CSC8634_TeraScope Report

Morgan Frodsham (210431461)

10/01/2022

Extended Technical Project: Performance Evaluation of Terapixel Rendering in Cloud (Super)computing

Understanding

Background

This report details the process and findings of an extended technical project on Newcastle University's cloud supercomputer architecture for visualising environmental data captured by the Newcastle Urban Observatory. It explores the data created during the computational production of *[this]* (http://terapixel.wasabi.1024.s3.eu-central-1.wasabisys.com/vtour/index.html) visualisation of Newcastle upon Tyne. Newcastle University and the Newcastle Urban Observatory started this initiative with the idea that "the only way to understand how to monitor a city is to try and monitor one" with a monitoring platform that matches "the scale, complexity and scope of our modern cities" (P James (2016)). It offers an unparalleled opportunity to improve decision making the city by accurately measuring and analysing our urban environment.

Objectives and success criteria

Urban Observatories attempt to replicate the "breadth, longevity and success of astronomical observatories"(P James (2016)) in understanding how cities operate. Funded in partnership with the UK Collaboratorium for Research in Infrastructure and Cities ("Details of Grant" (n.d.)), the Newcastle Urban Observatory collects environmental data about the city of Newcastle-upon-Tyne (Observatory (n.d.)). This is considered to be the most expansive set of "environmental data in the UK, with: over 74 urban indicators; 7,000 observations every minute; 3,600 sensor streams; 9 billion data points; 540 CCTV sensors; and hundreds of millions of images" (Newcastle University (2021)). Newcastle University has created a scalable cloud supercomputer software architecture for visualising this data as realistic terapixels (Forshaw (2021)). Terapixels are images containing over one trillion pixels (Holiman et. al (2019)) that allow viewers to interactively browse big data in intense detail across multiple scales (Forshaw (2021)). The terapixel visualisation created by Newcastle University's cloud supercomputer allows the viewer to "zoom in from an overview of just one square kilometre of the city… to see detail within a single room in an office or a house" (Holiman et. al (2019)).

In delivering this initative, Newcastle University had three objectives:

1. "create a supercomputer architecture for scalable viualisation using the public cloud;
2. produce a terapixel 3D city viusalisation supporting daily updates; and
3. undertake a rigorous evaluation of cloud supercomputing for compute intensive visualisation applications" (Forshaw (2021)).

This extended technical project focuses on the third objective. It explores three datasets created from the data produced by the supercomputer architecture during the creation of the terapixel visualisation, and aims to provide useful insights into the computational performance. The process and findings of this report explore the question: "How could the computation of a terapixel visualisation be more efficient?" The success criteria of the project is to identify useful insights about the stages of terapixel computation which are most resource intensive. Developing a better understanding of this is important as producing a high quality terapixel image is an intense process with significant computational costs (resource, energy, environmental and monetary) (Holiman et. al (2019)).

With a growing focus on doing "more with less" ("Efficiency" (n.d.)), cloud computing suppliers and users are trying to continue scaling with fewer computational costs (such as fewer servers and less energy). For humankind to fully reap the benefits of information provided by these visualisations, the computation needs to be efficient and associated costs must be sustainable. Improving the efficiency cloud computing has been a core focus for suppliers and users; indeed, Google returns 90.9 million results for "cloud computing efficiency." As the likelihood and potential impact of climate change is fully realised, users as well as wider society are increasingly concerned about the sustainability of digital technologies, such as cloud computing. Just five organisations - Amazon, Google, Microsoft, Facebook and Apple - use as much electricity annually as New Zealand (Kantor (2021)). Moreover, the Shift Project suggests that tech-related emissions are rising by 6 per cent annually (Kantor (2021)). Data centres are estimated to account for "1% of the total global energy demand" (Renee Obringer and Madani (2021)) Investigating the resource intensity of computing intensive visualisation applications, such as terapixels, is therefore a vital component of a rigorous evaluation and a necessary step towards greater sustainability.

Data mining goals and success criteria

To explore the question "How could the computation of a terapixel visualisation be more efficient?" this extended technical project seeks to identify useful insights about the stages of terapixel computation which are most resource intensive. To achieve this, the goals of data mining are:

1. Which events dominate task duration (run time)? The data mining success criteria is calculating the duration of each event to determine which has the longest duration.
2. What is the relationship between duration and GPU power draw? The data mining success criteria is identifying the power needed for duration time to better understand what demands more GPU power.
3. What is the relationship between GPU power draw and performance? The data mining success criteria is identifying the impact of GPU temperature, core utilisation and memory demand on GPU power to understand what demands more power.

[SECEOND ITERATION] 4. What is the variation in computation requirements for particular tiles? The data mining success criteria is identifying if there is a variation in computing requirements to determine which tiles are most intensive to produce.

5. Are there particular GPU cards (based on their serial numbers) whose performance differs to other cards? The data mining success criteria is identifying the outlier GPU cards (for example, cards that are perpetually slow) to determine which cards could benefit from further investigation to improve computation performance.

[THIRD ITERATION] 6. What can we learn about the efficiency of the task scheduling process? **...**

- we should see efficientl linear scalaing as we add more compute notdes - gustafosn-Baris' law Holiman et. al (2019) Rendering images is considered to be a good test of hardware performance as it is capable of absorbing all available compute resource, following Gustafson-Barsis' law that the problem size scales to fill the compute capacity (Holiman et. al (2019)).

Data Understanding

Initial data collection

The data for this extended technical project was created during a run (from application checkpoint and system metric output) using 1024 GPU nodes, and this run processes three levels of visualisation (levels 4, 8 and 12) to render the terapixel (Forshaw (2021)). It is unclear whether there have been any problems in data acquisition or extraction as Newcastle University has provided the data.

The data provided shows the timings of producing the terapixel, GPU card performance, and the coordinates for the part of the terapixel image that was rendered in each task (Forshaw (2021)) in three different data sets. This data appears to be raw, but of sufficient quality to be tided and analysed. It will provide sufficient information for an initial investigation into the stages of terapixel computation process which are most resource intensive.

Data description

The run to compute the terapixel visualisation provides the following data sets:

- the application checkpoint events throughout the execution of the render job (timestamp, hostname, eventName, eventType, jobId, taskId);
- the metrics output relating to the status of the GPU on the virtual machines (timestamp, hostname, gpuSerial, gpuUUID, powerDrawWatt, gpuTempC, gpuUtilPerc, gpuMemUtilPerc); and
- the x,y co-ordinates of the part of the image being rendered for each task (jobId, taskId, x, y, level).

Data exploration

The data exploration outlined in this report began with initial questions to better understand what each data set contained and visual investigation to obverse any obvious patterns. Further details of this initial exploration can be seen in the file 'eda.R' in the src folder. Examples are shared below:

1. There are 1024 unique host names, which suggests that these are the names of each virtual machine containing the GPU nodes.
2. There are 1024 unique GPU serials, which suggests that each virtual machine has one GPU node with its own unique serial number.
3. It appears that some virtual machines complete more 'TotalRender' than others, suggesting that some are more productive.
4. There are 65793 taskIds in the applications.checkpoint data set (copied as AC1), which matches the number of taskIds in the task.x.y data set.
5. There are 3 distinct jobIds in the applications.checkpoint data set (copied as AC1), which is the same in task.x.y, and seems to reflect the three visualisation levels (level 4, 8 and 12).

To address the key objective and success criteria, the exploratory analysis outlined in the rest of this section starts with applications.checkpoints, moves to gpu, and finishes with the task.x.y in line with the data mining goals and is structured by those goals. Each goal has it's own eda file in the src folder.

