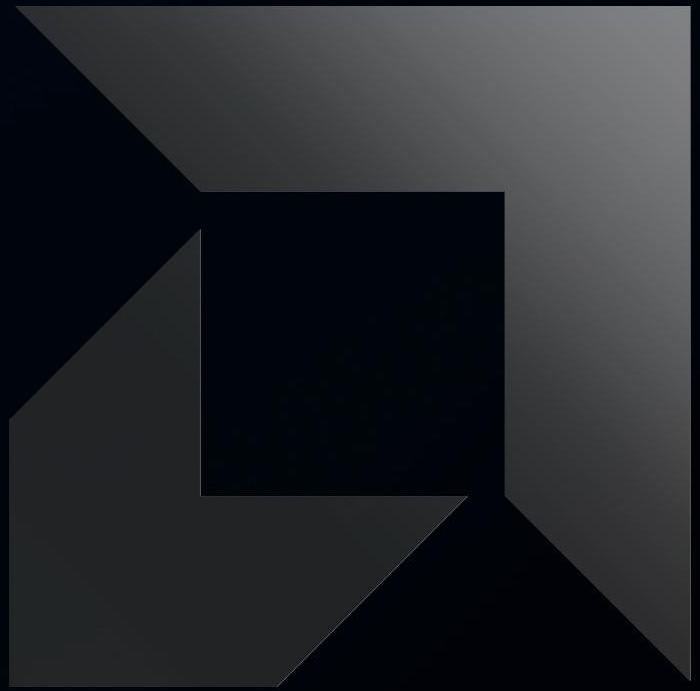


Pytorch Performance on AMD Radeon and Instinct GPUs

Dr. Joe Schoonover (Fluid Numerics)
Garrett Byrd (Fluid Numerics)

Special thanks to collaborators:
Dr. William K. Dewar (FSU)
Dr. Andrew Poje (CUNY)
Dr. Bruno Deremble (CNRS)

AMD
together we advance_

This is not a talk about AI

AMD
together we advance_

But if you use pytorch, you might find this interesting

AMD
together we advance_

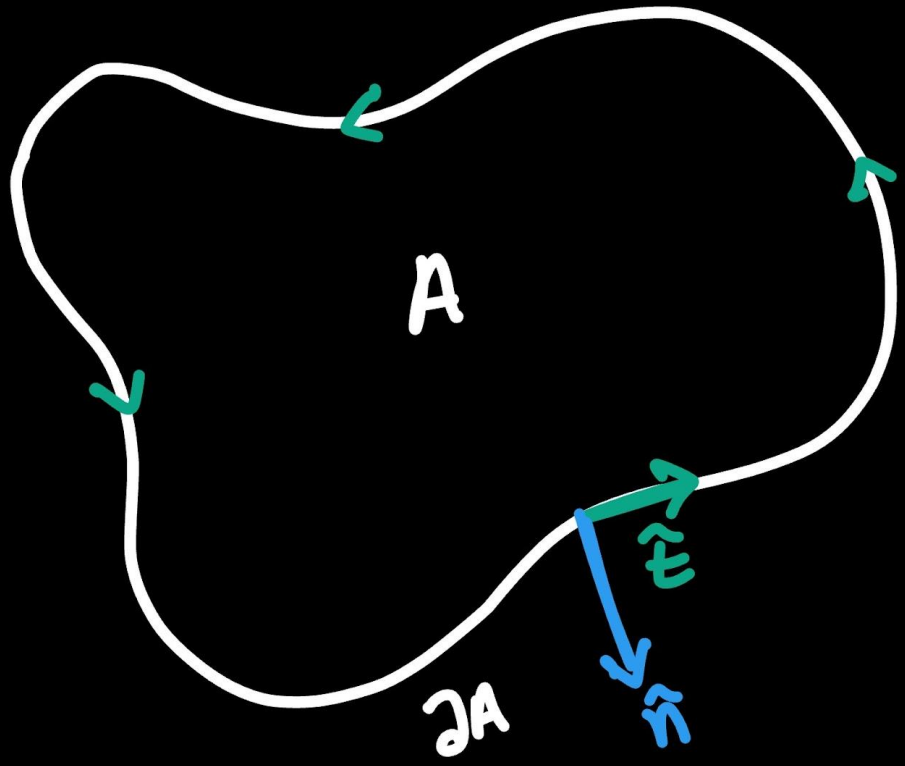
We are interested in calculating kinetic energy spectra in irregular geometry

AMD
together we advance_

Motivating Problem

AMD
together we advance_

Motivating Problem



Helmholtz Decomposition

$$\vec{u} = \hat{z} \times \nabla\psi + \nabla\chi$$

Rotational Part

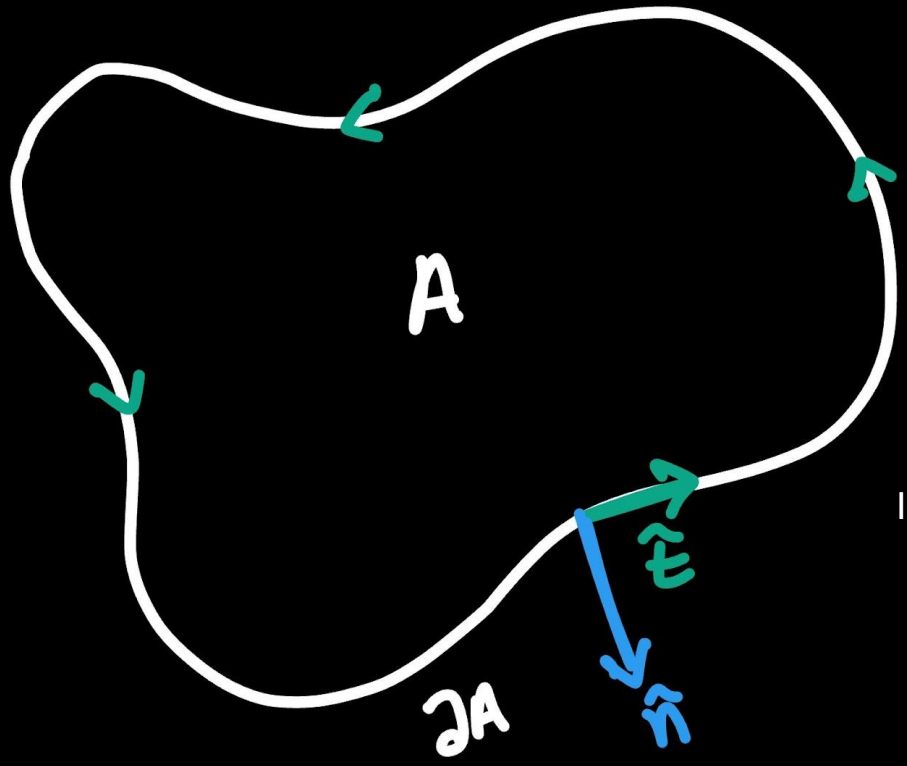
$$\nabla \cdot (\hat{z} \times \vec{u}) = \zeta = \nabla^2 \psi$$

