Emacs TeQ: (T$_{\text{E}}$X + Quail)

Input Method written in Quail for entering L$^{\text{A}}$T$_{\text{E}}$X math expressions

Garid Zorigoo

April 12, 2023

Contents

1   Alphabet related stuff :

1.1   Greek

`gifs/example-greek.gif`

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

1.4 Dots

Table 5: Dots

key	sym	latex (vec)	key	sym	latex (hat)
ad	$\dot{a}$	\dot{a}	Ad	$\dot{A}$	\dot{A}
bd	$\dot{b}$	\dot{b}	Bd	$\dot{B}$	\dot{B}
cd	$\dot{c}$	\dot{c}	Cd	$\dot{C}$	\dot{C}
dd	$\dot{d}$	\dot{d}	Dd	$\dot{D}$	\dot{D}
ed	$\dot{e}$	\dot{e}	Ed	$\dot{E}$	\dot{E}
fd	$\dot{f}$	\dot{f}	Fd	$\dot{F}$	\dot{F}
gd	$\dot{g}$	\dot{g}	Gd	$\dot{G}$	\dot{G}
hd	$\dot{h}$	\dot{h}	Hd	$\dot{H}$	\dot{H}
id	$\dot{i}$	\dot{i}	Id	$\dot{I}$	\dot{I}
jd	$\dot{j}$	\dot{j}	Jd	$\dot{J}$	\dot{J}
kd	$\dot{k}$	\dot{k}	Kd	$\dot{K}$	\dot{K}
ld	$\dot{l}$	\dot{l}	Ld	$\dot{L}$	\dot{L}
md	$\dot{m}$	\dot{m}	Md	$\dot{M}$	\dot{M}
nd	$\dot{n}$	\dot{n}	Nd	$\dot{N}$	\dot{N}
od	$\dot{o}$	\dot{o}	Od	$\dot{O}$	\dot{O}
pd	$\dot{p}$	\dot{p}	Pd	$\dot{P}$	\dot{P}
qd	$\dot{q}$	\dot{q}	Qd	$\dot{Q}$	\dot{Q}
rd	$\dot{r}$	\dot{r}	Rd	$\dot{R}$	\dot{R}
sd	$\dot{s}$	\dot{s}	Sd	$\dot{S}$	\dot{S}
td	$\dot{t}$	\dot{t}	Td	$\dot{T}$	\dot{T}
ud	$\dot{u}$	\dot{u}	Ud	$\dot{U}$	\dot{U}
vd	$\dot{v}$	\dot{v}	Vd	$\dot{V}$	\dot{V}
wd	$\dot{w}$	\dot{w}	Wd	$\dot{W}$	\dot{W}
xd	$\dot{x}$	\dot{x}	Xd	$\dot{X}$	\dot{X}
yd	$\dot{y}$	\dot{y}	Yd	$\dot{Y}$	\dot{Y}
zd	$\dot{z}$	\dot{z}	Zd	$\dot{Z}$	\dot{Z}

1.5 DDots

Table 6: DDots

key	sym	latex (vec)	key	sym	latex (hat)
ad.	$\ddot{a}$	\ddot{a}	Ad.	$\ddot{A}$	\ddot{A}
bd.	$\ddot{b}$	\ddot{b}	Bd.	$\ddot{B}$	\ddot{B}
cd.	$\ddot{c}$	\ddot{c}	Cd.	$\ddot{C}$	\ddot{C}
dd.	$\ddot{d}$	\ddot{d}	Dd.	$\ddot{D}$	\ddot{D}
ed.	$\ddot{e}$	\ddot{e}	Ed.	$\ddot{E}$	\ddot{E}
fd.	$\ddot{f}$	\ddot{f}	Fd.	$\ddot{F}$	\ddot{F}
gd.	$\ddot{g}$	\ddot{g}	Gd.	$\ddot{G}$	\ddot{G}
hd.	$\ddot{h}$	\ddot{h}	Hd.	$\ddot{H}$	\ddot{H}
id.	$\ddot{i}$	\ddot{i}	Id.	$\ddot{I}$	\ddot{I}
jd.	$\ddot{j}$	\ddot{j}	Jd.	$\ddot{J}$	\ddot{J}
kd.	$\ddot{k}$	\ddot{k}	Kd.	$\ddot{K}$	\ddot{K}
ld.	$\ddot{l}$	\ddot{l}	Ld.	$\ddot{L}$	\ddot{L}
md.	$\ddot{m}$	\ddot{m}	Md.	$\ddot{M}$	\ddot{M}
nd.	$\ddot{n}$	\ddot{n}	Nd.	$\ddot{N}$	\ddot{N}
od.	$\ddot{o}$	\ddot{o}	Od.	$\ddot{O}$	\ddot{O}
pd.	$\ddot{p}$	\ddot{p}	Pd.	$\ddot{P}$	\ddot{P}
qd.	$\ddot{q}$	\ddot{q}	Qd.	$\ddot{Q}$	\ddot{Q}
rd.	$\ddot{r}$	\ddot{r}	Rd.	$\ddot{R}$	\ddot{R}
sd.	$\ddot{s}$	\ddot{s}	Sd.	$\ddot{S}$	\ddot{S}
td.	$\ddot{t}$	\ddot{t}	Td.	$\ddot{T}$	\ddot{T}
ud.	$\ddot{u}$	\ddot{u}	Ud.	$\ddot{U}$	\ddot{U}
vd.	$\ddot{v}$	\ddot{v}	Vd.	$\ddot{V}$	\ddot{V}
wd.	$\ddot{w}$	\ddot{w}	Wd.	$\ddot{W}$	\ddot{W}
xd.	$\ddot{x}$	\ddot{x}	Xd.	$\ddot{X}$	\ddot{X}
yd.	$\ddot{y}$	\ddot{y}	Yd.	$\ddot{Y}$	\ddot{Y}
zd.	$\ddot{z}$	\ddot{z}	Zd.	$\ddot{Z}$	\ddot{Z}

2   Function Expansion

Table 7: Keys that will execute some elisp functions

key	sym	latex	description
/	▱	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

3   Symbols :

3.1   Dots related

Table 8: Multiple Dots Related

key	sym	latex	description
...	...	\dots	3 dots
.v	⋮	\vdots	vertical dots
.d	⋱	\ddots	diagonale dots
.l	...	\ldots	low dots

