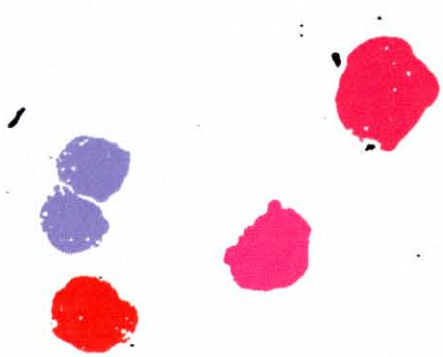
Angélique SEGRETIN

IUP Génie Informatique

Rapport de stage
de 3ème année d'IUP Génie Informatique



Sujet :

Caractérisation de types hémocytaires dans l'hémolymphe de l'huître plate, *Ostrea edulis*, par classification robuste.

Stage effectué à la station IFREMER de La Tremblade (17), au laboratoire de Génétique, Aquaculture et Pathologie de Ronce-Les-Bains, sous la direction de Tristan RENAULT.

Annee universitaire 96-97

Table des matières

INTRODUCTION

L'ostréiculture s'est développée réellement en France au milieu du siècle dernier après que des essais de captage de l'huître plate, *Ostrea edulis*, plus communément appelée Belon, soient effectuées par Coste (1861).

Contrairement à l'huître japonaise, *Crassostrea gigas*, qui constitue l'essentiel de la production conchylicole française actuelle (150 000 tonnes/an), la production annuelle d'*Ostrea edulis* a diminué considérablement entre les années 70 et 90, passant de 20 000 à 1 500 tonnes. Cette diminution est essentiellement due à l'apparition de deux pathologies qui sont spécifiques de cette espèce : la bonamiose et la marteiliose, deux maladies parasitaires.

L'action de ce type de bioagresseurs (parasites, mais aussi champignons, virus et bactéries) peut être spontanée ou induite par les pratiques zootechniques particulières. Les concentrations animales rencontrées en élevage peuvent également exacerber les effets des agents pathogènes.

En fait, il n'y a que peu de moyens de protéger les invertébrés marins d'intérêt aquacole vis à vis des maladies infectieuses du fait de certaines caractéristiques de ces animaux. D'une part, les traitements pour des espèces le plus généralement élevées en milieu ouvert, posent des problème de quantité de substance à utiliser, des fortes probabilités de recontamination et de l'accumulation de résidus dans le milieu. De ce fait, ce type d'approche pour contrôler les maladies infectieuses ne semble pas une voie à privilégier.

D'autre part, la vaccination reste sans objet chez les invertébrés du fait de leur absence de réponse immune spécifique.

Au vu de ces éléments, les seules façons de protéger efficacement les invertébrés marins d'intérêt économique vis à vis des maladies infectieuses sont de mettre au point des techniques de diagnostic rapides, sensibles et fiables, ainsi que d'obtenir des populations d'animaux présentant une certaine résistance aux principales maladies.

Dans ce cadre, les chercheurs de l'IFREMER ont entrepris depuis plusieurs années de sélectionner des animaux pour leur résistance à la bonamiose. Des huîtres sélectionnées présentant des caractères accrus de résistance à cette maladie ont ainsi été obtenues. Les

chercheurs du Laboratoire de Génétique Aquaculture et Pathologie de La Tremblade, en comparant ces individus et des animaux témoins ont mis en évidence le rôle probable des cellules de l'hémolymphe dans le processus de résistance vis à vis de l'infection parasitaire.

En effet, les pourcentages des types cellulaires présents dans l'hémolymphe montrent des différences marquées en fonction de l'état de résistance des huîtres analysées. Cependant, cette approche reste lourde à mener, dans la mesure où les examens sont effectués en microscopie photonique par un observateur et que le classement des types cellulaires observés peut être difficile.

Ainsi trois types cellulaires ont été caractérisés dans l'hémolymphe de l'huître plate, mais la difficulté d'établir un classement dans ces trois catégories et le fait qu'il peut différer selon la personne interrogée pour certaines cellules, pourraient laisser croire que ces trois types d'hémocytes ne sont pas exhaustifs.

Une première phase de prélèvement et de préparation a été nécessaire pour obtenir le matériel biologique : les cellules cytocentrifugées et colorées.

Pour caractériser les types hémocytaires, divers "variables" morphométriques ont été mesurées par analyse d'image de cellules à l'aide du logiciel IPS-SAMBA, en utilisant un microscope OLYMPUS BH2 et une caméra CCD Sony.

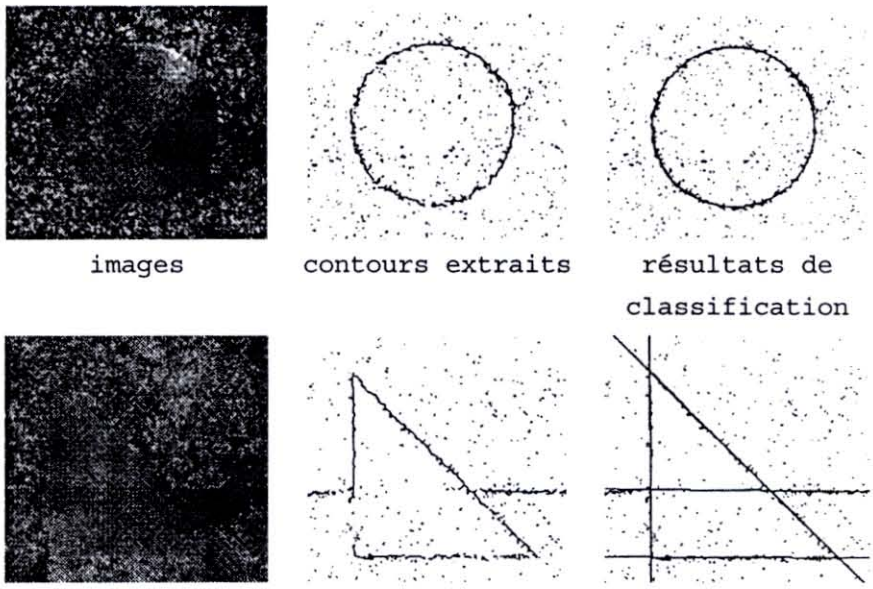
Différents algorithmes ont été utilisés pour tenter de classifier les cellules d'hémolymphe d'huître plate à partir de paramètres morphométriques sélectionnés.

2. - Rappels sur la classification
Des algorithmes de classification floue

Classifier, c'est analyser un ensemble d'objets dans un but de structuration (en classes) et/ou de représentation (prototypes).

La classification est souvent utilisée en vision par ordinateur, que ce soit :

- en traitement d'image (segmentation en régions ou contours)



- ou en analyse d'image (caractérisation d'objets dans une scène)

Dans notre étude, il s'agira de rechercher, par analyse de paramètres morphométriques issus d'images, différents types de cellules que l'on rencontre dans l'hémolymphe de l'huître plate. On s'intéressera autant au problème de struturation qu'à celui de représentation.

Le principe fondamental de toute méthode de classification est que des objets se ressemblant doivent appartenir à la même classe et des objets ne se ressemblant pas doivent appartenir à des classes différentes. La notion de ressemblance peut être fondée sur une relation ou plus souvent sur une métrique lorsque les objets sont des points de $\Re^p$, c'est-à-

dire des objets sur lesquels on a mesuré p paramètres. C'est dans ce cadre que nous aborderons ce problème.

2.1 - Les familles de méthodes de classification

Soit $\Omega = \{X_1, X_2, \ldots, X_N$, un ensemble de N points à classifier. Il existe deux grandes familles de méthodes de classification avec approche métrique :

- **La classification hiérarchique** ascendante (par agglomérations successives des points) ou descendante (par divisions successives de l'ensemble Ω), dont le résultat est une hiérarchie, chaque niveau de la hiérarchie correspondant à une partition stricte de Ω (voir définition ci-après). Lorsqu'on veut structurer en classes, il reste le problème du choix de la partition.

- **La classification par partition stricte** dont le résultat est un ensemble $P = \{\omega_1, \omega_2, \ldots, \omega_c$ de parties non vides de Ω tel que $\forall i \neq j, \omega_i \cap \omega_j = \varnothing$ et $\bigcup_{i=1}^{c} \omega_i = \Omega$. Le nombre c de classes doit être pré-défini. On présente souvent le résultat sous la forme d'une matrice de partition U de dimension $(c \times N)$ dont le terme général U_{ik} représente l'appartenance ou non du point X_k à la classe ω_i. On imposera les contraintes suivantes :

$$U_{ik} \in \{0,1 \quad \forall k = 1, N \quad \forall i = 1, c \qquad (1a)$$

$$\sum_{i=1}^{c} U_{ik} = 1 \qquad \forall k = 1, N \qquad (1b)$$

$$0 < \sum_{k=1}^{N} U_{ik} < N \qquad \forall i = 1, c \qquad (1c)$$

La contrainte (1a) traduit l'appartenance binaire des points aux classes, c'est-à-dire qu'un point appartient ou n'appartient pas à une classe. L'équation (1b) impose que l'appartenance soit totale et stricte, c'est-à-dire que chaque point appartient à une classe et une

seule. Enfin, la contrainte (1c) assure qu'aucune classe n'est vide et que le nombre de classes est plus grand que 1.

Les algorithmes de classification par partition procèdent itérativement à la recherche d'une matrice de partition stable (structuration) autour de prototypes (représentation) en optimisant un critère de type moindres carrés. Un cas particulier, l'algorithme des centres mobiles, et des exemples de classification sont donnés en Annexe 1.2.

Le choix d'établir une partition stricte s'avère souvent limitatif. Il est difficile de déterminer à quelle classe on doit associer un point pouvant appartenir à plusieurs classes (Annexe 1.2.2 - Partition stricte - Application sur le papillon). Il est aussi difficile de partitionner de manière stricte des classes de points qui manifestement ne se séparent pas nettement. Il est possible de pallier ces problèmes en permettant que l'appartenance soit non plus binaire mais graduelle. C'est le but de la **classification par partition floue**. Le résultat peut là encore être une matrice de partition U. Les contraintes (1b) et (1c) restent valides mais la contrainte (1a) est relachée en la contrainte (1'a) traduisant l'appartenance graduelle :

$$U_{ik} \in [0,1] \quad \forall k = 1, N \quad \forall i = 1, c \qquad (1'a)$$

Lorsqu'une classe ω_i est représentée par un point-prototype (par exemple, son centre de gravité V_i), il est commun de traduire l'appartenance U_{ik} d'un point X_k à ω_i par une fonction monotone décroissante de la distance $d(X_k, V_i$ de X_k au prototype V_i. La distance étant une mesure de dissemblance, une mesure de ressemblance traduira alors la notion d'appartenance. Ainsi, plus U_{ik} tendra vers 1 (respectivement 0), plus le point X_k appartiendra (respectivement n'appartiendra pas) à la classe ω_i.

Quelque soit le type de partition cherchée (stricte ou floue), le choix de la métrique définissant ces distances est crucial, car celle-ci influe sur la forme des classes. Par exemple, la métrique identité induisant la distance euclidienne contraint la recherche de classes de même forme hypersphérique.

En ce qui nous concerne, l'hémolymphe véhicule des cellules se situant à différentes phases de développement. Il parait donc évident que les classes cherchées (types hémocytaires) sont continues; d'où la nécessité d'employer des algorithmes de classification par partition floue. Il n'y a de plus aucune raison pour que ces classes soient des hypersphères.

2.2 - Les algorithmes de classification floue

2.2.1 - Les Fuzzy C-Means (FCM)

Le plus simple des algorithmes de classification floue, celui des Fuzzy C-Means (C-moyennes floues), est dû à J.C. Bezdek [1]. Le critère minimisé est :

$$J_m(U,V) = \sum_{i=1}^{c} \sum_{k=1}^{N} U_{ik}{}^m \, d_{ik}{}^2 \qquad (2)$$

où U est une matrice de partition floue vérifiant les contraintes (1'a), (1b) et (1c)

V est l'ensemble des centres V_i des classes ω_i $(i = 1, c)$

$m \in]1,+\infty[$ est un paramètre contrôlant le degré de flou de la partition

d_{ik} est la distance euclidienne entre X_k et V_i, définie dans $\Re^p$ par :

$$d_{ik}{}^2 = (X_k - V_i)^t \, (X_k - V_i \qquad \forall i = 1, c \qquad (3)$$

On peut montrer la convergence de l'algorithme itératif donné ci-après vers un minimum local de $J_m(U,V)$.

Etape 0 :

- fixer le nombre c de classes
- fixer le paramètre de flou m
- initialiser aléatoirement la matrice de partition U vérifiant (1'a), (1b) et (1c).

Etape 1 :

- calculer, à l'itération l, les centres V_i^l :

$$V_i^l = \frac{\sum_{k=1}^{N} U_{ik}^m X_k}{\sum_{k=1}^{N} U_{ik}^m} \qquad \forall i = 1, c \qquad (4)$$

Etape 2 :

- mettre à jour la matrice de partition U^l :

$$U_{ik}^l = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik}^2}{d_{jk}^2} \right)^{1/m-1}} \qquad \forall i = 1, c \quad \forall k = 1, N \qquad (5)$$

Etape 3 :

- estimer la stabilité de U^l en utilisant par exemple

  une norme matricielle :

  si $\left\| U^l - U^{l-1} \right\| \leq \varepsilon$ stop,

  sinon incrémenter l et reprendre à l'étape 1.

On obtient en fait une famille d'algorithmes selon la valeur de *m*. Plus *m* est grand, plus la partition est floue, c'est à dire que les valeurs U_{ik} s'éloignent de 0 et de 1 (les points ont tendance à appartenir à toutes les classes). Inversement, plus *m* tend vers 1, plus la partition est stricte, c'est à dire que les U_{ik} sont généralement proches de 0 ou de 1 (les points ont tendance à n'appartenir qu'à une seule classe). Il n'existe pas de critère permettant de choisir *m* de manière optimale, la plupart des auteurs préconisent *m*=2 .

Des exemples d'application montrant l'intérêt et les limites de cet algorithme sont donnés en Annexe 2. Les limites sont les suivantes :

1 - l'algorithme ainsi défini n'est pas du tout adapté pour détecter des classes non sphériques, car le calcul de distance ne s'y prête pas (annexe 2.1.2 - Fuzzy C-Means sur la croix de Gustafson : les points ne sont pas regroupés suivant la branche de la croix à laquelle ils appartiennent).

2 - les informations d'appartenance obtenues avec cet algorithme ne correspondent pas à ce qu'on attend intuitivement, notamment lorsque du "bruit" vient s'ajouter aux données. Il est logique qu'un point situé sur l'hyperplan séparateur de deux classes appartienne autant à l'une qu'à l'autre sauf s'il est loin des classes (annexe 2.1.3 - Fuzzy C-Means sur deux ensembles bruités). Un tel point, considéré comme du bruit, devrait appartenir aussi peu à une classe qu'à l'autre et non autant.

2.2.2 - Les Fuzzy C-Covariances (FCC)

E.E. Gustafson et W.C. Kessel (voir [1]), ont modifié l'algorithme des FCM afin de pallier le premier problème en changeant la façon de calculer la distance entre deux points (la métrique). L'idée est d'adapter la métrique selon chacune des classes. L'algorithme, qui cherche à détecter des classes de forme hyperellipsoïdale, est le même que l'algorithme FCM excepté que la distance d_{ik} (3) est :

$$d_{ik}{}^2 = (X_k - V_i)^t \, A_i (X_k - V_i \qquad \forall i = 1, c \qquad (6)$$

où les métriques A_i peuvent être obtenues sous contrainte de volume des classes ω_i en fixant par exemple $\det(A_i) = \rho_i > 0$. A volume fixé la métrique optimale est donnée par :

$$A_i = \left[\rho_i \det(S_i)\right]^{1/p} (S_i)^{-1} \qquad \forall i = 1, c \qquad (7)$$

où S_i représente la matrice de dispersion floue de la classe ω_i définie par :

$$S_i = \sum_{k=1}^{N} (U_{ik})^m (X_k - V_i)(X_k - V_i)^t \qquad \forall i = 1, c \qquad (8)$$

Les paramètres ρ_i permettent de fixer le rapport entre les volumes des classes détectées ω_i.

Etape 0 :

- fixer le nombre c de classes
- fixer le paramètre de flou m

- initialiser aléatoirement la matrice de partition U
  vérifiant (1'a), (1b) et (1c).
- fixer les contraintes des volume ρ_i

Etape 1 :

- calculer, à l'itération l, les centres V_i^l (4)

Etape 2 :

- mettre à jour la matrice de partition U^l (5)

  avec (6), (7) et (8).

Etape 3 :

- estimer la stabilité de U^l en utilisant par exemple

  une norme matricielle :

  si $\left\|U^l - U^{l-1}\right\| \leq \varepsilon$ stop,

  sinon incrémenter l et reprendre à l'étape 1.

Cet algorithme permet de détecter des classes de formes différentes ; des exemples sont donnés en annexe (Annexe 2.2.1 - Fuzzy C-Covariance sur la croix de Gustafson). Il faut cependant retenir que le choix des contraintes ρ_i laissé à l'utilisateur influe beaucoup sur les résultats. En l'absence d'idée précise, on choisit en général $\rho_i = \rho = 1$ ($\forall i = 1, c$).

Cependant, le second problème exposé pour l'algorithme précédent n'est toujours pas résolu, bien au contraire. Dans un cas comme celui des deux ensembles bruités, le fait de contraindre la métrique change totalement le résultat de la classification. En effet, les points de bruit sont intégrés aux classes et donc perturbent fortement leur forme (voir Annexe 2.2.2 - Fuzzy C-Covariance sur des ensembles bruités).

