TAIJI LABORATORY
FOR GRAVITATIONAL WAVE UNIVERSE

ICTP-AP
International Centre
for Theoretical Physics Asia-Pacific
国际理论物理中心-亚太地区

University of Chinese Academy of Sciences

引力波数据探索：编程与分析实战训练营

第 1 部分 编程开发环境与工作流

主讲老师：王赫
ICTP-AP, UCAS
2023/11/12

Taiji Program in Space
Gravitational wave physics
空间太极计划

关于上一讲的学员反馈

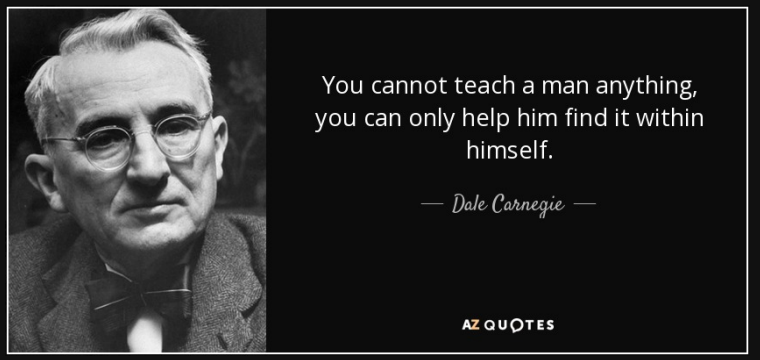
第8题：您有哪些建议或想法，能帮助我们改善线上的课程体验？ [填空题]

词频分析 | 观点分析 | 隐藏词云图 | 查看详细信息

第9题：请在此处写下任何额外的反馈或评论： [填空题]

词频分析 | 观点分析 | 隐藏词云图 | 查看详细信息

- 授课主要面向的是国科大以引力波数据处理为主要研究方向的研究生和高年级本科生。
- 不管“包学包会”，但会尽可能弥补和完善听课体验和授课内容。



目录

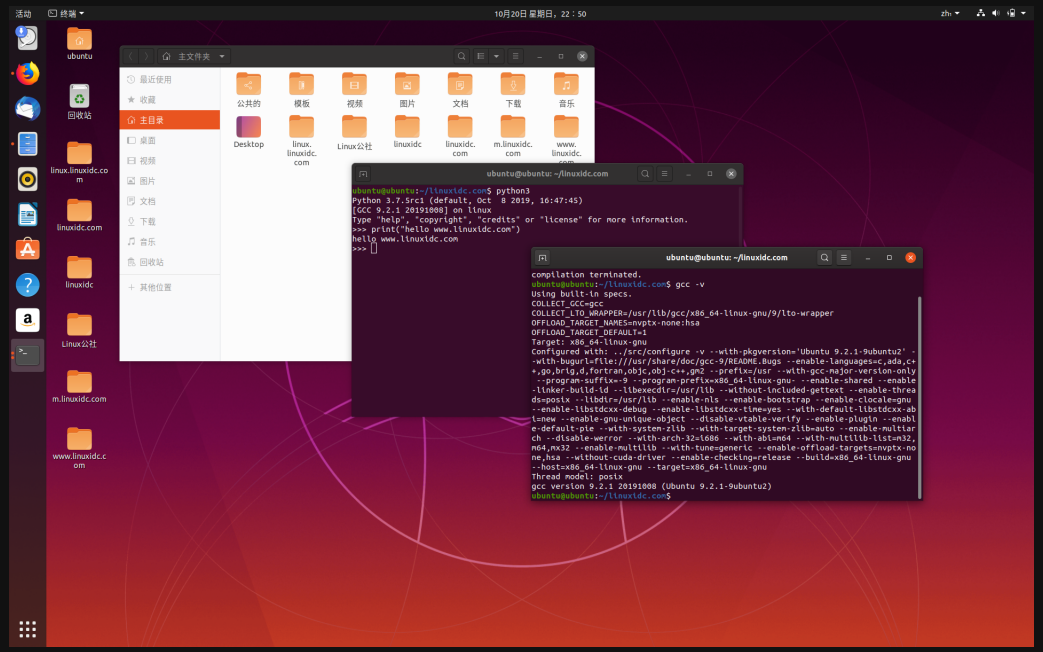
- 基础运维技术
  - 总说Linux，到底什么是Linux?
  - 强大好用的 Shell
  - 新手必须掌握的Linux命令
  - 管道符、重定向与环境变量
  - SSH服务管理远端设备
  - （Git 分布式版本控制系统）
- 基础容器化技术
  - Linux 容器虚拟化
  - Docker 的极简入门
  - Python / Jupyter 开发环境搭建
  - 远程连接 VS Code
  - （LALsuite / LISAcode 的源码编译）

基础运维技术

- 总说Linux，到底什么是Linux?
- 强大好用的 Shell
- 新手必须掌握的Linux命令
- 管道符、重定向与环境变量
- SSH服务管理远端设备
- （Git 分布式版本控制系统）

总说Linux，到底什么是Linux?

- 我们使用术语 “Linux” 来指代 Linux 内核，也是通常与Linux内核捆绑在一起的程序，工具和服务，以提供所有必需的组件全功能操作系统。
- Linux 与你以前使用的其他操作系统类似，如Windows，OS X或iOS，但是在许多重要方面也不同于其他操作系统，如 Linux 是开源软件，用于创建Linux的代码是免费的，可供公众查看，编辑和使用。



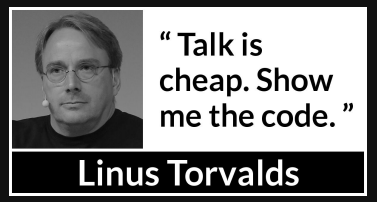
- See more: https://www.linuxprobe.com/whats-linux.html

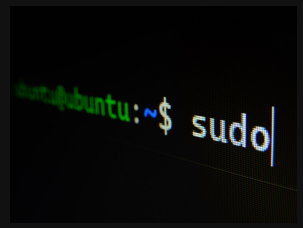
- Internet 上三分之二以上的网页都是由运行 Linux 的服务器生成。
- 公司和个人为他们的服务器选择 Linux，因为它是安全的，除了 Canonical，SUSE 和 Red Hat 等提供商业支持的公司之外，你还可以从大型用户社区获得极好的支持。
- 你可能拥有的许多设备，如Android手机，数字存储设备，个人录像机，相机，可穿戴设备等，也运行Linux。甚至你的车有Linux在运行下。

