Frame 1 - Page de garde Nous allons découvrir une méthode de traitement des signaux à plusieurs dimensions, basée sur le Théorème de superposition de Kolmogorov ; puis nous verrons son application à la compression d'images.

Nous verrons d'abord la preuve non-constructive proposée par Jean-Pierre Kahane, puis son application pratique.

Frame 2 - Introduction Comme souvent en mathématiques et en physique, la dimension est source de difficultés. Plus la dimension, le degré des polynômes, le nombre de degrés de liberté sont grands, plus les phénomènes mathématiques et physique sont compliqués.

Nous avons en physique l'exemple du problème à n corps, aisé pour 2 corps mais faisant intervenir la théorie du chaos pour 3 corps.
En mathématiques, l'exemple le plus frappant est l'inexistence de formules donnant les racines d'un polynôme de degré supérieur à 4, formules qui existent pourtant pour un degré jusqu'à 4.

Dans ce TIPE, je vais vous exposer le Théorème de Superposition de Kolmogorov, qui propose d'écrire les fonctions multidimensionnelles selon une composition de fonctions à une seule variable : de telles fonctions univariées et invariantes pour une même dimension permettent d'engendrer l'ensemble des fonctions à n variables sur le cube unité.

Frame 3 - Énoncé Voici l'énoncé du théorème.
BLABLA
Les fonctions phii étant donc invariantes de toutes fonctions !

Nous allons maintenant voir une preuve non constructive du théorème : elle ne permet d'expliciter ni les fonctions phi ni g, mais a le mérite d'être beaucoup plus simple que si on cherchait à déterminer les fonctions dont on montre l'existence.

Frame 3 - Prérequis Tout d'abord, quelques prérequis.

Le théorème de Baire (lire) Je pourrais vous démontrer ce résultat par la suite si vous le voulez.

La notion de propriété vraie pour quasi tout x de E : cela veut dire que cette propriété est vraie sur une intersection d'ouverts denses sur E, intersection appelée un Gdelta dense.

Enfin, je définis Phi, l'ensemble des fonctions continues croissantes sur [0,1], valant 0 en 0 et 1 en 1.

Frames 5,6 - Preuve Maintenant, place à la preuve.
Un point à relever est que l'on peut prendre les fonctions externes égales entre elles. C'est cette formule que nous allons montrer. (Lire) Par exemple, on peut prendre Q[X], par le théorème de Weierstrass.

Maintenant que nous avons vu une preuve de ce théorème, il est bine sûr intéressant d'en voir l'application pratique.

Frame 7 - Interpolation On transforme une image de n pixels de côté en une fonction sur $[0,1]^2$ par interpolation linéaire entre les points : plus la fonction a une valeur élevée, plus l'image est claire à l'endroit correspondant. On obtient alors une fonction continue quel l'on pourra "traiter" avec le théorème de superposition de Kolmogorov.

Frame 8 - Couches de fonctions Voici comment l'image, ou plutôt la fonction f que nous en avons déduit, est décomposée selon la formule de Kolmogorov.
L'image est la somme de 5 couches de fonctions à une seule variable, correspondant aux 5 fonctions externes g_i. Les fonctions internes sont invariantes sur l'ensemble des images.
Nous voyons enfin la première étape de l'algorithme (qui sera en fait récursif), qui reconstruit de façon déjà assez fidèle l'image d'origine.

Frame 9 - ϕ pour Sprecher Voici la représentation graphique des fonctions ϕ et ξ qui est construite à partir de ϕ, telles qu'elles sont définies pour l'algorithme de Sprecher. ξ sera utilisée par superposition, selon un décalage b. Les fonctions externes sont obtenues par approximations successives, d'où l'aspect récursif que j'ai évoqué. En réalité, deux étapes sont suffisantes pour obtenir l'image de façon strictement fidèle.

Frame 10 - Stockage Pour l'image, seules les fonctions externes g_i sont variables : les fonctions internes sont invariantes et peuvent être enregistrées par exemple dans le programme.
Ces 5 fonctions g_i ne sont pas données par une formule analytique, nous devons alors les discrétiser pour enregistrer leurs valeurs.
En pratique, nous enregistrons $10n$ points par fonction. On passe d'une complexité en $0(n^2)$ à $O(n)$.

Comparons alors le TSK avec le bitmap, format de données brutes, et le Jpg, le format le plus répandu en compression d'images de nos jours.
On voit que le TSK est extrêmement puissant pour les grandes images, ce qui était bien sûr prévisible par la complexité.

Frame 11 - Conclusion Au final, nous avons vu un outil d'analyse très original, mais aussi son application en compression d'images, qui se révèle extrêmement intéressante.
Néanmoins, son utilisation de nos jours ne dépasse pas les mathématiques théoriques ; pour cause : la complexité des algorithmes, dont nous n'avons entrevu que le plus simple, et leur nouveauté – 2010 pour la thèse de M. Leni.

J'ai néanmoins bon espoir que cette méthode s'ouvre d'ici quelques années au grand public (bof), voire à la compression de vidéo (le temps étant la 3^{e}dimension) où l'on passerait entre guillemets de n^3 à $7n$ valeurs à enregistrer.