Iris Bench Report

GPU Benchmarking for Performance and NetZero at IRIS

GPU Benchmarking for Performance and NetZero at IRIS

1. Introduction
2. Accomplishments Of Project So Far
3. Evaluating Accuracy
4. Results and Findings
5. Future Work and Expansion

1. Introduction

The IRIS GPU Bench project, led by the Science and Technology Facilities Council (STFC), aims to create an open-source GPU benchmarking suite tailored to the specific scientific needs of the IRIS community. Supporting a range of scientific fields from particle physics to astrophysics, IRIS seeks to optimize its computational resources and reduce its carbon footprint. The IRIS GPU Bench will address the gaps in current tools by providing performance insights and CO2 emission data relevant to scientific applications such as neutron spectroscopy, tomography, and machine learning for astrophysics, thus promoting both efficiency and sustainability.

2. Accomplishments Of Project So Far

- Code Repo with the necessary to deploy and run IRIS Bench: IRIS Bench Repo
- Accompanying Documentation necessary to deploy, run, maintain, and develop IRIS Bench: IRIS Bench Docs
  - (you may need to rebuild docs by clicking “run workflow” in GitHub Actions here).
  - The documentation contains the following pages an Overview, Installation Steps, Docker Image Management, Command-Line Interface Usage, Example Commands, Result Collection, Live Monitoring of Benchmarks, Accuracy Considerations, and Future Development Tasks. This report supports the Documentation, however, the documentation is worth reviewing as well.
- Accompanying Grafana Dashboards to visualise data produced by IRIS Bench: Dashboard
- Functional Scientific/User Benchmarks (Mantid/SciML) integrated with IRIS Bench

3. Evaluating Accuracy

3.1 Carbon Metrics

- **Data Source and Frequency**: Carbon data is sourced in real-time from the National Grid ESO Regional Carbon Intensity API, with updates every approximately 30 minutes. The monitor averages values at the start and end of each period, which may limit accuracy for containers running longer than 30 minutes.
- **Variability**: Carbon forecasts fluctuate based on weather, time, and energy demand, affecting total emissions estimates. To provide a broader context, total energy can be multiplied by the UK's average carbon emission rate of 162 gCO2/kWh in 2023.

3.2 GPU Metrics

- **Source and Error Margin**: GPU metrics are obtained from `pynvml` and `nvidia-smi`, with a reported power draw error margin of ±5% (up to ±30W) (source).
- **Energy Calculation**: Energy use is estimated using trapezoidal integration of power readings. Smaller monitoring intervals improve accuracy but increase overhead, which may affect results.

3.3 Benchmark Results: Integration with Meerkat (HIGH PRIORITY)

To enhance reliability, the benchmarks should be integrated with **Meerkat** for periodic execution. This will allow consistent testing over time, enabling calculation of medians, standard deviations, and error bars added to the benchmark results, which can be calculated through **Grafana** dashboards. Regular testing helps identify trends, maintain performance consistency, and provide statistically significant results.

Being able to run the Benchmark Periodically will also show how other external factors may affect the performance of the GPUs, such as hardware location, weather temperature (ie heatwaves vs winter) throughout the year.

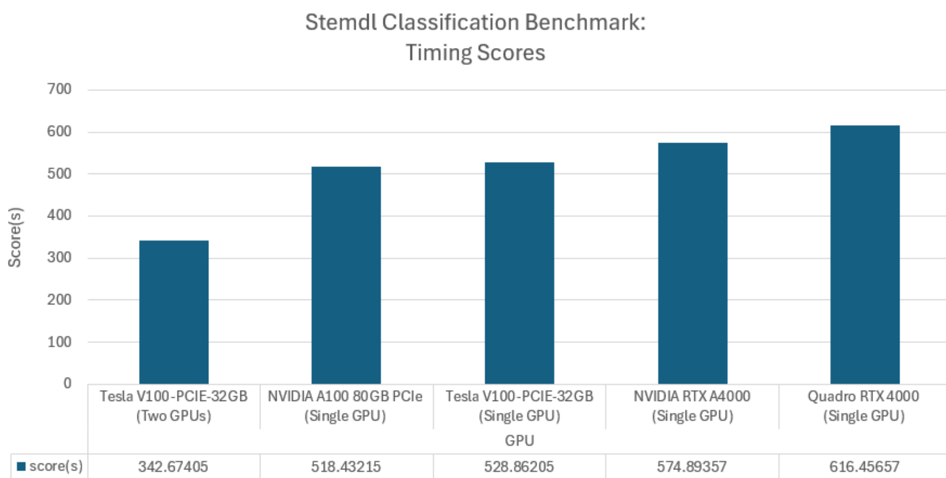
4. Results and Findings

4.1 Benchmark Results

The results below are from single runs of the benchmarks using IRIS Bench and containers. Hence, integration with Meerkat will allow for these tests to be repeated periodically (see 3.3) better validity.

Future work: review the results below to develop an understanding of GPU performance for various IRIS Workloads.

