Emacs TeQ: (TEX + Quail)

Input Method written in Quail for entering LATEX math expressions

Garid Zorigoo

April 10, 2023

Contents

1 Alphabet related stuff :

1.1 Greek

Table 1: Main Greek letters

key	sym	latex (lower greek)	key	sym	latex (upper greek)
a.	α	\alpha	A.	A	A
b.	β	\beta	B.	B	B
c.	ψ	\psi	C.	Ψ	\Psi
d.	δ	\delta	D.	Δ	\Delta
e.	ϵ	\epsilon	E.	E	E
f.	ϕ	\phi	F.	Φ	\Phi
g.	γ	\gamma	G.	Γ	\Gamma
h.	η	\eta	H.	H	H
i.	ι	\iota	I.	I	I
j.	ξ	\xi	J.	Ξ	\Xi
k.	κ	\kappa	K.	K	K
l.	λ	\lambda	L.	Λ	\Lambda
m.	μ	\mu	M.	M	M
n.	ν	\nu	N.	N	N
o.	o	o	O.	O	O
p.	π	\pi	P.	Π	\Pi
r.	ρ	\rho	R.	P	P
s.	σ	\sigma	S.	Σ	\Sigma
t.	τ	\tau	T.	T	T
th.	θ	\theta	Th.	Θ	\Theta
u.	υ	\upsilon	U.	Υ	\Upsilon
w.	ω	\omega	W.	Ω	\Omega
x.	χ	\chi	X.	X	X
z.	ζ	\zeta	Z.	Z	Z

Table 2: Variation Greek letters

key	sym	latex (lower greek)
`e..`	ε	`\varepsilon`
`f..`	φ	`\varphi`
`s..`	ς	`\varsigma`
`t..`	ϑ	`\vartheta`
`r..`	ϱ	`\varrho`
`p..`	ϖ	`\varpi`
`k..`	$\varkappa$	`\varkappa`

1.2 Matrix (aka bold)

Table 3: Matrix

key	sym	latex (upper bold)	key	sym	latex (lower bold)
Am	$\mathbf{A}$	\mathbf{A}	am	$\mathbf{a}$	\mathbf{a}
Bm	$\mathbf{B}$	\mathbf{B}	bm	$\mathbf{b}$	\mathbf{b}
Cm	$\mathbf{C}$	\mathbf{C}	cm	$\mathbf{c}$	\mathbf{c}
Dm	$\mathbf{D}$	\mathbf{D}	dm	$\mathbf{d}$	\mathbf{d}
Em	$\mathbf{E}$	\mathbf{E}	em	$\mathbf{e}$	\mathbf{e}
Fm	$\mathbf{F}$	\mathbf{F}	fm	$\mathbf{f}$	\mathbf{f}
Gm	$\mathbf{G}$	\mathbf{G}	gm	$\mathbf{g}$	\mathbf{g}
Hm	$\mathbf{H}$	\mathbf{H}	hm	$\mathbf{h}$	\mathbf{h}
Im	$\mathbf{I}$	\mathbf{I}	im	$\mathbf{i}$	\mathbf{i}
Jm	$\mathbf{J}$	\mathbf{J}	jm	$\mathbf{j}$	\mathbf{j}
Km	$\mathbf{K}$	\mathbf{K}	km	$\mathbf{k}$	\mathbf{k}
Lm	$\mathbf{L}$	\mathbf{L}	lm	$\mathbf{l}$	\mathbf{l}
Mm	$\mathbf{M}$	\mathbf{M}	mm	$\mathbf{m}$	\mathbf{m}
Nm	$\mathbf{N}$	\mathbf{N}	nm	$\mathbf{n}$	\mathbf{n}
Om	$\mathbf{O}$	\mathbf{O}	om	$\mathbf{o}$	\mathbf{o}
Pm	$\mathbf{P}$	\mathbf{P}	pm	$\mathbf{p}$	\mathbf{p}
Qm	$\mathbf{Q}$	\mathbf{Q}	qm	$\mathbf{q}$	\mathbf{q}
Rm	$\mathbf{R}$	\mathbf{R}	rm	$\mathbf{r}$	\mathbf{r}
Sm	$\mathbf{S}$	\mathbf{S}	sm	$\mathbf{s}$	\mathbf{s}
Tm	$\mathbf{T}$	\mathbf{T}	tm	$\mathbf{t}$	\mathbf{t}
Um	$\mathbf{U}$	\mathbf{U}	um	$\mathbf{u}$	\mathbf{u}
Vm	$\mathbf{V}$	\mathbf{V}	vm	$\mathbf{v}$	\mathbf{v}
Wm	$\mathbf{W}$	\mathbf{W}	wm	$\mathbf{w}$	\mathbf{w}
Xm	$\mathbf{X}$	\mathbf{X}	xm	$\mathbf{x}$	\mathbf{x}
Ym	$\mathbf{Y}$	\mathbf{Y}	ym	$\mathbf{y}$	\mathbf{y}
Zm	$\mathbf{Z}$	\mathbf{Z}	zm	$\mathbf{z}$	\mathbf{z}
0m	$\mathbf{0}$	\mathbf{0}	0m	$\mathbf{0}$	\mathbf{0}

1.3 Vector & Hat

Table 4: Vectors and Hats

key	sym	latex (vec)	key	sym	latex (hat)
av	$\vec{a}$	\vec{a}	ah	$\hat{a}$	\hat{a}
bv	$\vec{b}$	\vec{b}	bh	$\hat{b}$	\hat{b}
cv	$\vec{c}$	\vec{c}	ch	$\hat{c}$	\hat{c}
dv	$\vec{d}$	\vec{d}	dh	$\hat{d}$	\hat{d}
ev	$\vec{e}$	\vec{e}	eh	$\hat{e}$	\hat{e}
fv	$\vec{f}$	\vec{f}	fh	$\hat{f}$	\hat{f}
gv	$\vec{g}$	\vec{g}	gh	$\hat{g}$	\hat{g}
hv	$\vec{h}$	\vec{h}	hh	$\hat{h}$	\hat{h}
iv	$\vec{i}$	\vec{i}	ih	$\hat{i}$	\hat{i}
jv	$\vec{j}$	\vec{j}	jh	$\hat{j}$	\hat{j}
kv	$\vec{k}$	\vec{k}	kh	$\hat{k}$	\hat{k}
lv	$\vec{l}$	\vec{l}	lh	$\hat{l}$	\hat{l}
mv	$\vec{m}$	\vec{m}	mh	$\hat{m}$	\hat{m}
nv	$\vec{n}$	\vec{n}	nh	$\hat{n}$	\hat{n}
ov	$\vec{o}$	\vec{o}	oh	$\hat{o}$	\hat{o}
pv	$\vec{p}$	\vec{p}	ph	$\hat{p}$	\hat{p}
qv	$\vec{q}$	\vec{q}	qh	$\hat{q}$	\hat{q}
rv	$\vec{r}$	\vec{r}	rh	$\hat{r}$	\hat{r}
sv	$\vec{s}$	\vec{s}	sh	$\hat{s}$	\hat{s}
tv	$\vec{t}$	\vec{t}	th	$\hat{t}$	\hat{t}
uv	$\vec{u}$	\vec{u}	uh	$\hat{u}$	\hat{u}
vv	$\vec{v}$	\vec{v}	vh	$\hat{v}$	\hat{v}
wv	$\vec{w}$	\vec{w}	wh	$\hat{w}$	\hat{w}
xv	$\vec{x}$	\vec{x}	xh	$\hat{x}$	\hat{x}
yv	$\vec{y}$	\vec{y}	yh	$\hat{y}$	\hat{y}
zv	$\vec{z}$	\vec{z}	zh	$\hat{z}$	\hat{z}

2 Function Expansion

Table 5: Keys that will execute some elisp functions

key	trans	sym	description
/	quail-TeQ-frac	$\frac{\square}{\square}$	fraction on previous
eq	quail-TeQ-equation		equation environment
al	quail-TeQ-aligned		aligned environment
el	quail-TeQ-endofline		end of line

3 Symbols :

3.1 Dots related

Table 6: Multiple Dots Related

key	trans	sym	description
...	\dots	$\ldots$	3 dots
.v	\vdots	$\vdots$	vertical dots
.d	\ddots	$\ddots$	diagonale dots
.l	\ldots	$\ldots$	low dots

3.2 Geometry

Table 7:

key	trans	sym	description
perp	\perp	$\perp$	
perpn	\perp	$\not\perp$	
para	\parallel	$\parallel$	
paran	\nparallel	$\nparallel$	
ang	\angle	$\angle$	
ang.	\measuredangle	$\measuredangle$	

