OpenType Math Illuminated

Dr. Ulrik Vieth

Stuttgart, Germany

BachoTEX 2009

Developments in text typesetting

- Major trends in publishing
  - support for Unicode character sets
  - support for OpenType font technology
- Major developments in the TEX community
  - new TEX engines: XƎTEX, LuaTEX
  - new TEX fonts: Latin Modern, TEX Gyre
- Outside developments
  - OpenType supported by operating systems or libraries
  - OpenType supported by typesetting software
  - OpenType supported by commercial font suppliers
  - OpenType as a replacement for TrueType and Type 1

Developments in math typesetting

- Unicode math
  - encoding for math symbols and alphabets
  - developed by working group (input from STIX, AMS)
  - standard since 2001 (UTR#25 for Unicode 3.2)
- OpenType math
  - extension of OpenType font format
  - developed by Microsoft as a vendor-controlled format
  - officially *experimental*, but already *de facto* standard
  - first implemented in MS Office 2007
  - supported by reference fonts: Cambria Math
  - supported by font editors and tools: FontForge
  - supported by new TEX engines: XƎTEX, LuaTEX

Overview of OpenType math

- OpenType font format
  - extensible table structure (as in TrueType)
  - different flavors of font outlines (TrueType vs. CFF)
  - some tables required, e.g. glyph metrics, outlines
  - some tables optional, e.g. advanced typographic features
  - additions for OpenType math: new optional MATH table
- OpenType MATH table
  - global font parameters (similar to fontdimens of Appendix G)
  - variants and constructions (similar to charlists and extensibles)
  - additions to glyph metrics (similar to overloaded TFM fields)

Interactive Demo (I)

- Interactive Demo
  - open Cambria Math in FontForge
  - inspect parameters of MATH table

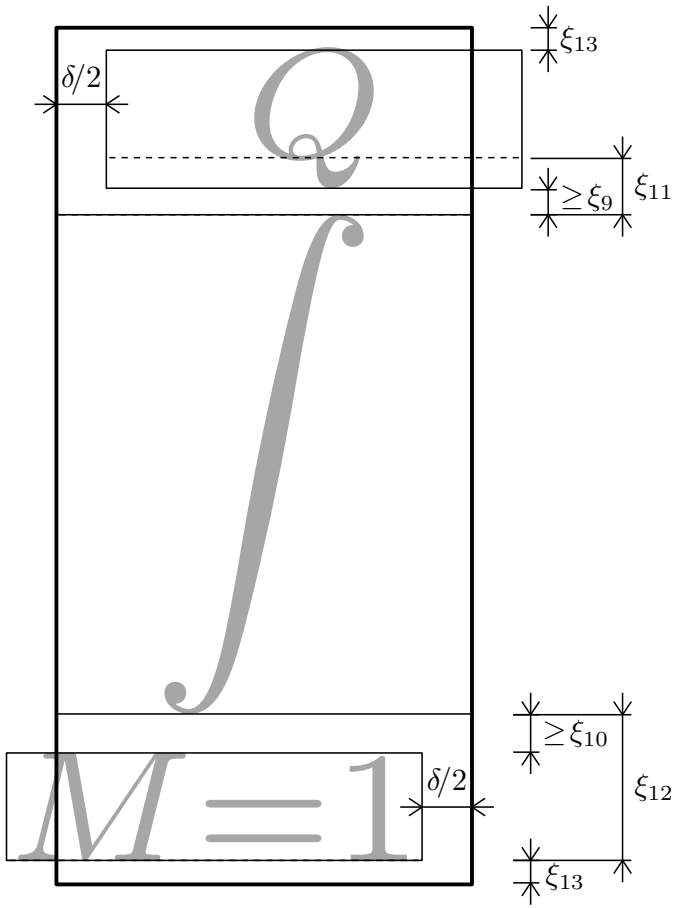
Font parameters

- Font parameters in TEX math fonts
  - approx. 20 parameters explicit in font metrics
  - many parameters implicit in typesetting algorithms
  - some parameters hidden in macro definitions
- Font parameters in OpenType math fonts
  - approx. 60 parameters explicit in MATH table
  - most TEX parameters have clear correspondence
  - some TEX parameters have *no* correspondence
  - some extensions / generalizations of TEX concepts
  - some cleanup of overloaded font data structures

Big Operators

- Spacing of limits on big operators
  - 5 parameters in TEX fontdimens
  - 4 parameters in OT MATH table
  - clear correspondence for ξ_9 to ξ_{12}
  - no correspondence for ξ_{13}
  - outside clearance assumed zero
- Parameter mapping

