Discovering a Causal DAG for genes via the Mappa Mundi algorithm

Robert R. Tucci
tucci@ar-tiste.com

April 7, 2025

Abstract

This paper can be viewed as an application and further refinement of the Mappa Mundi (MM) algorithm, an algorithm for which software and a paper are available at github. The MM algorithm can be used to extract causal DAGs from chronologically ordered data that is either in sentences text or tabular numeric form. In this paper, we describe how to apply it to finding what are called Gene Regulatory Networks (GRN), Autoregulon Nets and Network Motifs in the Genomics and Systems Biology literature.

1 Introduction

This paper can be viewed as an application and further refinement of the Mappa Mundi (MM) algorithm given in Ref.[4]. In this paper, we describe how to apply it to finding what are called Gene Regulatory Networks (GRN), autoregulon (AR) nets and Network Motifs in the Genomics and Systems Biology literature (Ref.[1]).

The MM algorithm was first proposed in Ref.[4] for DAG Extraction From Text (DEFT). In Ref.[4], it was used to extract causal DAGs from 3 P.G. Wodehouse short stories and the scripts of 3 PiXar movies. In general, the MM algorithm can extract causal DAGs from 2 or more text files, as long as each of those text files recounts actions in chronological order. So it will work with time stamped lab notebooks or medical records, but it won't work with textbooks or fiction with time travel or flashback shenanigans.

After Ref.[4], the MM algorithm was applied in Ref.[3] to extracting causal DAGs from FitBit times series data.

In this paper, we use the MM algorithm to extract DAGs from time series data for concentrations of gene expressions and transcription factors. The DAGs we obtain are called GRN.[1] GRN are a special case of AR networks.[2] AR nets are a special case of Dynamical systems (DS).[3]

2 Comparing 2 TS Records

We will use the term **time series (TS) record** to refer to a data file such as a spreadsheet with the first column giving time increasing downwards and additional columns giving the values of q_i (a state variable) and $\dot{q}_i$ (time derivative of q_i) for $i = 1, 2, \ldots N$ at the time indicated by the first column. Any q_i or $\dot{q}_i$ is called a **state variable**. The multidimensional space $(q_i)_{i=0}^N$ is called **configuration space**. The multidimensional space $(q_i, \dot{q}_i)_{i=0}^N$ is called **phase space**. A two dimensional space $(q_i, \dot{q}_i)$ for any i is called a **phase plane**.

A TS record may contain initially only a column for time and columns for configuration space, if the change in time between rows is small. In that case we can subtract two consecutive q_i readings and divide by the difference in time to obtain the $\dot{q}_i$ columns.

In this paper, each q_i represents either a **transcription factor (TF)** concentration, or a **gene expression (GE)** concentration.[4]

Suppose x and y are any two q_i. For $\xi \in \mathcal{A}_{xy} = \{x \to y, y \to x\}$, let
n^ξ_{acc}: number of arrows accepted, initially zero
n^ξ_{rej}: number of arrows rejected, initially zero
$N^\xi = n^\xi_{acc} + n^\xi_{rej}$: number of arrows detected
$p^\xi_{acc} = \frac{n^\xi_{acc}}{n^\xi_{acc}+n^\xi_{rej}}$: probability of causal arrow, initially zero.
N^*: threshold value for N^ξ
p^*_{acc}: threshold value for p^ξ_{acc}

The MM algorithm for finding an AR net from 2 TS records, consists of the following steps:

1. **Compare 2 TS records and score arrows**

   Consider Fig.1. In that figure, suppose the two ends of bridge a are approximately equal: $\mathcal{X}_1(t_a) \approx \mathcal{X}_2(t'_a)$ and the two ends of bridge b are approximately equal too: $\mathcal{Y}_1(t_b) \approx \mathcal{Y}_2(t'_b)$. [5]

[1] GRN are discussed in the chapter entitled "Gene Regulatory Networks" of my free book Ref.[2].

[2] AR nets are discussed in the chapter entitled "Autoregulon Networks (Network Motifs)" of my free book Ref.[2].

[3] DS are discussed in the chapter entitled "Dynamical Systems" of my free book Ref.[2].

[4] The terms "transcription factor" and "gene expression" are both defined in the AR networks chapter of my free book Ref.[2]

[5] By $\mathcal{X} \approx \mathcal{Y}$ we mean that both of these vectors are inside the same small bin or open ball of size given by a pre-specified precision.