3.3 Letter like

Table 8: Letter-like Symbold

key	trans	sym	description
inf	\infty	∞	
ex	\exists	$\exists$	
ex.	\nexists	$\nexists$	
fa	\forall	$\forall$	
hb	\hbar	$\hbar$	
hb.	\hslash	$\hbar$	
dd	\mathrm{d}	d	
dd.	\partial	∂	
ii	\imath	$\imath$	
jj	\jmath	$\jmath$	
nab	\nabla	∇	
cm	\checkmark	$\checkmark$	

3.4 Spaces

Table 9: Space Symbold

key	trans	sym	description
qu	\quad		
quu	\qquad		

3.5 Arrows:

3.5.1 Single:

Table 10: Single Line arrows

key	trans	sym	description
`<-`	`\leftarrow`	$\leftarrow$	left arrow
`->`	`\rightarrow`	$\rightarrow$	right arrow
`-^`	`\uparrow`	$\uparrow$	up arrow
`-v`	`\downarrow`	$\downarrow$	down arrow
`<->`	`\leftrightarrow`	$\leftrightarrow$	left-right arrow
`<-n`	`\nleftarrow`	$\nleftarrow$	not left arrow
`->n`	`\nrightarrow`	$\nrightarrow$	not right arrow
`-^n`	`\nuparrow`	$\nuparrow$	not up arrow
`-vn`	`\ndownarrow`	$\ndownarrow$	not down arrow
`<->`	`\nleftrightarrow`	$\nleftrightarrow$	not left-right arrow
`-->`	`\longrightarrow`	$\longrightarrow$	
`<--`	`\longleftarrow`	$\longleftarrow$	
`| ->`	`\mapsto`	$\mapsto$	

3.5.2 Double:

Table 11: Double Line arrows

key	trans	sym	description
`<=`	`\Leftarrow`	$\Leftarrow$	left arrow
`=>`	`\Rightarrow`	$\Rightarrow$	right arrow
`=^`	`\Uparrow`	$\Uparrow$	up arrow
`=v`	`\Downarrow`	$\Downarrow$	down arrow
`<=>`	`\Leftrightarrow`	$\Leftrightarrow$	left-right arrow
`iff`	`\Leftrightarrow`	$\Leftrightarrow$	left-right arrow
`<=n`	`\nLeftarrow`	$\nLeftarrow$	left arrow
`=>n`	`\nRightarrow`	$\nRightarrow$	right arrow
`<=>n`	`\nLeftrightarrow`	$\nLeftrightarrow$	left-right arrow
`iffn`	`\nLeftrightarrow`	$\nLeftrightarrow$	left-right arrow
`<==>`	`\Longleftrightarrow`	$\Longleftrightarrow$	left-right arrow
`<==`	`\Longleftarrow`	$\Longleftarrow$	left-right arrow
`==>`	`\Longrightarrow`	$\Longrightarrow$	left-right arrow

3.5.3 Long arrow with top-bottom entries

Table 12: Long arrow Line arrows

key	trans	sym	description
`<--`	`\xleftarrow[ ]{ }`	$\xleftarrow[\square]{\square}$	
`-->`	`\xrightarrow[ ]{ }`	$\xrightarrow[\square]{\square}$	
`===>`	`\xRightarrow[ ]{ }`	$\xRightarrow[\square]{\square}$	`mathtools` lib required
`<===`	`\xLeftarrow[ ]{ }`	$\xLeftarrow[\square]{\square}$	`mathtools` lib required

4 Symbol Modification

4.1 Accents (variable decoration?)

Table 13:

key	trans	sym	description
`vec`	`\vec`	$\vec{\square}$	
`bar`	`\bar`	$\bar{\square}$	
`hat`	`\hat`	$\hat{\square}$	
`dot`	`\dot`	$\dot{\square}$	
`dot.`	`\ddot`	$\ddot{\square}$	
`dot..`	`\dddot`	$\dddot{\square}$	
`dot...`	`\ddddot`	$\ddddot{\square}$	
`dag`	`^\dagger`	$\square^{\dagger}$	
`dag.`	`^\ddagger`	$\square^{\ddagger}$	
`*..`	`^*`	$\square^{*}$	
`deg`	`^\circ`	$\square^{\circ}$	
`tr`	`^T`	$\square^{T}$	
`tr.`	`^{-T}`	$\square^{-T}$	

5 Binary Operation Symbols

5.1 Simple Arithmetics:

Table 14: Simple Arithmetics operations

key	trans	sym
`+-`	`\pm`	$\pm$
`-+`	`\mp`	$\mp$
`*x`	`\times`	$\times$
`::`	`\div`	$\div$
`**`	`\cdot`	$\cdot$

5.2 Binary Relations:

Table 15:

key	trans	sym	description
=n	\neq	$\neq$	
=.	\equiv	$\equiv$	
=?	\stackrel{?}{=}	$\stackrel{?}{=}$	
=y	\stackrel{\checkmark}{=}	$\stackrel{\checkmark}{=}$	
3=	\equiv	$\equiv$	
=:	\coloneqq	$\coloneqq$	
:=	\coloneqq	$\coloneqq$	
~.	\sim	$\sim$	
~n	\nsim	$\nsim$	
~~	\approx	$\approx$	
<n	\nless	$\nless$	
<.	\leq	$\leq$	
<.n	\nleq	$\nleq$	
<?	\stackrel{?}{<}	$\stackrel{?}{<}$	
<y	\stackrel{\checkmark}{<}	$\stackrel{\checkmark}{<}$	
<.?	\stackrel{?}{\leq}	$\stackrel{?}{\leq}$	
<.y	\stackrel{\checkmark}{\leq}	$\stackrel{\checkmark}{\leq}$	
«	\ll	$\ll$	
«?	\stackrel{?}{\ll}	$\stackrel{?}{\ll}$	
«y	\stackrel{\checkmark}{\ll}	$\stackrel{\checkmark}{\ll}$	
>n	\ngtr	$\ngtr$	
>.	\geq	$\geq$	
>.n	\ngeq	$\ngeq$	
>?	\stackrel{?}{>}	$\stackrel{?}{>}$	
>y	\stackrel{\checkmark}{>}	$\stackrel{\checkmark}{>}$	
>.?	\stackrel{?}{\geq}	$\stackrel{?}{\geq}$	
>.y	\stackrel{\checkmark}{\geq}	$\stackrel{\checkmark}{\geq}$	
»	\gg	$\gg$	
»?	\stackrel{?}{\gg}	$\stackrel{?}{\gg}$	
»y	\stackrel{\checkmark}{\gg}	$\stackrel{\checkmark}{\gg}$	

5.3 Set symbols

Table 16:

key	trans	sym	description
in	\in	$\in$	
in.	\ni	$\ni$	
ni	\ni	$\ni$	
inn	\notin	$\notin$	
0/	\emptyset	$\emptyset$	
nsr	\mathbb{R}	$\mathbb{R}$	
nsc	\mathbb{C}	$\mathbb{C}$	
nsn	\mathbb{N}	$\mathbb{N}$	
nsp	\mathbb{P}	$\mathbb{P}$	
nsz	\mathbb{Z}	$\mathbb{Z}$	
nsi	\mathbb{I}	$\mathbb{I}$	
sub	\subset	$\subset$	
subn	\nssubseteq	$\nsubseteq$	
sub=	\subseteq	$\subseteq$	
sub=n	\nsubseteq	$\nsubseteq$	
subn=	\nsubseteq	$\nsubseteq$	
sup	\supset	$\supset$	
supn	\nsupseteq	$\not\supseteq$	
sup=	\supeseteq	$\supseteq$	
sup=n	\nsupseteq	$\not\supseteq$	
supn=	\nsupseteq	$\not\supseteq$	

5.4 Logic

Table 17:

key	trans	sym	description
or	\lor	$\lor$	
and	\lnd	$\land$	
not	\neg	$\neg$	
or.	\text{ or }	or	
and.	\text{ and }	and	
not.	\text{ not }	not	

6 Functions

6.1 Function

Table 18:

key	trans	sym	description
rank	\mathrm{rank}	rank	
arg	\arg	arg	
det	\det	det	
dim	\dim	dim	
exp	\exp	exp	
Im	\mathrm{Im}	Im	
Re	\mathrm{Re}	Re	
ln	\ln	ln	
log	\log	log	
max	\max	max	
min	\min	min	
dim	\dim	dim	
sqrt	\sqrt	$\sqrt[\square]{\square}$	
mod	\pmod	$\square$ (mod $\square$)	
mod.	\mod	$\square$ mod $\square$	
mod..	\bmod	$\square$ mod $\square$	