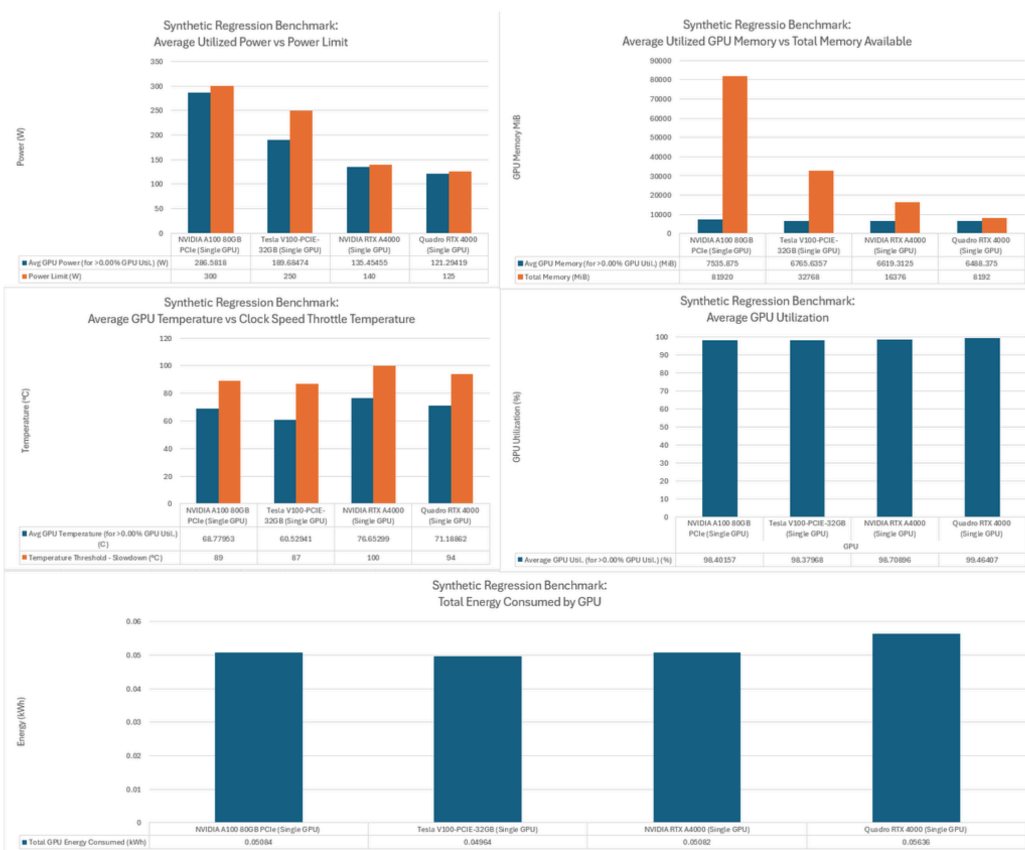
4.1.1 Sciml Bench: Stemdl Classification (Multi GPU Benchmark)



Benchmark GPU Runtime Score (s)

