the expression matrix. Furthermore, inputs directly linked to a single direct successor readout are eliminated to focus solely on gene regulation pathways involving various gene types (inputs, intermediates and readouts).

It is important to note that this method step is optional. Users possessing a PKN tailored to their specific case study have the flexibility to utilize it and execute the subsequent method steps. The approach aims to be as adaptable as possible to accommodate different user-specific PKNs.

3.4 Experimental Design Construction

In the context of developmental stage differentiation, an experimental design formulated for each class comprises the pseudo-perturbations and their associated readout values, maximizing differences between the two classes (Figure 1B). After obtaining pseudo-perturbations, thoroughly detailed in Section 3.4.1, we encounter various cells exhibiting identical Boolean vectors (Figure 1B.1), also named as *redundant cells*. To discern between redundant

cells we maximize the discrepancies in readout values for these cells across the two studied developmental stages (see Section 3.4.2). These pseudo-perturbations are amalgamated with readouts, fashioning specific experimental designs for each developmental stage under study. Although both experimental designs share the same pseudo-perturbation values (identical *entries*), they possess different readout values (distinct *outputs*). In the final step of our framework, we aim to understand the transition from identical entries in both classes to distinct outputs.

3.4.1 Pseudo-perturbation Identification

This method takes a binary matrix, E, where e_{ij} denotes the presence or absence of activity level of gene j in cell i. The objective is to generate a subset of genes and cells that adhere to various constraints, ensuring an identical number of pseudo-perturbations across different classes.

Problem Statement Let C represent the complete set of cells and G the complete set of genes in our experimental dataset. Each cell is uniquely associated with a class (A or B), hence $C = A \uplus B$. Using the binary matrix E, the relation I^G is defined such that $I^G(c_i) = \{g_j \in G \mid e_{ij} = 1\}$ represents the active genes for cell c_i. The restriction of I^G to a subset G' is denoted by $I^{G'}(c_i) = I^G(c_i) \cap G'$.

Problem formulation. Given the association matrix E between the set G of genes and the set C of cells (where C comprises cells from two disjoint sets, A and B), along with a parameter k limiting the number of selected genes, the aim is to identify a subset G' of genes and the largest subset C' ($C' = A' \uplus B' \subset C$, where $A' \subset A$ and $B' \subset B$) satisfying the following constraints:

1. The size of G' is fixed to k (for largest instances, $k << |G|$).

2. $\forall c_1, c_2 \in A'$ (resp. B') with $c_1 \neq c_2$, we ensure that $I^{G'}(c_1) \neq I^{G'}(c_2)$.

3. $\forall c_1 \in A'$ (resp. B'), $\exists c_2 \in B'$ (resp. A'), such that $I^{G'}(c_1) = I^{G'}(c_2)$.

This process results in a binary vector b^i for each $c_i \in C'$, where $b^i_j = 1$ (resp. $b^i_j = 0$) if gene $g_j \in I^{G'}(c_i)$ (resp. $\notin I^{G'}(c_i)$). This vector, termed a pseudo-perturbation, captures the gene expression profile within a subset of cells. Note that due to the non-uniqueness of sets G' and C', multiple pseudo-perturbation vectors may exist.

Constraints Justification. The imposed constraints play a crucial role within the framework, particularly in Boolean network inference and single-cell data analysis. *Constraint 1* reduces the search space, enhancing computational efficiency, and simplifying subsequent Boolean network learning steps. *Constraint 2* mitigates redundancy in gene selection from cells within the same class, essential due to zero values and redundancy in single-cell data. *Constraint 3* promotes similarity in gene expression between distinct classes, facilitating meaningful comparative analysis during Boolean network inference. Although cells from different classes exhibit evolutionary differences, selecting genes with similar expression patterns allows for comparable conditions, aiding accurate modeling of distinct regulatory mechanisms. Finally, a larger selection of pseudo-perturbations enhances the Boolean network inference, enabling exploration of various regulatory mechanisms.

In addition to this problem statement, we present in Supplementary Note 2 a line by line description of the ASP logic program.

3.4.2 Readout Difference Maximization

After obtaining pseudo-perturbations, we encounter various cells with identical Boolean vectors. To discern between redundant cells, our objective is to maximize the differences in readout values for these cells across the two

studied developmental stages (Figure 1B.2). Further details on this method are established in Supplementary Note 3. This process yields an association of each optimal pseudo-perturbation to a vector of normalized readout expressions, maximizing differences between the two classes.

3.5 Boolean Network Inference

Here, we outline the process of inferring and validating Boolean Networks (BNs) using our implemented method (Figure 1C). A *Boolean network* is a mathematical model employed to illustrate interactions among elements within a biological system. In this model, each component is represented as a node, and relationships between these components are defined by Boolean functions. These functions articulate the activation or deactivation of each node (such as genes or proteins) based on the states of the connected nodes (Kauffman, 1969).

We employed Caspo (Videla et al., 2017) to infer BNs for each studied classes. The goal is to derive BNs that accommodate both biological knowledge represented by gene interactions in the PKN and the observed gene expression in the case study. Using logical rules and constraints, Caspo computes BNs that best fit the data through optimizations (using the mean squared error function, MSE), allowing an optional fixed tolerance. The method outputs a set of BNs for each class, derived from the same input (prior knowledge and pseudo-perturbations data) but with different outputs (readout values). These BN families encapsulate knowledge and observations by establishing logical connections between genes. Comparing these BN families enables the identification of distinct behaviors between classes.