3.2   Geometry

Table 9:

key	sym	latex	description
perp	⊥	\perp	
perpn	⊥̸	\perp	⊥ n (neg)
para	∥	\parallel	
paran	∦	\nparallel	∥ n (neg)
ang	∠	\angle	
ang.	∡	\measuredangle	∠ . (var)
tri	△	\vartriangle	
trin	▽	\triangledown	△ n (neg)
squ	□	\square	
tri.	▲	\blacktriangle	△ . (var)
trin.	▼	\blacktriangledown	△ n. (neg,var)
squ.	■	\blacksquare	□ . (var)

8

3.3 Letter like

Table 10: Letter-like Symbold

key	sym	latex	description
inf	∞	\infty	
ex	$\exists$	\exists	
exn	$\nexists$	\nexists	$\exists$ + $\underline{n}$ (neg)
fa	$\forall$	\forall	
hb	$\hbar$	\hbar	
hb.	$\hbar$	\hslash	$\hbar$ + $\underline{.}$ (var)
dd	d	\mathrm{d}	
dd.	∂	\partial	d + $\underline{.}$ (var)
ii	$\imath$	\imath	
jj	$\jmath$	\jmath	
nab	∇	\nabla	
cm	$\checkmark$	\checkmark	

3.4 Spaces

Table 11: Space Symbold

key	sym	latex	description
qu		\quad	
quu		\qquad	

3.5 Arrows:

3.5.1 Single:

Table 12: Single Line arrows

key	sym	latex	description
<-	←	\leftarrow	
->	→	\rightarrow	
-^	↑	\uparrow	^ looks like up arrow head
-v	↓	\downarrow	v looks like down arrow head
<->	↔	\leftrightarrow	
<-n	↚	\nleftarrow	negate (n) of prev. section
->n	↛	\nrightarrow	arrows + n
-^n	↥	\nuparrow	
-vn	↧	\ndownarrow	
<->	↮	\nleftrightarrow	
-->	⟶	\longrightarrow	longer arrows, with 2 dashes
<--	⟵	\longleftarrow	
| ->	↦	\mapsto	vertical-bar + -> (this might rendered wrongly on Github)

3.5.2 Double:

Table 13: Double Line arrows

key	sym	latex	description
<=	⇐	\Leftarrow	compared to single arrrow
=>	⇒	\Rightarrow	these uses = as the arrow shaft
=^	⇑	\Uparrow	
=v	⇓	\Downarrow	
<=>	⇔	\Leftrightarrow	
iff	⇔	\Leftrightarrow	
<=n	⇍	\nLeftarrow	negate (n) of prev. section
=>n	⇏	\nRightarrow	arrows + n
<=>n	⇎	\nLeftrightarrow	
iffn	⇎	\nLeftrightarrow	
<==>	⟺	\Longleftrightarrow	longer arrows, with 2 dashes
<==	⟸	\Longleftarrow	
==>	⟹	\Longrightarrow	

3.5.3 Long arrow with top-bottom entries

Table 14: Long arrow Line arrows

key	sym	latex	description
<--	$\overset{\square}{\underset{\square}{\longleftarrow}}$	\xleftarrow[ ]{ }	these uses triple - or =
-->	$\overset{\square}{\underset{\square}{\longrightarrow}}$	\xrightarrow[ ]{ }	
===>	$\overset{\square}{\underset{\square}{\Longrightarrow}}$	\xRightarrow[ ]{ }	mathtools lib required
<===	$\overset{\square}{\underset{\square}{\Longleftarrow}}$	\xLeftarrow[ ]{ }	mathtools lib required

4 Symbol Modification

4.1 Accents (variable decoration?)

Table 15:

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

4.2 Superscripts & Subsripts (power & lower)

Table 16:

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

5 Binary Operation Symbols

5.1 Simple Arithmetics:

Table 17: Simple Arithmetics operations

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

5.2 Binary Relations:

Table 18:

key	sym	latex	description
<.	$\leq$	\leq	$< \, = \, >$
>.	$\geq$	\geq	symbols
«	$\ll$	\ll	
»	$\gg$	\gg	
=n	$\neq$	\neq	negation
~n	$\nsim$	\nsim	
<n	$\nless$	\nless	
>n	$\ngtr$	\ngtr	
<.n	$\nleq$	\nleq	
>.n	$\ngeq$	\ngeq	
=?	$\stackrel{?}{=}$	\stackrel{?}{=}	with question mark
<?	$\stackrel{?}{<}$	\stackrel{?}{<}	
>?	$\stackrel{?}{>}$	\stackrel{?}{>}	
<.?	$\stackrel{?}{\leq}$	\stackrel{?}{\leq}	
>.?	$\stackrel{?}{\geq}$	\stackrel{?}{\geq}	
«?	$\stackrel{?}{\ll}$	\stackrel{?}{\ll}	
»?	$\stackrel{?}{\gg}$	\stackrel{?}{\gg}	
=y	$\stackrel{\checkmark}{=}$	\stackrel{\checkmark}{=}	with check mark
<y	$\stackrel{\checkmark}{<}$	\stackrel{\checkmark}{<}	
>y	$\stackrel{\checkmark}{>}$	\stackrel{\checkmark}{>}	
<.y	$\stackrel{\checkmark}{\leq}$	\stackrel{\checkmark}{\leq}	
>.y	$\stackrel{\checkmark}{\geq}$	\stackrel{\checkmark}{\geq}	
«y	$\stackrel{\checkmark}{\ll}$	\stackrel{\checkmark}{\ll}	
»y	$\stackrel{\checkmark}{\gg}$	\stackrel{\checkmark}{\gg}	
=.	$\equiv$	\equiv	Variation on
-.	$\sim$	\sim	Another variations
=..	$\approx$	\approx	on =
3=	$\equiv$	\equiv	
=:	$:=$	\coloneqq	
:=	$:=$	\coloneqq	

