

Yeo Meng Han  
A0251772A  
EE3801 Lab 6A

Submit the following task for lab report for Part B:

### Step 14

checkfiles2.sh

```
Processing output - output/metrics.json
#!/bin/bash
["metrics_out":"/data/picasso/20181105/mountains/channel009/output/metrics.json"]
echo "Number of hkl files" ; story ...
find mountains -name *.hkl | grep -v -e spiketrain -e mountains | wc -l
[ Preparing temporary outputs... ]
echo "Number of mda files" ; picasso/20181105/mountains/channel009/output/metrics.json
find mountains -name "firings.mda" | wc -l
[ Initializing process... ]
# Separator... ] /data/miniconda3/envs/env1/bin/python3 /data/miniconda3/envs/env1/etc/mountainlab/packages/ml_ephys/basic
echo "#-----"
echo "#-----" ephys.compute_cluster_metrics /tmp/mountainlab-tmp/tempdir_57c04adada_swF5u5 --firings=/tmp/mountainlab-tmp/tempdir_57c04adada_swF5u5/input_firings_7pPjkFpD.mda --timeseries=/tmp/mountainlab-tmp/tempdir_57c04adada_swF5u5/output_metrics_out.json
for file in pipe-slurm*.out; do
    Read# Check if the file contains the start time pattern
    If [ $(grep -q "time.struct_time" "$file"); then
        Computing echo "=====$file <="
        Computing grep "time.struct_time" "$file" | head -n 1
        Writing output...
        done.

    echo "End Times" processor ephys.compute_cluster_metrics: 5.696 sec
    for file in pipe-slurm*.out; do
        [ Re# Check if the file contains the end time pattern
        Do if [ $(grep -q "time.struct_time" "$file"); then
            finished echo "=====$file <="
            time_struct=$(grep "time.struct_time" "$file" | tail -n 1)
            tm_hour=2, tm_min=23, tm_sec=4, tm_wday=0, tm_yday=296, tm_isdst=0
        fi
        230.# Output the time taken (s) and MessageId for clarity
        expect tail -n 3 "$file" channel level
        /date echo picasso/20181105/sessioneye/array01/channel009/cell01
        done
        count
        249
        echo "#-----"
        Object saved to file spiketrain_d41d.hkl
        /data/picasso/20181105/sessioneye/array01/channel009/cell02
    done
done
```

checkfiles2.sh output:

```
(env1) [ec2-user@ip-10-0-13-114 20181105]$ nano /data/src/PyHipp/checkfiles2.sh
(env1) [ec2-user@ip-10-0-13-114 20181105]$ bash /data/src/PyHipp/checkfiles2.sh
Number of hkl files
53
Number of mda files
8
=====
Start Times
==> rplpl-slurm.queue1-dy-m5a-4xlarge-1.2.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=23, tm_hour=1, tm_min=59, tm_sec=50, tm_wday=0, tm_yday=296, tm_isdst=0)
==> rplspl-slurm.queue1-dy-m5a-4xlarge-1.3.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=23, tm_hour=1, tm_min=59, tm_sec=50, tm_wday=0, tm_yday=296, tm_isdst=0)
End Times
==> rplpl-slurm.queue1-dy-m5a-4xlarge-1.2.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=23, tm_hour=2, tm_min=33, tm_sec=10, tm_wday=0, tm_yday=296, tm_isdst=0)
1999.3222522735596
{
    "MessageId": "67f7c943-ee88-58aa-bcd5-94ec332f62e5"
}

==> rplspl-slurm.queue1-dy-m5a-4xlarge-1.3.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=23, tm_hour=2, tm_min=45, tm_sec=45, tm_wday=0, tm_yday=296, tm_isdst=0)
2754.17416262665
{
    "MessageId": "f78d87ac-a680-568e-88a5-da03461a72b3"
}
=====
(env1) [ec2-user@ip-10-0-13-114 20181105]$ ]
```

