Diagrammatic Differentiation for Quantum Machine Learning

Alexis Toumi[⋆†], Richie Yeung[†], Giovanni de Felice[⋆†]

⋆ Department of Computer Science, University of Oxford † Cambridge Quantum Computing Ltd.

We introduce diagrammatic differentiation for tensor calculus by generalising the dual number construction from rigs to monoidal categories. Applying this to ZX diagrams, we show how to calculate diagrammatically the gradient of a linear map with respect to a phase parameter. For diagrams of parametrised quantum circuits, we get the well-known parameter-shift rule at the basis of many variational quantum algorithms. We then extend our method to the automatic differentation of hybrid classical-quantum circuits, using diagrams with bubbles to encode arbitrary non-linear operators. Moreover, diagrammatic differentiation comes with an open-source implementation in DisCoPy, the Python library for monoidal categories. Diagrammatic gradients of classical-quantum circuits can then be simplified using the PyZX library and executed on quantum hardware via the tket compiler. This opens the door to many practical applications harnessing both the structure of string diagrams and the computational power of quantum machine learning.

Introduction

String diagrams are a graphical language introduced by Penrose [1] to manipulate tensor expressions: wires represent vector spaces, nodes represent multi-linear maps between them. In [2], these diagrams are used to describe the geometry of space-time and an extra piece of notation is introduced: the covariant derivative is represented as a bubble around the tensor to be differentiated. Joyal and Street [3, 4] characterised string diagrams as the arrows of free monoidal categories, however their geometry of tensor calculus makes no mention of differential calculus, it only deals with composition and tensor.

In categorical quantum mechanics [5] string diagrams are used to axiomatise quantum theory in terms of dagger compact-closed categories. This culminated in the ZX-calculus [6], a graphical language that provides a complete set of rules for qubit quantum computing [7, 8]. ZX diagrams have recently been used for state-of-the-art quantum circuit optimisation [9, 10, 11], compilation [12, 13], extraction [14] and error correction [15, 16]. In recent work, ZX diagrams have been used to study quantum machine learning [17, 18] and its application to quantum natural language processing [19, 20].

In this work, we introduce diagrammatic differentiation: a graphical notation for manipulating tensor derivatives. On the theoretical side, we generalise the dual number construction (discussed in section 1) from rigs to monoidal categories (section 2). We then apply this construction to the category of ZX diagrams (section 3) and of quantum circuits (section 4). In section 5 we give a formal definition of diagrams with bubbles and their gradient with the chain rule. We use this to differentiate quantum circuits with neural networks as classical post-processing. The theory comes with an implementation in DisCoPy [21], the Python library for monoidal categories. The gradients of classical-quantum circuits can then be simplified using the PyZX library [22] and compiled on quantum hardware via the tket compiler [23].

Submitted to:
QPL 2021

Related work

The same bubble notation for vector calculus is proposed in [24], but they have mainly pedagogical motivations and restrict themselves to the case of three-dimensional Euclidean space. To the best of our knowledge, our definition is the first formal account of string diagrams with bubbles for tensor derivatives.

Differential categories [25] have been introduced to axiomatise the notion of derivative. More recently reverse derivative categories [26] generalised the notion of back-propagation, they have been proposed as a categorical foundation for gradient-based learning [27]. These frameworks all define the derivative of a morphism with respect to its domain. In our setup however, we define the derivative of parametrised morphism with respect to parameters that are in some sense external to the category. Investigating the relationship between these two definitions is left to future work.

1 Dual numbers

Dual numbers were first introduced by Clifford in 1873 [28]. Given a commutative rig (i.e. a riNg without Negatives) $\mathbb{S}$, the rig of dual numbers $\mathbb{D}[\mathbb{S}]$ extends $\mathbb{S}$ by adjoining a new element ϵ such that $\epsilon^2 = 0$. Concretely, elements of $\mathbb{D}[\mathbb{S}]$ are formal sums $s + s'\epsilon$ where s and s' are scalars in $\mathbb{S}$. We write $\pi_0, \pi_1 : \mathbb{D}[\mathbb{S}] \to \mathbb{S}$ for the projection on the real and epsilon component respectively. Addition and multiplication of dual numbers are given by:

$$(a + a'\,\epsilon) + (b + b'\,\epsilon) \quad = \quad (a + b) \ + \ (a + b')\,\epsilon \tag{1}$$

$$(a + a'\,\epsilon) \times (b + b'\,\epsilon) \quad = \quad (a \times b) \ + \ (a \times b' \ + \ a' \times b)\,\epsilon \tag{2}$$

A related notion is that of differential rig: a rig $\mathbb{S}$ equipped with a derivation, i.e. a map $\partial : \mathbb{S} \to \mathbb{S}$ which preserves sums and satisfies the Leibniz product rule $\partial(f \times g) = f \times \partial(g) + \partial(f) \times g$ for all $f, g \in \mathbb{S}$. An equivalent condition is that the map $f \mapsto f + (\partial f)\epsilon$ is a homomorphism of rigs $\mathbb{S} \to \mathbb{D}[\mathbb{S}]$. The correspondance also works the other way around: given a homorphism $\partial : \mathbb{S} \to \mathbb{D}[\mathbb{S}]$ such that $\pi_0 \circ \partial = \mathtt{id}_{\mathbb{S}}$, projecting on the epsilon component is a derivation $\pi_1 \circ \partial : \mathbb{S} \to \mathbb{S}$. The motivating example is the rig of smooth functions $\mathbb{S} = \mathbb{R} \to \mathbb{R}$, where differentiation is a derivation. Concretely, we can extend any smooth function $f : \mathbb{R} \to \mathbb{R}$ to a function $f : \mathbb{D}[\mathbb{R}] \to \mathbb{D}[\mathbb{R}]$ over the dual numbers defined by:

$$f(a + a'\epsilon) \quad = \quad f(a) \ + \ a' \times (\partial f)(a)\epsilon \tag{3}$$

We can use equations 1, 2 and 3 to derive the usual rules for gradients in terms of dual numbers. For the identity function we have $\mathtt{id}(a + a'\epsilon) = \mathtt{id}(a) + a'\epsilon$, i.e. $\partial \mathtt{id} = 1$. For the constant functions we have $c(a + a'\epsilon) = c(a) + 0\epsilon$, i.e. $\partial c = 0$. For addition, multiplication and composition of functions, we can derive the following *linearity*, *product* and *chain* rules:

$$(f + g)(a + a'\epsilon) \ = \ (f + g)(a) \ + \ a' \times (\partial f + \partial g)(a)\epsilon \tag{4}$$

$$(f \times g)(a + a'\epsilon) \ = \ (f \times g)(a) \ + \ a' \times (f \times \partial g \ + \ \partial f \times g)(a)\epsilon \tag{5}$$

$$(f \circ g)(a + a'\epsilon) \ = \ (f \circ g)(a) \ + \ a' \times (\partial g \ \times \ \partial f \circ g)(a)\epsilon \tag{6}$$

This generalises to smooth functions $\mathbb{R}^n \to \mathbb{R}^m$, where the partial derivative ∂_i is a derivation for each $i < n$. The functions $\mathbb{F}_2^n \to \mathbb{F}_2^m$ on the two-element field $\mathbb{F}_2$ with elementwise XOR as sum and conjunction as product also forms a differential rig. The partial derivative is given by $(\partial_i f)(\vec{x}) = f(\vec{x}_{[x_i \mapsto 0]}) \oplus f(\vec{x}_{[x_i \mapsto 1]})$. Intuitively, the $\mathbb{F}_2$ gradient $\partial_i f(\vec{x}) \in \mathbb{F}_2^m$ encodes which coordinates of $f(\vec{x})$

actually depend on the input x_i. An example of differential rig that isn't also a ring is given by the set $\mathbb{N}[X]$ of polynomials with natural number coefficients, again each partial derivative is a derivation.

