Updated Proposal for a Next-Generation BLAS

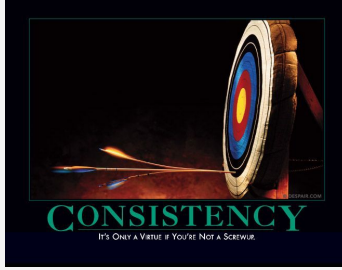
E. Jason Riedy[1], Greg Henry[2], James Demmel[3], Mark Gates[4], Xiaoye S. Li[5], Ping Tak P. Tang[2]

1: GT, 2: Intel, 3: UCB, 4: UTK, 5: LBNL

We invite feedback: https://tinyurl.com/yb7m7lov

Images from startrek.com and Wikipedia

Goal Reminder

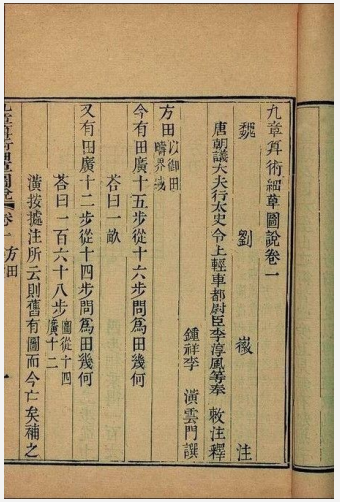


- XBLAS has 287 routines. Optimizing all of them is not a popular option.
  - Also missing necessities, *e.g.* parallel composition, lower precision.
- Establish a uniform naming convention at the library symbol level.
  - Low-level naming, others can provide high-level interfaces.
  - Mixed precision, extended precision, **reproducibility**, **batched**, fixed-point, …
  - **Ensure parallel routines can be built on top!**
- Enable more flexibility in data layout: Row strides, ...
- Provide clear platform optimization targets and minimal required routines.
  - Libraries need not support all possible names.
  - Many routines can be built on a few tuned microkernels as in BLIS.
- (Eventually) ***Provide a reference implementation. Help?***
  - Also, example higher level C++ & Fortran interfaces. *And C.*

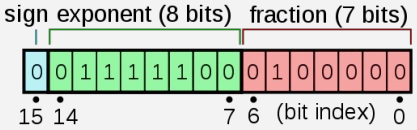
We invite feedback: https://tinyurl.com/yb7m7lov

Differences from Netlib and BLAS TF

- New names
  - Unified scheme for extended, reduced, and mixed precisions
  - Can be parsed and generated relatively easily
  - Simplification rules for common cases
- New API
  - Error reporting through a returned value
  - Well-defined, consistent propagation of exceptional values
  - Row strides to support interleaved and separate “extra” matrices (*e.g.* double-double tail)
- Identifies a minimal required set for known uses like LAPACK
- Explicitly intended to support new ideas predictably
  - Reproducible and batched BLAS alternatives
  - Fixed-point, flex-point, and other value formats



Nine Chapters on the Mathematical Art, circa 10th-2nd centuries BCE



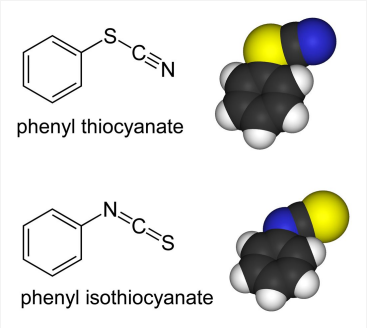
BFLOAT16 from Wikipedia

We invite feedback: https://tinyurl.com/yb7m7lov

Decisions: Nomenclature

Provide the low-level *naming* for high-level languages and dispatching.

```
BLAS_<blasFunction>_<typeSequence>[_<precisionLength>[<multiplier>]][_<suppl>](...parameter list...)
```

