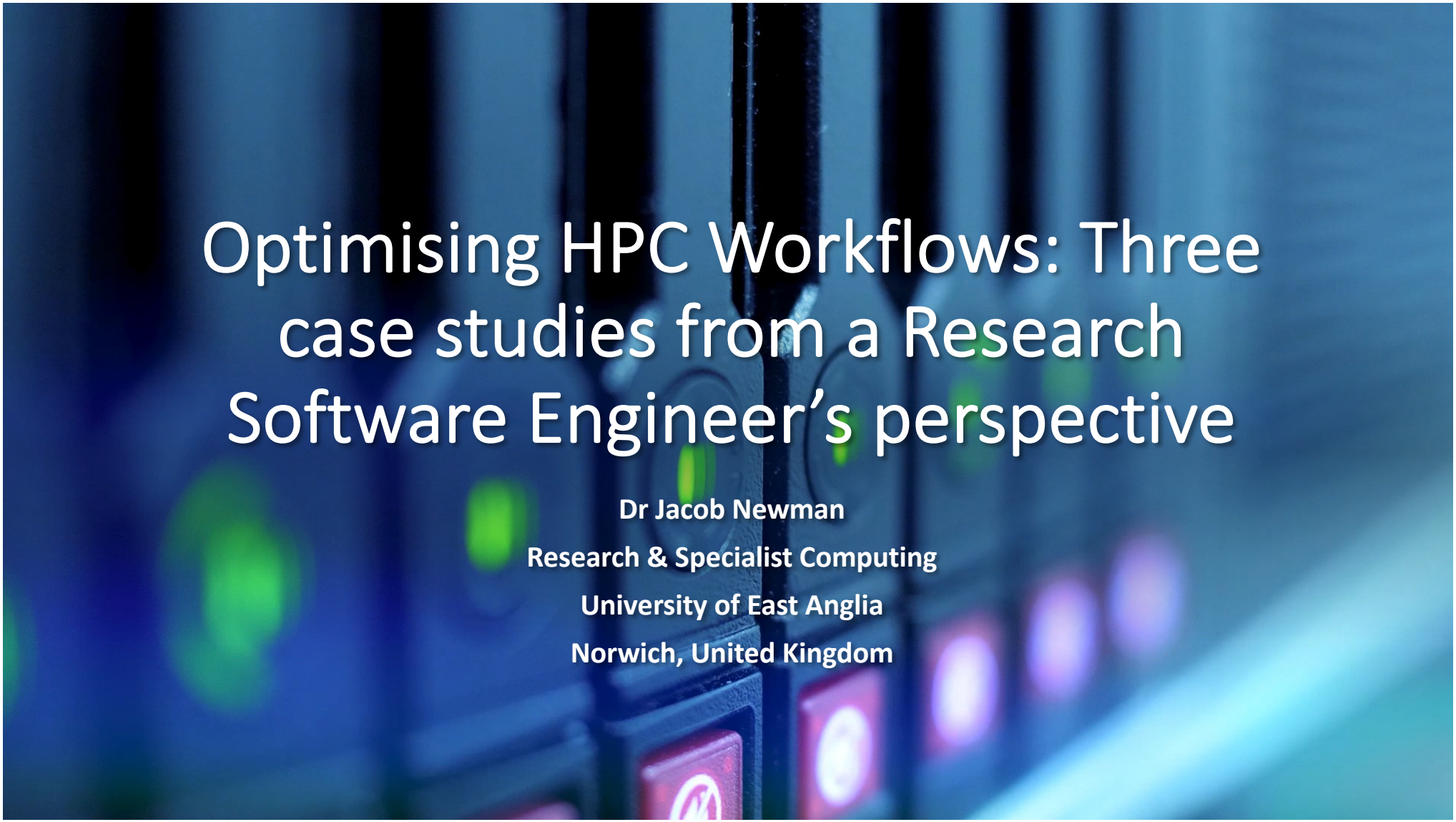


Optimising HPC Workflows: Three case studies from a Research Software Engineer's perspective

Dr Jacob Newman

Research & Specialist Computing

University of East Anglia

Norwich, United Kingdom

Me: Research Software Engineer

- Previously a computing researcher
- Extensive user experience of HPC
- Advocate for HPC in research
- Appointed RSE in May 2021