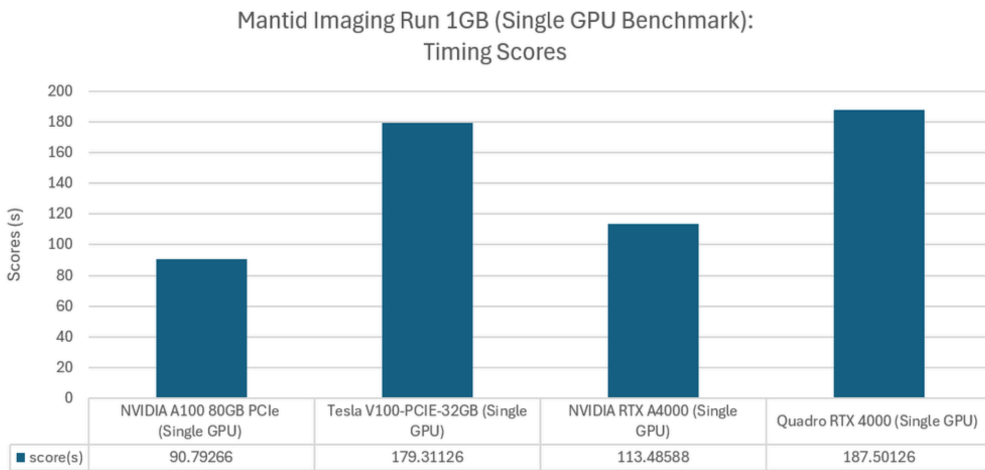
More Details on Performance

4.1.2 Sciml Bench: Synthetic Regression (Single GPU Benchmark)

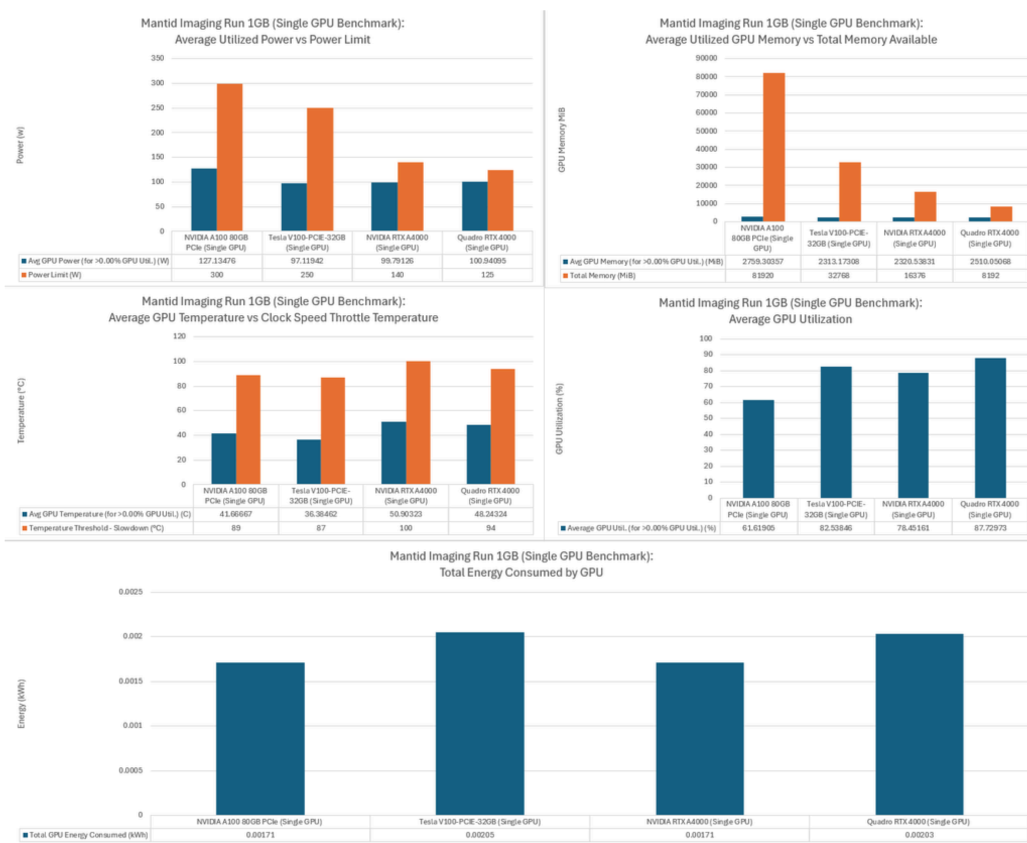


More Details on Performance

4.1.3 MANTID: Mantid Imaging Reconstruction Benchmark 1GB (Single GPU Benchmark)

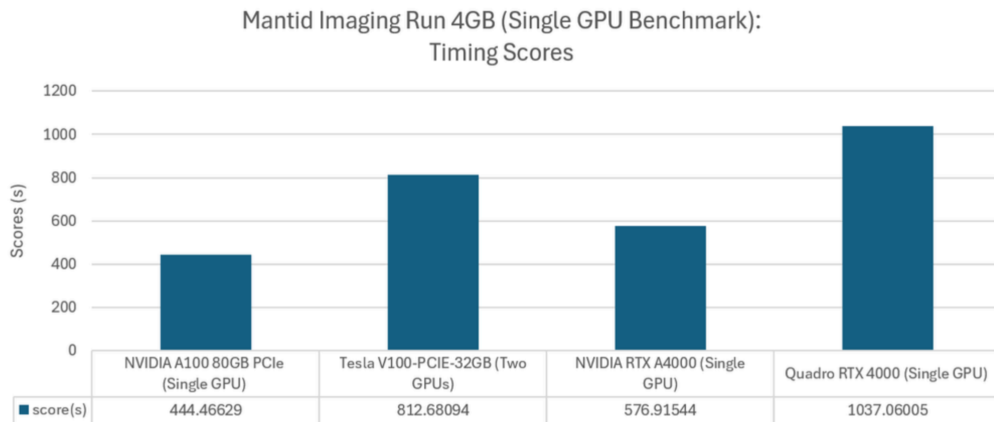


Benchmark GPU Runtime Score (s)

