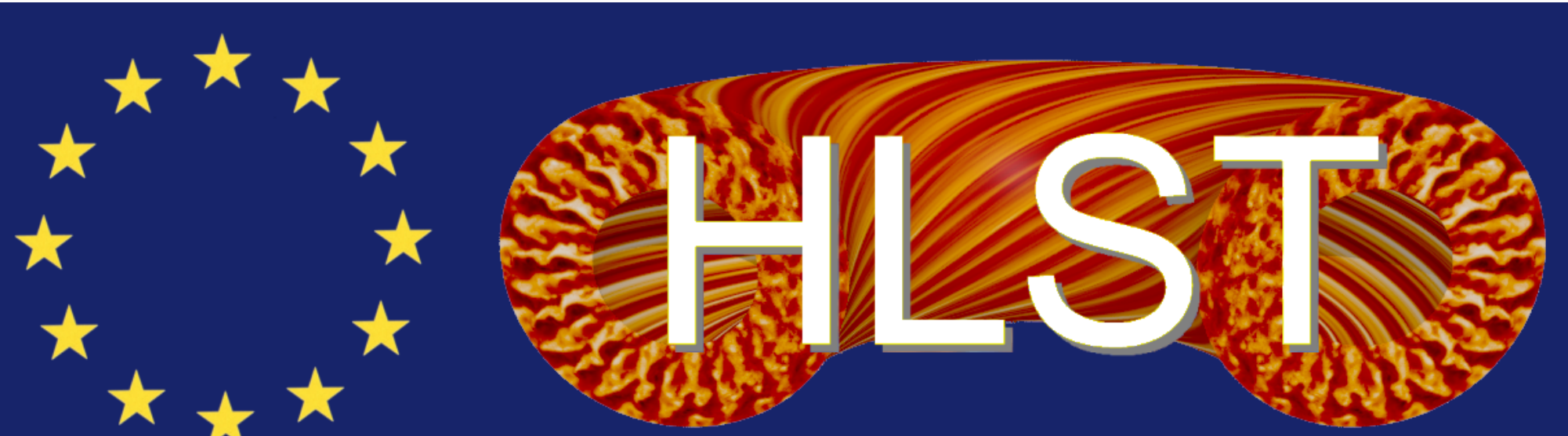
Performance Tuning Using Vectorization

Nicolay J. Hammer

GOTiT e-Seminar, Feb. 22, 2010



Overview

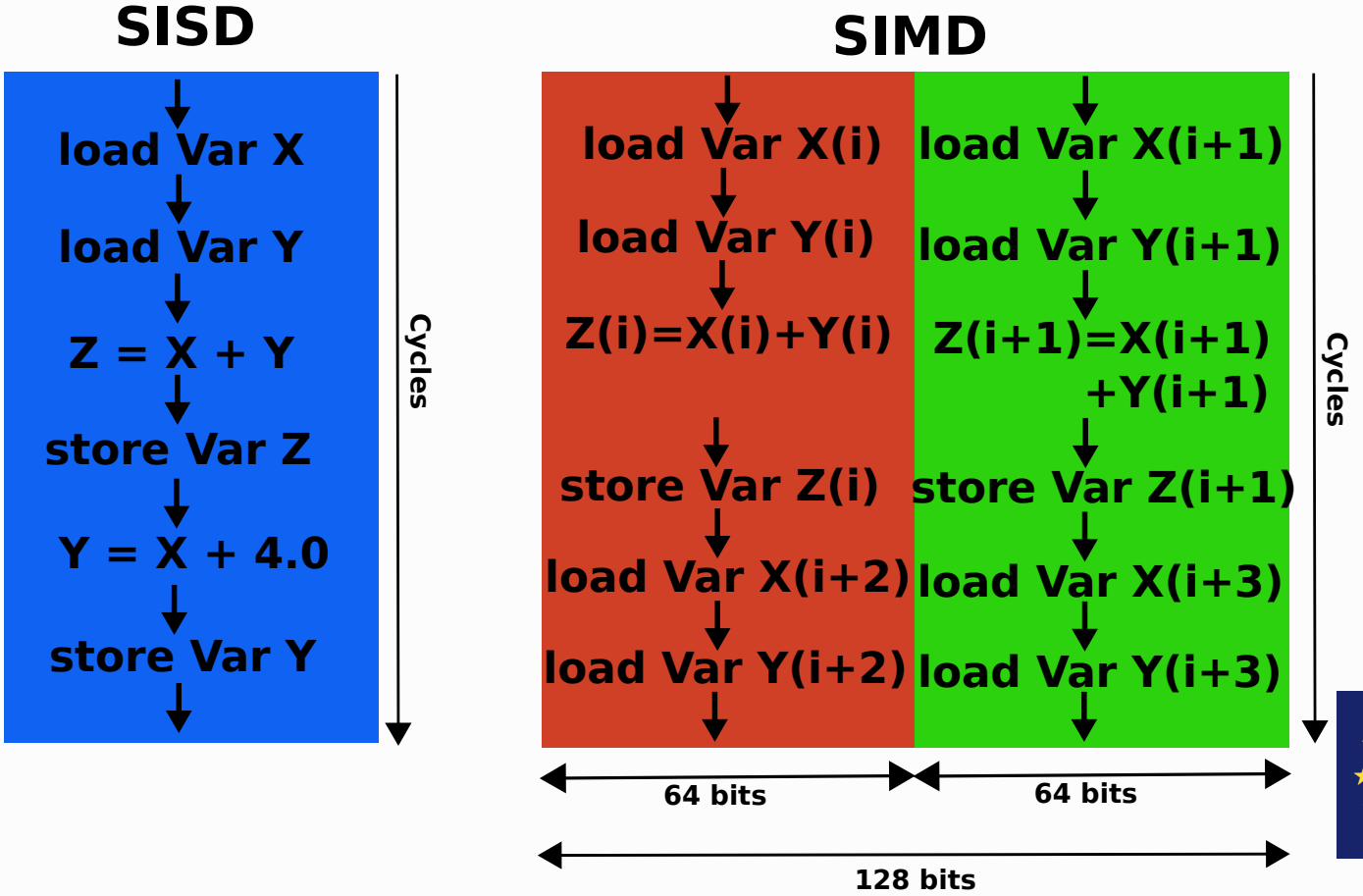
- Introduction
  - Flynn's Taxonomy & SIMD
  - Platform Setup
  - Vectorization Test Code
- Measurements
  - Basic Structure
  - Function Examples
- Problems and Issues
- Summary of the Study