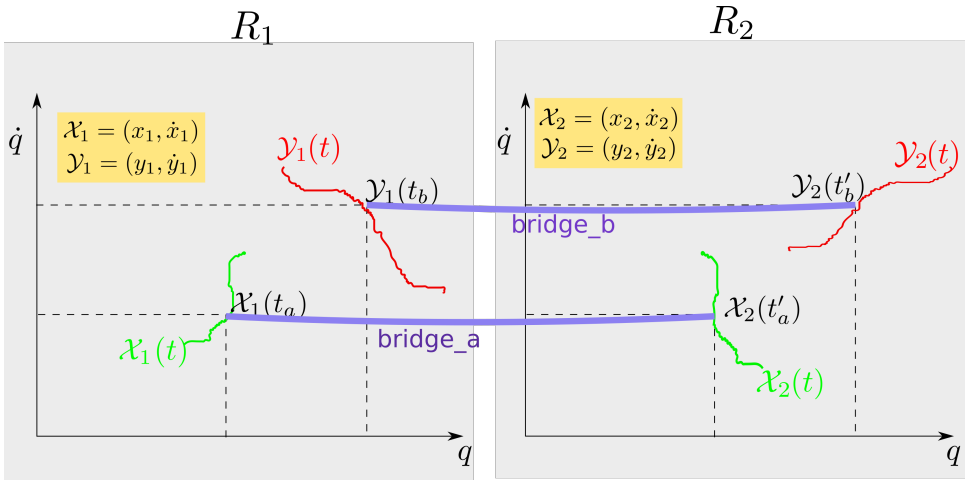


Figure 1: Causal bridges a and b spanning phase planes for TS records R_1 and R_2.

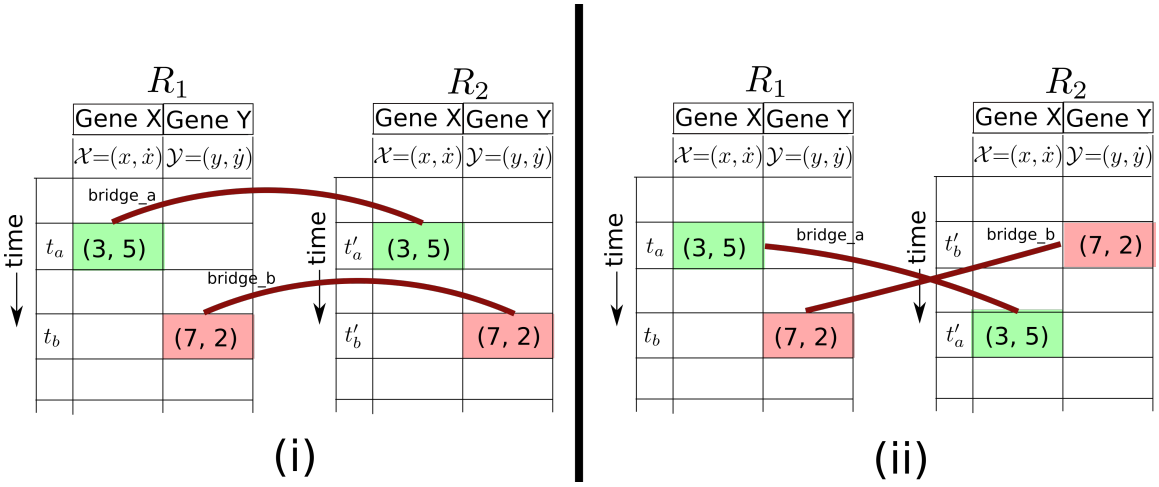


Figure 2: TS Records R_1 and R_2 portrayed as dataframes instead of phase planes as in Fig.1. In case (i), causal bridges are parallel, whereas in case (ii), they cross. X might be a TF instead of a gene. Ditto for Y.

At the very least, one must store (unless they are the default value zero) the current values of n^{ξ}_{acc} and n^{ξ}_{rec} for $\xi \in \mathcal{A}_{xy}$, where x and y are any two q_i.

