

Faster Code.... Faster

Intel® Math Kernel Library

aka

Intel® MKL

BLAST (from the past) Forum MINUTES

November 7-8, 1996

Cray Research, Eagan, MN

Shane Story of Intel Corporation spoke of the math library for Intel MP node machines provided by their Independent Software Vendor (ISV) -- Kuck and Associates. Basically, Intel **does not support a math libraries group anymore**, and instead hires third-party vendors to write software for their machines. They plan to tune existing libraries for their machines, and **not provide any additional functionality**.

Powered by Intel® MKL



Energy



Science &
Research



Engineering
Design



Financial
Analytics



Signal
Processing



Digital
Content
Creation

- Speeds math processing in scientific, engineering and financial applications
- Functionality for dense and sparse linear algebra (BLAS, LAPACK, PARDISO), FFTs, vector math, summary statistics and more
- Provides scientific programmers and domain scientists
 - Interfaces to de-facto standard APIs from C++, Fortran, C#, Python and more
 - Support for Linux*, Windows* and OS X* operating systems
 - Extract great performance with **minimal effort**
- Unleash the performance of Intel® Core, Intel® Xeon and Intel® Xeon Phi™ product families
 - Optimized for single core vectorization and cache utilization
 - Coupled with automatic parallelism for multi-core, manycore and coprocessors
 - Scales to PetaFlop (10^{15} floating-point operations/second) clusters and beyond
- Included in Intel® Parallel Studio XE, Intel® System Studio Suites and a variety of other new products

Optimized Mathematical Building Blocks

Intel MKL

Linear Algebra

- BLAS
- LAPACK
 - ScaLAPACK
 - Sparse BLAS
 - Sparse Solvers
 - Iterative
 - PARDISO* SMP & Cluster

Fast Fourier Transforms

- Multidimensional
- FFTW interfaces
- Cluster FFT

Vector Math

- Trigonometric
- Hyperbolic
- Exponential
- Log
- Power
- Root

Vector RNGs

- Congruential
- Wichmann-Hill
- Mersenne Twister
- Sobol
- Neiderreiter
- Non-deterministic

Summary Statistics

- Kurtosis
- Variation coefficient
- Order statistics
- Min/max
- Variance-covariance

