

Physics 234: Exercise 4

Imagine a hypothetical non-binary computer whose floating-point numbers are encoded using one sign bit and seven *decimal* digits. Each number is stored internally in the format

±		i		f ₄		f ₃		f ₂		f ₁		f ₀		e
---	--	---	--	----------------	--	----------------	--	----------------	--	----------------	--	----------------	--	---

and interpreted as the value $(\pm, i, f, e) = \pm i.f_4f_3f_2f_1f_0 \times 10^{e-5}$ with the exponent offset by five. For example, the numbers π , 1, and $-1/3$ are represented as follows:

$\pi \doteq 3.14159 \times 10^0 = (+, 3, 14159, 5)$	+	3		1	4		1	5		9		5
$1 = 1.00000 \times 10^0 = (+, 1, 00000, 5)$	+	1		0	0		0	0		0		5
$-1/3 \doteq -3.33333 \times 10^{-1} = (-, 3, 33333, 4)$	-	3		3	3		3	3		3		4

1. The integer part of the significand is specified explicitly. Why can't we throw it away, like the "hidden bit" in conventional binary floating point?
2. Let's assume that none of the exponent values are reserved to flag special states (such as **inf** or **nan**). What are the numbers of largest magnitude that can be represented? What are the smallest numbers that still retain their full six digits of significance?
3. Numbers that are smaller still can be represented in denormalized form by specifying $i = 0$. For normalized numbers, division by 10 is implemented as a decrement of the exponent ($e \rightarrow e - 1$) and for denormalized numbers as a rightward shift of the significand ($i.f_4f_3f_2f_1f_0 \rightarrow Z.IF_4F_3F_2F_1$). We'll use a rounding convention such that $Z.IF_4F_3F_2F_1 = 0.i f_4 f_3 f_2 f_1$ if $f_0 < 5$; otherwise, $F_1 = (f_1 + 1 \bmod 10)$ with the possibility of a carry passed upward to F_2 and likewise to F_3 , F_4 , I , and Z . So, for example, $(+, 9, 99999, 1)$ divided by 10 is $(+, 9, 99999, 0)$; but divided by 10 twice again gives $(+, 1, 00000, 0)$ and $(+, 0, 10000, 0)$. Write out the sequence formed by successive division of 1.47948 by 10. Show that the sequence goes through five denormalized values before it achieves a true zero.
4. Given a function $f(x) = 1 + x^2$, let's compute a naive estimate for the value of $f'(x_0)$ using only the numerical values of f evaluated at points $x_0 = 0.21$ and $x_1 = x_0 + 0.0125$. If we carry out the calculation using real numbers (with unrestricted accuracy), we find that $[f(x_1) - f(x_0)]/(x_1 - x_0) = (1.04950625 - 1.0441)/(0.0125) = 0.4325$. What is the result if we redo this within the floating-point scheme described above?
5. Since $f(x)$ is quadratic, the finite difference estimate goes linearly with the step size:

$$D^{(1)}(x_0) = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{(x_0 + \Delta x)^2 - x_0^2}{\Delta x} = 2x_0 + \Delta x.$$

The second column of the following table reports $D^{(1)}(0.21) = 0.42 + \Delta x$ computed exactly. Complete the missing entries in the third column, which represent floating-point calculations.

Δx	$(\Delta f/\Delta x)_{\mathbb{R}}$	$(\Delta f/\Delta x)_{\text{FP}}$
0.1	0.52	0.520000
0.05	0.47	0.470000
0.02	0.44	0.440000
0.01	0.43	0.430000
0.005	0.425	0.426000
0.002	0.422	
0.001	0.421	
0.0009	0.4209	
0.0008	0.4208	
0.0007	0.4207	
0.0006	0.4206	
0.0005	0.4205	
0.0004	0.4204	
0.0003	0.4203	
0.0002	0.4202	
0.0001	0.4201	

6. We know that first-order finite difference estimates in the limit of $\Delta x \rightarrow 0$ are numerically unreliable. A better strategy for computing derivatives is an extrapolation using higher-order differences computed at Δx values far from infinitesimal. Complete the Richardson tableau shown below.

Δx	$x_0 + \Delta x$	$f(x_0 + \Delta x)$	$D^{(1)}$	$D^{(2)}$	$D^{(3)}$	$D^{(4)}$
0	0.21	1.04410				
0.01	0.22	1.04840	0.430000			
0.02	0.23	1.05290	0.440000	0.420000		
0.05	0.26	1.06760	0.470000			
0.01	0.31	1.09610	0.520000			