JFP **35**, e13, 48 pages, 2025.

doi:10.1017/S0956796825000085

THEORETICAL PEARL

Point-free calculational proofs and program derivation in linear algebra using a graphical syntax

JÚLIA DE ARAÚJO MOTA
Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil

JOÃO A. PAIXÃO
Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil
(*e-mail:* jpaixao@dcc.ufrj.br)

LUCAS RUFINO MARTELOTTE
Instituto de Matemática Pura e Aplicada, Rio de Janeiro, Brazil

Abstract

This theoretical pearl shows how a graphical, relational, point-free, and calculational approach to linear algebra, known as graphical linear algebra, can be used to reason not only about matrices (and matrix algebra, as can be found in the literature) but also vector spaces and more generally linear relations. Linear algebra is usually seen as the study of vector spaces and linear transformations. However, to reason effectively with subspaces in a point-free and calculational manner, both can be generalized to an unifying concept: linear relations, much like relational algebra. While the semantics is relational, the syntax is graphical and uses string diagrams, 2-dimensional formal diagrams, which represent the linear relations. Most importantly, in a number of cases, the relational semantics allows algorithms and properties to be derived calculationally instead of just verified. Our approach is to proceed primarily by examples which involve finding inverses, switching from an implicit basis to an explicit basis (solving a homogeneous linear system), exploring both the exchange lemma and the Zassenhaus' algorithm.

1 Introduction

Besides calculus, linear algebra has always been present as a valuable mathematical tool for engineering and physics. Seemingly unrelated technologies are inherently linear algebraic: e.g., the Google PageRank concept is an eigenvector, and numerical linear algebra algorithms are at the core of convex optimization. Further afield, linear algebra is the backbone of much of modern mathematical physics; e.g., non-linear differential equations are

solved by iterating linear systems. It is difficult to overstate the sheer practicality of linear algebra theory and its influence in shaping modern science.

On the other hand, in computer science, the main mathematical tools have been more discrete rather than quantitative with logic, functions, relations, graphs, and combinatorics. Lately there is evidence of a shift towards integrating linear algebra into computing, with examples in machine learning, process semantics, quantum programming, natural language semantics, and data mining. This trend suggests a growing interest in applying linear algebra techniques to computing, driven by diverse research areas. For instance, many techniques in natural language processing, such as word2vec, aim to represent data as vectors in a suitable vector space; semantic relationship between words is thus captured by the usual euclidean distance formula. Other techniques can be cited, such as co-occurrence matrices which quantify how often words appear together in a body of text and works by Baroni & Zamparelli (2010) on adjective-noun composition, where nouns are represented by vectors and adjectives by matrices. Matrix multiplication lies at the heart of neural networks. The trendy attention mechanism of the Transformer Architecture (Vaswani *et al*., 2023), which enabled much of modern generative models such as ChatGPT and Stable Diffusion, is computed using dot products. Quantum computing and non-deterministic programming in general are also making use of linear algebra techniques, as for example in the paper by Sernadas *et al*. (2008) a method for deciding correctness of probabilistic programs is devised.

It is well known among the purist computer scientists that categorical (abstracting the data type), index-free, calculational reasoning and program derivation are good practices. Thus, it is natural to inquire whether linear algebra, with its recently popularized status as a programming tool, can also benefit from such practices. This can be found in the literature, see e.g. Gonthier (2011), Mac Lane (2013). As our main example, Macedo & Oliveira (2013) achieve point-free calculational reasoning in linear algebra by presenting the fundamental laws of matrix algebra as rewriting rules.

Relations and relational reasoning have already proven to be very important in programming (Bird & De Moor, 1996). As a natural extension of the ideas of Macedo and Oliveira, the aim of this paper is to showcase an approach for linear algebra where, instead of matrices, the main object of study is that of a *linear relation* (Arens, 1961; Mac Lane, 1961; Coddington, 1973; Cross, 1998). It generalizes the notion of a linear transformation in the same sense that ordinary relations generalize the notion of a function. We thereby quote from algebra of programming: "Our framework is relational because we need a degree of freedom in specification and proof that a calculus of functions alone would not provide." As we shall see in later sections, there is much to gain by also introducing the relational paradigm into linear algebra.

Our proofs make use of *string diagrams*. They are popular objects among category theorists, and are essentially formally defined drawings with certain rules for combining them (Selinger, 2011; Baez & Erbele, 2015). It is well known that string diagrams are a good tool for point-free reasoning and type-checking. Also, their graphical 2d-syntax allows one to omit parentheses around the two ways of composing relations, much like the usual 1d-syntax dismisses parentheses when functions are composed. Their use in linear algebra has recently been explored by Zanasi (2015), who developed what is called graphical linear algebra (GLA).

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

In Section 2, differences among the three notations used in this paper are discussed (which will be called pointwise, classical, and graphical). Tables 3 and 4, referred to as the Dictionary of Notations, show how to translate one into another. Additionally, this section exposes the main axioms and theorems that will be employed throughout this text, presenting their respective representations in the three notations.

In Sections 3 and 4, our approach is to proceed primarily by example. In Section 3, we construct proofs through verification. Two algorithms are explored: an algorithm for finding fundamental bases for the subspaces of a matrix (Beezer, 2014) and the Zassenhaus algorithm (Fischer, 2012). Section 4 shows two examples of program derivation and one last example where a *property* is derived instead. Two additional examples will be presented: calculating the right inverse of a wide triangular matrix and switching from implicit to explicit basis (solving a homogeneous linear system). The pseudocode for each algorithm (which will be a systematic translation of the proof steps) will be provided. Lastly, a proof of a result concerning the duality between Gaussian elimination and the exchange lemma will also be presented (Barańczuk & Szydło, 2021).

1.1 The category of linear relations

Point-free, calculational approaches for program derivation in linear algebra have already been explored in a previous work by Macedo & Oliveira (2013). The authors present the fundamental laws of matrix algebra and show how blocked matrix notation permits point-free equational reasoning and algorithm derivation in matrix algebra. Their approach makes use of the rich biproduct structure of $\mathsf{FinVect}_{\mathsf{k}}$ (where k stands for a field), a category whose arrows are linear transformations. Below is the full definition. In this paper, for simplicity, we will always consider $\mathsf{k} = \mathbb{R}$.

Definition 1 ($\mathsf{FinVect}_{\mathbb{R}}$)**.** *The category* $\mathsf{FinVect}_{\mathbb{R}}$ *is such that*

- *the objects are finite-dimensional vector spaces over* $\mathbb{R}$*,*
- *the arrows are linear transformations between them,*
- *composition stands for composition of functions,*
- *the identity arrow is the identity map.*

Despite its efficacy, subspaces appear only at the object level in $\mathsf{FinVect}_{\mathbb{R}}$, whereas arrows exclusively represent linear transformations. This setup poses challenges for point-free reasoning (without referencing objects) about subspaces. It is well known that when one wants point-free reasoning about relations, such as in relational algebra, it is a good idea to use the category of relations (Rel), instead of the category of functions (Set) (Bird & De Moor, 1996). In a similar vein, one may naturally inquire about the analogy of Rel in regards to $\mathsf{FinVect}_{\mathbb{R}}$. One possible solution is the category $\mathsf{LinRel}_{\mathbb{R}}$ (Zanasi, 2015; Baez & Erbele, 2015), in which the arrows represent linear relations (a generalization of both linear transformations and subspaces).

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Definition 2 ($\mathsf{LinRel}_{\mathbb{R}}$)**.** *Given vector spaces X and Y over $\mathbb{R}$, a linear relation R between them is a subspace of the product space $X \times Y$. There are two main ways of composing these objects, called relational composition and cartesian product.*

- ***Relational composition:*** *Let $R \subseteq X \times Z$ and $S \subseteq Z \times Y$ be two linear relations. Define their composition $SR \subseteq X \times Y$ as*

$$SR := \{(x, y) \mid \exists z \text{ such that } (x, z) \in R, (z, y) \in S\}.$$

.

- ***Cartesian product:*** *Let $R \subseteq A \times B$ and $S \subseteq X \times Y$ be two linear relations. Define their cartesian product $R \times S \subseteq (A \times X) \times (B \times Y)$ as*

$$R \times S := \{((a, x), (b, y)) \mid (a, b) \in R, (x, y) \in S\}.$$

The category $\mathsf{LinRel}_{\mathbb{R}}$ *is such that*

- *the objects are finite-dimensional vector spaces over $\mathbb{R}$,*
- *the arrows are linear relations between them,*
- *composition stands for relational composition,*
- *the identity arrow of an object X is the identity relation $\{(x, x) \mid x \in X\} \subseteq X \times X$.*

Definition 3 (Opposite)**.** *Given a relation $R \subseteq X \times Y$, we define its opposite $R^o \subseteq Y \times X$ as*

$$R^o := \{(y, x) \mid (x, y) \in R\}.$$

Linear transformations and vector spaces can be thought of as special cases of linear relations. The representation of a linear transformation $T : X \to Y$ as a linear relation is its graph $\{(x, Tx) \mid x \in X\}$. A subspace $X \subseteq Y$ can be thought of as the relation $\{*\} \times X \subseteq \{*\} \times Y$ where $\{*\}$ is the zero-dimensional space. When restricted to linear transformations, relational composition, and cartesian product yield the usual composition and cartesian product of linear transformations. Composing a linear subspace with a linear transformation corresponds to applying the transformation to the subspace. For simplicity, we will make no distinction between these objects and their representation as linear relations.

Example 1 (Image)**.** *Given a linear transformation $A : X \to Y$, we can represent the image of A as the composition*

$$\begin{aligned} AX &= \{(*, x) \mid \exists z \text{ such that } (*, z) \in X, (z, x) \in A\} \\ &= \{(*, x) \mid \exists z \in X \text{ such that } Az = x\} \\ &= \{(*, x) \mid x \in Im(A)\} \\ &= Im(A). \end{aligned}$$

We may extend the notion of blocked matrices to linear relations.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Definition 4 (Blocked relations)**.**

$$\begin{bmatrix} R & S \end{bmatrix} := \{((x,y), z+w) \mid (x,z) \in R, (y,w) \in S\}$$

$$\begin{bmatrix} R \\ S \end{bmatrix} := \{(x,(y,z)) \mid (x,y) \in R, (x,z) \in S\},$$

$$R + S := \{(x, y+z) \mid (x,y) \in R, (x,z) \in S\}.$$

Note that when U and V are subspaces, the operation $(U,V) \mapsto U+V$ corresponds to the usual sum of linear subspaces, i.e. $U+V = \{v+w \mid v \in U, w \in V\}$. In this case, we also have the identities

$$\begin{bmatrix} U & V \end{bmatrix} = U+V, \quad \begin{bmatrix} U \\ V \end{bmatrix} = U \times V.$$

1.2 Relational algebra unifies matrix and subspace laws

All matrix algebra rules from Macedo & Oliveira (2013) still hold when the variables are regarded as linear relations. These rules are summarized in Figure 1 from Santos & Oliveira (2020). As an example, consider the three rules below:

$$\text{Associativity:} \quad A(BC) = (AB)C; \tag{1.1}$$

$$\text{Absorption:} \quad \begin{bmatrix} A & B \end{bmatrix}(C \times D) = \begin{bmatrix} AC & BD \end{bmatrix}; \tag{1.2}$$

$$\text{Divide and conquer:} \quad \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} C \\ D \end{bmatrix} = AC + BD. \tag{1.3}$$

Suppose that A, B, and C now stand for linear relations. These equations are true given the definitions of relational composition and cartesian product. The benefit from this change of perspective comes from the fact that many laws concerning matrices and subspaces become special cases of these relational laws. First, the original rules can be readily recovered by restricting A, B, and C to matrices. Now, if we let C be the subspace corresponding to the domain of B, then composing with C in (1.1) is simply taking the image. So, as it turns out, the following rule is just a special case of the associativity of relational composition:

$$A(Im(B)) = Im(AB). \tag{1.4}$$

As another example, letting C and D in (1.3) be the domains of A and B respectively, we get the following common rule about the image of blocked matrices:

$$Im(\begin{bmatrix} A & B \end{bmatrix}) = Im(A) + Im(B). \tag{1.5}$$

Other common rules involving matrices and subspaces can also be attained via simple instantiation. This flexibility allows for calculational proofs in which both concepts are mixed together. A good example is the Zassenhaus algorithm, a method to find both the intersection and sum of two subspaces given their bases. As a proof of concept, this algorithm is verified in Section 3.2 in a point-free calculational manner.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

$$C\begin{bmatrix}A & B\end{bmatrix} = \begin{bmatrix}CA & CB\end{bmatrix} \qquad \begin{bmatrix}A & B\end{bmatrix}C = \begin{bmatrix}AC & BC\end{bmatrix}$$

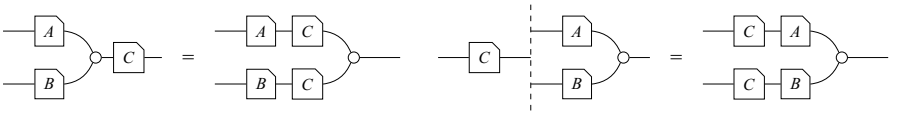


Table 1: Type checking in GLA. The wires of the diagram on the right do not connect.

The benefits are not restricted to verification proofs. It is well known that relational algebra is effective for point-free program derivation (Bird & De Moor, 1996). Thus, it is natural to expect that linear relations will play a similar role in regard to linear algebra. We give two examples of program derivation using relational algebra in Section 4.

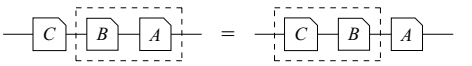
1.3 Problems with syntax

As noted before, blocked matrix notation is one of the key ingredients that allows an equational presentation of the laws of matrix algebra. However, the downside of such a choice is the loss of type information. For instance, many linear algebra students would probably need to think about the dimensions of A, B, and C to decide which of the following two equations is correct.

$$C\begin{bmatrix}A & B\end{bmatrix} = \begin{bmatrix}CA & CB\end{bmatrix} \text{ or } \begin{bmatrix}A & B\end{bmatrix}C = \begin{bmatrix}AC & BC\end{bmatrix}? \tag{1.6}$$

One way to solve this problem is through the use of string diagrams. As noted by Hinze & Marsden (2023), these are well-known objects in the category theory community able to allow equational reasoning without loss of type information. In a recent work, Paixão *et al*. (2022) make use of a graphical syntax for linear algebra, which uses string diagrams, known as GLA to present the laws of linear relations in an equational manner. As shown in Table 1 (the reader is not expected to fully understand the diagrams yet), type checking (1.6) in GLA amounts to checking whether the wires connect.

In our case, string diagrams also implicitly handle some non-trivial rules in the usual syntax, just as is commonly done with associativity by ignoring parentheses. As a simple example, below is the associativity law in graphical syntax, with the dotted lines simulating parenthesization. Table 2 exemplifies how other 6 rules are also handled implicitly by the graphical language.

Usual syntax	Graphical syntax
$A(BC) = (AB)C$	

In Section 3, we employ GLA to construct proofs through verification. Two algorithms are explored: an algorithm for finding fundamental bases for the subspaces of a matrix (Beezer, 2014) and the Zassenhaus algorithm (Fischer, 2012).

