正文标题

作者

潜伏

日期

2025-01-01

第一章

正文开始...

模板的基本使用

使用本模板之前，请阅读模板的使用说明文档。下面是本模板使用的基本样式:

```
#import "@preview/cumcm-muban:0.1.0": *
#show: thmrules



// 参考文献
#bib(bibliography("refs.bib"))

// 附录
= 附录 A
```

Listing 1: 基本样式

请确保你的文章内容符合要求，包括但不限于页数、格式、内容等。

下面给出写作与排版上的一些示例。

图片

在写作中，我们经常需要插入图片。Typst 支持的图片格式有 “png”, “jepg”, “”gif“”, “svg”，其他类型的图片无法插入文档。我们可以通过改变参数 `width` 来改变图片的大小，详见 typst/docs/image。下面是一些图片插入的示例:

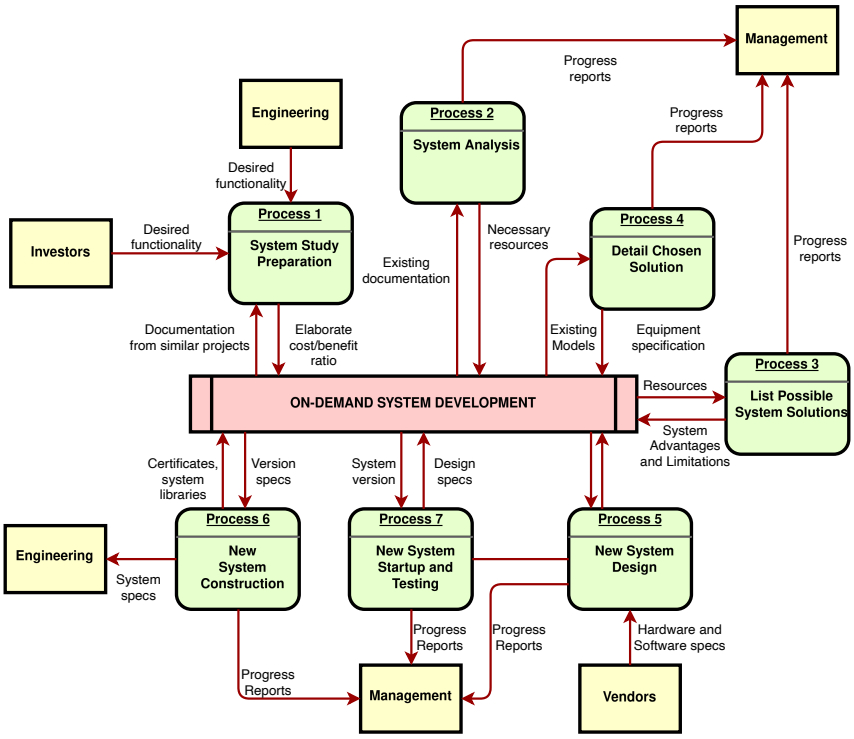
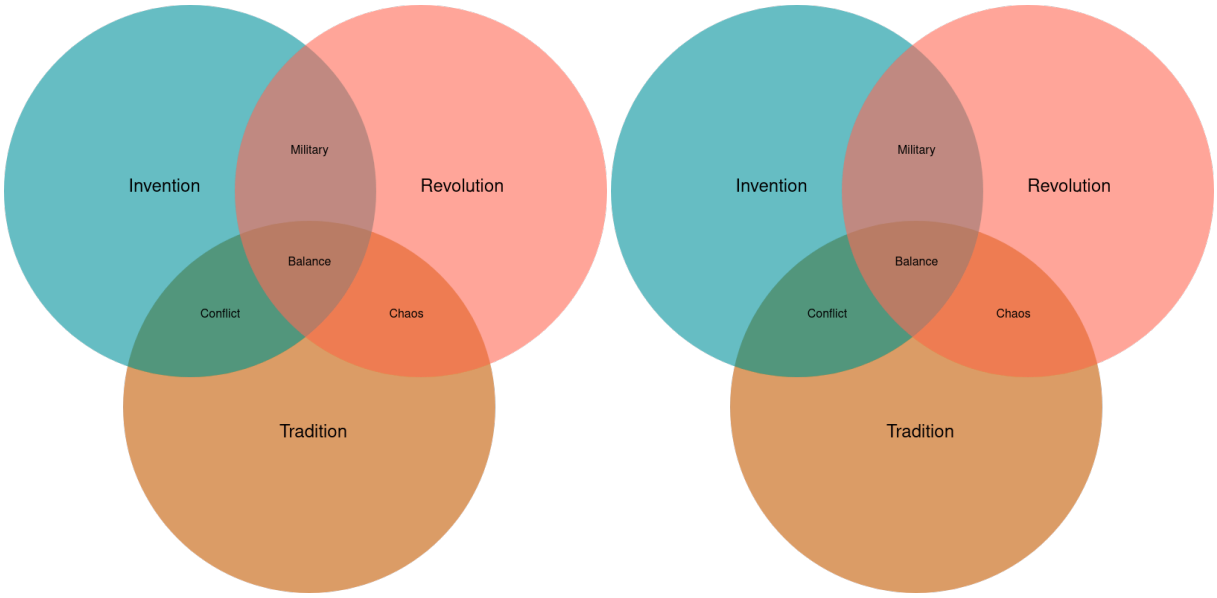


