

# Dual numbers

Miles Gould  
[@pozorvlak@mathstodon.xyz](mailto:@pozorvlak@mathstodon.xyz)

(1 of 2,147)110%



Contents

- ▶ Preface
- ▶ A Armv8-M Architecture Introduction and Overview
- ▼ B Armv8-M Architecture Rules
  - ▶ B1 Resets
  - ▶ B2 Power Management
  - ▶ B3 Programmers' Model
  - ▶ B4 Floating-point Support
  - ▶ B5 Vector Extension
  - ▶ B6 Pointer authentication and branch target identification Extension
  - ▶ B7 Memory Model

# Arm®v8-M Architecture Reference Manual

The image shows a screenshot of a document viewer displaying the table of contents for the 'Arm®v8-M Architecture Reference Manual'. On the left, there is a sidebar with a list of contents. The main area on the right displays the title 'Arm®v8-M Architecture Reference Manual' in a large, black, sans-serif font. The sidebar contains the following items:

- Contents
- ▶ Preface
- ▶ A Armv8-M Architecture Introduction and Overview
- ▼ B Armv8-M Architecture Rules
  - ▶ B1 Resets
  - ▶ B2 Power Management
  - ▶ B3 Programmers' Model
  - ▶ B4 Floating-point Support
  - ▶ B5 Vector Extension
  - ▶ B6 Pointer authentication and branch target identification Extension
  - ▶ B7 Memory Model

- The image shows the cover page of the 'Arm®v8-M Architecture Reference Manual'. On the left side, there is a grey sidebar with a table of contents. The contents include: 'Contents', '► Preface', '► A Armv8-M Architecture Introduction and Overview', '▼ B Armv8-M Architecture Rules' (which is expanded), and a list of sub-sections under B: '► B1 Resets', '► B2 Power Management', '► B3 Programmers' Model', '► B4 Floating-point Support', '► B5 Vector Extension', '► B6 Pointer authentication and branch target identification Extension', and '► B7 Memory Model'. The main area of the page is white and features the title 'Arm®v8-M Architecture Reference Manual' in a large, black, sans-serif font. At the top of the sidebar, there are several small icons: a document icon, a search icon, a list icon, and a refresh icon.

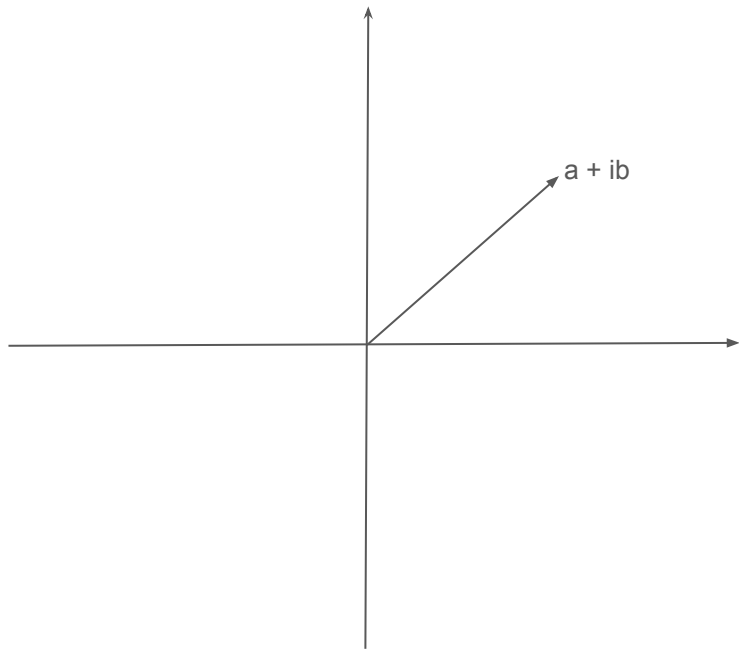
### **C2.4.381 VMLADAV Vector Multiply Add Dual Accumulate Across Vector.**

The elements of the vector registers are handled in pairs. In the base variant, corresponding elements from the two source registers are multiplied together, whereas the exchange variant swaps the values in each pair of values read from the first source register, before multiplying them with the values from the second source register. The results of the pairs of multiply operations are combined by adding them together. At the end of each beat these results are accumulated and the lower 32 bits written back to the general-purpose destination register. The initial value of the general-purpose destination register can optionally be added to the result.

