

到目前为止,我们配置了WSL下的RIES(符号近似计算软件),Docker封装后使用WSL启动的SageMath(Python为主的数学计算软件),以及VScode下的LaTeX环境.想要快速查询操作指南只需关注 >[!note] 的部分即可.

WSL(Windows Subsystem for Linux)下的工具

WSL的配置注意事项

如果你的Windows上还未搭载WSL,参考[设置 WSL 开发环境 | Microsoft Learn](#),而WSL配置Docker参见[WSL 上的 Docker 容器入门 | Microsoft Learn](#).请务必按顺序参考处理.

下面我们解决一点配置WSL和Docker时可能会遇到的问题.

② 为什么我在终端内输入 `wsl` 后显示 `docker-desktop:...` ?

若你在终端内输入 `wsl` 后看到的是 `docker-desktop:...`,那么说明你进入的是 **Docker自动创建的WSL发行版**.当Docker Desktop程序先被安装时,它会被自动设置为默认WSL发行版.

参考[WSL 上的 Docker 容器入门 | Microsoft Learn](#)里的教程以切换为其他你安装的WSL发行版,或按照以下方式操作以切换WSL默认发行版.

1. 在Powershell中查询WSL的发行版

输入

```
wsl -l -v
```

后,应该会返回类似

NAME	STATE	VERSION
* docker-desktop	Running	2
Ubuntu-22.04	Stopped	2

的文字.记下这个非docker-desktop的NAME.

2. 切换默认WSL发行版

在Powershell内输入

```
wsl --setdefault [Ubuntu-22.04]
```

将括号的NAME替换为实际的版本号,即可解决问题.这样之后每次输入 `wsl` 都会进入这个发行版的WSL.

② 为什么我进入WSL后显示 `root@...` ?

这表明你进入的是WSL的**root用户**.它拥有最高权限,但一般情况下不会影响我们的使用.

不过考虑到安全原因,我们可以按照以下操作手动添加一个用户.
在root会话中完成用户配置:

1. 创建普通用户

输入:

```
adduser [id]
```

替换 `[id]` 为你想使用的id,随后按提示设置密码和用户信息.

2. 授予 `sudo` 权限

随后,输入:

```
usermod -aG sudo [id]
```

以给这个名为id的用户添加sudo权限.sudo权限依然是linux的最高权限,但是在输入命令时需要带 `sudo` 前缀.

3. 设置默认用户

然后我们将这个用户设置为默认用户.输入:

```
bash echo -e "\n[user]\ndefault = [id]" >> /etc/wsl.conf
```

即可.

4. 退出并重启 WSL

在WSL内输入:

```
exit
```

随后,在Powershell中输入:

```
wsl --terminate [Ubuntu]
```

并将 [Ubuntu] 替换为你实际的WSL发行版.这会关掉正在运行的WSL,然后使用

```
wsl
```

以重新启动.

现在,重新进入WSL后你应该看到形如

```
(id)@...
```

的内容,表示用户设置完成.

🔗 **为什么我的Docker测试hello-world时报错:** <https://registry-1.docker.io/v2/> ?

请参见笔记末尾的网址,其中有完整解答.同时,笔记附上了可在Docker Desktop的Docker Engine处配置的json文件.复制后应该可以解决问题.

下面假设WSL和Docker都已经配置完成.

🔗 WSL启动

启动终端,输入 `wsl` 即可.

WSL内RIES的配置

接下来我们在WSL内配置RIES.网站:[RIES - Find Algebraic Equations, Given Their Solution at MROB](#).

1. 安装必要的工具

```
sudo apt update
sudo apt install -y gcc libgmp-dev libmpfr-dev
```

- gcc 是C编译器
- libgmp-dev 和 libmpfr-dev : 高精度数学库 (ries 依赖这些库)

2. 下载源码

```
wget https://www.mrob.com/pub/ries/src/ries.c.txt -O ries.c
```

3. 编译程序

```
gcc -O3 -o ries ries.c -lmpfr -lgmp -lm
```

- -O3 : 启用高级优化
- -o ries : 指定输出文件名
- -lmpfr -lgmp -lm : 链接数学库

于是我们完成了RIES的编译,接下来使用 `./ries` 就可以调用.我们测试一下RIES:

≡ RIES中 π 的"近似"

如果输入 `./ries 3.1415926` ,得到的应该是类似

```
Your target value: T = 3.1415926                                mrob.com/ries
x = pi                                                         for x = T + 5.35898e-08 {29}
...
```

的结果.♣

🔗 调用RIES

在WSL内输入 `./ries [target-value]` 即可调用.
若要查询RIES的操作,输入 `./ries` 即可.

Docker封装SageMath并在WSL内调用及添加alias

然后我们开始在Docker内封装SageMath.网址[SageMath - Open-Source Mathematical Software System](https://www.sagemath.org/open-source).主页Documentation内含有操作手册.