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

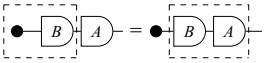
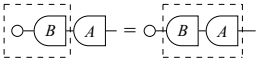
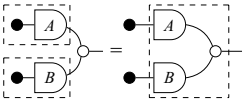
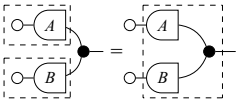
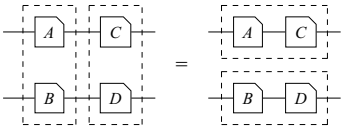
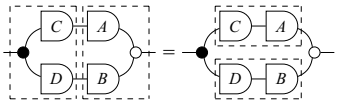
Usual syntax	Graphical syntax
$A(Im(B)) = Im(AB)$	
$A^o(Ker(B)) = Ker(BA)$	
$Im(A) + Im(B) = Im(\begin{bmatrix} A & B \end{bmatrix})$	
$Ker(A) \cap Ker(B) = Ker(\begin{bmatrix} A \\ B \end{bmatrix})$	
$(C \times D)(A \times B) = (CA) \times (DB)$	
$\begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} C \\ D \end{bmatrix} = AC + BD$	

Table 2: Implicit rules. The dotted lines represent parenthesization.

In Section 4, we utilize GLA for program derivation. Two additional examples will be presented: calculating the right inverse of a wide triangular matrix and switching from implicit to explicit basis (solving a homogeneous linear system). The pseudocode for each algorithm (which will be a systematic translation of the proof steps) will be provided. Lastly, a proof of a result concerning the duality between Gaussian elimination and the exchange lemma will also be presented (Barańczuk & Szydło, 2021).

2 Graphical linear algebra

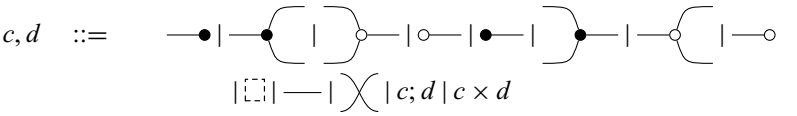
Three notations are used in this paper. The *pointwise notation* concerns fundamental mathematical explanations with notation of sets. The *classical notation* refers to terms commonly used in linear algebra such as image, kernel and block matrices. The *graphical notation* refers to the 2-dimensional diagrammatical notation used in GLA and will be the main focus of this section. We will briefly introduce it to the reader and show (Tables 3 and 4) how it can be translated into the other two.

Proofs in this context will make use of the diagrammatic language, in which the main ingredients are mathematical objects called *string diagrams*. This section explains how diagrams are constructed and outlines the structures of how to manipulate and reason with them. These diagrams are an instance of a particular class of string diagrams, which are well known to characterize the arrows of free strict monoidal categories (Selinger, 2010). They are, therefore, rigorous mathematical objects.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

2.1 Graphical syntax

This section will present the equational theory of interacting Hopf algebras (Zanasi, 2015) from which graphical linear algebra is based upon (Baez & Erbele, 2015; Bonchi *et al.*, 2017). It has been applied (and adapted) in a series of papers (Bonchi *et al.*, 2014, 2015, 2017, 2019) to model signal flow graphs, Petri nets and non-passive electrical circuits. In this paper, however, the aim is to elucidate its status as a notation for linear algebra.



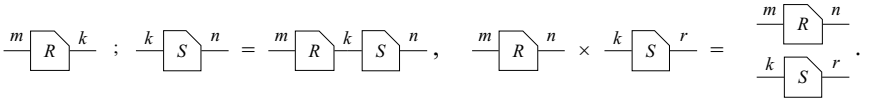
(2.1)

The graphical syntax is built from a series of initial diagrams (or symbols) called *generators*, shown in (2.1); diagrams can be combined according to two rules (; and ×) to form bigger diagrams. Semantically, each diagram canonically represents a linear relation (this will be better explained in Section 2.2), and the rules ; and × in diagrammatic notation correspond, respectively, to relational composition and cartesian product, as in Definition 2.

We use 

to denote a generic diagram with m wires on the left and n wires on the right (the chamfered edge on the right is necessary because eventually we will need to start "flipping diagrams horizontally", so it is important to retain directional information).

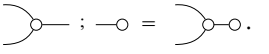
Definition 5 (Composing diagrams)**.** *The operations* ; *and* × *correspond to, respectively, attaching diagrams horizontally and vertically.*



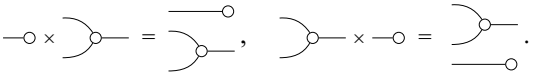
Notice that, similar to relational composition, to compute 

, R and S have to be compatible: the number of right-wires of R must be the same as the number of left-wires of S.

Example 2. *The composition*  *is ill-typed because the diagrams are incompatible (the first has one right-wire, whereas the second has two left-wires). The composition below, however, is valid.*

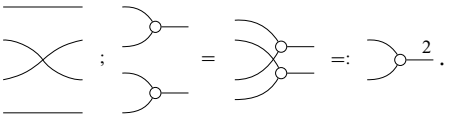


Example 3. *The operation* × *corresponds to stacking diagrams vertically. It can always be performed regardless of compatibility conditions.*



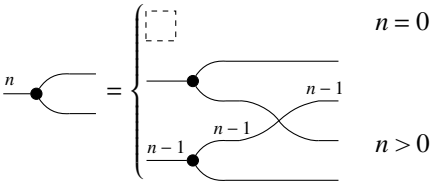
https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Example 4. *Here is a more complicated example of composition.*

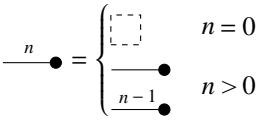


The last example above showcases a method of combining two  diagrams into a bigger one which we named  . This process can be continued inductively for any number of steps, forming what we call a *generalized generator*, written  , for any $n \in \mathbb{N}$. There are analogous constructions for each generator, as defined below.

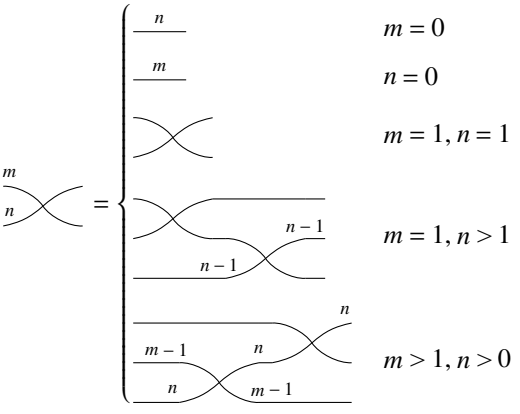
Definition 6 (Generalized generators)**.**



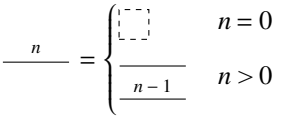
(2.2)



(2.3)

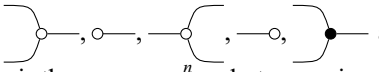


(2.4)

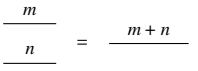


(2.5)

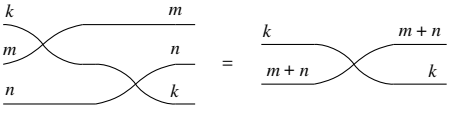
https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

The constructions for  , and  are what one might expect. For example,  is the same as  , but swapping the colors from black to white in the inductive definition. Similarly,  is the same as  but flipping the diagrams horizontally. This sort of double-symmetry (color-swapping and horizontal-flipping) is recurrent in GLA. For instance, the Theorem below is stated only for  and  , but the corresponding properties for the other generators can be readily guessed correctly.

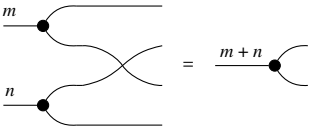
Theorem 1. *The types defined in Definition 6 are closed by the following operations.*



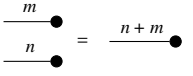
(2.6)



(2.7)



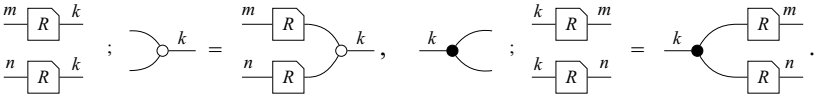
(2.8)



(2.9)

These higher-order structures help simplify notation, especially when defining certain types of diagrams inductively, like matrices (see Definition 9).

Example 5. *Here's an example of how generalized generators compose with other diagrams.*



By abuse of notation, we often omit the numbering on the wires when they are irrelevant or can be inferred from the given data.

2.1.1 Symmetric strict monoidal categories

Raw terms are quotiented with respect to the laws of symmetric strict monoidal (SSM) categories, summarized in Figure 1. We omit the (well-known) details (Selinger, 2010) here and mention only that this amounts to eschewing the need for "dotted line boxes" and ensuring that diagrams with the same topological connectivity are equated.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

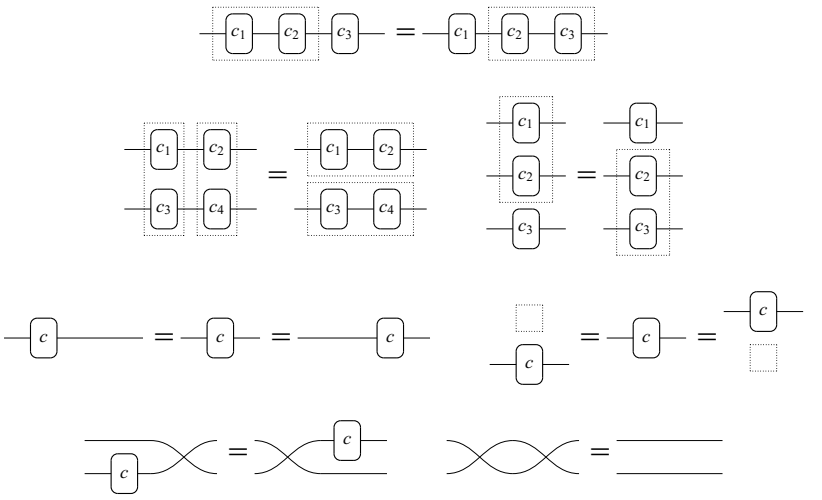
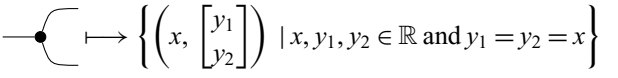


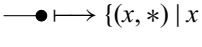
Fig. 1: Laws of symmetric strict monoidal (SSM) categories. The numbering on the wires is omitted for readability.

2.2 Translating graphical notation to pointwise notation

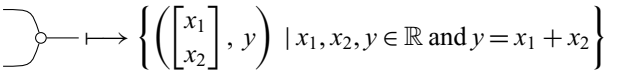
In the graphical notation, each diagram semantically corresponds to a linear relation. The translation of diagrams to pointwise notation is compositional and translates their semantic meaning – the meaning of a compound diagram is calculated from the meanings of its sub-diagrams. Here there is a connection with relational algebra, the two operations of diagram composition are mapped to the standard ways of composing relations: relational composition and cartesian product. In fact, the translation of diagrams to pointwise notation can be viewed as a *functor* from the category of diagrams to the category of linear relations over $\mathbb{R}$. Given that all diagrams are built from composing smaller diagrams, it suffices to show how to translate the generators.



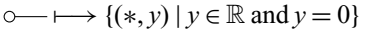
$$\longmapsto \left\{ \left(x, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) \mid x, y_1, y_2 \in \mathbb{R} \text{ and } y_1 = y_2 = x \right\}$$



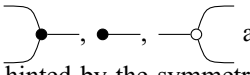
$$\longmapsto \{(x, *) \mid x \in \mathbb{R}\}$$



$$\longmapsto \left\{ \left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, y \right) \mid x_1, x_2, y \in \mathbb{R} \text{ and } y = x_1 + x_2 \right\}$$



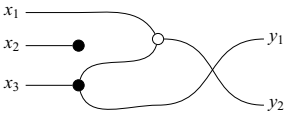
$$\longmapsto \{(*, y) \mid y \in \mathbb{R} \text{ and } y = 0\}$$

The generators  and  are, respectively, the opposite of the relations above, as hinted by the symmetric graphical syntax. Its interpretation in pointwise notation is therefore trivial.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Intuitively, the *black ball* in diagrams refers to copying and discarding – indeed  is *copy* and is *discard*. Meanwhile, the *white ball* refers to addition in $\mathbb{R}$ – indeed  is *add* and  is *zero*.

Example 6. *Here's an example of how to translate a more complicated diagram. The intuitive way to think about diagrams is like a series of logical gates, where the input starts on the left and flows to the right via the wires. In the image below we include some variable names x_1, x_2, x_3, y_1, y_2 (which are not wire numberings) to aid comprehension.*



By visualizing the x_i's flowing to the right, we can see x_2 is discarded and x_3 is copied. One of the copies is passed to y_1 while the other is added with x_1 to form y_2. So $y_1 = x_3$ and $y_2 = x_1 + x_3$. Thus, written in pointwise notation this diagram corresponds to the relation

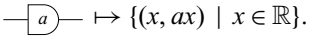
$$\left\{ \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) \in \mathbb{R}^3 \times \mathbb{R}^2 \mid y_1 = x_3 \text{ and } y_2 = x_1 + x_3 \right\}.$$

For the purpose of simplifying the notation, sometimes in this paper, the inclusion "$\in \mathbb{R}^3 \times \mathbb{R}^2$" will be omitted and diagrams such as the one presented above will be written in pointwise notation as follows.

$$\left\{ \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) \mid y_1 = x_3 \text{ and } y_2 = x_1 + x_3 \right\}.$$

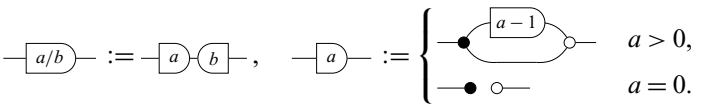
2.3 Scalars, matrices and other constructions

We've seen that linear transformation (or linear maps) can be thought of as special cases of linear relations. This section explains some basic types of maps which will be used throughout the paper. The most basic linear map is scalar multiplication: given $a \in \mathbb{R}$, one can define the map $[a] : \mathbb{R} \to \mathbb{R}$ given by $x \mapsto ax$. In diagrammatic notation, we write it as the 1×1 curved diagram , *always* with a lowercase letter. Semantically



If we restrict ourselves to scalars in $\mathbb{Q}$, any scalar multiplication can be built inductively from the generators. Below we define diagrammatically the maps $[q]$ for $0 \leq q \in \mathbb{Q}$.

Definition 7 (Scalars in $\mathbb{Q}$)**.** *For $a, b \in \mathbb{N}$, $b \neq 0$,*



https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

The maps corresponding to negative numbers can also be built, but defining $[-1]$ is somewhat confusing so we won't delve further into this discussion. For more information, see Sobociński (2015). As a last note on scalars, if we work under an uncountable field such as $\mathbb{R}$, there is no way to build all scalars from the generators for cardinality reasons. Hence, formally, one needs an uncountable amount of new generators $\text{—}\boxed{a}\text{—}$ with $a \in \mathbb{R}$.

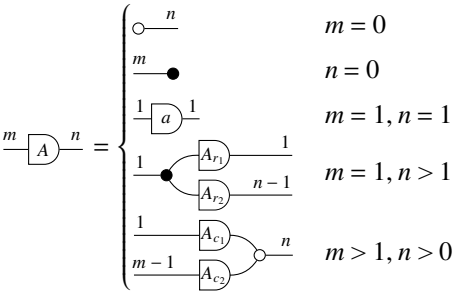
Aside from scalars, another important construction is that of a matrix. In the graphical language, a matrix is considered a linear transformation built inductively from the generators in a way that resembles the column-by-column construction of classical block matrices. More concretely, here's an inductive definition of a matrix in the usual sense.

Definition 8 (Matrix, column-by-column)**.**

$$A_{n\times m} = \begin{cases} [a] & m = 1, n = 1, \\ \begin{bmatrix} A^{r_1}_{1\times 1} \\ A^{r_2}_{n-1\times 1} \end{bmatrix} & m = 1, n > 1, \\ \begin{bmatrix} A^{c_1}_{n\times 1} & A^{c_2}_{n\times m-1} \end{bmatrix} & m > 1, n > 1. \end{cases}$$

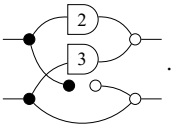
We can mimic this definition in diagrammatic notation.

