L'algorithme NMF

Léonard Blier

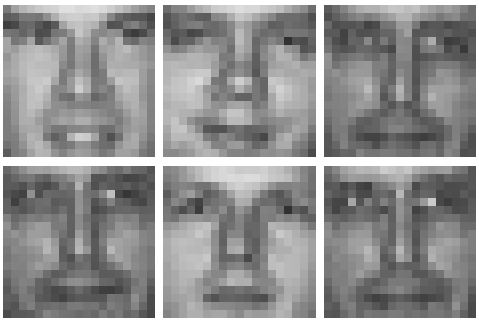
13 mai 2013

Résumé

L'algorithme NMF (*Non Negative Matrix Factorization*) est un algorithme de Data Mining, dont l'objectif est de déterminer et reconnaître les parties élémentaires des données fournies, comme par exemple les différentes parties d'un visage lorsque la base est composée de photographies de visages. L'algorithme consiste à factoriser des matrices avec une contrainte de positivité. Ici, il est appliqué sur deux bases de données, un ensemble de photographies de visage, et un ensemble d'articles de Wikipedia. Puis étudié d'un point de vue mathématique, en le ramenant à un cas de l'algorithme d'espérance maximisation, dont l'objectif est de maximiser la vraisemblance.

1 Présentation générale de l'algorithme

On applique dans cette partie l'algorithme à une base de donnée de photos de visages. Chaque photo est de taille $n = 19 \times 19$ pixels, en niveaux de gris. On assimile chaque pixel à un réel de [0,1], 0 pour blanc, 1 pour noir, et on représente ainsi chaque photographie comme un (19×19)-uplet de réels, soit un vecteur de $\mathbb{R}^{19\times 19}$. On dispose d'une base de donnée de $m = 2429$ photos, dont les vecteurs sont notés $(v_i)_{i\in[1,2429]}$. On pose $\mathcal{V} = \mathcal{M}((v_i)_{i\in[1,2429]})$, la matrice dont chaque colonne représente une photographie. On remarque ainsi que la matrice V est positive.



Six visages de la base de donnée

L'objectif de l'algorithme est de factoriser de façon approchée la matrice V en deux matrices positives de taille choisie, soit de choisir un paramètre $r \in \mathbb{N}$ et de déterminer deux matrices $\mathcal{W} \in \mathcal{M}_{19\times 19,r}(\mathbb{R}), \mathcal{H} \in \mathcal{M}_{r,2429}(\mathbb{R})$ positives telles que $\mathcal{V} = \mathcal{W}\mathcal{H}$.

Les intérêts d'une factorisation sont les suivants :

- Chaque photographie est exprimée comme combinaison linéaire des colonnes de $\mathcal{W}$. Si on note h_i les vecteurs colonnes de h, alors on a $v_i = \mathcal{W}h_i$. Les colonnes de W peuvent ainsi être interprétées comme les images élémentaires à partir desquelles on peut former toutes les images.
- L'information est compressée. Le paramètre r est choisi de telle sorte que $(n+m)\times r << n\times m$. Dans cet exemple, on choisit $r = 60$, donc $(n+m)\times r = 167400$, et $n\times m = 876869$, donc le taux de compression est de l'ordre de 5. Il sera bien plus important dans l'exemple présenté par la suite.

– On obtient une représentation des photographies dans un espace de dimension plus faible. Comme $v_i = Wh_i$, une photographie est déterminée uniquement par h_i, vecteur d'un espace de dimension $r = 60$, et non $n = 19 \times 19 = 361$, ce qui permet de travailler plus facilement sur ces données.

L'apport de NMF par rapport à d'autres algorithmes de factorisation de matrices [1] est la contrainte de positivité pour les matrices W et H, inspirée par des recherches en sciences cognitives. À la suite d'études montrant que la vision animale est basée sur une reconnaissance des parties d'un objet, les chercheurs Daniel D. Lee (Laboratoires Bell) et H. Sebastian Seung (Département du cerveau et des sciences cognitives, MIT) ont cherché un algorithme calculant $\mathcal{W}$ et $\mathcal{H}$ telles que $\mathcal{V} = \mathcal{WH}$, mais telles que les colonnes de $\mathcal{W}$ forment effectivement les différentes parties d'une photographie de visage reconnue par l'homme. Autrement dit, là où une simple factorisation fournissait une famille génératrice des photographies de visages, ici cette famille doit représenter les différentes parties des visages, et la matrice $\mathcal{H}$ indique de quelle façon ces images doivent être superposées pour former la matrice $\mathcal{V}$. Cette contrainte impose la positivité de $\mathcal{W}$ et $\mathcal{H}$, car il est naturel que la reconnaissance des parties soit fondée sur une superposition des parties, autrement dit, une combinaison linéaire à coefficients positifs d'éléments pouvant être interprétés comme des images, donc positifs.

