Some notes on `tenebris`

Felix Widmaier

December 12, 2025

Abstract

A small and simple `Python` library that supports automatic differentiation using dual numbers and solving some simple equations.

Getting started

The code of `tenebris` was originally published on

`https://github.com/fwidmaier/tenebris`

To install the library, simply use

```
pip install dist/tenebris-1.0-py2.py3-none-any.whl
```

To build the wheel file of the library, use

```
python setup.py bdist_wheel -universal
```

Dependencies. The required packages to work with `tenebris` are listed in `requirements.txt`. Currently, there are no dependencies. Only standard libraries are used.

1 Dual numbers

Define $\mathcal{D} = \mathbb{R}[\varepsilon]/\langle\varepsilon^2\rangle$ to be the polynomial ring with one variable ε modulo the ideal generated by ε^2. So any element in $\mathcal{D}$ is of the form

$$d = a + b\varepsilon$$

for some $a, b \in \mathbb{R}$. We call a the *real part* of d and b the *dual part* of d. Let us consider some analytic function $f\colon \mathbb{R} \to \mathbb{R}$. Then by the Taylor expansion of f we have for some $x + b\varepsilon \in \mathcal{D}$

$$\begin{aligned} f(x + b\varepsilon) &= f(x) + f'(x)b\varepsilon + \frac{f''(x)}{2}b^2\varepsilon^2 + \frac{f'''(x)}{6}b^3\varepsilon^3 + \cdots \\ &= f(x) + f'(x)b\varepsilon. \end{aligned}$$

Hence $f(x+\varepsilon) = f(x) + f'(x)\varepsilon$. `Et voilà!` This is basically all we need for automatic differentiation using dual numbers! We only need to know the dual part of $f(x+\varepsilon)$ and we obtain $f'(x)$.

Example. Let us go through a little toy example in order to get some better grasp on dual numbers. For instance, let $f\colon x \mapsto x^n$ for some $n \in \mathbb{N}$. Then

$$\begin{aligned} f(x+\varepsilon) = (x+\varepsilon)^n &= \sum_{k=0}^{n} \binom{n}{k} x^{n-k}\varepsilon^k \\ &= x^n + nx^{n-1}\varepsilon + \binom{n}{2}x^{n-2}\varepsilon^2 + \cdots = x^n + nx^{n-1}\varepsilon. \end{aligned}$$

Just as expected.

Basic computations show

$$(a+b\varepsilon) + (c+d\varepsilon) = a + c + (b+d)\varepsilon, \qquad (a+b\varepsilon)(c+d\varepsilon) = ac + (ad+bc)\varepsilon$$
$$\text{and} \qquad (a+b\varepsilon)(c+d\varepsilon)^{-1} = \frac{a}{c} + \frac{bc-ad}{c^2}\varepsilon \qquad \text{where } c \neq 0.$$

This arithmetic of dual numbers is basically encoded in the class `Dual`. The differentiating operator is implemented as `d`. It evaluates the given function on $x + \varepsilon$ (or rather `Dual(x,1)`) and returns the dual part of the result. We can now give a small example:

Example. A priori we can now differentiate any polynomial! For example, consider the function $f\colon x \mapsto x^2 - x - 1$. We expect the derivative to vanish at $\frac{1}{2}$. Using `tenebris`, we can write the following little piece of code:

```
from tenebris import d

f = lambda x: x * x - x - 1
df = d(f)  # the derivative of f. This is literally al it takes!
print(df(0.5))
```

And the output of this is `0.0` – just as we expected.