- Suppose $t_a < t_b$ and $t'_a < t'_b$. Hence, bridges are parallel in time as in Fig.2 (i). Then[6]

[6] $x++$ means add 1 to x

$$\begin{cases} n_{acc}^{x\to y}++ \\ N^{x\to y}++ \\ \text{Add } (t_a, t'_a, \mathcal{X}(t_a)) \text{ to a set } \mathbb{X}(R_1, R_2). \\ \text{Add } (t_b, t'_b, \mathcal{Y}(t_b)) \text{ to a set } \mathbb{Y}(R_1, R_2). \end{cases} \tag{1}$$

- Suppose $t_a > t_b$ and $t'_a > t'_b$. Hence, bridges are parallel in time. Then

$$\begin{cases} n_{acc}^{y\to x}++ \\ N^{y\to x}++ \\ \text{Add } (t_a, t'_a, \mathcal{X}(t_a)) \text{ to a set } \mathbb{X}(R_1, R_2). \\ \text{Add } (t_b, t'_b, \mathcal{Y}(t_b)) \text{ to a set } \mathbb{Y}(R_1, R_2). \end{cases} \tag{2}$$

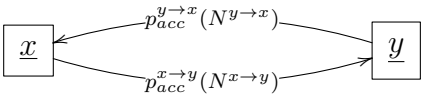
- Suppose $t_a < t_b$ and $t'_a > t'_b$. Hence, bridges cross in time as in Fig.2 (ii). Then

$$\begin{cases} n_{rej}^{x\to y}++ \\ N^{x\to y}++ \end{cases} \tag{3}$$

- Suppose $t_a > t_b$ and $t'_a < t'_b$. Hence, bridges cross in time. Then

$$\begin{cases} n_{rej}^{y\to x}++ \\ N^{y\to x}++ \end{cases} \tag{4}$$

2. **Draw AR net**



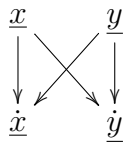
(5)

If x and y are any two q_i, draw an arrow from autoregulon $\boxed{\underline{x}}$ to autoregulon $\boxed{\underline{y}}$ iff both $p_{acc}^{x\to y} > p^*$ and $N^{x\to y} > N^*$ are true.

Likewise, draw an arrow from autoregulon $\boxed{\underline{y}}$ to autoregulon $\boxed{\underline{x}}$ iff both $p_{acc}^{y\to x} > p^*$ and $N^{y\to x} > N^*$ are true.

When drawing an arrow, write the values of p_{acc}^{ξ} and N^{ξ} over the arrow, where $\xi \in \mathcal{A}_{xy}$. See Eq.(5) where this is done with variables. Do it with the values of those variables instead.

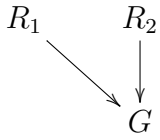
At first glance, Eq.(5) doesn't look like a DAG, because it has a cycle and DAGs are, by definition, acyclic. But Eq.(5) does indeed represent a DAG because, as explained in the AR net chapter of Ref.[2], Eq.(5) represents this net:



(6)

which is acyclic.

If R_1 and R_2 are two TS records and G is the **AR net** obtained by following the MM algorithm presented above, then one can represent that MM algorithm diagrammatically by the **meta net**



(7)

In the next section, we will try to define the merging of more than two records into a single AR net.

3 Comparing More Than 2 TS Records

3.1 Merging two or more AR nets into one AR net

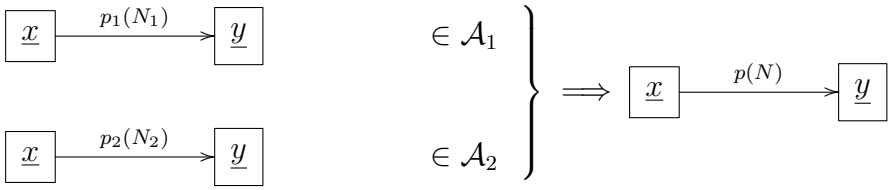
Suppose G_1 and G_2 are two AR nets that we wish to merge. Let $\mathcal{A}_i$ be the set of arrows of AR net G_i for $i = 1, 2$.

