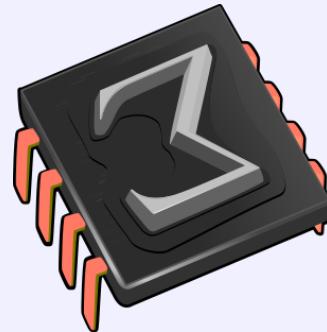


The Guile in TeXmacs



Massimiliano Gubinelli

- ▶ Visual **structured** editor: WYSWYG & WYSWYM
- ▶ Inspired by \TeX and EMACS
- ▶ High-quality typesetting algorithms (including microtypography)
- ▶ Special features for mathematical typesetting and input
- ▶ Support for interactive sessions: Scheme, Python, R, Octave, Maxima, Axiom, Mathemagix (and other CAS).
- ▶ Multi-platform: Unix, MacOS, Windows (via QT)
- ▶ Own format (XML like). Native output to PDF and PS. Export to \LaTeX , HTML
- ▶ Internal image editor, interfaces to SVN and GIT, versioning tool, database tool, encryption of documents.
- ▶ Website and documentation written in TeXmacs

Gallery

c2.tm

Buffer File Edit Insert Text Paragraph Document Project Options Help

□ ○ □ □ □ □ X | X □ □ □ □ □ □ | Σ () α \otimes \prec \rightarrow \wp | **B** *C* \mathfrak{F} **B**

Any series f in $C[[z_1; \dots; z_n]]$ may also be considered as a series in $C[[z_1]] \cdots [[z_n]]$ and we may recursively expand f as follows:

$$f = \sum_{\alpha_n \in A} f_{\alpha_n} z_n^{\alpha_n}$$
$$\vdots$$
$$f_{\alpha_n, \dots, \alpha_2} = \sum_{\alpha_1 \in A} f_{\alpha_1} z_1^{\alpha_1}.$$

Notice that $C[[z_1; \dots; z_n]] \not\subseteq C[[z_1]] \cdots [[z_n]]$, in general (see exercise 2.5).

Exercise 2.5. Show that, in general, $C[[z_1, \dots, z_n]] \not\subseteq C[[z_1; \dots; z_n]] \not\subseteq C[[z_1]] \cdots [[z_n]]$ and $C[[z_1; \dots; z_n]] \neq C[[z_{\sigma(1)}; \dots; z_{\sigma(n)}]]$, for non trivial permutations σ of $\{1, \dots, n\}$.

Exercise 2.6. Show that the definitions of this section generalize to the case

In math roman 12 example eqnarray* subscript <in>

The legacy X11 backend

TeXmacs File Edit Insert Focus Format Document Version View Go Tools Help fflogn.tm

$\sqrt{x}, \Sigma \{ \} \sim, \otimes, \prec, \succ, \neq, \Gamma, B, C, \mathfrak{F}, \mathfrak{B}, \mathfrak{o}, \mathfrak{p}, \mathfrak{D}, \mathfrak{S}, \mathfrak{E}, T, \mathfrak{C}, \mathfrak{L}, \mathfrak{R}, \mathfrak{I}, \mathfrak{H}$

Equations Cell (1.3)

$$\begin{aligned} \log x &:= \min \{k \in \mathbb{N} : \log^k x \leq 1\} \\ \log^k &:= \log \circ \dots \circ \log, \end{aligned}$$

and the bound (1.2) holds uniformly in q .

For the statement of our new results, we need to recall the concept of a “Linnik constant”. Given two integers $k > 0$ and $a \in \{1, \dots, k-1\}$ with $\gcd(a, k) = 1$, we define

$$\begin{aligned} P(a, k) &:= \min \{ck + a : c \in \mathbb{N}, ck + a \text{ is prime}\} \\ P(k) &:= \max \{P(a, k) : 0 < a < k, \gcd(a, k) = 1\}. \end{aligned}$$

We say that a number $L > 1$ is a *Linnik constant* if $P(k) = O(k^L)$. The smallest currently known Linnik constant $L = 5.18$ is due to Xylouris [57]. It is widely believed that any number $L > 1$ is actually a Linnik constant; see section 5 for more details.

In this paper, we prove the following two results:

THEOREM 1.1. Assume that there exists a Linnik constant with $L < 1 + \frac{1}{303}$. Then

$$l(n) = O(n \log n).$$

THEOREM 1.2. Assume that there exists a Linnik constant with $L < 1 + 2^{-1162}$. Then

$$M_q(n) = O(n \log q \log(n \log q)), \quad \text{uniformly in } q.$$

The QT backend, high quality typesetting

TeXmacs File Edit Insert Focus Format Document View Go Tools Help elliptic-stochastic-quant-12-arxiv-v1.tm Sat 13:15

obtaining, for any $x \in \mathbb{R}^2$, that

$$I_{\tau, \tau'}^{\pi, f_r}(0, x) \lesssim C_4^{k_1} I_{\tau_{k_2}, \tau_{k_3}}^{f_r}(0, x)$$

for some $k_1, k_2, k_3 \in \mathbb{N}$. The thesis follows from the previous inequality and the bounds obtained on $I_{\tau_{k_2}, \tau_{k_3}}^{f_r}(0, x)$. \square

Proof of Theorem 47. We write

$$L_{f_r}(t) := L(\phi_{f_r, t}(0)) \quad E_{f_r}(t) := \exp\left(4t \int_{\mathbb{R}^2} f'_r(x) V(\phi_{f_r, t}(x)) dx\right).$$

