University of Stuttgart
Germany
IPVS, Scientific Computing

Is RISC-V Ready for Machine Learning?

Portable Gaussian Processes Using Asynchronous Tasks

June 2026

Alexander Strack,
Patrick Diehl, and
Dirk Pflüger

Motivation

- x86-64 based CPUs dominated HPC for decades
- Shift to many-core chiplet designs, accelerators, and alternative ISAs such as ARM (LineShine and Fugaku) and RISC-V

Motivation

- x86-64 based CPUs dominated HPC for decades
- Shift to many-core chiplet designs, accelerators, and alternative ISAs such as ARM (LineShine and Fugaku) and RISC-V
- Are these emerging architectures ready for machine learning and scientific computing workloads?

Motivation

- MILK-V Pioneer Box: first publicly available RISC-V desktop-class system suitable for HPC

Motivation

Architecture	x86-64	ARM	RISC-V
CPU	**AMD EPYC 7742**	**Fujitsu A64FX**	**SOPHON SG2042**
Cores	64	48	64
Base clock	2.25GHz	2.20GHz	2.00GHz
Process	7nm/14nm	7nm	12nm
L3 Cache	256MB	n/a	64MB
RAM	2TB DDR4	32GB HBM2	128GB DDR4
SIMD	AVX2	SVE-512	RVV 0.7.1
TDP	225W	$\sim$140W	120W
Release	2019	2019	2023


Application:
Gaussian
Processes

What are Gaussian Processes?

- Widely used probabilistic models in machine learning
- Non-parametric: only a few tunable kernel hyperparameters
- Provide uncertainty estimates alongside predictions

Prediction & Uncertainty

Prediction and uncertainty computation:

$$\mathbf{y}_m = K_{mn} K_{nn}^{-1} \mathbf{y}_n \tag{1}$$

$$\mathrm{Cov}[\mathbf{y}_m] = K_{mm} - K_{mn} K_{nn}^{-1} K_{nm} \tag{2}$$

Prediction & Uncertainty

Prediction and uncertainty computation:

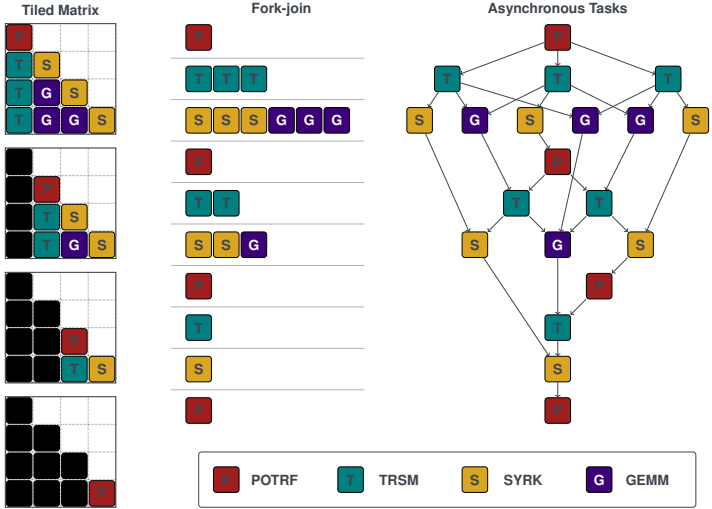
$$\mathbf{y}_m = K_{mn} K_{nn}^{-1} \mathbf{y}_n \tag{1}$$

$$\mathrm{Cov}[\mathbf{y}_m] = K_{mm} - K_{mn} K_{nn}^{-1} K_{nm} \tag{2}$$

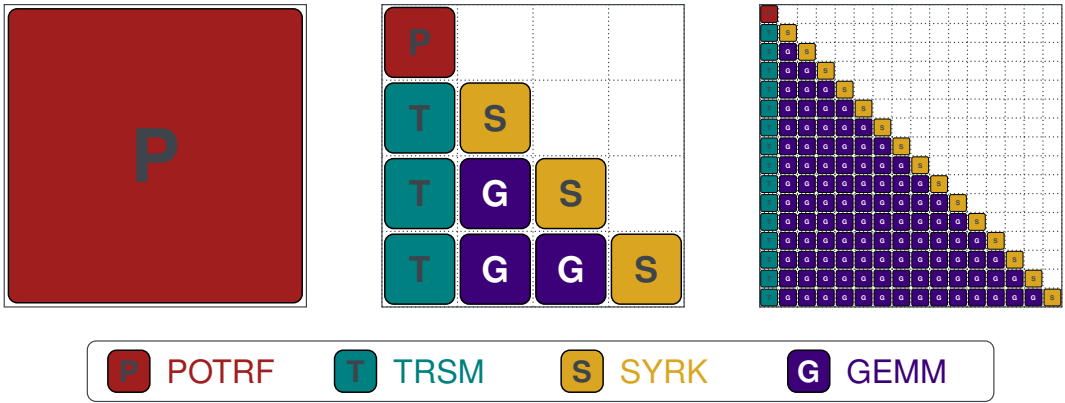
General approach to solve:

- Factorize symmetric positive-definite K_{nn} via **Cholesky decomposition**

Tiled Cholesky Decomposition



Tiled Cholesky Decomposition




Software Stack:
GPRat

GPRat

- Highly parallel C++ implementation of Gaussian process regression
- Parallelization at the **application level** using the asynchronous many-task runtime HPX
- Python bindings
- Accelerator support with CUDA and SYCL

BLAS Backend: From oneMKL to OpenBLAS

- GPRat needs a sequential, vectorized BLAS/LAPACK backend
- Historical choice: Intel oneMKL but no native ARM or RISC-V support