Flynn's Taxonomy & SIMD/Vector Pipeline

- Single Instruction, Single Data stream (SISD)
- Single Instruction, Multiple Data streams (SIMD)
- Multiple Instruction, Single Data streams (MISD)
- Multiple Instruction, Multiple Data streams (MIMD)

Flynn's Taxonomy & SIMD/Vector Pipeline

- Single Instruction, Single Data stream (SISD)
- Single Instruction, Multiple Data streams (SIMD)
- Multiple Instruction, Single Data streams (MISD)
- Multiple Instruction, Multiple Data streams (MIMD)



Platform Setup

- HPC-FF — Intel Platform
  - Intel Xeon X5570 (Nehalem) CPUs (2.93 GHz)
  - Intel `ifort` compiler
  - Intel MKL and IPP performance libs.
- VIP — IBM Platform
  - IBM POWER6 CPUs (4.7 GHz)
  - IBM XLF compiler
  - IBM MASS performance lib.

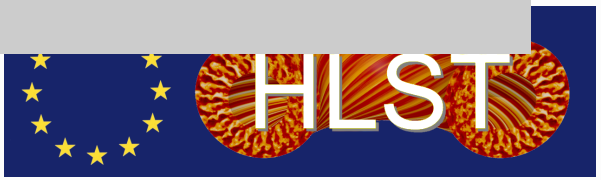
Vectorization Test - Source Code

```
CALL PERFON ('autovec')

DO nb = 0, nBlocks - 1

   nOfs   = nb * nChunk
   nSize  = MIN( nmax - nOfs , nChunk )
   nStart = nOfs + 1
   nStop  = nOfs + nSize

   vec1(1:nSize) = rearr1(nStart:nStop)

   !DEC$ VECTOR ALWAYS
   DO n = 1, nSize
      vec2(n) = EXP( vec1(n) )
   ENDDO

ENDDO ! nb = 0, nBlocks - 1

CALL PERFOFF
```

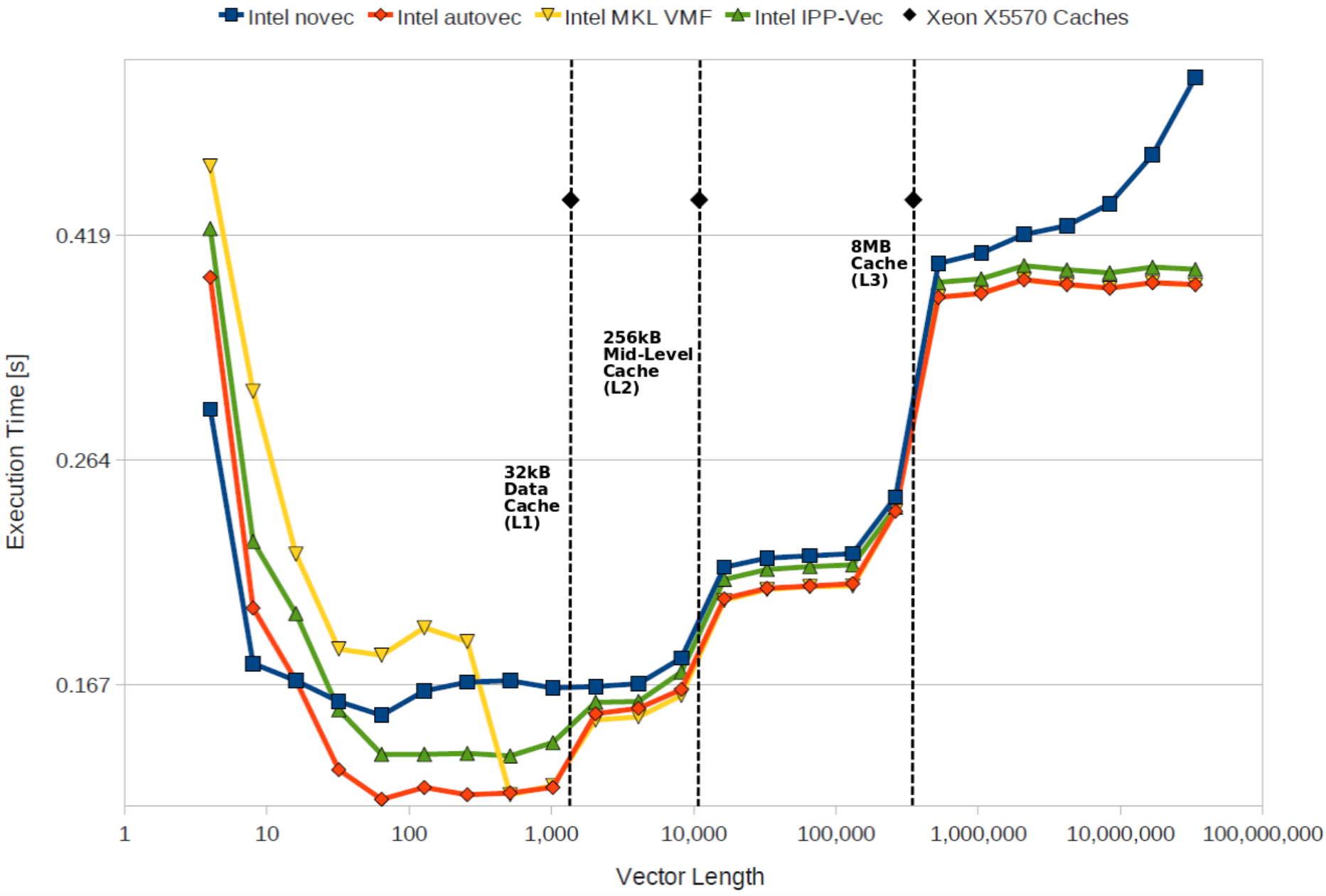
Vectorization Test - Source Code

```
CALL PERFON ('autovec')

DO nb = 0, nBlocks - 1

   nOfs   = nb * nChunk
   nSize  = MIN( nmax - nOfs , nChunk )
   nStart = nOfs + 1
   nStop  = nOfs + nSize

   vec1(1:nSize) = rearr1

   !DEC$ VECTOR ALWAYS
   DO n = 1, nSize
      vec2(n) = EXP( vec1
   ENDDO

ENDDO ! nb = 0, nBlocks -

CALL PERFOFF
```