We have

$$\begin{aligned} \partial_t^k F_{f_r}^L(t) &= \sum_{0 \leq l \leq k} \binom{k}{l} \mathbb{E}\left[L_{f_r}^{(k-l)}(0) \partial_t^l \left(\frac{E_{f_r}(t)}{\mathbb{E}[E_{f_r}(t)]}\right)\right]_{t=0} \\ &= \mathbb{E}[L_{f_r}^{(k)}(0)] + \sum_{1 \leq l \leq k} \sum_{0 \leq p \leq l-1} \binom{k}{l} \binom{l}{p} (\mathbb{E}[L_{f_r}^{(k-l)}(0) \cdot E_{f_r}^{(l-p)}(0)] + \\ &\quad - \mathbb{E}[L_{f_r}^{(k-l)}(0)] \mathbb{E}[E_{f_r}^{(l-p)}(0)]) \cdot \partial_t^p \left(\frac{1}{\mathbb{E}[E_{f_r}(t)]}\right)_{t=0}, \end{aligned}$$

where we used the Leibniz rule for the derivative of the product and the relation

$$\partial_t^l \left(\frac{1}{\mathbb{E}[E_{f_r}(t)]}\right)_{t=0} = - \sum_{0 \leq p \leq l-1} \binom{l}{p} \mathbb{E}[E_{f_r}^{(l-p)}(0)] \cdot \partial_t^p \left(\frac{1}{\mathbb{E}[E_{f_r}(t)]}\right)_{t=0}.$$

Since $\partial_t^p \left(\frac{1}{\mathbb{E}[E_{f_r}(t)]}\right)_{t=0}$ is bounded from above and below when $r \rightarrow +\infty$ if we are able to prove that

math math-stix 10 render-proof eqnarray* (2,3) around around* 0

Structured editing, high quality math typesetting

TeXmacs File Edit Insert Dynamic icourse Maxima Focus Format Document View Go Tools Help 100 % Thu 17:09:07 Joris van der Hoeven semdiff19.tm

Beamer Reddish vdh icourse + Screens 4:3 Pagella 10pt ? Slide 8: Théorème de densité (cas Fuchsien) 8/19

Théorème de densité (cas Fuchsien)

Monodromie

$$\begin{pmatrix} h_1(z') \\ h_2(z') \end{pmatrix} = M_{\alpha_1} \begin{pmatrix} h_1(z) \\ h_2(z) \end{pmatrix}$$

Théorème (SCHLESINGER)

Soient $M_{\alpha_1}, \dots, M_{\alpha_s} \in \mathrm{GL}_r(\mathbb{C})$ les matrices de monodromie autour des singularités d'un opérateur Fuchsien. Soit $\mathcal{G} = \langle M_{\alpha_1}, \dots, M_{\alpha_s} \rangle$ le plus petit sous groupe algébrique de $\mathrm{GL}_r(\mathbb{C})$ qui contient $M_{\alpha_1}, \dots, M_{\alpha_s}$. Alors $\mathcal{G}_{L,h} = \mathcal{G}$.

text pagella 10 screens shown unroll

Presentation mode

Autour du calcul numérique de groupes de Galois différentiels

Joris van der Hoeven
CNRS

I

Cover up your work

Séminaire Différentiel
Palaiseau, 2 avril 2019

Le problème

$K = \mathbb{Q} \subseteq \mathbb{C}$ (ou sous corps de \mathbb{C} avec $K = \mathbb{K}$ pour certains résultats)

$L = \partial - 1$,
 $H = (e^t)^L$

$\sigma(e^t) = ae^t, \quad a \in \mathbb{K}^\times$

Exemples

$L = \partial^2 + (1+z^{-1})\partial + z^{-1}$
 $(+ d\text{dérivée} \circ d : h' + h = z^{-1})$

$h = (\frac{1}{2} + \frac{1}{2}z + \frac{2}{3}z^2 + \frac{5}{4}z^3 + \dots, e^{-z})$

$\sigma(e^z) = ae^z, \quad a \in \mathbb{K}^\times$

Exemples (suite)

Historique abrégée

1895, 1897, SCHLESINGER : théorème de densité (cas Fuchsien)

Factorisation de l'opérateur L

1894, BEKE : facteurs d'ordre un

Point non singulier $z = \infty$. On peut prendre $h = h^{(1)}$ unique telle que

$$\begin{pmatrix} h_1(a) & \cdots & h_r(a) \\ h_1^{(r-1)}(a) & \cdots & h_r^{(r-1)}(a) \end{pmatrix} = \begin{pmatrix} 1 & & 0 \\ 0 & \ddots & \\ & & 1 \end{pmatrix}.$$

Solutions formelles autour d'une singularité

Théorème de densité (cas Fuchsien)

Monodromie

Théorème de densité (cas général)

Matrices exponentielles

Induites par automorphismes σ exponentiels $\sigma(e^{p(\sqrt[3]{t})}) = \lambda e^{p(\sqrt[3]{t})}$

Théorème de densité (cas général, suite)

Accéléro-summation d'échelle

$\begin{matrix} h \\ h_1 \\ \vdots \\ h_n \end{matrix} \xrightarrow{\lambda_{0 \rightarrow 1}} \begin{matrix} h_2 \\ \vdots \\ h_{n-1} \\ h_n \end{matrix} \xrightarrow{\lambda_{n-1 \rightarrow n}} \dots \xrightarrow{\lambda_{0 \rightarrow n}} \begin{matrix} h \\ h_1 \\ \vdots \\ h_n \end{matrix}$

Nombres holonomes

Théorème (CHUDNOVSKY², VAN DER HOEVEN)

Pour $K = \mathbb{Q}$ et en prenant un point de base non singulier dans \mathbb{K} , on peut approximer en temps $O(n \log^2 n)$ les matrices de monodromie de L autour de chaque singularité avec une erreur ϵ au plus 2^{-n} .

Factorisation de L

1. Calculer des générateurs $M_1, \dots, M_m \in \mathrm{GL}_n(\mathbb{C}^{\otimes k})$ de $\langle g_L \rangle$

2. Trouver une précision ϵ de calcul pour les tests à zéro.

3. Déterminer un sous espace invariant V non trivial pour M_1, \dots, M_m

4. Si un tel espace V n'existe pas, alors renvoyer V .

5. À partir de V , reconstruire une factorisation $L = L_1 \oplus L_2$ candidat.

6. Si $\|L - L_1\|_F < \epsilon$, alors renvoyer (V, L_1) .

7. Doubler la précision et renvoyer à l'étape 3

Zoom sur l'étape 3

1. Calculer des générateurs $M_1, \dots, M_m \in \mathrm{GL}_n(\mathbb{C}^{\otimes k})$ de $\langle g_L \rangle$

2. Fixer une précision ϵ de calcul pour les tests à zéro.

3. Déterminer un sous espace invariant V non trivial pour M_1, \dots, M_m .

4. Si un tel espace V n'existe pas, alors renvoyer V .

5. À partir de V , reconstruire une factorisation $L = L_1 \oplus L_2$ candidat.

