**61 PETSc users responded to the survey. In the following, the responses and the comments are in bold text.**

## Questionnaire on PETSc Experience

**These questions were created to receive feedback from the users of PETSc library and provide them with the requested functionalities. You may choose to answer as many questions as you can and skip the remaining ones. We appreciate additional comments for the answers you select.**

### *Installation and Packaging*

1. On which OS(s) did you use or anticipate using PETSc? Please provide further details such as distribution and flavor in the comments boxes.

   Linux
   UNIX
   Windows
   Mac OSX
   Solaris
   Other (Please specify)

   **Responses:**

   **Linux – 57/61 users**

   **Comments: OpenSUSE, Ubuntu, Arch, RedHat, Suse, Ubuntu, all 64-bit, SUSE Linux x64, The main platform we run the simulator, ubuntu 11.04, Ubuntu at home plus flavors installed on supercomputers, Ubuntu, red hat clones such as centOS, scientificLinux, Ubuntu 10.10, CentOS, Ubuntu, Ubuntu 11.04 and Fedora 14/15/16, Fedora, Ubuntu, Suse, CentOS, Slackware, Redhat, gentoo, RHEL4, Ubuntu, 64-bit Red Hat / CentOS, redhat, OpenSuse 11.4, Ubuntu 10.04 LTS, suse9,10,11 redHat/centos 4&5, ubuntu 64, Rocks, Opensuse 11.1, CentOS, ubuntu 10.04**

   **UNIX: 6 users**
   **Comments: AIX, FreeBSD, multi**

   **Windows: 10 users**
   **Comments: Windows 7 and later would be good as option, but not neccessary.**
   **Some customers require Windows.**
   **32-bit and 64-bit**
   **win32 & x64**

**MacOS: 20 users**
**Comments: 10.6, snow leopard, Leopard, Relatively recent versions, recent releases**

2. If you are using PETSc on a parallel machine (e.g., cluster or supercomputer), please provide some details (if known) on the configuration. This information may be obtained using the following commands on Unix like systems:

cat /proc/cpuinfo
sudo dmidecode –t cache (This command requires Dmidecode http://download.savannah.gnu.org/releases/dmidecode installed on the machine).

Number of nodes: **24, 1-300, 200+K, from 10 to hundreds, 4000, 1368, 2816, 5, it depends, 64, 120, 1000, 32, usually several thousand, 1888, thousand(s), 150, 900, 1-200, 14, 4, 680, 100-1000, 4000, 40, 4, 64, 50, 30, 16, 40, 300, 1, 1000, 8, 512**

Number of cores for each node: **8, 16-Apr, 12, from 4 to 12, 8, 12, 32, 8, 2, 8, 16-Aug, 24, ~8-12, 12, 12, 8, 4, 12, 8, 8, 8, 3 cores 4 sockets 12 cores, 16, 2, 8, 8, 8, single, 2, 2, 3600, 8, 12, 4**

Interconnect: **19 users said Infiniband, other responses include Cray Gemini, Ethernet, Ethernet, Myrinet, Infiniband and Gigabit Ethernet, ethernet, gigabit, GigaBit, Gigabit ethernet**

Sizes of L1 cache, L2 cache, L3 cache, and RAM:

**12288 KB cache, 24 GB RAM**
**16-Apr**
**variable**
**Intel Westmere**
**1 GB per core RAM**
**RAM 80 GB**
**x86 caches: socket [32 KB - 256 KB - 12 MB] – node [64 GB ram]**

**L2-Cache: 12x 512kB, L3-Cache: 12MB shared, 64 GB RAM**

**64k/256k/12M/24G**
**Cache 8192 KB, ram 1.5GB**

**L1 (32KB/core), L2 (256KB/core), L3 (12MB shared by 6 cores), RAM: 48 GB**

**L3 8MB, 24GB RAM**
**RAM 8G per node**
**8 GB RAM/node**
**128KB, 1024 KB, 16 MB, 96 GB**
**?, ? ,? , 2 GB**
**? ,? ,? ,1.5 TByte**
**12288kb cache, RAM 16GB/node**
**3072KB Cache 16GB RAM**
**128kb L1, 2048 kb L2**
**24MB L3, 24GB Memory**
**L3: 12 Mb, 48 GB RAM per node**
**?**

Other details:

**Look up Jaguar XT5/XK6 at ORNL, Hopper, Cray XE6, Intel(R) Xeon(R) CPU X5355 @ 2.66GHz, x86-64 architectures, 64bit**
**Intel(R) Xeon(R) CPU X5650 @ 2.67GHz, Fortran, ICE of http://www.lrz.de/services/compute/linux-cluster/overview/, Intel(R) Xeon(R) CPU, Intel Xeon X5650, Xeon X5650 @ 2.67GHz**

3. If you are using PETSc on a desktop workstation, please provide details on the configuration of the workstation. This information may be obtained using the following commands on Unix like systems:
cat /proc/cpuinfo
sudo dmidecode –t cache (This command requires Dmdiecode http://download.savannah.gnu.org/releases/dmidecode installed on the machine).

