PETSc TAO support for optimisation problems with FEniCSx

Paul T. Kühner, Michal Habera, Andreas Zilian
Department of Engineering | University of Luxembourg

18 June 2025 | FEniCS'25 conference

UNIVERSITÉ DU LUXEMBOURG

Optimisation Problems

$$\min_x f(x)$$

Optimisation Problems

Let $f : \mathbb{R}^n \to \mathbb{R}$

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$

- Unconstrained

Optimisation Problems

Let $f : \mathbb{R}^n \to \mathbb{R}$, $x^-, x^+ \in \overline{\mathbb{R}}^n$

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & x^- \leq x \leq x^+ \end{aligned}$$

- Unconstrained
- Bound constrained

Optimisation Problems

Let $f : \mathbb{R}^n \to \mathbb{R}$, $x^-, x^+ \in \overline{\mathbb{R}}^n$, $g : \mathbb{R}^n \to \mathbb{R}^m$ and $h : \mathbb{R}^n \to \mathbb{R}^l$.

$$
\begin{aligned}
\min_{x \in \mathbb{R}^n} \quad & f(x) \\
\text{subject to} \quad & x^- \leq x \leq x^+ \\
& g(x) = 0 \\
& h(x) \geq 0
\end{aligned}
$$

- Unconstrained
- Bound constrained
- General constrained

Optimisation Problems

Let $f : \mathbb{R}^n \to \mathbb{R}$, $x^-, x^+ \in \overline{\mathbb{R}}^n$, $g : \mathbb{R}^n \to \mathbb{R}^m$ and $h : \mathbb{R}^n \to \mathbb{R}^l$.

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & x^- \leq x \leq x^+ \\ & g(x) = 0 \\ & h(x) \geq 0 \end{aligned}$$

- Unconstrained
- Bound constrained
- General constrained

$\to$ PETSc's **T**oolkit for **A**dvanced **O**ptimization (TAO) [1]

Optimisation Problems over Function Spaces

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & x^- \le x \le x^+ \\ & g(x) = 0 \\ & h(x) \ge 0 \end{aligned}$$

Optimisation Problems over Function Spaces

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & x^- \le x \le x^+ \\ & g(x) = 0 \\ & h(x) \ge 0 \end{aligned}$$

$$\begin{aligned} \min_{u \in V} \quad & f(u) \\ \text{subject to} \quad & u^- \le u \le u^+ \\ & g(u) = 0 \\ & h(u) \ge 0 \end{aligned}$$

Motivation

- Non abstraction breaking
- Automatic differentiation, form manipulation, ... – `ufl`
- Uniform interface for un-, bound- or general constrained problems
- Multi field
- Optional abstractions

Previous Work

- Bound constrained, single field, no automatic differentiation
  - `FreeFem++` [2]
  - `comet-fenicsx` [3]
- SNES solver, blocked systems, `ufl`
  - `dolfiny` [4]
  - `DOLFINx` [5]
- Variational inequality solvers with SNES VI
  - `MFEM` [6]
  - `Firedrake` [7]
- Convex optimisation with `MOSEK`, multi field, `ufl`
  - `dolfinx_optim` [8]

```python
from dolfiny.taoproblem import TAOProblem
```

A Poisson Solver

We consider

$$-\Delta u = f \quad \text{in } \Omega = (0,1)^2$$
$$u = g \quad \text{on } \partial\Omega$$

as energy minimisation problem

$$\min_{\substack{u \in H^1(\Omega) \\ u|_{\partial\Omega}=g}} \frac{1}{2}\int_\Omega \nabla u^2 \, \mathrm{d}x - \int_\Omega f u \, \mathrm{d}x.$$

```python
from dolfinx import fem, mesh
from dolfiny.taoproblem import TAOProblem

domain = mesh.create_unit_square(
  MPI.COMM_WORLD, 32, 32
)
V = fem.functionspace(domain, ("P", 1))

u = fem.Function(V)
F = (
  .5 * ufl.inner(ufl.grad(u), ufl.grad(u)) * ufl.dx
  - f * u * ufl.dx
)
bc = fem.dirichletbc(g, facet_dofs, V)

opts = PETSc.Options("poisson")
opts["tao_type"] = "bqnls"
opts["tao_ls_type"] = "armijo"
opts["tao_max_it"] = 300
opts["tao_gttol"] = 1e-5

tao = TAOProblem(F, [u], bcs=[bc], prefix="poisson")
tao.solve()
```

But, SNES can line search

- Both TAO and SNES search for a stationary point

$$\delta f(u^\star) = R(u^\star) = 0.$$

- For a descent direction p, line search computes step size

$$\alpha = \operatorname*{argmin}_{\hat{\alpha}>0} f(u + \hat{\alpha} p).$$

- SNES will use (by default)

$$f(u) = ||R(u)||_2^2.$$

  - Dangerous – not respecting discretization.