Additionally, we present a method that uses the MSE score for sorting cells into one of the two studied classes. This approach evaluates the alignment with

Boolean Network (BN) families, determining the most suitable class for gene expression observations. The assessment includes the computation of the MSE between the BN readout predictions and the actual readout measurements for each cell's gene expression. Subsequently, the cell is categorized into the class with the lowest MSE. The overall accuracy, per-class accuracy, and Balanced Accuracy (BAC) are then analyzed to provide a comprehensive evaluation of the sorting process.

4 Results

4.1 Comparing Pseudo-perturbation Generation Programs

Our pseudo-perturbations identification program, inspired from Chebouba et al. (2018), proposes an additional constraint (see Section 3.4.1, Problem Statement, constraint 2), specifically aiming to ensure the generation of distinct pseudo-perturbations comprising k expressed genes within the same class. While increasing computational time, it proves valuable in handling redundant scRNAseq data.

Both programs were applied to datasets $A - SC$ (see Table 2). For comparison purposes, we post-processed the results from Chebouba et al. (2018) by removing redundant solutions (values in parenthesis in Table 2). For further insights into dataset specific features, please refer to Table 1.

Optimal solutions were attained by both programs for datasets A and B (see Table 2). Suboptimal results are denoted with an asterisk (*) over the fixed timeouts. Our program (O in Table 2) yielded suboptimal results for datasets $C - SC$, while Chebouba *et al.*'s program (C in Table 2) exhibited suboptimal outcomes for datasets D and P. Chebouba *et al.*'s version

demonstrated shorter execution times than our version when no timeout was imposed. Analysis of the number of distinct pseudo-perturbations generated by each program reveals two different behaviors contingent on dataset nature and complexity. Firstly, for single-cell datasets ($A - D$ and SC), Chebouba *et al.*'s program computes either an equal or a smaller count of pseudo-perturbations when removing redundancies. For instance, for dataset C, Chebouba *et al.*'s program derives an optimal solution comprising only 1 distinct pseudo-perturbation, while our program yields suboptimal results, generating 6 and 11 distinct pseudo-perturbations for $k = 3$ or $k = 10$ respectively. This disparity is more pronounced in larger datasets like D (10 *vs.* 22) or SC (3 *vs.* 20). It indicates that Chebouba *et al.*'s program infers numerous redundant solutions, lacking the ability to differentiate them effectively, highlighting our program's superiority in handling scRNAseq data. Secondly, our version showcases superior outcomes for phosphoproteomics data (23 *vs.* 25), affirming its adaptability across single-cell or averaged cell population gene-expression datasets.

Table 2: Comparison of ASP programs on different datasets.

Dataset	k	Execution time		Distinct Pseudo-Perturbations	
		C	O	C	O
A	3	0.008s	0.008s	3 (4)	3
B	3	0.048s	0.223s	1 (132)	4
C	3	1.420s	10 min*	1 (625)	6
	10	1.424s	10 min*	1 (600)	11
D	10	10 min*	10 min*	10 (2,436)	22
P	10	50h*	50h*	23 (64)	25
SC	10	5h 2 min	65h*	3 (77,618)	20

C corresponds to the Chebouba *et al.*'s logic program, while O corresponds to our logic program. For Chebouba *et al.*'s program, in parenthesis, the total number of pseudo-perturbations vectors (redundancy comprising). * Execution time corresponds to the fixed timeout.

4.2 Application on a Real Case Study

4.2.1 PKN Reconstruction

We used 438 transcription factor (TF) genes involved in human embryonic development as input for pyBRAvo software (Lefebvre et al., 2021) to reconstruct a PKN. These TF genes were identified through SCENIC (Aibar et al., 2017) analysis of scRNAseq data (TF list in Supplementary Note 4). The exploration depth parameter was fixed to 2, *i.e.* up to 2 levels upstream of the initial TFs. Only gene transcription events were queried, yielding a PKN of 327 nodes and 475 edges, with only 28 of the 438 initial TFs found in the database. We then reduced the network to 191 nodes (84 input genes, 27 intermediate genes, 14 readout genes, and 66 complexes) and 285 edges (Figure 2), limited to genes measured in scRNAseq data and complexes linked to these genes.

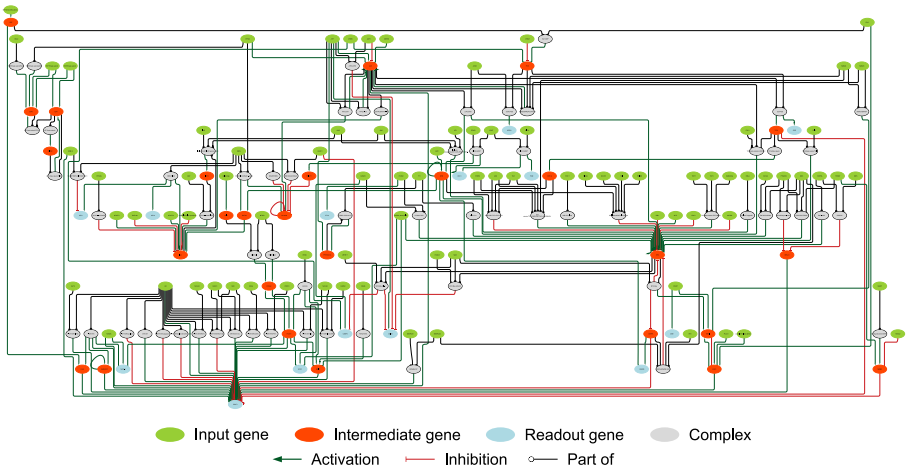

Figure 2: Reconstructed Prior Knowledge Network.

