The Principles of First Order Automatic Differentiation

Philipp H. W. Hoffmann

Abstract

This article provides a short overview of the theory of First Order Automatic Differentiation (AD) for readers unfamiliar with this topic. In particular, we summarize different characterisations of Forward AD, like the vector-matrix based approach, the idea of lifting functions to the algebra of dual numbers, the method of Taylor series expansion on dual numbers and the application of the push-forward operator. We give short, but precise mathematical descriptions of these methods and show why they all reduce to the same actual chain of computations (and are, hence, equivalent). Finally, we give a short summary of Reverse AD and again point out the underlying computational steps.

Keywords: Automatic Differentiation, Forward AD, Reverse AD, Dual Numbers

AMS Subject Classification (2010): 65-02, 65K99

1 Introduction

1.1 Content and Motivation

The evaluation of (often complicated) derivatives is a task that appears often in the work of many researchers in applied mathematics, physics, engineering or, in fact, any natural science. Typically, these computation will be performed with the help of a computer and there are two main distinct methods to determine a derivative of a function: On the one hand, there are numerical methods, usually based on finite differences, which only approximate the sought result and are inherently prone to rounding errors (which already occur due to floating point arithmetic). Computer Algebra systems (like Maple, Mathematica or the older system Maxima), on the other hand, evaluate the derivative symbolically, which may, in certain cases, lead to significantly long computation times.

Automatic Differentiation, also called Algorithmic Differentiation, (short AD) is now a third way to evaluate derivatives, which differs significantly from the two classical methods. The very first article on this method is probably due to Wengert [12]. Two further major publications regarding AD were published by Rall in the 1980s [9], [10] and, since then, there has been a growing community of researcher interested in this topic (see for example [2], [5], [6] or [7]). Of course, the majority of publications on Automatic Differentiation are concerned with certain improvements, extensions or implementations of AD, but usually

provide also short introductions to the topic for the unfamiliar reader. Furthermore, there are also excellent and comprehensive publications which describe the area as a whole (see for example Griewank [3] and Griewank and Walther [4]).

An interested reader new to the theory may find it nevertheless difficult to grasp the essence of Automatic Differentiation. The problem lies in the diversity in which the (actual simple) ideas can be described. While in [3] and [8, section 2] a vector-matrix approach is used, in [5] and [10] AD is defined via a certain multiplication on pairs (namely the multiplication which defines the algebra of *dual numbers*). Similarly, in [11] the lifting of a function to named dual numbers is presented as the main idea of AD, where in [7] this lifted functions is defined as a Taylor Series (on dual numbers). Finally, Manzyuk [6] bases his description on the push-forward operator known from differential geometry and gives a connection to category theory. While some of these descriptions are, in their core, quite similar, at the very least the vector-matrix based approach appears to differ from the remaining approaches quite a lot. For somebody unfamiliar with the theory, this may lead to the (wrong) impression that different authors essentially describe different methods which are only unified under the label of Automatic Differentiation. This is effectively not the case and this article hopes to clarify the situation.

We will in the following give short, but precise, overviews of the distinct descriptions of AD[1] mentioned above and show, why they all are just different expressions of the same principle. It is clear that the purpose of this article is completely educational and there is nothing intrinsically new in our elaborations. Indeed, in particular with regards to [3], we only give a extremely shorted and simplified version of the work in the original publication. Furthermore, there are actually at least two distinct versions, or modes, of AD. The so-called *Forward Mode* and the *Reverse Mode* (along with variants such as *Checkpoint Reverse Mode* [2]). The different descriptions mentioned above all refer to the Forward Mode only. We are, therefore, mainly concerned with Forward AD and will only briefly discuss the standard Reverse Mode at the end of this paper.

In addition, we will restrict ourselves to AD in its simplest form. Namely, First Order AD, that is Automatic Differentiation to compute first order derivatives, of a differentiable, multivariate function $f : X \to \mathbb{R}^m$, on an open set $X \subset \mathbb{R}^n$. Although, there are, of course, version which are able to evaluate higher order derivatives (see for example [7]), we view these as extensions or modifications of the original system and will, therefore, not consider them in this article. The same holds for Nested Forward Automatic Differentiation, which involves a kind of recursive calling of Forward AD (see, for example, [11]). Again, we will not be concerned with this extension in this paper.

1.2 Notation

Unfortunately, when one wants to describe the techniques of AD precisely, things get easily quite messy. In particular, a significant number of indices is necessary, which may make the reading of some parts of this article difficult. The reader should not feel discouraged by this, since the principle are really quite simple. We give, wherever possible, easy examples to bring the basic ideas across.

[1] To be more precise, of the Forward Mode of AD.

Further, we summarize the main results at the end of most section.

As mentioned above, the function we want to differentiate will be denoted by $f : X \to \mathbb{R}^m$ and defined on an open set $X \subset \mathbb{R}^n$. In particular, in this paper n always denotes the number of variables of f, while m denotes the dimension of its co-domain. If the image of f is multi-dimensional, we will use $f^{[j]}$ (instead of the commonly used f_j) for its one-dimensional sub-functions. (This is to avoid having too many sub-indices.) Open domains of functions other than f, will be denoted by U (usually with indices). In Sections 3 and 7, the notation x_i is reserved for variables of the function f, while other variables are denoted by v_i. The symbol c always denotes a fixed value (a constant). For real vectors, we use boldface letters like $\mathbf{x}$ or $\mathbf{c}$ (where the latter will be a constant vector). In particular, $\vec{\mathbf{x}}$ and $\overleftarrow{\mathbf{y}}$ will be (usually fixed) directional vectors or their transposed, respectively. Entries of $\vec{\mathbf{x}}$ or $\overleftarrow{\mathbf{y}}$ will be denoted by x'_i or y'_i, respectively[2].

1.3 The basic idea of Automatic Differentiation

Before we start with the theory, let us demonstrate the ideas of of AD in a very easy case: Let $f, \varphi_1, \varphi_2, \varphi_3 : \mathbb{R} \to \mathbb{R}$ be differentiable functions with $f = \varphi_3 \circ \varphi_2 \circ \varphi_1$. Let further $c, x', y' \in \mathbb{R}$ be real numbers. Assume we want to compute $f'(c) \cdot x'$ or $y' \cdot f'(c)$, respectively.

By the chain rule,

$$f'(c) \cdot x' = \varphi'_3\left(\varphi_2\left(\varphi_1(c)\right)\right) \cdot \varphi'_2\left(\varphi_1(c)\right) \cdot \varphi'_1(c) \cdot x'$$

As one easily sees, the evaluation of $f'(c) \cdot x'$ can be achieved by computing successively the following pairs:

$$\begin{gathered}
(c, x') \\
(\varphi_1(c), \varphi'_1(c) \cdot x') \\
\left(\varphi_2\left(\varphi_1(c)\right),\ \varphi'_2\left(\varphi_1(c)\right) \cdot \varphi'_1(c) x'\right) \\
\left(\varphi_3\left(\varphi_2\left(\varphi_1(c)\right)\right),\ \varphi'_3\left(\varphi_2\left(\varphi_1(c)\right)\right) \cdot \varphi'_2\left(\varphi_1(c)\right) \varphi'_1(c) x'\right)
\end{gathered}$$

and taking the second entry of the final pair. As we see, the first element of each pair appears as an argument of the functions φ_i, φ'_i in the following pair, while the second element appears as a factor (from the right).

Regarding the computation of $y' \cdot f'(c)$, we have obviously

$$y' \cdot f'(c) = y' \cdot \varphi'_3\left(\varphi_2\left(\varphi_1(c)\right)\right) \cdot \varphi'_2\left(\varphi_1(c)\right) \cdot \varphi'_1(c).$$

The computation of this derivative can now be achieved by the computing the following two lists of real numbers:

$$\begin{array}{cc}
c & y' \\
\varphi_1(c) & y' \cdot \varphi'_3\left(\varphi_2\left(\varphi_1(c)\right)\right) \\
\varphi_2(\varphi_1(c)) & y' \varphi'_3\left(\varphi_2\left(\varphi_1(c)\right)\right) \cdot \varphi'_2\left(\varphi_1(c)\right) \\
 & y' \varphi'_3\left(\varphi_2\left(\varphi_1(c)\right)\right) \varphi'_2\left(\varphi_1(c)\right) \cdot \varphi'_1(c)
\end{array}$$

[2]The notation x'_i for entries of $\vec{\mathbf{x}}$ is somewhat historical and based on the idea that, very often, x'_i may be considered as a derivative of either the identity function, or a constant function. For us, however, each $x'_i \in \mathbb{R}$ is simply a chosen real number. The same holds for the notation y'_i.

and taking the last entry of the second list. Here, each entry (apart from y') in the second list consists of values of φ_i' evaluated at an element of the first list (note that the order is reversed) and the previous entry as a factor (from the left).

If now values of the functions $\varphi_1, \varphi_2, \varphi_3$ and their derivatives $\varphi_1', \varphi_2', \varphi_3'$ are already implemented in the system, then the evaluation of the $\varphi_i(c), \varphi_i'(c)$ means simply calling these functions/derivatives, which can be achieved with limited computational time. Thus, the computation of $f'(c) \cdot x'$ and $y' \cdot f'(c)$ is nothing else than obtaining these values, performing some arithmetic operations on real numbers and passing the results on. That is, no actual differentiation takes place to compute the sought numbers. This is the main idea[3] of Automatic Differentiation.

In the following, we describe the setting in general.

2 Preliminaries

As mentioned above, (First Order) Automatic Differentiation, in its simplest form, is concerned with the computation of derivatives of a differentiable function $f : X \to \mathbb{R}^m$, on an open set $X \subset \mathbb{R}^n$. The assumption made is that each $f_j = f^{[j]} : X \to \mathbb{R}$ in

$$f(x_1, ..., x_n) = \begin{pmatrix} f_1(x_1, ..., x_n) \\ \vdots \\ f_m(x_1, ..., x_n) \end{pmatrix} = \begin{pmatrix} f^{[1]}(x_1, ..., x_n) \\ \vdots \\ f^{[m]}(x_1, ..., x_n) \end{pmatrix}$$

consists (to be defined more precisely later) of several sufficiently smooth so-called *elementary* functions $\varphi_i^{[j]} : U_i^{[j]} \to \mathbb{R}$, defined on open sets $U_i^{[j]} \subset \mathbb{R}^{n_i^{[j]}}$, such as addition and multiplications, constant, trigonometric, exponential or logarithmic functions etc. (Here, $i \in I^{[j]}$ for some index set $I^{[j]}$.) In particular, the $\varphi_i^{[j]}$ have to have the property that they and their partial derivatives $\frac{\partial \varphi_i^{[j]}}{\partial v_k}$ (and, therefore, their gradients $\nabla \varphi_i^{[j]}$) are already implemented (as functions) in the system.

Further, AD does not compute the actual mapping $\mathbf{x} \mapsto J_f(\mathbf{x})$, which maps a vector $\mathbf{x} \in X$ to the Jacobian $J_f(\mathbf{x})$ of f at $\mathbf{x}$. Instead, directional derivatives of f or left-hand products of row-vectors with its Jacobian at a fixed vector $\mathbf{c} \in X$ are determined.

That is, given $\mathbf{c} \in X$ and $\overrightarrow{\mathbf{x}} \in \mathbb{R}^n$ or $\overleftarrow{\mathbf{y}} \in \mathbb{R}^{1 \times m}$, we determine either

$$J_f(\mathbf{c}) \cdot \overrightarrow{\mathbf{x}} \qquad \text{or} \qquad \overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c}).$$

(This appears to be a subtle difference, however, while $\mathbf{x} \mapsto J_f(\mathbf{x})$ is a matrix-valued function, $J_f(\mathbf{c}) \cdot \overrightarrow{\mathbf{x}}$ and $\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})$ are vectors or one-row matrices, respectively, in euclidean space.)

The computation of directional derivatives of $J_f(\mathbf{c})$ is referred to as the Forward Mode of AD, or Forward AD (short FAD), while the computation of $\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})$ is referred to as the Reverse Mode of AD, or Reverse AD (RAD).[4]

[3] In our opinion, of course.

[4] As mentioned in the introduction, we will consider mainly the easier Forward Mode in this article.