Divergent Part

$$\nabla \circ \vec{u} = d = \nabla^2 \chi$$

AMD
together we advance_

Motivating Problem



We write the potentials in terms of the eigenmodes of the Laplacian operator

$$\begin{cases} \nabla^2 \xi_n = -\lambda_n^2 \xi_n \\ \xi_n|_{\partial A} = 0 \end{cases}$$

$$\psi = \sum_{n=0}^{\infty} \hat{\psi}_n \xi_n$$

$$\begin{cases} \nabla^2 \eta_n = -\sigma_n^2 \eta_n \\ \nabla \eta_n \cdot \hat{n}|_{\partial A} = 0 \end{cases}$$

$$\chi = \sum_{n=0}^{\infty} \hat{\chi}_n \eta_n$$

In general, we must find the eigenmodes numerically.

In our case, we use Pytorch to implement a matrix-free eigenpair solver.

AMD
together we advance_

Motivating Problem

We want to find eigenvalues and eigenfunctions of a Laplacian operator

Built-in methods with Pytorch like `torch.eig` and `torch.eigh` are not appropriate for our problem

- We have functions that implement the action of the Laplacian and Laplacian inverse
  - torch methods require we construct a matrix representation of the operator
- We don't want to waste time reshaping arrays
  - torch methods require data stored as column vectors
- For our problem sizes, returning all eigenvectors is impractical due to memory requirements for doing so.
  - torch methods are meant to return all eigenvalues and eigenvectors.
  - For a 500x500 grid, at single precision, we would need 250 GB memory to store just the eigenvectors. The memory footprint for exact eigenvalue decomposition would be ~ 1TB memory

AMD
together we advance_

Algorithm Sketch

Implicitly Restarted Lanczos Method

$$AV = VH + \vec{r}_m \hat{e}_m^T$$

[N x N][N x m] [N x m][m x m]

- A is a symmetric matrix
- V is an orthonormal matrix
- H is a tridiagonal symmetric matrix that shares eigenvalues with A (called Ritz values)

V and H are found by generating an orthonormal basis of A through a truncated Gram-Schmidt algorithm

AMD
together we advance_

Algorithm Sketch

Implicitly Restarted Lanczos Method

$$\mathbb{A}\mathbb{V} = \mathbb{V}\mathbb{H}$$

When $\mathbb{A}$ is the matrix representing the discretization of a Laplacian…

- Finding largest eigenmodes of $\mathbb{A}$ gives smallest length scales
- Finding largest eigenmodes of $\mathbb{A}^{-1}$ gives largest length scales (smallest eigenvalues of $\mathbb{A}$)
- Finding largest eigenmodes of $(\mathbb{A} - \sigma\mathbb{I})^{-1}$ gives modes closest to σ

AMD
together we advance_

Matrix-free IRLM implementation with Pytorch

We wrote a simple IRLM method that accepts a `matrixaction` method as an argument

```
def implicitly_restarted_lanczos(matrixaction, x, neigs, nkrylov, tol=1e-12, max_iter=100, arr_kwargs = {'dtype':torch.float64, 'device':'cpu'}):
    """Implicitly Restarted Lanczos Method (IRLM)
    This method applies the Implicitly Restarted Lanczos Method (IRLM) using a matrix-free
    method for applying the matrix action. Implicit restarts are applied by filtering out
    the eigenmodes in the Kryolv space associated with the smallest eigenvalues.

    Arguments

      matrixaction - A method that takes in data stored in a torch array of x.shape and returns a torch array
                     of shape x.shape. This method is assumed to a linear operator whose matrix-form equivalent
                     is a symmtric matrix

      x            - A seed vector for the IRLM

      neigs        - The number of eigenvalues/eigenmodes that you want to find

      nkrylov      - The size of the Krylov vector search space. Must be larger than neigs
```

github.com/fluidnumerics/FluidSpectraIG

AMD
together we advance_

Matrix-free IRLM implementation with Pytorch

For our test problems, our `matrixaction` method comes from **github.com/louity/MQGeometry**

```
def calculate_dirichlet_modes(self, tol=1e-12, max_iter=100):
    """Uses IRLM to calcualte the N smallest eigenmodes, corresponding to the
    largest length scales, of the Laplacian operator with homogeneous dirichlet
    boundary conditions."""

    # create an initial guess/seed vector for IRLM
    v0 = self.xg*(self.xg-self.Lx)*self.yg*(self.yg-self.Ly)

    evals, eigenvectors, r = implicitly_restarted_lanczos(self.laplacian_g_inverse, v0,
        self.nmodes, self.nkrylov, tol=tol, max_iter=max_iter,
        arr_kwargs = self.arr_kwargs)

    eigenvalues = 1.0/evals

    return eigenvalues, eigenvectors, r
```