A more exotic example is the rig of Boolean functions with elementwise disjunction as sum and conjunction as product. Boolean functions $\mathbb{B}^n \to \mathbb{B}^m$ can be represented as tuples of m propositional formulae over n variables. The partial derivative ∂_i for $i < n$ is defined by induction over the formulae: for variables we have $\partial_i x_j = \delta_{ij}$, for constants $\partial_i 0 = \partial_i 1 = 0$ and for negation $\partial_i \neg\varphi = \neg\partial_i\varphi$. The derivative of disjunctions and conjunctions are given by the linerarity and product rules. Equivalently, the gradient of a propositional formula can be given by $\partial_i\varphi = \neg\varphi_{[x_i\mapsto 0]} \wedge \varphi_{[x_i\mapsto 1]}$. Concretely, a model satisfies $\partial_i\varphi$ if and only if it satisfies $\varphi \leftrightarrow x_i$: the derivative is true when the variable and the formula are positively correlated. Substituting x_i with its negation, we get that a model satisfies $\partial_i\varphi_{[x_i\mapsto\neg x_i]}$ if and only if it satisfies $\varphi \leftrightarrow \neg x_i$, i.e. iff variable and formula are anti-correlated. Note that although $\mathbb{B}$ and $\mathbb{F}_2$ are isomorphic as sets, they are distinct rigs. Their derivations are related however by $\partial_i^{\mathbb{F}_2} f \mapsto \partial_i^{\mathbb{B}}\varphi \vee \partial_i^{\mathbb{B}}\varphi_{[x_i\mapsto\neg x_i]}$ for $\varphi : \mathbb{B}^n \to \mathbb{B}$ the formula corresponding to the function $f : \mathbb{F}_2^n \to \mathbb{F}_2$. That is, a Boolean function depends on an input variable precisely when either the corresponding formula is positively correlated or anti-correlated.

Dual numbers are a fundamental tool for *automatic differentiation* [29], i.e. they allow to compute the derivative of a function automatically from its definition. The key idea is that given a definition of $f : \mathbb{S}^n \to \mathbb{S}^m$ as a composition of elementary functions, we can compute $(\partial_i f)(a)$ by evaluating $f(a+\epsilon)$ and projecting on the epsilon component.

2 Dual diagrams

Our main technical contribution is to generalise derivations from rigs to monoidal categories with sums. Applying this to free monoidal categories, where the arrows are string diagrams, we say a derivation is diagrammatic when it commutes with the interpretation of the diagrams. We take two different flavours of the ZX-calculus as our main examples.

Let $(\mathbf{C}, \otimes, 1)$ be a monoidal category with sums, i.e. it has commutative monoids on each homset $(+) : \coprod_{x,y} \mathbf{C}(x,y) \times \mathbf{C}(x,y) \to \mathbf{C}(x,y)$ with unit $0 \in \coprod_{x,y} \mathbf{C}(x,y)$ such that composition and tensor distribute over the sum. Note that a one-object monoidal category with sums is simply a rig. Our motivating example is the category $\mathbf{Mat}_{\mathbb{S}}$ with natural numbers as objects and matrices valued in a commutative rig $\mathbb{S}$ as arrows, with matrix multiplication as composition, Kronecker product as tensor and entrywise sum. We define the category $\mathbb{D}[\mathbf{C}]$ by adjoining a scalar (i.e. an endomorphism of the monoidal unit) ϵ such that $\epsilon \otimes \epsilon = 0$ for all arrows $f \in \mathbf{C}$[1]. Concretely, the objects of $\mathbb{D}[\mathbf{C}]$ are the same as those of $\mathbf{C}$, the arrows are given by formal sums $f + f'\epsilon$ of parallel arrows $f, f' \in \mathbf{C}$. Composition and tensor are both given by the product rule:

$$(f + f'\epsilon) \fatsemi (g + g'\epsilon) \;=\; f \fatsemi g \;+\; (f' \fatsemi g \;+\; f \fatsemi g')\,\epsilon \tag{7}$$

$$(f + f'\epsilon) \otimes (g + g'\epsilon) \;=\; f \otimes g \;+\; (f' \otimes g \;+\; f \otimes g')\,\epsilon \tag{8}$$

We say that a unary operator on homsets $\partial : \coprod_{x,y} \mathbf{C}(x,y) \to \mathbf{C}(x,y)$ is a derivation whenever it satisfies the product rules for both composition $\partial(f \fatsemi g) = (\partial f) \fatsemi g + f \fatsemi (\partial g)$ and tensor $\partial(f \otimes g) = (\partial f) \otimes g + f \otimes (\partial g)$. An equivalent condition is that the map $f \mapsto f + (\partial f)\epsilon$ is a sum-preserving monoidal functor $\mathbf{C} \to \mathbb{D}[\mathbf{C}]$. Again, the correspondance between dual numbers and derivations works the other way around: given a sum-preserving monoidal functor $\partial : \mathbf{C} \to \mathbb{D}[\mathbf{C}]$ such that $\pi_0 \circ \partial = \mathtt{id}_{\mathbf{C}}$, projecting on the epsilon component gives a derivation $\pi_1 \circ \partial : \coprod_{x,y} \mathbf{C}(x,y) \to \mathbf{C}(x,y)$. The following propositions characterise the derivations on the category of matrices valued in a commutative rig $\mathbb{S}$.

[1] Note that in the case when $\mathbf{C}$ is not symmetric monoidal (or at least braided) the axiom $\epsilon \otimes f = f \otimes \epsilon$ is also needed.

Proposition 2.1. *Dual matrices are matrices of dual numbers, i.e.* $\mathbb{D}[\mathbf{Mat}_{\mathbb{S}}] \simeq \mathbf{Mat}_{\mathbb{D}[\mathbb{S}]}$.

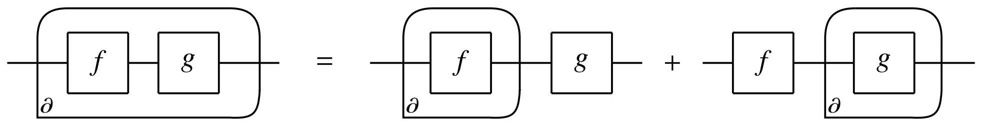
Proof. The isomorphism is given by $\left(\sum_{ij} f_{ij} |j\rangle\langle i|\right) + \left(f'_{ij} \sum_{ij} |j\rangle\langle i|\right)\epsilon \;\longleftrightarrow\; \sum_{ij}(f_{ij} + f'_{ij}\epsilon)|j\rangle\langle i|$. □

Proposition 2.2. *Derivations on* $\mathbf{Mat}_{\mathbb{S}}$ *are in one-to-one correspondance with derivations on* $\mathbb{S}$.