Poisson with Bound Constraints

We consider

$$-\Delta u = f \quad \text{in } \Omega = (0,1)^2$$
$$u = g \quad \text{on } \partial\Omega$$

as energy minimisation problem

$$\min_{u \in H^1(\Omega)} \frac{1}{2} \int_\Omega \nabla u^2 \,\mathrm{d}x - \int_\Omega f u \,\mathrm{d}x$$

subject to $\quad g - \iota_{\partial\Omega} \leq u \leq g + \iota_{\partial\Omega},$

where $\iota_{\partial\Omega} = \begin{cases} 0 & x \in \partial\Omega \\ \infty & \text{else.} \end{cases}$

```
u = fem.Function(V)
F = (
  .5 * ufl.inner(ufl.grad(u), ufl.grad(u)) * ufl.dx
  - f * u * ufl.dx
)

lb = fem.Function(V)
lb.interpolate(
  lambda x: np.where(
    np.isclose(x[0], 0) or np.isclose(x[0], 1) or
    np.isclose(x[1], 0) or np.isclose(x[1], 1),
    g,
    np.full(x.shape[1], -np.inf),
  )
)

ub = fem.Function(V)
ub.interpolate(...)

opts = PETSc.Options()
opts["tao_type"] = "bnls"
opts["tao_ls_type"] = "unit"
opts["tao_gatol"] = 1e-14

tao = TAOProblem(F, [u], ub=[ub], lb=[lb])
tao.solve()
```

Optimal Control

We consider

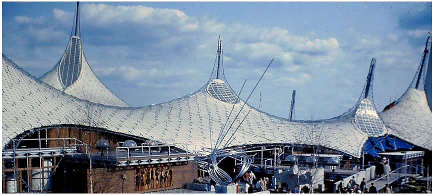
$$\min_{f \in H^1(\Omega)} ||u_d - u(f)||_{L^2}^2 + \frac{\alpha}{2} ||f||_{L^2}^2$$

where $u(f)$ is the solution of

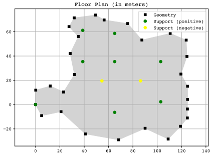
$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega = (0,1)^2 \\ u &= g \quad \text{on } \partial\Omega. \end{aligned}$$

```python
from dolfiny.taoproblem import sync_functions

state_problem = fem.petsc.LinearProblem(...)
adjoint_problem = fem.petsc.LinearProblem(...)

@sync_functions(f)
def F_hat(tao, x):
    state_problem.solve()
    return comm.allreduce(fem.assemble_scalar(F))

@sync_functions(f)
def J_hat(tao, x, J):
    state_problem.solve()

    adjoint_problem.solve()

    # jacobian = p + alpha * f
    jacobian.x.array[:] = (
        p.x.array[:] + alpha * f.x.array[:]
    )
    fem.petsc.assign(jacobian, J)

tao = TAOProblem(
    F_hat, [f], J=(J_hat, f.x.petsc_vec.copy())
)
tao.solve()
```

German Pavilion Expo ’67



German Pavilion at Expo ’67 in Montreal [9].

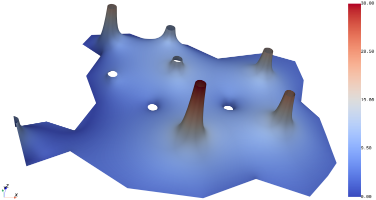


Floor plan of the Pavilion geometry [11].

$$\min_{\substack{g\in H^1(\Omega)\\ g|_{\partial\Omega}=h}} \int_\Omega \sqrt{|\nabla g|^2+1}\,\mathrm{d}x + \alpha||g||^2_{H^1}$$

subject to $\quad \phi \le u \le \psi.$

- For *smooth* h, ϕ and ψ this is well-posed [10].



Computed surface $\{(x, y, g(x, y)) \mid (x, y) \in \Omega\}$.

Truss Sizing Optimisation

$\Omega \subset \mathbb{R}^3$ truss structure [12]

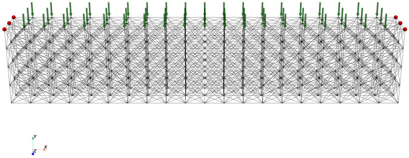
$$\min_{s \in DG_0(\Omega)} \int_{v \in \Gamma_f} u(s) \cdot f \, \mathrm{d}P$$

subject to $s_{\text{min}} \leq s \leq s_{\text{max}}$

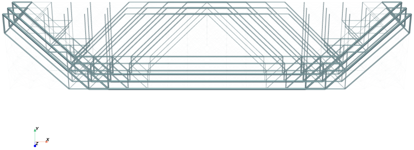
$$\int_\Omega s \, \mathrm{d}x = \frac{1}{20} \int_\Omega s_{\text{max}} \, \mathrm{d}x,$$

where $u(s) \in P_1(\Omega)$ solves

$$\int_\Omega s\sigma(u)\varepsilon(v) \, \mathrm{d}x = \int_{v \in \Gamma_f} f \cdot v \, \mathrm{d}x \quad \forall v.$$



Truss geometry $100 \times 20 \times 10\, m$.