6.2 Trignometry: function

Table 19:

key	sym	trans	key	sym	trans
cos	cos	\cos	cosh	cosh	\cosh
sin	sin	\sin	sinh	sinh	\sinh
tan	tan	\tan	tanh	tanh	\tanh
cot	cot	\cot	coth	coth	\coth
acos	arccos	\arccos	cos.	arccos	\arccos
asin	arcsin	\arcsin	sin.	arcsin	\arcsin
atan	arctan	\arctan	tan.	arctan	\arctan

6.3 Iterative-like operation:

Table 20: Integrals, Sums, Products

key	trans	sym	description
il	\limits_{ }^{ }	$\sum\limits_{here}^{here}$	
lim	\lim	$\lim$	
sum	\sum	$\sum$	
prod	\prod	$\prod$	
int	\int	$\int$	
inti	\iint	$\iint$	
intii	\iiint	$\iiint$	
intiii	\iiiint	$\iiiint$	
into	\oint	$\oint$	
sum.	\sum\limits_{ i=1 }^{ n }	$\sum\limits_{i=1}^{n}$	
prod.	\prod\limits_{ i=1 }^{ n }	$\prod\limits_{i=1}^{n}$	
int.	\int\limits_{ -\infty }^{ -\infty }	$\int\limits_{-\infty}^{-\infty}$	
inti.	\iint\limits_{ C }	$\iint\limits_{C}$	
intii.	\iiint\limits_{ C }	$\iiint\limits_{C}$	
intiii.	\iiiint\limits_{ C }	$\iiiint\limits_{C}$	
into.	\oint\limits_{ C }	$\oint\limits_{C}$	

7 Structural:

7.1 Parenthesis Related

Table 21:

key	trans	sym	description
`().`	`\left( \right)`	$(\square)$	
`()..`	`\left( \middle\vert \right)`	$(\square\|\square)$	
`[].`	`\left[ \right]`	$[\square]$	
`[]..`	`\left[ \middle\vert \right]`	$[\square\|\square]$	
`[].c`	`\lceil \rceil`	$\lceil\square\rceil$	
`[].f`	`\lfloor \floor`	$\lfloor\square\rfloor$	
`{}.`	`\left\{ \right\}`	$\{\square\}$	
`{}..`	`\left\{ \middle\vert \right\}`	$\{\square\|\square\}$	
`|| .`	`\left\vert \right\vert`	$\|\square\|$	

7.2 Texts:

Table 22:

key	trans	sym	description
`te`	`\text{}`	$a + \text{text}$	
`tr`	`\mathrm{}`	$a + \mathrm{mathrm}$	
`tb`	`\mathbf{}`	$a + \mathbf{mathbf}$	
`ti`	`\mathit{}`	$a + \mathit{mathit}$	

7.3 Superscripts (power) & Subsripts (lower)

Table 23:

key	sym	trans	key	sym	trans
pp	$\square^{\square}$	^{	ll	$\square_{\square}$	_{
p0	$\square^{0}$	^0	l0	$\square_{0}$	_0
p1	$\square^{1}$	^1	l1	$\square_{1}$	_1
p2	$\square^{2}$	^2	l2	$\square_{2}$	_2
p3	$\square^{3}$	^3	l3	$\square_{3}$	_3
p4	$\square^{4}$	^4	l4	$\square_{4}$	_4
pn	$\square^{n}$	^n	lnn	$\square_{n}$	_n
px	$\square^{x}$	^x	li	$\square_{i}$	_i
__	$\underset{\square}{\square}$	\underset{ }{ }	^^	$\overset{\square}{\square}$	\overset{ }{ }
__.	$\underbrace{\square}_{\square}$	\underbrace{ }_{ }	^^.	$\overbrace{\square}^{\square}$	\overbrace{ }^{ }
__..	$\underline{\square}$	\underline{ }	^^..	$\overline{\square}$	\overline{ }

7.4 Misc.

Table 24:

key	trans	sym	description
binom	\binom	$\binom{\square}{\square}$	
box	\boxed	$\boxed{\square}$	
can	\cancel	$\cancel{\square}$	requires cancel
&=	&=\n\\\\		
=&	&=\n\\\\		

7.5 xy Diagram related

Table 25:

