ADDITION AND DIFFERENTIATION OF ZX-DIAGRAMS

EMMANUEL JEANDEL a,b,c,d, SIMON PERDRIX a,b,c,d, AND MARGARITA VESHCHEZEROVA e

a LORIA, Campus scientifique, BP 239, 54506, Vandoeuvre-lès-Nancy Cedex, France
email address: emmanuel.jeandel@loria.fr, simon.perdrix@loria.fr
b CNRS, Délegation Centre-Est, 17 rue Notre-Dame des Pauvres, BP 10075, 54519, Vandoeuvre-lès-Nancy Cedex, France
c Université de Lorraine, 34 cours Léopold, 54052, Nancy Cedex, France
d Inria MOCQUA, 615 Rue du Jardin-Botanique, 54600 Villers-lès-Nancy, France
e Terra Quantum AG, St. Gallerstrasse 16A, 9400 Rorschach, Switzerland
e-mail address: mv@terraquantum.swiss

ABSTRACT. The ZX-calculus is a powerful framework for reasoning in quantum computing. It provides in particular a compact representation of matrices of interest. A peculiar property of the ZX-calculus is the absence of a formal sum allowing the linear combinations of arbitrary ZX-diagrams. The universality of the formalism guarantees however that for any two ZX-diagrams, the sum of their interpretations can be represented by a ZX-diagram. We introduce a general, inductive definition of the addition of ZX-diagrams, relying on the construction of controlled diagrams. Based on this addition technique, we provide an inductive differentiation of ZX-diagrams.

Indeed, given a ZX-diagram with variables in the description of its angles, one can differentiate the diagram according to one of these variables. Differentiation is ubiquitous in quantum mechanics and quantum computing (e.g. for solving optimization problems). Technically, the differentiation of ZX-diagrams is strongly related to summation as witnessed by the product rules.

We also introduce an alternative, non-inductive, differentiation technique rather based on the isolation of the variables. Finally, we apply our results to deduce a diagram for an Ising Hamiltonian.

1. INTRODUCTION

ZX-calculus, originally introduced in [CD11], is a graphical language that allows reasoning about quantum computing. In this language complex computations on qubits are represented with diagrams made out of elementary generators\(^1\). Each diagram corresponds to a linear transformation between Hilbert spaces of qubit states. A compact set of rewrite rules allows to transform diagrams into equivalent ones. The notable advantage of ZX-calculus compared

\(^1\)We use the TikZit software https://tikzit.github.io to draw ZX-diagrams
to other representations (e.g. linear maps, circuits, and tensor networks) is that in this language the computations may be done *entirely graphically*. A general introduction to the language alongside the overview of the main applications is available in [vdW20].

Due to its flexibility, ZX-calculus is widely used to address different problems of quantum computing. For instance, ZX-calculus allowed the derivation of important results in the field of *measurement-based quantum computing (MBQC)* [DP10, KvdW19]. It was also successfully applied to circuit optimization [CDD+20, dBBW20, KvdW20, DKPvdW20], design and verification of error-correction color codes [GD18], and to the analysis and compilation of *surface codes* [Hor11, dBH20, HEMN20].

However, the applications of the ZX-calculus to the rapidly growing field of variational algorithms [CAB+21] such as *Quantum Approximate Optimization Algorithm (QAOA)* [FGG14], *Variational Quantum Eigensolver (VQE)* [PMS+14] and variational quantum machine learning are so far limited. Nevertheless, as variational algorithms do not require error correction, the incoming emergence of *NISQ* devices makes them an object of particular attention [Pre18]. We believe that the reason why they are still unexplored with the means of ZX-calculus is the absence of a convenient way to *differentiate parametrized diagrams*. Indeed, the basic building blocks of variational algorithms are parametrized circuits, and the search for optimal parameter values is a crucial part of these algorithms. In theory, finding optimal parameters may be NP-hard [BK21]. In practice, the search is usually done by classical numerical optimization methods and most of them use derivatives [GS17].

The main difficulty for differentiation of ZX-diagrams comes from the *product rule*: 
\[ \partial (fg) = \partial f g + f \partial g \]
that involves adding two terms. This rule is crucial to evaluate the derivative of the sequential composition and the tensor product. As ZX-diagrams are defined inductively with these two compositions, the derivative of a complex diagram could be computed out of the derivatives of its parts if we were allowed to use the product rule, and therefore sum diagrams.

The works [ZG21, TYd21] use explicit sums of diagrams (usually referred to as *bags of diagrams*) to represent the derivative of diagrams with multiple occurrences of the parameter. The major disadvantage of this approach is that there are no rules to manipulate sums of ZX-diagrams. Therefore, we can’t fully exploit the power of graphical computation (while it’s still possible to use rewrite rules on summands [SH22]). In this work, we suggest an approach where *the derivative of a parametrized ZX-diagram is another ZX-diagram*. Hence we avoid the extension of the signature with formal sums. In order to tackle sums that appear in the product rule, we introduce an original technique to perform the addition of diagrams entirely in the ZX-calculus. For this purpose, we use special diagrams called controlled states [JPV19]. We suggest an inductive way to represent every ZX-diagram by such a state. As we know how to sum controlled states [JPV19] the addition for arbitrary diagrams follows. An inductive definition of the derivative is obtained by an explicit diagrammatic representation of the product rules.

In an attempt to give a ready-to-use toolbox for differentiation, we provide an easy and convenient way to compute the derivative for the family of linear diagrams $ZX(\beta)$ [JPV19], that is diagrams where angles might depend linearly (with integer coefficients) on one parameter $\beta$. Most of the circuits that are used for variational algorithms belong to $ZX(\beta)$ and we believe that our formulas will make their analysis much simpler.

A definition for a derivative similar to our formulas was obtained in the independent work [WY22]. This work uses *W-spin* [Had15] to handle product rules. In contrast to our result, the diagrammatic differentiation presented in [WY22] maps to a ZX-diagram a diagram
from another language called *algebraic ZX-calculus* [Wan20]. The algebraic ZX-calculus is convenient to represent arbitrary complex numbers, therefore their differentiation procedure can handle more general families of diagrams than \( ZX(\beta) \). This advantage comes with the cost of abandoning the legacy of the vanilla ZX-calculus which is by far the most popular graphical calculus for quantum computing.

In the end, we show how our result together with the Stone’s theorem [Sto32] allows finding a ZX-diagram representing an *Ising Hamiltonian*. In variational algorithms Ising Hamiltonians are typically used in the definition of the loss function for parameter optimization. In practical applications Hamiltonians are usually given as a sum of local terms \( H = \sum_{i=1}^{k} H_i \) where each \( H_i \) acts only on a limited number of qubits. Small individual terms \( H_i \) can be relatively easily represented as diagrams. For instance, for the Hamiltonian \( H = \sum_{u,v} Z_u Z_v \) diagrams for individual terms are trivial: \( H_{u,v} = \)

The ability to perform addition directly leads to a procedure for differentiation. On the other direction, with Stone’s theorem some specific kinds of sums might be straightforwardly expressed as diagrams if we can compute derivatives. We demonstrate how to combine our formula for the derivatives and the theorem to find a diagrammatic representation for example Hamiltonian.

**Structure of the paper.** In Section 2, we give a brief introduction to the ZX-calculus followed by some useful lemmas. In Section 3, we provide a general overview of addition and differentiation in quantum computing. In particular, we show how addition may be represented in circuit notation. We continue by giving an overview of the potential benefits coming from using ZX-diagrams to reason about derivatives. This section, alongside additional examples and explanations, constitutes the major extension of the current work with respect to the previous version [JPV22].

In the next Section 4 we introduce an inductive procedure for the transformation of a diagram into a controlled form. We show how this procedure leads to an algorithm for the addition of ZX-diagrams. In Section 5, we provide two approaches for the differentiation, including an inductive definition (directly inspired by the product rules) and compact formulas for derivatives of linear diagrams \( ZX(\beta) \) where \( \beta \) is a vector of parameters. In Section 6, we show how to apply our formulas to obtain a diagram for an Ising Hamiltonian.

### 2. ZX-calculus

#### Syntax and Semantics.

The ZX-diagrams are generated by green spiders \( \alpha \), red spiders \( \beta \) and Hadamard \( \chi \), where both kinds of spiders have an arbitrary number of inputs/outputs and are decorated with angles. ZX-diagrams are also made of wires: the identity \( | \), the swap \( \bigodot \) and also the possibility to bend wires with a cup \( \cup \) and a cap \( \cap \). A wire can connect two nodes or, alternatively, have free ends that point either up or down. Wires with free ends pointing towards the top (towards the bottom) of the diagram are called *inputs (outputs)*. We denote by \( D : n \to m \) a diagram with \( n \) inputs and \( m \) outputs. Finally, the empty diagram is denoted with \( \bot \).
Definition 2.1. ZX-diagrams are inductively defined as follows: for $n, m \in \mathbb{N}$ and $\alpha \in \mathbb{R}/2\pi\mathbb{Z}$,

\[
\begin{align*}
\begin{array}{ccc}
& \vdots & \\
\vdots & & \\
& \vdots & \\
\end{array} : & n \to m & \begin{array}{ccc}
& \vdots & \\
\vdots & & \\
& \vdots & \\
\end{array} : & n \to m & \begin{array}{ccc}
\circ & : & 1 \to 1 \\
\circ & : & 1 \to 1 \\
\end{array} & \begin{array}{ccc}
\circ & : & 1 \to 1 \\
\circ & : & 1 \to 1 \\
\end{array} \\
m & : & 2 \to 0 & m & : & 0 \to 2 & \begin{array}{ccc}
\cdot & : & 2 \to 2 \\
\cdot & : & 0 \to 0 \\
\end{array}
\end{align*}
\]

are ZX-diagrams, and for any ZX-diagrams $D_0 : a \to b$, $D_1 : b \to c$, and $D_2 : c \to d$, $D_1 \circ D_0 : a \to c$ and $D_0 \otimes D_2 : a + c \to b + d$ are ZX-diagrams. Pictorially:

\[
\begin{array}{ccc}
D_1 & \circ & D_0 \\
\downarrow & & \downarrow \\
D_0 & \circ & D_1 \\
\downarrow & & \downarrow \\
D_0 & \otimes & D_2 \\
\downarrow & & \downarrow \\
D_0 & \otimes & D_2
\end{array}
\quad \text{and} \quad
\begin{array}{ccc}
D_0 & \otimes & D_1 \\
\downarrow & & \downarrow \\
D_0 & \circ & D_1 \\
\downarrow & & \downarrow \\
D_0 & \otimes & D_2 \\
\downarrow & & \downarrow \\
D_0 & \otimes & D_2
\end{array}
\]

As in most works on ZX-calculus, an empty spider denotes the spider with $\alpha = 0$. A diagram with no input/output is called a scalar. To compactly write scalar factors, we introduce syntactic sugar $[-]^\otimes n$. For any scalar $d : 0 \to 0$ the notation $d^\otimes n$ corresponds to $d \otimes \cdots \otimes d$.

Semantically, ZX-diagrams are standardly interpreted as linear maps, and thus they can be used to represent quantum evolutions.

Definition 2.2. For any ZX-diagram $D : n \to m$, let $[D] \in \mathcal{M}_{2^n,2^m}(\mathbb{C})$ be inductively defined as: $[D_1 \circ D_0] = [D_1] \circ [D_0]$, $[D_0 \otimes D_2] = [D_0] \otimes [D_2]$, and

\[
\begin{align*}
\begin{bmatrix}
\vdots \\
\vdots \\
\vdots \\
m & \\
\end{bmatrix} : & |0\rangle^{\otimes m} \langle 0|^{\otimes n} + e^{i\alpha} |1\rangle^{\otimes m} \langle 1|^{\otimes n} & \begin{bmatrix}
\vdots \\
\vdots \\
\vdots \\
m & \\
\end{bmatrix} : & |+\rangle^{\otimes m} \langle +|^{\otimes n} + e^{i\alpha} |−\rangle^{\otimes m} \langle −|^{\otimes n} \\
\begin{bmatrix}
| \end{bmatrix} : & |0\rangle \langle 0| + |1\rangle \langle 1| & \begin{bmatrix}
\circ \end{bmatrix} : & |+\rangle \langle 0| + |−\rangle \langle 1| & \begin{bmatrix}
\circ \end{bmatrix} : & 1 \\
\begin{bmatrix}
\cdot \end{bmatrix} : & \langle 00| + \langle 11| & \begin{bmatrix}
\cdot \end{bmatrix} : & \langle 00| + \langle 11| & \begin{bmatrix}
\cdot \end{bmatrix} : & \sum_{i,j \in \{0,1\}} |ij\rangle \langle ji|
\end{align*}
\]

where bra-ket notations are used: $|0\rangle = (1\ 0)^t$, $|1\rangle = (0\ 1)^t$, $|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$, $|−\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$, $|xy\rangle = |x\rangle \otimes |y\rangle$ and $\langle x| = |x\rangle^\dagger$ is the adjoint (complex conjugate) of $|x\rangle$.

