Writing mathematical expressions

You can use a subset TeX markup in any matplotlib text string by placing it inside a pair of dollar signs ($).

Note that you do not need to have TeX installed, since matplotlib ships its own TeX expression parser, layout engine and fonts. The layout engine is a fairly direct adaptation of the layout algorithms in Donald Knuth's TeX, so the quality is quite good (matplotlib also provides a `usetex` option for those who do want to call out to TeX to generate their text (see Text rendering With LaTeX).

Any text element can use math text. You should use raw strings (precede the quotes with an `'r'`), and surround the math text with dollar signs ($), as in TeX. Regular text and mathtext can be interleaved within the same string. Mathtext can use DejaVu Sans (default), DejaVu Serif, the Computer Modern fonts (from (La)TeX), STIX fonts (with are designed to blend well with Times), or a Unicode font that you provide. The mathtext font can be selected with the customization variable `mathtext.fontset` (see Customizing matplotlib)

Note

On "narrow" builds of Python, if you use the STIX fonts you should also set `ps.fonttype` and `pdf.fonttype` to 3 (the default), not 42. Otherwise some characters will not be visible.

Here is a simple example:

```
# plain text
plt.title('alpha > beta')
```

produces "alpha > beta".

Whereas this:

```
# math text
plt.title(r'$\alpha > \beta$')
```

produces "$\alpha > \beta$".

Note

Depsy 100th percentile

Travis-CI: build passing

Table Of Contents

Related Topics

Documentation overview
- User's Guide
  - Working with text
    - Previous: Annotation
    - Next: Text rendering With LaTeX

This Page

Show Source

Quick search

Go

Mathtext should be placed between a pair of dollar signs ($). To make it easy to display monetary values, e.g., “$100.00”, if a single dollar sign is present in the entire string, it will be displayed verbatim as a dollar sign. This is a small change from regular TeX, where the dollar sign in non-math text would have to be escaped (‘\$’).

Note

While the syntax inside the pair of dollar signs ($) aims to be TeX-like, the text outside does not. In particular, characters such as:

```
# $ % & ~ _ ^ \ { } \( \) \[ \]
```

have special meaning outside of math mode in TeX. Therefore, these characters will behave differently depending on the rcParam `text.usetex` flag. See the usetex tutorial for more information.

Subscripts and superscripts

To make subscripts and superscripts, use the `'_'` and `'^'` symbols:

```
r'$\alpha_i > \beta_i$'
```

$$\alpha_i > \beta_i$$

Some symbols automatically put their sub/superscripts under and over the operator. For example, to write the sum of x_i from 0 to ∞, you could do:

```
r'$\sum_{i=0}^\infty x_i$'
```

$$\sum_{i=0}^\infty x_i$$

Fractions, binomials and stacked numbers

Fractions, binomials and stacked numbers can be created with the `\frac{}{}`, `\binom{}{}` and `\stackrel{}{}` commands, respectively:

```
r'$\frac{3}{4} \binom{3}{4} \stackrel{3}{4}$'
```

produces

$$\frac{3}{4} \binom{3}{4} \stackrel{3}{4}$$

Fractions can be arbitrarily nested:

```
r'$\frac{5 - \frac{1}{x}}{4}$'
```

produces

$$\frac{5 - \frac{1}{x}}{4}$$

Note that special care needs to be taken to place parentheses and brackets around fractions. Doing things the obvious way produces brackets that are too small:

```
r'$(\frac{5 - \frac{1}{x}}{4})$'
```

$$(\frac{5 - \frac{1}{x}}{4})$$

The solution is to precede the bracket with `\left` and `\right` to inform the parser that those brackets encompass the entire object:

```
r'$\left(\frac{5 - \frac{1}{x}}{4}\right)$'
```

$$\left(\frac{5 - \frac{1}{x}}{4}\right)$$

Radicals

Radicals can be produced with the `\sqrt[]{}` command. For example:

```
r'$\sqrt{2}$'
```

$$\sqrt{2}$$

Any base can (optionally) be provided inside square brackets. Note that the base must be a simple expression, and can not contain layout commands such as fractions or sub/superscripts:

```
r'$\sqrt[3]{x}$'
```

$$\sqrt[3]{x}$$

Fonts

The default font is *italics* for mathematical symbols.