Number of cores: **4, 8, 4, 8-Apr, 8, from 4 to 12, 8, 8, 2, 16, CPU XEON 5620 X2, ~4, 8, 6, 4, 6, 2 to 8, 4, 2, 2, 4, 4-Feb, 8, 8, 6, 8, 16, 4, 4, 4, 4, 12, 2, 16, 4, 4, 4, 2**

Sizes of L1 cache, L2 cache, L3 cache, and RAM:

**6MB**
**12288 KB cache, 24 GB RAM**
**8GB RAM**
**16-144**
**8192KB, 24GB**
**variable**
**16 GB**
**RAM 6 GB**
**32GB DDR3 memory**
**Varies**
**RAM 2GB per node**
**Cache 8192 KB, ram 12GB**
**32kb, 256kb, 12mb, 16 GB**

**L1(32KB/core), L2(3MB shared by 2 cores),**
**RAM:8GB**
**L3 3MB, 4GB RAM**
**128KB, 512KB, 6MB, 16GB**
**798kB 3MB 6MB**
**?,?,?, 8GB**
**12288kb cache, RAM 32GB total**
**3072KB Cache 16GBRAM**
**RAM 24g**
**L2 12M**
**64 KB L1 per core, 256 KB L2 per core, 12MB L3, 96GB RAM**
**--,--,--, 4 or 8 GB**
**32GB**
**L3: 8 Mb, 16 GB RAM**
**?**

Other details:

**Nvidia quadro 4500 GPU,**
**MacPro,**
**Intel(R) Core(TM) i7-2630QM CPU @ 2.00GHz, with centos 5 OS**
**64bit**
**Intel(R) Xeon(R) CPU X5675@3.07GHz 1 Processor, 6 Cores, 12 Threads**
**Intel(R) Core(TM)2 Quad CPU Q9400 @ 2.66GHz**
**AMD Phenom II X6 1090T**
**Intel(R) Core(TM) i7 CPU**
**Intel(R) Xeon(R) CPU**
**Intel xeon, dell precision T5500**
**2 Intel X5650 CPUs**
**New Dell dual 8 core sandybridge E5-2670**
**Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz**

4. Approximately how long does it take you to configure and install PETSc?

**Half an hour**

**10 min (using cmake for parallel builds)**

**N/A**

**Depends on the system, 20 mins - 1 day**

**Hours on my PC, ~hour on Jaguar, minutes on RedHat Linux box**

**Half an hour**

**Two hours**

**A few hours**

**2h**

**A few minutes**

**15-20 minutes**

**About 20 minutes I think**

**5 mins**

**10 minutes**

**30 minutes**

**Fast if all is ok, a lot of time when a configure.py test fails**
**~30 minutes**

**Half an hour on Linux, one day on Windows**

**15 minutes on my laptop, a full day on Blue Gene/P**

**An hour?**

**20 minutes**

**For 1st time 1.00 hr time frame**

**5mn**

**~ 1 hour**

**Less than 30 minutes**

**20-30 min**

**A few days for the first time, then less than an hour**

**2 hours**

**1 minute**

**5 min**

**30 minutes**

**10 minutes**

**10 minutes**

**On windows quite a long...**

**Standard: 3h; specific compilers or different MPI installation: 1 day**

**15 minutes**

**Don't remember**

**Five minutes**

**45min**

**1 hour**

**16 hours**

**30 min**

**15 min**

**5 minutes**

**2 hours**

**10 minutes (do it all the time, release and dev)**

**30 minutes**

**15 minutes**

**10 minutes**
**~1 hour**

5. Did you use or anticipate using the Debian Package for PETSc for binary installation?

Yes
No
No Answer

Reponses: **40 users said NO**
**7 users said YES**

**Comments: I use petsc-dev from the repository.**

**I use Redhat 5.**

**Like to build from source, often make small modifications to the source.**

**I recompile PETSc because I use commercial compilers**

**We use centos 5/6 and always compile petsc from source code.**

**Too many relevant configuration options.**

**With all the external software packages that can be added to PETSc, I was not expecting that to happen.**

**I have many versions of PETSc, each with different configuration. It's important to understand and control the configuration settings, most notably MPI and the compilers.**

**I used it a while but it was quickly outdated.**
**I build PETSc myself so I have total control (which MPI, external packages, etc)**

**It's easier to install and Taylor-make**

**Wanted to use it, but its FORTRAN include files were seemingly broken (lines were split without matching ampersands, if I recall correctly).**

**I prefer to compile PETSc as local.**

**I use too many custom packages (and it's just easy for me to rebuild). I also use environment modules to manage different PETSC_DIRS and ARCHs**

6. Do you desire any improvements to the Debian package for PETSc?
**NA**
**I don't know which advantages it gives. Maybe I do.**
**No**
**A repository (launchpad ppa?) would be awesome.**
**No comment**
**No**
**Make sure it works with a FORTRAN compiler. :)**
**No**

7. PETSc interfaces with several external packages. What external packages do you use or anticipate using in your application(s)?
Please select all that apply.

ADIC/ADIFOR - **2 users**
AMD - **1 user**
AnaMod
Caco – **2 users, Comments: Chaco, lf th**
ESSL – **3 users**
Euclid – **4 users**
FFTW – **8 users, Comments: paral**
HDF – **10 users, Comments: paral**
Hypre – 26 users, **Comments:**
1) **Mutli**
2) **Parallel ILU, however not so efficient**
3) **Boomeramg, Euclid ilu**
LUSOL
Mathematica
MATLAB – **9 users**
MUMPS – **22 users, Comments: The main linear solver used in our application.**
ParMETIS – **26 users, Comments:**
1) **Unstr**
2) **Required by mumps and superlu**
3) **For mesh decomposition**
Party -**1 user**
PasStiX - **2 users**
PLAPACK - **6 users**
Prometheus – **2 users**
Sundial/CVODE
SuperLU and SuperLU_Dist – 21 users, Another Linear solver

Trilinos/ML – 8 users
UMFPACK – 9 users
SPRNG -
PTScotch – 3 users
SPAI – 3 users
SPOOLES – 4 users
Other – 5 users, **Comments**:

1) None of above
2) Eventually, Elemental. Disclaimer: I am the author.
3) pARMS
4) BoomerAMG
5) mpich2, fblaslapack

8. Did you have any specific problems with configuring and building when trying to make use of the external packages?

Reponses: **12 users said No**
Comments:

1) Sometimes it's not easy to get them working same way as they work independently.
2) Few. I always remember the dependency of external libraries.
3) It is an annoyance that Petsc does not check for all dependencies before starting to build any external packages.
4) As a part of Hypre, I remember that I had downloaded the .tar file myself and PETSc could not do it automatically. I guess that had something to do with the hypre package registration.
5) yes, the default ML configure fails and requires some manual labor
6) not really -- the command line help and PETSc web site pretty much answers everything.
7) Yes, some external packages require additional packages and those informations are hard to find in the documentation or not at all.
8) matlab connection (this may be outdated ...)
9) compiling petsc code using automake tools
10) It is very difficult to make a customized build on Windows platforms

9. We are working to simplify the configuration and building of PETSc by prepackaging some of the external packages. Name the packages that you would like us to consider, if any.