14

5.3 Set symbols

Table 19:

key	sym	latex	description
in	∈	\in	
in.	∋	\ni	
ni	∋	\ni	
inn	∉	\notin	(neg)
0/	∅	\emptyset	
nsr	ℝ	\mathbb{R}	(n)umber (s)et (r)eal
nsc	ℂ	\mathbb{C}	(n)umber (s)et (c)omplex
nsn	ℕ	\mathbb{N}	...
nsp	ℙ	\mathbb{P}	...
nsz	ℤ	\mathbb{Z}	...
nsi	𝕀	\mathbb{I}	...
sub	⊂	\subset	
subn	⊄	\nssubseteq	(neg)
sub.	⊆	\subseteq	(var)
sub.n	⊄	\nsubseteq	(var, neg)
subn.	⊄	\nsubseteq	(neg, var)
sup	⊃	\supset	
supn	⊅	\nsupseteq	(neg)
sup.	⊇	\supeseteq	(var)
sup.n	⊅	\nsupseteq	(var, neg)
supn.	⊅	\nsupseteq	(neg, var)

5.4 Logic

Table 20:

key	sym	latex	description
or	∨	\lor	
and	∧	\lnd	
not	¬	\neg	
or.	or	\text{ or }	(var)
and.	and	\text{ and }	(var)
not.	not	\text{ not }	(var)

6 Functions

6.1 Function

Table 21:

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

6.2 Trignometry: function

Table 22:

key	sym	latex	key	sym	latex
cos	$\cos(\square)$	\cos(	cosh	$\cosh(\square)$	\cosh(
sin	$\sin(\square)$	\sin(	sinh	$\sinh(\square)$	\sinh(
tan	$\tan(\square)$	\tan(	tanh	$\tanh(\square)$	\tanh(
cot	$\cot(\square)$	\cot(	coth	$\coth(\square)$	\coth(
acos	$\arccos(\square)$	\arccos(	cos.	$\arccos(\square)$	\arccos(
asin	$\arcsin(\square)$	\arcsin(	sin.	$\arcsin(\square)$	\arcsin(
atan	$\arctan(\square)$	\arctan(	tan.	$\arctan(\square)$	\arctan(

6.3   Iterative-like operation:

Table 23: Integrals, Sums, Products

key	sym	latex	description
il	$\sum\limits_{\square}^{\square}$	\limits_{ }	
il	$\sum\limits_{\square}$	\limits_{ }^{ }	
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.	$\int\limits_{\square}^{\square}$	\int\limits_{ }^{ }	
int..	$\int\limits_{0}^{+\infty}$	\int\limits_{ 0 }^{ +\infty }	
int...	$\int\limits_{-\infty}^{+\infty}$	\int\limits_{ -\infty }^{ +\infty }	
inti.	$\iint\limits_{C}$	\iint\limits_{ }	
intii.	$\iiint\limits_{C}$	\iiint\limits_{ }	
intiii.	$\iiiint\limits_{C}$	\iiiint\limits_{ }	
into.	$\oint\limits_{C}$	\oint\limits_{ }	

7 Structural:

7.1 Parenthesis Related

Table 24:

key	sym	latex	description
().	(□)	\left(\right)	
()..	(□|□)	\left(\middle\vert \right)	
[].	[□]	\left[\right]	
[]..	[□|□]	\left[\middle\vert \right]	(var)
[].c	⌈□⌉	\lceil \rceil	(var) (ceil)
[].f	⌊□⌋	\lfloor \rfloor	(var) (floor)
{}.	{□}	\left\{ \right\}	
{}..	{□|□}	\left\{ \middle\vert \right\}	(var)
<>.	⟨□⟩	\left< \right>	
<>..	⟨□|□⟩	\left< \middle\vert \right>	(var)
|| .	|□|	\left\vert \right\vert	
|| ..	‖□‖	\left\Vert \right\Vert	(var)
(.	(□	\left(	half (
).	□)	\right)	half)
[.	[□	\left[	half [
].	□]	\right]	half]
{.	{□	\left\{	half {
}.	□}	\right\}	half }
<.	⟨□	\left<	half <
>.	□⟩	\right>	half >
(..		\left.	half left .
)..		\right.	half right .
| .	□⎸	\Bigg\vert_{ }^{ }	definite integral range

18

7.2 Texts:

Table 25:

key	sym	latex	description
te	$\square + \text{text}$	\text{	(te)xt
tr	$\square + \mathrm{mathrm}$	\mathrm{	(t)ext (r)oman
tb	$\square + \mathbf{mathbf}$	\mathbf{	(t)ext (b)old
ti	$\square + \mathit{mathit}$	\mathit{	(t)ext (i)talics

7.3 Misc.

Table 26:

key	sym	latex	description
binom	$\binom{\square}{\square}$	\binom{}{	Binom
box	$\boxed{\square}$	\boxed{}{	Putting box around object
fr	$\frac{\square}{\square}$	\frac{}{	Fractions
can	$\cancel{\square}$	\cancel	
&=		&=\n\\\\	
=&		&=\n\\\\	

7.4 xy Diagram related

Table 27:

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

8 Formatting Table into Elisp

```python
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("~", "").replace("\\\\texttt{\\\
        key1 = repr(key1).replace("\'", "\"").replace("~", "").replace("\\\\texttt{\\\
```