2.3 - Des algorithmes de classification possibiliste

Afin de nuancer l'appartenance d'un point à une classe et donc de gérer le "bruit" Krishnapuram et Keller [2] ont introduit la notion de classification possibiliste. Elle est fondée sur le relâchement de la contrainte d'appartenance totale (1b). Il suffit de la remplacer par :

$$0 \leq \sum_{i=1}^{c} U_{ik} \leq 1 \quad \forall k = 1, N \qquad (1'b)$$

Ainsi lorsque $U_j = \max_i(U_{ik})$ est inférieur à un seuil U_0 (par exemple $U_0 = \frac{1}{c}$), on peut considérer que le point X_k n'appartient pas à la classe ω_j dont il est le plus proche, et a fortiori à aucune autre. Un tel point est alors assimilé à un point de bruit.

L'algorithme des Possibilistic C-Means (C-moyennes floues) minimise le critère suivant :

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{k=1}^{N} {U_{ik}}^m \, {d_{ik}}^2 + \sum_{i=1}^{c} \eta_i \sum_{k=1}^{N} (1 - U_{ik})^m \qquad (9)$$

où U est une matrice de partittion possibiliste vérifiant (1'a), (1'b) et (1c)

V est l'ensemble des centres V_i des classes ω_i (i=1,c)

$m \in]1,+\infty[$ est un paramètre contrôlant le degré de flou de la partition

d_{ik} est la distance euclidienne entre X_k et V_i

$\eta_i > 0$ sont des paramètres de résolution

L'optimum du critère (9) est atteint pour des centres V_i vérifiant (4) et pour la matrice de partition définie comme suit :

$$U_{ik} = \frac{1}{1 + \left(\frac{d_{ik}^2}{\eta_i} \right)^{\frac{1}{m-1}}} \quad \forall i = 1, c \quad \forall k = 1, N \qquad (10)$$

où d_{ik} est définie par (3).

Les classes ainsi détectées sont, comme pour l'algorithme FCM, de forme hypersphériques. L'algorithme est :

Etape 0 :

- fixer le nombre c de classes
- fixer le paramètre de flou m
- initialiser aléatoirement la matrice de partition U vérifiant (1'a), (1'b) et (1c).
- initialiser les paramètres de résolution $\eta_i > 0$, par exemple

$$\eta_i = \sum_{k=1}^{N} {U_{ik}}^m \, {d_{ik}}^2 \Big/ \sum_{k=1}^{N} {U_{ik}}^m \qquad (11)$$

Etape 1 :

- calculer, à l'itération l, les centres V_i^l (4)

Etape 2 :

- mettre à jour la matrice de partition U^l (10)

Etape 3 :

- estimer la stabilité de U^l en utilisant par exemple

  une norme matricielle :

  si $\left\| U^l - U^{l-1} \right\| \leq \varepsilon$ stop,

  sinon incrémenter l et reprendre à l'étape 1.

Une version similaire à l'algorithme des FCC peut être utilisée, en définissant :

$$U_{ik} = \frac{1}{1 + d_{ik}^{\frac{2}{m-1}}} \qquad \forall i = 1, c \quad \forall k = 1, N \qquad (12)$$

où d_{ik} est définie par :

$$d_{ik}^2 = |F_i|^{1/p} \left(X_k - V_i\right)^t F_i^{-1} \left(X_k - V_i \right. \qquad \forall i = 1, c \quad \forall k = 1, N \qquad (13)$$

avec F_i, matrice de covariance floue de la classe ω_i, définie par :

$$F_i = \frac{S_i}{\sum_{k=1}^{N} U_{ik}^m} \qquad \forall i = 1, c \qquad (14)$$

Nous appelerons cet algorithme PCC (Possibilistic C-Covariances). Il présente l'avantage de chercher des classes de formes ellipsoïdales différentes d'une part, et de faire abstraction de la détermination des paramètres de résolution η_i.

Des exemples de classification sont donnés en annexe (Annexe 2.3 - PCM sur des ensembles bruités, Annexe 2.4 - PCC sur des ensembles bruités).

2.4 - Des algorithmes avec une classe de bruit

Si les algorithmes précédents permettent de traiter le "bruit" présent dans l'ensemble à classifier, il convient de noter que celui-ci est modélisé de manière intrinsèque. Une autre approche consiste à considérer le "bruit" comme une $(c+1)^{ème}$ classe. C'est l'objet des algorithmes présentés maintenant.

Dans le but de définir une version plus robuste au bruit de l'algorithme des FCM, Y. Ohashi [3] a proposé d'ajouter une classe ω_* (dite classe de bruit) au modèle, de telle sorte que tout point soit potentiellement du bruit. Il définit alors le critère suivant :

$$J_m(U,V,\alpha) = \alpha \sum_{i=1}^{c}\sum_{k=1}^{N} {U_{ik}}^m \, {d_{ik}}^2 + (1-\alpha)\sum_{k=1}^{N} {U_{*k}}^m \qquad (15)$$

où U est une matrice de partittion floue de dimension $(c+1, N)$, U_{*k} représentant l'appartenance à la classe de bruit

V est l'ensemble des centres V_i des classes ω_i $(i = 1, c)$

$m \in]1, +\infty[$ est un paramètre contrôlant le degré de flou de la partition

d_{ik} est la distance euclidienne entre X_k et V_i (3) ou (6)

$\alpha \in [0,1]$ est le paramètre contrôlant l'importance de la classe ω_*

A V fixé, l'optimum du critère est obtenu pour :

$$U_{*k} = \frac{\left(\frac{1-\alpha}{\alpha}\right)^{-\frac{1}{m-1}}}{\sum_{i=1}^{c} d_{ik}^{-\frac{2}{m-1}} + \left(\frac{1-\alpha}{\alpha}\right)^{-\frac{1}{m-1}}} \qquad \forall k = 1, N \qquad (16)$$

$$U_{ik} = \frac{d_{ik}^{-\frac{2}{m-1}}}{\sum_{i=1}^{c} d_{ik}^{-\frac{2}{m-1}} + \left(\frac{1-\alpha}{\alpha}\right)^{-\frac{1}{m-1}}} \qquad \forall i = 1, c \quad \forall k = 1, N \qquad (17)$$

La condition (1b) des FCM est remplacée par :

$$\sum_{i=1}^{c} U_{ik} + U_{*k} = 1 \qquad \forall k = 1, N \qquad (1''b)$$

Remarquons également que lorsque $\alpha = 1$, on retrouve bien évidemment l'algorithme FCM. L'algorithme de Ohashi est :

Etape 0 :

- fixer le nombre c de classes
- fixer le paramètre de flou m
- initialiser aléatoirement la matrice de partition U vérifiant (1a), (1''b) et (1c).
- fixer α

Etape 1 :

- calculer, à l'itération l, les centres V_i^l (4)

Etape 2 :

- mettre à jour la matrice de partition U^l (16), (17)

Etape 3 :

- estimer la stabilité de U^l en utilisant par exemple

  une norme matricielle :

  si $\left\|U^l - U^{l-1}\right\| \leq \varepsilon$ stop,

  sinon incrémenter l et reprendre à l'étape 1.

L'avantage de cette approche par rapport à la précédente, c'est qu'il n'est pas nécessaire de définir un seuil d'appartenance aux classes U_0 pour décider si un point X_k doit être considéré comme du "bruit". Il suffit que $U_{*k} = \max_{i=1,c+1}(U_{ik})$.

L'inconvénient réside dans le choix de α . $\frac{1}{c+1}$ semble être valeur acceptable.

Des exemples de classification sont donnés en Annexe 2.5.

R.N. Dave [4] définit la classe de bruit comme un entité universelle qui se situe toujours à la même distance de chaque point de l'ensemble à classifier. A chaque itération, la distance d'un point X_k à la classe ω_* est défini comme une constante $d_{*k} = \delta$ $(\forall k = 1, N)$. L'auteur propose :

$$\delta^2 = \lambda \frac{\sum_{i=1}^{c}\sum_{k=1}^{N} d_{ik}^2}{N\,c} \qquad (18)$$

où le facteur λ doit être choisi. Notons que plus λ tend vers 0, plus la classe ω_* est importante.

On peut écrire le critère défini par R.N. Dave comme suit :

$$J_n(U,V) = \sum_{i=1}^{c}\sum_{k=1}^{N} {U_{ik}}^m {d_{ik}}^2 + \sum_{k=1}^{N} {U_{*k}}^m {d_{*k}}^2 \qquad (19)$$

et la matrice de partition optimale correspondante est alors :

$$U_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik}^2}{d_{jk}^2} \right)^{1/m-1}} \qquad \forall i = 1, c+1 \quad \forall k = 1, N \qquad (20)$$

Le principe de l'algorithme reste inchangé :

Etape 0 :

- fixer le nombre c de classes
- fixer le paramètre de flou m
- initialiser aléatoirement la matrice de partition U vérifiant (1a), (1''b) et (1c).
- fixer λ

Etape 1 :

- calculer, à l'itération l, les centres V_i^l (4)

Etape 2 :

- mettre à jour δ^2 (18)
- mettre à jour la matrice de partition U^l (20),

Etape 3 :

- estimer la stabilité de U^l en utilisant par exemple

  une norme matricielle :

  si $\left\| U^l - U^{l-1} \right\| \leq \varepsilon$ stop,

  sinon incrémenter l et reprendre à l'étape 1.

Des résultats obtenus avec cette méthode sont présentés en Annexe 2.6.

3 - Matériels et Méthodes

3.1 Matériel biologique

Préparation 1 heure, pas de cuisson ...

3.1.1 - Prélèvement de l'hémolymphe.

Dans un premier temps, les huîtres sont brossées sous l'eau courante, afin de débarrasser la coquille des débris. Elles sont ensuite ouvertes par section du muscle adducteur (cette opération doit être effectuée avec soin afin de ne pas perforer la cavité péricardique).

On ponctionne alors l'hémolymphe directement dans la cavité péricardique (figure 1) à l'aide d'une seringue. Cette seringue contient au trois quart une solution d'Alsever dont la propriété est de limiter l'agrégation naturelle des cellules. 500 µl à 1000 µl d'hémolymphe sont ainsi prélevés par animal sur Alsever.

3.1.2 - Cytocentrifugation

Les cytocentrifugations sont des centrifugations de cellules effectuées sur lames histologiques permettant l'utilisation de faibles volumes de suspension cellulaire. Elle sont effectuées à faible vitesse et sont de courte durée, ce qui permet de conserver la morphologie des cellules.

Le programme de cytocentrifugation est le suivant : 500 trm, 1 minute, avec accélération progressive et décélération rapide.

On peut déposer une quantité variable de cellules sur une lame, généralement entre 50 000 et 500 000.

L'HUITRE PLATE

Ostrea edulis

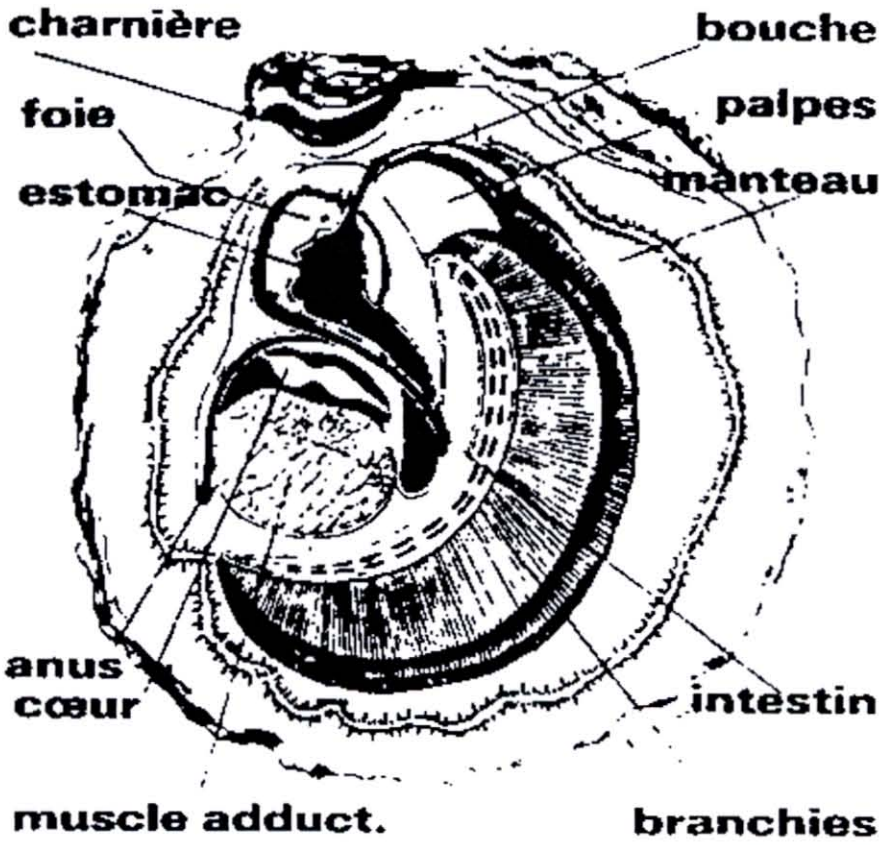


Figure 1

3.1.3 - Coloration

Les cellules étalées sur lame sont fixées pendant 30 secondes dans un bain de méthanol, puis colorées. Elles sont plongées dans deux bains de colorant (Kit Hémacolor - Merck). Le premier colore les éléments acidophiles des cellules en rouge, soit globalement le cytoplasme; le second bain colore les éléments basophiles des cellules en bleu, soit essentiellement la chromatine du noyau.

Le temps d'immersion dans chaque bain est d'environ 30 secondes.

D'autres colorants ont été testés (Giemsa, bleu de méthylène, bleu de UNA, Hématoxyline, Eosine) seuls ou en association.

3.2 - Matériel Optique et Informatique (Récupération des variables pour la classification)

Dans de nombreux domaines, l'analyse d'image est utilisée, d'une part comme outil descriptif, objectif, reproductible et précis, d'autre part comme un moyen d'automatisation.

C'est pour ces différents avantages que cette méthode a été retenue pour extraire des "variables" pour la classification.

Le logiciel IPS 4.6 - SAMBA (Système d'Analyse Micro photométrique à Balayage Automatique) de chez Alcatel, a été utilisé afin d'effectuer de nombreuses mesures sur les cellules cytocentrifugées et colorées.

3.2.1 - Présentation du logiciel

IPS est un logiciel d'analyse d'images offrant de nombreuses fonctions d'analyse et de traitement d'image; il facilite également la construction d'applications utilisant ces fonctions.

Des applications peuvent être créées, permettant d'automatiser un ensemble de requêtes. Elles sont implantées sous la forme de scripts de commandes SAMBA et sont compilées avant exécution par IPS.

Les commandes SAMBA reprennent l'ensemble des commandes de l'environnement interactif IPS et ajoutent d'autres fonctions du driver d'imagerie. SAMBA propose ainsi :

- différents modes d'acquisition des images,

- différents mode d'affichage,
- des fonctions de traitement d'image
  - filtrage,
  - manipulation des niveaux de gris et des vraies couleurs,
  - opérations sur les images en morphologie, en géométrie,
  - ...
- des fonctions d'analyse d'image
  - segmentation,
  - manipulation des objets, des contours,
  - ...
- des fonctions de mesure dans les images
  - histogrammes,
  - profils,
  - paramètres divers.
- ...

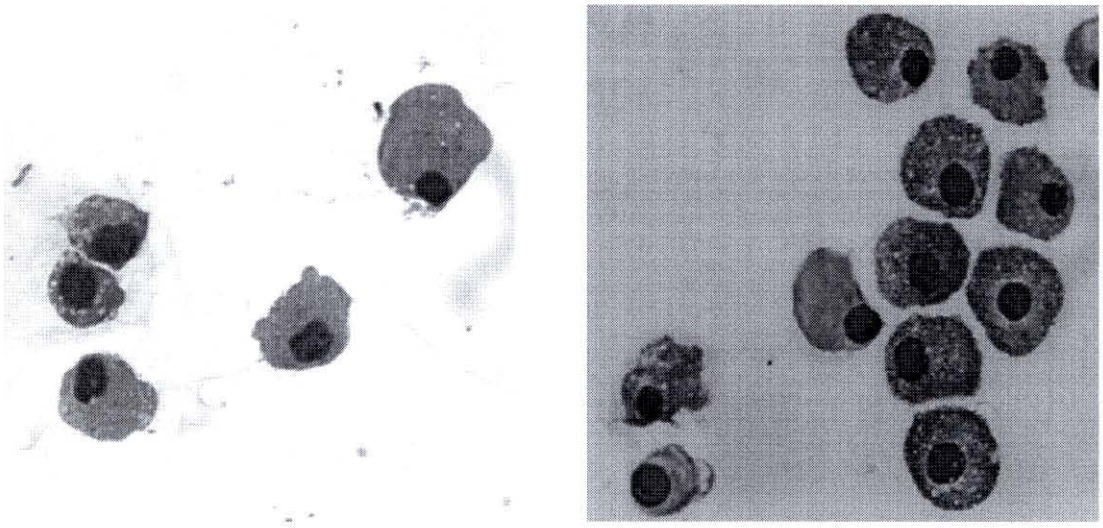
Il offre aussi la possibilité d'utiliser une platine motorisée. Cependant, la caméra en place est une caméra CCD Sony monochrome, montée sur un microscope OLYMPUS BH2, non équipé d'un système de motorisation.

SAMBA intègre également un langage de programmation permettant notamment :
- de faire et d'utiliser des variables de divers types,
- de faire des comparaisons,
- de réaliser des traitements itératifs
- de générer des macro-commandes
- ...