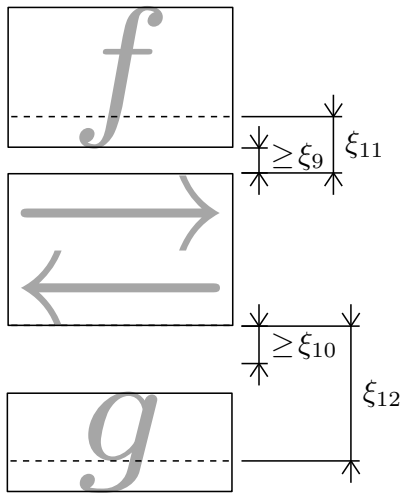
UpperLimitBaselineRiseMin	ξ_{11}
UpperLimitGapMin	ξ_9
LowerLimitGapMin	ξ_{10}
LowerLimitBaselineDropMin	ξ_{12}



Stretch Stacks

- Spacing of stretch stacks
  - generalization of stacked elements
  - e.g. labels above/below arrows
  - e.g. over/underbraces on formulas
  - correspondence at macro level in TEX
  - spacing similar to big operators
- Parameter mapping

StretchStackTopShiftUp	ξ_{11}
StretchStackGapAboveMin	ξ_9
StretchStackGapBelowMin	ξ_{10}
StretchStackBottomShiftDown	ξ_{12}



Over- and Underlines

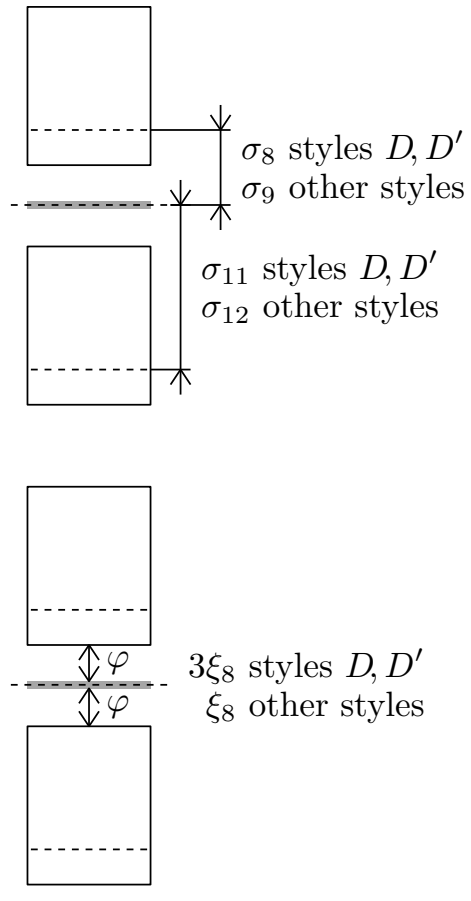
- Spacing of over- and underlines
  - 6 built-in rules in TEX algorithms
  - 6 parameters in OT MATH table
  - implicit rules made explicit in OT
  - greater flexibility of font designer
- Parameter mapping

OverbarExtraAscender	$(= \xi_8)$
OverbarRuleThickness	$(= \xi_8)$
OverbarVerticalGap	$(= 3\,\xi_8)$
UnderbarVerticalGap	$(= 3\,\xi_8)$
UnderbarRuleThickness	$(= \xi_8)$
UnderbarExtraDescender	$(= \xi_8)$

Fractions and Stacks (I)

- Spacing of regular fractions
  - 4 parameters in TEX fontdimens
  - 5 built-in rules in TEX algorithms
  - 9 parameters in OT MATH table
- Parameter mapping

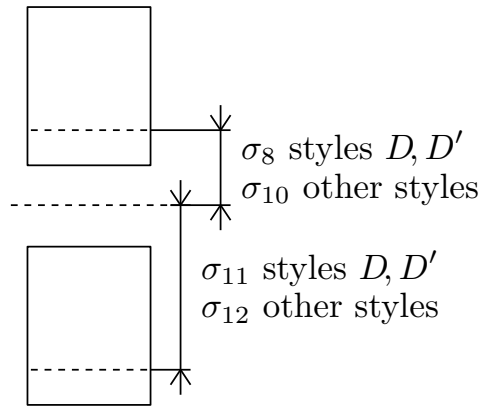
FractionNumeratorDisplayStyleShiftUp	σ_8
FractionNumeratorShiftUp	σ_9
FractionNumeratorDisplayStyleGapMin	$(= 3\,\xi_8)$
FractionNumeratorGapMin	$(= \xi_8)$
FractionRuleThickness	$(= \xi_8)$
FractionDenominatorDisplayStyleGapMin	$(= 3\,\xi_8)$
FractionDenominatorGapMin	$(= \xi_8)$
FractionDenominatorDisplayStyleShiftDown	σ_{11}
FractionDenominatorShiftDown	σ_{12}