BLAS Backend: From oneMKL to OpenBLAS

- GPRat needs a sequential, vectorized BLAS/LAPACK backend
- Historical choice: Intel oneMKL but no native ARM or RISC-V support
- This work: add a portable OpenBLAS backend which supports x86-64, ARM, and RISC-V

CPU	EPYC 7742	A64FX	SG2042
SIMD	AVX2	SVE-512	RVV 0.7.1
Register width	256-bit	512-bit	128-bit
FP64	4	8	2

BLAS Backend: From oneMKL to OpenBLAS

- GPRat needs a sequential, vectorized BLAS/LAPACK backend
- Historical choice: Intel oneMKL but no native ARM or RISC-V support
- This work: add a portable OpenBLAS backend which supports x86-64, ARM, and RISC-V

CPU	EPYC 7742	A64FX	SG2042
SIMD	AVX2	SVE-512	N/A
Register width	256-bit	512-bit	64-bit
FP64	4	8	**1**

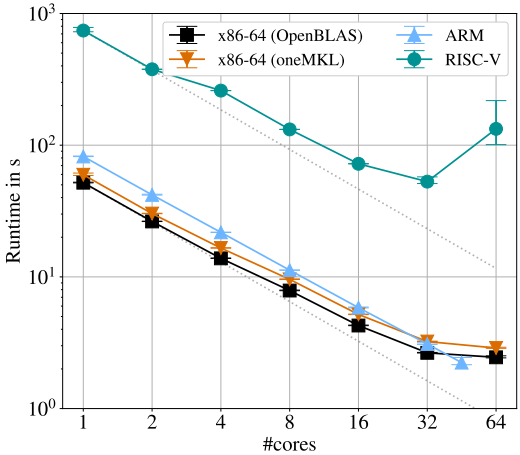

Results

Benchmark

Two benchmarks across x86-64, ARM, and RISC-V:

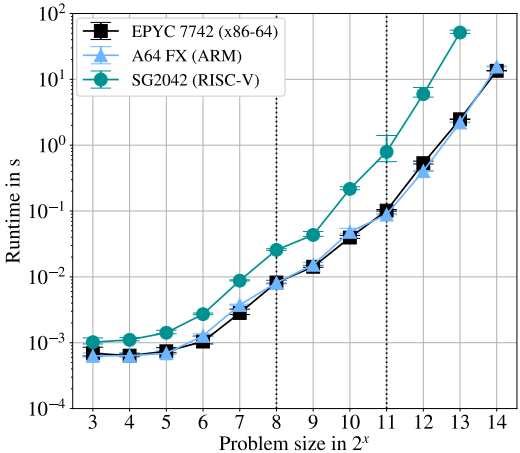
- Prediction with uncertainty (1)(2)
- FP64 (mostly) vectorized BLAS routines
- **Strong scaling**: $N = 2^{13}$ samples, 16 tiles per dimension
- **Problem size scaling**: N from 2^3 to 2^{14}, 1/4/16 tiles per dimension
- Runtimes averaged over 10 runs, 95% confidence intervals as error bars

Strong Scaling

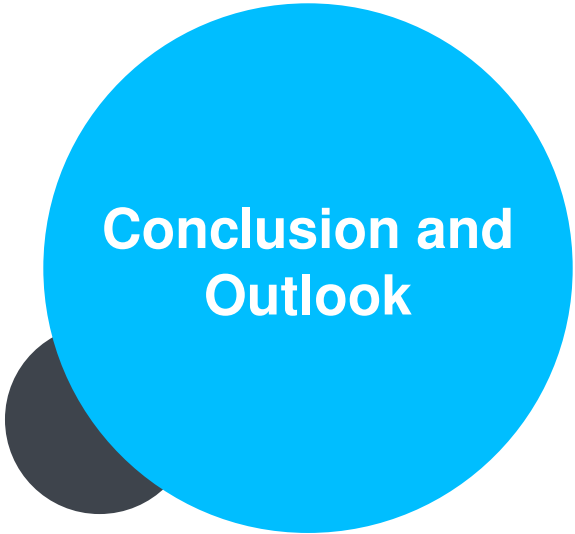


- x86-64: OpenBLAS 19% faster than oneMKL
- ARM: slower single-core, but 48 cores beat 64 Zen 2 cores by 9%
- RISC-V: more than 14× slower; parallel efficiency collapses to 9% at 64 cores

Problem Size Scaling



- EPYC 7742, A64FX, and SG2042 with 64, 48, and 32 cores respectively
- ARM and x86-64 remain competitive, staying within 23%
- RISC-V: performance gap $1.5\times$ to $3.1\times$ sequential, up to $24\times$ at scale


Conclusion and
Outlook

Conclusion

- Extended GPRat for portability across x86-64, ARM, and RISC-V
- Purpose-built ARM (A64FX) competes with contemporary x86-64
- RISC-V (SG2042) shows a significant performance gap especially when exercising the entire chip

Conclusion

- Extended GPRat for portability across x86-64, ARM, and RISC-V
- Purpose-built ARM (A64FX) competes with contemporary x86-64
- RISC-V (SG2042) shows a significant performance gap especially when exercising the entire chip

→ Future RISC-V CPUs targeting HPC need wide-register vectorization, a more mature software stack, and broader hardware availability.

Outlook

- Performance-per-watt across all three architectures
- Next-gen ARM and RISC-V and competing x86-64 designs
  - SOPHON SG2044: improved memory subsystem and RVV 1.0
  - NVIDIA GH200: superchip with 72 ARM cores
  - FUJITSU-MONAKA (2027): chiplet-based 144 ARM cores

University of Stuttgart
Germany

Thank You!

Alexander Strack
IPVS

Mail alexander.strack@ipvs.uni-stuttgart.de
Phone +49 711 685 88308
Internet https://www.alexanderstrack.com