WattOps × HorneSci

Case study · HorneSci

WattOps helps HorneSci prove where its HST runtime saves energy

HorneSci develops HST for sparse and incremental computation. With joules by WattOps, it measured on real hardware where HST saves energy, and where it does not.

39%
less GPU energy and 30% less time on an NVIDIA H200, with half the data changing per update

46%
less energy per job on Apple M1 Max ($p = 0.0006$)

10–25%
of the data changing per update: where HST overtook the baseline on H200 in this workload

The challenge

Energy results are noisy: repeats of the same command can differ by 15% or more, enough to hide or invent a difference. HorneSci needed one fair method across GPUs, Apple silicon and Intel CPUs, against a baseline that was already optimized.

The solution: joules by WattOps

`joules compare` measured each version's whole command:

- One warm-up, then 7 runs per side in alternating (ABBA) order, which reduces order bias (M1 Max and GPUs; the NUC used 6 phases per side)
- Idle power measured before and after
- Medians and p-values for energy and time
- Every run kept, in CSV and JSON, for audit

At a glance

Customer
HorneSci

Industry
HPC and scientific computing

Product
joules by WattOps

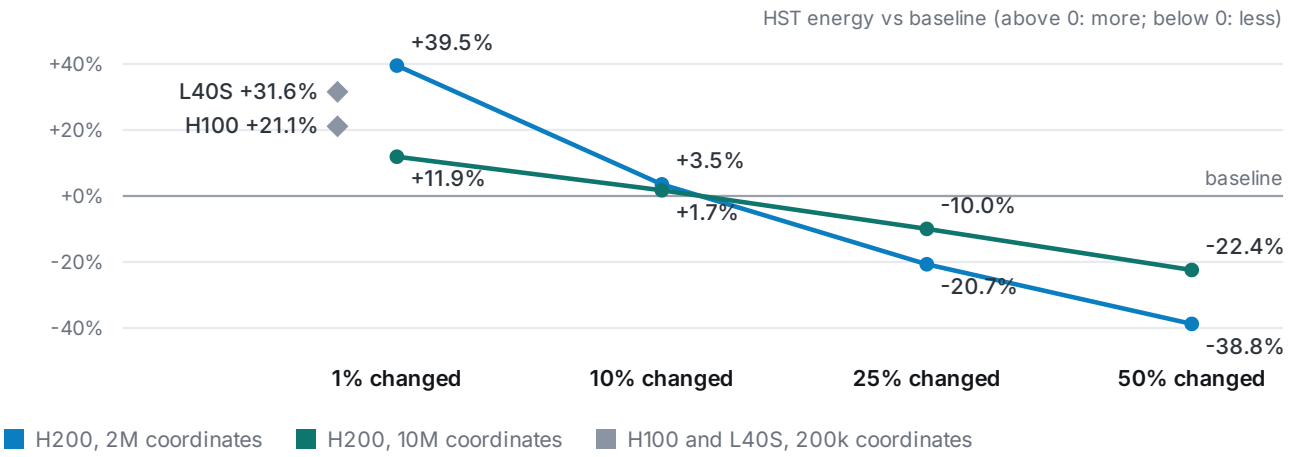
Hardware
NVIDIA H200, H100, L40S and T4, Apple M1 Max, Intel Core i7

Results
- Up to 39% less GPU energy
- 46% and 32% less on CPUs
- Shared-GPU split checked

NVIDIA H200

The more data changes, the more HST saves

HST was tested with 1% to 50% of the data changing per update, at 2 million and 10 million coordinates. With half the data changing, it used **38.8% less GPU energy** and **29.5% less time**. For this workload, the results suggest a crossover between 10% and 25% changed.