Note

This default can be changed using the `mathtext.default` rcParam. This is useful, for example, to use the same font as regular non-math text for math text, by setting it to `regular`.

To change fonts, e.g., to write “sin” in a Roman font, enclose the text in a font command:

```
r'$s(t) = \mathcal{A}\mathrm{sin}(2 \omega t)$'
```

$$s(t) = \mathcal{A}\mathrm{sin}(2 \omega t)$$

More conveniently, many commonly used function names that are typeset in a Roman font have shortcuts. So the expression above could be written as follows:

```
r'$s(t) = \mathcal{A}\sin(2 \omega t)$'
```

$$s(t) = \mathcal{A}\sin(2 \omega t)$$

Here “s” and “t” are variable in italics font (default), “sin” is in Roman font, and the amplitude “A” is in calligraphy font. Note in the example above the caligraphy A is squished into the `sin`. You can use a spacing command to add a little whitespace between them:

```
s(t) = \mathcal{A}\/\sin(2 \omega t)
```

$$s(t) = \mathcal{A}\/\sin(2 \omega t)$$

The choices available with all fonts are:

Command	Result
`\mathrm{Roman}`	Roman
`\mathit{Italic}`	Italic
`\mathtt{Typewriter}`	$\mathtt{Typewriter}$
`\mathcal{CALLIGRAPHY}`	$\mathcal{CALLIGRAPHY}$

When using the STIX fonts, you also have the choice of:

Command	Result
`\mathbb{blackboard}`	$\mathbb{blackboard}$
`\mathrm{\mathbb{blackboard}}`	$\mathrm{\mathbb{blackboard}}$
`\mathfrak{Fraktur}`	$\mathfrak{Fraktur}$
`\mathsf{sansserif}`	$\mathsf{sansserif}$
`\mathrm{\mathsf{sansserif}}`	$\mathrm{\mathsf{sansserif}}$
`\mathcircled{circled}`	ⓒⓘⓡⓒⓛⓔⓓ

There are also three global “font sets” to choose from, which are selected using the `mathtext.fontset` parameter in matplotlibrc.

`cm`: **Computer Modern (TeX)**

$$\mathcal{R}\prod_{i=\alpha_{i+1}}^\infty a_i\sin(2 \pi f x_i)$$

`stix`: **STIX** (designed to blend well with Times)

$$\mathcal{R}\prod_{i=\alpha_{i+1}}^\infty a_i\sin(2 \pi f x_i)$$

`stixsans`: **STIX sans-serif**

$$\mathcal{R}\prod_{i=\alpha_{i+1}}^\infty a_i\sin(2\pi fx_i)$$

Additionally, you can use `\mathdefault{...}` or its alias `\mathregular{...}` to use the font used for regular text outside of mathtext. There are a number of limitations to this approach, most notably that far fewer symbols will be available, but it can be useful to make math expressions blend well with other text in the plot.

Custom fonts

mathtext also provides a way to use custom fonts for math. This method is fairly tricky to use, and should be considered an experimental feature for patient users only. By setting the rcParam `mathtext.fontset` to `custom`, you can then set the following parameters, which control which font file to use for a particular set of math characters.

Parameter	Corresponds to
`mathtext.it`	`\mathit{}` or default italic
`mathtext.rm`	`\mathrm{}` Roman (upright)
`mathtext.tt`	`\mathtt{}` Typewriter (monospace)
`mathtext.bf`	`\mathbf{}` bold italic
`mathtext.cal`	`\mathcal{}` calligraphic
`mathtext.sf`	`\mathsf{}` sans-serif

Each parameter should be set to a fontconfig font descriptor (as defined in the yet-to-be-written font chapter).

