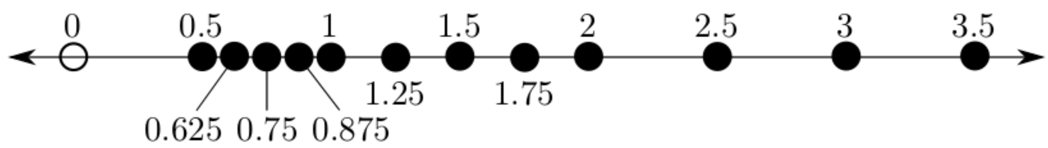
Upon paging down to section 2.2 we pass this number line...




note that the points are not equally spaced... this is typical of the way floating point or any type of scientific notation with a...

mantissa with a fixed number of digits and an exponent

and other approximations. Rounding
typically encountered in numerical sy

n is # of dec digits.

- *Rounding* or *truncation* error c
to deal with the fact that we
computational number systems
an IEEE 754 floating-point val
number of digits.

Categorize types of error...we've already discussed the first type

$2^{52} \approx 10^n$

$n \approx 52 \frac{\ln 2}{\ln 10}$

$\approx 15.6$

a little more than 15 digits of precision

```
julia> 52*log(2)/log(10)
15.653559774527022

julia> 52*log10(2)/log10(10)
15.653559774527022
```

- *Discretization* error comes fror
and other aspects of continuous
attempt to approximate the de

about 15 digits of precision in a 52-bit mantissa...

for some fixed choice of $\varepsilon > 0$. T
we will study in Chapter 14, bu
limit as $\varepsilon \to 0$, the resulting val

- *Modeling* error comes from hav
lems we wish to solve. For insta
choose to neglect the collective
displacement of air by these but
Furthermore, constants such as
be provided to the system with

Most of this course is about discretization error

- *Input* error can come from use
system (and from typos!). Simul
if" questions, in which explorat
idea of how a system behaves.

Input error can be a mistake like a typo which isn't really an error at all but simply a blunder...Input errors also include uncertainties in measured quantities...

**Definition 2.1** (Absolute error). The *absolute error* of a measurement is the difference between the approximate value and its underlying true value.

**Definition 2.2** (Relative error). The *relative error* of a measurement is the absolute error divided by the true value. # of sig digits

Absolute error: Let $x_*$ be an approximation of the exact value $x$.

$$E_{abs} = |x_* - x|$$

"signed error" $= x_* - x$, in the case $x \in \mathbb{R}$.

Relative error

$$E_{rel} = \frac{E_{abs}}{|x|}$$

Significant digits:

Suppose $x_*$ was an approx of 7 good to 4 significant digits.

$$\underbrace{7.000}_{4\text{-digits}}\boxed{1234} \approx 7 \quad \text{to 4 sig digits}$$

$$6.999\ 6843 = 7 \quad \text{to 4 sig. digits}$$

Extremes

7.0005

6.9995

$\boxed{\text{largest abs error } .0005}$

What is relative error? $\frac{.0005}{7} =$

More extremes:

largest it could have been $\frac{.0005}{1}$ (1)

smallest it could have been $\frac{.0005}{10}$ (2)

interesting..

In class it was not clear which of the above bounds should be used to define # of sig. digits... then Murphy's Law prevailed and I chose the wrong one!

0.00005

~~$x_*$ is good to 4 sig digits.~~

~~if $E_{rel} \le 0.5 \times 10^{-4}$~~

~~In general good to n sig. digits if~~

~~$\boxed{E_{rel} \le 0.5 \times 10^{n}}$~~

This part was added after the lecture...

actually, should have used bound (1) to define # of sig. digits.

Thus,

$\frac{.0005}{1}$ (1) $= 5\times 10^{-4}$ means

## Correct Definition:

$x_*$ is good to 4 sig digits.

if $E_{rel} \le 5 \times 10^{-4}$

In general good to n sig. digits if

$$\boxed{E_{rel} \le 5 \times 10^{-n}}$$

Consider:

$$f(x) \equiv \frac{e^x - 1}{x} - 1,$$

illustrate loss of precision

this example is related to approximating derivatives.

