AMPHOS 21

SCIENTIFIC AND STRATEGIC ENVIRONMENTAL CONSULTING

Use of PFLOTRAN on JUQUEEN - A User's Perspective

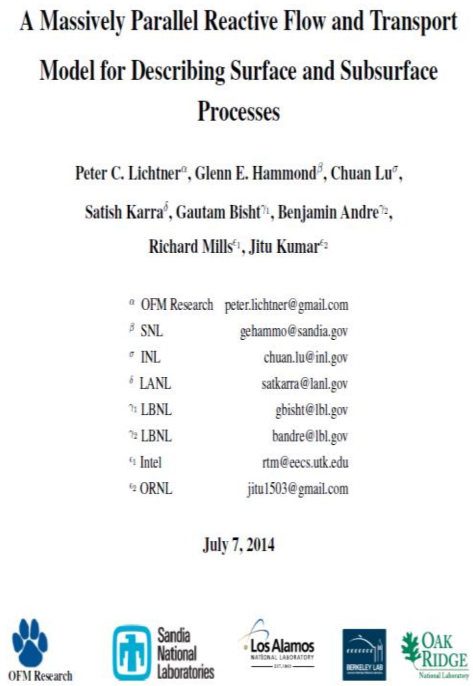
Hedieh Ebrahimi
Paolo Trinchero

1994-2014 20 YEARS FLOWING

February 2015

What is it?

- PFLOTRAN is reactive transport simulation program, developed with High Performance Computing in mind.
- It is under active development by a group of people connected to research institutes as open source code (LGPL license).
- It is programmed in Fortran 2003
- It heavily depends on other open source HPC frameworks, like MPI, PETSC and HDF.
- Main developers:
  - Peter C. Lichtner,
  - Glenn E. Hammond,
  - Chuan Lu, Satish Karra, Gautam Bisht, Benjamin Andre, Richard Mills, Jitu Kumar



A Massively Parallel Reactive Flow and Transport Model for Describing Surface and Subsurface Processes

Peter C. Lichtner[α], Glenn E. Hammond[β], Chuan Lu[σ], Satish Karra[δ], Gautam Bisht[γ1], Benjamin Andre[γ2], Richard Mills[ϵ1], Jitu Kumar[ϵ2]

[α] OFM Research	peter.lichtner@gmail.com
[β] SNL	gehammo@sandia.gov
[σ] INL	chuan.lu@inl.gov
[δ] LANL	satkarra@lanl.gov
[γ1] LBNL	gbisht@lbl.gov
[γ2] LBNL	bandre@lbl.gov
[ϵ1] Intel	rtm@eecs.utk.edu
[ϵ2] ORNL	jitu1503@gmail.com

July 7, 2014

OFM Research
Sandia National Laboratories
Los Alamos NATIONAL LABORATORY
BERKELEY LAB
OAK RIDGE National Laboratory

High Level Concepts

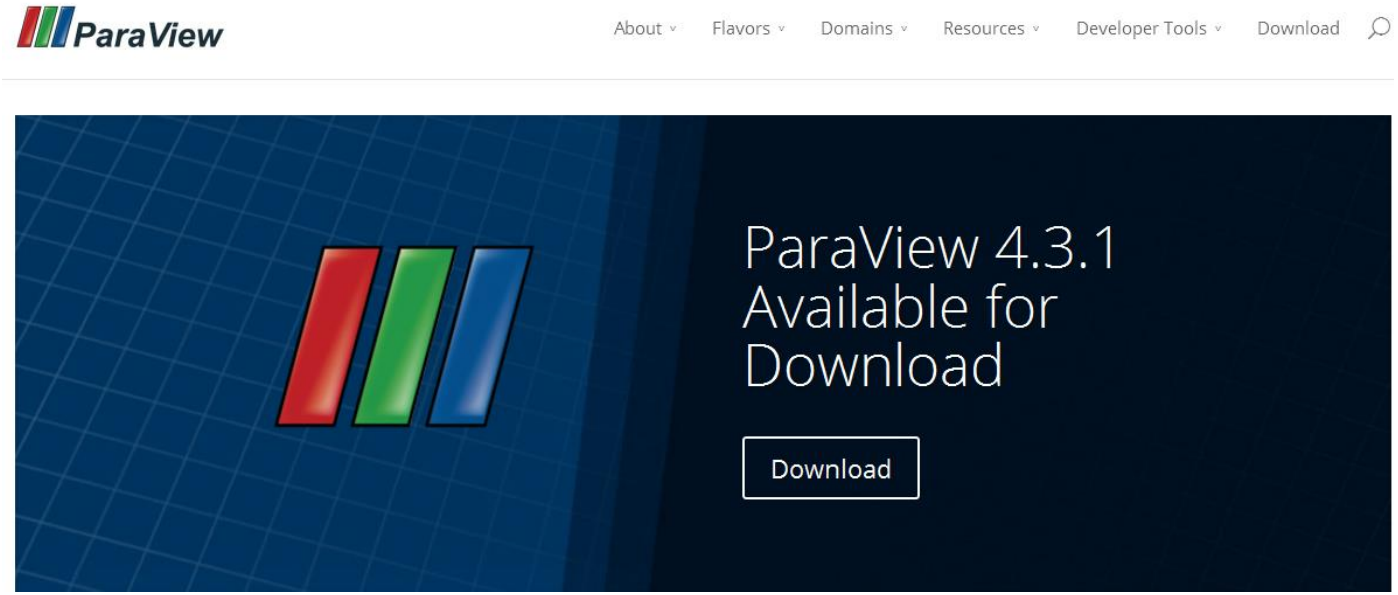
- Structured grids (generated)
- Unstructured grids (imported)
- Unsaturated & Saturated Flow
- Makes use of the very powerful…and very complex PETSc library
- Can read & write in parallel (HDF5 format)
- Open Source (LGPL license)
- Up to now provides several modes (e.g. RICHARDS, Water/Air, FLASH2, etc.)
- Complex chemistry options
- Moving to a more general multi-physics framework
- Under heavy development
- Limited documentation (and not always up to date)

A bit more about CHEMISTRY

- Makes use of primary & secondary species (comparable with ToughReact, Crunch, etc.)
  These species can be found in the thermodynamic database.
- (only) Kinetic mineral dissolution & precipitation (TST)
- Sorption:
  - Surface complexation
  - Ion Exchange
  - Mineral interaction
  - Colloid interaction
  - Isotherm reactions
- Operator Splitting (Could have been removed in refactoring…)
- Updating of permeability, porosity, tortuoisity, mineral surface area