6. Si $\|L - L_1\|_F < \epsilon$, alors renvoyer (V, L_1) .

7. Doubler la précision et renvoyer à l'étape 3

Groupe Galois différentiel

Idée : calculer $\langle f \rangle$ comme variété sous la forme

$\langle f \rangle = \mathcal{T}e^{\mathcal{L}}$ ($\forall N \in \mathcal{T}, \mathcal{N}e^{\mathcal{L}} = e^{\mathcal{L}}N$)

\mathcal{T} : ensemble fini contenu \mathbb{N}

\mathcal{L} : algèbre de Lie donnée par une base

Groupe Galois différentiel : l'algorithme

Étape 1. [Initialisation]

Calculer $(M_i) = \mathcal{T}e^{\mathcal{L}^i}$ pour tout $i \in \{1, \dots, m\}$

$\mathcal{T} := \mathcal{T}_1 \cup \dots \cup \mathcal{T}_m$

$L := \mathrm{Lie}(\mathcal{L}_1 \dashv \dots \dashv \mathcal{L}_m)$

Étape 2. [Clôture]

Tant qu'il existe un $N \in \mathcal{T} \setminus \{1\}$ avec $N.L.N^{-1} \not\subseteq \mathcal{L}$

$\mathcal{L} := \mathrm{Lie}(\mathcal{L} + N.L.N^{-1})$

Tant qu'il existe un $N \in \mathcal{T} \setminus \{1\}$ avec $N \in e^{\mathcal{L}}$, faire $\mathcal{T} := \mathcal{T} \setminus \{N\}$

Tant qu'il existe un $N \in \mathcal{T}^2$ avec $N \notin \mathcal{T}e^{\mathcal{L}}$ faire

 Calculer $\langle N \rangle = \mathcal{T}e^{\mathcal{L}^N}$.

 Si $\mathcal{L}' \not\subseteq L$, alors $L := \mathrm{Lie}(\mathcal{L} + \mathcal{L}')$, quitter la boucle, répéter l'étape 2

 Sinon, $\mathcal{T} := \mathcal{T} \cup \{N\}$

Retourner $\mathcal{T}e^{\mathcal{L}}$

Calculs plus rapides avec la partie discrète

Représentation compacte des éléments dans $\mathcal{H} = \mathcal{G}/e^{\mathcal{L}}$

- Réduire au cas où $\mathcal{G} \subseteq \mathrm{Norm}(e^{\mathcal{L}})^0$
- Premier élément $M = B_1 + e^{\mathcal{L}}$ de la base avec
 - $M e^{\mathcal{L}}$ est dans $\langle e^{\mathcal{L}} \cap \mathcal{G} \rangle$
 - $M e^{\mathcal{L}}$ génère $\langle e^{\mathcal{L}} \cap \mathcal{G} \rangle/e^{\mathcal{L}}$
 - M admet un ordre q maximal avec ces propriétés
- Posons $H' := (N \in \mathcal{H} : (M, N) = 0)$, $\mathcal{L}' := \mathcal{L} \oplus C(X, \mathcal{L})$, tels que
 - $H = (1, \dots, M^{q-1}) / (H' / e^{\mathcal{L}'})$.