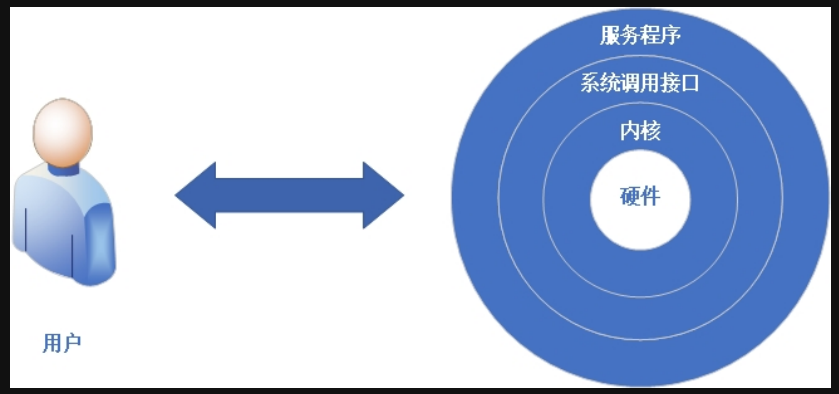
	Linux	Windows	macOS
Hardware Quality	Versatile, can run on low-spec PCs	Very versatile	Proprietary hardware, very high-end
Cost	Mostly free, some distros have paid versions	Freemium, accessing all features costs approximately $150	Free but comes on an expensive hardware
Software Compatibility	Open-source substitutes for proprietary software	Unparalleled	Has its own app ecosystem
Ease of Installation	Requires some computing knowledge	Easy	Very easy
Security and Stability	The safest and most stable OS	Generally great, requires plenty of frequent updates	Very good, requires only a few periodic updates
Ease of Use	Ease of use determined by the distro	Simple to use	Very easy to use

Understanding the Differences: Linux vs. Windows vs. Mac – A Comprehensive Guide

强大好用的 Shell







- Shell 就是终端程序的统称，它充当了人与内核（硬件）之间的翻译官，用户把一些命令“告诉”终端程序，它就会调用相应的程序服务去完成某些工作。
- 现在包括 Ubuntu 系统在内的许多主流Linux系统默认使用的终端是 Bash（Bourne-Again SHell）解释器，其是Shell终端程序中的一份子，它的语法和其他的 shell 都是类似的。

```
$man
# 空格键:          下翻一页
# PaGe down       向下翻一页
# PaGe up         向上翻一页
# home            直接前往首页
# end             直接前往尾页
# /      从上至下搜索某个关键词，如“/linux”
# ?      从下至上搜索某个关键词，如“?linux”
# n      定位到下一个搜索到的关键词
# N      定位到上一个搜索到的关键词
# q      退出帮助文档
```

- 执行命令的必备知识

```
Linux 命令的格式:
>> 命令名称   [命令参数]   [命令对象]
```

语法中的"动词"　　用于对命令的调整 “长格式” vs “短格式”　　命令执行后的“承受方”

Tips

- 约定俗成地将可选择的、可加或可不加的、非必需的参数使用中括号 [ ] 引起来。
- 命令名称、命令参数与命令对象之间要用空格进行分隔，且字母严格区分大小写。
- 用 man 命令查询指令的帮助手册
- **Tab键：**在Bash解释器的快捷键中，Tab键绝对是使用频率最高的，它能够实现对命令、参数或文件的内容补全。
- **Ctrl+c组合键：**当同时按下键盘上的Ctrl和字母c的时候，意味着终止当前进程的运行。
- **Ctrl+d组合键：**当同时按下键盘上的Ctrl和字母d的时候，表示键盘输入结束。
- **Ctrl+L组合键：**当同时按下键盘上的Ctrl和字母 L 的时候，会清空当前终端中已有的内容（相当于清屏操作）。

新手必须掌握的Linux命令

- 常用系统工作命令

```
$echo [字符串] [$变量]
$date [+指定的格式]
$reboot # 最好是以root管理员的身份来重启
$poweroff        # 同上
$shutdown now -r  # 立刻重启
$wget [参数] 网址
#       -b 后台下载模式
#       -P 下载到指定目录
#       -c 断点续传
#       -r 递归下载
$ps [参数]        # 查看系统中的进程状态
#       -a 显示所有进程（包括其他用户的进程）
#       -u 用户以及其他详细信息
#       -x 显示没有控制终端的进程
$ps -aux
$ps -ef # *所有进程*的*完整*信息
#       UID 进程的用户ID
#       PID 进程的唯一标识符，即进程ID
#       PPID：父进程的进程ID
#       C：进程的CPU使用率
#       STIME：进程的启动时间
#       TTY：进程所属的终端
#       TIME：进程的运行时间
#       CMD：进程的命令
```

$ps aux

https://www.linuxprobe.com/basic-learning-02.html

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
进程的所有者	进程ID号	运算器占用率	内存占用率	虚拟内存使用量(单位是KB)	占用的固定内存量(单位是KB)	所在终端	进程状态	被启动的时间	实际使用CPU的时间	命令名称与参数
root	1	0.0	0.5	244740	10636	?	Ss	07:54	0:02	/usr/lib/systemd/systemd --switched-root --system --deserialize 18
root	2	0.0	0.0	0	0	?	S	07:54	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	I<	07:54	0:00	[rcu_gp]
root	4	0.0	0.0	0	0	?	I<	07:54	0:00	[rcu_par_gp]
root	5	0.0	0.0	0	0	?	I<	07:54	0:00	[kworker/0:0H-kbl
root	6	0.0	0.0	0	0	?	I<	07:54	0:00	[mm_percpu_wq]
root	7	0.0	0.0	0	0	?	S	07:54	0:00	[ksoftirqd/0]
root	8	0.0	0.0	0	0	?	I	07:54	0:00	[rcu_sched]
root	9	0.0	0.0	0	0	?	S	07:54	0:00	[migration/0]

Tips

- 长格式和长格式之间不能合并，长格式和短格式之间也不能合并，但短格式和短格式之间是可以合并的，合并后仅保留一个减号（-）即可。
- 另外 ps 命令可允许参数不加减号（-），因此可直接写成 ps aux的样子。

