An Introduction to PYACTS

Tony Drummond
Lawrence Berkley National Laboratory
SIAM CSE05 - Orlando, FL, 2/11/05

Vicente Galiano
Miguel Hernandez University

Violeta Migallón and José Penadés
University of Alicante

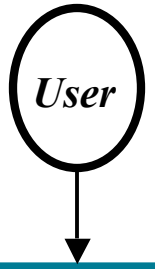
Motivation and Design Consideration

- High-level user friendly interface
- Hides details of parallelism from users
- Teaches users how to use the tools
- Flexible parameter reconfiguration
- Interoperability
- Choice of language: Python

PSE's

PMatlab

PyACTS



$Ax = b$

$Az = \lambda z$

View_field(T1)

$A = U\Sigma V^T$

High Level Interfaces

OPT++ PAWS Globus CUMULVS TAU

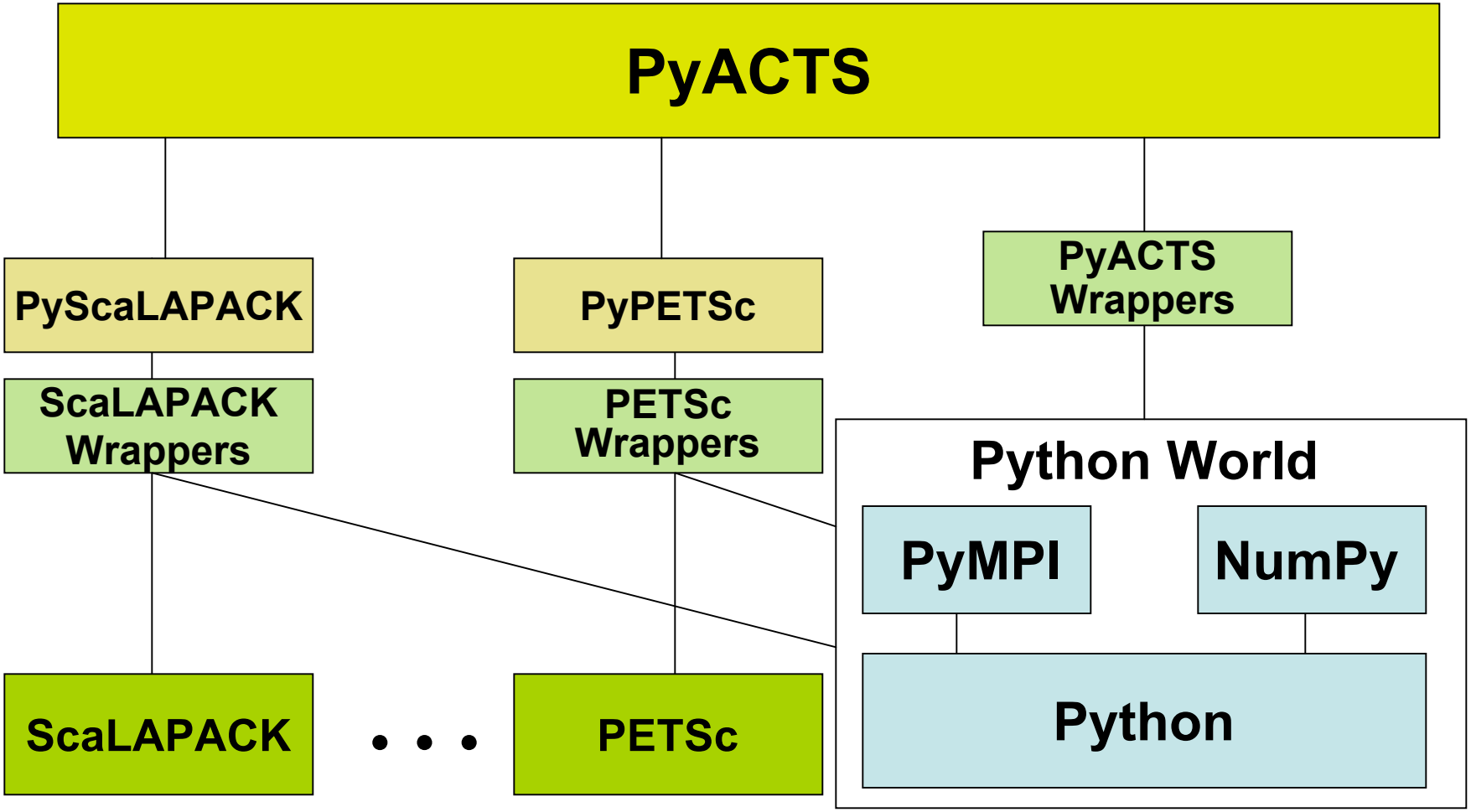
AZTEC Hypre PETSc Chombo Global Arrays