**See 7**

**Hypre, ParMETIS, HDF5, ASCEM-IO**

**SLEPc**

**ParMETIS, SuperLU and SuperLU_Dist**

**Mumps and its dependencies**

**N/A**

**I used Hypre BoomerAMG a lot, as this was often faster than the PETSc solvers. Same with SuperLU.**

**Mkl as low level package**

**ParMETIS, METIS, PTSctoch, MUMPS, SuperLU, HYPRE**

**Metis/ParMETIS, not only required by petsc, but also required by applications as mesh partitioner.**

**Hypre, hdf5, Matlab**

**Yes, I would like to have an interface for unstructured finite volume based CFD to set up the linear equations to solve on parallel machine.**

**Fftw**

**ParMETIS**

**SuperLU_dist, mumps**

**Trilinos/ml, MUMPS, ParMETIS**

**I use PETSc in conjunction with the finite element library libMesh. Anything to facilitate that would be a huge plus for me.**

**Matlab, spai**

**None**

**Umfpack**

**BoomerAMG**

**The sparse direct (umfpack, mumps) and AMG (Hypre, ML) solvers together with graph partitioners (metis/parmetis/maybe scotch...) probably are standard enough extension that it would be useful to have included.**

10. Did you face any other problems while configuring and building PETSc? Please provide details.

**See 8**

**Building on Windows with cygwin can be a bear.**

**PETSc does not allow to make a build that supports, at the same time, real and complex numbers. I think that, reviewing data type definitions and the API, it is possible to provide both with only one build.**

**Poor support under windows makes it very hard to get petsc working.**

**I had always encountered problems when I tried to use Intel compilers to compile and configure petsc on a couple of clusters. It finally worked out, but I had to do some fiddling around type of things.**

**Finding the correct settings for the Intel math kernel. It's important to switch off the multithreading**

**Not really. The command line help and PETSc web site pretty much answers everything.**

**Sometimes difficulty with non-gcc compiler, such as the Intel compiler.**

**Sometimes I cannot use local LABLAS PACK. By downloading them together with Petsc solved the problem.**

**Provide makefile examples. In some sense PETSc makes you use the PETSc makefile.**

11. What do you like the least in the PETSc configuration and build process?

**Configuring takes longer than the build.**

**See 8**
**Building on Windows is slow.**

**When something does not get built, it is difficult to understand why on my own. It is difficult to understand which are the proper configure options in some cases (e.g. specifying different BLAS and LAPACK ... examples are needed on the command-line help)**

**The many configuration arguments**

**Nothing**

**N/A**

**When a test fails a lot of time is spent to understand why, looking in the configure.py script**

**If PETSc doesn't find certainly specified library then it tries to substitute it. E.g. that happens with "--with-blas-lapack-dir", if no blas/lapack found in the specified directory PETSc will try to take them from "/usr/local/lib" for instance. I think it should warn user about that.**

**The configure procedure is very slow.{cr}{newline}On windows platform, configuration under cygwin needs great patient.**

**The amount of time required.**

**I wish the entire package could b linked as one single library file. I know it is too much to ask, but just wanted to share my dreams.**

**Quite long process...**

**It requires two separate pulls for PETSc and the BuildSystem. This is a minor problem, but it would be nice to have one single command to sync everything.**

**More install informations especially related to external packages and the configuration should start with an error checking for misspelling or for dependencies between given options that will be a real timesaver for new users.**

**The amount of time it takes.**

**The list of options when doing ./configure --help is very long maybe being able to do something like ./configure --help compilers would be nice**

**Having different MPI versions is a pain. It is often not clear whether the problem is the MPI installation or the petsc configuration.**

**make tests works, but linker errors occur when trying to use it.**

**For mere usage in serial mode: Why to bother with MPI stuff?**

**it's very good**

**Windows build support**

**I did not know that it was in debugging mode in the beginning. It will be nice to mention clearly for beginners.**

**Slow config. Also, some tests are allowed to fail when running "make testfortran", due to round-off differences. It would be better if these tests were done with some tolerance.**

**I suppose I've gotten used to it (and compared to the rest of the libraries I need to build it's relatively easy)**

**Simple and easy. Speed is not an issue.**

**Nothing, it works rather nicely.**

12. What do you like the most in the PETSc configuration and build process?

**That it pulls and builds dependencies automatically. No use for Hypre on Windows with no cygwin dependency though.**

**It is easy (and fast on Linux)**

**It automatically keeps different builds separated by architectures. This is great.**

**I like the automation**

**everything**

**N/A**

**The ease of configuration, and the ability to quickly switch between builds.**

**I will prefer classic configure tool (from autotools) instead of the specific python script**

**It downloads and compiles everything I need.**

**It is extremely robust.**

**Once all external packages that are needed have been selected, the build system has worked flawlessly every time that I have tried it. If this changes even slightly, the number of new PETSc users will dramatically decline.**

**The manual provided online is very helpful. Also, the team working on this package is very very helpful.**

**it's very flexible. It's easy to write a shell script to configure, build and install PETSc once you know which configure options you need**

