## The easing Library for pgfmath

Loh Ka-tsun

July 14, 2021

## 1 Usage

## 2 Implementation

\ifeasing@withfpu \easing@divide This library uses  $T_EX$  registers and pgf's mathematical engine for computations.

It is possible that the user is loading this library together with fpu. We save the basic routines from pgfmath so that when this happens, fpu doesn't break everything when it does a switcharoo with the pgfmath macros.

- 1 \newif\ifeasing@withfpu
- 2 \expandafter\ifx\csname pgflibraryfpuifactive\endcsname\relax
- 3 \easing@withfpufalse
- $4 \ensuremath{\setminus} else$
- $5 \geq 5$
- 6 \fi
- 7 \ifeasing@withfpu
- 8 \let\easing@divide\pgfmath@basic@divide@
- 9 \let\easing@cos\pgfmath@basic@cos@
- 10 \let\easing@pow\pgfmath@basic@pow@
- 11 \else
- 12 \let\easing@divide\pgfmathdivide@
- 13 \let\easing@cos\pgfmathcos@
- 14 \let\easing@pow\pgfmathpow@
- 15 \fi

\easing@linearstep@ne \easing@linearstep@fixed \easing@linearstep@float \easing@linearstep In absence of fpu, the next section of code defines \easing@linearstep, which expects as arguments plain numbers (i.e. things that can be assigned to dimension registers). The net effect of \easing@linearstep{#1}{#2}{#3} is to set \pgfmathresult to  $\frac{\#3-\#1}{\#2-\#1}$ , clamped to between 0 and 1.

If fpu is loaded, \easing@linearstep@sinstead named \easing@linearstep@fixed, and we additionally define \easing@linearstep@float, which expects fpuformat floats as arguments. We do not format the output as a float since fpu is smart enough to do that conversion quietly on its own.

The \easing@linearstep routine is the first step in the definition of all other routines that compute easing functions.

```
16 \def\easing@linearstep@ne#1{%
17
    \begingroup
    \pgf@x#1pt
18
   \ifdim1pt<\pgf@x\pgf@x 1pt\fi
19
20
   \ifdimOpt>\pgf@x\pgf@x Opt\fi
    \pgfmathreturn\pgf@x
22
   \endgroup
23 }%
24 \expandafter\def
\begingroup
27
    \pgf@xa#3pt
28
    \pgf@xb#2pt
    \pgf@xc#1pt
29
30
    \ifdim\pgf@xb=\pgf@xc
    \edef\pgfmathresult{\ifdim\pgf@xa>\pgf@xb 1\else 0\fi}%
31
    \else
32
33
    \advance\pgf@xa-\pgf@xc
    \advance\pgf@xb-\pgf@xc
35
    \easing@divide{\pgfmath@tonumber\pgf@xa}{\pgfmath@tonumber\pgf@xb}%
    \easing@linearstep@ne\pgfmathresult
36
37
    \pgfmathsmuggle\pgfmathresult
38
39
    \endgroup
40 }%
41 \ifeasing@withfpu
42 \def\easing@linearstep@float#1#2#3{%
43
    \begingroup
    \pgfmathfloatsubtract{#3}{#1}%
44
    \edef\pgf@tempa{\pgfmathresult}%
45
46
    \pgfmathfloatsubtract{#2}{#1}%
47
    \edef\pgf@tempb{\pgfmathresult}%
    \pgfmathfloatifflags{\pgf@tempb}{0}{%
48
      \pgfmathfloatifflags{\pgf@tempa}{-}{%
49
        \edef\pgfmathresult{0}%
50
      }{%
51
        \edef\pgfmathresult{1}%
52
      }%
53
    }{%
54
55
      \pgfmathfloatdivide\pgf@tempa\pgf@tempb
      \pgfmathfloattofixed{\pgfmathresult}%
56
      \easing@linearstep@ne\pgfmathresult
57
58
    \pgfmathsmuggle\pgfmathresult
59
60
    \endgroup
61 }%
62 \def\easing@linearstep#1#2#3{%
```

```
63 \pgflibraryfpuifactive{%
64 \easing@linearstep@float{#1}{#2}{#3}}{%
65 \easing@linearstep@fixed{#1}{#2}{#3}}%
66 }%
67 \fi
```