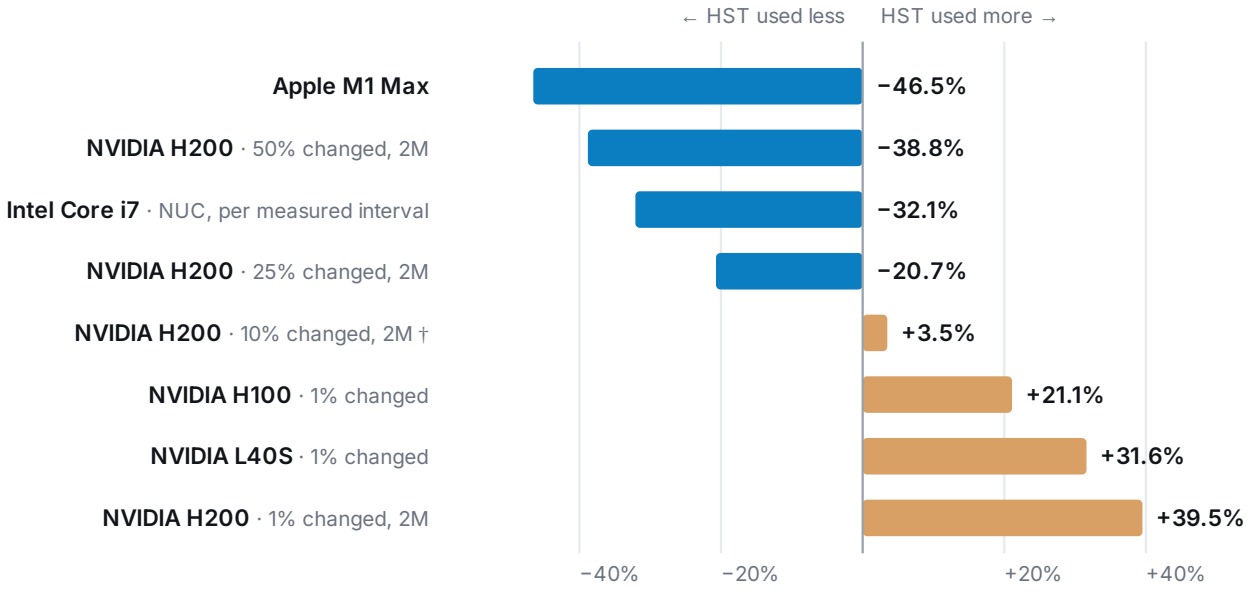
Data changed per update	2M · energy	2M · time	10M · energy	10M · time
1% ‡	+39.5%	+10.6%	+11.9%	+2.6%
10% ‡	+3.5% †	+0.6% †	+1.7% †	−0.8% †
25%	**−20.7%**	−13.3%	**−10.0%**	−5.2% *
50%	**−38.8%**	**−29.5%**	**−22.4%**	−10.8%

HST against the baseline, median of 7 runs each. † not statistically significant · * inconclusive (p = 0.053) · ‡ reported from summaries; the 25% and 50% cases were reconciled by HorneSci against raw data. Details on page 6.

Every machine, one method

Where HST saves energy, and where it doesn't

HST used less energy on both CPUs, and on H200 once 25% or more of the data changed. With 1% changing, the baseline used less on all three GPUs. Each machine ran its own workload, so each bar compares HST only with its own baseline.



Apple M1 Max: 46% less energy

HST finished the job in 12.2 seconds instead of 19.8, and used 133 joules instead of 249.

Median per command	Baseline	HST	Change	p
Runtime	19.821 s	12.226 s	**−38.3%**	0.000583
Integrated sampled Combined CPU+GPU+ANE power	249.165 J	133.426 J	**−46.5%**	0.000583
Energy above idle	47.216 J	23.314 J	−50.6%	0.053

The saving in energy above idle is not yet conclusive (p = 0.053). Details on page 6.

NVIDIA H100 and L40S

The gap a stopwatch misses

With 1% of the data changing, the baseline used less GPU energy on both GPUs. On the H100 the two finished almost together, HST 1.3% sooner, yet HST used **21% more energy**. Timing alone would not have shown it.

1% changed, 200k coordinates	Runtime	GPU energy	Energy above idle
NVIDIA H100 NVL	2.016 → 1.990 s · **−1.3%**	202.3 → 244.9 J · **+21.1%**	+54.2%
NVIDIA L40S	2.577 → 2.893 s · **+12.3%**	192.7 → 253.7 J · **+31.6%**	+133.4%

Baseline → HST, median of 7 runs each. All differences significant ($p \leq 0.011$). Details on page 6.

NVIDIA Tesla T4

A shared GPU, split fairly

Two jobs shared one GPU: a compute-heavy job, and a light job holding far more memory. joules split the energy by the time each kept the GPU busy, not by the memory each held.

Job	Allocated energy	Share	Memory snapshot
Compute-heavy job	8,154.379 J	**99.00%**	0.24 GB
Light job	82.763 J	1.00%	6.59 GB

Trust

Auditable to the sample

HorneSci kept the raw meter stream and audited it sample by sample. Eric Spencer found an extra GPU power line counted on Apple silicon when two samplers ran together. He reproduced it and proposed a fix; the corrected comparison reconciles all 1,529 samples one for one. WattOps shipped the fix in joules 0.8.0rc2.

Measure what your own jobs cost

joules is free: one command measures any job on NVIDIA, AMD, Intel or Apple hardware. WattOps adds energy and cost per job and per team across your GPU fleet. Start at wattops.dev

Run sparse or incremental workloads?

On H200, HST used up to 39% less energy once 25% or more of the data changed, and it used less on CPUs. Talk to HorneSci: hornesci.com · admin@hornesci.com

Method and notes

Measured domains: Apple CPU+GPU+ANE estimates, NVIDIA GPU energy and Intel package RAPL. These are not wall-plug readings. Each machine ran one workload, so results are separate per-machine comparisons; other workloads, sizes and libraries may behave differently. The GPU results compare HST Tile8×8 with an exact incremental CSC baseline, separate from cuSPARSE or a full application.

