

# Get\_MR2.0

---

## 写在前面

欢迎来到向量化与并行化的世界！本次更新就一个重大功能，就是并行化运行mr分析，以最优的效率批量跑大量的数据。家用电脑较新服务器基本可以实现2小时批量运行10000个因素，如果你有高性能服务器，恭喜你，30分钟以内就能跑完。

由于本次主要是思路与方法的分享，所以函数的帮助文档写的不多，主要还是看示例代码即可，应该还是很容易上手的。

公众号回复：“示例”即得示例代码

## cyclemr

---

### 描述

这个函数是用于执行循环Mendelian Randomization (CycleMR) 分析的功能函数，可以在R语言中使用。该函数可以将数据分配到多个计算节点中运行，提高MR分析的效率。

### 用法

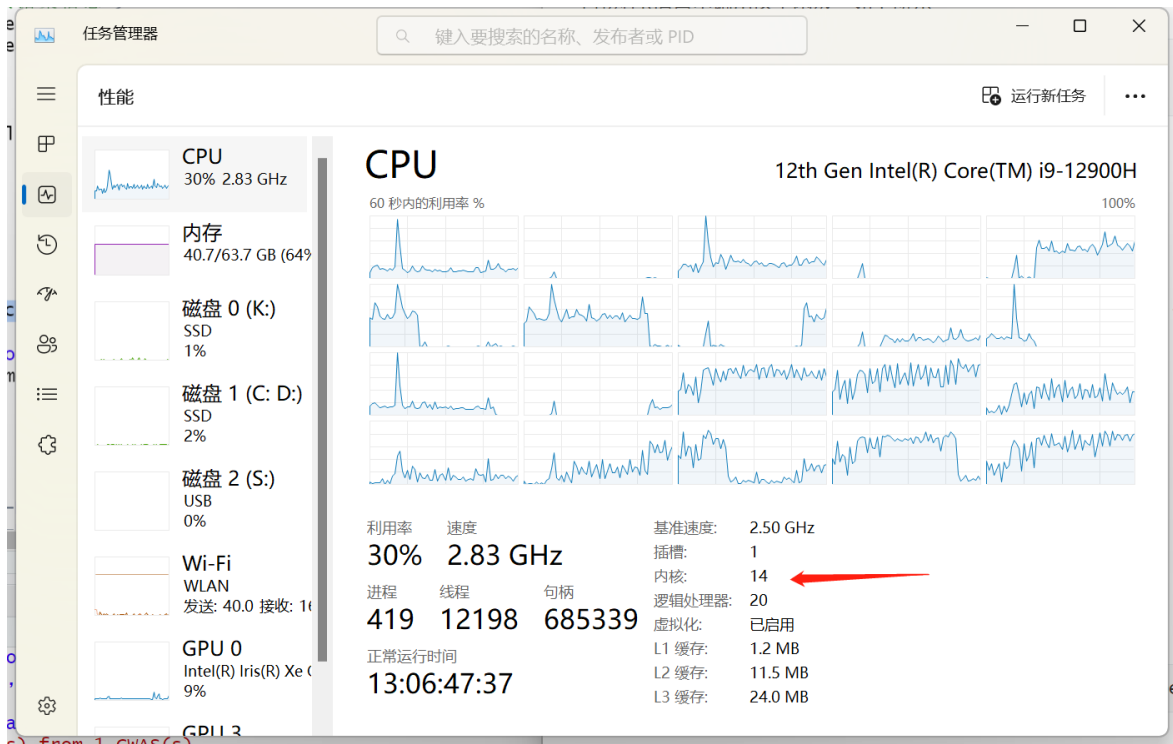
直接在R语言中调用这个函数，如下所示：

```
1 # 调用cyclemr函数
2 mr_results <- cyclemr(dat = data, cl_num = 4, type = "list")
```

### 参数

- `dat`: harmonise\_data后的数据，可以是数据框或列表类型。默认是list
- `cl_num`: 批量化线程数。
- `type`: MR分析数据类型，可以是"list"或"data"，默认为"list"。主要使用情况也是list

注： 如何判断自己的电脑能开启的最大线程数？



在任务管理器可看到

- 内核：计算机核心数
- 逻辑处理器：计算机总线线程数

比如说很多处理器宣传是8核16线程，这个8就指的是内核，这个16指的是逻辑处理器。

本质上，16只是将每个核心一分为2，但是他们能干的活是一样的。所以一般设置为内核数即可满载CPU

当然这不能一概而论，因为每个CPU和厂家调度不一样，如果你发现使用内核数不能让CPU跑到100%，则尝试用逻辑处理器数

## 返回值

- `cyclemr` 函数返回一个包含MR分析结果的数据框。

## 使用举例

下面是使用这个函数的一个示例：

```
1 | mr_results <- cyclemr(dat = data, cl_num = 16, type = "list")
```