\easing@linearstep@easein@ne \easing@linearstep@easeout@ne The linear ease-in and ease-out functions are identitical to the linear step function. We define the respective macros so as not to surprise the user with their absence.

```
68 \let\easing@lineareasein\easing@linearstep
69 \pgfmathdeclarefunction{lineareasein}{3}{%
70 \easing@lineareasein{#1}{#2}{#3}}%
71 \let\easing@lineareaseout\easing@linearstep
72 \pgfmathdeclarefunction{lineareaseout}{3}{%
73 \easing@lineareasein{#1}{#2}{#3}}%
```

the right to make space in the margins:

\easing@derive@easein@nefromstep@ne \easing@derive@easeout@nefromstep@ne \easing@derive@step@nefromeasein@ne \easing@derive@easeout@nefromeasein@ne The pattern in general is that, for each shape, we define the one-parameter version of the step, ease-in, and ease-out routines interpolating between values 0 at 1 at the ends of the unit interval. Then by composing with \easing@linearstep, we obtain the three-parameter versions that allow the user to specify the begin and end points of the interpolation.

Most of the time it suffices to define just one of the three one-parameter versions of a shape to be able to infer the form of all three. This is done with the \easing@derive-from- macros.

```
74 \def\easing@derive@easein@nefromstep@ne#1{%
    \expandafter\def\csname easing@#1easein@ne\endcsname##1{%
75
      \begingroup
76
77
      \pgf@x##1 pt
      \divide\pgf@x 2
78
79
      \csname easing@#1step@ne\endcsname{\pgfmath@tonumber\pgf@x}%
80
      \pgf@x\pgfmathresult pt
      \multiply\pgf@x 2
81
      \pgfmathreturn\pgf@x
82
      \endgroup
83
84
    }%
85 }%
86 \def\easing@derive@easeout@nefromstep@ne#1{%
    \expandafter\def\csname easing@#1easeout@ne\endcsname##1{%
87
      \begingroup
88
      \pgf@x##1 pt
89
      \divide\pgf@x 2
90
      \advance\pgf@x 0.5pt
91
92
      \csname easing@#1step@ne\endcsname{\pgfmath@tonumber\pgf@x}%
93
      \pgf@x\pgfmathresult pt
      \multiply\pgf@x 2
94
```

```
\advance\pgf@x -1pt
 95
        \pgfmathreturn\pgf@x
 96
       \endgroup
 97
     }%
 98
99 }%
100 \def\easing@derive@step@nefromeasein@ne#1{%
101
     \expandafter\def\csname easing@#1step@ne\endcsname##1{%
102
     \begingroup
        \pgf@x##1 pt
103
        \multiply\pgf@x 2
104
        \ifdim\pgf@x<1pt
105
        \csname easing@#1easein@ne\endcsname{\pgfmath@tonumber\pgf@x}%
106
107
        \pgf@x\pgfmathresult pt
        \divide\pgf@x 2
108
       \else
109
       \multiply\pgf@x -1
110
       \advance\pgf@x 2pt
111
       \csname easing@#1easein@ne\endcsname{\pgfmath@tonumber\pgf@x}%
112
113
       \pgf@x\pgfmathresult pt
114
        \divide\pgf@x 2
        \multiply\pgf@x -1
115
        \advance\pgf@x 1pt
116
117
        \pgfmathreturn\pgf@x
118
119
        \endgroup
     }%
120
121 }%
122 \def\easing@derive@easeout@nefromeasein@ne#1{%
     \expandafter\def\csname easing@#1easeout@ne\endcsname##1{%
123
       \begingroup
124
       \pgf@x##1pt
125
126
       \multiply\pgf@x -1
127
       \advance\pgf@x 1pt
       \csname easing@#1easein@ne\endcsname{\pgfmath@tonumber\pgf@x}%
128
        \pgf@x\pgfmathresult pt
129
        \multiply\pgf@x -1
130
       \advance\pgf@x 1pt
131
132
        \pgfmathreturn\pgf@x
133
        \endgroup
     }%
134
135 }
```