L'algorithme NMF est itératif. On commence par choisir aléatoirement deux matrices $\mathcal{W}$ et $\mathcal{H}$, puis, on applique à chacune de leurs coordonnées la formule suivante un certain nombre de fois (ici 50).

$$H_{k,j} \leftarrow H_{k,j} \frac{\sum_{i=1}^{n} \frac{W_{i,k} V_{i,j}}{(WH)_{i,j}}}{\sum_{k=1}^{n} W_{i,k}} \qquad W_{i,k} \leftarrow W_{i,k} \frac{\sum_{j=1}^{m} \frac{H_{k,j} V_{i,j}}{(WH)_{i,j}}}{\sum_{j=1}^{m} H_{k,j}}$$

Remarque : Les deux formules sont identiques, on a simplement pris la transposée.

La formule est calculée en plusieurs étapes lors de l'exécution. On commence par calculer le produit $\mathcal{WH}$ pour les matrices $\mathcal{W}$ et $\mathcal{H}$ d'origine, et on le garde en mémoire. Puis, à chaque fois que l'on calcule la nouvelle valeur d'une case (en n opérations élémentaires dans le cas de H), on modifie le produit $\mathcal{WH}$ sur les cases concernées (en n opérations dans le cas où on a modifié $\mathcal{H}$). On en déduit que l'exécution de l'algorithme sur toute la matrice $\mathcal{H}$ prend $O(n \times r \times m)$. Le résultat est donc identique pour $\mathcal{W}$, par symétrie des rôles de n et m. Or, le calcul de $\mathcal{WH}$ est au plus en $O(n \times r \times m)$ également. On en conclut que le nombre total d'opérations effectuées lors de l'execution de l'algorithme est bien $O(n \times r \times m)$, ceci devant être multiplié par le nombre d'exécutions de l'algorithme choisi.

Les résultats sont alors effectivement ceux attendus :

– Les images factorisées sont bien très proches des images originelles. (Sur la ligne supérieure, l'original, sur la ligne inférieure, le résultat obtenu en multipliant W et H)

1. Par exemple, l'analyse à composantes principales

– Les colonnes de W, réinterprétées comme des images, mettent bien en évidence les parties élémentaires du visage (nez, bouche, sourcil, ...). Ici, une sélection parmi les 60 images élémentaires.

L'application de l'algorithme NMF à une base de donnée de visages permet de mettre en évidence son efficacité, tant pour factoriser la matrice que pour identifier les parties élémentaires des données de la base, sur un exemple où on connaît les résultats attendus. Cette méthode permet bien une étude de la base de donnée fondée sur la reconnaissance des parties.

2 Application à l'analyse de Wikipedia

Dans cette partie, on s'intéresse à une nouvelle base de donnée, constituée d'articles de Wikipedia. On assimile un article au tableau contenant le nombre d'apparition de chaque mot dans cet article. La matrice V des données est alors la matrice dont les colonnes sont ces tableaux.

La base de donnée étant trop grande pour être traitée sur un simple ordinateur, on fait une sélection parmi tous les articles, et parmi tous les mots apparaissant dans les articles, dans le but de garder les articles les plus longs, et les mots assez fréquents, car il est plus facile avec ceux-ci d'établir des corrélations entre les articles. Dans un premier temps, on supprime une liste de 200 mots, les plus courants et n'apportant aucune information sur le sens (articles, verbes "être" et "avoir" conjugués, ...), et on fusionne les mots dont on est presque certains qu'ils signifient la même chose (verbes du premier groupes dans toutes leurs conjugaisons, pluriels en "s", "aux", "eaux", ...). Puis, on ne conserve que les articles qui, après cette opération, possèdent plus de 2500 mots [2], soit environ 40,000 articles. On ne conserve ensuite comme mots dont on étudie l'apparition que les 6000 mots les plus fréquents parmi ceux apparaissant dans les 40,000 articles, avec comme contrainte supplémentaire que ces mots doivent apparaître dans au moins cinq articles différents, pour éviter le phénomène des mots extrêmement spécifiques, qui n'apparaîtraient que dans un ou deux articles, mais un grand nombre de fois. Enfin, on choisit aléatoirement 10,000 articles, qui seront ceux que l'on étudiera. Il est important de ne pas choisir ces articles uniquement selon leur taille, car il y a un risque pour que les plus longs articles traitent tous sensiblement des mêmes thèmes.

On obtient ainsi une matrice $\mathcal{V} \in \mathcal{M}_{6000,10000}(\mathbb{R})$, que l'on va chercher comme dans la partie précédente à factoriser en $\mathcal{V} = \mathcal{W}\mathcal{H}$. Comment interpréter cette factorisation ici ? Là où les colonnes de $\mathcal{W}$ représentaient dans la partie précédente les parties élémentaires des photographies de visages, ici, elles représenteront les parties élémentaires des articles, soit l'ensemble des thèmes qui permettent de composer les articles.