**Detailed and straightforward process**

**cmake**

**Nothing in particular**

**its robust and intuitive**

**The options to automatically download and install packages is excellent.**

**It works!**

**The different architecture variants compiled inside one installation.**

**The make tests availability of many options**

**Straightforward method to have multiple configurations of PETSc in one place - debug - optimized - w/wo Hypre.**

**Very fast**

**Very easy, actually "hands off"**

**Hypre external package is very easy to use.**
**Big fan of cmake and parallel make.**

**Easy**

**It all works automatically.**

13. Using a 5-point scale (1 = easy, 5 = very difficult), please rate the difficulty level of configuring and installing PETSc?
    **5 – 1 user**
    **4 – 7 users**
    **3 – 17 users**
    **2 – 18 users**
    **1 – 9 users**

14. Did you use or anticipate using Python bindings (PETSc4py) for PETSc?

    Yes – **16 users**
    No – **30 users**
    No Answer

    **Comments:**

    **I did not know it exists. I may try it one day.**

    **No, but maybe in the future.**

    **I anticipate to use the Python bindings.**

    **anticipate to use**

    **I love PETSC4py but the documentation is very lacking.**

    **Not yet**

    **I would like to use it, but it's availability on Windows with packages such as PythonXY is not**

    **straightforward, and its integration with PySparse is bad.**

**Love python**

**Experimented with a little bit, for fun. Not normally using them.**

**Possibly in the feature when running multi-disciplinary simulations**

## *Profiling and Performance Analysis*

15. What is the minimum performance improvement for your structured grid application that would encourage you to upgrade PETSc?

    **I use mostly unstructured. Basically any improvements is worth upgrading.**

    **I upgrade PETSc on a whim as I track petsc-dev**

    **I already use PETSc for all my simulations.**

    **I do computations on unstrucutred grid. I wait for PETSc to do something for efficient computations on unstructured grids.**

    **use of multiple GPUs on the same workstation**

    **We upgrade every time there is a new release**

    **2X**

    **~20%**

    **2x faster**

    **Irrelevant. I use it for prototyping.**

    **20%**

    **convergence rate for simulation has reduced.**

    **I don't use structured grids**

    **10%**

    **I upgrade whenever there is a new release (regardless of performance). I assume newer version is better.**

**5%**

**2**

**50%**

**10%**

**5%**

**None**

**N/A**

**20%**

**15%**

**I upgrade all the time**

**I am using tetrahedron**

**15 percent**

16. Are the profiling options in PETSc (log_summary, info, log_trace, and log_mpe) sufficient for your profiling needs?

Yes – **35 users**
No – **5 users**

**Comments:**

**I have only use logging (including petsc routines and Petsc logging that I have embedded in my code). It would be nice to have an accurate breakdown of load imbalance in the solvers in order to verify whether an increase in global reduction time is due to communication bottlenecks or load imbalance projected onto a global synchronization point...make sense?**

**In fact, the performance of our application most dependents on 1) dspeed of irect linear solver such as mumps, 2) preconditioner which can reduce the number of iteration.**

**Honestly, I think profiling data shown by PETSc is still a bit complicated to interpret. I wish there was some simpler versions of that.**

**I use scalasca if i need detailed information**
**But I still don't use it too much.**

**Although I augment them with Macosx profiling tools**

17. Have you ever used the PETSc functionality for profiling user defined sections of code? This is typically done using the function calls PetscLogEventRegister (), PetscLogEventBegin (), and PetscLogEventEnd ().

    Yes – **20 users**
    No – **28 users**
    No Answer
    **Comments:**
    **All the time!**
    **I use it very often through my codes.**
    **But I plan to try it at some point.**
    **Super. I will have a try.**

18. Have you used any other performance tools for your PETSc application(s)?
    Yes - **21 users**
    No – **26 users**
    Comments:
    **Valgrind**
    **Valgrind**
    **TAU and CrayPat.**
    **Tau**
    **Self-written stuff usually.**
    **Valgrind (maybe not totally considered as performance tool).**
    **But I may start using Tau.**
    **Mac OS X Instruments**
    **Scalasca**
    **vtune**
    **cachegrind/valgrind**
    **It would be useful to know what others do here!**
    **gprof**
    **External Profilers**
    **some prof/gprof tools**
    **Instruments (macosx/xcode)**
    **We intend to do that as soon as things run stably.**

19. Would integration of additional performance analysis tools, such as PERISCOPE, TAU, SCALASCA, and KOJAK, into the PETSc distribution be useful to you?

    Yes – **13 users**
    No – **4 users**
    I don't know – **29 users**

    **I only use free, open-source performance tools.**
    **Tau is very useful.**
    **SCALASCA**
    **Have a GUI for performance analysis is cool.**
    **I don't know these tools.**
    **TAU**
    **I suspect this would be useful.**

20. Do you desire any additional profiling, tracing, and performance analysis capabilities in PETSc? If so, please comment on what additional capabilities you would like to see.

    **No**

    **It would be nice to have an accurate breakdown of load imbalance in the solvers in order to verify whether an increase in global reduction time is due to communication bottlenecks or load imbalance projected onto a global synchronization point...make sense?**

    **N/A**
    **No**
    **Have TAU integration sees cool.**
    **No.**
    **TAU**
    **Not at the moment**

21. Please select the performance tuning measures that you adopted for your application(s), if any?

    Memory Preallocation – **13 users**
    OpenMPI binding – **15 users**
    MPICH2 binding – **11 users**
    GPU binding – **3 users, Comments: Still in a preliminary stage**
    Tuning the fill ratio for sparse matrix – **26 users**
    **Comments: Custo**
    **This helps most**
    Data structure reuse – **7 users**
    Numerical experiments to determine solvers, preconditioners, and the parameters – **11 users**

Other – **3 users, Comments: Microsoft HPC Binding**

22. Would it be useful to you to have automatic OpenMPI, MPICH2, and GPU bindings in PETSc?

    Yes – **33 users**
    No – **4 users**

    Comments:

    **Don't know. I would have to better familiarize myself and investigate the benefits.**

    **Yes, but only for GPU bindings. I don't see any value in having direct bindings to OpenMPI and MPICH2 ... and why you didn't mention**

    **MVAPICH2? Please do not underestimate configurations with Infiniband.**

    **GPU binding maybe useful.**

    **I don't know or don't understand the term "automatic".**

    **Not the MPI bindings, but possibly the GPU ones.**

    **I do not understand the question**

23. We are integrating source transformation tools with PETSc to enable performance tuning of key computational kernels. PETSc can then be tuned for a set of benchmarks or for a given application instance at installation time. Would it be helpful to you to have such tools for your application(s)?

