II.2

NICE NUMBERS FOR GRAPH LABELS

Paul S. Heckbert
University of California
Berkeley, California

When creating a graph by computer, it is desirable to label the x and y axes with "nice" numbers: simple decimal numbers. For example, if the data range is 105 to 543, we'd probably want to plot the range from 100 to 600 and put tick marks every 100 units (see Fig. 1). Or if the data range is 2.03 to 2.17, we'd probably plot a range from 2.00 to 2.20 with a tick spacing of .05. Humans are good at choosing such "nice" numbers, but simplistic algorithms are not. The naive label-selection algorithm takes the data range and divides it into n equal intervals, but this usually results in ugly tick labels. We here describe a simple method for generating nice graph labels.

The primary observation is that the "nicest" numbers in decimal are 1, 2, and 5, and all power-of-ten multiples of these numbers. We will use only such numbers for the tick spacing, and place tick marks at multiples of the tick spacing. We choose the minimum and maximum of the graphed range in either of two ways: (a) loose: round the data minimum

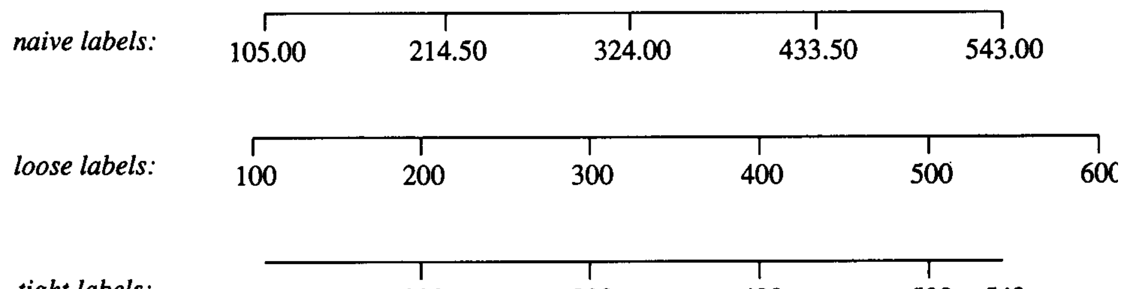

Figure 1.

down, and the data maximum up, to compute the graph minimum and maximum, respectively, or (b) tight: use the data minimum and maximum for the graph minimum and maximum. The relative merits of these two approaches are discussed in Tufte (1983). Below is some pseudo-code for the loose method:

```
const ntick ← 5;                          desired number of tick marks

loose_label: label the data range from min to max loosely.
(tight method is similar )

procedure loose_label(min, max: real);

nfrac: int;
d: real;                                  tick mark spacing
graphmin, graphmax: real;                 graph range min and max
range, x: real;
begin

    range ← nicenum(max – min, false);
    d ← nicenum(range/(ntick – 1), true);
    graphmin ← floor(min/d)*d;
    graphmax ← ceiling(max/d)*d;
    nfrac ← max( – floor(log10(d)), 0);   number of fractional digits to show

    for x ← graphmin to graphmax + .5*d step d do
        put tick mark at x, with a numerical label showing nfrac fraction digits
        endloop;
    endproc loose_label;

nicenum: find a "nice" number approximately equal to x.
Round the number if round = true, take ceiling if round = false.

function nicenum(x: real; round: boolean): real;
exp: int;                                 exponent of x
f: real;                                  fractional part of x
nf: real;                                 nice, rounded fraction
begin
    exp ← floor(log10(x));
    f ← x/expt(10., exp);                 between 1 and 10
```

```
if round then
    if f < 1.5 then nf ← 1.;
    else if f < 3. then nf ← 2.;
    else if f < 7. then nf ← 5.;
    else nf ← 10.;
else
    if f ≤ 1. then nf ← 1.;
    else if f ≤ 2. then nf ← 2.;
    else if f ≤ 5. then nf ← 5.;
    else nf ← 10.;
return nf*expt(10., exp);
endfunc nicenum;
```

We assume in the above that logl0(z) is log base 10 of z.

We also assume expt(a, n) = a^n for integer n. But note that the exponentiation routines in some math libraries are inexact for integer arguments, and such errors can cause the above code to fail. On early UNIX systems I found that pow(10.,2.) ≠ 100 exactly, so I wrote my own expt function by multiplying or dividing in a loop. The pow routine in current (BSD 4.3) UNIX is trustworthy, however.

See Appendix 2 for C Implementation (657)