4.2.2 Experimental Design Reconstruction

Pseudo-perturbation Identification For pseudo-perturbations identification we used $k = 10$. This choice was determined in a prior study (Bolteau et al., 2023), which explored different k values and identified $k = 10$ as the most optimal. The program was executed on a computer cluster equipped with 160 CPUs and 1.5 TB of RAM, requiring 65 hours to generate 20 pseudo-perturbations. It is crucial to emphasize that this outcome is sub-optimal, *i.e.* a greater number of pseudo-perturbations could potentially be identified. Additionally, the set of 20 pseudo-perturbations obtained is not unique: more than 1 million of various sets of 10-genes could also lead to 20 pseudo-perturbations.

The 20 selected matching cells represent a significant portion of the overall cell pool for both developmental stages for the 10 selected genes. Each stage includes redundant cells, with 242 for medium TE and 276 for late TE, collectively constituting, respectively, 75.2% (262/348) and 89.1% (296/332) of the total cells.

Experimental Design Reconstructed From the selected cells by pseudo-perturbations, we ran the readout difference maximization (see Section 3.4.2) to select the most different cells between both developmental stages, and combine these pseudo-perturbations with readout expression to form the experimental design. In Figure 3, we visualize the 20 pseudo-perturbations and the associated expression for the readouts retrieved in the inferred BNs (see Figure 4).

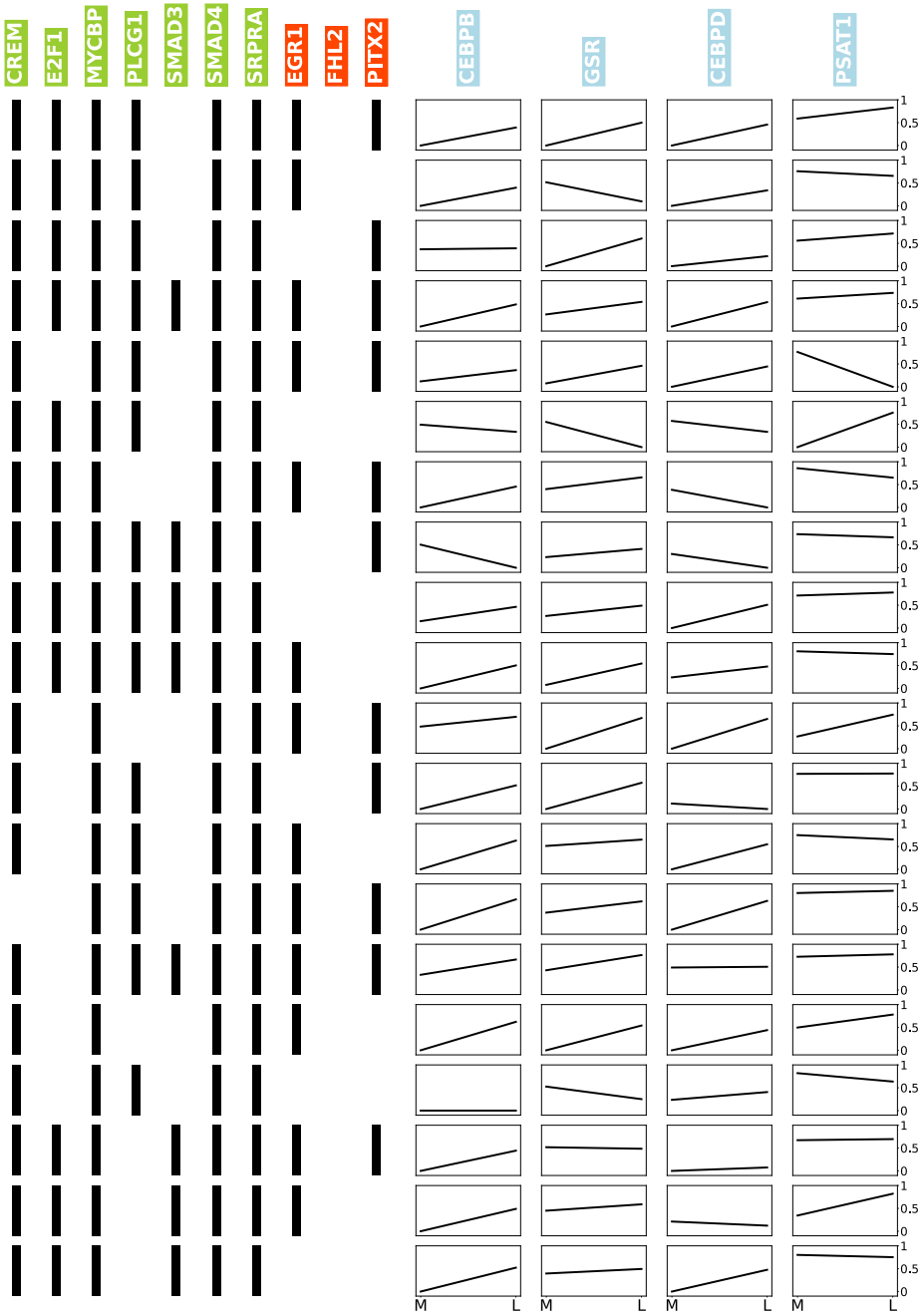

Figure 3: Graphical representation of computed experimental designs. Each row (left side) represents an optimal pseudo-perturbation on the 10 selected input (green) and intermediate (red) genes. Binarized vectors are illustrated using bars, where a black (resp. white) bar means the gene is active (resp. inactive). On the right side, readout genes in blue, present in the inferred BNs (cf. Figure 4), are shown. In each box, the curve represents the normalized readout gene expression evolution between the medium TE (M, left) and late TE (L, right) developmental stages.

