LaTeX Tips, Tricks, and Good Practice

What I have found that works well for me.

Ian P. Roberts

Updated: April 6, 2021

Outline

Fundamentals

Fundamentals

Creating Custom Macros

Fundamentals — Creating Custom Macros

Custom macros can be created using the `\newcommand{cmd}[N]{def}` macro, where `cmd` is the macro name, `N` is the number of input arguments (neglect `[N]` if none), and `def` is the macro definition.

A macro `\addvals{}{}` can be created using

```
\newcommand{\addvals}[2]{#1 + #2}
```

where `$\addvals{a}{b}$` produces $a + b$.

Note that `\addvals{a}{b}` produces a + b when not used within a math environment.

Fundamentals — Creating Custom Macros

To use this macro outside of math environments (e.g., without `$...$`) but behave as if it were, we can use `\ensuremath{}` to wrap our macro definition.

```
\newcommand{\addvals}[2]{\ensuremath{#1 + #2}}
```

where `\addvals{a}{b}` produces $a + b$.

Fundamentals — Creating Custom Macros

I have found the following macro useful for easily commenting out LaTeX.

```
\newcommand{\comment}[1]{}
```

Fundamentals

Referencing Figures, Tables, etc.

Fundamentals — Referencing Figures, Tables, etc.

To consistently reference figures, I define a custom macro `\figref{}`

```
\newcommand{\figref}[1]{Fig.~\ref{#1}}
```

This ensures my co-authors and I use consistent referencing. Changing the referencing style according to convention/preference can be done so in one place.

Fundamentals — Referencing Figures, Tables, etc.

This can be done likewise for tables, algorithms, theorems, sections, subsections, appendices, etc.

```
\newcommand{\tabref}[1]{Table~\ref{#1}}
\newcommand{\algref}[1]{Algorithm~\ref{#1}}
\newcommand{\thmref}[1]{Theorem~\ref{#1}}
\newcommand{\secref}[1]{Section~\ref{#1}}
\newcommand{\subsecref}[1]{Subsection~\ref{#1}}
\newcommand{\appref}[1]{Appendix~\ref{#1}}
```

Fundamentals

Using Multiple TEX Files

Fundamentals — Using Multiple TEX Files

I have found that using multiple TEX files can help me stay organized.

TEX files can be called using the `\input{my-file.tex}` macro, which "inserts" the contents of `my-file.tex` directly.

Fundamentals — Using Multiple TEX Files

I often use this for my "main" document file (e.g., `main.tex`) by organizing it in the follow fashion

```
...
\begin{document}

\input{sec-title.tex}
\input{sec-introduction.tex}
...
\input{sec-conclusion.tex}

\end{document}
```

This can be especially convenient when collaborating.

Figures

Figures

Inserting Figures

Figures — Inserting Figures

Figures can be inserted into a document using the following

```
\begin{figure}
    \centering
    \includegraphics[...]{...}
    \caption{Caption.}
    \label{fig:label}
\end{figure}
```

Figures — Inserting Figures

My go-to way to size figures is

```
\includegraphics[width=\linewidth,%
                 height=\textheight,%
                 keepaspectratio]%
                 {...}
```

The `keepaspectratio` option will force the figure to maintain its aspect ratio while constraining itself to the be within the width and height specified. In other words, it will restrict either the width or the height depending on which is more restrictive. I have found this to be extremely useful to align and size figures.

Figures

Vector Graphics

Figures — Vector Graphics

Vector graphics (e.g., .eps, .pdf) should be used when possible and practical.

There are many reasons for this, the primary being that they maintain their quality when zooming in, as they are mathematically described/rendered as opposed to bit-mapping pixels.

Figures — Vector Graphics

To output color .eps versions of figures in MATLAB, use the `print` function

```
print(filename,'-depsc')
```

The `export_fig` function (available online) is a very popular tool to use instead of the `print` function.

Figures

LaTeX Rendering in MATLAB

Figures — LaTeX Rendering in MATLAB

MATLAB can render its figure labels, legend, axes, etc. using LaTeX.

To set its interpreters to automatically use LaTeX, the following can be placed in your `startup.m` script.

```
set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaultPolarAxesTickLabelInterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');
set(groot,'defaultColorbarTickLabelInterpreter','latex');
```

With this in place, fonts will be Computer Modern Roman and labels, etc. can be typeset using LaTeX.