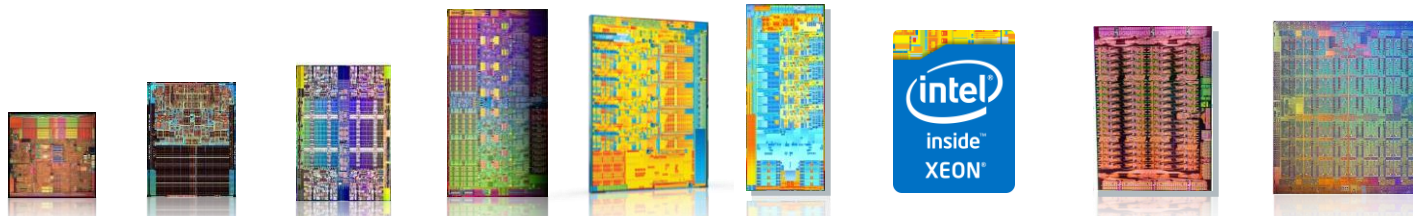
And More

- Splines
- Interpolation
- Trust Region
- Fast Poisson Solver

Automatic Dispatching to Tuned ISA-specific Code Paths

Intel® MKL

More cores → More Threads → Wider vectors



	Intel® Xeon® Processor 64-bit	Intel® Xeon® Processor 5100 series	Intel® Xeon® Processor 5500 series	Intel® Xeon® Processor 5600 series	Intel® Xeon® Processor E5-2600 v2 series	Intel® Xeon® Processor E5-2600 v3 series v4 series	Future Intel® Xeon® Processor ¹	Intel® Xeon Phi™ x100 Coprocessor (KNC)	Intel® Xeon Phi™ x200 Processor & Coprocessor (KNL)	Future Intel® Xeon Phi™ (KNH)
Up to Core(s)	1	2	4	6	12	18-22	TBD	61	72	TBD
Up to Threads	2	2	8	12	24	36-44	TBD	244	288	TBD
SIMD Width	128	128	128	128	256	256	512	512	512	TBD
Vector ISA	Intel® SSE3	Intel® SSE3	Intel® SSE4- 4.1	Intel® SSE 4.2	Intel® AVX	Intel® AVX2	Intel® AVX-512	IMCI 512	Intel® AVX-512	TBD

Product specification for launched and shipped products available on ark.intel.com. 1. Not launched or in planning.
All dates and products specified are for planning purposes only and are subject to change without notice

Intel MKL - 22 Years of Features and Performance

Year	Intel MKL Release	Processor	ISA	Features
1994	Intel® BLAS Library for Pentium Processor	Pentium	x87	BLAS
		Pentium II		
1996	Intel MKL 1.0			BLAS 3 Threaded
	2.0		MMX	2D FFTs
1998	2.1			Sparse BLAS 1
	3.0	Pentium III	Intel® SSE	LAPACK
2000	4.0			Vector Math
	5.0	Pentium 4	Intel® SSE2	
2002	6.0	Itanium®		DFTI & Vector Statistics
	7.0			PARDISO* & ScaLAPACK
2004	7.1	EM64T (Prescott)	Intel® SSE3	
	8.0	Nacona		Sparse L2/L3 BLAS & F95
2006	9.0	Merom	Intel® SSSE3	Trig Transforms & Poisson Solver
	9.1	Penryn	Intel® SSE4.1	Trust Region & Linpack Benchmark
2008	10.0/10.1			Out-of-core PARDISO*
	10.2	Xeon E5*(Nehalem)	Intel® SSE4.2	LAPACK 3.2
2010	10.3			LAPACKE & Data Fitting
		Xeon E5*(Westmere)		
2012	11.0	E5 (Sandy Bridge) & Intel Xeon® Phi (KNC)	Intel® AVX	Reproducibility & Extended Eigensolver & Automatic Offload
	11.1	E5 V2 (Ivybridge)		
2014	2015/11.2	E5 V3 (Haswell)	Intel® AVX2	Cluster Direct Sparse Solver & LAPACK 3.5
	2016/11.3			Community Licenses & HPCG & Enhanced Sparse & Batched BLAS
2016	2017	E5 V4 (Broadwell) & Intel Xeon® Phi (KNL)	Intel® AVX 512	MKL DNN

Latest version of Intel MKL runs on these processors

Using and Linking Intel MKL

With the Intel Fortran or
C/C++ compiler on Linux*,
simply

```
ifort prog.f -mkl[:lib]
```

or

```
icl prog.c -mkl[:lib]
```

Linking with other compilers requires additional
steps ...

Intel® Math Kernel Library Link Line Advisor

July 20, 2012

[Translate](#)

[f Share](#)

[t Tweet](#)

[g+ Share](#)

Introduction

The Intel® Math Kernel Library (Intel® MKL) is designed to run on multiple processors and operating systems. It is also compatible with several compilers and third party libraries, and provides different interfaces to the functionality. To support these different environments, tools and interfaces Intel MKL provides multiple libraries from which to choose.

To see what libraries are recommended for a particular use case, specify the parameters in the drop down lists below.