Definition 9 (Matrix in GLA)**.**



Except for the two first cases, which account for diagrams with zero left/right wires, the different cases in the graphical definition are exactly the different cases in the usual, column-by-column definition. Notice that we write a matrix $\overset{m}{\text{—}}\boxed{A}\overset{n}{\text{—}}$ as a curved diagram, in order to distinguish them from ordinary relations. We write a matrix with lowercase letters *if and only if* it is a 1×1 matrix, i.e. a scalar.

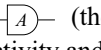
Example 7. *The triangular matrix* $\begin{bmatrix} 2 & 3 \\ 0 & 1 \end{bmatrix}$ *can be written in graphical syntax as*

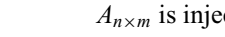


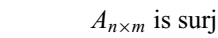
.

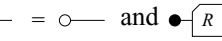
https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

However, this diagram presents an inefficient coding for the given matrix and can be simplified based on the Axioms and Theorems that will be exposed below.

We also mentioned in the Introduction that subspaces are special cases of linear relations. They can be thought of as relations $R \subseteq V \times W$ where $V = \{*\}$ is the zero-dimensional space. In diagrammatic notation, these correspond to the diagrams with 0 wires on the left. We write them as  . Given a matrix $A_{n\times m}$, some familiar subspaces are  (the image of A) and (the kernel of A). With these, we can define surjectivity and injectivity graphically.

$$A_{n\times m} \text{ is injective} \iff Ker(A) = \{0\} \iff$$ 

$$A_{n\times m} \text{ is surjective} \iff Img(A) = \mathbb{R}^n \iff$$ 

In fact, as a side note, linear transformations can be defined as linear relations satisfying  and . If these equations hold, we say the relation R is, respectively, single-valued and total (see Definition 11). It is not difficult to prove, under this definition, that matrices are linear transformations.

Tables 3 and 4 showcase some other constructions that will appear throughout the paper. Some new symbols are introduced: R, S denote mathematical relations; A, B, C, D represent matrices (linear transformations); a, b, c, d are real numbers; and V, W are subspaces. For simplicity, sometimes we write the expression $(x, y) \in R$ as xRy. It is worth noting that the dagger symbol $R^\dagger$ used below is *not* the dagger that appears in the context of dagger categories. A proper inductive definition of the dagger is given in Definition 10, and it corresponds to the intuitive notion of color-swapping, i.e. transforming white nodes into black nodes and vice-versa.

2.4 Diagrammatic reasoning

Now we present the axioms and theorems that will be used for the remainder of the paper. It is worth noting that here there are just three of the axioms of GLA. The complete presentation of the axioms can be found at Paixão *et al*. (2022). Most proofs in this section will be omitted as they are not the focus of this work, but can be found in the same reference. The axioms and theorems are inspired by the notion of an Abelian bicategory as defined by Carboni & Walters (1987). Here, two combinations of diagrams that play an important role are *bialgebras* and *special Frobenius algebras* (respectively, Theorems 2 and 3). As explained by Lack (2004), these are two canonical ways in which monoids and comonoids can interact.

Axiom 1 (Commutative comonoid)**.** *The copy diagram satisfies the equations of commutative comonoids, that is, associativity, commutativity and unitality, as given below:*

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

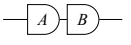
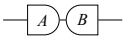
	Graphical notation	Pointwise notation	Name/Notation
1		$\{(x,y) \mid xRy\}$	Relation R
2		$\{(x,y) \mid yRx\}$	Opposite R^o
3		$\left\{(x^*,y^*) \mid x^*Ry^* \text{ and } x^{*\top}x - y^{*\top}y = 0\right\}$	Orthogonal complement (Stein & Samuelson, 2024) $R^\dagger$
4		$\{(x,y) \mid \exists z;\ xRz \text{ and } zSy\} \equiv$ $\{(x,y) \mid x(R \circ S)y\}$	Composition SR
5		$\left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},\begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid x_1Ry_1 \text{ and } x_2Sy_2\right\} \equiv$ $\{(x_1,y_1) \mid x_1Ry_1\} \times \{(x_2,y_2) \mid x_2Sy_2\}$	Cartesian product $R \times S$
6		$\{(x,y) \mid x = y\}$	Identity I
7		$\{(x,y) \mid Ax = y\}$	Matrix A
8		$\{(x,y) \mid B(Ax) = y\}$	Matrix composition BA
9		$\{(x,y) \mid Ax = By\}$	Relational division (Oliveira, 2018)
10		$\{(*,y) \mid \exists x;\ y = Ax\}$	Image $Im(A)$
11		$\{(x,*) \mid Ax = 0\}$	Kernel $Ker(A)$
12		$\left\{\left(x,\begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid \begin{bmatrix}A\\B\end{bmatrix}[x] = \begin{bmatrix}y_1\\y_2\end{bmatrix}\right\}$	Two-by-one block matrix $\begin{bmatrix}A\\B\end{bmatrix}$
13		$\left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},y\right) \mid \begin{bmatrix}A & B\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix} = [y]\right\}$	One-by-two block matrix $\begin{bmatrix}A & B\end{bmatrix}$

Table 3: Dictionary of notations (Part 1).

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

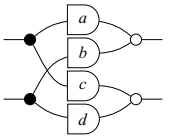
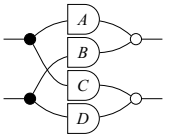
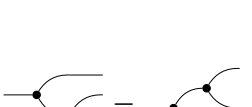
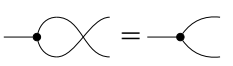
13		$\left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},\begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid \begin{bmatrix}a & b\\c & d\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=\begin{bmatrix}y_1\\y_2\end{bmatrix}\right\}$	Two-by-two matrix $\begin{bmatrix}a & b\\c & d\end{bmatrix}$
14		$\left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},\begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid \begin{bmatrix}A & B\\C & D\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=\begin{bmatrix}y_1\\y_2\end{bmatrix}\right\}$	Two-by-two block matrix $\begin{bmatrix}A & B\\C & D\end{bmatrix}$
15		$\{y \mid y \in V \text{ and } y \in W\}$	Intersection $V \cap W$
16		$\{y \mid \exists v \in V, w \in W; y = v + w\}$	Sum $V + W$

Table 4: Dictionary of notations (Part 2).



$$: \left\{\left(x, \begin{bmatrix}y_1\\y_2\\y_3\end{bmatrix}\right) \mid y_1 = x \text{ and } y_2 = y_3 = x\right\} = \left\{\left(x, \begin{bmatrix}y_1\\y_2\\y_3\end{bmatrix}\right) \mid y_1 = y_2 = x \text{ and } y_3 = x\right\}$$



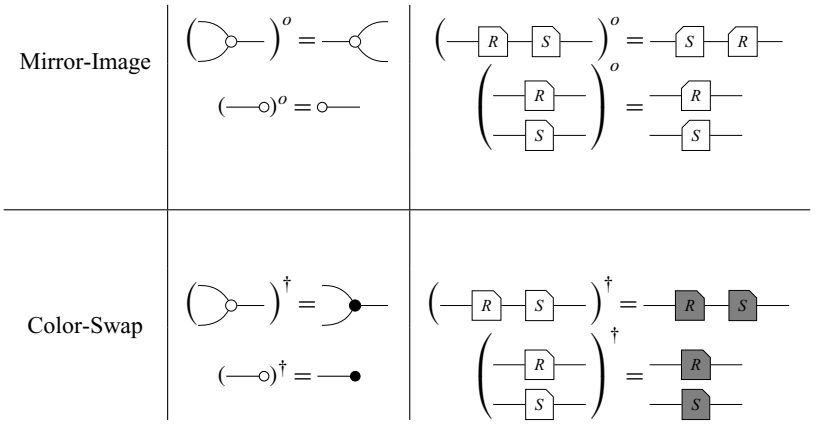
$$: \left\{\left(x, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid y_1 = y_2 = x\right\} = \left\{\left(x, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid y_2 = y_1 = x\right\}$$

$$: \begin{array}{c}\{(x,y) \mid \exists z; x = z = y\} = \\ \{(x,y) \mid x = y\} = \\ \{(x,y) \mid \exists z; x = y = z\}\end{array}$$

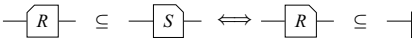
In graphical notation, the diagrams are closed under two symmetries: the "Mirror-Image" and the "Color-Swap".

Definition 10 (Mirror-Image and Color-Swap)**.** *Mirror-Image (denoted by* $(-)^o$*) and Color-Swap (denoted by* $(-)^\dagger$*) are defined on the generators in the obvious way and extended recursively as follows:*

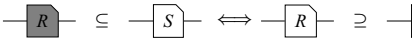
https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press



Axiom 2 (Symmetries)**.** *We have the equivalences* $R^o \subseteq S \iff R \subseteq S^o$*, i.e. in diagrams*

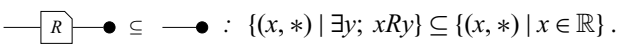


and $R^\dagger \subseteq S \iff R \supseteq S^\dagger$*, or in diagrams*

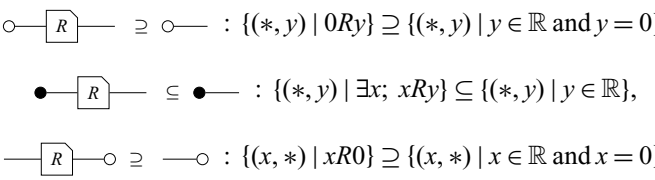


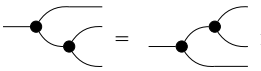
In Axiom 2,  represents the diagram with inverted colors, where black circles turn white, and vice versa. Axiom 2 denotes a particularly relevant advantage of graphical notation. As an example, consider the axiom below.

Axiom 3 (Discard)**.**

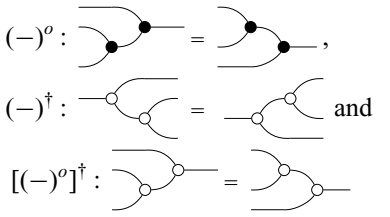


Applying Axiom 2 to Axiom 3, we can assert that:



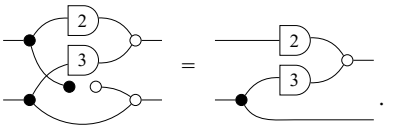
For equalities, this becomes even simpler. It can be assumed that each diagram has three other versions: the 'mirror image,' the 'color swap,' and both together. If an equation is valid, its other three versions are also valid. For example, from Axiom 1, it is known that the equality  is true. Therefore, by Axiom 2:

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press



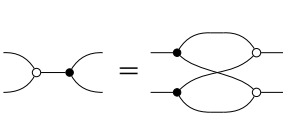
are also true. When a result is proven, it will be assumed that the other three versions are also true, just referencing the original result. For example, all of the above statements will be denoted simply by Axiom 1.

Example 8. *The triangular matrix diagram in Example 7 can be simplified using Axioms 1 and 2:*

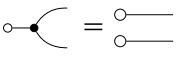


White and black diagrams act as bialgebras when they interact. This can be summarized by the following equations.

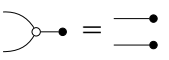
Theorem 2 (Bialgebra)**.**



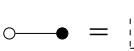
$$: \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid y_1 = y_2 = x_1 + x_2\right\} = \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid y_1 = x_1 + x_2 \text{ and } y_2 = x_1 + x_2\right\}$$



$$: \left\{\left(*, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid y_1 = y_2 = 0\right\} = \left\{\left(*, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid y_1 = 0 \text{ and } y_2 = 0\right\}$$



$$: \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, *\right) \mid \exists y;\ x_1 + x_2 = y\right\} = \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, *\right) \mid \exists \begin{bmatrix}y_1\\y_2\end{bmatrix};\ \begin{bmatrix}x_1\\x_2\end{bmatrix} = \begin{bmatrix}y_1\\y_2\end{bmatrix}\right\}$$

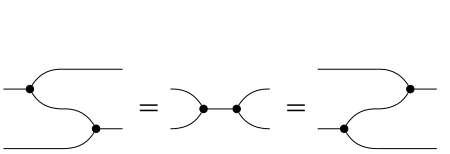


$$: \{(*,*) \mid \exists y;\ y = 0\} = \{(*,*)\}$$

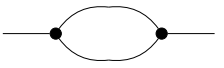
The white and black diagrams interact according to the rules of bialgebras. On the other hand, individually the white and black structures interact as (extraspecial) Frobenius algebras.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

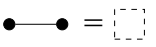
Theorem 3 (Frobenius Algebra)**.** *The following equations hold:*



$$: \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid y_1 = x_1 = y_2 = x_2\right\} = \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid x_1 = x_2 = y_1 = y_2\right\} = \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}, \begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid x_1 = y_1 = x_2 = y_2\right\}$$



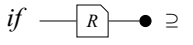
$$: \{(x,y) \mid \exists z_1, z_2;\ x = z_1 = z_2 = y\} = \{(x,y) \mid x = y\}$$

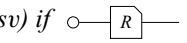


$$: \{(*,*) \mid \exists z;\ z \in \mathbb{R}\} = \{(*,*)\}$$

We now define the following important properties about linear relations.

Definition 11 (Types of relations)**.** *The R relation is called:*

Total (tot) if  $: \{(x,*) \mid \exists y;\ xRy\} \supseteq \{(x,*) \mid x \in \mathbb{R}\}$

Single-Valued (sv) if  $: \{(*,y) \mid 0Ry\} \subseteq \{(*,y) \mid y \in \mathbb{R} \text{ and } y = 0\}$

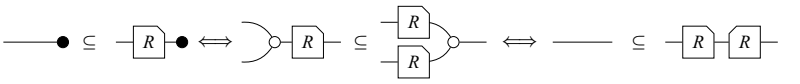
Surjective (sur) if  $: \{(*,y) \mid \exists x;\ xRy\} \supseteq \{(*,y) \mid y \in \mathbb{R}\}$

Injective (inj) if  $: \{(x,*) \mid xR0\} \subseteq \{(x,*) \mid x \in \mathbb{R} \text{ and } x = 0\}$

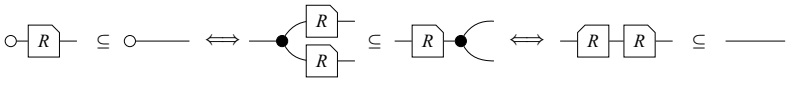
Definition 11 has equivalent statements, summarised in the Theorem below:

Theorem 4. *For every relation R, the following statements are equivalent:*

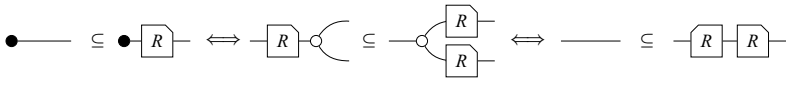
(Total)



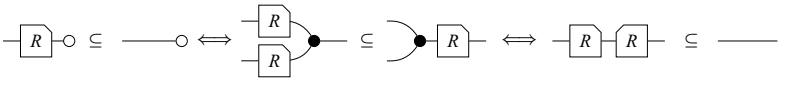
(Single Valued)



(Surjective)



(Injective)



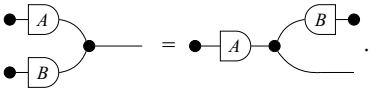
https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Remark. *A diagram is a map if it is total and single-valued. Also, a diagram is a Co-map if it is injective and surjective. In particular, a matrix*  *is a Map.*

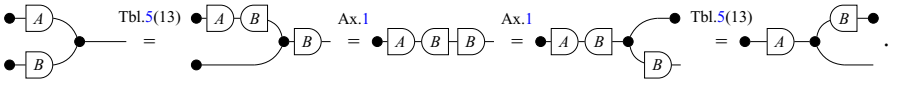
The other theorems relevant to this paper are summarized in Table 5. It is worth mentioning that the translations in pointwise notation presented are well-known theorems in linear algebra, which can be easily verified in a classical textbook, for example (Axler, 1997; Strang, 2009).