Figure 1: 单图示例



（a）venn 图　　（b）venn 图

Figure 2: 多图并排示例

表格

数学建模中表格有助于数据的整理与展示。Typst 支持使用 `table` 来插入表格，详见 typst/docs/table。下面是一些表格插入的示例:

	Area	Parameters
Cylinder	$\pi h \frac{D^2-d^2}{4}$	h: height D: outer radius d: inner radius
Tetrahedron	$\frac{\sqrt{2}}{12}a^3$	a: edge length

Table 1: 表格示例

Names	Properties		Creators
	Type	Size	
Machine	Steel	5 cm^3	John p& Kate
Frog	Animal	6 cm^3	Robert
Frog	Animal	6 cm^3	Robert

Table 2: 三线表示例

```
#figure(
  table(
    columns: 4,
    align: center + horizon,
    stroke: none,
    table.hline(),
    table.header(
      table.cell(rowspan: 2, [*Names*]),
      table.cell(colspan: 2,[*Properties*],),
      table.hline(stroke: 0.6pt),
      table.cell(rowspan: 2, [*Creators*]),
      [*Type*], [*Size*],

    ),
    table.hline(stroke: 0.4pt),
    [Machine], [Steel], [5 $"cm"^3$], [John p& Kate],
    [Frog], [Animal], [6 $"cm"^3$], [Robert],
    [Frog], [Animal], [6 $"cm"^3$], [Robert],
    table.hline()
  ),
  caption: "表格示例"
)
```

更多使用方法可以查看 typst/docs/table。

公式

数学建模中，公式的使用是必不可少的。Typst 可以使用 Typst 原生语法插入公式，参考 typst/docs/math。下面是一些公式插入的示例:

首先是行内公式，例如 $a^2 + b^2 = c^2$。行内公式使用 `$$` 包裹，公式和两端的 `$$` 之间没有空格。

其次是行间公式，例如:

$$\iiint_{\Omega}(\frac{\partial P}{\partial x}+\frac{\partial Q}{\partial y}+\frac{\partial R}{\partial z})dv=\oiint_{\Sigma}Pdydz+Qdzdx+Rdxdy$$

式（1）是高斯公式。行间公式使用 $$ 环境包裹，公式和两端的 $$ 之间至少有一个空格。

公式内可以使用换行符 \ 换行。若需要对齐，每行可以包含一个或多个对齐点 & 对其进行对齐。例如:

$$\begin{aligned}\sum_i b_i &= \sum_i \sum_{h,j\neq i} \frac{\sigma_{hj}(i)}{\sigma_{hj}} \\ &= \sum_{h\neq j} \frac{1}{\sigma_{hj}} \sum_{i\neq h,j} \sigma_{hj}(i)\end{aligned}$$

& 是对齐的位置，& 可以有多个，但是每行的个数要相同。

矩阵输入示例:

$$A=\begin{pmatrix}a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{pmatrix}$$

