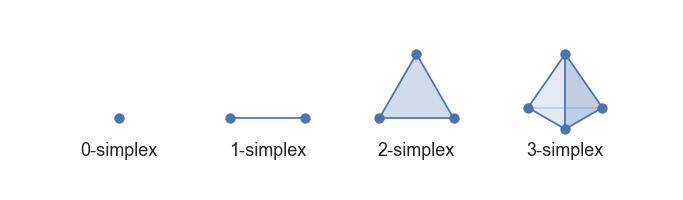
**Analyse de données topologiques et complexes simpliciaux**

Les complexes simpliciaux sont un moyen de construire des espaces topologiques à partir de composants combinatoires simples. Cela permet de réduire les complexités du traitement de la géométrie continue des espaces topologiques à la tâche de combinatoire et de comptage relativement simples. Cette méthode d’apprivoisement de la géométrie et de la topologie sera fondamentale pour notre approche de l’analyse des données topologiques en général, et de la réduction de dimension en particulier.

La première étape consiste à fournir des blocs de construction combinatoires simples [appelés \*simplices\*](https://en.wikipedia.org/wiki/Simplex). Géométriquement, un simplex est un moyen très simple de construire un objet de dimension k. Un simplex de dimension k est appelé k-simplex, et il est formé en prenant la coque convexe de k+1 points indépendants. Ainsi, un 0-simplex est un point, un 1-simplex est un segment de ligne (entre deux simplices nulles), un 2-simplex est un triangle (avec trois 1-simplices comme « faces »), et un 3-simplex est un tétraèdre (avec quatre 2-simplices comme « faces »). Une construction aussi simple permet une généralisation facile à des dimensions arbitraires.



Simplices de faible dimension[¶](https://umap-learn.readthedocs.io/en/latest/how_umap_works.html#id2)

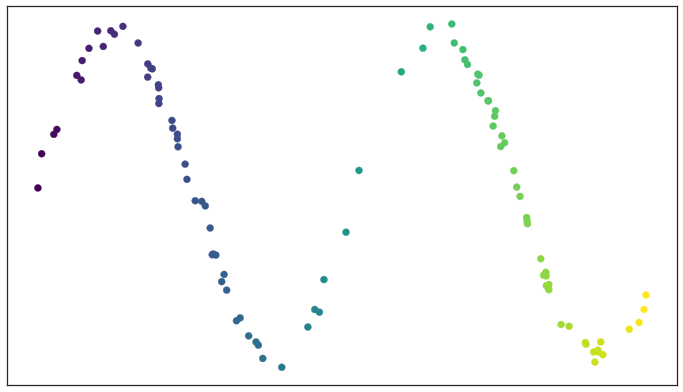
Cela a une structure combinatoire sous-jacente très simple, et finalement on peut considérer un k-simplex comme un ensemble arbitraire d’objets k+1 avec des faces (et des faces de faces, etc.) données par des sous-ensembles de taille appropriée – on peut toujours fournir une « réalisation géométrique » de cette description abstraite de cet ensemble en construisant le simplex géométrique correspondant.

Les simplices peuvent fournir des blocs de construction, mais pour construire des espaces topologiques intéressants, nous devons être en mesure de coller ensemble de tels blocs de construction. Cela peut être fait en construisant un [\*complexe simplicial\*](https://en.wikipedia.org/wiki/Simplicial_complex). Apparemment, un complexe simplicial est un ensemble de simplices collées ensemble le long des faces. Plus explicitement, un complexe simplicial \(\mathcal{K}\) est un ensemble de simplices tel que n’importe quelle face de n’importe quel simplex dans \(\mathcal{K}\) est également dans \(\mathcal{K}\) (en veillant à ce que toutes les faces existent), et l’intersection de deux simplices quelconques dans \(\mathcal{K}\) est une face des deux simplices. Une grande classe d’espaces topologiques peut être construite de cette manière – en collant simplement ensemble des simplices de différentes dimensions le long de leurs faces. Un peu plus d’abstraction arrivera à [des \*ensembles simpliciaux\*](https://en.wikipedia.org/wiki/Simplicial_set) qui sont purement combinatoires, ont une belle présentation théorique des catégories, et peuvent générer une classe beaucoup plus large d’espaces topologiques, mais cela nous mènera trop loin pour cet article. L’intuition des complexes simpliciaux suffira à illustrer les idées pertinentes et la motivation.