1. 拉取SageMath镜像

在WSL内输入:

```
docker pull sagemath/sagemath:latest
```

此时Docker会自动拉取SageMath最新的镜像.而如果在后续使用 `docker [operation] sagemath/sagemath:[最新的版本号, 如10.6]`, Docker会自动创建一个latest和版本号的映射.

现在你应该可以使用

```
docker run -it --rm sagemath/sagemath:latest sage
```

- `-it`: 进入交互模式
- `--rm`: 退出时自动删除容器

来启动SageMath在WSL内的交互页面了.我们用一个简单的例子测试一下SageMath:

≡ S_3 的子群和生成元

进入SageMath的交互页面,输入:

```
# 定义对称群 S3
G = SymmetricGroup(3)

# 获取所有子群
all_subgroups = G.subgroups()

# 打印子群信息
print("S3 的所有子群 (共 {} 个):".format(len(all_subgroups)))
for H in all_subgroups:
    print(f"阶: {H.order()}, 生成元: {H.gens()}")
```

得到结果:

S3 的所有子群（共 6 个）：

阶：1，生成元： $(())$ ，

...

阶：6，生成元： $((1,2,3), (2,3))$



如果想要退出交互界面,只需 `ctrl+d` 即可.

或者,我们也可以使用

```
docker run -it --rm -p 8888:8888 sagemath/sagemath:latest sage-jupyter
```

来运行Jupyter Notebook,其提供了一个类似Mathematica Notebook的网页端计算工具.

运行Jupyter Notebook后,需要在终端中找到Token,并且在浏览器中输入 `http://localhost:8888` 进入,而退出的方法是 `ctrl+c`. 注意,此时的Jupyter Notebook内的程序是临时储存的,终端停止运行就会消失.

(未alias)SageMath操作

1.(交互页面): 启动: `docker run -it --rm sagemath/sagemath:latest sage`, 关闭: `ctrl+d`.

2.(Jupyter Notebook): 启动: `docker run -it --rm -p 8888:8888 sagemath/sagemath:latest sage-jupyter`, 关闭: `ctrl+c` (还未分配指定保存地址)

但是,每次都要输入 `docker run ...` 那也太麻烦了,并且如果我们想要保存我们在Notebook里的程序,应该怎么做呢?

我们可以给对应代码添加一个alias.对应操作如下:

1. 创建Jupyter Notebook的保存路径

使用WSL,在 `\\wsl.localhost\Ubuntu\home\[你使用的WSL的用户名]` (或其他你想要的地方,只需对应更改地址即可),创建文件夹 `sage_notebooks` :

```
mkdir -p /home/[你的用户名]/sage_notebooks
```

这个文件夹将会储存你的Jupyter Notebook保存的笔记本.

2. 进入nano编辑器

在WSL内输入:

```
nano ~/.bashrc
```

3. 创建WSL内的SageMath交互页面的alias

在nano编辑器内输入

```
alias sage-shell='docker run -it --rm sagemath/sagemath:latest  
sage'
```

4. 创建可保存笔记本的Jupyter Notebook的alias

同样,在nano编辑器内输入:

```
alias sage-jupyter='docker run -it --rm -p 8888:8888 -v  
/home/[你的用户名]/sage_notebooks:/home/sage/notebooks  
sagemath/sagemath:latest sage-jupyter'
```

5. 退出nano编辑器并应用更改

确定alias输入完毕后,按 `ctrl+x`,接着按 `y` 确认保存,然后在WSL内输入

```
bash source ~/.bashrc
```

来应用保存.

然后我们就可以用alias来启动SageMath了.

(alias后)启动SageMath

1.(交互页面): 启动: `sage-shell`,关闭: `ctrl+d`.

2.(Jupyter Notebook): 启动: `sage-jupyter`,关闭: `ctrl+c`.

VSCode+Sumatra的LaTeX环境配置

VSCode的配置要简单很多,我在这里大量参考了[Visual Studio Code \(vscode\)配置LaTeX - 知乎](#)这篇文章,感谢Ali-loner大佬的付出.

1. 下载Texlive

Texlive可在清华镜像站下载:[Index of /CTAN/systems/texlive/Images/ | 清华大](#)

[学开源软件镜像站 | Tsinghua Open Source Mirror](#).因为我们在这里选用vscode, 所以不用勾选texlive自带的TeXworks作为前端.**注意得等到提示"欢迎进入TeX Live的世界!"才算安装完成.**

2. 下载VSCode

VSCode的安装和一般应用无异.官网:[Visual Studio Code - Code Editing. Redefined](#)

只有几个要点:(1) 记得**修改安装路径**; (2) 一定要选上**"添加到PATH"**这个选项,能省很多麻烦; (3) 可根据喜好决定是否添加快捷方式.