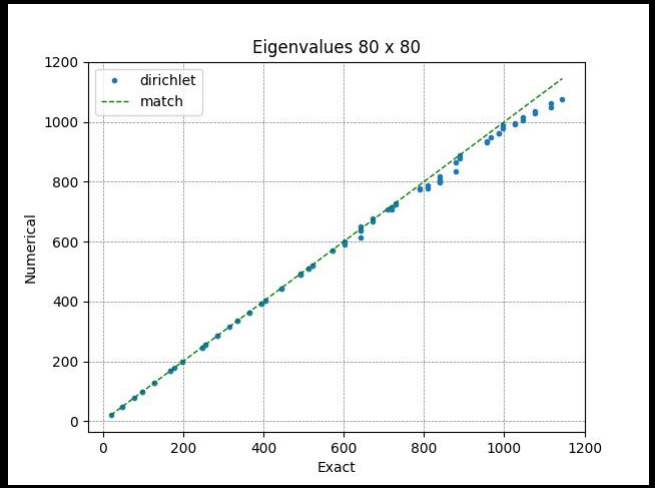
github.com/fluidnumerics/FluidSpectraIG

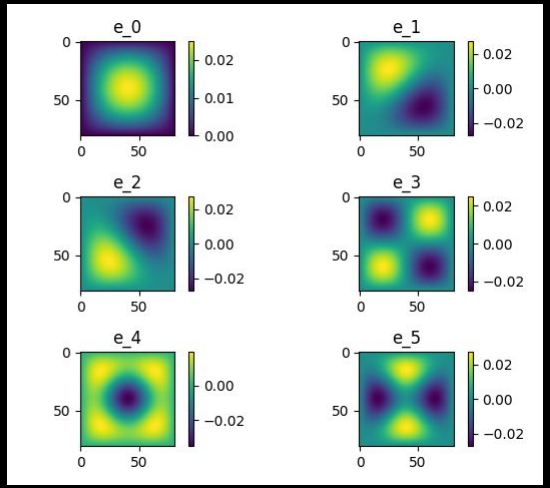
AMD
together we advance_

Test Case

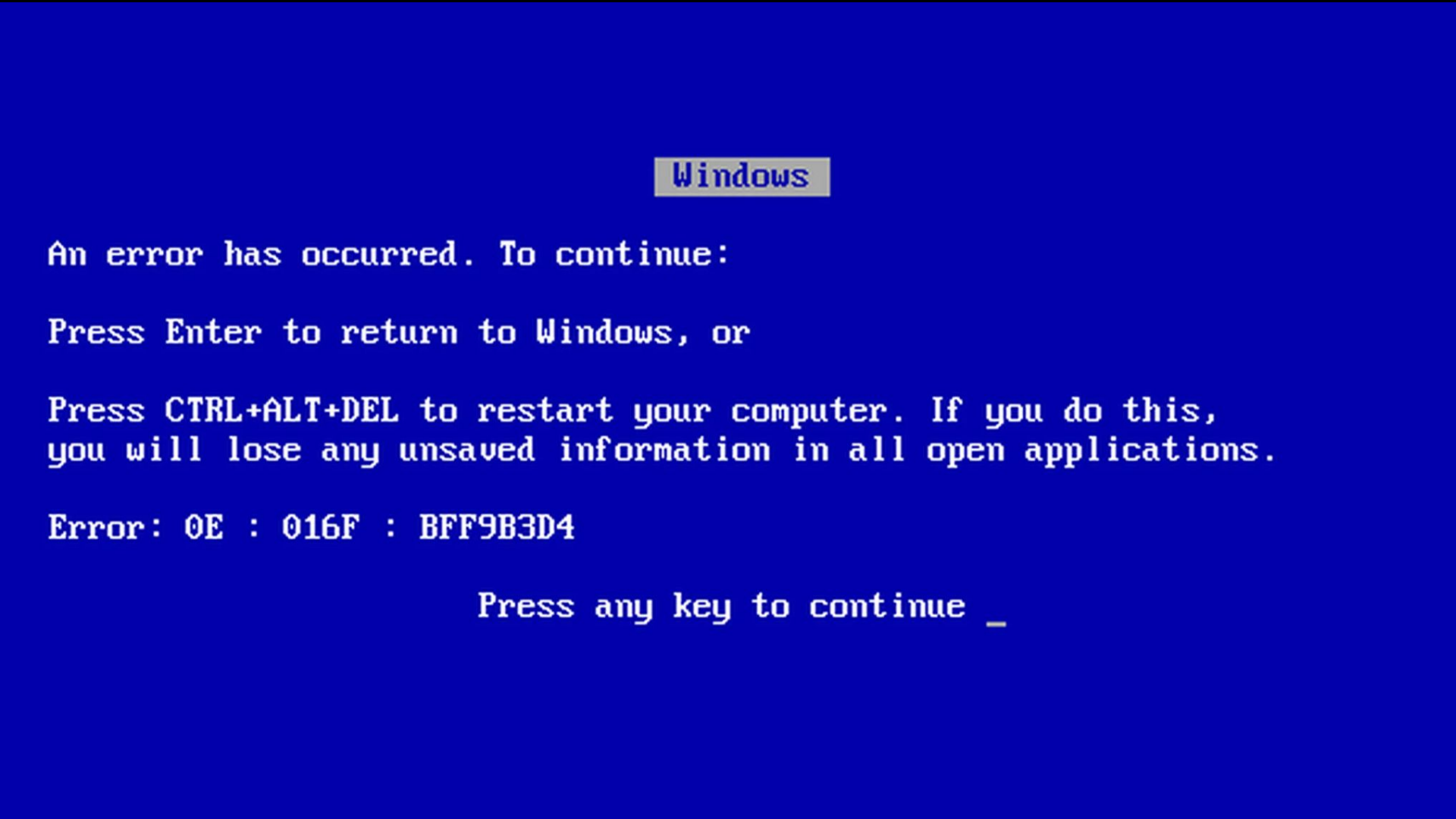
Laplacian on a square domain with Dirichlet boundary conditions

Calculate `m` eigenpairs on a `MxM` grid with a Krylov search space of `m+k`





AMD
together we advance_


Windows
An error has occurred. To continue:
Press Enter to return to Windows, or
Press CTRL+ALT+DEL to restart your computer. If you do this,
you will lose any unsaved information in all open applications.
Error: 0E : 016F : BFF9B3D4
Press any key to continue _

Set up ROCm 6.0.x and Pytorch

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Check supported operating systems at rocm.docs.amd.com

Supported operating systems

AMD ROCm™ Software supports the following Linux distributions.

Operating system	Kernel	Support
RHEL 9.3	5.14.0-362	✅
RHEL 9.2	5.14.0-362	✅
RHEL 8.9	4.18-513	✅
RHEL 8.8	4.18-513	✅
CentOS 7.9	3.10	✅
SLES 15 SP5	5.14.21-150500	✅
SLES 15 SP4	5.14.21-150500	✅
Ubuntu 22.04.4	6.6	✅ [2]
Ubuntu 22.04.3	6.2	✅
Ubuntu 22.04.2	5.19	✅
Ubuntu 20.04.6	5.15	✅
Ubuntu 20.04.5	5.15	✅

https://rocm.docs.amd.com/projects/install-on-linux/en/latest/reference/system-requirements.html *(Last accessed March 22, 2024)*

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Check your OS and Linux Kernel

```
$ cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.3 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.3 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy

$ uname -r
6.5.0-17-generic
```


AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Two installation methods for AMD GPU Drivers and ROCm

amdgpu-installer

- Provides a wrapper around the system package manager
- Handles adding repositories to system package manager, installation of amdgpu drivers and ROCm components based on use-cases
- Ideal if you're not as familiar with working with your system package manager

system package manager

- You add repositories and manually control installation of amdgpu drivers and ROCm components
- Use if you're comfortable working with the package manager directly

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Install AMD GPU drivers and ROCm using the amdgpu-installer