Example 2.3. Complex numbers can be represented with diagrams:

\[
\begin{align*}
[\square] = 2, & & [\bigcirc] = 1 + e^{i\alpha}, & & [\bigotimes] = \sqrt{2}, & & [\bigotimes] = \sqrt{2}e^{i\alpha}, & & [\bigodot] = \frac{1}{\sqrt{2}} (2.1)
\end{align*}
\]

Sometimes it is meaningful to consider diagrams with angles from a restricted sub-group $\mathcal{G}$ of $\mathbb{R}/2\pi\mathbb{Z}$. Such restrictions lead to fragments of the language, denoted ZX$\mathcal{G}$-calculus [JPV19]. The standard interpretation associates to each ZX$\mathcal{G}$-diagram $D : n \to m$ a matrix
In particular, the $\frac{\pi}{4}$- (resp. $\pi$-) fragment\footnote{i.e. the fragment of diagrams which angles are in the group generated by $\frac{\pi}{4}$ (resp. $\pi$)}, also called Clifford (resp. real Clifford) fragment, enjoys nice properties [Bac14, DP14] but is not universal for quantum computing, even approximately. Furthermore, any quantum computation that can be expressed in this fragment can be efficiently simulated on a classical computer. As soon as the group contains the angle $\frac{\pi}{4}$, the corresponding fragment is approximatively universal for quantum computing: any $2^n \times 2^n$ unitary transformation can be approximated by a ZX-diagram from this fragment with arbitrary precision. In particular the $\frac{\pi}{4}$-fragment, also called ‘Clifford+T’ fragment has be extensively studied [JPV18a, JPVW17, NW18]. Other finitely generated fragments have been considered in [JPV19].

Notice that for any sub-group $G$ of $\mathbb{R}/2\pi\mathbb{Z}$ that contains $\frac{\pi}{4}$, ZX$_G$-diagrams are universal [JPV19] in the sense that for any matrix $M \in \mathcal{M}_{2^m,2^n}(\mathcal{R}_G)$ there exists a ZX$_G$-diagram $D : n \rightarrow m$ such that $[D] = M$.
2.3. **Useful lemmas.** Theorems and demonstrations in this work extensively use triangle:  

- a syntactic sugar introduced in [JPV18a]. It corresponds to a non-unitary transformation:  

\[
\begin{bmatrix}
\begin{array}{c}
\uparrow
\end{array}
\end{bmatrix} = |0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 1|.
\]

The triangle may be written in terms of red and green spiders as:
We also use multiple lemmas introduced in the work [JPV19]:

Lemma 2.4.

\[ \otimes 2 = \begin{array}{c} \alpha + \beta \end{array} = \begin{array}{c} \alpha \end{array} \]

Lemma 2.5.

\[ \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.6.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.7.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.8.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.9.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.10.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.11.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.12.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.13.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.14.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.15.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.16.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.17.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.18.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

and

Lemma 2.19.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]

Lemma 2.20.

\[ \begin{array}{c} \alpha \beta \end{array} = \begin{array}{c} \alpha \beta \end{array} \]
Lemma 2.21.

\begin{align*}
\pi & = \pi \\
\beta & = \beta
\end{align*}

Lemma 2.22.

\begin{align*}
\pi & = \pi \\
\beta & = \beta
\end{align*}

We also introduce some new lemmas that will be useful in the following sections. The proofs serve as an illustration of the graphical computation process.

Lemma 2.23.

\begin{align*}
\pi & = \pi \\
\beta & = \beta
\end{align*}

Lemma 2.24.

\begin{align*}
\pi & = \pi \\
\beta & = \beta
\end{align*}

Lemma 2.25.

\begin{align*}
\pi & = \pi \\
\beta & = \beta
\end{align*}

Proof (lemma 2.23).

\begin{align*}
\pi & = \pi \\
\beta & = \beta
\end{align*}

(2.3)

\Box

Proof (lemma 2.25). The left hand side is:

\begin{align*}
\pi & = \pi \\
\beta & = \beta
\end{align*}

(2.4)
For the right hand side we get:

\[
\begin{align*}
\pi &= (B1)_{2.10} \\
\pi &= 2.6 \\
\pi &= 2.6_{2.11} \\
\pi &= 2.12
\end{align*}
\]  

(2.5)

Proof (lemma 2.24). To prove the lemma we adopt a slightly different technique than a usual sequence of rewrites. The technique consists in verification that the diagrams from the left hand side and the right hand side lead to the same result when applied on a set of basis states \(\{|0\rangle, |1\rangle\}\) corresponding to diagrams \(\bigcirc\) and \(\bigotimes\). As was pointed out in [CK17] for two matrices \(M_1, M_2 : \mathbb{C}^2 \rightarrow \mathbb{C}^{2^m}\) and a basis \(\{|b_1\rangle, \ldots, |b_{2^n}\rangle\}\) in \(\mathbb{C}^{2^n}\) the equality on basis input is equivalent to the equality of matrices, i.e.

\[
\forall i \in [1, \ldots, 2^n] : \ M_1 |b_i\rangle = M_2 |b_i\rangle \iff M_1 \equiv M_2
\]  

(2.6)

Therefore, by demonstrating that \(\text{lhs} = \text{rhs}\) and \(\text{lhs} \pi = \text{rhs} \pi\) we actually establish that

\[
\begin{bmatrix}
\text{lhs} \\
\vdots
\end{bmatrix} = \begin{bmatrix}
\text{rhs} \\
\vdots
\end{bmatrix}.
\]

(2.7)

(2.8)

The lemma condition follows from the completeness of ZX-calculus.

\[\square\]

2.4. Linear diagrams. In ZX-diagrams the angles inside spiders are real numbers or elements of some group \(\mathcal{G}\). However, in many equations such as (S1), (K), (H), and (SUP) we used symbols \(\alpha\) and \(\beta\) instead of numbers. The symbolic notation was extremely handy to define families of equalities, i.e. equalities that hold for any real values of angles. By explicitly keeping symbols inside spiders we can draw parameterized diagrams. A parameterized diagram with \(n\) different symbols can be interpreted as a function that associates to each element of the space \(\mathbb{R}^n\) a well-defined ZX-diagram.
More precisely, we say that a diagram $D$ is parametrized by $\beta_1, \ldots, \beta_k$ if its angles are some functions on $\beta_1, \ldots, \beta_k$. We denote such a diagram by $D(\beta_1, \ldots, \beta_k)$. It is possible to evaluate a parametrized diagram $D(\beta)$, $\beta \in \mathbb{R}^k$ in a point $\beta$ by replacing every occurrence of $\beta_i$ with the respective value $\beta_i$. The result of the evaluation is a diagram $D(\bar{\beta})$ from $\mathbb{R}^k$.

Among parameterized diagrams, we distinguish a family of linear diagrams denoted $ZX(\beta)$:

**Definition 2.26 (Linear diagrams [JPV18b]).** A ZX-diagram is linear in $\beta_1, \ldots, \beta_k$ with constants in $L \subset \mathbb{R}$ if it is generated by $\otimes, \mathfrak{1}, \frac{1}{\sqrt{2}}, \mathfrak{I}, \bigcup, \cap$ combined by tensor product and composition such that $\alpha$ is of the form $\sum_i n_i \beta_i + c$ with $n_i \in \mathbb{Z}$ and $c \in L$.

In other words, in a parameterized linear diagram $D = D(\beta_1, \ldots, \beta_k)$ each angle $\alpha$ of a (red/green) spider is of the form $\sum_i n_i \beta_i + c$ for some integer $n_i$ and $c \in L$.

It was shown in [JPV18b] that for $L = \{\frac{n\pi}{4}\}_{n \in \mathbb{Z}}$ the Clifford+T axiomatization (Fig. 1) is complete for linear diagrams.

The family of linear diagrams may appear restricted compared to ZX-diagrams that allow angles from a more general class of functions. It is, however, sufficient for applications in variational quantum algorithms as they use circuits where parameters appear in a linear fashion [Pre18]. More importantly, for this family we are able to demonstrate simple formulas for the derivative. We believe that such formulas are not obtainable even for a slightly more general fragment $ZX_{A_n}$ where angles belong to the group of affine functions $A_n = \{(\beta) \mapsto c^T \beta + c_0 | c \in \mathbb{R}^n, c_0 \in \mathbb{R}\}$ (i.e. where coefficients are real numbers). Intuitively, the difficulty comes from the absence of a simple representation of a general matrix over real numbers in terms of spiders. This restriction is removed in the algebraic ZX-calculus [Wan20] at the cost of an extended set of generators.

3. Related work

In this section we provide a broader context for addition and differentiation in quantum computing. In particular, we discuss how the addition of two unitary matrices may appear in quantum applications and how it can be expressed in the circuit notation [CW12]. We also review other attempts to manipulate sums of ZX-diagrams.

The differentiation in quantum computing appears in the context of variational algorithms. We review how derivatives of ZX-diagrams may be helpful for the analysis of these algorithms, in particular in the detection of barren plateaus and in the design of parameter shift rules. These ideas point out possible directions for future work.

3.1. Addition in quantum computing.

**Addition with circuits.** In pure qubit quantum mechanics computations are performed by unitary transformations. Unitary transformations form a group with respect to multiplication, but the sum of two unitary transformations $U_a$ and $U_b$ is not unitary.

However, in some context it is meaningful to consider transformations of the form $\frac{1}{2} (U_a + U_b)$. For instance, the work [CW12] suggests to approximate the evolution of a quantum state under some Hamiltonian by a linear combination of unitary operators. It was
shown that such an approach typically outperforms the traditional technique for simulation based on Trotterization.

The protocol for the addition of the unitaries $U_a$ and $U_b$ in [CW12] is:

$q_a : |0\rangle \rightarrow H \cdot X \cdot X \cdot H \cdot U_a \cdot U_b \cdot |b\rangle$ \hspace{1cm} (3.1)

where $q_a$ is an ancilla qubit initialized in a zero-state, and we have access to the controlled-version of the unitary gates $U_a$ and $U_b$.

The protocol (3.1) can be easily verified. Starting in the state $|0\rangle|\psi\rangle$ we obtain:

$|0\rangle|\psi\rangle \rightarrow \frac{|0\rangle + |1\rangle}{\sqrt{2}}|\psi\rangle \rightarrow \frac{1}{\sqrt{2}} (|0\rangle \otimes U_b|\psi\rangle + |1\rangle \otimes U_a|\psi\rangle)$ \hspace{1cm} (3.2)

$\rightarrow \frac{1}{c} (|0\rangle \otimes (U_b + U_a)|\psi\rangle + |1\rangle \otimes (U_b - U_a)|\psi\rangle)$ \hspace{1cm} (3.3)

If the ancilla qubit is measured in the $|0\rangle$ state, the input state $|\psi\rangle$ was successfully transformed to $\frac{1}{c}(U_a + U_b)|\psi\rangle$ ($c$ is the normalization constant).

We remark that protocol (3.1) uses an ancilla qubit and post-selection on the measurement output and, therefore, exceeds the framework of pure qubit computations. We also emphasize that the circuit involves controlled versions of unitary transformations. In order to be executed on a quantum computer, the controlled unitaries have to be compiled into a sequence of elementary gates. An efficient compilation is non-trivial even for a simple controlled-controlled-not gate usually called Toffoli gate. Interestingly, the use of graphical representations may enhance the compilation of controlled gates [KvdWK19].