新手必须掌握的Linux命令

• 常用系统工作命令

```
$top                          # 动态地监视进程活动及系统负载等信息
$kill [参数] 进程的PID        # 终止某个指定PID值的服务进程
$killall [参数] 服务名称      # 终止某个指定名称的服务所对应的全部进程
```

• 系统状态检测命令

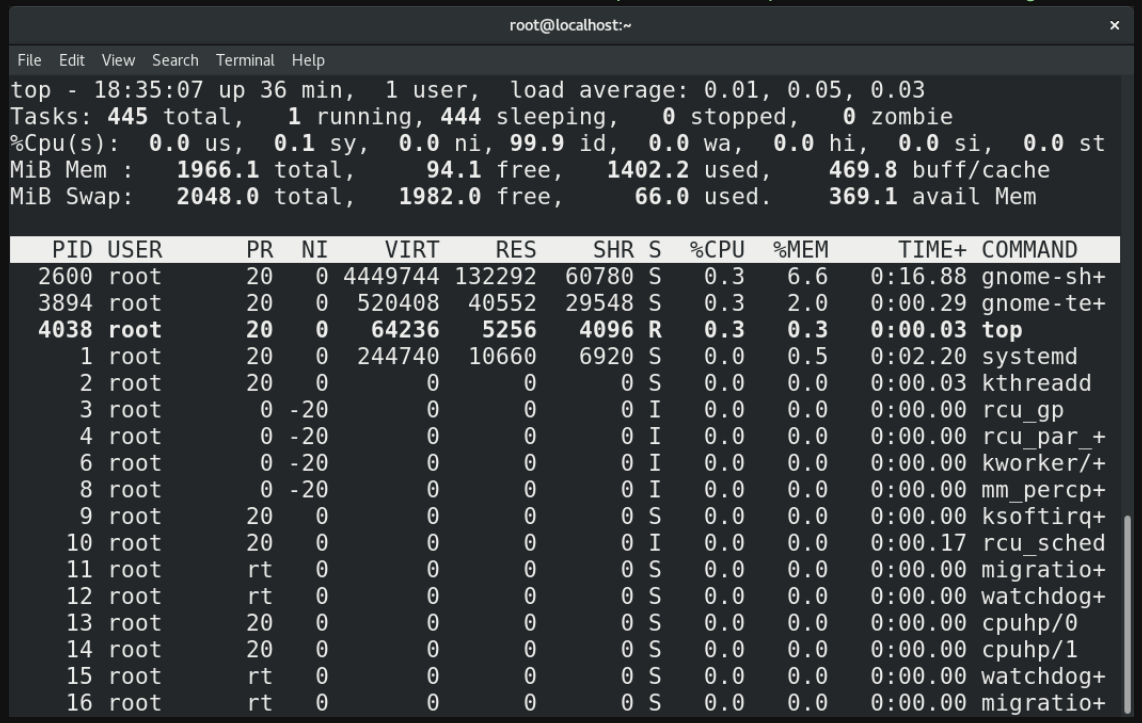
```
$ifconfig [参数] [网络设备]          # 获取网卡配置与网络状态等信息
$ping [参数] 主机地址                # 测试主机之间的网络连通性
$uname -a
#         内核名称、主机名、内核发行版本、节点名、压制时间、
#         硬件名称、硬件平台、处理器类型以及操作系统名称
$uptime # 查看系统的负载信息
$free -h          # 当前系统中内存的使用量信息
```

	内存总量	已用量	空闲量	共享使用的内存量	磁盘缓存的内存量	缓存的内存量	可用量
	total	used	free	shared	buffers	buff/cache	available
Mem:	1.9Gi	1.4Gi	99Mi	20Mi	450Mi	348Mi	
Swap:	2.0Gi	80Mi	1.9Gi				

Tips

- 如果有些命令在执行时不断地在屏幕上输出信息，影响到后续命令的输入，则可以在执行命令时在末尾添加一个&符号，这样命令将进入系统后台执行。
- 建议负载值保持在1左右，在生产环境中不要超过5就好。

https://www.linuxprobe.com/basic-learning-02.html



第1行：系统时间、运行时间、登录终端数、系统负载（3个数值分别为1分钟、5分钟、15分钟内的平均值，数值越小意味着负载越低）。

第2行：进程总数、运行中的进程数、睡眠中的进程数、停止的进程数、僵死的进程数。

第3行：用户占用资源百分比、系统内核占用资源百分比、改变过优先级的进程资源百分比、空闲的资源百分比等。其中数据均为CPU数据并以百分比格式显示，例如“99.9 id”意味着有99.9%的CPU处理器资源处于空闲。

第4行：物理内存总量、内存空闲量、内存使用量、作为内核缓存的内存量。

第5行：虚拟内存总量、虚拟内存空闲量、虚拟内存使用量、已被提前加载的内存量。

新手必须掌握的Linux命令

• 系统状态检测命令

```
$history [-c]    # 显示执行过的命令历史
# 历史命令会被保存到用户家目录中的.bash_history文件中
$cat ~/.bash_history
$htop            # 交互式进程查看器
$watch
```

Tips

- Linux系统中以点（.）开头的文件均代表隐藏文件，这些文件大多数为系统服务文件。
- **Ctrl+r快捷键**：在Linux中使用 Ctrl+r 可以进入反向搜索模式，在该模式下，输入关键字会搜索之前输入的命令历史记录，并显示最近一次匹配该关键字的命令。若想查找上一条匹配，可以重复按下 Ctrl+r, 直到找到所需的命令为止。另外，也可以按下Enter键执行当前匹配的命令

• 查找定位文件命令

```
$pwd     # 显示用户当前所处的工作目录
$cd [参数] [目录]
$ls [参数] [文件名称]
$ls -lth
$chmod   # 设置文件的一般权限及特殊权限
```

读写执行权限对于文件与目录可执行命令的区别

	文件	目录
读取（r）	cat	ls
写入（w）	vim	touch
执行(x)	./script	cd

文件权限的字符与数字表示

权限项	读	写	执行	读	写	执行	读	写	执行
字符表示	r	w	x	r	w	x	r	w	x
数字表示	4	2	1	4	2	1	4	2	1
权限分配	文件所属者			文件所属组			其他用户		

• 用户身份与文件权限

```
$id [用户名]      # 显示用户的详细信息
$useradd [参数] 用户名
$passwd [参数] 用户名
$userdel [参数] 用户名
$su [用户名]      # 切换用户身份
$sudo [参数] 用户名          # 把特定命令的执行权限赋予指定用户
$visudo [参数]    # 编辑、配置用户sudo的权限文件
```