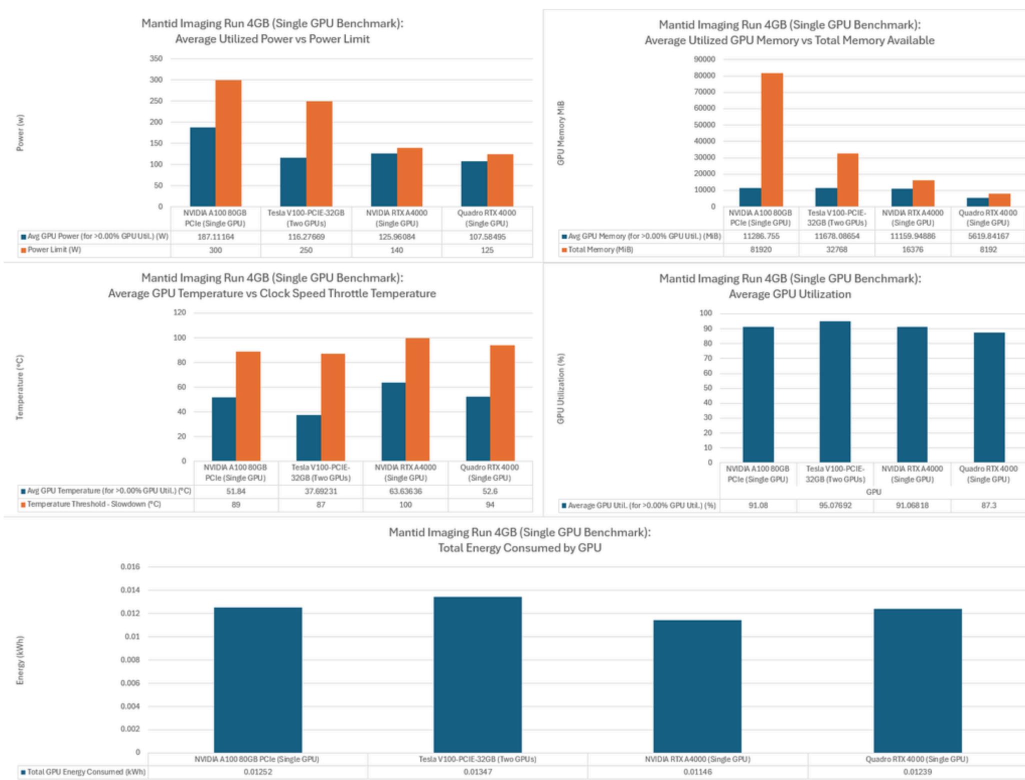


More Details on Performance

4.1.4 MANTID: Mantid Imaging Reconstruction Benchmark 4GB (Single GPU Benchmark)



Benchmark GPU Runtime Score (s)

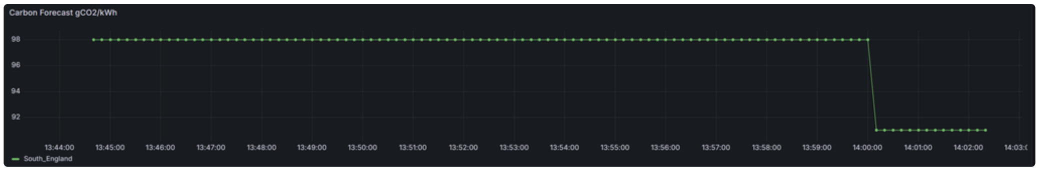


More Details on Performance

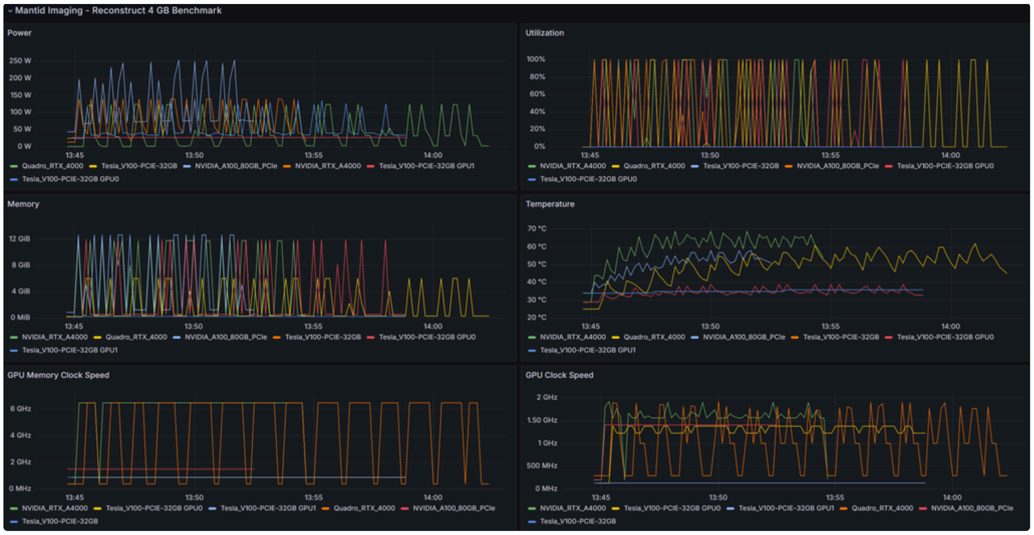
For this benchmark, a single V100 VM lacked sufficient memory, so the results are from a VM with two V100 GPUs. Consequently, the energy consumption for the dual-GPU setup is slightly higher than that of a single GPU due to the additional idle power.