Figures

Subfigures

Figures — Subfigures

The `subfig` package allows users to have multiple figures within a single figure element, which can be referenced as Fig. 1a, 1b, etc.

I load it using the following.

```
\usepackage[caption=false,font=footnotesize]{subfig}
```

Figures — Subfigures

Placing two figures side-by-side can be accomplished in the following fashion.

```
\begin{figure}
    \centering
    \subfloat[Caption a.]{%
        \includegraphics[...]{...}%
        \label{fig:subfig-a}}
    \quad
    \subfloat[Caption b.]{%
        \includegraphics[...]{...}%
        \label{fig:subfig-b}}
    \caption{Caption here.}
    \label{fig:subfigs}
\end{figure}
```

Figures — Subfigures

(a) Caption a.

(b) Caption b.

Figure 1: Caption here.

Figures — Subfigures

This is likely your friend here, for sizing each subfigure.

```
\includegraphics[width=0.48\linewidth,%
                 height=\textheight,%
                 keepaspectratio]%
                 {...}
```

For example, you may want them each take up around 50% of the width (the \quad takes some space). Or you may want to adjust them to have the same height.

Math

Math

Environments

Math — Environments

My go-to math environment is `align`.

```
\begin{align}
z = x + y
\end{align}
```

$$z = x + y \tag{1}$$

Math — Environments

You can use `gather` when you want your lines centered with each other.

```
\begin{gather}
w = t + u + v \\
z = x + y
\end{gather}
```

$$w = t + u + v \tag{2}$$

$$z = x + y \tag{3}$$

Math — Environments

The cases environment within a math environment can be used for piecewise definitions.

```
\begin{align}
u(t) =
\begin{cases}
1, & t \geq 0 \\
0, & t < 0
\end{cases}
\end{align}
```

$$u(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0 \end{cases} \tag{4}$$

Math — Environments

The `bmatrix` and other matrix styling environments can be used within a math environment.

```
\begin{align}
\mathbf{X} =
\begin{bmatrix}
1 & \cdots & 10 \\
\vdots &  \ddots & \vdots \\
10 & \cdots & 20
\end{bmatrix}
\end{align}
```

$$\mathbf{X} = \begin{bmatrix} 1 & \cdots & 10 \\ \vdots & \ddots & \vdots \\ 10 & \cdots & 20 \end{bmatrix} \tag{5}$$

Math

Styling Matrices, Vectors, etc.

Math — Styling Matrices, Vectors, etc.

Macros are very useful for **quickly** and **consistently** typesetting LaTeX, especially math.

For example, to typeset matrices consistently, we can define a command that applies boldface to the matrix letter.

```
\newcommand{\mat}[1]{\mathbf{#1}}
```

Then, the following

```
\mat{A}, \mat{B}, \mat{C}, \mat{\Sigma}, \mat{\Delta}
```

produces

$$\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{\Sigma}, \mathbf{\Delta} \tag{6}$$

Math — Styling Matrices, Vectors, etc.

To more quickly typeset this, we can define

```
\newcommand{\mA}{\mat{A}}
\newcommand{\mB}{\mat{B}}
...
```

Math — Styling Matrices, Vectors, etc.

If this seems overkill, imagine you want to change to $\mathbf{A}$ instead of $\mathbf{A}$.

Or imagine you want to be consistent when merging two documents that use different conventions (e.g., assembling your dissertation).

It sure would be convenient to change it in one place rather than each instance.

Math — Styling Matrices, Vectors, etc.

You can do this for:

- sets
- vectors
- matrices
- tensors
- frequency-domain versus time-domain notation
- ...

Math

Enclosures

Math — Enclosures

Don't do things like this. This is ridiculous.

$$(\frac{1}{N}), |\frac{1}{N}|, \{\frac{1}{N}\} \tag{7}$$

If you do this, no one will even read your paper. Just kidding.

But seriously.

Math — Enclosures

For enclosures (parentheses, brackets, etc.) that automatically scale with what's inside, use

```
\left( \frac{1}{N} \right)
```

This produces the much nicer

$$\left(\frac{1}{N}\right), \left|\frac{1}{N}\right|, \left\{\frac{1}{N}\right\} \tag{8}$$

Note that these do not work across line breaks, so you will have to manually use `\Bigg( \Bigg)`, etc.

