TP1 – Espaces de représentation des couleurs

Exercice 1 : corrélations et contrastes des canaux RVB

Le script `exercice_1.m` lit l'image `autumn.tif` codée en RVB (rouge, vert, bleu) et la stocke dans une matrice tridimensionnelle `I` : les valeurs entières `I(i,j,1)`, `I(i,j,2)` et `I(i,j,3)`, comprises entre 0 et 255 (8 bits), sont les *niveaux de couleur* du pixel situé sur la ligne `i` et la colonne `j`, dans les canaux R, V et B. La matrice `I` est scindée en trois matrices bidimensionnelles `R`, `V` et `B` correspondant aux trois canaux. Les matrices `I`, `R`, `V`, `B` sont affichées sous forme d'images. À part pour les régions très colorées (par exemple, l'arbre situé au centre de l'image « disparaît » dans le canal bleu), vous observez une forte corrélation entre les trois canaux.

Dans une deuxième fenêtre, les pixels sont considérés comme des points de $\mathbb{R}^3$, dans un repère dont les axes correspondent aux trois niveaux de couleur. Ils forment un nuage de forme très allongée, ce qui confirme l'observation précédente, à savoir que les trois canaux sont fortement corrélés.

Complétez ce script, de manière à calculer la matrice `Sigma` de variance/covariance des variables aléatoires correspondant aux trois canaux (matrice de taille 3×3). **Attention** : n'oubliez pas de centrer la matrice `X` des données ; n'utilisez pas les fonctions `var` et `covar` de Matlab, qui appliquent des prétraitements aux données.

Le *coefficient de corrélation linéaire* $r_{YZ} \in [-1, 1]$ entre les canaux Y et Z s'écrit $r_{YZ} = \frac{\sigma_{YZ}}{\sigma_Y \sigma_Z}$, où σ_Y désigne l'écart-type de Y, σ_Z l'écart-type de Z et σ_{YZ} la covariance de Y et Z. D'autre part, la *proportion de contraste* dans le canal Y s'écrit $c_Y = \sigma_Y^2/(\sigma_R^2 + \sigma_V^2 + \sigma_B^2)$, où σ_Y^2 désigne la variance de Y. Terminez l'écriture du script `exercice_1.m` en calculant les coefficients de corrélation linéaire et les proportions de contraste de cette image (utilisez la fonction `fprintf` pour l'affichage). **Attention** : ne confondez pas écart-type et variance.

Transmission d'une image couleur par un seul canal

Le choix d'un espace de représentation des couleurs s'est posé lorsque les chaînes de télévision sont passées à la couleur, dans les années 1960. En effet, il était inutile de transmettre trois canaux R, V et B aux utilisateurs (encore nombreux) possédant des téléviseurs noir et blanc, qui ne pouvaient afficher qu'un seul canal. Le principal critère était de maximiser la proportion de contraste de ce canal unique. Suffisait-il de transmettre un des trois canaux R, V ou B, choisi arbitrairement ? Lancez le script `exercice_1.m` sur l'image `gantrycrane.png`, qui est une image à dominante bleue. Déduisez-en pourquoi cette idée n'aurait pas été acceptable.

Or, la conversion d'une image couleur en une image en niveaux de gris consiste en une réduction de dimension, et l'analyse en composantes principales (ACP) est une technique très générale de réduction de dimension.

Exercice 2 : analyse en composantes principales

Effectuez une copie du script `exercice_1.m`, de nom `exercice_2.m`, que vous modifierez de manière à effectuer l'ACP des données contenues dans la matrice `X`.

La matrice `Sigma` de variance/covariance est symétrique et réelle. Elle admet donc une base orthonormée de vecteurs propres. Calculez ses valeurs et vecteurs propres à l'aide de l'appel : `[W,D] = eig(Sigma)`. Les valeurs propres de `Sigma` sont stockées sur la diagonale de la matrice `D`. Triez ces valeurs par ordre décroissant, à l'aide des fonctions `diag` et `sort` de Matlab (avec l'option `'descend'`). Les vecteurs propres de `Sigma`, appelés *vecteurs principaux* dans le cas de l'ACP, sont stockés sur les trois colonnes de la matrice `W`. La matrice `W` est donc orthogonale (son inverse est égale à sa transposée) et constitue la matrice de passage entre le repère RVB et le nouveau repère ayant pour axes les *axes principaux*. Calculez la matrice `C` des coordonnées des pixels dans ce nouveau repère, appelées *composantes principales* dans le cas de l'ACP. **Attention** : n'oubliez pas d'appliquer aux colonnes de `W` la permutation qui a permis de trier les valeurs propres de `Sigma` par ordre décroissant.