key	trans	sym	description
xy	\xymatrix{\n\n}		
bu	\bullet	$\bullet$	
ar	\ar		

8 Formatting Table into Elisp

```
def format_table_to_elisp_type6col(headcomment, table):
    print(f";; {headcomment}")
    table = table[1:]
    for line in table:
        key, sym, trans, key1, sym, trans1 = line
        key   = repr(key).replace("\'", "\"").replace("~", "")
        key1  = repr(key1).replace("\'", "\"").replace("~", "")
        trans = repr(trans).replace("\'", "\"").replace("~", "")
        trans1 = repr(trans1).replace("\'", "\"").replace("~", "")

        print(f"({key:<7}  [{trans:<17}])  ({key1:<7}  [{trans1:<17}])")

def format_table_to_elisp_type3col_type1(headcomment, table):
    print(f";; {headcomment}")
    table = table[1:]
    for line in table:
        key, trans, sym, description = line
        key   = repr(key).replace("\'", "\"").replace("~", "")
        trans = repr(trans).replace("\'", "\"").replace("~", "")

        print(f"({key:<8}  [{trans:<22}])  ; {description}")

def format_table_to_elisp_type3col_type2(headcomment, table):
    print(f";; {headcomment}")
    table = table[1:]
    for line in table:
        key, trans, sym, description = line
        key   = repr(key).replace("\'", "\"").replace("~", "")
        trans = trans.replace("~", "")

        print(f"({key:<8}  {trans:<22})  ; {description}")

format_table_to_elisp_type6col("Greek", tbl_1_greek)
format_table_to_elisp_type6col("Matrix", tbl_1_matrix)
format_table_to_elisp_type6col("Vector & Hat", tbl_1_vec)

format_table_to_elisp_type3col_type2("Expanding Func", tbl2_exec_func)
```

```
format_table_to_elisp_type3col_type1("Symbols-dots", tbl_3_sym_dots)
format_table_to_elisp_type3col_type1("Symbols-geo", tbl_3_sym_geo)
format_table_to_elisp_type3col_type1("Symbols", tbl_3_sym_letter)
format_table_to_elisp_type3col_type1("Symbols spaces", tbl_3_sym_spc)
format_table_to_elisp_type3col_type1("Symbols arrow1", tbl_3_sym_arrow_1)
format_table_to_elisp_type3col_type1("Symbols arrow2", tbl_3_sym_arrow_2)
format_table_to_elisp_type3col_type1("Symbols arrow3", tbl_3_sym_arrow_3)

format_table_to_elisp_type3col_type1("Symbols arrow3", tbl_4_sym_mod_1)

format_table_to_elisp_type3col_type1("Operation: arith", tbl_5_op_arith)
format_table_to_elisp_type3col_type1("Operation: arith", tbl_5_op_bin)
format_table_to_elisp_type3col_type1("Operation: arith", tbl_5_op_set)
format_table_to_elisp_type3col_type1("Operation: arith", tbl_5_op_logic)

format_table_to_elisp_type3col_type1("Func: main", tbl_6_func)
format_table_to_elisp_type6col("Func: Trig", tbl_6_func_trig_6col)
format_table_to_elisp_type3col_type1("Func: iter", tbl_6_func_iter)

format_table_to_elisp_type3col_type1("Structural: Parenthesis",  tbl_7_parenthesis)
format_table_to_elisp_type3col_type1("Structural: Text",  tbl_7_text)
format_table_to_elisp_type3col_type1("Structural: Text",  tbl_7_text)
format_table_to_elisp_type6col("Structural: Sub-sup-scripts",  tbl_7_supsubscripts)
format_table_to_elisp_type3col_type1("Structural: misc",  tbl_7_misc)
format_table_to_elisp_type3col_type1("Structural: xy",  tbl_7_xy)

;; Greek
("a."     ["\\alpha"        ])  ("A."     ["A"              ])
("b."     ["\\beta"         ])  ("B."     ["B"              ])
("c."     ["\\psi"          ])  ("C."     ["\\Psi"          ])
("d."     ["\\delta"        ])  ("D."     ["\\Delta"        ])
("e."     ["\\epsilon"      ])  ("E."     ["E"              ])
("f."     ["\\phi"          ])  ("F."     ["\\Phi"          ])
("g."     ["\\gamma"        ])  ("G."     ["\\Gamma"        ])
("h."     ["\\eta"          ])  ("H."     ["H"              ])
("i."     ["\\iota"         ])  ("I."     ["I"              ])
("j."     ["\\xi"           ])  ("J."     ["\\Xi"           ])
("k."     ["\\kappa"        ])  ("K."     ["K"              ])
("l."     ["\\lambda"       ])  ("L."     ["\\Lambda"       ])
("m."     ["\\mu"           ])  ("M."     ["M"              ])
```

```
("n."     ["\\nu"          ])  ("N."     ["N"             ])
("o."     ["o"             ])  ("O."     ["O"             ])
("p."     ["\\pi"          ])  ("P."     ["\\Pi"          ])
("r."     ["\\rho"         ])  ("R."     ["P"             ])
("s."     ["\\sigma"       ])  ("S."     ["\\Sigma"       ])
("t."     ["\\tau"         ])  ("T."     ["T"             ])
("th."    ["\\theta"       ])  ("Th."    ["\\Theta"       ])
("u."     ["\\upsilon"     ])  ("U."     ["\\Upsilon"     ])
("w."     ["\\omega"       ])  ("W."     ["\\Omega"       ])
("x."     ["\\chi"         ])  ("X."     ["X"             ])
("z."     ["\\zeta"        ])  ("Z."     ["Z"             ])
;; Matrix
("Am"     ["\\mathbf{A}"   ])  ("am"     ["\\mathbf{a}"   ])
("Bm"     ["\\mathbf{B}"   ])  ("bm"     ["\\mathbf{b}"   ])
("Cm"     ["\\mathbf{C}"   ])  ("cm"     ["\\mathbf{c}"   ])
("Dm"     ["\\mathbf{D}"   ])  ("dm"     ["\\mathbf{d}"   ])
("Em"     ["\\mathbf{E}"   ])  ("em"     ["\\mathbf{e}"   ])
("Fm"     ["\\mathbf{F}"   ])  ("fm"     ["\\mathbf{f}"   ])
("Gm"     ["\\mathbf{G}"   ])  ("gm"     ["\\mathbf{g}"   ])
("Hm"     ["\\mathbf{H}"   ])  ("hm"     ["\\mathbf{h}"   ])
("Im"     ["\\mathbf{I}"   ])  ("im"     ["\\mathbf{i}"   ])
("Jm"     ["\\mathbf{J}"   ])  ("jm"     ["\\mathbf{j}"   ])
("Km"     ["\\mathbf{K}"   ])  ("km"     ["\\mathbf{k}"   ])
("Lm"     ["\\mathbf{L}"   ])  ("lm"     ["\\mathbf{l}"   ])
("Mm"     ["\\mathbf{M}"   ])  ("mm"     ["\\mathbf{m}"   ])
("Nm"     ["\\mathbf{N}"   ])  ("nm"     ["\\mathbf{n}"   ])
("Om"     ["\\mathbf{O}"   ])  ("om"     ["\\mathbf{o}"   ])
("Pm"     ["\\mathbf{P}"   ])  ("pm"     ["\\mathbf{p}"   ])
("Qm"     ["\\mathbf{Q}"   ])  ("qm"     ["\\mathbf{q}"   ])
("Rm"     ["\\mathbf{R}"   ])  ("rm"     ["\\mathbf{r}"   ])
("Sm"     ["\\mathbf{S}"   ])  ("sm"     ["\\mathbf{s}"   ])
("Tm"     ["\\mathbf{T}"   ])  ("tm"     ["\\mathbf{t}"   ])
("Um"     ["\\mathbf{U}"   ])  ("um"     ["\\mathbf{u}"   ])
("Vm"     ["\\mathbf{V}"   ])  ("vm"     ["\\mathbf{v}"   ])
("Wm"     ["\\mathbf{W}"   ])  ("wm"     ["\\mathbf{w}"   ])
("Xm"     ["\\mathbf{X}"   ])  ("xm"     ["\\mathbf{x}"   ])
("Ym"     ["\\mathbf{Y}"   ])  ("ym"     ["\\mathbf{y}"   ])
("Zm"     ["\\mathbf{Z}"   ])  ("zm"     ["\\mathbf{z}"   ])
("Om"     ["\\mathbf{O}"   ])  ("0m"     ["\\mathbf{0}"   ])
;; Vector & Hat
```

```
("av"     ["\\vec{a}"       ])  ("ah"     ["\\hat{a}"       ])
("bv"     ["\\vec{b}"       ])  ("bh"     ["\\hat{b}"       ])
("cv"     ["\\vec{c}"       ])  ("ch"     ["\\hat{c}"       ])
("dv"     ["\\vec{d}"       ])  ("dh"     ["\\hat{d}"       ])
("ev"     ["\\vec{e}"       ])  ("eh"     ["\\hat{e}"       ])
("fv"     ["\\vec{f}"       ])  ("fh"     ["\\hat{f}"       ])
("gv"     ["\\vec{g}"       ])  ("gh"     ["\\hat{g}"       ])
("hv"     ["\\vec{h}"       ])  ("hh"     ["\\hat{h}"       ])
("iv"     ["\\vec{i}"       ])  ("ih"     ["\\hat{i}"       ])
("jv"     ["\\vec{j}"       ])  ("jh"     ["\\hat{j}"       ])
("kv"     ["\\vec{k}"       ])  ("kh"     ["\\hat{k}"       ])
("lv"     ["\\vec{l}"       ])  ("lh"     ["\\hat{l}"       ])
("mv"     ["\\vec{m}"       ])  ("mh"     ["\\hat{m}"       ])
("nv"     ["\\vec{n}"       ])  ("nh"     ["\\hat{n}"       ])
("ov"     ["\\vec{o}"       ])  ("oh"     ["\\hat{o}"       ])
("pv"     ["\\vec{p}"       ])  ("ph"     ["\\hat{p}"       ])
("qv"     ["\\vec{q}"       ])  ("qh"     ["\\hat{q}"       ])
("rv"     ["\\vec{r}"       ])  ("rh"     ["\\hat{r}"       ])
("sv"     ["\\vec{s}"       ])  ("sh"     ["\\hat{s}"       ])
("tv"     ["\\vec{t}"       ])  ("th"     ["\\hat{t}"       ])
("uv"     ["\\vec{u}"       ])  ("uh"     ["\\hat{u}"       ])
("vv"     ["\\vec{v}"       ])  ("vh"     ["\\hat{v}"       ])
("wv"     ["\\vec{w}"       ])  ("wh"     ["\\hat{w}"       ])
("xv"     ["\\vec{x}"       ])  ("xh"     ["\\hat{x}"       ])
("yv"     ["\\vec{y}"       ])  ("yh"     ["\\hat{y}"       ])
("zv"     ["\\vec{z}"       ])  ("zh"     ["\\hat{z}"       ])
;; Expanding Func
("/"      quail-TeQ-frac        )  ; fraction on previous
("eq"     quail-TeQ-equation    )  ; equation environment
("al"     quail-TeQ-aligned     )  ; aligned environment
("el"     quail-TeQ-endofline   )  ; end of line
;; Symbols-dots
("..."    ["\\dots"             ])  ; 3 dots
(".v"     ["\\vdots"            ])  ; vertical dots
(".d"     ["\\ddots"            ])  ; diagonale dots
(".l"     ["\\ldots"            ])  ; low dots
;; Symbols-geo
("perp"   ["\\perp"             ])  ;
("perpn"  ["\\perp"             ])  ;
("para"   ["\\parallel"         ])  ;
```

```
("paran"    ["\\nparallel"          ])  ;
("ang"      ["\\angle"              ])  ;
("ang."     ["\\measuredangle"      ])  ;
;; Symbols
("inf"      ["\\infty"              ])  ;
("ex"       ["\\exists"             ])  ;
("ex."      ["\\nexists"            ])  ;
("fa"       ["\\forall"             ])  ;
("hb"       ["\\hbar"               ])  ;
("hb."      ["\\hslash"             ])  ;
("dd"       ["\\mathrm{d}"          ])  ;