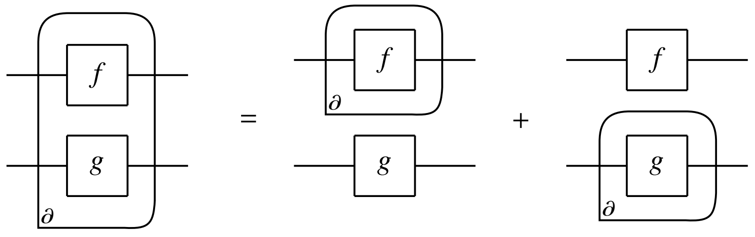
Proof. A derivation on $\mathbf{Mat}_{\mathbb{S}}$ is uniquely determined by its action on scalars in $\mathbb{S}$. Conversely, applying a derivation $\partial : \mathbb{S} \to \mathbb{S}$ entrywise on matrices yields a derivation on $\mathbf{Mat}_{\mathbb{S}}$. □

Fix a monoidal signature Σ with objects Σ_0 and boxes Σ_1. Let $\mathbf{C}_\Sigma$ be the free monoidal category it generates: the objects are types, i.e. lists of generating objects $t = t_1, \dots, t_n \in \Sigma_0^\star$, the arrows are string diagrams with boxes in Σ_1. Let $\mathbf{C}_\Sigma^+$ be the free monoidal category with sums: the objects are also given by types, the arrows are formal sums, i.e. bags[2], of string diagrams. We assume our diagrams are interpreted as matrices, i.e. we fix a sum-preserving monoidal functor $[\![-]\!] : \mathbf{C}_\Sigma^+ \to \mathbf{Mat}_{\mathbb{S}}$ for $\mathbb{S}$ a commutative rig with a derivation $\partial : \mathbb{S} \to \mathbb{S}$. Our main two examples are the standard ZX-calculus with smooth functions $\mathbb{R}^n \to \mathbb{R}$ as phases and the algebraic ZX-calculus over $\mathbb{S}$, introduced in [30].

Applying the dual number construction to $\mathbf{C}_\Sigma^+$, we get the category of dual diagrams $\mathbb{D}[\mathbf{C}_\Sigma^+]$ which is where diagrammatic differentiation happens. By the universal property of $\mathbf{C}_\Sigma^+$, every derivation $\partial : \mathbf{C}_\Sigma^+ \to \mathbb{D}[\mathbf{C}_\Sigma^+]$ is uniquely determined by its image on the generating boxes in Σ_1. Intuitively, if we're given the derivative for each box, we can compute the derivative for every sum of diagram using the product rule. We say that the interpretation $[\![-]\!] : \mathbf{C}_\Sigma^+ \to \mathbf{Mat}_{\mathbb{S}}$ admits diagrammatic differentiation if there is a derivation ∂ on $\mathbf{C}_\Sigma^+$ such that $[\![-]\!] \circ \partial = \partial \circ [\![-]\!]$, i.e. the interpretation of the gradient $[\![\partial d]\!]$ coincides with the gradient of the interpretation $\partial[\![d]\!]$ for all sums of diagrams $d \in \mathbf{C}_\Sigma^+$. We depict the gradient ∂d as a bubble surrounding the diagram d, we introduce bubbles formally in section 5. Once translated to string diagrams, the axioms for derivations on monoidal categories with sums become:



and



3 Differentiating ZX

This section applies the dual number construction to the diagrams of the ZX-calculus.

Definition 3.1. *The diagrams of the ZX-calculus with smooth maps* $\mathbb{R}^n \to \mathbb{R}$ *as phases form a category* $\mathbf{ZX}_n = \mathbf{C}_\Sigma$ *where* $\Sigma = \{H : x \to x,\ \sigma : x^{\otimes 2} \to x^{\otimes 2}\} + \{Z^{m,n}(\alpha) : x^{\otimes m} \to x^{\otimes n} \mid m, n \in \mathbb{N}, \alpha : \mathbb{R}^n \to \mathbb{R}\}$. *H is depicted as a yellow square,* σ *as a swap and* $Z^{m,n}(\alpha)$ *as a green spider. The interpretation* $[\![-]\!] : \mathbf{ZX}_n \to \mathbf{Mat}_{\mathbb{S}}$ *in matrices over* $\mathbb{S} = \mathbb{R}^n \to \mathbb{C}$ *is given by on objects by* $[\![x]\!] = 2$ *and on arrows by* $[\![H]\!] =$

[2]A bag of X, also called a multiset, is a function $X \to \mathbb{N}$. Addition of bags is done pointwise with unit the constant zero.

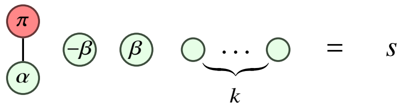
$\frac{1}{\sqrt{2}}\big(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|\big)$, $[\![\sigma]\!] = \sum_{i,j\in\{0,1\}} |j,i\rangle\langle i,j|$ and $[\![Z^{m,n}(\alpha)]\!] = e^{-i\alpha/2}|0\rangle^{\otimes n}\langle 0|^{\otimes m} + e^{i\alpha/2}|1\rangle^{\otimes n}\langle 1|^{\otimes m}$. We write $\mathbf{ZX}_n^+$ for the category of formal sums of parametrised ZX diagrams.

Remark 3.2. *Note that we've scaled the standard interpretation of the green spider by a global phase to match the usual definition of rotation gates in quantum circuits.*

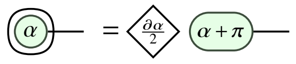
Remark 3.3. *For $n = 0$ we get $\mathbf{ZX}_0 = \mathbf{ZX}$ the ZX-calculus with no parameters. By currying, any ZX diagram $d \in \mathbf{ZX}_n$ can be seen as a function $d : \mathbb{R}^n \to Ar(\mathbf{ZX})$ such that $[\![-]\!] \circ d : \mathbb{R}^n \to \mathbf{Mat}_\mathbb{C}$ is smooth.*

Lemma 3.4. *A function $s : \mathbb{R}^n \to \mathbb{C}$ can be drawn as a scalar diagram in $\mathbf{ZX}_n$ if and only if it is bounded.*

Proof. Generalising [31, P. 8.101] to parametrised scalars, if there is a $k \in \mathbb{N}$ with $|s(\theta)| \le 2^k$ for all $\theta \in \mathbb{R}^n$ then there are parametrised phases $\alpha, \beta : \mathbb{R}^n \to \mathbb{R}$ such that



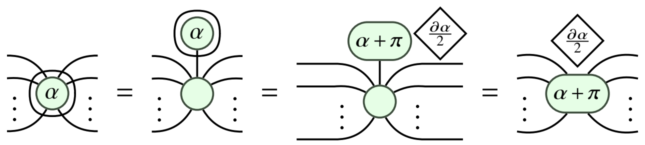
In the other direction, take any scalar diagram d in $\mathbf{ZX}_n$. Let k be the number of spider in the diagram and l the maximum number of legs. By decomposing each spider as a sum of two disconnected diagrams, we can write d as a sum of 2^k diagrams. Each term of the sum is a product of at most $\frac{1}{2} \times k \times l$ bone-shaped scalars. Each bone is bounded by 2, thus $[\![d]\!] : \mathbb{R}^n \to \mathbb{C}$ is bounded by $2^{k\times l}$. □

Lemma 3.5. *In $\mathbf{ZX}_n$, we have*  *for all affine phases $\alpha : \mathbb{R}^n \to \mathbb{R}$.*