4.2 Grafana Results Example

Example of Grafana Results for Mantid Imaging Benchmark Run with IRIS GPU Bench on a range of GPU.
From 2024-09-11 13:43:16 to 2024-09-11 14:03:04



Collected **Carbon Forecast** For South Of England During Benchmark Runs



Benchmark Run GPU Metrics

Benchmark Results

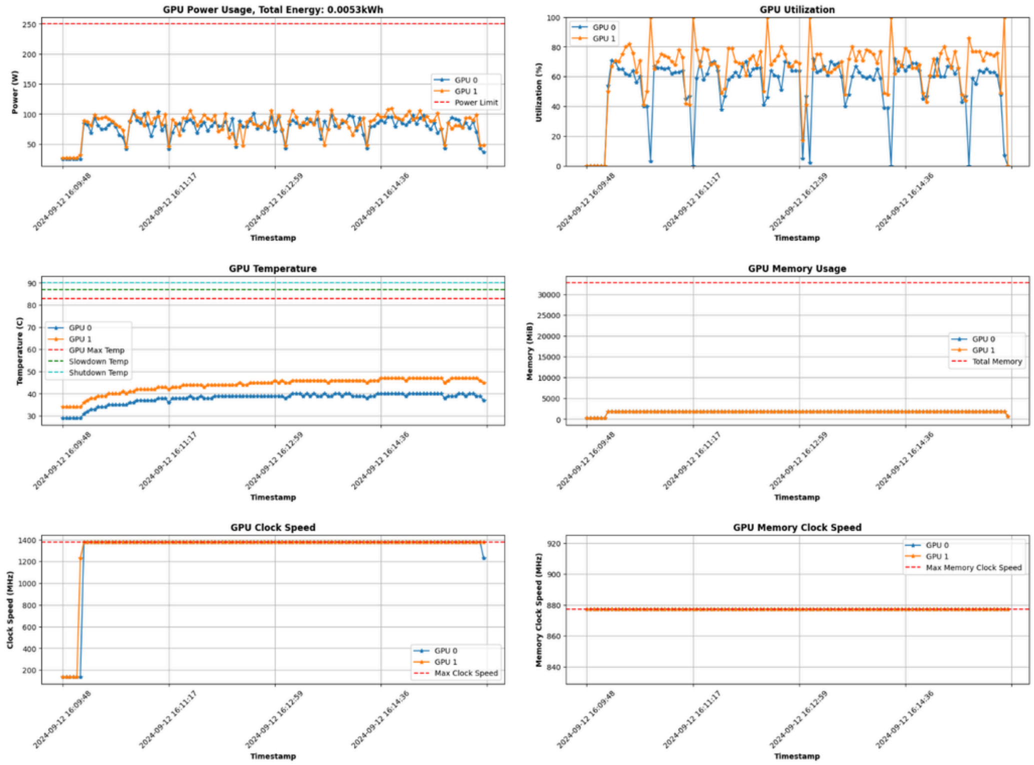
Time	GPU Type	Number of GPUs	Benchmark Score (s)	Total GPU Energy Consumed (kWh)	Total GPU Carbon Emissions (gCO2)	Avg Carbon Forecast (gCO2/kWh)	Avg GPU Memo
2024-09-11 14:02:30	Quadro_RTX_4000	1	1037.06005	0.01239	1.17106	94.50000	
2024-09-11 13:45:00	Tesla_V100-PCIE-32GB	1		0.00003	0.00340	98.00000	
2024-09-11 13:59:00	Tesla_V100-PCIE-32GB	2	812.68094	0.01347	1.32051	98.00000	
2024-09-11 13:52:30	NVIDIA_A100_80GB_PCIe	1	444.46629	0.01252	1.22660	98.00000	
2024-09-11 13:54:50	NVIDIA_RTX_A4000	1	576.91544	0.01146	1.12322	98.00000	

Grafana Table Containing Final Benchmark Results

4.3 IRIS Bench Results

When IRIS Bench is runs a benchmark on a VM the following results are generated locally (see more on this in the documentation). There are results for Benchmark Score, Energy Consumed, Carbon Emissions, Averaged GPU Metrics as well as a png figure providing more detail about the benchmark than is collected by Grafana.

The locally saved timeseries plot includes shutdown and slowdown temperatures, allowing us to better understand any performance drops if temperatures exceed these thresholds. The plot also features horizontal lines indicating total memory usage (though memory utilization might be a more relevant metric), power limits, maximum clock speed, and maximum memory clock speed. Additionally, while Grafana's minimum scrape interval is 10 seconds, shorter intervals can be selected for these locally saved plots to capture sharper results, such as when utilization fluctuates from 100% to 0% multiple times within a 10-second period.