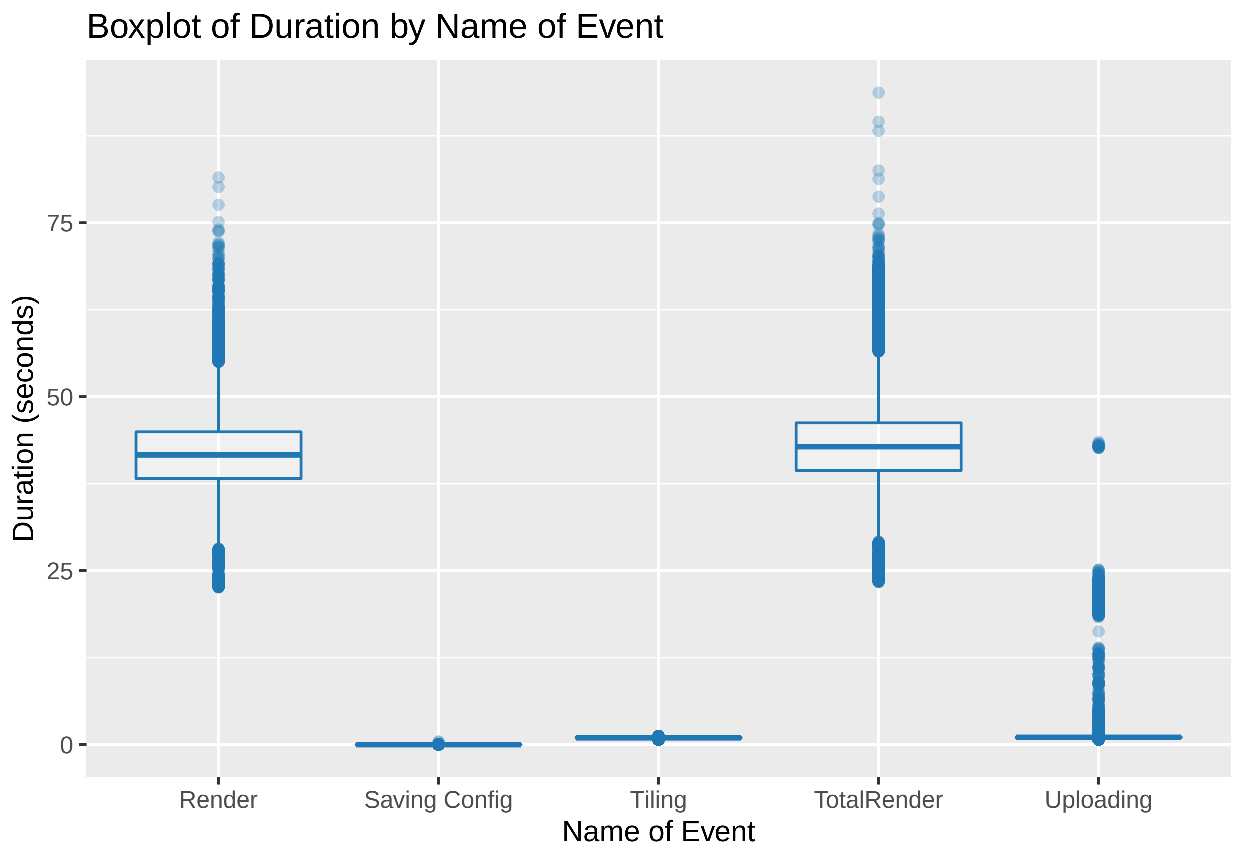
Goal 1 (Run time)

To create a terapixel, Newcastle University render levels 4, 8 and 12, "and subsample these to fill in each set of three intermediate levels" (Holiman et. al (2019)). There are 5 different events (eventName) that happen during the creation of a visualisation: Saving Config, Render, Tiling, Uploading and TotalRender. The data set application.checkpoints records the timestamp of when these events start and stop (eventType).

To determine which events have the longest duration (which means that they dominate the run time), the application.checkpoints data set is copied as AC1. AC2 applies a pivot wider function to AC1 so that eventType 'START' and 'STOP' become columns for the corresponding timestamp for each eventName. Both columns are mutated so that their values are date and time. In a new column, duration is calculated by subtracting START from STOP.

To investigate the duration of each event, a boxplot was created of duration by event name.

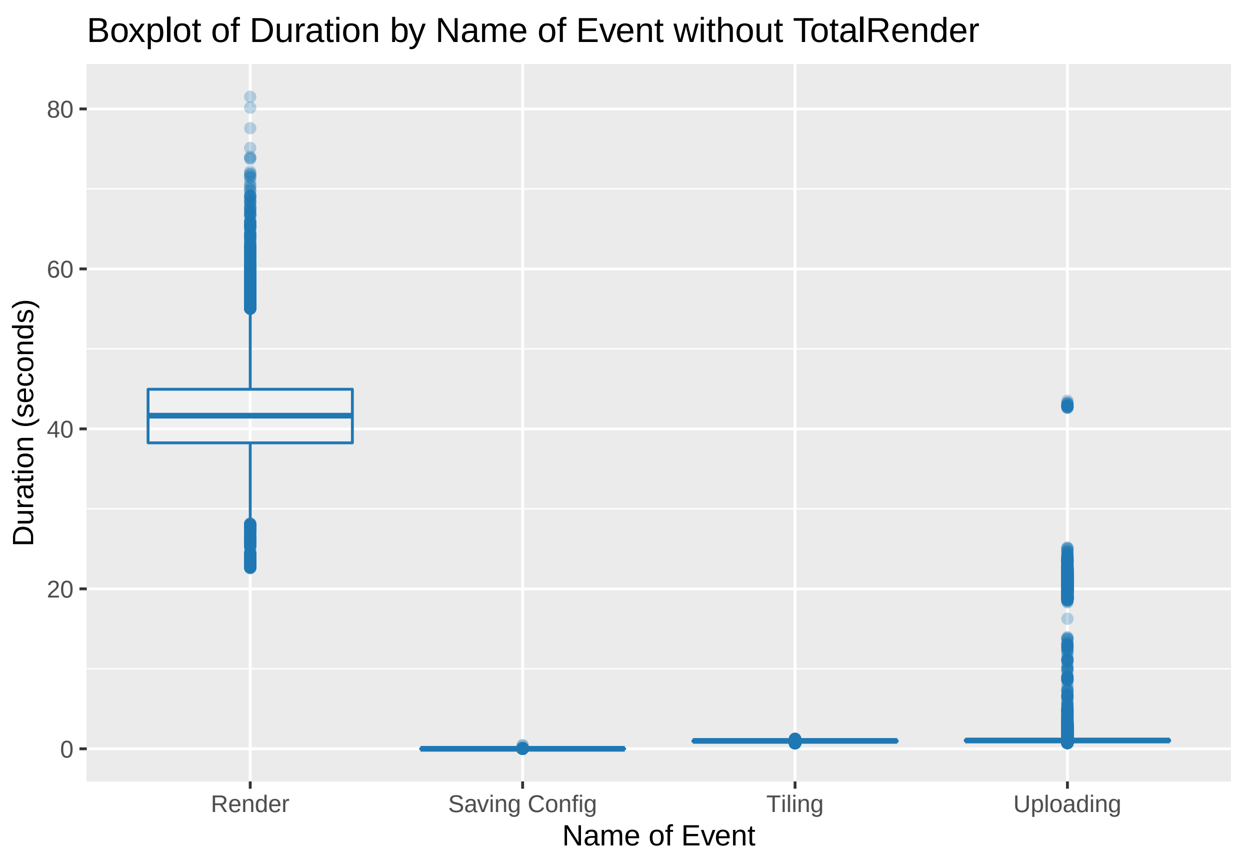
```
## Warning: Removed 1235 rows containing non-finite values (stat_boxplot).
```



This highlights that 'TotalRender' has the longest duration. It is assumed that this is because it includes the duration of all the other events.