    Yes - **24 users**
    No – **7 users**
    I don't know

    **Comments:**

    **Again, a better preconditioner is more important.**

    **Not for me, but I believe that this would be very useful in general.**

**I use complex scalar types. I switch to FORTRAN kernels some weeks ago and noticed a performance improvement. That would be great if automatic...**

**Our build system is convoluted enough already.**

**:) Of course, if this would result in a huge performance increase, it could be worth it after all.**

24. Please suggest any benchmarks or application data that you would like us to consider for creating custom-tuned versions of PETSc? Please provide URLs, if any.

    **Unstructured elasticity finite-element calculations using Sieve.**

    **Any solution of large transient (include a time derivative) global implicit system of equations to test iterative solvers.**

    **N/A**

    **Please make us available a benchmark for unstructured finite volume method. Specifically interfacing a**

    **PETSC to existing application solver to solve already decomposed domain into multiple zone.**

    **n/a**

    **Specific parallel preconditioners**

    **Large 3D multiphasic porous media simulations**

    **RANS flow calculations on highly stretched grids.**

25. What, in your opinion, are the main performance limitations of PETSc and how do they impact your goals?

    **Some data structures sacrifice larger memory use for generality. Sometimes it is necessary to create limited use data structures that are fast and have small memory use**.

    **The main drawback for me is not to exploit threads within a multicore node (as an option of course).**

**The main performance limitations are algorithmic in the solvers and I/O. The I/O has a larger impact at medium core counts (10K-50K). The solvers (large interation counts, cost of iterations) overwhelm at 50+K cores.**

**PETSc do not provide "block vectors". In other words, if I have 2 systems, A1*x = b1 and A2*x = b2, the only way I have to solve the vertically stacked system is to build a shell matrix for the stack of A1 and A2 (and that is OK) but I do not have any way for creating the "shell vector" that would represent the stack of b1 and b2. So, I need to create a new vector and scatter b1 and b2 into it. The loss of performance is due to the required VecScatterBegin and VecScatterEnd calls. Other performance issue are present in all the functions that compose operations. For example (but, I stress, this is only an example) some time ago I've seen that the MatCreateComposite function does a lot of allocation and de-allocation during its operations. I think that this can be avoided pre-allocating auxiliary vectors and never destroying them. I've also experienced severe performance degradation while loading/saving dense matrices to binary files. I've seen programs crashing due to these operations, and the reason is that those operations are currently memory consuming (a lot more than what expected, and I don't know why).**

**Linear solves are always the bottleneck. PETSc is fine**

**The condition number in our application is very poor. As a result, iterative solver usually can't converge. Direct solver is the bottom neck.**

**The existing options for sparse-direct solvers (e.g., MUMPS and SuperLU_Dist) were not nearly scalable enough, both time and memory wise, so I had to write my own. Also, PETSc does not yet support Krylov algorithms with multiple right-hand sides, which was crucial for my application.**

**Maybe not having vectors of different types at the same time.**

**n/a**

**As more computing power moves to GPUs I am concerned that PETSc will not be as useful.**
**The main performance limitations of PETSc are the basic limitations of large scale sparse scientific computing: lack of optimal black box preconditioners (especially in matrix free situations), and low arithmetic intensity in sparse matrix multiplication. Eliminating these problems is difficult to impossible, though algebraic multigrid is promising.**
**matrix assembly for unstructured grids**

**none? Maybe available preconditioners**

**I stopped using petsc about 3 years ago because of a lack of support for general block sparse matrices (i.e variable sized blocks). If support for this were improved I might consider using petsc again.**

**Slow convergence of the linear solvers for RANS flow calculations. Basically the preconditioners are not good enough. I tried AMG, but that does not work too well. The only thing that does work is underrelaxation by making the matrix more diagonally dominant. However, this increases the CPU time significantly.**

## *Linear Solvers, Preconditioners, and Data Types*

**The questions in this section are specific to structured grid applications. Please skip to the next section if you don't use PETSc to develop structured grid applications.**

26. Have you developed or anticipate developing structured grid applications with the PETSc library?
    Yes – **31 users**
    No – **12 users**
    **Multiple solvers for unstructured problems. Structured are considered.**
    **The structured side of PFLOTRAN is founded upon the PETSc DM/DA.**
    **I would like to have more support concerning staggered structured grids.**
    **With LLNL samrai**
    **Structured grid generation is a pain, it's just not flexible enough for CFD apps**
    **Various types**

27. What is the domain of your application(s)? In the textboxes, please provide any details on the subdomain such as CFD, Seismic Wave Modeling, and Reactive Flow simulation.