In Table 5, the dotted lines in some of the rows have no syntactic meaning and are included solely to visually illustrate the two ways in which the diagram in question can be semantically interpreted as it was done in introduction. For instance, the equation in row 18, which is a theorem in classical notation, implicitly holds true in the graphical syntax.

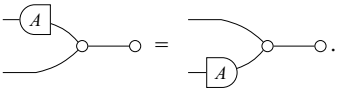
Lastly, we state three additional Lemmas which will be useful in the next sections. Lemmas 1 and 2 will be used in the proof of the Zassenhaus' algorithm (Theorem 9), while Lemma 3 will appear later, in the proof of the Exchange Lemma (Theorem 16). The reason for providing specific proofs of these three lemmas is to use them as examples of how Tables 3 and 5 can help in translating proofs from graphical notation into two other notations.

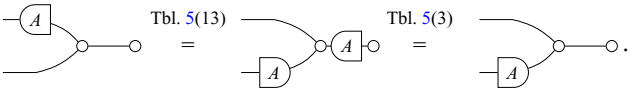
Lemma 1. 

Proof

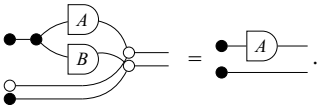


■

Lemma 2. 

Proof 

■

Lemma 3. 

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

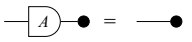
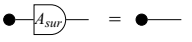
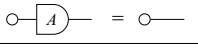
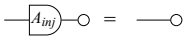
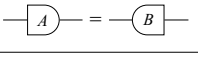
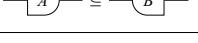
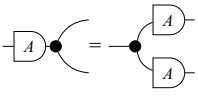
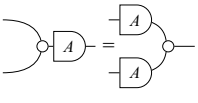
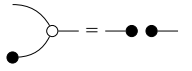
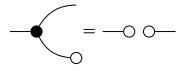
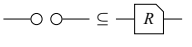
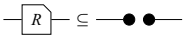
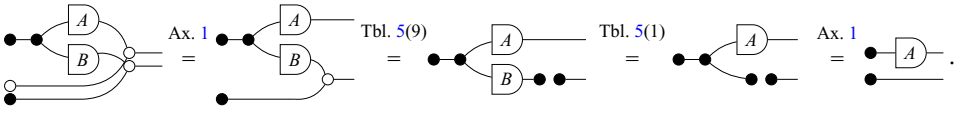
	Graphical notation	Pointwise notation	Classical notation
1		$\{(x,*) \mid \exists y;\ Ax = y\} = \{(x,*) \mid x \in \mathbb{R}\}$	–
2		A is surjective iff $\{(*,y) \mid \exists x;\ Ax = y\} = \{(*,y) \mid y \in \mathbb{R}\}$	A is surjective iff $Im(A) = \mathbb{R}^n$
3		$\{(*,y) \mid A0 = y\} = \{(*,y) \mid y = 0\}$	–
4		A is injective iff $\{(x,*) \mid Ax = 0\} = \{(x,*) \mid x = 0\}$	A is injective iff $Ker(A) = \{0\}$
5		$\{(x,y) \mid y = Ax\} = \{(x,y) \mid By = x\}$	$A = B^{-1}$
6		$\{(x,y) \mid y = Ax\} \subseteq \{(x,y) \mid By = x\}$	$AB = I$
7		$\left\{\left(x, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \mid y_2 = y_1 = I(Ax)\right\} = \left\{\left(x, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \mid y_2 = y_1 = (IA)x\right\}$	$\begin{bmatrix} I \\ I \end{bmatrix} A = \begin{bmatrix} A \\ A \end{bmatrix}$
8		$\left\{\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, y\right) \mid y = A(x_1 + x_2)\right\} = \left\{\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, y\right) \mid y = Ax_1 + Ax_2\right\}$	$A \begin{bmatrix} I & I \end{bmatrix} = \begin{bmatrix} A & A \end{bmatrix}$
9		$\{(x,y) \mid \exists z;\ y = z + x\} = \{(x,y) \mid x,y \in \mathbb{R}\}$	–
10		$\{(x,y) \mid x = y = 0\} = \{(x,y) \mid x = 0 \text{ and } y = 0\}$	–
11		$\{(x,y) \mid x = 0 \text{ and } y = 0\} \subseteq \{(x,y) \mid xRy\}$	Bottom linear relation
12		$\{(x,y) \mid xRy\} \subseteq \{(x,y) \mid \exists z_1, z_2;\ x = z_1 \text{ and } z_2 = y\}$	Top linear relation

Table 5: Theorems, presented in the three different notations (Part 1).

Proof



■

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

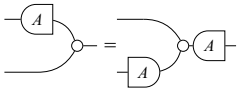
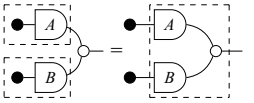
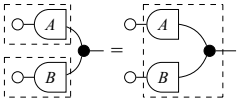
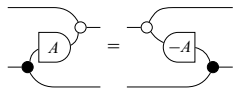
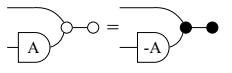
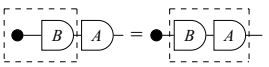
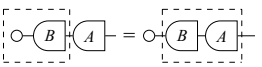
13		$\left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},y\right) \mid \exists z; Az=x_1 \text{ and } z+x_2=y\right\} = \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},y\right) \mid x_1+Ax_2=Ay\right\}$	–
14		$\{(*,y) \mid \exists x_1,x_2;\ Ax_1+Bx_2=y\} = \left\{(*,y) \mid \exists x_1,x_2;\ \begin{bmatrix}A & B\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=y\right\}$	$Im(A)+Im(B) = Im(\begin{bmatrix}A & B\end{bmatrix})$
15		$\{(*,y) \mid Ay=0 \text{ and } By=0\} = \left\{(*,y) \mid \begin{bmatrix}A\\B\end{bmatrix}y=0\right\}$	$Ker(A)\cap Ker(B) = Ker\left(\begin{bmatrix}A\\B\end{bmatrix}\right)$
16		$\left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},\begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid \begin{bmatrix}I & A\\0 & I\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=\begin{bmatrix}y_1\\y_2\end{bmatrix}\right\} = \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},\begin{bmatrix}y_1\\y_2\end{bmatrix}\right) \mid \begin{bmatrix}x_1\\x_2\end{bmatrix}=\begin{bmatrix}I & -A\\0 & I\end{bmatrix}\begin{bmatrix}y_1\\y_2\end{bmatrix}\right\}$	$\begin{bmatrix}I & A\\0 & I\end{bmatrix}^{-1} = \begin{bmatrix}I & -A\\0 & I\end{bmatrix}$
17		$\left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},*\right) \mid x_1+Ax_2=0\right\} = \left\{\left(\begin{bmatrix}x_1\\x_2\end{bmatrix},*\right) \mid x_1=-Ax_2\right\}$	–
18		$\{(*,y) \mid \exists x,\ A(Bx)=y\} = \{(*,y) \mid \exists x,\ (AB)x=y\}$	$A(Im(B)) = Im(AB)$
19		$\{(*,y) \mid B(Ay)=0\} = \{(*,y) \mid (BA)y=0\}$	$A^o(Ker(B)) = Ker(BA)$

Table 6: Theorems, presented in the three different notations (Part 2).

We can almost immediately obtain the same proofs (Lemmas 1, 2 and 3) in pointwise notation by systematically translating each step according to the Tables 3 and 5.

Lemma 4 (Lemma 1 in pointwise version)**.**

$$\{(*,y) \mid \exists x_1,x_2;\ Ax_1=y \textit{ and } Bx_2=y\} = \{(*,y) \mid \exists x_1,x_2;\ Ax_1=y \textit{ and } Ax_1=Bx_2\}$$

Proof

$$\begin{aligned}
& \{(*,y) \mid \exists x_1,x_2;\ Ax_1=y \text{ and } Bx_2=y\} \\
= & \quad \{ \text{ Table 5(13) } \} \\
& \{(*,y) \mid \exists x_1,x_2,\exists z;\ Ax_1=Bz \text{ and } Bx_2=y\} \\
= & \quad \{ \text{ Axiom 1 } \} \\
& \{(*,y) \mid \exists x_1,\exists z;\ Ax_1=Bz \text{ and } Bz=y\} \\
= & \quad \{ \text{ Axiom 1 } \}
\end{aligned}$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

$$
\begin{array}{ll}
 & \{(*, y) \mid \exists x_1, x_2, \exists z;\ Ax_1 = Bz \text{ and } Bz = y \text{ and } z = x_2\} \\
= & \quad \{\ \text{Table 5(13)}\ \} \\
 & \{(*, y) \mid \exists x_1, x_2;\ Ax_1 = y \text{ and } Ax_1 = Bx_2\}
\end{array}
$$

■

Lemma 5 (Lemma 2 in pointwise version)**.** *Let A be a matrix, then*

$$\left\{\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, *\right) \mid \exists z;\ Az = x_1 \text{ and } z + x_2 = 0\right\} = \left\{\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, *\right) \mid x_1 + Ax_2 = 0\right\}$$

Proof

$$
\begin{array}{ll}
 & \left\{\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, *\right) \mid \exists z;\ Az = x_1 \text{ and } z + x_2 = 0\right\} \\
= & \quad \{\ \text{Table 5(13)}\ \} \\
 & \left\{\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, *\right) \mid x_1 + Ax_2 = A0\right\} \\
= & \quad \{\ \text{Table 5(3)}\ \} \\
 & \left\{\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, *\right) \mid\ x_1 + Ax_2 = 0\right\}
\end{array}
$$

■

Lemma 6 (Lemma 3 in pointwise notation)**.** *Given two matrices $A_{n_1 \times m}$ and $B_{n_2 \times m}$, then*

$$Im\left(\begin{bmatrix} A & 0 \\ B & I \end{bmatrix}\right) = Im(A) \times \mathbb{R}^{n_2}$$

Proof

$$
\begin{array}{ll}
 & Im\left(\begin{bmatrix} A & 0 \\ B & I \end{bmatrix}\right) \\
= & \quad \{\ \text{Definition of image - Table 3(9)}\ \} \\
 & \left\{\left(*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \mid \exists \begin{bmatrix} x_1 \\ x_2 \end{bmatrix};\ \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} A & 0 \\ B & I \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right\} \\
= & \quad \{\ \text{Arithmetic}\ \} \\
 & \left\{\left(*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \mid \exists x_1, x_2;\ Ax_1 + 0 = y_1 \text{ and } Bx_1 + x_2 = y_2\right\} \\
= & \quad \{\ \text{Axiom 1}\ \} \\
 & \left\{\left(*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \mid \exists x_1, x_2;\ Ax_1 = y_1 \text{ and } Bx_1 + x_2 = y_2\right\} \\
= & \quad \{\ \text{Table 5(9)}\ \}
\end{array}
$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

$$
\begin{aligned}
& \left\{ \left(*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) \mid \exists x_1, z_1, z_2;\ Ax_1 = y_1,\ Bx_1 = z_1 \text{ and } z_2 = y_2 \right\} \\
= \quad & \{ \text{ Table 5(1) } \} \\
& \left\{ \left(*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) \mid \exists x_1, z_1, z_2;\ Ax_1 = y_1,\ x_1 = z_1 \text{ and } z_2 = y_2 \right\} \\
= \quad & \{ \text{ Axiom 1 } \} \\
& \left\{ \left(*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) \mid \exists x_1, z_2;\ Ax_1 = y_1 \text{ and } z_2 = y_2 \right\} \\
= \quad & \{ \text{ Table 3(4) } \} \\
& \{(*, y_1) \mid \exists x_1;\ Ax_1 = y_1\} \times \{(*, y_2) \mid y_2 \in \mathbb{R}^{n_2}\} \\
= \quad & \{ \text{ Definition of image - Table 3(9) } \} \\
& Im(A) \times \mathbb{R}^{n_2}
\end{aligned}
$$

■

Note that in the proof of Lemma 6, some steps (gray letters) are needed in addition to those used in the graphical version (Lemma 3). In fact, this is quite common. Informally speaking, proofs in GLA often have fewer steps than their classical counterparts. There are two main reasons for this. The first is that, as seen in rows 14, 15, and 18 of Table 5, some non-trivial rules in the usual notation become implicitly true in GLA. The second is that the definitions of kernel and image do not require specific terminology; instead, they can be built from the pre-established fundamental symbols (zero, discard, and matrices).

3 Verification proofs in GLA

This section presents two distinct verification proofs in GLA about linear subspaces. The first one is an introductory example from which the reader can draw intuition. Thus, in this case, the same proof will be written in all three types of notation. The second example will be compared to the widely available Wikipedia proof.

3.1 Introductory example: finding the fundamental subspaces

Definition 12 (Fundamental Subspaces)**.** *For any matrix A, there are four fundamental subspaces: Im*(A)*, Im*$(A^\top)$*, Ker*(A) *and Ker*$(A^\top)$*.*

The algorithm presented by Beezer (2014) determines bases for these four subspaces in a single calculation using Reduced Row Echelon Form (RREF) obtained through row operations. This algorithm can be stated as follows.

Theorem 5 (Bases for Fundamental Subspaces)**.** *For every matrix* $A_{m\times n}$*, if there are surjective matrices C and L, a matrix K, and an invertible matrix X such that*

$$\begin{bmatrix} A & I \end{bmatrix} = X \begin{bmatrix} C & K \\ 0 & L \end{bmatrix}$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

the following equalities hold.

1. $Im(A^\top) = Im(C^\top)$,
2. $Ker(A) = Ker(C)$,
3. $Im(A) = Ker(L)$,
4. $Ker(A^\top) = Im(L^\top)$.

Example 9. *Let* $A = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}$. *Then*

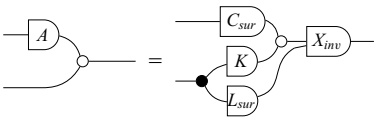
$$\begin{bmatrix} A & I \end{bmatrix} = \left[\begin{array}{cc|ccc} 1 & 1 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 \\ 1 & 3 & 0 & 0 & 1 \end{array}\right] \xrightarrow{RREF} X \left[\begin{array}{cc|ccc} 1 & 0 & 0 & 3 & -2 \\ 0 & 1 & 0 & -1 & 1 \\ \hline 0 & 0 & 1 & -2 & 1 \end{array}\right] = X \begin{bmatrix} C & K \\ 0 & L \end{bmatrix}.$$

See that the equations 1, 2, 3 *and* 4 *of Theorem 5 hold:*

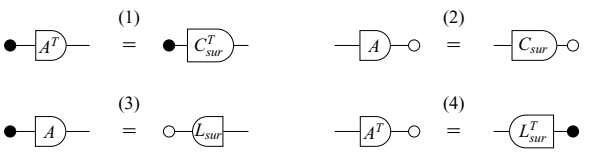
1. $Im(A^\top) = \left\{\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}\right\} = \mathbb{R}^2$ *and* $Im(C^\top) = \left\{\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right\} = \mathbb{R}^2$;
2. $Ker(A) = Ker(C) = 0$;
3. $Ker(L) = \left\{\begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}\right\} = \mathbb{R}^3$ *and* $Im(A) = \left\{\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}\right\} = \mathbb{R}^3$;
4. $Ker(A^\top) = Im(L^\top) = \left\{\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}\right\}$.

