ngsPETSc

S. Zampini[†], P. E. Farrell*, U. Zerbinati*

* *Mathematical Institute*
University of Oxford

† *Extreme Computing Research Center*
King Abdullah University of Science and Technology

4th NGSolve Useer Meeting, Portland, 10th of July 2023

UNIVERSITY OF OXFORD
Mathematical Institute

Oxford Mathematics

NGSolve-PETSc

January 2023 — First NetGen to DMPlex interface for Firedrake.

NGSolve-PETSc

UNIVERSITY OF OXFORD
Mathematical Institute

- January 2023 — First NetGen to DMPlex interface for Firedrake.
- March 2023 — PETSc DMPlex Transform wrapped in petsc4py.

NGSolve-PETSc

UNIVERSITY OF OXFORD
Mathematical Institute

- January 2023 — First NetGen to DMPlex interface for Firedrake.
- March 2023 — PETSc DMPlex Transform wrapped in petsc4py.
- April 2023 — DMPlex to NetGen interface for NGSolve.

NGSolve-PETSc

January 2023	First NetGen to DMPlex interface for Firedrake.
March 2023	PETSc DMPlex Transform wrapped in petsc4py.
April 2023	DMPlex to NetGen interface for NGSolve.
May 2023	NGSolve-DMPlex used in Firedrake master.

NGSolve-PETSc

- January 2023 — First NetGen to DMPlex interface for Firedrake.
- March 2023 — PETSc DMPlex Transform wrapped in petsc4py.
- April 2023 — DMPlex to NetGen interface for NGSolve.
- May 2023 — NGSolve-DMPlex used in Firedrake master.
- June 2023 — NGSolve-DMPlex can be used in FEniCSx

NGSolve-PETSc

UNIVERSITY OF OXFORD
Mathematical Institute

- January 2023 — First NetGen to DMPlex interface for Firedrake.
- March 2023 — PETSc DMPlex Transform wrapped in petsc4py.
- April 2023 — DMPlex to NetGen interface for NGSolve.
- May 2023 — NGSolve-DMPlex used in Firedrake master.
- June 2023 — NGSolve-DMPlex can be used in FEniCSx
- July 2023 — Can we have more PETSc features in NGSolve ?

NGSolve-PETSc

University of Oxford
Mathematical Institute

January 2023	First NetGen to DMPlex interface for Firedrake.
March 2023	PETSc DMPlex Transform wrapped in petsc4py.
April 2023	DMPlex to NetGen interface for NGSolve.
May 2023	NGSolve-DMPlex used in Firedrake master.
June 2023	NGSolve-DMPlex can be used in FEniCSx
July 2023	Can we have more PETSc features in NGSolve ?

Why PETSc?

UNIVERSITY OF OXFORD
Mathematical Institute

PETSc is a suite of data structures and routines developed by Argonne National Laboratory for the scalable (parallel) solution of scientific applications modeled by partial differential equations.

- PETSc implementation have **scalability** in mind, for this reason most linear algebra operations have **efficient parallel** implementation.
- Wide variety of **Krylov linear solvers (KSP)**.
- Wide variety of **preconditioners (PC)**.
- PETSc easily **wraps** many external libraries, such as: **HYPRE**, **MUMPS**, **H2OPUS**, **Intel MKL**, **...**

Setting up ngsPETSc

UNIVERSITY OF OXFORD
Mathematical Institute

MPI Support

To interface NGSolve with PETSc, you need to have NGSolve compiled with a MPI support, you achieve this using the **USE_MPI** flag, when building from source.

pip install will <u>not</u> work !