Intel® Math Kernel Library (Intel® MKL) Link Line Advisor v4.6

[Reset](#)

Select Intel® product:	Intel(R) MKL 11.3.2
Select OS:	Linux*
Select usage model of Intel® Xeon Phi™ Coprocessor:	Native
Select compiler:	Intel(R) C/C++
Select architecture:	Intel(R) 64
Select dynamic or static linking:	Dynamic
Select interface layer:	LP64 (32-bit integer)
Select threading layer:	OpenMP threading
Select OpenMP library:	Intel(R) (libiomp5)
Select cluster library:	<input type="checkbox"/> Cluster PARDISO (BLACS required) <input type="checkbox"/> CDFT (BLACS required) <input type="checkbox"/> ScaLAPACK (BLACS required) <input type="checkbox"/> BLACS

<http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor/>

TAKE FLIGHT LET YOUR CODE SOAR

Get Community Licensing >



Unbelievable
...
Intel® MKL
for FREE!



THE NEST

Get the whole flock. Modernize your code quickly and easily with this powerful collection of high performance libraries:

- Intel® Threading Building Blocks
- Intel® Integrated Performance Primitives
- Intel® Math Kernel Library
- Intel® Data Analytics Acceleration Library

Get access to the latest versions with no up-front cost. Intel® Premier Support and access to previous library versions can be purchased separately as needed.

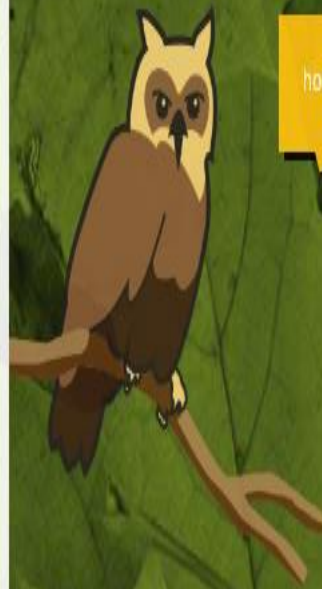
Licenses:

Community >

Teach/Research >

Evaluation >

Full Support >



hoot.

INTEL® MATH KERNEL LIBRARY

Accelerate application performance and reduce development time with the fastest and most used math library for Intel® and compatible processors. Harness the power of today's processors—with increasing core counts, wider vector units, and more varied architectures. Includes highly vectorized and threaded linear algebra, fast Fourier Transforms, vector math, and statistics functions. Through a single API call, these functions automatically scale for future processor architectures by selecting the best code path for each.

Licenses:

Community >

Teach/Research >

Evaluation >

Full Support >

<https://software.intel.com/sites/campaigns/nest/>

Notable Intel® MKL 11.0-11.3 (2013-2016) Enhancements

Intel® MKL

■ *Optimizations*

- Always tuned for the latest Intel® Xeon® processors & Intel® Xeon Phi™ Processors/Coprocessors

■ Features

- Conditional Numerical Reproducibility (CNR)
- Extended Eigensolvers based on and compatible with FEAST¹
- Parallel Direct Sparse Solver for Clusters
- Small Matrix Multiply enhancements (Direct Call)
- Intel® Optimized High Performance Conjugate Gradient (HPCG) Benchmark
- Batch GEMM
- GEMMT functions calculating U/L triangular part of $C = \alpha * A * B + \beta * C$
- Intel Threading Building Blocks (TBB) composability
- Sparse BLAS inspector-executor API
- Extended cluster support (MPI wrappers and OS X)

¹ <http://www.ecs.umass.edu/~polizzi/feast/>

Conditional Numerical Reproducibility (CNR)

Intel® MKL

What causes a variation in results?

- With floating-point numbers, the order of computation matters!
- Associativity does not always hold ... **$(a+b)+c \neq a+(b+c)$**

$$2^{-63} + 1 + -1 = 2^{-63}$$

(infinitely precise result)

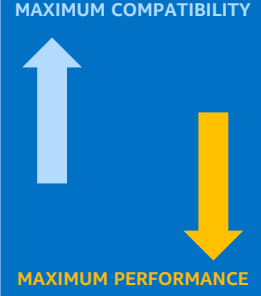
$$(2^{-63} + 1) + -1 = 0$$

(correct IEEE double precision result)

$$2^{-63} + (1 + -1) = 2^{-63}$$

(correct IEEE double precision result)