Tips

- 授权原则：在保证普通用户完成相应工作的前提下，尽可能少地赋予额外的权限。
- visudo 格式：谁可以使用 允许使用的主机 = （以谁的身份） 可执行命令的列表

新手必须掌握的Linux命令

- 查找定位文件命令

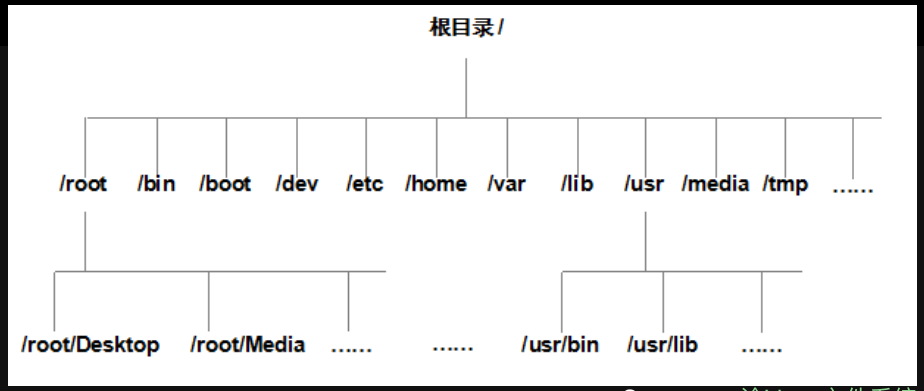
```
$find [查找范围] 寻找条件  # 按照指定条件来查找文件所对应的位置
#        -name    匹配名称
#        -perm    匹配权限（mode为完全匹配，-mode为包含即可）
#        -user    匹配所有者
#        -group   匹配所有组
#        -mtime -n +n     匹配修改内容的时间（-n指n天以内，+n指n天以前）
#        -atime -n +n     匹配访问文件的时间（-n指n天以内，+n指n天以前）
#        -ctime -n +n     匹配修改文件权限的时间（-n指n天以内，+n指n天以前）
#        -nouser 匹配无所有者的文件
#        -nogroup         匹配无所有组的文件
#        -newer f1 !f2    匹配比文件f1新但比f2旧的文件
#        -type b/d/c/p/l/f        匹配文件类型（后面的字幕字母依次表示块设备、目录、字符设备、管道、链接文件、文本文件）
#        -size    匹配文件的大小（+50KB为查找超过50KB的文件，而-50KB为查找小于50KB的文件）
#        -prune   忽略某个目录
#        -exec …… {}\;    后面可跟用于进一步处理搜索结果的命令

# 获取保存系统配置文件的目录/etc中所有以host开头的文件列表
$find /etc -name "host*" -print
# 在整个文件系统中找出所有归属于 ubuntu 用户的文件并复制到 /root/findresults 目录中
$find / -user ubuntu -exec cp -a {} /root/findresults/ \;
# 重点是"-exec {} \;"参数，其中的{}表示find命令搜索出的每一个文件，并且命令的结尾必须是"\;"。
```

```
$whereis 命令名称          # 按照名称快速搜索二进制程序（命令）、源代码以及帮助文件所对应的位置
$which 命令名称    # 按照指定名称快速搜索二进制程序（命令）所对应的位置
```

Tips

- Linux系统中的一切都是文件。在Linux系统中，目录、字符设备、硬盘、光驱、打印机等都被抽象成文件形式。
- 一切从“/”开始。

Linux系统中的文件存储结构



See more: 论Linux文件系统

```
$df -h                  # 查看磁盘占用的空间
$du -sh 目录/文件       # 查看目录/文件的真实大小
```

新手必须掌握的Linux命令

- 文本文件编辑命令

```
$cat [参数] 文件名称        # 查看纯文本文件（内容较少的）
$cat -n 文件名称  # 顺便显示行号
$more [参数] 文件名称       # 查看纯文本文件（内容较多的）
$head [参数] 文件名称       # 查看纯文本文件的前 N 行
$tail [参数] 文件名称       # 查看纯文本文件的后 N 行
$tail -f 文件名称           # 能够持续刷新一个文件的内容
$wc [参数] 文件名称         # 统计指定文本文件的行数、字数或字节数
#        -l       只显示行数
#        -w       只显示单词数
#        -c       只显示字节数
$wc -l /etc/passwd        # 统计当前系统中有多少个用户
$stat 文件名称    # 查看文件的具体存储细节和时间等信息
#        Access Time（内容最后一次被访问的时间，简称为Atime）
#        Modify Time（内容最后一次被修改的时间，简称为Mtime）
#        Change Time（文件属性最后一次被修改的时间，简称为Ctime）
$grep [参数] 文件名称       # 按"行"提取文本内容
$grep -n 文件名称           # 顺便显示行号
$grep -v 文件名称           # 反选信息
$grep /sbin/nologin /etc/passwd # 查找出当前系统中不允许登录系统的
所有用户的信息
```

- 文件目录管理命令

```
$touch [参数] 文件名称     # 创建空白文件或设置文件的时间
$mkdir [参数] 目录名称     # 创建空白的目录
$mkdir -p 嵌套层叠关系的目录
$cp [参数] 源文件名称 目标文件名称
#         如果目标文件是目录，则会把源文件复制到该目录中；
#         如果目标文件也是普通文件，则会询问是否要覆盖它；
#         如果目标文件不存在，则执行正常的复制操作。
#         -p        保留原始文件的属性
#         -d        若对象为“链接文件”，则保留该“链接文件”的属性
#         -r        递归持续复制（用于目录）
#         -i        若目标文件存在则询问是否覆盖
#         -a        相当于-pdr（p、d、r为上述参数）
$rm [参数] 文件名称
#         -f        强制执行
#         -i        删除前询问
#         -r        删除目录
#         -v        显示过程
```

Tips

- 写好并准备执行终端Linux命令之前，默读指令检查一遍！

管道符、重定向与环境变量

输入输出重定向

- 一般键盘作为输入“流”，显示器作为输出“流”，也可以重定向这些流！
- 标准输入重定向（STDIN，文件描述符为0）：默认从键盘输入，也可从其他文件或命令中输入。
- 标准输出重定向（STDOUT，文件描述符为1）：默认输出到屏幕。
- 错误输出重定向（STDERR，文件描述符为2）：默认输出到屏幕。