Proof. $\partial[\![Z(\alpha)]\!] = \partial\big(e^{-i\alpha/2}|0\rangle + e^{i\alpha/2}|1\rangle\big) = \frac{i\partial\alpha}{2}\big(-e^{-i\alpha/2}|0\rangle + e^{i\alpha/2}|1\rangle\big) = \frac{\partial\alpha}{2}\big(e^{-i\frac{\alpha+\pi}{2}}|0\rangle + e^{i\frac{\alpha+\pi}{2}}|1\rangle\big)$. α is affine so $\partial\alpha$ is constant, hence bounded and from lemma 3.4 we know it can be drawn in $\mathbf{ZX}_n$. □

Theorem 3.6. *The ZX-calculus with affine maps $\mathbb{R}^n \to \mathbb{R}$ as phases admits diagrammatic differentiation.*

Proof. The Hadamard H and swap σ have derivative zero. For the green spiders, we can extend lemma 3.5 from single qubit rotations to arbitrary many legs using spider fusion:



□

Note that there is no diagrammatic differentiation for the ZX-calculus with smooth maps as phases, even when restricted to bounded functions. Take for example $\alpha : \mathbb{R} \to \mathbb{R}$ with $\alpha(\theta) = \sin\theta^2$, it is smooth and bounded by 1 but its derivative $\partial\alpha$ is unbounded. Thus, from lemma 3.4 we know it cannot be represented as a scalar diagram in $\mathbf{ZX}_1$: there can be no diagrammatic differentiation $\partial : \mathbf{ZX}_1 \to \mathbb{D}[\mathbf{ZX}_1]$. In such cases, we can always extend the signature by adjoining a new box for each derivative.

Proposition 3.7. *For every interpretation $[\![-]\!] : \mathbf{C}_\Sigma^+ \to \mathbf{Mat}_\mathbb{S}$, there is an extended signature $\Sigma' \supset \Sigma$ and interpretation $[\![-]\!] : \mathbf{C}_{\Sigma'}^+ \to \mathbf{Mat}_\mathbb{S}$ such that $\mathbf{C}_{\Sigma'}^+$ admits digrammatic differentiation.*

Proof. Let $\Sigma' = \cup_{n\in\mathbb{N}}\Sigma^n$ where $\Sigma^0 = \Sigma$ and $\Sigma^{n+1} = \Sigma^n \cup \{\partial f \mid f \in \Sigma^n\}$ with $[\![\partial f]\!] = \partial[\![f]\!]$. □

The issue of being able to represent arbitrary scalars disappears if we work with the algebraic ZX-calculus instead. Furthermore, we can generalise from $\mathbb{S} = \mathbb{R}^n \to \mathbb{C}$ to any commutative rig.

Definition 3.8. *The diagrams of the algebraic ZX-calculus over a commutative rig $\mathbb{S}$ form a category $\mathbf{ZX}_\mathbb{S} = \mathbf{C}_\Sigma$ where the signature Σ is given in [?, Table 2] and the interpretation is given in [?, §6]. In particular, there is a green square $R_Z^{m,n}(a) \in \Sigma_1$ for each $a \in S$ and $m, n \in \mathbb{N}$ with $[\![R_Z^{m,n}(a)]\!] = |0\rangle^{\otimes n}\langle 0|^{\otimes m} + a|1\rangle^{\otimes n}\langle 1|^{\otimes m}$. Let $\mathbf{ZX}_\mathbb{S}^+$ be the category of formal sums of algebraic ZX diagrams over $\mathbb{S}$.*

Theorem 3.9. *Diagrammatic derivations on $[\![-]\!] : \mathbf{ZX}_\mathbb{S}^+ \to \mathbf{Mat}_\mathbb{S}$ are in one-to-one correspondance with rig derivations $\partial : \mathbb{S} \to \mathbb{S}$.*

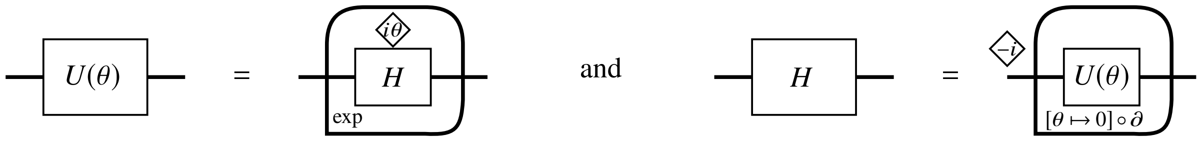
Proof. Given a derivation ∂ on $\mathbb{S}$, we have $\partial[\![R_Z^{m,n}(a)]\!] = (\partial a)|1\rangle^{\otimes n}\langle 1|^{\otimes m}$ and ∂a can be represented by the scalar diagram $R_Z^{1,0}(\partial a)|1\rangle$. In the other direction, a diagrammatic derivation ∂ on $\mathbf{ZX}_\mathbb{S}^+$ is uniquely determined by its action on scalars $R_Z^{1,0}(a)|1\rangle$ for $a \in \mathbb{S}$. □

One application of diagrammatic differentiation is to solve differential equations between diagrams. As a first step, we apply Stone's theorem [32] on one-parameter unitary groups to the ZX-calculus.

Definition 3.10. *A one-parameter unitary group is a unitary matrix $U : n \to n$ in $\mathbf{Mat}_{\mathbb{R}\to\mathbb{C}}$ with $U(0) = \mathtt{id}_n$ and $U(\theta)U(\theta') = U(\theta + \theta')$ for all $\theta, \theta' \in \mathbb{R}$. It is strongly continuous when $\lim_{\theta\to\theta_0} U(\theta) = U(\theta_0)$ for all $\theta_0 \in \mathbb{R}$. We say a one-parameter diagram $d : x^{\otimes n} \to x^{\otimes n}$ is a unitary group if its interpretation $[\![d]\!]$ is.*

Remark 3.11. *The interpretation of diagrams with smooth maps as phases must be strongly continuous.*

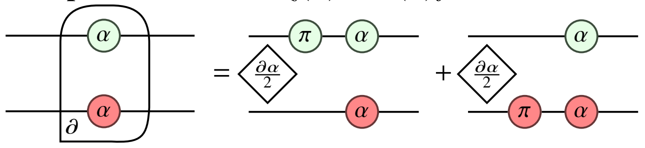
Theorem 3.12 (Stone)**.** *There is a one-to-one correspondance between strongly continuous one-parameter unitary groups $U : n \to n$ in $\mathbf{Mat}_{\mathbb{R}\to\mathbb{C}}$ and self-adjoint matrices $H : n \to n$ in $\mathbf{Mat}_\mathbb{C}$. The bijection is given explicitly by $U(\theta) = \exp(i\theta H)$ and $H = -i(\partial U)(0)$, translated in terms of diagrams with bubbles we get:*



Corollary 3.13. *A one-parameter diagram $d : x^{\otimes n} \to x^{\otimes n}$ in $\mathbf{ZX}_1$ is a unitary group if and only if there is a constant self-adjoint diagram $h : x^{\otimes n} \to x^{\otimes n}$ such that $\partial d = ih \mathbin{⨾} d$.*

Proof. Given the diagram for a unitary group d, we compute its diagrammatic differentiation ∂d and get h by pattern matching. Conversely given a self-adjoint h, the diagram $d = \exp(i\theta h)$ is a unitary group. □

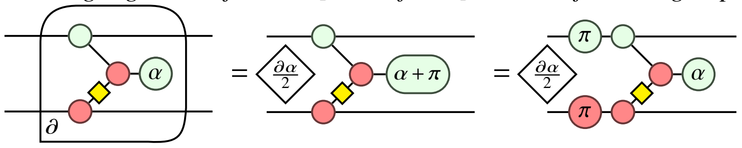
Example 3.14. *Let $d = R_z(\alpha) \otimes R_x(\alpha)$ for a smooth $\alpha : \mathbb{R} \to \mathbb{R}$, then the following implies $d(\theta) = \exp(i\theta h)$*



for $h = -i\frac{\partial\alpha}{2}(Z \otimes I + I \otimes X)$.

