TP3 – Classification bayésienne

Rappels de cours

La segmentation d'une image en niveaux de gris $\mathbf{x} = (x_s)_{s\in\mathcal{S}}$ peut être effectuée par *classification*. En choisissant un nombre N de classes, supposées gaussiennes, et en supposant connues les moyennes $\mu_1, \ldots, \mu_N$ et les écarts-types $\sigma_1, \ldots, \sigma_N$ des différentes classes, le résultat est la configuration $\widehat{\mathbf{k}} = (\widehat{k}_s)_{s\in\mathcal{S}}$ qui maximise la *probabilité a posteriori* de la configuration $\mathbf{k} = (k_s)_{s\in\mathcal{S}}$, sachant $\mathbf{x}$. Or, d'après le théorème de Bayes :

$$p(\mathbf{K}=\mathbf{k}|\mathbf{X}=\mathbf{x}) = \frac{p(\mathbf{X}=\mathbf{x}|\mathbf{K}=\mathbf{k})\,p(\mathbf{K}=\mathbf{k})}{p(\mathbf{X}=\mathbf{x})} \;\propto\; p(\mathbf{X}=\mathbf{x}|\mathbf{K}=\mathbf{k})\,p(\mathbf{K}=\mathbf{k}) \tag{1}$$

L'hypothèse d'indépendance des données permet d'écrire la *vraisemblance* sous la forme d'un produit :

$$p(\mathbf{X}=\mathbf{x}|\mathbf{K}=\mathbf{k}) = \prod_{s\in\mathcal{S}} p(X_s=x_s|K_s=k_s) = \prod_{s\in\mathcal{S}} \frac{1}{\sigma_{k_s}\sqrt{2\pi}}\,\exp\left\{-\frac{(x_s-\mu_{k_s})^2}{2\,\sigma_{k_s}^2}\right\} \tag{2}$$

Quant à la *probabilité a priori* de la configuration $\mathbf{k}$, elle est donnée par le *modèle de Potts* :

$$p(\mathbf{K}=\mathbf{k}) \;\propto\; \exp\left\{-\beta \sum_{\{s,t\}\in\mathcal{C}_2} [1-\delta(k_s,k_t)]\right\} \tag{3}$$

où $\mathcal{C}_2$ contient les paires $\{s,t\}$ de pixels voisins (« 8 plus proches voisins »). De (1), (2) et (3), il vient :

$$p(\mathbf{K}=\mathbf{k}|\mathbf{X}=\mathbf{x}) \;\propto\; \exp\left\{-\frac{1}{2}\sum_{s\in\mathcal{S}}\left[\ln\sigma_{k_s}^2 + \frac{(x_s-\mu_{k_s})^2}{\sigma_{k_s}^2}\right] - \beta\sum_{\{s,t\}\in\mathcal{C}_2}[1-\delta(k_s,k_t)]\right\} \tag{4}$$

Chercher le maximum de $p(\mathbf{k}|\mathbf{x})$ équivaut à chercher le minimum de l'*énergie* $U(\mathbf{k})$:

$$U(\mathbf{k}) = \frac{1}{2}\sum_{s\in\mathcal{S}}\left[\ln\sigma_{k_s}^2 + \frac{(x_s-\mu_{k_s})^2}{\sigma_{k_s}^2}\right] + \beta\sum_{\{s,t\}\in\mathcal{C}_2}[1-\delta(k_s,k_t)] \tag{5}$$

Étant donné qu'il est impossible, en pratique, de tester les $N^{\mathrm{card}(\mathcal{S})}$ configurations $\mathbf{k}$, il faut recourir à une *méta-heuristique*, en l'occurrence le *recuit simulé*, qui fait décroître un paramètre T appelé *température*, à chaque itération. L'algorithme complet s'écrit :

1. **Initialisations :** $T \leftarrow T_0$; $\mathbf{k} \leftarrow$ Configuration obtenue par maximisation de la vraisemblance.
2. **Parcours de tous les pixels s de l'image, visitée ligne par ligne et colonne par colonne :**
   - Tirer une valeur $k'_s \in E\backslash\{k_s\}$, où $E = \{1, \ldots, N\}$, et calculer les deux énergies locales suivantes :

$$\begin{cases} U_s = \dfrac{1}{2}\left[\ln\sigma_{k_s}^2 + \dfrac{(x_s-\mu_{k_s})^2}{\sigma_{k_s}^2}\right] + \beta \displaystyle\sum_{t\,\in\,\mathcal{V}(s)}[1-\delta(k_s,k_t)] \\ U'_s = \dfrac{1}{2}\left[\ln\sigma_{k'_s}^2 + \dfrac{(x_s-\mu_{k'_s})^2}{\sigma_{k'_s}^2}\right] + \beta \displaystyle\sum_{t\,\in\,\mathcal{V}(s)}[1-\delta(k'_s,k_t)] \end{cases} \tag{6}$$

   où $\mathcal{V}(s)$ désigne l'ensemble des pixels voisins de s.
   - Si $U'_s < U_s$, remplacer k_s par k'_s. Sinon, faire de même, mais avec une probabilité $\exp\left\{-\frac{U'_s-U_s}{T}\right\}$ qui décroît avec la température T. Une particularité du recuit simulé est donc de ne pas systématiquement rejeter les changements de configuration qui font croître l'énergie.
3. **Mises à jour :** $T \leftarrow \alpha\,T$, puis retour en 2, tant que le nombre maximal d'itérations n'est pas atteint.