### **C2.4.381 VMLADAV Vector Multiply Add **Dual** Accumulate Across Vector.**

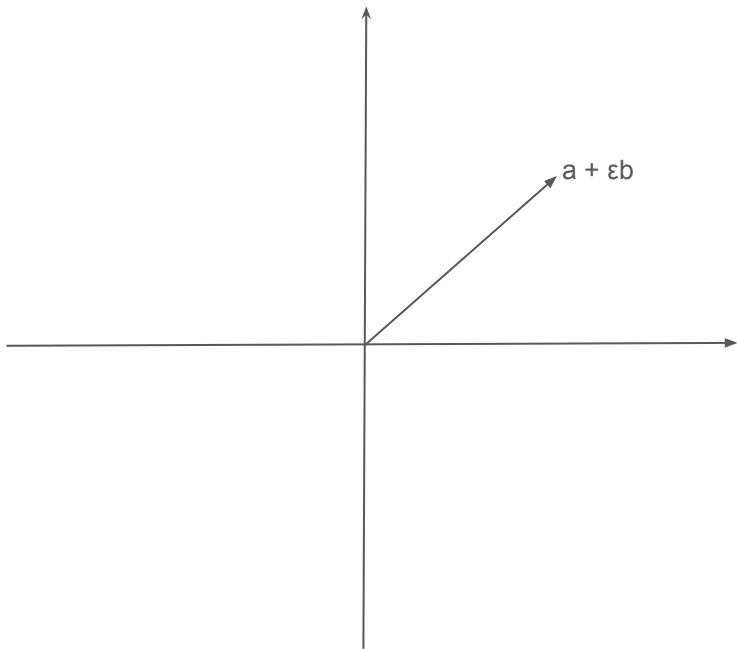
The elements of the vector registers are handled in pairs. In the base variant, corresponding elements from the two source registers are multiplied together, whereas the exchange variant swaps the values in each pair of values read from the first source register, before multiplying them with the values from the second source register. The results of the pairs of multiply operations are combined by adding them together. At the end of each beat these results are accumulated and the lower 32 bits written back to the general-purpose destination register. The initial value of the general-purpose destination register can optionally be added to the result.

# Complex numbers



- 2d vectors
- Write “ $a + ib$ ” for  $(a, b)$
- “Real” and “imaginary” parts
- $(a + ib) + (c + id) = (a + c) + i(b + d)$
- $i^2 = -1$
- $(a + ib)(c + id) = (ac - bd) + i(bc + ad)$

# Dual numbers



- 2d vectors
- Write “ $a + \epsilon b$ ” for  $(a, b)$
- “Body” and “soul”
- $(a + \epsilon b) + (c + \epsilon d) = (a + c) + \epsilon(b + d)$
- $\epsilon^2 = 0$
- $(a + \epsilon b)(c + \epsilon d) = (ac + \cancel{bd}) + \epsilon(bc + ad)$

You have seen these formulae before!

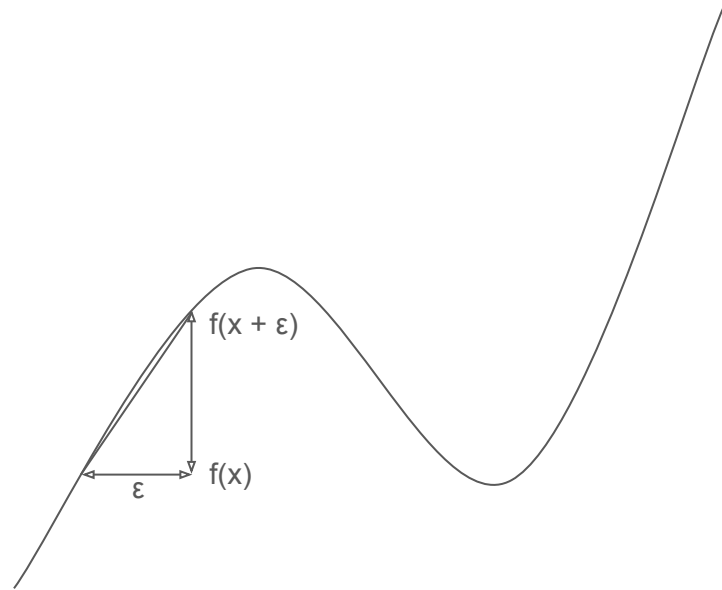
$$(a + \varepsilon b) + (c + \varepsilon d) = (a + c) + \varepsilon(b + d) \quad (f(x) + g(x))' = f'(x) + g'(x)$$

$$(a + \varepsilon b)(c + \varepsilon d) = ac + \varepsilon(bc + ad) \quad (f(x)g(x))' = f'(x)g(x) + f(x)g'(x)$$