19

```python
        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,  sym, trans, description = line
        key   = repr(key).replace("\'", "\"").replace("~", "").replace("\\\\texttt{\\\
        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, sym, trans, description = line
        key   = repr(key).replace("\'", "\"").replace("~", "").replace("\\\\texttt{\\\
        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_type6col("Dot", tbl_alphabet_dot_6column)
format_table_to_elisp_type6col("DDot", tbl_alphabet_ddot_6column)

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}"  ])  ("Om"      ["\\mathbf{O}"  ])
;; 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}"         ])
;; Dot
("ad"       ["\\dot{a}"         ])   ("Ad"       ["\\dot{A}"         ])
("bd"       ["\\dot{b}"         ])   ("Bd"       ["\\dot{B}"         ])
("cd"       ["\\dot{c}"         ])   ("Cd"       ["\\dot{C}"         ])
("dd"       ["\\dot{d}"         ])   ("Dd"       ["\\dot{D}"         ])
("ed"       ["\\dot{e}"         ])   ("Ed"       ["\\dot{E}"         ])
("fd"       ["\\dot{f}"         ])   ("Fd"       ["\\dot{F}"         ])
("gd"       ["\\dot{g}"         ])   ("Gd"       ["\\dot{G}"         ])
("hd"       ["\\dot{h}"         ])   ("Hd"       ["\\dot{H}"         ])
("id"       ["\\dot{i}"         ])   ("Id"       ["\\dot{I}"         ])
("jd"       ["\\dot{j}"         ])   ("Jd"       ["\\dot{J}"         ])
("kd"       ["\\dot{k}"         ])   ("Kd"       ["\\dot{K}"         ])
("ld"       ["\\dot{l}"         ])   ("Ld"       ["\\dot{L}"         ])
("md"       ["\\dot{m}"         ])   ("Md"       ["\\dot{M}"         ])
("nd"       ["\\dot{n}"         ])   ("Nd"       ["\\dot{N}"         ])
("od"       ["\\dot{o}"         ])   ("Od"       ["\\dot{O}"         ])
("pd"       ["\\dot{p}"         ])   ("Pd"       ["\\dot{P}"         ])
("qd"       ["\\dot{q}"         ])   ("Qd"       ["\\dot{Q}"         ])
("rd"       ["\\dot{r}"         ])   ("Rd"       ["\\dot{R}"         ])
("sd"       ["\\dot{s}"         ])   ("Sd"       ["\\dot{S}"         ])
("td"       ["\\dot{t}"         ])   ("Td"       ["\\dot{T}"         ])
```

```
("ud"      ["\\dot{u}"      ])  ("Ud"      ["\\dot{U}"      ])
("vd"      ["\\dot{v}"      ])  ("Vd"      ["\\dot{V}"      ])
("wd"      ["\\dot{w}"      ])  ("Wd"      ["\\dot{W}"      ])
("xd"      ["\\dot{x}"      ])  ("Xd"      ["\\dot{X}"      ])
("yd"      ["\\dot{y}"      ])  ("Yd"      ["\\dot{Y}"      ])
("zd"      ["\\dot{z}"      ])  ("Zd"      ["\\dot{Z}"      ])
;; DDot
("ad."     ["\\ddot{a}"     ])  ("Ad."     ["\\ddot{A}"     ])
("bd."     ["\\ddot{b}"     ])  ("Bd."     ["\\ddot{B}"     ])
("cd."     ["\\ddot{c}"     ])  ("Cd."     ["\\ddot{C}"     ])
("dd."     ["\\ddot{d}"     ])  ("Dd."     ["\\ddot{D}"     ])
("ed."     ["\\ddot{e}"     ])  ("Ed."     ["\\ddot{E}"     ])
("fd."     ["\\ddot{f}"     ])  ("Fd."     ["\\ddot{F}"     ])
("gd."     ["\\ddot{g}"     ])  ("Gd."     ["\\ddot{G}"     ])
("hd."     ["\\ddot{h}"     ])  ("Hd."     ["\\ddot{H}"     ])
("id."     ["\\ddot{i}"     ])  ("Id."     ["\\ddot{I}"     ])
("jd."     ["\\ddot{j}"     ])  ("Jd."     ["\\ddot{J}"     ])
("kd."     ["\\ddot{k}"     ])  ("Kd."     ["\\ddot{K}"     ])
("ld."     ["\\ddot{l}"     ])  ("Ld."     ["\\ddot{L}"     ])
("md."     ["\\ddot{m}"     ])  ("Md."     ["\\ddot{M}"     ])
("nd."     ["\\ddot{n}"     ])  ("Nd."     ["\\ddot{N}"     ])
("od."     ["\\ddot{o}"     ])  ("Od."     ["\\ddot{O}"     ])
("pd."     ["\\ddot{p}"     ])  ("Pd."     ["\\ddot{P}"     ])
("qd."     ["\\ddot{q}"     ])  ("Qd."     ["\\ddot{Q}"     ])
("rd."     ["\\ddot{r}"     ])  ("Rd."     ["\\ddot{R}"     ])
("sd."     ["\\ddot{s}"     ])  ("Sd."     ["\\ddot{S}"     ])
("td."     ["\\ddot{t}"     ])  ("Td."     ["\\ddot{T}"     ])
("ud."     ["\\ddot{u}"     ])  ("Ud."     ["\\ddot{U}"     ])
("vd."     ["\\ddot{v}"     ])  ("Vd."     ["\\ddot{V}"     ])
("wd."     ["\\ddot{w}"     ])  ("Wd."     ["\\ddot{W}"     ])
("xd."     ["\\ddot{x}"     ])  ("Xd."     ["\\ddot{X}"     ])
("yd."     ["\\ddot{y}"     ])  ("Yd."     ["\\ddot{Y}"     ])
("zd."     ["\\ddot{z}"     ])  ("Zd."     ["\\ddot{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"                 ])  ; $\perp$  ~n~ (neg)
("para"     ["\\parallel"             ])  ;
("paran"    ["\\nparallel"            ])  ; $\parallel$ ~n~  (neg)
("ang"      ["\\angle"                ])  ;
("ang."     ["\\measuredangle"        ])  ; $\angle$ ~.~ (var)
("tri"      ["\\vartriangle"          ])  ;
("trin"     ["\\triangledown"         ])  ; $\vartriangle$ ~n~ (neg)
("squ"      ["\\square"               ])  ;
("tri."     ["\\blacktriangle"        ])  ; $\vartriangle$ ~.~ (var)
("trin."    ["\\blacktriangledown"    ])  ; $\vartriangle$ ~n.~ (neg,var)
("squ."     ["\\blacksquare"          ])  ; $\square$ ~.~ (var)
;; Symbols
("inf"      ["\\infty"                ])  ;
("ex"       ["\\exists"               ])  ;
("exn"      ["\\nexists"              ])  ; $\exists$ + _n_ (neg)
("fa"       ["\\forall"               ])  ;
("hb"       ["\\hbar"                 ])  ;
("hb."      ["\\hslash"               ])  ; $\hbar$ + _._  (var)
("dd"       ["\\mathrm{d}"            ])  ;
("dd."      ["\\partial"              ])  ; $\mathrm{d}$  + _._ (var)
("ii"       ["\\imath"                ])  ;
("jj"       ["\\jmath"                ])  ;
("nab"      ["\\nabla"                ])  ;
("cm"       ["\\checkmark"            ])  ;
;; Symbols spaces
("qu"       ["\\quad"                 ])  ;
("quu"      ["\\qquad"                ])  ;
;; Symbols arrow1
("<-"       ["\\leftarrow"            ])  ;
("->"       ["\\rightarrow"           ])  ;
("-^"       ["\\uparrow"              ])  ; ~^~ looks like up arrow head
("-v"       ["\\downarrow"            ])  ; ~v~ looks like down arrow head
("<->"      ["\\leftrightarrow"       ])  ;
("<-n"      ["\\nleftarrow"           ])  ; negate (~n~) of prev. section
("->n"      ["\\nrightarrow"          ])  ; arrows + _n_
```

```
("-^n"      ["\\nuparrow"          ])  ;
("-vn"      ["\\ndownarrow"        ])  ;