PETSc/SLEPc

You need to have PETSc and SLEPc configured on your system with slepc4py and petsc4py installed as well.

https://ngspetsc.readthedocs.io/en/latest/ngsPETSc.html

ngsPETSc - Basic Components

UNIVERSITY OF OXFORD
Mathematical Institute

ngsPETSc wraps the following PETSc objects in such a way that can be used easily in NGSolve:

- **PETSc Vec** and **PETSc Mat**, those are the basic linear algebra object corresponding to vectors and matrices.
- **PETSc KSP**, this is the object representing a **large sparse linear solver**.
- **PETSc DMPlex**, this is the object representing meshes in PETSc.
- **SLEPc EPS**, this is the object representing an eigenvalue problem solver in SLEPc.

ngsPETSc - PETSc Vec

Mathematical Institute

ngsPETSc is capable of creating a PETSc **Vec** from a NGSolve **BaseVector** and **ParallelVector** using the **VectorMapping** object.

```
from ngsPETSc import VectorMapping
Map = VectorMapping(fes)
petscVec = Map.petscVec(ngsVec)
```

ngsPETSc - PETSc Mat

UNIVERSITY OF OXFORD
Mathematical Institute

ngsPETSc is capable of creating a PETSc **Mat** from a NGSolve **BaseMatrix** object, thanks to the **Matrix** class.

```
a = BilinearForm(grad(u)*grad(v)*dx).Assemble()
M = a.mat
from ngsPETSc import Matrix
M = Matrix(m.mat, fes.FreeDofs(), matType="aij")
M.view()
```

- We can easily host PETSc **Mat** on devices, in particular PETSc support using **Kokkos** GPU manufactured by **NVIDIA**, **INTEL** and **AMD**.

ngsPETSc - PETSc KSP

ngsPETSc is capable of creating a PETSc **KSP** from a NGSolve **BilinearForm** object, thanks to the **KrylovSolver** class.

```
f = LinearForm(fes)
f += 32 * (y*(1-y)+x*(1-x)) * v * dx
from ngsPETSc import KrylovSolver
sol = KrylovSolver(a, fes, solverParameters={
           'ksp_type': 'cg',
           'pc_type': 'lu',
           'pc_factor_mat_solver_type': 'mumps'
          })
gfu = sol.solve(f)
```

ngsPETSc - PETSc KSP

University of Oxford Mathematical Institute

Algorithm	Associated Type	External Packages	Parallel	Complex
Richardson	KSPRICHARDSON	—	X	X
Chebyshev	KSPCHEBYSHEV	—	X	X
GMRES	KSPGMRES	—	X	X
Flexible GMRES	KSPFGMRES	—	X	X
LGMRES	KSPLGMRES	—	X	X
Deflated GMRES	KSPDGMRES	—	X	
Two-stage with least squares residual minimization	KSPTSIRM	—	X	X
Conjugate Gradient	KSPCG	—	X	X
Conjugate Gradient Squared	KSPCGS	—	X	X
Conjugate Gradient for Least Squares	KSPCGLS	—	X	X
Conjugate Gradient on Normal Equations	KSPCGNE	—	X	X

ngsPETSc - PETSc KSP

Nash Conjugate Gradient with trust region constraint	`KSPNASH`	—	X	X
Conjugate Gradient with trust region constraint	`KSPSTCG`	—	X	X
Gould et al Conjugate Gradient with trust region constraint	`KSPGLTR`	—	X	X
Steinhaug Conjugate Gradient with trust region constraint	`KSPQCG`	—	X	X
Left Conjugate Direction	`KSPLCD`	—	X	X
Bi-Conjugate Gradient	`KSPBICG`	—	X	X
Stabilized Bi-Conjugate Gradient	`KSPBCGS`	—	X	X
Improved Stabilized Bi-Conjugate Gradient	`KSPIBCGS`	—	X	X
Transpose-free QMR	`KSPTFQMR`	—	X	X

ngsPETSc - PETSc KSP

Mathematical Institute

Tony Chan QMR	KSPTCQMR	—	X	X
QMR BiCGStab	KSPQMRCGS	—	X	X
Flexible Conjugate Gradients	KSPFCG	—	X	X
Flexible stabilized Bi-Conjugate Gradients	KSPFBCGS	—	X	X
Flexible stabilized Bi-Conjugate Gradients with fewer reductions	KSPFBCGSR	—	X	X
Stabilized Bi-Conjugate Gradients with length ℓ recurrence	KSPBCGSL	—	X	X
Conjugate Residual	KSPCR	—	X	X
Generalized Conjugate Residual	KSPGCR	—	X	X
Generalized Conjugate Residual (with inner normalization and deflated restarts)	KSPHPDDM	HPDDM	X	X
Minimum Residual	KSPMINRES	—	X	X