("dd."      ["\\partial"            ])  ;
("ii"       ["\\imath"              ])  ;
("jj"       ["\\jmath"              ])  ;
("nab"      ["\\nabla"              ])  ;
("cm"       ["\\checkmark"          ])  ;
;; Symbols spaces
("qu"       ["\\quad"               ])  ;
("quu"      ["\\qquad"              ])  ;
;; Symbols arrow1
("<-"       ["\\leftarrow"          ])  ; left arrow
("->"       ["\\rightarrow"         ])  ; right arrow
("-^"       ["\\uparrow"            ])  ; up arrow
("-v"       ["\\downarrow"          ])  ; down arrow
("<->"      ["\\leftrightarrow"     ])  ; left-right arrow
("<-n"      ["\\nleftarrow"         ])  ; not left arrow
("->n"      ["\\nrightarrow"        ])  ; not right arrow
("-^n"      ["\\nuparrow"           ])  ; not up arrow
("-vn"      ["\\ndownarrow"         ])  ; not down arrow
("<->"      ["\\nleftrightarrow"    ])  ; not left-right arrow
("-->"      ["\\longrightarrow"     ])  ;
("<--"      ["\\longleftarrow"      ])  ;
("\\vert ->"  ["\\mapsto"             ])  ;
;; Symbols arrow2
("<="       ["\\Leftarrow"          ])  ; left arrow
("=>"       ["\\Rightarrow"         ])  ; right arrow
("=^"       ["\\Uparrow"            ])  ; up arrow
("=v"       ["\\Downarrow"          ])  ; down arrow
("<=>"      ["\\Leftrightarrow"     ])  ; left-right arrow
("iff"      ["\\Leftrightarrow"     ])  ; left-right arrow
```

```
("<=n"     ["\\nLeftarrow"         ])  ; left arrow
("=>n"     ["\\nRightarrow"        ])  ; right arrow
("<=>n"    ["\\nLeftrightarrow"    ])  ; left-right arrow
("iffn"    ["\\nLeftrightarrow"    ])  ; left-right arrow
("<==>"    ["\\Longleftrightarrow"])  ; left-right arrow
("<=="     ["\\Longleftarrow"      ])  ; left-right arrow
("==>"     ["\\Longrightarrow"     ])  ; left-right arrow
;; Symbols arrow3
("<---"    ["\\xleftarrow[ ]{ }"   ])  ;
("--->"    ["\\xrightarrow[ ]{ }"  ])  ;
("===>"    ["\\xRightarrow[ ]{ }"  ])  ; ~mathtools~ lib required
("<==="    ["\\xLeftarrow[ ]{ }"   ])  ; ~mathtools~ lib required
;; Symbols arrow3
("vec"     ["\\vec"                ])  ;
("bar"     ["\\bar"                ])  ;
("hat"     ["\\hat"                ])  ;
("dot"     ["\\dot"                ])  ;
("dot."    ["\\ddot"               ])  ;
("dot.."   ["\\dddot"              ])  ;
("dot..."  ["\\ddddot"             ])  ;
("dag"     ["^\\dagger"            ])  ;
("dag."    ["^\\ddagger"           ])  ;
("*.."     ["^*"                   ])  ;
("deg"     ["^\\circ"              ])  ;
("tr"      ["^T"                   ])  ;
("tr."     ["^{-T}"                ])  ;
;; Operation: arith
("+-"      ["\\pm"                 ])  ;
("-+"      ["\\mp"                 ])  ;
("*x"      ["\\times"              ])  ;
("::"      ["\\div"                ])  ;
("**"      ["\\cdot"               ])  ;
;; Operation: arith
("=n"      ["\\neq"                ])  ;
("=."      ["\\equiv"              ])  ;
("=?"      ["\\stackrel{?}{=}"     ])  ;
("=y"      ["\\stackrel{\\checkmark}{=}"])  ;
("3="      ["\\equiv"              ])  ;
("=:"      ["\\coloneqq"           ])  ;
(":="      ["\\coloneqq"           ])  ;
```

```
("=.="      ["\\sim"                ])  ;
("=n="      ["\\nsim"               ])  ;
(""         ["\\approx"             ])  ;
("<n"       ["\\nless"              ])  ;
("<."       ["\\leq"                ])  ;
("<.n"      ["\\nleq"               ])  ;
("<?"       ["\\stackrel{?}{<}"     ])  ;
("<y"       ["\\stackrel{\\checkmark}{<}"])  ;
("<.?"      ["\\stackrel{?}{\\leq}"])  ;
("<.y"      ["\\stackrel{\\checkmark}{\\leq}"])  ;
("<<"       ["\\ll"                 ])  ;
("<<?"      ["\\stackrel{?}{\\ll}" ])  ;
("<<y"      ["\\stackrel{\\checkmark}{\\ll}"])  ;
(">n"       ["\\ngtr"               ])  ;
(">."       ["\\geq"                ])  ;
(">.n"      ["\\ngeq"               ])  ;
(">?"       ["\\stackrel{?}{>}"     ])  ;
(">y"       ["\\stackrel{\\checkmark}{>}"])  ;
(">.?"      ["\\stackrel{?}{\\geq}"])  ;
(">.y"      ["\\stackrel{\\checkmark}{\\geq}"])  ;
(">>"       ["\\gg"                 ])  ;
(">>?"      ["\\stackrel{?}{\\gg}" ])  ;
(">>y"      ["\\stackrel{\\checkmark}{\\gg}"])  ;
;; Operation: arith
("in"       ["\\in"                 ])  ;
("in."      ["\\ni"                 ])  ;
("ni"       ["\\ni"                 ])  ;
("inn"      ["\\notin"              ])  ;
("0/"       ["\\emptyset"           ])  ;
("nsr"      ["\\mathbb{R}"          ])  ;
("nsc"      ["\\mathbb{C}"          ])  ;
("nsn"      ["\\mathbb{N}"          ])  ;
("nsp"      ["\\mathbb{P}"          ])  ;
("nsz"      ["\\mathbb{Z}"          ])  ;
("nsi"      ["\\mathbb{I}"          ])  ;
("sub"      ["\\subset"             ])  ;
("subn"     ["\\nssubseteq"         ])  ;
("sub="     ["\\subseteq"           ])  ;
("sub=n"    ["\\nsubseteq"          ])  ;
("subn="    ["\\nsubseteq"          ])  ;
```

```
("sup"     ["\\supset"           ])  ;
("supn"    ["\\nsupseteq"        ])  ;
("sup="    ["\\supeseteq"        ])  ;
("sup=n"   ["\\nsupseteq"        ])  ;
("supn="   ["\\nsupseteq"        ])  ;
;; Operation: arith
("or"      ["\\lor"              ])  ;
("and"     ["\\lnd"              ])  ;
("not"     ["\\neg"              ])  ;
("or."     ["\\text{ or }"       ])  ;
("and."    ["\\text{ and }"      ])  ;
("not."    ["\\text{ not }"      ])  ;
;; Func: main
("rank"    ["\\mathrm{rank}"     ])  ;
("arg"     ["\\arg"              ])  ;
("det"     ["\\det"              ])  ;
("dim"     ["\\dim"              ])  ;
("exp"     ["\\exp"              ])  ;
("Im"      ["\\mathrm{Im}"       ])  ;
("Re"      ["\\mathrm{Re}"       ])  ;
("ln"      ["\\ln"               ])  ;
("log"     ["\\log"              ])  ;
("max"     ["\\max"              ])  ;
("min"     ["\\min"              ])  ;
("dim"     ["\\dim"              ])  ;
("sqrt"    ["\\sqrt"             ])  ;
("mod"     ["\\pmod"             ])  ;
("mod."    ["\\mod"              ])  ;
("mod.."   ["\\bmod"             ])  ;
;; Func: Trig
("cos"    ["\\cos"          ])  ("cosh"   ["\\cosh"         ])
("sin"    ["\\sin"          ])  ("sinh"   ["\\sinh"         ])
("tan"    ["\\tan"          ])  ("tanh"   ["\\tanh"         ])
("cot"    ["\\cot"          ])  ("coth"   ["\\coth"         ])
("acos"   ["\\arccos"       ])  ("cos."   ["\\arccos"       ])
("asin"   ["\\arcsin"       ])  ("sin."   ["\\arcsin"       ])
("atan"   ["\\arctan"       ])  ("tan."   ["\\arctan"       ])
;; Func: iter
("il"      ["\\limits_{ }^{ }"   ])  ;
("lim"     ["\\lim"              ])  ;
```

```
("sum"     ["\\sum"              ])  ;
("prod"    ["\\prod"             ])  ;
("int"     ["\\int"              ])  ;
("inti"    ["\\iint"             ])  ;
("intii"   ["\\iiint"            ])  ;
("intiii"  ["\\iiiint"           ])  ;
("into"    ["\\oint"             ])  ;
("sum."    ["\\sum\\limits_{ i=1 }^{ n }"])  ;
("prod."   ["\\prod\\limits_{ i=1 }^{ n }"])  ;
("int."    ["\\int\\limits_{ -\\infty }^{ -\\infty }"])  ;
("inti."   ["\\iint\\limits_{ C }"])  ;
("intii."  ["\\iiint\\limits_{ C }"])  ;
("intiii."  ["\\iiiint\\limits_{ C }"])  ;
("into."   ["\\oint\\limits_{ C }"])  ;
;; Structural: Parenthesis
("()."     ["\\left( \\right)"    ])  ;
("().."    ["\\left( \\middle\\vert  \\right)"])  ;
("[]."     ["\\left[ \\right]"    ])  ;
("[].."    ["\\left[ \\middle\\vert  \\right]"])  ;
("[].c"    ["\\lceil \\rceil"     ])  ;
("[].f"    ["\\lfloor \\floor"    ])  ;
("{}."     ["\\left\\{ \\right\\}"])  ;
("{}.."    ["\\left\\{ \\middle\\vert  \\right\\}"])  ;
("\\vert\\vert ."  ["\\left\\vert \\right\\vert"])  ;
;; Structural: Text
("te"      ["\\text{}"           ])  ;
("tr"      ["\\mathrm{}"         ])  ;
("tb"      ["\\mathbf{}"         ])  ;
("ti"      ["\\mathit{}"         ])  ;
;; Structural: Text
("te"      ["\\text{}"           ])  ;
("tr"      ["\\mathrm{}"         ])  ;
("tb"      ["\\mathbf{}"         ])  ;
("ti"      ["\\mathit{}"         ])  ;
;; Structural: Sub-sup-scripts
("pp"     ["^{"            ])  ("ll"     ["_{"            ])
("p0"     ["^0"            ])  ("l0"     ["_0"            ])
("p1"     ["^1"            ])  ("l1"     ["_1"            ])
("p2"     ["^2"            ])  ("l2"     ["_2"            ])
("p3"     ["^3"            ])  ("l3"     ["_3"            ])
```

```
("p4"     ["^4"            ])  ("l4"     ["_4"            ])
("pn"     ["^n"            ])  ("lnn"    ["_n"            ])
("px"     ["^x"            ])  ("li"     ["_i"            ])
("__"     ["\\underset{ }{ }"])  ("^^"     ["\\overset{ }{ }"])
("__."    ["\\underbrace{ }_{ }"])  ("^^."    ["\\overbrace{ }^{ }"])
("__.."   ["\\underline{ }" ])  ("^^.."   ["\\overline{ }"  ])
;; Structural: misc
("binom"   ["\\binom"            ])  ;
("box"     ["\\boxed"            ])  ;
("can"     ["\\cancel"           ])  ; requires ~cancel~
("&="      ["&=\\n\\\\\\\\\"     ])  ;
("=&"      ["&=\\n\\\\\\\\\"     ])  ;
;; Structural: xy
("xy"      ["\\xymatrix{\\n\\n}" ])  ;
("bu"      ["\\bullet"           ])  ;
("ar"      ["\\ar"               ])  ;
```

