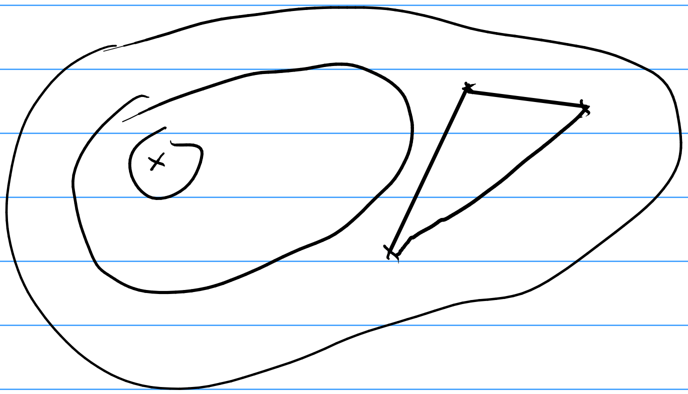
What if we don't even have one derivative, let alone two?!

$$x_{k+1} = x_k - \underbrace{\partial_f(x_k)}_{n \times n} \underbrace{f(x_k)}_{n}$$

Secant-updating methods: BFGS



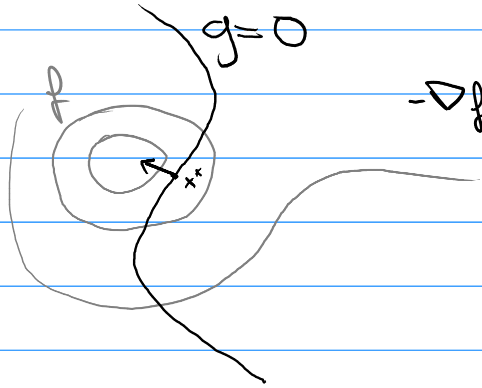
Constrained Optimization

Modify the problem statement of optimization to accommodate a constraint.

$$f: \mathbb{R}^n \to \mathbb{R} \qquad \vec{g}(\vec{x}) = 0$$

What does a solution/minimum x^* of this problem look like?

I.e. what are some necessary conditions on x^* ?



$$-\nabla f = \nabla g \cdot \lambda = \nabla g^T \lambda$$
$$= J_g^T \lambda \quad \text{for some } \lambda$$

Miracle: Reduce constrained to un-constrained optimization.

Define a new function of more unknowns: x and λ, $\lambda \in \mathbb{R}^m$

$$\mathcal{L}(x, \lambda) := f(x) + g(x)^T \vec{\lambda}$$

What are the necessary conditions for an un-constrained minimum of $\mathcal{L}$?

$$\nabla \mathcal{L}(\vec{x}, \vec{\lambda}) = \begin{array}{c} \frac{\partial}{\partial x} \rightarrow \\ \frac{\partial}{\partial \lambda} \rightarrow \end{array} \begin{pmatrix} \nabla f + J_g^T \lambda \\ g(x) \end{pmatrix} = 0$$

Using Newton's method on $\mathcal{L}$ gets a new name:

Sequential Quadratic Programming

Can you do an example?

Minimize $(x-2)^4 + 2(y-1)^2$ subject to $x+4y=3$

Uncondr. min: $x=2$, $y=1$

$$\mathcal{L}(x,\lambda) = (x-2)^4 + 2(y-1)^2 + \lambda(x+4y-3)$$

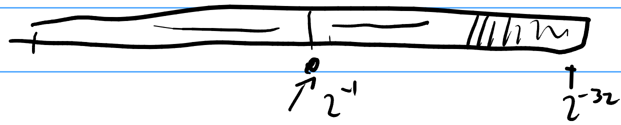
$$\nabla\mathcal{L} = \begin{pmatrix} 4(x-2)^3 + \lambda \\ 4(y-1) + 4\lambda \\ x+4y-3 \end{pmatrix}$$

$$H_{\mathcal{L}} = \begin{pmatrix} 12(x-2)^2 & 0 & 1 \\ 0 & 4 & 4 \\ 1 & 4 & 0 \end{pmatrix}$$

13

Floating Point Arithmetic

What are the main problems with fixed point arithmetic?



limited range

super-inaccurate for small numbers

Idea: Set a few bits aside to store the largest exponent. How?

$$2^3 + 2^2 + 2^0 = (\overset{2^3}{\downarrow}1101)_2 = (\underbrace{1.101}_{\text{significand}})_2 \cdot 2^{\underbrace{3}_{\text{exponent}}}$$

Write these here as a floating point number:

$$12.25 = (\overset{2^3}{\downarrow}1100.01)_2 = (1.10001) \cdot 2^3$$

$$0.125 = (0.001)_2 = (1.000)_2 \cdot 2^{-3}$$

$$2^{50} = (1.000)_2 \cdot 2^{50}$$

$$2^{-100} = (1.000) \cdot 2^{-100}$$

The first digit of the significand seems to always be a one. Do we need to store it?

No.

$$\left(\underbrace{1}.\underbrace{10001}\right)_2 \cdot 2^{\underbrace{3}}$$

what we actually store

This is called the "implied one".

How is zero represented?

$$0 = 1.\underbrace{\quad\quad\quad\quad} \cdot 2^{\underbrace{\quad\quad}}$$

Make a new rule: If the exponent has a special value,

then turn off the implied one. -1023

Stored for 0:

significand: 000000 exp: -1023

What happens if you run out of digits to store in your significand?