Post-Processing

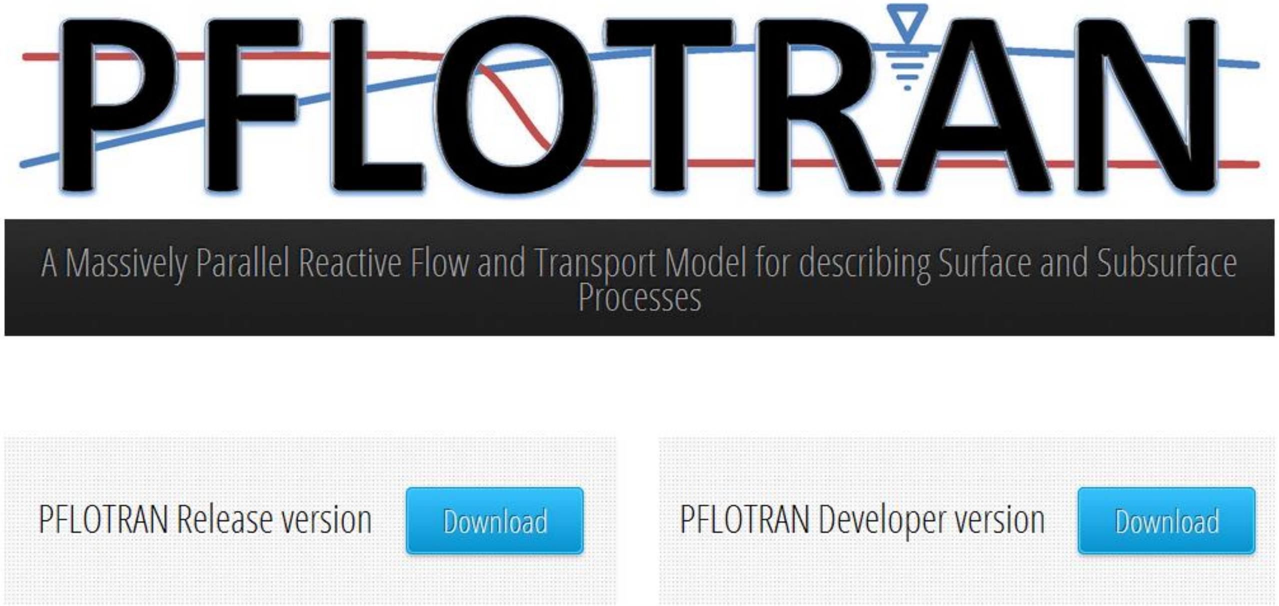
- PFLOTRAN can output to
  - Tecplot
  - PFLOTRAN HDF5
- PFLOTRAN HDF5 can be opened directly in ParaView
- ParaView download: http://www.paraview.org/download/



ParaView 4.3.1 Available for Download

More Information

- Home Page:
  http://www.pflotran.org/
- Wiki & Source Code:
  https://bitbucket.org/pflotran/pflotran-dev



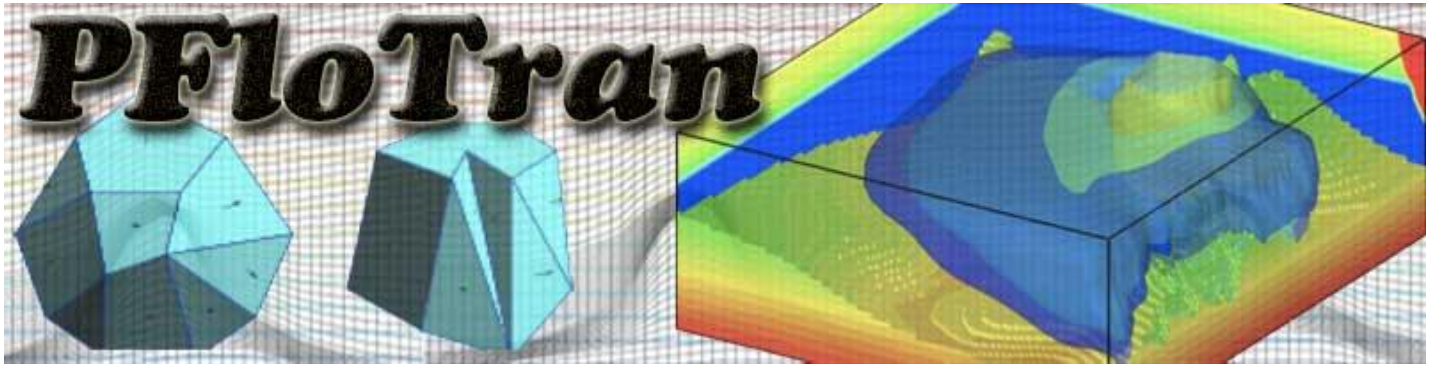
PFLOTRAN

A Massively Parallel Reactive Flow and Transport Model for describing Surface and Subsurface Processes

PFLOTRAN Release version

PFLOTRAN Developer version

PFLOTRAN Installation :



- **Build on Linux :** Although PFLOTRAN can be built for Windows, here only the Linux version is discussed. Compiling the Windows version involves installing cygwin (out of scope for this presentation). It is easier anyway to compile it directly on Linux Systems or on a virtual machine containing a linux distribution (that can run on windows).

- **No Installer available / Needs Compilation:** Because it is under current development, there is no executable available. If you want to use it, you need to know how to compile it (or you need to know someone who knows..).

PFLOTRAN – Making an executable

- There are 2 steps in building a PFLOTRAN executable
  - Building the PETSc library
  - Building the PFLOTRAN executable

PETSc

- PETSc is a library for solving partial differential equations http://www.mcs.anl.gov/petsc/
- PETSC is a suite of data structures and routines for the Parallel solution of scientific applications modeled by partial diferential equations. (Parallel Solution of PDE)
- PFLOTRAN -> Parallelization achieved through domain decompositon through PETSC.