Example 3.15. *Let $d = P(\alpha, ZX)$ be a Pauli gadget as defined in [33, def. 4.1] then the following implies*

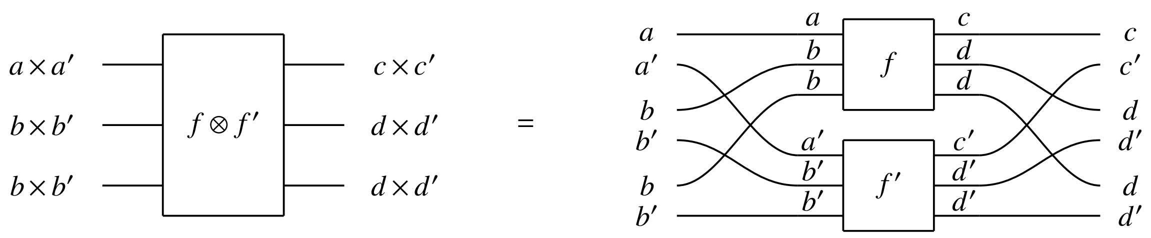
$d(\theta) = \exp(i\theta h)$ for $h = -i\frac{\partial\alpha}{2}Z \otimes X$.



4 Differentiating quantum circuits

In this section, we extend diagrammatic differentiation to classical-quantum circuits. These circuit diagrams have two kinds of wires for bits and qubits, and boxes for pure quantum processes, measurements and preparations. We interpret these classical-quantum circuits in terms of parametrised matrices, where the tensor product reorders the indices to keep the classical and quantum dimensions in order. Borrowing the term from Coecke and Kissinger [31], we call these matrices cq-maps. In this context, diagrammatic derivations correspond to the notion of gradient recipe for parametrised quantum gates [34].

We first give the definition of parametrised cq-maps which is at the basis of our Python implementation. The category $\mathbf{CQMap}_n$ has objects given by pairs of natural numbers $\mathrm{Ob}(\mathbf{CQMap}_n) = \mathbb{N}\times\mathbb{N}$, where the first and second element of the pair encode the classical and the quantum dimension of the system respectively. Arrows $f : (a,b) \to (c,d)$ are given by $a\times b^2 \to c\times d^2$ parametrised complex matrices, i.e. with entries in $\mathbb{R}^n \to \mathbb{C}$. Composition of cq-maps is given by multiplying their underlying matrices. Tensor is given on objects by pointwise multiplication and on arrows by the following diagram in $\mathbf{Mat}_{\mathbb{R}^n\to\mathbb{C}}$:



Each pure map $f : a \to b$ in $\mathbf{Mat}_{\mathbb{R}^n\to\mathbb{C}}$ embeds as a cq-map $(1,a) \to (1,b)$ by "doubling", i.e. tensoring with its complex conjugate $f \mapsto \bar{f}\otimes f$. Note that doubling is faithful up to a global phase. For each dimension $a \in \mathbb{N}$, there are distinguished cq-maps $M_a : (1,a) \to (a,1)$, $E_a : (a,1) \to (1,a)$ for measurement and preparation in the computational basis with matrices given by $M_a = \sum_{i<a} |i\rangle\langle i,i|$ and $E_a = \sum_{i<a} |i,i\rangle\langle i|$. The sum of two cq-maps is given by entrywise addition of their underlying matrix. Note that doubling does not preserve sums, i.e. $\overline{(\sum_i f_i)}\otimes(\sum_i f_i) \neq \sum_i(\overline{f_i}\otimes f_i)$. In quantum mechanical terms, this corresponds to the distinction between quantum superposition and probabilistic mixing.

Remark 4.1. *The cq-maps we have defined here differ from [31] in two minor ways. First, we take the algebraic conjugate rather than the diagrammatic conjugate, i.e. we take $\overline{f\otimes g} = \bar{f}\otimes\bar{g} \neq \bar{g}\otimes\bar{f}$. This is just a choice of convention that makes numerical computation easier. Second, our category* **CQMap** *contains matrices that have no physical interpretation, e.g. we do not ask for complete positivity. This can be fixed by considering the subcategory in the image of the interpretation functor defined below.*

Take a monoidal signature Σ with one object $\Sigma_0 = \{q\}$ interpreted as a qubit, and boxes interpreted as pure quantum processes with n parameters. That is, we fix a parametrised interpretation functor $[\![-]\!] : \mathbf{C}_\Sigma \to \mathbf{Mat}_{\mathbb{R}^n\to\mathbb{C}}$ with $[\![q]\!] = 2$. This could be the signatures for parametrised or algebraic ZX from the previous section, or any universal quantum gate set plus boxes for scalars, bras and kets. We define an extended signature $cq(\Sigma) \supset \Sigma$ with two objects $cq(\Sigma)_0 = \{c,q\}$ interpreted as bit and qubit respectively. Boxes are given by $cq(\Sigma)_1 = \{\hat{f} : q^{\otimes a} \to q^{\otimes b} \mid f \in \Sigma_1\} + \{M : q \to c,\ E : c \to q\}$. Let $\mathbf{C}_{cq(\Sigma)}$ be the free monoidal category it generates, i.e. arrows are classical-quantum circuits. Their interpretation is given by a monoidal functor $[\![-]\!] : \mathbf{C}_{cq(\Sigma)} \to \mathbf{CQMap}_n$ with $[\![c]\!] = (2,1)$ and $[\![q]\!] = (1,2)$ on objects. On arrows we define $[\![M]\!] = M_2$, $[\![E]\!] = E_2$ and $[\![\hat{f}]\!] = \overline{[\![f]\!]}\otimes[\![f]\!]$. We write $cq(\mathbf{ZX}_n)$ for the category of classical-quantum circuits with parametrised ZX diagrams as pure processes.

Let $\mathbf{C}^+_{cq(\Sigma)}$ be the free monoidal category with sums, i.e. arrows are bags of circuits. Again, we want to find a diagrammatic derivation $\partial : \mathbf{C}^+_{cq(\Sigma)} \to \mathbb{D}[\mathbf{C}^+_{cq(\Sigma)}]$ which commutes with the interpretation, i.e.

such that $[\![\partial \hat{f}]\!] = \partial[\![\hat{f}]\!] = \partial\big(\overline{[\![f]\!]} \otimes [\![f]\!]\big)$ for all pure maps $f \in \Sigma_1$. Note that a diagrammatic derivation for pure processes in $\mathbf{C}^+_\Sigma$ does not in general lift to one for classical-quantum circuits in $\mathbf{C}_\Sigma$. Indeed, using the product rule we get $\partial\big(\overline{[\![f]\!]} \otimes [\![f]\!]\big) \; = \; \partial\overline{[\![f]\!]} \otimes [\![f]\!] \; + \; \overline{[\![f]\!]} \otimes \partial[\![f]\!] \;\; \neq \;\; \overline{[\![\partial f]\!]} \otimes [\![\partial f]\!]$.