Outreach

Optimisation

UEA's HPC: Ada

- Centos 7
- Slurm 19.05
- About 10,000 cores
- Across 173 CPU nodes
- 57 GPUs
- Including 30 Nvidia Quadro RTX6000s
- InfiniBand 100Gb/sec
- GPU/Compute/IB/Hi-Mem/Vis partitions

Our users

- About 1000 users
- Range of disciplines
- Simulations, modelling, machine learning, data processing, code development, etc.
- Associated levels of computing proficiency


3 Case Studies

User A

Senior lecturer in Chemistry

Researching molecular dynamics

GROMACS

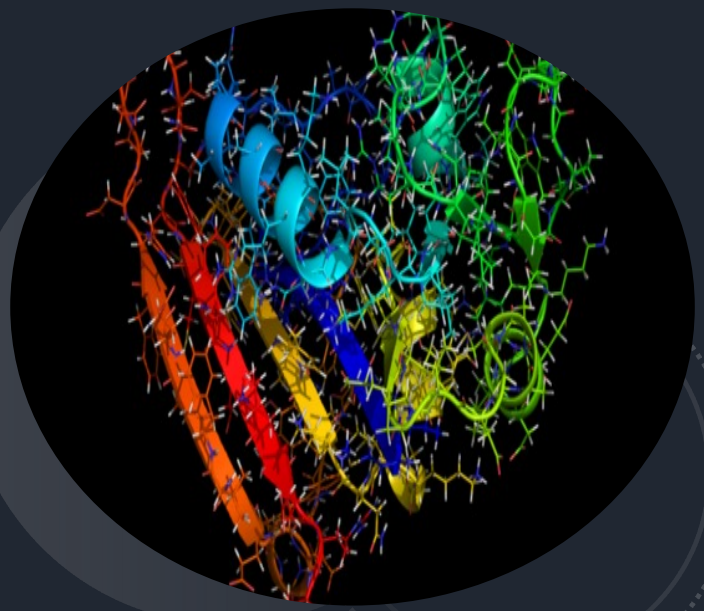
User A

“Currently I am trying to run molecular dynamics calculations using **gromacs** on the **compute** queues. These show **0.32 ns/day** on a shared node, **1.31 ns/day** on an exclusive node. Perhaps I am not using the queue properly because on my desktop **at home** (Intel i5-2500 @3.3GHz) I can achieve over **8 ns/day**. Anyway, **I need** a simulation of **200 ns** or so, hence I am wondering whether the other hardware available will offer better performance. If so, can I be given access to the appropriate queue. gromacs is **GPU aware** though whether this offers a significant improvement depends on the balance of calculations and the advice is to test the system.”