    Mechanical Engineering – **19 users**
    **Comments:**
    **CFD**
    **CFD, Fluid structure interaction**
    **CFD**
    **Multiphase flow simulations**
    **Levelset + stress**

**CFD, Incompresible flow**
**CFD**
**CFD**
**CFD, particle methods**
**CFD, porous media simulation**
**CFD of wind turbines**

Civil Engineering – **6 users, Comments: Fluid, CFD**

Chemical Engineering – **3 users Comments: CFD + energy conversion**

Electrical Engineering – 3 users, **Comments:**
**Electro hydrodynamics, Sparse ill-conditioned electromagnetic wave problems**

Aeronautical Engineering – **7 users, Comments:**
**Computational aerodynamics Reactive Flow simulation**
**CFD**
**cfd**
**CFD**
**CFD on gas turbine engines**

Geophysics – **11 users Comments:**
**Earth**
**FD Ma**
**Seismic Wave Modeling**
**EM modeling, Inverse problem**
**Reflection seismology**
**Gravity and turbidity currents occurring in oceans**
**Multiphase flow**
**Solid earth geodynamics**
**EM simulation**

Earth Sciences – **5 users**
**Comments:**
**Fluid**
**Long and short term crustal processes**
**Magma dynamics/reactive flow**

Other (Please specify)
**Astrophysics**
**Physics**
**Maritimn engineering**
**Graphics**
**Computer Science**
**Computer Science**

**Wave (quantum) mechanics**
**Plasmaphysics**

**Comments:**

**Fusion energy**
**CFD, volumetric solids, solid membranes**
**Turbulence**

28. Please provide known details on the memory requirement of your application(s). For example, how much memory is required per grid point?

**Responses:**

**Always more :)**

**Couldn't tell you as it varies based on problem complexity. We generally solve for 1-3 dofs per cell for flow and 1-20 dofs per cell for reactive transport, but there are all kinds of supporting variables/arrays depending on the number of processes incorporated in a simulation. I suspect that each grid cell could require storage of up to 5KB for a large complex geochemistry problem with multirate kinetic sorption.**

**1e6 grid points, 1e4 bytes/grid point, in total ~10 GB memories.**

**1.5 MB**

**About 1 MB per grid point**

**Variable,**

**Around 30 double precision values per grid point.**

**Sizeof (float) * 20**

**Up to 200 kb per grid point**

**Approx. 1K per node.**

**All of the memory usage is in the preconditioner.**

**3 Kb.**
**Approx. 15+3=18 variables per grid point (5 variables, each having 3 time levels; x, y, z coordinate values)**

**Typically 3 floats of solver variable per node, plus between 6 and 30 ints/floats of known information.**

**120KB**

**Requirement is fairly large, 4 GB/node for a bigger run**

**2GB**

**Approximately 4kb per grid block (maximum application use, unoptimized)**

**100mb - 1000 GB**

**O(5-7) doubles/dof**

**Without PETSc it is roughly 1Kb per grid point. With PETSc it really depends on the preconditioner, but typically a factor of 25 to 100 more.**

29. Did you ever use PETSc in single precision mode ("--with-precision=single" when running ./configure)?

    Yes – **3 users**
    No – **41 users**
    No Answer

    **Comments:**

    **I don't think it is worth to try. Precision is important**

    **double and long double.**

    **I may try it at some point (to test performance gains)**

    **I have in the past, but am not currently. It is unfortunate that common automatic installers such as MacPorts support only double precision.**

30. Did you ever use PETSc in mixed precision mode ("--with-mixed-precision=1" when running ./configure)?

    Yes
    No – **40 users**
    No Answer

    **Comments:**

    **This might be a good thing to try, though**

**Meaning less**
**No, but I do think that this is useful.**
**I may try it at some point (to test performance gain).**

31. Please list any "software design tradeoffs" you made while developing your structured grid application with PETSc.
    For example, simulation time for accuracy of results.

    **The main tradeoff is to give up native threads on multicore CPUs. Using only MPI results in memory copying within a node.**

    **We generally chose better accuracy and throw more cores at it, although our flow and transport algorithms are not high order accuracy to begin with (upwinding for advection terms).**

    **Accuracy and maintainability always more important than ultimate performance.**

    **Simulation time**

    **Solver transient nonlinear problem with only one newton iteration by linearize the problem.**

    **KSP solvers and preconditioners are very memory consuming. I had to make some bad design decisions in my code to "optimize" memory usage. For instance, I do not use any preconditioners for the solution of velocity and scalar(s) transport equation to save memory. I precondition the pressure Poisson equation with BoomerAMG though which is very memory demanding.**

    **n/a**

    **Unsure**

    **Took a bit of a performance hit on serial behavior wrt to my hand-rolled serial multi-grid codes but gained considerable flexibility**

    **None.**

32. What linear solver(s) did you use for your application? Please select more than one if applicable. In the text boxes, please provide reasons, if any, for choosing the particular solver.

Richardson – **5 users, comments: using BoomerAMG thru PETSC**
Chebychev – **2 users**
Conjugate Gradient – 20 users

Comments:

**For normal equation in inverse problem**
**Symmetric Laplacian operator**
**Typically, my problems can be formulated as SPD.**
**Matrix symmetric**

BiConjugate Gradient – 5 users

Comments:

**This is the fastest BiConjugate Gradient method. I've seen working on relatively simple problems.**

Generalized Minimal Residual – **29 users**

**Bench**

**Non symmetric, non-positive definite matrix**

**GMRES (20)**

**It showed the best performance with boomerAMG preconditioner.**

**Flexible GMRES**

**My problems are almost always symmetric, but are occasionally indefinite. It's unfortunately that PETSc is missing dedicated symmetric indefinite solvers.**

**Matrix non-symmetric, rel. stable**

**Standard**

BiCGSTAB – **17 users**

**2x fa**

**For Electromagnetic modeling. Since matrix is non-hermitian.**

**Powerful than GMRES, default solver in our application**

**Fast and less memory demanding than GMRES**

**Following recommendations from journal papers**

**Matrix non-symmetric**

Conjugate Gradient Squared – **3 users**

Transpose-Free Quasi-Minimal Residual – 3 users

**Comments:**

**For Electromagnetic modeling. Gives smooth convergence behaviour in comparison with BiCGStab.**

Conjugate Residual – **2 users**

Least Squares Method – **3 users**

**Comments:**

**For normal equation in inverse problem**

**Matrix non-symmetric, first tests**

Shell for no KSP method – **9 users, Comments: Used**

Other (Please specify) – **fgmres, none, MUMPS, GCR, mumps, GMRES, multigrid**

**Comments:**

**For iterative solver as preconditioner**

**Used for most difficult problems.**

**Direct solvers are much better than iterative solvers**

33. What preconditioners have you used or anticipate using for your application? In the text boxes, please provide reasons, if any, for choosing the particular preconditioner.