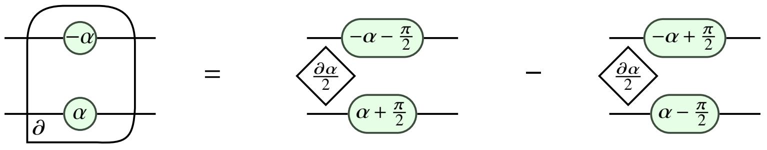
Hence we need equations, called gradient recipes, to rewrite the gradient of a pure map $\partial[\![\hat{f}]\!]$ as the pure map of a gradient $[\![\partial \hat{f}]\!]$. In the special case of Hermitian operators with at most two unique eigenvalues, gradient recipes are given by the parameter-shift rule. In the general case where the parameter-shift rule does not apply, gradient recipes require the introduction of an ancilla qubit.

Theorem 4.2 (Schuld et al.)**.** *For a one-parameter unitary group f with $[\![f(\theta)]\!] = \exp(i\theta H)$, if H has at most two eigenvalues $\pm r$, then there is a shift $s \in [0, 2\pi)$ such that $[\![r\big(f(\theta+s) - f(\theta-s)\big)]\!] = \partial[\![f(\theta)]\!]$.*

Proof. The shift is given by $s = \frac{\pi}{4r}$, see the Taylor expansion given in [34, Theorem 1]. □

Corollary 4.3. *Classical-quantum circuits $cq(\mathbf{ZX}_n)$ with parametrised ZX diagrams as pure processes admit diagrammatic differentiation.*

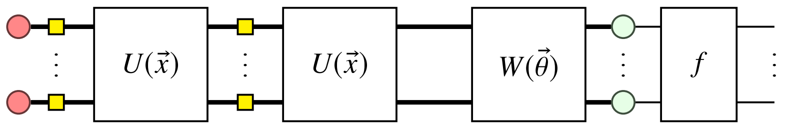
Proof. The Z rotation has eigenvalues ± 1, hence the spiders with two legs have diagrammatic differentiation given by the parameter-shift rule:


$-\alpha$
α
∂
$=$
$-\alpha - \frac{\pi}{2}$
$\frac{\partial\alpha}{2}$
$\alpha + \frac{\pi}{2}$
$-$
$-\alpha + \frac{\pi}{2}$
$\frac{\partial\alpha}{2}$
$\alpha - \frac{\pi}{2}$

As for theorem 3.6, this extends to arbitrary-many legs using spider fusion. □

Remark 4.4. *All scalars in $cq(\mathbf{ZX}_n)$ are non-negative real numbers. Thus in order to encode the substraction of the parameter shift-rule diagrammatically, we need either to consider formal sums with minus signs (a.k.a. enrichment in Abelian groups) or simply to extend the signature with the -1 scalar.*

Example 4.5. *The quantum enhanced feature spaces of [35] are parametrised classical-quantum circuits. The quantum classifier can be drawn as a diagram:*



where $U(\vec{x})$ depends on the input, $W(\vec{\theta})$ depends on the trainable parameters and f is a fixed Boolean function encoded as a linear map.

5 Bubbles and the chain rule

This section introduces an extension of the language of string diagrams that encodes arbitrary non-linear operators on matrices: bubbles. Previous sections already used two kinds of bubbles informally: matrix exponentials and gradients. We give a formal definition of bubbles and their gradients with the chain rule. We then use them to compute the gradient of hybrid classical-quantum circuits where the measurement results can be post-processed by any classical feed-forward neural network.

Fix a set of colours C. Take a monoidal signature Σ, we construct the free monoidal category with sums and bubbles $\mathbf{C}^+_{\beta(\Sigma)}$, i.e. arrows are formal sums of diagrams with bubbles. We define the signature of bubbled diagrams as a union $\beta(\Sigma) = \bigcup_{n \in \mathbb{N}} \beta(\Sigma, n)$ where the signature of $(\leq n)$-nested bubbles $\beta(\Sigma, n)$ is defined by induction:

$$\beta(\Sigma, 0) \;=\; \Sigma \qquad \text{and} \qquad \beta(\Sigma, n+1) \;=\; \left\{ \beta^c(d) : x \to y \;\mid\; c \in C,\; d : x \to y \in \mathbf{C}^+_{\beta(\Sigma,n)} \right\}$$

That is, we put a formal sum of diagrams $d \in \mathbf{C}^+_{\beta(\Sigma,n)}$ with $(\leq n)$-nested bubbles inside a c-coloured bubble and take it as a box $\beta^c(d) \in \beta(\Sigma, n+1)$ for diagrams with $(n+1)$-nested bubbles. We say a monoidal category $\mathbf{C}$ has bubbles when it comes equipped with a unary operator on homsets $\beta^c : \coprod_{x,y} \mathbf{C}(x,y) \to \mathbf{C}(x,y)$ for each colour $c \in C$. Although it makes the bureaucracy heavier, we may consider bubbles that change the domain and codomain of the diagram inside. Such a bubble is defined by two operators on objects $\beta^c_{\mathtt{dom}}, \beta^c_{\mathtt{cod}} : \mathrm{Ob}(\mathbf{C}) \to \mathrm{Ob}(\mathbf{C})$ and an operator on homsets $\beta^c : \coprod_{x,y} \mathbf{C}(x,y) \to \mathbf{C}(\beta^c_{\mathtt{dom}}(x), \beta^c_{\mathtt{cod}}(y))$.

Example 5.1. *Bubbles first appear in Penrose and Rindler [2] where they are used to encode the covariant derivative. An extra wire comes in the bubble to encode the dimension of the tangent vector.*

Example 5.2. *The functorial boxes of Melliès [36] can be thought of as well-behaved bubbles, i.e. such that the composition of bubbles is the bubble of the composition. Indeed, a functor $F : \mathbf{C} \to \mathbf{D}$ between two categories $\mathbf{C}$ and $\mathbf{D}$ defines a bubble on the subcategory of their coproduct $\mathbf{C} \coprod \mathbf{D}$ spanned by $\mathbf{C}$.*

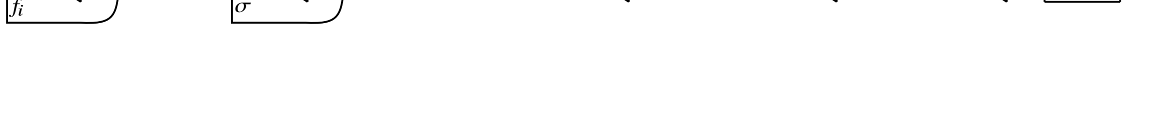
Example 5.3. *Bubbles appear under the name "uooh" (unary operator on homsets) in [37] where they are used to encode the sep lines of C.S.Peirce's existential graphs. Take the predicates of a first-order logic as signature, i.e. one generating object x and each predicate P with arity k as a box with $\mathtt{dom}(P) = 1$ and $\mathtt{cod}(P) = x^{\otimes k}$. Add generators for spiders to encode lines of identity. Then bubbled diagrams encode first-order logic formulae, and every formula can be represented in this way. Logical deduction rules may be given entirely in terms of diagrammatic rules. The evaluation of first-order logic formulae is a bubble-preserving functor $F : \mathbf{C}_{B(\Sigma)} \to \mathbf{Mat}_{\mathbb{B}}$, where bubbles are interpreted as pointwise negation.*