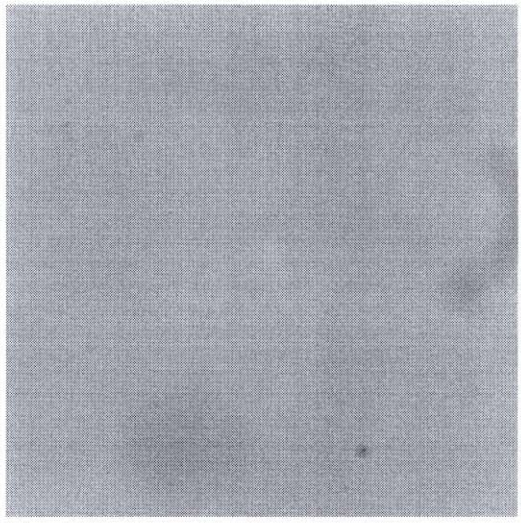
3.2.2 - Utilisation de IPS - SAMBA

Le logiciel IPS a été utilisé pour effectuer l'acquisition d'images sur les lames de cytocentrifugation, afin de récupérer des paramètres de cellules.

Une phase "d'étalonnage" est effectuée pour chaque lame : la soustraction (***AbsDif***) entre deux images (une image de cellules et une image de "référence") gomme les impuretés résidentes sur les objectifs et "met à zéro" le fond de l'image. (Exemple : planche 1)

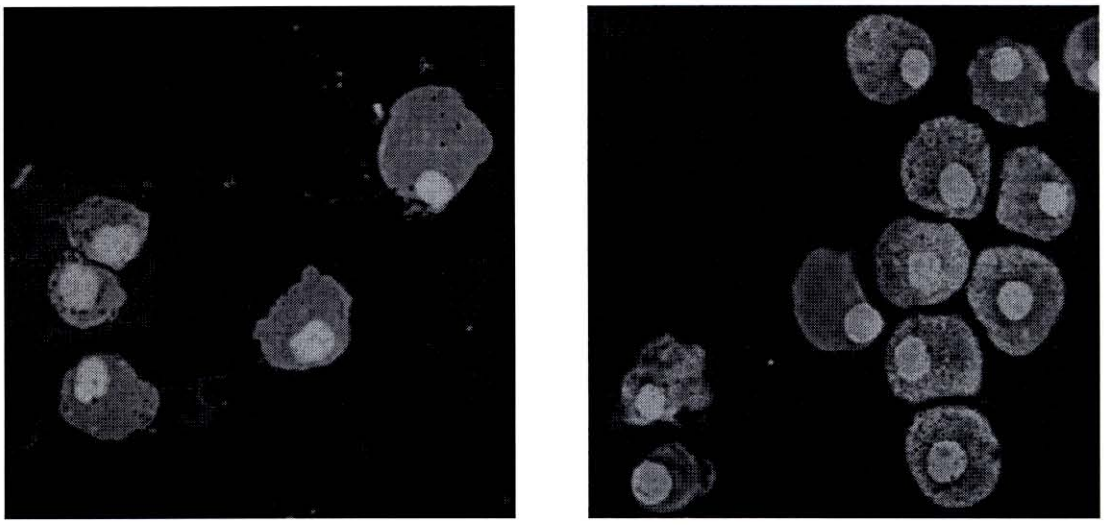


Images acquises



Images de référence

Soustraction des images acquises des images de référence



Les images obtenues sont de même intensité.

Planche 1

A ce stade, les cellules sont individualisées dans les images.

A partir d'un histogramme d'intensité, calculé sur l'image entière (commande ***ImgHisto***), la commande ***Fisher*** calcule automatiquement un seuil de binarisation. Ce seuil sépare au mieux deux zones dans l'histogramme; en règle générale, le fond de l'image est noir et les cellules se détachent du fond car elles sont claires : les zones séparées sont donc le fond et les cellules. (Planche 2)

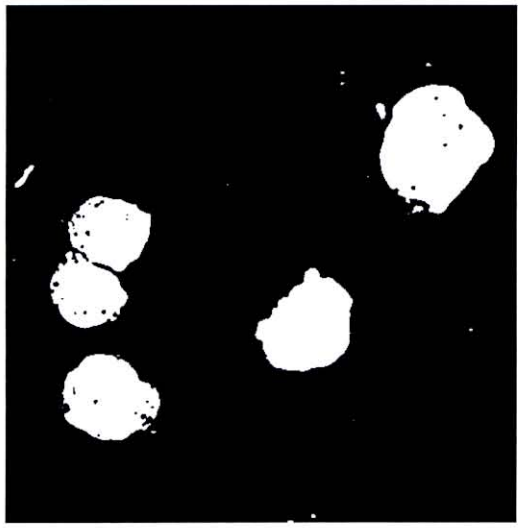
Une inversion vidéo (***NotImg***) va ensuite mettre le fond de l'image et les trous en blanc. Puis, un étiquetage (***Label***) des objets assez grands (c'est à dire du fond et pas des trous) va produire une image dans laquelle se trouve l'objet "fond" seul en niveau de gris 1. Enfin une binarisation (***Thresh***) au niveau 1, en retenant la partie foncée donne une image binaire contenant les cellules sans trous. (Planche 2)

La commande ***Label*** permet ensuite d'étiqueter les objets de cette image (ici les cellules). La commande renvoie le nombre d'objets trouvés (au maximum 255, mais dans notre cas, c'est largement suffisant).

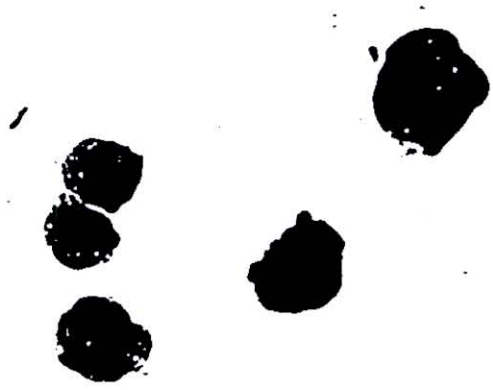
La commande ***Extract*** permet alors d'extraire un objet donné de l'image. Elle retourne quatre paramètres de position (coordonnée maximale droite, coordonnée maximale gauche, coordonnée maximale haute, coordonnée maximale basse), la surface de la cellule et une autre image binaire dans laquelle est isolée la cellule spécifiée. Cette image sert de masque pour isoler la cellule dans l'image en niveaux de gris (commande ***AndImg***). (Planche 2)

Le masque permet également d'appeler la fonction ***GetRegHisto*** qui calcule un histogramme juste sur la cellule.

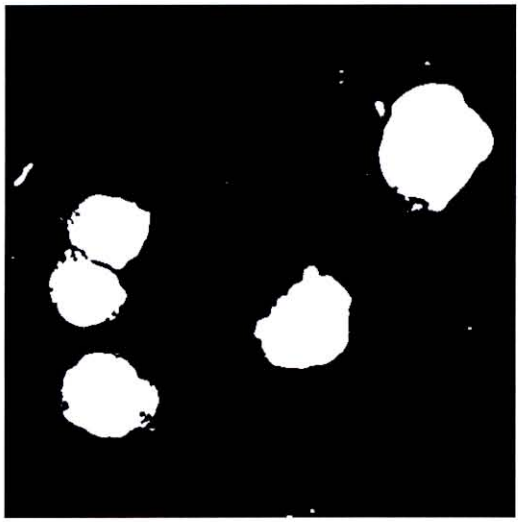
La démarche qui isole les cellules dans l'image est de nouveau effectuée, mais cette fois pour isoler le noyau dans la cellule. (Planche 2)



Binarisation

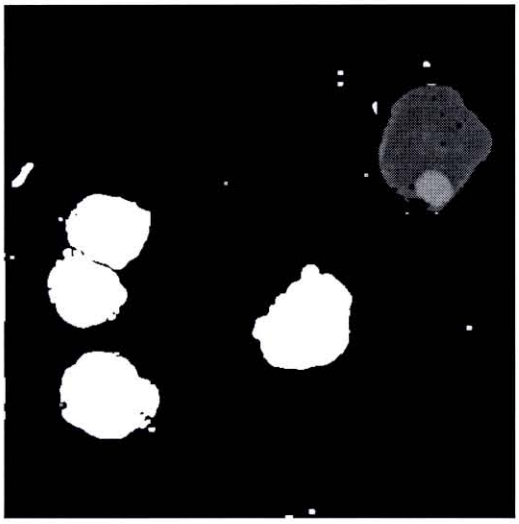


Inversion vidéo

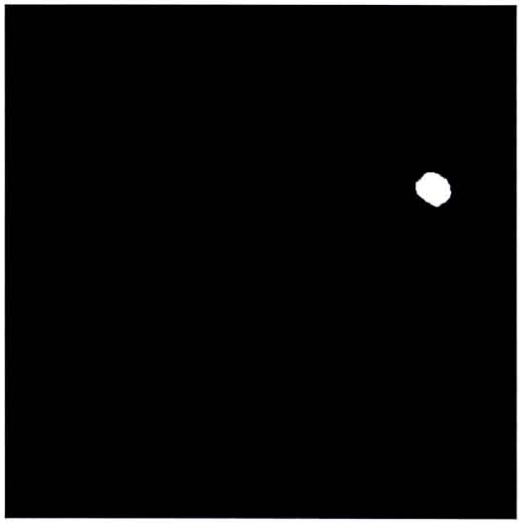


Rebouchage

Isolement



Utilisation du masque



Identification du noyau

Planche 2

Pour chaque cellule retenue, plusieurs fonctions d'extraction de paramètres sont utilisées. Ces fonctions sont :

- ***Densito*** qui renvoie des mesures densitométriques
  - Surface
  - Moyenne en terme de niveau de gris,
  - Ecart-type à la moyenne,
  - Degré de symétrie de l'histogramme,
  - Mesure d'aplatissement de l'histogramme.

- ***TextureCooc*** qui renvoie des paramètres de texture par analyse de la matrice de cooccurrence P
  - Moyenne locale $\left(\frac{\sum_{ij} (i+j) \cdot P_{ij}}{nClasses} \right)$
  - Energie $\left(\sum_{ij} P_{ij}^2 \right)$
  - Entropie $\left(-\sum_{ij} P_{ij} \cdot \ln(P_{ij}) \right)$
  - Inertie $\left(\frac{\sum_{ij} (i-j)^2 \cdot P_{ij}}{nClasses} \right)$

(où P_{ij} est la probabilité du niveau de gris i situé à une certaine distance de la classe de gris j).

- ***TextureRLS*** qui renvoie des paramètres de texture d'après la méthode des longueurs de section.

(matrice des longueurs de section R : R_{lg} = probabilité des segments de la classe de gris g ayant la longueur l).

  - SRE importance des sections courtes $\left(\sum_{lg} \frac{R_{lg}}{l^2} \right)$

- LRE importance des sections longues $\left(\sum_{lg} R_{lg} \cdot l^2\right)$
- GLD distribution des niveaux de gris $\left(\sum_{g}\left(\sum_{l} R_{lg}\right)^2\right)$
- RLD distribution des longueurs de sections $\left(\sum_{l}\left(\sum_{g} R_{lg}\right)^2\right)$
- RLP pourcentage des longueurs de sections $\left(100 \cdot \sum_{lg} R_{lg}\right)$

Lorsque c'est possible, l'espace de travail est reduit à une portion de l'image avec la commande ***SetWindow***. A la fin du traitement, on restitue l'image entière avec la commande ***RestWindow***.

La phase "d'analyse d'image" se fait en deux étapes : l'acquisition d'image supervisée par l'opérateur dans un premier temps et l'analyse automatique dans un second temps. Deux programmes sont donc exécutés pour effectuer une analyse complète

4 - Résultats - Discussion

4.1 - PARTIE 1 : Préparations cytologiques et traitement informatique.

4.1.1 Les cytocentrifugations

Le nombre de cellules déposées sur les lames a varié de 50 000 à 500 000.

Si le nombre de cellules est trop élevé par lame, il risque d'y avoir des amas et dans ce cas, il sera impossible d'identifier ou mesurer les hémocytes. Les cellules étant accolées, il est impossible de bien définir leur limite.

En revanche, si les cellules sont trop peu nombreuses sur la lame, il faudra faire l'acquisition d'un plus grand nombre d'images pour analyser une quantité suffisante d'hémocytes et le temps de traitement et de calcul est plus long.

Une quantité de 300 000 à 400 000 cellules est un bon compromis.

4.1.2 - La coloration

La phase de coloration revêt un caractère important, dans la mesure où l'identification ne sera pas faite par l'oeil de l'observateur, mais à travers une caméra. Si les cellules et leur noyau ne sont pas suffisamment colorés et contrastés, il sera impossible d'effectuer les mesures nécessaires à leur identification.

Plusieurs essais de coloration ont été effectués. Il est nécessaire que le colorant permette de séparer le fond et les cellules, et à l'intérieur de la cellule, le cytoplasme et le noyau.

Coloration au Giemsa (Photos A, B et C)

On note ici un contraste général insuffisant, aussi bien entre le fond de l'image et les cellules qu'entre le cytoplasme et le noyau.

Coloration au Bleu de méthylène (Photos D et E)

La coloration prend mal et le noyau est peu coloré : avec la caméra monochrome, il est impossible de distinguer le noyau dans le cytoplasme.

Coloration au Bleu de méthylène suivi d'une coloration au Giemsa (Photo F)

Les cellules sont mieux colorées, mais le noyau ne se distingue pas dans la cellule en niveaux de gris.

Coloration à l'Eosine (Photos G, H et I)

La coloration est bonne, mais le contraste noyau / cytoplasme est insuffisant une fois l'image ramenée en niveaux de gris.

Coloration à l'Hématoxyline (Photos J et K)

La coloration est faible, le contraste insuffisant.

Coloration au Bleu de UNA (Photo L)

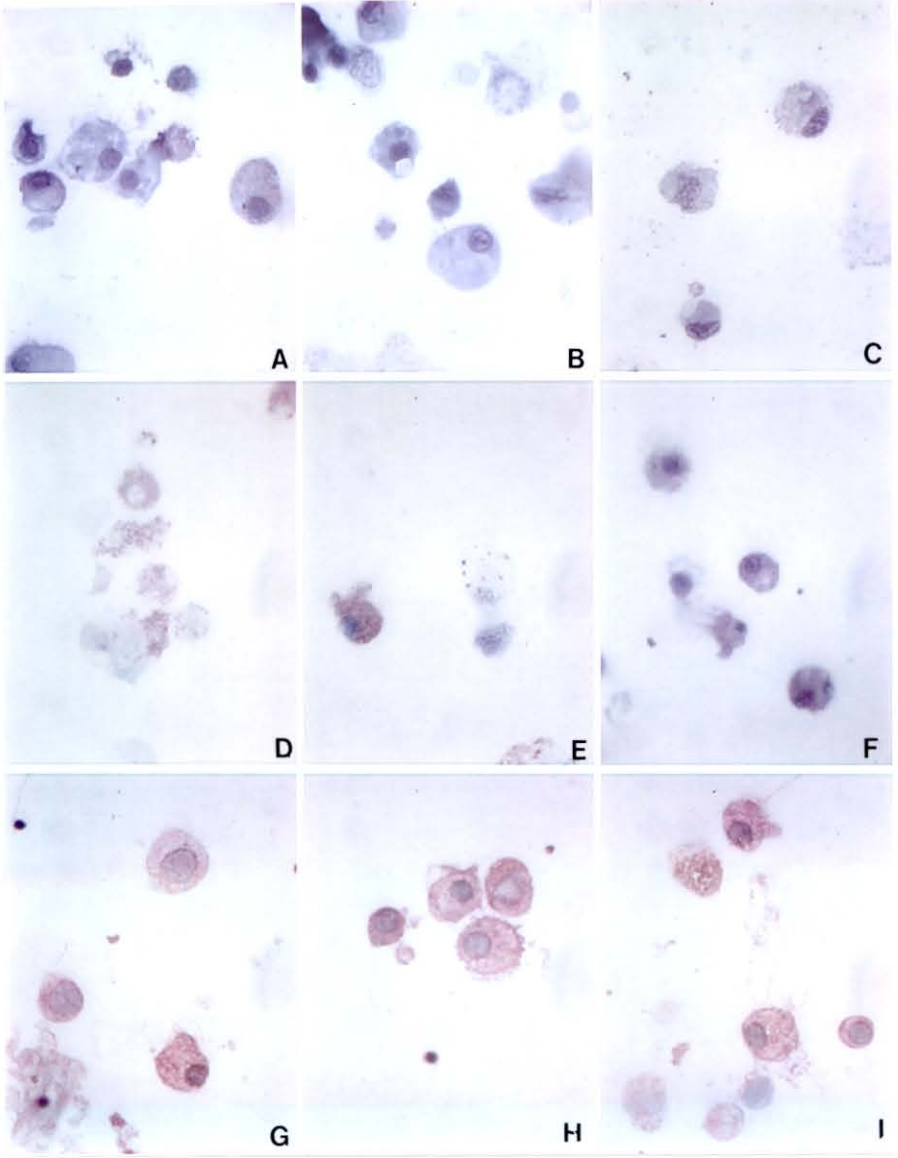
Les cellules ne sont pas colorées

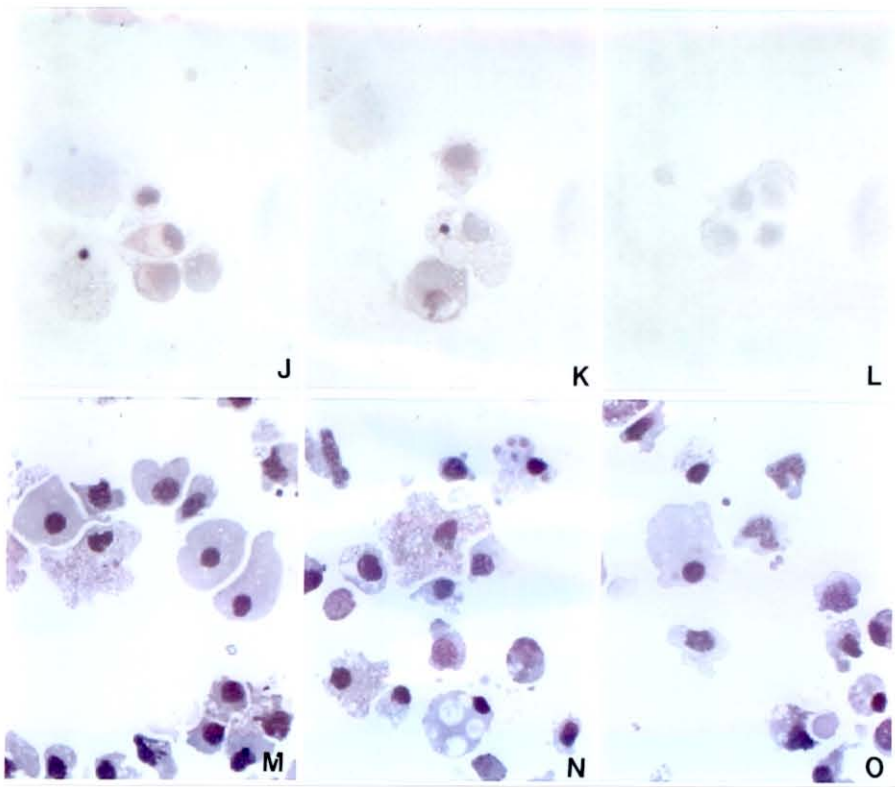
Coloration à l'Hémacolor (Photos M, N et O)

La coloration est intense et les contrastes nets.