Réduction de réseaux non commutative

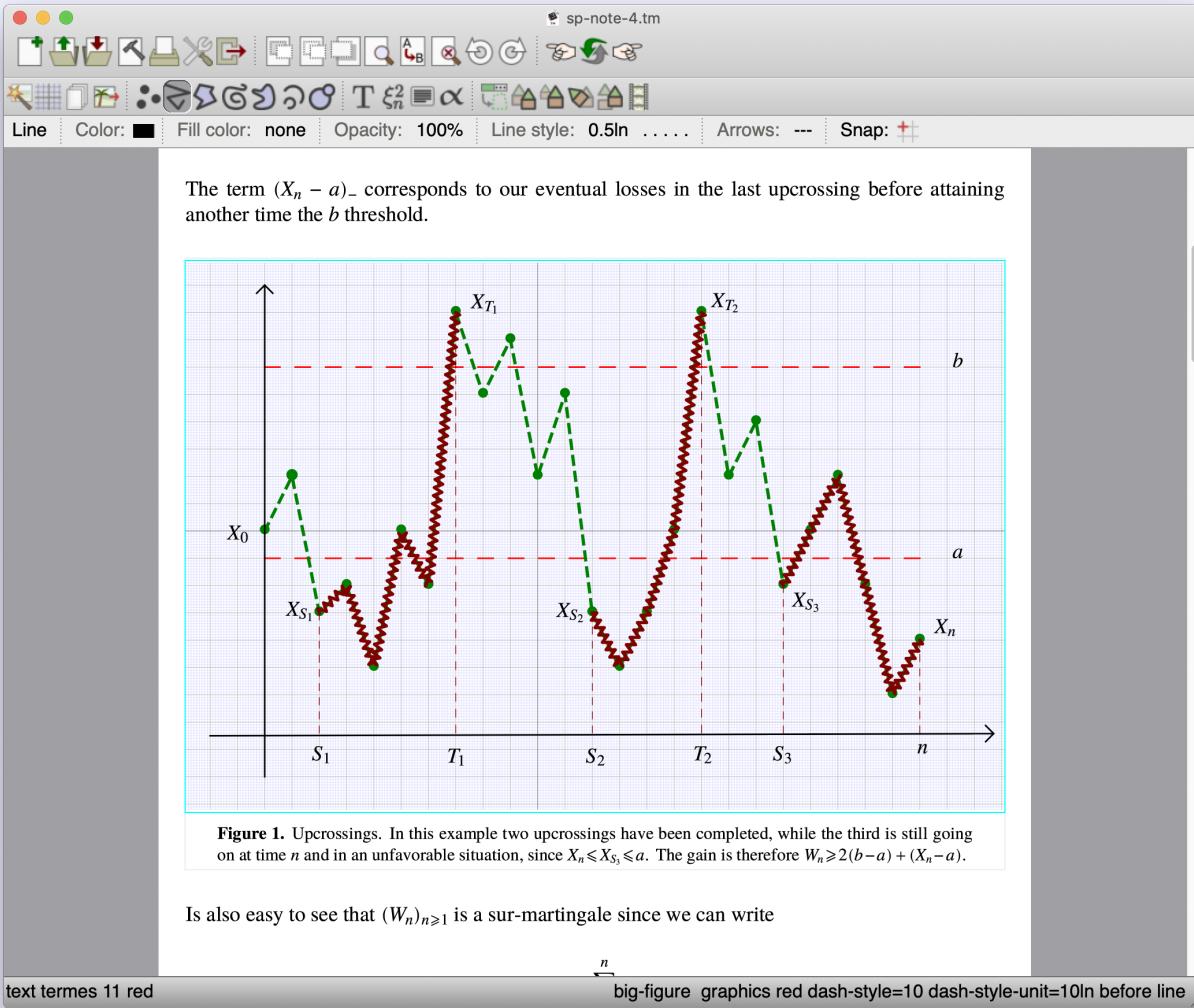
B_1, \dots, B_q

- Si $\{B_1, B_2\} = 0$, alors réduire en utilisant LLL.
- Autres éléments B_3, \dots, B_q de la base par récurrence, avec $\|B_i\|_{\mathcal{L}'} \leq \dots \leq \|B_1\|_{\mathcal{L}'}$

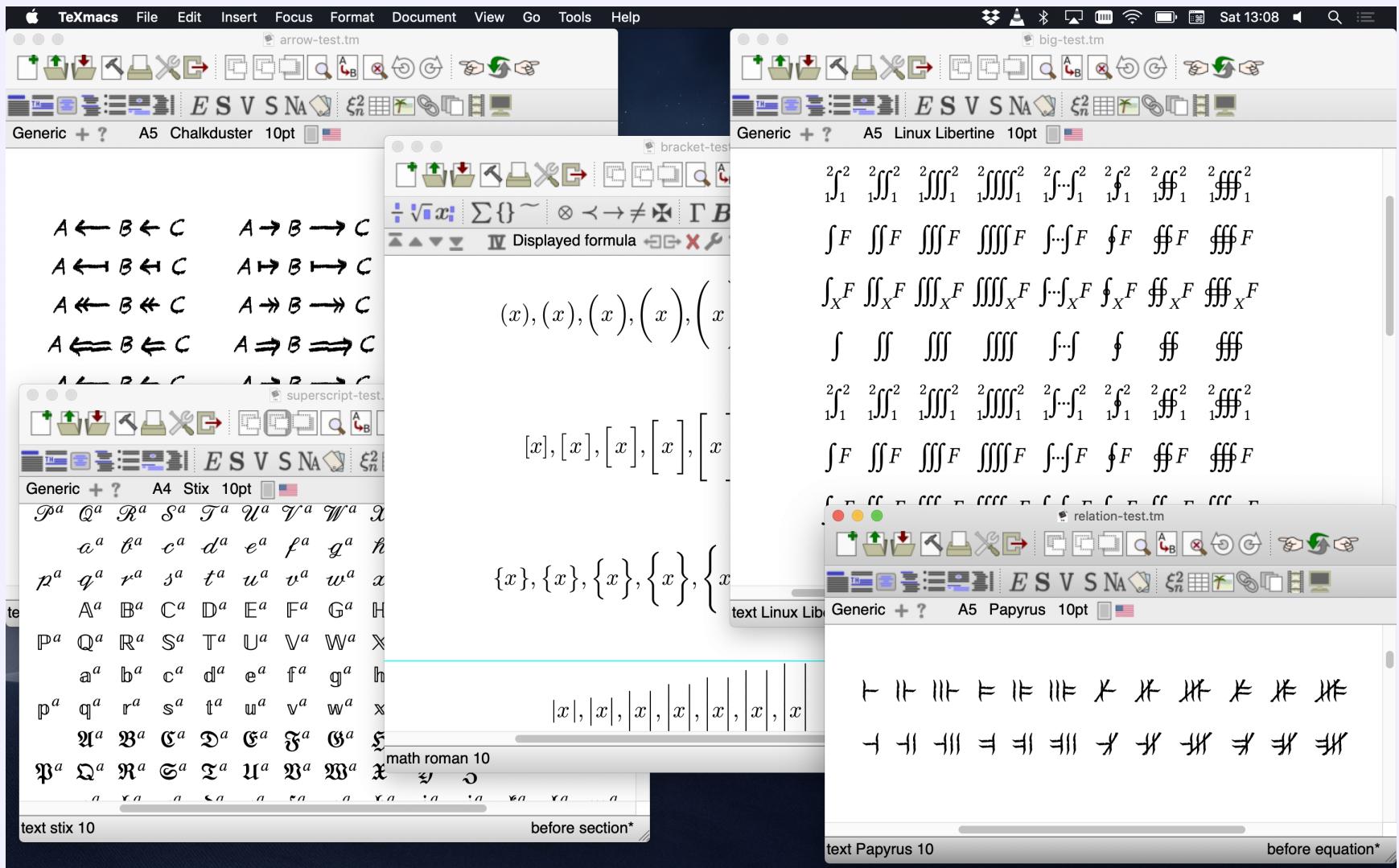
Merci !