timeseries_plot.png from stemdl_classification with two V100 GPUs

```
GPU and Carbon Performance Results

+-----------------------------------+----------------------+
| Metric                            | Value                |
+===================================+======================+
| Benchmark:                        | stemdl_classification |
+-----------------------------------+----------------------+
| Benchmark Score (s)               | 1053.29596           |
+-----------------------------------+----------------------+
| Elapsed Monitor Time (s)          | 1202.78311           |
+-----------------------------------+----------------------+
| Total GPU Energy Consumed (kWh)   | 0.00561              |
+-----------------------------------+----------------------+
| Total GPU Carbon Emissions (gCO2) | 0.54695              |
+-----------------------------------+----------------------+

Carbon Information

+------------------------------------+---------------------+
| Metric                             | Value               |
+====================================+=====================+
| Average Carbon Forecast (gCO2/kWh) | 97.5                |
+------------------------------------+---------------------+
| Carbon Forecast Start Time         | 2024-09-11 14:25:55 |
+------------------------------------+---------------------+
| Carbon Forecast End Time           | 2024-09-11 14:45:58 |
+------------------------------------+---------------------+

GPU Information

+------------------------------------------------------+-----------------------------------+
| Metric                                               | Value                             |
+======================================================+===================================+
| GPU Type                                             | Tesla V100-PCIE-32GB              |
+------------------------------------------------------+-----------------------------------+
| No. of GPUs                                          | 2                                 |
+------------------------------------------------------+-----------------------------------+
| Average GPU Utilization (for >0.00% GPU Util.) (%)   | 43.03604                          |
+------------------------------------------------------+-----------------------------------+
| Average GPU Power (for >0.00% GPU Util.) (W)         | 65.31081 (Power Limit: 250)       |
+------------------------------------------------------+-----------------------------------+
| Average GPU Temperature (for >0.00% GPU Util.) (°C)  | 38.58559                          |
+------------------------------------------------------+-----------------------------------+
| Temperature Threshold - Slowdown (°C)                | 87.00                             |
+------------------------------------------------------+-----------------------------------+
| Average GPU Memory (for >0.00% GPU Util.) (MiB)      | 1877.58896 (Total Memory: 32768.0) |
+------------------------------------------------------+-----------------------------------+
| Average Clock Speed (MHz)                            | 1340.87 (Max: 1380.00)            |
+------------------------------------------------------+-----------------------------------+
| Average Memory Clock Speed (MHz)                     | 877.00 (Max: 877.00)              |
+------------------------------------------------------+-----------------------------------+
```

formatted_metrics.txt

5. Future Work and Expansion

5.1 Additional Benchmarks to Integrate

- McStas Neutron Monte Carlo Ray-Tracing
- **RFI GPU Benchmarks**: Explore and integrate relevant Radio Frequency Interference (RFI) GPU benchmarks to cover a wider range of scientific performance evaluations.
- **Highly Optimized NVIDIA HPL Benchmarks**: Add the NVIDIA HPC Benchmarks, focusing on the HPL benchmarks designed for Grace Hopper tests. This integration will require referencing the NVIDIA HPL Benchmark documentation to ensure proper implementation and optimization.

5.2 Enhancing Benchmark Results

- **Integration into Meerkat (HIGH PRIORITY)**: Complete the integration with Meerkat to facilitate regular benchmark testing. This will allow the generation of more robust statistical data, such as error bars, by calculating mean and standard deviation across multiple tests.
- **Normalization of Results**: Implement normalization techniques to standardize benchmark results across different GPU models and workloads, facilitating meaningful comparisons.
- **FLOP Estimations and Efficiency Metrics**: Calculate Floating Point Operations per Second (FLOPs) to determine performance per watt, providing insights into computational efficiency and energy consumption.

5.2.1 Estimating Floating Point Operations Per Second (FLOPs)

Comparisons based solely on clock speed can be misleading due to differences in GPU architectures, CUDA cores, tensor cores, and operations per clock cycle. Note clock speeds vary for various reason such as size of workload, temperature, power supply.

To estimate the FLOPs of a GPU:

`FLOPs = CUDA Cores × Clock Speed (Hz) × Operations per Clock Cycle`

Steps to calculate FLOPs:

- Determine the number and type of cores (e.g., CUDA cores, tensor cores) and if they are being utilized.
- Clock speed can be found using IRIS BENCH.
- Identify the operations per clock cycle, specific to the GPU's architecture (ie core generation). "The A100 SM includes new third-generation Tensor Cores that each perform 256 FP16/FP32 FMA operations per clock"
- Investigate how `nvidia-smi` calculates utilization and whether more granular utilization data can be obtained.
- Review how tools like `sciml-bench` and pytorch report core usage and incorporate these insights into metrics collection.

Data Sheets Contain Information About the Number of Cores
V100 datasheet
A4000 Datasheet
NVIDIA A100
RTX4000 DataSheet

5.3 Explore NVIDIA Nsight Systems

Utilize NVIDIA Nsight Systems for detailed performance profiling, identifying optimization opportunities, and potentially getting insight into the activated cores for FLOP calculations. This tool offers in-depth insights into GPU performance across various workloads and configurations.

5.4 Enhancing Carbon Footprint Accuracy

To achieve a more accurate total carbon To more accurately estimate the total carbon footprint, emissions from the GPU's manufacturing, delivery, and full lifecycle should be included. This requires calculating the proportion of the GPU's lifespan used by a specific workload and converting it to equivalent carbon emissions, which are then added to emissions from the API and electricity use.

Cooling power should also be considered; while `nvidia-smi` does not report fan power directly, fan speed data can be used to estimate it.