输入重定向中用到的符号及其作用

符号	作用
命令 < 文件	将文件作为命令的标准输入
命令 << 分界符	从标准输入中读入，直到遇见分界符才停止
命令 < 文件1 > 文件2	将文件1作为命令的标准输入并将标准输出到文件2

输出重定向中用到的符号及其作用

符号	作用
命令 > 文件	将标准输出重定向到一个文件中（清空原有文件的数据）
命令 2> 文件	将错误输出重定向到一个文件中（清空原有文件的数据）
命令 >> 文件	将标准输出重定向到一个文件中（追加到原有内容的后面）
命令 2>> 文件	将错误输出重定向到一个文件中（追加到原有内容的后面）
命令 >> 文件 2>&1 或 命令 &>> 文件	将标准输出与错误输出共同写入到文件中（追加到原有内容的后面）

管道命令符 "|"

- 把前一个命令原本要输出到屏幕的信息当作后一个命令的标准输入。

See more: 通配符和转义字符

重要的环境变量

命令在Linux中的执行分为4个步骤：

1. 判断用户是否以绝对路径或相对路径的方式输入命令（如/bin/ls）
2. Linux系统检查用户输入的命令是否为“别名命令”
3. Bash解释器判断用户输入的是内部命令还是外部命令
4. 系统在多个路径中查找用户输入的命令文件，而定义这些路径的变量叫作 PATH，Bash解释器在这些位置中逐个查找

变量名称	作用
HOME	用户的主目录（即家目录）
SHELL	用户在使用的Shell解释器名称
HISTSIZE	输出的历史命令记录条数
HISTFILESIZE	保存的历史命令记录条数
MAIL	邮件保存路径
LANG	系统语言、语系名称
RANDOM	生成一个随机数字
PS1	Bash解释器的提示符
PATH	定义解释器搜索用户执行命令的路径
EDITOR	用户默认的文本编辑器

Linux系统中最重要的10个环境变量

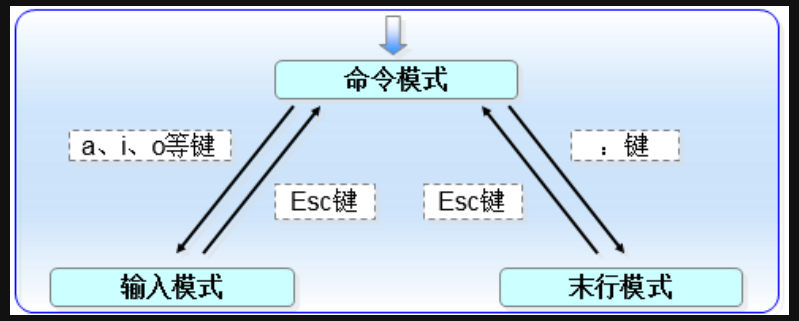
Tips

- 将变量和变量值写入到 .bashrc 或者 .bash_profile 文件中，以确保永久能使用它们。

Vim编辑器与Shell命令脚本

- Vim

- “在Linux系统中一切都是文件，而配置一个服务就是在修改其配置文件的参数。”



命令模式中最常用的一些命令

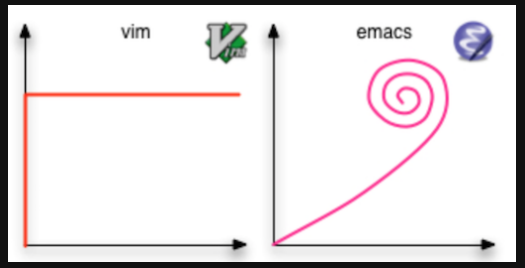
命令	作用
dd	删除（剪切）光标所在整行
5dd	删除（剪切）从光标处开始的5行
yy	复制光标所在整行
5yy	复制从光标处开始的5行
n	显示搜索命令定位到的下一个字符串
N	显示搜索命令定位到的上一个字符串
u	撤销上一步的操作
p	将之前删除（dd）或复制（yy）过的数据粘贴到光标后面

末行模式中最常用的一些命令

命令	作用
:w	保存
:q	退出
:q!	强制退出（放弃对文档的修改内容）
:wq!	强制保存退出
:set nu	显示行号
:set nonu	不显示行号
:命令	执行该命令
:整数	跳转到该行
:s/one/two	将当前光标所在行的第一个one替换成two
:s/one/two/g	将当前光标所在行的所有one替换成two
:%s/one/two/g	将全文中的所有one替换成two
?字符串	在文本中从下至上搜索该字符串
/字符串	在文本中从上至下搜索该字符串

See more:

- Lecture 3: Editors (vim) (2020)
- 《用聪明的方法学习 Vim》： https://gitlab.com/wsdjeg/Learn-Vim_zh_cn



- 编写和执行简单的Shell脚本

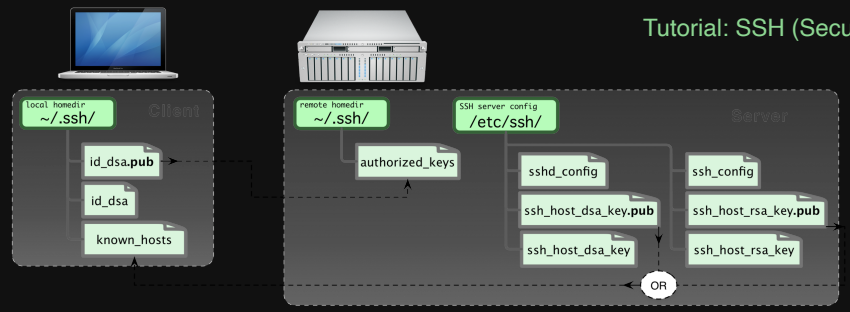
Tips

- && 是逻辑“与”，只有当前面的语句执行成功的时候才会执行后面的语句。
- || 是逻辑“或”，只有当前面的语句执行失败的时候才会执行后面的语句。
- ! 是逻辑“非”，代表对逻辑测试结果取反值；之前若为正确则变成错误，若为错误则变成正确。
- & 表示程序要在后台执行
- ; 表示每个命令按照从左到右的顺序来执行，每个命令彼此之间无任何关联，所有命令都要执行