ScaLAPACK SuperLU TAO PVODE Overture

Motivation and Design Consideration

- Choice of scripting language: Python
- Uses PyMPI and Numeric
- Intended for testing and not high-performing production runs.

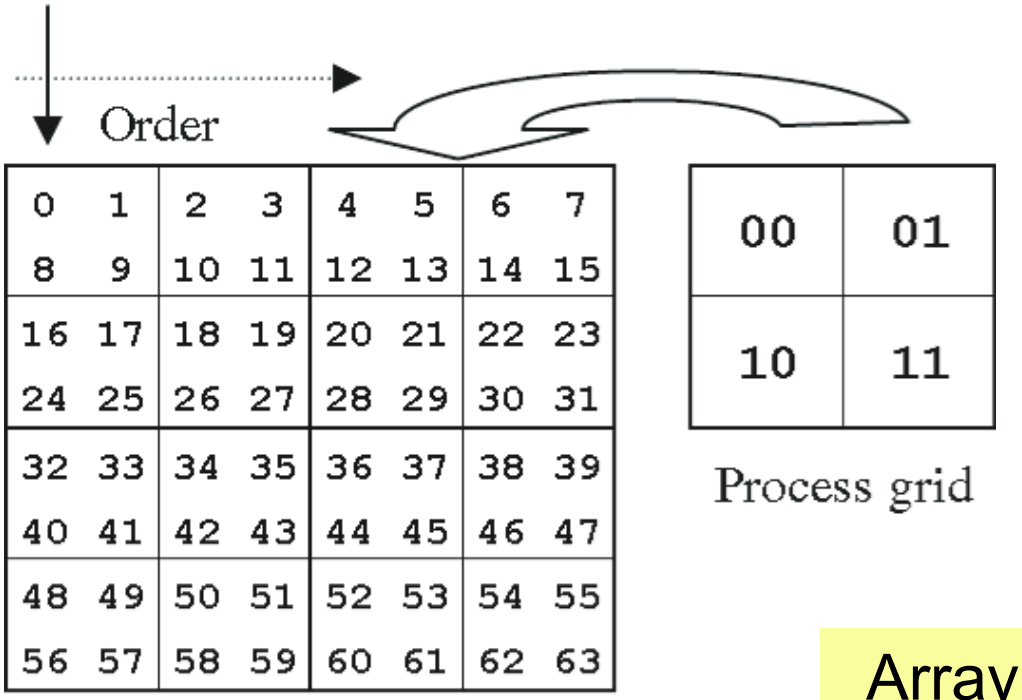
A Conceptual View Of PyACTS



PyBLACS Example Operation

```
> import PyACTS
> import Numeric
> n=8
> PyACTS.gridinit(nb=2)
> ACTS_lib=1 # ScaLAPACK library
> if PyACTS.iread==1:
> a=Numeric.reshape(range(n*n),[n,n])
> else:
> a=None
> a=PyACTS.Num2PyACTS(a,ACTS_lib) # convert array to PyACTS array
> print "PyACTS Array Properties in
      [",PyACTS.myrow,",",PyACTS.mycol,"]"
> print " lib=",a.lib
> print " desc=",a.desc
> print " data=",a.data
> PyACTS.gridexit().
```

