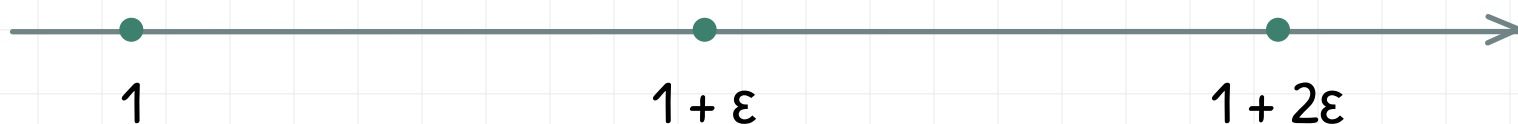


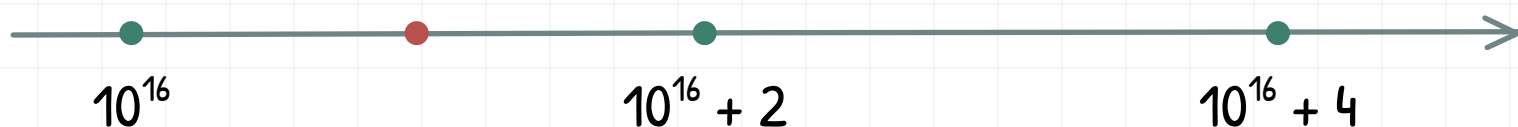
Float là những điểm rời rạc

Khoảng cách giữa hai số biểu diễn được tăng theo độ lớn của chúng.

quanh 1: bước $\approx 2.22 \times 10^{-16}$



quanh 10^{16} : bước = 2



+1: ở giữa, làm tròn về 10^{16}

Từ hình đến công thức

Số chuẩn float64: $(-1)^s \cdot (1 + \text{phần_lẻ}) \cdot 2^e$; 1 bit dấu, 11 bit mũ, 52 bit phần lẻ.

Gần 10^{16} , bước giữa hai float là 2. Số $10^{16} + 1$ nằm giữa và làm tròn về 10^{16} .

0.1 và 0.2 không có biểu diễn nhị phân hữu hạn; 0.5 và 0.25 thì có.

Sai số có cấu trúc theo thang số và quy tắc làm tròn.

Trừ số lớn để lấy số rất nhỏ

Cùng một phương sai toán học, hai cách tính cho kết quả khác nhau.

$x: 10^8+1, 10^8+2, 10^8+3, 10^8+4, 10^8+5$

$$E[x^2] - E[x]^2$$



4.0 (sai)

$$E[(x-\bar{x})^2]$$



2.0 (đúng)

Từ hình đến công thức

$E[x^2]$ và $E[x]^2$ đều cỡ 10^{16} . Sai số của chúng có thể lấn át hiệu nhỏ cần tìm.

Trừ trung bình trước: $x-\bar{x}=(-2,-1,0,1,2)$; mean bình phương bằng 2.

Dùng phương pháp hai lượt hoặc Welford thay vì hiệu hai moment lớn.

Triệt tiêu phơi bày sai số có sẵn; không phải mọi phép trừ gần nhau đều hỏng.

Đổi ngoặc, đổi kết quả

Số 1 bị mất nếu cộng riêng vào 10^{16} ; cộng hai số 1 trước thì còn.

Thứ tự	Kết quả
$(10^{16}+1)+1$	10^{16}
$10^{16}+(1+1)$	$10^{16}+2$
toán số thực	cả hai = $10^{16}+2$

Từ hình đến công thức

Phép cộng float không kết hợp. Đổi thứ tự reduction có thể đổi kết quả.