PETSC Main Components:

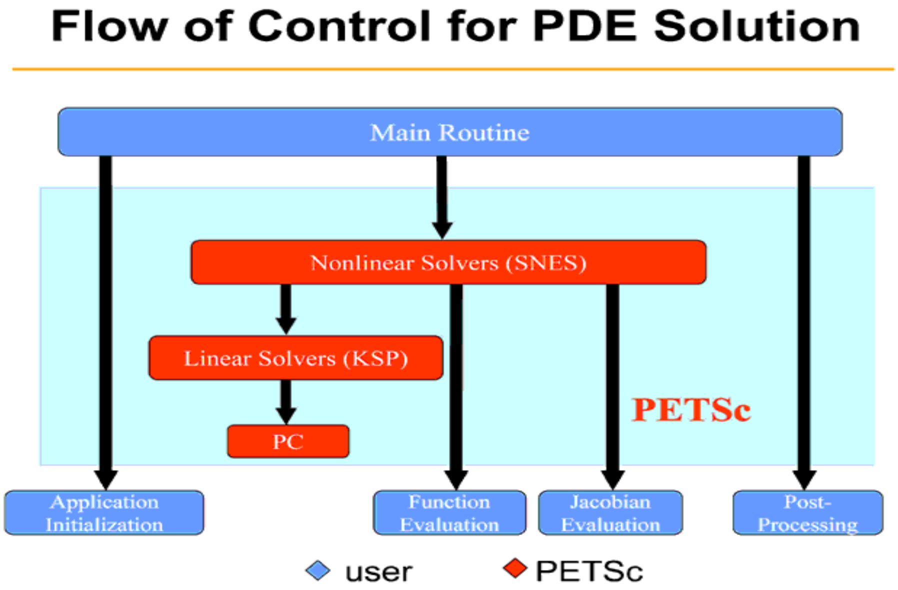
PETSc components provide the functionality required for many parallel solutions of PDEs.

SNES: Data Structure-neutral implementation of Newton-like

Methods for non-linear systems.

KSP: Parallel Implementation of many popular Krylov subspace iterative methods.

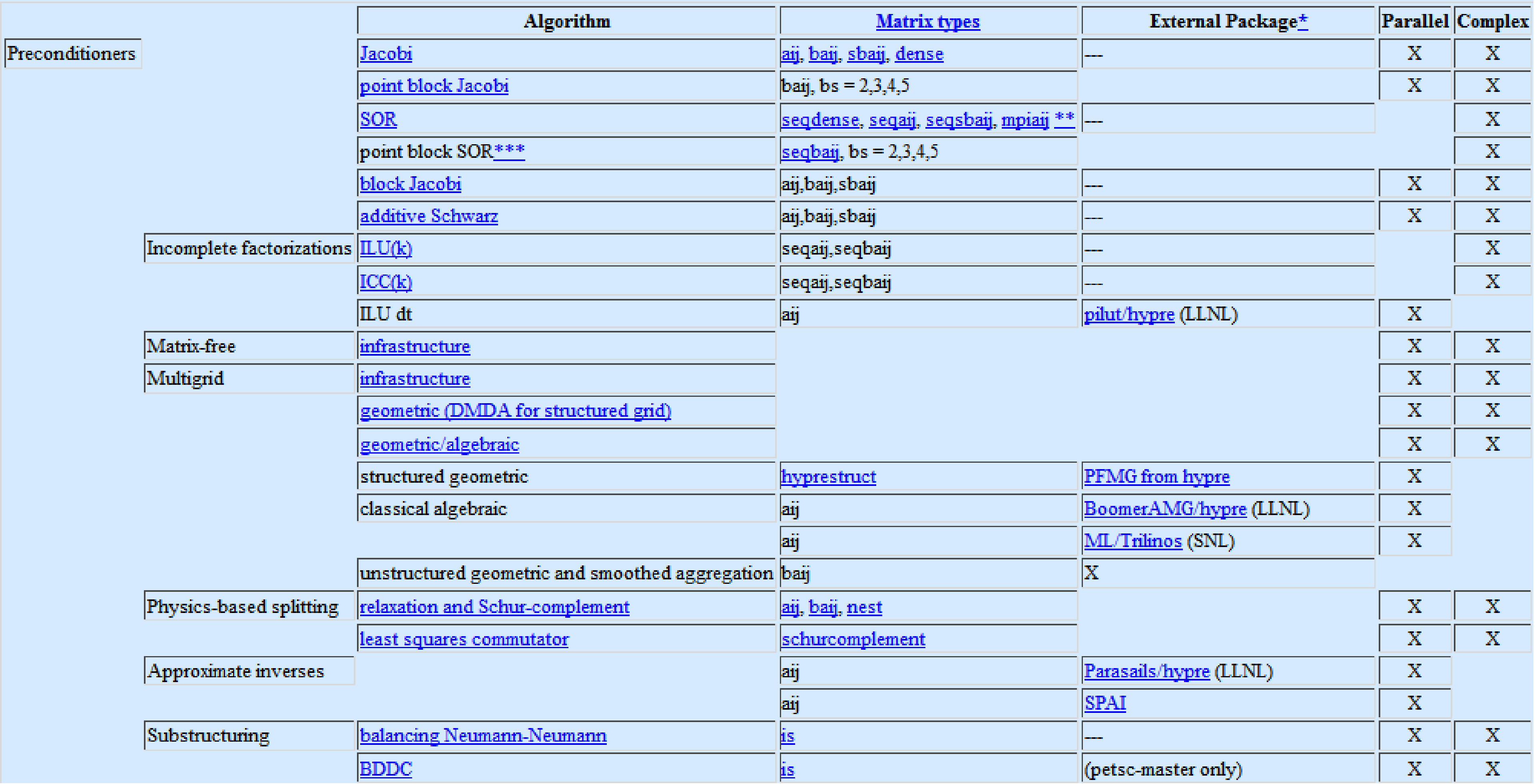
PC: A collection of sequential and Parallel Preconditioners, that condition problem in to a form that is more suitable for numerical solving Methods.



PETSC Solvers:

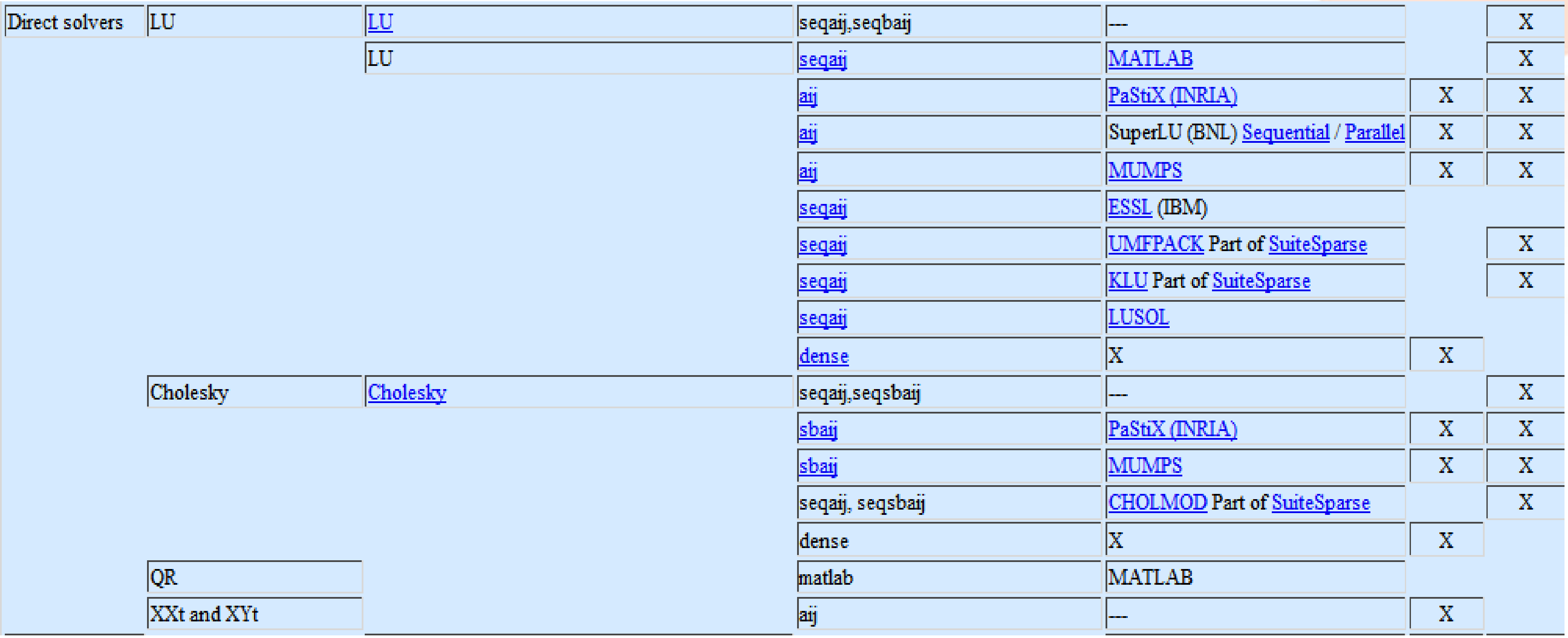
- Preconditioners
- Direct Solvers
- Krylov Methods

Preconditioners:



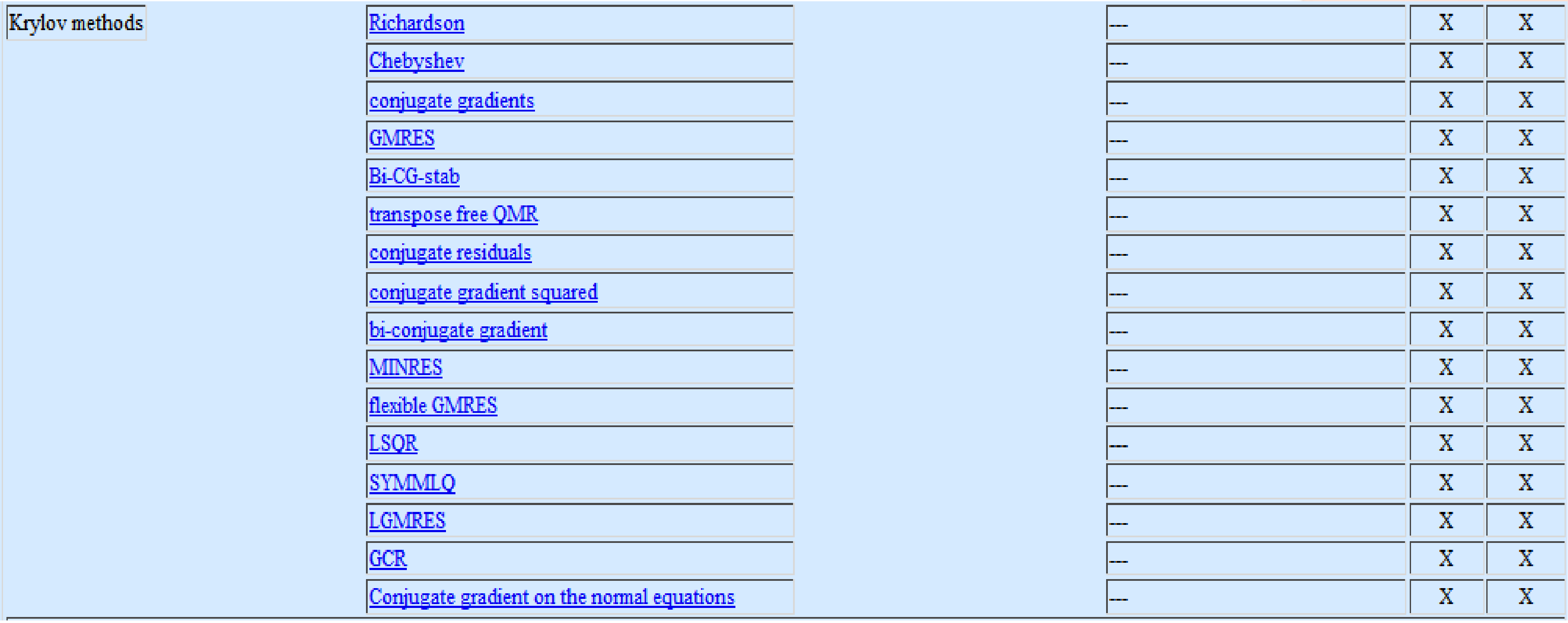
		Algorithm	Matrix types	External Package*	Parallel	Complex
Preconditioners		Jacobi	aij, baij, sbaij, dense	---	X	X
		point block Jacobi	baij, bs = 2,3,4,5		X	X
		SOR	seqdense, seqaij, seqsbaij, mpiaij **	---		X
		point block SOR***	seqbaij, bs = 2,3,4,5			X
		block Jacobi	aij,baij,sbaij	---	X	X
		additive Schwarz	aij,baij,sbaij	---	X	X
	Incomplete factorizations	ILU(k)	seqaij,seqbaij	---		X
		ICC(k)	seqaij,seqbaij	---		X
		ILU dt	aij	pilut/hypre (LLNL)	X	
	Matrix-free	infrastructure			X	X
	Multigrid	infrastructure			X	X
		geometric (DMDA for structured grid)			X	X
		geometric/algebraic			X	X
		structured geometric	hyprestruct	PFMG from hypre	X	
		classical algebraic	aij	BoomerAMG/hypre (LLNL)	X	
			aij	ML/Trilinos (SNL)	X	
		unstructured geometric and smoothed aggregation	baij	X		
	Physics-based splitting	relaxation and Schur-complement	aij, baij, nest		X	X
		least squares commutator	schurcomplement		X	X
	Approximate inverses		aij	Parasails/hypre (LLNL)	X	
			aij	SPAI	X	
	Substructuring	balancing Neumann-Neumann	is	---	X	X
		BDDC	is	(petsc-master only)	X	X