User A

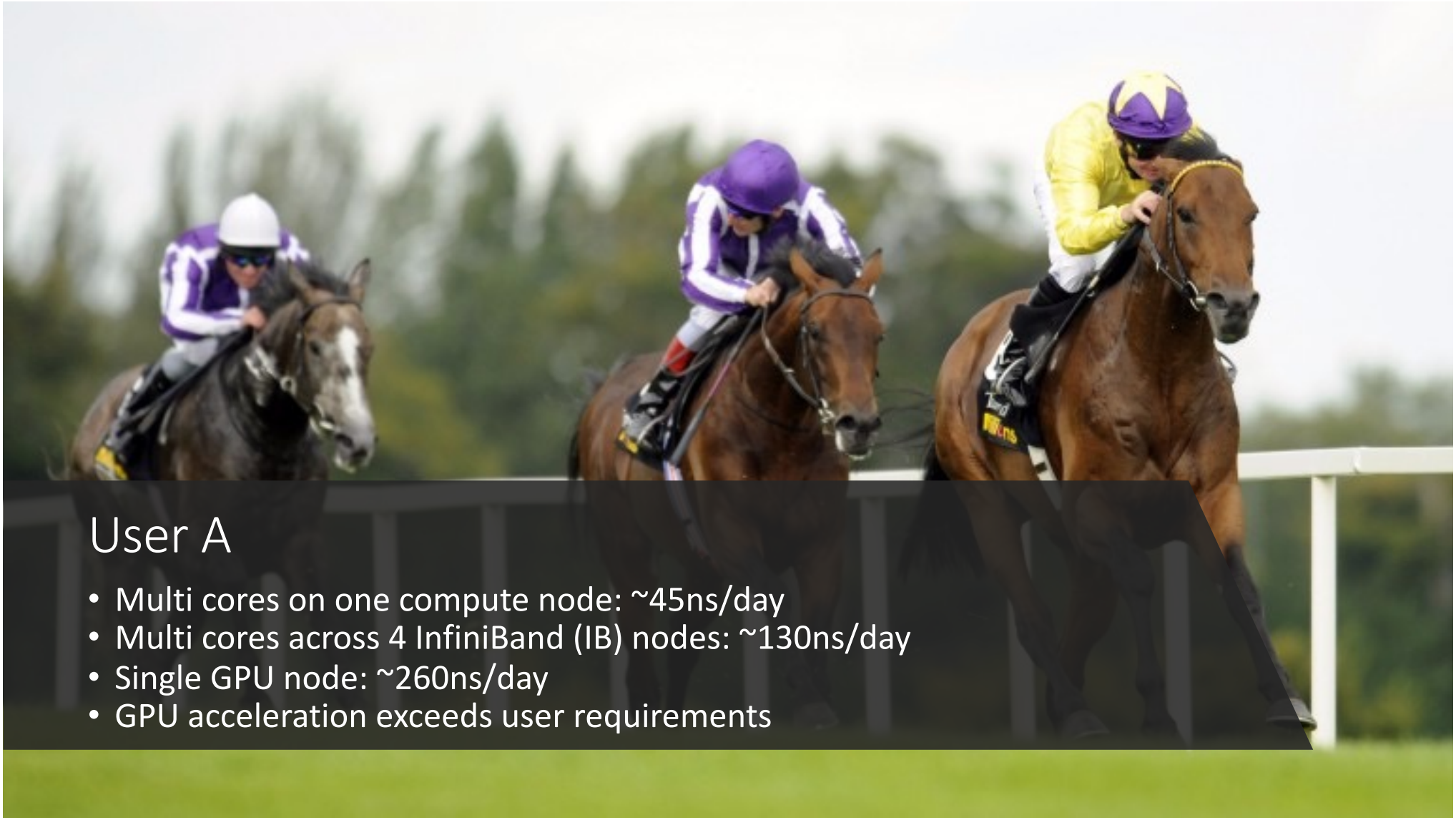
- Replicate results
- Now what?
- 2019.4
- 2020.4
- 34x speed increase
- ~45 ns/day, which is still not close to 200 ns/day

```
   gromacs/2019.4/gcc                    (D)
   gromacs/2020.4/gcc-gpu-cuda11.0
   gromacs/2020.4/gcc-openmpi-eth        (D)
```



MPI

GPU



User A

- Multi cores on one compute node: ~45ns/day
- Multi cores across 4 InfiniBand (IB) nodes: ~130ns/day
- Single GPU node: ~260ns/day
- GPU acceleration exceeds user requirements

