

Sequential quadratic and convex optimisation with FEniCSx and PETSc

Paul T. Kühner, Michal Habera, Andreas Zilian

Department of Engineering | University of Luxembourg

7 October 2025 | KLAIM 2025



Contents

Motivation

Quadratic and convex approximations

Interfacing and extending PETSc

Examples

Generally (PDE-)constrained optimisation problems

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

$$\begin{array}{ll} \min_{y, u \in Y \times U} & J(y, u) \\ \text{subject to} & F(y, u) = 0 \\ & y^- \leq y \leq y^+ \\ & g(y) = 0 \\ & h(y) \leq 0 \end{array}$$

Motivation – our interest

PDE-constrained optimisation for additively manufactured light-weight parts.

► Scales and scalar products

- A FEM function is not a coefficient vector
- Scales/units of controls/states

► No black-box oracle

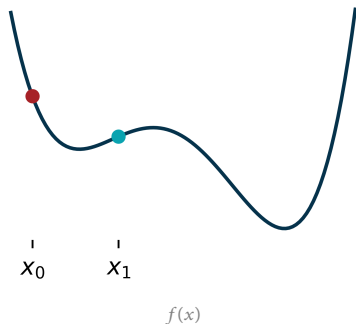
- Efficient optimisation algorithms employ a-priori problem knowledge
- Formulation is a moving target

⇒ Problem specific optimisation solvers.

Motivation – general interest

- ▶ Libraries, like IPOPT [1], CLARABEL [2], NLOPT [3], ...are powerful tools, but hard to extend
 - Integration of custom solvers/preconditioners
 - Parallelism not generally available
 - If problem breaks the provided abstractions, no re-usability
 - ▶ Well established algorithms (SQP/MMA/CONLIN/...) lack modern and performant implementations in a common interface.
- ⇒ Optimisation solver integration into modern HPC software tool chain.

Sequential optimisation



Formulate an approximate sub-problem to P around $x_k \in \mathbb{R}^n$.

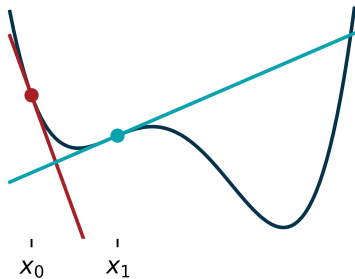
$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \hat{f}(x) \\ \text{subject to} \quad & x^- \leq x \leq x^+ \\ & \hat{g}(x) = 0 \\ & \hat{h}(x) \leq 0 \end{aligned} \quad (P_k)$$

Pick $\hat{f}, \hat{g}, \hat{h}$ such that [4]

- ▶ P_k can be efficiently solved
- ▶ Precise approximation (convergence)
- ▶ Sufficiently conservative (feasibility)

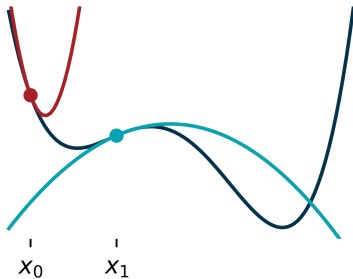
Linear and quadratic approximations

Sequential Linear Programming



$$\hat{f}(x) = f(x_0) + \nabla f(x_0)^T (x - x_0)$$

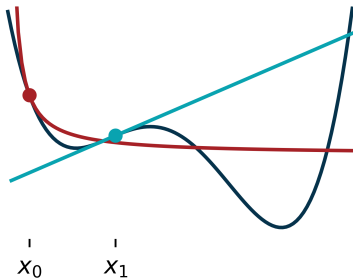
Sequential Quadratic Programming [5]



$$\hat{f}(x) = f(x_0) + \nabla f(x_0)^T (x - x_0) + \frac{1}{2} (x - x_0)^T H(x_0) (x - x_0)$$

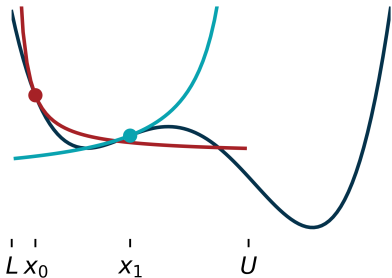
CONLIN and MMA

CONvex LINearisation [6]



$$\hat{f}(x) = f(x_0) + p^T(x - x_0) + q^T x_0^2 \odot \left(\frac{1}{x} - \frac{1}{x_0} \right)$$
$$p = \nabla f^+(x_0)$$
$$q = \nabla f^-(x_0)$$

Method of Moving Asymptotes [7]



$$\hat{f}(x) = r + p^T(U - x)^{-1} + q^T(x_0 - L)^{-1}$$
$$r = f(x_0) - p^T(U - x_0)^{-1} - q^T(x_0 - L)^{-1}$$
$$p = \nabla f^+(x_0)$$
$$q = \nabla f^-(x_0)$$