Math — Enclosures

Defining macros like these can make things even easier.

```
\newcommand{\parens}[1]{\left(#1\right)}
\newcommand{\brackets}[1]{\left[#1\right]}
\newcommand{\braces}[1]{\left\{#1\right\}}
\newcommand{\bars}[1]{\left\vert#1\right\vert}
\newcommand{\doublebars}[1]{\left\Vert#1\right\Vert}
\newcommand{\angles}[1]{\left\langle#1\right\rangle}
\newcommand{\ceil}[1]{\left\lceil{#1}\right\rceil}
\newcommand{\floor}[1]{\left\lfloor{#1}\right\rfloor}
```

Math — Enclosures

So you simply have to write `\parens{\frac{1}{N}}` to produce

$$\left(\frac{1}{N}\right) \tag{9}$$

This also offers the convenience of changing the enclosure in one place.

Math

Subscripts and Superscripts

Math — Subscripts and Superscripts

There are generally two types of subscripts:

1. x_i — i is a **variable** and thus should be in math font `x_i`.
2. x_{tx} — tx is a **label** and should not be in math font.
   - x_{tx} — `x_{\mathrm{tx}}`
   - x_{tx} — `x_{\mathsf{tx}}`

Note that x_{tx} (`x_{tx}`) should not be used. Don't you think x_{tx} looks better?

The same goes for superscripts.

Math — Subscripts and Superscripts

Do not refer to indices as i^{th} (`$i^{th}$`).

Instead, use either `$i^{\mathrm{th}}$` or `$i$-th` to produce i^{th} or i-th, respectively.

For 2-D indexing, we can use $(i, j)^{\mathrm{th}}$ or (i, j)-th, for example.

Math

Custom Math Operators

Math — Custom Math Operators

Custom math operators/functions can be created in the following fashion

```
\newcommand{\svdop}[1]{\mathrm{SVD}\parens{#1}}
```

which can be used as `\svdop{\mA}` to represent taking the SVD of a matrix as, for example,

$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \mathrm{SVD}\left(\mathbf{A}\right) \tag{10}$$

We use `\mathrm{}` here to style the operator nicely.

Plenty of uses: $\mathrm{maxk}\left(\mathbf{x}, K\right)$, $\mathrm{null}\left(\mathbf{A}\right)$, $\mathrm{diag}\left(\mathbf{X}\right)$, ...

Math — Custom Math Operators

Transpose, conjugate, conjugate transpose, inverse, and pseudoinverse can be typeset using

```
\newcommand{\trans}{^{\mathrm{T}}}
\newcommand{\conj}{^{\mathrm{c}}}
\newcommand{\ctrans}{^{*}}
\newcommand{\inv}{^{-1}}
\newcommand{\pinv}{^{\dagger}}
```

which can be used as `\mA\inv`, `\mA\ctrans`, etc. to produce

$$\mathbf{A}^{\mathrm{T}}, \mathbf{A}^{\mathrm{c}}, \mathbf{A}^{*}, \mathbf{A}^{-1}, \mathbf{A}^{\dagger} \tag{11}$$

Math — Custom Math Operators

For norms, it is convenient to have a `\pnorm{}{}` macro to typeset the p-norm as

```
\newcommand{\pnorm}[2]{\doublebars{#2}_{#1}}
```

where `\pnorm{2}{\vx}` produces

$$\|\mathbf{x}\|_p \tag{12}$$

Math — Custom Math Operators

The `\pnorm{}{}` macro can be used to create macros for common norms

```
\newcommand{\normzero}[1]{\pnorm{0}{#1}}
\newcommand{\normone}[1]{\pnorm{1}{#1}}
\newcommand{\normtwo}[1]{\pnorm{2}{#1}}
\newcommand{\norminf}[1]{\pnorm{\infty}{#1}}
\newcommand{\normnuc}[1]{\pnorm{*}{#1}}
\newcommand{\normfro}[1]{\pnorm{\mathrm{F}}{#1}}
\newcommand{\normmax}[1]{\pnorm{\mathrm{max}}{#1}}
```

which take only the argument as input and produce, for example,

$$\|\mathbf{x}\|_0 , \|\mathbf{x}\|_1 , \|\mathbf{x}\|_2 , \|\mathbf{x}\|_\infty , \|\mathbf{X}\|_* , \|\mathbf{X}\|_\mathrm{F} , \|\mathbf{X}\|_\mathrm{max} \tag{13}$$

