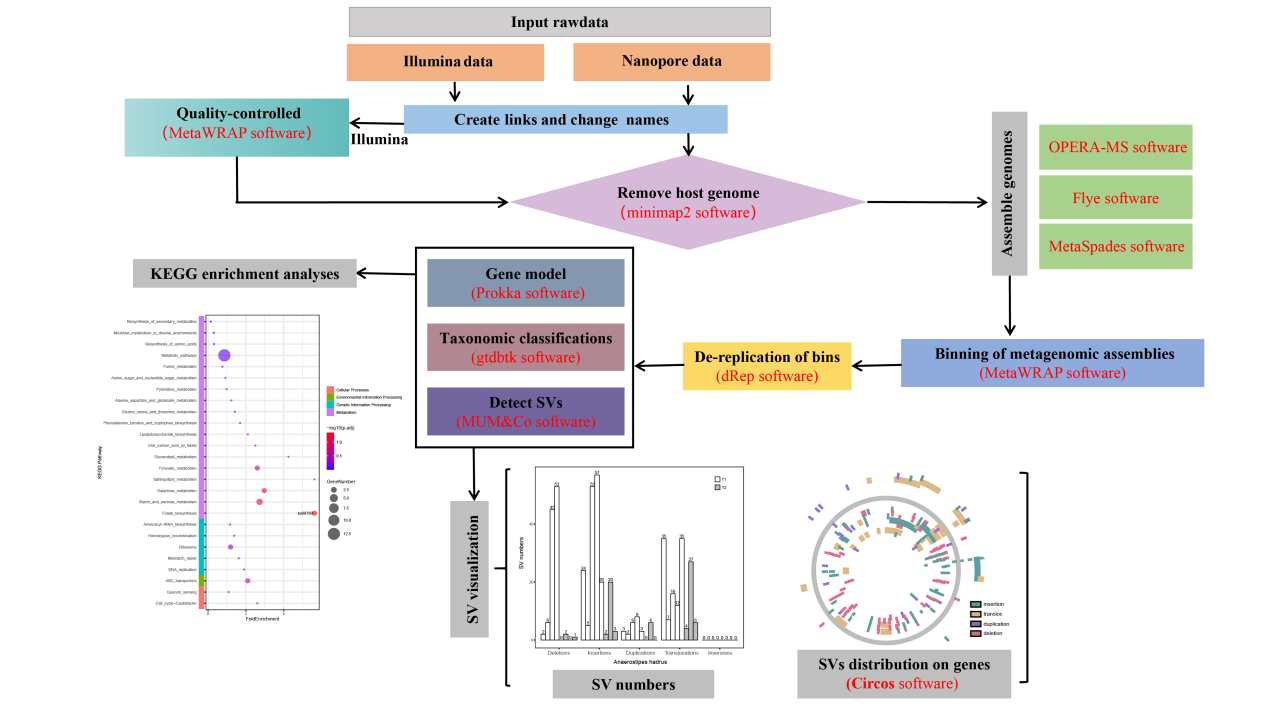
**Introduction**

MetaSVs is a pipeline combining Nanopore long reads and Illumina short reads to analyze SVs in the microbial genomes from gut microbiome, and further identify differential SVs that can be reflective of metabolic differences. The pipeline integrates multiple software tools and its core mission consists of 13 steps, including the creation of soft links, quality control and sequence statistics, removal of host reads, metagenome assembly and evaluation, extraction of high-quality draft genomes (bins) and de-replication, species taxonomy and gene models of bins, detection and visualization of SVs, and KEGG enrichment analyses. MetaSVs gives researchers easy access to SVs and relevant metabolites in the microbial genomes without the requirement of specific technical expertise, which is particularly useful to researchers interested in metagenomic SVs but lacking sophisticated bioinformatic knowledge.