4.2.3 Boolean Network Inference

Families of BNs Using Caspo software (Videla et al., 2017), we infer families of BNs for medium and late TE using the reconstructed PKN and specific experimental designs. Caspo generates BNs adhering to PKN topology, optimizing the Mean Square Error (MSE) between Boolean predictions of readout nodes and experimental measurements. Our Caspo parameters are set as: (i) $length = 2$ to restrict nodes receiving at most 2 "AND" logical functions, and (ii) $fit = 0.001$ to permit exploration beyond the optimal BN up to a distance of 0.1% from the optimal MSE.

Figure 4 displays the learned BNs union for the two studied developmental stages. The size is equal to 8 for medium TE and 15 for late TE, with respective optimal MSEs of 0.1421 and 0.1924. The higher MSE for late TE indicates a more intricate fitting between inferred BNs and experimental data, resulting in less precise BNs compared to medium TE. While the medium TE family contains 2 BNs, the late TE family comprises 4. Notably, these BN families diverge in gene regulatory mechanisms. Late TE BNs display more extensive connectivity with 3 inputs and 4 readouts, compared to 2 inputs and 1 readout in medium TE. Both share 4 genes, including 2 inputs (*SMAD3* and *E2F1*), 1 intermediate (*EGR1*), and 1 readout (*PSAT1*). Late BNs exhibit supplementary readout genes, namely *GSR*, *CEBPB*, and *CEBPD*, indicating that the readout measurements matched the late TE BNs prediction, given the selected pseudo-perturbation Boolean vectors. However, medium TE BNs could not predict the observed measurements with minimal error on these three genes. This suggests higher complexity in late TE regulatory mechanisms compared to medium TE.

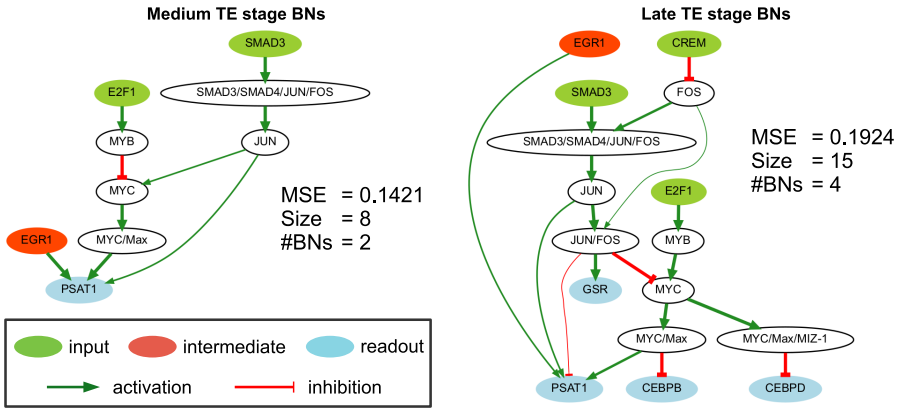

Figure 4: Families of inferred Boolean networks (BNs) for medium and late TE developmental stages. Each network represents the union of (sub-)optimal BNs learned from the reduced PKN and the experimental design. The width of the arcs represents the frequency of occurrence of this arc in the BNs.

Single Pseudo-perturbation Analysis We employed the Mean Square Error (MSE) metric to sort each 20 cells identified as pseudo-perturbations in both medium and late TE stages, comparing the calculated gene expression across the inferred Boolean Networks (BNs) with actual data (Figure 5). Notably, the sorting accuracy for medium TE BNs stands for 70% (Figure 5A), highlighting their reliability in individual pseudo-perturbation predictions. In contrast, late TE stage outcomes reveal a lower accuracy of 15%, once again emphasizing the challenges in modeling the regulatory mechanisms of late TE. Examining the proximity of MSE values for cells in both medium and late TE, we observe that late TE cells are predominantly sorted into the medium TE class. Additionally, we note that for poorly sorted late TE cells, the MSE values obtained with late TE BNs are generally close to those obtained with medium TE BNs (Figure 5B). These observations could be explained by the fact that late TE BNs involve fitting 4 readouts, as opposed to 1 readout for medium, aligning with the lower sorting outcomes for

A.

#Cells	Accuracy	Medium TE Accuracy	Late TE Accuracy	BAC
40	43 % (17/40)	70 % (14/20)	15 % (3/20)	43 %

B.

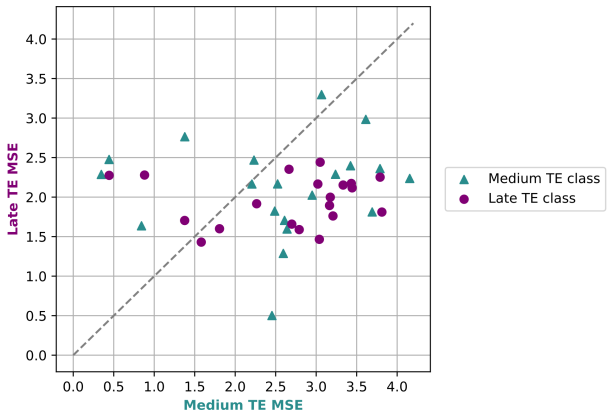

Figure 5: Mean Squared Error (MSE) scores outcomes. **A.** Description of the accuracy results obtain using our MSE score method. **B.** Medium TE *vs.* Late TE MSE obtain for each 20 cells involved in pseudo-perturbations (values in negative logarithmic scale).

late TE. Additionally, the higher MSE for late TE (see Section 4.2.3, Families of BNs), indicates a lesser accuracy with experimental data.

In conclusion, this analysis, bridging BN network logical behavior to single cell data, could be useful for selecting the best answers of selected genes maximizing pseudo-perturbations. As mentioned before, numerous equivalent answers exist for 10-genes (over 1 million of equivalent solutions – cf. Section 4.2.2). Therefore, numerous sets of BN families could be proposed.