$0/0$

$$\lim_{x \to 0} f(x) = \lim_{x \to 0} \frac{e^x - 1}{x} - \lim_{x \to 1} 1$$

$$= \left( \lim_{x \to 0} \frac{e^x - 0}{1} \right) - 1 = \frac{e^0}{1} - 1 = 1 - 1 = 0$$

Let's use Julia to draw Figure 2.2 from page 34 of the text...

```
julia> f(x)=(exp(x)-1)/x-1
f (generic function with 1 method)
```

Plug in values of x close to zero

```
julia> f(0.1)
0.05170918075647724

julia> f(0.01)
0.005016708416794913

julia> f(0.0001)
5.000166714097531e-5
```

As expected, f(x) gets close to zero as x gets close to zero, so what could go wrong?

There is loss of precision in the numerator $e^x - 1$

caused by subtracting two nearly equal numbers

Loss of precision

3.24689 6 sig. digits

subtract 3.24521 6 sig digits

$$
\begin{array}{r}
3.24689 \\
-\ 3.24521 \\
\hline
0.00168
\end{array}
$$

only 3 significant digits left.

Therefore, the values of f(x) become less precise as x gets smaller

```
julia> exp(0.1)
1.1051709180756477
```

one digit cancels

```
julia> exp(0.01)
1.010050167084168
```

two digits cancel

```
julia> exp(0.0001)
1.0001000050001667
```

four digits

```
julia> exp(0.00001)
1.00001000005
```

5 digits cancel

Subtract 1 from each of these numbers... How many digits cancel.

If 5 digits cancel, then 10 are left, because the computations are done using 15-digit double precision floating point...

for smaller values of x even more digits will cancel...and eventually all of them...a catastrophy...

**Example 2.5** (Loss of precision in practice). Figure 2.2 plots the functi

$$f(x) \equiv \frac{e^x - 1}{x} - 1,$$

for evenly spaced inputs $x \in [-10^{-8}, 10^{-8}]$, computed using IEEE floa
metic. The numerator and denominator approach 0 at approximately th

```
julia> exp(1e-8)
1.00000001
```

8 digits cancel

more than half the digits are lost... answer only good to 7 sig. digits...

Thus, x=1e-8 is in the danger zone of catastrophic loss of precision due to subtraction of two nearly equal numbers.

Graph that range...

## Ranges in Julia

start increment end

```
julia> x=[1:2:12;]
6-element Vector{Int64}:
  1
  3
  5
  7
  9
 11
```

don't forget this weird semicolon

actual range we want to plot...

```
julia> x=[-1e-8:1e-10:1e-8;]
201-element Vector{Float64}:
 -1.0e-8
 -9.900000000000001e-9
 -9.8e-9
 -9.700000000000001e-9
 -9.6e-9
  ⋮
  9.700000000000001e-9
  9.800000000000002e-9
  9.9e-9
  1.0e-8
```

If you are using your own computer you may need to add the Plots package before proceeding...

This doesn't need to be done in the lab, but to load a package type the ] key and then at the package manager prompt type

skip this if in the lab

```
(@v1.6) pkg> add Plots
    Updating registry at `~/.julia/registries/General`
   Resolving package versions...
   Installed Libiconv_jll — v1.16.0+8
   Installed FFTW_jll ——— v3.3.9+8
```

Press backspace to leave the package manager when its done.

Making a plot in Julia...