SSH 服务管理远端设备

- SSH（Secure Shell）是一种能够以安全的方式提供远程登录的协议，也是目前远程管理Linux系统的首选方式。

Tutorial: SSH (Secure Shell)




```
$ssh [参数] 远程账户@[远程IP地址]
#       -p      指定远程主机的sshd端口号
```

- scp: 通过 SSH 复制文件，不仅能够通过网络传送数据，而且所有的数据都将进行加密处理。

```
$scp [参数] 本地文件 远程账户@远程IP地址:远程目录
#       -v      显示详细的连接进度
#       -P      指定远程主机的sshd端口号
#       -r      用于传送文件夹
#       -6      使用IPv6协议
```

- rsync: 对 scp 进行了改进，它可以检测本地和远端的文件以防止重复拷贝。它还可以提供一些诸如符号连接、权限管理等精心打磨的功能。甚至还可以基于 --partial 标记实现断点续传。

```
$rsync -avzP -e 'ssh -p 端口号' 本地文件 远程账户@远程IP地址:远程目录
```

- 不间断会话服务：Screen

See more: Linux/Unix 中 Screen 命令详解

Screen 是一款由 GNU 计划开发的用于命令行终端切换的自由软件。用户可以通过该软件同时连接多个本地或远程的命令行会话，并在其间自由切换。GNU Screen 可以看作是窗口管理器的命令行界面版本。它提供了统一的管理多个会话的界面和相应的功能。

- 会话恢复
- 多窗口

```
$screen -S yourname     #-> 新建一个叫 yourname 的 session
$screen -ls             #-> 列出当前所有的 session
$screen -r yourname     #-> 回到 yourname 这个session
$screen -x yourname     #-> 回到 yourname 这个已经 attached session
$screen -d yourname     #-> 远程 detach 某个session
$screen -d -r yourname  #-> 结束当前 session 并回到 yourname 这个 session
```

容器化技术

- Linux 容器虚拟化
- Docker 的极简入门
- Python / Jupyter 开发环境搭建
- 远程连接 VS Code
- （LALsuite / LISAcode 的源码编译）

Linux 容器虚拟化

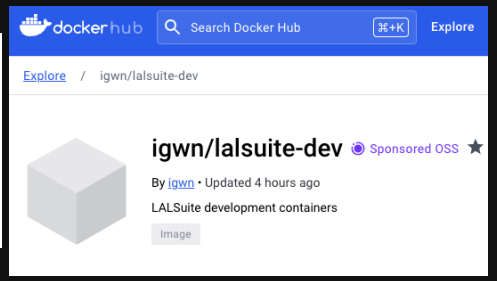
- 环境配置的难题
- 虚拟机（virtual machine）
  - 资源占用多、冗余步骤多、启动慢
- Linux 容器（Linux Containers, LXC）：不是模拟一个完整的操作系统，而是对进程进行隔离。
  - 启动快、资源占用少、体积小

为什么要使用 Docker

2013年发布至今， Docker 一直广受瞩目，被认为可能会改变软件行业。

- 更高效的利用系统资源
- 更快速的启动时间
- 一致的运行环境
- 持续交付和部署（DevOps）
- 更轻松的迁移
- 更轻松的维护和拓展

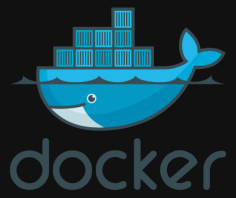
详细的 Docker 介绍可见：
https://docker_practice.gitee.io/introduction/what.html



Also see: http://https://computing.docs.ligo.org/guide/dhtc/containers/

容器化 vs 传统虚拟机

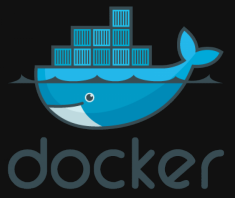
特性	容器	虚拟机
启动	秒级	分钟级
硬盘使用	一般为 MB	一般为 GB
性能	接近原生	弱于
系统支持量	单机支持上千个容器	一般几十个



Docker 的用途

1. **提供一次性的环境。** 比如，本地测试他人的软件、持续集成的时候提供单元测试和构建的环境。
2. **提供弹性的云服务。** 因为 Docker 容器可以随开随关，很适合动态扩容和缩容。
3. 组建微服务架构。 通过多个容器，一台机器可以跑多个服务，因此在本机就可以模拟出微服务架构。

Docker 的极简入门



• Docker 的安装

- Docker 是一个开源的商业产品，有两个版本：社区版（Community Edition，缩写为 CE）和企业版（Enterprise Edition，缩写为 EE）。
- Docker CE 的安装请参考官方文档，也可参考这本中文线上电子书教程。

```
# 安装完成后，运行下面的命令，验证是否安装成功。
$docker version
$docker info

# 如果 docker 服务没有启动，可以用下面的命令启动
# 以下命令仅适用于 Linux 系统 service 命令的用法
$sudo service docker start

# systemctl 命令的用法 (RHEL7/Centos7)
$sudo systemctl start docker
```

Tips

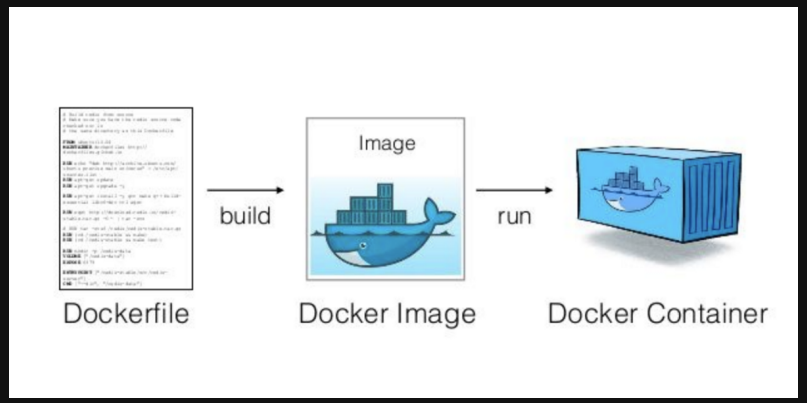
- 镜像加速器：修改 Docker 的官方仓库到国内的镜像网站 https://docker_practice.gitee.io/install/mirror.html
- 《Docker — 从入门到实践》https://docker_practice.gitee.io