**Addition with the ZX-Calculus.** In the context of the ZX-calculus, there is no natural way to perform a linear combination of arbitrary ZX-diagrams. Fundamentally, this deficiency is due to the absence of a proper physical interpretation for a sum of diagrams. Indeed, ZX-calculus is a language for a process theory representation of quantum computing - i.e. each diagram is a process [CK17]. The composition of processes can be interpreted as their sequential application, the tensor product corresponds to the parallel application, but the addition doesn’t have a meaningful physical interpretation.

Several attempts were made to address this issue. The first option selected in [TYd21, SH22] consists in an extension of the formalism from diagrams to bags of diagrams, i.e. formal sums of diagrams. The use of formal sums means computations and diagram simplifications cannot be done purely in the context of diagrams, but also require basic calculus.

Independently from our result, [YWS] suggested an alternative protocol for the addition of two ZX-diagrams. This article uses the algebraic ZXW-calculus instead of vanilla ZX-calculus.

There are some similarities with our work, in the sense that their result can also be seen as based on a translation to a controlled form. However, their definition of a controlled form is not equivalent to ours. In addition, instead of inductive procedure, they rely on a decomposition of a diagram on elementary matrices that have an exponential complexity in the worst-case. A more detailed comparison of the two approaches is presented in Section 4.3.
3.2. Differentiation in variational algorithms. In quantum computing a variational algorithm (VQA) is defined by a parameterized quantum circuit (PQC) (also called ansatz) and a loss function that helps to discriminate between the different values of the parameters. Usually, the loss function is given as an expectation of an energy operator in the state prepared by PQC. Such expectation can be represented as a scalar ZX-diagram with some angles depending on the parameters.

Many state-of-the-art methods for the optimization of the loss function such as Quantum Analytic Descent [KB22] and meta-learning optimizers [WSW+21] use derivatives [GS17]. For efficient training it is important to detect deficiencies in the parameter landscape. In particular, gradient-based methods fail to converge to an optimal solution in the presence of so-called barren plateaus - zones with exponentially vanishing gradient values. It was shown that some specific structure of the cost function [CSV+21] or, alternatively, the structure of the ansatz itself [MBS+18] may lead to a landscape containing such zones.

As was demonstrated in [ZG21], the diagrammatic representation is beneficial in the analysis of the barren plateau phenomena. Yet the work [ZG21] pointed out a crucial obstacle limiting the application of ZX-calculus to variational algorithms - notably the absence of convenient tools for the addition and differentiation of parametrized diagrams. Due to this obstacle, in [ZG21] the analysis was limited to diagrams with only two occurrences of the parameter. In [TYd21] an arbitrary number of parameter occurrences is tackled with explicitly represented product rules leading to bags of diagrams.

Other than in the analysis of the barren plateau, diagrammatic representation can be used to design parameter shift rules.

Parameter shift rules. Parameter shift rules appeared as a solution to the problem of gradient evaluation. Indeed, the gradient of the loss function \( \frac{\partial}{\partial \alpha_i} \langle \psi_0 | U(\alpha) O U(\alpha) | \psi_0 \rangle \) (where \( O \) is the energy operator) in general can’t be directly evaluated on quantum hardware. However, in some cases the gradient can be expressed as a linear combination of the loss function values at different points:

\[
\frac{\partial L}{\partial \alpha_i} (\tilde{\alpha}) = \sum_{k=1}^{m} \epsilon_k L(\alpha^k) \tag{3.4}
\]

where \( \alpha^k_i = \begin{cases} \tilde{\alpha}_i, & \forall i \neq \iota^* \\ \tilde{\alpha}_i + \phi_i, & i = \iota^* \end{cases} \). The values \( L(\alpha^k) \) can be computed with stochastic approximation.

Decompositions of the form (3.4) are called parameter shift rules. We remark that the parameter shift rules lead to an unbiased estimator for the gradient compared to the finite difference approximation \( \frac{\partial L}{\partial \alpha} (\alpha_0) = \frac{L(\alpha_0 + \delta) - L(\alpha_0)}{\delta} \) which, in turn, enhances the parameter optimization process [MBK21].

The first shift rule was derived for the case \( U(\alpha) = W e^{i\alpha H} V \) where \( W \) and \( V \) don’t depend on \( \alpha \), while \( H \) (called generator) has two eigenvalues \( +1 \) and \( -1 \) [MNKF18]. It was further extended to generators with arbitrary symmetric eigenvalues \( +r \) and \( -r \) [SBG+19]:

\[
\frac{\partial L}{\partial \alpha} (\alpha) = \frac{r}{4} \left( L \left( \alpha + \frac{\pi}{4r} \right) - L \left( \alpha - \frac{\pi}{4r} \right) \right) \tag{3.5}
\]
For more complex $U(\alpha)$ we can decompose $U(\alpha)$ on a product of elementary matrices $U(\alpha) = E_1(\alpha) \times \cdots \times E_M(\alpha)$ and use product rules. Alternatively, more sophisticated techniques [WIWL22, KE21, ILY21] involve a spectral decomposition of the generator $H$. These advanced rules imply a smaller number of elements in the linear combination (3.4) compared to the decomposition, leading to an economy in the number of calls made to the quantum computer.

We remark that simply having a ZX-diagram for the derivative of the loss function is not enough to design new shift rules. We need to further extend the toolbox of ZX-calculus, for instance, we may need a procedure that compiles a diagram into a linear combination of other diagrams corresponding to valid quantum expectations. This extension constitutes one of the possible directions for future work.

4. Addition of ZX-diagrams

The addition is a natural operation on matrices from a Hilbert space. However, it doesn’t correspond to a physical process, hence it is not reflected in the standard ZX-calculus [CK17].

It is not hard to show that it is not possible directly, from two diagrams $D_1$ and $D_2$ corresponding to two matrices $M_1$ and $M_2$, to obtain a diagram corresponding to the sum $M_1 + M_2$ by just plugging $D_1$ and $D_2$ into a bigger diagram. Indeed, any diagram that contains a subdiagram representing the 0 matrix also represents a 0 matrix. So if $D_1$ represents a 0 matrix and $D_2$ represents a nonzero matrix, then plugging $D_1$ and $D_2$ into a bigger diagram will give a zero matrix, and therefore will not represent the sum of the two matrices.

On the other hand, for any two diagrams $D_1, D_2 : n \to m$, the universality of the ZX-calculus guarantees that there exists a diagram $D : n \to m$ such that $[D] = [D_1] + [D_2]$.

We provide in this section a general construction for such a diagram.

The key ingredient is to use controlled versions of the original diagrams, rather than the original diagrams. As pointed out in [JPV19], one can inductively define the addition on ‘controlled’ versions of the diagrams. A controlled version of a diagram $D_0$ is roughly speaking a diagram with an extra input such that when this extra input is set to $|1\rangle$ the diagram behaves as $D_0$ and when it is set to $|0\rangle$ the diagram behaves as a neutral diagram. Roughly speaking, the addition procedure puts controlled versions of two diagrams aside and plugs a diagram corresponding to $\text{const}([|10\rangle + |01\rangle])$ to the inputs.

To build a controlled version of a diagram our construction uses controlled states originally introduced in [JPV19].

4.1. Controlled states.

Definition 4.1 (Controlled state [JPV19]). A ZX-diagram $C : 1 \to n$ is a controlled state if $\langle C \mid 0 \rangle = \sum_{x \in \{0,1\}^n} |x\rangle = \left[ \begin{array}{c} \cdots \cdots \cdots \\ n \end{array} \right]$. Intuitively, a controlled state is a way to encode the state $\langle C \mid 1 \rangle$. A controlled state with no outputs is called controlled scalar. For instance, the scalar 0 could be encoded by the controlled scalar $C_0 = \left[ \begin{array}{c} \cdots \cdots \cdots \\ n \end{array} \right]$. Indeed, we can explicitly check that $\langle C_0 \mid 0 \rangle = 1$ and
\[ [C_0] |1\rangle = 0: \]
\[
\begin{align*}
\begin{bmatrix}
\text{\textbullet} & \text{\textbullet} \\
\text{\textbullet} & \text{\textbullet}
\end{bmatrix}_{2,4} & = \begin{bmatrix} 1 \end{bmatrix} = 1, \\
\begin{bmatrix}
\text{\textbullet} & \text{\textbullet} & \text{\textbullet} & \text{\textbullet} \\
\text{\textbullet} & \text{\textbullet} & \text{\textbullet} & \text{\textbullet}
\end{bmatrix} & = \begin{bmatrix} \otimes^2 \end{bmatrix} \times \begin{bmatrix}
\text{\textbullet} \\
\text{\textbullet}
\end{bmatrix}_{2.1} = 0
\end{align*}
\]

It will be more useful in what follows to perform \textit{graphical proofs} instead of matrix computation to manipulate controlled states. With these rules, we verify the condition \[ [C] |0\rangle = \sum_{x \in \{0,1\}^n} |x\rangle \] by checking if \[ ZX \vdash C. . . \not= C. . . ] \[ n \). The diagram corresponding to the encoded state is simply \[ \begin{bmatrix} D \end{bmatrix} = \begin{bmatrix} C \end{bmatrix}. \]

**Example 4.2** (Proven in [JPV19]). Diagrams \( C_{1/\sqrt{2}} = \) and \( C_2 = \) are controlled scalars:

\[
\begin{align*}
\begin{bmatrix} C_{1/\sqrt{2}} \end{bmatrix} & = \begin{bmatrix} \text{\textbullet} & \text{\textbullet} \\
\text{\textbullet} & \text{\textbullet}
\end{bmatrix}, & \begin{bmatrix} C_2 \end{bmatrix} & = \begin{bmatrix} \text{\textbullet} \\
\text{\textbullet}
\end{bmatrix}
\end{align*}
\]

representing respectively \( \frac{1}{\sqrt{2}} \) and 2.

Controlled states have nice properties that allows to perform element-wise addition and tensor product of corresponding vectors:

**Lemma 4.3** (Sum and tensor product [JPV19]). For any controlled states \( C_1, C_2 : 1 \rightarrow n \) and \( C_3 : 1 \rightarrow m \) the diagrams:

\[
\begin{align*}
C_+ & = C_1 + C_2, \quad \text{and} \quad C_\otimes & = C_1 \otimes C_3
\end{align*}
\]

are controlled states such that \[ [C_+] |1\rangle = [C_1] |1\rangle + [C_2] |1\rangle \] and \[ [C_\otimes] |1\rangle = [C_1] |1\rangle \otimes [C_3] |1\rangle. \]

The previous lemma can be used to add two controlled states. More precisely, if one of the controlled states encodes a state diagram \( D_1 : 0 \rightarrow n \) and the second controlled state encodes \( D_2 : 0 \rightarrow n \) the diagram \[ D_+ = C_1 + C_2 \] correspond to the vector \[ [D_1] + [D_2]. \] Moreover, as the construction returns a controlled state we can directly proceed to the addition of new
terms: for $N$ controlled versions $C_1, \ldots, C_N$ of $D_1, \ldots, D_N$ the diagram:

$$D = \begin{array}{cccc}
C_1 & C_2 & C_3 & \cdots & C_N \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
\end{array} \tag{4.4}$$

correspond to the state $[D] = [D_1] + \cdots + [D_N]$.

We remark that for each number of output wires $n$ the zero-element $C^0_n : 1 \rightarrow n$ is a controlled state. Indeed, we can prove the following lemma:

**Lemma 4.4.** For any controlled state $C : 1 \rightarrow n$

$$C \circ C^m_0 = \begin{array}{cccc}
C & C^m_0 \\
\cdots & \cdots \\
\end{array} \quad \text{and} \quad C \circ C^m_0 = C^{m+m}_0 \tag{4.5}$$

*Proof.* We start by proving the first equality. The left-hand side is:

$$C \circ C^m_0 = \begin{array}{cccc}
C & \cdots & \cdots & \cdots \\
\cdots & \cdots & \cdots & \cdots \\
\end{array} \tag{4.6}$$
The top part of the diagram can be transformed as follows:

\[ \begin{array}{c}
\includegraphics{diagram1} \\
\includegraphics{diagram2} \\
\includegraphics{diagram3}
\end{array} \]

The statement follows from (S2) and (S1).

For the second equality we have:

\[ \begin{array}{c}
\includegraphics{diagram4} \\
\includegraphics{diagram5}
\end{array} \]

The lemma follows from the definition of the controlled state.