PyBLACS Example Operation



Array Distribution and Processor Layout

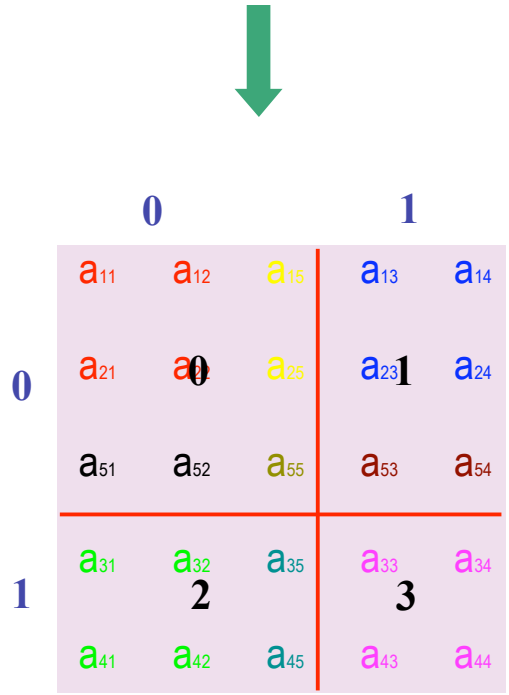
PyBLACS operation

```
PyACTS Array Properties in [ 0 , 0 ]
       lib= 1; desc= [1 0 8 8 2 2 0 0 4]
       data= [ 0  8 32 40  1  9 33 41
               4 12 36 44  5 13 37 45]
PyACTS Array Properties in [ 1 , 0 ]
       lib= 1;desc= [1 0 8 8 2 2 0 0 4]
       data= [16 24 48 56 17 25 49 57
              20 28 52 60 21 29 53 61]
PyACTS Array Properties in [ 1 , 1 ]
       lib= 1; desc= [1 0 8 8 2 2 0 0 4]
       data= [18 26 50 58 19 27 51 59
              22 30 54 62 23 31 55 63]
PyACTS Array Properties in [ 0 , 1 ]
       lib= 1; desc= [1 0 8 8 2 2 0 0 4]
       data= [ 2 10 34 42  3 11 35 43
               6 14 38 46  7 15 39 47]
```

Output from code

2D Block-Cyclic Distribution

$$\begin{bmatrix} 1.1 & 1.2 & 1.3 & 1.4 & 1.5 \\ -2.1 & 2.2 & 2.3 & 2.4 & 2.5 \\ -3.1 & -3.2 & 3.3 & 3.4 & 3.5 \\ -4.1 & -4.2 & -4.3 & 4.4 & 4.5 \\ -5.1 & -5.2 & -5.3 & -5.4 & 5.5 \end{bmatrix}$$

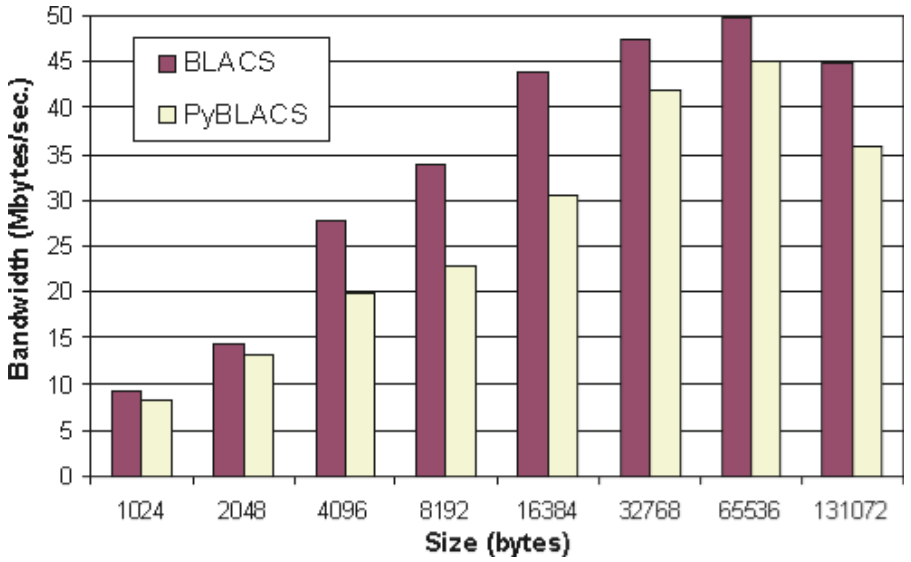



```
CALL BLACS_GRIDINFO( ICTXT, NPROW, NPCOL, MYROW, MYCOL )

IF      ( MYROW.EQ.0 .AND. MYCOL.EQ.0 ) THEN
        A(1) = 1.1; A(2) = -2.1; A(3) = -5.1;
        A(1+LDA) = 1.2; A(2+LDA) = 2.2; A(3+LDA) = -5.2;
        A(1+2*LDA) = 1.5; A(2+3*LDA) = 2.5; A(3+4*LDA) = -5.5;
ELSE IF ( MYROW.EQ.0 .AND. MYCOL.EQ.1 ) THEN
        A(1) = 1.3; A(2) = 2.3; A(3) = -5.3;
        A(1+LDA) = 1.4; A(2+LDA) = 2.4; A(3+LDA) = -5.4;
ELSE IF ( MYROW.EQ.1 .AND. MYCOL.EQ.0 ) THEN
        A(1) = -3.1; A(2) = -4.1;
        A(1+LDA) = -3.2; A(2+LDA) = -4.2;
        A(1+2*LDA) = 3.5; A(2+3*LDA) = 4.5;
ELSE IF ( MYROW.EQ.1 .AND. MYCOL.EQ.1 ) THEN
        A(1) = 3.3; A(2) = -4.3;
        A(1+LDA) = 3.4; A(2+LDA) = 4.4;
END IF

CALL PDGESVD( JOBU, JOBVT, M, N, A, IA, JA, DESCA, S, U, IU,
              JU, DESCU, VT, IVT, JVT, DESCVT, WORK, LWORK,
              INFO )
```