("<->"      ["\\nleftrightarrow"   ])  ;
("-->"      ["\\longrightarrow"    ])  ; longer arrows, with 2 dashes
("<--"      ["\\longleftarrow"     ])  ;
("|->"      ["\\mapsto"            ])  ; vertical-bar + ~->~  (this might rendered wrong
;; Symbols arrow2
("<="       ["\\Leftarrow"         ])  ; compared to single arrrow
("=>"       ["\\Rightarrow"        ])  ; these uses ~=~ as the arrow shaft
("=^"       ["\\Uparrow"           ])  ;
("=v"       ["\\Downarrow"         ])  ;
("<=>"      ["\\Leftrightarrow"    ])  ;
("iff"      ["\\Leftrightarrow"    ])  ;
("<=n"      ["\\nLeftarrow"        ])  ; negate (~n~) of prev. section
("=>n"      ["\\nRightarrow"       ])  ; arrows + _n_
("<=>n"     ["\\nLeftrightarrow"   ])  ;
("iffn"     ["\\nLeftrightarrow"   ])  ;
("<==>"     ["\\Longleftrightarrow"])  ; longer arrows, with 2 dashes
("<=="      ["\\Longleftarrow"     ])  ;
("==>"      ["\\Longrightarrow"    ])  ;
;; Symbols arrow3
("<---"     ["\\xleftarrow[ ]{ }"  ])  ; these uses triple - or =
("--->"     ["\\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
("<."      ["\\leq"              ])   ; < = >
(">."      ["\\geq"              ])   ; symbols
("<<"      ["\\ll"               ])   ;
(">>"      ["\\gg"               ])   ;
("=n"      ["\\neq"              ])   ; negation
("n"       ["\\nsim"             ])   ;
("<n"      ["\\nless"            ])   ;
(">n"      ["\\ngtr"             ])   ;
("<.n"     ["\\nleq"             ])   ;
(">.n"     ["\\ngeq"             ])   ;
("=?"      ["\\stackrel{?}{=}"    ])   ; with question mark
("<?"      ["\\stackrel{?}{<}"    ])   ;
(">?"      ["\\stackrel{?}{>}"    ])   ;
("<.?"     ["\\stackrel{?}{\\leq}"])   ;
(">.?"     ["\\stackrel{?}{\\geq}"])   ;
("<<?"     ["\\stackrel{?}{\\ll}" ])   ;
(">>?"     ["\\stackrel{?}{\\gg}" ])   ;
("=y"      ["\\stackrel{\\checkmark}{=}"])   ; with check mark
("<y"      ["\\stackrel{\\checkmark}{<}"])   ;
(">y"      ["\\stackrel{\\checkmark}{>}"])   ;
("<.y"     ["\\stackrel{\\checkmark}{\\leq}"])   ;
(">.y"     ["\\stackrel{\\checkmark}{\\geq}"])   ;
("<<y"     ["\\stackrel{\\checkmark}{\\ll}"])   ;
(">>y"     ["\\stackrel{\\checkmark}{\\gg}"])   ;
("=."      ["\\equiv"            ])   ; Variation on
("-."      ["\\sim"             ])   ; Another variations
("=.."     ["\\approx"           ])   ; on =
("3="      ["\\equiv"            ])   ;
("=:"      ["\\coloneqq"         ])   ;
(":="      ["\\coloneqq"         ])   ;
;; Operation: arith
("in"      ["\\in"               ])   ;
("in."     ["\\ni"               ])   ;
("ni"      ["\\ni"               ])   ;
```

```
("inn"     ["\\notin"               ])  ; (neg)
("0/"      ["\\emptyset"            ])  ;
("nsr"     ["\\mathbb{R}"           ])  ; (n)umber (s)et (r)eal
("nsc"     ["\\mathbb{C}"           ])  ; (n)umber (s)et (c)omplex
("nsn"     ["\\mathbb{N}"           ])  ; ...
("nsp"     ["\\mathbb{P}"           ])  ; ...
("nsz"     ["\\mathbb{Z}"           ])  ; ...
("nsi"     ["\\mathbb{I}"           ])  ; ...
("sub"     ["\\subset"             ])  ;
("subn"    ["\\nssubseteq"         ])  ; (neg)
("sub."    ["\\subseteq"           ])  ; (var)
("sub.n"   ["\\nsubseteq"          ])  ; (var, neg)
("subn."   ["\\nsubseteq"          ])  ; (neg, var)
("sup"     ["\\supset"             ])  ;
("supn"    ["\\nsupseteq"          ])  ; (neg)
("sup."    ["\\supeseteq"          ])  ; (var)
("sup.n"   ["\\nsupseteq"          ])  ; (var, neg)
("supn."   ["\\nsupseteq"          ])  ; (neg, var)
;; Operation: arith
("or"      ["\\lor"                ])  ;
("and"     ["\\lnd"                ])  ;
("not"     ["\\neg"                ])  ;
("or."     ["\\text{ or }"         ])  ; (var)
("and."    ["\\text{ and }"        ])  ; (var)
("not."    ["\\text{ not }"        ])  ; (var)
;; 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_{ }"          ])  ;
("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_{ }^{ }"])  ;
("int.."    ["\\int\\limits_{ 0 }^{ +\\infty }"])  ;
("int..."   ["\\int\\limits_{ -\\infty }^{ +\\infty }"])  ;
("inti."    ["\\iint\\limits_{ }" ])  ;
("intii."   ["\\iiint\\limits_{ }" ])  ;
("intiii."  ["\\iiiint\\limits_{ }"])  ;
("into."    ["\\oint\\limits_{ }" ])  ;
;; Structural: Parenthesis
("()."      ["\\left( \\right)"     ])  ;
("().."     ["\\left( \\middle\\vert  \\right)"])  ;
("[]."      ["\\left[ \\right]"     ])  ;
("[].."     ["\\left[ \\middle\\vert  \\right]"])  ; (var)
("[].c"     ["\\lceil \\rceil"      ])  ; (var) (ceil)
("[].f"     ["\\lfloor \\rfloor"    ])  ; (var) (floor)
("{}."      ["\\left\\{ \\right\\}"])  ;
("{}.."     ["\\left\\{ \\middle\\vert  \\right\\}"])  ; (var)
("<>."      ["\\left< \\right>"     ])  ;
```

```
("<>.."      ["\\left< \\middle\\vert \\right>"])   ; (var)
("||."       ["\\left\\vert \\right\\vert"])   ;
("||.."      ["\\left\\Vert \\right\\Vert"])   ; (var)
("(."        ["\\left("              ])  ; half (
(")."        ["\\right)"             ])  ; half )
("[."        ["\\left["              ])  ; half [
("]."        ["\\right]"             ])  ; half ]
("{."        ["\\left\\{"            ])  ; half {
("}."        ["\\right\\}"           ])  ; half }
("<."        ["\\left<"              ])  ; half <
(">."        ["\\right>"             ])  ; half >
("(.."       ["\\left."              ])  ; half left .
(")..."      ["\\right."             ])  ; half right .
("|."        ["\\Bigg\\vert_{ }^{ }"])  ; definite integral range
;; Structural: Text
("te"        ["\\text{"              ])  ; (te)xt
("tr"        ["\\mathrm{"            ])  ; (t)ext (r)oman
("tb"        ["\\mathbf{"            ])  ; (t)ext (b)old
("ti"        ["\\mathit{"            ])  ; (t)ext (i)talics
;; Structural: Text
("te"        ["\\text{"              ])  ; (te)xt
("tr"        ["\\mathrm{"            ])  ; (t)ext (r)oman
("tb"        ["\\mathbf{"            ])  ; (t)ext (b)old
("ti"        ["\\mathit{"            ])  ; (t)ext (i)talics
;; 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{}{"          ])  ; Binom
("box"       ["\\boxed{}{"          ])  ; Putting box around object
```