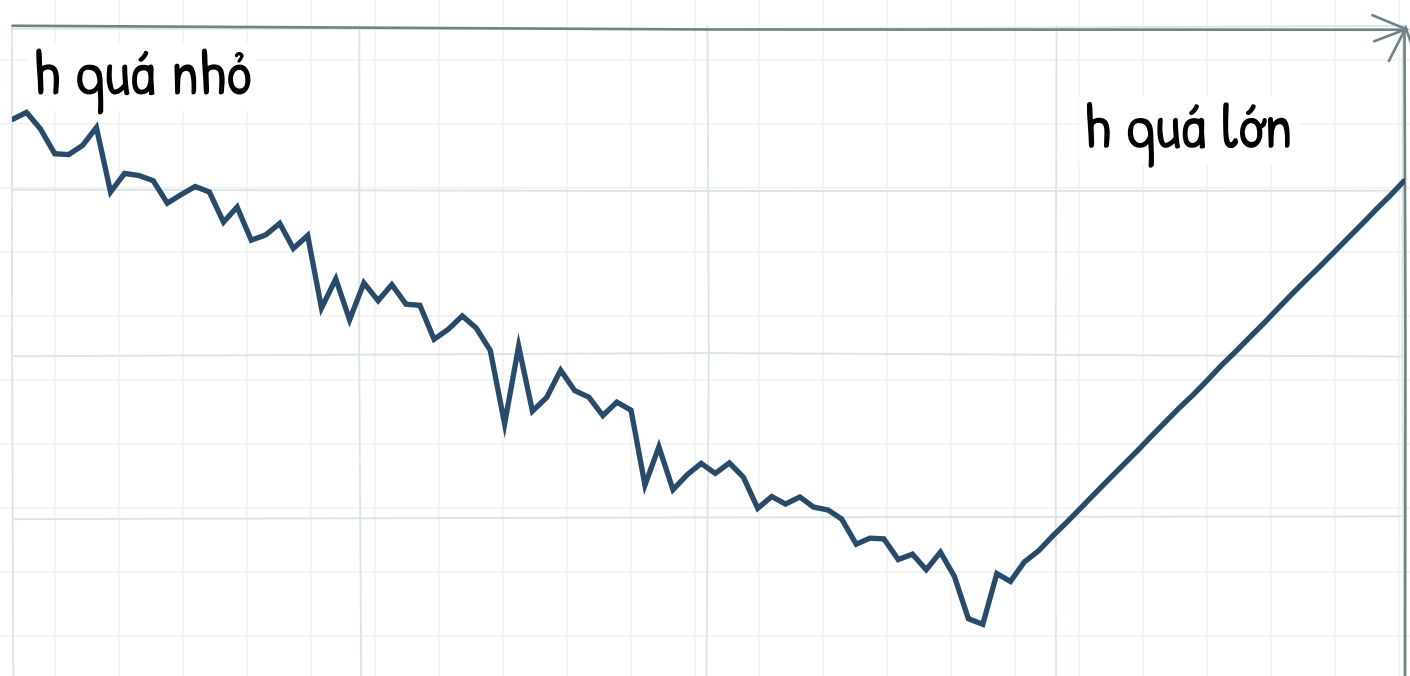
Kahan giữ phần làm tròn bị mất để bù vào các bước sau; vẫn không chữa mọi tình huống.

Cố định seed chưa đủ: còn dtype, kernel, phần cứng và thứ tự cộng.

Khi so hai lần chạy, phân biệt giống từng bit với gần nhau trong sai số.

Hai sai số kéo ngược nhau

Đường sai số đo cho sin tại 1: giảm h ban đầu có lợi, rồi phản tác dụng.



— trục ngang: $\log_{10} h$

— dọc: $\log_{10}$ sai số

Từ hình đến công thức

Sai phân trung tâm của hàm đủ trơn: cắt cụt thường cỡ $C \cdot h^2$.