AC3 is created by filtering out TotalRender from AC2. Another boxplot was created of duration by event name.

```
## Warning: Removed 988 rows containing non-finite values (stat_boxplot).
```



It is clear that 'Render' has the longest duration, and therefore is the event that dominates the run time.

```
## # A tibble: 50 x 7
## # Groups:   hostname [47]
##   hostname                               eventName
##   <chr>                                  <chr>
## 1 0d56a730076643d585f77e00d2d8521a00000I Render
## 2 4a79b6d2616049edbf06c6aa58ab426a000003 Render
## 3 95b4ae6d890e4c46986d91d7ac4bf08200000G Render
## 4 6139a35676de44d6b61ec247f0ed865700000I Render
## 5 5903af3699134795af7eafc605ae5fc700000U Render
##   jobId
##   <chr>
## 1 1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705
## 2 1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705
## 3 1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705
## 4 1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705
## 5 1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705
##   taskId                               START               STOP
##   <chr>                                <dttm>              <dttm>
## 1 a95d501e-d5d5-4fb4-9119-98120bf6f4d5 2018-11-08 08:06:34 2018-11-08 08:07:55
## 2 25b410b5-f5ef-4a2f-8b21-29175bca35fc 2018-11-08 07:50:57 2018-11-08 07:52:17
## 3 d194b27d-d421-47d3-ae41-eed07a00e8d4 2018-11-08 08:15:39 2018-11-08 08:16:56
## 4 f4a61a45-2e92-4aa0-9219-4425ce0ec17e 2018-11-08 08:19:18 2018-11-08 08:20:34
## 5 94bfb9b3-80c2-44e7-8869-c29a0007bbe0 2018-11-08 07:45:41 2018-11-08 07:46:55
##   duration
##      <dbl>
## 1     81.5
## 2     80.2
## 3     77.6
## 4     75.1
## 5     74.0
## # ... with 45 more rows
```

The hostnames of the virtual machines with the longest Render durations (over 74 seconds) are:

1. 0d56a730076643d585f77e00d2d8521a00000I with 81.51 seconds;
2. 4a79b6d2616049edbf06c6aa58ab426a000003 with 80.16 seconds;
3. 95b4ae6d890e4c46986d91d7ac4bf08200000G with 77.59 seconds;
4. 6139a35676de44d6b61ec247f0ed865700000I with 75.12 seconds; and
5. 5903af3699134795af7eafc605ae5fc700000U with 74.02 seconds.

Further exploration related to this data goal can be viewed in EDA1.

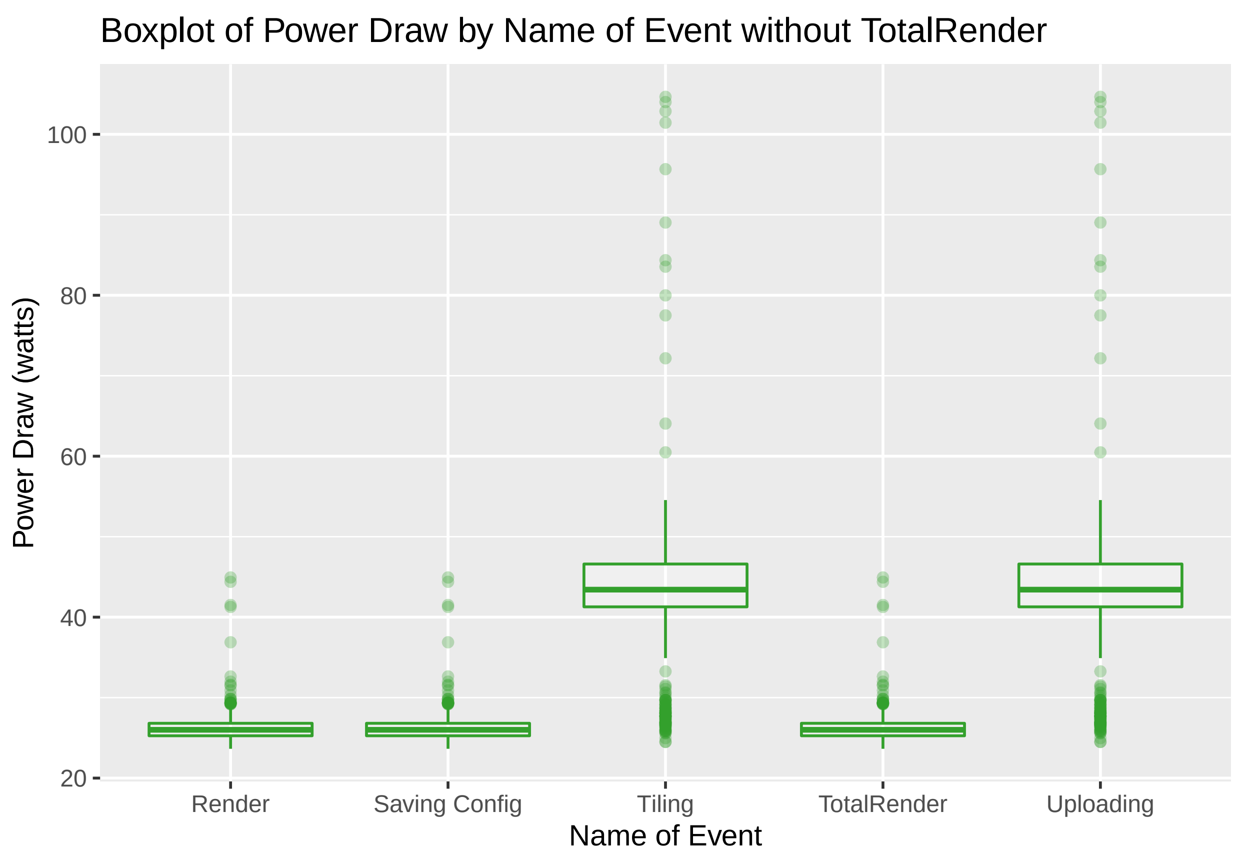
Goal 2 (Power draw)

To identify the power needed for duration time, the application.checkpoints and gpu data set needed to be combined. This could be achieved by either timestamp or hostname as the common values across both data sets. Joining by hostname was unlikely to provide us with the desired result as hostnames appeared more than once in both data sets.

The code chunk below sets out how the two data sets were joined by timestamp.