User B

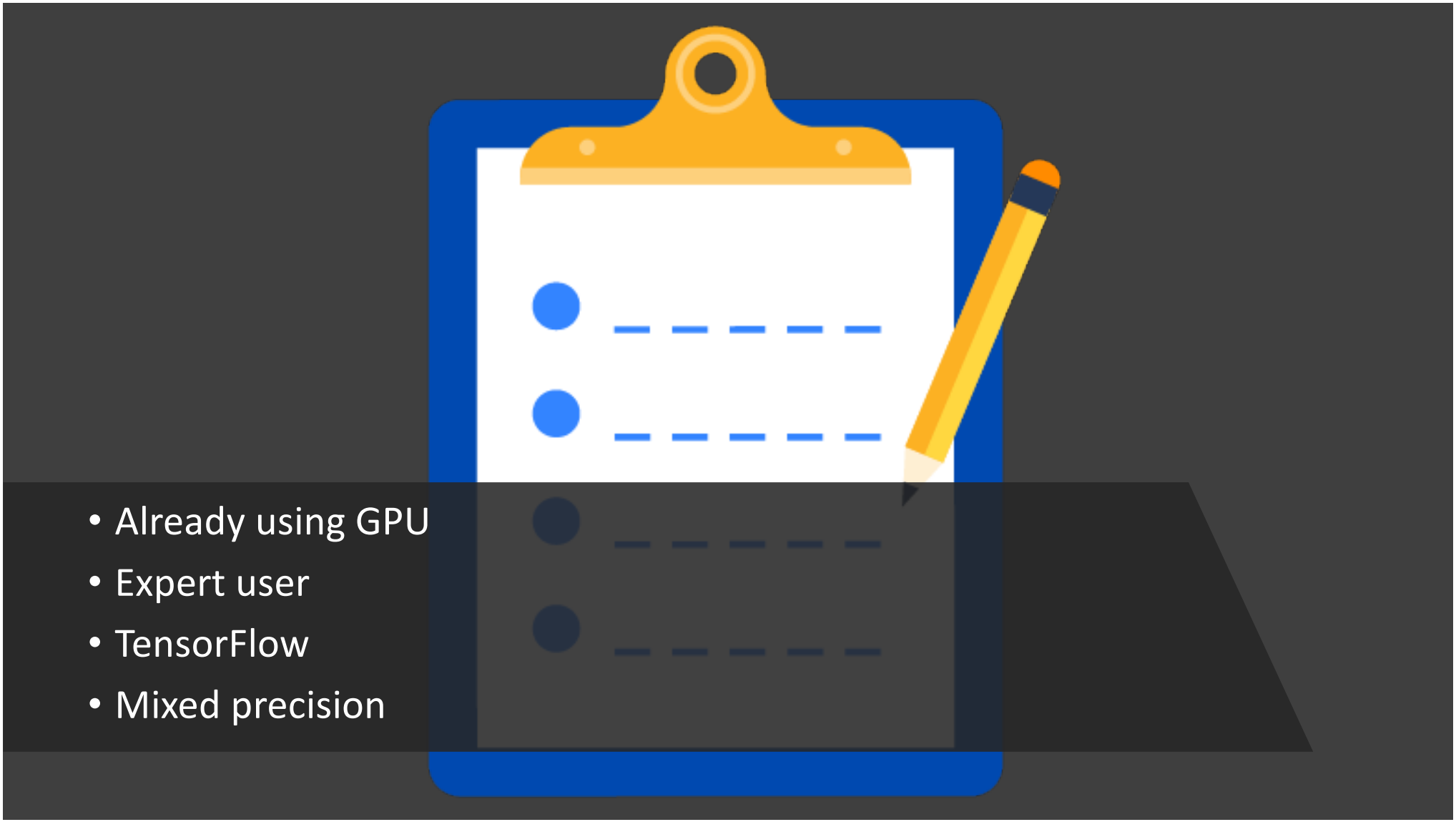
Postgraduate student in Computing

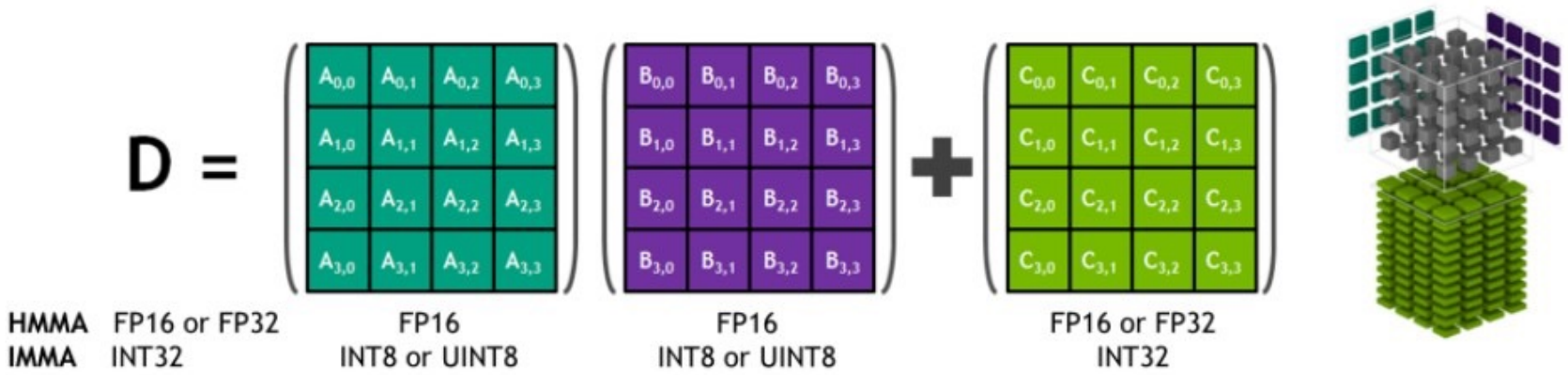
Time-series classification

TensorFlow

User B

Are there any way you can think of that might improve efficiency/speed up GPU jobs?