Affichez les trois colonnes de la matrice `C` sous forme d'images (avec les fonctions `reshape` et `size` de Matlab). Calculez les coefficients de corrélation linéaire et les proportions de contraste dans le nouveau repère. Calculez la proportion de contraste de la première composante principale. Commentez.

La fonction `eig` de Matlab ne garantit rien sur le signe des vecteurs propres. Comme ce signe est aléatoire (mais constant, pour une même session Matlab), il y a une chance sur deux pour que le contraste de la première composante principale soit inversé (ciel noir). Ce défaut imprévisible sera corrigé dans l'exercice 3.

Exercice 3 : combinaisons linéaires des trois canaux RVB

Si le critère retenu est celui du contraste, l'ACP est le meilleur moyen pour convertir une image couleur en une image en niveaux de gris. Cependant, la matrice de passage dépend de l'image considérée. Or, il n'était pas envisageable, dans les années 1960, d'effectuer une ACP pour chaque image d'une séquence télédiffusée. C'est pourquoi une matrice de passage commune à toutes les images couleur a été choisie.

Que ce soit pour `autumn.tif` ou pour `pears.png`, vérifiez que les coordonnées du vecteur propre de `Sigma` correspondant à sa plus grande valeur propre sont toutes trois très proches de $1/\sqrt{3} \approx 0,5774$. Cela montre-t-il que la maximisation de la proportion de contraste, c'est-à-dire de la variance, qui caractérise la première composante principale (cf. cours), s'obtient en donnant le même poids aux trois canaux ? Bien que cette affirmation soit erronée lorsque la proportion de contraste diffère beaucoup d'un canal à l'autre (faites le test sur l'image `gantrycrane.png`), elle est vraie *en moyenne*. Or, donner le même poids aux trois canaux revient à transformer une image couleur en une image en niveaux de gris de la manière la plus intuitive qui soit, à savoir :

$$I_{\text{nvg}} = \frac{1}{3}\,(R + V + B) \tag{1}$$

Pourtant, la fonction `rgb2gray` de Matlab, qui a été spécifiquement conçue pour transformer une image RVB en une image en niveaux de gris, effectue une autre combinaison linéaire (cf. `help rgb2gray`) :

$$Y = 0,2989\,R + 0,5870\,V + 0,1140\,B \tag{2}$$

qui disymétrise les canaux R, V et B, contrairement à (1). Cela vient de ce que le système visuel humain n'est pas également sensible aux différentes fréquences lumineuses. En particulier, comme sa courbe de sensibilité est maximale dans le vert, le poids du canal V dans (2) est supérieur aux poids des deux autres canaux.

Effectuez une copie du script `exercice_2.m`, de nom `exercice_3.m`, que vous modifierez de manière à afficher sur une même figure les quatre images suivantes : l'image couleur d'origine, sa première composante principale, et les deux images correspondant aux combinaisons linéaires (1) et (2). **Attention** : pour garantir que le contraste de la première composante principale ne soit pas inversé, il suffit de calculer son coefficient de corrélation linéaire avec la moyenne (1) des trois canaux. Ce coefficient doit toujours être très proche de 1 en valeur absolue, mais son signe permet de déceler une éventuelle inversion de contraste.

Parmi les trois images en niveaux de gris affichées par le script `exercice_3.m`, laquelle vous semble-t-elle présenter le meilleur contraste ? Testez différentes images internes de Matlab (`autumn.tif`, `pears.png`, etc.). La liste des images internes de Matlab s'obtient en tapant : `dir(fileparts(which('autumn.tif')))`.

Espace de représentation des couleurs YCbCr

L'expression (2) n'est autre que la première des trois formules de passage de l'espace RVB à un autre espace de représentation des couleurs appelé YCbCr : Y est la *luminance*, Cb la *chrominance bleue* et Cr la *chrominance rouge*. Les deux autres formules de passage s'écrivent comme suit (fonction `rgb2ycbcr` de Matlab) :

$$\begin{aligned} C_b &= -0,1687\,R - 0,3313\,V + 0,5\,B + 128 \\ C_r &= 0,5\,R - 0,4187\,V - 0,0813\,B + 128 \end{aligned} \tag{3}$$

L'ajout de 128 à C_b et C_r force ces valeurs à être comprises entre 0 et 255.

Il est notable que l'espace de représentation des couleurs YCbCr n'est pas orthonormé : en utilisant les formules (2) et (3), vérifiez que la matrice de passage n'est pas orthogonale. L'espace YCbCr est néanmoins très connu, car il est utilisé par le format JPEG. En effet, comme le système visuel humain est plus sensible à la luminance Y qu'aux chrominances Cb et Cr, il est possible de réduire la taille d'une image en dégradant ses deux chrominances, sans que cela altère trop sa qualité.