Any autoregulon $\boxed{\underline{x}}$ in an AR net G stores as "cargo" a dictionary $\mathcal{D}_X$ mapping (R_1, R_2) to $\mathbb{X}(R_1, R_2)$, where the $\mathbb{X}(R_1, R_2)$ are of the form

$$\{(t_1, t_1', \mathcal{X}(t_1)), (t_2, t_2', \mathcal{X}(t_2)), (t_3, t_3', \mathcal{X}(t_3)), \ldots\} \tag{8}$$

If AR nodes $\boxed{\underline{x}}$ and $\boxed{\underline{x}'}$ are merged, the merged node should store both $\mathcal{D}_X$ and $\mathcal{D}_{X'}$.

- merging doubly overlapping arrows (overlap in head and tail)



(9)

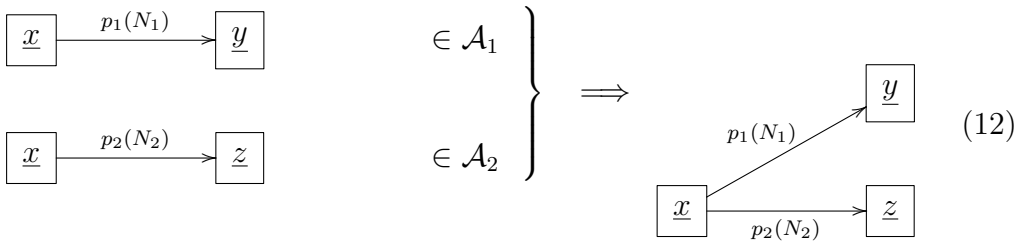
where

$$N = N_1 + N_2 \tag{10}$$

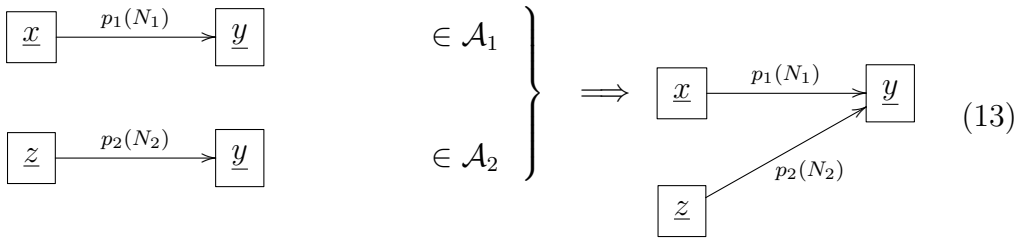
and

$$p = \frac{\sum_{i=1}^{2} n_{acc,i}}{\sum_{i=1}^{2} (n_{acc,i} + n_{rej,i})} = \frac{p_1 N_1 + p_2 N_2}{N_1 + N_2} = p_1 \frac{N_1}{N} + p_2 \frac{N_2}{N} \tag{11}$$

- merging singly overlapping arrows (overlap in head or tail but not both)

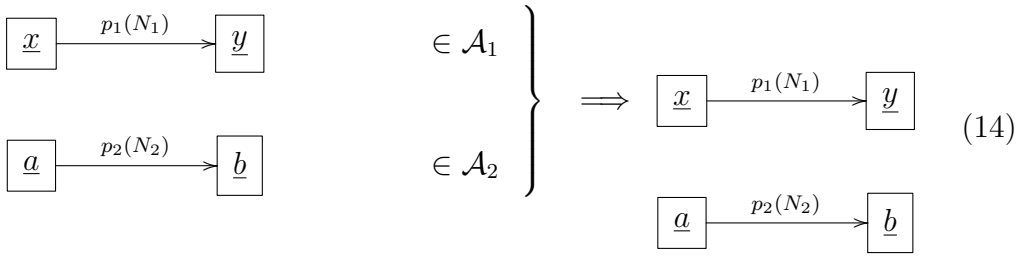


(12)



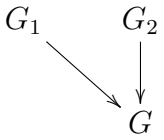
(13)

- merging non-overlapping arrows



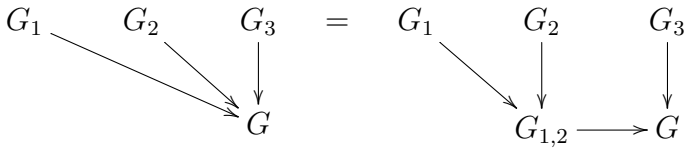
(14)

If G_1 and G_2 are two AR nets and G is the AR net obtained by following the AR net merging algorithm presented above, then one can represent that AR net merging algorithm diagrammatically by