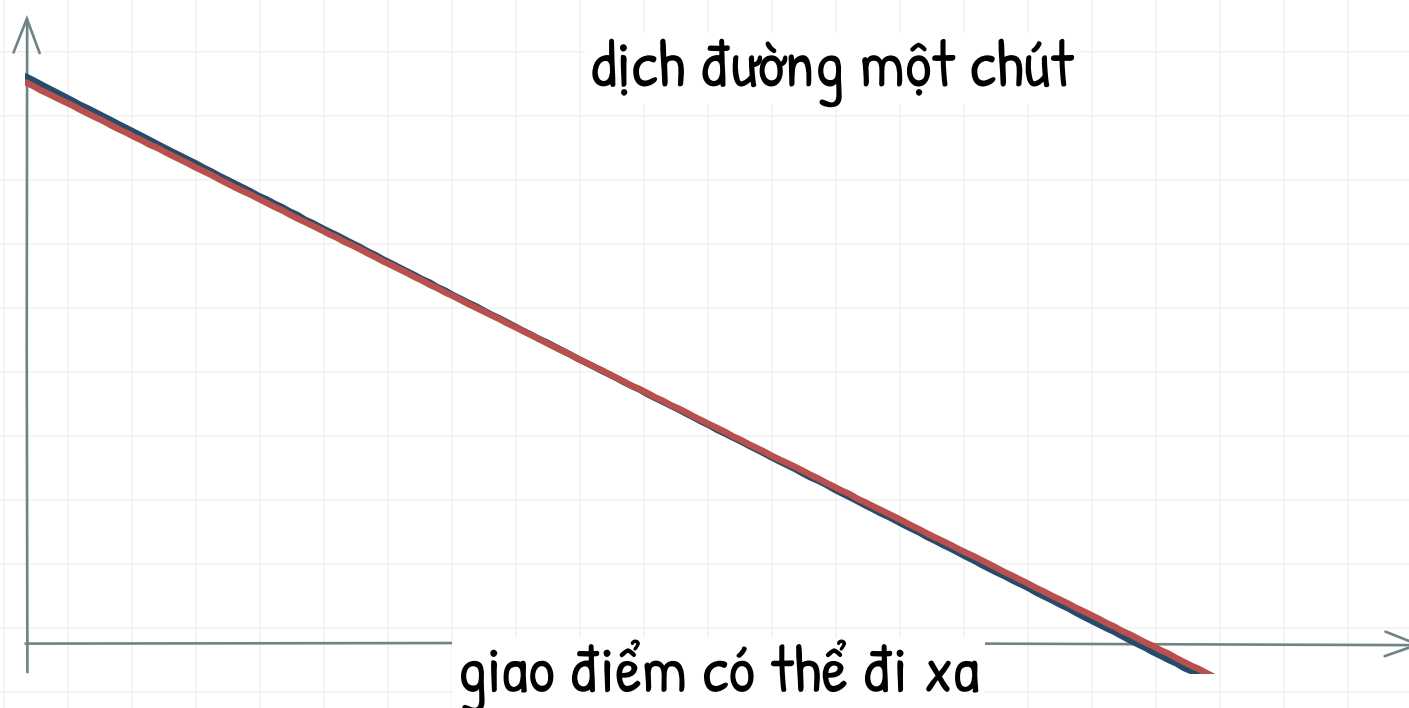
Làm tròn và triệt tiêu thường góp cỡ ε/h , phụ thuộc thang của hàm.

Với x^2 , trung tâm có cắt cụt bằng 0 trong số học chính xác; không dùng nó để minh họa đủ hai phía.

Quét h , dùng float64 khi kiểm gradient; tránh h nhỏ đến mức $x+h=x$.

Bài toán nhảy với đầu vào

Hai đường gần song song: giao điểm nhảy với một dịch chuyển nhỏ.



— hai đường gần song song

Từ hình đến công thức

$\kappa_2(A) = \sigma_{\max} / \sigma_{\min}$ cho A khả nghịch. κ lớn báo khả năng khuếch đại sai số tương đối.