The first and foremost principle of AD is now that the computation of named derivatives should essentially only involve computations (which means calling) of the elementary functions $\varphi_i^{[j]}$ and their partial derivatives plus some real number arithmetic. Indeed, we may give the following, informal descriptions:

- *Forward Automatic Differentiation is the computation of* $J_f(\mathbf{c}) \cdot \overrightarrow{\mathbf{x}}$ *for fixed* $\mathbf{c} \in X$ *and* $\overrightarrow{\mathbf{x}} \in \mathbb{R}^n$ *through the successive computation of the pairs of real numbers*
$$\left(\varphi_i^{[j]}(\mathbf{c}^{[\mathbf{i},\mathbf{j}]}),\ \nabla \varphi_i^{[j]}(\mathbf{c}^{[\mathbf{i},\mathbf{j}]}) \cdot \overrightarrow{\mathbf{x}}^{[\mathbf{i},\mathbf{j}]} \right) \in \mathbb{R}^2$$
in suitable order, for suitable vectors $\mathbf{c}^{[\mathbf{i},\mathbf{j}]}, \overrightarrow{\mathbf{x}}^{[\mathbf{i},\mathbf{j}]} \in \mathbb{R}^{n_i^{[j]}}$.

- *Reverse Automatic Differentiation is the computation of* $\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})$ *for fixed* $\mathbf{c} \in X$ *and* $\overleftarrow{\mathbf{y}} \in \mathbb{R}^{1 \times m}$ *through the computation of the real numbers*
$$\varphi_i^{[j]}(\mathbf{c}^{[\mathbf{i},\mathbf{j}]}) \in \mathbb{R} \quad \textit{and} \quad v_{i,j,k} + \frac{\partial \varphi_i^{[j]}}{\partial v_k}(\mathbf{c}^{[\mathbf{i},\mathbf{j}]}) \cdot v_{i,j} \ \in \mathbb{R}, \quad k = 1, ..., n_i^{[j]},$$
in suitable order, for suitable vectors $\mathbf{c}^{[i,j]} \in \mathbb{R}^{n_i^{[j]}}$ *and suitable numbers* $v_{i,j,k}, v_{i,j} \in \mathbb{R}$.

(Of course, the vectors and numbers $\mathbf{c}^{[\mathbf{i},\mathbf{j}]}, \overrightarrow{\mathbf{x}}^{[\mathbf{i},\mathbf{j}]}, v_{i,j,k}, v_{i,j}$ will be determined in a certain way; as will be the order in which the computations are performed.)

The advantage of Forward AD in the case of a function of one variable $f : \mathbb{R} \to \mathbb{R}^m$ is clear: If we choose in that case $\overrightarrow{\mathbf{x}} = 1$, we obtain the whole Jacobian of f. Conversely, if $f : \mathbb{R}^n \to \mathbb{R}$ is real valued, Reverse AD is advantageous: If we choose $\overleftarrow{\mathbf{y}} = 1$ in that case, we obtain the whole gradient of f.

As mentioned above, addition and multiplication are considered to be elementary functions. That is, the mappings
$$\text{sum} : \mathbb{R}^2 \to \mathbb{R} \quad \text{with} \quad \text{sum}(x_1, x_2) = x_1 + x_2$$
and
$$\text{prod} : \mathbb{R}^2 \to \mathbb{R} \quad \text{with} \quad \text{prod}(x_1, x_2) = x_1 \cdot x_2$$
are part of the $\varphi_i^{[j]}$, whose partial derivatives are already implemented in the system. (It is clear that sums and products of more than two summands or factors can be expressed by iterating the mappings sum and prod.)

For reasons of convenience, we will not use the notations sum, prod but instead express sums and products using $+$ or $\cdot$ as usual.

Now each $f^{[j]}$ will in general consist of several levels of elementary functions. To express this fact accurately, we will have to use a subtler way of indexing the elementary functions $\varphi_i^{[j]}$. We first give an example, before we describe the (actually trivial) situation formally.

Example 2.1. Consider the function $f : \mathbb{R}^2 \to \mathbb{R}^2$ given by
$$f(x_1, x_2) = \begin{pmatrix} \sin(x_2) + \cos(x_1^2 + 3) \cdot 5x_2 \\ \exp(\sin(x_1)) + 4x_2^3 \end{pmatrix}.$$

In this case, $f^{[1]}$ depends on the following elementary functions:

$$\varphi_{1,1}^{[1]}(x_2) = \sin(x_2),\ \varphi_{1,2}^{[1]}(x_1) = x_1^2,\ \varphi_{1,3}^{[1]}(x_2) = 5x_2,$$
$$\varphi_{2,1}^{[1]}(x_1^2) = x_1^2 + 3,$$
$$\varphi_{3,1}^{[1]}(x_1^2 + 3) = \cos(x_1^2 + 3),$$
$$\varphi_{4,1}^{[1]}(\cos(x_1^2 + 3), 5x_2) = \cos(x_1^2 + 3) \cdot 5x_2,$$
$$\varphi_{5}^{[1]}(\sin(x_2), \cos(x_1^2 + 3) \cdot 5x_2) = \sin(x_2) + \cos(x_1^2 + 3) \cdot 5x_2 = f^{[1]}(x_1, x_2).$$

Analogously, for $f^{[2]}$:

$$\varphi_{1,1}^{[2]}(x_1) = \sin(x_1),\ \varphi_{1,2}^{[2]}(x_2) = x_2^2,$$
$$\varphi_{2,1}^{[2]}(\sin(x_1)) = \exp(\sin(x_1)),\ \varphi_{2,3}^{[2]}(x_2, x_2^2) = x_2^2 \cdot x_2 = x_2^3,$$
$$\varphi_{3,1}^{[2]}(x_2^3) = 4x_2^3,$$
$$\varphi_{4}^{[2]}(\exp(\sin(x_1)), 4x_2^3) = \exp(\sin(x_1)) + 4x_2^3 = f^{[2]}(x_1, x_2).$$

The general situation is now that each $f^{[j]}$ may depend on elementary functions

$$\varphi_{1,1}^{[j]}, ..., \varphi_{1,k_{j,1}}^{[j]},\ \varphi_{2,1}^{[j]}, ..., \varphi_{2,k_{j,2}}^{[j]},\quad ...\quad , \varphi_{\ell_j-1,1}^{[j]}, ..., \varphi_{\ell_j-1,k_{j,\ell_j-1}}^{[j]},\quad \varphi_{\ell_j}^{[j]},$$

defined on open sets $U_{\nu,i}^{[j]} \subset \mathbb{R}^{n_{\nu,i}^{[j]}}$, $\nu = 1, ..., \ell_j - 1$, $i = 1, ..., k_{j,\nu}$, and $U_{\ell_j}^{[j]} \subset \mathbb{R}^{n_{\ell_j}^{[j]}}$, with the properties that

$$f^{[j]}(x_1, ..., x_n) = \varphi_{\ell_j}^{[j]}(u_{\ell_j,1}^{[j]}, ..., u_{\ell_j,l_{\ell_j}}^{[j]}) \qquad \text{with}$$

$$u_{\ell_j,1}^{[j]}, ..., u_{\ell_j,l_{\ell_j}}^{[j]}$$
$$\in \left\{x_1, ..., x_n, \varphi_{1,1}^{[j]}(\cdots), ..., \varphi_{1,k_{j,1}}^{[j]}(\cdots),\quad ...\quad , \varphi_{\ell_j-1,1}^{[j]}(\cdots), ..., \varphi_{\ell_j-1,k_{j,\ell_j-1}}^{[j]}(\cdots)\right\},$$

where

$$\varphi_{\ell_j-1,i}^{[j]}(\cdots) = \varphi_{\ell_j-1,i}^{[j]}(u_{\ell_j-1,i,1}^{[j]}, ..., u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]}) \quad \text{with}$$

$$u_{\ell_j-1,i,1}^{[j]}, ..., u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]}$$
$$\in \left\{x_1, ..., x_n, \varphi_{1,1}^{[j]}(\cdots), ..., \varphi_{1,k_{j,1}}^{[j]}(\cdots),\quad ...\quad , \varphi_{\ell_j-2,1}^{[j]}(\cdots), ..., \varphi_{\ell_j-2,k_{j,\ell_j-2}}^{[j]}(\cdots)\right\},$$

where

$$\varphi_{\ell_j-2,i}^{[j]}(\cdots) = \varphi_{\ell_j-2,i}^{[j]}(u_{\ell_j-2,i,1}^{[j]}, ..., u_{\ell_j-2,i,l_{\ell_j-2,i}}^{[j]}) \quad \text{with}$$

$$u_{\ell_j-2,i,1}^{[j]}, ..., u_{\ell_j-2,i,l_{\ell_j-2,i}}^{[j]}$$
$$\in \left\{x_1, ..., x_n, \varphi_{1,1}^{[j]}(\cdots), ..., \varphi_{1,k_{j,1}}^{[j]}(\cdots),\quad ...\quad , \varphi_{\ell_j-3,1}^{[j]}(\cdots), ..., \varphi_{\ell_j-3,k_{j,\ell_j-3}}^{[j]}(\cdots)\right\},$$

$$\vdots$$

where

$$\varphi_{2,i}^{[j]}(...) = \varphi_{2,i}^{[j]}(u_{2,i,1}^{[j]}, ..., u_{2,i,l_{2,i}}^{[j]}) \qquad \text{with}$$

$$u_{2,i,1}^{[j]}, ..., u_{2,i,l_{2,i}}^{[j]}$$
$$\in \left\{x_1, ..., x_n, \varphi_{1,1}^{[j]}(u_{1,1,1}^{[j]}..., u_{1,1,l_{1,1}}^{[j]}), ..., \varphi_{1,k_{j,1}}^{[j]}(u_{1,k_{j,1},1}^{[j]}..., u_{1,k_{j,1},l_{1,k_{j,1}}}^{[j]})\right\},$$
$$\text{where} \quad u_{1,i,1}^{[j]}, ..., u_{1,i,l_{1,i}}^{[j]} \in \{x_1, ..., x_n\},$$

for all $x_1, ..., x_n \in X$ and for $i = 1, ..., k_{j,\nu}$, $\nu = 1, ..., \ell_j - 1$.

This description is somewhat necessary for formally discussing the Forward Mode of AD using the algebra of dual numbers (see Section 4). It is indeed not too important for the approach we will discuss first.

3 Forward AD—An elementary approach

In this approach, the function f is described as a composition of multi-variate and multi-dimensional mappings. Differentiating this composition to obtain $J_f(\mathbf{c}) \cdot \vec{\mathbf{x}}$, for given $\mathbf{c} \in X \subset \mathbb{R}^n$ and $\vec{\mathbf{x}} \in \mathbb{R}^n$, leads, by the chain rule, to a product of matrices. This method has, for example, been described in [8, section 2] and, comprehensively, in [3] and [4]. We follow mainly the notation of [3].

The simple idea is to express f as a composition of the form

$$f = P_Y \circ \Phi_\mu \circ \cdots \circ \Phi_1 \circ P_X.$$

Here, $P_X : X \to H$ is the (linear) natural embedding of the domain $X \subset \mathbb{R}^n$ into the so-called *state space* $H = \mathbb{R}^{\dim H}$, where $\dim H = n + \mu$ for μ being the total number of elementary functions $\varphi_{\nu,k}^{[j]}, \varphi_{\ell_j}^{[j]}$ which make up the function f. Each $\Phi_i : H \to H$, referred to as an *elementary transition*, corresponds to exactly one such elementary function. The mapping $P_Y : H \to Y = f(U)$ is some suitable linear projection of H down onto $Y = f(U) \subset \mathbb{R}^m$.

Determining now $J_f(\mathbf{c}) \cdot \vec{\mathbf{x}}$ for fixed $\mathbf{c} \in X$ and fixed $\vec{\mathbf{x}} \in \mathbb{R}^n$ becomes, by the chain rule, the evaluation of the matrix-vector product

$$J_f(\mathbf{c}) \cdot \vec{\mathbf{x}} = P_Y \cdot \Phi'_{\mu,\mathbf{c}} \ \cdots \ \Phi'_{1,\mathbf{c}} \cdot P_X \cdot \vec{\mathbf{x}}, \tag{3.1}$$

where $\Phi'_{i,\mathbf{c}}$ denotes the Jacobian of Φ_i at $(\Phi_{i-1} \circ \cdots \circ \Phi_1 \circ P_X)(\mathbf{c})$.