Direct Solvers:



Direct solvers	LU	LU	seqaij,seqbaij	---		X
		LU	seqaij	MATLAB		X
			aij	PaStiX (INRIA)	X	X
			aij	SuperLU (BNL) Sequential / Parallel	X	X
			aij	MUMPS	X	X
			seqaij	ESSL (IBM)		
			seqaij	UMFPACK Part of SuiteSparse		X
			seqaij	KLU Part of SuiteSparse		X
			seqaij	LUSOL		
			dense	X	X	
	Cholesky	Cholesky	seqaij,seqsbaij	---		X
			sbaij	PaStiX (INRIA)	X	X
			sbaij	MUMPS	X	X
			seqaij, seqsbaij	CHOLMOD Part of SuiteSparse		X
			dense	X	X	
	QR		matlab	MATLAB		
	XXt and XYt		aij	---	X	

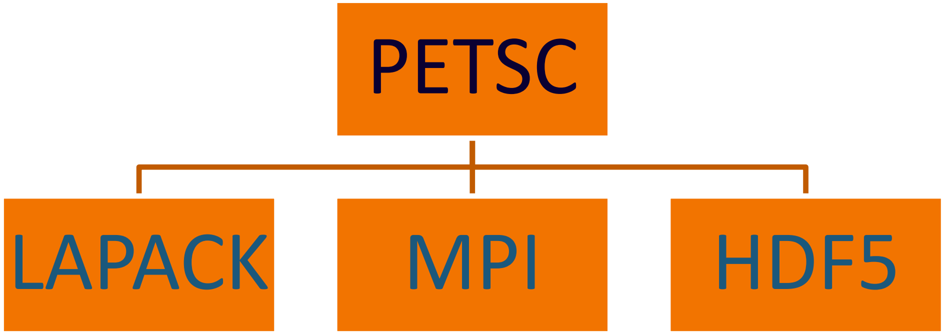
Krylov Methods:



Krylov methods	Richardson	---	X	X
	Chebyshev	---	X	X
	conjugate gradients	---	X	X
	GMRES	---	X	X
	Bi-CG-stab	---	X	X
	transpose free QMR	---	X	X
	conjugate residuals	---	X	X
	conjugate gradient squared	---	X	X
	bi-conjugate gradient	---	X	X
	MINRES	---	X	X
	flexible GMRES	---	X	X
	LSQR	---	X	X
	SYMMLQ	---	X	X
	LGMRES	---	X	X
	GCR	---	X	X
	Conjugate gradient on the normal equations	---	X	X

PETSC Uses External Packages/Libraries:

- PETSC is a solvers Package, but PETSC is more than a Solvers Package!
- Parallel Mesh and linear algebra object Management.
- Nonlinear Solvers
- Linear (Iterative) Solvers
- Performance Logging and Debugging
- Interface to many other Packages.



Linear Algebra Libraries

- GNU Scientific Library
- LAPACK
- MKL
- ….

Linear Algebra Libraries	Language	Details
GNU Scientific Library	C	GNU Project FREE / GPL
LAPACK	Fortran	FREE
MKL	C/C++/Fortran	Intel
Scipy	Python	Free/BSD

- **How PETSC Related to PFLOTRAN ?**

PFLOTRAN Flow Mode:

- **MODE Card**: Specifies the flow mode.

- Options:
  - TH
  - *Thermo-hydro - Coupled groundwater flow and thermal*
  - **RICHARDS**
  - ***Single-phase variable saturated groundwater flow using Richards Equation***
  - MPHASE
  - *Multiphase CO2-water-energy*
  - GENERAL
  - *Multiphase air-water-energy*
  - IMS, IMMIS, THS
  - *Immissible CO2-water-energy*

- Note: Flow comes from iDP which uses the DarcyTools MIGAL Solver.

PFLOTRAN Solvers:

- The flow and heat equations (Modes: RICHARDS, MPHASE, FLASH2, THC, . . .) are solved
- using a fully implicit backward Euler approach based on Newton-Krylov iteration.
- Both fully implicit backward Euler and operator splitting solution methods are supported for reactive transport.

- **NEWTON_SOLVER** CARD: Specifies nonlinear solver parameters associated with convergence and Jacobian matrix format for FLOW or TRANSPORT.

- **NEWTON_SOLVER TRANSPORT**
  - **ATOL 1D-11**
  - **RTOL 1D-07**
  - **STOL 1D-07**
  - **DTOL 1.D10**
  - **ITOL 1D-7**
  - **MAXIT 50**
  - **MAXF 100**
  - **MAX_NORM 1D4**
- **END**

- **Other Libraries**

MPI: Message Passing Interface Implementations:

- MPI is a standardized and portable message passing system designed by a group of researchers from academia and industry to function on a wide variety of parallel computers.

- MPICH 2
- IBM MPI
- INTEL MPI
- OPEN MPI
- MICROSOFT MPI
- CRAY MPI
- …

HDF

- HDF5 is a data model, library and file format for storing and managing data. It supports an unlimited variey of datatypes and is designed for flexible and efficient I/O and for high volume and complex data.
- HDF5 is portable and is extensible, allowing applications to evolve in their use of HDF5.

- includes tools and applications for managing , viewing and analyzing data in HDF5 format.