C'est donc le kit Hémacolor (Merck) qui donne les meilleurs résultats pour un usage simple et rapide : 30 s dans chacun des deux bains de colorant.

Il est inutile voire défavorable de colorer plus longtemps, en utilisant ce kit.


A
B
C
D
E
F
G
H
I



J - K : Coloration à l'Hématoxyline

L : Coloration au Bleu de UNA

M - N - O : Coloration à l'Hémacolor

4.1.3 - Les types hémocytaires

Pour les études effectuées à l'IFREMER, trois types de cellules ont été définis sur la base de préparations cytocentrifugées : les granulocytes, les grandes cellules agranuleuses et les petits hyalinocytes.

Les hémocytes granuleux se caractérisent par l'aspect très hétérogène de leur cytoplasme qui contient des granulations en nombre et de forme variables. De taille homogène, ils ont un rapport nucléocytoplasmique (taille noyau / taille cellule) faible.

Les grandes cellules agranuleuses possèdent un cytoplasme d'aspect homogène, gris rosé après coloration au kit Hémacolor (Merck), qui peut contenir quelques vacuoles claires. Leur taille est très variable et elles ont un rapport nucléocytoplasmique faible.

Les petits hyalinocytes se distinguent des autres cellules par un rapport nucléocytoplasmique élevé, mais variable. Ils possèdent un cytoplasme basophile réduit à une mince auréole autour du noyau. Quelques cellules présentent de courts prolongements cytoplasmiques.

Cependant, il n'y a pas de consensus sur le nombre et la distinction des types cellulaires décrits. La classification automatique des hémocytes présente donc deux aspects intéressants : confirmer ou infirmer l'hypothèse de la répartition en trois catégories et obtenir un moyen de classement des cellules fiable et sans à priori de l'observateur.

4.1.4 - Utilisation d'IPS-SAMBA

L'utilisation du logiciel dans le cadre du travail rapporté ici est légèrement limitée, puisque l'acquisition des images se fait par l'intermédiaire d'une caméra monochrome, montée sur un microscope sans motorisation de la platine.

Il est peut-être regrettable que l'on ne puisse pas faire d'acquisitions couleur, car les lames sont colorées avec des teintes à dominante violette. On aurait pu faire les analyses et les traitements sur le plan rouge des images, dans lequel elles auraient présenté un meilleur contraste.

Pour pallier cet inconvénient, un filtre vert a été utilisé sur la lampe du microscope; le contraste est légèrement meilleur que sans filtre. En effet, il permet de récupérer essentiellement l'information de contraste bleue et rouge soit la totalité de l'information violet tout en éliminant l'information "verte" qui n'apporte rien de mieux.

En ce qui concerne la platine, la motorisation peut se heurter au fait que la position du spot cellulaire peut varier. En effet, le système de dépôt sur lame se monte manuellement, ce qui peut impliquer un décalage de quelques millimètres. Ce décalage grossit 100x10 fois au microscope, imposerait un balayage très grand et surtout un temps de traitement très long.

4.1.5 - Le programme de traitement d'image

Les objectifs du microscope et/ou de la caméra peuvent présenter des tâches ou des rayures. De plus, l'acquisition des images ne se fait pas toujours avec la même intensité lumineuse dans la mesure où celle-ci se règle manuellement et parce qu'il faut en changer selon la qualité de la coloration. C'est pour cette raison qu'une phase "d'étalonnage" est nécessaire pour chaque lame : l'acquisition d'un champ vide sert de référence (pour la correction de la luminosité et des défauts des objectifs) et il faudra garder la même intensité lumineuse pendant la série d'acquisition sur une lame donnée.

Un inconvénient intervient également dû à la structure des cellules. Les cellules possédant des vacuoles (les vacuoles ont une couleur proche de celle du fond de l'image) vont présenter un aspect "troué" après binarisation.

Un moyen de boucher ces trous est d'appliquer quelques transformations à l'image binaire : l'inversion vidéo, l'étiquetage du fond de l'image, et enfin la binarisation au niveau 1.

Un autre problème peut se poser, quand deux cellules se touchent. Le programme n'en considère qu'une. Il ne sera pas possible de la classer correctement. Une façon d'éviter ce problème, est d'éliminer les cellules à deux noyaux ou plus; cependant, il existe réellement des cellules à deux noyaux; pour simplifier, elle seront éliminées.

Pour accélérer le traitement, il est possible de réduire l'espace de travail à la fenêtre englobant l'objet traité.

Les coordonnées de cette fenêtre sont celles obtenues avec la fonction ***extract***. Le positionnement de l'espace de travail se fait avec la commande ***SetWindow***. A la fin du traitement, on restitue l'image entière avec la commande ***RestWindow***.

Comme l'acquisition des paramètres sur une image reste une opération très longue, la phase "d'analyse d'image" est divisée en deux étapes : la numérisation supervisée par l'opérateur dans un premier temps et l'analyse dans un second temps. L'acquisition se fait de façon semi-automatique, c'est à dire que l'utilisateur doit positionner la lame sous la caméra et la déplacer pour visualiser les champs à l'écran. Pour chaque lame, il spécifie le répertoire de sauvegarde des images, règle l'intensité lumineuse du microscope, saisit un champs vide (sans cellules), puis saisit les différentes images de cellules sans changer les réglages relatifs à l'intensité lumineuse. L'image "vide" porte systématiquement le nom "imvide.bmp". Il faudra donc créer autant de répertoires sur le disque qu'il y a de lames à observer.

L'analyse d'image se fait de façon totalement automatique après avoir spécifié les répertoires de travail et le nombre d'images acquises sur chaque lame.

Le second programme effectue les mesures des cellules des différentes images et, en fonction de ces mesures, établit un classement des hémocytes dans l'une des trois catégories (ou éventuellement ne les classe pas), puis renvoie la composition hémocytaire de chaque lame.

4.2 - PARTIE 2 : La classification.

Les 15 paramètres retenus (taille de la cellule, taille du noyau, rapport nucléocytoplasmique, moyenne, dissymétrie de l'histogramme, aplatissement de l'histogramme, moyenne locale, énergie, entropie, inertie, SRE, LRE, GLD, RLD, RLP voir paragraphe 3.2.2) pour caractériser les hémocytes ont été mesurés sur environ 800 cellules provenant de 8 lames histologiques. Ces lames sont issues de prélèvements différents et ont été colorées indépendamment.

Les différents algorithmes ont été utilisés sur ces données.

Pour chacun des quatre algorithmes (Fuzzy, Possibilistic, Ohashi et Dave), deux variantes sont proposées dans le choix de la métrique : distance euclidienne ou distance de Mahalanobis.

4.2.1 - En distance euclidienne.

Pour l'algorithme des Fuzzy C-Means et celui de DAVE, lorsque la distance euclidienne est utilisée, on constate (quelques soient les paramètres utilisés : nombre de classes et

paramètre de flou), que la classification des cellules s'effectue suivant la valeur ayant la plus forte variance, c'est à dire la taille de la cellule.

Comme les trois types cellulaires déjà définis ne se différencient pas seulement par la taille de la cellule, les résultats obtenus ainsi ne permettent pas d'obtenir des classes regroupant des cellules homogènes pour l'ensemble des paramètres considérés.

L'utilisation de cette même métrique avec l'algorithme de OHASHI donne des résultats non exploitables, dans la mesure où tous les points se retrouvent dans la classe de bruit.

4.2.2 - Algorithme possibiliste

L'avantage de cet algorithme, c'est qu'il peut regrouper des classes s'il ne trouve pas suffisamment de différence pour les séparer : les éléments classifiés appartiennent fortement à plusieurs classes à la fois, et celles-ci sont centrées au même endroit.

Ainsi, si on propose de trouver trop de classes, l'algorithme les regroupe.

L'inconvénient ici, c'est que les paramètres ne doivent pas êtres suffisamment discriminants pour les cellules (quelque soit la métrique utilisée), et celles-ci appartiennent autant à chaque classe lorsqu'on utilise l'algorithme : il n'y a en fait qu'une classe de cellules.

4.2.3 - Autres algorithmes (tenant compte de la dispersion autour des axes) Fuzzy C-Covariances, OHASHI , DAVE

Les résultats obtenus avec ces trois algorithmes sont sensiblement identiques.

L'influence du paramètre de flou est notable puisque les grandes valeurs de m conduisent à rassembler les classes : $U_{ik} \approx \frac{1}{C}$ $\forall i, \forall k$, et les classes ont même centre.

Le nombre de classes demandées modifie leur position dans l'espace en dimension 15 et leur contenu. Il existe des critères de validation du nombre de classe.

Les différents critères utilisés sont de deux types : des critères spécifiques à la classification floue :

- Coefficient de partition : $F(U,C) = \frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{C} U_{ik}^2$ à maximiser.

La valeur de ce coefficient est comprise entre $\frac{1}{C}$ et 1; elle augmente avec les grandes valeur de U_{ik}. Plus il existe un U_{ik} proche de 1 pour un k donné, plus $\sum_{i=1}^{C} U_{ik}^2$ est grand.

- Entropie de classification : $H(U,C) = -\frac{1}{N}\sum_k \sum_i U_{ik} \log_a(U_{ik})$

$$H2(U,C) = \frac{H(U,C)}{\log_a(C)} \quad \text{à minimiser.}$$

Ce paramètre représente le désordre de la classification.

- Max : $P1(U,C) = \frac{1}{N}\sum_k \max_i U_{ik}$ à maximiser.

Ce paramètre représente l'appartenance moyenne d'un point à la classe dont il est le plus proche.

- Min-Max : $P2(U,C) = \frac{1}{N}\sum_k \frac{\min_i U_{ik}}{\max_i U_{ik}}$ à minimiser.

Ce paramètre donne une information quant à la forte appartenance d'un point à la classe dont il est le plus proche, et la faible appartenance de ce point à la classe dont il est le plus loin.

et des critères géométriques :

- Hypervolume flou : $FHV(U,C) = \sum_i \det(\sum_i)^{1/2}$ à minimiser

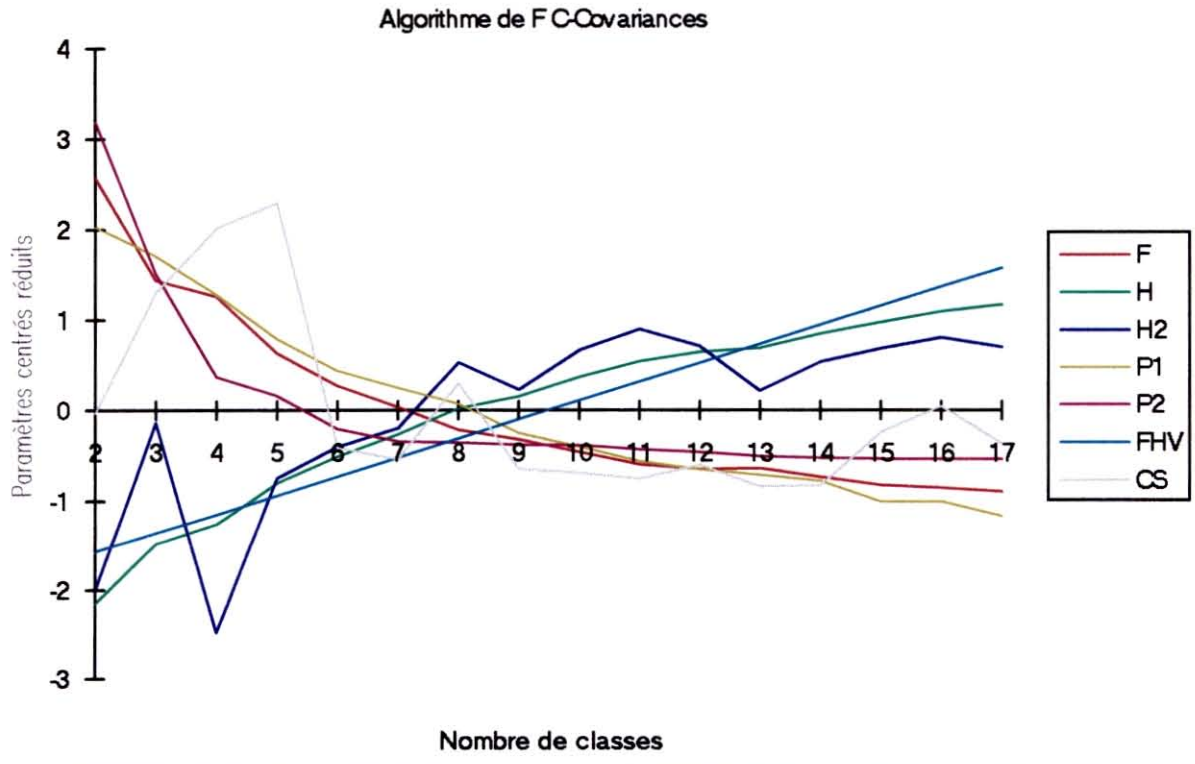
Σ_i : matrice de covariance floue

Ce paramètre donne une information sur la compacité des classes.

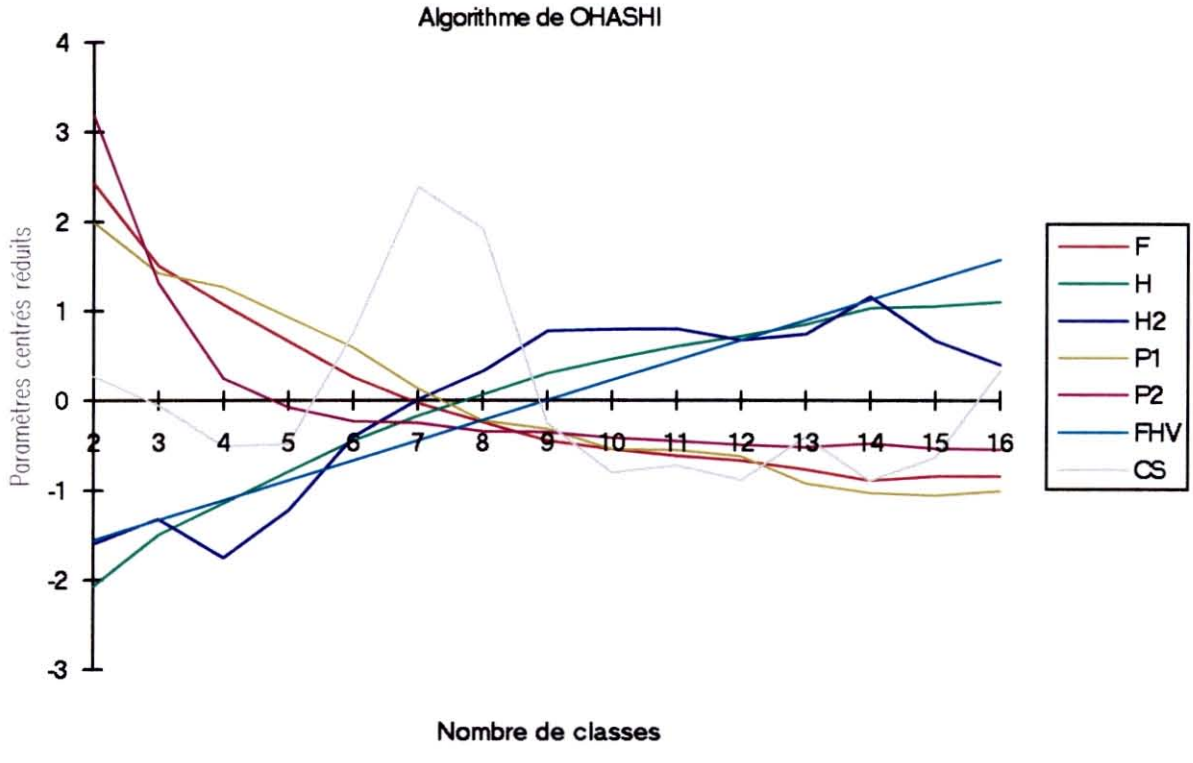
- Compacité-séparabilité : $CS(U,C) = \frac{\frac{1}{N}\sum_i \sum_k (U_{ik} \|X_k - V_i\|)^2}{\min_{i,J,J\neq i} \|V_i - V_j\|^2}$ à minimiser

Ce paramètre donne une information sur la compacité et la séparabilité des classes.