Ubuntu 22.04 (Jammy)

```
wget https://repo.radeon.com/amdgpu-install/6.0.2/ubuntu/jammy/amdgpu-install_6.0.60002-1_all.deb
sudo apt install ./amdgpu-install_6.0.60002-1_all.deb
sudo amdgpu-install --usecase=graphics,rocm
```

For other operating systems, see

https://rocm.docs.amd.com/projects/install-on-linux/en/latest/tutorial/quick-start.html

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Check supported GPUs at rocm.docs.amd.com

```
$ rocminfo | grep gfx
  Name:                    gfx1100
      Name:                    amdgcn-amd-amdhsa--gfx1100
```

Some GPUs may not be supported, but they might still work (e.g. AMD Radeon RX 6700 XT)

```
$ rocminfo | grep gfx
  Name:                    gfx1031
      Name:                    amdgcn-amd-amdhsa--gfx1031

$ export HSA_OVERRIDE_GFX_VERSION=10.3.0
```

Older GPUs might also be supported on older versions of ROCm

https://rocm.docs.amd.com/projects/install-on-linux/en/latest/reference/system-requirements.html *(Last accessed March 22, 2024)*

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

ask Ubuntu

Home
Questions
Tags
Users
Companies
Unanswered

TEAMS

Ask questions, find answers and collaborate at work with Stack Overflow for Teams.

Explore Teams

Create a free Team

Errors with amdgpu-dmks on Ubuntu 22.04

Asked 1 year, 3 months ago Modified 9 months ago Viewed 3k times

Microsoft Azure
Protect your customers with built-in multilayered security
Sign up ❯

0

I have been trying to install AMD GPU drivers (for the RX5500M) and have been facing multiple issues. Now, whenever I run `sudo apt-get upgrade` (or any install with `apt-get`), I recieve the following error:

```
dpkg: error processing package amdgpu-dkms (--configure):
installed amdgpu-dkms package post-installation script subprocess returned error e
Errors were encountered while processing:
amdgpu-dkms
E: Sub-process /usr/bin/dpkg returned an error code (1)
```

Looking at the crash log, the following can be seen:

```
ProblemType: Package
DKMSBuildLog:
```

AMD Community › Support Forums › Drivers & Software › MAJOR issue with amdgpu-dkms...

tyalassio Adept I

11-08-2023 08:40 AM

MAJOR issue with amdgpu-dkms...

Hi guys,

I am a professional IT staff for over 20 years. As of yesterday, please be noted that amd-dkms package for Ubuntu 22.04, 23.04 and 23.10 cannot be compiled anymore. Furthermore, on 23.04 it totally broken the dpkg system, and had to manually remove the entry after it failed to compile. The problem started yesterday right after a system update, I wasn't too sure what it was, but NOT a kernel update. It seemed a compiler update or a package updated from AMD as well.

The problem is that the failure to compile renders the system completely broken, with black screen and unable to install/remove any more packages unless dpkg is manually fixed.

Thank you,
Best Regards,
Simone

2 Likes REPLY

r/Ubuntu Search in r/Ubuntu

r/Ubuntu • 5 mo. ago
gokacinlar

Errors were encountered while processing: amdgpu-dkms & warning: the compiler differs from the one used to build the kernel?

Hello, I was trying to install AMD graphics driver via `amdgpu-install` script for my fresh Ubuntu 22.04.3 LTS install and encountered an error saying:

```
Error! Bad return status for module build on kernel: 6.2.0-36-generic (x86_64)
Consult /var/lib/dkms/amdgpu/6.2.4-1646729.22.04/build/make.log for more information. dpkg: error
```

When I look at the `/var/lib/dkms/amdgpu/6.2.4-1646729.22.04/build/make.log` I get a warning saying:

```
DKMS make.log for amdgpu-6.2.4-1646729.22.04 for kernel 6.2.0-36-generic (x86_64)
Prş 02 Kas 2023 19:40:09 +03 make: Entering directory '/usr/src/linux-headers-6.2.0-36-generic' w
```

I was unable to solve problem. Any ideas how can I solve this issue?Thanks in advance.

4 10 Share

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Some potential issues installing **amdgpu-dkms** on Ubuntu 20.04 and 22.04

- When the make.log shows compilation errors, often it's because you're using an unsupported kernel

```
dpkg: error processing package amdgpu-dkms (--configure):
installed amdgpu-dkms package post-installation script subprocess returned error
exit status 10
Errors were encountered while processing:
amdgpu-dkms
E: Sub-process /usr/bin/dpkg returned an error code (1)
```

Ensure your Linux kernel version is supported, e.g.

- On Ubuntu 20.04.6, install Linux Kernel 5.15 with Hardware Enablement (HWE) stack
  `sudo apt install --install-recommends linux-generic-hwe-20.04`

- On Ubuntu 22.04.4, install Linux Kernel 6.5 with
  `sudo apt install --install-recommends linux-generic-hwe-22.04`

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

- Pytorch is a python package based on the Torch machine learning library
- In March 2021, Pytorch (v1.8) was made available for AMD GPUs with ROCm 4.0
- Torch uses MIOpen, ROCBlas, and RCCL to provide optimal performance on AMD GPUs
- Pytorch can be installed with ROCm support via `pip`
- Use the `cuda` device type to run on GPUs

https://pytorch.org/blog/pytorch-for-amd-rocm-platform-now-available-as-python-package/ (Last accessed March 25, 2024)

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Install pytorch in a conda environment

- As of March 2024, ROCm 6.x release available through nightly builds

```
(base) $ conda create -n pytorch_env python=3.9

(base) $ conda activate pytorch_env

(pytorch_env) $ pip install --pre torch torchvision torchaudio --index-url https://download.pytorch.org/whl/nightly/rocm6.0
```

pytorch with ROCm 6.0.x support will be available in stable release with pytorch 2.3 (eta July 2024)

https://github.com/pytorch/pytorch/issues/120433 (Last accessed March 25, 2024)
Pytorch release cadence : https://github.com/pytorch/pytorch/blob/main/RELEASE.md#release-cadence (Last accessed March 25, 2024)

AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Verifying the installation

```
#!/usr/bin/env python
import torch
print(torch.cuda.is_available())
if torch.cuda.is_available():
  print(torch.version.hip)
```

```
$ python torch-test.py
True
```


AMD
together we advance_

Set up ROCm 6.0.x and Pytorch

Using a docker image

```
$ docker pull rocm/pytorch:latest

$ docker run -it --cap-add=SYS_PTRACE --security-opt seccomp=unconfined \
--device=/dev/kfd --device=/dev/dri --group-add video \
--ipc=host --shm-size 8G rocm/pytorch:latest
```

https://rocm.docs.amd.com/projects/install-on-linux/en/develop/how-to/3rd-party/pytorch-install.html#using-docker-with-pytorch-pre-installed (Last accessed March 25, 2024)