5 Discussion and Conclusion

Studying human embryonic development poses challenges, often requiring system perturbations, which are infeasible here. To overcome this, we propose an original framework utilizing human embryonic scRNAseq data to identify

pseudo-perturbations and construct Boolean network families representing two specific developmental stages.

In this study, our method identifies 20 pseudo-perturbations that show gene expression entry-output behavior across cells in two stages. Our pipeline computes Boolean networks families modeling medium and late trophectoderm (TE) stages, unveiling higher complexity in late TE. Notably, our program outperforms existing techniques like Chebouba et al. (2018) in pseudo-perturbation identification. This pipeline infers Boolean networks to model and compare developmental stages, presenting a novel methodology. Compared to other methods proposing computational models from scRNAseq data (Dunn et al., 2019; Chevalier et al., 2020), our method outputs logic models when no perturbation data is available, and uses a pool of cell behaviors to represent a single developmental stage. Extending this approach to other developmental stages could deepen our understanding of regulatory mechanisms in the human embryonic development. In addition, its adaptability makes it versatile for diverse case studies. As part of our work, we aim to enhance the metric for sorting single cells. While the Mean Square Error (MSE) utilized in our study is a commonly used metric, it may yield improved class predictions when integrated with additional types of information, such as the significance of gene expression readouts. Another aspect we are presently addressing involves refining the efficiency of our logic program.

Acknowledgments

We are grateful to the Bioinformatics Core Facility of Nantes BiRD, member of Biogenouest, Institut Français de Bioinformatique (IFB) (ANR-11-INBS-0013) for the use of its resources and for its technical support.

In addition, we acknowledge the use of the GLiCID computing center's HPC resources for some of the experimental procedures.

Authors' Contributions

M.B.: Conceptualization; Data curation; Formal analysis; Investigation; Methodology; Software; Validation; Visualization; Writing – original draft; Writing – review & editing. **L.C.**: Formal analysis; Investigation; Writing – review & editing (supporting). **L.D.**: Conceptualization; Funding acquisition; Supervision; Writing – review & editing. **J.B.**: Conceptualization; Funding acquisition; Investigation; Methodology; Supervision; Writing – review & editing. **C.G.**: Conceptualization; Funding acquisition; Investigation; Methodology; Supervision; Writing – review & editing.

Authors' Disclosure Statement

The authors declare no conflicts of interest or financial disclosures related to this research.

Funding Statement

This work is supported in part by funds from the Agence Nationale de la Recherche [ANR-20-THIA-0011 to M.B., ANR-20-CE17-0007 to L.D. and M.B.].

References

Aibar, S., González-Blas, C. B., Moerman, T., Huynh-Thu, V. A., Imrichova, H., Hulselmans, G., Rambow, F., Marine, J.-C., Geurts, P., Aerts, J.,

et al. Scenic: single-cell regulatory network inference and clustering. *Nature methods*, 14(11):1083–1086, 2017. doi: https://doi.org/10.1038/nmeth.4463.

Baral, C. *Knowledge Representation, Reasoning, and Declarative Problem Solving*. Cambridge University Press, New York, NY, USA, 2003. ISBN 0521818028.

Bolteau, M., Bourdon, J., David, L., and Guziolowski, C. Inferring Boolean Networks from Single-Cell Human Embryo Datasets. In Guo, X., Mangul, S., Patterson, M., and Zelikovsky, A., editors, *Bioinformatics Research and Applications*, Lecture Notes in Computer Science, pages 431–441, Singapore, 2023. Springer Nature. ISBN 9789819970742. doi: https://doi.org/10.1007/978-981-99-7074-2_34.

Chebouba, L., Miannay, B., Boughaci, D., and Guziolowski, C. Discriminate the response of acute myeloid leukemia patients to treatment by using proteomics data and answer set programming. *BMC Bioinformatics*, 19(2):15–26, 2018. doi: https://doi.org/10.1186/s12859-018-2034-4.

Chevalier, S., Noël, V., Calzone, L., Zinovyev, A., and Paulevé, L. Synthesis and simulation of ensembles of boolean networks for cell fate decision. In Abate, A., Petrov, T., and Wolf, V., editors, *Computational Methods in Systems Biology*, pages 193–209, Cham, 2020. Springer International Publishing. ISBN 978-3-030-60327-4.

Dunn, S. J., Li, M. A., Carbognin, E., Smith, A., and Martello, G. A common molecular logic determines embryonic stem cell self-renewal and reprogramming. *EMBO J*, 38(1), Jan 2019. doi: https://doi.org/10.15252/embj.2018100003.

Jiang, R., Sun, T., Song, D., and Li, J. J. Statistics or biology: the zero-inflation

controversy about scrna-seq data. *Genome Biology*, 23(1):31, 2022. ISSN 1474-760X. doi: 10.1186/s13059-022-02601-5.

Kauffman, S. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467, 1969. ISSN 0022-5193. doi: https://doi.org/10.1016/0022-5193(69)90015-0.

Lefebvre, M., Gaignard, A., Folschette, M., Bourdon, J., and Guziolowski, C. Large-scale regulatory and signaling network assembly through linked open data. *Database*, 2021, 2021. doi: https://doi.org/10.1093/database/baaa113.

Meistermann, D., Bruneau, A., Loubersac, S., Reignier, A., Firmin, J., François-Campion, V., Kilens, S., Lelièvre, Y., Lammers, J., Feyeux, M., Hulin, P., Nedellec, S., Bretin, B., Castel, G., Allègre, N., Covin, S., Bihouée, A., Soumillon, M., Mikkelsen, T., Barrière, P., Chazaud, C., Chappell, J., Pasque, V., Bourdon, J., Fréour, T., and David, L. Integrated pseudotime analysis of human pre-implantation embryo single-cell transcriptomes reveals the dynamics of lineage specification. *Cell Stem Cell*, 28(9):1625–1640.e6, 2021. doi: https://doi.org/10.1016/j.stem.2021.04.027.