<http://www.Texeucs.org>

Panorama mode



Graphics editor



Microtypography, synthetic math fonts

TeXmacs File Edit Insert Focus Format Document View Go Tools Help dratex.tm Sat 13:16

I B S T ξⁿ Dratex GO STOP

TeXmacs interface to (Al)DraTeX (High Level Drawing Facilities)

```
DraTex] \Draw
  \Tree()
  2,atom //
  0,electrons & 2,nucleus //
  0,protons & 0,neutrons //
\EndDraw
```

atom

electrons nucleus

 protons neutrons

```
DraTex] \Draw
  \TreeAlign(H,0,0)(0,0,0)
  \TreeSpace(S,5,15)
  \Tree()
  2,European //
  3,Latin & 2,Celtic //
  0,French & 0,Italian & 0,Spanish & 0,Irish & 0,Welsh //
\EndDraw
```

European

 Latin French

 Celtic Italian

 Spanish

 Irish

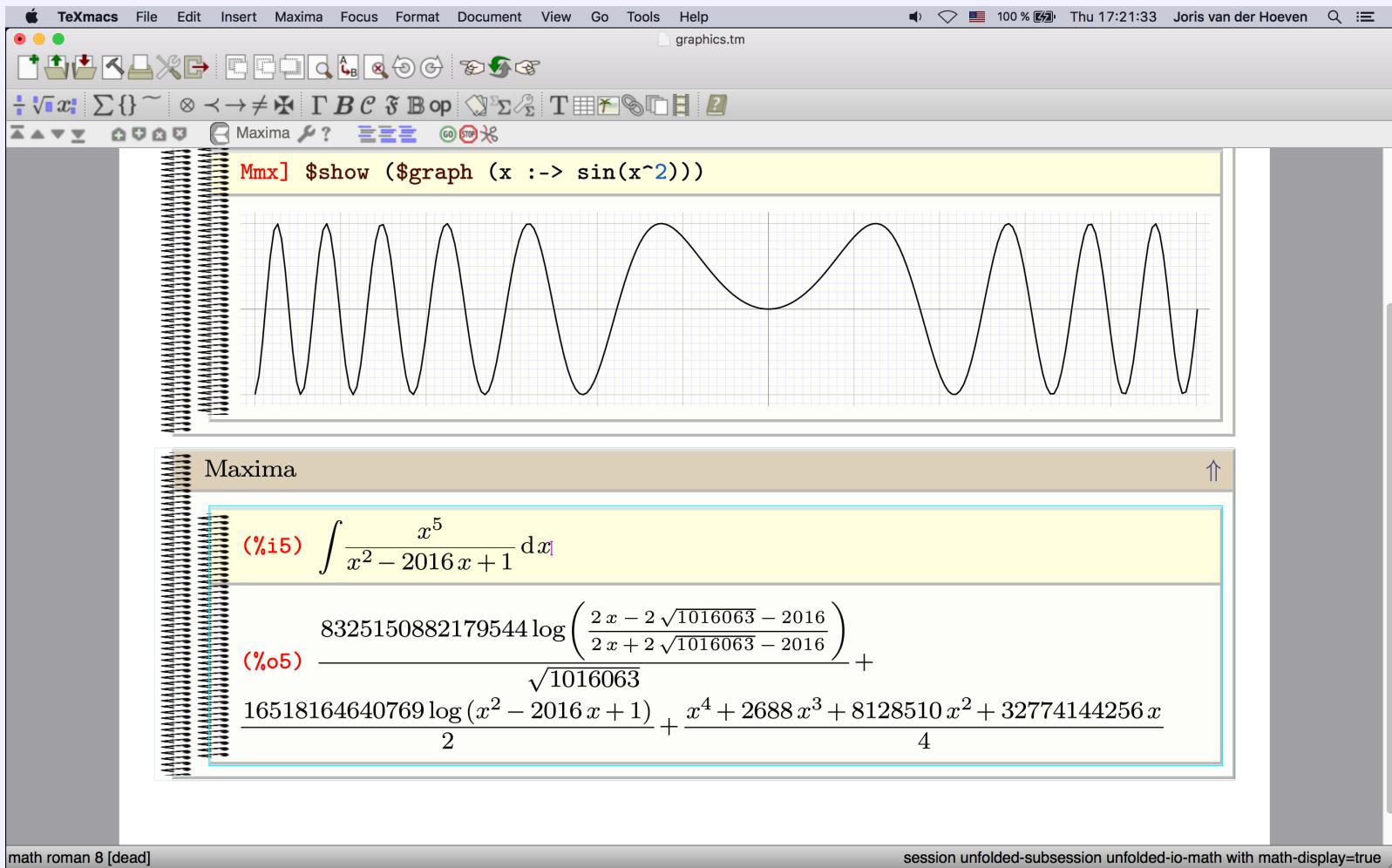
 Welsh

```
DraTex] I
```

program roman 8 [dead]

session input start

Interfaces to external packages (here DraTeX)



Interfaces to external packages (here MATHEMAGIX and MAXIMA)

TeXmacs File Edit Insert Focus Format Document View Go Tools Help

Help - 数学公式

Tmdoc + ? A4 Sys-chinese 10pt

插入→数学→多行公式或者 Alt+&.

这个条目创建了一个 eqnarray*，一个三列宽的类似表格的环境（详见创建表格）。在这个环境中可以方便地输入多行公式，比如：

$$\begin{aligned} x + 0 &= x \\ x + (-x) &= 0 \\ x + y &= y + x \\ (x + y) + z &= x + (y + z) \end{aligned}$$

第一列右对齐，第二列居中，第三列左对齐。 eqnarray* 环境的典型用途就是显示多步计算：

$$\begin{aligned} (e^{\sin x} + \sin e^x)' &= (e^{\sin x})' + (\sin e^x)' \\ &= (\sin x)' e^{\sin x} + (e^x)' \cos e^x \\ &= e^{\sin x} \cos x + e^x \cos e^x, \end{aligned}$$

在这个例子中，大多数行的第一列是空着的。

2. 输入数学符号

使用 $\text{\textasciitilde} F7$ 输入希腊字母，比如， $\text{\textasciitilde} F7 A$ 输入 α ， $\text{\textasciitilde} F7 \text{\textasciitilde} G$ 输入 Γ 。类似地，F6，F7，F8 和 $\text{\textasciitilde} F6$ 可用来输入粗体，毛体，毛体和黑板粗体。例如 $\text{\textasciitilde} F8 M$ 得到 M ， $\text{\textasciitilde} F6 \text{\textasciitilde} G$ 得到 Γ ， $\text{\textasciitilde} F6 F7$ 得到 α 。

text sys-chinese 10

before tmdoc-title

Support for oriental scripts

- ▶ Started in 1998 by JORIS VAN DER HOEVEN
 - v0.2.3β released 26 Oct 1999
 - v1.0 (2002)
 - QT backend in v1.0.7 (2008)
 - native PDF support in v1.99.1 (2013)
 - currently version 1.99.9 (soon 2.1)
- ▶ Written in C++ (~300.000 loc) and SCHEME (~150.000 loc) (from [\[openhub\]](#)).
- ▶ Fully modular, external dependencies (mostly) isolated via tight interfaces.
- ▶ Two UI backends: legacy X11 with custom widget library, modern QT backend (cross-platform support).
- ▶ **GNU Guile as extension language.** C++ export basic manipulation routines and few internal datatypes.



