

N_OT_EX: L^AT_EX Macros for Mathematical Notation

Emin Martinian
emin@alum.mit.edu

Abstract

The N_OT_EX package proposes a consistent system for mathematical notation as well as a corresponding set of L^AT_EX macros. The goal of the N_OT_EX system is to avoid low-level T_EX commands for notational concepts so that changing notation can be accomplished by modifying a few macros instead of replacing every occurrence of a given notation in the entire manuscript.

1 Introduction

Notation can have a dramatic effect on readability. Unfortunately, an author often stumbles upon the best notation near the end of the writing process. At that point, rewriting the manuscript in the new notation is often time-consuming and error prone. One solution is to use macros instead of direct, low-level formatting commands. This allows the author to easily modify a few macros instead of manually changing every occurrence of an entry. The N_OT_EX package available from http://csua.berkeley.edu/~emin/source_code/notex/notex.tgz provides a system of notation and a set of L^AT_EX macros that I have found useful in achieving this goal.

1.1 Usage

To use the N_OT_EX package simply put the provided L^AT_EX files in the place where you keep your style files and add the line `\usepackage{all_defs}` to your L^AT_EX file. You can then use the macros provided in the sub-packages described below.

The basic philosophy of the N_OT_EX package is that no low-level L^AT_EX commands should appear in your manuscript. Instead, notational concepts should be defined in the appropriate style files. This has two main advantages. First, if you want to change notation you only need to modify the appropriate sub-package. Second, if you break out your notation into a separate style file, then other people can easily adopt your notation leading to more consistency in the literature.

1.2 Generics

The main tool in the N_OT_EX package is the `generic_defs.tex` sub-package, which defines a set of generic concepts such as sequences, random variables, *etc.* The idea is that instead of using a low-level formatting command to indicate information about a variable (*e.g.*, formatting a vector in bold type), one can use the generic macro.

For example, I prefer to represent random variables in a sans-serif format and vectors or sequences in bold. Thus, the commands `\genericRV{x}`, `\genericS{x}`, and `\genericRVS{x}` would produce x , \mathbf{x} , and \mathbf{x} representing a random variable, a non-random sequence, and a random sequence all of which are named “ x ”.

A key counterpart to any vector or sequence notation is a notation to index components of the vector or sequence. The command `\genericC` is used for this purpose. For example, the command `\genericC{x}{i}` would produce x_i indicating the i th component of the

vector \mathbf{x} . Note that we use `\genericC{x}{i}` instead of `\genericC{\genericS{x}}{i}` because we are taking about the i th component, which is a scalar element of the vector \mathbf{x} . The command `\genericCSub` is used to indicating a subsequence. For example, the command `\genericCSub{\genericRVS{x}}{i}{j}` would produce \mathbf{x}_i^j indicating components i through j of the random vector \mathbf{x} .

Often one has a series of vectors in time and needs to denote the i th vector in time of the sequence. In this case, the command `\genericT{\genericS{x}}{i}` can be used to produce $\mathbf{x}[i]$. Sub-sequences in time can be indicated via `\genericTSub{\genericS{x}}{i}{j}` to produce $\mathbf{x}[i:j]$. You will also need to use the `amsmath` package (via `\usepackage{amsmath}`) to use the `\genericTSub` command.

Various combinations of these commands are also available. For example, to indicate components i through j of the vector at time t , you could use

```
\genericTCSub{\genericS{x}}{t}{i}{j}
```

to produce $\mathbf{x}_i^j[t]$.

Finally, it make become somewhat tedious to build long expressions using the various generic commands. Therefore, one can define a sequence of abbreviations such as

```
\newcommand{\src}{s} % name of basic source variable
\newcommand{\sSrc}{\genericS{\src}} % source sequence
\newcommand{\rvSrc}{\genericRV{\src}} % random source sample
\newcommand{\rvSrcT}[1]{\genericT{\rvSrc}{#1}} % random source sample at time t
\newcommand{\rvsSrc}{\genericRVS{\src}} % random source vector
\newcommand{\srcT}[1]{\genericT{\src}{#1}} % source at time t
```

in a separate style file and use these commands in the manuscript. Then, to switch the name of the source variable, one only needs to change a single character in the `\src` macro and all occurrences of the source variable (whether they are random variables, vectors, *etc*) will change accordingly.

1.3 Non-mathematical Definitions

The `defs.tex` sub-package provides a variety of non-mathematical notation such as the macros `\ie`, `\etc`, `\eg`, `\iid`, `\apriori`, `\etal`, and `\cf`, for simple abbreviations. In addition, the macros `\thrm`, `\prop`, `\lemma`, `\defn`, and `\corol` are provided to define Theorem like notation. Finally, `\chapref`, `\secref`, `\figref`, `\tabref`, `\thrmref`, `\lemref`, `\propref`, `\corolref`, `\appref`, and `\defref` are available to refer to other parts of the manuscript in a consistent way.

1.4 Mathematical Definitions

The `math_defs.tex` sub-package provides basic mathematical notation most of which are self explanatory.

1.5 Random Variables

The `rv_defs.tex` sub-package defines bold sans-serif fonts used for random variables.