Petropoulos, S., Edsgärd, D., Reinius, B., Deng, Q., Panula, S. P., Codeluppi, S., Reyes, A. P., Linnarsson, S., Sandberg, R., and Lanner, F. Single-cell rna-seq reveals lineage and x chromosome dynamics in human preimplantation embryos. *Cell*, 165(4):1012–1026, 2016. doi: https://doi.org/10.1016/j.cell.2016.03.023.

Radulescu, O., Lagarrigue, S., Siegel, A., Veber, P., and Le Borgne, M. Topology and static response of interaction networks in molecular biology. *Journal of The Royal Society Interface*, 3(6):185–196, 2006. doi: 10.1098/rsif.2005.0092.

Rodchenkov, I., Babur, O., Luna, A., Aksoy, B. A., Wong, J. V., Fong, D., Franz, M., Siper, M. C., Cheung, M., Wrana, M., Mistry, H., Mosier, L.,

Dlin, J., Wen, Q., O'Callaghan, C., Li, W., Elder, G., Smith, P. T., Dallago, C., Cerami, E., Gross, B., Dogrusoz, U., Demir, E., Bader, G. D., and Sander, C. Pathway Commons 2019 Update: integration, analysis and exploration of pathway data. *Nucleic Acids Research*, 48(D1):D489–D497, 10 2019. ISSN 0305-1048. doi: https://doi.org/10.1093/nar/gkz946.

Videla, S., Saez-Rodriguez, J., Guziolowski, C., Siegel, A., and Wren, J. caspo: a toolbox for automated reasoning on the response of logical signaling networks families. *Bioinformatics*, 33(6):947–950, 2017. ISSN 33/6/947/2585024. doi: https://doi.org/10.1093/bioinformatics/btw738.

Boolean Network Models of Human Preimplantation Development

Mathieu Bolteau[1*], Lokmane Chebouba[2,3], Laurent David[4],

Jérémie Bourdon[1], Carito Guziolowski[1*]

[1]Nantes Université, École Centrale Nantes,

CNRS, LS2N, UMR 6004, F-44000, Nantes, France

[2]University of Science and Technology Houari Boumediene (USTHB),

El-Alia BP 32 Bab-Ezzouar, Algiers, 16111, Algeria

[3]University of Constantine 1

[4]Nantes Université, CHU Nantes, INSERM, Center for

Research in Transplantation and Translational Immunology,

UMR 1064, F-44000, Nantes, France

[*]To whom correspondence should be addressed;

E-mail: {firstname.lastname}@ls2n.fr.

January 30, 2024

Supplementary Notes

Supplementary Note 1: Small example of an ASP program

We introduce a concise logic program example, demonstrating the concepts presented in the paper, in selecting potential perturbations for medium TE and late TE developmental stages (see Listing 1).

```
1  cell(c1). cell(c2). cell(c3).
2  class(early_TE). class(medium_TE). class(late_TE).
3  be_part(c1,early_TE). be_part(c2,medium_TE). be_part(c3,late_TE).
4  gene(g1). gene(g2).
5  expr(c1,g1,0). expr(c1,g2,0).
6  expr(c2,g1,0). expr(c2,g2,1).
7  expr(c3,g1,1). expr(c3,g2,1).
8  pert(C,G,S,CL) :-expr(C,G,S), cell(C), gene(G), be_part(C,CL).
9  {sel_pert(C,G,S,CL) : pert(C,G,S,CL)}.
10 :-sel_pert(_,_,_,early_TE).
11 #show sel_pert/4.
```

Listing 1: Example of a logic program.

The logic program delineates knowledge through facts (lines 1-7). For instance, the predicate `expr(c1,g1,0)` on line 4 signifies that in cell `c1`, the gene `g1` exhibits an expression value of `0`. Line 8 introduces a rule defining a predicate `pert/4` (with an arity of 4, *i.e.* composed of 4 terms), modeling experimental data, where gene G demonstrates an expression value S in cell C, associated with class CL. Line 9 selects a subset of predicates `pert/4` using `sel_pert/4`, representing the chosen perturbations. This construct, known as *choice rules*, is central in ASP modeling, generating potential combinations of candidate solutions. These solutions are typically filtered using constraints. For instance, line 10 restricts the solution candidate `sel_pert/4` linked with the class *early TE*. Finally, line 11 presents the answer to this program, focusing solely on the `sel_pert/4` predicate. This answer is rendered as a set of assignments (answer sets) of constants to the terms of the `sel_pert/4` predicate; each assignment validates all program rules.

Supplementary Note 2: Implemented ASP program

In this note, we provide a step-by-step explanation of the ASP program devised to identify pseudo-perturbations. Our program, based on the method proposed in Chebouba et al. (2018), differs primarily in the rule governing the generation of distinct Boolean pseudo-perturbation vectors. Our logic program is tailored specifically to handle scRNAseq data, which often exhibits redundancy due to cells within the same developmental stage sharing identical gene expressions. Additionally, scRNAseq data commonly presents a strong abundance of zero values.