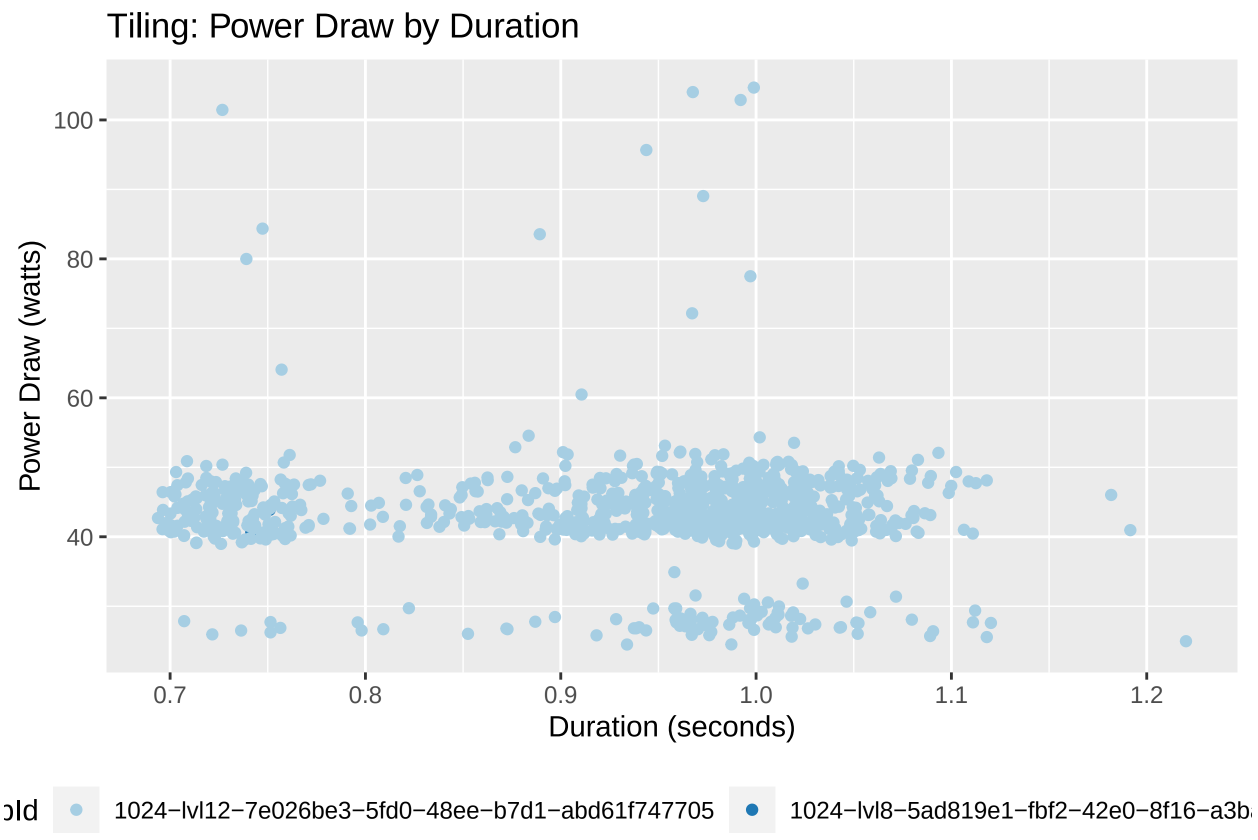
```
# Create a single data set with power usage and duration for every event other than total render
BEGIN <- as_datetime("2018-11-0807:41:27.242", tz = "Europe/London")
END <- as_datetime("2018-11-0807:42:27.242", tz = "Europe/London")

AC9 <- AC2 %>%
  filter(START >= BEGIN, START <= END) %>% # Select start times between BEGIN and END
  mutate(timestamp = round_date(START, unit = "2 seconds")) # Create new timestamp column where start t

GPU2 <- GPU1 %>%
  filter(timestamp >= BEGIN, timestamp <= END) %>% # Select timestamp that is between BEGIN and END
  mutate(timestamp = round_date(timestamp, unit = "2 seconds")) # Alter the timestamp column so that it

Power_Duration2 <- full_join(AC9, GPU2, by = c("timestamp", "hostname")) %>% # Join AC8 and GPU2
  na.omit() %>% # Omit rows with NA
  group_by(hostname) %>% # Group by hostname
  arrange(timestamp, .by_group = TRUE) # Arrange by timestamp
```

Initially, a boxplot was created of power draw by event name to see if there was an obvious correlation between duration and power draw.



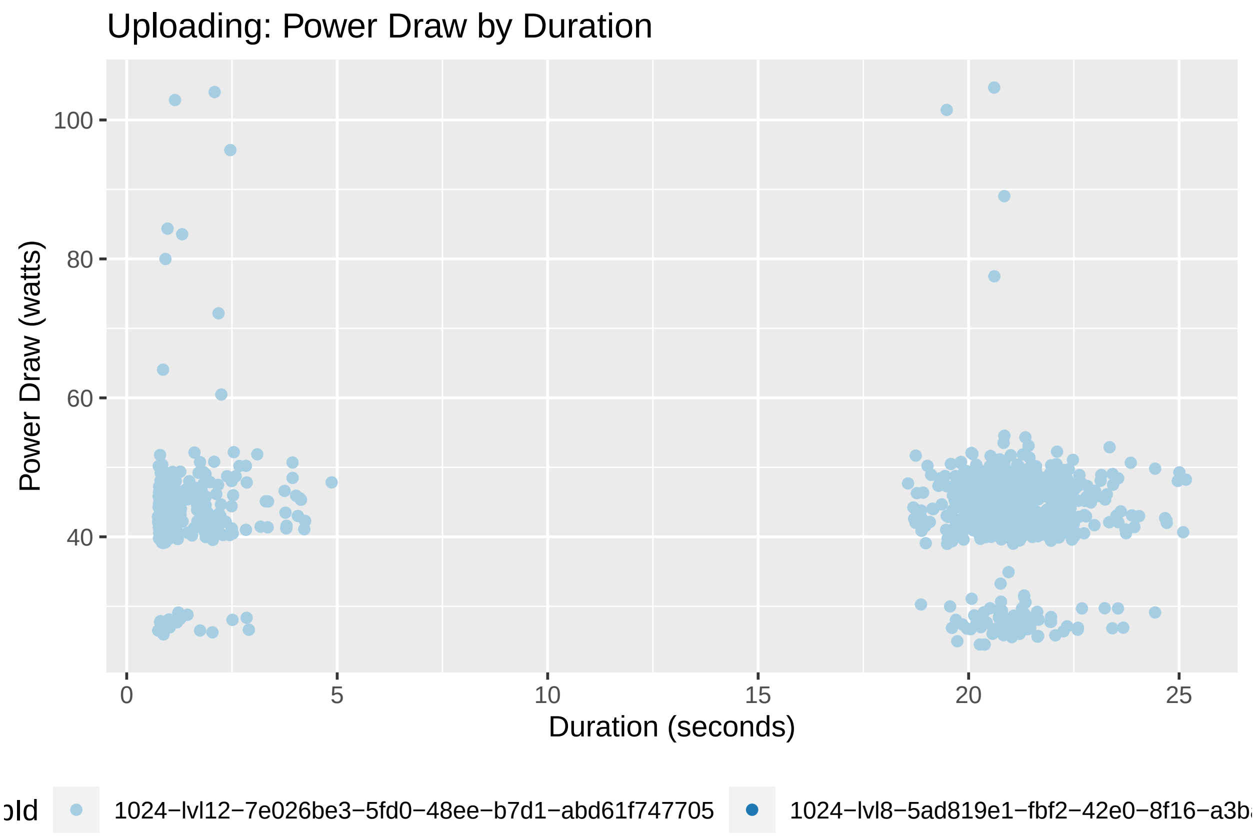
Interestingly, the events with the highest duration (TotalRender and Render) do not have the largest power draw. Tiling and Upload seem to consume the most power.

