The trig package*

David Carlisle

2021/08/11

1 Introduction

These macros implement the trigonometric functions, sin, cos and tan. In each case two commands are defined. For instance the command `\CalculateSin{33}` may be issued at some point, and then anywhere later in the document, the command `\UseSin{33}` will return the decimal expansion of $\sin(33°)$.

The arguments to these macros do not have to be whole numbers, although in the case of whole numbers, LaTeX or plain TeX counters may be used. In TeXBook syntax, arguments must be of type: ⟨*optional signs*⟩⟨*factor*⟩

Some other examples are:
`\CalculateSin{22.5}`, `\UseTan{\value{mycounter}}`, `\UseCos{\count@}`.

Note that unlike the psfig macros, these save all previously computed values. This could easily be changed, but I thought that in many applications one would want many instances of the same value. (eg rotating all the headings of a table by the *same* amount).

I don't really like this need to pre-calculate the values, I originally implemented `\UseSin` so that it automatically calculated the value if it was not pre-stored. This worked fine in testing, until I remembered why one needs these values. You want to be able to say `\dimen2=\UseSin{30}\dimen0`. Which means that `\UseSin` must *expand* to a ⟨*factor*⟩.

2 The Macros

```
⟨*package⟩
```

`\nin@ty` `\@clxxx` `\@lxxi` `\@mmmmlxviii` Some useful constants for converting between degrees and radians.

$$\frac{\pi}{180} \simeq \frac{355}{113 \times 180} = \frac{71}{4068}$$

```
\chardef\nin@ty=90
\chardef\@clxxx=180
```

*This file has version number v1.11, last revised 2021/08/11.

```
\chardef\@lxxi=71
\mathchardef\@mmmmlxviii=4068
```

The approximation to sin. I experimented with various approximations based on Tchebicheff polynomials, and also some approximations from a SIAM handbook 'Computer Approximations' However the standard Taylor series seems sufficiently accurate, and used by far the fewest TEX tokens, as the coefficients are all rational.

$$\begin{aligned}\sin(x) &\simeq x-(1/3!)x^3+(1/5!)x^5-(1/7!)x^7+(1/9!)x^9\\ &\simeq \frac{((((7!/9!x^2-7!/7!)x^2+7!/5!)x^2+7!/3!)x^2+7!/1!)x}{7!}\\ &= \frac{((((1/72x^2-1)x^2+42)x^2+840)x^2+5040)x}{5040}\end{aligned}$$

The nested form used above reduces the number of operations required. In order to further reduce the number of operations, and more importantly reduce the number of tokens used, we can precompute the coefficients. Note that we can not use 9! as the denominator as this would cause overflow of TEX's arithmetic.

`\@coeffz` `\@coeffa` `\@coeffb` `\@coeffc` `\@coeffd` Save the coefficients as `\`(`math`)`chars`.

```
\chardef\@coeffz=72
%\chardef\@coefa=1
\chardef\@coefb=42
\mathchardef\@coefc=840
\mathchardef\@coefd=5040
```

`\TG@rem@pt` The standard trick of getting a real number out of a ⟨*dimen*⟩. This gives a maximum accuracy of approx. 5 decimal places, which should be sufficient. It puts a space after the number, perhaps it shouldn't.

```
{\catcode`t=12\catcode`p=12\gdef\noPT#1pt{#1}}
\def\TG@rem@pt#1{\expandafter\noPT\the#1\space}
```

`\TG@term` Compute one term of the above nested series. Multiply the previous sum by x^2 (stored in `\@tempb`, then add the next coefficient, `#1`.

```
\def\TG@term#1{%
  \dimen@\@tempb\dimen@
  \advance\dimen@ #1\p@}
```

`\TG@series` Compute the above series. the value in degrees will be in `\dimen@` before this is called.

```
\def\TG@series{%
  \dimen@\@lxxi\dimen@
  \divide \dimen@ \@mmmmlxviii
```

`\dimen@` now contains the angle in radians, as a ⟨*dimen*⟩. We need to remove the units, so store the same value as a ⟨*factor*⟩ in `\@tempa`.

```
  \edef\@tempa{\TG@rem@pt\dimen@}%
```

Now put x^2 in `\dimen@` and `\@tempb`.

```
  \dimen@\@tempa\dimen@
  \edef\@tempb{\TG@rem@pt\dimen@}%
```

The first coefficient is 1/72.

```
\divide\dimen@\@coeffz
\advance\dimen@\m@ne\p@
\TG@term\@coefb
\TG@term{-\@coefc}%
\TG@term\@coefd
```

Now the cubic in x^2 is completed, so we need to multiply by x and divide by 7!.

```
\dimen@\@tempa\dimen@
\divide\dimen@ \@coefd}
```

