

## **PISCO : Plateforme d'optimisation topologique par lignes de niveaux**

F. Bordeu<sup>1</sup>, C. Nardoni<sup>2</sup>, D. Danan<sup>2</sup>, C. Mang<sup>2</sup>, J. Cortial<sup>1</sup>

<sup>1</sup> Safran Tech, Plateforme Sciences & techniques du numérique, France, {felipe.bordeu, julien.cortial}@safrangroup.com

<sup>2</sup> IRT SystemX, France, {chiara.nardoni, david.danan, chetra.mang}@irt-systemx.fr

---

**Résumé** — PISCO est un environnement de prototypage rapide pour l'optimisation topologique par la méthode des lignes de niveaux. Il est caractérisé par une architecture logicielle modulaire qui vise à découpler autant que possible le moteur d'optimisation topologique et le solveur physique. Il repose essentiellement sur deux bibliothèques *open source* qui fournissent des briques numériques suffisamment génériques pour mettre en œuvre une grande variété d'applications métiers.

**Mots clefs** — Éléments finis ; Optimisation topologique ; Lignes de niveaux ; Open source

---

### **1. Contexte**

L'optimisation topologique est une discipline d'optimisation de structures qui consiste à déterminer, au sein d'un espace de conception donné, la distribution de matériau la plus performante possible telle que la solution obtenue respecte un certain nombre de contraintes fonctionnelles.

Ses caractéristiques intrinsèques la rendent très attractive pour les utilisateurs industriels. Premièrement, l'optimisation topologique permet en effet une liberté de conception plus large que l'optimisation paramétrique voire l'optimisation de forme classique et facilite l'obtention des concepts non contraints de manière artificielle. Cet aspect offre également une bonne synergie avec les procédés de fabrication additive [1], qui ne sont généralement pas aussi restrictifs que les procédés traditionnels. Deuxièmement, la mise en œuvre de l'optimisation topologique ne nécessite pas de disposer d'une géométrie préliminaire complète et détaillée d'une pièce à concevoir, mais seulement d'un cahier des charges.

L'optimisation topologique est couramment utilisée dans l'industrie pour la conception de pièces mécaniques et de nombreuses solutions logicielles disponibles dans le commerce sont disponibles à cet effet. Cependant, certains verrous techniques subsistent, comme par exemple la prise en compte des spécificités des procédés de fabrication additive, les problèmes multi-physiques (par exemple, mécanique, thermique et fluide) et la conception de structures à plusieurs échelles.

La plateforme PISCO vise à répondre aux objectifs suivants :

- Améliorer la compréhension des aspects numériques de l'optimisation topologique par la méthode des lignes de niveaux ;
- Accompagner la montée en maturité de technologies prometteuses, en particulier l'utilisation d'un maillage du domaine de calcul conforme à l'interface [2] ;

- Rendre possible l'utilisation de logiciels externes pour la résolution des problèmes physiques pour profiter de leurs capacités de calcul (méthodes numériques avancées, calcul haute performance, etc.) ;
- Permettre le prototypage rapide de développements adaptés aux besoins industriels.

## 2. Architecture de la plateforme

Le logiciel d'optimisation topologique par lignes de niveaux se décompose de la manière suivante :

- Un moteur d'optimisation qui pilote les itérations successives ;
- Une machinerie lignes de niveaux pour la gestion et l'évolution du champ distance ;
- Un solveur physique (ici, un solveur mécanique éléments finis).

La Figure 1 schématise les différentes interactions entre ces trois sous-systèmes. Chaque composant est suffisamment indépendant pour être modifié ou remplacé séparément. Par exemple, une interface avec Code\_Aster<sup>1</sup> est proposée au sein de la plateforme et peut se substituer facilement au solveur éléments finis nativement intégré.