In graphical syntax, Theorem 5 takes the following form:

Theorem 6 (Bases for Fundamental Subspaces – Graphical version)**.** *For every matrix* $A_{m\times n}$*, if there are surjective matrices C and L, a matrix K, and an invertible matrix X such that*



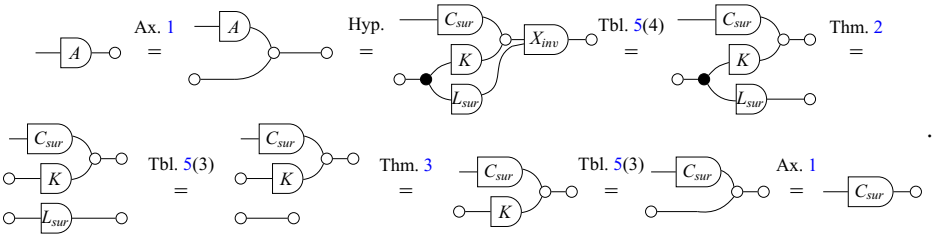
then



We begin by exploring the proof of items 1 and 2.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Proof of (2)



■

Proof of (1) It is proved by the Proof (2) and the Theorem 2. ■

As seen in Section 2, each diagram has a semantic translation to pointwise notation. Therefore, based on the graphical proof above, we immediately obtain the following proof by translating each step systematically.

Proof of (2) [Pointwise version]

$$
\begin{aligned}
& Ker(A) \\
= \quad & \{ \text{ Definition of kernel - Table 3(10) } \} \\
& \{(x, *) \mid Ax = 0\} \\
= \quad & \{ \text{ Axiom 1 } \} \\
& \{(x, *) \mid Ax + 0 = 0\} \\
= \quad & \{ \text{ Matrix notation } \} \\
& \left\{(x, *) \mid \begin{bmatrix} A & \mid I \end{bmatrix} \begin{bmatrix} x \\ 0 \end{bmatrix} = 0 \right\} \\
= \quad & \{ \text{ Hypothesis } \} \\
& \left\{(x, *) \mid X \begin{bmatrix} C & K \\ 0 & L \end{bmatrix} \begin{bmatrix} x \\ 0 \end{bmatrix} = 0 \right\} \\
= \quad & \{ \text{ Arithmetic } \} \\
& \left\{(x, *) \mid X \begin{bmatrix} Cx + K0 \\ L0 \end{bmatrix} = 0 \right\} \\
= \quad & \{ \ X \text{ invertible - Table 5(4) } \} \\
& \left\{(x, *) \mid \begin{bmatrix} Cx + K0 \\ L0 \end{bmatrix} = 0 \right\} \\
= \quad & \{ \text{ Theorem 2 } \} \\
& \{(x, *) \mid Cx + K0 = 0 \text{ and } L0 = 0\} \\
= \quad & \{ \text{ Table 5(3) } \} \\
& \{(x, *) \mid Cx + K0 = 0 \text{ and } 0 = 0\} \\
= \quad & \{ \text{ Theorem 3 } \}
\end{aligned}
$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

$$
\begin{array}{ll}
 & \{(x,*) \mid Cx + K0 = 0\} \\
= & \quad \{ \text{ Table 5(3) } \} \\
 & \{(x,*) \mid Cx + 0 = 0\} \\
= & \quad \{ \text{ Axiom 1 } \} \\
 & \{(x,*) \mid Cx = 0\} \\
= & \quad \{ \text{ Definition of kernel - Table 3(10) } \} \\
 & Ker(C)
\end{array}
$$

■

In a similar way, using Table 5 the same proof can be written in classical notation.

Proof of (2) [Classical version]

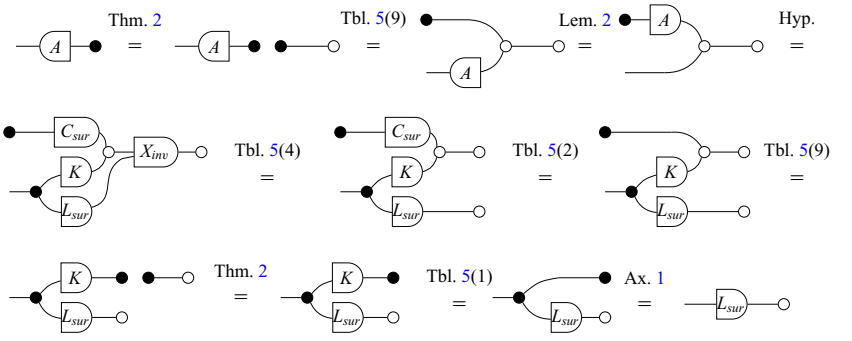
$$
\begin{array}{ll}
 & Ker(A) \\
= & \quad \{ \text{ Breakdown of } A \text{ } \} \\
 & Ker\left(\begin{bmatrix} A & \mid I \end{bmatrix} \begin{bmatrix} I \\ 0 \end{bmatrix} \right) \\
= & \quad \{ \text{ Hypothesis } \} \\
 & Ker\left(X \begin{bmatrix} C & K \\ 0 & L \end{bmatrix} \begin{bmatrix} I \\ 0 \end{bmatrix} \right) \\
= & \quad \{ X \text{ invertible – Table 5(4) } \} \\
 & Ker\left(\begin{bmatrix} C & K \\ 0 & L \end{bmatrix} \begin{bmatrix} I \\ 0 \end{bmatrix} \right) \\
= & \quad \{ \text{ Arithmetic } \} \\
 & Ker\left(\begin{bmatrix} C \\ 0 \end{bmatrix} \right) \\
= & \quad \{ \text{ Kernel intersection – Table 5(15) } \} \\
 & Ker(C) \cap Ker(0) \\
= & \quad \{ Ker(0) = \mathbb{R}^n \} \\
 & Ker(C) \cap \mathbb{R}^n \\
= & \quad \{ \forall A, A \cap \mathbb{R}^n = A \} \\
 & Ker(C)
\end{array}
$$

■

Now we prove items 3 and 4.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Proof of (3) [Graphical version]



■

Proof of (4) [Graphical version] It is proved by the Proof (3) and the Theorem 2. ■

Again, translating each step systematically, we obtain the proof in pointwise notation:

Proof of (3) [Pointwise version]

$$
\begin{aligned}
& Im(A) \\
= & \quad \{ \text{ Definition of image - Table 3(9) } \} \\
& \{(*, y) \mid \exists x;\ Ax = y\} \\
= & \quad \{ \text{ Theorem 2 } \} \\
& \{(*, y) \mid \exists x, z;\ z = 0 \text{ and } Ax = y\} \\
= & \quad \{ \text{ Table 5(9) } \} \\
& \{(*, y) \mid \exists \tilde{x};\ x + \tilde{x} = 0; \text{ and } Ax = y\} \\
= & \quad \{ \text{ Lemma 2 } \} \\
& \{(*, y) \mid \exists \tilde{x};\ A\tilde{x} + y = 0\} \\
= & \quad \{ \text{ Matrix notation } \} \\
& \left\{(*, y) \mid \exists \tilde{x};\ \begin{bmatrix} A \mid & I \end{bmatrix} \begin{bmatrix} \tilde{x} \\ y \end{bmatrix} = 0 \right\} \\
= & \quad \{ \text{ Hypothesis } \} \\
& \left\{(*, y) \mid \exists \tilde{x};\ X \begin{bmatrix} C & K \\ 0 & L \end{bmatrix} \begin{bmatrix} \tilde{x} \\ y \end{bmatrix} = 0 \right\} \\
= & \quad \{ \text{ Arithmetic } \} \\
& \left\{(*, y) \mid \exists \tilde{x};\ X \begin{bmatrix} C\tilde{x} + Ky \\ Ly \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\} \\
= & \quad \{ \ X \text{ invertible} - \text{Table 5(4) } \} \\
& \{(*, y) \mid \exists \tilde{x};\ C\tilde{x} + Ky = 0 \text{ and } Ly = 0\} \\
= & \quad \{ \ C \text{ surjective} - \text{Table 5(2) } \}
\end{aligned}
$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

$$
\begin{aligned}
& \{(*, y) \mid \exists \tilde{x};\ \tilde{x} + Ky = 0 \text{ and } Ly = 0\} \\
= & \quad \{\ \text{Table 5(9)}\ \} \\
& \{(*, y) \mid \exists \tilde{x}_1, \tilde{x}_2;\ Ky = \tilde{x}_1,\ \tilde{x}_2 = 0 \text{ and } Ly = 0\} \\
= & \quad \{\ \text{Theorem 2}\ \} \\
& \{(*, y) \mid \exists \tilde{x}_1;\ Ky = \tilde{x}_1 \text{ and } Ly = 0\} \\
= & \quad \{\ \text{Table 5(1)}\ \} \\
& \{(*, y) \mid \exists \tilde{x}_1;\ y = \tilde{x}_1 \text{ and } Ly = 0\} \\
= & \quad \{\ \text{Axiom 1}\ \} \\
& \{(*, y) \mid Ly = 0\} \\
= & \quad \{\ \text{Definition of kernel – Table 3(10)}\ \} \\
& Ker(L)
\end{aligned}
$$

■

In this case, as shown in Table 5, some steps have no direct translations into classical notation. Hence, the following translation of the graphical proof relies on pointwise notation to fill in the gaps. Informally speaking, this often happens when multiple matrices interact with each other in a non-trivial way. The graphical language is expressive enough to handle these complex interactions without the need to introduce new symbols.

3.2 The Zassenhaus' algorithm

The Zassenhaus algorithm is a well-known method that calculates a basis for the intersection and another for the sum of two subspaces of a vector space. According to Fischer (2012), despite being attributed to Hans Zassenhaus (1912–1991), there is no publication of this algorithm in the mathematician's works. However, there is a historical association regarding its application in Zassenhaus' classic coffee grinder manufacturing company. This algorithm serves as a compelling example of how graphical syntax can be used to reason about linear subspaces.

Although there is published research exploring the Zassenhaus algorithm, the most widely available source is presented in Wikipedia contributors (2023), which describes the algorithm verbatim as presented in Appendix A.

The pointwise Wikipedia proof is using much subspace notation, has to define π_1, $\pi_1|_H$, has the word "obviously", is full of indices to connect the matrices and subspaces, and uses dimension.

We begin by reformulating the algorithm as a theorem to make the assumptions clearer and write the proof in calculational style. In classical notation, the Zassenhaus algorithm, as presented by Wikipedia contributors (2023), can be summarized with the following theorem:

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Theorem 7. *For all subspaces A and B, if there are injective C and D, a matrix E, and invertible X, such that:*

$$\begin{bmatrix} A^\top & A^\top \\ B^\top & 0 \end{bmatrix} = X \begin{bmatrix} C^\top & E^\top \\ 0 & D^\top \\ 0 & 0 \end{bmatrix},$$

then,

1. $Im(A) \cap Im(B) = Im(D)$
2. $Im(A) + Im(B) = Im(C)$

Example 10. *Let* $A = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \end{bmatrix}$ *and* $B = \begin{bmatrix} 5 & 0 \\ 0 & 5 \\ -3 & -3 \end{bmatrix}$. *Then*

$$\begin{bmatrix} A^\top & A^\top \\ B^\top & 0 \end{bmatrix} = X \left[\begin{array}{ccc|ccc} 1 & -1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ \hline 5 & 0 & -3 & 0 & 0 & 0 \\ 0 & 5 & -3 & 0 & 0 & 0 \end{array}\right] \xrightarrow{RREF}$$

$$X \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0.6 \\ 0 & 1 & 0 & 0 & 0 & 0.6 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 & -1 & 0 \end{array}\right] = X \begin{bmatrix} C^\top & E^\top \\ 0 & D^\top \end{bmatrix}.$$

Therefore

1. $Im(A) \cap Im(B) = Im(D) = \left\{ \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \right\}$
2. $Im(A) + Im(B) = Im(C) = \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\} = \mathbb{R}^3$

Note that the matrices are transposed because Wikipedia contributors (2023) formulate the matrix in blocks with row vectors. To facilitate notation, we will consider the block matrix in the following equivalent (transposed) form.

Theorem 8 (Zassenhaus)**.** *For all matrices A and B, if there are injective matrices C and D, a matrix E, and an invertible matrix X such that*

$$\begin{bmatrix} A & B \\ A & 0 \end{bmatrix} = \begin{bmatrix} C & 0 & 0 \\ E & D & 0 \end{bmatrix} X^\top,$$

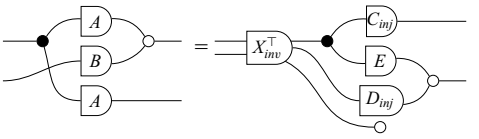
then

1. $Im(A) \cap Im(B) = Im(D)$,
2. $Im(A) + Im(B) = Im(C)$.

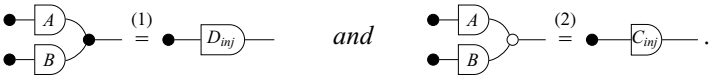
https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Similarly, this theorem can be expressed in graphical syntax.

Theorem 9 (Zassenhaus – Graphical version)**.** *For all matrices A and B, if there are injective matrices C and D, a matrix E, and an invertible matrix X such that*

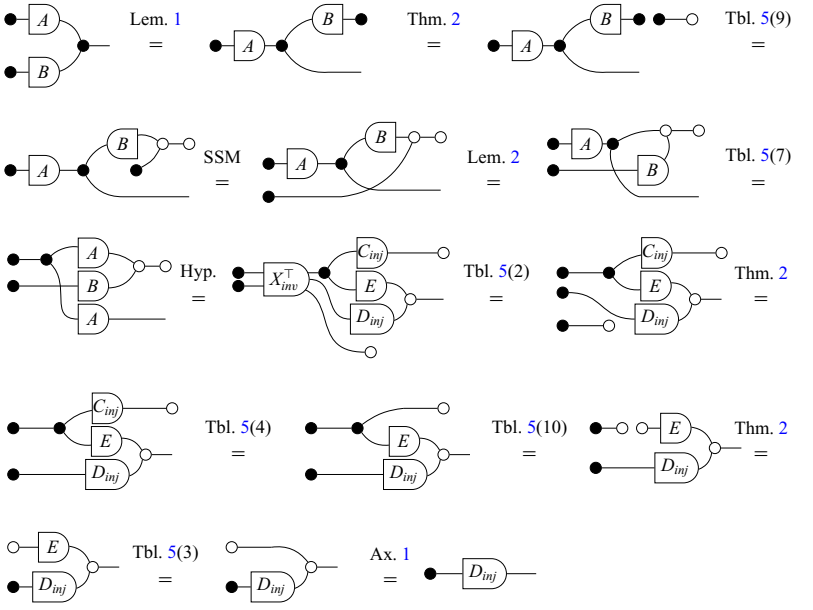


then



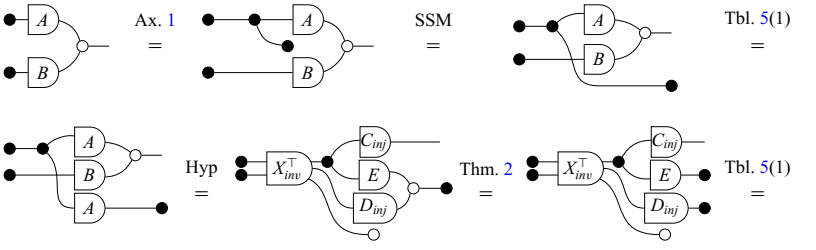
Let us rephrase the proof of the theorem using graphical syntax.