Each computation of $\Phi'_{i,\mathbf{c}}$ should now only involve one computation of (some directional derivative of) the gradient $\nabla\varphi_i(\mathbf{c}^{[i]})$ of some elementary function φ_i (equal to $\varphi_{\nu,k}^{[j]}$ or $\varphi_{\ell_j}^{[j]}$ for some j, ν, k) at some $\mathbf{c}^{[i]}$. Again, the computation of the $\nabla\varphi_i(\mathbf{c}^{[i]})$ simply reduces to calling partial derivatives, and, similarly, computing the numbers $\mathbf{c}^{[i]}$ shall also only require the calling of some other elementary function φ_l. This way, $J_f(\mathbf{c}) \cdot \vec{\mathbf{x}}$ is computed 'automatically' and, at no time in this process, any actual differentiation takes place.

The process is now performed in a particular ordered fashion, which we describe in the following.

The evaluation of f at some point $\mathbf{x} = (x_1, ..., x_n)$ can be described by a so-called *evaluation trace* $\mathbf{v}^{[0]} = \mathbf{v}^{[0]}(\mathbf{x}), ..., \mathbf{v}^{[\mu]} = \mathbf{v}^{[\mu]}(\mathbf{x})$, where each $\mathbf{v}^{[i]} \in H$

is a so-called *state vector*, representing the state of the evaluation after i steps. More precisely, we set

$$\mathbf{v}^{[0]} := P_X(x_1, ..., x_n) = (x_1, ..., x_n, 0, ..., 0) \quad \text{and} \quad \mathbf{v}^{[i]} = \Phi_i(\mathbf{v}^{[i-1]}), \quad i = 1, ..., \mu.$$

The elementary transitions Φ_i are now given by imposing an ordering on the $\varphi_{\nu,k}^{[j]}, \varphi_{\ell_j}^{[j]}$, such that φ_i is the i-th elementary function with respect to this order,

$$\text{and by setting} \quad \Phi_i \begin{pmatrix} v_1 \\ \vdots \\ v_{n+\mu} \end{pmatrix} = \begin{pmatrix} v_1 \\ \vdots \\ v_{n+i-1} \\ \varphi_i(v_{i_1}, ..., v_{i_{n_i}}) \\ v_{n+i+1} \\ \vdots \\ v_{n+\mu} \end{pmatrix}, \text{ for all } \begin{pmatrix} v_1 \\ \vdots \\ v_{n+\mu} \end{pmatrix} \in H,$$

and $v_{i_1}, ..., v_{i_{n_i}} \in \{v_1, ..., v_{n+i-1}\} \cap U_i$, where $U_i \subset \mathbb{R}^{n_i}$ is the open domain of φ_i. Note that this is not a definition in the strict sense, since we do not specify the arguments $v_{i_1}, ..., v_{i_{n_i}}$ of φ_i. These will depend on the actual functions f and φ_i. (Compare the example below.)

Therefore, we have

$$\mathbf{v}^{[i]}(\mathbf{x}) = \Phi_i(\mathbf{v}^{[i-1]}(\mathbf{x})) = \begin{pmatrix} \mathbf{v}_1^{[i-1]}(\mathbf{x}) = x_1 \\ \vdots \\ \mathbf{v}_n^{[i-1]}(\mathbf{x}) = x_n \\ \vdots \\ \mathbf{v}_{n+i-1}^{[i-1]}(\mathbf{x}) \\ \varphi_i(v_{i_1}(\mathbf{x}), ..., v_{i_{n_i}}(\mathbf{x})) \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} \mathbf{v}_1^{[i-1]} = x_1 \\ \vdots \\ \mathbf{v}_n^{[i-1]} = x_n \\ \vdots \\ \mathbf{v}_{n+i-1}^{[i-1]} \\ \varphi_i(v_{i_1}, ..., v_{i_{n_i}}) \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

for $v_{i_1} = v_{i_1}(\mathbf{x}), ..., v_{i_{n_i}} = v_{i_{n_i}}(\mathbf{x}) \in \{\mathbf{v}_1^{[i-1]}, ..., \mathbf{v}_{n+i-1}^{[i-1]}\} \cap U_i$.

It is clear that, for the above to make sense, the ordering imposed on the $\varphi_{\nu,k}^{[j]}, \varphi_{\ell_j}^{[j]}$ must have the property that all arguments in $\varphi_i(v_{i_1}, ..., v_{i_{n_i}})$ have already been evaluated.

The definition of the projection $P_Y : H \to Y = f(U) \subset \mathbb{R}^m$ depends on the ordering imposed on the $\varphi_{\nu,k}^{[j]}, \varphi_{\ell_j}^{[j]}$. If this ordering is such that we have for the top-level elementary functions $\varphi_{\ell_1}^{[1]} = \varphi_{n+\mu-m}$, ..., $\varphi_{\ell_m}^{[m]} = \varphi_{n+\mu}$, then we will have

$$f^{[1]}(x_1, ..., x_n) = \mathbf{v}_{n+\mu-m}^{[\mu]}, ..., \quad f^{[m]}(x_1, ..., x_n) = \mathbf{v}_{n+\mu}^{[\mu]}$$

and we can choose $P_Y(v_1, ..., v_{n+\mu}) = (v_{n+\mu-m}, ..., v_{n+\mu})$.

Example 3.1. The following is taken from [3, page 332].

Consider the function $f : \mathbb{R}^2 \to \mathbb{R}$ given by $f(x_1, x_2) = \exp(x_1) \cdot \sin(x_1 + x_2)$.

Choose $H = \mathbb{R}^6$ and $f = P_Y \circ \Phi_4 \circ \Phi_3 \circ \Phi_2 \circ \Phi_1 \circ P_X$ with

$$P_X : \mathbb{R}^2 \to \mathbb{R}^6, \quad \text{with} \quad P_X(x_1, x_2) = (x_1, x_2, 0, 0, 0, 0),$$

$$\Phi_1 : \mathbb{R}^6 \to \mathbb{R}^6, \quad \text{with} \quad \Phi_1(v_1, v_2, v_3, v_4, v_5, v_6) = (v_1, v_2, \exp(v_1), v_4, v_5, v_6),$$

$$\Phi_2 : \mathbb{R}^6 \to \mathbb{R}^6, \quad \text{with} \quad \Phi_2(v_1, v_2, v_3, v_4, v_5, v_6) = (v_1, v_2, v_3, v_1 + v_2, v_5, v_6),$$

$$\Phi_3 : \mathbb{R}^6 \to \mathbb{R}^6, \quad \text{with} \quad \Phi_3(v_1, v_2, v_3, v_4, v_5, v_6) = (v_1, v_2, v_3, v_4, \sin(v_4), v_6),$$

$$\Phi_4 : \mathbb{R}^6 \to \mathbb{R}^6, \quad \text{with} \quad \Phi_4(v_1, v_2, v_3, v_4, v_5, v_6) = (v_1, v_2, v_3, v_4, v_5, v_3 \cdot v_5),$$

$$P_Y : \mathbb{R}^6 \to \mathbb{R}, \quad \text{with} \quad P_Y(v_1, v_2, v_3, v_4, v_5, v_6) = v_6.$$

Analogously to the evaluation of $f(\mathbf{x})$, the evaluation of the matrix-vector product (3.1) for some $\mathbf{c} \in \mathbb{R}^n$ and some $\overrightarrow{\mathbf{x}} = (x'_1, ..., x'_n) \in \mathbb{R}^n$ can be expressed as an evaluation trace $\mathbf{v}'^{[0]} = \mathbf{v}'^{[0]}(\mathbf{c}, \overrightarrow{\mathbf{x}}), ..., \mathbf{v}'^{[\mu]} = \mathbf{v}'^{[\mu]}(\mathbf{c}, \overrightarrow{\mathbf{x}})$, where

$$\mathbf{v}'^{[0]} := P_X \cdot \overrightarrow{\mathbf{x}} = (x'_1, ..., x'_m, 0, ..., 0) \quad \text{and} \quad \mathbf{v}'^{[i]} := \Phi'_{i,\mathbf{c}} \cdot \mathbf{v}'^{[i-1]}, \quad i = 1, ..., \mu.$$

By the nature of the elementary transformations Φ_i, each Jacobian $\Phi'_{i,\mathbf{c}} := J_{\Phi_i}(\mathbf{v}^{[i-1]}(\mathbf{c}))$ will be of the form

$$\Phi'_{i,\mathbf{c}} = \begin{pmatrix} 1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 1 & 0 & \cdots & 0 \\ \frac{\partial \varphi_i}{\partial v_1}(\cdots) & \cdots & \cdots & \cdots & \cdots & \frac{\partial \varphi_i}{\partial v_{n+\mu}}(\cdots) \\ 0 & \cdots & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 1 \end{pmatrix} \leftarrow (n+i)\text{-th row}, \tag{3.2}$$

where $\frac{\partial \varphi_i}{\partial v_k}(\cdots) = \frac{\partial \varphi_i}{\partial v_k}(v_{i_1}(\mathbf{c}), ..., v_{i_{n_i}}(\mathbf{c}))$ is interpreted as 0 if φ_i does not depend on v_k.

Thus, each $\mathbf{v}'^{[i]}$ will be of the form

$$\mathbf{v}'^{[i]} = \begin{pmatrix} \mathbf{v}'^{[i-1]}_1 = x'_1 \\ \vdots \\ \mathbf{v}'^{[i-1]}_n = x'_n \\ \vdots \\ \mathbf{v}'^{[i-1]}_{n+i-1} \\ \nabla \varphi_i(v_{i_1}, ..., v_{i_{n_i}}) \cdot \begin{pmatrix} v'_{i_1} \\ \vdots \\ v'_{i_{n_i}} \end{pmatrix} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

for $v'_{i_1} = v'_{i_1}(\mathbf{c}, \overrightarrow{\mathbf{x}}), ..., v'_{i_{n_i}} = v'_{i_{n_i}}(\mathbf{c}, \overrightarrow{\mathbf{x}}) \in \{\mathbf{v'}_1^{[i-1]}, ..., \mathbf{v'}_{n+i-1}^{[i-1]}\}$, where the $v'_{i_1}, ..., v'_{i_{n_i}}$ correspond exactly to the $v_{i_1}, ..., v_{i_{n_i}}$. That is, if $v_{i_j} = \mathbf{v}_l^{[i-1]}(\mathbf{c})$, then $v'_{i_j} = \mathbf{v'}_l^{[i-1]}(\mathbf{c}, \overrightarrow{\mathbf{x}})$.

The directional derivative of f at $\mathbf{c}$ in direction of $\overrightarrow{\mathbf{x}}$ is then

$$J_f(\mathbf{c}) \cdot \overrightarrow{\mathbf{x}} = P_Y \cdot \mathbf{v'}^{[\mu]}.$$

Example 3.2. The computation of $J_f(c_1, c_2) \cdot \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix}$ for $f : \mathbb{R}^2 \to \mathbb{R}$ given by

$$f(x_1, x_2) = \exp(x_1) \cdot \sin(x_1 + x_2)$$

has four *evaluation trace pairs* $[\mathbf{v}^{[0]}, \mathbf{v'}^{[0]}], ..., [\mathbf{v}^{[4]}, \mathbf{v'}^{[4]}]$, where

$$\mathbf{v}^{[0]} = \begin{pmatrix} c_1 \\ c_2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{v'}^{[0]} = \begin{pmatrix} x'_1 \\ x'_2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

and

$$\mathbf{v}^{[4]} = \begin{pmatrix} c_1 \\ c_2 \\ \exp(c_1) \\ c_1 + c_2 \\ \sin(c_1 + c_2) \\ \exp(c_1) \cdot \sin(c_1 + c_2) \end{pmatrix},$$

$$\overline{\mathbf{v'}}^{[4]} = \begin{pmatrix} x'_1 \\ x'_2 \\ \exp(c_1)x'_1 \\ x'_1 + x'_2 \\ \cos(c_1 + c_2)(x'_1 + x'_2) \\ \sin(c_1 + c_2)\exp(c_1)x'_1 + \exp(c_1)\cos(c_1 + c_2)(x'_1 + x'_2) \end{pmatrix}.$$

Then

$$\nabla J_f((c_1, c_2)) \cdot \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix} = P_Y \cdot \mathbf{v'}^{[4]}$$
$$= \big(\exp(c_1)\sin(c_1 + c_2) + \exp(c_1)\cos(c_1 + c_2)\big)x'_1 + \exp(c_1)\cos(c_1 + c_2)x'_2.$$

Note that in the evaluation process, given the Φ_i, each pair $[\mathbf{v}^{[i]}, \mathbf{v'}^{[i]}]$ depends only on the previous pair $[\mathbf{v}^{[i-1]}, \mathbf{v'}^{[i-1]}]$ and the given vectors $\mathbf{c}, \overrightarrow{\mathbf{x}}$. (Since $\mathbf{v}^{[i]} = \Phi_i(\mathbf{v}^{[i-1]})$ and $\mathbf{v'}^{[i]} = J_{\Phi_i}(\mathbf{v}^{[i-1]}(\mathbf{c})) \cdot \mathbf{v'}^{[i-1]}$.) Therefore, in an implementation, one can actually overwrite $[\mathbf{v}^{[i-1]}, \mathbf{v'}^{[i-1]}]$ by $[\mathbf{v}^{[i]}, \mathbf{v'}^{[i]}]$ in each step.

