On Reals and Floats

The set of normalized IEEE floating point numbers is discrete and finite. It has a largest element we'll call *maxnfloat*, and a smallest (positive) element, *minnfloat*. All of the real numbers in the interval $[minnfloat,\ maxnfloat]$, together with their negatives, is called the floating point range (FPR):

$$\text{FPR} = \{x \in \mathbb{R} : minnfloat \leq |x| \leq maxnfloat\}.$$

Notice that the FPR is the (disjoint) union of two real intervals. If $|x| < minnfloat$, it is said to *underflow*, while if $|x| > maxnfloat$, it is said to *overflow*.

Each real number in FPR gets represented by ("rounded to") the floating point number nearest it. Every normalized floating point number (normal float) has the form

$$f = \pm(0.b_1 b_2 \ldots b_t)_2 \times 2^e, \text{ where}$$

$(0.b_1 b_2 \ldots b_t)_2$ is interpreted as a base-2 fraction, $b_1 \neq 0$, and e is an integer satisfying $minexp \leq e \leq maxexp$. The difference between $|f|$ and the next-larger float is 2^{e-t}, and therefore the relative difference between two neighboring floats is bounded above by 2^{1-t}. The maximum relative distance between any $x \in$ FPR and its floating point representative is called the *machine epsilon*: $\epsilon_{mach} = 2^{1-t}$. The *unit round-off* is half of this: $\boldsymbol{\mu} = \epsilon_{mach}/2 = 2^{-t}$.

There are also non-normal floats. These are floating point "numbers" which do not fit the scheme above. These include ± 0, $\pm$ Inf, NaN, and the subnormals (which are evenly spaced in the interval $(-minnfloat, minnfloat)$). Most compilers allow the programmer to use subnormals, but it's typically not the default. The common default is to set underflow to ± 0, and overflow to $\pm$ Inf, although most compilers allow the programmer to set error flags in these cases.

Arithmetic with Inf can make sense: Inf+Inf = Inf, -Inf-Inf= -Inf, PositiveNumber*Inf =Inf, NegativeNumber*Inf=-Inf, ect., but not always: NaN, which stands for "not a number", is used for undefined results, like 0/0, Inf - Inf, Inf/Inf, 0∗Inf, etc.

With an 8-byte word (double precision), we are able to represent $2^{64} \approx 1.8 \times 10^{19}$ floats. Of these, 2^{54} are non-normal, and half of these non-normals are subnormals. Thus about 99.9% of the 8-byte floats are normal. Here $maxnfloat \approx 10^{308}$, $minnfloat \approx 10^{-308}$, and $\boldsymbol{\mu} \approx 10^{-16}$.

With a 4-byte word (single precision), we have $2^{32} \approx 4.3 \times 10^{9}$ floats, about 99.2% of which are normal. Here $maxnfloat \approx 10^{38}$, $minnfloat \approx 10^{-38}$, and $\boldsymbol{\mu} \approx 10^{-7}$.

"Number" x	Classification	fl(x)	Other
minnfloat $\leq |x| \leq$ *maxnfloat*	Normal	$\pm(0.b_1 b_2 \ldots b_t)_2 \times 2^e$	$\text{fl}(x) = x(1+\delta),\ |\delta| \leq \boldsymbol{\mu}$
$|x| <$ *minnfloat*	Underflow	± 0	subnormals...
$|x| >$ *maxnfloat*	Overflow	$\pm$ Inf	error flags...
0/0, 0*Inf, Inf - Inf, etc.	Undefined	NaN	error flags...