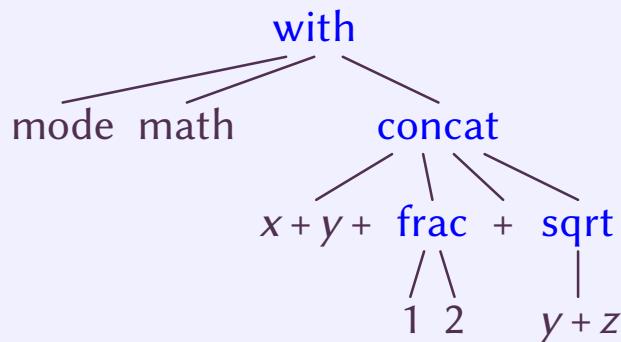
Some of the (current) developers

TeXmacs' content model

16/38

All TeX_{MACS} documents or document fragments can be thought of as *trees*.

For instance, the tree



typically represents the formula

$$x + y + \frac{1}{2} + \sqrt{y + z}$$

External representations

Serialization of TeXmacs documents without loss of informations

- $\text{\TeX}_{\text{MACS}}$ format

```
<with|mode|math|x+y+<frac|1|2>+<sqrt|y+z>>
```

- XML format

```
<frac><tm-arg>1</tm-arg><tm-arg>2</tm-arg></frac>+<sqrt>y+z</sqrt>
```

- SCHEME format

```
(with "mode" "math" (concat "x+y+" (frac "1" "2") "+" (sqrt "y+z"))))
```

Typesetting process converts TeXmacs trees into boxes:



The **typesetting primitives** are designed to be very fast and they are built-in into the editor:

e.g. typesetting primitives for horizontal concatenations (`concat`), page breaks (`page-break`), mathematical fractions (`frac`), hyperlinks (`hlink`), and so on.

The rendering of many of the primitives may be customized through the **built-in environment variables**.

e.g. the environment variable `color` specifies the current color of objects, `par-left` the current left margin of paragraphs, etc.

The **stylesheet language** allows the user to write new primitives (macros) on top of the built-in primitives.

Contains primitives for defining macros, conditional statements, computations, delayed execution, etc. and a special `extern` tag to inject SCHEME expressions in order to write macros.

Evaluation of TeXmacs trees proceeds by reduction of the primitives, essentially by evaluation of macro applications.

```
<assign|hello|<macro|name|Hello name, how are you today?>>
```

Macros have editable input fields. Examples here below (activate the macros):

```
<assign|hello|<macro|name|Hello name, how are you today?>>
```

```
<hello|dsdjskjds>
```

```
<assign|seq|<macro|val|(val1, ..., valn)>>
```

$$\langle \text{seq} | f \rangle = \langle \text{seq} | g \rangle$$

TeXmacs is extendable and customizable in various ways:

- ▶ **GUILE** embedded as extension and scripting language
- ▶ A plugin system allows asynchronous communication with external programs
- ▶ Mechanism to dynamically load external code (via C interface)

GUILE is easy to embed and provides a reasonably fast implementation of SCHEME.

Why SCHEME?

1. Allows to mix programs and data in a common framework.
2. Allows to customize the language itself, by adding new programming constructs.
3. Allows to write programs on a very abstract level.

```
(menu-bind file-menu
  ("New" (new-buffer))
  ("Load" (choose-file load-buffer "Load file" ""))
  ("Save" (save-buffer))
  ...)
```

can be easily extended from user code:

```
(menu-bind insert-menu
  (former)
  ---
  (-> "Opening"
    ("Dear Sir" (insert "Dear Sir,"))
    ("Dear Madam" (insert "Dear Madam,")))
  (-> "Closing"
    ("Yours sincerely" (insert "Yours sincerely,"))
    ("Greetings" (insert "Greetings,"))))
```

Keybindings

```
(kbd-map
  ("D e f ." (make 'definition))
  ("L e m ." (make 'lemma))
  ("P r o p ." (make 'proposition))
  ("T h ." (make 'theorem)))
```

The file `my-init-buffer.scm` is executed every time a buffer is loaded, it allows some specific customizations. For example:

```
(if (not (buffer-has-name? (current-buffer)))
  (begin
    (init-style "article")
    (buffer-pretend-saved (current-buffer)))))

(if (not (buffer-has-name? (current-buffer)))
  (make-session "maxima" (url->string (current-buffer))))
```

Scheme invocation

SCHEME commands can be invoked interactively (like in EMACS) using the `⌘^X` shortcut.

A SCHEME session is started using the Insert→Session→Scheme menu item:

```
scheme] (define (square x) (* x x))
scheme] (square 1111111)
scheme] (kbd-map ("h i ." (insert "Hi there!")))
scheme] ;; try typing `hi.'
```

SCHEME commands can be invoked from the command line:

```
texmacs text.tm -x "(print)" -q
```

Or scheme statement executed from inside TeXmacs macros:

```
<extern|lambda (x) `(concat "Hallo " ,x))|Piet>
```

Scheme invocation

SCHEME commands can be invoked interactively (like in EMACS) using the `⌘^X` shortcut.