• Docker 的 Hello World 示例

```
# 抓取官方的 hello-world 镜像:
$sudo docker image pull hello-world

# 查看
$sudo docker images

# 运行这个 image 文件。
$sudo docker container run hello-world

# Hello from Docker!
# This message shows that your installation appears to

# ... ...
# (运行成功! )
```

```
$sudo docker images      # 查看本机所有的镜像
$sudo docker ps -a       # 查看本机所有的容器

$sudo docker rm [containerID]    # 删除容器
$sudo docker image rm [imageID] # 删除镜像
```

Docker 的极简入门

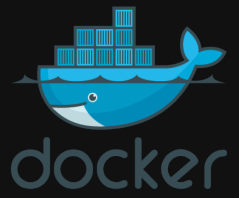



```bash
#!/bin/bash
var="Hello World"

# Run date and hostname command and s
# tore output to shell variables
now="$(date)"
computer_name="$(hostname)"

# print it or use the variable
# Variable names are case sensitive $now
# and $NOW are different names
#
echo "$var"
echo "Current date and time : $now"
echo "Computer name : $computer_name"
echo ""
```

- 一个简单的 APP 示例

```dockerfile
# Base Image
FROM python:3.10

# 将当前目录下的所有文件(除了.dockerignore排除的路径)
# 都拷贝进入 image 里的/home/ictp_ap目录
COPY . /home/ictp_ap

# 将容器 8000 端口暴露出来， 允许外部连接这个端口
EXPOSE 8000

# 指定接下来的工作路径为/home/ictp_ap
WORKDIR /home/ictp_ap

RUN apt-get update && apt-get install \
        -y --no-install-recommends \
        python3-setuptools \
        python3-pip \
        python3-dev \
        python3-venv \
        git \
        && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

RUN python -c "print('hello world')"

# 将这个 image 做成一个 app 可执行程序
# 容器启动后自动执行下面指令
# (ENTRYPOINT 可加额外的shell参数)
ENTRYPOINT ["bash", "setup.sh"]
```

Python / Jupyter 开发环境搭建



- A unified Dockerfile

```
FROM ubuntu:20.04
LABEL MAINTAINER="He Wang<hewang@ucas.ac.cn>"
ARG RELEASE_NAME=focal
ARG ROOT_PASSWD=root
ENV SSH_PORT 22
ENV PATH /root/miniconda3/bin:$PATH
# Open port
EXPOSE 8888
EXPOSE 22
#
# UTF-8 encoding to support chs characters
#
RUN echo "export LANG=C.UTF-8" >>/etc/profile \
#
# apt sources backup
#
&& cp /etc/apt/sources.list /etc/apt/sources.list.bak \
#
# switch apt source to TUNA
#
&& echo "deb http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ release-name main restricte
deb http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ release-name-updates main restricted
deb http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ release-name-backports main restrict
deb http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ release-name-security main restricte
&& sed -i 's/release-name/'$RELEASE_NAME'/g' /etc/apt/sources.list \
#
# remove nvidia sources
#
# && rm /etc/apt/sources.list.d/cuda.list /etc/apt/sources.list.d/nvidia-ml.list \
#
# install necessary packages
#
&& apt-get update \
&& apt-get install -y --no-install-recommends \
  build-essential \
  apt-utils \
  vim \
  openssh-server \
  net-tools \
```

```
#!/bin/bash
sudo docker build -t ictp_ap_jupyter .
```

```
#!/bin/bash
sudo docker run -itd \
        -p 1234:22 \
        -p 19999:8888 \
        --name ictp_ap_dev \
        ictp_ap_jupyter
```

```
ssh -p 1234 root@0.0.0.0
```

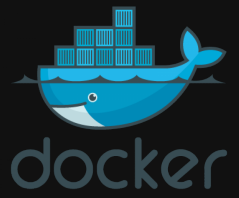
```
$sudo docker stop [containerID] # 终止容器运行

# 标注用户名和版本
$sudo docker tag [imageName] [username]/[respository]:[tag]

# 发布image文件
$sudo docker image push [username]/[respository]:[tag]
```

```
ubuntu@ubuntu:~/ICTP_AP_course$ sudo docker image push iphysresearch/ictp_ap_jupyter:1.0
The push refers to repository [docker.io/iphysresearch/ictp_ap_jupyter]
09e4bd03abb6: Pushed
6c3e7df31590: Layer already exists
1.0: digest: sha256:f65b5957d8ba7d45ee9a57c0fc88483a1432c5d081ebb4104d7b56a5ac4e0003 size: 742
```

查看一下这个image发布后的效果：

https://hub.docker.com/repository/docker/iphysresearch/ictp_ap_jupyter/general

Python / Jupyter 开发环境搭建



- Inside the Container

```
# 创建 conda 环境
conda create -n ictp-ap python=3.10 --yes
# 激活 conda 环境
conda activate ictp-ap
# 安装 JupyterNotebook 内核
pip install --upgrade ipykernel
# 安装一个扩展插件
pip install jupyterlab_nvdashboard

# 给 ictp-ap conda环境创建同名 kernel
python -s -m ipykernel install \
    --user \
    --name=ictp-ap \
    --display-name="ictp-ap"
```

```
sudo docker pull iphysresearch/ictp_ap_jupyter:1.5
```

```
sudo docker run -itd --platform linux/amd64 \
        -p 1234:22 \
        -p 19999:8888 \
        -v /Users/herb/ICTP_AP_course:/home/ICTP_AP_course \
        --name ictp_ap_dev \
        iphysresearch/ictp_ap_jupyter:1.0
```

```
ssh -p 1234 root@0.0.0.0
```