Example 5.4. *Take colours to be arbitrary rig-valued functions $\mathbb{S} \to \mathbb{S}$, then the category of matrices $\mathbf{Mat}_{\mathbb{S}}$ has bubbles given by pointwise application. Gradient bubbles $\partial : \mathbb{S} \to \mathbb{S}$ are a special case.*

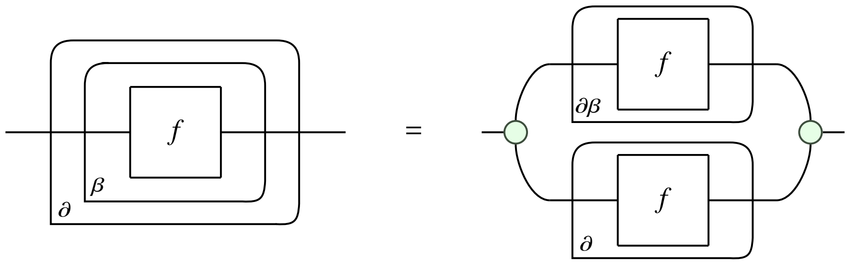
Example 5.5. *In the subcategory of square matrices, matrix exponential is an example of bubble for $\mathbb{S} = \mathbb{R}, \mathbb{C}$. When $\mathbb{S} = \mathbb{B}$, square matrices are finite graphs and reflexive transitive closure is an example.*

Example 5.6. *Bubbles can encode the standard non-linear operators used in machine learning. The sigmoid $\sigma(x) = 1/(1 + e^{-x})$ and rectified linear unit $\sigma(x) = \max(0, x)$ are pointwise bubbles $\sigma : \mathbb{R} \to \mathbb{R}$. The softmax function $\sigma : \mathbb{R}^n \to \mathbb{R}^n$ takes a vector $\vec{x}$, applies exponential pointwise then normalises by $\sum_{i<n} e^{\vec{x}_i}$. It can be drawn as a bubble around the diagram for the vector $\vec{x}$. Bubbles may also depend on the labels from the dataset. Take a loss function such as the relative entropy $l(\vec{y}, \vec{y}^\star) = \sum_{i<m} \vec{y}_i \log(\vec{y}_i / \vec{y}^\star_i)$. The partially-applied loss function $l(-, \vec{y}^\star) : \mathbb{R}^m \to \mathbb{R}$ for the label $\vec{y}^\star \in \mathbb{R}^m$ can be drawn as a bubble around the diagram for the prediction $\vec{y} \in \mathbb{R}^m$.*

Bubbles compose by nesting, this defines a category of post-processes $pp(\mathbf{C})$. The objects are pairs of objects from $\mathbf{C}$, arrows $(x,y) \to (x',y')$ are c-coloured bubbles such that $\beta^c_{\mathtt{dom}}(x) = x'$ and $\beta^c_{\mathtt{cod}}(y) = y'$. If we apply this to the category of matrices, $pp(\mathbf{Mat}_{\mathbb{S}})$ is the category of all matrix-valued functions. In particular, this includes any feed-forward neural networks. Indeed, take $f = f_n \circ \cdots \circ f_1 : \mathbb{R}^a \to \mathbb{R}^b$ where each layer is given by $f_i(\vec{x}_i) = \sigma(W_i \vec{x}_i + \beta_i)$ for the input vector $\vec{x}_i : 1 \to a_i$, the parametrised weight matrix $W_i : a_i \to b_i$ and bias vector $\beta_i : 1 \to b_i$ in $\mathbf{Mat}_{\mathbb{R}^n \to \mathbb{R}}$ for n the total number of parameters. Drawing both f_i and $\sigma : \mathbb{R} \to \mathbb{R}$ as bubbles we get the following definition:



When the bubble β has a derivative $\partial\beta$, we may define the gradient of bubbled diagrams with the chain rule $\partial(\beta(f)) = (\partial\beta)(f) \times \partial f$. In order to make sense of the multiplication, we assume that the homsets of our category $\mathbf{C}$ have a product on homsets which is compatible with the sum, i.e. each homset forms a rig[3] and which commutes with the tensor, i.e. $(f \times f') \otimes (g \times g') = (f \otimes g) \times (f' \otimes g')$. The category of matrices $\mathbf{Mat}_{\mathbb{S}}$ over a rig $\mathbb{S}$ is an example, each homset $\mathbf{Mat}_{\mathbb{S}}(m,n)$ is a rig with entrywise sums and products. Another example is the category of diagrams with spiders on each object, where the product is given by pre/post-composition with the co/monoid structure. We get the following diagrammatic equation:



For scalar diagrams, spiders are empty diagrams and the equation simplifies to the usual chain rule.

Thus, we can draw both a parametrised quantum circuit and its classical post-processing as one bubbled diagram in $cq(\mathbf{ZX}_n)$. By applying the product rule to the quantum circuit and the chain rule to its post-processing, we can compute a diagram for the overall gradient. This applies to parametrised quantum circuits seen as machine learning models [38], to the patterns of measurement-based quantum computing seen as ZX-diagrams [39] as well as the quantum natural language processing of [19].

Conclusion, implementation & future work

We introduced diagrammatic differentiation for tensor calculus, using bubbles to represent the partial derivative of a subdiagram. The product rule allows to compute the gradient of a diagram from the gradient of its boxes. Applying this to ZX diagrams, we showed how to compute the gradient of any linear map with respect to a phase parameter. We then extended this to quantum circuits with the parameter-shift rule and to neural networks with the chain rule.

Although this work focused on the theoretical foundations of diagrammatic differentiation, we briefly describe its implementation as part of the open-source DisCoPy library [21]. A notebook with examples is available in the documentation[4]. The `cqmap` module implements classical-quantum maps as NumPy arrays [40], with SymPy [41] symbols as parameters. The two modules `zx` and `circuit` build upon `monoidal`, the implementation of diagrams in monoidal categories. They both come with an `eval` method which evaluates a diagram as a NumPy array and a `grad` method which returns a formal sum of diagrams given a SymPy symbol. The `zx` module comes with back-and-forth translations with the PyZX library [22] for automated diagram simplification. The `circuit` module interfaces with the tket compiler [23], allowing to execute the diagrams for circuits and their gradient on quantum hardware.

For now, we have only defined gradients of diagrams with respect to one parameter at a time. In future work, we plan to extend our definition to compute the Jacobian of a tensor with respect to a vector of variables. Other promising directions for research include the study of diagrammatic differential equations, as well as a definition of integration for diagrams.

[3] We do not assume that products are compatible with composition, in other words $\mathbf{C}$ need not be rig-enriched.

[4] https://discopy.readthedocs.io/en/main/notebooks/diag-diff.html

Acknowledgements

The authors would like to thank the members of the Oxford quantum group for their insightful feedback. Special thanks go to Stefano Gogioso for developing the idea of diagrammatic differentiation with RY during his MSc project [17]. We thank the QPL reviewers for their constructive feedback which improved the presentation of this work. We also thank Nicola Mariella for his contribution to the implementation. AT thanks Simon Harrison for the Wolfson Harrison Quantum Foundation Scholarship.