A SCHEME session is started using the Insert→Session→Scheme menu item:

```
scheme] (define (square x) (* x x))
```

```
scheme] (square 1111111)
```

```
scheme] (kbd-map ("h i ." (insert "Hi there!")))
```

```
scheme] ;; try typing `hi.'
```

SCHEME commands can be invoked from the command line:

```
texmacs text.tm -x "(print)" -q
```

Or scheme statement executed from inside TeXmacs macros:

```
<extern|(\lambda (x) `(concat "Hallo " ,x))|Piet>
```

Scheme invocation

23/38

SCHEME commands can be invoked interactively (like in EMACS) using the `⌘^X` shortcut.

A SCHEME session is started using the Insert→Session→Scheme menu item:

```
scheme] (define (square x) (* x x))
```

```
scheme] (square 1111111)
```

```
scheme] (kbd-map ("h i ." (insert "Hi there!")))
```

```
scheme] ;; try typing `hi.'
```

SCHEME commands can be invoked from the command line:

```
texmacs text.tm -x "(print)" -q
```

Or scheme statement executed from inside TeXmacs macros:

```
<extern|lambda (x) `(concat "Hallo " ,x))|Piet>
```

Scheme invocation

SCHEME commands can be invoked interactively (like in EMACS) using the `⌘^X` shortcut.

A SCHEME session is started using the Insert→Session→Scheme menu item:

```
scheme] (define (square x) (* x x))
```

```
scheme] (square 1111111)
```

```
scheme] (kbd-map ("h i ." (insert "Hi there!")))
```

```
scheme] ;; try typing `hi.'
```

SCHEME commands can be invoked from the command line:

```
texmacs text.tm -x "(print)" -q
```

Or scheme statement executed from inside TeXmacs macros:

```
<extern|lambda (x) `(concat "Hallo " ,x))|Piet>
```

Scheme invocation

23/38

SCHEME commands can be invoked interactively (like in EMACS) using the `⌘^X` shortcut.

A SCHEME session is started using the Insert→Session→Scheme menu item:

```
scheme] (define (square x) (* x x))
```

```
scheme] (square 1111111)
```

```
scheme] (kbd-map ("h i ." (insert "Hi there!")))
```

```
scheme] ;; try typing `hi.'
```

SCHEME commands can be invoked from the command line:

```
texmacs text.tm -x "(print)" -q
```

Or scheme statement executed from inside TeXmacs macros:

```
<extern|lambda (x) `(concat "Hallo " ,x))|Piet>
```

Contextual overloading

Function definition can depend on several run-time conditions (e.g. editor mode). This allows to develop modular user interfaces.

```
(tm-define (hello) (insert "Hello"))
(tm-define (hello) (:require (in-math?)) (insert-go-to "hello()" '(6)))
```

```
(tm-define (hello)
  (if (in-math?) (insert-go-to "hello()" '(6)) (former)))
```

```
(tm-define (my-replace what by)
  default-implementation)
```

```
(tm-define (my-replace what by)
  (:require (== what by))
  (noop))
```

Meta informations

25/38

```
(tm#define (square x)
  (:synopsis "Compute the square of @x")
  (:argument x "A number")
  (:returns "The square of @x")
  (* x x))
```

Used via e.g. (`help square`). Allows for interactive input of parameters: typing `⌘^↑X` followed by `square` and `↶` and you will be prompted for “A number” on the footer (or in a dialog). Tab-completion.

```
(tm#property (choose-file fun text type)
  (:interactive #t))
```

to indicate interactive commands in menu items like:

```
("Load" (choose-file load-buffer "Load file" ""))
```

Check-marks for menu items:

```
(tm-define (toggle-session-math-input)
  (:check-mark "v" session-math-input?)
  (session-use-math-input (not (session-math-input?))))
```

```
(tm-define mouse-unfold
  (:secure #t)
  (with-action t
    (tree-go-to t :start)
    (fold)))
```

- This is a fold/unfold environment

Check-marks for menu items:

```
(tm-define (toggle-session-math-input)
  (:check-mark "v" session-math-input?)
  (session-use-math-input (not (session-math-input?))))
```

```
(tm-define mouse-unfold
  (:secure #t)
  (with-action t
    (tree-go-to t :start)
    (fold)))
```

- This is a **fold/unfold** environment

It allows to toggle the display of its content by switching the tag from **fold** to **unfold** and back.

- **Passive trees** (stree)

$$\frac{a^2}{b+c}$$

is typically represented by

```
(frac (concat "a" (rsup "2")) "b+c")
```

convenient to manipulate content directly using standard SCHEME routines on lists.

- **Active trees** (tree). $\text{\TeX}_{\text{MACS}}$ internal C++ type tree which is exported to SCHEME via the glue. Keeps track of the *position* of the tree inside the global document tree and can be used to programmatically modify documents.
- **Hybrid representation** (content). an expression of the type content is either a string, a tree or a list whose first element is a symbol and whose remaining elements are other expressions of type content.

```
scheme] (tree-set! t '(document "First line." "Second line."))
scheme] (tree-set t 1 "New second line.")
scheme] (tree-set t 0 '(strong ,(tree-ref t 0)))
```

A full example

28/38

```
(tm#define (swap-numerator-denominator t)
  (:require (tree-is? t 'frac))
  (with p (tree-cursor-path t)
    (tree-set! t '(frac ,(tree-ref t 1) ,(tree-ref t 0)))
    (tree-go-to t (cons (- 1 (car p)) (cdr p)))
    (tree-focus t)))
```

To be called as (swap-numerator-denominator (focus-tree)), or just add it as a structured variant to **frac**

```
(tm#define (variant-circulate t forward?)
  (:require (tree-is? t 'frac))
  (swap-numerator-denominator t))
```

Regular expressions

29/38

T_EX_{MACS} implements the routines `match?` and `select` for matching regular expressions and selecting subexpressions along a “path”. For instance: in the current buffer search all expressions of the form

$$\frac{a}{1 + \sqrt{b}}$$

where *a* and *b* are general expressions:

```
Scheme] (select (buffer-tree) '(:* (:match (frac :%1 (concat "1+" (sqrt :%1))))))
```

User preferences

```
(define-preferences
  ("Gnu's hair color" "brown" notify-gnu-hair-change)
  ("Snail's cruising speed" "1mm/sec" notify-Achilles))
```

New data formats and converters

```
(define-format blablah
  (:name "Blablah")
  (:suffix "bla"))

(converter blablah-file latex-file
  (:require (url-exists-in-path? "bla2tex"))
  (:shell "bla2tex" from ">" to))
```

When a format can be converted from or into $\text{\TeX}_{\text{MACS}}$, then it will automatically appear into the File→Export and File→Import menus. Similarly, when a format can be converted to POSTSCRIPT, then it also becomes a valid format for images. $\text{\TeX}_{\text{MACS}}$ also attempts to combine explicitly declared converters into new ones.