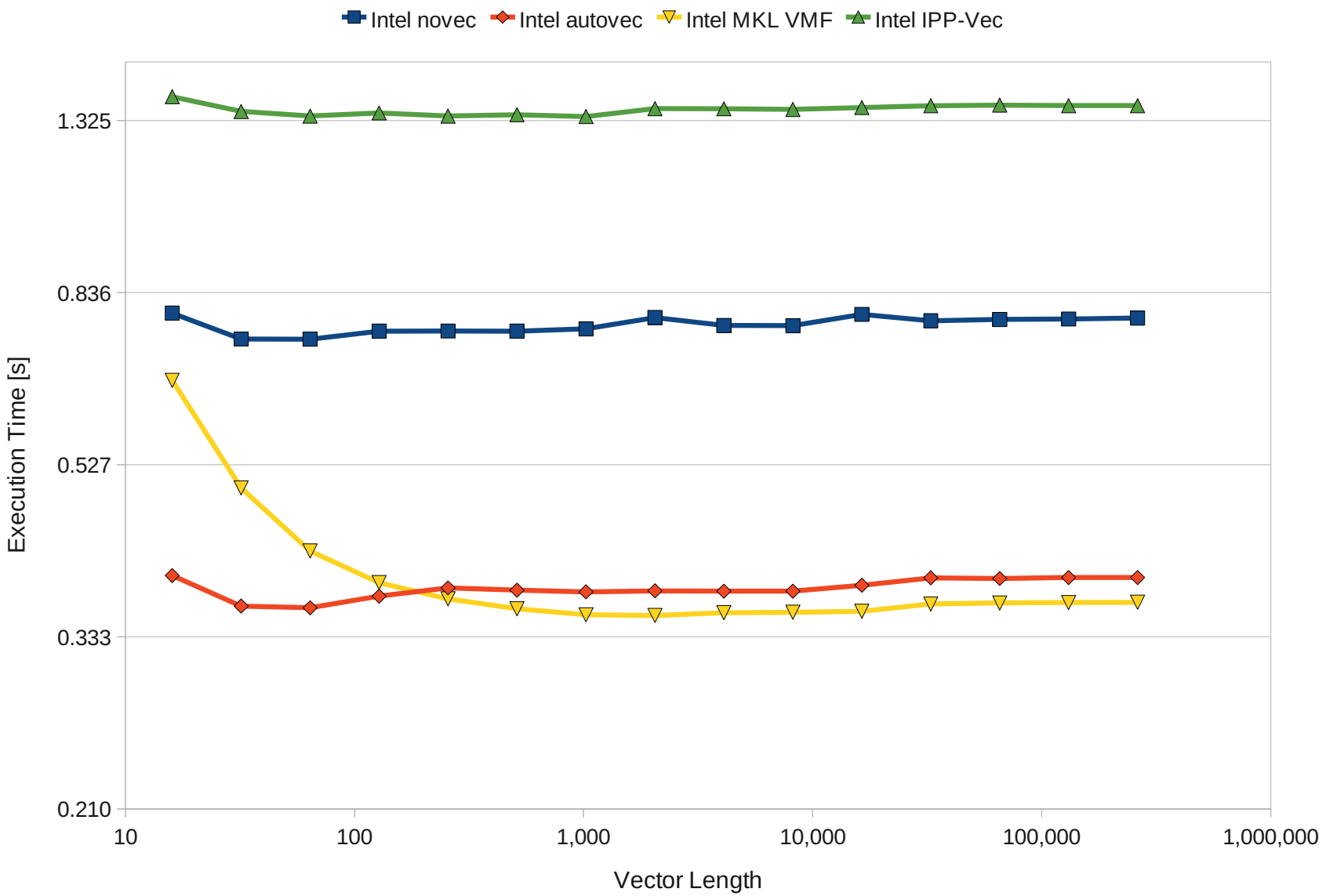
```
CALL PERFON ('mkl-vmf')

DO nb = 0, nBlocks - 1

   nOfs   = nb * nChunk
   nSize  = MIN( nmax - nOfs , nChunk )
   nStart = nOfs + 1
   nStop  = nOfs + nSize

   vec1(1:nSize) = rearr1(nStart:nStop)

   CALL VDEXP( nSize, vec1(1:nSize), vec2(1:nSize) )

ENDDO ! nb = 0, nBlocks - 1

CALL PERFOFF
```