(15)

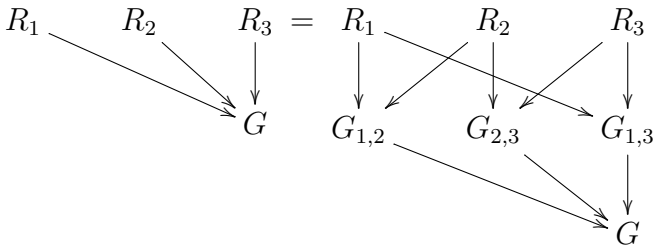
Once one defines the meaning of meta net Eq.(15), one can use it as a building block to define the merging of more than 2 AR nets into a single AR net. For example, we can define the merging of 3 AR nets by the meta net



(16)

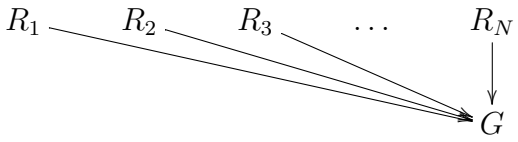
3.2 Extracting single AR net from more than 2 TS records

Once we define by Eq.(16) the merging of 3 AR nets into a single AR net, one can define the extraction of a single AR net from 3 TS records, as follows.



(17)

Suppose we have N records $R_1, R_2, \ldots, R_N$. Eq.(17) can be generalized to define

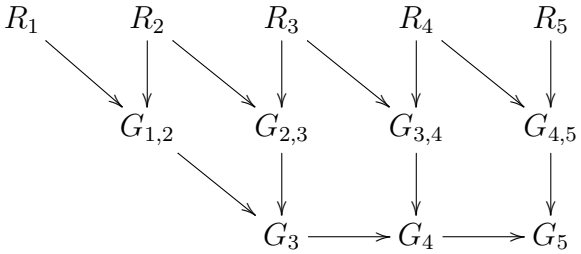


(18)

The generalization extracts an AR net from every pair of records $\{R_i, R_j\}$ where $1 \leq i < j \leq N$, and merges all the extracted AR nets into a single one.

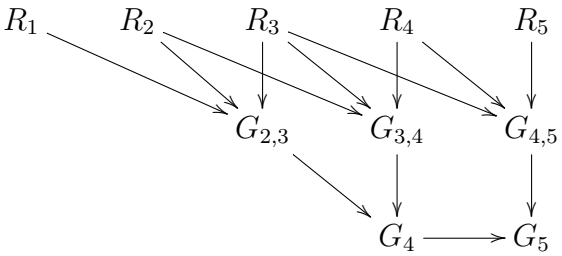
Eqs. (17) and (18) involve the comparison of $\binom{N}{2}$ record pairs. For large N, performing that many record comparisons is untenable. So we need a less laborious algorithm (even if less informative too) for extracting a single AR net from N records.

One family of such extraction algorithms is the following. Order the records R_i so that i denotes the time at which they are presented to us. If R_i is the current record, one can extract an AR net from R_i and the previous record R_{i-1} as follows



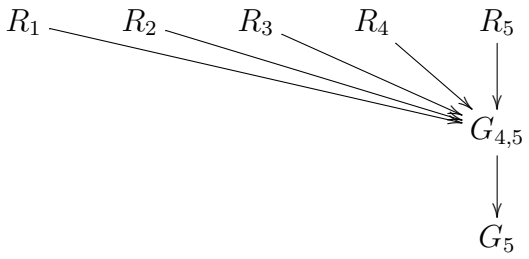
(19)

Alternatively, if one is willing to do more work, one can extract an AR net from R_i and the previous 2 records R_{i-1} and R_{i-2}, as follows



(20)

And so on. If we assume perfect memory of all past records, then we get the following, which is the same as Eq.(18).



(21)

References

[1] Uri Alon. *An introduction to systems biology: design principles of biological circuits*. Chapman and Hall/CRC, 2019.

[2] Robert R. Tucci. Bayesuvius (free book). `https://github.com/rrtucci/Bayesuvius`.

[3] Robert R. Tucci. CausalFitbit (software and paper). `https://github.com/rrtucci/CausalFitbit`.

[4] Robert R. Tucci. Mappa Mundi (software and paper). `https://github.com/rrtucci/mappa_mundi`.