The fonts used should have a Unicode mapping in order to find any non-Latin characters, such as Greek. If you want to use a math symbol that is not contained in your custom fonts, you can set the rcParam `mathtext.fallback_to_cm` to `True` which will cause the mathtext system to use characters from the default Computer Modern fonts whenever a particular character can not be found in the custom font.

Note that the math glyphs specified in Unicode have evolved over time, and many fonts may not have glyphs in the correct place for mathtext.

Accents

An accent command may precede any symbol to add an accent above it. There are long and short forms for some of them.

Command	Result
`\acute a` or `\'a`	$\acute{a}$
`\bar a`	$\bar{a}$
`\breve a`	$\breve{a}$
`\ddot a` or `\"a`	$\ddot{a}$
`\dot a` or `\.a`	$\dot{a}$

Command	Result
`\grave a` or `` \`a ``	$\grave{a}$
`\hat a` or `\^a`	$\hat{a}$
`\tilde a` or `\~a`	$\tilde{a}$
`\vec a`	$\vec{a}$
`\overline{abc}`	$\overline{abc}$

In addition, there are two special accents that automatically adjust to the width of the symbols below:

Command	Result
`\widehat{xyz}`	$\widehat{xyz}$
`\widetilde{xyz}`	$\widetilde{xyz}$

Care should be taken when putting accents on lower-case i's and j's. Note that in the following `\imath` is used to avoid the extra dot over the i:

```
r"$\hat i\ \ \hat \imath$"
```

$\hat i\ \ \hat \imath$

Symbols

You can also use a large number of the TeX symbols, as in `\infty`, `\leftarrow`, `\sum`, `\int`.

Lower-case Greek

α `\alpha`	β `\beta`	χ `\chi`	δ `\delta`	$\digamma$ `\digamma`
ϵ `\epsilon`	η `\eta`	γ `\gamma`	ι `\iota`	κ `\kappa`
λ `\lambda`	μ `\mu`	ν `\nu`	ω `\omega`	ϕ `\phi`
π `\pi`	ψ `\psi`	ρ `\rho`	σ `\sigma`	τ `\tau`
θ `\theta`	υ `\upsilon`	ε `\varepsilon`	$\varkappa$ `\varkappa`	φ `\varphi`
ϖ `\varpi`	ϱ `\varrho`	ς `\varsigma`	ϑ `\vartheta`	ξ `\xi`
ζ `\zeta`				

Upper-case Greek

Δ `\Delta`	Γ `\Gamma`	Λ `\Lambda`	Ω `\Omega`	Φ `\Phi`	Π `\Pi`
Ψ `\Psi`	Σ `\Sigma`	Θ `\Theta`	Υ `\Upsilon`	Ξ `\Xi`	$\mho$ `\mho`
∇ `\nabla`					

Hebrew

ℵ \aleph	ℶ \beth	ℸ \daleth	ℷ \gimel

Delimiters

/ /	[[	⇓ \Downarrow	⇑ \Uparrow	‖ \Vert	\ \backslash
↓ \downarrow	⟨ \langle	⌈ \lceil	⌊ \lfloor	⌞ \llcorner	⌟ \lrcorner
⟩ \rangle	⌉ \rceil	⌋ \rfloor	⌜ \ulcorner	↑ \uparrow	⌝ \urcorner
| \vert	{ \{	‖ \\|	} \}	]]	| |

Big symbols

⋂ \bigcap	⋃ \bigcup	⨀ \bigodot	⨁ \bigoplus	⨂ \bigotimes
⨄ \biguplus	⋁ \bigvee	⋀ \bigwedge	∐ \coprod	∫ \int
∮ \oint	∏ \prod	∑ \sum		

Standard function names

Pr \Pr	arccos \arccos	arcsin \arcsin	arctan \arctan
arg \arg	cos \cos	cosh \cosh	cot \cot
coth \coth	csc \csc	deg \deg	det \det
dim \dim	exp \exp	gcd \gcd	hom \hom
inf \inf	ker \ker	lg \lg	lim \lim
liminf \liminf	limsup \limsup	ln \ln	log \log
max \max	min \min	sec \sec	sin \sin
sinh \sinh	sup \sup	tan \tan	tanh \tanh