30

```
("fr"      ["\\frac{}{"         ])  ; Fractions
("can"     ["\\cancel"          ])  ;
("&="      ["&=\\n\\\\\\\\\\\\" ])  ;
("=&"      ["&=\\n\\\\\\\\\\\"  ])  ;
;; Structural: xy
("xy"      ["\\xymatrix{\\n\\n}" ])  ;
("bu"      ["\\bullet"          ])  ;
("ar"      ["\\ar"              ])  ;
```

9   Executable elisp function definition

```
(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))

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

10   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))
```

```lisp
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(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))
```

33

```elisp
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(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{O}"    ])  ("Om"      ["\\mathbf{O}"    ])
;; 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}"        ])
;; Dot
("ad"      ["\\dot{a}"        ])  ("Ad"    ["\\dot{A}"        ])
("bd"      ["\\dot{b}"        ])  ("Bd"    ["\\dot{B}"        ])
("cd"      ["\\dot{c}"        ])  ("Cd"    ["\\dot{C}"        ])
("dd"      ["\\dot{d}"        ])  ("Dd"    ["\\dot{D}"        ])
("ed"      ["\\dot{e}"        ])  ("Ed"    ["\\dot{E}"        ])
("fd"      ["\\dot{f}"        ])  ("Fd"    ["\\dot{F}"        ])
("gd"      ["\\dot{g}"        ])  ("Gd"    ["\\dot{G}"        ])
("hd"      ["\\dot{h}"        ])  ("Hd"    ["\\dot{H}"        ])
("id"      ["\\dot{i}"        ])  ("Id"    ["\\dot{I}"        ])
("jd"      ["\\dot{j}"        ])  ("Jd"    ["\\dot{J}"        ])
("kd"      ["\\dot{k}"        ])  ("Kd"    ["\\dot{K}"        ])
("ld"      ["\\dot{l}"        ])  ("Ld"    ["\\dot{L}"        ])
("md"      ["\\dot{m}"        ])  ("Md"    ["\\dot{M}"        ])
("nd"      ["\\dot{n}"        ])  ("Nd"    ["\\dot{N}"        ])
("od"      ["\\dot{o}"        ])  ("Od"    ["\\dot{O}"        ])
("pd"      ["\\dot{p}"        ])  ("Pd"    ["\\dot{P}"        ])
("qd"      ["\\dot{q}"        ])  ("Qd"    ["\\dot{Q}"        ])
("rd"      ["\\dot{r}"        ])  ("Rd"    ["\\dot{R}"        ])
("sd"      ["\\dot{s}"        ])  ("Sd"    ["\\dot{S}"        ])
("td"      ["\\dot{t}"        ])  ("Td"    ["\\dot{T}"        ])
("ud"      ["\\dot{u}"        ])  ("Ud"    ["\\dot{U}"        ])
("vd"      ["\\dot{v}"        ])  ("Vd"    ["\\dot{V}"        ])
("wd"      ["\\dot{w}"        ])  ("Wd"    ["\\dot{W}"        ])
("xd"      ["\\dot{x}"        ])  ("Xd"    ["\\dot{X}"        ])
("yd"      ["\\dot{y}"        ])  ("Yd"    ["\\dot{Y}"        ])
("zd"      ["\\dot{z}"        ])  ("Zd"    ["\\dot{Z}"        ])
;; DDot
```

```
("ad."      ["\\ddot{a}"      ])  ("Ad."      ["\\ddot{A}"      ])
("bd."      ["\\ddot{b}"      ])  ("Bd."      ["\\ddot{B}"      ])
("cd."      ["\\ddot{c}"      ])  ("Cd."      ["\\ddot{C}"      ])
("dd."      ["\\ddot{d}"      ])  ("Dd."      ["\\ddot{D}"      ])
("ed."      ["\\ddot{e}"      ])  ("Ed."      ["\\ddot{E}"      ])
("fd."      ["\\ddot{f}"      ])  ("Fd."      ["\\ddot{F}"      ])
("gd."      ["\\ddot{g}"      ])  ("Gd."      ["\\ddot{G}"      ])
("hd."      ["\\ddot{h}"      ])  ("Hd."      ["\\ddot{H}"      ])
("id."      ["\\ddot{i}"      ])  ("Id."      ["\\ddot{I}"      ])
("jd."      ["\\ddot{j}"      ])  ("Jd."      ["\\ddot{J}"      ])
("kd."      ["\\ddot{k}"      ])  ("Kd."      ["\\ddot{K}"      ])
("ld."      ["\\ddot{l}"      ])  ("Ld."      ["\\ddot{L}"      ])
("md."      ["\\ddot{m}"      ])  ("Md."      ["\\ddot{M}"      ])
("nd."      ["\\ddot{n}"      ])  ("Nd."      ["\\ddot{N}"      ])
("od."      ["\\ddot{o}"      ])  ("Od."      ["\\ddot{O}"      ])
("pd."      ["\\ddot{p}"      ])  ("Pd."      ["\\ddot{P}"      ])
("qd."      ["\\ddot{q}"      ])  ("Qd."      ["\\ddot{Q}"      ])
("rd."      ["\\ddot{r}"      ])  ("Rd."      ["\\ddot{R}"      ])
("sd."      ["\\ddot{s}"      ])  ("Sd."      ["\\ddot{S}"      ])
("td."      ["\\ddot{t}"      ])  ("Td."      ["\\ddot{T}"      ])
("ud."      ["\\ddot{u}"      ])  ("Ud."      ["\\ddot{U}"      ])
("vd."      ["\\ddot{v}"      ])  ("Vd."      ["\\ddot{V}"      ])
("wd."      ["\\ddot{w}"      ])  ("Wd."      ["\\ddot{W}"      ])
("xd."      ["\\ddot{x}"      ])  ("Xd."      ["\\ddot{X}"      ])
("yd."      ["\\ddot{y}"      ])  ("Yd."      ["\\ddot{Y}"      ])