9 Making the `el`

```
(require 'quail)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


(defun quail-func-init ()
  (quail-delete-region)
  (setq quail-current-str nil
        quail-converting nil
        quail-conversion-str ""))

(defun quail-func-end ()
  (throw 'quail-tag nil))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(defun quail-TeQ-equation (key idx)
  (quail-func-init)
  (insert "\\begin{equation}\n\n\\end{equation}")
  (previous-line)
  (quail-func-end))
```

```
(defun quail-TeQ-aligned (key idx)
  (quail-func-init)
  (insert "\\begin{aligned}\n\n\\end{aligned}")
  (previous-line)
  (quail-func-end))

(defun quail-TeQ-endofline (key idx)
  (quail-func-init)
  (end-of-line)
  (insert "\\\\\n")
  (quail-func-end))

(defun quail-TeQ-frac (key idx)
  (quail-func-init)

  (backward-sexp) (kill-sexp)
  (if (looking-back "[a-zA-Z]" 0)
      (progn
        (backward-word)
        (if (= (preceding-char) ?\\ )
            (progn (message "yes") (kill-word 1)
                   (backward-delete-char 1) (insert "\\frac{\\")
                   (yank 1) (yank 2) (insert "}{}"))
          (progn (message "no") (forward-word)
                 (insert "\\frac{") (yank) (insert "}{}")))
        )
    (progn (message "no")
                                        ;(forward-word)
           (insert "\\frac{") (yank) (insert "}{}"))
    )
  (backward-char)

  (quail-func-end))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(quail-define-package
 "TeQ-Math" "Emacs-Teq-Latex"  "TeQ-" t
 "TeQ-Math input"
 nil t t t t nil nil nil nil nil t)
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(quail-define-rules
    ;; Greek Alphabets
    ;; Greek
    ("a."     ["\\alpha"        ])  ("A."     ["A"              ])
    ("b."     ["\\beta"         ])  ("B."     ["B"              ])
    ("c."     ["\\psi"          ])  ("C."     ["\\Psi"          ])
    ("d."     ["\\delta"        ])  ("D."     ["\\Delta"        ])
    ("e."     ["\\epsilon"      ])  ("E."     ["E"              ])
    ("f."     ["\\phi"          ])  ("F."     ["\\Phi"          ])
    ("g."     ["\\gamma"        ])  ("G."     ["\\Gamma"        ])
    ("h."     ["\\eta"          ])  ("H."     ["H"              ])
    ("i."     ["\\iota"         ])  ("I."     ["I"              ])
    ("j."     ["\\xi"           ])  ("J."     ["\\Xi"           ])
    ("k."     ["\\kappa"        ])  ("K."     ["K"              ])
    ("l."     ["\\lambda"       ])  ("L."     ["\\Lambda"       ])
    ("m."     ["\\mu"           ])  ("M."     ["M"              ])
    ("n."     ["\\nu"           ])  ("N."     ["N"              ])
    ("o."     ["o"              ])  ("O."     ["O"              ])
    ("p."     ["\\pi"           ])  ("P."     ["\\Pi"           ])
    ("r."     ["\\rho"          ])  ("R."     ["P"              ])
    ("s."     ["\\sigma"        ])  ("S."     ["\\Sigma"        ])
    ("t."     ["\\tau"          ])  ("T."     ["T"              ])
    ("th."    ["\\theta"        ])  ("Th."    ["\\Theta"        ])
    ("u."     ["\\upsilon"      ])  ("U."     ["\\Upsilon"      ])
    ("w."     ["\\omega"        ])  ("W."     ["\\Omega"        ])
    ("x."     ["\\chi"          ])  ("X."     ["X"              ])
    ("z."     ["\\zeta"         ])  ("Z."     ["Z"              ])
    ;; Matrix
    ("Am"     ["\\mathbf{A}"    ])  ("am"     ["\\mathbf{a}"    ])
    ("Bm"     ["\\mathbf{B}"    ])  ("bm"     ["\\mathbf{b}"    ])
    ("Cm"     ["\\mathbf{C}"    ])  ("cm"     ["\\mathbf{c}"    ])
    ("Dm"     ["\\mathbf{D}"    ])  ("dm"     ["\\mathbf{d}"    ])
    ("Em"     ["\\mathbf{E}"    ])  ("em"     ["\\mathbf{e}"    ])
    ("Fm"     ["\\mathbf{F}"    ])  ("fm"     ["\\mathbf{f}"    ])
    ("Gm"     ["\\mathbf{G}"    ])  ("gm"     ["\\mathbf{g}"    ])
    ("Hm"     ["\\mathbf{H}"    ])  ("hm"     ["\\mathbf{h}"    ])
    ("Im"     ["\\mathbf{I}"    ])  ("im"     ["\\mathbf{i}"    ])
```

```
("Jm"     ["\\mathbf{J}"    ])  ("jm"     ["\\mathbf{j}"    ])
("Km"     ["\\mathbf{K}"    ])  ("km"     ["\\mathbf{k}"    ])
("Lm"     ["\\mathbf{L}"    ])  ("lm"     ["\\mathbf{l}"    ])
("Mm"     ["\\mathbf{M}"    ])  ("mm"     ["\\mathbf{m}"    ])
("Nm"     ["\\mathbf{N}"    ])  ("nm"     ["\\mathbf{n}"    ])
("Om"     ["\\mathbf{O}"    ])  ("om"     ["\\mathbf{o}"    ])
("Pm"     ["\\mathbf{P}"    ])  ("pm"     ["\\mathbf{p}"    ])
("Qm"     ["\\mathbf{Q}"    ])  ("qm"     ["\\mathbf{q}"    ])
("Rm"     ["\\mathbf{R}"    ])  ("rm"     ["\\mathbf{r}"    ])
("Sm"     ["\\mathbf{S}"    ])  ("sm"     ["\\mathbf{s}"    ])
("Tm"     ["\\mathbf{T}"    ])  ("tm"     ["\\mathbf{t}"    ])
("Um"     ["\\mathbf{U}"    ])  ("um"     ["\\mathbf{u}"    ])
("Vm"     ["\\mathbf{V}"    ])  ("vm"     ["\\mathbf{v}"    ])
("Wm"     ["\\mathbf{W}"    ])  ("wm"     ["\\mathbf{w}"    ])
("Xm"     ["\\mathbf{X}"    ])  ("xm"     ["\\mathbf{x}"    ])
("Ym"     ["\\mathbf{Y}"    ])  ("ym"     ["\\mathbf{y}"    ])
("Zm"     ["\\mathbf{Z}"    ])  ("zm"     ["\\mathbf{z}"    ])
("Om"     ["\\mathbf{0}"    ])  ("0m"     ["\\mathbf{0}"    ])
;; Vector & Hat
("av"     ["\\vec{a}"       ])  ("ah"     ["\\hat{a}"       ])
("bv"     ["\\vec{b}"       ])  ("bh"     ["\\hat{b}"       ])
("cv"     ["\\vec{c}"       ])  ("ch"     ["\\hat{c}"       ])
("dv"     ["\\vec{d}"       ])  ("dh"     ["\\hat{d}"       ])
("ev"     ["\\vec{e}"       ])  ("eh"     ["\\hat{e}"       ])
("fv"     ["\\vec{f}"       ])  ("fh"     ["\\hat{f}"       ])
("gv"     ["\\vec{g}"       ])  ("gh"     ["\\hat{g}"       ])
("hv"     ["\\vec{h}"       ])  ("hh"     ["\\hat{h}"       ])
("iv"     ["\\vec{i}"       ])  ("ih"     ["\\hat{i}"       ])
("jv"     ["\\vec{j}"       ])  ("jh"     ["\\hat{j}"       ])
("kv"     ["\\vec{k}"       ])  ("kh"     ["\\hat{k}"       ])
("lv"     ["\\vec{l}"       ])  ("lh"     ["\\hat{l}"       ])
("mv"     ["\\vec{m}"       ])  ("mh"     ["\\hat{m}"       ])
("nv"     ["\\vec{n}"       ])  ("nh"     ["\\hat{n}"       ])
("ov"     ["\\vec{o}"       ])  ("oh"     ["\\hat{o}"       ])
("pv"     ["\\vec{p}"       ])  ("ph"     ["\\hat{p}"       ])
("qv"     ["\\vec{q}"       ])  ("qh"     ["\\hat{q}"       ])
("rv"     ["\\vec{r}"       ])  ("rh"     ["\\hat{r}"       ])
("sv"     ["\\vec{s}"       ])  ("sh"     ["\\hat{s}"       ])
("tv"     ["\\vec{t}"       ])  ("th"     ["\\hat{t}"       ])
("uv"     ["\\vec{u}"       ])  ("uh"     ["\\hat{u}"       ])
```

```
("vv"     ["\\vec{v}"       ])  ("vh"     ["\\hat{v}"       ])
("wv"     ["\\vec{w}"       ])  ("wh"     ["\\hat{w}"       ])
("xv"     ["\\vec{x}"       ])  ("xh"     ["\\hat{x}"       ])
("yv"     ["\\vec{y}"       ])  ("yh"     ["\\hat{y}"       ])
("zv"     ["\\vec{z}"       ])  ("zh"     ["\\hat{z}"       ])
;; Expanding Func
("/"       quail-TeQ-frac       )  ; fraction on previous
("eq"      quail-TeQ-equation   )  ; equation environment
("al"      quail-TeQ-aligned    )  ; aligned environment
("el"      quail-TeQ-endofline  )  ; end of line
;; Symbols-dots
("..."     ["\\dots"             ])  ; 3 dots
(".v"      ["\\vdots"            ])  ; vertical dots
(".d"      ["\\ddots"            ])  ; diagonale dots
(".l"      ["\\ldots"            ])  ; low dots
;; Symbols-geo
("perp"    ["\\perp"             ])  ;
("perpn"   ["\\perp"             ])  ;
("para"    ["\\parallel"         ])  ;
("paran"   ["\\nparallel"        ])  ;
("ang"     ["\\angle"            ])  ;
("ang."    ["\\measuredangle"    ])  ;
;; Symbols
("inf"     ["\\infty"            ])  ;
("ex"      ["\\exists"           ])  ;
("ex."     ["\\nexists"          ])  ;
("fa"      ["\\forall"           ])  ;
("hb"      ["\\hbar"             ])  ;
("hb."     ["\\hslash"           ])  ;
("dd"      ["\\mathrm{d}"        ])  ;
("dd."     ["\\partial"          ])  ;
("ii"      ["\\imath"            ])  ;
("jj"      ["\\jmath"            ])  ;
("nab"     ["\\nabla"            ])  ;
("cm"      ["\\checkmark"        ])  ;
;; Symbols spaces
("qu"      ["\\quad"             ])  ;
("quu"     ["\\qquad"            ])  ;
;; Symbols arrow1
("<-"      ["\\leftarrow"        ])  ; left arrow
```

```
("->"      ["\\rightarrow"        ])  ; right arrow
("-^"      ["\\uparrow"           ])  ; up arrow
("-v"      ["\\downarrow"         ])  ; down arrow
("<->"     ["\\leftrightarrow"    ])  ; left-right arrow
("<-n"     ["\\nleftarrow"        ])  ; not left arrow