Comment appliquer ces outils théoriques de la topologie à des ensembles finis de points de données ? Pour commencer, nous allons examiner comment on pourrait construire un complexe simplicial à partir d’un espace topologique. L’outil que nous allons considérer est la construction d’un [complexe de Čech](https://en.wikipedia.org/wiki/%C4%8Cech_cohomology) étant donné une couverture ouverte d’un espace topologique. C’est beaucoup de verbiage si vous n’avez pas fait beaucoup de topologie, mais nous pouvons la décomposer assez facilement pour notre cas d’utilisation. Une couverture ouverte n’est essentiellement qu’une famille d’ensembles dont l’union est l’ensemble de l’espace, et un complexe de Čech est un moyen combinatoire de convertir cela en un complexe simplicial. Cela fonctionne assez simplement: que chaque ensemble de la couverture soit un 0-simplex; créer un 1-simplex entre deux de ces ensembles s’ils ont une intersection non vide ; créer un 2-simplex entre trois de ces ensembles si la triple intersection des trois n’est pas vide ; et ainsi de suite. Maintenant, cela ne semble pas très avancé – il suffit de regarder les intersections d’ensembles. La clé est que la théorie topologique de fond fournit en fait des garanties sur la façon dont ce processus simple peut produire quelque chose qui représente l’espace topologique lui-même d’une manière significative (le [théorème nerveux](https://en.wikipedia.org/wiki/Nerve_of_a_covering) est le résultat pertinent pour ceux qui s’y intéressent). De toute évidence, la qualité de la couverture est importante, et les couvertures plus fines offrent plus de précision, mais la réalité est que, malgré sa simplicité, le processus capture une grande partie de la topologie.

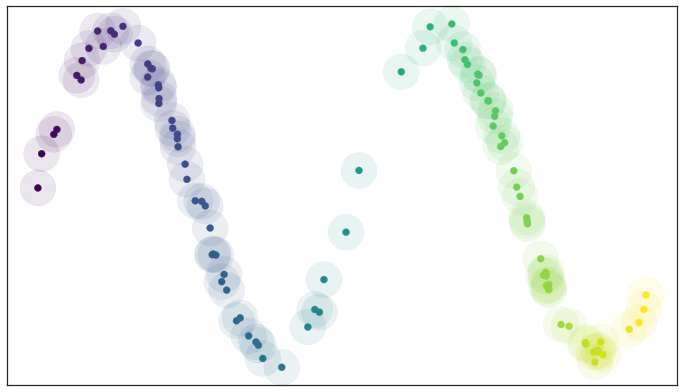
Ensuite, nous devons comprendre comment appliquer ce processus à un ensemble fini d’échantillons de données. Si nous supposons que les échantillons de données sont tirés d’un espace topologique sous-jacent, nous devons en générer une couverture ouverte pour en apprendre davantage sur la topologie de cet espace. Si nos données se trouvent réellement dans un espace métrique (c’est-à-dire que nous pouvons mesurer la distance entre les points), une façon d’approximer une couverture ouverte consiste simplement à créer des boules d’un rayon fixe autour de chaque point de données. Puisque nous n’avons que des échantillons finis, et non l’espace topologique lui-même, nous ne pouvons pas être sûrs qu’il s’agit vraiment d’une couverture ouverte, mais cela pourrait être une approximation aussi bonne que nous pourrions raisonnablement nous y attendre. Cette approche présente également l’avantage que le complexe de Čech associé à la couverture aura un 0-simplex pour chaque point de données.

Pour illustrer le processus, considérons un jeu de données de test comme celui-ci



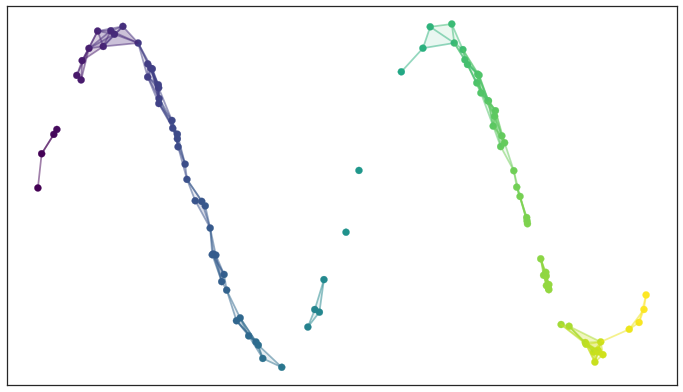
Ensemble de données de test d’une onde sinusoïdale bruyante[¶](https://umap-learn.readthedocs.io/en/latest/how_umap_works.html#id3)

Si nous fixons un rayon, nous pouvons alors imaginer les ensembles ouverts de notre couverture sous forme de cercles (puisque nous sommes dans un joli boîtier bidimensionnel visualisable). Le résultat est quelque chose comme ceci



Un couvercle ouvert de base des données d’essai[¶](https://umap-learn.readthedocs.io/en/latest/how_umap_works.html#id4)

Nous pouvons alors représenter le complexe simplicial des simplices 0, 1 et 2 sous forme de points, de lignes et de triangles.



Un complexe simplicial construit à partir des données de test[¶](https://umap-learn.readthedocs.io/en/latest/how_umap_works.html#id5)

Il est plus difficile de représenter facilement les simplices de dimension supérieure, mais vous pouvez imaginer comment ils s’intégreraient. Il y a deux choses à noter ici : premièrement, le complexe simplicial fait un travail raisonnable pour commencer à capturer la topologie fondamentale de l’ensemble de données; deuxièmement, la majeure partie du travail est vraiment effectuée par les simplices 0 et 1, qui sont plus faciles à traiter informatiquement (il ne s’agit que d’un graphique, au sens des nœuds et des arêtes). La deuxième observation motive le [complexe de Vietoris-Rips](https://en.wikipedia.org/wiki/Vietoris%E2%80%93Rips_complex), qui est similaire au complexe de Čech mais est entièrement déterminé par les simplices 0 et 1. Les complexes Vietoris-Rips sont beaucoup plus faciles à utiliser en calcul, en particulier pour les grands ensembles de données, et sont l’un des principaux outils d’analyse de données topologiques.