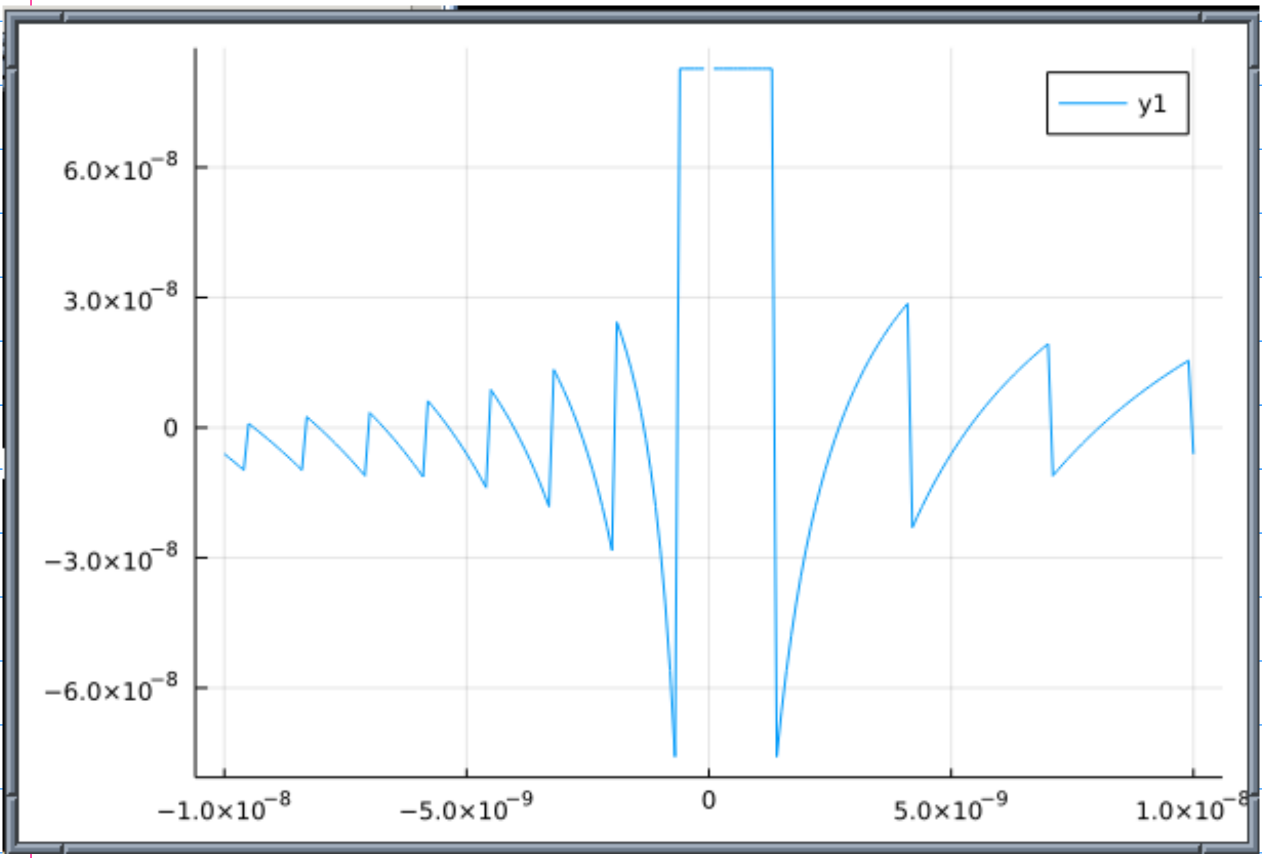
```
julia> using Plots

julia> plot(x,f.(x))
```

note $f(x)$ is a vector valued function whereas $f.(x)$ means a vector of f values...

$$f.(x) = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_{201}) \end{bmatrix}$$

The plot should look like




- note that the graph, surprisingly is not symmetric about the origin...
- If you can explain the lack of symmetry... please write this down and turn it in for extra credit...

Backwards error vs forwards error

$x = \sqrt{2}$ correct answer to the problem
find the root of $\boxed{f(x) = x^2 - 2}$.

$x_* = 1.4$

$E_{abs} = |x_* - x| = |1.4 - \sqrt{2}| \approx 0.0142$

By how much do I have to change the problem so $x_*$ is the exact solution

$$g(x) = \boxed{x^2 - (1.4)^2}$$

The difference of the two problems

$$|g(x) - f(x)| = |(1.4)^2 - 2| \approx 0.04$$

condition # in this case

$$C \approx \frac{0.0142}{0.04} = \frac{\text{forward error}}{\text{backwards error}}$$

maximize over all small errors