• Already using GPU
• Expert user
• TensorFlow
• Mixed precision



Tensor Cores

- Reduces memory usage
- Speeds up data transfer
- Performance increases, shortening job duration

Mixed precision

```
from tensorflow.keras import mixed_precision

# The feature is experimental in TensorFlow < 2.4. >= 2.4
it is a main feature.
mixed_precision.experimental.set_policy('mixed_float16’)

# The output layer needs to be float32 for numerical
stability
dense_1 = Dense(nclass, activation=activations.softmax,
dtype='float32', name="dense_3_ptbdb")(dense_1)
```

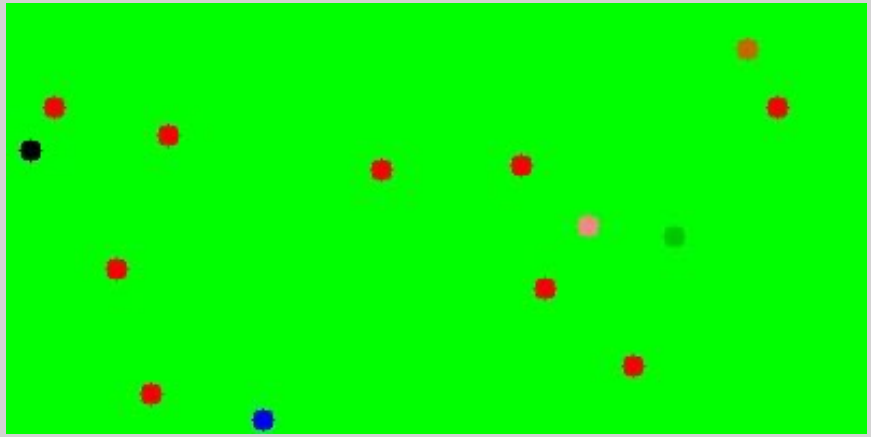
My own tests

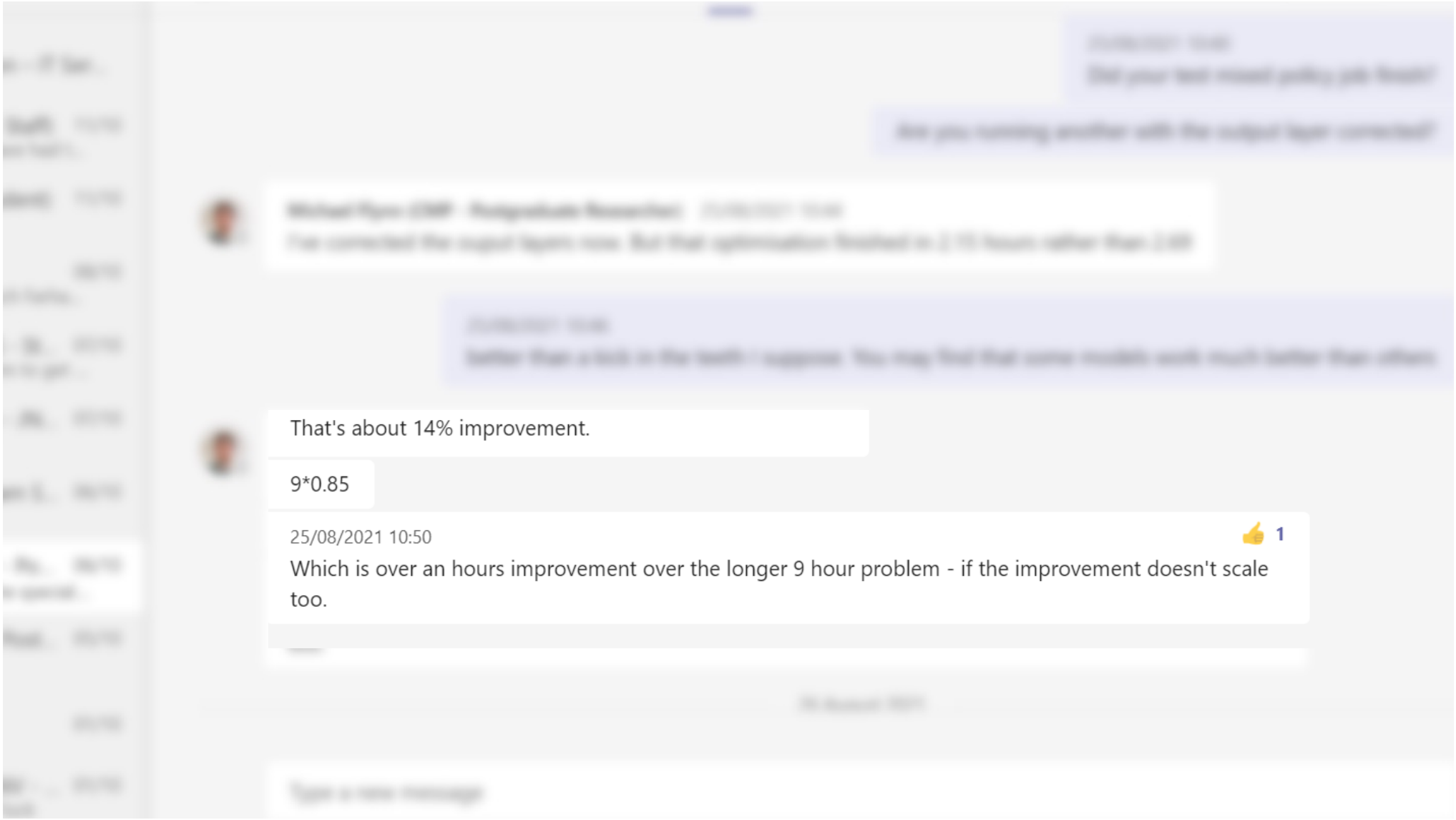
Without the mixed policy…

```
Epoch 1/100
1490/1490 [======] -
1124s 754ms/step
```