Efficient sub-problem optimisation

- SQP – Solving the KKT system (linear solve)

$$\begin{bmatrix} H_f(x_0) & \nabla g(x_0)^T \\ \nabla g(x_0) & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} H_f(x_0)x_0 - \nabla f(x_0) \\ \nabla g(x_0)x_0 - g(x_0) \end{bmatrix}$$

- CONLIN/MMA – Dual method (bound constrained optimisation)

$$\min_x \max_{\lambda \geq 0} L(x, \lambda) \stackrel{\text{strong duality}}{=} \max_{\lambda \geq 0} \underbrace{\min_x L(x, \lambda)}_{W(\lambda)} = \hat{f}(x) + \lambda^T \hat{h}(x)$$

Sub-problem P_k is equivalent to

$$\max_{\lambda \geq 0} W(\lambda).$$

- The separability of the CONLIN/MMA approximations, yield closed form expressions for $W(\lambda)$.

Interface – MMA implementation

```
class MMA:
    def create(self, tao):
        subsolver = PETSc.TAO().create()

    def setUp(self, tao):
        def W(tao,  $\lambda$ , G) -> float:
            # compute  $x(\lambda)$ 
            # evaluate  $W(\lambda)$ , grad  $W(\lambda)$ .

        subsolver.setSolution( $\lambda$ )
        subsolver.setObjectiveGradient(W, J_ $\lambda$ )

    def solve(self, tao):
        for k in range(1, tao.getMaximumIterations()):
            # evaluate  $f(x)$ , grad  $f(x)$ 
            f = tao.computeObjectiveGradient(x, gradient)

            if h:
                # evaluate  $h(x)$ , grad  $h(x)$ 
                h(tao, x, c, *h_args, **h_kwargs)
                Jh(tao, x, J, None, *Jh_args, **Jh_kwargs)

            # update move limits, bounds, intermediates

            # solve subproblem with dual method
            subsolver.solve()

            tao.setIterationNumber(k)
            tao.monitor(f=..., res=...)
            tao.checkConverged()
```

- Builds on top of petsc4py [8]
- Vectors, matrices, solvers, etc. are PETSc [9] objects
- Integrates no different than a native TAO optimiser
 - tao_type python
 - tao_python_type MMA

Example: topology optimisation

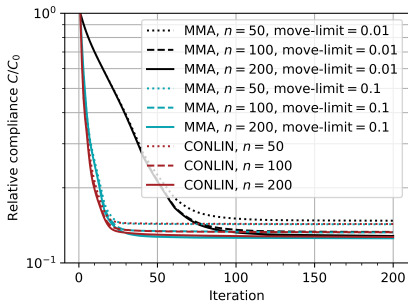
Let $d \in \{2, 3\}$, $\Omega = (0, 1)^{d-1} \times (0, 2)$, $\Gamma \subset \partial\Omega$, $f \in \mathbb{R}^d$, $\rho_{\min} = 10^{-9}$, $V_f \in (0, 1)$

$$\begin{aligned} \min_{\rho, \hat{\rho}, u \in \mathcal{DP}_0 \times \mathcal{P}_1 \times \mathcal{P}_1^d} \quad & \int_{\Gamma} f \cdot u \, dS \\ \text{subject to} \quad & \rho_{\min} \leq \rho \leq 1 \\ & \int_{\Omega} \sigma(u, \hat{\rho}) : \epsilon(v) \, dx = \int_{\Gamma} f \cdot v \, dS \quad \forall v \in \mathcal{P}_1^d \\ & \int_{\Omega} r^2 \nabla \hat{\rho} \cdot \nabla \tau + \hat{\rho} \tau \, dx = \int_{\Omega} \rho \tau \, dx \quad \forall \tau \in \mathcal{P}_1 \\ & \frac{1}{\text{Vol}(\Omega)} \int_{\Omega} \rho \, dx \leq V_f \end{aligned}$$

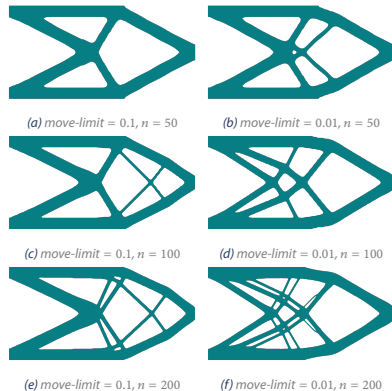
σ the Cauchy stress tensor of linear isotropic elastic material, with SIMP penalised [10] Young's modulus $E(\hat{\rho}) = (\rho_{\min} + (1 - \rho_{\min})\hat{\rho}^p)E_0$ (here $p = 3$).

