**Project Lyra:** HPC APIs for LQCD

- Latent Quantum Chromodynamics (LQCD): latents physics
- a major fraction of HPC usage
- PRACE-6P Grant agreement ID: 823767, Project Lyra
- synergy of expertise and codes
  - numerical linear algebra, Inria Bordeaux-Paris
  - LQCD: The Cerny Institute
  - portable performance kernels: L2: LIBRRS (this poster)

**Numerical Techniques of Interest**

- iterative methods: block-Krylov
- require efficient Sparse Matrix-Matrix multiplication aka SpMM
- SpMM in matrix form:

```
\begin{align*}
\text{matrix} & = \begin{pmatrix}
0 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 0 \\
\end{pmatrix}
\end{align*}
```

More on the methods of interest:

- Van Teuk, Jerome / ScaLapack (Yahoo) (2007, 2014)
- Sparse matrices and applications
- parallel right-hand side.

**LIBRS: A Sparse BLAS Library**

- Recursive Sparse Blocks (RBS) layout
- operations for distributed-memory applications
  - e.g. shared-memory SpMM, triangular solve, ...
  - but also
  - operations for interactive applications:
    - e.g. matrix update, sparse-sparse sum, multiplication, convolutions
  - dual API
    - low-level interface in C/C++ and Fortran
    - Sparse BLAS (ENVI) standard
  - portable LGPA3 (flex standard)

**Project page:** [http://librrs.at.net](http://librrs.at.net)

---

**Figure 1:** Program using the new C = 2.2 interface to create a matrix, name it with respect to specific operation parameters, and multiply it by a vector. Here error handling is execution-based.

**One Library, many Interfaces**

- original rth-b
- machine-translation:
  - FORTRAN module, via script
  - Sparse BLAS header and module, via GNU Make
- **GNU OCTAVE** written in C++ using rth-b and librrs interface.

**Features access:**

- all C/C++
- more: Fortran
- many: PYTHON (OCTAVE-SparseRBS)
- hybrid: Sparse BLAS API

**Performance:**

- in C and FORTRAN, zero overhead
- optimal if used redundantly (block dot-vectorization) (also available on REACH and EU HPC)
- portable if used precompiled (e.g. #api install librrs octave-sparsemat)

**Python + Cython + LIBRSB = PyRBS**

- Cython
- **PyRBS** extension module to access LIBRSB transparently

**Printable: Figure 2:**

- RSB Layout and Algorithms
  - S=128MB of CSR = OctaveRBS = generated specialized kernels
  - intended for large matrices, where it favours:
    - cache locality
    - sparse thread parallelism
  - supports online empirical tuning

**Figure 3:** Instance of matrix representing the Wilson Dirac Operator. Used in Lattice QCD to simulate quarks. Square, 12x12 matrices, 7x7x7 nearest. Matrix: courtesy of Dr. Jacob Fislerbeer.

**Figure 5:** Same matrix. White auto-tuning-determined layout for SpMM with 2 RHS: 170 COO and 1330 CSR blocks. Using shared matrices (foreground MOS2, MOSRF) reducing averages to only 2.4 bytes per entry. Performed SpMM 40% faster than an initial “first guess” structure with 7712 blocks. Auto-tuning took less than a second. SpMM operation time slightly (12%) faster than CSR with ENVI. MKL: 2013.02.22

**Current Development Focus**

- performance improvement of SpMM kernels
- partners’ use cases, novel architectures (GPU and Isra methods)
- quality improvement of internals (bugfixes, C/C) and interfaces

---

**Figure 6:**

- Python + Cython + LIBRSB = PyRBS

- Cython
- **PyRBS** extension module to access LIBRSB transparently

---

**Figure 7:**

- rth matrix usage is edited after SpCrsr matrix generation.