("zd."      ["\\ddot{z}"      ])  ("Zd."      ["\\ddot{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"              ])  ; $\perp$  ~n~ (neg)
("para"     ["\\parallel"          ])  ;
```

```
("paran"    ["\\nparallel"           ])  ; $\parallel$ ~n~  (neg)
("ang"      ["\\angle"               ])  ;
("ang."     ["\\measuredangle"       ])  ; $\angle$ ~.~ (var)
("tri"      ["\\vartriangle"         ])  ;
("trin"     ["\\triangledown"        ])  ; $\vartriangle$ ~n~ (neg)
("squ"      ["\\square"              ])  ;
("tri."     ["\\blacktriangle"       ])  ; $\vartriangle$ ~.~ (var)
("trin."    ["\\blacktriangledown"   ])  ; $\vartriangle$ ~n.~ (neg,var)
("squ."     ["\\blacksquare"         ])  ; $\square$ ~.~ (var)
;; Symbols
("inf"      ["\\infty"               ])  ;
("ex"       ["\\exists"              ])  ;
("exn"      ["\\nexists"             ])  ; $\exists$ + _n_ (neg)
("fa"       ["\\forall"              ])  ;
("hb"       ["\\hbar"                ])  ;
("hb."      ["\\hslash"              ])  ; $\hbar$ + _._  (var)
("dd"       ["\\mathrm{d}"           ])  ;
("dd."      ["\\partial"             ])  ; $\mathrm{d}$  + _._ (var)
("ii"       ["\\imath"               ])  ;
("jj"       ["\\jmath"               ])  ;
("nab"      ["\\nabla"               ])  ;
("cm"       ["\\checkmark"           ])  ;
;; Symbols spaces
("qu"       ["\\quad"                ])  ;
("quu"      ["\\qquad"               ])  ;
;; Symbols arrow1
("<-"       ["\\leftarrow"           ])  ;
("->"       ["\\rightarrow"          ])  ;
("-^"       ["\\uparrow"             ])  ; ~^~ looks like up arrow head
("-v"       ["\\downarrow"           ])  ; ~v~ looks like down arrow head
("<->"      ["\\leftrightarrow"      ])  ;
("<-n"      ["\\nleftarrow"          ])  ; negate (~n~) of prev. section
("->n"      ["\\nrightarrow"         ])  ; arrows + _n_
("-^n"      ["\\nuparrow"            ])  ;
("-vn"      ["\\ndownarrow"          ])  ;
("<->"      ["\\nleftrightarrow"     ])  ;
("-->"      ["\\longrightarrow"      ])  ; longer arrows, with 2 dashes
("<--"      ["\\longleftarrow"       ])  ;
("|->"      ["\\mapsto"              ])  ; vertical-bar + ~->~  (this might rendered
;; Symbols arrow2
```

```
("<="        ["\\Leftarrow"                ])  ; compared to single arrrow
("=>"        ["\\Rightarrow"               ])  ; these uses ~=~ as the arrow shaft
("=^"        ["\\Uparrow"                  ])  ;
("=v"        ["\\Downarrow"                ])  ;
("<=>"       ["\\Leftrightarrow"           ])  ;
("iff"       ["\\Leftrightarrow"           ])  ;
("<=n"       ["\\nLeftarrow"               ])  ; negate (~n~) of prev. section
("=>n"       ["\\nRightarrow"              ])  ; arrows + _n_
("<=>n"      ["\\nLeftrightarrow"          ])  ;
("iffn"      ["\\nLeftrightarrow"          ])  ;
("<==>"      ["\\Longleftrightarrow"])       ; longer arrows, with 2 dashes
("<=="       ["\\Longleftarrow"            ])  ;
("==>"       ["\\Longrightarrow"           ])  ;
;; Symbols arrow3
("<---"      ["\\xleftarrow[ ]{ }"  ])  ; these uses triple - or =
("--->"      ["\\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
("<."        ["\\leq"                      ])  ; < = >
```

```
(">."       ["\\geq"                ])  ; symbols
("<<"       ["\\ll"                 ])  ;
(">>"       ["\\gg"                 ])  ;
("=n"       ["\\neq"                ])  ; negation
("n"        ["\\nsim"               ])  ;
("<n"       ["\\nless"              ])  ;
(">n"       ["\\ngtr"               ])  ;
("<.n"      ["\\nleq"               ])  ;
(">.n"      ["\\ngeq"               ])  ;
("=?"       ["\\stackrel{?}{=}"     ])  ; with question mark
("<?"       ["\\stackrel{?}{<}"     ])  ;
(">?"       ["\\stackrel{?}{>}"     ])  ;
("<.?"      ["\\stackrel{?}{\\leq}"])  ;
(">.?"      ["\\stackrel{?}{\\geq}"])  ;
("<<?"      ["\\stackrel{?}{\\ll}" ])  ;
(">>?"      ["\\stackrel{?}{\\gg}" ])  ;
("=y"       ["\\stackrel{\\checkmark}{=}"])  ; with check mark
("<y"       ["\\stackrel{\\checkmark}{<}"])  ;
(">y"       ["\\stackrel{\\checkmark}{>}"])  ;
("<.y"      ["\\stackrel{\\checkmark}{\\leq}"])  ;
(">.y"      ["\\stackrel{\\checkmark}{\\geq}"])  ;
("<<y"      ["\\stackrel{\\checkmark}{\\ll}"])  ;
(">>y"      ["\\stackrel{\\checkmark}{\\gg}"])  ;