CNR run-time controls

 MAXIMUM COMPATIBILITY MAXIMUM PERFORMANCE	For consistent results ...	Function Call: <code>mkl_cbwr_set(...)</code>	Env. Variable: <code>MKL_CBWR =</code>
	on Intel or Intel-compatible CPUs supporting SSE2 instructions or later	<code>MKL_CBWR_COMPATIBLE</code>	<code>COMPATIBLE</code>
	on Intel processors supporting SSE4.2 instructions or later	<code>MKL_CBWR_SSE4_2</code>	<code>SSE4_2</code>
	on Intel processors supporting Intel® AVX or later	<code>MKL_CBWR_AVX</code>	<code>AVX</code>
	on Intel processors supporting Intel® AVX2 or later	<code>MKL_CBWR_AVX2</code>	<code>AVX2</code>
	on Intel processors supporting Intel® AVX512 or later	<code>MKL_CBWR_AVX512</code>	<code>AVX512</code>
	from run-to-run (but not processor-to-processor)	<code>MKL_CBWR_AUTO</code>	<code>AUTO</code>

Intel® MKL 2017 - SGEMM for packed A and/or B

$C = \alpha * \text{op}(A) * \text{op}(B) + \beta * C$, with $\text{op} = \text{N}, \text{T}$ or $\text{P}(\text{packed})$

float* mat = sgemm_alloc(char *identifier, int *m, int *n, int *k)

- Description: allocate storage for the packed matrix
- *identifier = 'A' or 'a', allocate storage to pack the A matrix
- *identifier = 'B' or 'b', allocate storage to pack the B matrix

void sgemm_pack(char *identifier, char *trans, int *m, int *n, int *k, float *alpha, float *src, int *ld, float *dest)

- Description: pack the matrix, scale by alpha
- *identifier = 'A' or 'a', src is the A matrix, scaled by alpha and packed into dest
- *identifier = 'B' or 'b', src is the B matrix, scaled by alpha and packed into dest

void sgemm_compute(char *transa, char *transb, int *m, int *n, int *k, float *A, int *lda, float *B, int *ldb, float *beta, float *C, int *ldc)

- Description: compute sgemm where one of A or B is stored in a packed data structure
- *transa = 'P' or 'p', matrix A packed and lda ignored
- *transa = 'N' or 'n' or 'T' or 't', matrix A meets sgemm requirements
- *transb = 'P' or 'p', matrix B packed and ldb ignored
- *transb = 'N' or 'n' or 'T' or 't', matrix B meets sgemm requirements

void sgemm_free(float* mat) - free the storage previously allocated for the packed matrix

- Description: free the storage previously allocated for the packed matrix

IGEMM

$IC = ialpha * op(IA) * op(IB) + ibeta * IC$, with $op=N,T$



How do I request Intel MKL features?

<https://software.intel.com/en-us/forums/intel-math-kernel-library>

Intel® Developer Zone:

Intel® Math Kernel Library



Announcements

This is the place to ask questions of and share information with other users of the Intel® Math Kernel Library (Intel MKL), a software library for Windows*, Linux* and Mac OS* X. Other useful links: [Support Resources](#), [Product Information](#) and [Documentation](#).

Please do not include private information such as your email address or product serial number in your posts.

[Login](#) to post new content in forum.

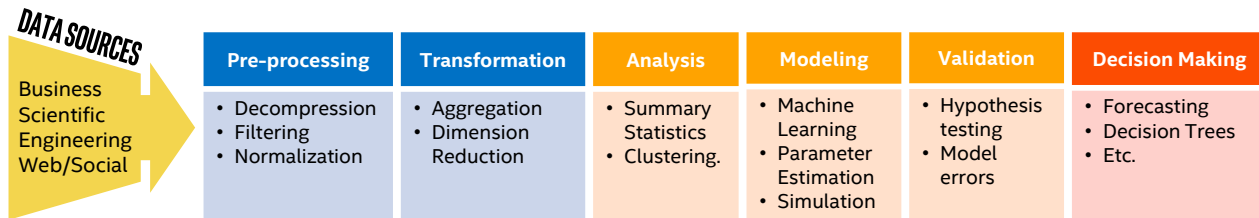
- Forum Tools -

1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | [next >](#) | [last >>](#)