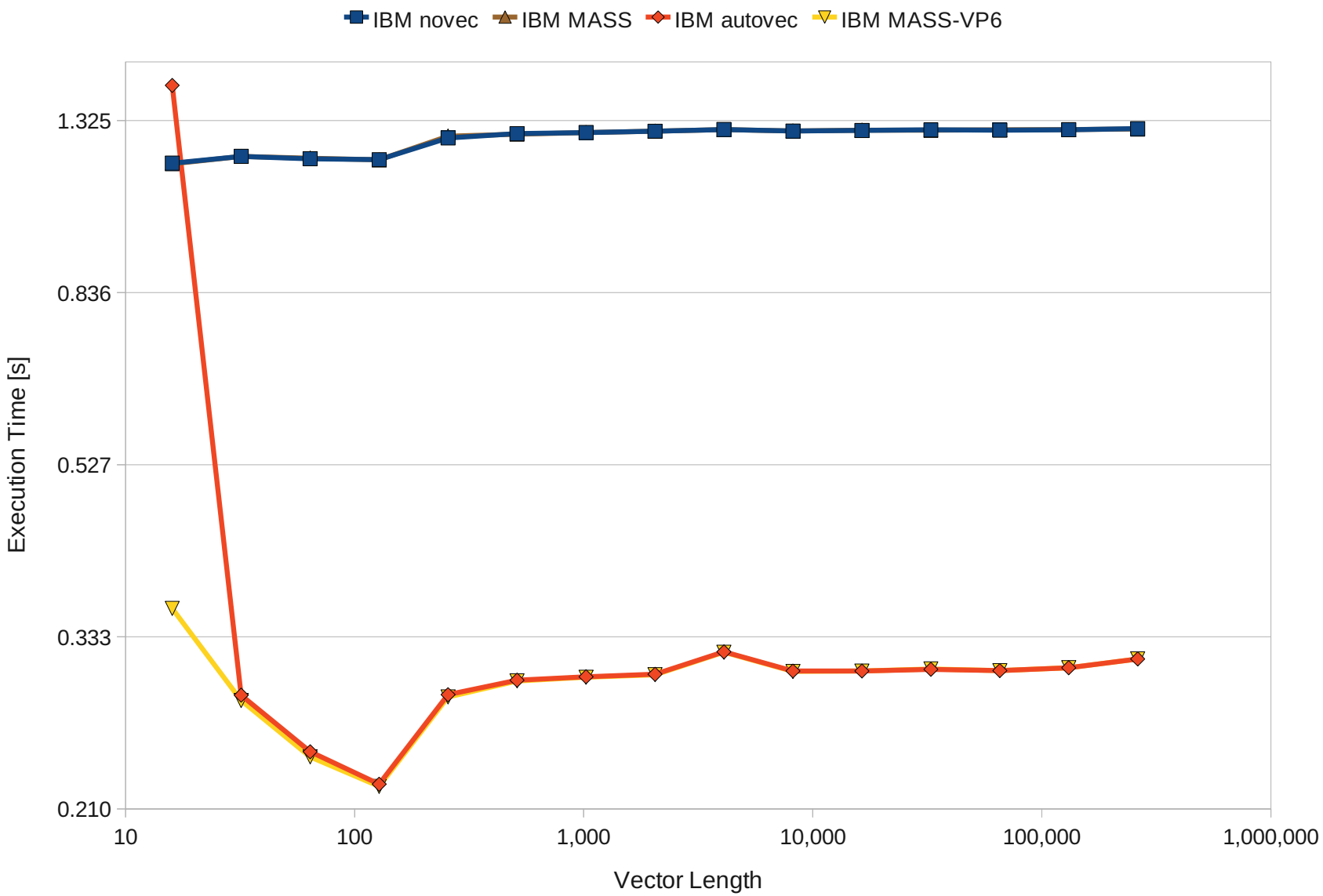
Basic Structure of the Measurements



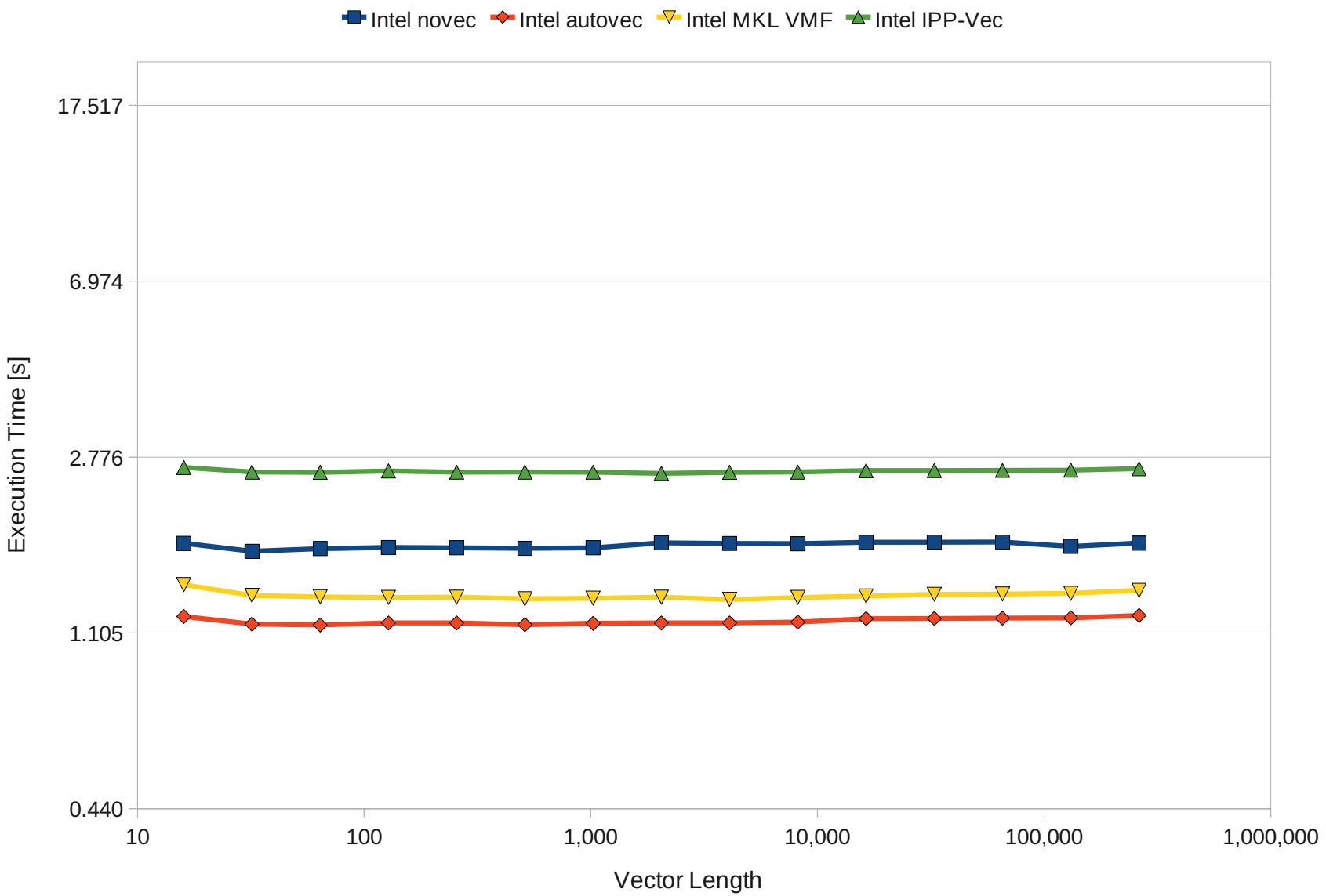
EXP function



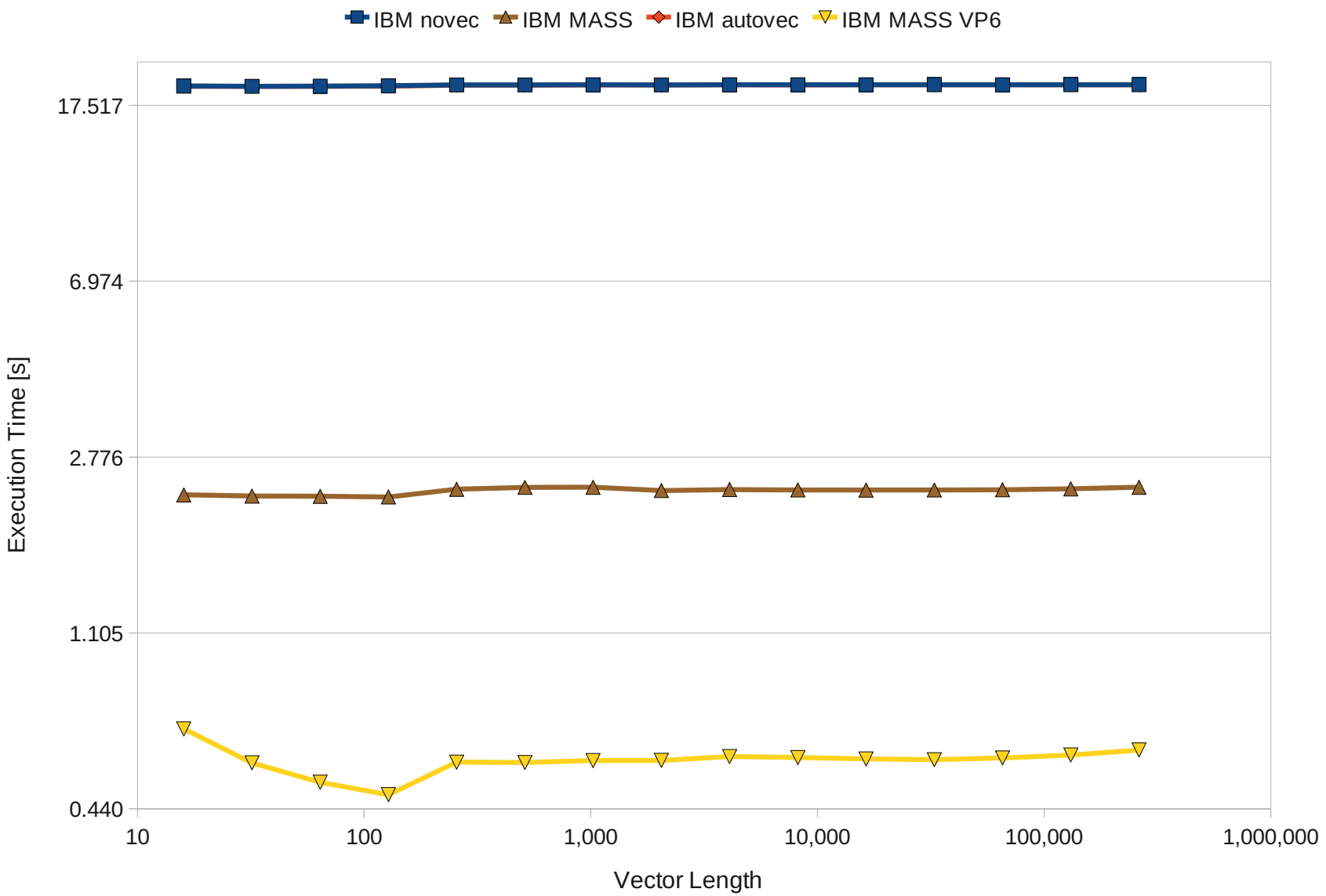
EXP function