## Calculations:

The screenshot shows a Mac desktop environment. On the left, a terminal window displays a bash script for calculating job durations. On the right, a Finder window titled 'Lab 7' lists several files: '(L7) Lab Handout.docx', '(L7) Lab Handout.pdf', 'Lab7\_WangZihan.docx', 'Lab7\_WangZihan.pdf', and 'Stuff.txt'. The terminal window contains the following script:

```
#!/bin/bash

# Defined durations for each job in seconds
duration_job1=1999.3222522735596
duration_job2=2754.17416262665

# Extracted start and end times for each job
start_job1="2023-10-23 01:59:50"
end_job1="2023-10-23 02:33:10"

start_job2="2023-10-23 01:59:50"
end_job2="2023-10-23 02:45:45"

# Convert to epoch for calculations
startEpoch1=$(date --date="$start_job1" +%s)
endEpoch1=$(date --date="$end_job1" +%s)

startEpoch2=$(date --date="$start_job2" +%s)
endEpoch2=$(date --date="$end_job2" +%s)

# c. Sum of the two job durations in hours, minutes, and seconds
total_duration=$(echo "$duration_job1 + $duration_job2" | bc)
hours=$(bc <<< "$total_duration/3600")
minutes=$(bc <<< "($total_duration%3600)/60")
seconds=$(bc <<< "$total_duration%60")

echo "c. Total Duration: $hours hours, $minutes minutes, $seconds seconds"

# d. Actual time taken for both jobs to complete
actual_duration=$(echo "$endEpoch2 - $startEpoch1" | bc)
hours=$(bc <<< "$actual_duration/3600")
minutes=$(bc <<< "($actual_duration%3600)/60")
seconds=$(bc <<< "$actual_duration%60")

echo "d. Actual time taken: $hours hours, $minutes minutes, $seconds seconds"

# e. Time saved in parallelizing the jobs
time_saved=$(echo "$total_duration - $actual_duration" | bc)
hours=$(bc <<< "$time_saved/3600")
minutes=$(bc <<< "($time_saved%3600)/60")
seconds=$(bc <<< "$time_saved%60")

echo "e. Time saved: $hours hours, $minutes minutes, $seconds seconds"

# f. Time it will take to process all 110 channels
channels=110
current_channels=8
time_for_channels=$(echo "$actual_duration * ($channels/$current_channels)" | bc)
hours=$(bc <<< "$time_for_channels/3600")
minutes=$(bc <<< "($time_for_channels%3600)/60")
seconds=$(bc <<< "$time_for_channels%60")

echo "f. Time for 110 channels (estimated): $hours hours, $minutes minutes, $seconds seconds"
```

## Calculations output:

- c. Total Duration: 1 hours, 19 minutes, 13.4964148998261 seconds
- d. Actual time taken: 0 hours, 45 minutes, 55 seconds
- e. Time saved: 0 hours, 33 minutes, 18.4964148998261 seconds
- f. Time for 110 channels (estimated): 9 hours, 56 minutes, 55 seconds

## Step 62

Part a: checkfiles2.sh number of files created:

```
[cenv1] [ec2-user@ip-10-0-14-36 20181105]$ bash /data/src/PyHipp/checkfiles2.sh
Number of hkl files
665
Number of mda files
110
"
```

## Part b: checkfiles2.sh start and end times

```
#=====
Start Times
==> rplpl-slurm.queue1-dy-m5a-4xlarge-1.19.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=16, tm_min=8, tm_sec=43, tm_wday=1, tm_yday=297, tm_isdst=0)
==> rs1-slurm.queue1-dy-m5a-4xlarge-1.20.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=16, tm_min=8, tm_sec=43, tm_wday=1, tm_yday=297, tm_isdst=0)
==> rs2-slurm.queue1-dy-m5a-4xlarge-2.21.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=16, tm_min=8, tm_sec=43, tm_wday=1, tm_yday=297, tm_isdst=0)
==> rs3-slurm.queue1-dy-m5a-4xlarge-3.22.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=16, tm_min=8, tm_sec=43, tm_wday=1, tm_yday=297, tm_isdst=0)
==> rs4-slurm.queue1-dy-m5a-4xlarge-1.23.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=18, tm_min=7, tm_sec=4, tm_wday=1, tm_yday=297, tm_isdst=0)
End Times
==> rplpl-slurm.queue1-dy-m5a-4xlarge-1.19.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=16, tm_min=46, tm_sec=12, tm_wday=1, tm_yday=297, tm_isdst=0)
2248.4354314804077
{
    "MessageId": "8bf5ac60-9ba2-578b-86b8-f1d16cb733d9"
}

==> rs1-slurm.queue1-dy-m5a-4xlarge-1.20.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=18, tm_min=7, tm_sec=0, tm_wday=1, tm_yday=297, tm_isdst=0)
7096.6148455142975
{
    "MessageId": "72f6f1bb-2178-5932-9d73-ac092b7dc60"
}

==> rs2-slurm.queue1-dy-m5a-4xlarge-2.21.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=18, tm_min=21, tm_sec=48, tm_wday=1, tm_yday=297, tm_isdst=0)
7984.86262345314
{
    "MessageId": "30a196c5-d02a-5269-8d30-cb77dd54e438"
}

==> rs3-slurm.queue1-dy-m5a-4xlarge-3.22.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=18, tm_min=14, tm_sec=12, tm_wday=1, tm_yday=297, tm_isdst=0)
7528.819412469864
{
    "MessageId": "4bceac8f-261b-5619-b704-f13d2dad2bac"
}

==> rs4-slurm.queue1-dy-m5a-4xlarge-1.23.out <==
time.struct_time(tm_year=2023, tm_mon=10, tm_mday=24, tm_hour=19, tm_min=28, tm_sec=20, tm_wday=1, tm_yday=297, tm_isdst=0)
4875.603684902191
{
    "MessageId": "91541001-7338-5c59-a3da-4db36e2d280c"
}
#=====
```

This part consists of:

5. Checking and resubmitting jobs (Estimated time: 20 mins)
6. Processing all remaining data (Estimated time: 5 mins)
7. Wrapping up (Estimated time: 5 mins)

Repeat these steps from Lab 5:

### Step 34 (showing that you only have 1 instance and 1 volume running)

The screenshot shows the AWS EC2 Dashboard for the Asia Pacific (Singapore) Region. The top navigation bar includes options like [Option+S], EC2 Global view, and account information for Singapore and user 'yeo\_menghan'.