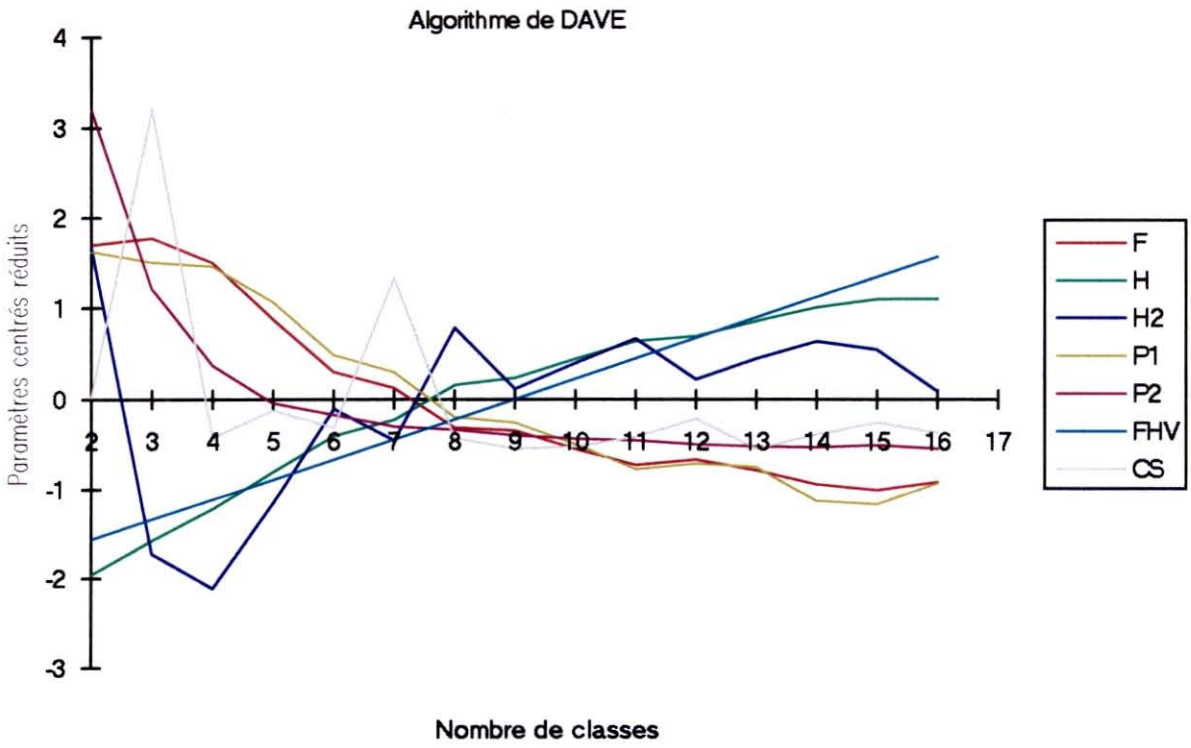
Les trois graphiques suivants montrent l'évolution des critères en fonction du nombre de classes demandées.



Graphique 1



Graphique 2



Graphique 3

Dans le cas de l'algorithme des F C-Covariances, il semble que 7 soit le nombre de classes qui satisfasse au mieux les différentes contraintes. C'est en effet un minimum local pour le critère CS, et un compromis pour tous les autres critères qui sont globalement monotones.

Pour l'algorithme de Ohashi, le nombre de classes optimal est 5. C'est de nouveau le compromis entre les différents critères monotones et surtout un minimum local pour CS.

Concernant l'algorithme de Dave, le nombre de classes qui satisfait au mieux les contraintes est 4. C'est un minimum pour le critère H2, un minimum pour CS, et un bon compromis pour les autres critères.

On constate donc que le nombre de classes optimum diffère selon l'algorithme utilisé, ce qui montre bien que les classes ne sont pas nettement séparables.

En choisissant le nombre de classe approprié pour chaque algorithme, les classes obtenues ne regroupent pas plus spécialement des cellules homogènes du point de vue de l'observateur, qu'un utilisant un nombre de classes quelconque.

4.2.4 - Bilan de la classification

Malgré différents changements de paramètres, il n'y a pas de configuration qui ait donné des résultats directement exploitables pour effectuer un classement, en utilisant les variables sélectionnées de morphométrie. Cependant, en faisant varier leur nombre, on obtient des classes dont le contenu peut être très homogène. Par exemple, en prenant 8 classes, les cellules de grande taille sont divisées en trois classes, dont
une contient exclusivement les cellules dont la texture est granuleuse : les granulocytes. Par ailleurs, les cellules de taille plus petite ne sont pas discriminées de façon correcte selon le jugement de l'observateur (mélange des petits hyalinocytes et des cellules agranuleuses).

En diminuant le nombre de classes à 5, la classe contenant les granulocytes englobe également d'autres cellules dont la texture n'a rien à voir avec les granulocytes. Ce sont simplement de grandes cellules hyalines. Néanmoins, une classe contient exclusivement des petits hyalinocytes.

4.2.5 - ACP sur les données

En effectuant une analyse en composantes principales sur l'ensemble des 15 variables, il est possible de déterminer des variables qui sont peu discriminantes. Mais les essais de classification qui on été réalisés en enlevant ces variables du modèle ne permettent pas de retrouver des classes très homogènes pour l'observateur.

4.2.6 - Classement

Bien que la classification n'est pas aboutit à un résultat concret, un classement peut être effectué en utilisant certaines données.

En effet, dans le cas où les cellules granuleuses sont correctement regroupées, il est possible d'analyser la composition de la classe (notamment la moyenne et l'écart-type des différentes variables) et de dresser un profil du groupe des granulocytes.

Ainsi, on constate que ces cellules se caractérisent par :

- une taille de cellule > 4500 pixels
- une taille de noyau > 700 pixels
- un rapport nucléocytoplasmique faible (<0.26)

- une moyenne de gris > 80
- une moyenne locale > 0.35
- une énergie < 0.1
- une entropie > 0.28
- une importance des courtes section > 0.45
- une importance des grandes section > 20
- un pourcentage des longueurs de sections > 70
- une distribution des longueurs de sections > 0.25

D'autre part, lorsque les petits hyalinocytes sont correctement regroupés, le groupe se caractérise par :

- une petite taille de cellule (< 3500 pixels)
- un rapport nucléocytoplasmique élevé (> 0.3)
- une moyenne de gris importante (> 100)
- un histogramme asymétrique
- une moyenne locale élevée
- une énergie > 0.05
- un rapport des sections courtes > 0.5
- une importance des sections longues > 10

A partir de ces informations, les granulocytes et les petits hyalinocytes sont globalement retrouvés (avec une erreur de 10 à 15 % environ).

La présence de cellules hyalines est déduite si celles-ci n'entrent pas dans les deux catégories précédentes.

Sur un échantillon de 200 cellules comprenant :

- 24 granulocytes
- 59 petits hyalinocytes
- 117 cellules hyalines

le programme réalisé retrouve :

nature des cellule / position dans les classes	Granulocytes	Hyalinocytes	Petits hyalinocytes
Granulocytes	18	3	2
Hyalinocytes	4	110	11
Petits hyalinocytes	0	0	40
Non classées	2	4	6

CONCLUSION

Globalement, les résultats obtenus par classification semblent confirmer ceux obtenus par d'autres techniques d'analyse telles que la cytométrie de flux quant à la séparabilité des types cellulaires : un profil général sort, mais il n'est pas possible de discriminer très clairement plusieurs populations hémocytaires.

En fait des hypothèses sont émises par les chercheurs. Parmi les cellules de l'hémolymphe, il y a peut-être une famille de cellules souche qui évolue progressivement vers un type cellulaire précis; mais la présence de cellules à tous les stades de développement peut être à l'origine de l'impossibilité de les séparer nettement.

Cependant, dans le cas de la classification basée sur des paramètres morphométriques d'analyse d'image, c'est peut-être le fait d'utiliser de tels paramètres qui rend impossible la séparabilité des classes.

Les mesures effectuées ne sont peut-être pas les plus pertinentes même si elles représentent à priori les critères utilisés par l'observateur.

Il faudrait donc rechercher de nouvelles variables, et/ou en éliminer quelques unes, et rappliquer les différents algorithmes de classification en faisant varier plus largement les paramètres, ou alors abandonner l'analyse d'image pour une technique plus adaptée aux cellules.

BIBLIOGRAPHIE

[1] J.C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*, Plenum Press, 1987 (2nd edition).

[2] R. Krishnapuram, J.M. Keller, « A possibilistic approach to clustering », *I.E.E.E. Trans. On Fuzzy Systems*, **1** (2), 98-110, May 1993.

[3] Y. Ohashi, « Fuzzy clustering and robust estimation », in *Proc. Of 9th meeting SAS users Group International*, Hollywood Beach, Florida, 1984.

[4] R.N. Dave, « Characterization and detection of noise in clustering », *Pattern Recognition Letters*, **12**, 657-664, November 1991.

[5] N. COCHENNEC, *La Bonamiose : Mise au point de réactifs pour le diagnostic et étude des mécanismes cellulaires de défense développés par l'hôte l'huître plate, Ostrea edulis. Rapport de première année pour l'obtention du diplôme de l'Ecole Pratique des Hautes Etudes*, 1995.

[6] N. COCHENNEC, *La Bonamiose : Etude du parasite Bonamia ostreae et des mécanismes cellulaires de défense mis en jeu par l'huître plate vis à vis de ce parasite. Rapport de deuxième année pour l'obtention du diplôme de l'Ecole Pratique des Hautes Etudes*, 1996.

[7] N. PARTHUISOT, *Infestation des hémocytes par le parasite Bonamia ostreae : Cytologie comparative in vitro*. Rapport de stage post-DUT, mars-juin 1996.

[8] Valérie BIRAUD, *Application de reconnaissance et de comptage par imagerie numérique des différents types hémocytaires présents dans l'hémolymphe de l'huître plate, Ostrea edulis*. Rapport de stage de maîtrise de Génie Informatique, février-juin 1996.

[9] SAMBA - Manuel de référence. Alcatel TITN Answare 1995.

[10] IPS 4.02 - Manuel d'utilisation. Alcatel TITN Answare 1995.

[11] G. MOREL, J.-G. FOURNIER. *Hybridation in situ en microscopie photonique*. CNRS URA 1459, Institut Pasteur de Lyon.

ANNEXES

Table des matières

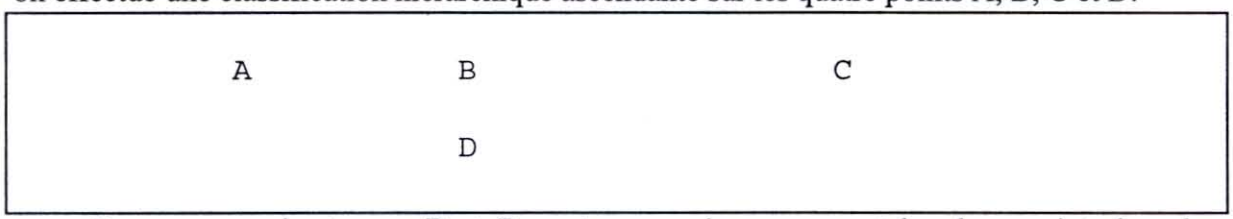
1. - La Classification stricte

Classifier, c'est analyser un ensemble d'objets dans le but de les regrouper en classes homogènes, c'est à dire, de telle sorte que deux objets d'une même classe se ressemblent plus que deux objets de deux classes différentes. Le(s) critère(s) de ressemblance (ou similarité) reste(nt) à déterminer.

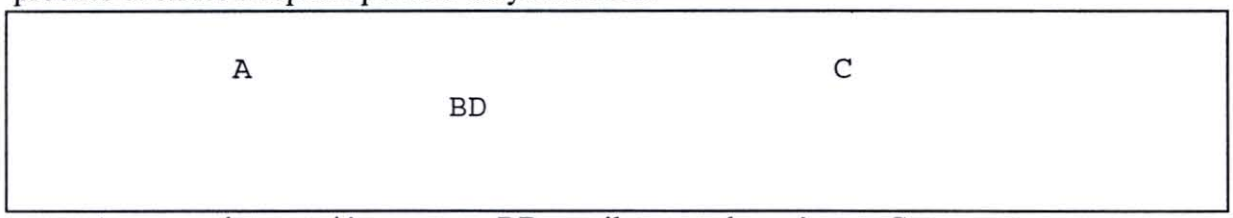
1.1 - Classification hiérarchique

La classification hiérarchique est la démarche qui consiste à regrouper les éléments successivement selon un ou plusieurs critères d'agrégation, jusqu'au regroupement total (classification ascendante) ou bien à diviser successivement les groupes d'éléments jusqu'à l'isolement de tous les éléments (classification descendante).

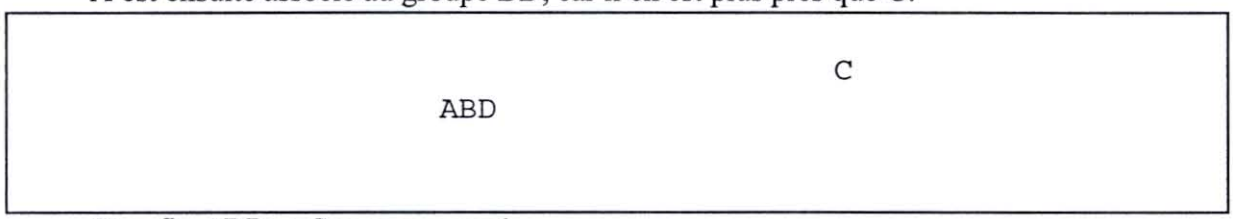
Dans l'exemple suivant, la distance euclidienne est prise comme critère d'agrégation et on effectue une classification hiérarchique ascendante sur les quatre points A, B, C et D.



Dans un premier temps, B et D sont regroupés car ce sont les deux points les plus proches et on les remplace par leur barycentre BD.



A est ensuite associé au groupe BD, car il en est plus près que C.



Et enfin ABD et C sont regroupés.

Aux différents niveaux de la hiérarchie, on avait les partitions suivantes :

- niveau 0 : { {A}, {B}, {C}, {D} }
- niveau 1 : { {A}, {B, D}, {C} }
- niveau 2 : { {A, B, D}, {C} }
- niveau 3 : { {A, B, C, D} }

Il faut alors définir un critère supplémentaire pour décider de la partition à retenir.

1.2 - Classification par partition stricte

La classification par partition stricte consiste à attribuer un élément à une et une seule classe : celle dont il est le plus proche (la proximité étant définie par un critère qui n'est pas forcément la distance euclidienne).

Exemple d'algorithme de classification stricte :

étape 1 :

- initialiser aléatoirement les centres des classes

étape 2 :

- attribuer chaque point à la classe la plus proche
- calculer les centres des classes

étape 3 :

- si la position des centres a beaucoup changé, reprendre à l'étape 2 sinon stop.

1.2.1 - Application sur deux ensembles

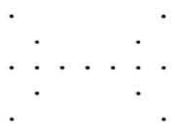
Soit la configuration en deux ensembles suivante :

Après quelques itérations de l'algorithme précédent, on obtient la répartition des points comme suit :

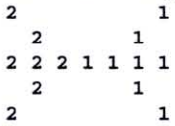
✔ ***Les classes ont été retrouvées sans problème.***

1.2.2 - Application sur le papillon

Soit la configuration en papillon suivante :



Après quelques itérations, on obtient la répartition suivante :



✘ ***Le point central de la configuration en papillon a été attribué à l'une des deux classes alors qu'il n'est pas plus proche du côté droit du papillon que du côté gauche.***

2. - La Classification floue

La différence entre la classification stricte et la classification floue, c'est qu'on autorise un élément à appartenir à plusieurs classes à la fois.

Il existe plusieurs types d'algorithme en classification floue, dans la mesure où on est totalement libre de définir et de gérer le degré d'appartenance d'un point à une classe.

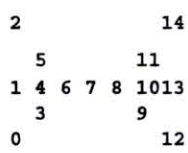
Les exemples suivants montrent comment les algorithmes (présentés au chapitre 1 : Classification) quantifient l'appartenance d'un élément à une classe et éventuellement la non appartenance d'un élément à toutes les classes.

2.1 - Les Fuzzy C-Means

Cet algorithme (présenté dans le paragraphe 1.2) répartit simplement l'appartenance totale d'un point dans les classes selon la proximité du point aux différentes classes.

2.1.1 - Fuzzy C-Means sur le papillon

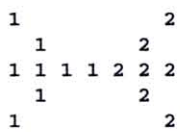
numérotation des points :



Si on choisit le fuzzifier m=2. Les valeurs d'appartenance des points du papillon aux classes sont les suivantes :

numér	0	1	2	3	4	5	6	*7*	8	9	10	11	12	13	14
classe 1	0.86486	0.97230	0.86504	0.94686	0.99899	0.94703	0.88515	*0.50247*	0.11744	0.05333	0.00119	0.05354	0.13491	0.02716	0.13513
classe 2	0.13514	0.02770	0.13496	0.05314	0.00101	0.05297	0.11485	*0.49753*	0.88256	0.94667	0.99881	0.94646	0.86509	0.97284	0.86487

La figure suivante représente l'attribution faite aux classes en considérant la valeur maximale d'appartenance (k appartient à i si $\max_j(U_{jk}) = U_{ik}$).