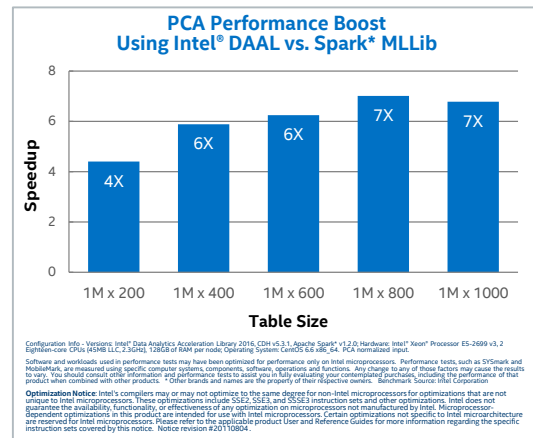
	Topic / Topic starter	Post date	Replies	Last Post
	Sticky: Intel® Math Kernel Library 11.3 Update 3 is now available by Gennady Fedorov...	Mon, 05/02/2016 - 05:51	0	by Gennady Fedorov... Mon, 05/02/2016 - 05:51
	Sticky: Intel® Parallel Studio XE 2017 Beta invitation – please register and provide feedback! by Sridevi (Intel)	Wed, 03/30/2016 - 13:33	1	by steve o. Sat, 04/30/2016 - 07:40
	Sticky: Deep Neural Network extensions for Intel MKL by Gennady Fedorov...	Tue, 12/22/2015 - 10:15	0	by Gennady Fedorov... Tue, 12/22/2015 - 10:15
	Sticky: Calling Python Developers - High performance Python powered by Intel MKL is here! by Zhang Z. (Intel)	Thu, 11/12/2015 - 14:16	0	by Zhang Z. (Intel) Thu, 11/12/2015 - 14:16
	Sticky: Intel Math Kernel Library for Free – requires registration, includes forum support, and permits royalty-free use by Zhang Z. (Intel)	Tue, 09/01/2015 - 10:45	1	by Tim S. Wed, 09/02/2015 - 10:50
	vdExp 32 bit vs 64 bit different results by Mark L.	Wed, 05/18/2016 - 04:34	4	by Jingwei Z. (Intel) Wed, 05/18/2016 - 13:49

Turn Big Data Into Information Faster with Intel® Data Analytics Acceleration Library

- Advanced analytics algorithms supporting all stages of data analysis



- Simple object-oriented APIs for C++ & Java
- Easy connections to
 - Popular analytics platforms (Hadoop, Spark)
 - Data sources (SQL, non-SQL, files, in-memory)



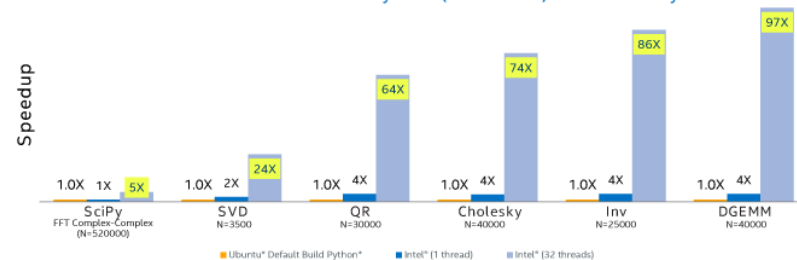
Boost NumPy/SciPy performance with Intel® MKL