("=."       ["\\equiv"              ])  ; Variation on
("-."       ["\\sim"               ])  ; Another variations
("=.."      ["\\approx"             ])  ; on =
("3="       ["\\equiv"              ])  ;
("=:"       ["\\coloneqq"           ])  ;
(":="       ["\\coloneqq"           ])  ;
;; Operation: arith
("in"       ["\\in"                 ])  ;
("in."      ["\\ni"                 ])  ;
("ni"       ["\\ni"                 ])  ;
("inn"      ["\\notin"              ])  ; (neg)
("0/"       ["\\emptyset"           ])  ;
("nsr"      ["\\mathbb{R}"          ])  ; (n)umber (s)et (r)eal
("nsc"      ["\\mathbb{C}"          ])  ; (n)umber (s)et (c)omplex
("nsn"      ["\\mathbb{N}"          ])  ; ...
("nsp"      ["\\mathbb{P}"          ])  ; ...
("nsz"      ["\\mathbb{Z}"          ])  ; ...
```

```
("nsi"      ["\\mathbb{I}"          ])  ; ...
("sub"      ["\\subset"             ])  ;
("subn"     ["\\nssubseteq"         ])  ; (neg)
("sub."     ["\\subseteq"           ])  ; (var)
("sub.n"    ["\\nsubseteq"          ])  ; (var, neg)
("subn."    ["\\nsubseteq"          ])  ; (neg, var)
("sup"      ["\\supset"             ])  ;
("supn"     ["\\nsupseteq"          ])  ; (neg)
("sup."     ["\\supeseteq"          ])  ; (var)
("sup.n"    ["\\nsupseteq"          ])  ; (var, neg)
("supn."    ["\\nsupseteq"          ])  ; (neg, var)
;; Operation: arith
("or"       ["\\lor"                ])  ;
("and"      ["\\lnd"                ])  ;
("not"      ["\\neg"                ])  ;
("or."      ["\\text{ or }"         ])  ; (var)
("and."     ["\\text{ and }"        ])  ; (var)
("not."     ["\\text{ not }"        ])  ; (var)
;; 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_{ }"           ])  ;
("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_{ }^{ }"])  ;
("int.."    ["\\int\\limits_{ 0 }^{ +\\infty }"])  ;
("int..."  ["\\int\\limits_{ -\\infty }^{ +\\infty }"])  ;
("inti."    ["\\iint\\limits_{ }"  ])  ;
("intii."   ["\\iiint\\limits_{ }"  ])  ;
("intiii."  ["\\iiiint\\limits_{ }"])  ;
("into."    ["\\oint\\limits_{ }"  ])  ;
;; Structural: Parenthesis
("()."      ["\\left( \\right)"    ])  ;
("().."     ["\\left( \\middle\\vert  \\right)"])  ;
("[]."      ["\\left[ \\right]"    ])  ;
("[].."     ["\\left[ \\middle\\vert  \\right]"])  ; (var)
("[].c"     ["\\lceil \\rceil"     ])  ; (var) (ceil)
("[].f"     ["\\lfloor \\rfloor"   ])  ; (var) (floor)
("{}."      ["\\left\\{ \\right\\}"])  ;
("{}.."     ["\\left\\{ \\middle\\vert  \\right\\}"])  ; (var)
("<>."      ["\\left< \\right>"    ])  ;
("<>.."     ["\\left< \\middle\\vert \\right>"])  ; (var)
("||."      ["\\left\\vert \\right\\vert"])  ;
("||.."     ["\\left\\Vert \\right\\Vert"])  ; (var)
("(."       ["\\left("            ])  ; half (
(")."       ["\\right)"           ])  ; half )
("[."       ["\\left["            ])  ; half [
("]."       ["\\right]"           ])  ; half ]
```

```
("{."       ["\\left\\{"               ])  ; half {
("}."       ["\\right\\}"              ])  ; half }
("<."       ["\\left<"                 ])  ; half <
(">."       ["\\right>"                ])  ; half >
("(.."      ["\\left."                 ])  ; half left .
(")..      ["\\right."                 ])  ; half right .
("|."       ["\\Bigg\\vert_{ }^{ }"])  ; definite integral range
;; Structural: Text
("te"       ["\\text{"                 ])  ; (te)xt
("tr"       ["\\mathrm{"               ])  ; (t)ext (r)oman
("tb"       ["\\mathbf{"               ])  ; (t)ext (b)old
("ti"       ["\\mathit{"               ])  ; (t)ext (i)talics
;; Structural: Text
("te"       ["\\text{"                 ])  ; (te)xt
("tr"       ["\\mathrm{"               ])  ; (t)ext (r)oman
("tb"       ["\\mathbf{"               ])  ; (t)ext (b)old
("ti"       ["\\mathit{"               ])  ; (t)ext (i)talics
;; 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{}{"               ])  ; Binom
("box"      ["\\boxed{}{"               ])  ; Putting box around object
("fr"       ["\\frac{}{"                ])  ; Fractions
("can"      ["\\cancel"                 ])  ;
("&="       ["&=\\n\\\\\\\\\\\\\\\\"    ])  ;
("=&"       ["&=\\n\\\\\\\\\\\\\\\\"    ])  ;
;; Structural: xy
("xy"       ["\\xymatrix{\\n\\n}"  ])  ;
("bu"       ["\\bullet"               ])  ;
```

43

```
        ("ar"        ["\\ar"                    ])  ;
)
```