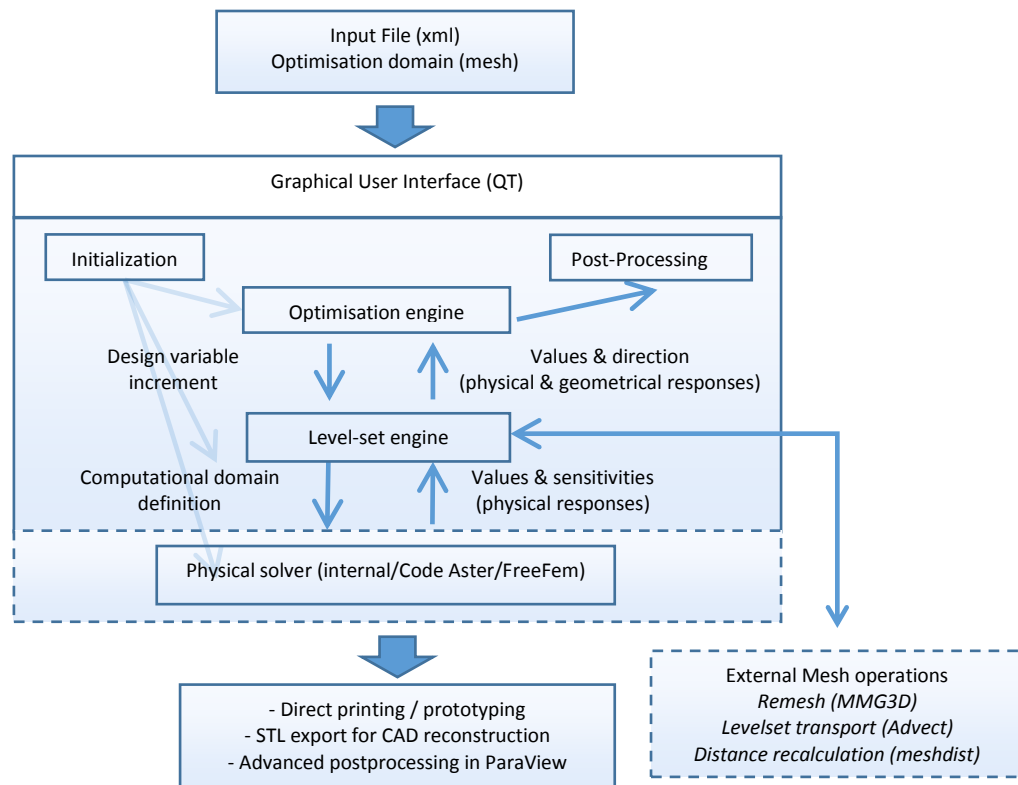
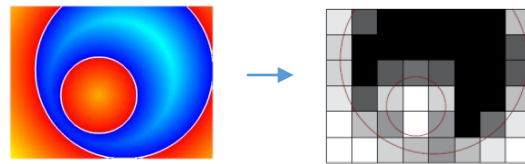


Figure 1 : Interactions entre les différents composants du démonstrateur

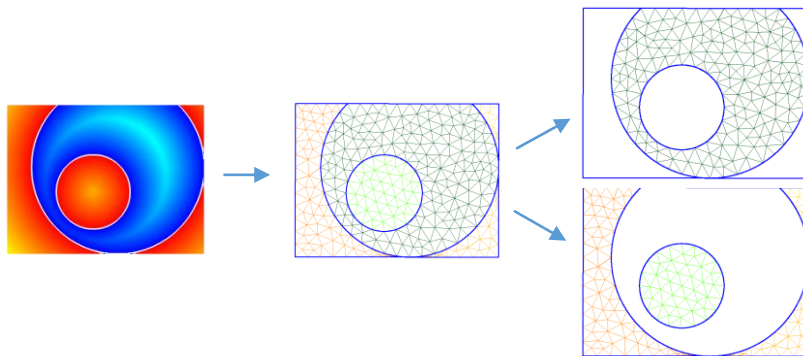
<sup>1</sup> <https://www.code-aster.org/>

Dans la mesure du possible, les différentes briques ont été découplées pour obtenir une bibliothèque de composants modulaires, capables de s'adapter aux besoins futurs. Par exemple, il n'est pas nécessaire d'avoir la même discrétisation pour le problème éléments finis et pour le support de la variable de conception, c'est-à-dire, dans le cadre d'une méthode de lignes de niveaux, le champ de distance signé. De même, l'utilisateur peut décider d'interfacier un solveur physique différent et ainsi effectuer une optimisation topologique sur des problèmes de natures physiques différentes (par exemple, un problème couplé fluide/structure).