`\CalculateSin` If this angle has already been computed, do nothing, else store the angle, and call `\TG@@sin`.

```
\def\CalculateSin#1{{%
  \expandafter\ifx\csname sin(\number#1)\endcsname\relax
    \dimen@=#1\p@\TG@@sin
    \expandafter\xdef\csname sin(\number#1)\endcsname
                                        {\TG@rem@pt\dimen@}%
  \fi}}
```

`\CalculateCos` As above, but use the relation $\cos(x) = \sin(90 - x)$.

```
\def\CalculateCos#1{{%
  \expandafter\ifx\csname cos(\number#1)\endcsname\relax
    \dimen@=\nin@ty\p@
    \advance\dimen@-#1\p@
    \TG@@sin
    \expandafter\xdef\csname cos(\number#1)\endcsname
                                         {\TG@rem@pt\dimen@}%
  \fi}}
```

`\TG@reduce` Repeatedly use one of the the relations $\sin(x) = \sin(180 - x) = \sin(-180 - x)$ to get x in the range $-90 \leq x \leq 90$. Then call `\TG@series`.

```
\def\TG@reduce#1#2{%
\dimen@#1#2\nin@ty\p@
  \advance\dimen@#2-\@clxxx\p@
  \dimen@-\dimen@
  \TG@@sin}
```

`\TG@@sin` Slightly cryptic, but it seems to work...

```
\def\TG@@sin{%
  \ifdim\TG@reduce>+%
  \else\ifdim\TG@reduce<-%
  \else\TG@series\fi\fi}%
```

`\UseSin` `\UseCos` Use a pre-computed value.

```
\def\UseSin#1{\csname sin(\number#1)\endcsname}
\def\UseCos#1{\csname cos(\number#1)\endcsname}
```

A few shortcuts to save space.

```
\def\z@num{0 }
\def\@tempa{1 }
\def\@tempb{-1 }
```

```
\expandafter\let\csname sin(0)\endcsname\z@num
\expandafter\let\csname cos(0)\endcsname\@tempa
\expandafter\let\csname sin(90)\endcsname\@tempa
\expandafter\let\csname cos(90)\endcsname\z@num
\expandafter\let\csname sin(-90)\endcsname\@tempb
\expandafter\let\csname cos(-90)\endcsname\z@num
\expandafter\let\csname sin(180)\endcsname\z@num
\expandafter\let\csname cos(180)\endcsname\@tempb
```

A few more added in 1.10 (previously in pdftex.def)

```
\expandafter\let\csname sin(270)\endcsname\@tempb
\expandafter\let\csname cos(270)\endcsname\z@num
\expandafter\let\csname sin(360)\endcsname\z@num
\expandafter\let\csname cos(360)\endcsname\@tempa
\expandafter\let\csname sin(-180)\endcsname\z@num
\expandafter\let\csname cos(-180)\endcsname\@tempb
\expandafter\let\csname sin(-270)\endcsname\@tempa
\expandafter\let\csname cos(-270)\endcsname\z@num
\expandafter\let\csname sin(-360)\endcsname\z@num
\expandafter\let\csname cos(-360)\endcsname\@tempa
```

`\CalculateTan` Originally I coded the Taylor series for tan, but it seems to be more accurate to just take the ratio of the sine and cosine. This is accurate to 4 decimal places for angles up to 50°, after that the accuracy tails off, giving 57.47894 instead of 57.2900 for 89°.

```
\def\CalculateTan#1{{%
  \expandafter\ifx\csname tan(\number#1)\endcsname\relax
    \CalculateSin{#1}%
    \CalculateCos{#1}%
    \@tempdima\UseCos{#1}\p@
    \divide\@tempdima\@iv
    \@tempdimb\UseSin{#1}\p@
    \@tempdimb\two@fourteen\@tempdimb
    \divide\@tempdimb\@tempdima
    \expandafter\xdef\csname tan(\number#1)\endcsname
                                      {\TG@rem@pt\@tempdimb}%
  \fi}}
```

`\UseTan` Just like `\UseSin`.

```
\def\UseTan#1{\csname tan(\number#1)\endcsname}
```

`\two@fourteen` `\@iv` two constants needed to keep the division within TeX's range.

```
\mathchardef\two@fourteen=16384
\chardef\@iv=4
```

Predefine tan($\pm$90) to be an error.

```
\expandafter\def\csname tan(90)\endcsname{\errmessage{Infinite tan !}}
\expandafter\let\csname tan(-90)\expandafter\endcsname
                                       \csname tan(90)\endcsname
⟨/package⟩
```