## 运行时间参考

在设置无误情况下，这是我手头有的所有电脑测试出的运行时间：

运行10000个数据。（ieu批量数据前10000个）除了服务器外，其他均使用Windows系统

CPU	核数（运行时开的线程数）	时间
-----	--------------	----

CPU	核数（运行时开的线程数）	时间
i9-12900H（拯救者2022 Y9000P 狂暴模式）	14核20线程（14）	1小时28分
r7-5700X（台式）	8核16线程（16）	1小时38分
r9-6800H (yoga2022 14S 性能模式)	8核16线程（16）	1小时54分
r5-3500X (台式)	6核12线程（12）	3小时47分
r5-4600H（拯救者 2020 R7000 狂暴模式）	6核12线程（12）	约4小时
双路 EPYC 7T83（服务器 Linux）	128核256线程（128）	11分钟

欢迎各位补充自己手头的机器的运行时间数据，尤其M系列的苹果处理器数据

## 一些小工具

### get\_rsid

#### 描述

根据CHR和POS，从ensemble官网中获取rsID。

#### 用法

```
1 | get_rsid(chr, pos, version = 'hg38')
```

#### 参数

- chr：染色体号。
- pos：基因位置。
- version：表示使用的基因组版本，默认为最新的版本（'hg38'）。也可选择hg19。
- 注：GRCh37=hg19，GRCh38=hg38

#### 详细说明

该函数基于生物信息学数据库Ensembl SNP Mart来查询给定位点的相关信息。如果未指定基因组版本，则默认使用最新的版本(hg38)。该函数会根据指定的基因组版本选择正确的URL。如果您想查询其他版本的数据，可以将version参数设置为相应版本的字符串。

参考：[How to find rsID with biomaRt in R \(bioconductor.org\)](#)

## 使用举例

```
1 ds4 <- data.frame(CHR = c("8", "8", "8", "8", "8"), POS = c('101592213',  
2   '106973048', '108690829', '102569817', '108580746'))  
2 res<-get_rsid(chr=ds4$CHR, pos=ds4$POS, version = 'hg38')
```

## 错误说明

如果出现：

```
1 Error in curl::curl_fetch_memory(url, handle = handle) :  
2   Timeout was reached: [grch37.ensembl.org:80] Operation timed out after 300013  
   milliseconds with 7909 bytes received
```

这并不是代码问题，而是网络超时了，ensembl的API经常拥堵，多试几次即可。当然也有可能请求的数据量太大，也可能出现这个问题。

## get\_exposure 和 get\_outcome

### 描述

这两个函数是用于进行双样本MR（Two-sample Mendelian Randomization）分析的数据处理和提取过程的功能函数。

- `get_exposure` 函数用于从给定的遗传仪器ID中提取出暴露（exposure）数据。
- `get_outcome` 函数用于从给定的遗传仪器ID和暴露数据中提取出结果（outcome）数据。

### 用法

使用这两个函数前，需要先安装并加载TwoSampleMR包。

```
1 library(TwoSampleMR)
```

然后可以直接在R语言中调用这两个函数，如下所示：

```
1 # 调用get_exposure函数  
2 exposure_data <- get_exposure(id = "ieu-a-1", pval = 5e-8, r2 = 0.001, kb =  
   10000)  
3  
4 # 调用get_outcome函数  
5 outcome_data <- get_outcome(id = "ieu-a-1", expo = exposure_data)
```

## 参数

- `get_exposure` 函数的参数说明：
  - `id`: 遗传仪器ID。
  - `pval`: 提取暴露数据的P值阈值，默认为5e-08。
  - `r2`: 遗传仪器间的LD (linkage disequilibrium) 值的阈值，默认为0.001。
  - `kb`: 遗传仪器的范围 (以kb为单位) , 默认为10000。
- `get_outcome` 函数的参数说明：
  - `id`: 遗传仪器ID。
  - `expo`: 暴露数据, TwoSampleMR包格式

## get\_ao

### 描述

获取OPEN GWAS数据库所有可用的ID。可以指定获取某个前缀的ID

### 用法

使用这个函数前, 需要先安装并加载TwoSampleMR包。然后可以直接在R语言中调用这个函数, 如下所示:

```
1 # 调用get_ao函数
2 ao <- get_ao()## 不限定来源则返回所有id
3 ao <- get_ao(a = "finn")## 这样就会返回finn的所有可用id
```