We follow in this work the definition of a controlled state in [JPV19]. There exist other definitions, see e.g. [YWS], which essentially differ in the output of the zero-case.

4.2. **Controlizer.** The lemma 4.3 provides a way to obtain the sum of two diagrams in a controlled state form.

The definition and the lemma technically allow only to solve the problem for states rather than arbitrary matrices, but notice that going from a matrix \( n \to m \) to a state \( 0 \to n + m \) is just a matter of bending wires. We remark that bending wires alone can’t lead to a procedure for addition, as there is no obvious way to directly sum state diagrams without controlled versions.

What remains to be done is to explain how to compute algorithmically, from a state (diagram) \( D \), a controlled version of the same state (diagram).

We introduce for this the notion of a controlizer - a map that associates diagrams with the corresponding controlled states.

**Notation 4.5.** \( ZX[n,m] \) denotes the family of all ZX-diagrams with \( n \) inputs and \( m \) outputs.

The formal definition of the controlizer is:

**Definition 4.6 (Controlizer).** We say that a map \( C : ZX[n,m] \to ZX[1,n+m] \) that associates to every diagram \( D : n \to m \) a diagram \( C(D) : 1 \to n + m \) is a controlizer if the following conditions hold for any ZX-diagram \( D \):

- \( C(D) \) is a controlled state
- The state represents \( D \) up to some scalars in the sense that:

\[ [D] = \begin{bmatrix}
\otimes n + m & \cdots \\
\cdots & C(D) \\
\cdots & \cdots \\
\cdots & \cdots \\
m & \cdots
\end{bmatrix} \]
Several maps satisfy the definition of the controller. For instance, the map presented in [JPV19] that associates to each diagram its normal form is in fact a controller. In [YWS] an alternative controlled state form is constructed from the $2^n$ coefficients of the algebraic normal form.

In our approach the controller is defined by induction over the composition and the tensor product and doesn’t rely on normal forms. It means that the shape of the controller will follow closely the shape of the original diagram. In terms of complexity, this also implies that the size of the controller output is polynomial in the size of the diagram.

**Definition 4.7 (Inductive controller).** We define the map $C : \text{ZX}[n, m] \to \text{ZX}[1, n+m]$ that associates to each diagram $D : n \to m$ a diagram $C(D) : 1 \to n+m$ as follows:

- For the generators $\otimes$, $\otimes$, $\oplus$, $\bigotimes$, $\bigoplus$, $\bigotimes'$:

  $C(\otimes) = \quad \bigotimes'$

- Generators $\bigotimes$ and $\bigoplus$ can be decomposed as follows using the above generators:

  $C(\bigotimes) = C(\bigotimes) = \quad C(\bigotimes) = \quad C(\bigotimes) = \quad C(\bigotimes)$

- For tensor product $D_\otimes = D_2 \otimes D_1$ and composition $D_\circ = D_3 \circ D_1$ where $D_1 : n \to m$, $D_2 : k \to l$ and $D_3 : m \to k$:

  $C(D_\otimes) = C(D_2) \quad C(D_3) \quad \bigotimes' \quad C(D_\circ) = C(D_3) \quad C(D_1)$

We remark that the output of the inductive controller is not unique as it depends on the decomposition order. For instance, two diagrams that are equivalent up to deformation (the topology matters rule) will produce different outputs, that are not equivalent up to deformation. However, the semantics is preserved: both representations will of course represent the same matrix.

We now verify that our inductive controller 4.7 satisfies the definition 4.6:
**Proof.** We prove the claim for the generators $\beta$, $\alpha$, $\cup$, $\cup$, $\cup$, $\cup$ and both compositions. The case of $\hdash\wr$ and $\hdash\otimes$ is a direct consequence of the axiom (S1).

First, we inductively prove that for all diagrams $D$ the diagram $C(D)$ is controlled state.

- **Generators:**

\[
\begin{align*}
C(\bullet) & \circ \ = \ 1 \ = \ (B1) \quad (4.12) \\
C(\downarrow) & \circ \ = \ 2 \ = \ (B1) \quad (4.13) \\
C(\uparrow) & \circ \ = \ 4 \ = \ (B1) \quad (4.14) \\
C(\bigcup) & \circ \ = \ 2 \ = \ (B1) \quad (4.15) \\
C(\bigcap) & \circ \ = \ 2 \ = \ (B1) \quad (4.16)
\end{align*}
\]

- **Tensor product:** it directly follows from lemma (4.3) that if $C(D_1)$ and $C(D_2)$ are controlled states, then the diagram $C(D_1 \otimes D_2)$ from (4.11) is controlled state.
• **Composition:** let’s assume that $C(D_1)$ and $C(D_2)$ are controlled states. We show that $C(D_1 \circ D_2)$ is also a controlled state:

\[
C(D_1 \circ D_2) \circ \begin{array}{c}
\ast
\end{array} = C(D_1) \circ \begin{array}{c}
\ast
\end{array} \circ C(D_2) = C(D_1) \circ \begin{array}{c}
\ast
\end{array} \circ C(D_2)
\]

\[
\cong (B1)
\]

\[
\cong 2.4
\]

(4.17)

Next we show that the map $C$ satisfy the property (4.9), i.e. it correctly encodes the input diagram.

• **Generators:**

\[
C \left( \begin{array}{c}
\ast
\end{array} \right) \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array}
\]

(4.18)

\[
C \left( \begin{array}{c}
\ast
\end{array} \right) \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array}
\]

(4.19)

\[
C \left( \begin{array}{c}
\ast
\end{array} \right) \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} = \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array} \circ \begin{array}{c}
\ast
\end{array}
\]

(4.20)
\[
C\left(\bigcirc\right) \circ 2 \pi = (B1) \quad \text{(B1)} \\
C\left(\bigotimes\right) \circ 2 \pi = (B1) \quad \text{(B1)}
\]

\[C\left(\bigotimes\right) \circ \pi = 2.8 \quad \text{(S2)}
\]

**Composition:** Let’s assume that the property (4.9) holds for diagrams \(D_1 : n \to m\) and \(D_2 : m \to k\). The controller for their composition \(D_1 \circ D_2\) is given by:

\[
\otimes n + k \quad \pi \quad \otimes n + m \quad \pi \quad \otimes k + m \quad \pi \quad \otimes 2m
\]

\[= 
\]

\[
\otimes n + k \quad \pi \quad \otimes n + m \quad \pi \quad \otimes k + m \quad \pi \quad \otimes 2m
\]

**Tensor product:** Let’s assume that the property (4.9) holds for diagrams \(D_1 : n \to m\) and \(D_2 : k \to l\). The controller for their tensor \(D_1 \circ D_2\) is given by:

\[
\otimes n + k \quad \pi \quad \otimes n + m \quad \pi \quad \otimes k + m \quad \pi \quad \otimes 2m
\]

\[= 
\]

\[
\otimes n + k \quad \pi \quad \otimes n + m \quad \pi \quad \otimes k + m \quad \pi \quad \otimes 2m
\]

\[= D_2 \circ D_1 \quad \text{(4.23)}
\]
The diagram (4.24) is equal to
\[ C(D_1) \otimes n + m + k + l = C(D_2), \]
by the induction assumption.

To give a flavor of our computation process we show how to obtain a controlled version of \( \pi \):\n
\[ C(\pi) = \frac{2}{2.11} \]

Addition of arbitrary diagrams. The controller allows to map every two ZX-diagrams \( D_1 : n \to m \) and \( D_2 : n \to m \) to controlled states \( C_1 : 1 \to n + m \) and \( C_2 : 1 \to n + m \). The lemma 4.3 provides a way to obtain a sum of resulting controlled states. By combining this two results we can recover the diagram for addition of \( D_1 \) and \( D_2 \). The addition protocol is resumed in the following theorem:

Theorem 4.8. For diagrams \( D_1 : n \to m \) and \( D_2 : n \to m \) the diagram
\[ D_+ = C_+ \otimes n + m \]

where \( C_+ = C(D_1) \otimes \ldots \otimes C(D_2) \).

is such that \([D_+] = [D_1] + [D_2] \).

Proof (theorem 4.8). The theorem follows from the definition of the controller and the lemma 4.3.

We illustrate the diagrammatic addition with a simple example.

**Example 4.9.** Using 4.8, we construct a diagram \( D \) corresponding to the addition of \( \bigotimes \) and \( \otimes \).

\[
D = \begin{array}{c}
\otimes n + m \\
\bigotimes
\end{array}
\]

In this example we simplified the diagram with graphical rewriting to obtain a known result:

\[
[\bigotimes] + [\otimes] = (|00\rangle + |11\rangle) + (|01\rangle + |10\rangle) = 2|+\rangle = \begin{array}{c}
\otimes
\end{array}
\]

4.3. Discussion. The previous example shows that our inductive procedure leads to large diagrams, even when the initial diagrams are fairly simple. The resulting diagrams roughly follow the shape of the initial diagrams, but are still quite different. This defect is however consistent with intuition. Indeed, contrary to sequential and parallel compositions, the addition is not a physical operation, hence it is not surprising that can’t be easily incorporated into the framework of ZX-diagrams. On the positive side, we can rely on the powerful equation theory of the ZX-calculus to simplify, when it is possible, the diagrams representing the sum of diagrams.

An alternative approach for the addition in the algebraic ZWX-calculus was presented in [YWS]. It also relies on the controlled forms of the diagram, but their controlled forms are different from ours. Notably, contrary to our approach that accepts diagrams with an arbitrary number of input and output wires, the controlled forms in [YWS] are defined only for two families of diagrams: state diagrams \( \text{ZX}[0, n] \) and diagrams \( \text{ZX}[m, m] \) that correspond
to square matrices. Controlled matrices [YWS] are built from the decomposition of the initial diagram \(D\) on a product of diagrams for so-called elementary matrices [WY21]. In the general case, the decomposition on elementary matrices seems to require an explicit computation of the matrix corresponding to the diagram. Therefore, although for given controlled forms the addition from [YWS] is done diagrammatically with the help of \(W\)-spider, the computation of the controlled forms uses the semantics. In contrast, in our inductive procedure the entire process is maintained in the graphical framework.

However, in some special cases of practical importance the approach suggested in [YWS] is beneficial. For instance, as was shown in [YWS] that for local Hamiltonians made out of Pauli tensors the decomposition on elementary matrices is straightforward. As a consequence, the controlled versions are easy to compute and the overall procedure leads to better-looking diagrams.

5. Differentiation of ZX-diagrams

A basic definition of the derivative is the derivative for a smooth function \(f : \mathbb{R} \to \mathbb{R}\):

\[
\frac{\partial f}{\partial x}(x_0) = \lim_{\delta \to 0} \frac{f(x_0 + \delta) - f(x_0)}{\delta}
\]  

(5.1)

In traditional calculus the derivative usually is computed from the decomposition of the function \(f : \mathbb{R} \to \mathbb{R}\) on a sum, product, and composition of elementary functions. Derivatives for compositions of function functions are computed using linearity and the product rules:

\[\partial(f \circ g) = f \circ \partial g + \partial f \circ g\]

and similarly for the tensor product.

A parameterized ZX-diagram \(D(\beta)\) can be understood as a function \(f : \mathbb{R} \to \text{ZX}\) that associates to each real value \(\beta \in \mathbb{R}\) a ZX-diagram. In the context of variational algorithms, we consider the derivatives over the parameters.

In order to keep the notations simple in what follows we restrict our attention to the set of linear diagrams over one variable, denoted by \(\text{ZX}(\beta)\). We denote the corresponding matrices by \(\mathcal{M}(\beta)\). The elements of matrices in \(\mathcal{M}(\beta)\) belong to the ring generated by complex numbers and the smooth functions \(e^{iA}\) where \(A = \{k\beta \mid k \in \mathbb{Z}\}\) is the group of affine functions with integer coefficients. We remark that even if we restrict the consideration to diagrams over one variable, all results are easily extended to the case of partial derivatives \(\partial_{\beta_i}\) for linear diagrams with an arbitrary number of variables mutatis mutandis.