Math — Custom Math Operators

There are many ways to denote taking the entries of a vector/matrix/tensor. I use

$$[\mathbf{x}]_i \, , [\mathbf{X}]_{i,j} \, , [\boldsymbol{\mathcal{X}}]_{i,j,k} \tag{14}$$

which uses the following `\entry{}{}` command

```
\newcommand{\entry}[2]{\brackets{#1}_{#2}}
```

Math

Special Variables and Values

Math — Special Variables and Values

Special math variables can (should!) be styled using `\mathrm{}`, `\mathsf{}` or the like.

For example, to denote the signal-to-noise ratio, we see that

```
SNR \qquad \mathrm{SNR}
```

produces

$$SNR \qquad \mathrm{SNR} \tag{15}$$

Clearly, SNR looks better and is not ambiguous.

Is $SNR = S \times N \times R$?

Math — Special Variables and Values

We can create macros to more quickly type common variables like SNR.

```
\newcommand{\snr}{\mathrm{SNR}}
```

Other good examples: SINR, ENOB, PAPR.

`\mathsf{}` also looks good: SNR, SINR.

The math sans serif font is the same sans serif one used here in this slide deck. In your papers, it will stand out better among serif text.

Math — Special Variables and Values

Some people, including myself, use `\mathrm{}` to style special values like $\mathrm{j} = \sqrt{-1}$ and $\mathrm{e} = 2.718\ldots$ because they are not variables. Their values are fixed.

And this frees j for indexing.

Math — Special Variables and Values

If there are variables like P_{tx}, $\mathbf{F}_{\mathrm{RF}}$, G_{rx}, SNR, and N_{t} that are used throughout your paper, make them macros so that you can type `\Ptx` rather than `P_{\mathrm{tx}}` every time.

Note that using `\powertransmit` instead of `\Ptx` for P_{tx} can make changes to notation easier down the road.

Math

Denoting Values in Decibels

Math — Denoting Values in Decibels

I use the convention that variables are in linear units, almost universally. Their log-based units can be explicitly denoted to help prevent confusion.

SNR and P_{tx}, for example, would be in linear units and their values in dB and dBm, respectively, would be expressed as

$$[\mathrm{SNR}]_{\mathrm{dB}}\,, [P_{\mathrm{tx}}]_{\mathrm{dBm}} \tag{16}$$

This can be achieved using macros

```
\newcommand{\todB}[1]{\brackets{#1}_{\mathrm{dB}}}
\newcommand{\todBm}[1]{\brackets{#1}_{\mathrm{dBm}}}
```

Math

Optimization Problems

Math — Optimization Problems

I typeset optimization problems using the following form

$$a = \min_{x} \ f(x) \tag{17}$$

$$\text{s.t. } g(x) \leq 0 \tag{18}$$

$$h(x) = 0 \tag{19}$$

via

```
a = \min_{x} \ & f(x) \\
\st
& g(x) \leq 0 \\
& h(x) = 0
```

where `\st` is a macro to typeset s.t. .

Math — Optimization Problems

I like using the `subequations` environment from the `amsmath` package for optimization problems.

$$a = \min_x \; f(x) \tag{20a}$$

$$\text{s.t. } g(x) \leq 0 \tag{20b}$$

$$h(x) = 0 \tag{20c}$$

This allows me to refer to this problem as "problem (20)".

Math — Optimization Problems

This was achieved using the following

```
\begin{subequations} \label{eq:subequations-main}
\begin{align}
a = \min_{x} \ & f(x) \\
\st
& g(x) \leq 0 \\
& h(x) = 0
\end{align}
\end{subequations}
```

to create the problem and `\eqref{eq:subequations-main}` to reference it. Each line within the align can be labeled as usual.

Math

Theorems, Proofs, Etc.

Math — Theorems, Proofs, Etc.

Theorems, proofs, corollaries, lemmas, definitions, and remarks can be formatted nicely using the `amsthm` package.

Theorems can be typeset using the following, for example,

```
\begin{theorem}
Theorem here...
\begin{proof}
Proof of theorem here...
\end{proof}
\end{theorem}
```

Math — Theorems, Proofs, Etc.

Based on my Beamer settings, we have

Theorem

Theorem here...