Intel® Distribution for Python*

Easy Access to High Performance Python

- NumPy/SciPy/Scikit-Learn/Pandas accelerated with Intel® MKL
 - Close to 100X performance speedups on select functions
 - Includes Python optimized modules for Intel® Threading Building Blocks (Intel® TBB), Intel® Data Analytics Acceleration Library (Intel® DAAL)
- Integrated Distribution, Out-of-the-Box access to performance
- Latest Optimizations for Intel® Xeon and Intel® Xeon Phi™ Processors
- Available as free standalone, via conda* and Intel® Parallel Studio XE 2017

Python* Performance Boost on Select Numerical Functions
with Intel® Distribution for Python (2017 beta) vs. Ubuntu Python



Configuration info: - Versions: Intel® Distribution for Python 2.7.11 2017, Beta (Mar 08, 2016), Ubuntu* built Python* Python 2.7.11, NumPy 1.10.4, SciPy 0.17.0 built with gcc 4.8.4; Hardware: Intel® Xeon® CPU E5-2698 v3 @ 2.30GHz (2 sockets, 16 cores each, HT=OFF), 64 GB of RAM, 8 DIMMs of 8GB@2133MHz; Operating System: Ubuntu 14.04 LTS; MKL version 11.3.2 for Intel Distribution for Python 2017, Beta

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSE4.2 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804.

"I expected Intel's numpy to be fast but it is significant that plain old python code is much faster with the Intel version too."

- Dr. Donald Kinghorn, Puget Systems
Public [review](#)

Download beta at <https://software.intel.com/en-us/python-distribution>

A Very Good Kitty, Indeed

DreamWorks Animation's *Puss in Boots*
Uses Intel® Math Kernel Library to Help
Create Dazzling Special Effects

BY GARRET ROMAINE



Intel[®] MKL Resources

Intel[®] MKL website

- <https://software.intel.com/en-us/intel-mkl>

Intel MKL forum

- <https://software.intel.com/en-us/forums/intel-math-kernel-library>

Intel[®] MKL benchmarks

- <https://software.intel.com/en-us/intel-mkl/benchmarks#>

Intel[®] MKL link line advisor

- <http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor/>

Backup

Intel® Xeon Phi™ Coprocessor Support

Intel® MKL

Automatic Offload

- No code changes required
- Automatically uses both host and target
- Transparent data transfer and execution management

Compiler Assisted Offload

- Explicit controls for data transfer and remote execution using compiler pragmas
- Can be used together with Automatic Offload

Native Execution

- Uses the coprocessors as independent nodes
- Input data and binaries copied to targets in advance



Using Automatic Offload

- Set **MKL_MIC_ENABLE** env. variable
- Ensure **MIC_LD_LIBRARY_PATH** contains `<mkl directory>/lib/mic`
- Link and run as you would normally

?GEMMT Functionality

Intel® MKL

Calculates $C = \alpha * \text{op}(A) * \text{op}(B) + \beta * C$, updating only upper or lower triangular part of C

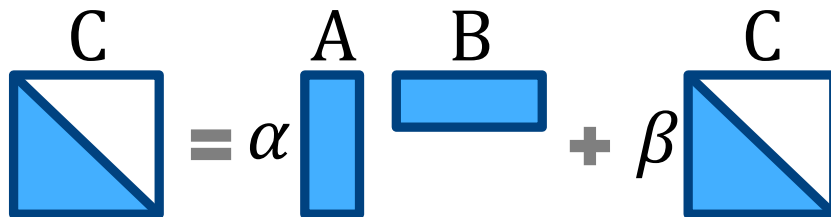
`?gemmt(uplo, transa, transb, n, k, alpha, a, lda, b, ldb, beta, c, ldc)`

Naming

- S/D/C/Z: supported precisions
- GE: general matrix
- MM: matrix-matrix Product
- T: triangular part of result matrix updated

Parameters

- uplo: specifies whether upper or lower triangular part of the array c is used
- transa, transb: specifies form of $\text{op}(A, B)$ used in the matrix multiplication
- n, k: sizes of the input matrices
- lda, ldb, ldc: leading dimensions of the input matrices
- a, b: arrays containing matrices to be multiplied
- c: array containing triangular part of result matrix