In Listing 2, we present the ASP encoding.

```
1  {selgene(G):pert(C,G,S,CL)} = k.
2  selpert(C,G,S,CL) :-selgene(G), pert(C,G,S,CL).
3  equal(I,J,G) :-selpert(I,G,S1,C1), selpert(J,G,S2,C2), C1<C2, S1 = S2.
4  countequal(I,J,M) :-M={equal(I,J,_)}.
5  0{affinity(I,J)}1 :-countequal(I,J,k).
6  nbInputOnes(C, N) :-N={pert(C,G,1,_) : selinput(G)}, affinity(C,_).
7  :-affinity(C,_), nbInputOnes(C,N), N<1.
8  diff(I1,I2,G) :-selpert(I1,G,S1,C1), selpert(I2,G,S2,C2), C1==C2, S1!=S2, I1<I2.
9  countdiff(I1,I2,M) :-M={diff(I1,I2,_)}.
10 :-countdiff(I1,I2,0), affinity(I1,_), affinity(I2,_), I1<I2.
11 :-countdiff(I1,I2,0), affinity(_,I1), affinity(_,I2), I1<I2.
12 #maximize{1,I: affinity(I,_)}
```

Listing 2: ASP encoding of pseudo-perturbation generation.

The `pert/4` predicate, an instance in our program referring to experimental data, emerges from discretized scRNAseq data associated with input and intermediate genes. It delineates the expression of gene G at value S in cell C, linked to class CL. Starting from line 1, our logic program initiates by selecting a set of k genes from all potential input and intermediate genes using the `selgene/1` predicate. This action generates $\binom{m}{k}$ answer sets, where m denotes the total count of input and intermediate genes. The subsequent rules aim to filter these candidate answer sets. Line 2 introduces the `selpert/4` predicate, summarizing experimental data for the selected genes. Following this, line 3 employs the `equal(I,J,G)` predicate to identify pairs of cells I and J from distinct classes (`C1<C2`), exhibiting the same measured value S for gene G (`S1=S2`). Moving for-

ward to line 4, the `countequal(I,J,M)` predicate enumerates the count of genes M demonstrating identical values across cells I and J. Recall that we are interested in finding k identical values associations for k genes. Therefore, in line 5, we define the predicate `affinity(I,J)`, which will be generated 0 or 1 times when there are k similarities for cells I and J, aiming to identify potential optimal pseudo-perturbations. Notably, the terms `affinity/2`, represented by variables I and J, pertain to cells in the first and second classes, respectively.

To handle data sparsity, lines 6-7 introduce rules. Line 6 defines `nbInputOnes/2`, calculating the count of input genes having a value of `1` for a cell C selected by the `affinity/2` predicate. Then, line 7 prohibits the selection of an `affinity(C,_)` if the count of 1-valued input genes in cell C is less than 1 (`N<1`). This ensures that each pseudo-perturbation has at least one active input gene, disallowing vectors where all genes remain inactive (equal to 0).

To discern distinct pseudo-perturbations, additional rules (lines 8-11) are introduced. Line 8 defines the `diff(I1,I2,G)` predicate, selecting cells $I1$ and $I2$ from the same class but with differing values for gene G. The subsequent `countdiff/3` predicate on line 9 records the disparities in expression values of the selected genes between cells $I1$ and $I2$. In line 10 (resp. line 11), the constraint forbids predicates `countdiff(I1,I2,0)`, where there is no difference in expression values for the selected genes, for cells $I1$ and $I2$ selected to be affinities in line 5 for the first class (resp. for the second class). Combined lines 5, 10, and 11 keep only one cell association when the same cell is associated with other cells, such as retaining only `affinity(c1,c2)` and not `affinity(c1,c3)` from possible associations.

Finally, line 12 aims to maximize associations specified by the `affinity/2` predicate concerning the first class, left term.

Supplementary Note 3: Readout difference maximization

Diverse cells have the same Boolean vectors as identified pseudo-perturbations. We search to maximize the difference in readout values for these redundant cells across the two studied developmental stages.

Mathematical definition Given a set of pseudo-perturbation binary vectors, O, and the matrix of preprocessed scRNAseq data with normalized readout values, the aim is to find sets of cells A' and B' associated by all pseudo-perturbation vectors in O that maximize the differences between their readout vectors, $r^{A'}$ (for readouts of cells in A') and $r^{B'}$ (for readouts of cells in B').

Algorithm For each vector b in the set of optimal pseudo-perturbations, relating cells c_1 (in A') and c_2 (in B'):

1. Identify a set of *redundant cells* for each class by finding cells in class A and B with identical binarized vector b (sets R_b^A and R_b^B). Both sets include cells c_1 and c_2 respectively.

2. Compare all pairs of cells in $R_b^A \times R_b^B$, calculating the readout gene value differences while retaining the maximum difference.

This process yields an association of each optimal pseudo-perturbation to a vector of normalized readout expressions, maximizing differences between the two classes.

Supplementary Note 4: Transcription factors list

Below we list the transcription factors used as input of pyBRAvo tool for the PKN reconstruction (see Section 4.2.1).