Note further that the $(n+i)$-th entry in each pair $[\mathbf{v}^{[i]}, \mathbf{v'}^{[i]}]$ is of the form

$$\left(\varphi_i(v_{i_1}, ..., v_{i_{n_i}}), \nabla\varphi_i(v_{i_1}, ..., v_{i_{n_i}}) \cdot \begin{pmatrix} v'_{i_1} \\ \vdots \\ v'_{i_{n_i}} \end{pmatrix}\right) \in \mathbb{R}^2, \tag{3.3}$$

i.e. consisting of a value of φ_i and a directional derivative of this elementary function. Since the previous $n+i-1$ entries are identical to the first $n+i-1$ entries of $[\mathbf{v}^{[i-1]}, \mathbf{v}'^{[i-1]}]$, the computation of $[\mathbf{v}^{[i]}, \mathbf{v}'^{[i]}]$ is effectively the computation of (3.3).

We summarize the discussion of this section in the following theorem:

Theorem 3.3. *By the above, the evaluation of* $J_f(\mathbf{c}) \cdot \vec{\mathbf{x}}$ *can be achieved by the process of computing the evaluation trace pairs* $[\mathbf{v}^{[i]}, \mathbf{v}'^{[i]}]$. *This process is equivalent to the process of computing the pairs* (3.3).

The following section is concerned with a method which uses this last fact directly from the start. This approach also provides a better understanding on how an Automatic Differentiation system could actually be implemented. A question which may not be quite clear from the discussion so far.

4 Forward AD—An approach using Dual Numbers

Many descriptions and implementation of Forward AD actually use a slightly different approach than the elementary one that we have just described. Instead of expressing the function whose derivative one wants to compute as a composition, the main idea in this 'alternative' approach[5] is to lift this function (and all elementary functions) to (a subset of) the algebra of *dual numbers* $\mathcal{D}$. This approach has, for example, been described in [5], [7] and [10].

Dual numbers, introduced by Clifford [1], are defined as $\mathcal{D} := (\mathbb{R}^2, +, \cdot)$, where addition is defined component wise as usual and multiplication is defined as

$$(x_1, y_1) \cdot (x_2, y_2) := (x_1 x_2, x_1 y_2 + y_1 x_2), \quad \forall\ (x_1, y_1), (x_2, y_2) \in \mathbb{R}^2.$$

It is easy to verify that $\mathcal{D}$ with these operations is an associative and commutative algebra over $\mathbb{R}$ with multiplicative unit $(1, 0)$ and that the element $\varepsilon := (0, 1)$ is nilpotent of order two.

Analogously to a complex number, we write a dual number $z = (x, y)$ as $z = x + y\varepsilon$, where we identify each $x \in \mathbb{R}$ with $(x, 0)$. We will further use the notation (x, x') instead of (x, y), i.e. we write $z = x + x'\varepsilon$. The x' in this representation is referred to as the *dual part* of z.

We now define an extension of a differentiable function $g : U \to \mathbb{R}$, on an open set $U \subset \mathbb{R}^\mu$, to a function $\widehat{g} : \mathcal{D}^\mu \supset U \times \mathbb{R}^\mu \to \mathcal{D}$ on a subset of the dual numbers[6], by setting

$$\widehat{g}(x_1 + x_1'\varepsilon, ..., x_\mu + x_\mu'\varepsilon) := g(x_1, ..., x_\mu) + \left(\nabla g(x_1, ..., x_\mu) \cdot \begin{pmatrix} x_1' \\ \vdots \\ x_\mu' \end{pmatrix} \right) \cdot \varepsilon. \tag{4.1}$$

[5] Indeed, we will see at the end of this section, that Forward AD using dual numbers is completely equivalent to the method of expressing f as $P_Y \circ \Phi_\mu \circ \cdots \circ \Phi_1 \circ P_X$.

[6] Here, $\mu \in \mathbb{N}^+$ is just some positive integer and does not have the same meaning as in the previous section.

This definition easily extends to a differentiable function $f : X \to \mathbb{R}^m$, on open $X \subset \mathbb{R}^n$, as $\widehat{f} : \mathcal{D}^n \supset X \times \mathbb{R}^n \to \mathcal{D}^m$ with

$$\widehat{f}(x_1 + x_1'\varepsilon, ..., x_n + x_n'\varepsilon) := \begin{pmatrix} \widehat{f^{[1]}}(x_1 + x_1'\varepsilon, ..., x_n + x_n'\varepsilon) \\ \vdots \\ \widehat{f^{[m]}}(x_1 + x_1'\varepsilon, ..., x_n + x_n'\varepsilon) \end{pmatrix} \tag{4.2}$$

$$= f(x_1, ..., x_n) + J_f(x_1, ..., x_n) \cdot \begin{pmatrix} x_1' \\ \vdots \\ x_\mu' \end{pmatrix} \varepsilon.$$

We first have to show that definition (4.1) makes sense. I.e., that it is compatible with the natural extension of functions on U which are defined via usual arithmetics. That is, we show the following:

Proposition 4.1. *Definition (4.1) is compatible with the natural extension of*

(i) the identity function,

(ii) constant functions,

(iii) the mappings sum_k, $prod_k : \mathbb{R}^k \to \mathbb{R}$, defined by

$$\mathrm{sum}_k(x_1, ..., x_k) := x_1 + \cdots + x_k \quad \textit{and} \quad \mathrm{prod}_k(x_1, ..., x_k) := x_1 \cdots x_k,$$

(iv) (multivariate) polynomials

to the dual numbers $\mathcal{D}$.

Proof. (i) and (ii) follow directly from the definition.
(iii): We have

$$\begin{aligned}
&\widehat{\mathrm{sum}}_k(x_1 + x_1'\varepsilon, ..., x_k + x_k'\varepsilon) \\
&= x_1 + x_1'\varepsilon + \cdots + x_k + x_k'\varepsilon \\
&= (x_1 + \cdots + x_k) + (x_1' + \cdots + x_k')\varepsilon \\
&= (x_1 + \cdots + x_k) + \left(\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}^T \cdot \begin{pmatrix} x_1' \\ \vdots \\ x_k' \end{pmatrix} \right) \varepsilon \\
&= \mathrm{sum}_k(x_1, ..., x_k) + \left(\nabla \mathrm{sum}_k(x_1, ..., x_k) \cdot \begin{pmatrix} x_1' \\ \vdots \\ x_k' \end{pmatrix} \right) \varepsilon
\end{aligned}$$

and, since $\varepsilon^2 = 0$,

$$\begin{aligned}
&\widehat{\mathrm{prod}}_k(x_1 + x_1'\varepsilon, ..., x_k + x_k'\varepsilon) \\
&= (x_1 + x_1'\varepsilon) \cdot \cdots \cdot (x_k + x_k'\varepsilon) \\
&= (x_1 \cdot \cdots \cdot x_k) + \left(\sum_{i=1}^{k} x_i' \prod_{\substack{j=1 \\ j\neq i}}^{k} x_j \right) \varepsilon \\
&= (x_1 \cdot \cdots \cdot x_k) + \left(\begin{pmatrix} x_2 x_3 \cdots x_k \\ x_1 x_3 x_4 \cdots x_k \\ \vdots \\ x_1 x_2 \cdots x_{k-1} \end{pmatrix}^T \cdot \begin{pmatrix} x_1' \\ \vdots \\ x_k' \end{pmatrix} \right) \varepsilon \\
&= \mathrm{prod}_k(x_1, ..., x_k) + \left(\nabla \mathrm{prod}_k(x_1, ..., x_k) \cdot \begin{pmatrix} x_1' \\ \vdots \\ x_k' \end{pmatrix} \right) \varepsilon
\end{aligned}$$

(iv): This follows from (i)-(iii). □

We further need to prove that definition (4.1) behaves well for functions which depend on other functions. That is, we show the following:

Proposition 4.2. *Let $g : U \to \mathbb{R}$ be differentiable on the open set $U \subset \mathbb{R}^\mu$, such that*

$$g(x_1, ..., x_\mu) = g_\ell(x_{0,1}, ..., x_{0,\mu_0}, g_1(x_{1,1}, ..., x_{1,\mu_1}), ..., g_{\ell-1}(x_{\ell-1,1}, ..., x_{\ell-1,\mu_{\ell-1}})),$$

where $g_i : U_i \to \mathbb{R}$ are differentiable on the open sets $U_i \subset \mathbb{R}^{\mu_i}$ with $x_{i,j} \in \{x_1, ..., x_\mu\}$. Then

$$\begin{aligned}
&\widehat{g}(x_1 + \varepsilon x_1', ..., x_\mu + \varepsilon x_\mu') \\
&= \widehat{g}_\ell(x_{0,1} + \varepsilon x_{0,1}', ..., x_{0,\mu_0} + \varepsilon x_{0,\mu}', \widehat{g}_1(x_{1,1} + \varepsilon x_{1,1}', ..., x_{1,\mu_1} + \varepsilon x_{1,\mu_1}'), ..., \\
&\qquad \widehat{g_{\ell-1}}(x_{\ell-1,1} + \varepsilon x_{\ell-1,1}', ..., x_{\ell-1,\mu_{\ell-1}} + \varepsilon x_{\ell-1,\mu_{\ell-1}}')).
\end{aligned} \tag{4.3}$$

Proof. The right hand-side of equation (4.3) is equal to

$$\begin{aligned}
&\widehat{g}_\ell \Bigg(x_{0,1} + \varepsilon x_{0,1}', ..., x_{0,\mu_0} + \varepsilon x_{0,\mu}', \\
&\qquad g_1(x_{1,1}, ..., x_{1,\mu_1}) + \left(\nabla g_1(\cdots) \cdot \begin{pmatrix} x_{1,1}' \\ \vdots \\ x_{1,\mu}' \end{pmatrix} \right) \varepsilon, ..., \\
&\qquad g_{\ell-1}(x_{\ell-1,1}, ..., x_{\ell-1,\mu_{\ell-1}}) + \left(\nabla g_{\ell-1}(\cdots) \cdot \begin{pmatrix} x_{\ell-1,1}' \\ \vdots \\ x_{\ell-1,\mu_{\ell-1}}' \end{pmatrix} \right) \varepsilon \Bigg)
\end{aligned}$$

$$= g_\ell\left(x_{0,1},...,x_{0,\mu_0},g_1(x_{1,1},...,x_{1,\mu_1}),...,g_{\ell-1}(x_{\ell-1,1},...,x_{\ell-1,\mu_{\ell-1}})\right)$$
$$+\left(\nabla g_\ell(\cdots)\cdot\begin{pmatrix} x'_{0,1}\\ \vdots\\ x'_{0,\mu}\\ \nabla g_1(\cdots)\cdot\begin{pmatrix} x'_{1,1}\\ \vdots\\ x'_{1,\mu}\end{pmatrix}\\ \vdots\\ \nabla g_{\ell-1}(\cdots)\cdot\begin{pmatrix} x'_{\ell-1,1}\\ \vdots\\ x'_{\ell-1,\mu_{\ell-1}}\end{pmatrix}\end{pmatrix}\right)\varepsilon, \tag{4.4}$$

where $\nabla g_\ell(\cdots)=\frac{dg_\ell}{d\left(x_{0,1},...,x_{0,\mu_0},g_1(\cdots),...,g_{\ell-1}(\cdots)\right)}(\cdots)$.