The revised calculation includes:

- Manufacturing Emissions per Hour:
  - Manufacturing Emissions per Hour = Total Manufacturing Emissions (kg CO_2e) / Expected Lifespan (hours)
- Delivery Emissions per Hour:
  - Delivery Emissions per Hour = Total Delivery Emissions (kg CO_2e) / Expected Lifespan (hours)
- Use Emissions for the Run (already calculated by IRIS Bench):

- Use Emissions for the Run = (Power Consumption (Watts) / 1000) * Run Time (hours) * Carbon Intensity from API (kg CO_2e per kWh)

- Cooling Emissions for the Run (based on fan speed):
  - Cooling Emissions for the Run = (Estimated Fan Power (Watts) / 1000) * Run Time (hours) * Carbon Intensity from API (kg CO_2e per kWh)
- Total Emissions for the Run:
  - Total Emissions for the Run = (Manufacturing Emissions per Hour + Delivery Emissions per Hour) * Run Time (hours) + Use Emissions for the Run + Cooling Emissions for the Run

This approach will provide a more comprehensive estimate of the carbon footprint for GPU workloads.

5.4.1 Estimating Embodied Carbon if Stats aren't available from Manufacturers

Finding specific statistics on the embodied carbon emissions associated with the manufacturing and packaging of NVIDIA GPUs proved challenging. However, the paper Toward Sustainable HPC: Carbon Footprint Estimation and Environmental Implications of HPC Systems (arxiv.org) provides a model for estimating embodied carbon using available data on GPUs. For detailed insights, please refer to Section 2.1, titled "Embodied Carbon Footprint Modeling."

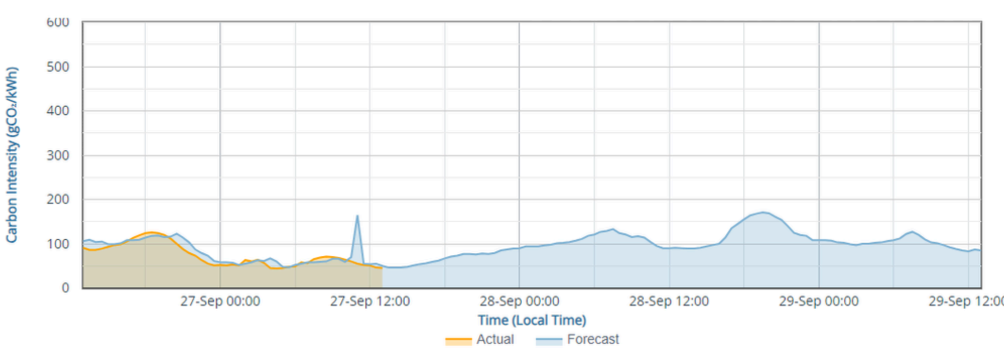
5.5 Strategic Planning for Energy-Intensive Tasks Based on Carbon Intensity Forecasts

This information is less about benchmarking but is intended to guide policies for executing energy-intensive tasks at STFC. Additionally, providing a carbon intensity forecast from the National Grid Carbon Intensity API prior to a run could assist users in determining the optimal timing for these energy-intensive operations. By scheduling such tasks during periods of low carbon intensity or in regions with lower emissions—such as comparing Daresbury and RAL—users can effectively reduce their carbon footprint. Factors such as seasonal variations (summer vs. winter) and time of day significantly impact carbon intensity, and making informed decisions based on these variables could lead to substantial reductions in carbon emissions.

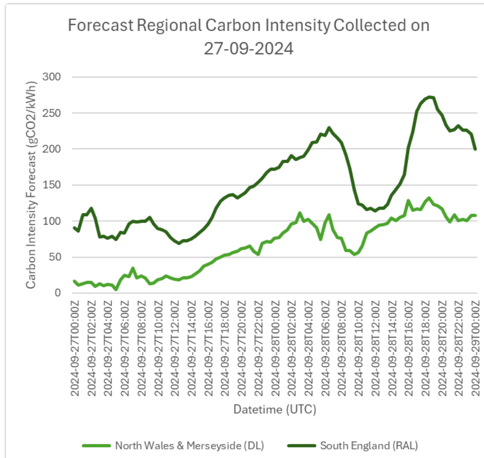
The table below, from Carbon Intensity API , illustrates the variation in carbon intensity across different regions. At the time this screenshot was captured, the carbon intensity for the Daresbury Lab region was 21 gCO2/kWh, while RAL's region recorded 76 gCO2/kWh. Running a program at the Hartree Centre instead of RAL during this period could potentially reduce carbon emissions by approximately 70%, assuming the same hardware and setup are available.