References

[1] Roger Penrose. Applications of Negative Dimensional Tensors. *Scribd*, 1971.

[2] Roger Penrose and Wolfgang Rindler. *Spinors and Space-Time: Volume 1, Two-Spinor Calculus and Relativistic Fields*. Cambridge University Press, 1984.

[3] André Joyal and Ross Street. Planar diagrams and tensor algebra. *Unpublished manuscript, available from Ross Street's website*, 1988.

[4] André Joyal and Ross Street. The geometry of tensor calculus, I. *Advances in Mathematics*, 88(1):55–112, July 1991.

[5] Samson Abramsky and Bob Coecke. Categorical quantum mechanics. *arXiv:0808.1023 [quant-ph]*, August 2008.

[6] Bob Coecke and Ross Duncan. Interacting Quantum Observables. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming*, Lecture Notes in Computer Science, pages 298–310. Springer Berlin Heidelberg, 2008.

[7] Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. A Complete Axiomatisation of the ZX-Calculus for Clifford+T Quantum Mechanics. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 559–568, New York, NY, USA, 2018. ACM.

[8] Amar Hadzihasanovic, Kang Feng Ng, and Quanlong Wang. Two Complete Axiomatisations of Pure-state Qubit Quantum Computing. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 502–511, New York, NY, USA, 2018. ACM.

[9] Aleks Kissinger and John van de Wetering. Reducing T-count with the ZX-calculus. *Physical Review A*, 102(2):022406, August 2020.

[10] Ross Duncan, Aleks Kissinger, Simon Perdrix, and John van de Wetering. Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus. *Quantum*, 4:279, June 2020.

[11] Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. Fast and effective techniques for T-count reduction via spider nest identities. *arXiv:2004.05164 [quant-ph]*, April 2020.

[12] Alexander Cowtan, Will Simmons, and Ross Duncan. A Generic Compilation Strategy for the Unitary Coupled Cluster Ansatz. *arXiv:2007.10515 [quant-ph]*, August 2020.

[13] Arianne Meijer-van de Griend and Ross Duncan. Architecture-aware synthesis of phase polynomials for NISQ devices. *arXiv:2004.06052 [quant-ph]*, April 2020.

[14] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski, and John van de Wetering. There and back again: A circuit extraction tale. *arXiv:2003.01664 [quant-ph]*, March 2020.

[15] Nicholas Chancellor, Aleks Kissinger, Joschka Roffe, Stefan Zohren, and Dominic Horsman. Graphical Structures for Design and Verification of Quantum Error Correction. *arXiv:1611.08012 [quant-ph]*, January 2018.

[16] Craig Gidney and Austin G. Fowler. Flexible layout of surface code computations using AutoCCZ states. *arXiv:1905.08916 [quant-ph]*, May 2019.

[17] Richie Yeung. Diagrammatic Design and Study of Ans\"{a}tze for Quantum Machine Learning. *arXiv:2011.11073 [quant-ph]*, November 2020.

[18] Chen Zhao and Xiao-Shan Gao. Analyzing the barren plateau phenomenon in training quantum neural network with the ZX-calculus. *arXiv:2102.01828 [quant-ph]*, February 2021.

[19] Konstantinos Meichanetzidis, Stefano Gogioso, Giovanni De Felice, Nicolò Chiappori, Alexis Toumi, and Bob Coecke. Quantum Natural Language Processing on Near-Term Quantum Computers. In *Quantum Physics and Logic (QPL)*, 2020.

[20] Bob Coecke, Giovanni de Felice, Konstantinos Meichanetzidis, and Alexis Toumi. Foundations for Near-Term Quantum Natural Language Processing. *ArXiv e-prints*, 2020.

[21] Giovanni de Felice, Alexis Toumi, and Bob Coecke. DisCoPy: Monoidal Categories in Python. In *Proceedings of the 3rd Annual International Applied Category Theory Conference, ACT*, volume 333. EPTCS, 2020.

[22] Aleks Kissinger and John van de Wetering. PyZX: Large Scale Automated Diagrammatic Reasoning. *arXiv:1904.04735 [quant-ph]*, April 2019.

[23] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. Tket : A Retargetable Compiler for NISQ Devices. *arXiv:2003.10611 [quant-ph]*, March 2020.

[24] Joon-Hwi Kim, Maverick S. H. Oh, and Keun-Young Kim. Boosting Vector Calculus with the Graphical Notation. *arXiv:1911.00892 [hep-th, physics:physics]*, January 2020.

[25] R. F. Blute, J. R. B. Cockett, and R. A. G. Seely. Differential categories. *Mathematical Structures in Computer Science*, 16(06):1049, December 2006.

[26] Robin Cockett, Geoffrey Cruttwell, Jonathan Gallagher, Jean-Simon Pacaud Lemay, Benjamin MacAdam, Gordon Plotkin, and Dorette Pronk. Reverse derivative categories. *arXiv:1910.07065 [cs, math]*, October 2019.

[27] G. S. H. Cruttwell, Bruno Gavranović, Neil Ghani, Paul Wilson, and Fabio Zanasi. Categorical Foundations of Gradient-Based Learning. *arXiv:2103.01931 [cs, math]*, March 2021.

[28] William Kingdon Clifford. *A Preliminary Sketch of Biquaternions*. 1873.

[29] Philipp H. W. Hoffmann. A Hitchhiker's Guide to Automatic Differentiation. *Numerical Algorithms*, 72(3):775–811, July 2016.

[30] Quanlong Wang. Completeness of algebraic ZX-calculus over arbitrary commutative rings and semirings. *arXiv:1912.01003 [quant-ph]*, October 2020.

[31] Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, 2017.

[32] M. H. Stone. On one-parameter unitary groups in hilbert space. *Annals of Mathematics*, 33(3):643–648, 1932.

[33] Alexander Cowtan, Silas Dilkes, Ross Duncan, Will Simmons, and Seyon Sivarajah. Phase Gadget Synthesis for Shallow Circuits. *Electronic Proceedings in Theoretical Computer Science*, 318:213–228, May 2020.

[34] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. Evaluating analytic gradients on quantum hardware. *Physical Review A*, 99(3):032331, March 2019.

[35] Vojtech Havlicek, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. Supervised learning with quantum enhanced feature spaces. *Nature*, 567(7747):209–212, March 2019.

[36] Paul-André Melliès. Functorial Boxes in String Diagrams. In Zoltán Ésik, editor, *Computer Science Logic*, Lecture Notes in Computer Science, pages 1–30. Springer Berlin Heidelberg, 2006.

[37] Nathan Haydon and Pawel Sobocinski. Compositional Diagrammatic First-Order Logic. page 16, 2020.

[38] Marcello Benedetti, Erika Lloyd, and Stefan Sack. Parameterized quantum circuits as machine learning models. *arXiv:1906.07682 [quant-ph]*, June 2019.

[39] Ross Duncan and Simon Perdrix. Rewriting measurement-based quantum computations with generalised flow. In *International Colloquium on Automata, Languages, and Programming*, pages 285–296. Springer, 2010.

[40] Stefan van der Walt, S. Chris Colbert, and Gael Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation. *Computing in Science Engineering*, 13(2):22–30, March 2011.

[41] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. SymPy: Symbolic computing in Python. *PeerJ Computer Science*, 3:e103, January 2017.