The left hand side of (4.3) is obviously equal to

$$g(x_1,...,x_\mu)+\left(\nabla g(x_1,...,x_\mu)\cdot\begin{pmatrix} x'_1\\ \vdots\\ x'_\mu\end{pmatrix}\right)\cdot\varepsilon. \tag{4.5}$$

By assumption,

$$g(x_1,...,x_\mu)=g_\ell(x_{0,1},...,x_{0,\mu_0},g_1(x_{1,1},...,x_{1,\mu_1}),...,g_{\ell-1}(x_{\ell-1,1},...,x_{\ell-1,\mu_{\ell-1}})).$$

Further,

$$\begin{aligned}&\nabla g(x_1,...,x_\mu)\\ &=\frac{d}{d(x_1,...,x_\mu)}g_\ell(x_{0,1},...,x_{0,\mu_0},g_1(x_{1,1},...,x_{1,\mu_1}),...,g_{\ell-1}(x_{\ell-1,1},...,x_{\ell-1,\mu_{\ell-1}}))\\ &=\nabla g_\ell(x_{0,1},...,x_{0,\mu_0},g_1(x_{1,1},...,x_{1,\mu_1}),...,g_{\ell-1}(x_{\ell-1,1},...,x_{\ell-1,\mu_{\ell-1}}))\\ &\quad\cdot\frac{d}{d(x_1,...,x_\mu)}\left((x_1,...,x_\mu)\mapsto\left(x_{0,1},...,x_{0,\mu_0},g_1(\cdots),...,g_{\ell-1}(\cdots)\right)\right),\end{aligned}$$

where $\nabla g_\ell(\cdots)=\frac{dg_\ell}{d\left(x_{0,1},...,x_{0,\mu_0},g_1(\cdots),...,g_{\ell-1}(\cdots)\right)}(\cdots)$.

Now, clearly

$$\frac{d}{d(x_1,...,x_\mu)}\left((x_1,...,x_\mu)\mapsto\left(x_{0,1},...,x_{0,\mu_0},g_1(\cdots),...,g_{\ell-1}(\cdots)\right)\right)\cdot\begin{pmatrix} x'_1\\ \vdots\\ x'_\mu\end{pmatrix}$$

equals

$$\begin{pmatrix} \frac{\partial x_{0,1}}{dx_1} & \dots & \frac{\partial x_{0,1}}{dx_\mu} \\ \frac{\partial x_{0,\mu_0}}{dx_1} & \dots & \frac{\partial x_{0,\mu_0}}{dx_\mu} \\ \frac{\partial g_1(\cdots)}{dx_1} & \dots & \frac{\partial g_1(\cdots)}{dx_\mu} \\ \vdots & \vdots & \vdots \\ \frac{\partial g_{\ell-1}(\cdots)}{dx_1} & \dots & \frac{\partial g_{\ell-1}(\cdots)}{dx_\mu} \end{pmatrix} \begin{pmatrix} x'_1 \\ \vdots \\ x'_\mu \end{pmatrix} = \begin{pmatrix} x'_{0,1} \\ \vdots \\ x'_{0,\mu} \\ \nabla g_1(\cdots) \begin{pmatrix} x'_{1,1} \\ \vdots \\ x'_{1,\mu} \end{pmatrix} \\ \vdots \\ \nabla g_{\ell-1}(\cdots) \begin{pmatrix} x'_{\ell-1,1} \\ \vdots \\ x'_{\ell-1,\mu_{\ell-1}} \end{pmatrix} \end{pmatrix}.$$

Hence, (4.4) equals (4.5) and we are done. □

One can now determine the directional derivative $J_f(\mathbf{c}) \cdot \vec{\mathbf{x}}$ of a differentiable $f : \mathbb{R}^m \supset X \to \mathbb{R}^m$, X open, at a fixed $\mathbf{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} \in \mathbb{R}^n$ in direction of a fixed $\vec{\mathbf{x}} = \begin{pmatrix} x'_1 \\ \vdots \\ x'_n \end{pmatrix} \in \mathbb{R}^n$, by extending each $f^{[j]}$ to the dual numbers and. That is, one replaces the arguments $(x_1, ..., x_n)$ in $f^{[j]}$ by $(c_1 + x'_1\varepsilon, ..., c_n + x'_n\varepsilon)$ and applies first (4.1) to the elementary functions $\varphi^{[j]}_{1,1}, ..., \varphi^{[j]}_{1,k_{j,1}}$. This then leads to the extension of the $\varphi^{[j]}_{2,1}, ..., \varphi^{[j]}_{2,k_{j,2}}$ to the dual numbers and one can apply (4.1) again and so on.

We obtain therefore for $i = 1, ..., k_{j,\nu}$, $\nu = 1, ..., \ell_j - 1$:

$$\widehat{\varphi^{[j]}_{1,i}} \left(u^{[j]}_{1,i,1} + \left(u^{[j]}_{1,i,1}\right)' \varepsilon, ..., u^{[j]}_{1,i,l_{1,i}} + \left(u^{[j]}_{1,i,l_{1,i}}\right)' \varepsilon \right)$$

$$= \varphi^{[j]}_{1,i}(u^{[j]}_{1,i,1}, ..., u^{[j]}_{1,i,l_{1,i}}) + \left(\nabla \varphi^{[j]}_{1,i}(u^{[j]}_{1,i,1}, ..., u^{[j]}_{1,i,l_{1,i}}) \cdot \begin{pmatrix} \left(u^{[j]}_{1,i,1}\right)' \\ \vdots \\ \left(u^{[j]}_{1,i,l_{1,i}}\right)' \end{pmatrix} \right) \varepsilon,$$

with

$$u^{[j]}_{1,i,1} + \left(u^{[j]}_{1,i,1}\right)' \varepsilon, ..., u^{[j]}_{1,i,l_{1,i}} + \left(u^{[j]}_{1,i,l_{1,i}}\right)' \varepsilon \ \in \{c_1 + x'_1\varepsilon, ..., c_n + x'_n\varepsilon\}.$$

Then,

$$\widehat{\varphi^{[j]}_{2,i}} \left(u^{[j]}_{2,i,1} + \left(u^{[j]}_{2,i,1}\right)' \varepsilon, ..., u^{[j]}_{2,i,l_{2,i}} + \left(u^{[j]}_{2,i,l_{2,i}}\right)' \varepsilon \right)$$

$$= \varphi_{2,i}^{[j]}(u_{2,i,1}^{[j]}, ..., u_{2,i,l_{2,i}}^{[j]}) + \left(\nabla \varphi_{2,i}^{[j]}(u_{2,i,1}^{[j]}, ..., u_{2,i,l_{2,i}}^{[j]}) \cdot \begin{pmatrix} \left(u_{2,i,1}^{[j]}\right)' \\ \vdots \\ \left(u_{2,i,l_{2,i}}^{[j]}\right)' \end{pmatrix} \right) \varepsilon,$$

with

$$u_{2,i,1}^{[j]} + \left(u_{2,i,1}^{[j]}\right)' \varepsilon, ..., u_{2,i,l_{2,i}}^{[j]} + \left(u_{2,i,l_{2,i}}^{[j]}\right)' \varepsilon$$
$$\in \left\{ c_1 + x_1' \varepsilon, ..., c_n + x_n' \varepsilon, \ \widehat{\varphi_{1,1}^{[j]}}(\cdots), ..., \widehat{\varphi_{1,k_{j,1}}^{[j]}}(\cdots) \right\}$$

$$\vdots$$

$$\widehat{\varphi_{\ell_j-1,i}^{[j]}} \left(u_{\ell_j-1,i,1}^{[j]} + \left(u_{\ell_j-1,i,1}^{[j]}\right)' \varepsilon, ..., u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]} + \left(u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]}\right)' \varepsilon \right)$$
$$= \varphi_{\ell_j-1,i}^{[j]} \left(u_{\ell_j-1,i,1}^{[j]}, ..., u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]} \right)$$
$$+ \left(\nabla \varphi_{\ell_j-1,i}^{[j]} \left(u_{\ell_j-1,i,1}^{[j]}, ..., u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]} \right) \cdot \begin{pmatrix} \left(u_{\ell_j-1,i,1}^{[j]}\right)' \\ \vdots \\ \left(u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]}\right)' \end{pmatrix} \right) \varepsilon,$$

with

$$u_{\ell_j-1,i,1}^{[j]} + \left(u_{\ell_j-1,i,1}^{[j]}\right)' \varepsilon, ..., u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]} + \left(u_{\ell_j-1,i,l_{\ell_j-1,i}}^{[j]}\right)' \varepsilon$$
$$\in \Big\{ c_1 + x_1' \varepsilon, ..., c_n + x_n' \varepsilon, \ \widehat{\varphi_{1,1}^{[j]}}(\cdots), ..., \widehat{\varphi_{1,k_{j,1}}^{[j]}}(\cdots), \widehat{\varphi_{2,1}^{[j]}}(\cdots), ..., \widehat{\varphi_{2,k_{j,2}}^{[j]}}(\cdots), ...$$
$$..., \widehat{\varphi_{\ell_j-2,1}^{[j]}}(\cdots), ..., \widehat{\varphi_{\ell_j-2,k_{j,\ell_j-2}}^{[j]}}(\cdots) \Big\}.$$

Finally,

$$\widehat{\varphi_{\ell_j}^{[j]}} \left(u_{\ell_j,1}^{[j]} + \left(u_{\ell_j,1}^{[j]}\right)' \varepsilon, ..., u_{\ell_j,l_{\ell_j}}^{[j]} + \left(u_{\ell_j,l_{\ell_j}}^{[j]}\right)' \varepsilon \right)$$
$$= \varphi_{\ell_j}^{[j]} \left(u_{\ell_j,1}^{[j]}, ..., u_{\ell_j,l_{\ell_j}}^{[j]} \right) + \left(\nabla \varphi_{\ell_j}^{[j]} \left(u_{\ell_j,1}^{[j]}, ..., u_{\ell_j,l_{\ell_j}}^{[j]} \right) \cdot \begin{pmatrix} \left(u_{\ell_j,1}^{[j]}\right)' \\ \vdots \\ \left(u_{\ell_j,l_{\ell_j}}^{[j]}\right)' \end{pmatrix} \right) \varepsilon, \tag{4.6}$$

with

$$u_{\ell_j,1}^{[j]} + \left(u_{\ell_j,1}^{[j]}\right)' \varepsilon, ..., u_{\ell_j,l_{\ell_j}}^{[j]} + \left(u_{\ell_j,l_{\ell_j}}^{[j]}\right)' \varepsilon$$

$$\in \left\{ c_1 + x_1'\varepsilon, ..., c_n + x_n'\varepsilon,\ \widehat{\varphi_{1,1}^{[j]}}(\cdots), ..., \widehat{\varphi_{1,k_{j,1}}^{[j]}}(\cdots), \widehat{\varphi_{2,1}^{[j]}}(\cdots), ..., \widehat{\varphi_{2,k_{j,2}}^{[j]}}(\cdots), ... \right.$$
$$\left. ..., \widehat{\varphi_{\ell_j-1,1}^{[j]}}(\cdots), ..., \widehat{\varphi_{\ell_j-1,k_{j,\ell_j-1}}^{[j]}}(\cdots) \right\}.$$

Now, we have the following:

Theorem 4.3. *The lifting process of $f^{[j]}$ to a subset of the dual numbers, as described above, computes the directional derivative $\nabla f^{[j]}(\mathbf{c}) \cdot \vec{\mathbf{x}}$ of $f^{[j]}$ at fixed $\mathbf{c} \in \mathbb{R}^n$ in direction of fixed $\vec{\mathbf{x}} \in \mathbb{R}^n$ as the dual part of $\widehat{\varphi_{\ell_j}^{[j]}}(\cdots)$.*