The sum for the matrices in \(\mathcal{M}(\beta)\) is naturally defined by entry-wise addition. Therefore, we can define the derivative in \(\mathcal{M}(\beta)\) by specifying the general definition for monoidal categories with sums given in [TYd21]:

**Definition 5.1 (Derivative in \(\mathcal{M}(\beta)\)).** A derivative \(\partial_M\) in \(\mathcal{M}(\beta)\) is a linear unary operator on \(\mathcal{M}(\beta)\) that associates to a matrix \(L \in \mathcal{M}(\beta)\), \(L : n \to m\) another matrix \(L' \in \mathcal{M}(\beta)\), \(L' : n \to m\) of the same dimensions. It satisfies the following axioms:

- \(\circ\)-product rule: \(\partial_M[A \circ B] = \partial_M[A] \circ B + A \circ \partial_M[B]\)
- \(\otimes\)-product rule: \(\partial_M[A \otimes B] = \partial_M[A] \otimes B + A \otimes \partial_M[B]\)

Rather than defining the derivative as a limit of finite difference and deriving the product rules, the work [TYd21] suggests an axiomatic definition that proclaims product rules as the fundamental property of the derivative operator. We remark that the entry-wise differentiation of matrix elements satisfies this definition.

Although our theorem 4.8 provides a fully diagrammatic way for the addition of diagrams, we avoid extending the language with a formal introduction of a sum operator in order
to circumvent unnecessary complications. Therefore, the general definition from [TYd21] doesn’t apply. Instead, we suggest an alternative semantics for the derivative in the ZX-calculus requiring the coherence between the derivative of a diagram and the derivative of the corresponding linear map:

**Definition 5.2 (ZX-derivative).** A derivative \( \partial_{ZX} : ZX[n,m] \rightarrow ZX[n,m] \) is a unary operator that commutes with the standard interpretation:

\[
[\partial_{ZX} D] = \partial_M [D]
\]  

(5.2)

where \( \partial_M \) is an (arbitrary) fixed derivative in \( \mathcal{M}(\beta) \) satisfying the definition (5.1).

The property (5.2) is called **diagramatic differentiation.**

5.1. **Diagrammatic differentiation with controlizers.** The derivative in \( \mathcal{M}(\beta) \) is defined through product rules that involve sums that will be translated to the diagrammatic framework using **controlizers.**

**Notation 5.3.** In what follows we denote by \( C \) any map that satisfies the definition 4.6 of controlizer, for instance the one of 4.7.

**Definition 5.4.** Given a controlizer \( C \), the \( C \)-derivative is the map \( \Delta : ZX(\beta) \rightarrow ZX(\beta) \) that associates to a parameterized diagram \( D : n \rightarrow m \) another diagram \( \Delta(D) : 1 \rightarrow n + m \) defined as follows:

- **Generators:** For parametrized spiders:

\[
\Delta \left( \begin{array}{c} \beta \\ \end{array} \right) = \begin{array}{c} \circ \\ \end{array}, \quad \Delta \left( \begin{array}{c} \beta \\ \end{array} \right) = \begin{array}{c} \circ \\ \end{array}
\]  \hfill (5.3)

\[
\Delta \left( \begin{array}{c} n \\ \ldots \\ m \\ \end{array} \right) = \Delta \left( \begin{array}{c} k \\ \ldots \\ n \\ \end{array} \right), \quad \Delta \left( \begin{array}{c} n \\ \ldots \\ m \\ \end{array} \right) = \Delta \left( \begin{array}{c} n \\ \ldots \\ m \\ \end{array} \right)
\]  \hfill (5.4)

For all other generators \( g : n \rightarrow m \) that do not contain an instance of the variable \( \beta \), we define \( \Delta(g) = \begin{array}{c} \circ \\ \end{array} \)

- **Tensor product:** for \( D_1 : n \rightarrow m \) and \( D_2 : k \rightarrow l \) the diagram \( \Delta(D_1 \otimes D_2) \) is:

\[
\Delta(D_1 \otimes D_2) = \Delta(D_1) \otimes C(D_2) \otimes C(D_2) \otimes \Delta(D_2)
\]  \hfill (5.5)
**Composition:** for $D_1 : n \rightarrow m$ and $D_2 : m \rightarrow k$ the diagram $\Delta(D_2 \circ D_1)$ is:

$$\Delta(D_2 \circ D_1) = \Delta(D_1) \otimes C(D_2) \otimes C(D_1) \otimes \Delta(D_2)$$ (5.6)

As should be clear from the definition, the $C$-derivative of a diagram is not derivative of a diagram. Rather, it corresponds to a controlled-version of the derivative.

A simple verification shows that for any generator $G$ the $C$-derivative $\Delta(G)$ is a controlled state. From lemma 4.3 it follows that for every diagram $D : n \rightarrow m$ the $C$-derivative $\Delta(D) : 1 \rightarrow n + m$ is a controlled state. Moreover, as was proven in lemma 4.4 the $C$-derivative $\Delta(G)$ of generators that are independent on $\beta$ is precisely the zero-element $C_0^{n+m}$ for the addition of controlled states. The same lemma implies, in particular, that the $C$-derivative of any diagram $D : n \rightarrow m$ that is independent on $\beta$ is $\Delta(G)$.

Similarly to controlizers, a step-by-step application of the map $\Delta$ may lead to different diagrams depending on the order of decomposition on tensor products and compositions. However, all possible outputs are semantically equivalent and, by the completeness of ZX-calculus, are equivalent as diagrams.

By combining $C$-derivatives and controlizers we can reproduce the semantics of product rules in a graphical representation:

**Definition 5.5.** Given a $C$-derivative $\Delta$, let $\partial_C : ZX(\beta) \rightarrow ZX(\beta)$ be the unary operator such that for any diagram $D : n \rightarrow m$

$$\partial_C[D] = \Delta(D)$$ (5.7)

**Theorem 5.6.** The operator $\partial_C$ satisfies the definition (5.2) of diagrammatic differentiation.

**Proof (theorem 5.6).** We give the proof by induction over tensor product and composition.

- **Generators:** First, we show that $\partial_M \begin{bmatrix} \begin{smallmatrix} 0 \end{smallmatrix} \end{bmatrix} = ie^{i\beta}|-\rangle$ and $\partial_M \begin{bmatrix} \begin{smallmatrix} -\beta \end{smallmatrix} \end{bmatrix} = -ie^{-i\beta}|-\rangle$.

Indeed, we have:

$$\partial_M \begin{bmatrix} \begin{smallmatrix} 0 \end{smallmatrix} \end{bmatrix} = \partial_M \left( |+\rangle + e^{i\beta}|-\rangle \right) = ie^{i\beta}|-\rangle$$ (5.8)

$$\partial_M \begin{bmatrix} \begin{smallmatrix} -\beta \end{smallmatrix} \end{bmatrix} = \partial_M \left( |+\rangle + e^{-i\beta}|-\rangle \right) = -ie^{-i\beta}|-\rangle$$ (5.9)
For the diagrams $\partial C (\emptyset)$ and $\partial C (-\beta)$ we verify the property of diagrammatic differentiation:

$$\left[\begin{array}{c}
\text{Diagram 1} \\
\text{Diagram 2}
\end{array}\right] = \left[\begin{array}{c}
\text{Diagram 3} \\
\text{Diagram 4}
\end{array}\right] = i e^{i \beta} | - \rangle$$

(5.10)

$$\left[\begin{array}{c}
\text{Diagram 5} \\
\text{Diagram 6}
\end{array}\right] = \left[\begin{array}{c}
\text{Diagram 7} \\
\text{Diagram 8}
\end{array}\right] = -i e^{-i \beta} | - \rangle$$

(5.11)

For a generator $g : n \to m$ that doesn’t depend on $\beta$:

$$[\partial_C(g)] = \left[\begin{array}{c}
\text{Diagram 9} \\
\text{Diagram 10}
\end{array}\right] = \left[\begin{array}{c}
\text{Diagram 11} \\
\text{Diagram 12}
\end{array}\right] = (S1) = (0)_{n \times m}$$

(5.12)

- **Tensor product**: Let’s assume that $[\partial_C(D_1)] = \partial_M [D_1]$ and $[\partial_C(D_2)] = \partial_M [D_2]$ for $D_1 : n \to m$ and $D_2 : l \to k$. For $D = D_1 \otimes D_2$:

$$[\partial_C(D)] = \left[\begin{array}{c}
\text{Diagram 13} \\
\text{Diagram 14}
\end{array}\right] = \left[\begin{array}{c}
\text{Diagram 15} \\
\text{Diagram 16}
\end{array}\right]$$

(5.13)

By applying the definition 5.4 of the $C$-derivative for the tensor product we obtain the diagram:

$$\left[\begin{array}{c}
\text{Diagram 17} \\
\text{Diagram 18}
\end{array}\right]$$

(5.14)
We denote by the part of the diagram surrounded by the red frame. According to the definition, $C$-derivatives $\Delta(D_1)$, $\Delta(D_2)$ and controlizer outputs $C(D_1)$, $C(D_2)$ are controlled states. Therefore, from lemma 4.3 we get:

\[
\begin{bmatrix}
\begin{array}{c}
T \\
\ldots \ldots \\
\eta + m \\
k + l
\end{array}
\end{bmatrix} = \begin{bmatrix}
\begin{array}{c}
C(D_1) \\
\ldots \\
m \\
l
\end{array}
\end{bmatrix} + \begin{bmatrix}
\begin{array}{c}
\Delta(D_1) \\
\ldots \\
m \\
l
\end{array}
\end{bmatrix} \quad (5.15)
\]

Meanwhile, by bending wires we see that the resting part of the diagram (5.14) is:

\[
B = \begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} + \begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} \quad (5.16)
\]

The sum in $M(\beta)$ is distributive over the composition. Therefore,

\[
\begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} + \begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} \quad (5.17)
\]

\[
\begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} + \begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} \quad (5.18)
\]

\[
\begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} + \begin{bmatrix}
\begin{array}{c}
\otimes \eta + m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} \quad (5.19)
\]

The claim follows from the induction assumption and the definition of controlizer.

- **Composition:** Following the same reasoning as for the tensor product we obtain for $D_1 : n \rightarrow m$ and $D_2 : m \rightarrow l$

\[
\partial_C[D_2 \circ D_1] = \begin{bmatrix}
\begin{array}{c}
\otimes \eta + k + 2m \\
\ldots \\
m \\
\end{array}
\end{bmatrix} \quad (5.20)
\]
where the diagram $T$ is exactly the same as in (5.15). On more time, by distributivity of the sum in $\mathcal{M}(\beta)$ over the composition we obtain

\begin{equation}
[\partial_C[D_2 \circ D_1]] = \text{diagram}
\end{equation}

(5.21)

\begin{equation}
+ \text{diagram}
\end{equation}

(5.22)

The claim follows from the definition of controllerizer and the induction assumption.

We consider a simple example to illustrate how to perform the inductive procedure.

**Example 5.7.** We apply the definition 5.5 to the simple scalar diagram $\pi \pi = \pi \pi$.

Notice that $C(\pi \otimes \pi) = \text{diagram}$, moreover $C(\pi \pi)$ was already found in the example 4.25. We obtain a diagram for $C(\pi \otimes \pi)$:

\begin{equation}
\text{diagram}
\end{equation}

(5.23)
We know from 4.4 that $\Delta \left( \begin{array}{c} A \end{array} \right) = \begin{array}{c} \bullet \end{array}$. By definition, $\Delta \left( \begin{array}{c} B \end{array} \right) = \begin{array}{c} \bullet \end{array}$.

5.2. Formula for derivatives in $\text{ZX}(\beta)$. Although perfectly correct, the differentiation procedure described above leads to very puzzling output even for small diagrams (see example 5.7). In this section we provide a simpler approach to obtain the derivative of a diagram in $\text{ZX}(\beta)$ written in a special form. We formalize it in definitions $\partial_{\text{ZX}}$ and $\partial_{\text{P}}$ of unary operators that satisfy the property of diagrammatic differentiation (5.2).