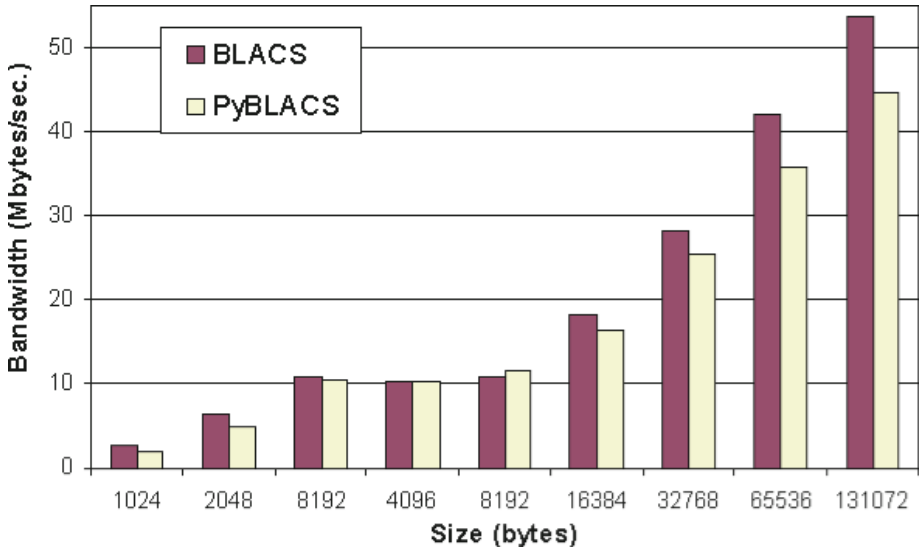
LDA is the leading dimension of the local array (see next slides)

Array descriptor for A (see next slides)

BLACS vs PyBLACS



Linux Cluster (2Ghz)