2. 2500 mots représentent environ trois pleines pages de texte en Times police 12.

Dans un premier temps, on peut observer quels sont ces thèmes élémentaires. Ici, une sélection parmi les 100 thèmes mis en évidence. On affiche à chaque fois les vingts mots apparaissant le plus dans chaque thème :

Thème 1	**Thème 2**	**Thème 3**	**Thème 4**	**Thème 5**	**Thème 6**	**Thème 7**
informatique	économique	organisation	film	science	vin	église
logiciel	entreprise	pays	cinéma	théorie	blanc	dieu
version	travail	nation	théâtre	philosophie	provence	religion
système	économie	état	acteur	scientifique	rouge	catholique
windows	marcher	internationale	scène	connaissance	mont	homme
ordinateur	développement	unie	meilleur	dioscorea	rhône	vie
microsoft	service	international	réalisateur	monde	alpes	religieuse
internet	société	membre	rôle	physique	appellation	foi
fichier	état	arabe	naissance	idée	jpg	monde
carter	activité	monde	oscar	logique	département	même
projet	politique	israël	festival	être	aoc	cette
mac	sociale	al	mise	esprit	vigne	esprit
numérique	gestion	guerre	award	pensée	unité	être
apple	financer	irak	télévision	nature	france	jean
os	social	onu	mort	philosophe	savoie	citation
sous	banque	union	actrice	cette	fromage	religieux
sourcer	produit	entrer	cannes	concept	origine	amour
libre	pays	accord	image	méthode	vaucluse	toute
licence	production	article	année	expérience	vignoble	chrétien
utiliser	cette	sécurité	producteur	sens	production	chrétienne

On peut également construire un dendrogramme des articles. Un dendrogramme est un graphique représentant un arbre binaire dont les feuilles sont les éléments de la base de donnée, et construit de la façon suivante : On assimile ici un article v_i au vecteur colonne h_i de $\mathcal{H}$ tel que $v_i = Wh_i$. Ainsi, on représente l'ensemble des articles dans $\mathbb{R}^{100}$. On munit cet espace de la norme euclidienne canonique. On définit un type arbre binaire dont les sommets sont étiquetés par leur poids p, la distance à ses fils d, et sa position dans l'espace x. Les feuilles portent en plus le titre de l'article qu'elles représentent. L'algorithme est alors celui-ci :

Étape 1 On construit une liste des feuilles, $liste := [Feuille(x = h_i, p = \text{longueur de l'article i}, d = 0, titre = titre_i), i \in [1, m]]$

Étape 2 Si $liste = [a]$, on renvoie a. Sinon :

Étape 3 On construit un tableau D contenant les distances entre N_1 et N_2 pour tout $(N_1, N_2) \in liste$. La distance entre deux noeuds est la distance entre leurs positions.

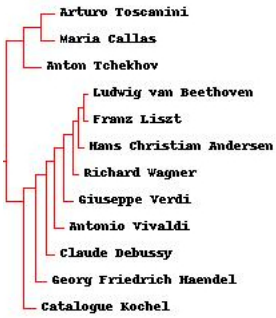
Étape 4 On détermine $N_1 = Noeud(x_1, p_1, d_1)$ et $N_2 = Noeud(x_2, p_2, d_2)$ tels que $||x_1 - x_2||$ soit le minimum de D.

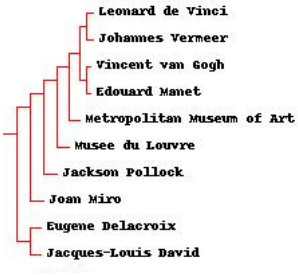
Étape 5 On supprime N_1 et N_2 de $liste$, et on y ajoute leur barycentre pondéré, soit $M = Noeud(x = \frac{p_1 x_1 + p_2 x_2}{p_1 + p_2},\ p = p_1 + p_2,\ d = ||x_1 - x_2||)$