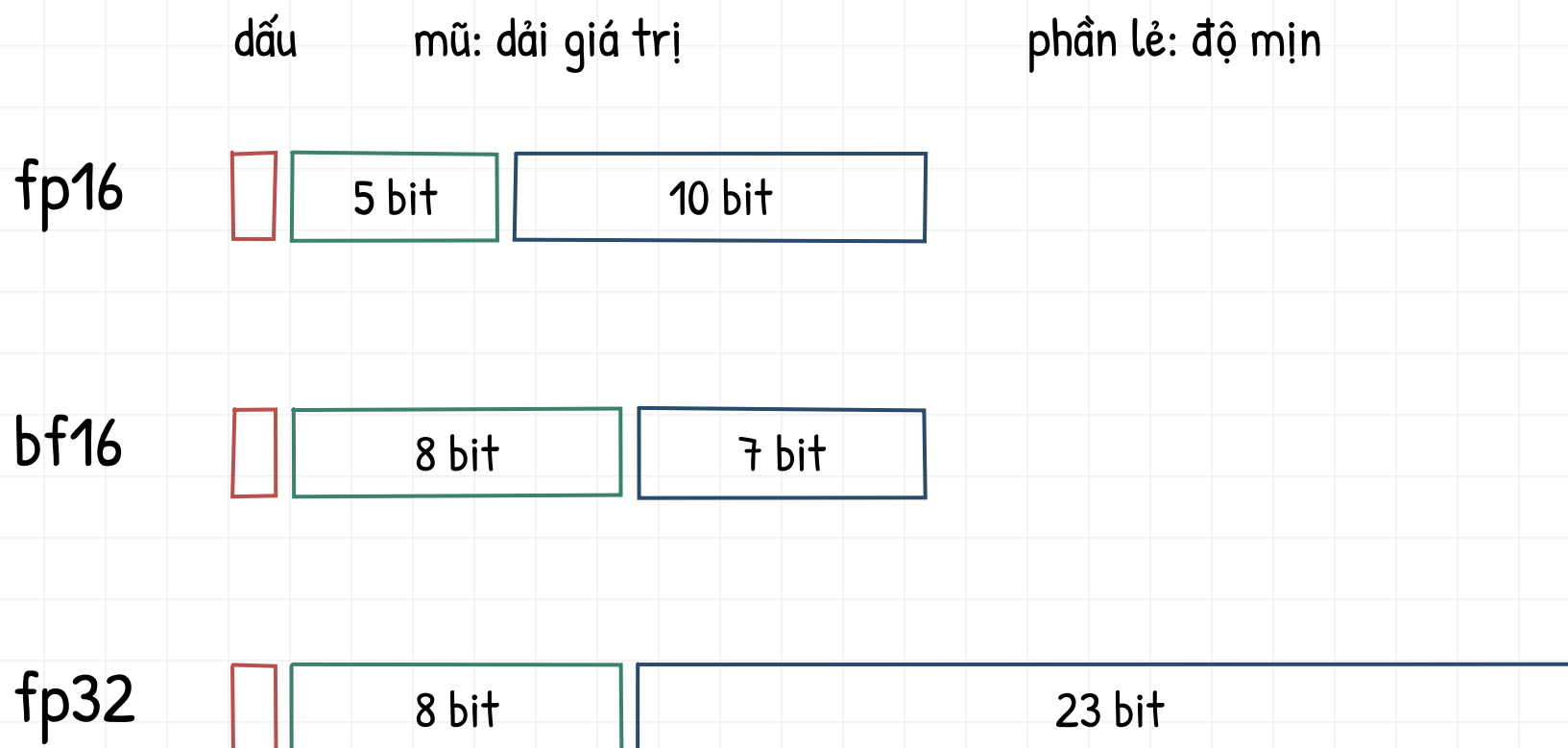
$A = \begin{bmatrix} 1 & 1 \\ 1 & 1+10^{-8} \end{bmatrix}$, $\kappa \approx 4 \times 10^8$. $b = (2, 2)$ đổi thêm 10^{-10} ở phần tử 2: nghiệm $(2, 0) \rightarrow (1.99, 0.01)$.

det nhỏ chưa đủ: $A = 0.1I_{10}$ có $\det = 10^{-10}$ nhưng $\kappa = 1$.

Điều kiện thuộc bài toán; ổn định số thuộc thuật toán. Cần xét cả hai.

Dải rộng hay các nấc mịn?