IBM SP - PWR 3

Berkeley Lab

Office of Science
U.S. Department of Energy

ACTS Collection

Example of PBLAS: pvgemm

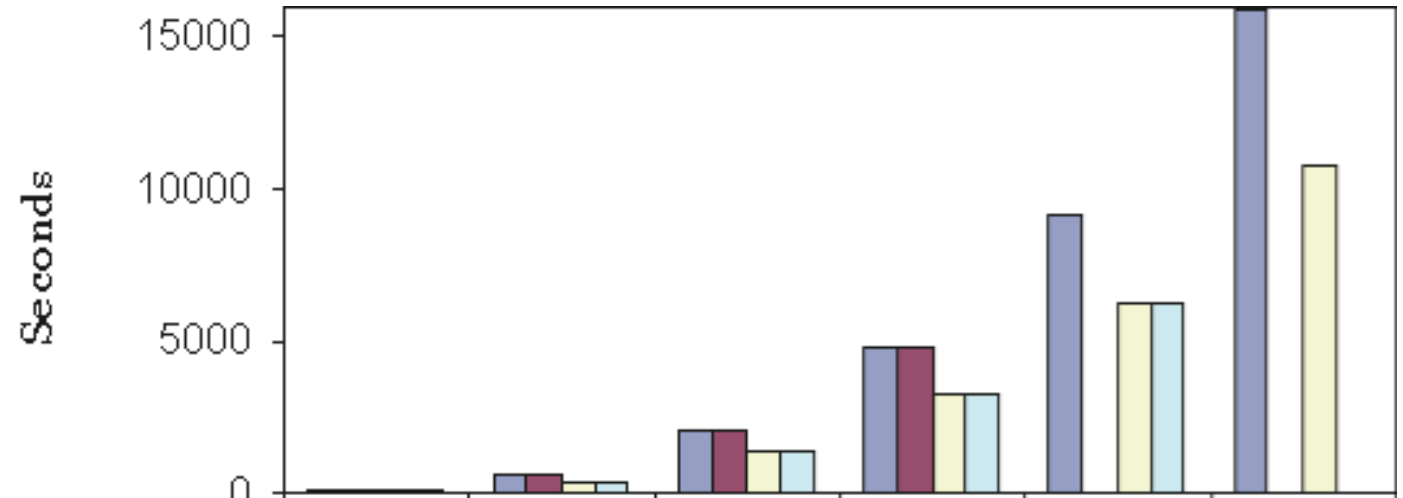
```
PvGEMM( TRANSA, TRANSB, M, N, K, ALPHA,
        A, IA, JA, DESCA,
        B, IB, JB, DESCB,
        BETA, C, IC, JC, DESCC )
```

- User needs to know about the parallel environment (data layout)
- User needs to initialize the process grid (BLACS)
- User needs to distribute data arrays
- Know details about the BLAS 3 call

Example of PyPBLAS: pvgemm

```
> from PyACTS import *
> import PyACTS.PyPBLAS as PyPBLAS
> import time
> n=500
> ACTS_lib=1 # ScaLAPACK library
> PyACTS.gridinit() # grid initialization
> alpha=Scal2PyACTS(2,ACTS_lib) # convert scalar
                                # to PyACTS scalar
> beta=Scal2PyACTS(3,ACTS_lib)
> a=Rand2PyACTS(n,n,ACTS_lib) # generate a random
                              # PyACTS array
> b=Rand2PyACTS(n,n,ACTS_lib)
> c=Rand2PyACTS(n,n,ACTS_lib)
> c=PyPBLAS.pvgemm(alpha,a,b,beta,c) # call level 3
                                     # PBLAS routine
> PyACTS.gridexit()
```