Proof of (1) [Graphical version]

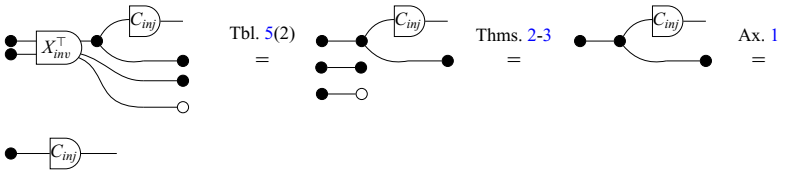


■

Proof of (2) [Graphical version]



https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press



■

See that we have achieved a completely calculational and point-free proof. Moreover, the proof in graphical syntax serves as a guide to produce a new proof in pointwise notation (Appendix B).

4 Programs and properties derivation in GLA

In this section, four main algorithms will be explored: how to find the right inverse of a wide triangular matrix, how to switch from an implicit basis description to an explicit basis description of a subspace, how to find a basis for the intersection of two subspaces and the exchange lemma. The first three algorithms are good examples of how it is possible to use graphical syntax to perform program derivations concerning linear subspaces. The last algorithm exemplifies the use of graphical syntax to derive important properties to be considered for problem-solving.

4.1 Simple example: Right inverse of a wide triangular matrix

As an introductory example, we will discuss the problem of finding the right inverse of a wide triangular matrix.

Definition 13 (Recursive Wide Triangular Matrix/Classical notation)**.** *A wide triangular matrix $T_{m\times n}$ is recursively defined as*

$$T_{m\times n} = \begin{cases} T_{1\times n} \text{ is a non-zero vector,} & \text{if } m = 1 \\ \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}, & \text{if } m > 1 \end{cases}$$

where T_{12} is a matrix and T_{11} and T_{22} are smaller wide triangular matrices.

This definition is almost like that of a triangular matrix, but it allows for more columns than rows. Note that it is tricky to define this matrix non-recursively. This definition leads to matrices that are surjective but not necessarily injective. In fact, every matrix constructed by the above recursive rule is surjective. Below, we present a proof by verification in classical notation.

Theorem 10 (Verification/Classical version)**.** *Let T be a wide triangular matrix, then T has a right inverse T^{+}. In other words, there exists an T^{+} such that $TT^{+} = I$ and*

$$T^{+}_{m\times n} = \begin{cases} \begin{bmatrix} 0 & \cdots & 1/t_i & \cdots & 0 \end{bmatrix}^{\top} & \text{if } m = 1 \\ \begin{bmatrix} T^{+}_{11} & -T^{+}_{11}T_{12}T^{+}_{22} \\ 0 & T^{+}_{22} \end{bmatrix}, & \text{if } m > 1 \end{cases}$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Proof The proof will be done by strong induction. First, for the base case, consider a wide triangular matrix $T_{1\times n} = \begin{bmatrix} t_1 & t_2 & \ldots & t_n \end{bmatrix}$.

$$
\begin{array}{ll}
 & TT^{+} \\
= & \quad \{ \text{ Definitions of } T \text{ and } T^{+} \ \} \\
 & \begin{bmatrix} t_1 & \cdots & t_i & \cdots & t_n \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 1/t_i \\ \vdots \\ 0 \end{bmatrix} \\
= & \quad \{ \ T \text{ is non-zero so there exists a } i \text{ such that } t_i \text{ is non-zero } \} \\
 & t_i/t_i \\
= & \quad \{ \text{ Arithmetic } \} \\
 & 1
\end{array}
$$

For the inductive step, consider T in blocks as in the recursive definition.

$$
\begin{array}{ll}
 & TT^{+} \\
= & \quad \{ \text{ Definitions of } T \text{ and } T^{+} \ \} \\
 & \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{bmatrix} T_{11}^{+} & -T_{11}^{+} T_{12} T_{22}^{+} \\ 0 & T_{22}^{+} \end{bmatrix} \\
= & \quad \{ \text{ Matrix multiplication } \} \\
 & \begin{bmatrix} T_{11} T_{11}^{+} & -T_{11} T_{11}^{+} T_{12} T_{22}^{+} + T_{12} T_{22}^{+} \\ 0 & T_{22} T_{22}^{+} \end{bmatrix} \\
= & \quad \{ \text{ Induction Hypothesis } \} \\
 & \begin{bmatrix} I & -T_{12} T_{22}^{+} + T_{12} T_{22}^{+} \\ 0 & I \end{bmatrix} \\
= & \quad \{ \text{ Arithmetic } \} \\
 & \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}
\end{array}
$$

■

Example 11. *The matrix* $T = \begin{bmatrix} 3 & 2 & 3 & 4 \\ 0 & 5 & 2 & 1 \end{bmatrix}$ *is wide triangular. So* $T_{11}^{+} = \begin{bmatrix} 1/3 \end{bmatrix}$, $T_{22}^{+} = \begin{bmatrix} 1/5 & 1/2 & 1 \end{bmatrix}^{\top}$ *and*

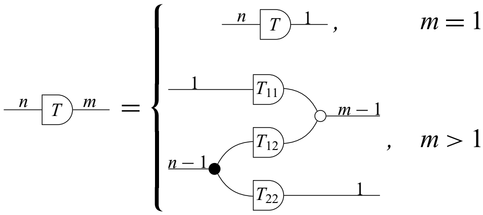
$$T^{+} = \begin{bmatrix} 1/3 & -59/30 \\ 0 & 1/5 \\ 0 & 1/2 \\ 0 & 1 \end{bmatrix}.$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Note that $TT^+ = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. *Therefore,* T^+ *is the right inverse of* T.

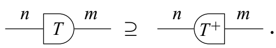
The disadvantage of this method has already been pointed out. One has to provide a candidate for the right inverse prior to the verification. In order to derive the inverse, we first reformulate the problem in GLA.

Definition 14 (Recursive Wide Triangular Matrix/Graphical version)**.** *A triangular wide matrix* $T_{m\times n}$ *is defined recursively by*

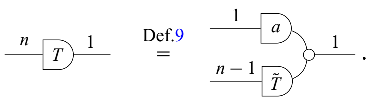


where T_{11} *and* T_{22} *are smaller wide triangular matrices.*

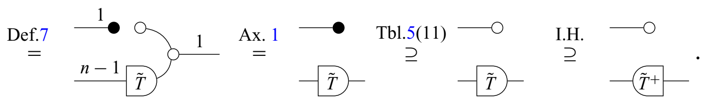
Theorem 11 (Derivation/Graphical version)**.** *Let* T *be a wide triangular matrix. There exists a wide triangular matrix* T^+ *such that:*



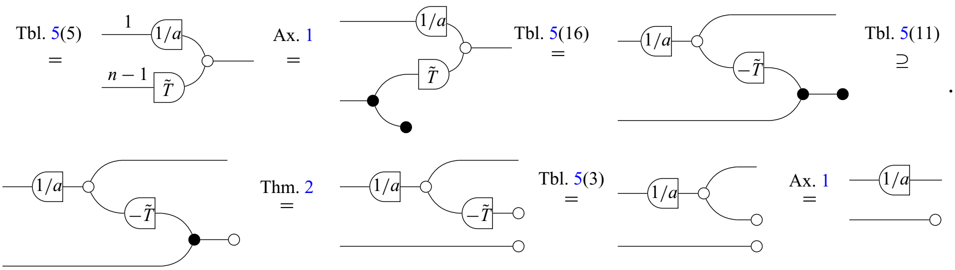
Proof The proof will be done by induction. For the base case ($T_{1\times n}$) we have

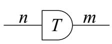


If $a = 0$,

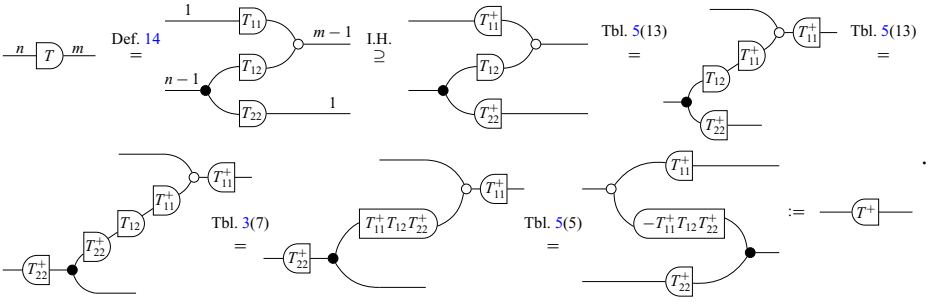


If $a \neq 0$,



For the inductive step, consider  in blocks as in the recursive Definition 14.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press



■

This automatically leads to a computer implementation (by simply translating each step of the proof into the pseudocode).

Algorithm 1. Right Inverse Triangular

procedure `TriMatrix` INVERSE(`TriMatrix` $T_{m\times n}$)
  If $m > 1$:
    $\begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \leftarrow A$
    $T_{11}^{+} \leftarrow \text{INVERSE}(A_{11})$
    $T_{22}^{+} \leftarrow \text{INVERSE}(A_{22})$
    $T_{12}^{+} \leftarrow -T_{11}^{+} T_{12} T_{22}^{+}$
    $T^{+} \leftarrow \begin{bmatrix} T_{11}^{+} & T_{12}^{+} \\ 0 & T_{22}^{+} \end{bmatrix}$
    return T^{+}
  If $m = 1$:
    $\begin{bmatrix} t_{1\times 1} & T' \end{bmatrix} \leftarrow T$
    If $t = 0$:
      return $\begin{bmatrix} 0 \\ \text{INVERSE}(T') \end{bmatrix}$
    If $t \neq 0$:
      return $\begin{bmatrix} 1/t \\ 0 \end{bmatrix}$

4.2 Switching from implicit to explicit basis

A subspace can always be represented by implicit and explicit bases, as presented by Zanasi (2015). Let $B \subseteq \mathbb{R}^n$ be a subspace, and let $a_1, a_2, ..., a_m$ be vectors orthogonal to B, as depicted in Figure 2. An implicit basis of this subspace is given by

$$\{x \mid x \perp a_1, x \perp a_2, \ldots, x \perp a_m\} = \left\{x \mid a_1^{\top} x = 0, a_2^{\top} x = 0, \ldots, a_m^{\top} x = 0\right\} = \left\{x \mid A^{\top} x = 0\right\}.$$

On the other hand, given $b_1, b_2, \ldots, b_r \in B$, an explicit basis of B is given by

$$\{x \mid \exists c_1, c_2, \cdots c_n \in \mathbb{R};\ c_1 b_1 + c_2 b_2 + \cdots + c_r b_r = x\} = \{x \mid \exists c \in \mathbb{R}^n;\ Bc = x\}.$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

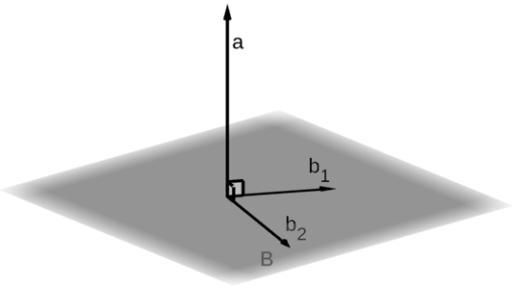
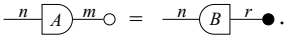


Fig. 2: Geometric representation of the implicit and explicit bases of the B subspace.

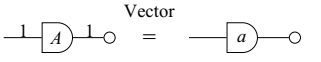
Example 12. *The subspace orthogonal to the vector* $a = \begin{bmatrix}1 & 2 & 3\end{bmatrix}^\top$ *can also be generated by the column vectors of the matrix* $B = \begin{bmatrix}-2 & -3\\ 1 & 0\\ 0 & 1\end{bmatrix}$.

This can be denoted by the following theorem and proved recursively in graphical syntax.

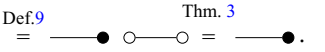
Theorem 12 (Switch from implicit to explicit basis)**.** *For every matrix* $A_{m\times n}$*, there exists a matrix* $B_{n\times r}$ *such that*



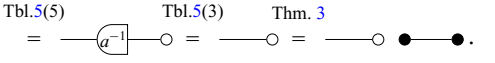
Proof The proof will be done by induction. For the base case, if $m = 1$, there are two cases: **Case** 1 **(**$n = 1$**):**



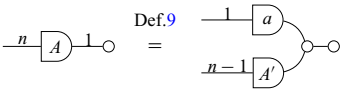
If $a = 0$,



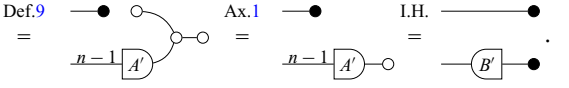
If $a \neq 0$,



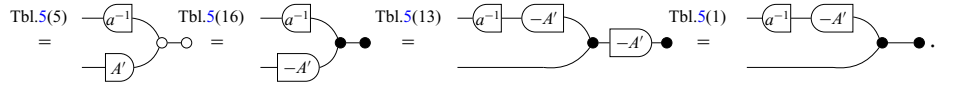
Case 2 **(**$n > 1$**):**



If $a = 0$,

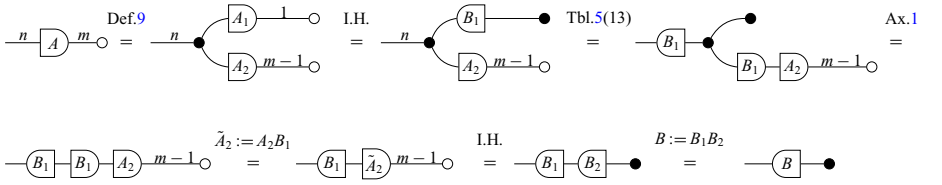


If $a \neq 0$,



https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Inductive step:



■

The inductive step can of course be written in classical linear algebra notation mixed with relational algebra notation as follows.

Proof [Classical version]

$$
\begin{aligned}
& Ker(A) \\
= & \quad \{ \text{ Definition 9 } \} \\
& Ker\left(\begin{bmatrix} A_1 \\ A_2 \end{bmatrix}\right) \\
= & \quad \{ \text{ Table 5 (15) } \} \\
& Ker(A_1) \cap Ker(A_2) \\
= & \quad \{ \text{ Induction hypothesis } \} \\
& Im(B_1) \cap Ker(A_2) \\
= & \quad \{ \text{ Definition 1 } \} \\
& B_1\mathbb{R}^n \cap Ker(A_2) \\
= & \quad \{ \text{ Table 5 (13) } \} \\
& B_1(\mathbb{R}^n \cap B_1^o Ker(A_2)) \\
= & \quad \{ \text{ Table 5 (19) } \} \\
& B_1(\mathbb{R}^n \cap Ker(A_2B_1)) \\
= & \quad \{ \forall X, \mathbb{R}^n \cap X = X \ \} \\
& B_1(Ker(A_2B_1)) \\
= & \quad \{ \ \tilde{A}_2 := A_2B_1 \ \} \\
& B_1(Ker(\tilde{A}_2)) \\
= & \quad \{ \text{ Induction hypothesis } \} \\
& B_1(Im(B_2)) \\
= & \quad \{ \text{ Table 5 (18) } \} \\
& Im(B_1B_2) \\
= & \quad \{ \ B := B_1B_2 \ \} \\
& Im(B)
\end{aligned}
$$

■

Furthermore, this proof provides a nice recursive algorithm presented below:

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Algorithm 2. Implicit to Explicit Basis

procedure `Matrix` IMPLICITEXPLICIT(`Matrix` $A_{m\times n}$)

If $m > 1$:

$\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \leftarrow A$

$B_1 \leftarrow$ IMPLICITEXPLICIT(A_1)

$\tilde{A}_2 \leftarrow A_2 \cdot B_1$

$B_2 \leftarrow$ IMPLICITEXPLICIT($\tilde{A}_2$)

$B \leftarrow B_1 \cdot B_2$

return B

If $n > 1$:

$\begin{bmatrix} a_{1\times 1} & A' \end{bmatrix} \leftarrow A$

If $a = 0$:

return $\begin{bmatrix} I & 0 \\ 0 & \text{IMPLICITEXPLICIT}(A') \end{bmatrix}$

If $a \neq 0$:

return $\begin{bmatrix} -a^{-1}A' \\ I \end{bmatrix}$

If $n = 1$:

$\begin{bmatrix} a_{1\times 1} \end{bmatrix} \leftarrow A$

If $a = 0$:

return [1]

If $a \neq 0$:

return [0]

4.2.1 Calculating a basis for the intersection

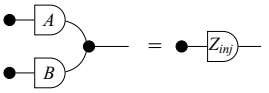
The example of section (4.2) motivates the next Theorem, in which we want to build a basis for the intersection of two subspaces.