✔ ***Le point central est attribué à la classe 1, mais on constate que sa valeur d'appartenance est quasiment la même pour les deux classes.***

En choisissant le fuzzifier m=1.25 on a les valeurs d'appartenance suivantes :

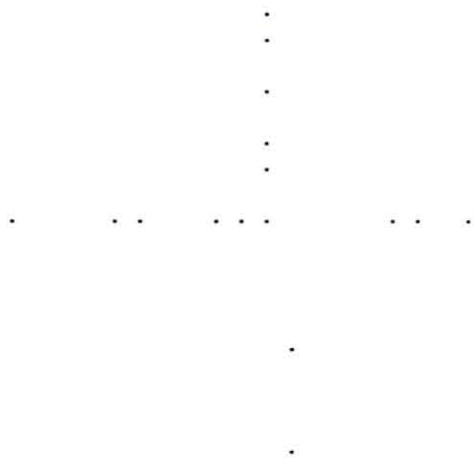
numér	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
classe 1	0.99944	1.00000	0.99944	0.99999	1.00000	0.99999	0.99966	0.49233	0.00031	0.00001	0.00000	0.00001	0.00056	0.00000	0.00056
classe 2	0.00056	0.00000	0.00056	0.00001	0.00000	0.00001	0.00034	0.50767	0.99969	0.99999	1.00000	0.99999	0.99944	1.00000	0.99944

Ce qui nous donne la même répartition.

➽ ***Dans le cas où m est petit (proche de 1), la partition est presque stricte.***

2.1.2 - Fuzzy C-Means sur la croix de Gustafson

La configuration de la croix de Gustafson est approximativement la suivante :



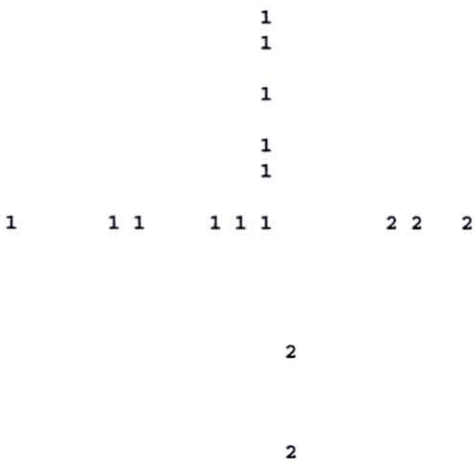
Après utilisation de l'algorithme des FC-M, on attribue un point à la classe pour laquelle il a la plus grande valeur d'appartenance :

pour m=1.25, les valeurs d'appartenance sont les suivantes :

numéro	0	1	2	3	4	5	6	7	8	9
classe 0:	0.98866	0.99719	0.99967	0.99966	0.99881	0.98441	0.00020	0.00023	0.00038	0.00039
classe 1:	0.01134	0.00281	0.00033	0.00034	0.00119	0.01559	0.99980	0.99977	0.99962	0.99961

numéro	10	11	12	13	14	15	16	17	18	19
classe 0:	0.99317	0.99510	0.99962	0.99995	0.99987	0.99993	0.99998	0.93573	0.02876	0.04716
classe 1:	0.00683	0.00490	0.00038	0.00005	0.00013	0.00007	0.00002	0.06427	0.97124	0.95284

La partition est pour ainsi dire stricte, mais les classes obtenues ne sont pas celles attendues.



pour m=2.

numéro	0	1	2	3	4	5	6	7	8	9
classe 0:	0.24520	0.17718	0.11008	0.07549	0.09999	0.22378	0.98834	0.97065	0.91673	0.91245
classe 1:	0.75480	0.82282	0.88992	0.92451	0.90001	0.77622	0.01166	0.02935	0.08327	0.08755

numéro	10	11	12	13	14	15	16	17	18	19
classe 0	0.31145	0.30301	0.20351	0.13795	0.14639	0.13241	0.08142	0.28525	0.56041	0.56649
classe 1	0.68855	0.69699	0.79649	0.86205	0.85361	0.86759	0.91858	0.71475	0.43959	0.43351

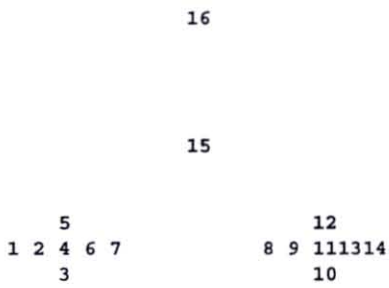
La partition est plus floue, mais les classes ne sont toujours pas celles attendues.
La répartition des points dans les classes est toujours la même.

✗ Il aurait fallut que chaque branche soit considérée comme une classe !

➽ Cet algorithme n'est pas conçu pour retrouver des classes non sphériques.

2.1.3 - Fuzzy C-Means sur deux ensembles bruités

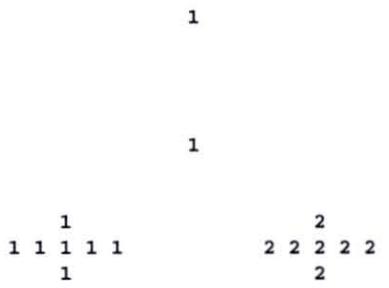
En rajoutant deux points de "bruit" à la configuration du 1.2.1 les valeurs d'appartenance obtenues ne sont plus tout à fait satisfaisantes.
numérotation des points :



m=2

numéro	0	1	2	3	4	5	6	7
classe 0:	0.94897	0.97309	0.99646	0.98935	0.96168	0.98806	0.95353	0.04574
classe 1:	0.05103	0.02691	0.00354	0.01065	0.03832	0.01194	0.04647	0.95426

numéro	8	9	10	11	12	13	*14*	*15*
classe 0:	0.00840	0.02887	0.00584	0.00430	0.09515	0.25325	*0.55586*	*0.55586*
classe 1:	0.99160	0.97113	0.99416	0.99570	0.90485	0.74675	*0.44414*	*0.44414*



✔ Les classes sont correctes et les points situés sur la médiatrice des classes n'appartiennent pas plus à l'une qu'à l'autre classe.

✗ La valeur d'appartenance des deux points de bruit est la même pour les deux classes alors que l'un des deux est plus loin des classes que l'autre.

➽ *Cet algorithme n'est donc pas adapté pour nuancer l'appartenance dans le cas de configurations "bruitées".*

2.2 - Les Fuzzy C-Covariance

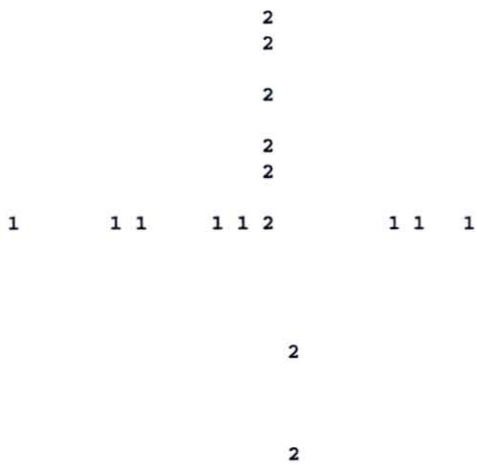
En utilisant une métrique différente, on n'impose plus aux classes d'être sphérique.

2.2.1 - Fuzzy C-Covariance sur la croix de Gustafson

Avec l'algorithme de Fuzzy C- et une métrique utilisant la covariance, il est possible de détecter des classes non sphériques. Par exemple sur la configuration précédente, on obtient la répartition des points comme suit :

numéro	0	1	2	3	4	5	6	7	8	9
classe 0:	0.99789	0.99701	0.99370	0.98504	0.95129	0.23619	0.99549	0.99563	0.99811	0.99778
classe 1:	0.00211	0.00299	0.00630	0.01496	0.04871	0.76381	0.00451	0.00437	0.00189	0.00222

numéro	10	11	12	13	14	15	16	17	18	19
classe 0:	0.00122	0.00277	0.00584	0.00073	0.00824	0.00196	0.01315	0.16879	0.00387	0.00260
classe 1:	0.99878	0.99723	0.99416	0.99927	0.99176	0.99804	0.98685	0.83121	0.99613	0.99740



✔ *Chaque branche de la croix a bien été identifiée comme une classe.*

2.2.2 - Fuzzy C-Covariance sur des ensembles bruités

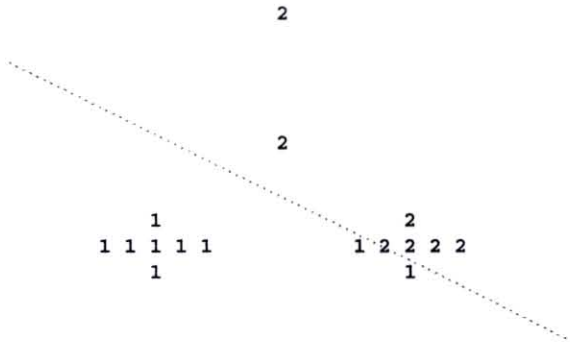
✘ *Sur les deux ensembles vus précédemment, la modification apportée par le changement de métrique n'est pas robuste au bruit.*

Valeurs d'appartenance des points :
m=2

numéro	0	1	2	3	4	5	6	7
classe 0:	0.99920	0.99953	0.99514	0.99981	0.99585	0.99998	0.99990	0.77725
classe 1:	0.00080	0.00047	0.00486	0.00019	0.00415	0.00002	0.00010	0.22275

numéro	8	9	10	11	12	13	14	15
classe 0	0.08069	0.61937	0.47686	0.13418	0.02500	0.01856	0.00732	0.00732
classe 1	0.91931	0.38063	0.52314	0.86582	0.97500	0.98144	0.99268	0.99268

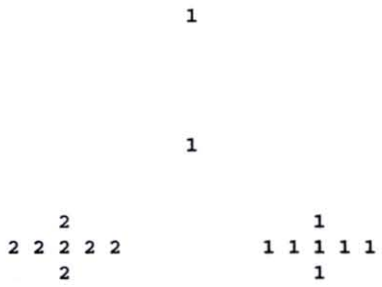
La structure des classes est complètement perturbée par le bruit.



En changeant les volumes de classe, le bruit entre entièrement dans la classe la plus volumineuse.

numéro	0	1	2	3	4	5	6	7
classe 0:	0.00230	0.00078	0.00688	0.00001	0.00590	0.00071	0.00423	0.70526
classe 1:	0.99770	0.99922	0.99312	0.99999	0.99410	0.99929	0.99577	0.29474

numéro	8	9	10	11	12	13	14	15
classe 0:	0.98182	0.85237	0.92400	0.97792	0.98328	0.97709	0.98658	0.98658
classe 1:	0.01818	0.14763	0.07600	0.02208	0.01672	0.02291	0.01342	0.01342



➽ ***Cet algorithme n'est donc pas non plus adapté pour des configurations "bruitées".***

2.3 - PC-M sur des ensembles bruités

2.3.1 - Les deux ensembles

avec m=2, les classes sont regroupées.
Les classes sont centrées en :

c 0	c 1
15.372	15.375
11.000	11.000

numér	0	1	2	3	4	5	6	7
classe	0.9997	0.9998	0.0000	0.9999	0.0000	0.9999	0.9999	0.9998
classe	1.0000	1.0000	0.0000	1.0000	0.0000	1.0000	1.0000	1.0000

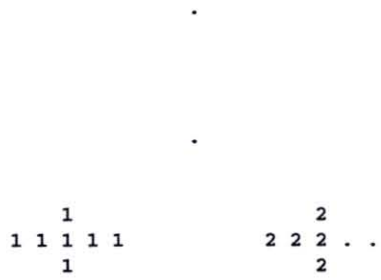
numér	8	9	10	11	12	13	14	15
classe	0.9997	0.0000	0.9996	0.0000	0.0000	0.0000	0.0000	0.0000
classe	1.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000

avec m=1.25, les classes sont séparées.

numér	0	1	2	3	4	5	6	7
classe	0.99936	1.00000	1.00000	1.00000	1.00000	1.00000	0.99946	0.02566
classe	0.00011	0.00022	0.00047	0.00050	0.00048	0.00122	0.00333	0.99933

numér	8	9	10	11	12	13	14	15
classe	0.01014	0.00421	0.00438	0.00421	0.00124	0.00017	0.00096	0.00096
classe	1.00000	0.99978	1.00000	0.99997	0.26943	0.00062	0.00007	0.00007

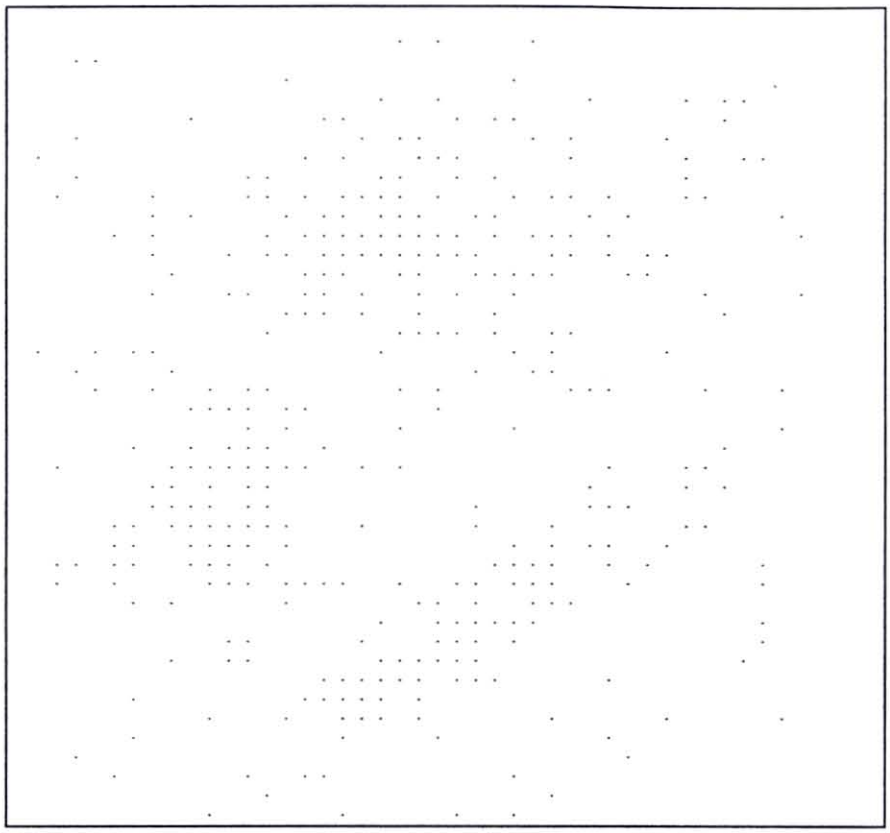
La répartition des points, en considérant qu'un point appartient à une classe si sa valeur d'appartenance est supérieure à 0.5, est la suivante :



✔ ***Les classes ne sont pas parfaites, mais le bruit a été rejeté.***

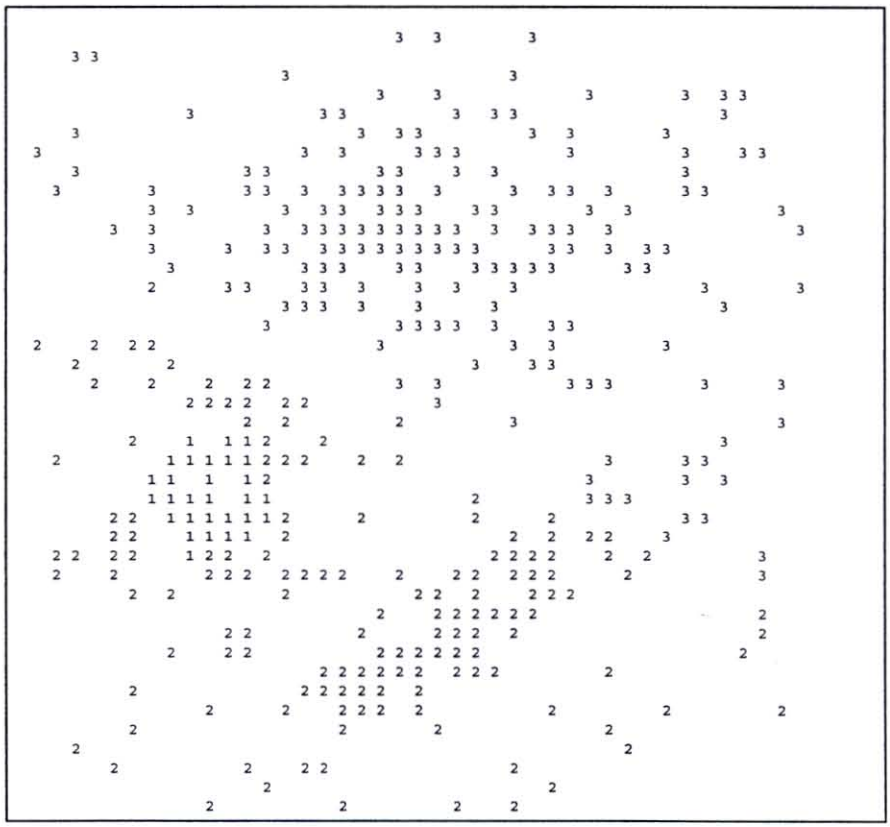
2.3.2 - Autre exemple

Soient la répartition de points dans l'espace suivante :