Bit mũ mở rộng dải; bit phần lẻ làm dày các số trong cùng một khoảng.



Từ hình đến công thức

fp16: 5 bit mũ, 10 bit phần lẻ, $\max = 65504$.

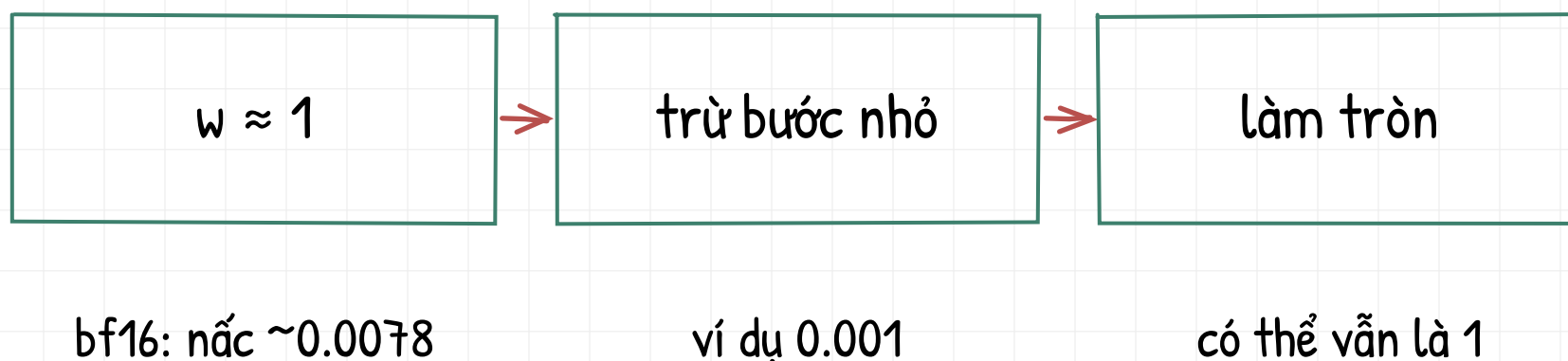
bf16: 8 bit mũ, 7 bit phần lẻ, $\max \approx 3.39 \times 10^{38}$; fp32
 $\max \approx 3.40 \times 10^{38}$.

bf16 ít overflow hơn fp16 trong nhiều tác vụ, nhưng làm tròn thô hơn; cần kiểm theo mô hình.

Cùng 16 bit không có nghĩa cùng độ chính xác hay cùng dải giá trị.

Vì sao giữ bản trọng số fp32?

Một bước cập nhật nhỏ có thể lọt giữa hai số bf16 biểu diễn được.



Từ hình đến công thức

Mixed precision thường dùng số thấp chính xác cho phép nhân lớn, giữ một số phép cộng / trọng số ở fp32.

Bản master fp32 tích lũy cập nhật nhỏ tốt hơn; chi tiết phụ thuộc cách triển khai.

Chẩn đoán: kiểm finite, dtype, cách gộp loss, so bản float64 nhỏ, rồi thử công thức ổn định.

Tối ưu tốc độ sau khi đã có bản tham chiếu chạy đúng.

Tự kiểm trước khi đi tiếp

Che đáp án, thử nói bằng lời

1. Vì sao $10^{16}+1$ có thể vẫn bằng 10^{16} ?
2. Tính variance của $10^8+[1,2,3,4,5]$ ra sao?
3. Seed giống nhau đủ cho kết quả giống từng bit?
4. $\text{det}=10^{-10}$ có buộc K lớn không?
5. bf16 hơn fp16 ở đâu, thua ở đâu?

Đáp nhanh

1. Khoảng cách float64 ở đó là 2; phép cộng bị làm tròn.
2. Trừ mean trước rồi mean bình phương: 2.0.
3. Chưa: còn phần cứng, kernel, dtype, thứ tự reduction.
4. Không. 0.1_{10} có định thức đó nhưng $K=1$.
5. Dải số rộng hơn; ít bit phần lẻ nên độ mịn kém hơn.