Exercice 1 : segmentation par classification supervisée

Écrivez les fonctions `estimation_loi_normale`, `attache_aux_donnees` et `recuit_simule`, appelées par `exercice_1` :

- La fonction `estimation_loi_normale` permet d'estimer la moyenne et la variance de chaque classe à partir d'un échantillon sélectionné par l'utilisateur, d'où le caractère *supervisé* de la classification.
- La fonction `attache_aux_donnees` doit retourner une matrice tridimensionnelle contenant, pour chaque pixel s, la valeur de l'attache aux données $\dfrac{1}{2}\left[\ln \sigma_{k_s}^2 + \dfrac{(x_s - \mu_{k_s})^2}{\sigma_{k_s}^2}\right]$, relativement à chacune des N classes.
- La fonction `recuit_simule` doit effectuer autant d'itérations de l'algorithme ci-dessus qu'il y a de pixels dans l'image. Utilisez la fonction `randi` de Matlab pour tirer aléatoirement la nouvelle classe d'un pixel, et l'opérateur `~=` (« différent de ») pour calculer le terme de régularisation des expressions (6).

Observez ce qui se passe dans les cas suivants : si le nombre N de classes est différent de 4 ; lorsque les échantillons sont mal sélectionnés ; si $T_0 = 0$ (dans ce cas, on force l'énergie à décroître à chaque itération).

Classification non supervisée

Pour éviter à l'utilisateur de sélectionner à la main un échantillon de chaque classe, il est envisageable d'estimer les paramètres des N classes, en cherchant un mélange de N gaussiennes coïncidant avec l'histogramme $f(x)$ de l'image en niveaux de gris :

$$f(x) = \sum_{i=1}^{N} \frac{p_i}{\sigma_i\sqrt{2\pi}} \exp\left\{-\frac{(x-\mu_i)^2}{2\,\sigma_i^2}\right\}, \qquad x \in [0, 255] \tag{7}$$

où μ_i, σ_i et p_i désignent, respectivement, la moyenne, l'écart-type et le poids de la $i^{\text{ème}}$ gaussienne. L'estimation des paramètres de ce modèle revient donc à résoudre un problème en moindres carrés linéaire vis-à-vis de p_i, mais non linéaire vis-à-vis de μ_i et σ_i :

$$(\widehat{\mu_i}, \widehat{\sigma_i}, \widehat{p_i})_{i\in E} = \underset{(\mu_i,\sigma_i,p_i)_{i\in E}}{\arg\min} \sum_{x=0}^{255} \left[f(x) - \sum_{i=1}^{N} \frac{p_i}{\sigma_i\sqrt{2\pi}} \exp\left\{-\frac{(x-\mu_i)^2}{2\,\sigma_i^2}\right\}\right]^2 \tag{8}$$

Exercice 2 : segmentation par classification non supervisée

Lisez le script `exercice_2`, dans lequel l'estimation des paramètres μ_i et σ_i est effectuée en minimisant l'argument de (8) par tirages aléatoires : les moyennes μ_i sont recherchées dans l'intervalle $[0, 255]$, mais les écarts-types σ_i sont recherchés dans l'intervalle $[10, 25]$ afin d'accélérer la résolution. Quant à l'estimation des poids p_i, elle est facilitée par le fait que le problème en moindres carrés (8) est linéaire en p_i. Pour estimer les poids, à chaque tirage aléatoire de $2N$ valeurs réelles (μ_i, σ_i), $i \in \{1, \ldots, N\}$, il faut donc résoudre un système linéaire du type $\mathbf{A}\,\mathbf{P} = \mathbf{F}$, où $\mathbf{P} = [p_1, \ldots, p_N]^\top$ et où $\mathbf{F}$ contient les 256 valeurs de l'histogramme.

Écrivez la fonction `estimation_poids`, appelée par le script `exercice_2`, permettant de résoudre la partie linéaire du problème (8), c'est-à-dire relativement aux poids $(p_i)_{i\in E}$.

Bien que beaucoup plus lente, à cause de l'estimation des paramètres par tirages aléatoires, cette méthode doit vous permettre d'atteindre un pourcentage de bonnes classifications comparable à celui de l'exercice 1, et ce de manière entièrement non supervisée !

Exercice 3 : segmentation par partitionnement (facultatif)

Écrivez un script, de nom `exercice_3`, permettant de segmenter l'image de synthèse précédente en vous inspirant d'une des méthodes de partitionnement (*clustering*) vues en 1A : choisissez comme caractéristiques d'un pixel son niveau de gris et sa position dans l'image, puis étendez cette méthode à des images couleur.