**Resources** section:

Instances (running)	1	Auto Scaling Groups	0	Dedicated Hosts	0
Elastic IPs	0	Instances	1	Key pairs	1
Load balancers	0	Placement groups	0	Security groups	4
Snapshots	2	Volumes	1		

**Account attributes** section:

- Default VPC**: [vpc-016d3bd5850764ca4](#)
- Settings**: [Data protection and security](#), [Zones](#), [EC2 Serial Console](#), [Default credit specification](#), [Console experiments](#)

**Launch instance** section:

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

**Launch instance** button (orange), [Migrate a server](#)

Note: Your instances will launch in the Asia Pacific (Singapore) Region

**Scheduled events** section:

[Asia Pacific \(Singapore\)](#)

**Service health** section:

AWS Health Dashboard

Region: Asia Pacific (Singapore)

**Zones** table:

Zone name	Zone ID
ap-southeast-1a	apse1-az1
ap-southeast-1b	apse1-az2
ap-southeast-1c	apse1-az3

**Explore AWS** sidebar:

- Amazon GuardDuty Malware Protection**: GuardDuty now provides agentless malware detection in Amazon EC2 & EC2 container workloads. [Learn more](#)
- Get Up to 40% Better Price Performance**: T4g instances deliver the best price performance for burstable general purpose workloads in Amazon EC2. [Learn more](#)
- Save up to 90% on EC2 with Spot Instances**: Optimize price-performance by combining EC2 purchase options in a single EC2 ASG. [Learn more](#)

## Step 36

The screenshot shows the AWS Billing Bills page. At the top, there are navigation links, a search bar with placeholder '[Option+S]', and various global settings like 'Global' and 'yeo\_menghan'. Below the header, the main content area is titled 'Bills' with a 'Info' link. It includes buttons for 'Download all to CSV', 'Print', and a dropdown for 'Billing period: October 2023'. A note says 'Page refresh time: Wednesday, October 25, 2023 at 3:51:20 AM GMT+8'.

**AWS estimated bill summary** [Info](#)

Total charges and payment information

Account ID <b>547215547739</b>	Billing period <a href="#">Info</a> <b>October 1 - October 31, 2023</b>
Service provider <b>Amazon Web Services Singapore Private Limited</b>	Total in USD <b>USD 0.00</b>

**Estimated grand total:** **USD 0.00**  
Payable by Account ID: 870849704511

**▶ Payment information** [Info](#)

**Highest estimated cost by service provider** [Info](#)

Viewing Amazon Web Services Singapore Private Limited

Highest service spend <b>USD 0.00</b>	Trend compared to prior month No data to display.	Highest AWS Region spend <b>USD 30.52</b>	Trend compared to prior month No data to display.
Service name	Region name		

## Step 37

AWS Cost Management > Home [Option+S]

Home [Info](#)

**Cost summary**

Current month costs [Info](#) \$30.58 Up 100% over last month

Forecasted month-end costs [Info](#) - Down 0% over last month

**Daily unblended costs** [View in Cost Explorer](#)

Cost (\$)

Date	Cost (\$)
Sep 03	0.5
Sep 08	0.5
Sep 13	6.0
Sep 18	2.8
Sep 23	2.0
Sep 28	1.5
Oct 03	1.5
Oct 08	1.5
Oct 13	1.5
Oct 18	1.5
Oct 23	1.5
Oct 28	8.0
Oct 29	5.5

**October trends** [Info](#)

Once you have more usage across AWS, we will provide helpful cost and usage insights.

**More resources** [View](#)

[What is AWS Billing and Cost Management?](#)

[Documentation](#)

[FAQ](#)

## Step 75

The screenshot shows the AWS Billing Overview page. At the top, there is a navigation bar with icons for search, refresh, help, and user information (Global and yeo\_menghan). Below the navigation bar, the breadcrumb path is AWS Billing > Budgets > Overview. The main title is "Overview" with an "Info" link. A sub-header "Budgets (1) Info" is followed by a "Create budget" button. There is a search bar with placeholder text "Find a budget" and a "Show all budgets" dropdown. Below these are buttons for navigating between pages and filtering results. A table displays the single budget entry:

<input type="checkbox"/>	Name	Thresholds	Budget	Amount us...	Forecasted...	Current vs. budgeted
<input type="checkbox"/>	EE3801 Budget	OK	\$200.00	\$30.58	-	15.29