3. 下载Sumatra

依然是官网下载即可:<https://sumatrapdfreader.org/free-pdf-reader>

4. 汉化VSCode和配置Latex Workshop

按快捷键 `ctrl+shift+x` 或找到"Extensions"打开扩展页面.搜索"Chinese",选择"Chinese (Simplified) Language Pack for Visual Studio Code"插件;搜索"latex workshop",选择第一个"LaTeX Workshop"插件.重启确保安装完毕.

5. 配置json文件

按 `ctrl+,` 或找到"管理"然后点击"设置"进入设置页面.在左上角找到"打开设置(json)".

输入:

```
{
  "latex-workshop.latex.autoBuild.run": "never",
  "latex-workshop.showContextMenu": true,
  "latex-workshop.intellisense.package.enabled": true,
  "latex-workshop.message.error.show": false,
  "latex-workshop.message.warning.show": false,
  "latex-workshop.latex.tools": [
    {
      "name": "pdflatex",
      "command": "pdflatex",
      "args": [
        "-synctex=1",
        "-interaction=nonstopmode",
        "-file-line-error",
        "%DOCFILE%"
      ]
    },
    {
      "name": "latexmk",
      "command": "latexmk",
      "args": [
        "-synctex=1",
```



```

        "-interaction=nonstopmode",
        "-file-line-error",
        "-pdf",
        "-outdir=%OUTDIR%",
        "%DOCFILE%"
    ]
},
{
    "name": "xelatex",
    "command": "xelatex",
    "args": [
        "-synctex=1",
        "-interaction=nonstopmode",
        "%DOCFILE%"
    ]
},
{
    "name": "biber",
    "command": "biber",
    "args": [
        "%DOCFILE%"
    ]
}
],
"latex-workshop.latex.recipes": [
    {
        "name": "XeLaTeX",
        "tools": [
            "xelatex"
        ]
    },
    {
        "name": "PDFLaTeX",
        "tools": [
            "pdflatex"
        ]
    },
    {
        "name": "Biber",
        "tools": [
            "biber"
        ]
    }
],
{

```

```

        "name": "LaTeXmk",
        "tools": [
            "latexmk"
        ]
    },
    {
        "name": "xelatex -> biber -> xelatex*2",
        "tools": [
            "xelatex",
            "biber",
            "xelatex",
            "xelatex"
        ]
    },
    {
        "name": "pdflatex -> biber -> pdflatex*2",
        "tools": [
            "pdflatex",
            "biber",
            "pdflatex",
            "pdflatex"
        ]
    }
],
"latex-workshop.latex.clean.fileTypes": [
    "*.aux",
    "*.bbl",
    "*.blg",
    "*.idx",
    "*.ind",
    "*.lof",
    "*.lot",
    "*.out",
    "*.toc",
    "*.acn",
    "*.acr",
    "*.alg",
    "*.glg",
    "*.glo",
    "*.gls",
    "*.ist",
    "*.fls",
    "*.log",
    "*.fdb_latexmk"
]

```

```

],
"latex-workshop.latex.autoClean.run": "onFailed",
"latex-workshop.latex.recipe.default": "xelatex -> biber ->
xelatex*2",
"latex-workshop.view.pdf.internal.synctex.keybinding": "double-
click",
"editor.unicodeHighlight.nonBasicASCII": false,
"editor.largeFileOptimizations": false,
"latex-workshop.view.pdf.viewer": "external",
"latex-workshop.view.pdf.external.viewer.command": "[你的
SumatraPDF的地址,形如C:/.../SumatraPDF.exe]",
"latex-workshop.view.pdf.external.viewer.args": [
    "-reuse-instance",
    "-forward-search",
    "%TEX%",
    "%LINE%",
    "%PDF%"
],
"latex-workshop.view.pdf.external.synctex.command": "[你的
SumatraPDF的地址,形如C:/.../SumatraPDF.exe]",
"latex-workshop.view.pdf.external.synctex.args": [
    "-reuse-instance",
    "-forward-search",
    "%TEX%",
    "%LINE%",
    "%PDF%"
],
"latex-workshop.synctex.afterBuild.enabled": true,
"security.workspace.trust.untrustedFiles": "open",
"editor.mouseWheelZoom": true,
"editor.minimap.enabled": false
}

```

注意把括号附近的文字替换为你的Sumatra的安装地址,并且我在这个json文件中做了对Ali-Ioner的json的一些改动,但不影响正常使用.比较重要的有:

- 将 BibTeX 替换为 Biber ,支持UTF-8编码(这会导致引用参考文献时使用不同的代码),
- 设置默认编译为 xelatex -> biber -> xelatex*2 ,
- ctrl+鼠标滚轮 可以缩放代码界面.