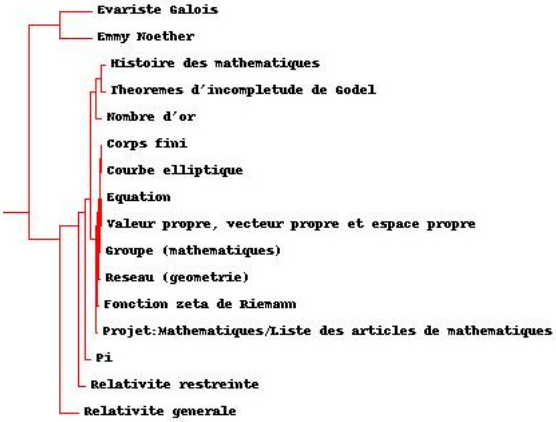
Étape 6 Retour à l'étape 2

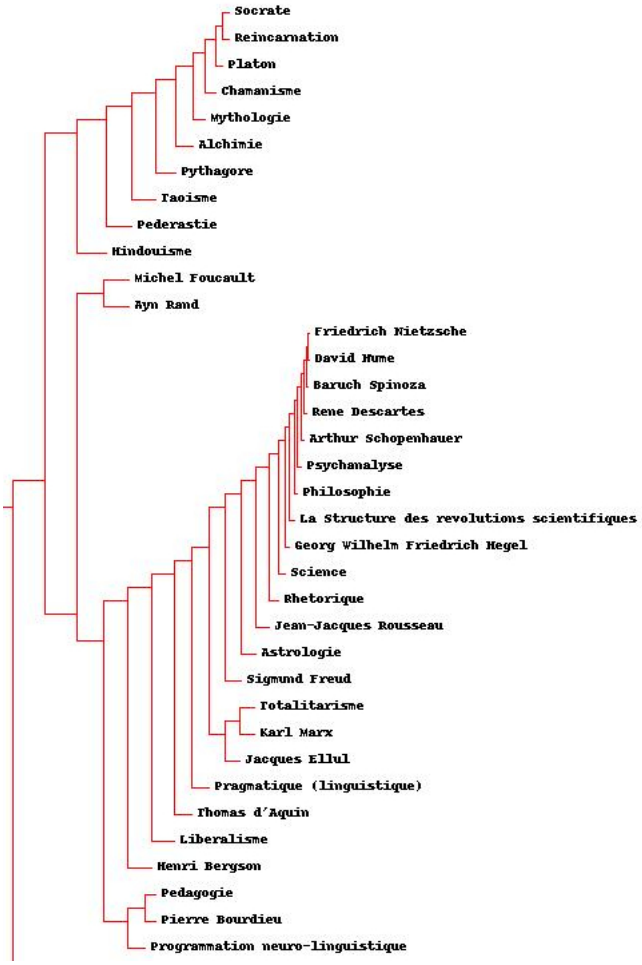
Dans la représentation de l'arbre, la longueur horizontale d'une arête est croissante avec la distance du noeud père à ses fils [3], ce qui permet d'observer si les rapprochements effectués sont très pertinents ou non. Voici quelques extraits d'un dendrogramme contenant 1300 articles.

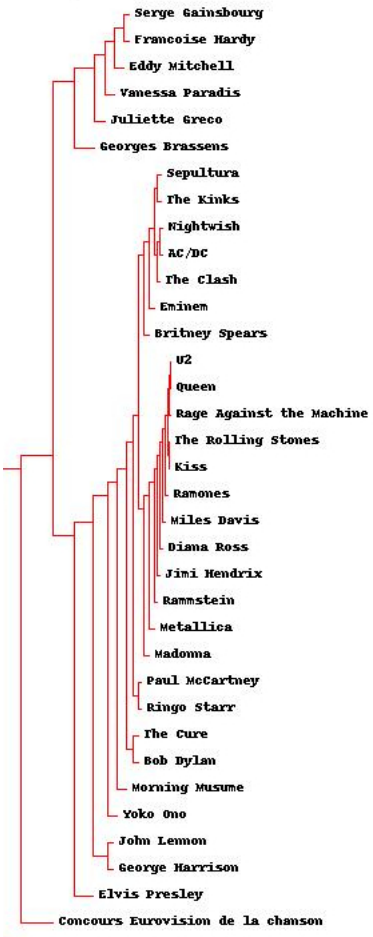
3. Dans la représentation, la longueur d'une arête est en réalité proportionnelle à $\log(1 + d)$, pour pouvoir représenter les petites comme les grandes distances sur un même document


Arturo Toscanini
Maria Callas
Anton Tchekhov
Ludwig van Beethoven
Franz Liszt
Hans Christian Andersen
Richard Wagner
Giuseppe Verdi
Antonio Vivaldi
Claude Debussy
Georg Friedrich Haendel
Catalogue Kochel


Leonard de Vinci
Johannes Vermeer
Vincent van Gogh
Edouard Manet
Metropolitan Museum of Art
Musee du Louvre
Jackson Pollock
Joan Miro
Eugene Delacroix
Jacques-Louis David


Evariste Galois
Emmy Noether
Histoire des mathematiques
Theoremes d'incompletude de Godel
Nombre d'or
Corps fini
Courbe elliptique
Equation
Valeur propre, vecteur propre et espace propre
Groupe (mathematiques)
Reseau (geometrie)
Fonction zeta de Riemann
Projet:Mathematiques/Liste des articles de mathematiques
Pi
Relativite restreinte
Relativite generale