The combined data set was then filtered to see the power draw by duration for 'Tiling.' This was graphed in a scatter plot, coloured by 'jobId,' below:



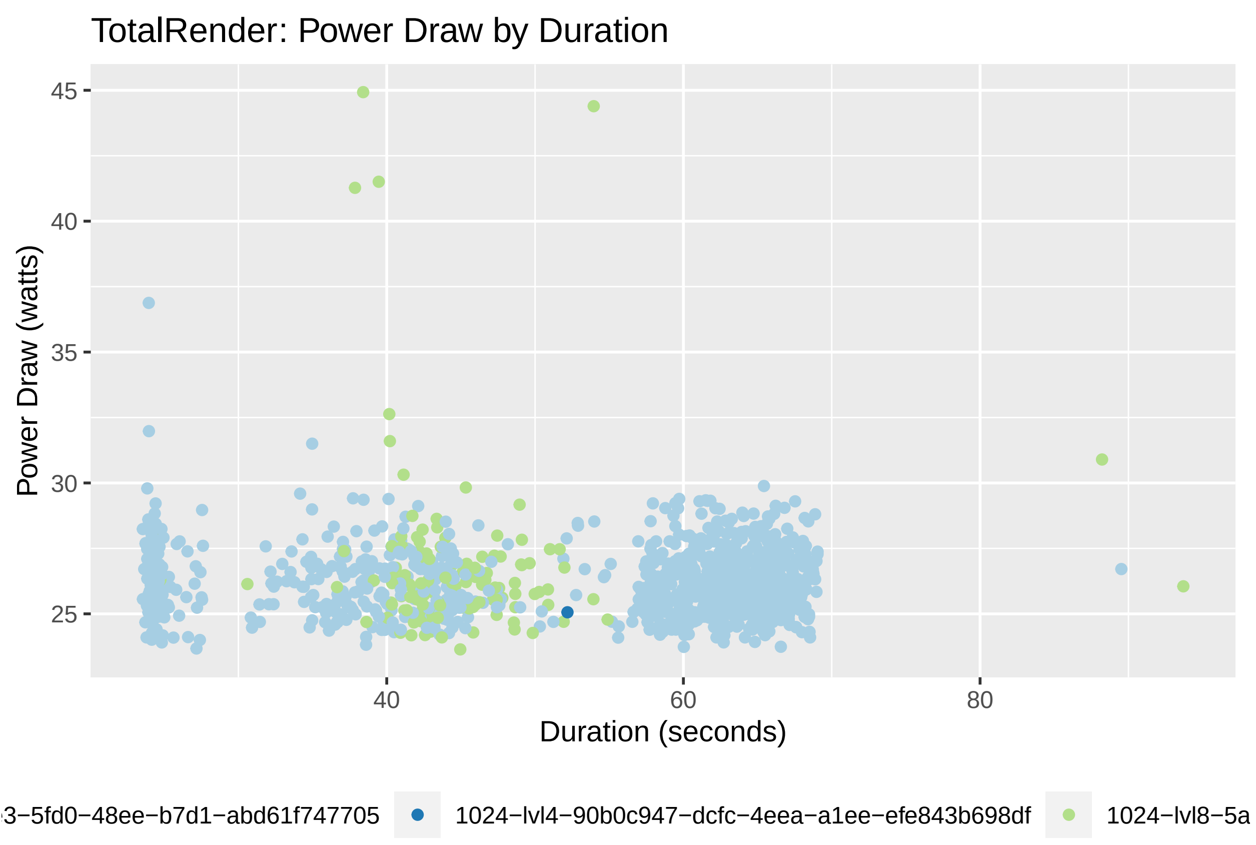
Given that we know there are 2 jobIds, it is interesting to observe that most of the tiling occurs for the jobId of level 12 (1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705). It is unclear if this is a problem with the data wrangling, or a true result. Further investigation is required.

The combined data set was then filtered to see the power draw by duration for 'Uploading.' This was graphed in a scatter plot, coloured by 'jobId,' below:



The same pattern is observed in uploading as for tiling, where there is only power draw and duration data for 2 jobIds (level 8 and 12). Interestingly, here there are also two distinct groups for duration. This would benefit from further exploration.

The combined data set was then filtered to see the power draw by duration for 'TotalRender.' This was graphed in a scatter plot, coloured by 'jobId,' below:



There are 3 jobIds visible in the scatter plot for TotalRender, although data for level 4 (1024-lvl4-90b0c947-dcfc-4eea-a1ee-efe843b698df) only appears once. It also seems that there are 3 groups for duration.

In the scatter plots above, however, there is not an obvious correlation between power draw and duration. In the tiling and uploading scatter plots it seems that there is some sort of OFF/ON division in the data, as there is an observable gap in the power draw. There also seems to be some sort of limitation on the virtual machine, perhaps specific virtual machines are tasked with tasks that are more intensive as there appears to be three separate groups by duration, not necessarily by jobId (visualisation level).

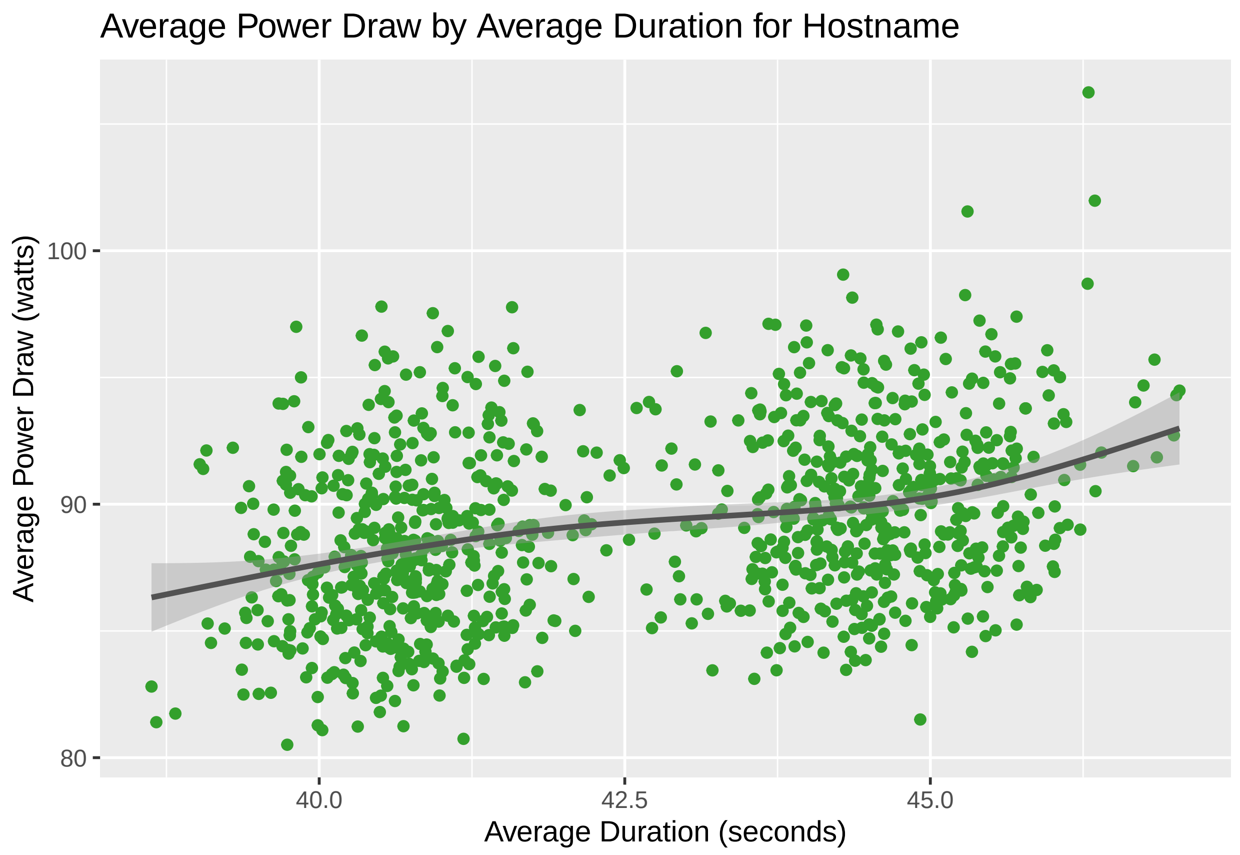
To better understand whether there is any observable pattern, the mean, standard deviation and coefficient of variation for power draw and duration was calculated in the code chunk below.