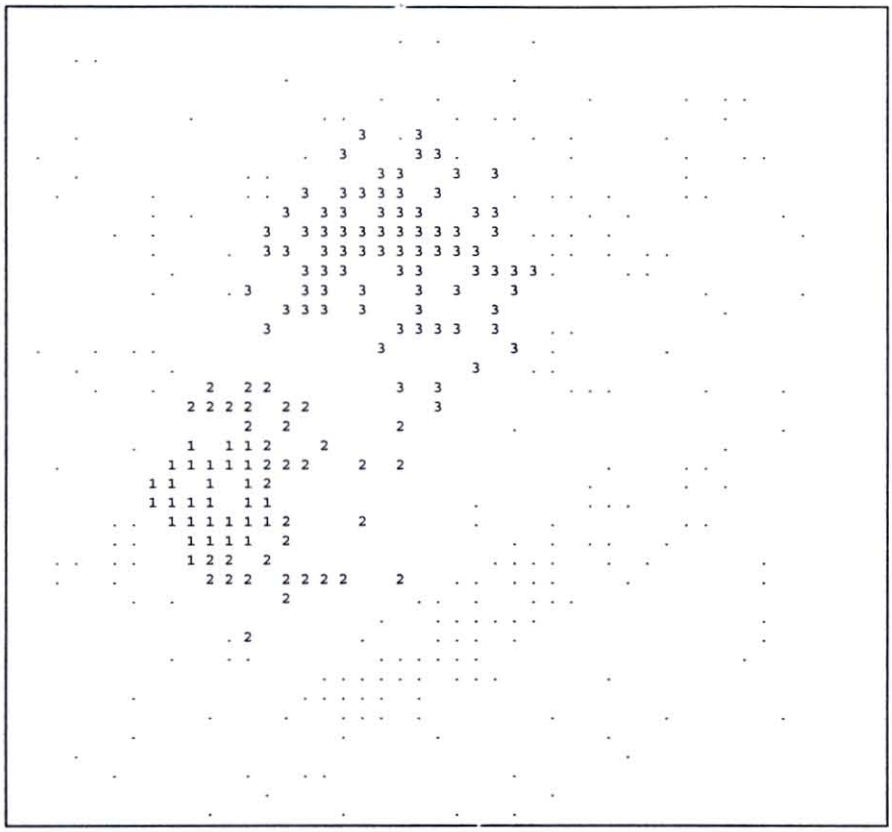


Recherche de 3 classes :

Si on attribue chaque point à la classe dont il est le plus proche, on obtient la répartition suivante :



En considérant qu'un point est "bruit" si son appartenance est inférieure à 0.5, on obtient la répartition suivante :

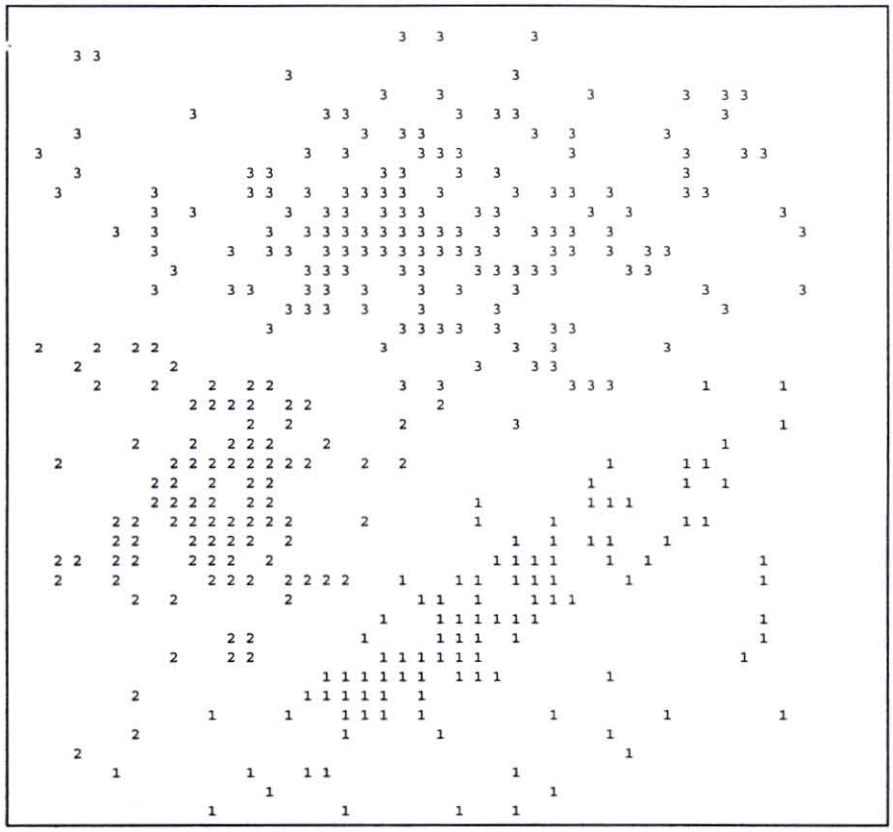


✘ La répartition n'est pas très bonne, cependant, deux des trois classes ont été retrouvées.

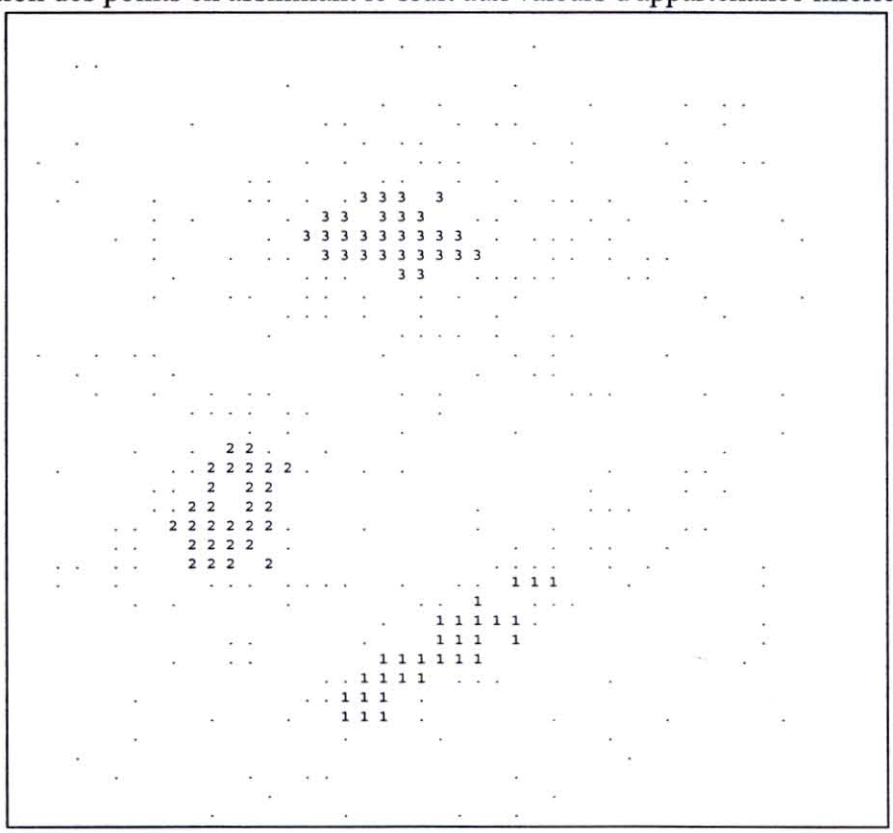
2.4 - Possibilistic C-Covariance

✔ En utilisant la méthode de Kessel-Gustafson pour le calcul de la métrique, avec des volumes de classe appropriés, les classes sont correctes.

Répartition des points (en considérant le maximum) :



Répartition des points en assimilant le bruit aux valeurs d'appartenance inférieures à 0.5:



2.5 - OHASHI

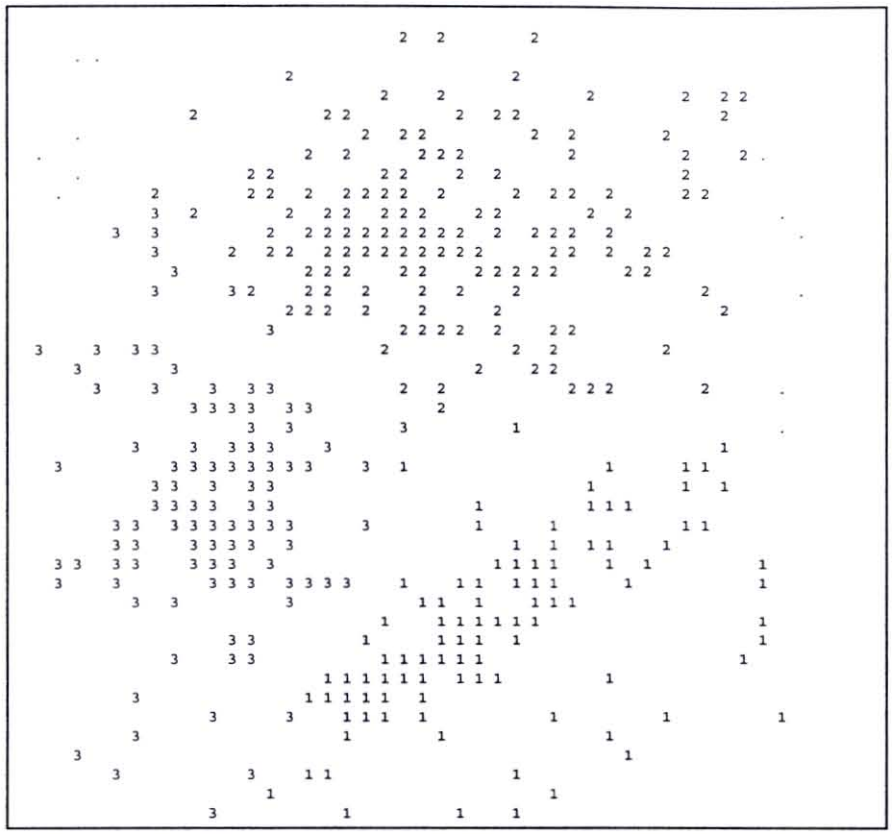
Utilisation de l'algorithme de OHASHI avec la métrique euclidienne pour retrouver 3 classes de points plongées dans du bruit.

✔ ***Les classes sont bien centrées, mais leur forme ne correspond pas très bien aux données.***

✔ ***En utilisant le même algorithme, mais avec une métrique qui tient compte de la dispersion des points sur les axes, les formes des classes sont meilleures :***

2.6 - DAVE

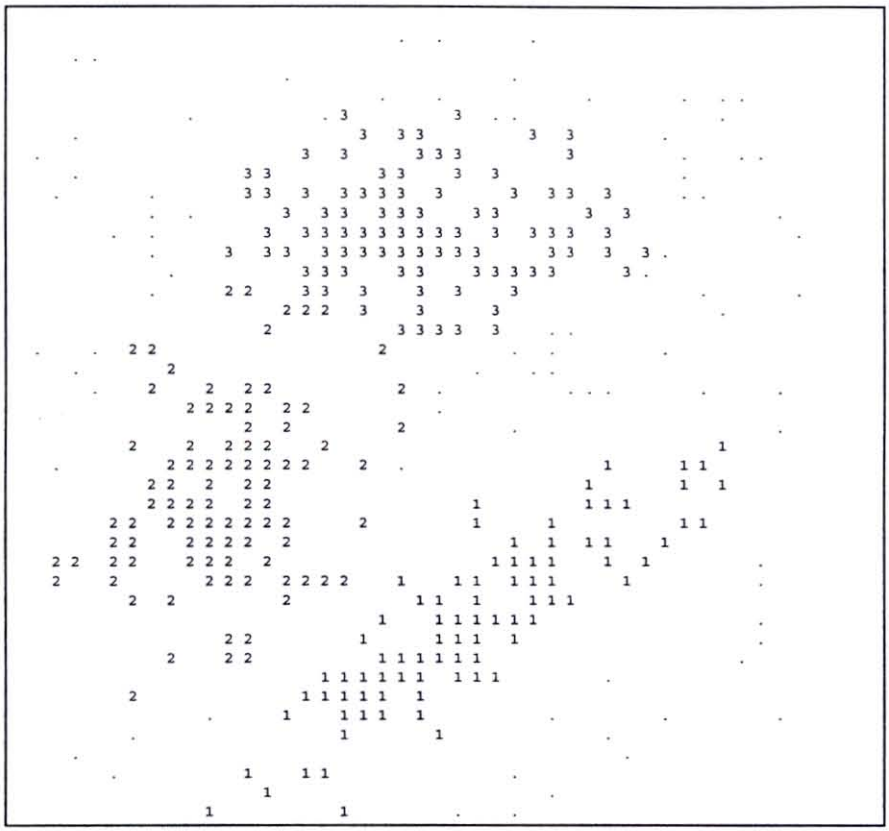
Utilisation de l'algorithme de DAVE avec la métrique euclidienne pour retrouver les trois classes, et avec $\lambda=1$.



En prenant λ=0.2, la classe de bruit est beaucoup plus importante.

✘ ***Les classes sont sphériques.***

En utilisant la dispersion autour des axes et une valeur de λ suffisamment faible pour marquer la classe de bruit (λ=0.2)



✔ ***Les classes sont correctement positionnées et le bruit correctement isolé.***

3. - ACP sur les données.

C'est à l'aide du logiciel SAS qu'une analyse en composantes principales a été effectuée sur les 15 variables retenues pour la classification, en considérant l'ensemble des 800 cellules initialement mesurées

Le script de commande est le suivant :

```
libname basstat 'q:\classifi\sas';
data basstat.data;
   infile 'q:\classifi\sas\data';
input v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15;

proc princomp out=prin;
run;
```

Les résultats obtenus sont :

Principal Component Analysis

806 Observations
15 Variables

Simple Statistics

	V1	V2	V3	V4	V5
Mean	5105.636476	1103.043424	0.2811815782	120.7051146	0.3528873598
StD	2925.190172	292.795677	0.1536137870	26.4950351	0.8127251847

	V6	V7	V8	V9	V10
Mean	2.811941731	0.4769977605	0.1139023313	2.746405917	0.0019044194
StD	1.422919526	0.1122027521	0.0506835982	0.340276469	0.0006572230

	V11	V12	V13	V14	V15
Mean	0.5076005670	24.12304319	0.1511002270	0.2583394739	66.45672385
StD	0.0764115510	14.47492533	0.0294984297	0.0614093260	14.52722184

Correlation Matrix

	V1	V2	V3	V4	V5	V6	V7	V8
V1	1.0000	-.0046	-.7410	-.5029	0.6178	0.6147	-.5096	0.1077
V2	-.0046	1.0000	0.4152	0.0587	-.3575	-.2121	0.0587	-.1355
V3	-.7410	0.4152	1.0000	0.6002	-.8049	-.3817	0.6072	-.0193
V4	-.5029	0.0587	0.6002	1.0000	-.6726	-.2420	0.9998	-.2691
V5	0.6178	-.3575	-.8049	-.6726	1.0000	0.4834	-.6768	0.2956
V6	0.6147	-.2121	-.3817	-.2420	0.4834	1.0000	-.2472	0.5605
V7	-.5096	0.0587	0.6072	0.9998	-.6768	-.2472	1.0000	-.2683
V8	0.1077	-.1355	-.0193	-.2691	0.2956	0.5605	-.2683	1.0000
V9	-.1032	0.1354	0.0784	0.3814	-.3188	-.4653	0.3803	-.9276
V10	-.2738	0.1220	0.4061	0.5993	-.5589	-.2566	0.6005	-.5640
V11	-.3437	0.0926	0.4715	0.6109	-.5347	-.2519	0.6130	-.4011
V12	0.2887	-.2000	-.2839	-.3605	0.4918	0.5422	-.3621	0.8115
V13	0.4073	-.2231	-.4598	-.6775	0.5370	0.5051	-.6787	0.6228
V14	-.3488	0.1015	0.4934	0.6172	-.5864	-.2770	0.6198	-.4426
V15	-.1772	0.1361	0.2563	0.4373	-.4743	-.3621	0.4381	-.7405

Correlation Matrix

	V9	V10	V11	V12	V13	V14	V15
V1	-.1032	-.2738	-.3437	0.2887	0.4073	-.3488	-.1772
V2	0.1354	0.1220	0.0926	-.2000	-.2231	0.1015	0.1361
V3	0.0784	0.4061	0.4715	-.2839	-.4598	0.4934	0.2563
V4	0.3814	0.5993	0.6109	-.3605	-.6775	0.6172	0.4373
V5	-.3188	-.5589	-.5347	0.4918	0.5370	-.5864	-.4743
V6	-.4653	-.2566	-.2519	0.5422	0.5051	-.2770	-.3621
V7	0.3803	0.6005	0.6130	-.3621	-.6787	0.6198	0.4381
V8	-.9276	-.5640	-.4011	0.8115	0.6228	-.4426	-.7405
V9	1.0000	0.7447	0.6143	-.8113	-.7294	0.6502	0.8572
V10	0.7447	1.0000	0.8827	-.7141	-.5816	0.9257	0.8964
V11	0.6143	0.8827	1.0000	-.5983	-.5878	0.9762	0.8008
V12	-.8113	-.7141	-.5983	1.0000	0.5235	-.6334	-.8655
V13	-.7294	-.5816	-.5878	0.5235	1.0000	-.5921	-.5384
V14	0.6502	0.9257	0.9762	-.6334	-.5921	1.0000	0.8490
V15	0.8572	0.8964	0.8008	-.8655	-.5384	0.8490	1.0000