# Automatic differentiation

$$f(x + \varepsilon) = f(x) + \varepsilon f'(x)$$



# Does it work?

Let  $f(x) = x$

$f(x + \varepsilon) = x + \varepsilon$

# Does it work?

$$\text{Let } f(x) = x^2$$

$$f(x + \varepsilon) = (x + \varepsilon)^2$$

$$= x^2 + 2x\varepsilon + \varepsilon^2$$

$$= x^2 + 2x\varepsilon$$

# Does it work?

$$\text{Let } f(x) = x^3$$

$$f(x + \varepsilon) = (x + \varepsilon)^3$$

$$= x^3 + 3x^2\varepsilon + 3x\varepsilon^2 + \varepsilon^3$$

$$= x^3 + 3x^2\varepsilon$$

# Does it work?

$$\text{Let } f(x) = x^n$$

$$f(x + \varepsilon) = (x + \varepsilon)^n$$

$$= x^n + nx^{n-1}\varepsilon + \dots$$

$$= x^n + nx^{n-1}\varepsilon$$

# Yes!

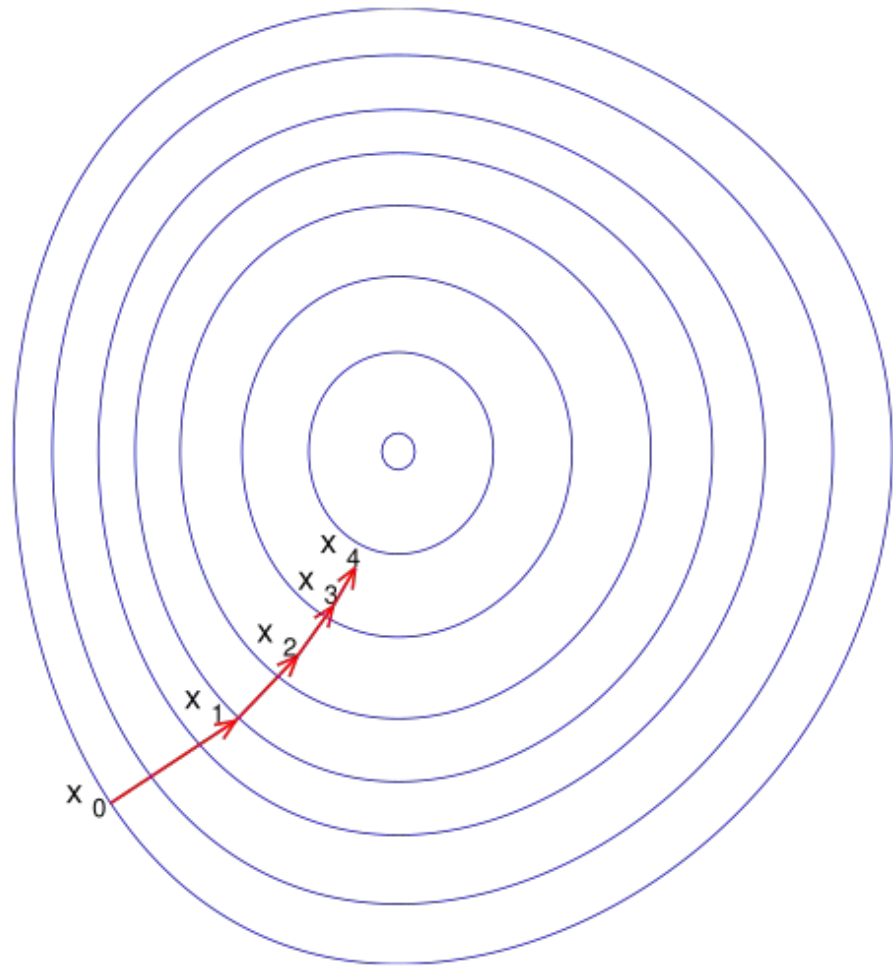
- Powers of  $n$  work
- Sums and products work
- So polynomial functions work
- So approximations to power series work
- But in fact, *arbitrary code* works
- We can extend this to higher or partial derivatives

# Why do we care?

Physics simulations!

Computer graphics!

Gradient descent!



# Historical note

- I first learned about these from Dan Piponi
- He used them in 2005 to do special effects for the *Matrix* sequels
- At the time they weren't widely-known in the graphics community
- But apparently Yoshua Bengio's deep-learning group already knew about them
  - "Source: rooftop beers" – Paul Khuong