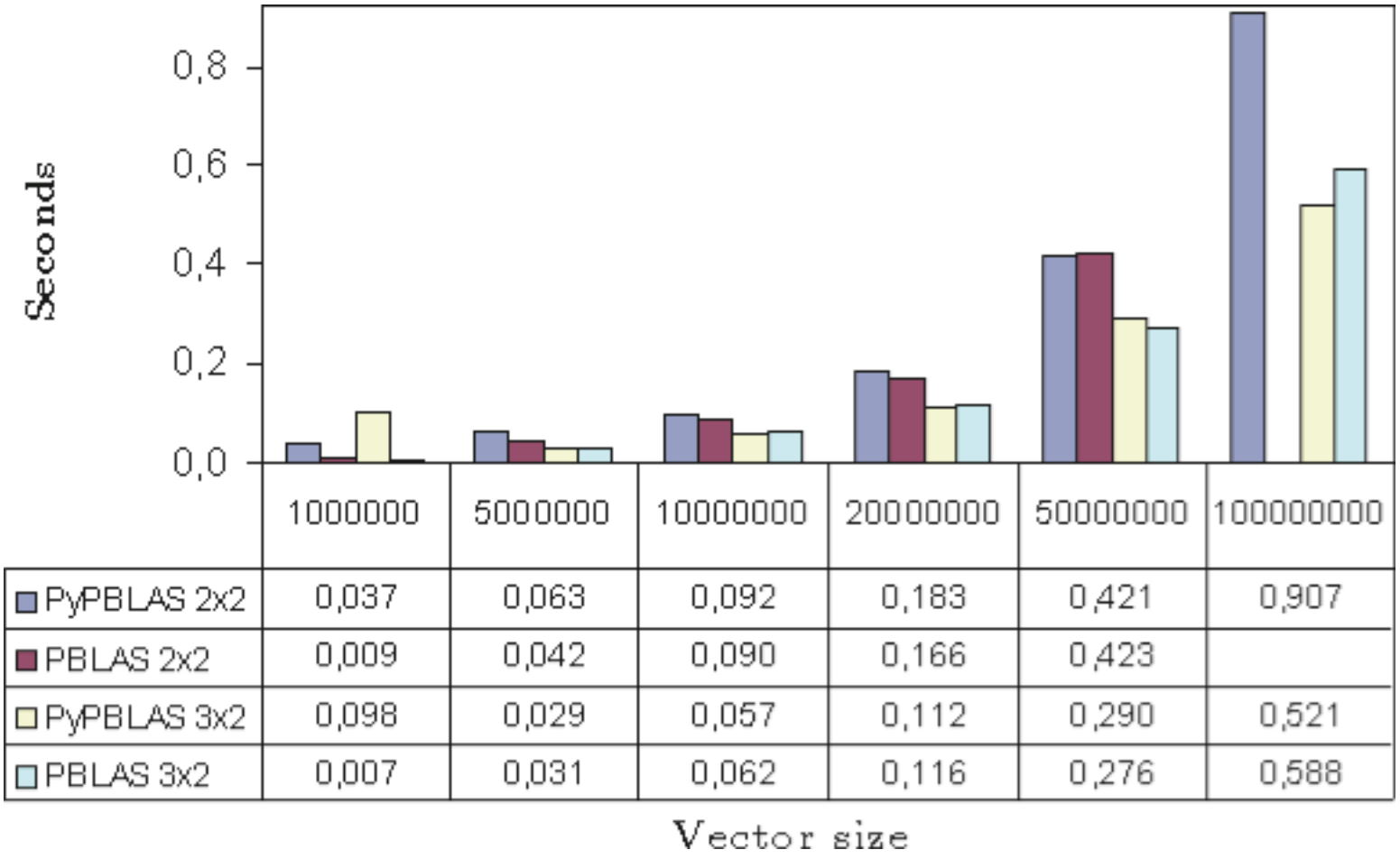
Example of PBLAS: pdgemm (cluster)



	2000	4000	6000	8000	10000	12000
PyPBLAS 2x2	76,0	608,3	2038,7	4812,6	9149,1	15890,7
PBLAS 2x2	76,0	610,2	2043,3	4805,9		
PyPBLAS 3x2	51,3	403,1	1360,0	3236,3	6319,4	10826,5
PBLAS 3x2	51,0	404,0	1367,2	3241,1	6310,4	