("->n"     ["\\nrightarrow"       ])  ; not right arrow
("-^n"     ["\\nuparrow"          ])  ; not up arrow
("-vn"     ["\\ndownarrow"        ])  ; not down arrow
("<->"     ["\\nleftrightarrow"   ])  ; not left-right arrow
("-->"     ["\\longrightarrow"    ])  ;
("<--"     ["\\longleftarrow"     ])  ;
("\\vert ->"  ["\\mapsto"           ])  ;
;; Symbols arrow2
("<="      ["\\Leftarrow"         ])  ; left arrow
("=>"      ["\\Rightarrow"        ])  ; right arrow
("=^"      ["\\Uparrow"           ])  ; up arrow
("=v"      ["\\Downarrow"         ])  ; down arrow
("<=>"     ["\\Leftrightarrow"    ])  ; left-right arrow
("iff"     ["\\Leftrightarrow"    ])  ; left-right arrow
("<=n"     ["\\nLeftarrow"        ])  ; left arrow
("=>n"     ["\\nRightarrow"       ])  ; right arrow
("<=>n"    ["\\nLeftrightarrow"   ])  ; left-right arrow
("iffn"    ["\\nLeftrightarrow"   ])  ; left-right arrow
("<==>"    ["\\Longleftrightarrow"])  ; left-right arrow
("<=="     ["\\Longleftarrow"     ])  ; left-right arrow
("==>"     ["\\Longrightarrow"    ])  ; left-right arrow
;; Symbols arrow3
("<---"    ["\\xleftarrow[ ]{ }"  ])  ;
("--->"    ["\\xrightarrow[ ]{ }" ])  ;
("===>"    ["\\xRightarrow[ ]{ }" ])  ; ~mathtools~ lib required
("<==="    ["\\xLeftarrow[ ]{ }"  ])  ; ~mathtools~ lib required
;; Symbols arrow3
("vec"     ["\\vec"               ])  ;
("bar"     ["\\bar"               ])  ;
("hat"     ["\\hat"               ])  ;
("dot"     ["\\dot"               ])  ;
("dot."    ["\\ddot"              ])  ;
("dot.."   ["\\dddot"             ])  ;
("dot..."  ["\\ddddot"            ])  ;
("dag"     ["^\\dagger"           ])  ;
```

```
("dag."    ["^\\ddagger"          ])  ;
("*."      ["^*"                  ])  ;
("deg"     ["^\\circ"             ])  ;
("tr"      ["^T"                  ])  ;
("tr."     ["^{-T}"               ])  ;
;; Operation: arith
("+-"      ["\\pm"                ])  ;
("-+"      ["\\mp"                ])  ;
("*x"      ["\\times"             ])  ;
("::"      ["\\div"               ])  ;
("**"      ["\\cdot"              ])  ;
;; Operation: arith
("=n"      ["\\neq"               ])  ;
("=."      ["\\equiv"             ])  ;
("=?"      ["\\stackrel{?}{=}"    ])  ;
("=y"      ["\\stackrel{\\checkmark}{=}"])  ;
("3="      ["\\equiv"             ])  ;
("=:"      ["\\coloneqq"          ])  ;
(":="      ["\\coloneqq"          ])  ;
("=.="     ["\\sim"               ])  ;
("=n="     ["\\nsim"              ])  ;
(""        ["\\approx"            ])  ;
("<n"      ["\\nless"             ])  ;
("<."      ["\\leq"               ])  ;
("<.n"     ["\\nleq"              ])  ;
("<?"      ["\\stackrel{?}{<}"    ])  ;
("<y"      ["\\stackrel{\\checkmark}{<}"])  ;
("<.?"     ["\\stackrel{?}{\\leq}"])  ;
("<.y"     ["\\stackrel{\\checkmark}{\\leq}"])  ;
("<<"      ["\\ll"                ])  ;
("<<?"     ["\\stackrel{?}{\\ll}" ])  ;
("<<y"     ["\\stackrel{\\checkmark}{\\ll}"])  ;
(">n"      ["\\ngtr"              ])  ;
(">."      ["\\geq"               ])  ;
(">.n"     ["\\ngeq"              ])  ;
(">?"      ["\\stackrel{?}{>}"    ])  ;
(">y"      ["\\stackrel{\\checkmark}{>}"])  ;
(">.?"     ["\\stackrel{?}{\\geq}"])  ;
(">.y"     ["\\stackrel{\\checkmark}{\\geq}"])  ;
(">>"      ["\\gg"                ])  ;
```

```
(">>?"     ["\\stackrel{?}{\\gg}" ])  ;
(">>y"     ["\\stackrel{\\checkmark}{\\gg}"])  ;
;; Operation: arith
("in"      ["\\in"               ])  ;
("in."     ["\\ni"               ])  ;
("ni"      ["\\ni"               ])  ;
("inn"     ["\\notin"            ])  ;
("0/"      ["\\emptyset"         ])  ;
("nsr"     ["\\mathbb{R}"        ])  ;
("nsc"     ["\\mathbb{C}"        ])  ;
("nsn"     ["\\mathbb{N}"        ])  ;
("nsp"     ["\\mathbb{P}"        ])  ;
("nsz"     ["\\mathbb{Z}"        ])  ;
("nsi"     ["\\mathbb{I}"        ])  ;
("sub"     ["\\subset"           ])  ;
("subn"    ["\\nssubseteq"       ])  ;
("sub="    ["\\subseteq"         ])  ;
("sub=n"   ["\\nsubseteq"        ])  ;
("subn="   ["\\nsubseteq"        ])  ;
("sup"     ["\\supset"           ])  ;
("supn"    ["\\nsupseteq"        ])  ;
("sup="    ["\\supeseteq"        ])  ;
("sup=n"   ["\\nsupseteq"        ])  ;
("supn="   ["\\nsupseteq"        ])  ;
;; Operation: arith
("or"      ["\\lor"              ])  ;
("and"     ["\\lnd"              ])  ;
("not"     ["\\neg"              ])  ;
("or."     ["\\text{ or }"       ])  ;
("and."    ["\\text{ and }"      ])  ;
("not."    ["\\text{ not }"      ])  ;
;; Func: main
("rank"    ["\\mathrm{rank}"     ])  ;
("arg"     ["\\arg"              ])  ;
("det"     ["\\det"              ])  ;
("dim"     ["\\dim"              ])  ;
("exp"     ["\\exp"              ])  ;
("Im"      ["\\mathrm{Im}"       ])  ;
("Re"      ["\\mathrm{Re}"       ])  ;
("ln"      ["\\ln"               ])  ;
```

```
("log"     ["\\log"              ])  ;
("max"     ["\\max"              ])  ;
("min"     ["\\min"              ])  ;
("dim"     ["\\dim"              ])  ;
("sqrt"    ["\\sqrt"             ])  ;
("mod"     ["\\pmod"             ])  ;
("mod."    ["\\mod"              ])  ;
("mod.."   ["\\bmod"             ])  ;
;; Func: Trig
("cos"    ["\\cos"          ])  ("cosh"   ["\\cosh"         ])
("sin"    ["\\sin"          ])  ("sinh"   ["\\sinh"         ])
("tan"    ["\\tan"          ])  ("tanh"   ["\\tanh"         ])
("cot"    ["\\cot"          ])  ("coth"   ["\\coth"         ])
("acos"   ["\\arccos"       ])  ("cos."   ["\\arccos"       ])
("asin"   ["\\arcsin"       ])  ("sin."   ["\\arcsin"       ])
("atan"   ["\\arctan"       ])  ("tan."   ["\\arctan"       ])
;; Func: iter
("il"      ["\\limits_{ }^{ }"    ])  ;
("lim"     ["\\lim"               ])  ;
("sum"     ["\\sum"               ])  ;
("prod"    ["\\prod"              ])  ;
("int"     ["\\int"               ])  ;
("inti"    ["\\iint"              ])  ;
("intii"   ["\\iiint"             ])  ;
("intiii"  ["\\iiiint"            ])  ;
("into"    ["\\oint"              ])  ;
("sum."    ["\\sum\\limits_{ i=1 }^{ n }"])  ;
("prod."   ["\\prod\\limits_{ i=1 }^{ n }"])  ;
("int."    ["\\int\\limits_{ -\\infty }^{ -\\infty }"])  ;
("inti."   ["\\iint\\limits_{ C }"])  ;
("intii."  ["\\iiint\\limits_{ C }"])  ;
("intiii."  ["\\iiiint\\limits_{ C }"])  ;
("into."   ["\\oint\\limits_{ C }"])  ;
;; Structural: Parenthesis
("()."     ["\\left( \\right)"    ])  ;
("().."    ["\\left( \\middle\\vert  \\right)"])  ;
("[]."     ["\\left[ \\right]"    ])  ;
("[].."    ["\\left[ \\middle\\vert  \\right]"])  ;
("[].c"    ["\\lceil \\rceil"     ])  ;
("[].f"    ["\\lfloor \\floor"    ])  ;
```

```
    ("{}."      ["\\left\\{ \\right\\}"])  ;
    ("{}.."     ["\\left\\{ \\middle\\vert  \\right\\}"])  ;
    ("\\vert\\vert ."  ["\\left\\vert \\right\\vert"])  ;
    ;; Structural: Text
    ("te"       ["\\text{}"             ])  ;
    ("tr"       ["\\mathrm{}"           ])  ;
    ("tb"       ["\\mathbf{}"           ])  ;
    ("ti"       ["\\mathit{}"           ])  ;
    ;; Structural: Text
    ("te"       ["\\text{}"             ])  ;
    ("tr"       ["\\mathrm{}"           ])  ;
    ("tb"       ["\\mathbf{}"           ])  ;
    ("ti"       ["\\mathit{}"           ])  ;
    ;; Structural: Sub-sup-scripts
    ("pp"     ["^{"             ])  ("ll"     ["_{"             ])
    ("p0"     ["^0"             ])  ("l0"     ["_0"             ])
    ("p1"     ["^1"             ])  ("l1"     ["_1"             ])
    ("p2"     ["^2"             ])  ("l2"     ["_2"             ])
    ("p3"     ["^3"             ])  ("l3"     ["_3"             ])
    ("p4"     ["^4"             ])  ("l4"     ["_4"             ])
    ("pn"     ["^n"             ])  ("lnn"    ["_n"             ])
    ("px"     ["^x"             ])  ("li"     ["_i"             ])
    ("__"     ["\\underset{ }{ }"])  ("^^"     ["\\overset{ }{ }"])
    ("__."    ["\\underbrace{ }_{ }"])  ("^^."    ["\\overbrace{ }^{ }"])
    ("__.."   ["\\underline{ }" ])  ("^^.."   ["\\overline{ }"  ])
    ;; Structural: misc
    ("binom"    ["\\binom"              ])  ;
    ("box"      ["\\boxed"              ])  ;
    ("can"      ["\\cancel"             ])  ; requires ~cancel~
    ("&="       ["&=\\n\\\\\\\\\"       ])  ;
    ("=&"       ["&=\\n\\\\\\\\\"       ])  ;
    ;; Structural: xy
    ("xy"       ["\\xymatrix{\\n\\n}"  ])  ;
    ("bu"       ["\\bullet"             ])  ;
    ("ar"       ["\\ar"                 ])  ;
)
```