Socrate
Reincarnation
Platon
Chamanisme
Mythologie
Alchimie
Pythagore
Taoisme
Pederastie
Hindouisme
Michel Foucault
Ayn Rand
Friedrich Nietzsche
David Hume
Baruch Spinoza
Rene Descartes
Arthur Schopenhauer
Psychanalyse
Philosophie
La Structure des revolutions scientifiques
Georg Wilhelm Friedrich Hegel
Science
Rhetorique
Jean-Jacques Rousseau
Astrologie
Sigmund Freud
Totalitarisme
Karl Marx
Jacques Ellul
Pragmatique (linguistique)
Thomas d'Aquin
Liberalisme
Henri Beryson
Pedagogie
Pierre Bourdieu
Programmation neuro-linguistique


Serge Gainsbourg
Francoise Hardy
Eddy Mitchell
Vanessa Paradis
Juliette Greco
Georges Brassens
Sepultura
The Kinks
Nightwish
AC/DC
The Clash
Eminem
Britney Spears
U2
Queen
Rage Against the Machine
The Rolling Stones
Kiss
Ramones
Miles Davis
Diana Ross
Jimi Hendrix
Rammstein
Metallica
Madonna
Paul McCartney
Ringo Starr
The Cure
Bob Dylan
Morning Musume
Yoko Ono
John Lennon
George Harrison
Elvis Presley
Concours Eurovision de la chanson

.


Pier Paolo Pasolini
Abbas Kiarostami
Pulp Fiction
Harrison Ford
Kirikou et la Sorciere
Gladiator (film, 2000)
Alfred Hitchcock
Les Oiseaux (film, 1963)
Clint Eastwood
Le Bon, la Brute et le Truand
Angelina Jolie
Les Sept Samourais
Stanley Kubrick
John Cassavetes
Steven Spielberg
Roman Polanski
Catherine Deneuve
Festival de Cannes
Louis de Funes
Marilyn Monroe
Akira Kurosawa
Cinema
Transgenre
Peplum
Kevin Bacon
Natalie Portman
United Artists
Paramount Pictures
Sebastien Japrisot
Stephen King
J. R. R. Tolkien
Homer Simpson
Bart Simpson
Lucky Luke
Mickey Mouse
Donald Duck
Walt Disney
Le Seigneur des anneaux : Les Deux Tours
Fantasy
Andre Franquin
Gollum
Jack Kirby
Bilbo le Hobbit
Le Seigneur des anneaux
Gaston (bande dessinee)
James Bond
Le Seigneur des anneaux : Le Retour du roi
Warner Bros.
Buffy contre les vampires
Les Soprano
Valerian et Laureline
Jean-Claude Mezieres
Lara Croft
Dracula
Western
X-Files : Aux frontieres du reel
NCIS : Enquetes speciales
Friends
Futurama
Scrubs
Vampire
Les Tetes brulees (serie televisee)
Sense and Sensibility

3 Approche mathématique

3.1 NMF : Un problème de maximisation de vraisemblance

L'algorithme NMF ne désigne non pas la formule énoncée précédemment, mais le fait de factoriser la matrice des données en deux matrices positives. Il existe de nombreuses façons d'appliquer cette méthode, qui cherchent à minimiser $||\mathcal{V} - \mathcal{WH}||$ pour différentes normes. Dans leurs premiers articles, Lee et Seung proposent deux méthodes, toutes deux multiplicatives, la première minimisant la norme euclidienne canonique, la seconde la divergence de Kullback-Leibler. Les mérites de ces formules étaient selon eux une implémentation facile, et une convergence relativement rapide. Depuis, d'autres méthodes ont été développées, basées notamment sur la descente du gradient. Elles ne sont alors plus multiplicatives mais additives, et garantissent des convergences encore plus rapides. Ici, on choisit de ne s'intéresser qu'à l'algorithme NMF minimisant la divergence de Kullback-Leibler dans sa version multiplicative, car il est possible d'en donner une interprétation probabiliste.

On va ramener le problème de factorisation de matrice à un problème de maximisation de vraisemblance, et pour cela, on ramène notre tableau de donnée à un tirage aléatoire. Ici, un article n'est plus assimilé à un tableau d'occurrence des mots, mais à un ensemble de probabilité d'apparition de chaque mot (inconnu), et chaque tableau d'occurrence d'un article d'une longueur de k mots est un tirage aléatoire de k mots selon les probabilités d'apparition de chaque mot dans cet article.

L'hypothèse de NMF, c'est à dire qu'un article est caractérisé par l'importance donnée à un chaque thème élémentaire, se traduit de la façon suivante : On suppose l'existence de thèmes, qui correspondent aux colonnes de $\mathcal{W}$, et chaque thème est également un ensemble de probabilités d'apparition de chaque mot. Chaque article est caractérisé par l'importance donnée à chaque thème, ce que l'on modélise ici en considérant qu'un article est un ensemble de probabilités d'apparition de chaque thème. Par hypothèse, on considère que choisir aléatoirement un mot dans un article revient à choisir aléatoirement un thème parmi ceux exprimés par cet article, puis à choisir aléatoirement un mot dans ce thème. La matrice $\mathcal{V}$ des données est alors considéré comme les résultats de tirages successifs.