Binary operation and relation symbols

≎ \Bumpeq	⋒ \Cap	⋓ \Cup
≑ \Doteq	⨝ \Join	⋐ \Subset
⋑ \Supset	⊩ \Vdash	⊪ \Vvdash
≈ \approx	≊ \approxeq	∗ \ast
≍ \asymp	϶ \backepsilon	∽ \backsim

⋍ \backsimeq	⊼ \barwedge	∵ \because
≬ \between	◯ \bigcirc	▽ \bigtriangledown
△ \bigtriangleup	◀ \blacktriangleleft	▶ \blacktriangleright
⊥ \bot	⋈ \bowtie	⊡ \boxdot
⊟ \boxminus	⊞ \boxplus	⊠ \boxtimes
● \bullet	≏ \bumpeq	∩ \cap
· \cdot	∘ \circ	≗ \circeq
≔ \coloneq	≅ \cong	∪ \cup
⋞ \curlyeqprec	⋟ \curlyeqsucc	⋎ \curlyvee
⋏ \curlywedge	† \dag	⊣ \dashv
‡ \ddag	⋄ \diamond	÷ \div
⋇ \divideontimes	≐ \doteq	≑ \doteqdot
∔ \dotplus	⩞ \doublebarwedge	≖ \eqcirc
≕ \eqcolon	≂ \eqsim	⪖ \eqslantgtr
⪕ \eqslantless	≡ \equiv	≒ \fallingdotseq

⌢ \frown	≥ \geq	≧ \geqq
⩾ \geqslant	≫ \gg	⋙ \ggg
⪊ \gnapprox	≩ \gneqq	⋧ \gnsim
⪆ \gtrapprox	⋗ \gtrdot	⋛ \gtreqless
⪌ \gtreqqless	≷ \gtrless	≳ \gtrsim
∈ \in	⊺ \intercal	⋋ \leftthreetimes
≤ \leq	≦ \leqq	⩽ \leqslant
⪅ \lessapprox	⋖ \lessdot	⋚ \lesseqgtr
⪋ \lesseqqgtr	≶ \lessgtr	≲ \lesssim
≪ \ll	⋘ \lll	⪉ \lnapprox
≨ \lneqq	⋦ \lnsim	⋉ \ltimes
∣ \mid	⊧ \models	∓ \mp
⊯ \nVDash	⊮ \nVdash	≉ \napprox
≇ \ncong	≠ \ne	≠ \neq
≠ \neq	≢ \nequiv	≱ \ngeq
≯ \ngtr	∋ \ni	≰ \nleq

≮ \nless	∤ \nmid	∉ \notin
∦ \nparallel	⊀ \nprec	≁ \nsim
⊄ \nsubset	⊈ \nsubseteq	⊁ \nsucc
⊅ \nsupset	⊉ \nsupseteq	⋪ \ntriangleleft

⋬ \ntrianglelefteq	⋫ \ntriangleright	⋭ \ntrianglerighteq
⊭ \nvDash	⊬ \nvdash	⊙ \odot
⊖ \ominus	⊕ \oplus	⊘ \oslash
⊗ \otimes	∥ \parallel	⟂ \perp
⋔ \pitchfork	± \pm	≺ \prec
⪷ \precapprox	≼ \preccurlyeq	⪯ \preceq
⪹ \precnapprox	⋨ \precnsim	≾ \precsim
∝ \propto	⋌ \rightthreetimes	≓ \risingdotseq
⋊ \rtimes	∼ \sim	≃ \simeq
∕ \slash	⌣ \smile	⊓ \sqcap
⊔ \sqcup	⊏ \sqsubset	⊏ \sqsubset
⊑ \sqsubseteq	⊐ \sqsupset	⊐ \sqsupset
⊒ \sqsupseteq	⋆ \star	⊂ \subset
⊆ \subseteq	⫅ \subseteqq	⊊ \subsetneq
⫋ \subsetneqq	≻ \succ	⪸ \succapprox
≽ \succcurlyeq	⪰ \succeq	⪺ \succnapprox
⋩ \succnsim	≿ \succsim	⊃ \supset
⊇ \supseteq	⫆ \supseteqq	⊋ \supsetneq
⫌ \supsetneqq	∴ \therefore	× \times
⊤ \top	◁ \triangleleft	⊴ \trianglelefteq