Proof. By applying proposition 4.2 successively, we obtain

$$\widehat{\varphi_{\ell_j}^{[j]}}\left(u_{\ell_j,1}^{[j]} + \left(u_{\ell_j,1}^{[j]}\right)'\varepsilon, ..., u_{\ell_j,l_{\ell_j}}^{[j]} + \left(u_{\ell_j,l_{\ell_j}}^{[j]}\right)'\varepsilon\right) = f^{[j]}(c_1 + x_1'\varepsilon, ..., c_n + x_n'\varepsilon).$$

Since $\varphi_{\ell_j}^{[j]}\left(u_{\ell_j,1}^{[j]}, ..., u_{\ell_j,l_{\ell_j}}^{[j]}\right) = f^{[j]}(c_1, ..., c_n)$ by construction, (4.6) implies

$$\nabla\varphi_{\ell_j}^{[j]}\left(u_{\ell_j,1}^{[j]}, ..., u_{\ell_j,l_{\ell_j}}^{[j]}\right) \cdot \begin{pmatrix} \left(u_{\ell_j,1}^{[j]}\right)' \\ \vdots \\ \left(u_{\ell_j,l_{\ell_j}}^{[j]}\right)' \end{pmatrix} = \nabla f^{[j]}(c_1, ..., c_n) \cdot \begin{pmatrix} x_1' \\ \vdots \\ x_n' \end{pmatrix}.$$

□

Example 4.4. Consider again the function $f : \mathbb{R}^2 \to \mathbb{R}^2$ given by

$$f(x_1, x_2) = \begin{pmatrix} \sin(x_2) + \cos(x_1^2 + 3) \cdot 5x_2 \\ \exp(\sin(x_1)) + 4x_2^3 \end{pmatrix}.$$

We replace x_1, x_2 in $f^{[1]}$ and $f^{[2]}$ by $c_1 + x_1'\varepsilon$ and $c_2 + x_2'\varepsilon$.

We have, by definition (4.1), proposition 4.1 and proposition 4.2,

$$\begin{aligned}
&\widehat{f}^{[1]}(c_1 + x_1'\varepsilon, c_2 + x_2'\varepsilon) \\
&= \sin(c_2 + x_2'\varepsilon) + 5(c_2 + x_2'\varepsilon) \cdot \cos((c_1 + x_1'\varepsilon)^2 + 3) \\
&= \sin(c_2) + \cos(c_2)x_2'\varepsilon + (5c_2 + 5x_2'\varepsilon) \cdot \cos(c_1^2 + 3 + 2c_1x_1'\varepsilon) \\
&= \sin(c_2) + \cos(c_2)x_2'\varepsilon + (5c_2 + 5x_2'\varepsilon) \cdot (\cos(c_1^2 + 3) - \sin(c_1^2 + 3)2c_1x_1'\varepsilon) \\
&= \sin(c_2) + \cos(c_2)x_2'\varepsilon + 5c_2\cos(c_1^2 + 3) - 10c_1c_2\sin(c_1^2 + 3)x_1'\varepsilon + 5\cos(c_1^2 + 3)x_2'\varepsilon \\
&= \sin(c_2) + 5c_2\cos(c_1^2 + 3) + \left(-10c_1c_2\sin(c_1^2 + 3)x_1' + (\cos(c_2) + 5\cos(c_1^2 + 3))x_2'\right)\varepsilon.
\end{aligned}$$

By theorem 4.3, the dual part of this expression is $\nabla f^{[1]}(c_1, c_2) \cdot \begin{pmatrix} x_1' \\ x_2' \end{pmatrix}$. I.e.,

$$\nabla f^{[1]}(c_1, c_2) \cdot \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = -10c_1c_2\sin(c_1^2 + 3)x_1' + (\cos(c_2) + 5\cos(c_1^2 + 3))x_2'.$$

Similarly,

$$\begin{aligned}
&\widehat{f}^{[2]}(c_1 + x_1'\varepsilon, c_2 + x_2'\varepsilon) \\
&= \exp(\sin(c_1 + x_1'\varepsilon)) + 4 \cdot (c_2 + x_2'\varepsilon) \cdot (c_2 + x_2'\varepsilon)^2 \\
&= \exp\left(\sin(c_1) + \cos(c_1)x_1'\varepsilon\right) + 4(c_2^3 + c_2^2 x_2'\varepsilon + 2c_2^2 x_2'\varepsilon) \\
&= \exp(\sin(c_1)) + \exp(\sin(c_1))\cos(c_1)x_1'\varepsilon + 4c_2^3 + (4c_2^2 x_2' + 8c_2^2 x_2')\varepsilon \\
&= \exp(\sin(c_1)) + 4c_2^3 + \left(\exp(\sin(c_1))\cos(c_1)x_1' + 12c_2^2 x_2'\right)\varepsilon.
\end{aligned}$$

Then, by theorem 4.3,

$$\nabla f^{[2]}(c_1, c_2) \cdot \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \exp(\sin(c_1))\cos(c_1)x_1' + 12c_2^2 x_2'.$$

Therefore,

$$J_f(c_1, c_2) \cdot \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} -10c_1c_2\sin(c_1^2+3)x_1' + (\cos(c_2) + 5\cos(c_1^2+3))x_2' \\ \exp(\sin(c_1))\cos(c_1)x_1' + 12c_2^2 x_2' \end{pmatrix}.$$

Thus, a (basic) implementation of an Automatic Differentiation System can be realised by implementing (4.1) for all elementary functions together with the rules of addition and multiplication on dual numbers. The actual computation of a directional derivative of a function f is then performed by simply replacing the arguments of each $f^{[j]}$ by dual numbers $c_1 + x_1'\varepsilon, ..., c_n + x_n'\varepsilon$ and evaluating $\widehat{f}^{[j]}(c_1 + x_1'\varepsilon, ..., c_n + x_n'\varepsilon)$.

Note again that, at no time during this process, any actual differentiation takes place. Instead, we are computing and passing on the pairs

$$\widehat{\varphi_{\nu,k}^{[j]}}(\cdots) = \begin{pmatrix} \varphi_{\nu,k}^{[j]}(\cdots),\ \nabla\varphi_{\nu,k}^{[j]}(\cdots) \cdot \begin{pmatrix} \left(u_{\nu,k,1}^{[j]}\right)' \\ \vdots \\ \left(u_{\nu,k,l_{\nu,k}}^{[j]}\right)' \end{pmatrix} \end{pmatrix} \tag{4.7}$$

$$\text{and} \quad \widehat{\varphi_{\ell_j}^{[j]}}(\cdots) = \begin{pmatrix} \widehat{\varphi_{\ell_j}^{[j]}}(\cdots),\ \nabla\widehat{\varphi_{\ell_j}^{[j]}}(\cdots) \cdot \begin{pmatrix} \left(u_{\ell_j,1}^{[j]}\right)' \\ \vdots \\ \left(u_{\ell_j,l_{\ell_j}}^{[j]}\right)' \end{pmatrix} \end{pmatrix}, \tag{4.8}$$

which computes 'automatically' the directional derivatives $\nabla f^{[j]}(\mathbf{c}) \cdot \vec{\mathbf{x}}$ and, therefore, the directional derivative $J_f(\mathbf{c}) \cdot \vec{\mathbf{x}}$..

Note further that the pairs (4.7), (4.8) are exactly the same pairs (just with different indices) as the ones in (3.3). This means that the processes described in this and in the previous section reduce to exactly the same arithmetic operations.

Indeed, if we impose a suitable ordering on the $\varphi_{\nu,k}^{[j]}, \varphi_{\ell_j}^{[j]}$, let φ_i the i-th function with respect to this ordering, and store the pairs (c_i, x_i') and $\begin{pmatrix} \varphi_i(\cdots), \nabla\varphi_i(\cdots) \begin{pmatrix} u_{i,1}' \\ \vdots \\ u_{i,l_i}' \end{pmatrix} \end{pmatrix}$ in an array, we obtain the evaluation trace pairs $[\mathbf{v}^{[i]}, \mathbf{v}'^{[i]}]$. In summary:

Theorem 4.5. *The evaluation of the directional derivative $J_f(\mathbf{c}) \cdot \vec{\mathbf{x}}$ through the lifting of each $f^{[j]}$ to a function defined on a subset of $\mathcal{D}$, is equivalent to the process of computing the evaluation trace pairs $[\mathbf{v}^{[i]}, \mathbf{v}'^{[i]}]$, as described in the previous section. This process is equivalent to the computation of the pairs (4.7), (4.8) in suitable order.*

5 Forward AD and Taylor Series expansion

In the literature (see, for example, [7]), definition (4.1) is often described as being obtained by evaluating the Taylor Series Expansion of $\widehat{g}$ about $(x_1 + x'_1\varepsilon, ..., x_\mu + x'_\mu\varepsilon)$.

To understand this argument, recall that the Taylor series of an infinitely many times differentiable multivariate function $g : U \to \mathbb{R}$ on an open set $U \subset \mathbb{R}^\mu$ about some point $\mathbf{c} = (c_1, ..., c_\mu) \in U$ is given by

$$T(g; \mathbf{c})(\mathbf{x}) = \sum_{n_1+\cdots+n_\mu=0}^{\infty} \frac{(x_1 - c_1)^{n_1} \cdots (x_\mu - c_\mu)^{n_\mu}}{n_1! \cdots n_\mu!} \frac{\partial^{n_1+\cdots+n_\mu} g}{\partial x_1^{n_1} \cdots \partial x_\mu^{n_\mu}}(\mathbf{c}),$$

for all $\mathbf{x} = (x_1, ..., x_\mu) \in U$.

Let now $\tilde{g} : \mathcal{D}^\mu \supset U \times \mathbb{R}^\mu \to \mathcal{D}$ be an extension of g to the dual numbers (that is $\tilde{g}|_U = g$). We define the Taylor series of $\tilde{g}$ about some vector of dual numbers $(\mathbf{c}, \vec{\mathbf{c}}) := (c_1 + c'_1\varepsilon, ..., c_\mu + c'_\mu\varepsilon) \in U \times \mathbb{R}^\mu$ analogously to the real case. That is,

$$\begin{aligned} &T(\tilde{g}; (\mathbf{c}, \vec{\mathbf{c}}))((\mathbf{x}, \vec{\mathbf{x}})) \\ &= \sum_{n_1+\cdots+n_\mu=0}^{\infty} \left(\frac{(x_1 - c_1 + (x'_1 - c'_1)\varepsilon)^{n_1} \cdots (x_\mu - c_\mu + (x'_\mu - c'_\mu)\varepsilon)^{n_\mu}}{n_1! \cdots n_\mu!} \right. \\ &\qquad \left. \cdot \frac{\partial^{n_1+\cdots+n_\mu} \tilde{g}}{\partial x_1^{n_1} \cdots \partial x_\mu^{n_\mu}}((\mathbf{c}, \vec{\mathbf{c}})) \right), \end{aligned}$$

for all $(\mathbf{x}, \vec{\mathbf{x}}) := (x_1 + x'_1\varepsilon, ..., x_\mu + x'_\mu\varepsilon) \in U \times \mathbb{R}^\mu$.