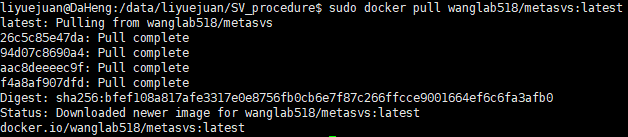


**Installation**

To install with Docker, run:

# Docker installation

docker pull wanglab518/metasvs:latest



**Databases**

In addition to the Docker installation, you will need to configure the paths to some databases that you downloaded onto your system according to the following guide.

|  |  |  |
| --- | --- | --- |
| **Database** | **Size** | **Used in module** |
| CheckM\_db | 1.4G | binning, de-replication |
| gtdbtk\_db | 49G | taxonomy |
| kobas\_db | 54G | KEGG enrichment |

1. **Downloading the CheckM database:**

mkdir CheckM\_db && cd CheckM\_db

wget https://data.ace.uq.edu.au/public/CheckM\_databases/checkm\_data\_2015\_01\_16.tar.gz

tar xvzf checkm\_data\_2015\_01\_16.tar.gz

rm checkm\_data\_2015\_01\_16.tar.gz

1. **Downloading the gtdbtk database:**

wget https://data.gtdb.ecogenomic.org/releases/release202/202.0/auxillary\_files/gtdbtk\_r202\_data.tar.gz

tar xvzf gtdbtk\_r202\_data.tar.gz

mv release202 gtdbtk\_db

rm gtdbtk\_r202\_data.tar.gz

1. **Downloading the** **kobas** **database:**

mkdir kobas\_db && cd kobas\_db

#to download sqlite3.tar.gz and seq\_pep.tar.gz

Link: <ftp://ftp.cbi.pku.edu.cn/pub/KOBAS_3.0_DOWNLOAD/>

tar xvzf sqlite3.tar.gz && tar xvzf seq\_pep.tar.gz

rm sqlite3.tar.gz seq\_pep.tar.gz

**Usage**

1. To create a working directory

mkdir -p $workpath && cd $workpath

1. To run a MetaSVs docker container:

docker run --name sv\_project -v $workpath:/opt/project -idt wanglab518/metasvs

docker start sv\_project

docker exec -it sv\_project /bin/bash

1. To prepare the table of the data information according to the “SV\_procedure/test/sample\_info.txt” file

##“sample\_info.txt” file used to save the table of sample information, including 4 columns separated by TAB

#sample\_IDs: sample names

#NGS\_rawnames: the names of Illumina sequence data

#ONT\_rawnames: the names of Nanopore sequence data

#group: the group names of the sample

vi sample\_info.txt

1. To prepare the configuration file as the pipeline input according to the “SV\_procedure/test/config.ini” file

vi config.in #including the path of database downloaded above, raw sequence data and so on