These values are combined into a single data set and visualised with the scatter plot below.

```
## ‘geom_smooth()‘ using method = ’gam’ and formula ’y ~ s(x, bs = "cs")’
```

```
## Warning: Removed 4 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```



It does appear that there are two clear groups of virtual machines (hostname), but it is unclear why.

Goal 3 (Performance)

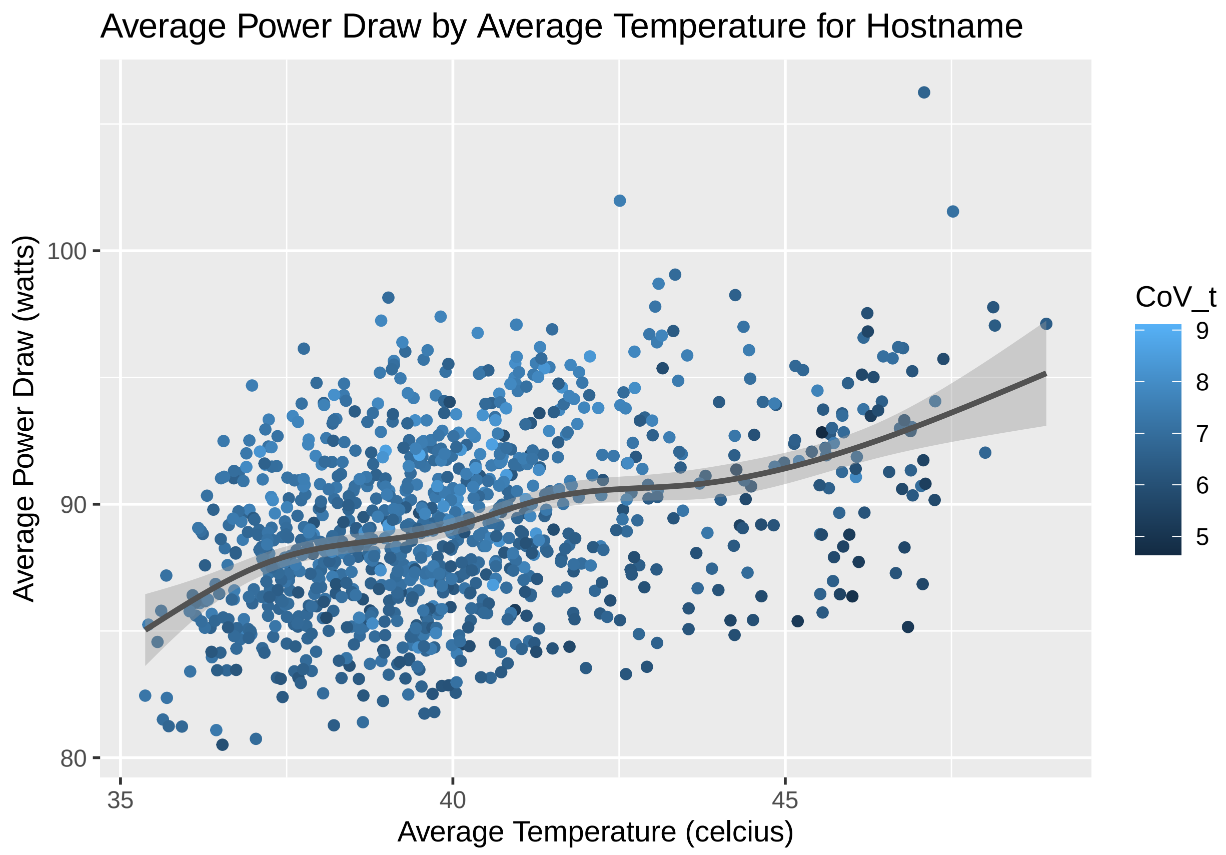
To understand the relationship between GPU power draw and performance, the impact of GPU temperature, core utilisation and memory demand on GPU power needs to be determined.

First, the mean, standard deviation and coefficient of variation are calculated.

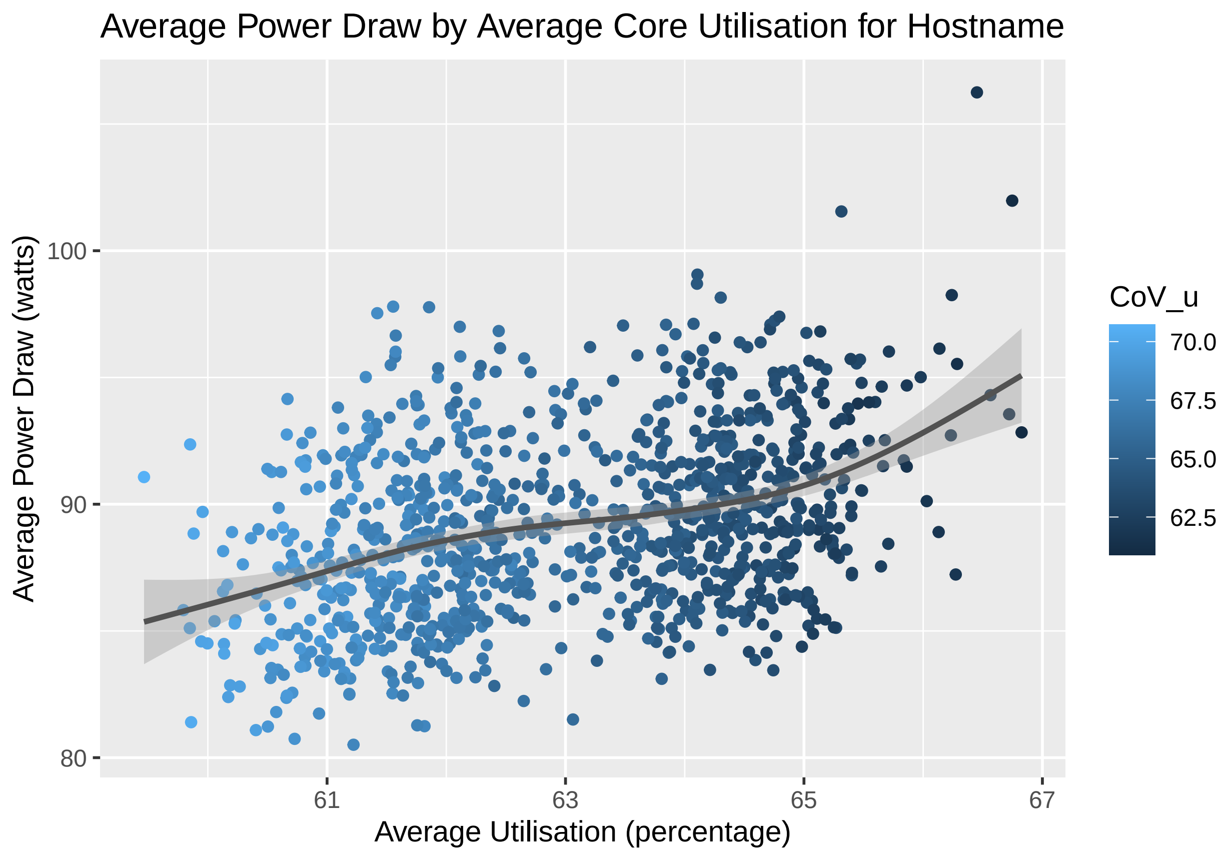
```