L'objectif est alors, à partir des tirages donnés (la matrice $\mathcal{V}$), de déterminer la probabilité d'apparition de chaque mot dans chaque thème, et la probabilité d'apparition de chaque thème dans chaque article. On montre que ceci revient bien à factoriser $\mathcal{V}$ et deux matrices positives $\mathcal{W}$ et $\mathcal{H}$. On note W la variable aléatoire pouvant prendre toutes les valeurs $(w_i)_{i\in[1,n]}$, c'est-à-dire tous les mots, D la variable aléatoire pouvant prendre toutes les valeurs $(d_j)_{j\in[1,m]}$, où chaque d_j représente un article, et enfin H la variable aléatoire pouvant prendre toutes les valeurs $(h_k)_{k\in[1,r]}$, c'est-à-dire tous les thèmes élémentaires.

Soit $\mathcal{M} = (P(W = w_i, D = d_j))_{i,j}$, la matrice contenant la probabilité d'obtenir le mot w_i dans d_j pour tout (i, j) en tirant un mot aléatoirement dans un article choisi aléatoirement. On a :

$$\mathcal{M}_{i,j} = P(W = w_i, D = d_j)$$

$$\mathcal{M}_{i,j} = \sum_{k=1}^{r} P(W = w_i, H = h_k | D = d_j) \times P(D = d_j)$$

$$\mathcal{M}_{i,j} = \sum_{k=1}^{r} P(W = w_i | H = h_k, D = d_j) \times P(H = h_k | D = d_j) \times P(D = d_j)$$

Or $P(W = w_i|H = h_k, D = d_j) = P(W = w_i|H = h_k)$, car l'article n'a plus d'influence si on connait le thème dont va être issu le mot, par hypothèse. Donc :

$$\mathcal{M}_{i,j} = \sum_{k=1}^{r} P(W = w_i|H = h_k) \times P(H = h_k|D = d_j) \times P(D = d_j)$$

On définit :

$$\mathcal{W} := (P(W = w_i|H = h_k))_{i\in[1,n],j\in[1,r]}$$
$$\mathcal{H} := (P(H = h_k|D = d_j) \times P(D = d_j))_{k\in[1,r],j\in[1,m]}$$

On a alors immédiatement :

$$\mathcal{M} = \mathcal{W}\mathcal{H}$$

Or, $\mathcal{M}$ représente la distribution de probabilité dont sont issus les tirages de V. Donc, chercher $\mathcal{W}$ et $\mathcal{H}$ les plus vraisemblables revient à maximiser la fonction de vraisemblance de loi multinomiale suivante :

$$L = \frac{(\sum_{i,j} V_{i,j})!}{\prod_{i,j}(V_{i,j}!)} \prod_{i=1}^{n}\prod_{j=1}^{m} P(W = w_i, D = d_j)^{V_{i,j}}$$

Donc :

$$L(\mathcal{W},\mathcal{H}) = \frac{(\sum_{i,j} V_{i,j})!}{\prod_{i,j}(V_{i,j}!)} \prod_{i=1}^{N}\prod_{j=1}^{M}(\sum_{k=0}^{r} P(D = d_j)P(W = w_i|H = h_k)P(H = h_k|D = d_j))^{V_{i,j}}$$

$$L(\mathcal{W},\mathcal{H}) = \frac{(\sum_{i,j} V_{i,j})!}{\prod_{i,j}(V_{i,j}!)} \prod_{i=1}^{N}\prod_{j=1}^{M}(\sum_{k=0}^{r} \mathcal{W}_{i,k}\mathcal{H}_{k,j})^{V_{i,j}}$$

Comme on ne cherche qu'à maximiser cette fonction, on ne considère pas le terme multiplicatif constant, et on prend la log-vraisemblance afin de faciliter les calculs :

$$\log(L(\mathcal{W},\mathcal{H})) = \sum_{i=1}^{n}\sum_{j=1}^{m} V_{i,j} \log(\sum_{k=0}^{r} \mathcal{W}_{i,k}\mathcal{H}_{k,j})$$

Remarque : Minimiser cette fonction est équivalent à minimiser la divergence de Kullback-Leibler entre $\mathcal{V}$ et $\mathcal{W}\mathcal{H}$.

On ne peut minimiser cette fonction directement par le calcul, et on est donc amenés à utiliser l'algorithme d'Espérance Maximisation.