NVIDIA H200 NVL

600 W limit, driver 595.84, NVML GPU-device energy, joules 0.8.0rc1. Seven runs per arm after one warm-up, ABBA, 30 s idle before/after, 60 C start ceiling. Fixed FP64 fixture, seed 23: 10,000 reset/update steps per command at 2,000,000 coordinates, 2,000 at 10,000,000. Energy is the median whole-command GPU energy, excluding CPU energy; p-values are two-sided Mann-Whitney U. Each iteration resets the output and applies the same fixed delta, with changes grouped into eight-coordinate blocks. Whole commands include common CPU setup that both arms share, a larger part of each command at 10,000,000 coordinates: there, with 50% changed, GPU energy above idle fell 34.6%. At 10% changed the reported differences were not statistically significant; at 10,000,000 coordinates with 25% changed the runtime difference was inconclusive ($p = 0.053$). The 10–25% crossover is suggested by this workload, not an established threshold. Every case passed the kit's validator, including reference/final relative L2 below 1e-12. HorneSci reconciled the 25% and 50% cases against their raw data and source manifests; the 1% and 10% cases are reported results from summaries only.

NVIDIA H100 NVL

400 W limit, driver 580.173.02, NVML GPU-device energy, joules 0.8.0rc1. Seven runs per arm after one warm-up, ABBA, 30 s idle before/after, 60 C start ceiling. Fixed FP64 fixture: 200,000 coordinates, 2,000 changed, 100,000 reset/update operations. Runtime $p = 0.011$, energy $p = 0.000583$. Idle 59.783 to 62.582 W; starts 32-35 C; observed 1785 MHz, no clock locking or reported throttling. All 16 processes passed reference/final relative L2 below 1e-12; the independent raw-file audit raised no flags.

NVIDIA L40S

NVML GPU-device energy, joules 0.8.0rc1. Fixed FP64 sparse updates: 200,000 coordinates, 1% changed, 100,000 reset/update operations per command. Both arms reuse operator, changed set and buffers. All three differences $p = 0.000583$. Idle 62.721 to 62.687 W; starts 32-34 C; observed 2040 MHz, no clock locking or reported throttling. All 16 processes passed reference/final checks below 1e-12.

Apple M1 Max

MacBook Pro, joules 0.8.0rc1 with the powermetrics cpu_power sampler; whole command, integrated sampled Combined CPU+GPU+ANE power, 7 runs per arm. Idle moved from 7.597 W to 11.681 W during the session on a desktop in use; energy above idle uses a straight-line correction between the two. Average power 12.571 to 10.877 W (−13.5%). All 16 processes, including warm-ups, passed at relative L2 error 1.177e-16.

Intel Core i7-1165G7 (NUC)

Earlier joules 0.7.10 cohort, 6 setup-excluded phases per arm, 20,000 updates, 30 s idle diagnostics before/after, 3 s rest; gross package RAPL energy, without idle subtraction. 142.41 to 96.70 J; runtime 17.69 to 11.58 s, average power 8.06 to 8.34 W. An M2 cohort remains outside these results pending the new meter audit.

NVIDIA Tesla T4

AWS, driver 580.126.09, PyTorch 2.7.0+cu128, Python 3.12; 121.755 s, 100% SM-time coverage and matched process IDs. Tested emailed joules 0.7.11. Energy is allocated by SM time; memory is a snapshot.

Try it

Run your own comparison

joules is free and runs on Linux, macOS and Windows with NVIDIA, AMD, Intel and Apple hardware. A comparison like HorneSci's takes one command, with the two quoted commands replaced by your own:

```
pip install landauer-gap==0.8.0

joules compare --names hst,baseline --baseline baseline \
  --runs 7 --order abba --warmup 1 --subtract-idle 30 \
  --markdown results.md --csv runs.csv --out results.json \
  "./run --runtime hst" "./run --runtime baseline"
```

It alternates the commands and measures idle before and after. It reports median energy, runtime, average power and energy x time, with p-values for energy and runtime. CSV and JSON keep every measured run.

About WattOps

WattOps measures the energy and cost of GPU and AI workloads, per job and per team. Its command-line tool, joules, reads the hardware's own energy counters. WattOps is built by PHENEX TECHNOLOGIES INC. in Vancouver, Canada. wattops.dev

About HorneSci

HorneSci is a software and research company co-founded by Hunter Horne and Eric Spencer. It helps teams reduce repeated computation in existing software, updating results from changed inputs and measuring each approach against a defined baseline. HST is part of its work on sparse and incremental computation. hornesci.com