Region	Forecast Carbon Intensity (gCO_2/kWh)	Index
North Wales & Merseyside	21	very low
South England	76	low

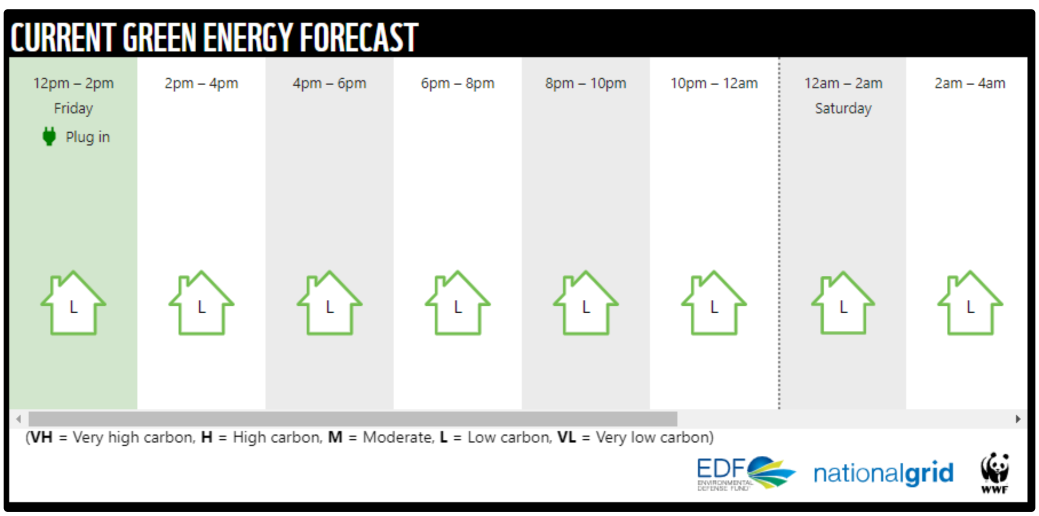
As shown in the plot below, from Carbon Intensity API, actual emissions closely align with the forecasted values. Running an energy-intensive task between September 27 at 00:00 and 12:00, rather than from September 28 at 00:00 to 12:00, could result in an approximately 50% reduction in carbon usage.



Connecting this back to the benchmarks—IRIS Bench provides users with insights into the duration and energy consumption of workloads similar to theirs on the available GPUs. When combined with carbon intensity forecasts, which can be gathered regionally before the intended time of execution, we can estimate the carbon emissions for a given workload before it is even run. This predictive capability would be an invaluable addition to IRIS Bench, allowing users to make informed decisions about the environmental impact of their tasks. The example below, taken from the API on 27-09-2024, provides a carbon forecast through 29-09-2024 and demonstrates the ability to select specific regions for more tailored predictions.



WWF also has integrated the API above into a reusable widget designed to help you reduce emissions by turning on devices during periods of green energy and turning them off when it's not. STFC Cloud could create a similar solution for its users to enhance their carbon emission management.



Screen print from WWF-UK's Green Energy Forecast | WWF

5.6 Additional Metrics and Areas for Measurement in IRIS Bench

- **Utilization Time**: Measure the total time the GPU is actively utilized, which can provide insights into idle periods and workload efficiency.
- **Stability: Crash Frequency**: Track and report any GPU crashes or visual artifacts during benchmarks to assess stability.
- **Throttling Events**: Monitor instances of clock speed reductions due to high temperatures or power constraints.
- **Memory Bandwidth**: Measure the data transfer rate between the GPU and system memory to identify potential bottlenecks and optimize performance

5.7 Other Ideas for Future Implementation

- **Consistent Hardware Configurations**: Ensure that all GPUs being tested use the same hardware configurations (e.g., memory, CPUs) to eliminate variability and produce consistent results.
- **Continuous Integration for Performance Testing**: Encourage IRIS users to integrate GPU benchmarks into their CI workflows. Implement automated performance tests on every pull request; if performance drops by a specified percentage, the pull request would fail, ensuring that code changes do not degrade performance.
- **Experimenting with Precision to Utilize Tensor Cores Fully**: For GPUs equipped with tensor cores, utilize lower precisions (e.g., FP16) for matrix operations where feasible. This can lead to significant performance gains, depending on the workload's precision requirements.

5.8 Improvements for GitHub Repository

- **Continuous Integration (CI) Tests**: Develop and integrate comprehensive CI tests using GitHub Actions to maintain code reliability, ensure consistent performance, and catch issues early in the development cycle.
- **Carbon Index Calculation**: Enhance the environmental impact analysis by calculating the carbon index throughout the entire benchmarking run, rather than just at the start and end, to provide a more accurate representation.
- **Use Best Practices for Naming Dockerfiles**: Ensure all Dockerfiles follow standard naming conventions for clarity and maintainability.
- **Include Logging Levels**: Implement various logging levels (e.g., debug, info, error) and log tagging to improve traceability and debugging.
- **Add Shell Check Workflow**: Integrate a shell check workflow, similar to the one used in the SCD-OpenStack-Utils repository, to catch errors in shell scripts.
- **Run Shell Check from Bash Scripts**: Use shell check (similar to pylint) to analyze bash scripts for potential issues and maintain code quality.
- **Add Dependabot to GitHub Actions**: Implement Dependabot for automated dependency updates, improving security and ensuring compatibility with new releases.