Dialogues

```
Scheme] (user-ask "First number:"
  (lambda (a)
    (user-ask "Second number:"
      (lambda (b)
        (set-message (number->string (* (string->number a)
                                         (string->number b))))
        "product")))))
```

Widgets

```
Scheme] (tm-widget (example3)
  (hlist
    (bold (text "Hello"))
    >>>
    (inert (explicit-buttons ("world" (display "\n"))))))
```

```
Scheme] (top-window example3 "Some text")
```

```
Scheme]
```

tree-view

32/38

```
Scheme] (define t
  (stree->tree
    '(root
      (library "Library" "$TEXMACS_PIXMAP_PATH/tm_german.xpm" 01
        (collection "Cool stuff" 001)
        (collection "Things to read" 002)
        (collection "Current work" 003
          (collection "Forever current" 004)
          (collection "Read me" 005))))))
```

```
Scheme] (define dd
  (stree->tree
    '(list (library DisplayRole DecorationRole UserRole:1)
      (collection DisplayRole UserRole:1))))
```

```
Scheme] (define (action clicked cmd-role . user-roles)
  (display* "clicked= " clicked ", cmd-role= " cmd-role
    ", user-roles= " user-roles "\n")))
```

```
Scheme] (tm-widget (widget-library)
  (resize ("150px" "400px" "9000px") ("300px" "600px" "9000px")
    (vertical
      (bold (text "Testing tree-view")))
      ===
      (tree-view action t dd))))
```

```
Scheme] (top-window widget-library "Tree View")
```

```
Scheme]
```

Forms

```

Scheme] (tm-widget (form3 cmd)
  (resize "500px" "500px"
  (padded
    (form "Test"
      (aligned
        (item (text "Input:")
          (form-input "fieldname1" "string" '("one") "1w"))
        (item === ===)
        (item (text "Enum:")
          (form-enum "fieldname2" '("one" "two" "three") "two" "2w"))
        (item === ===)
        (item (text "Choice:")
          (form-choice "fieldname3" '("one" "two" "three") "one"))
        (item === ===)
        (item (text "Choices:")
          (form-choices "fieldname4"
            '("one" "two" "three")
            '("one" "two")))))
      (bottom-buttons
        ("Cancel" (cmd "cancel")) >>
        ("Ok"
          (display* (form-fields) " -> " (form-values) "\n")
          (cmd "ok"))))))))

```

```

Scheme] (dialogue-window form3 (lambda (x) (display* x "\n")) "Test of form3")

```

```

Scheme]

```

Bibliography styles

New styles can be defined via SCHEME modules like `example.scm` defined as follows:

```
(texmacs-module (bibtex example)
  (:use (bibtex bib-utils)))

(bib-define-style "example" "plain")

(tm-define (bib-format-date e)
  (:mode bib-example?)
  (bib-format-field e "year"))
```

This example style behaves in a similar way as the plain style, except that all dates are formatted according to our custom routine. Styles are stored in `$TEXMACS_PATH/progs/bibtex` and referred to as e.g. `tm-example` (for when used in a $\text{\TeX}_{\text{MACS}}$ document).

Graphics

Graphics objects are also part of the TeXmacs format and can be manipulated programmatically from Scheme.

Actually, part of the graphics editor is written in Scheme.

```
Scheme] (stree->tree
  '(with gr-geometry (tuple "geometry" "200px" "100px" "center")
      color "blue"
      (graphics (text-at "TeXmacs" (point "-2.5" "-1"))
                (point 0 -1)
                (line (point 0 0) (point 0 1)
                      (point 1 1) (point 1 0) (point 0 0)))))
```

```
Scheme]
```

Graphics

Graphics objects are also part of the TeXmacs format and can be manipulated programmatically from Scheme.

Actually, part of the graphics editor is written in Scheme.

```
Scheme] (stree->tree
  '(with gr-geometry (tuple "geometry" "200px" "100px" "center")
      color "blue"
      (graphics (text-at "TeXmacs" (point "-2.5" "-1"))
                (point 0 -1)
                (line (point 0 0) (point 0 1)
                      (point 1 1) (point 1 0) (point 0 0))))))
```



TeXmacs ●

```
Scheme]
```

Many improvements ahead

- ▶ Version 2.1 to be released soon
- ▶ Update the backend to QT 5 (currently QT 4.8) [almost there]
- ▶ Adapt the scheme code to run on GUILE 3. (currently GUILE 1.8) [WIP]
- ▶ New website, documentations, videos [WIP]
- ▶ JUPYTER plugins (protocol to interface to many computational kernels, e.g. PYTHON, JULIA, R, HASKELL, GUILE, ...)
- ▶ Improvements to the styling of presentations and posters [WIP]
- ▶ More documentation, more tutorial, grow community [Stackexchange proposal]
- ▶ Collaboration tools
- ▶ Bibliography management with ZOTERO

Many opportunities for contributions for all tastes

- ▶ From the outside

- ▶ Write and review documentations, tutorials, videos, improve community, advertise
- ▶ Develop plugins to your preferred system or to add your preferred feature, e.g.: literate programming tools with beautiful output
- ▶ Write new document styles, templates, presentation styles, poster styles
- ▶ Font tuning

- ▶ Hack the C++ code

- ▶ Understand the code and write developer documentation
- ▶ Improve the QT backend, fix bugs, add features, improve stability, better image handling and PDF export of TeXmacs features
- ▶ Write new backends (COCOA), port to tablets or to the browser

- ▶ Hack SCHEME

- ▶ Help porting to GUILE 3, improve speed
- ▶ fix bugs, review code, add new cool features

Happy TeXmacsing!