Une infrastructure de modules d'extension (« plugins ») permet à l'utilisateur d'intégrer ses propres fonctionnalités (lecture de maillages, algorithmes d'optimisations, problèmes d'optimisation, solveur physique, etc.) sans devoir modifier le code source existant. En particulier, la flexibilité obtenue permet d'isoler les développements expérimentaux des fonctionnalités plus matures disponibles dans la plateforme. La méthode d'optimisation topologique par la méthode des lignes de niveaux repose sur une série d'opérations sur le champ de distance signée. La nature des algorithmes associés dépend du type de support de calcul, en particulier maillage structuré fixe (Figure 2) ou non-structuré (Figure 3). De plus, dans le cas de maillages non-structurés épousant l'interface, une opération de remaillage doit être appliquée.



**Figure 2 : Lignes de niveau et interprétation sous la forme d'un champ de densité dans le cadre d'un maillage fixe.**



**Figure 3 : Champ de distance signée, domaine remaillé conforme à l'interface, maillages intérieur et extérieur pour la simulation physique.**

### 3. Bibliothèques *open source*

Le langage Python a été utilisé du fait de sa simplicité d'accès, de la familiarité de nombreux développeurs avec ce langage, et de la disponibilité de bibliothèques comme NumPy<sup>2</sup> et SciPy<sup>3</sup> qui permettent d'obtenir un niveau de performance acceptable. Toutefois, plusieurs routines ont été programmées en langage C++ pour obtenir des performances acceptables pour certaines opérations très spécifiques, comme par exemple l'assemblage des matrices élémentaires. Nous avons utilisé Cython<sup>4</sup> pour l'interfaçage entre Python et C++.

Toutes les briques élémentaires de PISCO sont disponibles dans deux bibliothèques *open source* (BasicTools et TopoTools) disponibles sur <https://gitlab.com/drti/>.

BasicTools est une bibliothèque qui propose un ensemble d'outils de base pour la manipulation de maillages, de champs, et de données dans le contexte des éléments finis. Les principales fonctionnalités sont :

- Lecture et écriture de fichiers de maillages. BasicTools ne définit pas de format spécifique, qui serait incompatible avec les formats existants, mais cherche à s'adapter aux besoins des utilisateurs en favorisant la compatibilité et la flexibilité.
- Manipulation de maillages, notamment les opérations de création, d'inspection, de modification, et de filtrage qui sont utiles pour les analyses éléments finis.
- Manipulation de champs. Les champs éléments finis peuvent être définis avec différentes interpolations (P0, P1, P2, ...), sur le maillage complet ou seulement sur une zone restreinte. Des champs définis aux points d'intégrations sont aussi disponibles.
- Intégration numérique de formulations faibles pour l'assemblage de matrices, de vecteurs et de quantité scalaires.
- Transfert de champs entre maillages.
- Interfaçage d'outils externes : Des adaptateurs facilitent la conversion de maillages et des champs entre les structures de données BasicTools et des bibliothèques comme vtk, meshio, et pyvista.
- Moteur de géométrie implicite : Moteur pour la création et la manipulation (intersection, union, différence...) des objets géométriques définis de manière explicite.

---

<sup>2</sup> <https://numpy.org/>

<sup>3</sup> <https://scipy.org/>

<sup>4</sup> <http://cython.org>

TopoTools est la bibliothèque plus spécifiquement dédiée à la l'optimisation topologique, dont les principales fonctionnalités sont :