≜ \triangleq	▷ \triangleright	⊵ \trianglerighteq
⊎ \uplus	⊨ \vDash	∝ \varpropto
⊲ \vartriangleleft	⊳ \vartriangleright	⊢ \vdash
∨ \vee	⊻ \veebar	∧ \wedge
≀ \wr		

Arrow symbols

⇓ \Downarrow	⇐ \Leftarrow

⇔ \Leftrightarrow	⇚ \Lleftarrow
⟸ \Longleftarrow	⟺ \Longleftrightarrow
⟹ \Longrightarrow	↰ \Lsh
⇗ \Nearrow	⇖ \Nwarrow
⇒ \Rightarrow	⇛ \Rrightarrow
↱ \Rsh	⇘ \Searrow
⇙ \Swarrow	⇑ \Uparrow
⇕ \Updownarrow	↺ \circlearrowleft
↻ \circlearrowright	↶ \curvearrowleft
↷ \curvearrowright	⇠ \dashleftarrow
⇢ \dashrightarrow	↓ \downarrow
⇊ \downdownarrows	⇃ \downharpoonleft
⇂ \downharpoonright	↩ \hookleftarrow
↪ \hookrightarrow	⇝ \leadsto
← \leftarrow	↢ \leftarrowtail
↽ \leftharpoondown	↼ \leftharpoonup
⇇ \leftleftarrows	↔ \leftrightarrow
⇆ \leftrightarrows	⇋ \leftrightharpoons
↭ \leftrightsquigarrow	⇜ \leftsquigarrow
⟵ \longleftarrow	⟷ \longleftrightarrow
⟼ \longmapsto	⟶ \longrightarrow
↫ \looparrowleft	↬ \looparrowright
↦ \mapsto	⊸ \multimap
⇍ \nLeftarrow	⇎ \nLeftrightarrow
⇏ \nRightarrow	↗ \nearrow
↚ \nleftarrow	↮ \nleftrightarrow
↛ \nrightarrow	↖ \nwarrow
→ \rightarrow	↣ \rightarrowtail
⇁ \rightharpoondown	⇀ \rightharpoonup
⇄ \rightleftarrows	⇄ \rightleftarrows
⇌ \rightleftharpoons	⇌ \rightleftharpoons
⇉ \rightrightarrows	⇉ \rightrightarrows
⇝ \rightsquigarrow	↘ \searrow
↙ \swarrow	→ \to
↞ \twoheadleftarrow	↠ \twoheadrightarrow
↑ \uparrow	↕ \updownarrow

↕ `\updownarrow`	↿ `\upharpoonleft`
↾ `\upharpoonright`	⇈ `\upuparrows`

Miscellaneous symbols

$ `\$`	Å `\AA`	Ⅎ `\Finv`
⅁ `\Game`	ℑ `\Im`	¶ `\P`
ℜ `\Re`	§ `\S`	∠ `\angle`
‵ `\backprime`	★ `\bigstar`	■ `\blacksquare`
▲ `\blacktriangle`	▼ `\blacktriangledown`	⋯ `\cdots`
✓ `\checkmark`	® `\circledR`	Ⓢ `\circledS`
♣ `\clubsuit`	∁ `\complement`	© `\copyright`
⋱ `\ddots`	♢ `\diamondsuit`	ℓ `\ell`
∅ `\emptyset`	ð `\eth`	∃ `\exists`
♭ `\flat`	∀ `\forall`	ℏ `\hbar`
♡ `\heartsuit`	ℏ `\hslash`	∭ `\iiint`
∬ `\iint`	∬ `\iint`	ı `\imath`
∞ `\infty`	ȷ `\jmath`	… `\ldots`
∡ `\measuredangle`	♮ `\natural`	¬ `\neg`
∄ `\nexists`	∰ `\oiiint`	∂ `\partial`
′ `\prime`	♯ `\sharp`	♠ `\spadesuit`
∢ `\sphericalangle`	ß `\ss`	▽ `\triangledown`
∅ `\varnothing`	△ `\vartriangle`	⋮ `\vdots`
℘ `\wp`	¥ `\yen`	