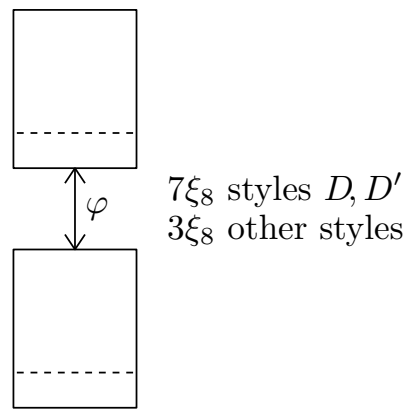


Fractions and Stacks (II)

- Spacing of generalized fractions (stacks)
  - 4 parameters in TEX fontdimens (overlap between fractions and stacks)
  - 2 built-in rules in TEX algorithms
  - 6 parameters in OT MATH table (no overlap between fractions and stacks)
- Parameter mapping

StackTopDisplayStyleShiftUp	σ_8
StackTopShiftUp	σ_{10}
StackDisplayStyleGapMin	$(= 7\,\xi_8)$
StackGapMin	$(= 3\,\xi_8)$
StackBottomDisplayStyleShiftDown	σ_{11}
StackBottomShiftDown	σ_{12}

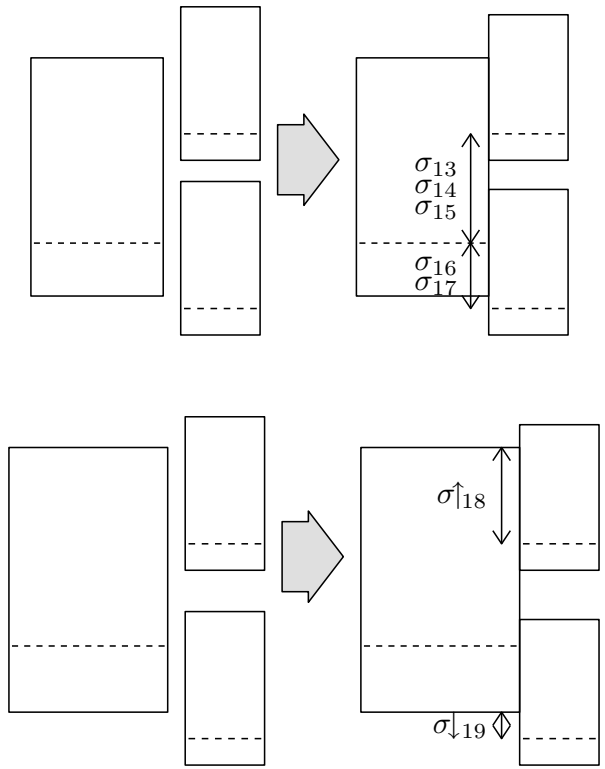




Superscripts and Subscripts (I)

- Spacing of superscripts and subscripts
  - 7 parameters in TEX fontdimens
  - 5 parameters in OT MATH table
  - no distinction between σ_{13} and σ_{14} (superscripts in display or text style)
  - no distinction between σ_{16} and σ_{17} (subscripts with or w/o superscripts)
- Parameter mapping

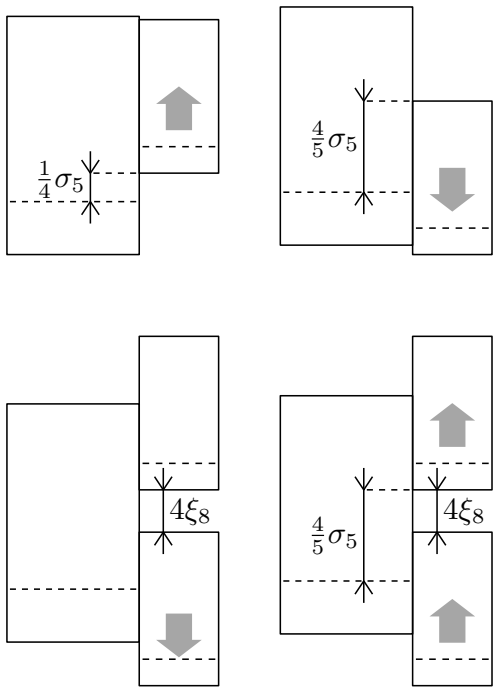
SuperscriptShiftUp	σ_{13}, σ_{14}
SuperscriptShiftUpCramped	σ_{15}
SubscriptShiftDown	σ_{16}, σ_{17}
SuperscriptBaselineDropMax	σ_{18}
SubscriptBaselineDropMin	σ_{19}