\easing@pgfmathinstall The three-parameter versions of each routine is installed into the mathematical engine, so that they are available in \pgfmathparse.

```
136 \def\easing@pgfmathinstall#1{%
137 \pgfmathdeclarefunction{#1step}{3}{%
138 \easing@linearstep{##1}{##2}{##3}%
139 \csname easing@#1step@ne\endcsname\pgfmathresult
140 }%
```

```
\pgfmathdeclarefunction{#1easein}{3}{%
141
       \easing@linearstep{##1}{##2}{##3}%
142
       \csname easing@#1easein@ne\endcsname\pgfmathresult
143
144
     \pgfmathdeclarefunction{#1easeout}{3}{%
145
146
       \easing@linearstep{##1}{##2}{##3}%
147
       \csname easing@#1easeout@ne\endcsname\pgfmathresult
    }%
148
149 }%
The smooth shape.
150 \def\easing@smoothstep@ne#1{%
```

\easing@smoothstep@ne \easing@smootheasein@ne \easing@smootheaseout@ne

- \begingroup 151
- \pgf@x#1pt 152
- \edef\pgf@temp{\pgfmath@tonumber\pgf@x}% 153
- \multiply\pgf@x-2 154
- \advance\pgf@x 3pt 155
- \pgf@x\pgf@temp\pgf@x 156
- \pgf@x\pgf@temp\pgf@x 157
- \pgfmathreturn\pgf@x 158
- 159\endgroup
- 160 }%
- 161 \easing@derive@easein@nefromstep@ne{smooth}%
- 162 \easing@derive@easeout@nefromstep@ne{smooth}%
- 163 \easing@pgfmathinstall{smooth}%

\easing@sinestep@ne \easing@sineeasein@ne \easing@sineeaseout@ne The sine shape.

We write down both the easein and step forms of this, since they are simple compared to what would have been obtained by \easing@derive-.

```
164 \def\easing@sineeasein@ne#1{%
```

- 165 \begingroup
- 166 \pgf@x#1pt
- \multiply\pgf@x 90 167
- \easing@cos{\pgfmath@tonumber\pgf@x}% 168
- \pgf@x\pgfmathresult pt 169
- \multiply\pgf@x -1 170
- 171 \advance\pgf@x 1pt
- 172 \pgfmathreturn\pgf@x
- 173 \endgroup
- 174 }%
- 175 \def\easing@sinestep@ne#1{%
- 176 \begingroup
- 177 \pgf@x#1pt
- \multiply\pgf@x 180 178
- \easing@cos{\pgfmath@tonumber\pgf@x}% 179
- \pgf@x\pgfmathresult pt
- 181 \divide\pgf@x 2

```
182 \multiply\pgf@x -1
183 \advance\pgf@x 0.5pt
184 \pgfmathreturn\pgf@x
185 \endgroup
186 }%
187 \easing@derive@easeout@nefromeasein@ne{sine}%
188 \easing@pgfmathinstall{sine}%
```

\easing@powstep@ne \easing@poweasein@ne \easing@poweaseout@ne The pow shape.

Because of some wonkiness in in fpu, instead of invoking the pow function from pgfmath, we compute  $t^n$  approximately by computing  $e^{n \ln t}$  using ln and exp instead (which is what pgfmath does anyway when the exponent is not an integer.)

```
189 \pgfkeys{/easing/.is family}%
190 \pgfkeys{easing,
     pow/exponent/.estore in=\easing@param@pow@exponent,
     pow/exponent/.default=2.4,
     pow/exponent}%
193
194 \def\easing@poweasein@ne#1{%
     \begingroup
195
196
     \pgf@x#1pt
     \ifdim\pgf@x=0pt
197
     \edef\pgfmathresult{0}%
198
199
200
     \pgfmath@basic@ln@{#1}%
     \pgf@x\pgfmathresult pt
201
     \pgf@x\easing@param@pow@exponent\pgf@x
202
     \pgfmath@basic@exp@{\pgfmath@tonumber\pgf@x}%
203
204
205
     \pgfmathsmuggle\pgfmathresult
     \endgroup
206
207 }%
208 \easing@derive@easeout@nefromeasein@ne{pow}%
209 \easing@derive@step@nefromeasein@ne{pow}%
210 \easing@pgfmathinstall{pow}%
```

\easing@quadstep@ne The
\easing@quadeasein@ne smal
\easing@quadeaseout@ne faste
\easing@cubiceasein@ne
\easing@cubiceaseout@ne
\easing@quartstep@ne
\easing@quarteasein@ne
\easing@quarteaseout@ne
\easing@quarteaseout@ne
\easing@quintstep@ne
\easing@quintstep@ne
\easing@quinteasein@ne
211 \d
212
213
214
215
216
easing@quinteasein@ne
217
\easing@quinteaseout@ne
218 }%