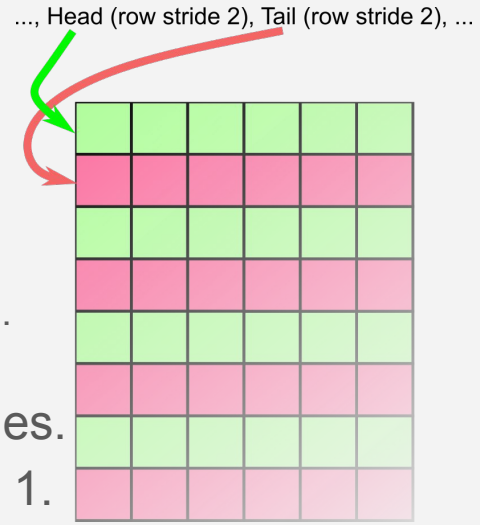


- Details in the proposal.
- Intention: Automatedly derive names from uses.
  - Very high-level interfaces can query optimized implementations.
  - Possible automatic generation of routines, *e.g.* `libxsmm`
  - Can support ***different*** reproducibility implementations
  - Downside could be interface explosion, but query+generation could be a route for systems like Julia. Could we support batched differences also?
- Intermediates and scalars for mixed precision: Always the max.

We invite feedback: https://tinyurl.com/yb7m7lov

Image from Wikipedia

Decisions: Storage and Strides



- Supporting double-double is complex (*cough*).
  - Some applications will want to drop the tail, so stick it in workspace.
  - Others want everything interleaved.
- Applying BLAS to general N-D arrays requires more copies.
- “Solution:” Support row strides everywhere, not just level 1.
  - Double-double still passed as two arrays, but row stride of 2 can interleave as if one.
  - Many tensor slices now are possible.
  - Maybe unify Matlab complex and sane complex?
- Arguments in the low-level API now include more matrices, column strides, and row strides. Could be painful for everyday use.
  - Somewhat alleviated by higher-level interfaces, **even in C**.
  - But the flexibility could be very useful for interfaces like Numpy or under Julia.

We invite feedback: https://tinyurl.com/yb7m7lov

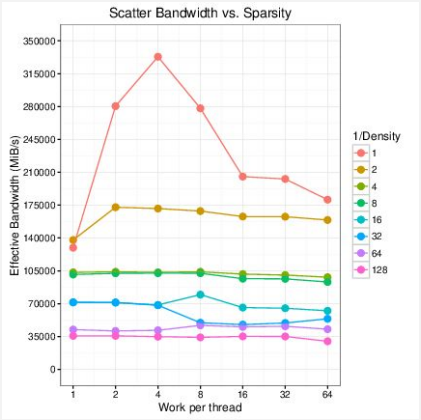
Decisions: Error/Exception Checking and Reporting

- Keep checking dimensional consistency.
  - Removing the checks only helps tiny matrices, better methods exist.
- Cast XERBLA in Game of Thrones (aka get rid of it).
  - Return an error code universally, affecting ddot.
  - High-end implementations (*e.g.* NAG, MKL) can provide intercept layers for trapping.
    - Parse-able names plus flexible object models (ELF): Dynamic intercepts.
- More difficult: Exceptional values in data.
  - Not operating on zeros can improve performance, but $0 \times \infty = \text{NaN}$.
  - Vendors' customers request such consistency indirectly via "bug" reports.
  - *Agreement:* Zero scalars elide the matrices, but zero entries do not skip operations.
  - Even without a new interface, existing routines (ISAMAX) need fixing.
  - *See slides from Monday's Correctness 2018 workshop: James Demmel, "Correctness of Floating Point Programs - Exception Handling and Reproducibility"*

We invite feedback: https://tinyurl.com/yb7m7lov

Where Are We Now?

- We could use assistance in implementation.
  - Many of us are in research, not development…
  - Patrick Lavin (GT) working on gather-compute-scatter architectural characterization.
- Most feedback: “I’m glad someone is thinking about this.”
  - We need to move past the thinking stage.
  - We are very thankful for the detailed feedback although slow to respond.
- Other major goal: **Finding the right *ways* forward.**
  - There are as many ways as there are scientific problems to address.
  - Hence: Naming and semantics. The rest will follow use.
- Extended abstract: https://tinyurl.com/yb7m7lov
- Proposal: https://tinyurl.com/y9dz8kmy



https://spatter.io

And then there are “funny” architectures: FPGAs, memory-centric, analog, neuromorphic, ...

We invite feedback: https://tinyurl.com/yb7m7lov

WIP image from Wikipedia