Superscripts and Subscripts (II)

- Spacing when resolving collisions between superscripts and subscripts
  - 4 built-in rules in TEX algorithms
  - 4 parameters in OT MATH table
  - implicit rules made explicit in OT
- Parameter mapping

SuperscriptBottomMin	$(= \frac{1}{4}\sigma_5)$
SubscriptTopMax	$(= \frac{4}{5}\sigma_5)$
SubSuperscriptGapMin	$(= 4\,\xi_8)$
SuperscriptBottomMaxWithSubscript	$(= \frac{4}{5}\sigma_5)$



Radicals (I)

- Spacing of radicals (square roots)
  - 4 built-in rules in TEX algorithms
  - 4 parameters in OT MATH table
  - unusual metrics in TEX: $h_{\surd} = \xi_8$
  - no need for unusual metrics in OT
- Parameter mapping

RadicalExtraAscender	$(= \xi_8)$
RadicalRuleThickness	$(= h_{\surd} = \xi_8)$
RadicalDisplayStyleVerticalGap	$(= \xi_8 + \frac{1}{4}\sigma_5)$
RadicalVerticalGap	$(= \xi_8 + \frac{1}{4}\xi_8)$

Radicals (II)

- Spacing of radicals (n-th roots)
  - 3 parameters hidden in TEX macros
  - 3 parameters in OT MATH table
  - implicit rules made explicit in OT
  - replacement of macros by primitives
- Parameter mapping

RadicalKernBeforeDegree	e. g. 5/18 em
RadicalKernAfterDegree	e. g. 10/18 em
RadicalDegreeBottomRaisePercent	e. g. 60 %

General parameters

- Mixed bag of parameters
  - some related to font sizes of script fonts
  - some related to size of delimited fractions
  - some related to placement of math accents
- Parameter mapping

ScriptPercentScaleDown	e. g. 70–80 %
ScriptScriptPercentScaleDown	e. g. 50–60 %
DisplayOperatorMinHeight	?? (e. g. 12–15 pt)
??	σ_{20} (e. g. 20–24 pt)
DelimitedSubFormulaMinHeight	σ_{21} (e. g. 10–12 pt)
AxisHeight	σ_{22} (axis height)
AccentBaseHeight	σ_5 (x-height)
FlattenedAccentBaseHeight	?? (capital height)

Font sizes of script fonts (I)

- In TEX math fonts
  - math families always loaded at 3 font sizes
  - font sizes of script fonts defined outside the font
  - font sizes defined in macro packages or format files
- Example (Computer Modern, using optical design sizes)

```
\newfam\symbols
\textfont\symbols=cmsy10 % at 10.0 pt
\scriptfont\symbols=cmsy7 % at 7.0 pt
\scriptscriptfont\symbols=cmsy5 % at 5.0 pt
```

- Example (Math Times, using scaled-down sizes)

```
\newfam\symbols
\textfont\symbols=mtsy10 % at 10.0 pt
\scriptfont\symbols=mtsy10 scaled 760 % at 7.6 pt
\scriptscriptfont\symbols=mtsy10 scaled 600 % at 6.0 pt
```

Font sizes of script fonts (II)

- In OpenType math fonts
  - font sizes of script fonts specified inside the font
  - optical variants for script sizes packaged in base font
  - optical variants activated by OpenType feature tags
  - potentially only a single OpenType math font needed
  - math families can still be loaded at 3 font sizes
- Example (using scaling factors and features)

```
\newfam\symbols
\textfont\symbols="CambriaMath" % at 10.0 pt
\scriptfont\symbols="CambriaMath:+ssty0"
  scaled \OTvalue{ScriptPercentScaleDown}
\scriptscriptfont\symbols="CambriaMath:+ssty1"
  scaled \OTvalue{ScriptScriptPercentScaleDown}
```