And with the mixed policy…

```
Epoch 1/100
1490/1490 [======] -
485s 325ms/step
```




That's about 14% improvement.
9*0.85
25/08/2021 10:50
1
Which is over an hours improvement over the longer 9 hour problem - if the improvement doesn't scale too.

User C

Researcher in Biology

Quantifying bird abundance

R

User C

Hi Jake,

I hope you are well. I have been using the code you made for Claire to run some analysis on the cluster, however after 2 weeks it timed out.

I think the best idea is to try and split it up using the site index but I have no idea how to do 2 array indexes or if that is even possible? I have attached all the code. Do you have some time to talk about this soon?

Many thanks,
Cat

Current workflow

- For 16 datasets
  - For 15k sites
    - Process 330k records
- Takes at least two weeks to complete
- Current implementation submits a job for each dataset
- Bash script to submit an array job from idx 1 to 16

- Suggestion to process the 15k sites in smaller batches

New workflow

		Dataset ID		
		1	2	3
Site ID	1	Array ID 1	Array ID 2	Array ID 3
	2	Array ID 4	Array ID 5	Array ID 6
	3	Array ID 7	Array ID 8	Array ID 9

New workflow

```
#!/bin/bash
SBATCH_ARRAY_INX=$(ls ./datasets/*.csv | wc -l)
SBATCH_ARRAY_INX=$((SBATCH_ARRAY_INX*10))
sbatch --array=1-$SBATCH_ARRAY_INX pairwise_site.sh
```

New workflow

```
idx <- strtoi(args[1],base=10)
num_datasets <- strtoi(args[2],base=10)

dataset_idx <- ((idx-1)%%num_datasets)+1
site_idx <- ceiling(idx/num_datasets)
```

Worked example, where idx=19 and num_datasets=16:

dataset_idx = ((19-1)%%16)+1 = 3

site_idx = ceiling(19/16) = ceiling(1.1875) = 2

Results

- Longest job completed in 4.5 days compared to "over two weeks"
- The cumulative time is 389 days
- But more work to be done…
- Could increase granularity
- Could ensure the jobs are distributed equally
- But in this case, five days is acceptable

Benefits of an RSE service

Since you started helping I've managed to get 1,200 gpu jobs completed

[redacted]

Jake was brilliant and worked with me for weeks very patiently to fix this, I can't thank him enough.

[redacted]

Hi Jake,

This is looking great. I think it is exactly what I want.

it work, thaaaaaank you

Thanks a lot Jacob for your kind help!

Benefits of an RSE service

- Relationship building
- Recognition of RSCS
- Freeing time
- Proactive mitigation
- Jobs complete more quickly
- Improved productivity: global problems solved faster

Increased efficiency?

“REDUCE POWER CONSUMPTION”?

INCREASE RESEARCH OUTPUT

POWER MANAGEMENT

Conclusions

- No two jobs are the same
- Impressive improvements can be made using simple solutions
- It is worth trying several approaches
- Productive users and good relationships

Thank you for listening