First, we remark that even in the inductive procedure the effort can be significantly reduced if we separate the parts that depend on the parameter from the rest of the diagram that is constant on $\beta$. Indeed, for constant diagrams $A : k \to l$ and $B : m \to k$ that do not depend on $\beta$, the $C$-derivative is the controlled version of the zero diagrams $C_{0}^{k+l}$ and $C_{0}^{m+k}$. Therefore, for a parameterized diagram $D(\beta) : n \to m$ the lemma 4.4 implies:

\[
\Delta(D(\beta) \otimes A) = \Delta(D(\beta)) \otimes C(A), \quad \Delta(D(\beta) \circ A) = \Delta(D(\beta)) \circ C(B). \tag{5.24}
\]

These expressions are significantly simpler that the general case (5.4) where both parts of the compositions depend on $\beta$.

We want to further explore this property to find simplified formulas for the derivative. Let’s denote by $X_{\beta}(n, m)$ the family of diagrams $\begin{array}{c} \begin{array}{c} \beta \end{array} \end{array}$ from $\text{ZX}(\beta)$. 
**Claim 5.8.** Using the rules of ZX-calculus, each diagram \(D(\beta)\) from \(ZX(\beta)\) may be transformed into the form

\[
D(\beta) = D_2^{\beta} D_1 = D_2^{\beta - \gamma} = D_2^{\beta - \gamma - \gamma}
\]

where \(n, m\) are some integer numbers and \(D_1, D_2\) are constant with respect to \(\beta\). We call diagrams in this form \(\beta\)-factored.

A rigorous demonstration of claim 5.8 may be found in [JPV18b]. It essentially follows from the (H) rule to transform any green spider into a red spider, the (S1) rule to separate the \(\beta\) part from the rest of the diagram, and the paradigm *Only topology matters* to rearrange the diagram.

The diagram \(X_{\beta}(n, m)\) has a particularly regular form, so we expect it to have a nice-looking derivative. We would like to develop the intuition from the semantics \(\partial M \left[ X_{\beta}(n, m) \right] \).

### 5.2.1. Intuition

Due to the form obtained in the previous claim, it remains to find the derivative of \(X_{\beta}(n, m)\) to obtain the desired results.

For reasons that will become more clear later, we first focus on a variant of \(X_{\beta}(n, m)\) with green spiders instead of red spiders.

We denote by \(|\beta\rangle = |0\rangle + e^{i\beta}|1\rangle\) the state \(\begin{bmatrix} \circ \end{bmatrix}\) and by \(|-\beta\rangle = |0\rangle + e^{-i\beta}|1\rangle\) the state \(\begin{bmatrix} \circ \end{bmatrix}\).

We also introduce the notation:

\[
|\beta(n, m)\rangle = |\beta\rangle^\otimes n \otimes |-\beta\rangle^\otimes m = \sum_{x \in \{0, 1\}^{n+m}} e^{i\beta(|x^+| - |x^-|)} |x^+\rangle \otimes |x^-\rangle
\]

where \(x^+ \in \{0, 1\}^n\) corresponds to the first \(n\) qubits that are in the state \(|\beta\rangle^\otimes n\) and \(x^- \in \{0, 1\}^m\) corresponds to the last \(m\) qubits that are in the state \(|-\beta\rangle^\otimes m\). The notation \(|x| : \{0, 1\}^k \to \mathbb{Z}_+\) stands for the Hamming weight of the binary string \(x\).

By applying entrywise differentiation to the vector \(|\beta(n, m)\rangle\), we obtain the following result:

\[
\partial M \left( |\beta(n, m)\rangle \right) = \sum_{x \in \{0, 1\}^{n+m}} i(|x^+| - |x^-|) e^{i\beta(|x^+| - |x^-|)} |x^+\rangle \otimes |x^-\rangle
\]

We observe that \(\partial M (|\beta(n, m)\rangle) = iM_\Delta |\beta(n, m)\rangle\) where \(M_\Delta : 2^{n+m} \to 2^{n+m}\) is a linear map such that \(M_\Delta : |x\rangle \mapsto (|x^+| - |x^-|)|x\rangle\).

Therefore, obtaining the derivative of \(|\beta(n, m)\rangle\) requires the ability to produce the matrix \(M_\Delta\).

The first step is finding a diagram for the controlled-triangle \(D_\lambda : 2 \to 2\). Semantically, we want the diagram \(D_\lambda\) to apply the triangle \(\uparrow\) to the second input if the first one is in the \(|1\rangle\) state and do nothing otherwise. We denote the matrix corresponding to \(D_\lambda\) by...
Crucially, when the matrix $M_{\lambda}$ is applied to a vector $|b\rangle \otimes (k|0\rangle + |1\rangle)$ for $b \in \{0, 1\}$ the coefficient $k$ is modified depending on the value of $b$:

$$|b\rangle \otimes (k|0\rangle + |1\rangle) \xrightarrow{M_{\lambda}} |b\rangle \otimes ((k + b)|0\rangle + |1\rangle)$$ (5.28)

Similarly, when the input vector is $|b\rangle \otimes (k|0\rangle - |1\rangle)$ the value of $b$ is subtracted from $k$:

$$|b\rangle \otimes (k|0\rangle + |1\rangle) \xrightarrow{M_{\lambda}} |b\rangle \otimes ((k - b)|0\rangle - |1\rangle)$$ (5.29)

We remark the matrix $M_{\lambda}$ is not norm-preserving, so it doesn’t correspond to a valid quantum gate. However, it doesn’t matter as ZX-diagrams are universal for general linear maps of dimensions $2^n \times 2^m$.

We prove that $D_{\lambda} = \Lambda$ by explicitly checking cases when the first input is in $|0\rangle$ and in $|1\rangle$ states.

For the $|0\rangle$-case we have:

$$= (B1) \quad = (H) \quad 2.13 \quad = \quad 2.14$$ (5.30)

When the first input is in the $|1\rangle$ state we get:

$$= (B1) \quad = (H) \quad 2.13 \quad = \quad 2.14 $$ (5.31)

It turns out that the matrix $M_{\Delta}$ can be written as an evolution in $n + m + 1$ space with one ancilla input followed by a projection on $n + m$ dimensional space. The ancilla input is used to accumulate the coefficient $\alpha_x$ of the basis state $|x\rangle$. We suggest a construction in which the coefficient $\alpha_x$ is recovered from the projection $\langle 0|\psi_a \rangle$ of the state $|\psi_a\rangle$ of the ancilla input.

To be exact, we design an iterative procedure that multiplies the input vector $|b_0, \ldots, b_{m+1}\rangle$ by the matrix $M_{\Delta}$. Initially, we set $|\psi_a\rangle = |1\rangle$. Then, for $i \in \{1, \ldots, n\}$ we apply the transformation $M_{\lambda}$ to $|b_i\rangle|\psi_a\rangle$ controlled by the input $i$. After $n$ applications the state $|\psi_a\rangle$ becomes $|\psi_a\rangle = (\sum_{i=1}^{n} b_i) |0\rangle + |1\rangle$. To proceed further we apply $Z$ matrix to $|\psi_a\rangle$ and obtain $|\psi_a\rangle = (\sum_{i=1}^{n} b_i) |0\rangle - |1\rangle$. Then the state $|\psi_a\rangle$ is transformed to $|\psi_a\rangle = (\sum_{i=1}^{n} b_i - \sum_{i=n+1}^{m} b_i) |0\rangle - |1\rangle$ by iterative application of $M_{\lambda}$ to $|b_i\rangle|\psi_a\rangle$. Finally, the
coefficient $\alpha_x$ is:

$$\alpha_x = \langle 0 | \psi_a \rangle = \left( \sum_{i=1}^{n} b_i - \sum_{i=n+1}^{m} b_i \right)$$  \hspace{1cm} (5.32)$$

As the original input wires are used only in control, the output of such a procedure is $\alpha_x | b_0, \ldots, b_{n+m} \rangle$.

This procedure can be directly translated in a diagram $D_\lambda$:

The diagram for the derivative of $X_\beta(n, m) = \cdots \cdots \cdots$ can be obtained by conjugating every input with Hadamard boxes:

$$\partial M \left[ \begin{array}{c} \cdots \cdots \cdots \end{array} \right] = \partial M \left[ \begin{array}{c} \cdots \cdots \cdots \end{array} \right] = \left[ \begin{array}{c} \cdots \cdots \cdots \end{array} \right] \circ \partial M \left[ \begin{array}{c} \cdots \cdots \cdots \end{array} \right]$$  \hspace{1cm} (5.34)$$
By applying the rule (H) to the final we obtain

$$\partial_{M} [X_{\beta}(n,m)] = \left[\begin{array}{c}
\overset{\otimes 3}{\beta} \\
\otimes n + m \\
\end{array}\right] \overset{n}{\overset{m}{\overrightarrow{\ldots}}} \left[\begin{array}{c}
\overset{\otimes 3}{\beta} \\
\otimes n + m \\
\end{array}\right] (5.36)$$

Notice that the Hadamard boxes from $D_{\lambda}$ have disappeared in the expression of $\partial_{M} [X_{\beta}(n,m)]$, which explains why we chose initially red spiders instead of green spiders.

5.2.2. Diagrammatic differentiation of $\beta$-factored forms. Motivated by the simplicity of the result (5.36), we suggest an alternative non-inductive definition for derivatives of diagrams in $\beta$-factored forms:

**Definition 5.9.** Given a diagram $D(\beta) = D_{2} \circ (D_{1} \otimes X_{\beta}(n,m))$ in $\beta$-factored form, let $\partial_{ZX}[D] = D_{2} \circ (D_{1} \otimes \partial_{ZX}[X_{\beta}(n,m)])$ where

$$\partial_{ZX}[X_{\beta}(n,m)] = \partial_{ZX} \left[\begin{array}{c}
\overset{\otimes 3}{\beta} \\
\otimes n + m \\
\end{array}\right] (5.37)$$

$$= \left[\begin{array}{c}
\overset{\otimes 3}{\beta} \\
\otimes n + m \\
\end{array}\right] (5.38)$$

Although our definition is inspired by the intuitive reflection presented above, we provide a rigorous diagrammatic demonstration of the soundness of our definition. More precisely, we prove the following theorem:

**Theorem 5.10.** The operator $\partial_{ZX}[\cdot]$ defined at (5.9) satisfies the property of diagrammatic differentiation: for any diagram $D(\beta) \in ZX(\beta)$ in $\beta$-factored form,

$$[[\partial_{ZX}D(\beta)] = \partial_{M} [[D(\beta)]] (5.39)$$

We remark that according to the definition 5.2 the derivative $D' : n \rightarrow m$ of a diagram $D : n \rightarrow m$ that is constant on $\beta$ is such that $[D'] = \partial_{M} [[D]] = (0)_{n \times m}$. Therefore, theorem 5.10 is a direct consequence of the following lemma:

**Lemma 5.11.** For any $n, m$,

$$[[\partial_{ZX}X_{\beta}(n,m)]] = \partial_{M} [[X_{\beta}(n,m)]] (5.40)$$

**Proof.** We prove the lemma by induction. We provide the demonstration for $n = n + 1$, the proof for $m = m + 1$ is directly obtainable in the same way.

**Base:** We already know that $\partial_{M} [[\overset{\otimes 3}{\beta}]] = i e^{i \beta} [\cdot]$. Thus, for the case $n = 1, m = 0$ the lemma statement follows from:

$$[[\partial_{ZX}X_{\beta}(1,0)]] = \left[\begin{array}{c}
\overset{\otimes 3}{\beta} \\
\otimes n + m \\
\end{array}\right] = \left[\begin{array}{c}
\overset{\otimes 3}{\beta} \\
\otimes n + m \\
\end{array}\right] = i e^{i \beta} [\cdot] (5.41)$$
Step: By induction, we assume that the equation (5.40) holds for some \( n \) and \( m \). We show that under this assumption \( \partial \mathbf{M} \left[ X_\beta(n + 1, m) \right] = \partial_M \left[ X_\beta(n + 1, m) \right] \).

In order to prove the induction we will proceed in three steps. Firstly we apply the product rule to the matrix \( \partial \mathbf{M} \left[ X_\beta(n + 1, m) \right] \):

\[
\partial \mathbf{M} \left[ X_\beta(n + 1, m) \right] = \partial \mathbf{M} \left[ X_\beta(1, 0) \right] \otimes \left[ X_\beta(n, m) \right] + \left[ X_\beta(1, 0) \right] \otimes \partial_M \left[ X_\beta(n, m) \right]
\]

(5.42)

where the second equality follows from the inductive assumption.