- **How to Build PETSC and PFLOTRAN ?**

PFLOTRAN – Getting the PETSc source code

- The source code is stored here: https://bitbucket.org/petsc/petsc
- PETSc is under continuous development. The source code is stored in a git repository (git = software configuration management (SCM) system)
- To copy the remote repository into your local repository:

```
git clone https://bitbucket.org/petsc/petsc petsc
```

- Check on the PFLOTRAN wiki which git commit you need to use (https://bitbucket.org/pflotran/pflotran-dev/wiki/Installation/Linux Installation instructions – step 3.1):

```
cd petsc
git checkout 0c8c14c5
```

(This code will be regularly updated on the website)

(note that you only need to "run clone" once, afterwards you use "git pull" to update the local repository)

PFLOTRAN – Configuring PETSc source code

- Now that you have the source code, you need to configure it.
  This step will check the compilers, download required libraries and set all the compilation (and link) flags correctly.
  If the compilers are not installed correctly, this step will fail, so make sure that the fortran and c compilers are installed.
- A basic configure command looks as follows:

  ./configure --with-debugging=0 --download-mpich=yes --download-hdf5=yes
  (line continues:)   --download-fblaslapack=yes PETSC_ARCH=rel_ 0c8c14c5

- The flag "—with-debugging=0" defines if you want to having debugging information in the library. To create a fast library use 0, If you are having problems, use 1.
- MPI is the library to communicate between different computers (or processes on the same computer). There are 2 main version of MPI: openMPI and MPICH2. the flag "—download-mpich=yes" will download and compile the MPICH2 library.
- HDF5 is used for reading and writing HDF5 files. BLAS and LAPACK are numerical libraries to solve systems of equations.
- To use different versions of the petsc library at the same time, you can configure & compile them each in their separate folder. The folder is specified (optionally) by The PETSC_ARCH variable.

PFLOTRAN – Configuring PETSc source code

- If the configure finishes successfully (this will take quite some time), a message similar to this will be written to the screen:

```
xxx==========================================================================xxx
Configure stage complete. Now build PETSc libraries with (gnumake build):
make PETSC_DIR=/home/manuel/dev/petsc PETSC_ARCH=arch-linux2-c-opt all
xxx==========================================================================xxx
=================================================================================
Finishing Configure Run at Wed Apr 23 14:33:32 2014
=================================================================================
```

- Note that PETSC_DIR refers to the directory where you cloned petsc, and PETSC_ARCH refers to the current configuration dir that you specified (if not, PETSc will generate a default name).
- All the information about the configure process can be found in configure.log (which actually links to a configure.log in your PETSC_ARCH folder).

PFLOTRAN – Building the PETSc library

- The next step is compiling the PETSc source code and linking it into a library. As mentioned at the end of the configure process, you will need to run the following command (with your own paths and variables of course):

```
make PETSC_DIR=/home/manuel/dev/petsc PETSC_ARCH=0c8c14c5 all
```

- The actual command is “make all”. If you specified PETSC_DIR and PETSC_ARCH in your environment, that’s all you need. If not, make understands the variables in front of “all”. (setting env variables in the bash console: “export VARNAME=VALUE”)
- If everything goes ok, you will see a message like this:

```
=========================================
gmake[1]: Leaving directory `/home/manuel/dev/petsc'
Now to check if the libraries are working do:
make PETSC_DIR=/home/manuel/dev/petsc PETSC_ARCH=arch-linux2-c-opt test
=========================================
Finishing at: Wed 23 Apr 14:36:15 CEST 2014
```

- Again, all the output of the make command can be found in make.log (which actually links to a make.log in your PETSC_ARCH folder).

PFLOTRAN – Testing the PETSc library

- The next step is to make sure that the PETSC library works as expected. As mentioned at the end of the make process, you will need to run the following command (with your own paths and variables of course):

```
make PETSC_DIR=/home/manuel/dev/petsc PETSC_ARCH=0c8c14c5 test
```

- The actual command is “make test”. If you specified PETSC_DIR and PETSC_ARCH in your environment, that’s all you need. If not, make understands the variables in front of “all”.
- If everything goes ok, you will see a message that says that the test example have completed.

- If the examples do not run:
  - confirm that “**hostname**” appears as an alias for localhost in as “**cat /etc/hosts**” (if not update /etc/hosts). You can you the Yast tool Hostnames to do this as well.
  - Make sure that there were no errors in configure.log or make.log
- That’s it, you now have a working PETSc library. Remember that any program that wants to use it, requires the PETSC_DIR and PETSC_ARCH variables defined correctly.

PFLOTRAN – Getting the PFLOTRAN source code

- PFLOTRAN is also under continuous development.
- The source code is also stored in a code repository, but unlike PETSC, PFLOTRAN it is not using git, but mercurial.
- The source code is stored on: https://bitbucket.org/pflotran/pflotran-dev
- To be able to retrieve the code (see https://bitbucket.org/pflotran/pflotran-dev/wiki/Installation/Linux step 4) :

```
hg clone https://bitbucket.org/pflotran/pflotran-dev
```

- You only have to do the "clone" command once. After that, you use "**hg pull –u**", to update to the latest version.

- If you want an easier interface for mercurial, you can install tortoiseHg (http://tortoisehg.bitbucket.org/).

PFLOTRAN - Supercomputers

- PFLOTRAN is designed to run on supercomputers
- Within Amphos 21, PFLOTRAN was succesfully used on
  - local PCs running OpenSuse Linux
  - MareNostrum (Barcelona Supercomputing Center)
  - Juqeen (Jülich Supercomputing Centre)

- To run on a supercomputer, you normally connect to a pc that provides access to their network (the access node). The connection provides a remote shell login with ssh security.
  - On Linux, you can connect directly from the terminal with the ssh command.
  - On Windows, you can connect using Putty (http://www.chiark.greenend.org.uk/~sgtatham/putty/)
- Some basic software is installed using the module system. These module set the environment variables to point to specific versions of software (compilers, cmake, libaries, etc.)
  - module avail: shows modules that are available
  - module list: shows modules loaded
  - module load: loads a module for that session

PFLOTRAN - Supercomputers

- Besides the standard software, you can install your own software.
- On MareNostrum, the admins installed everything on request. There was no direct internet connection.
- On Julich there is a direct internet connection, and you can install your own software.