Matrix size

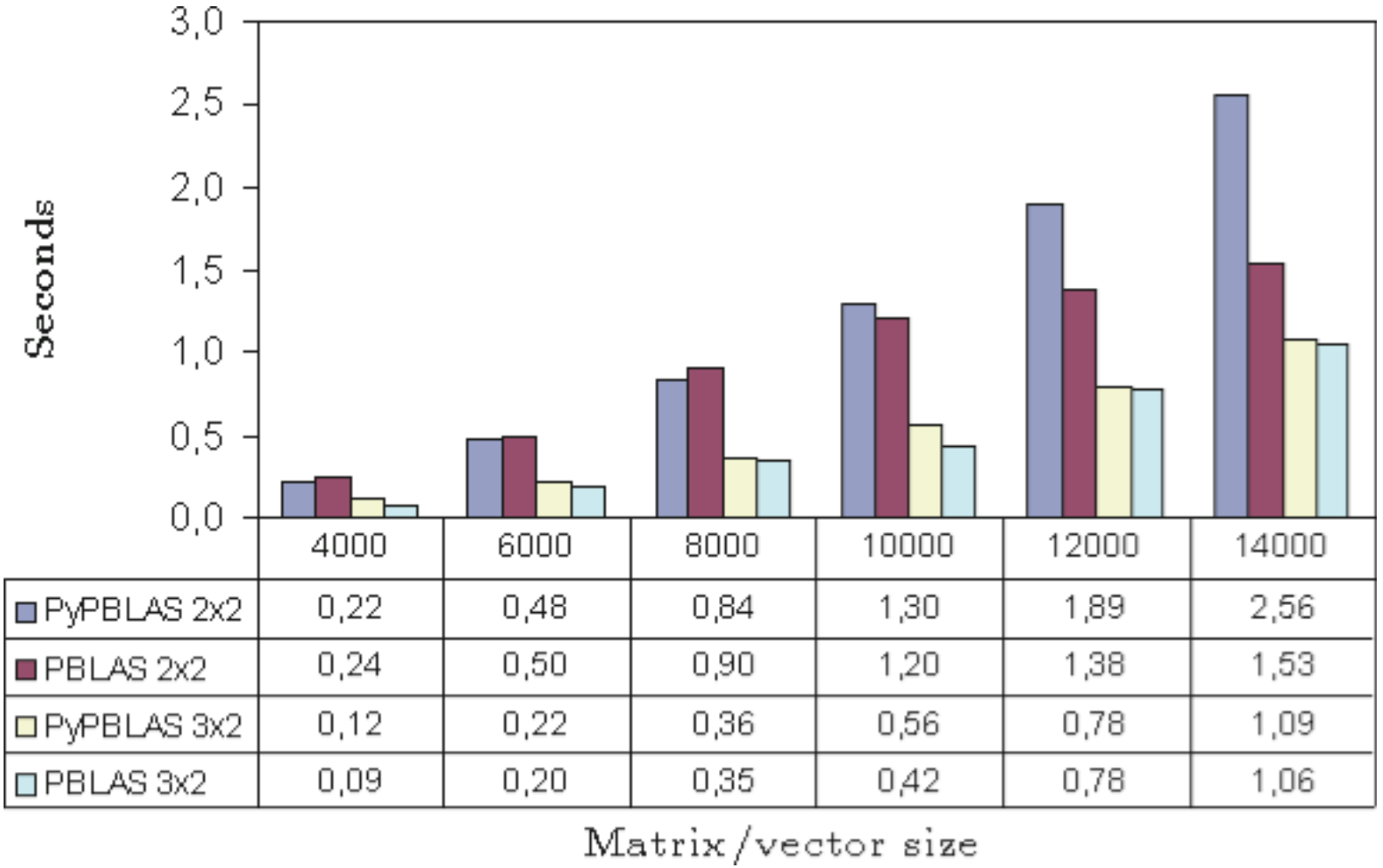
Example of PBLAS: pdaxpy (cluster)



	1000000	5000000	10000000	20000000	50000000	100000000
PyPBLAS 2x2	0,037	0,063	0,092	0,183	0,421	0,907
PBLAS 2x2	0,009	0,042	0,090	0,166	0,423	
PyPBLAS 3x2	0,098	0,029	0,057	0,112	0,290	0,521
PBLAS 3x2	0,007	0,031	0,062	0,116	0,276	0,588

Berkeley Lab
Office of Science
U.S. Department of Energy
ACTS Collection

Example of PBLAS: pdger (cluster)



	4000	6000	8000	10000	12000	14000
PyPBLAS 2x2	0,22	0,48	0,84	1,30	1,89	2,56
PBLAS 2x2	0,24	0,50	0,90	1,20	1,38	1,53
PyPBLAS 3x2	0,12	0,22	0,36	0,56	0,78	1,09
PBLAS 3x2	0,09	0,20	0,35	0,42	0,78	1,06

Future Work

- Work on release and documentation
- Include other tools and data formats
- Scriber function ⇒ high performance codes in C, C++ and Fortran flavors.
- Visit http://acts.nersc.gov for updates