The quad—, cubic—, quart—, and quint— routines have explicit definitions. The small integer exponents are computed with TEX registers, which is probably a little faster and more accurate than setting the argument then evaluating the equivalent pow— routine.

```
211 \def\easing@quadeasein@ne#1{%
212 \begingroup
213 \pgf@x#1pt
214 \edef\pgf@temp{\pgfmath@tonumber\pgf@x}%
215 \pgf@x\pgf@temp\pgf@x
216 \pgfmathreturn\pgf@x
217 \endgroup
```

```
223 \def\easing@cubiceasein@ne#1{%
                        224
                             \begingroup
                        225
                             \pgf@x#1pt
                             \edef\pgf@temp{\pgfmath@tonumber\pgf@x}%
                        226
                             \pgf@x\pgf@temp\pgf@x
                        227
                             \pgf@x\pgf@temp\pgf@x
                        228
                             \pgfmathreturn\pgf@x
                        229
                        230
                             \endgroup
                        231 }%
                        232 \easing@derive@step@nefromeasein@ne{cubic}%
                        233 \easing@derive@easeout@nefromeasein@ne{cubic}%
                        234 \verb|\easing@pgfmathinstall{cubic}| \%
                        235
                        236 \def\easing@quarteasein@ne#1{%
                        237
                             \begingroup
                        238
                             \pgf@x#1pt
                             \edef\pgf@temp{\pgfmath@tonumber\pgf@x}%
                        239
                             \pgf@x\pgf@temp\pgf@x
                        240
                             \pgf@x\pgf@temp\pgf@x
                        241
                             \pgf@x\pgf@temp\pgf@x
                        242
                        243
                             \pgfmathreturn\pgf@x
                        244
                             \endgroup
                        245 }%
                        246 \easing@derive@step@nefromeasein@ne{quart}%
                        247 \easing@derive@easeout@nefromeasein@ne{quart}%
                        248 \easing@pgfmathinstall{quart}%
                        249
                        250 \def\easing@quinteasein@ne#1{%
                        251
                             \begingroup
                        252
                             \pgf@x#1pt
                             \edef\pgf@temp{\pgfmath@tonumber\pgf@x}%
                        253
                             \pgf@x\pgf@temp\pgf@x
                        254
                             \pgf@x\pgf@temp\pgf@x
                        255
                        256
                             \pgf@x\pgf@temp\pgf@x
                        257
                              \pgf@x\pgf@temp\pgf@x
                             \pgfmathreturn\pgf@x
                        258
                        259
                             \endgroup
                        260 }%
                        261 \verb|\easing@derive@step@nefromeasein@ne{quint}|| \%
                        262 \easing@derive@easeout@nefromeasein@ne{quint}%
                        263 \easing@pgfmathinstall{quint}%
                         The back shape.
   \easing@backstep@ne
 \easing@backeasein@ne
                        264 \pgfkeys{easing,
\easing@backeaseout@ne
                        265 back/overshoot/.estore in=\easing@param@back@overshoot,
```

219 \easing@derive@step@nefromeasein@ne{quad}% 220 \easing@derive@easeout@nefromeasein@ne{quad}%

221 \easing@pgfmathinstall{quad}%

222

```
back/overshoot/.default=1.6,
266
                                back/overshoot}%
267
268 \ensuremath{\mbox{\sc def}\mbox{\sc easein@ne\#1}}\%
                                \begingroup
269
                                \protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\protect\pro
270
                                271
272
                                \advance\pgf@x -1pt
273
                                \verb|\pgf@x| easing@param@back@overshoot|pgf@x|
                                \advance\pgf@x\pgf@temp pt
274
275
                                \verb|\pgf@x\pgf@temp\pgf@x|
                                \pgf@x\pgf@temp\pgf@x
276
                                 \verb|\pgfmathreturn\pgf@x|
277
278
                                \endgroup
279 }%
280 \verb|\easing@derive@step@nefromeasein@ne{back}|| % \\
```

 $281 \verb|\easing@derive@easeout@nefromeasein@ne{back}|| \%$ 

 $282 \verb|\easing@pgfmathinstall{back}| %$