Secondly, we show the last sum in the equation (5.42) can be represented diagrammatically with a controlled state, i.e. the following claim holds:

**Claim 5.12.** We can find a controlled state \( \tilde{X} : 1 \rightarrow n + 1 + m \) and a constant scalar \( c \in ZX_\pi^2 \) such that

\[
\left[ \begin{array}{c}
\tilde{X} \\
\ldots
\end{array} \right] = \left[ \begin{array}{c}
\partial \mathbf{M} \left[ X_\beta(1, 0) \right] \otimes X_\beta(n, m) \\
[ X_\beta(1, 0) \otimes \partial \mathbf{M} \left[ X_\beta(n, m) \right] ]
\end{array} \right] \quad (5.43)
\]

Finally, we will show that we can transform the diagram \( \partial ZX X_\beta(n + 1, m) \) to

\[
\left[ \begin{array}{c}
\tilde{X} \\
\ldots
\end{array} \right] \quad \text{with a sequence of graphical rewrites:}
\]

**Claim 5.13.**

\[
\partial ZX X_\beta(n + 1, m) = c \quad (5.44)
\]

As the rewrite rules preserve the semantics, it follows from the claims above that:

\[
\left[ \partial ZX X_\beta(n + 1, m) \right] = 5.13 \left[ \begin{array}{c}
\tilde{X} \\
\ldots
\end{array} \right] = 5.12 \left[ \begin{array}{c}
\partial \mathbf{M} \left[ X_\beta(1, 0) \right] \otimes X_\beta(n, m) \\
[ X_\beta(1, 0) \otimes \partial \mathbf{M} \left[ X_\beta(n, m) \right] ]
\end{array} \right] = 5.42 \partial M \left[ X_\beta(n + 1, m) \right]
\]

(5.45)
Proofs. While proving the claims 5.12 and 5.13 we will repeatedly use the following lemmas:

Lemma 5.14.

![Diagram](image1)

Proof. The right hand side transformation:

![Transformation Diagrams](image2)

(5.46)

Lemma 5.15. Diagrams:

\[ C_{\pm \beta} = \begin{pmatrix} \pm \beta \\ \end{pmatrix}, \quad C_{\beta}^{(n,m)} = \begin{pmatrix} \otimes_{n+m} \\ \end{pmatrix} \]

are controlled states such that

\[ [C_{\pm \beta}] |1\rangle = \begin{pmatrix} \otimes (n+m-1) \\ \end{pmatrix}, \quad [C_{\beta}^{(n,m)}] |1\rangle = \begin{pmatrix} \otimes_{n+m}, & X(n,m) \end{pmatrix} \]

Proof. For the diagram \( C_{\pi \beta} \) the claim follows directly from the definition. For the diagram \( C_{\beta}^{(n,m)} \) we check two cases:
\[
\tilde{X} = C_{\beta}^{(1,0)} C_{+\beta} \cdots C_{-\beta} C_{+\beta} C_{\beta}^{(n,m)}
\]  

(5.49)

where \( C_{\pm \beta} \) and \( C_{\beta}^{(n,m)} \) are defined in the previous lemma (5.15). From these definitions and the lemma (4.3) follows that

\[
\begin{align*}
\left[ \tilde{X} \right]|1\rangle &= \left[ C_{\beta}^{(1,0)} \right]|1\rangle \otimes \left[ C_{+\beta} \right]|1\rangle \otimes \cdots \otimes \left[ C_{-\beta} \right]|1\rangle + \left[ C_{+\beta} \right]|1\rangle \otimes \left[ C_{\beta}^{(n,m)} \right]|1\rangle \\
&= \left[ \otimes (n+m) \right] \left[ [\partial_{ZX} [X_{\beta}(1,0)] \otimes X_{\beta}(n, m)] + [X_{\beta}(1,0) \otimes \partial_{ZX} [X_{\beta}(n, m)]] \right]
\end{align*}
\]

(5.50)
Therefore the claim holds for the controlled state (5.49) and the scalar \( c = \otimes (n + m - 2) \)

**claim 5.13.** Firstly, we simplify the expression \( \tilde{X} \circ \frac{\pi}{2} \):{\small \[
\begin{align*}
\tilde{X} \circ \frac{\pi}{2} &= \otimes (n + m) \\
2.6 &= \otimes (n + m) \\
2.11 \otimes (n + m) &= \otimes (n + m) \\
2.24 \otimes (n + m) &= \otimes (n + m) \quad (5.51)
\end{align*}
\]}

We apply the lemma 5.14 to parts in the red frame and obtain:

\( \frac{\pi}{2} \)

\[
\begin{align*}
\frac{\pi}{2} &= \otimes (n + m) \\
2.11 \otimes (n + m) &= \otimes (n + m) \quad (5.52)
\end{align*}
\]
The claim holds for the scalar $c = \pi (n + m - 2)$.

5.3. **Simplified formula for paired spiders.** Variational quantum algorithms use gradients in the search for optimal parameter values. The objective minimized by these algorithms can be expressed as $\langle \psi(\beta)|H|\psi(\beta)\rangle$ where the diagram for $\langle \psi(\beta)|H|\psi(\beta)\rangle^\dagger$ is obtained out of the diagram for $|\psi(\beta)\rangle$ by flipping upside down followed by the change of signs in spiders. Therefore, parameters in the diagram for $\langle \psi(\beta)|H|\psi(\beta)\rangle$ appear in pairs $-\beta$.

We suggest a more compact formula for diagrams in what we call *pair-factored form*:

$$D_2 \circ (D_1 \otimes Y(n)).$$

In this expression $Y_\beta(n) = \left( \begin{array}{c} \beta \\ \vdots \\ \beta \end{array} \right)$.

**Lemma 5.16.** The diagram:

$$\partial_P (Y_\beta(n)) =$$

satisfies $[\partial_P (Y_\beta(n))] = \partial_M [Y_\beta(n)]$.

The lemma (5.16) can be proven by applying the same approach as in the proof of the formula (5.11) for individual spiders. We leave the exact proof to the reader.

It is possible to extend 5.16 to find the derivative for $X_\beta(n, m)$ when $n \neq m$. Indeed, it is always possible to transform a diagram into an equivalent diagram that has the same number of occurrences of $\beta$ and $-\beta$, using the fact that $\pm \beta = \frac{\pi}{2}$. For instance, if $n > m$:

$$\partial_P (X_\beta(n, m)) = \sigma \circ \left[ \begin{array}{c} \otimes n - m \\ \vdots \\ \otimes 2n - 1 \\ \otimes n - m \\ \vdots \\ \otimes 2m - 1 \end{array} \right]$$

where $\sigma$ is some wire permutation and

$$\left( \begin{array}{c} \partial_P (Y_\beta(n)) \\ \vdots \\ \partial_P (Y_\beta(n)) \end{array} \right) = \left( \begin{array}{c} \otimes n - m \\ \vdots \\ \otimes 2n - 1 \end{array} \right).$$

**Example 5.17.** We apply 5.16 to the same diagram as in 5.7:

$$\partial_P \left( -\beta \right) =$$

We observe that using the formula for the pair-factored form (5.53) we obtain a much more compact result than with the inductive procedure (see the example 5.7). Even if both approaches lead to diagrams with the same semantics, in practice the diagrams obtained with the formulas (5.53) and (5.38) are less verbose. For this reason they are easier to manipulate.
5.4. Discussion. Contrary to derivatives defined in [TYd21] and [ZG21] in our approaches a derivative of a ZX-diagram is another ZX-diagram.

In the inductive approach we proceed by integration of the product rules using controlizers. We also observe that as the derivative of a constant diagram is trivial, it is beneficial to "factor-out" the part that depends on the parameter. We adopted this approach in formulas for diagrams in β-factored form and pair-factored forms. To derive the formulas, we got the intuition in the desired semantics of the diagrams. Thenceforth, our formulas were rigorously proven by induction.

A result similar to our simplified formula for β-factored forms was independently derived in [WY22]. The major difference between our formula and the method shown in [WY22] is the considered language. Indeed, in our work we operate ZX-diagrams while Wang and Yeung consider diagrams from the more expressive algebraic ZX-calculus.

The difficulty to represent derivatives for non-linear diagrams follows from the fact that there is no simple way to represent real numbers in the vanilla ZX-calculus. In the algebraic ZX-calculus this restriction is removed. As a consequence, when an algebraic ZX-diagram is parameterized by an arbitrary derivable function \( f(x) \), the differentiated algebraic ZX-diagram is parametrized by \( f'(x) \).

6. Diagrammatic representation of Ising Hamiltonians

The addition of diagrams naturally appears in the representation of Hamiltonians. Usually, Hamiltonians are given as a weighted sum of Pauli tensors (see [Ves22] for an introduction to Hamiltonians):

\[
H = \sum_k \alpha_k \sigma_k^0 \otimes \cdots \otimes \sigma_k^n, \quad \alpha_k \in \mathbb{R}, \sigma_k^0 = I \text{ or } X \text{ or } Y \text{ or } Z
\] (6.1)

Pauli tensors have a particularly simple decomposition on elementary matrices. Therefore, the procedure described in [YWS] is well-suited for the construction of diagrammatic representation for such Hamiltonians.

In addition, the work [YWS] also suggests a way to exponentiate diagrams representing Hamiltonians. The exponentiate of a Hamiltonian \( H \) is a unitary \( e^{i\theta H} \). The exponentiation is done with the aid of the Cayley-Hamilton theorem. The Cayley-Hamilton theorem allows to decompose the exponent \( e^{i\theta H} \) on a linear combination of \( H^k = H \times \cdots \times H \) for \( k \in [1, \ldots, 2^n] \).

We remark that in a particular case when \( H = \sum_i H_i \) is a sum of commuting terms it is usually straightforward to obtain \( e^{i\theta H} \) as a product of individual \( e^{i\theta H_i} \). However, if some local terms in \( H \) don’t commute the product decomposition doesn’t apply. For instance, this is the case for modified mixer Hamiltonians used in the extension of QAOA to constrained optimization problems [HWO+19].

In this section, we will also consider an inverse problem. Using Stone’s theorem, we demonstrate how to obtain a diagram for \( H \) given the derivative of the diagram for \( e^{i\theta H} \).

The Stone theorem relating the derivative of a unitary group \( e^{i\gamma H} \) to its generator \( H \) was used before in [TYd21]. However, in [TYd21] the derivatives were defined as formal sums of diagrams. As a consequence, the work [TYd21] didn’t suggest a single diagram representation for a Hamiltonian but rather a representation as a sum of diagrams.
We remark that for an Ising Hamiltonian $H$ the diagram $D_U(\beta)$ of the linear map $U(\beta) = e^{i\beta H}$ is easy to find. For Hamiltonians with integer coefficients the matrix $U(\beta) = e^{i\beta H}$ belongs to $\mathcal{M}(\beta)$. It satisfies the definition of a strongly continuous one-parameter unitary group:

**Definition 6.1** (Unitary group [TYd21]). A one-parameter unitary group is a unitary matrix $U : n \to n$ in $\mathcal{M}(\beta)$ with $U(0) = id_n$ and $U(\beta)U(\beta') = U(\beta + \beta')$ for all $\beta, \beta' \in \mathbb{R}$. It is strongly continuous when $\lim_{\beta \to \beta_0} U(\beta) = U(\beta_0)$ for all $\beta_0 \in \mathbb{R}$.

**Theorem 6.2** (Stone ([Sto32])). There is a one-to-one correspondence between strongly continuous one-parameter unitary groups $U : n \to n$ in $\mathcal{M}(\beta)$ and self-adjoint matrices $H : n \to n$ in $\mathcal{M}$. The bijection is given explicitly by $U(\beta) = e^{i\beta H}$ and $H = -i(\partial_M U)(0)$.

We use the bijection from Stone’s theorem to find the diagram $h \in ZX_\mathbb{R}$ such that $[h] = H$. Using the property $U(0) = id_n$ we obtain:

$$H = -i[\partial_M U(\beta)](0) = -i \otimes [[\partial_{ZX} D_U](\beta)](0) = -i \otimes [[\partial_{ZX} D_U](0)]$$

where the third equality is due to the fact that the evaluation commutes with the standard interpretation.

We give an example of a diagram for an Ising Hamiltonian obtained via our approach.