10 Execuation el

```
(defun quail-func-init ()
  (quail-delete-region)
  (setq quail-current-str nil
        quail-converting nil
        quail-conversion-str ""))

(defun quail-func-end ()
  (throw 'quail-tag nil))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(defun quail-TeQ-equation (key idx)
  (quail-func-init)
  (insert "\\begin{equation}\n\n\\end{equation}")
  (previous-line)
  (quail-func-end))

(defun quail-TeQ-aligned (key idx)
  (quail-func-init)
  (insert "\\begin{aligned}\n\n\\end{aligned}")
  (previous-line)
  (quail-func-end))

(defun quail-TeQ-endofline (key idx)
  (quail-func-init)
  (end-of-line)
  (insert "\\\\\n")
  (quail-func-end))

(defun quail-TeQ-frac (key idx)
  (quail-func-init)

  (backward-sexp) (kill-sexp)
  (if (looking-back "[a-zA-Z]" 0)
      (progn
        (backward-word)
        (if (= (preceding-char) ?\\ )
            (progn (message "yes") (kill-word 1)
                   (backward-delete-char 1) (insert "\\frac{\\")
```

```
                    (yank 1) (yank 2) (insert "}{}"))
            (progn (message "no") (forward-word)
                   (insert "\\frac{") (yank) (insert "}{}")))
        )
    (progn (message "no")
                                          ;(forward-word)
           (insert "\\frac{") (yank) (insert "}{}"))
    )
  (backward-char)

  (quail-func-end))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```