AMD
together we advance_

Benchmarks

AMD
together we advance_

Benchmark Systems

Hosted at Fluid Numerics shop

“Sega”

- 1x AMD Ryzen Threadripper 7960X (24-Cores Zen4)
- PCIe v4 x16
- 1x AMD Radeon RX7600 GPU
- 1x AMD Radeon Pro W7800 GPU
- More economical pricing

“Noether”

- 2x AMD EPYC 7313 CPU (16-cores Zen3)
- 2x PCIe v4 x16
- 4x AMD MI210 GPU
- “Call for pricing”

When running pytorch on Radeon RX7600 (gfx1102), override GFX version

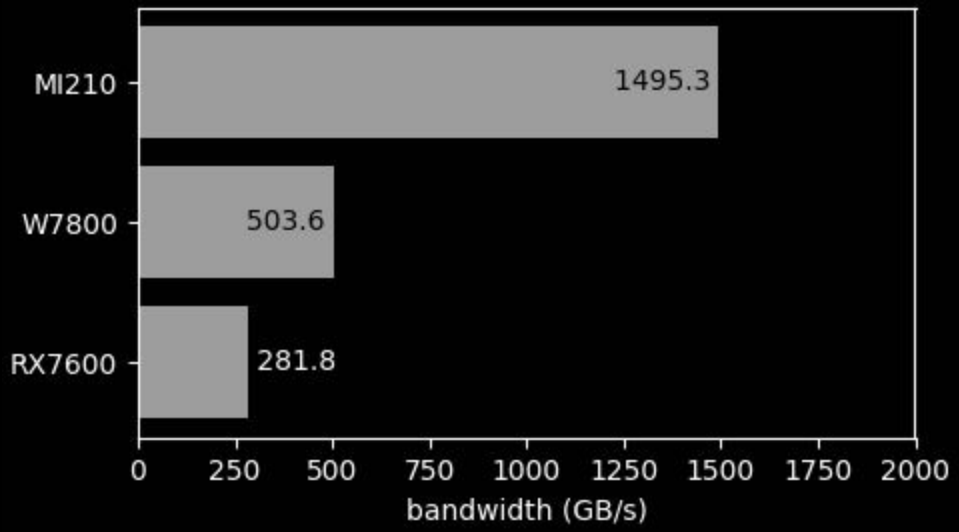
```
HSA_OVERRIDE_GFX_VERSION=11.0.0 python test/eigenmodes_squarebasin_hr.py
```


AMD
together we advance_

Benchmark Systems - Microbenchmark performance

Hosted at Fluid Numerics shop

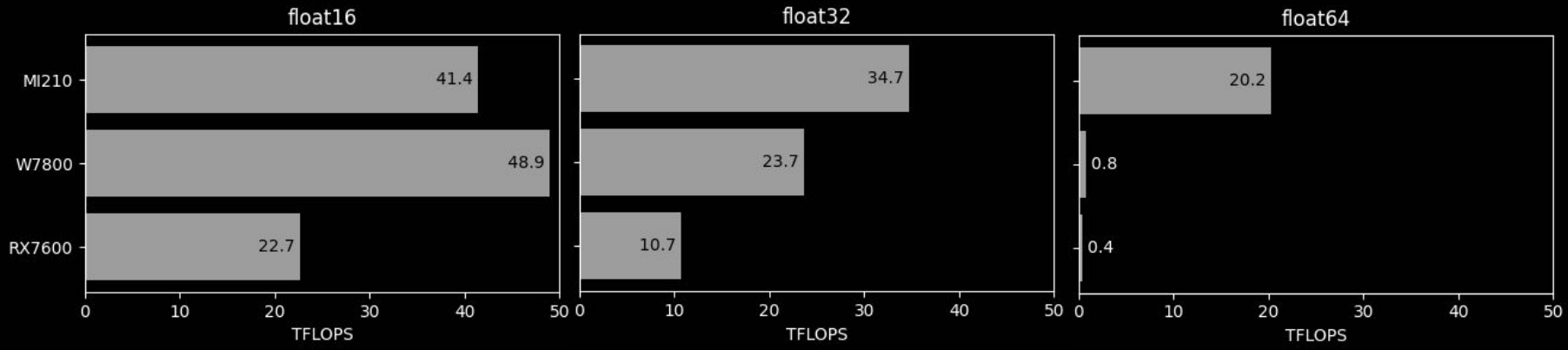
We use mixbench (github.com/ekondis/mixbench) to assess memory and compute performance