Delimited Fractions

- What's a delimited fraction?
  - `$$ \left( {n \atop k} \right) $$` (regular fraction)
  - `$$ {n \atopwithdelims() k} $$` (delimited fraction)
- What's the difference?
  - size depends on delimiterfactor, delimitershortfall
    $\Rightarrow$ 18 pt or 24 pt delimiters (depending on contents)
  - size depends on fontdimens σ_{20} (display), σ_{21} (text)
    $\Rightarrow$ always 24 pt delimiters (regardless of contents)
- What's the problem?
  - only one OpenType parameter `DelimitedSubFormulaMinHeight`
  - no suitable correspondence for TEX parameters σ_{20}, σ_{21}
  - no suitable implementation for `\atopwithdelims`

Variants and Constructions

- In TEX math fonts:
  - charlists and extensibles only used in specific contexts
  - big operators: 2 vertical sizes (text style vs. display style)
  - big delimiters: n vertical sizes + extensible version
  - wide accents: n horizontal sizes, but no extensible version
- In OpenType math fonts:
  - generalization of variants and constructions
  - big operators: can have more than 2 vertical sizes
  - big operators: can even have extensible version
  - wide accents: can also have extensible version
  - wide accents: can be applied to overbrace/underbrace
  - long arrows: can be represented by horizontal constructions

Big Operators

- In TEX math fonts:
  - only 2 sizes of operators (text style vs. display style)
  - no support for additional sizes or extensible versions
- In OpenType math fonts:
  - possible to have additional sizes of operators
  - OpenType parameter `DisplayOperatorMinHeight` needed to determine which size to use in display style
  - possible to have extensible versions of operators (depends on glyph shape, e.g. straight integral)
  - semantics may be difficult to implement in TEX (need context to determine size of operators)
  - semantics may be easier to implement in MathML

Big Delimiters

- In TEX math fonts:
  - usually 4 sizes of delimiters + extensible version
  - usual progression of sizes: 12 pt, 18 pt, 24 pt, 30 pt
  - macros to select specific sizes: big, Big, bigg, Bigg
  - no requirement to have 4 sizes, just a convention
- In OpenType math fonts:
  - possible to have additional or intermediate sizes, e.g. 4 of the usual sizes + 3 intermediate sizes
  - no limitations such as 16 TFM heights/depths
  - only base size of delimiters encoded in Unicode slots
  - additional sizes encoded in private-use area using internal glyph names `symbol.vsize<n>` or `symbolbig<n>`

Wide Accents

- In TEX math fonts:
  - only limited range of wide accents provided in fonts
  - no support for extensible versions of math accents
  - macro constructions used as a workaround (leaders)
- In OpenType math fonts:
  - possible to have extensible versions of math accents
  - possible to rewrite/simplify macro constructions
  - possible to redefine overbrace/underbrace as math accents (may require different semantics for labels on braces)
  - only base size of math accents encoded in Unicode slots
  - additional sizes encoded in private-use area using internal glyph names `symbol.hsize<n>` or `symbolwide<n>`

Interactive Demo (II)

- Interactive Demo
  - open Cambria Math in FontForge
  - inspect variants and constructions

Font parameters

- OpenType MATH extends many TEX concepts
  - many built-in rules replaced by explicit parameters
  - some overlap in multi-purpose parameters avoided
  - some macro parameters integrated (e.g. degree of radicals)
  - some extensions of concepts integrated (e.g. stretch stacks)
- OpenType MATH falls short on a few TEX concepts
  - outside clearance on big operators ($\xi_{13} = 0$)
  - superscripts in display or text style ($\sigma_{13} \neq \sigma_{14}$)
  - subscripts with or w/o superscripts ($\sigma_{16} \neq \sigma_{17}$)
  - nothing suitable for delimited fractions (σ_{20}, σ_{21})
- OpenType MATH cannot reproduce 100% of TEX behavior
- TEX engines can add the missing bits, if really needed

Variants and Constructions

- OpenType MATH extends many TEX concepts
  - generalization of applicable context
  - additional sizes + extensible versions of big operators
  - additional sizes + extensible versions of wide accents
  - horizontal constructions not limited to math accents, also applicable for long arrows or over/under delimiters
- TEX engines may need to implement new semantics
  - big operators may need context to determine size
  - new primitives needed for over/under delimiters
  - new primitives needed for labels on long arrows
  - macros can be rewritten/simplified using new primitives