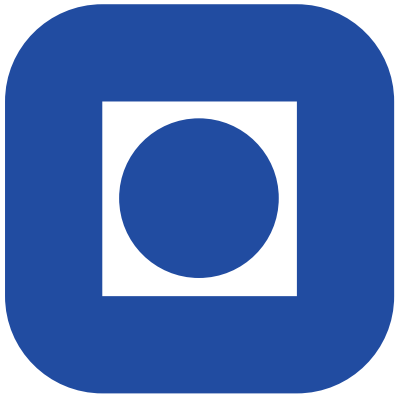
OPM and PETSc

Jørgen Kvalsvik



jorgekva@stud.ntnu.no

March 11, 2015

- Portable, Extensible Toolkit for Scientific Computing
- Mature, free (libre), C library of data structures and solvers
- Used in many industry applications

Extensive solver support.

type	algorithm
preconditioner	(point block) jacobi
	SOR
	additive schwarz
	ILU(k)
	ICC(k)
	relaxation & schur-complement
direct	LU
	LU
	cholesky
krylov	richardson
	chebyshev
	cg
	gmres

Unfortunately, terrible from a user perspective.

```
matrix::builder::inserter builder( 2, 2 );
builder.insert( 0, 0, 2 );
builder.insert( 1, 1, 2 );

// [ 2, 0 ] [ x1 ] = [ 2 ]
// [ 0, 2 ] [ x2 ] = [ 1 ]

matrix A = commit( builder );

vector b = { 0, 1 };
vector result = { 1, 0.5 };

auto x = solve( A, b );
auto y = solve( A, b, A );
auto z = solve( A, b, A,
        solver::pc_type( "sor" ),
        solver::ksp_type( "cg" ) );
```

- Memory automatically managed
- Functionally oriented, designed to be quick and easy to use
- Comparable performance
- Debug & development tools.

```
/* rhs_ and soln_ are std::vector */
petsc::solver::linear_tolerance ltol;
ltol.relative_tolerance = residual_tolerance;
ltol.absolute_tolerance = 1e-05;
ltol.maximum_iterations = linsolver_maxit;

petsc::vector b( this->rhs_ );

auto x = petsc::solve( A, b, ltol );

petsc::vector::scalar* raw_arr;
VecGetArray( x, &raw_arr );
soln_.resize( x.size() );
soln_.insert( soln_.begin(),
              raw_arr,
              raw_arr + x.size() );
```

```
/* rhs_ and soln_ are dune vectors */
Adapter opS(S_);

Dune::SeqILU0<Matrix,Vector,Vector>
precond(S_, 1.0);

Dune::CGSolver<Vector>
linsolve(opS, precond, residual_tolerance,
         maxit, verbosity_level);

Dune::InverseOperatorResult result;

linsolve.apply(soln_, rhs_, result);
if (!result.converged) {
    OPM_THROW(std::runtime_error,
    "failed to converge" );
}
```

Questions?