- ▶ FEM analysis in FEniCSx [11]
- ▶ Custom interface to TAO [12]

Example: 2D cantilever

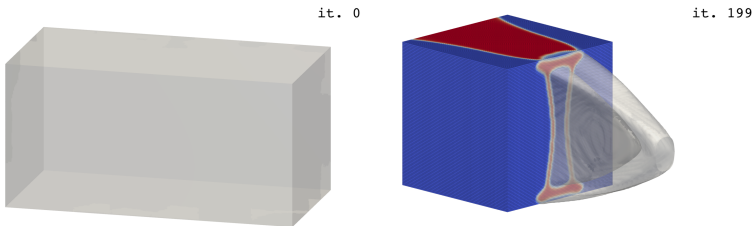


Plot of compliance $\int_{\Gamma} f \cdot u \, dS$ relative to first iterate.



Converged MMA results with varying move-limit and discretisations.

Example: large scale



0.5 and 0.1 density contours for iterations 0-200 (left) and cross section of final design (right).

- ▶ $\{2, 6\} \times 10^6$ DOFs in {density, displacement} field
- ▶ single node, 96 cores¹, approx. 12 hours runtime

¹AMD EPYC 7742

Example: options

<code>-tao_type</code>	<code>python</code>
<code>-tao_python_type</code>	<code>dolfiny.mma.MMA</code>
<code>-tao_mma_move_limit</code>	<code>0.01</code>
<code>-tao_mma_subsolver_tao_type</code>	<code>bqnls</code>
<code>-tao_mma_subsolver_tao_ls_type</code>	<code>more-thuente</code>
<code>-elasticity_ksp_type</code>	<code>cg</code>
<code>-elasticity_pc_type</code>	<code>gamg</code>
<code>-elasticity_mg_levels_ksp_type</code>	<code>chebyshev</code>
<code>-elasticity_mg_levels_pc_type</code>	<code>sor</code>
<code>-filter_ksp_type</code>	<code>cg</code>
<code>-filter_pc_type</code>	<code>jacobi</code>

 `dolfiny.uni.lu`

 `fenics-dolfiny/dolfiny`

 `pip install dolfiny`

References

- [1] Andreas Wächter and Lorenz T. Biegler. 'On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming'. In: *Mathematical Programming* 106.1 (Apr. 2005), pp. 25–57. ISSN: 1436-4646. DOI: 10.1007/s10107-004-0559-y.
- [2] Paul J. Goulart and Yuwen Chen. *Clarabel: An interior-point solver for conic programs with quadratic objectives*. 2024. arXiv: 2405.12762.
- [3] Steven G. Johnson. *The NLOpt nonlinear-optimization package*. <https://github.com/stevengj/nlopt>. 2007.
- [4] Pierre Duysinx, Michaël Bruyneel and Claude Fleury. 'Solution of large scale optimization problems with sequential convex programming'. In: (Jan. 2009).
- [5] Joaquim R. R. A. Martins and Andrew Ning. *Engineering Design Optimization*. Cambridge, UK: Cambridge University Press, Jan. 2022. ISBN: 9781108833417. DOI: 10.1017/9781108980647.
- [6] C. Fleury. 'CONLIN: An efficient dual optimizer based on convex approximation concepts'. In: *Structural Optimization* 1.2 (June 1989), pp. 81–89. ISSN: 1615-1488. DOI: 10.1007/bf01637664.
- [7] Krister Svanberg. 'The method of moving asymptotes—a new method for structural optimization'. In: *Internat. J. Numer. Methods Engrg.* 24.2 (1987), pp. 359–373. ISSN: 0029-5981, 1097-0207. DOI: 10.1002/nme.1620240207.
- [8] Lisandro D. Dalcin et al. 'Parallel distributed computing using Python'. In: *Advances in Water Resources* 34.9 (2011). New Computational Methods and Software Tools, pp. 1124–1139. ISSN: 0309-1708. DOI: 10.1016/j.advwatres.2011.04.013.
- [9] 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.
- [10] Martin Philip Bendsøe and Noboru Kikuchi. 'Generating optimal topologies in structural design using a homogenization method'. In: *Computer Methods in Applied Mechanics and Engineering* 71.2 (1988), pp. 197–224. ISSN: 0045-7825. DOI: 10.1016/0045-7825(88)90086-2.
- [11] Igor A. Baratta et al. *DOLFINx: The next generation FEniCS problem solving environment*. Dec. 2023. DOI: 10.5281/zenodo.10447666.
- [12] Paul Theodor Kühner, Michal Habera and Andreas Zilian. 'PETSc TAO support for optimisation problems with FEniCSx'. In: Groningen, Netherlands, June 2025. URL: <https://hdl.handle.net/10993/65289>.