3.2 L'algorithme d'espérance maximisation

L'algorithme d'espérance maximisation s'utilise dans le cas suivant :

On dispose de données, y, obtenues aléatoirement selon une loi $P_\theta(Y)$, où θ est le paramètre que l'on souhaite déterminer, de telle sorte à maximiser $L : \theta \to P_\theta(Y = y)$. Mais cette fonction est trop complexe, et on ne peut trouver son maximum directement.

On suppose ici qu'il existe des "variables cachées", x, telles que leur connaissance détermine à elles seules, sans besoin de θ, la loi suivie par Y. On dispose donc des lois $P_\theta(X = x)$ et $P_\theta(Y = y|X = x)$, cette dernière telle que $P_\theta(Y = y|X = x) = P(Y = y|X = x)$, autrement dit, qu'elle ne dépende pas de θ. L'algorithme est le suivant :

Étape 1 $m \leftarrow 0$. On fait une première évaluation de θ, souvent de manière aléatoire, et on la note θ_0

Étape 2 On détermine la fonction $x \to P_{\theta_m}(X = x|Y = y)$

Étape 3 On en déduit la fonction

$$Q(\theta, \theta_m) = \sum_x \log(P_\theta(X = x, Y = y)) P_{\theta_m}(X = x|Y = y)$$

$$= E_{X|Y=y,\theta_m}(\log(P_\theta(X = x, Y = y)))$$

Q s'interprète comme l'espérance de la log-vraisemblance selon les valeurs de X, d'après la loi déterminée à l'étape 2.

Étape 4 On maximise Q, et on définit $\theta_{m+1} \leftarrow \arg\max_\theta Q(\theta, \theta_m)$

Étape 5 $m \leftarrow m + 1$. Retour à l'étape 2.

Remarque : Décrit de cette façon, l'algorithme ne termine pas. Il est nécessaire de choisir à l'avance lorsque l'algorithme s'arrête, soit après un certain nombre d'itérations, soit lorsque $||\theta_{m+1} - \theta_m|| < \epsilon$, où on a préalablement défini ϵ.

Cet algorithme ne garantit pas la convergence de $(\theta_m)_m$ mais la décroissance stricte [4] de la log-vraisemblance à chaque étape, comme cela est démontré en annexe.

La fonction Q représente, à une constante additive près, l'opposé de la divergence de Kullback-Leibler entre $P_{\theta_m}(X|Y = y)$ et $P_\theta(X|Y = y)$. Maximiser Q revient donc à minimiser la divergence entre ces deux distributions. Pour résumer, l'algorithme consiste, à partir d'une estimation θ_m à déterminer une distribution de probabilité $P_{\theta_m}(X|Y = y)$, puis à déterminer le paramètre θ qui rende cette distribution la plus vraisemblable.

Le problème restant est de maximiser la fonction Q, mais comme on le verra dans la suite, ce problème est beaucoup plus facile que le précédent.

3.3 Application de l'Espérance-Maximisation à NMF

On choisit comme variable cachée la variable x, qui indique, pour chaque tirage, le thème dont provient le mot qui a été tiré. On a effectué $\sum_{i,j} V_{i,j}$ tirages, et il y a r thèmes. Il y a donc $r^{\sum_{i,j} V_{i,j}}$ valeurs possibles de x.

On suppose que l'on est à l'étape m, et on connaît donc des estimation $\mathcal{H}^{(m)}$ et $\mathcal{W}^{(m)}$.

On détermine $P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x|W = w_i, D = d_j)$. On a :

$$P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|W = w_i, D = d_j) = \frac{P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i|X = x_k, D = d_j) \times P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|D = d_j)}{P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i|D = d_j)}$$

Or $P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i|X = x_k, D = d_j) = P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i|X = x_k)$, donc :

$$P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|W = w_i, D = d_j) = \frac{P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i|X = x_k) \times P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|D = d_j)}{\sum_s P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i, X = x_s|D = d_j)}$$

$$= \frac{P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i|X = x_k) \times P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|D = d_j)}{\sum_s P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(W = w_i|X = x_s) P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_s|D = d_j)}$$

4. La suite ne devient stationnaire que si θ_m et θ_{m+1} décrivent exactement la même loi de probabilité.

$$= \frac{\mathcal{W}^{(m)}_{i,k}\mathcal{H}^{(m)}_{k,j}}{(\mathcal{W}^{(m)}\mathcal{H}^{(m)})_{i,j}}$$

On détermine ensuite $E_{X|\mathcal{V},\mathcal{H}^{(m)},\mathcal{W}^{(m)}}[\log(P_{\mathcal{H},\mathcal{W}}(\mathcal{V}, X = x))]$

$$E_{X|\mathcal{V},\mathcal{H}^{(m)},\mathcal{W}^{(m)}}[\log(P_{\mathcal{H}}(\mathcal{V}, X = x)] = \sum_{x_k} P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|V = \mathcal{V}) \sum_{i=1}^{n}\sum_{j=1}^{m} V_{i,j} \log(P(W = w_i, D = d_j, X = x_k))$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{x_k} V_{i,j} P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|V = \mathcal{V}) \log(P(W = w_i|X = x_k)P(X = x_k|D = d_j))$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{x_k} V_{i,j} P_{\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(X = x_k|V = \mathcal{V}) \log(\mathcal{W}_{i,k}\mathcal{H}_{k,j})$$