# What is the mean, standard deviation and coefficient of variation for all GPU metrics by host
Performance <- GPU1 %>%
  group_by(hostname) %>% # Group by hostname
  summarise(av_power = mean(powerDrawWatt), sd_power = sd(powerDrawWatt), av_temp = mean(gpuTempC), sd_
  mutate(CoV_p = (sd_power/av_power)*100, CoV_t = (sd_temp/av_temp)*100, CoV_u = (sd_util/av_util)*100,
```

Can I turn the co-effieinct of variation into a percentage?

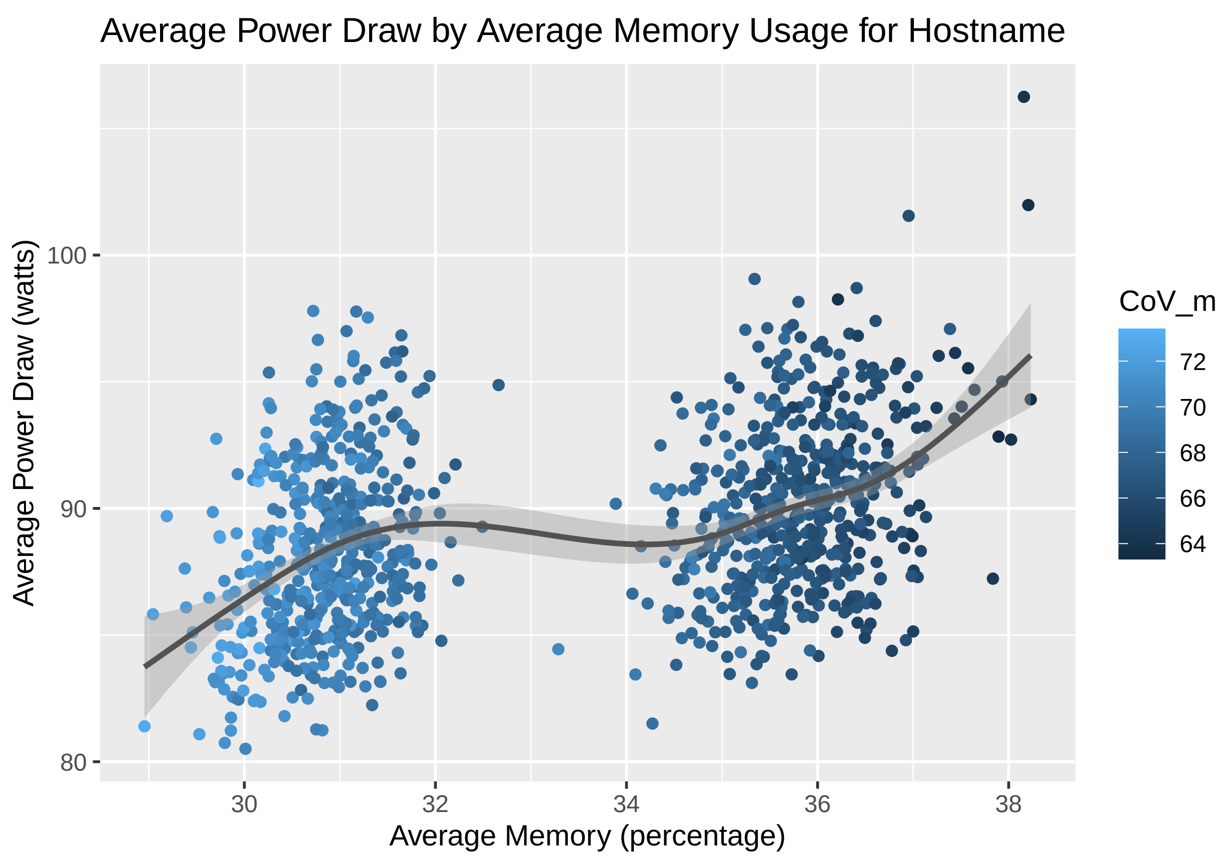
```
## ‘geom_smooth()‘ using method = ’gam’ and formula ’y ~ s(x, bs = "cs")’
```



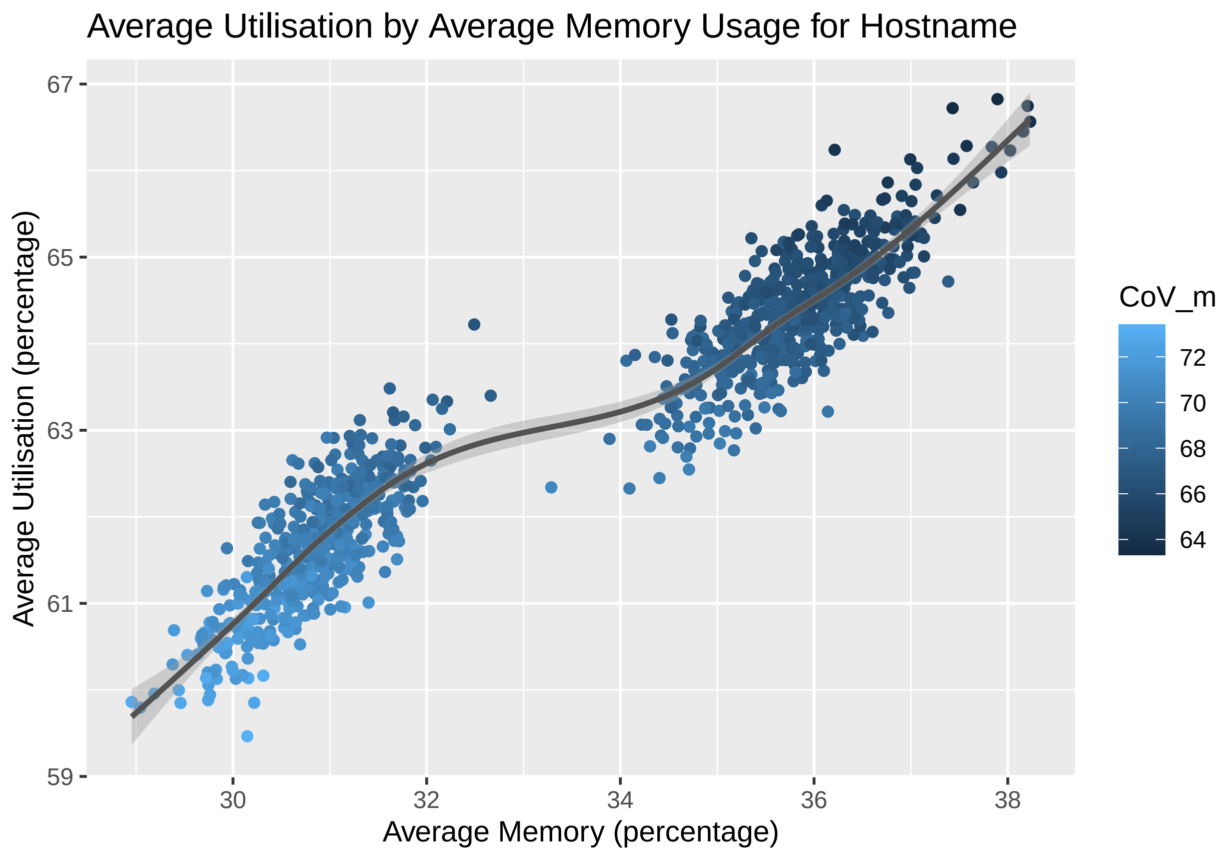
```
## ‘geom_smooth()‘ using method = ’gam’ and formula ’y ~ s(x, bs = "cs")’
```



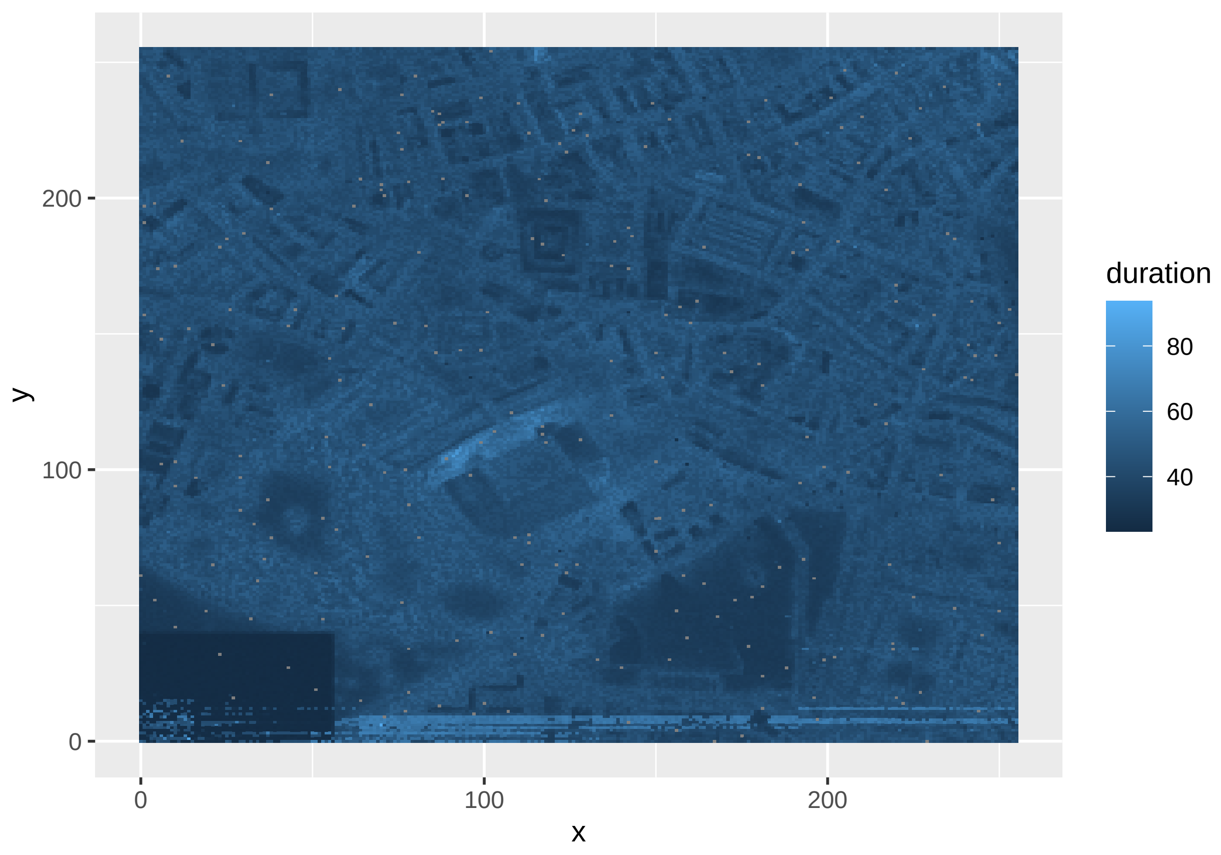
```
## ‘geom_smooth()‘ using method = ’gam’ and formula ’y ~ s(x, bs = "cs")’
```