Jacobi – **8 users, Comments: Cheap**

Block Jacobi – **16 users, Comments: ILU (0), Easy to use from PETSc command line.**

SOR (and SSOR) – **3 users**

SOR with Eisenstat Trick – **1 user**

Incomplete Cholesky – **8 users**

Incomplete LU – **24 users**

**Comments:**

**Works**
**Local PC for linear solver/ block jacobi**
**ILU (K), k in 0-2**

Additive Schwartz – **12 users**

**Comments:**
**Fast**
**We ha**
**Anticipate to try**

Linear Solver – **6 users**

**Hypre**
**To parallelize ILU, empirically found to work well**

Combination of Preconditioners – **5 users**

LU - **14 users**

**Comments:**
**When using SuperLU_Dist**
**Only for small problems for which ILU fails**

Cholesky – **4 users**

**When using MUMPS**

No Preconditioning – **6 users, Comments: cheap; special discretisation no allowing any standard pc**

User-Defined Preconditioning – **18 users**

**Comments:**

**Field**
**Physi**
**None**
**Helmholtz is hard.**
**BoomerAMG (part of hypre package)**
**Block preconditioners for saddle point problems**
**GMRES**
**For my own testing**
**Block diagonal**
**AMG hypre/ml, MG**

34. If you used parallel preconditioning (Block Jacobi or ASM), what are the local preconditioners?

**BJacobi with ILU [0]**
**ILU**
**LU**
**ILU**
**ILU or LU**
**ILU**
**ILU**

**Incomplete Cholesky or block diagonal, typically.**
**ILU**
**Jacobi**
**ILU**
**ILU and ICC**

35. If you used composite preconditioning, what is the combination of preconditioners?

**Field split with ML on displacement DOF with custom preconditioner for Lagrange multipliers**
**associated with the fault**
**hypre/boomeramg/jacobi/custom**
**N/**
**Multiplicative**
**deflation + ASM + Fieldsplit + ILU**
**ML, SOR, MUMPS**
**n/a**

36. Do you desire any preconditioners that do not exist in PETSc? If so, please provide any details.

**Native (within PETSc) multilevel preconditioners**

**I would like to have a parallel Cholesky decomposition that returns the Cholesky factor and, optionally, directly the inverse of the Cholesky factor**

**Well-defined and documented multigrid is a dream.**

**Some smart precoditioner for schur complement.**

**Sweeping preconditioners, but they are...complicated and unlikely to be incorporated into PETSc in the near future.**

**Approximate inverse PC for complex scalar type**
**Geometric multigrid from Hypre's struct and sstruct interfaces**

**AMG**
**Boomeramg**

**Memory efficient implementations of ILU**

**Algebraic Multigrid**

**Easy way to create your own preconditioner.**

37. What data types in PETSc have you used for storing the matrices? In the text boxes, please provide reasons, if any, for choosing the particular data type.

    The Default type (The default MATAIJ format is identical to MATSEQAIJ when constructed with single processor communicator and MATMPIAIJ otherwise) – **22 users, Comments: A lot**

    Dense Matrix (MATDENSE) - **4 users, Comments: My Jacobian is dense, poten**

    Dense Sequential Matrix (MATSEQDENSE) – **2 users**

    Dense Parallel Matrix (MATMPIDENSE) – **2 users, Comments: My Jacobian is dense**

    Sparse Sequential Matrix (MATSEQAIJ) – **11 users**
    **Serial only**
    **Discretisation gives sparse matrix, serial**
    Sparse Parallel Matrix (MATMPIAIJ) – **21 users**
    **Minim**

    **Data is sparse and parallel**

    **For parallel simulation**

    **Returned by DMDAGetMatrix (). I use a finite difference method. So, I make use of the distribute arrays**

    **To use MPI across multiple compute nodes**

    **Discretisation gives sparse matrix, parallel**

    Blocked Sparse Matrix (MATBAIJ) - **9 users, Comments: Often**
    Blocked Sparse Sequential Matrix (MATSEQBAIJ) – **3 users**
    Blocked Sparse Parallel Matrix (MATMPIBAIJ) – **7 users**
    Other (Please specify): **Shell, MATSHELL**
    **Comments: Interested in MatNEST**

38. Have you ever implemented your own PETSc data type for storing a matrix or vector? If so, please provide details on your custom data type.
    **No.**

    **Only as shell matrix. Basically, I've re-implemented block matrices, to overcome the memory allocation/deallocation inefficiencies that PETSc provides (and that I was talking of before)**

**Yes, basic sparse block row binary storage**

**No**

**I had extends MPIAIJ for dynamic memory allcation support. And send the patch to petsc developers. However, they are always busy.**

**n/a**
**No.**
**No**
**No**
**no**
**No**

39. Do you desire any custom data structures for storing your matrices, vectors, and other application data that could enhance the performance of your applications?

**Not sure.**

**Yes. There is the need for a block vector or shell vector. See my explanation in the section of performances.**

**No**

**A matrix format where the five (or n) main diagonals are stored in an array similar to the one used by lapack for banded matrix inversion would be useful for implementing a block Gauss Seidel relaxation process on structured grids.**

**No**

**Maybe a sparse vector is useful? BAIJ matrix only support fixed block size. This is too bad for a multi-physics problem. We have to use AIJ matrix instead.**

**Parallel (direct) tridiagonal (periodic and non-periodic) solver**

**No.**
**No**

**It might be interesting to expand the DA or DM to include staggered structured meshes**

**No**

40. What stencil types did you use or anticipate using for your structured grid application along with distributed arrays?