**Example 6.3.** Let $H : 2 \to 2$ be the Hamiltonian given by $H = Z_1 - Z_2 + Z_1Z_2$. The diagram $D_U(\beta)$ for $U(\beta) = e^{i\beta H}$ is:

$$D_U(\beta) = \includegraphics[scale=0.5]{diagram1} \quad \includegraphics[scale=0.5]{diagram2}$$

Using the formula (5.38) we find the derivative of $D_U(\beta)$:

$$\partial_{ZX} D_U(\beta) = \includegraphics[scale=0.5]{diagram3} \quad \includegraphics[scale=0.5]{diagram4}$$

It is of course possible to obtain a diagram for $H$ using the addition procedure we obtained earlier, rather than going through computing its exponential and then differentiating. However, in the case when $H$ is Ising, the exponential of $H$ is typically easier to obtain than $H$ itself.
7. Discussions

In this work, we have introduced an inductive definition for addition of ZX-diagrams, that we have then used to introduce an inductive definition of the differentiation of ZX-diagrams. Addition and differentiation are essential tools for the development and the study of quantum algorithms, but, as a matter of fact, both of them are leading to large diagrams, even when the initial diagrams are fairly simple.

In sections 5.2 and 5.3, we have shown that instead of simplifying the resulting diagrams \textit{a posteriori}, one can \textit{a priori} put the initial diagrams in an appropriate form. While this approach is not inductive anymore, it seems to ease the differentiation of diagrams in practice. As an application we have shown that our result allows the construction of diagrams for Ising Hamiltonians and for derivatives of parametrized circuits. Therefore, it becomes possible to study variational algorithms entirely within the ZX-calculus. In particular, we can use rewrite rules to simplify such expressions as $\langle \psi(\beta) | H_f | \psi(\beta) \rangle$ and $\frac{\partial}{\partial \beta} \langle \psi(\beta) | H_f | \psi(\beta) \rangle$. We believe that it will lead to a better understanding of the potential of variational algorithms and of their applications to real-world problems.

Acknowledgements

This work was supported in part by the CIFRE EDF/Loria Quantum Computing for Combinatorial Optimisation, the French National Research Agency (ANR) under the research projects SoftQPro ANR-17-CE25-0009-02 and VanQuTe ANR-17-CE24-0035, the PEPR integrated project EPIQ ANR-22-PETQ-0007 part of Plan France 2030, by the STIC-AmSud project Qapla’ 21-STIC-10, and by the European projects NEASQC (funded from the European Union’s Horizon 2020 research and innovation programme grant agreement No 951821) and HPCQS (European High-Performance Computing Joint Undertaking under grant agreement No 101018180).

References

[Bac14] Miriam Backens. The ZX-calculus is complete for stabilizer quantum mechanics. \textit{New Journal of Physics}, 16(9):093021, sep 2014. doi:10.1088/1367-2630/16/9/093021.

[BK21] Lennart Bittel and Martin Kliesch. Training variational quantum algorithms is NP-hard. \textit{Physical Review Letters}, 127(12), sep 2021. doi:10.1103/physrevlett.127.120502.

[CAB+21] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. Variational quantum algorithms. \textit{Nature Reviews Physics}, 3(9):625–644, Aug 2021. doi:10.1038/s42254-021-00348-9.

[CD11] Bob Coecke and Ross Duncan. Interacting quantum observables: categorical algebra and diagrammatics. \textit{New Journal of Physics}, 13(4):043016, apr 2011. doi:10.1088/1367-2630/13/4/043016.

[CDD+20] Alexander Cowtan, Silas Dilkes, Ross Duncan, Will Simmons, and Seyon Sivarajah. Phase gadget synthesis for shallow circuits. \textit{Electronic Proceedings in Theoretical Computer Science}, 318:213–228, may 2020. doi:10.4204/eptcs.318.13.

[CK17] Bob Coecke and Aleks Kissinger. \textit{Picturing Phases and Complementarity}, pages 510–623. Cambridge University Press, 2017. doi:10.1017/9781316219317.010.

[CSV+21] M. Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J. Coles. Cost function dependent barren plateaus in shallow parametrized quantum circuits. \textit{Nature Communications}, 12(1), mar 2021. doi:10.1038/s41467-021-21728-x.
[CW12] Andrew M. Childs and Nathan Wiebe. Hamiltonian Simulation Using Linear Combinations of Unitary Operations. \textit{arXiv e-prints}, page arXiv:1202.5822, February 2012. arXiv:1202.5822.

[dBBW20] Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. Techniques to reduce $-4$-parity-phase circuits, motivated by the ZX calculus. \textit{Electronic Proceedings in Theoretical Computer Science}, 318:131–149, may 2020. doi:10.4204/eptcs.318.9.

[dBH20] Niel de Beaudrap and Dominic Horsman. The ZX calculus is a language for surface code lattice surgery. \textit{Quantum}, 4:218, January 2020. doi:10.22331/q-2020-01-09-218.

[DKPvdW20] Ross Duncan, Aleks Kissinger, Simon Perdrix, and John van de Wetering. Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus. \textit{Quantum}, 4:279, jun 2020. doi:10.22331/q-2020-06-04-279.

[DP10] Ross Duncan and Simon Perdrix. Rewriting measurement-based quantum computations with generalised flow. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, \textit{Automata, Languages and Programming}, pages 285–296, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[DP14] Ross Duncan and Simon Perdrix. Pivoting makes the ZX-calculus complete for real stabilizers. \textit{Electronic Proceedings in Theoretical Computer Science}, 171:50–62, Dec 2014. doi:10.4204/eptcs.171.5.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014. doi:10.48550/ARXIV.1411.4028.

[GD18] Liam Garvie and Ross Duncan. Verifying the smallest interesting colour code with quantomatic. \textit{Electronic Proceedings in Theoretical Computer Science}, 266:147–163, feb 2018. doi:10.4204/eptcs.266.10.

[GS17] Gian Giacomo Guerreschi and Mikhail Smelyanskiy. Practical optimization for hybrid quantum-classical algorithms, 2017. arXiv:1701.01450.

[Had15] Amar Hadzihasanovic. A diagrammatic axiomatisation for qubit entanglement. In \textit{2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science}, pages 573–584, 2015. doi:10.1109/LICS.2015.59.

[HEDN20] Michael Hanks, Marta P. Estarellas, William J. Munro, and Kae Nemoto. Effective compression of quantum braided circuits aided by ZX-calculus. \textit{Physical Review X}, 10(4), nov 2020. doi:10.1103/physrevx.10.041030.

[HNW18] Amar Hadzihasanovic, Kang Feng Ng, and Quanlong Wang. Two complete axiomatisations of pure-state qubit quantum computing. In \textit{Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science}, LICS ‘18, pages 502–511, New York, NY, USA, 2018. ACM. doi:10.1145/3209108.3209128.

[Hor11] Clare Horsman. Quantum pictorialism for topological cluster-state computing. \textit{New Journal of Physics}, 13(9):095011, sep 2011. doi:10.1088/1367-2630/13/9/095011.

[HWO+19] Stuart Hadfield, Zhihui Wang, Bryan O’Gorman, Eleanor Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. \textit{Algorithms}, 12(2):34, feb 2019. doi:10.3390/a12020034.

[ILY21] Artur F. Izmaylov, Robert A. Lang, and Tzu-Ching Yen. Analytic gradients in variational quantum algorithms: Algebraic extensions of the parameter-shift rule to general unitary transformations. \textit{Physical Review A}, 104(6), dec 2021. doi:10.1103/physreva.104.062443.

[JPV18a] Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. A Complete Axiomatization of the ZX-Calculus for Clifford+T Quantum Mechanics. In \textit{The 33rd Annual {ACM/IEEE} Symposium on Logic in Computer Science, {LICS} 2018}, Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, pages 559–568, Oxford, United Kingdom, July 2018. doi:10.1145/3209108.3209131.

[JPV18b] Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. Diagrammatic Reasoning beyond Clifford+T Quantum Mechanics. In \textit{The 33rd Annual Symposium on Logic in Computer Science}, Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, pages 569–578, Oxford, United Kingdom, July 2018. doi:10.1145/3209108.3209139.

[JPV19] Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. A generic normal form for ZX-diagrams and application to the rational angle completeness. In \textit{Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science}, LICS ’19. IEEE Press, 2019.
[JPV22] Emmanuel Jeandel, Simon Perdrix, and Margarita Veshchezerova. Addition and Differentiation of ZX-diagrams. arXiv e-prints, page arXiv:2202.11386, February 2022. arXiv:2202.11386.

[JPVV17] Emmanuel Jeandel, Simon Perdrix, Renaud Vilmart, and Quanlong Wang. ZX-Calculus: Cyclotomic Suplementarity and Incompleteness for Clifford+T Quantum Mechanics. In Kim G. Larsen, Hans L. Bodlaender, and Jean-Francois Raskin, editors, 42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017), volume 83 of Leibniz International Proceedings in Informatics (LIPIcs), pages 11:1–11:13, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.MFCS.2017.11.

[BK22] Bálint Koczor and Simon C. Benjamin. Quantum analytic descent. Physical Review Research, 4(2), apr 2022. doi:10.1103/physrevresearch.4.023017.

[KBE21] Oleksandr Kyriienko and Vincent E. Elfving. Generalized quantum circuit differentiation rules. Phys. Rev. A, 104:052417, Nov 2021. doi:10.1103/PhysRevA.104.052417.

[MBK21] Andrea Mari, Thomas R. Bromley, and Nathan Killoran. Estimating the gradient and higher-order derivatives on quantum hardware. Physical Review A, 103(1), jan 2021. doi:10.1103/physreva.103.012405.

[MBS +18] Jarrod R. McClean, Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. Nature Communications, 9(1), nov 2018. doi:10.1038/s41467-018-07090-4.

[Pre18] John Preskill. Quantum computing in the NISQ era and beyond. Quantum, 2:79, aug 2018. doi:10.22331/q-2018-08-06-79.

[SBG +19] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. Evaluating analytic gradients on quantum hardware. Physical Review A, 99(3), mar 2019. doi:10.1103/physreva.99.032331.

[SH22] Tobias Stollenwerk and Stuart Hadfield. Diagrammatic Analysis for Parameterized Quantum Circuits. arXiv e-prints, page arXiv:2204.01307, April 2022. arXiv:2204.01307.

[Sto32] M. H. Stone. On one-parameter unitary groups in hilbert space. Annals of Mathematics, 33(3):643–648, 1932. URL: http://www.jstor.org/stable/1968538.

[TYd21] Alexis Toumi, Richie Yeung, and Giovanni de Felice. Diagrammatic differentiation for quantum machine learning. arXiv e-prints, page arXiv:2103.07960, March 2021. arXiv:2103.07960.

[vvdW20] John van de Wetering. ZX-calculus for the working quantum computer scientist, 2020. arXiv:2012.13966.

[Ves22] Margarita Veshchezerova. Quantum algorithms for energy management optimization problems. Theses, Université de Lorraine, December 2022. URL: https://hal.univ-lorraine.fr/tel-04105922.

[Pre19] John Preskill. Quantum Computing in the NISQ Era and Beyond. Quantum, 2:79, August 2018. doi:10.22331/q-2018-08-06-79.
[WSW+21] Max Wilson, Rachel Stromswold, Filip Wudarski, Stuart Hadfield, Norm Tubman, and Eleanor Rieffel. Optimizing quantum heuristics with meta-learning. Quantum Machine Intelligence, 3, 06 2021. doi:10.1007/s42484-020-00022-w.

[WY21] Quanlong Wang and Richie Yeung. Representing and implementing matrices using algebraic ZX-calculus, 2021. doi:10.48550/ARXIV.2110.06898.

[WY22] Quanlong Wang and Richie Yeung. Differentiating and integrating ZX diagrams, 2022. arXiv:2201.13250.

[YWS] Richie Yeung, Quanlong Wang, and Razin A. Shaikh. How to sum and exponentiate Hamiltonians in ZXW calculus. URL: https://www.qplconference.org/proceedings2022/QPL_2022_paper_85.pdf.

[ZG21] Chen Zhao and Xiao-Shan Gao. Analyzing the barren plateau phenomenon in training quantum neural networks with the ZX-calculus. Quantum, 5:466, June 2021. doi:10.22331/q-2021-06-04-466.