## 参数

- `a`: (可选) 数据来源。

### 备注: 来源的名称

来自OPEN GWAS [Browse the IEU OpenGWAS project \(mrcieu.ac.uk\)](https://www.ebi.ac.uk/gwas/browse/ieuoopen) 5.1获取

Batch	Description	<a href="#">Count</a>
<a href="#">bbj-a</a>	<a href="#">Biobank Japan release of disease traits</a>	120
<a href="#">ebi-a</a>	<a href="#">Datasets that satisfy minimum requirements imported from the EBI database of complete GWAS summary data</a>	2,585
<a href="#">eqtl-a</a>	<a href="#">eQTLGen 2019 results, comprising all cis and some trans regions of gene expression in whole blood</a>	19,942
<a href="#">finn-b</a>	<a href="#">FinnGen biobank analysis round 5</a>	2,803

Batch	Description	Count
<a href="#">ieu-a</a>	<a href="#">GWAS summary datasets generated by many different consortia that have been manually collected and curated, initially developed for MR-Base</a>	440
<a href="#">ieu-b</a>	<a href="#">GWAS summary datasets generated by many different consortia that have been manually collected and curated, initially developed for MR-Base (round 2)</a>	207
<a href="#">met-a</a>	<a href="#">Human blood metabolites analysed by Shin et al 2014</a>	452
<a href="#">met-b</a>	<a href="#">Human immune system traits analysed by Roederer et al 2015</a>	150
<a href="#">met-c</a>	<a href="#">Circulating metabolites analysed by Kettunen et al 2016</a>	123
<a href="#">met-d</a>	<a href="#">Metabolic biomarkers in the UK Biobank measured by Nightingale Health 2020</a>	249
<a href="#">prot-a</a>	<a href="#">Complete GWAS summary data on protein levels as described by Sun et al 2018</a>	3,282
<a href="#">prot-b</a>	<a href="#">Complete GWAS summary data on protein levels as described by Folkersen et al 2017</a>	83
<a href="#">prot-c</a>	<a href="#">Complete GWAS summary data on protein levels as described by Suhre et al 2017</a>	1,124
<a href="#">ubm-a</a>	<a href="#">Complete GWAS summary data on brain region volumes as described by Elliott et al 2018</a>	3,143
<a href="#">ukb-a</a>	<a href="#">Neale lab analysis of UK Biobank phenotypes, round 1</a>	596
<a href="#">ukb-b</a>	<a href="#">IEU analysis of UK Biobank phenotypes</a>	2,514
<a href="#">ukb-d</a>	<a href="#">Neale lab analysis of UK Biobank phenotypes, round 2</a>	904
<a href="#">ukb-e</a>	<a href="#">Pan-ancestry genetic analysis of the UK Biobank performed at the Broad Institute</a>	3,873

## clean\_outcome\_from\_exposure

### 描述

(主要用于向量化) 用于清洗outcome。将exposure中 (list形式, 也就是批量化的形式存在的exposure) 不存在的SNP从outcome中剔除, 大幅精简outcome, 并大幅提升harmonise\_data的速度。

**实测清洗与不清洗outcome对比, 速度相差一百倍以上。**

## 用法

直接在R语言中调用这个函数，如下所示：

```
1 # 调用clean_outcome_from_exposure函数
2 cleaned_outcome <- clean_outcome_from_exposure(expo = exposure_data, outcome = outcome_data)
```

## 参数

- `expo`: 暴露数据，格式为**list**，TwoSampleMR包格式。比如get\_exposure批量化获取下来的数据
- `outcome`: 结果数据，TwoSampleMR包格式。

## clean\_GWAS

---

### 描述

这个函数是用于清洗遗传关联数据集，使其符合特定的数据集要求的功能函数。

### 用法

直接在R语言中调用这个函数，如下所示：

```
1 # 调用clean_GWAS函数
2 cleaned_GWAS_list <- clean_GWAS(list = GWAS_list, clean = c("bbj", "eqtl"))
```

## 参数

- `list`: 一个list。里面包含每个暴露的data.frame。具体参考批量运行get\_exposure后的结果
- `clean`: 需要清洗的数据集名称，一个字符型向量类型。具体参考get\_ao的附注。

## 返回值

- `clean_GWAS` 函数返回一个清洗后的遗传关联数据集列表，符合特定数据集要求。