pymatsolver Documentation

Release 0.1.3

Rowan Cockett

May 17, 2021

Contents

A (sparse) matrix solver for python.

Solving Ax = b should be as easy as:

```
Ainv = Solver(A)
x = Ainv * b
```

In pymatsolver we provide a number of wrappers to existing numerical packages. Nothing fancy here.

CHAPTER 1

Solvers Available

All solvers work with `scipy.sparse` matricies, and a single or multiple right hand sides using `numpy`:

- L/U Triangular Solves
- Wrapping of SciPy matrix solvers (direct and indirect)
- Pardiso solvers now that MKL comes with conda!
- Mumps solver with nice error messages

CHAPTER 2

Installing Mumps

We have not been able to get the pip install to work because of multiple dependencies on fortran libraries. However, the linux and mac installs are relatively easy. Note that you must have mumps pre-installed, currently we have only got this working for the sequential version, so when you are installing, you will need to point to that one. You can also look at the *.travis.yml* file for how to get it working on TravisCI.

2.1 Linux

From a clean install on Ubuntu:

```
apt-get update
apt-get -y install gcc gfortran git libopenmpi-dev libmumps-seq-dev libblas-dev␣
↪liblapack-dev

# Install all the python you need!
wget http://repo.continuum.io/miniconda/Miniconda-3.8.3-Linux-x86_64.sh -O miniconda.
↪sh;
chmod +x miniconda.sh
./miniconda.sh -b
export PATH=/root/anaconda/bin:/root/miniconda/bin:$PATH
conda update --yes conda
conda install --yes numpy scipy matplotlib cython ipython nose

git clone https://github.com/rowanc1/pymatsolver.git
cd pymatsolver
make mumps
```

2.2 Mac

This assumes that you have Brew and some python installed (numpy, scipy):

```
brew install mumps --with-scotch5 --without-mpi

git clone https://github.com/rowanc1/pymatsolver.git
cd pymatsolver
make mumps_mac
```

If you have problems you may have to go into the Makefile and update the pointers to Lib and Include for the various libraries.

This command is helpful for finding dependencies. You should also take note of have happens when brew installs mumps.

```
mpicc --showme
```

Code: https://github.com/simpeg/pymatsolver

Tests: https://travis-ci.org/simpeg/pymatsolver

Bugs & Issues: https://github.com/simpeg/pymatsolver/issues

License: MIT

2.2.1 The API

class pymatsolver.solvers.**Base**(*A*)

Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

T

accuracy_tol

accuracy_tol (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06

check_accuracy

check_accuracy (Boolean): check the accuracy of the solve?, a boolean, Default: False

clean()

is_hermitian

is_positive_definite

is_real

is_symmetric

set_kwargs(*ignore=None*, ***kwargs*)

Sets key word arguments (kwargs) that are present in the object, throw a warning if they don't exist.

Basic Solvers

class pymatsolver.wrappers.**Solver**(*A*, ***kwargs*)

Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

clean()

class pymatsolver.wrappers.**SolverLU**(*A*, ***kwargs*)
Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

clean()

class pymatsolver.wrappers.**SolverCG**(*A*, ***kwargs*)
Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

Diagonal

class pymatsolver.solvers.**Diagonal**(*A*)
Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

Triangular

class pymatsolver.solvers.**Forward**(*A*)
Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

class pymatsolver.solvers.**Backward**(*A*)
Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

Iterative Solvers

class pymatsolver.iterative.**BicgJacobi**(*A*, *symmetric=True*)
Bicg Solver with Jacobi preconditioner

Required Properties:

- **accuracy_tol** (Float): tolerance on the accuracy of the solver, a float, Default: 1e-06
- **check_accuracy** (Boolean): check the accuracy of the solve?, a boolean, Default: False

clean()

factor()

maxiter = 1000

solver = None

```
tol = 1e-06
```

Direct Solvers

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Index