Si nous adoptons cette approche pour obtenir une représentation topologique, nous pouvons construire un algorithme de réduction de dimension en trouvant une représentation de faible dimension des données qui a une représentation topologique similaire. Si nous ne nous soucions que des simplices 0 et 1, alors la représentation topologique n’est qu’un graphique, et trouver une représentation de faible dimension peut être décrit comme un [problème de disposition du graphique](https://en.wikipedia.org/wiki/Graph_drawing). Si l’on veut utiliser, par exemple, des méthodes spectrales pour la mise en page des graphes, nous arrivons à des algorithmes comme [les cartes propres laplaciennes](https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction#Laplacian_eigenmaps) et [les cartes de diffusion](https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction#Diffusion_maps). Les mises en page dirigées par la force sont également une option et fournissent des algorithmes plus proches de [la cartographie MDS](https://en.wikipedia.org/wiki/Multidimensional_scaling) ou [Sammon](https://en.wikipedia.org/wiki/Sammon_mapping) en termes de saveur.

Je ne blâmerais pas ceux qui ont lu jusqu’ici de se demander pourquoi nous avons pris une route aussi abstraite pour simplement construire un graphique de quartier sur les données et ensuite disposer ce graphique. Il y a plusieurs raisons. La première raison est que l’approche topologique, bien qu’abstraite, fournit une justification théorique solide de ce que nous faisons. Bien que la construction d’un graphe de voisinage et sa disposition dans un espace de dimension inférieure aient un sens heuristique et soient traitables sur le plan informatique, cela ne fournit pas la même motivation sous-jacente de capturer fidèlement la structure topologique sous-jacente des données – pour cela, nous devons faire appel à la puissante machinerie topologique que j’ai suggérée se trouve en arrière-plan. La deuxième raison est que c’est cette approche topologique plus abstraite qui nous permettra de généraliser l’approche et de contourner certaines des difficultés des types d’algorithmes décrits ci-dessus. Bien qu’en fin de compte, nous nous retrouverons avec un processus assez simple sur le plan informatique, il est important de comprendre *pourquoi* diverses manipulations sont importantes pour vraiment comprendre l’algorithme (par opposition à simplement calculer avec lui).

## L’algorithme UMAP

En rassemblant toutes ces pièces, nous pouvons construire l’algorithme UMAP. La première phase consiste à construire une représentation topologique floue, essentiellement comme décrit ci-dessus. La deuxième phase consiste simplement à optimiser la représentation de faible dimension pour avoir une représentation topologique floue aussi proche que possible telle que mesurée par l’entropie croisée.

Lors de la construction de la représentation topologique floue initiale, nous pouvons prendre quelques raccourcis. En pratique, puisque les forces d’appartenance floues se décomposent pour être extrêmement petites, nous n’avons besoin de les calculer que pour les voisins les plus proches de chaque point. En fin de compte, cela signifie que nous avons besoin d’un moyen de calculer rapidement (approximativement) les voisins les plus proches efficacement, même dans les espaces de haute dimension. Nous pouvons le faire en tirant parti de [l’algorithme Nearest-Neighbor-Descent de Dong et al](http://www.cs.princeton.edu/cass/papers/www11.pdf). Les calculs restants ne traitent plus qu’avec les voisins locaux de chaque point et sont donc très efficaces.

En optimisant l’intégration de faible dimension, nous pouvons à nouveau prendre quelques raccourcis. Nous pouvons utiliser la descente de gradient stochastique pour le processus d’optimisation. Pour faciliter le problème de descente de pente, il est avantageux que la fonction d’objectif final soit différentiable. Nous pouvons organiser cela en utilisant une approximation lisse de la fonction de force d’appartenance réelle pour la représentation de faible dimension, en sélectionnant parmi une famille convenablement polyvalente. En pratique, UMAP utilise la famille de courbes de la forme \(\frac{1}{1 + a x^{2b}}\). De même, nous ne voulons pas avoir à traiter avec toutes les arêtes possibles, nous pouvons donc utiliser l’astuce d’échantillonnage négatif (telle qu’utilisée par word2vec et LargeVis), pour simplement échantillonner des exemples négatifs si nécessaire. Enfin, puisque le Laplacien de la représentation topologique est une approximation de l’opérateur de Laplace-Beltrami de la variété, nous pouvons utiliser des techniques d’incorporation spectrale pour initialiser la représentation de faible dimension dans un bon état.

En rassemblant toutes ces pièces, nous arrivons à un algorithme rapide et évolutif, tout en restant construit à partir d’une théorie mathématique solide. J’espère que cette introduction a contribué à fournir une certaine intuition pour cette théorie sous-jacente et pour la façon dont l’algorithme UMAP fonctionne dans la pratique.