AMD
together we advance_

Benchmark Systems - Microbenchmark performance

Hosted at Fluid Numerics shop

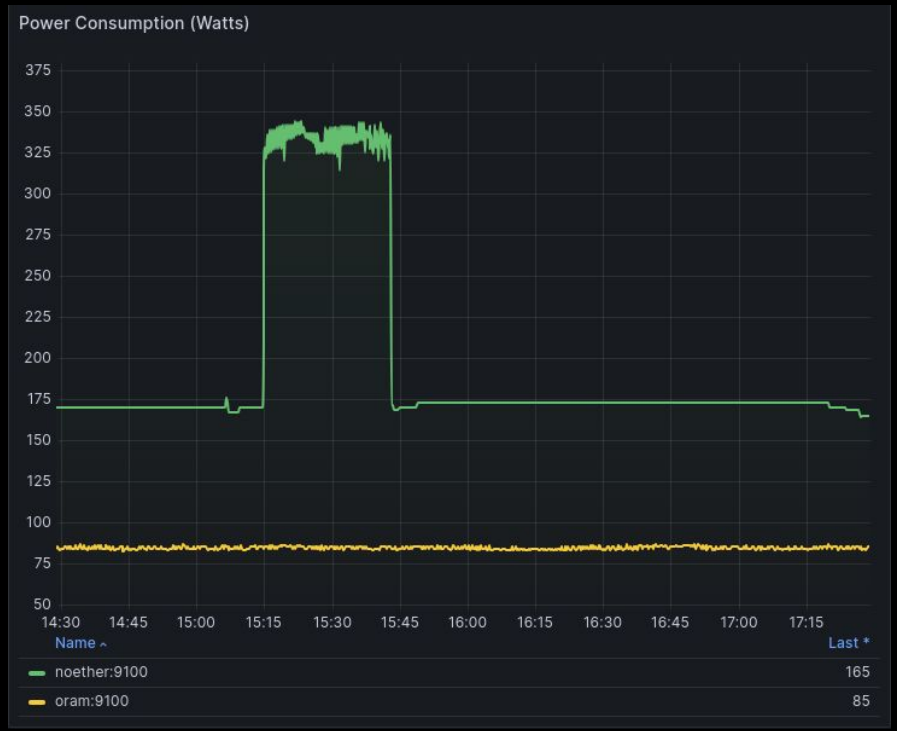


AMD
together we advance_

Benchmark Metrics

Search Time (per iterate)	Energy Consumption

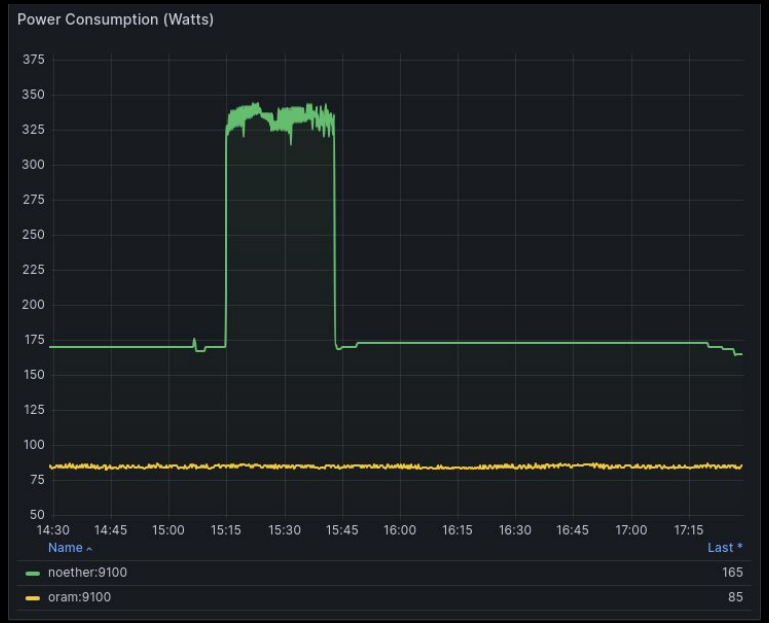
```python
import time
tic = time.perf_counter()
evals, evecs, r =
  model.calculate_dirichlet_modes(tol=1e-9,
                                  max_iter=150)
toc = time.perf_counter()
```



AMD
together we advance_

Estimating energy consumption

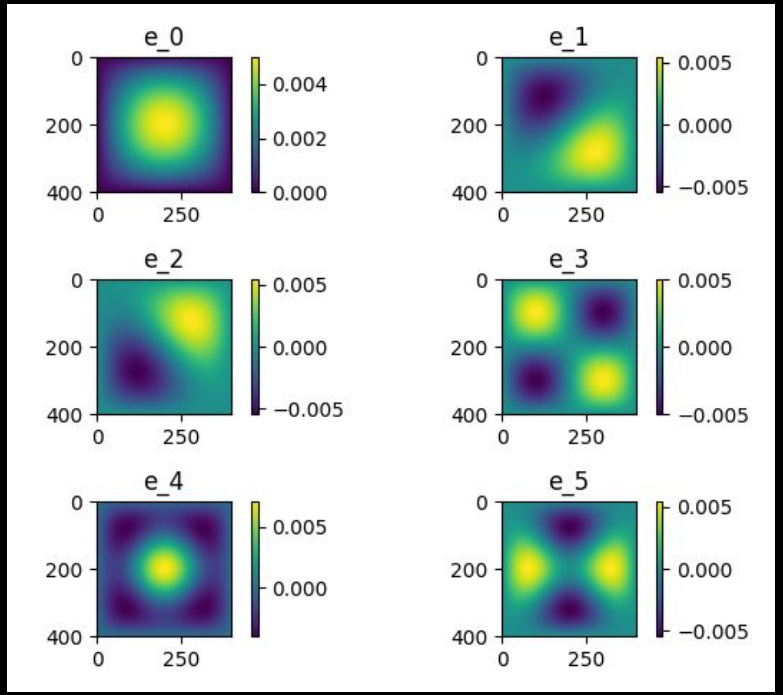
- We use `node_exporter` with prometheus to serve energy consumption metrics and Grafana to monitor and query measurements.
- Use `--exclusive` Slurm job to ensure our job is the only one running on a node.
- Energy consumption directly attributed to the workload is the integral of energy consumption rate from start to end of the job.



AMD
together we advance_

IRLM Eigenvalue Search Benchmark

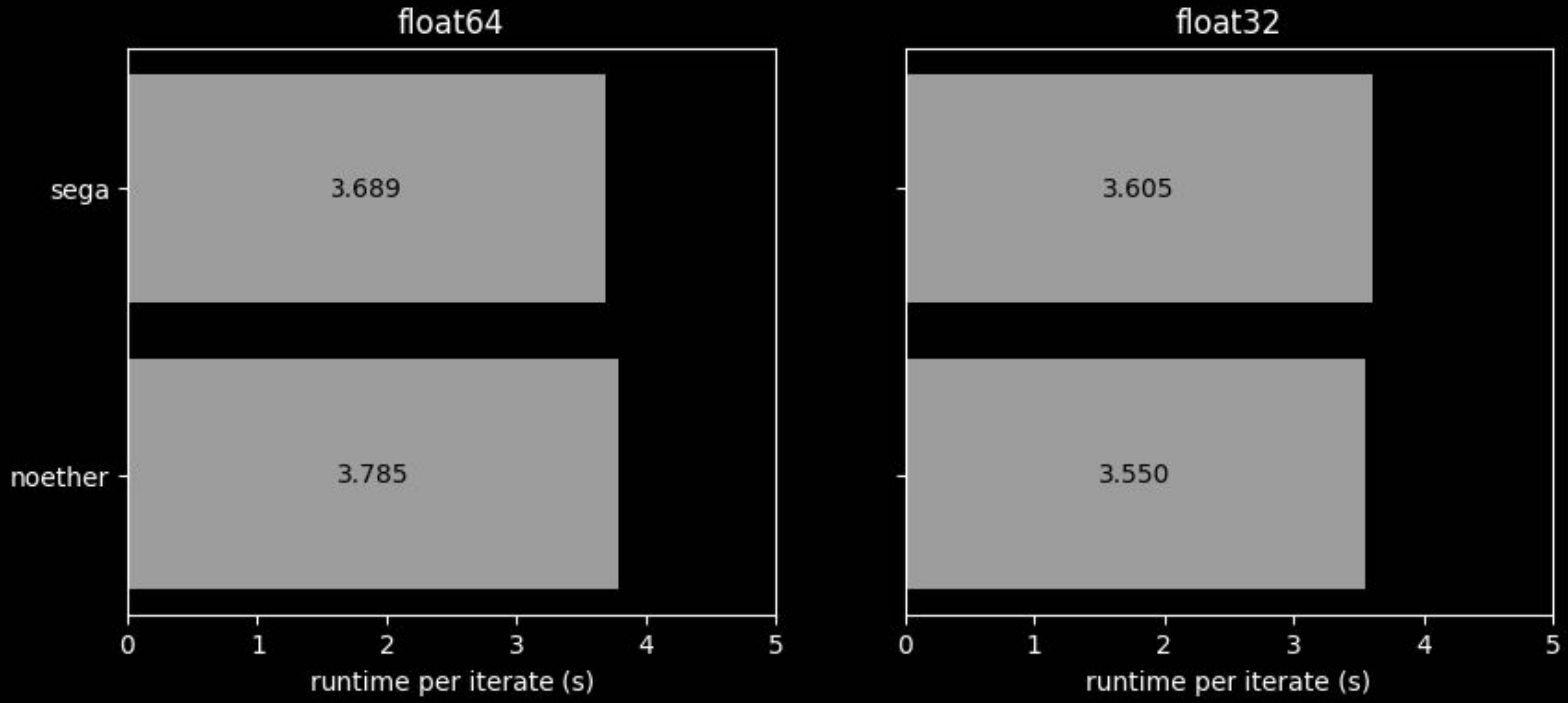
- Grid size - 400x400
- Number of eigenmodes - 500 (fixed)
- Krylov search space size - 515 (fixed) (< 1GB memory)
- Precision - fp32, fp64