Trivially, this series converges for $(\mathbf{x}, \vec{\mathbf{x}}) = (\mathbf{c}, \vec{\mathbf{c}})$. Further, due to $\varepsilon^2 = 0$, the Taylor series about any $(\mathbf{x}, \mathbf{0}) = (x_1 + 0\varepsilon, ..., x_\mu + 0\varepsilon) \in U \times \mathbb{R}^\mu$ converges for the arguments $(\mathbf{x}, \vec{\mathbf{x}}) = (x_1 + x'_1\varepsilon, ..., x_\mu + x'_\mu\varepsilon)$, for all $\vec{\mathbf{x}} \in \mathbb{R}^\mu$. We have, identifying $\mathbf{x}$ with $(\mathbf{x}, \mathbf{0})$,

$$\begin{aligned} T(\tilde{g}; \mathbf{x})(\mathbf{x}, \vec{\mathbf{x}}) &= \sum_{n_1+\cdots+n_\mu=0}^{\infty} \frac{(x'_1\varepsilon)^{n_1} \cdots (x'_\mu\varepsilon)^{n_\mu}}{n_1! \cdots n_\mu!} \frac{\partial^{n_1+\cdots+n_\mu}}{\partial x_1^{n_1} \cdots \partial x_\mu^{n_\mu}} \tilde{g}(\mathbf{x}) \qquad (5.1) \\ &= \sum_{n_1+\cdots+n_\mu=0}^{1} \frac{(x'_1\varepsilon)^{n_1} \cdots (x'_\mu\varepsilon)^{n_\mu}}{n_1! \cdots n_\mu!} \frac{\partial^{n_1+\cdots+n_\mu}}{\partial x_1^{n_1} \cdots \partial x_\mu^{n_\mu}} \tilde{g}(\mathbf{x}) \\ &= \tilde{g}(\mathbf{x}) + \sum_{k=1}^{\mu} \frac{\partial}{\partial x_k} \tilde{g}(\mathbf{x}) \cdot x'_k\varepsilon \end{aligned}$$

$$= \tilde{g}(x_1, ..., x_\mu) + \left(\nabla \tilde{g}(x_1, ..., x_\mu) \cdot \begin{pmatrix} x'_1 \\ \vdots \\ x'_\mu \end{pmatrix} \right) \varepsilon$$

$$= g(x_1, ..., x_\mu) + \left(\nabla g(x_1, ..., x_\mu) \cdot \begin{pmatrix} x'_1 \\ \vdots \\ x'_\mu \end{pmatrix} \right) \varepsilon.$$

As we see, the right-hand side of the last equation is equal to $\widehat{g}(x_1 + x'_1\varepsilon, ..., x_\mu + x'_\mu\varepsilon)$ in definition (4.1). Hence, if we choose $\tilde{g}$ as $\widehat{g}$, we obtain

$$\widehat{g}(x_1 + x'_1\varepsilon, ..., x_\mu + x'_\mu\varepsilon) = T(\widehat{g}; (x_1, ..., x_\mu))(x_1 + x'_1\varepsilon, ..., x_\mu + x'_\mu\varepsilon). \quad (5.2)$$

That is:

Proposition 5.1. *The extension of a differentiable $g : \mathbb{R}^\mu \supset U \to \mathbb{R}$, U open, to a subset of the dual number as defined in (4.1), is the (unique) function $\widehat{g}$, with the property that the images of any $(x_1 + x'_1\varepsilon, ..., x_\mu + x'_\mu\varepsilon)$ under $\widehat{g}$ and $T(\widehat{g}; (x_1, ..., x_\mu))$ are equal.*

(Note that this does not mean that $\widehat{g}$ is analytic on U.)

Since we identify U with its natural embedding into $\mathcal{D}^\mu$, we can replace $\tilde{g}(\mathbf{x})$ by $g(\mathbf{x})$ in the right-hand side of (5.1). It is custom to do this in the left-hand side of (5.1) as well. That is, one usually writes $T(g; (x_1, ..., x_\mu))$ instead of $T(\tilde{g}; (x_1, ..., x_\mu))$ or $T(\widehat{g}; (x_1, ..., x_\mu))$.

By the above, it is obvious that one can describe the process of determining the directional derivatives $\nabla f^{[j]}(\mathbf{c}) \cdot \vec{\mathbf{x}}$ of each $f^{[j]}$ in terms of Taylor series expansion. I.e., one determines these derivatives by replacing the $(x_1, ..., x_n)$ in the elementary functions $\varphi^{[j]}_{1,1}, ..., \varphi^{[j]}_{1,k_{j,1}}$ by $(c_1 + x'_1\varepsilon, ..., c_n + x'_n\varepsilon)$ and evaluates the Taylor series of these functions or, more precisely, of their extensions to the dual numbers, about the real parts of their arguments. This leads to the extension of the $\varphi^{[j]}_{2,1}, ..., \varphi^{[j]}_{2,k_{j,2}}$ to the dual numbers and one evaluates the Taylor series of these functions and so on.

This process is, by (5.2), obviously identical to the one described on the previous pages.

6 Forward AD, Differential Geometry and Category Theory

In recent literature (see [6]) the extension of differentiable functions $g : U \to \mathbb{R}$ on open $U \subset \mathbb{R}^\mu$ to a function $\widehat{g} : \mathcal{D}^\mu \supset U \times \mathbb{R}^\mu \to \mathcal{D}$ is described in terms of the *push-forward* operator known from Differential Geometry. We shortly summarize the discussion provided in [6].

Let M, N be differentiable manifolds, TM, TN their tangent spaces and let $h : M \to N$ be a linearly approximatable function. The push-forward (or *differential*) $T(h) = dh$ of h can be defined[7] as

$$T(h) : TM \to TN \quad \text{with} \quad T(h)(\mathbf{x}, \vec{\mathbf{x}}) = (h(\mathbf{x}), d_{\mathbf{x}}h(\vec{\mathbf{x}})),$$

[7] The definition we are using here is the same as in [6]. Some authors define $T(h) = dh$ via $dh(\mathbf{x}, \vec{\mathbf{x}}) = d_{\mathbf{x}}h(\vec{\mathbf{x}})$.

where $d_{\mathbf{x}}h(\overrightarrow{\mathbf{x}})$ is the *push-forward (or differential) of h at $\mathbf{x}$* applied to $\overrightarrow{\mathbf{x}}$.

If now $f : X \to \mathbb{R}^m$ on open $X \subset \mathbb{R}^n$ is a differentiable function, this reads

$$T(f) : X \times \mathbb{R}^n \to \mathbb{R}^{2m} \quad \text{with} \quad T(f)(\mathbf{x}, \overrightarrow{\mathbf{x}}) = \left(f(\mathbf{x}),\ J_f(\mathbf{x}) \cdot \overrightarrow{\mathbf{x}}\right).$$

Identifying $\mathbb{R}^{2m}$ with $\mathcal{D}^m$ and in light of (4.2), this means nothing else than

$$T(f) = \widehat{f}.$$

Furthermore, in the special case of a real-valued and infinitely many times differentiable function $g : U \to \mathbb{R}$ on open $U \subset \mathbb{R}^\mu$ we also have, by equation (5.2),

$$T(g)(\mathbf{x}, \overrightarrow{\mathbf{x}}) = T(g; \mathbf{x})(\mathbf{x}, \overrightarrow{\mathbf{x}}), \quad \forall (\mathbf{x}, \overrightarrow{\mathbf{x}}) \in U \times \mathbb{R}^\mu,$$

which justifies using the letter T for both, the push-forward and the Taylor-series of g in this setting.

It is now easy to check that $T(id_M) = id_{TM}$ and that

$$T(h \circ g) = T(h) \circ T(g),$$

for all $g : M \to N$ and $h : N \to L$, for differentiable manifolds M, N, L. Hence, the mapping T given by

$$\begin{aligned} M &\mapsto TM \\ h &\mapsto T(h) \end{aligned}$$

is a functor from the category of differentiable manifolds to itself. Furthermore, since $T\mathbb{R} = \mathbb{R}^2$, one can even consider the ring of dual number $\mathcal{D}$ as the image of $\mathbb{R}$ under T, equipped with the push-forwards of addition and multiplication. That is,

$$(\mathcal{D}, +, \cdot) = (T\mathbb{R}, T(\text{sum}), T(\text{prod}))\,.$$

Extending this to higher dimensions, the lifting of a differentiable function $f : X \to \mathbb{R}^m$ on open $X \subset \mathbb{R}^n$ to a function $\widehat{f}$ on a subset of the dual numbers $X \times \mathbb{R}^n \subset \mathcal{D}^n$ may be considered as the application of the functor T to X, $\mathbb{R}^m$ and f. In other words,

$$\widehat{f} : X \times \mathbb{R}^n \to \mathcal{D}^m \quad = \quad T(f) : TX \to T\mathbb{R}^m \quad = \quad T\left(f : X \to \mathbb{R}^m\right).$$

In summary, the Forward Mode of AD may also be studied from viewpoints of Differential Geometry and Category Theory.

7 The Reverse Mode of AD

Let, as before, $f : X \to \mathbb{R}^m$ on open $X \subset \mathbb{R}^n$ be differentiable. As already mentioned, the Reverse Mode of Automatic Differentiation evaluates products of the Jacobian of f with row vectors. That is, it computes

$$\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c}), \qquad \text{for fixed } \mathbf{c} \in X \text{ and } \overleftarrow{\mathbf{y}} \in \mathbb{R}^{1 \times m}.$$

It not really possible to encode this evaluation using dual numbers. Instead, an elementary approach, similar to the FAD approach in section 3 will have to

suffice. The Reverse Mode is, for example, described in [3], [4] and [8]. We follow mainly the discussion in [3].

Express again f as the composition $P_Y \circ \Phi_\mu \circ \cdots \circ \Phi_1 \circ P_X$, with P_X, P_Y and the Φ_i as in section 3. The computation of $\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})$ is, by the chain rule, the evaluation of the product

$$\begin{aligned} & \overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c}) = \overleftarrow{\mathbf{y}} \cdot P_Y \cdot \Phi'_{\mu,\mathbf{c}} \ \cdots \ \Phi'_{1,\mathbf{c}} \cdot P_X \\ \Leftrightarrow \quad & J_f(\mathbf{c})^T \cdot \overleftarrow{\mathbf{y}}^T = P_X^T \cdot \Phi'^T_{1,\mathbf{c}} \ \cdots \ \Phi'^T_{\mu,\mathbf{c}} \cdot P_Y^T \cdot \overleftarrow{\mathbf{y}}^T, \end{aligned} \tag{7.1}$$

where again $\Phi'_{i,\mathbf{c}}$ denotes the Jacobian of Φ_i at $(\Phi_{i-1} \circ \cdots \circ \Phi_1 \circ P_X)(\mathbf{c})$.

Obviously, the sequence of state vectors $\mathbf{v}^{[i]} \in H$ is the same as in the Forward Mode case. The difference lies in the computation of the evaluation trace of (7.1), which we denote by $\overline{\mathbf{v}}^{[\mu]} = \overline{\mathbf{v}}^{[\mu]}(\mathbf{c}, \overleftarrow{\mathbf{y}}), ..., \overline{\mathbf{v}}^{[0]} = \overline{\mathbf{v}}^{[0]}(\mathbf{c}, \overleftarrow{\mathbf{y}})$.

For simplicity, assume $P_Y(v_1, ..., v_{n+\mu}) = (v_{n+\mu-m}, ..., v_{n+\mu})$, for all $(v_1, ..., v_{n+\mu}) \in H$ and denote $\overleftarrow{\mathbf{y}}^T = (y'_1, ..., y'_m)$. We define the evaluation trace of (7.1) as

$$\overline{\mathbf{v}}^{[\mu]} := P_Y^T \cdot \overleftarrow{\mathbf{y}}^T = (0, ..., 0, y'_1, ..., y'_m) \ \text{ and } \ \overline{\mathbf{v}}^{[i-1]} := \Phi'^T_{i,\mathbf{c}} \cdot \overline{\mathbf{v}}^{[i]}.$$

By (3.2)

$$\Phi'^T_{i,\mathbf{c}} = \begin{pmatrix} 1 & \cdots & 0 & \frac{\partial \varphi_i}{\partial v_1}(\cdots) & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 1 & \vdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \vdots & 1 & \cdots & 0 \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{\partial \varphi_i}{\partial v_{n+\mu}}(\cdots) & 0 & \cdots & 1 \end{pmatrix},$$

$$\begin{matrix} & & & & (n+i)\text{-th column} & & & \\ & & & & \downarrow & & & \end{matrix}$$

where $\frac{\partial \varphi_i}{\partial v_k}(\cdots) = \frac{\partial \varphi_i}{\partial v_k}(v_{i_1}(\mathbf{c}), ..., v_{i_{n_i}}(\mathbf{c}))$ is interpreted as 0 if φ_i does not depend on v_k, and the $v_{i_1}(\mathbf{c}), ..., v_{i_{n_i}}(\mathbf{c}) \in \{\mathbf{v}_1^{[i-1]}(\mathbf{c}), ..., \mathbf{v}_{n+i-1}^{[i-1]}(\mathbf{c})\}$.