```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



Goal 4 (Tile computation)



Goal 5 (GPU serials)

Goal 6 (Tasking Efficiency)

Evaluation - How successful has it been? Provide evidence sing appropriate evaluation methodologies, and comment on the strengths/weaknesses of your evidence in answering this question.

- What are the future implications for work in this area? If applicable, which areas of extension work are now possible due to the foundational work you have performed in this project?
- execution time if often the best metric for accessing comuter performance (Hoefler)

Reflection

- A brief relfection on your personal and professional learning in undertaking this project. Here you can comment on how you found the process, what you learned about the technologies and methodologies you used, which aspects you found most difficult/straightforward, and any conclusions which will inform the way you undertake similar projects in future.

Project Plan

(reference approaches to understanding this - which data will be prioritised, and mention iterations.)

Costs and benefits

This not a resource intensive project (as outlined by the resource invitory below). If the insights generated by this extended technical project are significant, there could be benefits for both Newcastle University and the Newcastle Urban Observatory. For example, the analysis and findings could provide further evidence, or indeed highlight new evidence, to support Newcastle University in delivering its objective to evaluate cloud supercomputing for compute intensive visualisation applications.

Terminology

The terminology used throughout this report includes:

- R, which is a programming language for statistical computing and graphics.
- RStudio, which is an Interated Development Environment for R.
- R packages, which are extensions to the R statistcial programming language. R packages contain code, data, and documentation in a standardised collection format that can be installed by users of R. The R packages used by this report are listed in the section below.
- Git, which is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code. GitHub is a provider of internet hosting for software development and version control using Git. Git Bash is an application for Microsoft Windows environments which provides an emulation layer for a Git command line experience.
- Tibble, which is a data frame that stores data in R and appears as a table.

(is there anything else I use?)

Inventory of resources

This project has been undertaken by one part-time person, on a Dell G7 17" laptop. Git, RStudio and a selection of R packages (detailed in the next section) have been used to deliver the project. Newcastle University also provided monitoring data from its cloud supercomputer architecture.

Initial assessment of tools and techniques

This project utilises the following tools and techniques:

- CRISP-DM (Chapman et. al. and Wirth (2000)) is the methodology used to deliver data workflow best practice;
- Tidyverse for R is the R package used for the analysis;
- ggplot2 for R is the R package used for visualisations;
- RColorBrewer is the R package used to provide accessible colour palettes for visualisations;
- ProjectTemplate is the R package used to improve the reproducibility of this analysis;
- Git (specifically GitHub and GitBash) is used for version control;
- all written documentation is created with the R package, RMarkdown; and
- natbib is an R package used for the bibliography

Bibliography

"Details of Grant." n.d. https://gow.epsrc.ukri.org/NGBOViewGrant.aspx?GrantRef=EP/P016782/1.

"Efficiency." n.d. https://www.google.co.uk/about/datacenters/efficiency.

Forshaw, Matthew. 2021. *Summary.* https://github.com/NewcastleDataScience/StudentProjects202122/blob/master/TeraScope/Summary.md.

Holiman et. al, Manu Antony, Nicolas S. Holliman. 2019. "Petascale Cloud Supercomputing for Terapixel Visualization of a Digital Twin." *CoRR* abs/1902.04820. http://arxiv.org/abs/1902.04820.

Kantor, Alice. 2021. "Big Tech Races to Clean up Act as Cloud Energy Use Grows," May. https://www.ft.com/content/c719f655-149c-4ce0-a7a5-18527c7776cf.

Newcastle University, Proffessor P James on behalf of. 2021. "Written Evidence Submitted by Newcastle University (Evp0115)." https://committees.parliament.uk/writtenevidence/22873/html/.

Observatory, Newcastle Urban. n.d. "Our Urban Observatory." https://www.ncl.ac.uk/who-we-are/vision/urban-observatory/.

P James, J Jonczyk, RJ Dawson. 2016. "The UK Urban Observatory Programme - Newcastle: Towards Slightly Less Dumb Cities." https://huckg.is/gisruk2017/GISRUK_2017_paper_15.pdf.

Renee Obringer, Debora Maia-Silva, Benjamin Rachunok, and Kaveh Madani. 2021. "The Overlooked Environmental Footprint of Increasing Internet Use." *Resources, Conservation & Recycling* 167. https://doi.org/10.1016/j.resconrec.2020.105389.