ngsPETSc - PETSc KSP

University of Oxford
Mathematical Institute

Minimum Residual	`KSPMINRES`	—	X	X
LSQR	`KSPLSQR`	—	X	X
SYMMLQ	`KSPSYMMLQ`	—	X	X
FETI-DP (reduction to dual-primal sub-problem)	`KSPFETIDP`	—	X	X
Gropp's overlapped reduction Conjugate Gradient	`KSPGROPPCG`	—	X	X

ngsPETSc - PETSc PC

It is now possible to use any PETSc **PC** in NGSolve using the **“PETScPC”** preconditioner in NGSolve.

```
pre = Preconditioner(a, "PETScPC", pc_type="hypre")
gfu = GridFunction(fes)
gfu.vec.data = CG(a.mat, rhs=f.vec, pre=pre, printrates=mesh.comm.rank==0)
```

ngsPETSc - PETSc PC

	Algorithm	Associated Type	Matrix Types	External Packages
Generic	Jacobi	PCJACOBI	MATAIJ, MATBAIJ, MATSBAIJ, MATDENSE	—
	Point Block Jacobi	PCPBJACOBI	MATAIJ, MATBAIJ, MATSBAIJ, MATKAIJ, MATMPISELL, MATIS	—
	Variable Point Block Jacobi	PCPBJACOBI	MATAIJ, MATBAIJ, MATSBAIJ	—
	Block Jacobi	PCBJACOBI	MATAIJ, MATBAIJ, MATSBAIJ	—
	SOR	PCSOR	MATAIJ, MATSEQDENSE, MATSEQSBAIJ	—

ngsPETSc - PETSc PC

Mathematical Institute

	Point Block SOR		MATSEQBAIJ (only for bs = 2,3,4,5)	—
	Kaczmarz	PCKACZMARZ	MATAIJ	—
	Additive Schwarz	PCASM	MATAIJ, MATBAIJ, MATSBAIJ	—
	Vanka/overlapping patches	PCPATCH	MATAIJ	—
	Deflation	PCDEFLATION	All	—
Incomplete	ILU	PCILU	MATSEQAIJ, MATSEQBAIJ	—
	ILU with drop tolerance	PCILU	MATSEQAIJ	SuperLU Sequential ILU solver
		PCILU	MATAIJ	Euclid/hypre (PCHYPRE)

ngsPETSc - PETSc PC

Mathematical Institute

	ICholesky	PCICC	MATSEQAIJ, MATSEQBAIJ, MATSEQSBAIJ	—
	Algebraic recursive multilevel	PCPARMS	MATSEQAIJ	pARMS
Matrix Free	Infrastructure	PCSHELL	All	—
Multigrid	Infrastructure	PCMG	All	—
	Geometric		All	—
	Smoothed Aggregation	PCGAMG	MATAIJ	—
	Smoothed Aggregation (ML)	PCML	MATAIJ	ML/Trilinos
	Structured Geometric	PCPFMG, PCSYSPFMG, PCSMG	MATHYPRESTRUCT	hypre

ngsPETSc - PETSc PC

Mathematical Institute

	Classical Algebraic	PCHYPRE, PCAMGX	MATAIJ	BoomerAMG/hypre, AmgX
	Multi-group MG	PCHMG	MATAIJ	—
	Domain Decomposition	PCHPDDM	MATAIJ, MATBAIJ, MATSBAIJ, MATIS	HPDDM
Hierarchical matrices	$\mathcal{H}^2$	PCH2OPUS	MATHTOOL, MATH2OPUS	H2OPUS
Physics-based Splitting	Relaxation & Schur Complement	PCFIELDSPLIT	MATAIJ, MATBAIJ, MATNEST	—
	Galerkin composition	PCGALERKIN	Any	—
	Additive/multiplicative	PCCOMPOSITE	Any	—
	Least Squares Commutator	PCLSC	MATSCHURCOMPLEMENT	—