    Star type – **5 users**
    Box type – **4 users**
    Both – **14 users**
    None – **1 user**

    **Our code is unstructured.**

    **I use unstructured for now.**

    **We have custom dynamic stencils that choose values depending on gradients etc (ghost fluid method, immersed interface method)**

    **Will be switching to manual spectral elements soon.**

    **Box type is used since I am dealing with complex geometry and some interpolation schemes which require non-orthogonal stencil.**

    **Star for Poisson equation. Box for staggered-grid Navier-Stokes equations (finite difference).**

    **For inviscid applications it is a star type stencil, but for viscous it is a box type.**

41. What are the typical stencil width(s) for your application(s)? For a Star type stencil, a width of 1 corresponds to the standard 5-point stencil. For a box type stencil, a width of 1 corresponds to the standard 9-point stencil.

    **1, but considering 2 for higher order methods in future.**

    **3-D spectral, i.e. ~20**

    **Standard 9-point stencil**

    **2 is typical width**

    **2**

    **2**

**Between 1 and 4.**

**9-point stencil**
**As high as 3 for fifth order WENO scheme used in levelset.**

**7-point 3d stencil.**

**Star-stencil: width 1 (in both 3D and 2D) Box-stencil: width 3 (both 3D and 2D). I do not allocated the matrix and vectors based on the Box-stencil. I only use the DA layout to know the indices of the nodes and for very few node extra non-zeros are added to the matrix originally created based upon the start-stencil.**

**1 (but I will use "2" for an high-order version discretization)**
**2**
**1, 3**
**1**
**1 and 2**

**1, 2 for some semi-lagrangian methods**

**2 to 5. 2 for the second order schemes. 5 for the fifth order scheme, although this scheme is usually run in matrix-free mode.**

**5**

42. Are the box-type and star-type stencils in PETSc sufficient for your needs? Do you desire any other stencils for your application(s)?
    **No, we need unstructured finite-element meshes.**
    **Yes.**
    **Didn't know PETSC provides stencils.**
    **Yes, sufficient**
    **They are sufficient**
    **They are sufficient**
    **YES**
    **Enough**
    **They are enough.**
    **Yes. I currently do not need other stencils.**
    **Yes**
    **Yes, but is easy to create with additional scatters**
    **General unstructured**
    **Yes**
    **Sufficient**

## *Miscellaneous*

43. What debuggers have you used with your PETSc application?
    dbx
    xdbx
    gdb – **32 users**
    Other (Please specify) **Visual Studio, Totalview, kdbg, Valgrind, None, Totalview, TotalView, Valgrind, Valgrind, Msvc, Valgrind, Totalview**

    **Comments:**

    **KDBG does not work as expected. It cannot be launched automatically by PETSc**

    **Use Valgrind to check memory problem**

44. Are there any other debugging capabilities that you desire in PETSc?

    **No**

    **I plan to try out the parallel debugger available in Eclipse PTP module.**

    **Yes; parallel debugging is difficult**

    **No**

45. Would it be useful to you if we integrate the most recent Zope framework into the PETSc distribution? Zope (http://bluebream.zope.org) is a tool that generates a web page containing all output of a PETSc program. It stores the output of previous run and makes it easy to compare the outputs.
    Yes – **13 users**
    No – **10 users**
    No Answer

    **Comments:**

    **Don't know that I would use it. Is it better than diff?**

    **I am not aware of that software.**

    **Zope seems heavy, but ok if it will ease debugging and visualization**

    **Haven't used Zope**

46. Do you desire any other capabilities for visualizing and analyzing the results from your PETSc applications?
    **No.**

    **I always disable PETSc graphical tools. I think PETSc should concentrate on the computations, and let petsc4py to interface with Python, where there are a lot of tools for visualization. Once**

**you have saved in binary files the results, you can view them from Python.**

**I use PetscViewerBinaryMatlabXXX() to generate self-documenting output containing a variety of PETSc data structures (PetscBag, Vectors, Vectors on DAs). It would be nice if there was a similar thing for HDF5. Jed has been working on something but I don't think it can handle PetscBag objects, and not sure about its ease-of-use.**

**Yes, I would prefer PETSc to write vtk files in parallel format for post processing with ParaView.**

**No**

**No**

**No, we have our own visualization tool base on VTK.**

**Too much to ask, I know, but can i also be interfaced to VTK package?**

**More VTK and Paraview support, if possible. For Paraview, it would be nice to support parallel (XML) output format (as defined in http://www.vtk.org/VTK/img/file-formats.pdf) so it will be easier to use it in parallel paraview (for large data set).**

**Built in vtk writer**

**No**

**Support for residual plot, introspection, etc., on windows... for linear and non-linear solvers.**

**Maybe improved hdf5/vtk support**

**To MatLab**

**No**

47. Please use the following box to further comment on any aspects of PETSc.

**Keep up the good work!**

**I would only like to summarize features that could enhance PETSc a lot: Possibility to use real and complex numbers with only one build.**

**Possibility to use shell vectors. Possibility to have Cholesky decomposition able to return the Cholesky factor or the inverse of the Cholesky factor. More optimization on memory allocation/deallocation. More optimization on the load/save operations for dense matrices - Enhance the overall documentation. A lot of features are difficult to be used simply because the documentation is poor.**

**PETSc is a great library. Keep going ;) Hopefully you can extract something from my answers.**

**I think you should not focus on structured grids. It's useful for academic purposes but not for engineering**

**PETSc and the PETSc team is one of the best things that could possibly happen to application developers like myself. Really!**

**Really great piece of software**

**I would like to use a very fast Jacobian-free non-linear newton solver implemented in PETSc**

**Note that my most recent experience with PETSc was with version 3.0.0-p9, i.e. my comments may be invalidated by changes in newer versions.**