Therefore, each $\overline{\mathbf{v}}^{[i-1]}$ is of the form

$$\overline{\mathbf{v}}^{[i-1]} = \begin{pmatrix} \overline{\mathbf{v}}_1^{[i]} + \frac{\partial \varphi_i}{\partial v_1}(\cdots) \cdot \overline{\mathbf{v}}_{n+i}^{[i]} \\ \vdots \\ \overline{\mathbf{v}}_{n+i-1}^{[i]} + \frac{\partial \varphi_i}{\partial v_{n+i-1}}(\cdots) \cdot \overline{\mathbf{v}}_{n+i}^{[i]} \\ \frac{\partial \varphi_i}{\partial v_{n+i}}(\cdots) \cdot \overline{\mathbf{v}}_{n+i}^{[i]} \\ \overline{\mathbf{v}}_{n+i+1}^{[i]} + \frac{\partial \varphi_i}{\partial v_{n+i+1}}(\cdots) \cdot \overline{\mathbf{v}}_{n+i}^{[i]} \\ \vdots \\ \overline{\mathbf{v}}_{n+\mu}^{[i]} + \frac{\partial \varphi_i}{\partial v_{n+\mu}}(\cdots) \cdot \overline{\mathbf{v}}_{n+i}^{[i]} \end{pmatrix}.$$

The value $\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})$ is then given by

$$\left(\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})\right)^T = P_X^T \cdot \overline{\mathbf{v}}^{[0]}.$$

Note that, in contrast to Forward AD, the sequence of evaluation trace pairs $[\mathbf{v}^{[i]}, \overline{\mathbf{v}}^{[i]}]$ appears in reverse order (that is, $[\mathbf{v}^{[\mu]}, \overline{\mathbf{v}}^{[\mu]}], ..., [\mathbf{v}^{[1]}, \overline{\mathbf{v}}^{[1]}]$). In particular, unlike to Forward AD, it is not efficient to overwrite the previous pair in each computational step. Indeed, since the state vector $\mathbf{v}^{[i]}$ is needed to compute $\overline{\mathbf{v}}^{[i]}$, the pairs $[\mathbf{v}^{[i]}, \overline{\mathbf{v}}^{[i]}]$ should not be computed (as pairs) at all. Instead, it is more efficient to first evaluate the evaluation trace $\mathbf{v}^{[1]}, ..., \mathbf{v}^{[\mu]}$, store these values, and then use them to compute the $\overline{\mathbf{v}}^{[\mu]}, ..., \overline{\mathbf{v}}^{[1]}$ afterwards.

Example 7.1. Consider the function

$$f : \mathbb{R} \to \mathbb{R}^2, \quad \text{with} \quad f(x) = \begin{pmatrix} x \\ \exp(x)\sin(x) \end{pmatrix}.$$

We want to determine $(y_1' \quad y_2') \cdot J_f(c)$ for fixed $\overleftarrow{\mathbf{y}} = (y_1' \quad y_2') \in \mathbb{R}^{1\times 2}$ and $\mathbf{c} = c \in \mathbb{R}$.

Set $H = \mathbb{R}^5$ and $f = P_Y \circ \Phi_4 \circ \Phi_3 \circ \Phi_2 \circ \Phi_1 \circ P_X$ with

$$P_X : \mathbb{R} \to \mathbb{R}^5, \quad \text{with} \quad P_X(x) = (x, 0, 0, 0, 0),$$

$$\Phi_1 : \mathbb{R}^5 \to \mathbb{R}^5, \quad \text{with} \quad \Phi_1(v_1, v_2, v_3, v_4, v_5) = (v_1, \exp(v_1), v_3, v_4,),$$

$$\Phi_2 : \mathbb{R}^5 \to \mathbb{R}^5, \quad \text{with} \quad \Phi_2(v_1, v_2, v_3, v_4, v_5) = (v_1, v_2, \sin(v_1), v_4, v_5),$$

$$\Phi_3 : \mathbb{R}^5 \to \mathbb{R}^5, \quad \text{with} \quad \Phi_3(v_1, v_2, v_3, v_4, v_5) = (v_1, v_2, v_3, v_1, v_5),$$

$$\Phi_4 : \mathbb{R}^5 \to \mathbb{R}^5, \quad \text{with} \quad \Phi_4(v_1, v_2, v_3, v_4, v_5) = (v_1, v_2, v_3, v_4, v_2 \cdot v_3),$$

$$P_Y : \mathbb{R}^5 \to \mathbb{R}, \quad \text{with} \quad P_Y(v_1, v_2, v_3, v_4, v_5) = (v_4, v_5).$$

Clearly, we obtain the evaluation trace $\mathbf{v}^{[0]}(c), ..., \mathbf{v}^{[4]}(c)$ with

$$\mathbf{v}^{[0]}(c) = \begin{pmatrix} c \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, ..., \mathbf{v}^{[4]}(c) = \begin{pmatrix} c \\ \exp(c) \\ \sin(c) \\ c \\ \exp(c)\sin(c) \end{pmatrix}.$$

The Reverse Mode of Automatic Differentiation produces now the vectors $\overline{\mathbf{v}}^{[4]}, ..., \overline{\mathbf{v}}^{[0]}$ with

$$\overline{\mathbf{v}}^{[4]} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ y_1' \\ y_2' \end{pmatrix}, \overline{\mathbf{v}}^{[3]} = \begin{pmatrix} 0 \\ y_2'\sin(c) \\ y_2'\exp(c) \\ y_1' \\ 0 \end{pmatrix}, \overline{\mathbf{v}}^{[2]} = \begin{pmatrix} y_1' \\ y_2'\sin(c) \\ y_2'\exp(c) \\ 0 \\ 0 \end{pmatrix},$$

$$\overline{\mathbf{v}}^{[1]} = \begin{pmatrix} y_1' + y_2'\cos(c)\exp(c) \\ y_2'\sin(c) \\ 0 \\ 0 \\ 0 \end{pmatrix}, \overline{\mathbf{v}}^{[0]} = \begin{pmatrix} y_1' + y_2'\exp(c)(\sin(c) + \cos(c)) \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Then

$$(y_1' \quad y_2') \cdot J_f(c) = P_X^T \cdot \overline{\mathbf{v}}^0 = y_1' + y_2' \exp(c)(\sin(c) + \cos(c)).$$

We summarize:

Theorem 7.2. *The evaluation of* $\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})$ *can be achieved by computing the vectors* $\mathbf{v}^{[0]}, ..., \mathbf{v}^{[\mu]}$ *and* $\overline{\mathbf{v}}^{[\mu]}, ..., \overline{\mathbf{v}}^{[0]}$*, where the computation of each* $\overline{\mathbf{v}}^{[i-1]}$ *is effectively the computation of the real numbers*

$$\overline{\mathbf{v}}_k^{[i]} + \frac{\partial \varphi_i}{\partial v_k}(v_{i_1}(\mathbf{c}), ..., v_{i_{n_i}}(\mathbf{c})) \cdot \overline{\mathbf{v}}_{n+i}^{[i]}, \quad k \neq n+i,$$

and

$$\frac{\partial \varphi_i}{\partial v_{n+i}}(v_{i_1}(\mathbf{c}), ..., v_{i_{n_i}}(\mathbf{c})) \cdot \overline{\mathbf{v}}_{n+i}^{[i]}.$$

Remark 7.3. (i) It is not hard to see that the Reverse Mode is the *dual* concept of the Forward Mode. As a matter of fact, while $J_f(\mathbf{x}) \cdot \overrightarrow{\mathbf{x}}$ is the push-forward of f at $\mathbf{x}$ applied to $\overrightarrow{\mathbf{x}}$, the matrix product $\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{x})$ is the *pull-back* of f at $\mathbf{x}$ applied to the linear map $\overleftarrow{\mathbf{y}}$ (which is an element of $\mathbb{R}^{1\times m}$, the dual space of $\mathbb{R}^m$). Indeed, given $\mathbf{x}$, we have trivially

$$\left(\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{x})\right) \cdot \overrightarrow{\mathbf{x}} = \overleftarrow{\mathbf{y}} \cdot \left(J_f(\mathbf{x}) \cdot \overrightarrow{\mathbf{x}}\right), \tag{7.2}$$

for all $\overleftarrow{\mathbf{y}} \in \mathbb{R}^{1\times m}, \overrightarrow{\mathbf{x}} \in \mathbb{R}^n$.

(ii) If we use the notations $\mathbf{v'}^{[-1]} := \overrightarrow{\mathbf{x}}, \mathbf{v'}^{[\mu+1]} := J_f(\mathbf{c}) \cdot \overrightarrow{\mathbf{x}}$ and $\overline{\mathbf{v}}^{[\mu+1]} := \overleftarrow{\mathbf{y}}^T$, $\overline{\mathbf{v}}^{[-1]} := \left(\overleftarrow{\mathbf{y}} \cdot J_f(\mathbf{c})\right)^T$, then (7.2) (with $\mathbf{x} = \mathbf{c}$) reads

$$\overline{\mathbf{v}}^{[-1]^T} \cdot \mathbf{v'}^{[-1]} = \overline{\mathbf{v}}^{[\mu+1]^T} \cdot \mathbf{v'}^{[\mu+1]}.$$

In fact, given $\mathbf{c}, \overrightarrow{\mathbf{x}}$ and $\overleftarrow{\mathbf{y}}$, it follows immediately from the definitions of $\mathbf{v'}^{[i]}$ and $\overline{\mathbf{v}}^{[i]}$ that the scalar products of the evaluation trace vectors of the Forward and Reverse Mode

$$\overline{\mathbf{v}}^{[i]^T} \cdot \mathbf{v'}^{[i]}$$

are constant (equal to $\overleftarrow{\mathbf{y}} J_f(\mathbf{c}) \overrightarrow{\mathbf{x}}$) for all $i = -1, ..., \mu+1$. (That is, at each time of the computations.)

Acknowledgement I

I am thankful to Barak Pearlmutter for valuable suggestions which led to an improvement of this paper.

Acknowledgment II

This work was supported by Science Foundation Ireland grant 09/IN.1/I2637.

References

[1] W. K. Clifford, Preliminary Sketch of Bi-quaternions,*Proceedings of the London Mathematical Society* 4 (1873), 381–395.

[2] A. Griewank, Achieving Logarithmic Growth of Temporal and Spatial Complexity in Reverse Automatic Differentiation, *Optimization Methods and Software* 1 (1992), 35–54.

[3] A. Griewank, A Mathematical View of Automatic Differentiation, *Acta Numerica* 12 (2003), 321–398.

[4] A. Griewank and A. Walther, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, second edition, SIAM, Philadelphia, PA, 2008.

[5] D. Kalman, Double Recursive Multivariate Automatic Differentiation, *Mathematics Magazine* 75 (3) (2002), 187–202.

[6] O. Manzyuk, A Simply Typed λ-Calculus of Forward Automatic Differentiation, *Electronic Notes in Theoretical Computer Science* 286 (2012), 257–272.

[7] B. A. Pearlmutter and J. M. Siskind, Lazy Multivariate Higher-Order Forward-Mode AD, *Proc of the 2007 Symposium on Principles of Programming Languages, Nice, France* (2007), 155–160.

[8] B. A. Pearlmutter and J. M. Siskind, Reverse-Mode AD in a Functional Framework: Lambda the Ultimate Backpropagator, *TOPLAS* 30 (2) (2008), 1–36.

[9] L. B. Rall, Differentiation and generation of Taylor coefficients in Pascal-SC, In: A New Approach to Scientific Computation, Academic Press, New York, 1983, 291–309.

[10] L. B. Rall, The Arithmetic of Differentiation, *Mathematics Magazine* 59, 1986, 275–282.

[11] J. M. Siskind and B. A. Pearlmutter, Nesting Forward-Mode AD in a Functional Framework, *Higher-Order and Symbolic Computation* 21 (4) (2008),361–376.

[12] R. Wengert, A Simple Automatic Derivative Evaluation Program, *Communications of the ACM* 7 (8) (1964), 463–464.

Philipp Hoffmann
National University of Ireland Maynooth
Department of Computer Science
Maynooth, Co. Kildare
Ireland
email: philipp.hoffmann@cs.nuim.ie
philip.hoffmann@maths.ucd.ie