Example 13. *As shown in Example 10, a basis for the intersection of the images of the subspaces* $A = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \end{bmatrix}$ *and* $B = \begin{bmatrix} 5 & 0 \\ 0 & 5 \\ -3 & -3 \end{bmatrix}$ *is given by* $\begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$, *ie,*

$$Im(A) \cap Im(B) = \left\{ \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \right\}.$$

See the theorem:

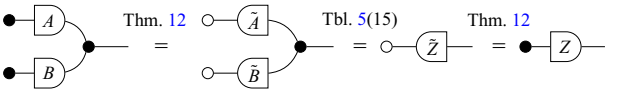
Theorem 13 (Base for intersection)**.** *For all matrices* $A_{k\times m}$ *and* $B_{k\times n}$, *there is an injective matrix* Z *such that*



https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Using the Theorem 12, we obtain the derivation.

Proof



■

This derivation provides yet another algorithm that returns a basis for the intersection of two subspaces given as input:

Algorithm 3. Basis for the intersection

procedure `Matrix` BASISINTERSECTION(`Matrix` A, `Matrix` B)
$\tilde{A} \leftarrow$ IMPLICITEXPLICIT(A)
$\tilde{B} \leftarrow$ IMPLICITEXPLICIT(B)
$\tilde{Z} \leftarrow \begin{bmatrix} \tilde{A} \\ \tilde{B} \end{bmatrix}$
$Z \leftarrow$ IMPLICITEXPLICIT($\tilde{Z}$)
return Z

From the symmetries property of the graphical syntax (Axiom 2), it is also possible to obtain a program derivation and, consequently, an algorithm for the sum of two subspaces. This is the dual of Theorem 13.

4.3 The exchange lemma

Let V be a finite-dimensional vector space and let $A = (a_1, ..., a_r)$ and $B = (b_1, ..., b_s)$ be two subspaces of V such that A is linearly independent and

$$Im(A) \subseteq Im(B).$$

The exchange lemma says that $r \leq s$ and that there exists a subspace C formed by the vectors of A and $s - r$ vectors of B such that $Im(C) = Im(B)$. More details can be found in (Barańczuk & Szydło, 2021).

Example 14. *Let* (e_1, e_2, e_3, e_4) *be the standard basis in* $\mathbb{R}^4$,

$$A = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 1 \\ -1 & -1 & 1 \\ 1 & -1 & -1 \\ 0 & 0 & -2 \end{bmatrix} \quad \textit{and}$$

$$B = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 & b_5 \end{bmatrix} = \begin{bmatrix} e_1 & e_2 & e_3 & e_4 & -e_4 \end{bmatrix}.$$

The subspace $C = \begin{bmatrix} a_1 & a_2 & a_3 & b_3 & b_4 \end{bmatrix}$ *is such that* $Im(C) = Im(B) = \mathbb{R}^4$.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

The major issue of the lemma is "selecting" which vectors of B make this true, i.e:

$$Im(A) + Im(B\,[\text{Selector}]) = Im(B).$$

Inspired by the solution presented by Barańczuk & Szydło (2021), this problem can be stated as the following theorem:

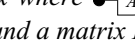
Theorem 14 (Exchange lemma – Verification/Classical version)**.** *Let V be a finite-dimensional vector space and let A and B be two subspaces of V such that A is linearly independent and $Im(A) \subseteq Im(B)$. Thus, there exists $X = P \begin{bmatrix} I \\ M \end{bmatrix} Z$, where P is a permutation matrix, I is the identity and Z is invertible, such that $A = BX$ and*

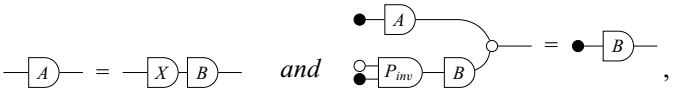
$$Im(A) + Im\left(BP \begin{bmatrix} 0 \\ I \end{bmatrix}\right) = Im(B).$$

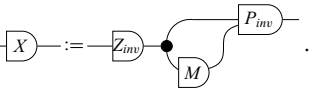
The expression $P \begin{bmatrix} 0 \\ I \end{bmatrix}$ is the *selector* that determines the proper vectors of B. The matrix $\begin{bmatrix} 0 \\ I \end{bmatrix}$ has the function of selecting or discarding vectors, while the permutation matrix P determines the order in which this happens. For example, considering B with size $n \times 4$, a possible selector would be

$$Im\left(\begin{bmatrix}(b_1) & (b_2) & (b_3) & (b_4)\end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}\right) = Im\left(\begin{bmatrix}(b_4) & (b_2)\end{bmatrix}\right).$$

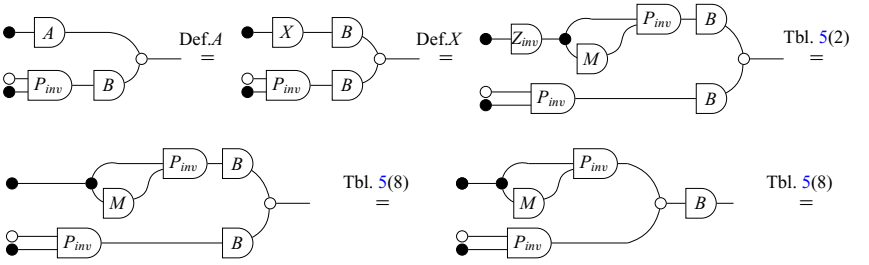
Theorem 14 can be rewritten in graphical syntax as follows:

Theorem 15 (Exchange lemma – Verification/Graphical version)**.** *Let A be an injective matrix where*  *for some matrix B. Then, there exist invertible matrices Z, P and a matrix M such that*

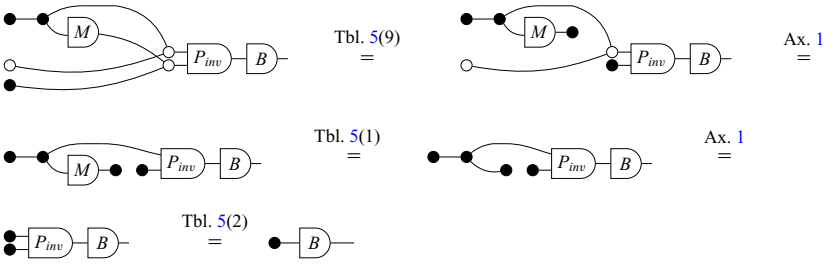


where 

Proof

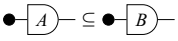


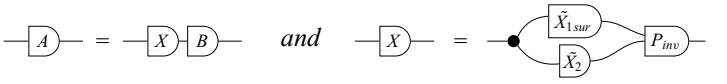
https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press



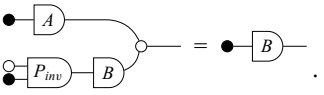
■

To solve this problem, Barańczuk & Szydło (2021) decomposed the matrix X into row-echelon form. It is possible to do a property derivation to find a way to decompose X by rewriting Theorem 15 as

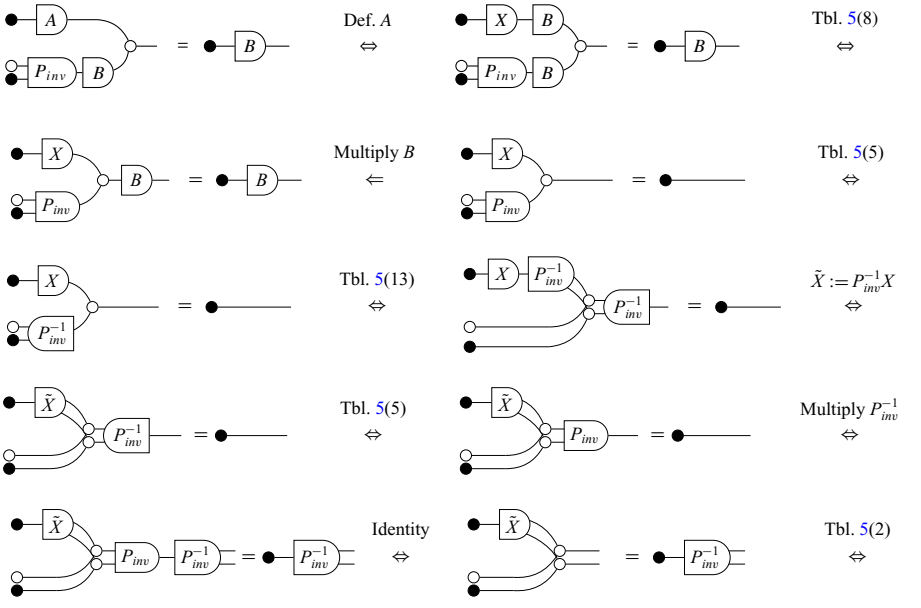
Theorem 16 (Exchange lemma – Derivation/Graphical version)**.** *Let A be an injective matrix and*  *for some matrix B. Suppose that*



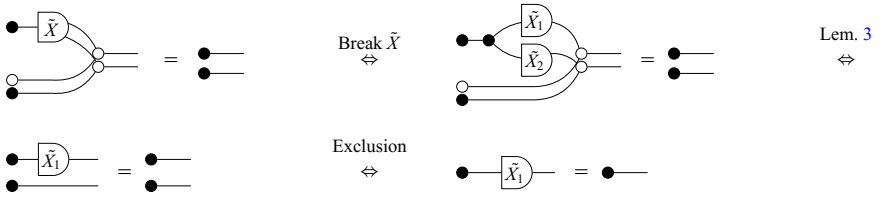
for some matrix X, surjective matrix $\tilde{X}_1$, matrix $\tilde{X}_2$ and invertible matrix P. Then,

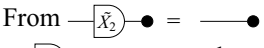
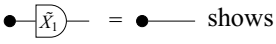


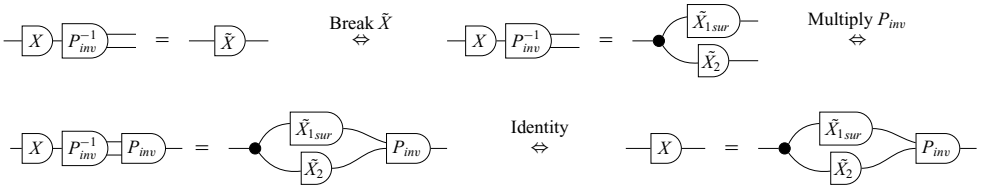
Proof



https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press



From  , we can say that $\tilde{X}_2$ is any matrix. On the other hand, the equality  shows us that $\tilde{X}_1$ must be at least surjective. Thus,



■

Note that $X = P \begin{bmatrix} \tilde{X}_1 \\ \tilde{X}_2 \end{bmatrix}$ gives the form of some of the possible solutions to the problem. This derivation also extends the solution initially presented by the reference, but this time is constructed according to the requirements of the problem. We can then write an algorithm that receives the matrix X, uses the derived property, and returns the desired "selector":

Algorithm 4. Selector for the Exchange Lemma

procedure `Matrix` SELECTOR(`Matrix` X)

$P \begin{bmatrix} I \\ M \end{bmatrix} Z \leftarrow$ ROWECHELON(X)

[Selector] $\leftarrow P \begin{bmatrix} 0 \\ I \end{bmatrix}$

return [Selector]

5 Conclusions and future work

Throughout this paper, a series of examples involving problems of different nature were presented to demonstrate that graphical linear algebra allows for calculational, point-free reasoning, and program derivation on matrices and linear subspaces.

Future work will mainly focus on graphical syntax. The next step will be to build a characterization for general relationships in graphic notation, expanding their use. Furthermore, a certainly productive direction will be to establish, in diagrams, a schematic description of several other normal forms and matrix factorizations, allowing the exploration and elucidation of the theory of several known theorems and algorithms.

From the development of graphical syntax and its consequent review of concepts, another area to focus on in future work is program derivation. For this, a good start will be to find derivations for theorems previously proven only by verification.

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Acknowledgements

Several anonymous individuals are thanked for contributions to these instructions.

Data availability statement

Data availability is not applicable to this article as no new data were created or analysed in this study.

Author contributions

All authors contributed equally to analysing data and reaching conclusions and in writing the paper.

Funding

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

Conflicts of interest

The authors report no conflict of interest.

References

Arens, R. (1961) Operational calculus of linear relations. *Pac. J. Math*. **11**(1), 9–23.

Axler, S. J. (1997) *Linear Algebra Done Right*. Undergraduate Texts in Mathematics. Springer.

Baez, J. & Erbele, J. (2015) Categories in control. *Theory Appl. Categ*. **30**, 836–881.

Barańczuk, S. & Szydło, B. (2021) Two algorithms for the exchange lemma. *Numerical Algorithms* **86**(3), 1041–1050.

Baroni, M. & Zamparelli, R. (2010) Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 1183–1193.

Beezer, R. A. (2014) Extended echelon form and four subspaces. *Am. Math. Mon*. **121**(7), 644–647.

Bird, R. & De Moor, O. (1996) The algebra of programming. In NATO ASI DPD, pp. 167–203.

Bonchi, F., Holland, J., Pavlovic, D. & Sobocinski, P. (2017) Refinement for signal flow graphs. In 28th International Conference on Concurrency Theory (CONCUR 2017), Dagstuhl, Germany. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, pp. 24:1–24:16.

Bonchi, F., Piedeleu, R., Sobociński, P. & Zanasi, F. (2019) Graphical affine algebra, pp. 1–12.

Bonchi, F., Sobociński, P. & Zanasi, F. (2014) A categorical semantics of signal flow graphs. In International Conference on Concurrency Theory. Springer, pp. 435–450.

Bonchi, F., Sobocinski, P. & Zanasi, F. (2015) Full abstraction for signal flow graphs. In POPL 2015. ACM, pp. 515–526.

Bonchi, F., Sobociński, P. & Zanasi, F. (2017) Interacting Hopf algebras. *J. Pure Appl. Algebra* **221**(1), 144–184.

Carboni, A. & Walters, R. F. (1987) Cartesian bicategories I. *J. Pure Appl. Algebra* **49**(1-2), 11–32.

Coddington, A. (1973) *Extension Theory of Formally Normal and Symmetric Subspaces*. Memoirs of the American Mathematical Society, nr. 134. ISBN 0-8218-1834-1.

Cross, R. (1998) *Multivalued Linear Operators*. Chapman & Hall/CRC Pure and Applied Mathematics. Taylor & Francis.

Fischer, G. (2012) *Lernbuch Lineare Algebra Und Analytische Geometrie*, 2nd ed. Springer Vieweg.