ACO1	CERS3	E2F8	FOSL1	HESX1	IRF8	LHX5
AKR1A1	CERS6	EBF1	FOXA2	HEY1	IRX2	LHX8
ANXA1	CLOCK	EBF2	FOXD2	HEY2	IRX3	LRRFIP1
ANXA11	CPEB1	EBF3	FOXN2	HHAT	IRX4	LSM6
ARG2	CREB5	ECSIT	FOXO3	HHEX	IRX5	LUZP2
ARID5B	CREBL2	EGR1	FOXP1	HIF1A	ISL2	MAF
ASH2L	CTBP1	EGR2	FOXQ1	HIRIP3	JAZF1	MCTP2
ATF2	CTCFL	EIF5A2	GATA2	HIST1H2BN	JDP2	MECOM
BACH2	CTNNB1	ELF3	GATA3	HIST2H2BE	JRKL	MIXL1
BARHL2	CUX2	ELK3	GATA4	HKR1	KDM4A	MLXIPL
BARX1	CYB5R1	EMX1	GBX1	HLF	KDM4D	MSI2
BARX2	DAB2	EN1	GBX2	HNF1A	KDM4E	MSRA
BATF	DBP	ERG	GCM1	HNF4A	KLF11	MSRB3
BCL11A	DDIT3	ESRRG	GIT2	HOXA4	KLF12	MTHFD1
BCL3	DDX4	ETFB	GOT1	HOXA7	KLF17	MYCL
BHLHE40	DDX43	ETS2	GPD1	HOXA9	KLF18	MYLK
CARF	DLX1	ETV1	GRHL1	HOXB13	KLF2	NANOG
CBFA2T2	DLX2	ETV4	GRHL2	HOXB6	KLF3	NANOGP8
CCDC25	DLX3	ETV5	GRHL3	HOXC10	KLF4	NCALD
CDX1	DLX4	EXO5	GRHPR	HOXD8	KLF6	NEUROG2
CDX2	DLX5	EZR	GTF2A1L	HTATIP2	KLF7	NFATC1
CEBPA	DMRTB1	FEZF2	GTF3A	HUNK	KLF9	NFE2
CEBPB	DMRTC2	FHL2	H2AFY	ING3	KLRG1	NFIX
CEBPD	DNMT1	FIGLA	HAND1	IRF3	LARP1	NFKB1
CELF5	DPRX	FLI1	HCFC2	IRF4	LEF1	NFKB2
CERS2	DUXA	FOSB	HES1	IRF6	LHX2	NFYA

NKX2-5	PPARGC1A	SOX11	TPI1	ZNF140	ZNF439	ZNF697
NKX3-1	PRDM1	SOX15	TPPP	ZNF146	ZNF440	ZNF701
NKX3-2	PRDM10	SOX17	TRIB2	ZNF155	ZNF479	ZNF702P
NKX6-1	PRDM11	SOX2	TRIB3	ZNF157	ZNF490	ZNF705A
NKX6-2	PRDM14	SOX30	TRIP10	ZNF16	ZNF506	ZNF705CP
NMI	PRDM16	SOX4	TULP1	ZNF165	ZNF528	ZNF705D
NNT	PRDX5	SOX5	UBE2V1	ZNF17	ZNF530	ZNF705G
NOBOX	PRKAA1	SOX9	UGP2	ZNF174	ZNF534	ZNF706
NR1H4	PRKAA2	SP110	VENTX	ZNF18	ZNF541	ZNF708
NR2E1	PSMC2	SP6	YWHAZ	ZNF182	ZNF549	ZNF714
NR2F2	PSMD12	SPIC	YY2	ZNF184	ZNF555	ZNF716
NR3C1	RAB14	SSX3	ZBTB11	ZNF19	ZNF557	ZNF727
NR3C2	RAB18	STAT3	ZBTB16	ZNF200	ZNF559	ZNF735
NR4A3	RARB	STAT5A	ZBTB49	ZNF211	ZNF561	ZNF736
NR5A2	RAX2	SUCLG1	ZBTB7B	ZNF214	ZNF569	ZNF766
NR6A1	RBBP9	TAF7	ZCCHC14	ZNF215	ZNF574	ZNF79
NRF1	RELB	TAGLN2	ZEB1	ZNF226	ZNF578	ZNF814
OLIG1	RFX4	TBPL2	ZFHX3	ZNF23	ZNF584	ZNF829
OSR1	RFXANK	TBX2	ZFP3	ZNF230	ZNF586	ZNF830
OSR2	RLF	TBX3	ZFP37	ZNF25	ZNF595	ZNF831
OTX1	RORB	TBX5	ZFP42	ZNF256	ZNF596	ZNF844
OTX2	RUNX1	TCF24	ZFP62	ZNF266	ZNF597	ZNF845
OVOL1	RUNX2	TCF7L1	ZFP64	ZNF280A	ZNF599	ZNF878
OVOL2	RUVBL1	TCF7L2	ZFP90	ZNF284	ZNF606	ZNF880
P4HB	RXRA	TEAD1	ZHX2	ZNF304	ZNF610	ZNF891
PARP1	SALL2	TEAD3	ZHX3	ZNF329	ZNF616	ZNF92
PAX9	SATB1	TFAP2A	ZIC3	ZNF331	ZNF630	ZRSR2
PBX3	SETBP1	TFAP2B	ZIK1	ZNF341	ZNF654	ZSCAN10
PIR	SHOX2	TFAP2D	ZIM2	ZNF343	ZNF668	ZSCAN18
PITX2	SIN3A	TFCP2L1	ZIM3	ZNF350	ZNF669	ZSCAN32
PKM	SMAD5	TFEB	ZKSCAN4	ZNF354A	ZNF674	ZSCAN4
PKNOX2	SMAD6	TGIF2LX	ZKSCAN5	ZNF362	ZNF675	ZSCAN5A
PLAG1	SNAI1	THAP1	ZNF10	ZNF385A	ZNF677	ZSCAN5B
PLAGL1	SNAI3	THRB	ZNF117	ZNF394	ZNF679	ZSCAN5C
POLD2	SND1	TIGD2	ZNF132	ZNF408	ZNF684	
POU5F1B	SOD1	TOPORS	ZNF134	ZNF416	ZNF689	
PPARG	SOX1	TP63	ZNF136	ZNF438	ZNF69	

References

Chebouba, L., Miannay, B., Boughaci, D., and Guziolowski, C. Discriminate the response of acute myeloid leukemia patients to treatment by using proteomics data and answer set programming. *BMC Bioinformatics*, 19(2):15–26, 2018.