Optimised truss structure, radii scaled by factor 5.

Experience with PETSc TAO

- Python TAO interface sparsely used
  - `GitHub` search for `PETSc.TAO()` yields 70 results
  - Routines for general constrained problems previously not exported
- Unconstrained and bound constrained problems – *just work*
- General constrained problems – *not battle tested*
  - `ALMM` is the only usable general constrained solver – Bugs identified
  - `PDIPM` does not support ghosted vectors – `IPOPT` [13] contender?

Outlook

- NEST matrices
- Multiple constraint support
- Further applications in shape- and topology optimisation
- Custom optimiser through PETSc Python interface
- Automatic reduced functional – pyadjoint-x? [14]

dolfiny.uni.lu

fenics-dolfiny/dolfiny

pip install dolfiny

References

[1] Satish Balay et al. 'Efficient Management of Parallelism in Object-Oriented Numerical Software Libraries'. In: *Modern Software Tools for Scientific Computing*. Ed. by Erlend Arge, Are Magnus Bruaset and Hans Petter Langtangen. Boston, MA: Birkhäuser Boston, 1997, pp. 163–202. ISBN: 978-1-4612-1986-6. DOI: 10.1007/978-1-4612-1986-6_8. URL: https://doi.org/10.1007/978-1-4612-1986-6_8.

[2] Frédéric Hecht. 'New development in FreeFem++'. In: *J. Numer. Math.* 20.3-4 (2012), pp. 251–265. ISSN: 1570-2820. URL: https://freefem.org/.

[3] Jeremy Bleyer. *Numerical tours of Computational Mechanics with FEniCSx*. Version v0.2. Sept. 2024. DOI: 10.5281/zenodo.13838486. URL: https://doi.org/10.5281/zenodo.13838486.

[4] Andreas ZILIAN and Michal HABERA. 'dolfiny: Convenience wrappers for DOLFINx'. English. In: https://dx.doi.org/10.6084/m9.figshare.14495262. Cambridge, United Kingdom, 23 March 2021. URL: https://fenics2021.com/talks/zilian.html.

[5] Igor A. Baratta et al. *DOLFINx: The next generation FEniCS problem solving environment*. Dec. 2023. DOI: 10.5281/zenodo.10447666. URL: https://doi.org/10.5281/zenodo.10447666.

[6] R. Anderson et al. 'MFEM: A Modular Finite Element Methods Library'. In: *Computers & Mathematics with Applications* 81 (2021), pp. 42–74. DOI: 10.1016/j.camwa.2020.06.009.

[7] David A. Ham et al. *Firedrake User Manual*. First edition. Imperial College London et al. May 2023. DOI: 10.25561/104839.

[8] Jeremy Bleyer. 'Applications of conic programming in non-smooth mechanics'. In: *Journal of Optimization Theory and Applications* (2022), pp. 1–33.

[9] René Lavigne. *Pavillon de l'Allemagne (2).jpg*. [Online; accessed June 14, 2025]. 1967. URL: https://commons.wikimedia.org/wiki/File:Pavillon_de_l%27Allemagne_(2).jpg.

[10] L. A. Caffarelli. 'The obstacle problem revisited'. In: *J. Fourier Anal. Appl.* 4.4-5 (1998), pp. 383–402. ISSN: 1069-5869,1531-5851. DOI: 10.1007/BF02498216. URL: https://doi.org/10.1007/BF02498216.

[11] Alba López Gómez, Diego Paniagua Padilla and Luis Ruiz Andrés. 'Pabellón alemán: EXPO'67, Montreal (Canadá): Frei Otto, Rolf Gutbrod: arquitectura textil, análisis del edificio'. Working Paper. 2015. URL: https://oa.upm.es/39614/.

[12] Peter W. Christensen and Anders Klarbring. 'Sizing Stiffness Optimization of a Truss'. In: *An Introduction to Structural Optimization*. Dordrecht: Springer Netherlands, 2009, pp. 77–95. ISBN: 978-1-4020-8666-3. DOI: 10.1007/978-1-4020-8666-3_5.

[13] Andreas Wächter and Lorenz T. Biegler. 'On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming'. In: *Math. Program.* 106.1 (2006), pp. 25–57. ISSN: 0025-5610,1436-4646. DOI: 10.1007/s10107-004-0559-y. URL: https://doi.org/10.1007/s10107-004-0559-y.

[14] Sebastian K. Mitusch, Simon W. Funke and Jørgen S. Dokken. 'dolfin-adjoint 2018.1: automated adjoints for FEniCS and Firedrake'. In: *Journal of Open Source Software* 4.38 (2019), p. 1292. DOI: 10.21105/joss.01292. URL: https://doi.org/10.21105/joss.01292.