ngsPETSc - PETSc PC

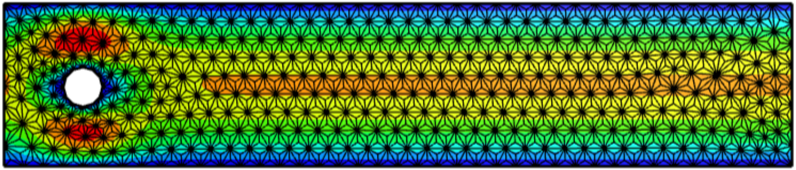
Parallel transformation	Redistribution	PCREDISTRIBUTE	MATAIJ	—
	Telescoping communicator	PCTELESCOPE	MATAIJ	—
	Distribute for MPI	PCMPI	MATAIJ	—
Approximate Inverse	AIV	PCHYPRE, PCSPAI	MATAIJ	Parasails/hypre, SPAI
Substructuring	Balancing Neumann-Neumann	PCNN	MATIS	—
	Balancing Domain Decomposition	PCBDDC	MATIS	—
	2-level Schwarz wire basket	PCEXOTIC	MATAIJ	—

ngsPETSc - PETSc DMPlex

ngsPETSc is capable of creating a mapping PETSc **DMPlex** and NGSolve **Mesh** object, thanks to the **MeshMapping** class. This will allow to apply **DMPlexTransform** to NGSolve Meshes.

```
Map = MeshMapping(mesh)
tr = PETSc.DMPlexTransform().create(comm=PETSc.COMM_WORLD)
tr.setType(PETSc.DMPlexTransformType.REFINEALFELD)
tr.setDM(Map.petscPlex)
tr.setUp()
newplex = tr.apply(Map.petscPlex)
mesh = Mesh(MeshMapping(newplex).ngMesh)
```

ngsPETSc - PETSc DMPlex



ngsPETSc - PETSc EPS

Mathematical Institute

ngsPETSc is capable of creating easily a SLEPc **EPS** one can use in NGSolve to solve eigenvalue problems. In particular, we are interested in solving an eigenvalue pencil

$$-\lambda(\vec{u}_h, \vec{v}_h) + (\nabla \vec{u}_h, \nabla \vec{v}_h) = 0$$

```
    solver = EigenSolver((m, a), fes, 4, solverParameters={"eps_type":SLEPc.EPS.Type.ARNOLDI,
                   "eps_smallest_magnitude":None,
                   "eps_tol": 1e-6,
                   "eps_target": 2,
                   "st_type": "sinvert",
                   "st_pc_type": "lu"})
```

ngsPETSc - Future Features

University of Oxford Mathematical Institute

- NGSolve **BaseMatrix** with block structures should be mapped to a PETSc **NestMat**. This will allow for **field split** preconditioning.
- Use PETSc **SNES** to solve non-linear system in NGSolve, this will allow for **line search** and **trust region**.
- Use PETSc **TS** to solve time dependent partial differential equations.
- Use SLEPc **PEP** to solve polynomial eigenvalue problems.

ngsPETSc - Other Solvers (It's a secret ...)

Having access to a PETSc **DMPlex** allows using NetGen meshes also in other finite element libraries,

- **Firedrake**, thanks to the ngsPETSc interface is now possible to use NetGen mesh and do **adaptive mesh refinement**.

```
ngmsh = unit_square.GenerateMesh(maxh=0.1)
msh = firedrake.Mesh(ngmsh)
File("output/MeshExample1.pvd").write(msh)
```

- **FEniCSx**, thanks to the ngsPETSc interface is now possible to use NetGen mesh.