6. 配置Sumatra

打开Sumatra,在左上角找到 设置-高级选项 .使用 ctrl+f 搜索

EnableTeXEnhancements ,更改为 EnableTeXEnhancements = true .这是为了启

动反向搜索.保存.

随后,在 `EnableTeXEnhancements` 附近找到 `InverseSearchCmdLine`,更改为

```
InverseSearchCmdLine = "[你的VSCode的地址,如C:\Microsoft VS Code\Code.exe]" --goto %f:%l .
```

或者,在 设置-选项 中找到"设置反向搜索命令行",输入:

```
"[你的VSCode的地址,如C:\Microsoft VS Code\Code.exe]" --goto %f:%l
```

随即你应该可以使用VSCode进行LaTeX的写作,并实现和Sumatra的双向查看.

7. VSCode自动编译LaTeX为pdf,格式化TeX文档(可选)

首先我们设置LaTeX的自动编译,在设置中找到 `Files:Auto Save`,选择 `afterDelay`,使得每一秒可以自动保存.

随后,在 `json` 文件中找到 `"latex-workshop.latex.autoBuild.run":`

`"never"` 一项,将 `never` 改为 `onSave`.这样我们就实现了LaTeX的自动编译.

然后我们设置TeX文档的自动格式化,使得代码更加美观.在 `json` 文件末尾复制:

```
{
  "editor.formatOnSave": true,
  "[tex]": {
    "editor.formatOnSave": true
  },
  "latex-workshop.latexindent.platform": "texlive",
  "latex-workshop.formatting.latexindent.path": "latexindent",
  "latex-workshop.formatting.enabled": true
}
```

保存即可.

注意:此方法会导致自动编译出来的pdf页面频繁地跳出.

☰ VSCode内测试LaTeX.

在VSCode内输入

```
\documentclass{article} % 定义文档类型
\usepackage{ctex} % 引入中文宏包
\begin{document}
你好, \LaTeX!
\end{document}
```

保存为 .tex 格式(代码被高亮),在右边找到"TeX"拓展,选择带有"XeLaTeX"的编译,随后按 `ctrl+alt+v` 便可调用Sumatra查看编译后的结果.同时,双击Sumatra检查反向链接,右键VSCode中的文本,选择"从光标同步SyncTeX",或按快捷键 `ctrl+alt+j` 检查正向链接.

如果一切正常,恭喜你来到 $LaTeX$ 的世界!



如果你需要LaTeX的文档模板,参见:

- [AMS-LaTeX](#),提供了AMS格式的LaTeX模板.
- [模板-Overleaf](#),Overleaf是在线的LaTeX编辑网站,其中也提供了大量的LaTeX模板.第一次使用可能需要注册.
- 种种提供TeX源码的开源项目.

VSCode+Sumatra的操作

- 1.(打开外部查看器): 快捷键 `ctrl+alt+v` .
 - 2.(快速默认编译):快捷键 `ctrl+alt+b` .
 - 3.(正向链接): 右键选择"从光标同步SyncTeX",或快捷键 `ctrl+alt+j` .
 - 4.(反向链接): 在Sumatra内左键双击即可.
- 其余快捷键在VSCode内可自行查询与设置.

Docker Engine配置

直接复制即可,提供了许多可供使用的Docker镜像.其中绝大部分来自[docker配置镜像](#)[Docker pull时报错: https://registry-1.docker.io/v2/ - 知乎](#),依然感谢大佬的付出.

```
{
  "builder": {
    "gc": {
      "defaultKeepStorage": "20GB",
      "enabled": true
    }
  },
  "debug": true,
  "experimental": false,
  "insecure-registries": [
```

```
    "registry.docker-cn.com",
    "docker.mirrors.ustc.edu.cn"
],
"registry-mirrors": [
    "https://docker.registry.cyou",
    "https://docker-cf.registry.cyou",
    "https://dockercf.jsdelivr.fyi",
    "https://docker.jsdelivr.fyi",
    "https://dockertest.jsdelivr.fyi",
    "https://mirror.aliyuncs.com",
    "https://dockerproxy.com",
    "https://mirror.baidubce.com",
    "https://docker.m.daocloud.io",
    "https://docker.nju.edu.cn",
    "https://docker.mirrors.sjtug.sjtu.edu.cn",
    "https://docker.mirrors.ustc.edu.cn",
    "https://mirror.iscas.ac.cn",
    "https://docker.rainbond.cc",
    "https://do.nark.eu.org",
    "https://dc.j8.work",
    "https://dockerproxy.com",
    "https://gst6rzl9.mirror.aliyuncs.com",
    "https://registry.docker-cn.com",
    "http://hub-mirror.c.163.com",
    "http://mirrors.ustc.edu.cn/",
    "https://mirrors.tuna.tsinghua.edu.cn/",
    "http://mirrors.sohu.com/"
]
}
```