AMD
together we advance_

IRLM Eigenvalue Search Benchmark

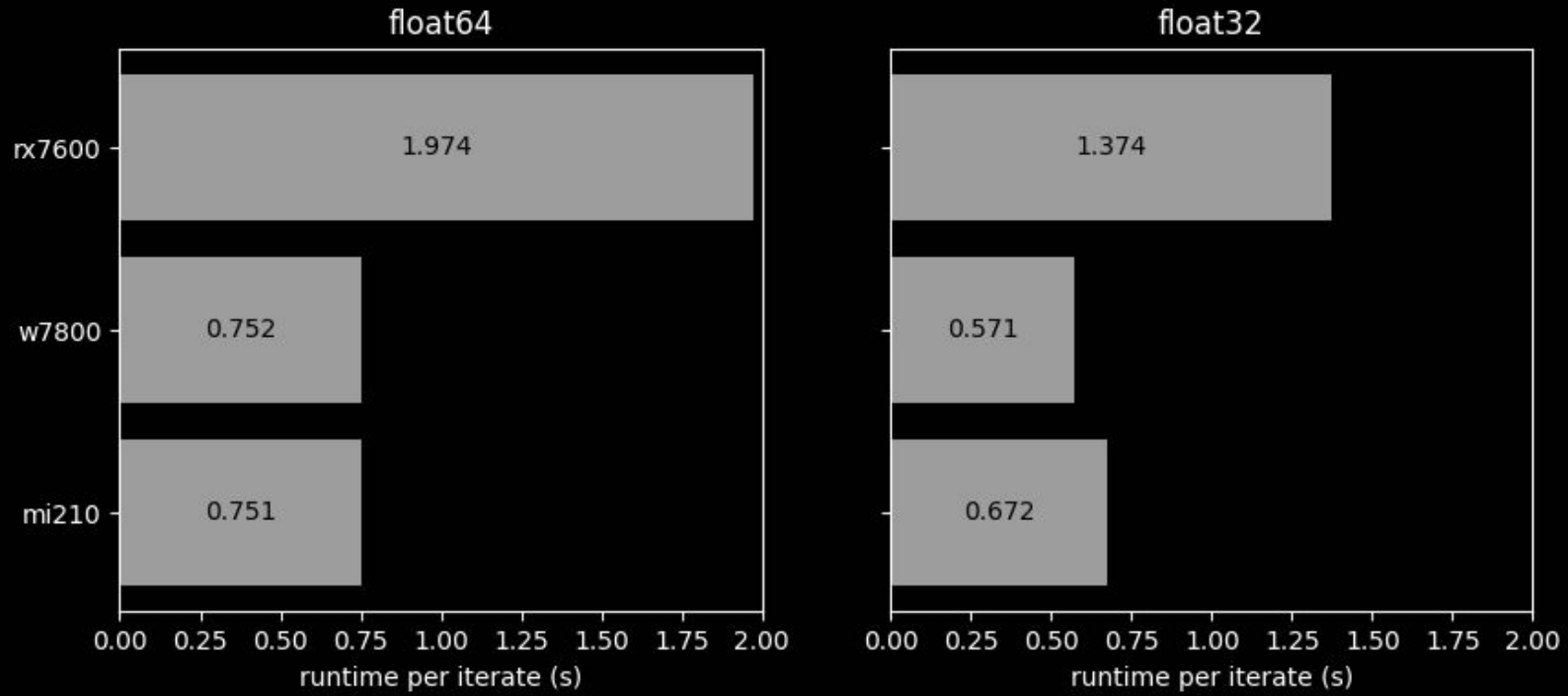
Serial CPU Runtime



AMD
together we advance_

IRLM Eigenvalue Search Benchmark

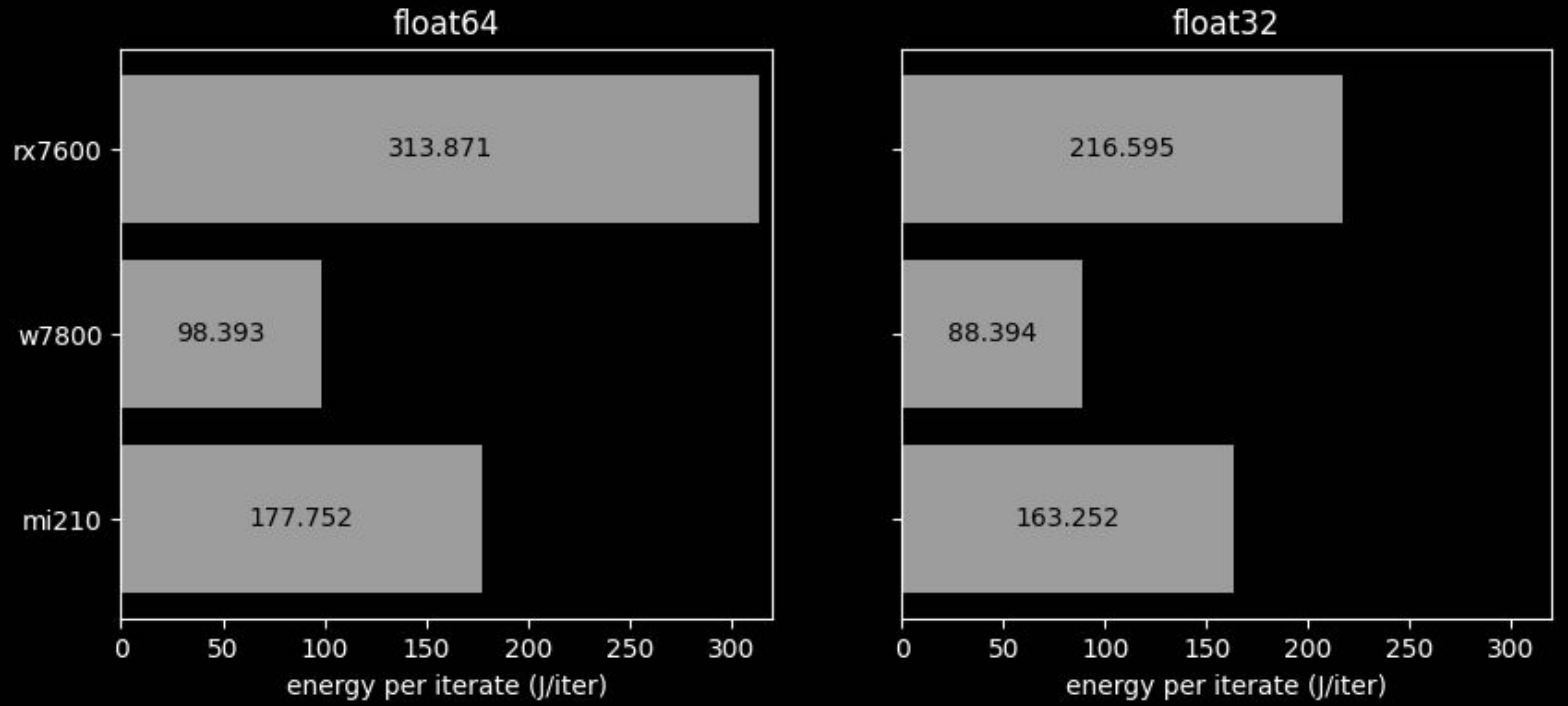
GPU Runtime



AMD
together we advance_

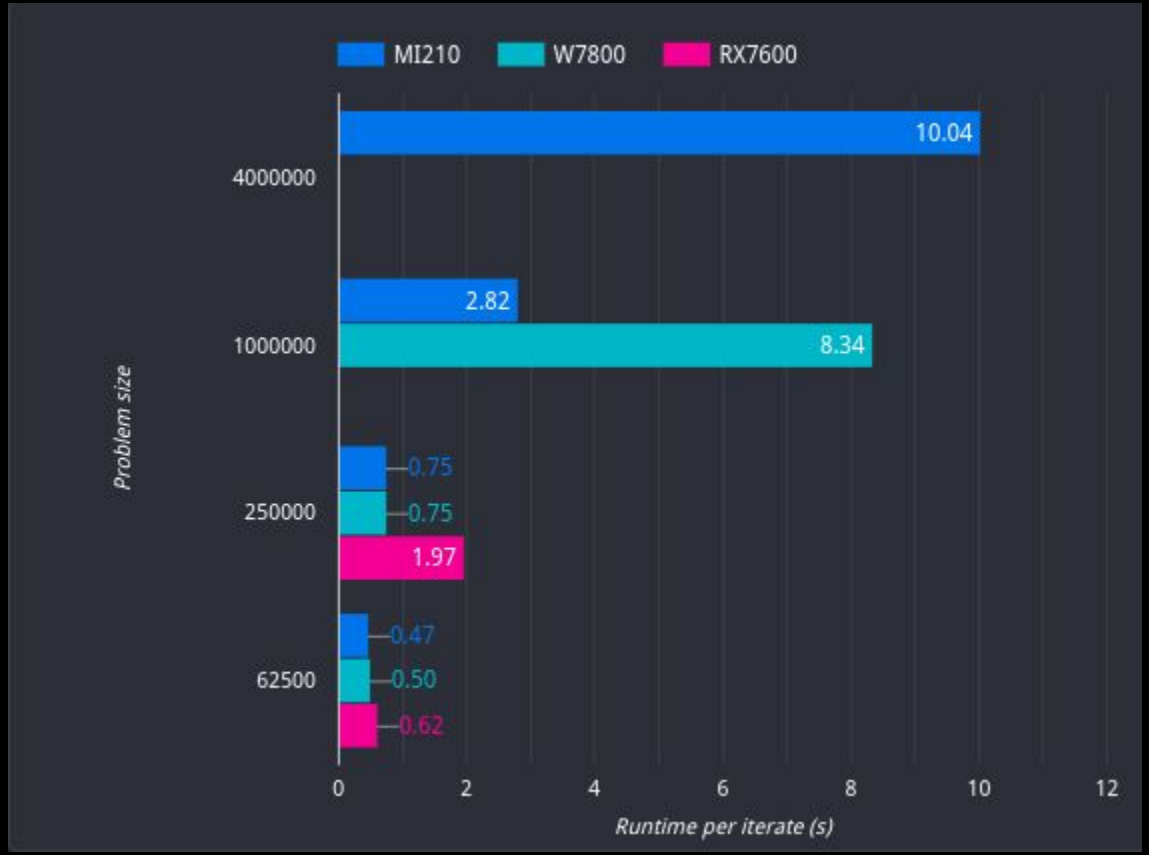
IRLM Eigenvalue Search Benchmark

GPU Estimated energy per iterate



AMD
together we advance_

IRLM Eigenvalue Search Benchmark



AMD
together we advance_

IRLM Eigenvalue Search Benchmark

A few remarks

- These results are preliminary, but suggest the RDNA3 architecture is an economical choice for some of our problems.
- The IRLM is sensitive to round-off errors; additional work is needed to obtain converged solutions for fp32
- Generalizing our results to other applications is not necessarily appropriate. The only benchmark that matters is yours.

AMD
together we advance_

Find Fluid Numerics Online

www.fluidnumerics.com


r/fluidnumerics


github.com/fluidnumerics


youtube.com/@FluidNumerics


linkedin.com/company/fluidnumerics

AMD
together we advance_

Q & A

AMD
together we advance_