This is a sample test, just to test Typst capabilities and make sure my environment is properly set up. Apparently everything is working properly! Using Typst in VSCode is super simple, I just had to download the Typst LSP extention and everything was setup automatically.

Section title

This is a section, and there is a title just above this paragraph. It is formated as a header because I used a "=" symbol before it in the source code.

Subsection title

To change header level, I simply use more "=" symbols. The subsection title above this paragraph has two "=" in front of it in the source code and is thus rendered as a level 2 header.

What about paragraphs?

Creating a paragraph is easy: symply write some text.

To create another paragraph, I just have to use two line breaks, just like this!

Comparison with LaTeX

At first glance, Typst source code seems more straight forward that LaTeX, where I would have had to use "\section{}" and "\subsection{}" for sections and subsections.

New paragraphs require "\\" in LaTeX, but not in Typst.

Note that I have to escape the backslashes ("\") in the source code, using another backslash.

Compiling

Compiling is very fast and happens in a matter of tenths of a second. Very good for iterating quickly! I would be interested in knowing how fast an actual article with many figures and images would perform.

With the Typst LSP extention for VSCode, compiling happens on save. It creates a pdf file next to the typ source file, with no additional junk. Pretty clean!

Without the extension, compiling a typ file can be done through a simple command, calling the Typst compiler on the source file.

Lists

Unumbered lists are done using "-":

- this is an item
- this is another item
- …

Numbered list simply use "+" instead of "-":

1. this is the first item
2. this is the second item
3. …

Obviously, lists can be nested:

1. first of all, I can nest numbered lists:
   1. this is sub-item 1
   2. this is sub-item 2
2. and unumbered lists too:
   - a sub-item

- another sub-item that has a nested, numbered, list:
  1. look at me!
  2. and at me too!

Figures

Cute kitties are super cool so I need to have a way to put some in my Typst documents! Look at this cute kitty in Game of Thrones:

Figure 1: A cute kitty in *Game of Thrones*

On a side note, the Figure 1 has been generated by the *Flux.1* AI model. Have you noticed that I just made a clickable reference to a figure AND that I just learned how to write *italicized* text?

About this document

This document has been written to learn and test the Typst language. As someone with some LaTeX experience, I'm curious to see how Typst feels and how it could be usefull to me so I just write this to try out it features.

I'm following the official Typst tutorial [1].

Math

Obviously, since I'm passionate about generative AI, I need to be able to type quite advanced math pretty easily. Let's test this out in this section!

Inline equations

Inline equations work similarly to LaTeX, simply by using "$" (I had to escape this symbol using a backslash).

Diffusion models usually try to sample new a data point x_0 following the training data distribution, using what's called a reverse diffusion process to progressively denoise a pure gaussian noise sample $x_T \hookrightarrow \mathcal{N}(0, 1)$.

It wasn't obvious how to write the round $\mathcal{N}$, but apparently you can symply use the "cal" function in math mode [2]. I can't wait to learn how to write inline code just to be able to properly write "cal" instead of using double quotes by the way! It also demonstrates how Typst uses functions in math

mode to define custom symbols. I guess I will learn how to define custom functions latter on in this tutorial?

Writing $\hookrightarrow$ is interesting. Indeed, it shows that symbols are defined as elements of classes. $\hookrightarrow$ is the "hook" element of the "arrow.r" (right arrow) class.

Full equations

Writing inline equations is easy. But it's as easy to put them on their own line! I just need to put the equation on it's own line in the source code, separating it from the previous and next paragraphs with line breaks and placing spaces between the "$" symbols and the equation.

The reverse diffusion process of a diffusion model is defined by it's *reverse diffusion kernel* $q(x_t \ |x_{t-1})$. Since the process is markovian, $x_T, x_{T-1}, ..., x_1$ are independent and we can thus get the joint probability distribution of these variables conditionally to x_0:

$$q(x_1, ..., x_T \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

I notice two things right of the bat:

- to group things together, we use parenthesis "()" and not curly brackets "{}"
- the size of the parenthesis in the compiled equations automatically adapt to what is inside them, no need for the "\left(... \right)" synthax.

Bibliography

[1] Typst, "Tutorial – Typst Documentation." 2024.

[2] Typst, "Variants Functions – Typst Documentation." 2024.