- Gestion des lignes de niveaux : création et manipulation de champs de lignes de niveaux sur maillages structurés et non structurés. Certaines fonctionnalités (redistantiation, advection) reposent sur un interfaçage avec des outils externes (ISCD toolbox<sup>5</sup>).
- Adaptation de maillage : une interface avancée avec le logiciel de remaillage Mmg3d<sup>6</sup> est disponible.
- Solveurs physiques : des solveurs physiques sont disponibles pour la construction de critère nécessitant une résolution d'un problème physique (comme par exemple la compliance).
- Critères et problème d'optimisation: des critères varies (géométriques, physiques) sont disponible afin de définir l'objectif et les contraintes du problème d'optimisation. Les critères peuvent être utilisés comme des contraintes d'égalités et d'inégalités, et ceci dans le contexte d'une optimisation par une méthode de gradient.
- Optimiseurs : un optimiseur spécialement conçu pour le traitement de problèmes d'optimisation topologique ainsi que des optimiseurs plus classiques (du type lagrangien Augmenté)
- Application interactive : visualisation d'un problème d'optimisation topologique afin de valider, modifier et surveiller un calcul en temps réel.

## 4. Application graphique interactive

Toutes les fonctionnalités présentées précédemment ont été intégrées dans une application graphique qui utilise un fichier d'entrée au format XML permettant d'initialiser l'ensemble des composants nécessaires au calcul. La Figure 4 montre la fenêtre principale de l'application. Cette dernière a été programmée entièrement en langage Python en utilisant les bibliothèques Qt pour la gestion de fenêtres et VTK pour la visualisation 3D.

---

<sup>5</sup> <https://github.com/ISCDtoolbox/>

<sup>6</sup> <https://www.mmgtools.org>

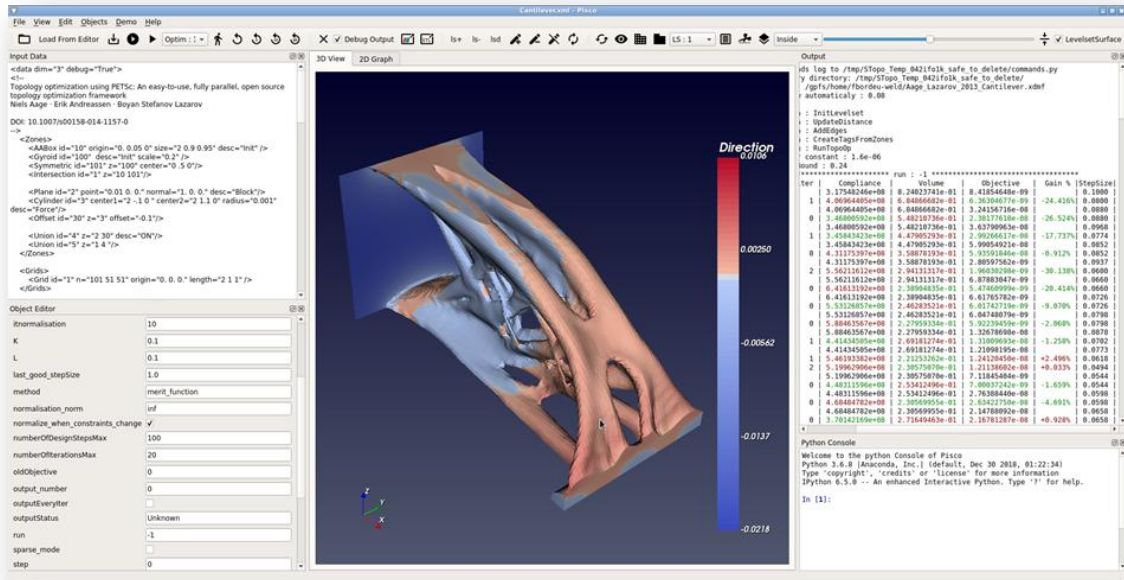


Figure 4 : Interface graphique utilisateur de PISCO

## Références

- [1] G. Allaire, M. Bihr, B. Bogosel, Support optimization in additive manufacturing for geometric and thermo-mechanical constraints, Structural and Multidisciplinary Optimization 61, Springer, 2377-2399, 2020.
- [2] G. Allaire, C. Dapogny, P. Frey, A mesh evolution algorithm based on the level set method for geometry and topology optimization, Structural and Multidisciplinary Optimization 48, Springer, 711-715, 2013.