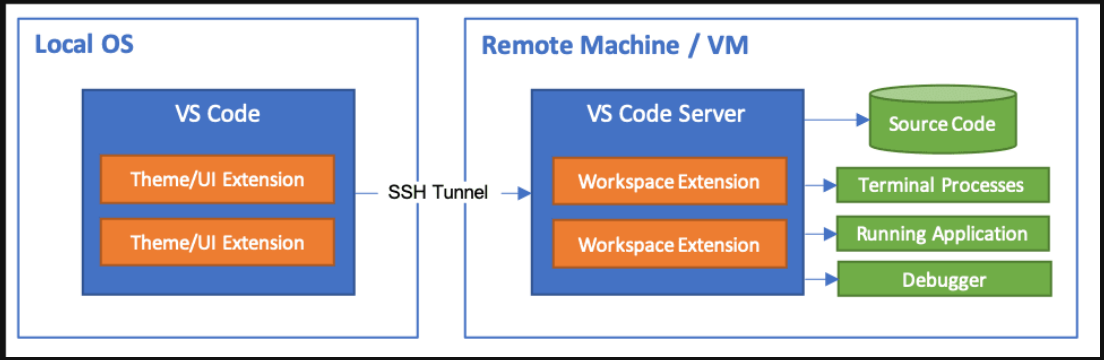
```
#!/bin/bash
nohup jupyter-lab \
    --ip='*' \
    --port=8888 \
    --no-browser \
    --NotebookApp.token='' \
    --allow-root \
    --autoreload \
    --notebook-dir=/home \
    > jupyter.log 2>&1 &
```

http://127.0.0.1:19999

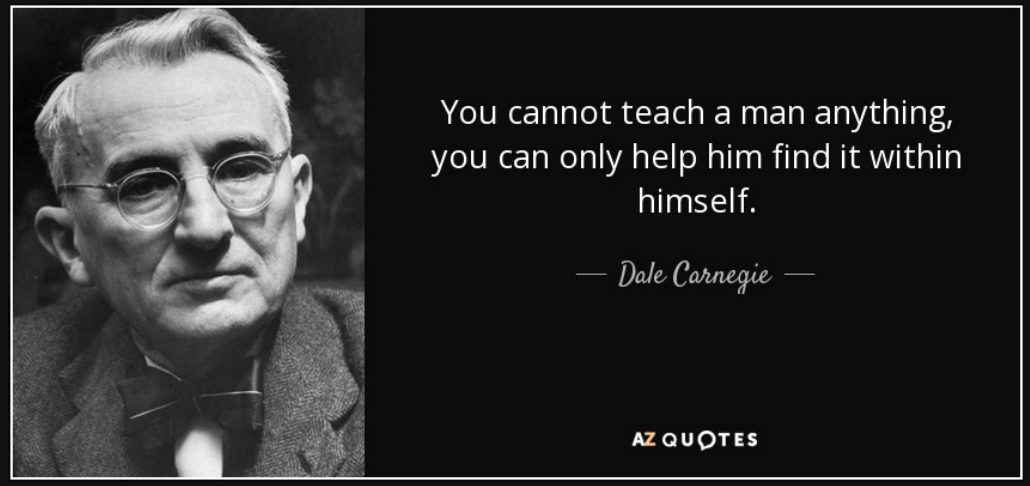
远程连接 VS Code

- Remote Development using SSH

- Visual Studio Code Remote - SSH 扩展允许你在任何地方运行 SSH 服务器的远程机器、虚拟机或容器上打开远程文件夹，并充分利用 VS Code 的全部功能。一旦连接到服务器，你就可以与远程文件系统上的任何文件和文件夹进行交互。




```
# ~/.ssh/config
Host ICTP_AP_course
  HostName 0.0.0.0
  User root
  Port 1234
```



Repo of the course: https://github.com/iphysresearch/GWData-Bootcamp

Homework

基础作业：

1. 在自己的本地笔记本电脑/远程服务器上，用 Docker 容器创建 Python / Jupyter 开发环境。
2. 自行安装好 VS Code 后，用 SSH remote 扩展远程连接自己创建好的 Docker 容器。

扩展作业：

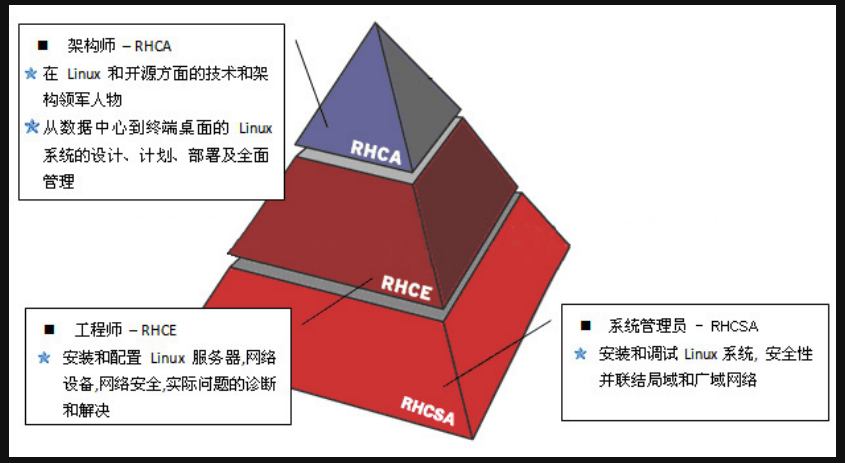
1. 在自己本地/远程服务器上，用 Docker 容器创建支持 GPUs 的 Python / Jupyter 开发环境（需要有GPU卡）
2. 创建 LALsuite / LISAcode 的源码编译好的容器镜像

Hints:

- https://hub.docker.com/r/nvidia/cuda
- https://github.com/NVIDIA/nvidia-container-toolkit
- https://git.ligo.org/lscsoft/lalsuite
- https://hub.docker.com/r/igwn/lalsuite-dev
- https://hub.docker.com/repository/docker/iphysresearch/lal_bilby
- https://lisa-ldc.lal.in2p3.fr/

通向自我实现之路

Linux就该这么学



- Linux 系统运维工程师必备的红帽认证（RHCSA, RHCE, RHCA）
- 《The Missing Semester of Your CS Education》（计算机教育中缺失的一课）

- Docker 入门教程 - 阮一峰 （我的最初级入门资料）
- Docker — 从入门到实践 or 在线阅读国内镜像（非常全的中文教程电子书Irepo）
- 10张图带你深入理解Docker容器和镜像（英文原文：Visualizing Docker Containers and Images）
- Docker系列之一：入门介绍（美团技术资料）
- Docker系列之二：基于容器的自动构建（美团技术资料）
- 如何使用docker部署c/c++程序
- *Docker Get Started Tutorial*
- How to use Docker Build Args and Environment Variables
- Docker进阶之Dockerfile

- 主办单位
  - 中国科学院大学 · 国际理论物理中心（亚太地区）
  - 引力波宇宙太极实验室

United Nations Educational, Scientific and Cultural Organization

ICTP-AP INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS ASIA-PACIFIC

TAIJI LABORATORY FOR GRAVITATIONAL WAVE UNIVERSE

- 赞助单位
  - 中科曙光

计算服务 | Sugon 曙光智算