If a particular symbol does not have a name (as is true of many of the more obscure symbols in the STIX fonts), Unicode characters can also be used:

```
ur'$\u23ce$'
```

Example

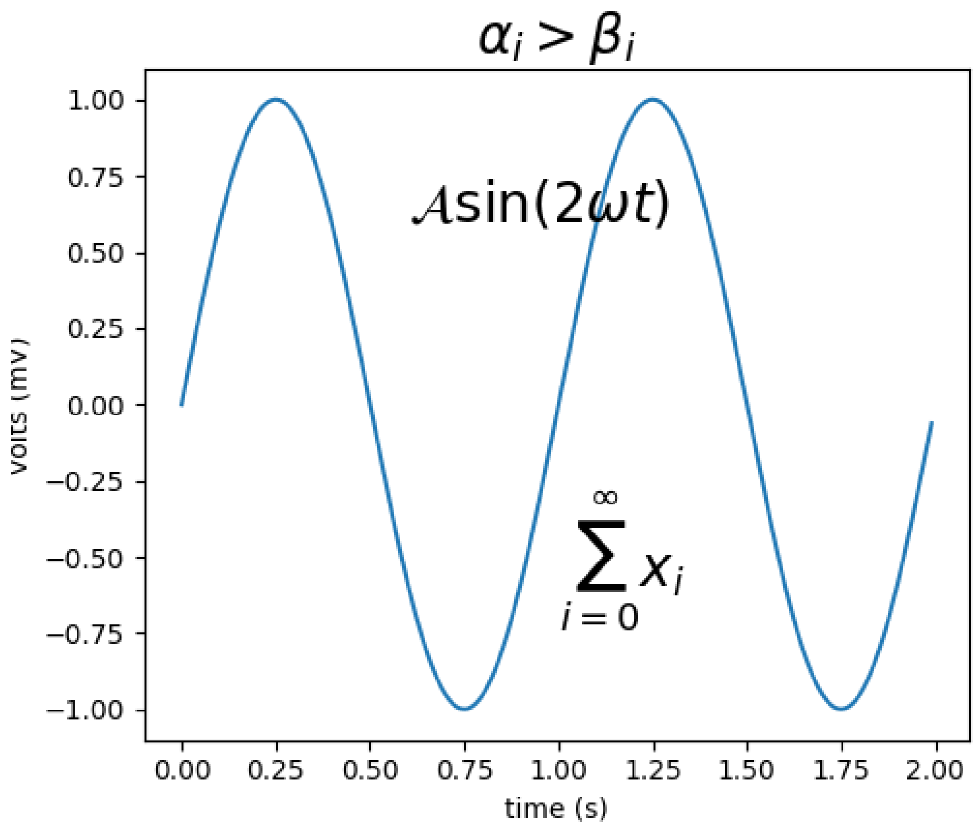
Here is an example illustrating many of these features in context.

```
import numpy as np
import matplotlib.pyplot as plt
t = np.arange(0.0, 2.0, 0.01)
```

```
s = np.sin(2*np.pi*t)

plt.plot(t,s)
plt.title(r'$\alpha_i > \beta_i$', fontsize=20)
plt.text(1, -0.6, r'$\sum_{i=0}^\infty x_i$', fontsize
plt.text(0.6, 0.6, r'$\mathcal{A}\mathrm{sin}(2 \omega
         fontsize=20)
plt.xlabel('time (s)')
plt.ylabel('volts (mV)')
plt.show()
```

(Source code, png, pdf)


$\alpha_i > \beta_i$
$\mathcal{A}\mathrm{sin}(2\omega t)$
$\sum_{i=0}^\infty x_i$
volts (mV)
time (s)