- On supercomputers you do not run things directly from the command line. Instead you write a file/job that describes which command you want to run, with what parameters, on how many nodes and with how many cores. Then you send this job to the "queue" and it will run when it reaches its turn.
- Different supercomputers use different job schedulers (or queue systems).
  - BSC: LSF (http://www.bsc.es/support/LSF/9.1.2/)
  - Julich: LoadLeveler (https://docs.loni.org/wiki/Useful_LoadLeveler_Commands)
- On julich:
  - **llsubmit jobfile** Submits a new job
  - **llq –s "JOBIDProvidedDuringSubmit"** checks the status of a job

Libraries necessary For PETSC on Juqueen

- LAPACK
- HDF5 (You can choose to use a local library already compiled or ask PETSC to download and build HDF5 with –download-hdf5=1 option)

PFLOTRAN – PETSc compilation

- Because the access node can be a different type of computer from the cluster nodes, you cannot simply do a configure. (this would get all the settings for the local node only). You will need to run it with special flags.
- The following configure parameters might be useful:
  - Setting the compilers (c,c++ and fortran):

    ```
    --with-cc=mpixlc_r
    --with-cxx=mpixlcxx_r
    --with-fc=mpixlf2003_r
    ```

    In this case IBM XL compilers are used that are already compiled against MPI, so you will also need the flag “**--known-mpi-shared-libraries=0**”
  - Setting compiler optimization flags:

    ```
    COPTFLAGS="-O2 -qarch=qp -qtune=qp"
    CXXOPTFLAGS="-O2 -qarch=qp -qtune=qp"
    FOPTFLAGS="-O2 -qarch=qp -qtune=qp“
    ```
  - Special BLAS and/or lapack lib (instead of downloading and compiling):

    ```
    --with-blas-lapack-lib="-L/bgsys/local/lapack/3.4.2/lib -llapack -L/bgsys/local/lib -lesslbg“
    ```
  - Let PETSc know that you are not running on the cluster:

    ```
    --with-batch
    ```

    The configure script will now generate an executable to be run in a job on the server. The output of this executable is a configuration file that can be run with python:
    Note that when you run the new python config, you need to comment the **'--with-batch‘** command with a “**#**”.

Compilation Problems

- IBM XL Fortran Compiler giving syntax errors when compiling ; had to ask developers to fix the code so the compiler would not stop on those syntax compile time errors.

- Compilation Warning that did not disappear
- Although PETSC was built with :
- --with-shared-libraries=0 \
- --known-mpi-shared-libraries=0 \
- ***warning: Using 'dlopen' in statically linked applications requires at runtime the shared libraries from the glibc version used for linking***
- This warning did not prevent pflotran from running

Run Time Problems:

- 1. Deallocation Problem
- 2. Restart Problem

Deallocation Problem:

- "pmc_subsurface.F90", line 1: 1525-109 Error encountered while attempting to deallocate a data object. The program will stop.

- 2015-01-08 14:52:27.357 (WARN) [0x40001298f30] :1890767:ibm.runjob.client.Job: normal termination with status 1 from rank 0

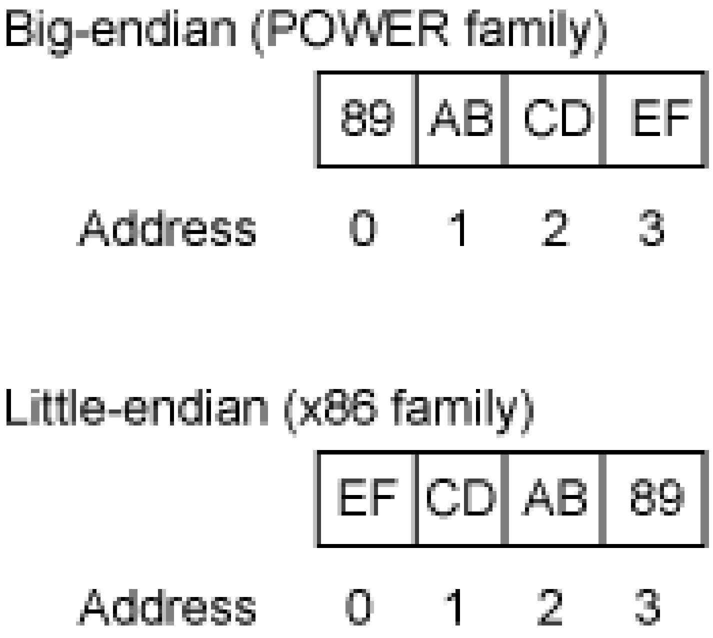
RESTART Problem

- Case1: No output
- Case2: Output with Format HDF5 (Single File)
- Case3: Output with Format HDF5 Multiple_Files.

- Solution?

Postprocess Problems:

- Endianness :The terms **endian** and **endianness** refer to the convention used to interpret the bytes making up a data word when those bytes are stored in computer memory in which memory stores binary data by organizing it into 8-bit (1 byte) .
- Endianness → the interpretation of the data word.



Big-endian (POWER family)

	89	AB	CD	EF
Address	0	1	2	3

Little-endian (x86 family)

	EF	CD	AB	89
Address	0	1	2	3

ParaView PFLOTRAN Reader :

-
  ```
  if (H5Tequal(intype, H5T_NATIVE_FLOAT) ||
          H5Tequal(intype, H5T_NATIVE_DOUBLE) ||
          H5Tequal(intype, H5T_NATIVE_LDOUBLE) ||
          H5Tequal(intype, H5T_IEEE_F32BE) ||
          H5Tequal(intype, H5T_IEEE_F64BE) )
  ```

- Adding the red lines now the ParaView PFLOTRAN Reader will also accept Big-Indian types!

- Thanks to Herwig

Builds Trials:

- Using GCC Compilers on juqueen.
- Using local hdf5 library
- Trying a different lapack libraries on juqueen than default. (default=lapack/3.4.2) (trial=lapack/3.4.2_simd)
- Built with debugging=on /off
- Built –with-64-bit-indices=0 and –q64
- Built –with-64-bit-indices=1 and –q64
- Built using PETSC 3.5.2
- Build using O3 strict optimization

Scalability tests:

- Doing tests using standard PFLOTRAN Problem (regional_doublet.in) and (cu_leaching.in)

- Comparing runtimes with previous data available on running on several number of processes(1,2,8,16,32…)

- Runtimes for 32 processes ~4.11 times slower than previous tests on article ??? And previous experience on Cesga.

- Scalability tests are done all with writing no outputs.

- **During Workshop PFLOTRAN with O4 optimization was built. The simulation output needs to be compared with O2 for consistency. –qstrict might be needed.**

PFLOTRAN – Troubleshooting

- If you have problems compiling petsc:
  - first make sure that you are using the correct commit version
  - Try a newer (or the latest) commit version to see if the problem still occurs
  - Try to narrow down the problem, by reading the configure.log & make.log
  - Search in the mailing list if the problem has not already been reported.
  - If the above does not work, you can write to petsc-maint@mcs.anl.gov for help.
    Be sure to include the configure.log and make.log as attachments and
    to write the commit version that you are using. Follow these instructions:
    http://www.mcs.anl.gov/petsc/documentation/bugreporting.html

- If you have problems compiling PFLOTRAN:
  - Make sure you are using the latest version. If you are using an older version, verify that the problem still exists in the latest version
  - Verify in the output where the first error occurs.
    When you run **make pflotran** again, it will continue from the last successfully compiled file.
  - Search in the mailing list if the problem has not already been reported.
  - If you cannot figure out what the problem is, write an email to
    pflotran-dev@googlegroups.com
    Again, make sure to include the commit version of petsc, pflotran and to mention on what system you are trying to compile PFLOTRAN.