Donc, d'après le calcul précédent :

$$E_{X|\mathcal{V},\mathcal{H}^{(m)},\mathcal{W}^{(m)}}(\log(P_{\mathcal{H},\mathcal{W}}(\mathcal{V}, X = x)) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=0}^{r} V_{i,j} \frac{\mathcal{W}^{(m)}_{i,k}\mathcal{H}^{(m)}_{k,j}}{(\mathcal{W}^{(m)}\mathcal{H}^{(m)})_{i,j}} \log(\mathcal{W}_{i,k}\mathcal{H}_{k,j})$$

Pour maximiser cette fonction, on utilise les multiplicateurs de Lagrange (démonstration complète en annexe). Le résultat est alors bien celui attendu :

$$\mathcal{H}^{(m+1)}_{k,j} = \mathcal{H}^{(m)}_{k,j} \frac{\sum_{i=1}^{n} \frac{\mathcal{W}^{(m)}_{i,k}\mathcal{V}_{i,j}}{(\mathcal{W}^{(m)}\mathcal{H}^{(m)})_{i,j}}}{\sum_{k=1}^{n} \mathcal{W}^{(m)}_{i,k}}$$

A Annexe : La vraisemblance

On dispose d'une loi de probabilité $P_\theta(X)$, sur un ensemble fini d'événements, où θ joue un rôle de paramètre. Étant donné un évènement x, la vraisemblance d'un paramètre θ est donnée par la fonction $L : \theta \to P_\theta(X = x)$. L'objectif est de maximiser cette fonction de vraisemblance, soit de déterminer le paramètre θ rendant le plus probable les évènements qui se sont effectivement produits.

B Annexe : Démonstration de l'algorithme Espérance Maximisation

La démonstration de cet algorithme repose essentiellement sur la concavité du logarithme. (On reprend ici les notations de la partie précédente).

Soit $l(\theta) = \log(P_\theta(Y = y))$, la log-vraisemblance. On a :

$$l(\theta) = \log(\sum_x P_\theta(Y = y, X = x))$$

En multipliant numérateur et dénominateur par le même facteur :

$$l(\theta) = \log(\sum_x \frac{P_\theta(X=x, Y=y) \times P_{\theta_m}(X=x|Y=y)}{P_{\theta_m}(X=x|Y=y)})$$

Or, $\sum_x P_{\theta_m}(X=x|Y=y) = 1$ et log est concave. On a donc :

$$l(\theta) \geq \sum_x P_{\theta_m}(X=x|Y=y) \log(\frac{P_\theta(X=x, Y=y)}{P_{\theta_m}(X=x|Y=y)})$$

$$= E_{X|Y=y,\theta_m}[\log(\frac{P_\theta(X=x, Y=y)}{P_{\theta_m}(X=x|Y=y)})]$$

$$= E_{X|Y=y,\theta_m}[\log(\frac{P_\theta(X=x, Y=y) \times P_{\theta_m}(Y=y)}{P_{\theta_m}(X=x, Y=y)})]$$

$$= E_{X|Y=y,\theta_m}[\log(P_\theta(X=x, Y=y))] + \log(P_{\theta_m}(Y=y)) - E_{X|Y=y,\theta_m}[P_{\theta_m}(X=x, Y=y))]$$

$$= Q(\theta, \theta_m) + l(\theta_m) - Q(\theta_m, \theta_m)$$

On en conclut que :

$$l(\theta) - l(\theta_m) \geq Q(\theta, \theta_m) - Q(\theta_m, \theta_m)$$

Or, $Q(\theta_{m+1}, \theta_m) \geq Q(\theta_m, \theta_m)$ car θ_{m+1} maximise cette fonction. Donc

$$l(\theta) - l(\theta_m) \geq 0$$

C Bibliographie

- Daniel D. Lee, H. Sebastian Seung *Learning the parts of objects by non-negative matrix factorization*, Nature 401, October 1999
- Daniel D. Lee, H. Sebastian Seung *Algorithms for Non-negative Matrix Factorization*
- Toby Segaran *Collective intelligence*, O'Reilly.
- Maya R. Gupta, Yihua Chen, *Theory and use of the EM algorithm*, Now publisher
- Thomas Hofmann, *Unsupervised Learning by Probabilistic Latent Semantic Analysis*, Machine Learning (Boston, MA : Kluwer Academic Publishers)