分段函数可以使用 case 环境:

$$f(x)=\begin{cases}0 & x \text{ 为无理数},\\ 1 & x \text{ 为有理数}.\end{cases}$$

假如要公式里面有个别文字，需要把这部分放在 text 环境里面,即 `text[文本内容]` 。

如果公式中有个别需要加粗的字母，可以使用 `bold()` 进行加粗。如，$\alpha a \boldsymbol{\alpha} \boldsymbol{a}$。

以上仅为一些简单的公式示例，更多的公式使用方法可以查看 typst/docs/math

另外，如果需要插入 LaTeX 公式可以使用外部包 mitex。

定理环境

在本模板中，我们定义了一些常用的定理环境，包括 theorem 、 lemma、corollary、assumption、 conjecture、axiom、principle、problem、example、proof、solution。可以根据论文的实际需求合理使用。

定义 1: 这是一个定义

引理 1: 这是一个引理

推论 1: 这是一个推论

假设 1: 这是一个假设

猜想 1: 这是一个猜想

公理 1: 这是一个公理

定律 1: 这是一个定律

问题 1: 这是一个问题

例 1: 这是一个例子

证明 1: 这是一个证明

解 1: 这是一个解

其他功能

脚注

利用 `#footnote(脚注内容)`可以生产脚注[1]

无序列表与有序列表

无序列表例:

- 元素 1
- 元素 2
- ...

有序列表例:

1. 元素 1
2. 元素 2
3. ...

字体粗体与斜体

如果想强调部分内容,可以使用加粗的手段来实现。加粗字体可以用 `*需要加粗的内容*` 或 `#strong[需要加粗的内容]` 来实现。例如: 这是加粗的字体，**This is bold fonts**。

中文字体没有斜体设计，但是英文字体有。斜体 *Italics*。

参考文献与引用

参考文献对于一篇正式的论文来说是必不可的，在建模中重要的参考文献当然应该列出。Typst 支持使用 BibTeX 来管理参考文献。在 `refs.bib` 文件中添加参考文献的信息，然后在正文中使用 `#cite(<引用的文献的 key>)` 来引用文献。例如: [1]。最后通过 `#bib(bibliography("refs.bib"))` 来生成参考文献列表。

参考文献

[1] ASTLEY R, MORRIS L. At-scale impact of the Net Wok: A culinarically holistic investigation of distributed dumplings[J]. Armenian Journal of Proceedings, 2020, 61: 192-219.

[1] 脚注例

附录 1:

线性规划 - Python 源程序

```
import numpy as np
from scipy.optimize import linprog

c = np.array([2, 3, 1])
A_up = np.array([[-1, -4, -2], [-3, -2, 0]])
b_up = np.array([-8, -6])

r = linprog(c, A_ub=A_up, b_ub=b_up, bounds=((0, None), (0, None), (0, None)))

print(r)
```

附录 2:

非线性规划 - Python 源程序

```
from scipy import optimize as opt
import numpy as np
from scipy.optimize import minimize

# 目标函数
def objective(x):
  return x[0] ** 2 + x[1] ** 2 + x[2] ** 2 + 8

# 约束条件
def constraint1(x):
  return x[0] ** 2 - x[1] + x[2] ** 2  # 不等约束

def constraint2(x):
  return -(x[0] + x[1] ** 2 + x[2] ** 2 - 20)  # 不等约束

def constraint3(x):
  return -x[0] - x[1] ** 2 + 2

def constraint4(x):
  return x[1] + 2 * x[2] ** 2 - 3  # 不等约束

# 边界约束
b = (0.0, None)
bnds = (b, b, b)

con1 = {'type': 'ineq', 'fun': constraint1}
con2 = {'type': 'ineq', 'fun': constraint2}
con3 = {'type': 'eq', 'fun': constraint3}
con4 = {'type': 'eq', 'fun': constraint4}
cons = ([con1, con2, con3, con4])  # 4个约束条件
x0 = np.array([0, 0, 0])
# 计算
solution = minimize(objective, x0, method='SLSQP',  bounds=bnds, constraints=cons)
x = solution.x

print('目标值: ' + str(objective(x)))
print('答案为')
```

```
print('x1 = ' + str(x[0]))
print('x2 = ' + str(x[1]))
```