Proof.

Proof of theorem here... □

Math — Theorems, Proofs, Etc.

Corollary

Corollary here...

Lemma

Lemma here...

Remark

Remark here...

Definition (My term here)

Definition here...

Math — Theorems, Proofs, Etc.

The theorem environment can be customized using the following in the preamble.

```
\newtheoremstyle{customname}
 {spaceabove}{spacebelow}%
 {bodyfont}{indentamt}{headfont}%
 {headpunct}{headspace}{headspec}%

\theoremstyle{customname}
\newtheorem{theorem}{Theorem}
```

For more information on styling and formatting of these different environments, refer to online documentation.

Glossary of Acronyms

Glossary of Acronyms

Acronyms are used frequently in technical documents. To ensure you define each term once and only once and use the acronym consistently, the `glossaries` package comes in handy.

I usually load it in the following fashion for papers/slides, though it can be adjusted as you need it.

```
\usepackage[acronym,%
            nogroupskip,%
            nonumberlist,%
            nopostdot]{glossaries}
\makeglossaries
```

Please refer to online documentation for more information about its settings.

Glossary of Acronyms

Creating a Glossary

Glossary of Acronyms — Creating a Glossary

I create my glossary of terms in a file `glossary.tex`, which can be incorporated using `\input{glossary.tex}`.

Acronyms can be defined in the following fashion.

```
\newacronym{<key>}{<acronym>}{<full term>}
```

For example, signal-to-noise ratio (SNR) would be defined via

```
\newacronym{snr}{SNR}{signal-to-noise ratio}
```

Simply add as many `\newacronym`'s as you'd like.

Glossary of Acronyms

Usage

Glossary of Acronyms — Usage

To use the SNR acronym, simply type `\gls{snr}` anywhere you want to use it.

The first time the acronym is used, the acronym will be defined. Thereafter, it will be referred to by its acronym only. For example,

```
The \gls{snr} is very low. The \gls{snr} should be higher.
```

produces

> *The signal-to-noise ratio (SNR) is very low. The SNR should be higher.*

Glossary of Acronyms

Shorthand Macros

Glossary of Acronyms — Shorthand Macros

It can be cumbersome to type `\gls{snr}` each time for SNR, which may be frequently used. Instead, I can create a custom macro `\snr` that inserts the acronym. It's a little quicker to type and is more readable.

```
\newcommand{\snr}{\gls{snr}\xspace}
```

I have used the `xspace` package here to dynamically insert a space when appropriate.

Now, I can type

```
The \snr is very low. The \snr should be higher.
```

Glossary of Acronyms

Miscellany

Glossary of Acronyms — Miscellany

The glossary definition can be "reset" using

```
\glsresetall
```

which will redefine acronyms thereafter.

Glossaries can be printed using

```
\printglossary[type=\acronymtype]
\printglossaries
```

Refer to online documentation for more. Handling acronyms may differ for your dissertation.

Algorithms

Algorithms

I use the following two packages for algorithms, especially in IEEE papers.

```
\usepackage{algorithm}
\usepackage{algorithmic}
```

The following changes the words “Require” to “Input:” and “Ensure” to “Output:”. This is optional.

```
\renewcommand{\algorithmicrequire}{\textbf{Input:}}
\renewcommand{\algorithmicensure}{\textbf{Output:}}
```

Algorithms

An algorithm can be written as, for example,

```
\begin{algorithm}[t]
    \begin{algorithmic}[0]
        \REQUIRE $K$
        \STATE $T = 0$
        \FOR{$k = 1:K$}
            \STATE $T = T + 1$
        \ENDFOR
        \ENSURE $T$
    \end{algorithmic}
    \caption{My algorithm.}
    \label{alg:my-algorithm}
\end{algorithm}
```

Algorithms

Algorithm 1 My algorithm.

Input: K

$T = 0$

for $k = 1 : K$ **do**

$T = T + 1$

end for

Output: T

Note that `\begin{algorithm}[H]` should be used when using this in Beamer.