Suppose, for the sake of argument, that we store four bits of the significand. But the true number we would like to represent has seven binary digits.

$(1110.011)_2 \cdot 2^0 = (1.\underbrace{1100}11)_2 \cdot 2^3$

3.14159

3.142

Idea: Round the number.

$(1110.011)_2 = (1.\underbrace{1100}11)_2 \cdot 2^3 \approx (1.\underbrace{1101})_2 \cdot 2^3$

What is a denormal number?

Suppose the smallest exponent you can represent is -1022.

How would you represent 2^{-1025}?

$2^{-1025} = \cancel{1}0.\underbrace{001\quad\quad}\cdot 2^{-1022}$

-1023

Idea: Use the special ("turn off the implied one") exponent (-1023)

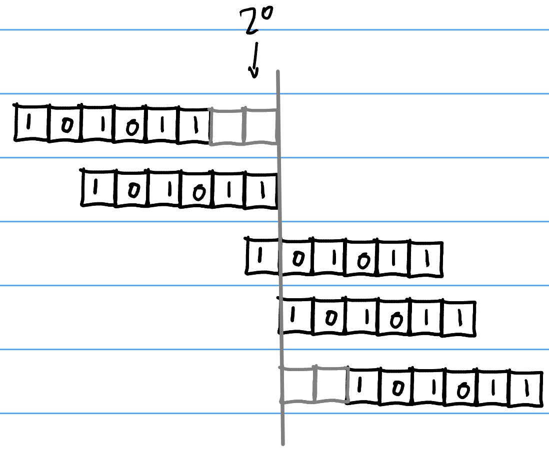
and have it *mean* -1022.

	Significand	Exponent
Intended meaning:	$0.\underbrace{001}$	-1022
Stored values:	001	-1023

A number that makes use of the special exponent value to turn off the implied one is called *denormal*.

What is the...

exponent? significand? value?



In our 64-bit example:

- 1 bit for sign (+/-)
- 11 bits for largest exponent
- 52 bits for "bits"

Exponent ranges from
-1023 to 1024

This is called "double precision".

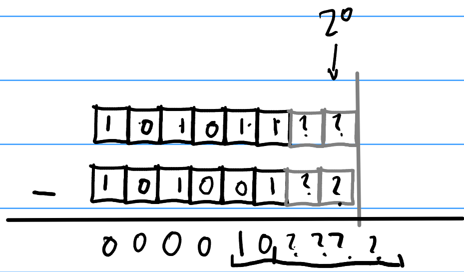
What is (very roughly) the smallest number we can represent?

What is (very roughly) the largest number we can represent?

How many accurate decimal digits do we have in the largest representable number?

How many accurate decimal digits do we have in the smallest representable number?

So what could possibly go wrong?



"Catastrophic Cancellation"

How many accurate (binary) digits are there in the above result?

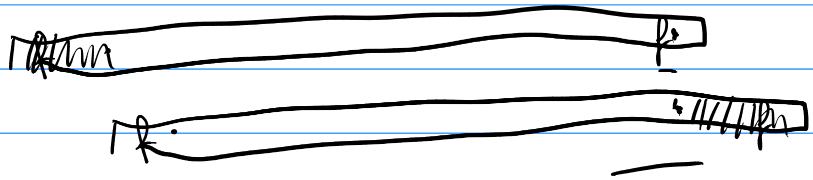
2

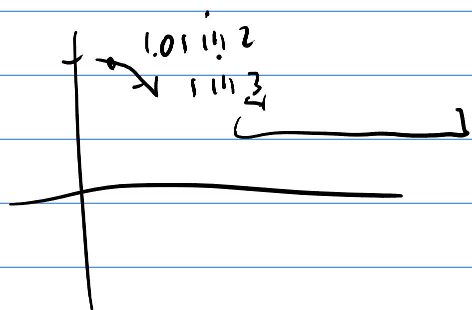
$(s_1 . s_2 s_3) \cdot 2^0$

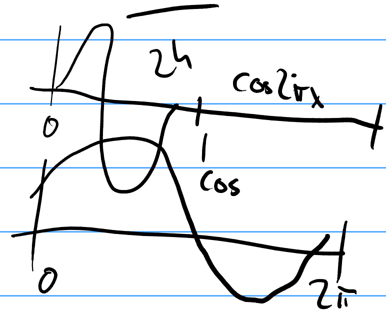
$1.00 \cdot 2^0$

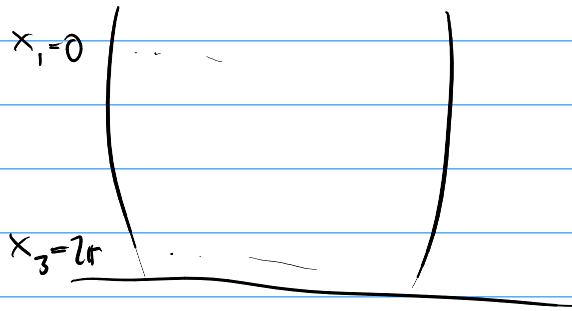
$(1.01) \cdot 2^0$

2^{-2}









$$f(x) = a\cdot\cos(0x) + b\cdot\sin(x) + c\cdot\cos(x)$$

$$f(0) = f(2\pi)$$

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} = \left(\frac{1}{2h}\right)\cdot f(x+h) + \left(\frac{1}{2h}\right) f(x-h)$$