ATAN2 function



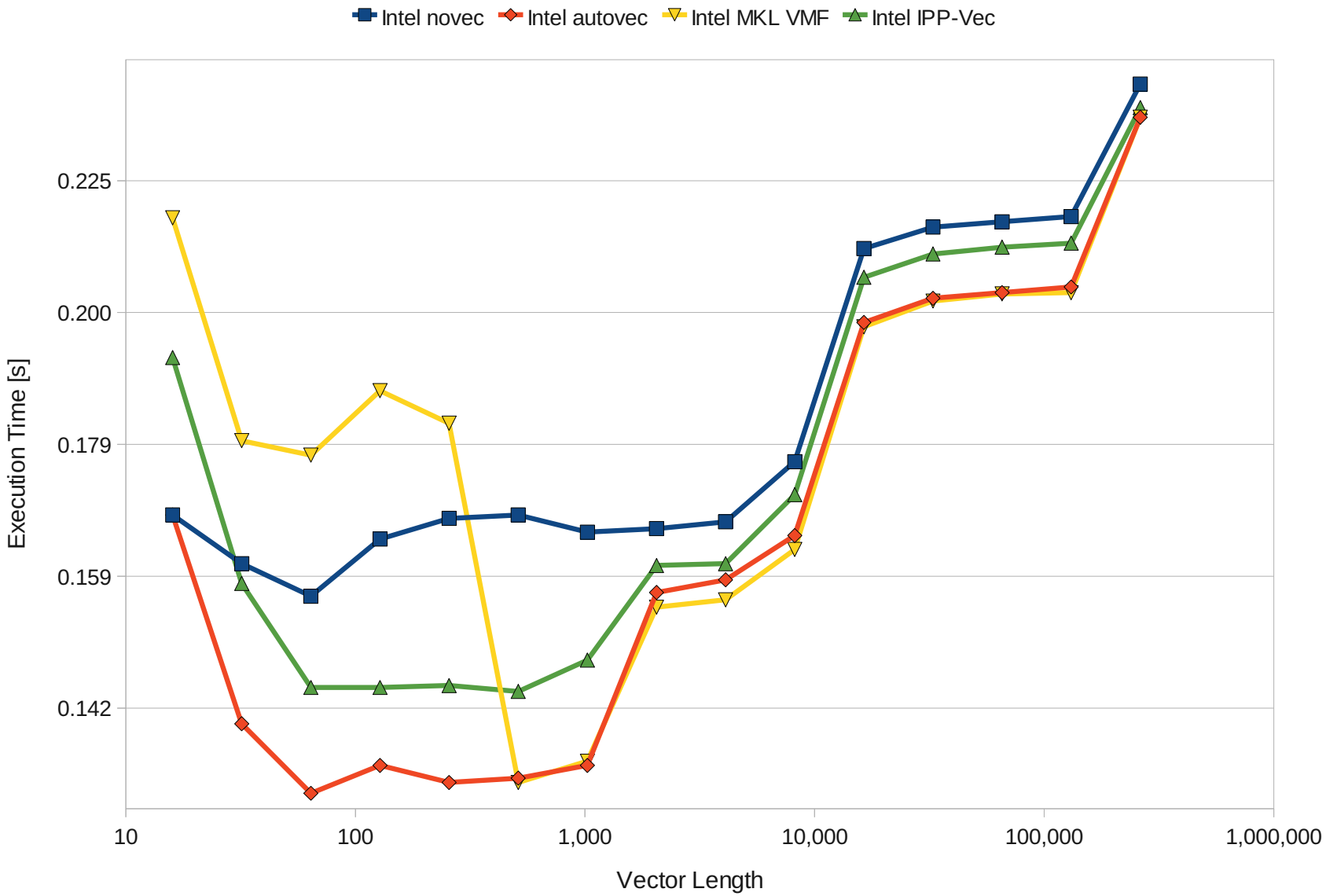
ATAN2 function



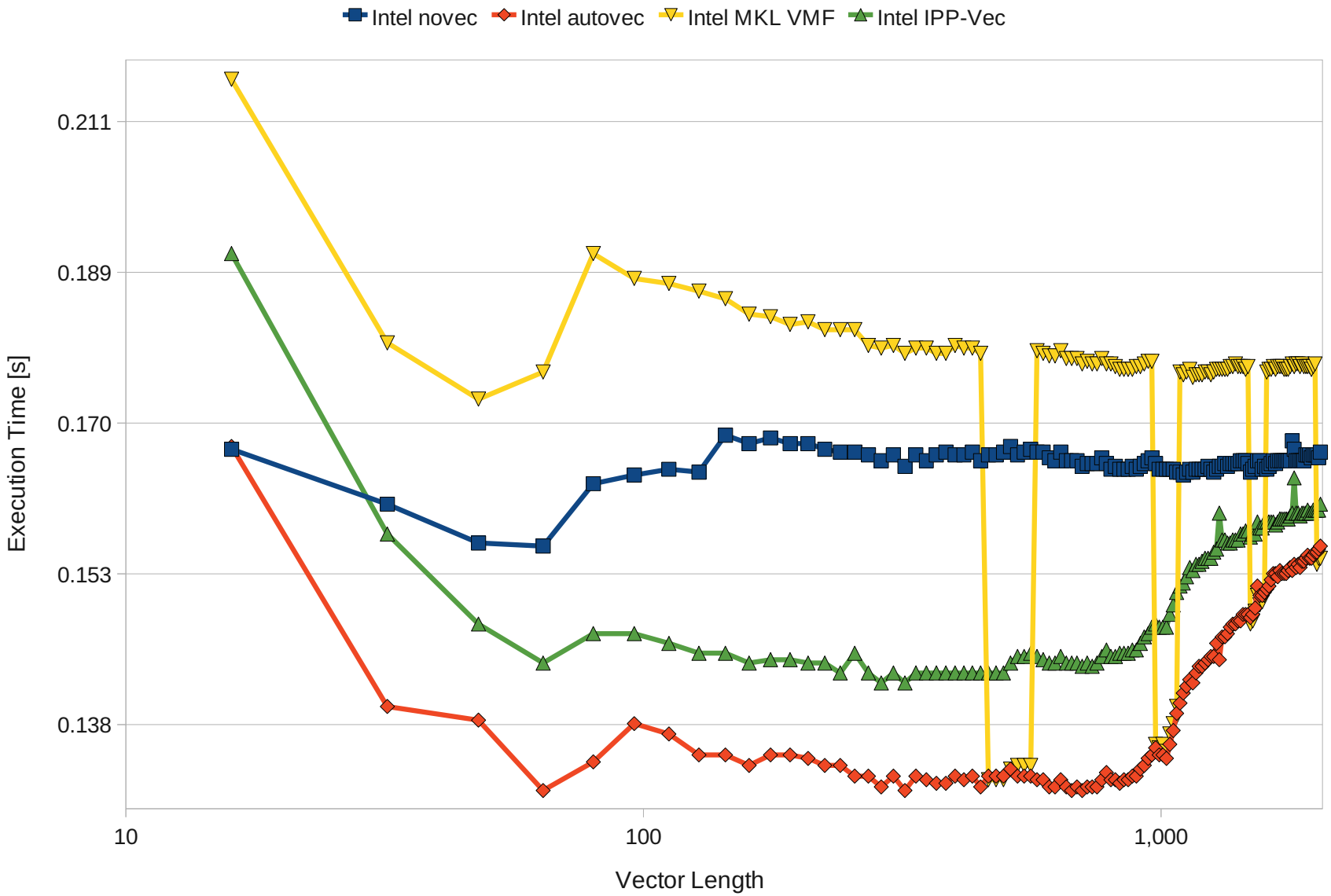
Problems and Issues

- AltiVec provides only single precission statements ($\Longrightarrow$ VSX)
- Intel `ifort` auto-vectorizer cannot vectorize the Fortran COMPLEX(8) type
- Problems with "basic" operations in Intel MKL v10.1 – v10.2 ($\Longrightarrow$ v10.3)

