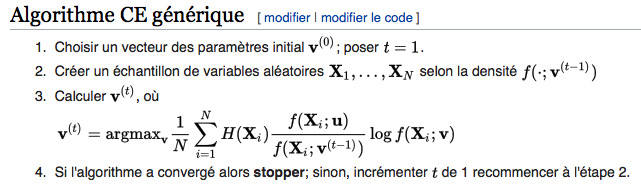
**Algorithme de type CE (cross-entropy) basé sur la famille paramétrique des produits de lois de Bernoulli**

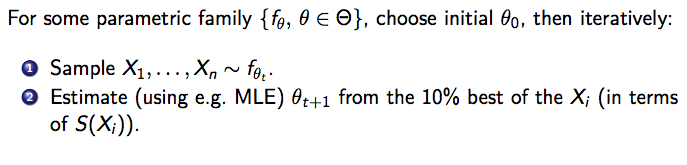
La méthode CE se décompose en deux phases :

1. Créer aléatoirement un échantillon de données (trajectoires, vecteurs, etc.) selon un mécanisme spécifique.
2. Mettre à jour les paramètres du mécanisme de création aléatoire à partir de l'échantillon de données pour produire un meilleur échantillon à l'itération suivante. Cette étape implique de minimiser l'[entropie croisée](https://fr.wikipedia.org/wiki/Entropie_crois%C3%A9e) ou la [divergence de Kullback-Leibler](https://fr.wikipedia.org/wiki/Divergence_de_Kullback-Leibler).



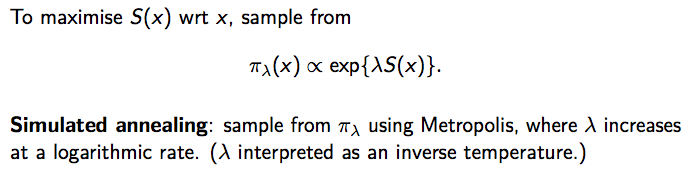
Lien intéressant sur la méthode: https://rfia2012.files.wordpress.com/2012/02/la-mc3a9thode-lentropie-croisc3a9e.pdf

Dans le cours, dernière slide (et sujet du dernier TD):



**Algorithme de type recuit simulé, basé sur un noyau de type Metropolis, avec idée: choisir une composante au hasard, et tenter de la modifier.**

Dans le cours, avant-dernière slide (simulated annealing):



Définition Wikipédia:

Le recuit simulé s'appuie sur l'[algorithme de Metropolis-Hastings](https://fr.wikipedia.org/wiki/Algorithme_de_Metropolis-Hastings), qui permet de décrire l'évolution d'un système [thermodynamique](https://fr.wikipedia.org/wiki/Thermodynamique). Par analogie avec le processus physique, la fonction à minimiser deviendra l'énergie E du système. On introduit également un paramètre fictif, la température T du système.

Partant d'un état donné du système, en le modifiant, on obtient un autre état. Soit celui-ci améliore le critère que l'on cherche à optimiser, on dit alors qu'on a fait baisser l'énergie du système, soit celui-ci le dégrade. Si on accepte un état améliorant le critère, on tend ainsi à chercher l'optimum dans le voisinage de l'état de départ. L'acceptation d'un « mauvais » état permet alors d'explorer une plus grande partie de l'espace des états et tend à éviter de s'enfermer trop vite dans la recherche d'un optimum local.

**Etat initial de l'algorithme:**

L'état initial peut être pris au hasard dans l'espace des états possibles. À cet état correspond une énergie initiale E=E0

Cette énergie est calculée en fonction du critère que l'on cherche à optimiser. Une température initiale T=T0

élevée est également choisie. Ce choix est alors totalement arbitraire et va dépendre de la loi de décroissance utilisée.

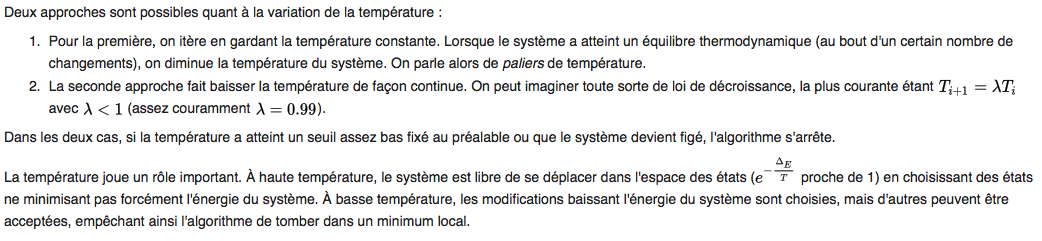
**Itérations de l'algorithme:**

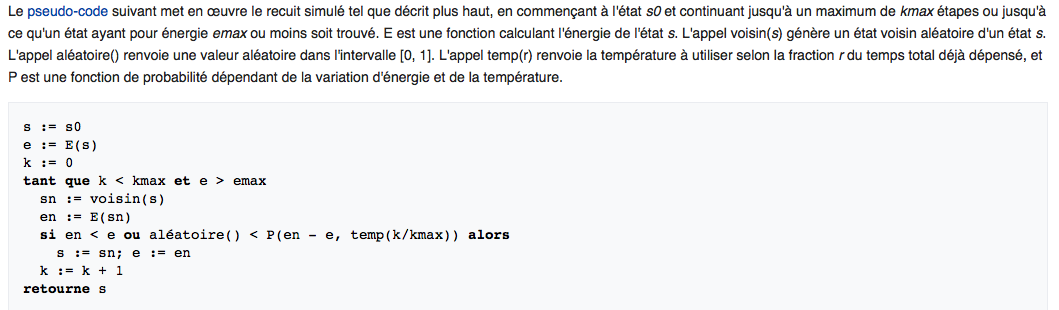
À chaque itération de l'algorithme une modification élémentaire de l'état est effectuée. Cette modification entraîne une variation ΔE de l'énergie du système (toujours calculée à partir du critère que l'on cherche à optimiser). Si cette variation est négative (c'est-à-dire qu'elle fait baisser l'énergie du système), elle est appliquée à l'état courant. Sinon, elle est acceptée avec une probabilité

e ^(− ΔE/T). Ce choix de l'exponentielle pour la probabilité s'appelle **règle de Metropolis.**

On itère ensuite selon ce procédé en gardant la température constante.

**Programme de recuit:**



Pseudo-code:

**Algorithme de type recuit simulé, basé sur un noyau de Gibbs (“deterministic scan”: on simule en boucle la composante 1, puis 2, puis 3, etc.)**

🡪 cf p97 de http://perso.telecom-paristech.fr/~tupin/cours/polymrf.pdf