PFLOTRAN – Troubleshooting

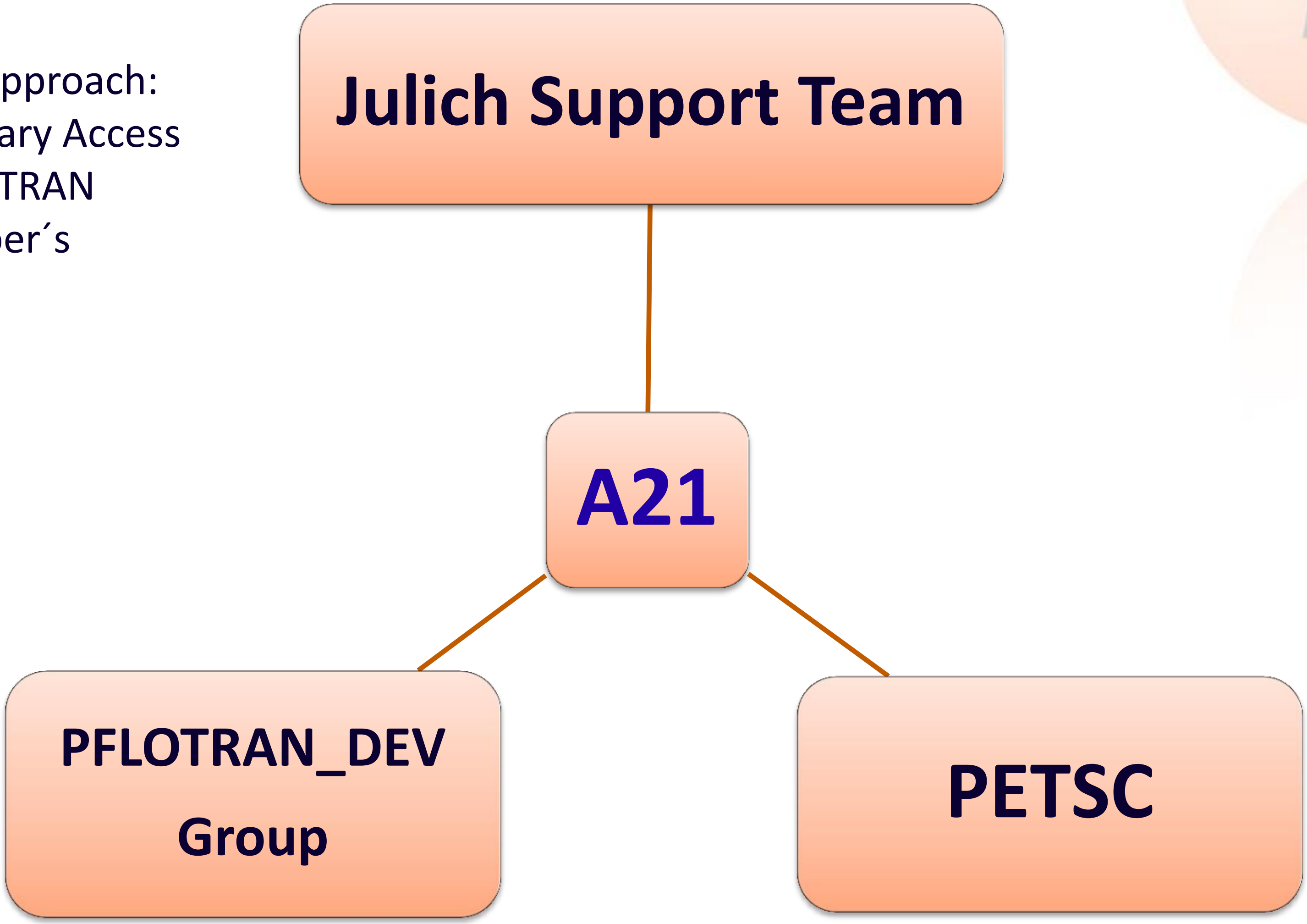
- If you have problems running PFLOTRAN :
  - make sure that the PFLOTRAN regression test runs fine.
  - Simplify your problem and see if it runs.
  - If it does not run, double-check the syntax of your input (by simplifying the input even further)
  - If it does not run because of convergence problems, try different discetizations (different grid, different time steps)
  - Check if there is no similar example in the shortcourse/examples folder.
  - Search in the mailing list if the problem has not already been reported.
  - Search in the mailing list if someone has asked questions about a similar problem
  - If you are still stuck, send an email to the following mailing list, with a complete description of your problem and which version of petsc and pflotran you are using: pflotran-users@googlegroups.com
  - See for more information: https://bitbucket.org/pflotran/pflotran-dev/wiki/Documentation/FAQ#rst-header-getting-help

Hope to solve issues related to:

- Runtime Errors
- Optimization
- Scorpio

Troubleshooting ?

Faster approach: Temporary Access to PFLOTRAN developer´s



Julich Support Team

A21

PFLOTRAN_DEV Group

PETSC

With Special Thanks to :

- Julich Support team especially
  - Inge Gutheil
  - Daniel Rohe
  - Herwig Zilken
  - Florian
- Guido Deissmann
- PFLOTRAN Developers´ team especially (Glenn Hammond)

AMPHOS 21

SCIENTIFIC AND STRATEGIC ENVIRONMENTAL CONSULTING

ESPAÑA
AMPHOS 21 CONSULTING S.L.
Paseo de García Faria, 49-51
08019 BARCELONA
Tel.: +34 93 583 05 00
Fax : +34 93 307 59 28

CHILE
AMPHOS 21 CONSULTING CHILE Ltda.
Av. Nueva Tajamar 481 of. 1005 (Torre Sur)
Las Condes 7550099
SANTIAGO DE CHILE
Tel.: +562 27991630

PERÚ
AMPHOS 21 CONSULTING PERU S.A.C.
Av. del Parque Sur 661, San Borja
Lima 41
Tel.: +511 5921275

FRANCE
AMPHOS 21 CONSULTING FRANCE S.A.R.L.
14 Avenue de l’Opéra
75001 PARIS
Tel.: +33 1 69345030

1994-2014
20 YEARS
FLOWING

www.amphos21.com