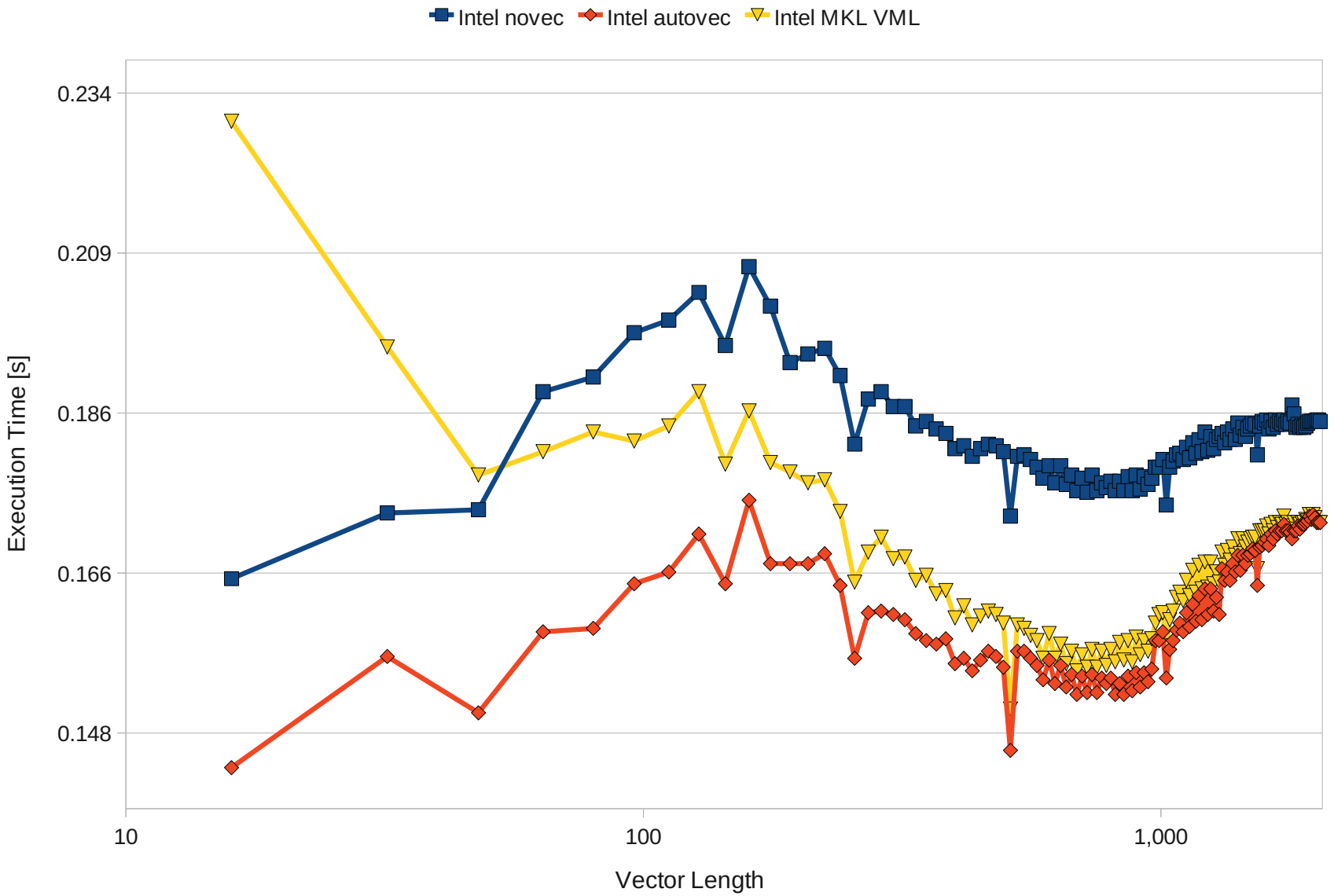
Multiplication operation



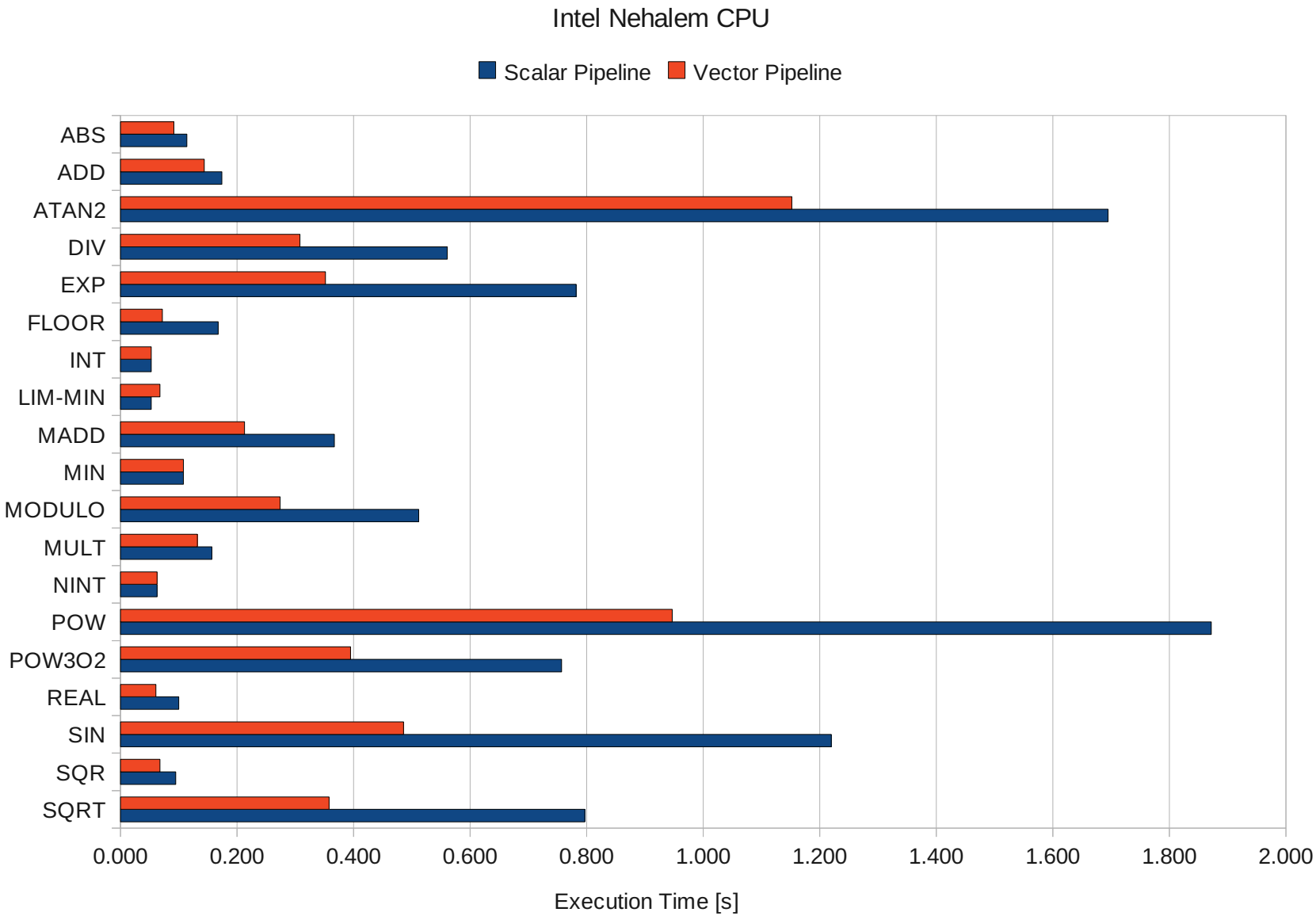
Multiplication operation



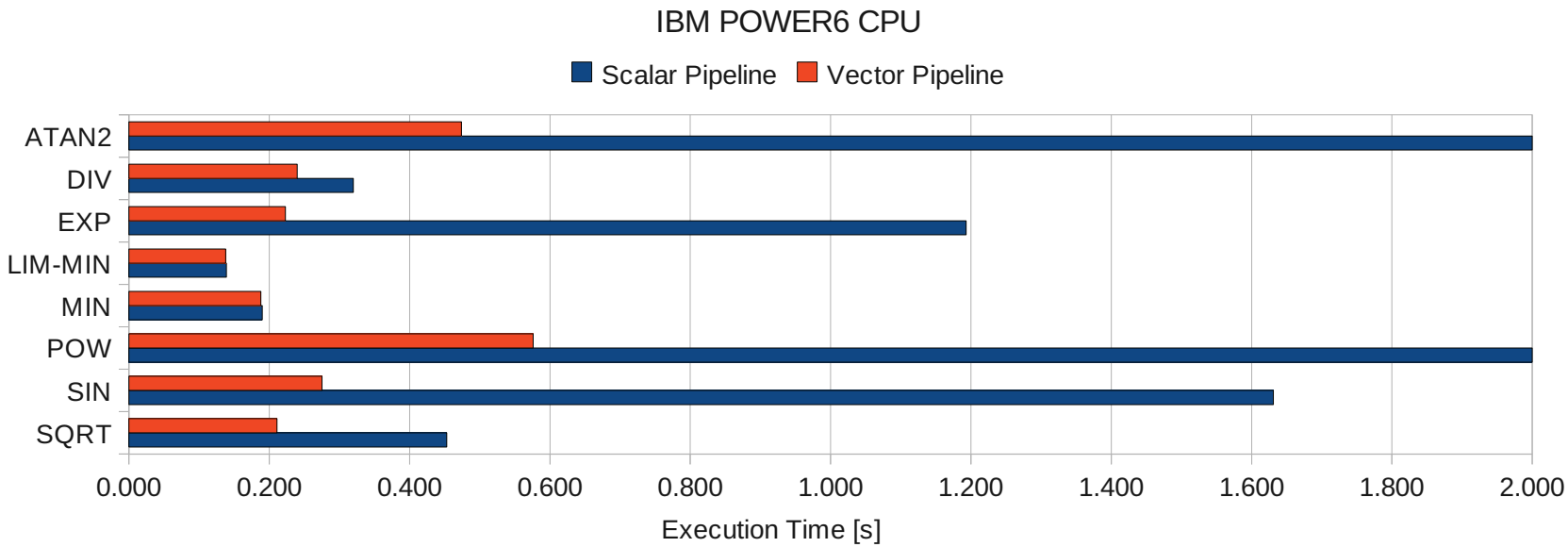
Multiplication operation



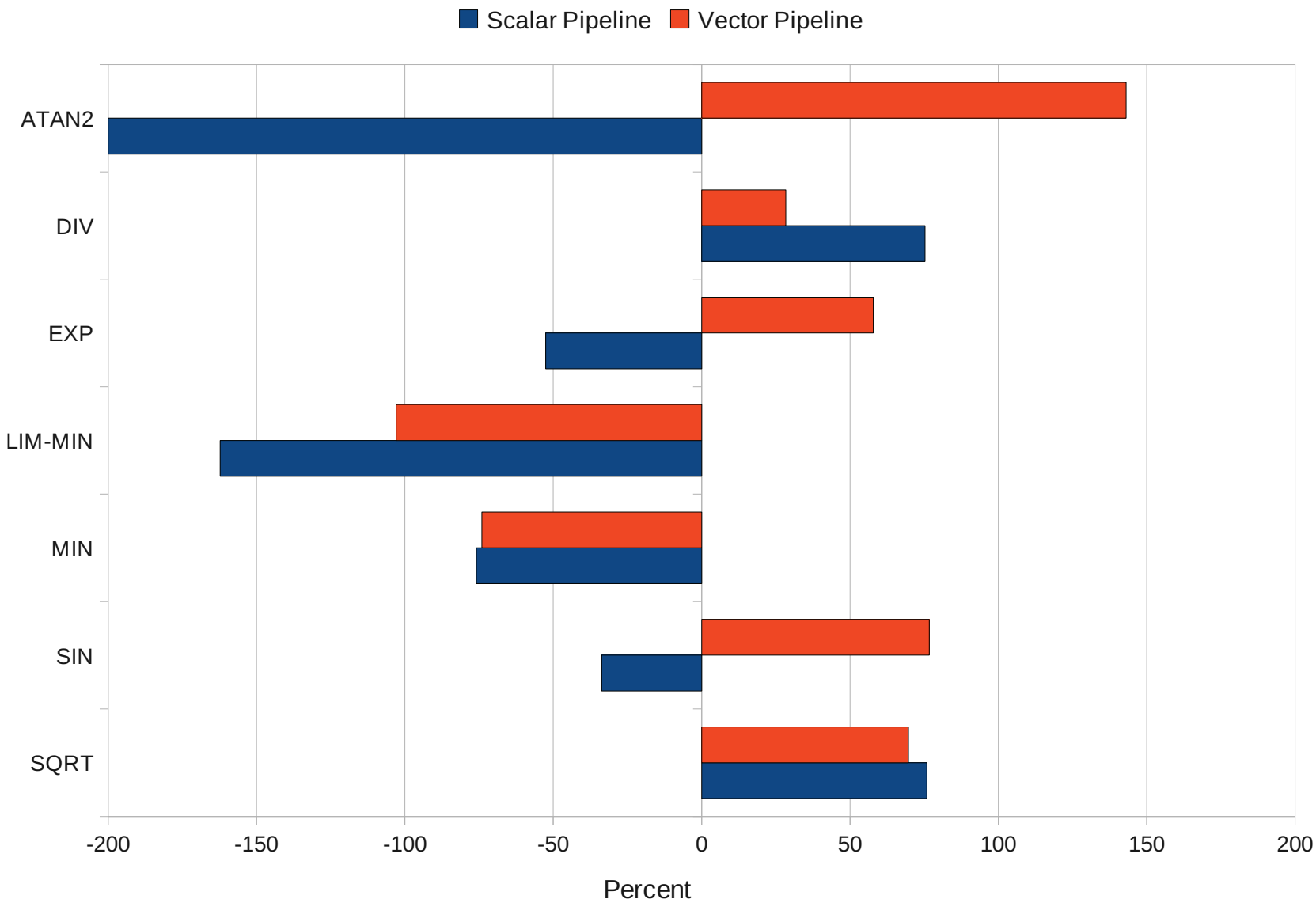
Functions on Intel Nehalem



Functions on IBM POWER6



IBM POWER6 vs. Intel Nehalem



Details of the Study

More details can be found in the technical report

Performance Tuning Using Vectorization

`http://edoc.mpg.de/display.epl?mode=doc&id=536180`