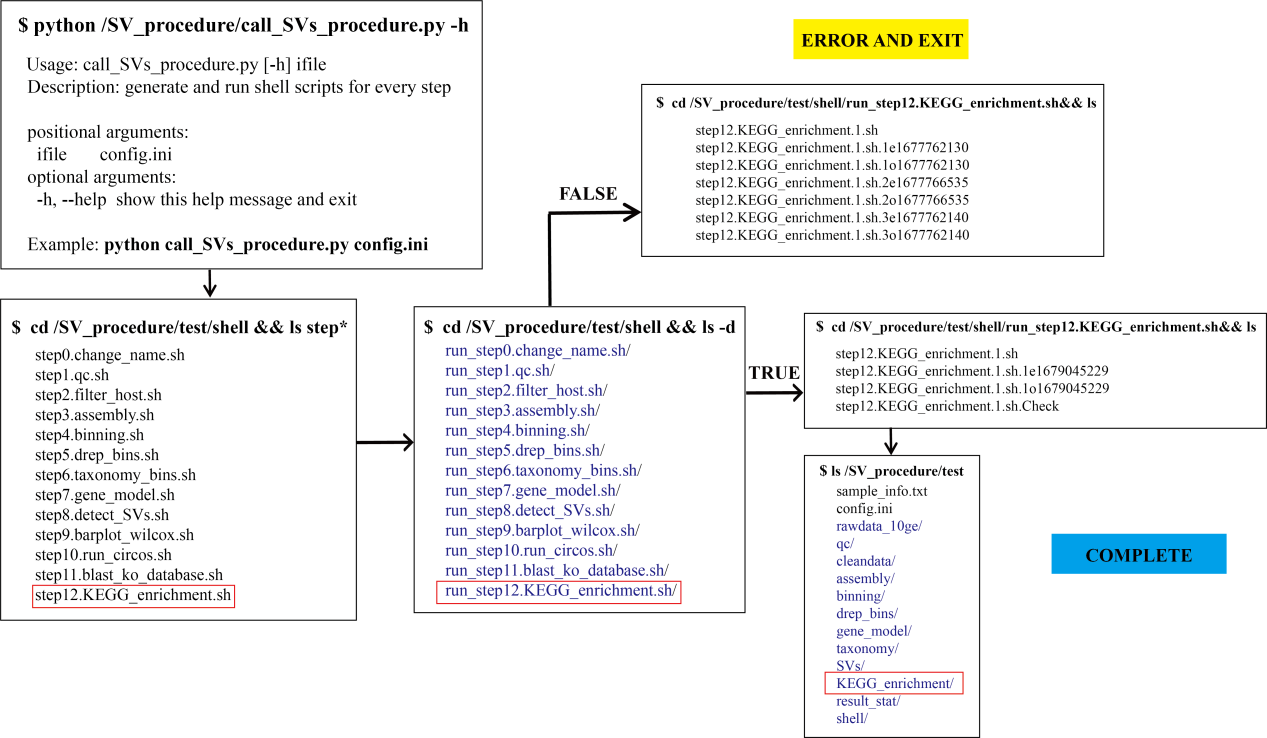
1. To run the pipeline in a docker container, simply use the following command (we recommend using the “nohup” command to run the pipeline in the background):

cd /opt/project

python /opt/conda/SV\_procedure/call\_SVs\_procedure.py config.ini

**Output**

The output includes the source code, the running process and the running results of the scripts. In order to support the process traceability, a "shell/" directory is first generated to hold the script files, and then the specific process of script execution is stored in the "step\*.sh.\*e\*" (stored error information) and the "step\*.sh.\*o\*" (stored the normal process) files in the "run\_\*.sh" sub-folders of the current folder. Taking the step12 KEGG enrichment analysis as an example, the running process of the "step12.KEGG\_enrichment.sh" script is stored in the "run\_step12.KEGG\_enrichment.sh" folder. If the script is functioning normally, the "step12.KEGG\_enrichment.1.sh.Check" file is generated. If not, the execution will exit after the loop has executed three times. The resulting output files from step 0 to step 12 are respectively stored in the following directories: "rawdata\_10ge/", "qc/", "cleandata/", "assembly/", "binning/", "drep\_bins/", "gene\_model/", "taxonomy/", "SV/", "KEGG\_enrichment/" and "result\_stat/".



**Error**

when the program failure occurs due to a variety of corruption or bugs (such as file, network or disk issues), the program execution does not simply exit but enters the next execution using an identical scripting code. After the loop executes three times, if the failure still exists, the program will exit and errors in during this step will be stored in the “shell/run\_step\*/\*.e\*” files. After troubleshooting, we can delete the “\*.1o\*, \*.1e\*, \*.2o\*, \*.2e\*, \*.3o\*, \*.3e\*” files or this “run\_step\*” folder, and then rerun the pipeline:

python /opt/conda/SV\_procedure/call\_SVs\_procedure.py config.ini

By now, the program will be able to recover from the nearest checkpoint rather than overwrite otherwise usable intermediate files, obviating the need to restart from the beginning of a process.

**Examples**

The example project can be found in the following directory: SV\_procedure/test