Gonthier, G. (2011) Point-free, set-free concrete linear algebra. In Interactive Theorem Proving: Second International Conference, ITP 2011, Berg en Dal, The Netherlands, August 22–25, 2011. Proceedings 2. Springer, pp. 103–118.
Hinze, R. & Marsden, D. (2023) *Introducing String Diagrams: The Art of Category Theory*. Cambridge University Press.
Lack, S. (2004) Composing props. *Theory Appl. Categ*. **13**(9), 147–163.
Mac Lane, S. (1961) An algebra of additive relations. *Proc. Natl. Acad. Sci. U.S.A.* **47**(7), 1043.
Mac Lane, S. (2013) *Categories for the Working Mathematician*, vol. 5. Springer Science & Business Media.
Macedo, H. D. & Oliveira, J. N. (2013) Typing linear algebra: A biproduct-oriented approach. *Sci. Comput. Program*. **78**(11), 2160–2191.
Oliveira, J. N. (2018) Programming from metaphorisms. *J. Logical Algebraic Methods Program*. **94**, 15–44.
Paixão, J., Rufino, L. & Sobociński, P. (2022) High-level axioms for graphical linear algebra. *Sci. Comput. Program*. **218**, 102791.
Santos, A. & Oliveira, J. N. (2020) Type your matrices for great good: A haskell library of typed matrices and applications (functional pearl). In Proceedings of the 13th ACM SIGPLAN International Symposium on Haskell. New York, NY, USA: Association for Computing Machinery, pp. 54–66.
Selinger, P. (2010) A survey of graphical languages for monoidal categories. In *New Structures for Physics*. Springer, pp. 289–355.
Selinger, P. (2011) A survey of graphical languages for monoidal categories. In *New Structures for Physics*, pp. 289–355.
Sernadas, A., Ramos, J. & Mateus, P. (2008) Linear algebra techniques for deciding the correctness of probabilistic programs with bounded resources. *Preprint, SQIG-IT and IST-TU Lisbon*, pp. 1049–001.
Sobociński, P. (2015) Graphical linear algebra. *Mathematical blog.[Online]*. Available at: https://graphicallinearalgebra.net.
Stein, D. & Samuelson, R. (2024) Towards a compositional framework for convex analysis (with applications to probability theory). In Foundations of Software Science and Computation Structures. Cham: Springer Nature Switzerland, pp. 166–187.
Strang, G. (2009) *Introduction to Linear Algebra*, 4th ed. Wellesley, MA: Wellesley-Cambridge Press.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2023) Attention is all you need.
Wikipedia contributors. (2023) Zassenhaus algorithm — Wikipedia, the free encyclopedia. [Online; accessed 16-November-2023].
Zanasi, F. (2015) *Interacting Hopf Algebras: The Theory of Linear Systems*. PhD Thesis. Ecole Normale Supérieure de Lyon.

Appendix A Wikipedia's Zassenhaus algorithm

Input

Let V be a vector space and U, W two finite-dimensional subspaces of V with the following spanning sets:

$$U = \langle u_1, \ldots, u_n \rangle \quad \text{and} \quad W = \langle w_1, ..., w_k \rangle .$$

Finally, let $B_1, \ldots, B_m$ be linearly independent vectors so that u_i and w_i can be written as

$$u_i = \sum_{j=1}^{m} a_{i,j} B_j \quad \text{and} \quad w_i = \sum_{j=1}^{m} b_{i,j} B_j.$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Output

The algorithm computes the base of the sum $U + W$ and a base of the intersection $U \cap W$.

Algorithm

The algorithm creates the following block matrix of size $((n + k) \times (2m))$:

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m} & a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,m} & a_{n,1} & a_{n,2} & \cdots & a_{n,m} \\ b_{1,1} & b_{1,2} & \cdots & b_{1,m} & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ b_{k,1} & b_{k,2} & \cdots & b_{k,m} & 0 & 0 & \cdots & 0 \end{pmatrix}$$

Using elementary row operations, this matrix is transformed to the row echelon form. Then, it has the following shape:

$$\begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,m} & \bullet & \bullet & \cdots & \bullet \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ c_{q,1} & c_{q,2} & \cdots & c_{q,m} & \bullet & \bullet & \cdots & \bullet \\ 0 & 0 & \cdots & 0 & d_{1,1} & d_{1,2} & \cdots & d_{1,m} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & d_{\ell,1} & d_{\ell,2} & \cdots & d_{\ell,m} \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

Here, $\bullet$ stands for arbitrary numbers, and the vectors $(c_{p,1}, c_{p,2}, \ldots, c_{p,m})$ for every $p \in \{1, ..., q\}$ and $(d_{p,1}, d_{p,2}, \ldots, d_{p,m})$ for every for every $p \in \{1, ..., \ell\}$ are nonzero.

Then $(y_1, \ldots, y_q)$ with

$$y_i := \sum_{j=1}^{m} c_{i,j} B_j$$

is a basis of $U + W$ and $(z_1, \ldots, z_\ell)$ with

$$z_i := \sum_{j=1}^{m} d_{i,j} B_j$$

is a basis of $U \cap W$.

Proof of correctness

First, we define $\pi_1 : V \times V \to V, (a, b) \mapsto a$ a to be the projection to the first component. Let $H := \{(u, u) \mid u \in U\} + \{(w, 0) \mid w \in W\} \subseteq V \times V$. Then $\pi_1(H) = U + W$ and $H \cap (0 \times V) = 0 \times (U \cap W)$. Also, $H \cap (0 \times V)$ is the kernel of $\pi_1|_H$, the projection restricted to H. Therefore, $\dim(H) = \dim(U + W) + \dim(U \cap W)$. The Zassenhaus algorithm calculates a basis of H. In the first m columns of this matrix, there is a basis y_i of $U + W$. The rows of the form $(0, z_i)$ (with $z_i \neq 0$) are obviously in $H \cap (0 \times V)$. Because the matrix is in row echelon form, they are also linearly independent. All rows which are different from zero $((y_i, \bullet)$ and $(0, z_i))$ are a basis of H, so there are $\dim(U \cap W)$ such z_is. Therefore, the z_is form a basis of $U \cap W$.

(Wikipedia contributors, 2023)

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

Appendix B Zassenhaus: Pointwise proof

Proof [Pointwise notation – Calculational but pointwise]

(1)

$$
\begin{aligned}
& Im(A) \cap Im(B) \\
= \quad & \{ \text{ Definition of the intersection – Table 3(15) } \} \\
& \{(*, y) |\, y \in Im(A) \text{ and } y \in Im(B)\} \\
= \quad & \{ \text{ Definition of image – Table 3(9) } \} \\
& \{(*, y) |\, \exists x_1, x_2;\ Ax_1 = y \text{ and } Bx_2 = y\} \\
= \quad & \{ \text{ Lemma 1 } \} \\
& \{(*, y) |\, \exists x_1, x_2;\ Ax_1 = y \text{ and } Ax_1 = Bx_2\} \\
= \quad & \{ \text{ Theorem 2 } \} \\
& \{(*, y) |\, \exists x_1, x_2, x_3;\ Ax_1 = y,\ Ax_1 = Bx_2 \text{ and } x_3 = 0\} \\
= \quad & \{ \text{ Table 5(9) } \} \\
& \{(*, y) |\, \exists x_1, x_4;\ Ax_1 = y,\ Ax_1 = Bx_2 \text{ and } x_2 + x_4 = 0\} \\
= \quad & \{ \text{ Lemma 2 } \} \\
& \{(*, y) |\, \exists x_1, x_4;\ Ax_1 = y \text{ and } Ax_1 + Bx_4 = 0\} \\
= \quad & \{ \text{ Matrix notation } \} \\
& \left\{(*, y) |\, \exists x_1, x_4;\ \begin{bmatrix} A & B \\ A & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}\right\} \\
= \quad & \{ \text{ Hypothesis } \} \\
& \left\{(*, y) |\, \exists x_1, x_4;\ \begin{bmatrix} C & 0 & 0 \\ E & D & 0 \end{bmatrix} X^{\top} \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}\right\} \\
= \quad & \{ \text{ Variable change } \} \\
& \left\{(*, y) |\, \exists x_1, x_4,\ \exists \tilde{x}_1, \tilde{x}_2, \tilde{x}_3;\ \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \tilde{x}_3 \end{bmatrix} = X^{\top} \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} \text{ and } \begin{bmatrix} C & 0 & 0 \\ E & D & 0 \end{bmatrix} \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \tilde{x}_3 \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}\right\} \\
= \quad & \{ X \text{ invertible – Table 5(2) } \} \\
& \left\{(*, y) |\, \exists \tilde{x}_1, \tilde{x}_2, \tilde{x}_3;\ \begin{bmatrix} C & 0 & 0 \\ E & D & 0 \end{bmatrix} \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \tilde{x}_3 \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}\right\} \\
= \quad & \{ \text{ Arithmetic } \} \\
& \{(*, y) |\, \exists \tilde{x}_1, \tilde{x}_2, \tilde{x}_3;\ C\tilde{x}_1 = 0 \text{ and } E\tilde{x}_1 + D\tilde{x}_2 = y\} \\
= \quad & \{ \text{ Theorem 2 } \} \\
& \{(*, y) |\, \exists \tilde{x}_1, \tilde{x}_2;\ C\tilde{x}_1 = 0 \text{ and } E\tilde{x}_1 + D\tilde{x}_2 = y\} \\
= \quad & \{ C \text{ injective – Table 5(4) } \}
\end{aligned}
$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

$$
\begin{array}{ll}
 & \{(*,y)|\ \exists \tilde{x}_1,\tilde{x}_2;\ \tilde{x}_1=0 \text{ and } E\tilde{x}_1+D\tilde{x}_2=y\} \\
= & \quad \{\ \text{Table 5(10)}\ \} \\
 & \{(*,y)|\ \exists \tilde{x}_1,\tilde{x}_2;\ \tilde{x}_1=0 \text{ and } E0+D\tilde{x}_2=y\} \\
= & \quad \{\ \text{Theorem 2}\ \} \\
 & \{(*,y)|\ \exists \tilde{x}_2;\ E0+D\tilde{x}_2=y\} \\
= & \quad \{\ \text{Table 5(3)}\ \} \\
 & \{(*,y)|\ \exists \tilde{x}_2;\ 0+D\tilde{x}_2=y\} \\
= & \quad \{\ \text{Axiom 1}\ \} \\
 & \{(*,y)|\ \exists \tilde{x}_2;\ D\tilde{x}_2=y\} \\
= & \quad \{\ \text{Definition of image – Table 3(9)}\ \} \\
 & Im(D)
\end{array}
$$

(2)

$$
\begin{array}{ll}
 & Im(A)+Im(B) \\
= & \quad \{\ \text{Definition of sum – Table 3(16)}\ \} \\
 & \{(*,y)|\ y\in Im(A)+Im(B)\} \\
= & \quad \{\ \text{Definition of image – Table 3(9)}\ \} \\
 & \{(*,y)|\ \exists x_1,x_2;\ Ax_1+Bx_2=y\} \\
= & \quad \{\ \text{Axiom 1}\ \} \\
 & \{(*,y)|\ \exists x_1,x_2,\tilde{y};\ x_1=\tilde{y} \text{ and } Ax_1+Bx_2=y\} \\
= & \quad \{\ \text{Table 5(1)}\ \} \\
 & \{(*,y)|\ \exists x_1,x_2,\tilde{y};\ Ax_1=\tilde{y} \text{ and } Ax_1+Bx_2=y\} \\
= & \quad \{\ \text{Matrix notation}\ \} \\
 & \left\{(*,y)|\ \exists x_1,x_2,\tilde{y};\ \begin{bmatrix}A & B\\ A & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=\begin{bmatrix}y\\ \tilde{y}\end{bmatrix}\right\} \\
= & \quad \{\ \text{Hypothesis}\ \} \\
 & \left\{(*,y)|\ \exists x_1,x_2,\tilde{y};\ \begin{bmatrix}C & 0 & 0\\ E & D & 0\end{bmatrix}X^{\top}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=\begin{bmatrix}y\\ \tilde{y}\end{bmatrix}\right\} \\
= & \quad \{\ \text{Variable change}\ \} \\
 & \left\{(*,y)|\ \exists x_1,x_2,\tilde{y},\exists \tilde{x}_1,\tilde{x}_2,\tilde{x}_3;\ \begin{bmatrix}\tilde{x}_1\\ \tilde{x}_2\\ \tilde{x}_3\end{bmatrix}=X^{\top}\begin{bmatrix}x_1\\ x_2\end{bmatrix} \text{ and } \begin{bmatrix}C & 0 & 0\\ E & D & 0\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ \tilde{x}_2\\ \tilde{x}_3\end{bmatrix}=\begin{bmatrix}y\\ \tilde{y}\end{bmatrix}\right\} \\
= & \quad \{\ X \text{ invertible – Table 5(2)}\ \} \\
 & \left\{(*,y)|\ \exists \tilde{y},\tilde{x}_1,\tilde{x}_2,\tilde{x}_3;\ \begin{bmatrix}C & 0 & 0\\ E & D & 0\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ \tilde{x}_2\\ \tilde{x}_3\end{bmatrix}=\begin{bmatrix}y\\ \tilde{y}\end{bmatrix}\right\} \\
= & \quad \{\ \text{Arithmetic}\ \}
\end{array}
$$

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press

$$
\begin{aligned}
& \{(*, y) \mid \exists \tilde{y}, \tilde{x}_1, \tilde{x}_2, \tilde{x}_3;\ C\tilde{x}_1 = y \text{ and } E\tilde{x}_1 + D\tilde{x}_2 = \tilde{y}\} \\
= \quad & \{ \text{ Theorem 2 } \} \\
& \{(*, y) \mid \exists \tilde{y}, \tilde{x}_1, \tilde{x}_2;\ C\tilde{x}_1 = y \text{ and } E\tilde{x}_1 + D\tilde{x}_2 = \tilde{y}\} \\
= \quad & \{ \text{ Variable change } \} \\
& \{(*, y) \mid \exists \tilde{y}, \tilde{x}_1, \tilde{x}_2, \exists \tilde{y}_1, \tilde{y}_2;\ C\tilde{x}_1 = y,\ E\tilde{x}_1 = \tilde{y}_1,\ D\tilde{x}_2 = \tilde{y}_2 \text{ and } \tilde{y} = \tilde{y}_1 + \tilde{y}_2\} \\
= \quad & \{ \text{ Theorem 2 } \} \\
& \{(*, y) \mid \exists \tilde{x}_1, \tilde{x}_2, \tilde{y}_1, \tilde{y}_2;\ C\tilde{x}_1 = y,\ E\tilde{x}_1 = \tilde{y}_1 \text{ and } D\tilde{x}_2 = \tilde{y}_2\} \\
= \quad & \{ \text{ Table 5(1) } \} \\
& \{(*, y) \mid \exists \tilde{x}_1, \tilde{x}_2, \tilde{y}_1, \tilde{y}_2;\ C\tilde{x}_1 = y,\ \tilde{x}_1 = \tilde{y}_1 \text{ and } \tilde{x}_2 = \tilde{y}_2\} \\
= \quad & \{ \text{ Theorem 3 } \} \\
& \{(*, y) \mid \exists \tilde{x}_1, \tilde{y}_1;\ C\tilde{x}_1 = y \text{ and } \tilde{x}_1 = \tilde{y}_1\} \\
= \quad & \{ \text{ Axiom 1 } \} \\
& \{(*, y) \mid \exists \tilde{x}_1;\ C\tilde{x}_1 = y\} \\
= \quad & \{ \text{ Definition of image – Table 3(9) } \} \\
& Im(C)
\end{aligned}
$$

■

https://doi.org/10.1017/S0956796825000085 Published online by Cambridge University Press