Algorithms

The following commands can be used within algorithms. Please refer to online documentation for more on algorithms.

```
\STATE <text>
\IF{<cond>} \STATE {<text>} \ELSE \STATE{<text>} \ENDIF
\IF{<cond>} \STATE {<text>} \ELSIF{<cond>} \STATE{<text>} \ENDIF
\FOR{<cond>} \STATE {<text>} \ENDFOR
\FOR{<cond> \TO <cond> } \STATE {<text>} \ENDFOR
\FORALL{<cond>} \STATE{<text>} \ENDFOR
\WHILE{<cond>} \STATE{<text>} \ENDWHILE
\REPEAT \STATE{<text>} \UNTIL{<cond>}
\LOOP \STATE{<text>} \ENDLOOP
\REQUIRE <text>
\ENSURE <text>
\RETURN <text>
\PRINT <text>
\COMMENT{<text>}
\AND, \OR, \XOR, \NOT, \TO, \TRUE, \FALSE
```

IEEEtran

IEEEtran

Getting Started

IEEEtran — Getting Started

The document styling provided by `IEEEtran` is really great. Its documentation offers a concise summary of how to use it.

A single-column journal (draft) can be constructed using, for example,

```
\documentclass[journal,%
               draftclsnofoot,%
               onecolumn,%
               12pt,%
               oneside]{IEEEtran}
```

IEEEtran

Authors, Affiliations, and Acknowledgments

IEEEtran — Authors, Affiliations, and Acknowledgments

Authors, affiliations, and acknowledgments in journals can be done so using the `\thanks{...}` command.

```
\author{First Last, %
        First Last, and %
        First Last%
        \thanks{F.~Last is with ...}
}
```

Line breaks `\\` can be used within the list of authors.

IEEEtran — Authors, Affiliations, and Acknowledgments

A simple author block for conference publications can be created using

```
\author{
  \IEEEauthorblockN{First Last\IEEEauthorrefmark{1}%
                    and %
                    First Last\IEEEauthorrefmark{2}}
  \IEEEauthorblockA{\IEEEauthorrefmark{1}Affiliation 1.}
  \IEEEauthorblockA{\IEEEauthorrefmark{2}Affiliation 2.}
}
```

IEEEtran

Bibliographies

IEEEtran — Bibliographies

Bibliographies using `IEEEtran` use the `cite` package.

```
\usepackage{cite}
```

Bibliographies can be printed using

```
\bibliographystyle{bibtex/IEEEtran}
\bibliography{refs}
```

which will be populated with BibTeX entries in `refs.bib` that were cited using `\cite{}`.

IEEEtran — Bibliographies

Citing a journal/magazine in IEEE format is of the form

> *J. K. Author, "Name of paper," Abbrev. Title of Periodical, vol. x, no. x, pp. xxx–xxx, Abbrev. Month, year.*

Citing a conference paper in IEEE format is of the form

> *J. K. Author, "Title of paper," in Abbreviated Name of Conf., (location of conference is optional), (Month and day(s) if provided) year, pp. xxx–xxx.*

Note that these both use **abbreviated** names of the periodical and conference. Also note that each word in the publication titles are **not capitalized**.

IEEEtran — Bibliographies

In addition to `bibtex/IEEEtran`, the `IEEEtran` styling files also come with `bibtex/IEEEabrv`, which is an abbreviations file containing the preferred abbreviations for a variety of IEEE publications.

For example, it contains the following, which maps the identifier `IEEE_J_WCOM` to the abbreviation "IEEE Trans. Wireless Commun." via the line

```
@STRING{IEEE_J_WCOM = "{IEEE} Trans. Wireless Commun."}
```

This allows you to use `journal = IEEE_J_WCOM` in your BibTeX entries, rather than explicitly writing out the abbreviated periodical title each time.

You can create custom abbreviations as you'd like.

IEEEtran — Bibliographies

To print your bibliography using these abbreviations, you use

```
\bibliographystyle{bibtex/IEEEtran}
\bibliography{bibtex/IEEEabrv,refs}
```

where `refs.bib` contains BibTeX entries that use these abbreviations. (Not every single entry needs to.)

IEEEtran — Bibliographies

Always be sure to use an en dash `--` for page ranges and to use accented characters as needed.

Also, wrap capitalized acronyms/words using `{}` (e.g., `{MIMO}`) in the title to ensure that when displayed it will be uppercase, whereas the rest of the title's capitalization will be properly handled by the IEEE BibTeX file.

Wrapping your entire title in double quotes will preserve its entire styling. This should be avoided so as to let the IEEE BibTeX properly style your bibliography.

Hope this helps.