Inspector-Executor Sparse BLAS API

Intel® MKL

Inspect step – analyze matrix to choose best strategy

- Computational kernels for portrait
- Balancing strategy for parallel execution

Execute step – use analysis data to get better performance

- Optimization applied to get better performance
- Level chosen based on expected number of iterations

Compared to existing implementation new API provides

- Parallel triangular solver
- Improved sparse matrix by sparse matrix multiplication
- Both 0-based and 1-based indexing, row-major and column-major ordering
- Extended BSR support

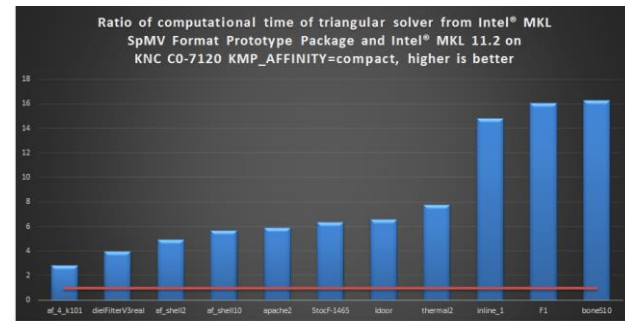
```
mkl_sparse_create_d_csr(&A, SPARSE_INDEX_BASE_ZERO,
                        rows, cols, rowsStart, rowsEnd, colIndx, values );

mkl_sparse_set_mv_hint(A,
                      SPARSE_OPERATION_NON_TRANSPOSE,
                      SPARSE_FULL,
                      n_iter);

mkl_sparse_set_memory_hint(A, SPARSE_MEMORY_AGRESSIVE);
mkl_sparse_optimize(A);

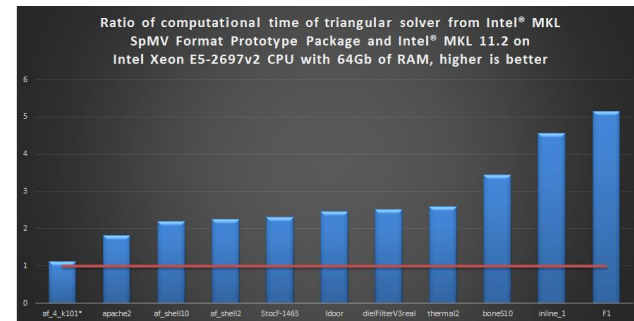
for (int i=0;i<n_iter;i++) {
    mkl_sparse_d_mv(SPARSE_OPERATION_NON_TRANSPOSE,
                  alpha, A, SPARSE_FULL, x, beta, y);
    ...
}

mkl_sparse_destroy(A);
```



Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, refer to <http://www.intel.com/content/www/us/en/benchmarks/resources/benchmark-limitations.html>. Refer to our Optimization Notice for more information regarding performance and optimization choices in Intel software products at: <http://software.intel.com/en-us/articles/optimization-notice>.

*Other brands and names are the property of their respective owners.



Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, refer to <http://www.intel.com/content/www/us/en/benchmarks/resources/benchmark-limitations.html>. Refer to our Optimization Notice for more information regarding performance and optimization choices in Intel software products at: <http://software.intel.com/en-us/articles/optimization-notice>.

*Other brands and names are the property of their respective owners.

How is Intel® MKL Parallelized

Domain	SIMD	Open MP	MPI
BLAS 1, 2, 3	X	X	
FFTs	X	X	
LAPACK (dense LA solvers)	X (relies on BLAS 3)	X	
ScaLAPACK (cluster dense LA solvers)		X (hybrid)	X
PARDISO (sparse solver)	X (relies on BLAS 3)	X	X
VML/VSL	X	X	
Cluster FFT		X	X

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Copyright © 2016, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