Eigenvalues of the Correlation Matrix

	Eigenvalue	Difference	Proportion	Cumulative
PRIN1	8.05526	5.63296	0.537017	0.53702
PRIN2	2.42229	0.91067	0.161486	0.69850
PRIN3	1.51163	0.43948	0.100775	0.79928
PRIN4	1.07215	0.24619	0.071476	0.87075
PRIN5	0.82596	0.42873	0.055064	0.92582
PRIN6	0.39723	0.18770	0.026482	0.95230
PRIN7	0.20953	0.01516	0.013969	0.96627
PRIN8	0.19437	0.08343	0.012958	0.97923
PRIN9	0.11093	0.04201	0.007396	0.98662
PRIN10	0.06892	0.00476	0.004595	0.99122
PRIN11	0.06416	0.02148	0.004277	0.99549
PRIN12	0.04268	0.02971	0.002846	0.99834
PRIN13	0.01298	0.00117	0.000865	0.99920
PRIN14	0.01181	0.01170	0.000787	0.99999
PRIN15	0.00011	.	0.000008	1.00000

Eigenvectors

	PRIN1	PRIN2	PRIN3	PRIN4	PRIN5	PRIN6	PRIN7	PRIN8
V1	-.185859	-.382070	0.226696	0.329818	0.356322	-.115265	0.300150	-.300243
V2	0.082793	0.085364	-.371118	0.763003	0.279270	0.103731	0.125951	0.296324
V3	0.214490	0.442460	-.113515	0.210520	-.146141	0.031357	-.418104	-.003014
V4	0.265249	0.264452	0.234694	-.167369	0.371025	-.209010	0.199759	0.236638
V5	-.265730	-.291581	0.121611	-.142861	0.057576	0.412525	-.097009	0.725352
V6	-.195344	-.027200	0.565586	0.270960	0.168568	-.189898	-.632064	-.002616
V7	0.266148	0.267266	0.231937	-.167976	0.363710	-.205353	0.195154	0.232390
V8	-.238144	0.370538	0.265571	0.155991	-.188508	0.164457	0.140151	0.069076
V9	0.276715	-.361958	-.081814	-.075799	0.175323	0.105562	-.150265	-.044290
V10	0.310264	-.123865	0.243971	0.132890	-.149998	-.012739	-.012633	-.012151
V11	0.298290	-.031472	0.300271	0.123029	-.230458	0.357362	0.179186	0.048270
V12	-.280753	0.249087	0.179362	-.018097	0.170196	0.419812	0.202484	-.317873
V13	-.282342	-.014809	0.094158	0.147847	-.454942	-.503799	0.307333	0.256528
V14	0.309144	-.041440	0.279954	0.132492	-.243396	0.247769	0.149081	-.084811
V15	0.299998	-.274250	0.090135	0.109801	-.186942	-.159288	0.012746	0.051367

	PRIN9	PRIN10	PRIN11	PRIN12	PRIN13	PRIN14	PRIN15
V1	-.324381	0.468797	0.005111	-.098356	0.042794	0.047809	-.003016
V2	0.141470	-.221877	-.007123	0.020446	-.036865	-.049397	0.004705
V3	-.103922	0.635392	0.255565	-.030665	0.120160	0.092343	-.010760
V4	0.009946	0.019952	0.045157	0.023478	0.029138	-.006018	-.703844
V5	0.009132	0.300669	0.040575	-.000121	-.075292	-.048217	-.002443
V6	0.046301	-.296322	0.106933	0.009656	-.020643	-.043391	0.004261
V7	0.005055	0.038083	0.045179	0.022392	0.023899	-.016520	0.710152
V8	-.213382	-.007216	-.578803	0.320249	0.369127	0.068706	0.000749
V9	0.216364	-.031580	0.046379	0.146210	0.798646	0.065974	0.005078
V10	0.544463	0.225068	-.510924	-.382282	-.143013	0.116237	-.001193
V11	-.293366	-.276814	0.293733	-.266047	0.021852	0.518866	0.005384
V12	0.560635	0.080575	0.311339	0.210341	-.060537	0.117208	0.001233
V13	0.258897	0.045904	0.334737	-.089483	0.281047	0.010420	0.002529
V14	-.057299	0.002329	0.151421	0.041461	0.024138	-.792568	-.007177
V15	0.046270	0.101409	0.032443	0.770062	-.310807	0.220030	0.000152

4. - Critères de validation du nombre de classes

Afin de valider le nombre de classes utilisées lors de la classification, plusieurs critères ont été mesurés. Cinq des sept critères mesurés sont spécifiques à la classification floue :

•Coefficient de partition : $F(U,C)=\frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{C}U_{ik}^2$ à maximiser.

La valeur de ce coefficient est comprise entre $\frac{1}{C}$ et 1; elle augmente avec les grandes valeur de U_{ik}. Plus il existe un U_{ik} proche de 1 pour un k donné, plus $\sum_{i=1}^{C}U_{ik}^2$ est grand.

• Entropie de classification : $H(U,C)=-\frac{1}{N}\sum_{k}\sum_{i}U_{ik}\log_a(U_{ik})$

$$H2(U,C)=\frac{H(U,C)}{\log_a(C)}$$ à minimiser.

Ce paramètre représente le désordre de la classification.

• Max : $P1(U,C)=\frac{1}{N}\sum_{k}\max_{l}U_{ik}$ à maximiser.

Ce paramètre représente l'appartenance moyenne d'un point à la classe dont il est le plus proche.

• Min-Max : $P2(U,C)=\frac{1}{N}\sum_{k}\frac{\min_{i}U_{ik}}{\max_{l}U_{ik}}$ à minimiser.

Ce paramètre donne une information quant à la forte appartenance d'un point à la classe dont il est le plus proche, et la faible appartenance de ce point à la classe dont il est le plus loin.

les deux autres sont des critères géométriques :

• Hypervolume flou : $FHV(U,C)=\sum_{i}\det(\sum_{i})^{1/2}$ à minimiser

Σi : matrice de covariance floue

Ce paramètre donne une information sur la compacité des classes.

• Compacité-séparabilité : $CS(U,C)=\frac{\frac{1}{N}\sum_{i}\sum_{k}(U_{ik}\|X_k-V_i\|)^2}{\min_{i,j,j\neq i}\|V_i-V_j\|^2}$ à minimiser

Ce paramètre donne une information sur la compacité et la séparabilité des classes.

Valeurs des critères selon les différents algorithmes :

Algorithme des Fuzzy-C Means

critère / classe	F	H	H2	P1	P2	FHV	CS
2	0,702991	0,457451	0,659963	0,879759	0,273768	5,236068	989,1609
3	0,538459	0,785034	0,714569	0,829076	0,150395	6,236068	2426,603
4	0,512802	0,895588	0,64603	0,762726	0,067051	7,236068	3194,663
5	0,422177	1,121457	0,696801	0,686577	0,051651	8,236068	3489,268
6	0,369693	1,266552	0,706876	0,631452	0,024762	9,236068	578,8625
7	0,33507	1,387164	0,712861	0,601515	0,014964	10,23606	441,4043
8	0,299115	1,526927	0,734297	0,573909	0,014057	11,23606	1350,772
9	0,283687	1,594006	0,725463	0,523922	0,012129	12,23606	332,4821
10	0,263634	1,700053	0,738324	0,500029	0,011897	13,23606	282,7375
11	0,243985	1,786797	0,745152	0,47553	0,008626	14,23606	215,9593
12	0,236407	1,837908	0,739628	0,461542	0,006426	15,23606	377,2951
13	0,238088	1,85983	0,725094	0,451997	0,003156	16,23606	126,3516
14	0,224363	1,938246	0,734446	0,441883	0,001955	17,23606	138,9194
15	0,211484	2,000806	0,738836	0,406346	0,001173	18,23606	764,6596
16	0,206961	2,058347	0,742392	0,406706	0,001021	19,23606	1072,184
17	0,201199	2,094677	0,739329	0,381023	0,000997	20,23606	629,6239
Moyonne	0,330632	1,519427	0,718753	0,563374	0,040251	12,73606	1025,684
Ecart-type	0,144943	0,493398	0,029319	0,156011	0,073150	4,760952	1074,405

Algorithme de OHASHI

critère / classe	F	H	H2	P1	P2	FHV	CS
2	0,703906	0,45796	0,660697	0,871216	0,267656	2	1587,562
3	0,566018	0,735492	0,669474	0,783526	0,133375	3	1176,396
4	0,501463	0,909261	0,655893	0,759859	0,057171	4	612,4532
5	0,441261	1,082446	0,672562	0,709337	0,03406	5	645,0423
6	0,381436	1,251696	0,698585	0,657742	0,023059	6	2199,245
7	0,337901	1,384884	0,711689	0,587717	0,02164	7	4236,378
8	0,306758	1,501053	0,721854	0,532763	0,015108	8	3654,281
9	0,276486	1,617349	0,736087	0,519104	0,014428	9	933,8788
10	0,262075	1,696201	0,736651	0,484625	0,010053	10	241,7605
11	0,251343	1,766869	0,736842	0,483713	0,007569	11	344,1760
12	0,243262	1,820798	0,732743	0,472871	0,004484	12	141,6761
13	0,2289	1,884932	0,734881	0,426964	0,002702	13	741,4477
14	0,210263	1,974456	0,748167	0,410215	0,005463	14	129,1601
15	0,217895	1,983934	0,732606	0,406468	0,00173	15	453,2063
16	0,217768	2,00744	0,724031	0,414027	0,000977	16	1665,839
17							
M	0,343115	1,471651	0,711517	0,568009	0,039965	9	1250,833
E	0,148532	0,489012	0,031683	0,152349	0,071491	4,472135	1252,995

Algorithme de DAVE

critère / classe	F	H	H2	P1	P2	FHV	CS
2	0,495645	0,540338	0,779543	0,764388	0,265541	2	1256,196
3	0,503598	0,725636	0,660503	0,748461	0,125191	3	6554,914
4	0,475015	0,89666	0,646804	0,742605	0,066033	4	509,6244
5	0,408008	1,095395	0,680607	0,689363	0,036715	5	1013,765
6	0,346647	1,284978	0,71716	0,612155	0,02777	6	695,9552
7	0,328085	1,371648	0,704888	0,587053	0,018636	7	3441,747
8	0,281289	1,555848	0,748205	0,521004	0,016154	8	504,8898
9	0,277714	1,592591	0,724819	0,512462	0,011875	9	306,5650
10	0,25606	1,691638	0,734669	0,480452	0,009126	10	339,6070
11	0,237088	1,784281	0,744103	0,443847	0,00775	11	521,9407
12	0,243415	1,810197	0,728477	0,45195	0,004746	12	860,3645
13	0,230735	1,888785	0,736383	0,447127	0,002951	13	304,3501
14	0,214171	1,960555	0,7429	0,396639	0,002387	14	570,6662
15	0,207129	2,003355	0,739778	0,391266	0,003934	15	790,5799
16	0,217094	2,006931	0,723847	0,423236	0,001218	16	600,5569
17							
Moyenne	0,314779	1,480589	0,720845	0,547467	0,040001	9	1218,114
Ecart-type	0,106674	0,478422	0,034955	0,133722	0,070512	4,472135	1664,463

Valeurs centrées réduites pour affichage sur un même graphique :

Algorithme des Fuzzy-C Means

critère / classe	F	H	H2	P1	P2	FHV	CS
2	2,568997	-2,152369	-2,005186	2,027960	3,192285	-1,575315	-0,033993
3	1,433849	-1,488437	-0,142731	1,703092	1,505714	-1,365273	1,303901
4	1,256835	-1,264371	-2,480400	1,277802	0,366359	-1,155231	2,018770
5	0,631590	-0,806590	-0,748747	0,789702	0,155833	-0,945189	2,292973
6	0,269490	-0,512517	-0,405118	0,436362	-0,211752	-0,735147	-0,415878
7	0,030617	-0,268066	-0,200986	0,244472	-0,345696	-0,525105	-0,543816
8	-0,217445	0,015199	0,530133	0,067524	-0,358095	-0,315063	0,302575
9	-0,323886	0,151152	0,228831	-0,252882	-0,384452	-0,105021	-0,645195
10	-0,462237	0,366083	0,667483	-0,406031	-0,387623	0,105021	-0,691495
11	-0,597800	0,541892	0,900366	-0,563065	-0,432340	0,315063	-0,753649
12	-0,650083	0,645482	0,711958	-0,652725	-0,462415	0,525105	-0,603486
13	-0,638485	0,689912	0,216245	-0,713907	-0,507117	0,735147	-0,837051
14	-0,733177	0,848843	0,535215	-0,778735	-0,523536	0,945189	-0,825353
15	-0,822033	0,975637	0,684945	-1,006520	-0,534226	1,155231	-0,242947
16	-0,853238	1,092258	0,806230	-1,004213	-0,536304	1,365273	0,043280
17	-0,892992	1,165890	0,701760	-1,168836	-0,536632	1,575315	-0,368632

Algorithme de OHASHI

critère / classe	F	H	H2	P1	P2	FHV	CS
2	2,429033	-2,072936	-1,604012	1,990201	3,184852	-1,565247	0,268738
3	1,500697	-1,505400	-1,326989	1,414617	1,306582	-1,341640	-0,059407
4	1,066079	-1,150053	-1,755637	1,259270	0,240670	-1,118033	-0,509483
5	0,660766	-0,795901	-1,229525	0,927651	-0,082596	-0,894427	-0,483474
6	0,257993	-0,449795	-0,408178	0,588989	-0,236474	-0,670820	0,756915
7	-0,035107	-0,177434	0,005413	0,129355	-0,256322	-0,447213	2,382725
8	-0,244779	0,060124	0,326245	-0,231354	-0,347689	-0,223606	1,918161
9	-0,448586	0,297942	0,775471	-0,321010	-0,357201	0	-0,252957
10	-0,545609	0,459190	0,793272	-0,547325	-0,418397	0,223606	-0,805328
11	-0,617862	0,603701	0,799301	-0,553312	-0,453142	0,447213	-0,723591
12	-0,672268	0,713983	0,669927	-0,624477	-0,496294	0,670820	-0,885204
13	-0,768961	0,845133	0,737407	-0,925804	-0,521220	0,894427	-0,406534
14	-0,894435	1,028204	1,156744	-1,035742	-0,482600	1,118033	-0,895193
15	-0,843052	1,047586	0,665603	-1,060337	-0,534816	1,341640	-0,636576
16	-0,843907	1,095654	0,394956	-1,010720	-0,545348	1,565247	0,331210
17							

Algorithme de DAVE

critère / classe	F	H	H2	P1	P2	FHV	CS
2	1,695484	-1,965314	1,679222	1,622168	3,198559	-1,565247	0,022879
3	1,770037	-1,578003	-1,726296	1,503063	1,208138	-1,341640	3,206318
4	1,502092	-1,220529	-2,118199	1,459271	0,369170	-1,118033	-0,425656
5	0,873950	-0,805133	-1,151157	1,061119	-0,046612	-0,894427	-0,122772
6	0,298734	-0,408866	-0,105442	0,483745	-0,173469	-0,670820	-0,313710
7	0,124729	-0,227708	-0,456521	0,296028	-0,303006	-0,447213	1,335945
8	-0,313949	0,157306	0,782699	-0,197896	-0,338205	-0,223606	-0,428501
9	-0,347462	0,234106	0,113667	-0,261774	-0,398889	0	-0,547653
10	-0,550453	0,441134	0,395458	-0,501150	-0,437875	0,223606	-0,527802
11	-0,728302	0,634777	0,665348	-0,774888	-0,457389	0,447213	-0,418257
12	-0,668991	0,688946	0,218316	-0,714292	-0,499991	0,670820	-0,214934
13	-0,787857	0,853211	0,444493	-0,750360	-0,525448	0,894427	-0,548984
14	-0,943132	1,003225	0,630932	-1,127917	-0,533446	1,118033	-0,388983
15	-1,009146	1,092686	0,541617	-1,168097	-0,511507	1,341640	-0,256860
16	-0,915731	1,100160	0,085860	-0,929020	-0,550025	1,565247	-0,371025
17							

D'après les valeurs mesurées pour ces différents critères, le nombre de classes le mieux adapté aux données est variable selon l'algorithme utilisé.

5. - Le classement

5.1 - Mode d'emploi

Mode d'emploi
de l'application de comptage
des trois types hémocytaires
de l'huître plate
sur lames histologiques.

PARTIE I - L'ACQUISITION D'IMAGES

A _ Allumer l'imprimante.

B _ Créer un répertoire* sur le disque pour stocker les images (Créer un répertoire par lame histologique à analyser).
Assurez-vous qu'il y a suffisamment de place sur le disque pour sauvegarder les images.

C _ Sous Windows exécuter IPS. Ouvrir le programme ***acq_auto.sam*** situé dans le répertoire C:\ANGELI\SCRIPTS\ à l'aide de la commande OUVRIR du menu FICHIER. Exécuter le programme avec la commande EXECUTER du menu PROGRAMME ou avec l'icône raccourci.

D _ Déroulement de la procédure d'acquisition :

1- Répertoire d'acquisition

Donner le chemin complet du répertoire précédemment créé et valider (OK). Ne pas oublier le '\' final.*

2 - Acquérir un champs vide ?

choisir OUI

3 - Réglage du numériseur

Régler le microscope :

- *Diaphragmes ouverts au maximum*
- *Positionner le condenseur*
- *Positionner le filtre vert sur la lampe*
- *Régler l'intensité lumineuse pour obtenir le meilleur contraste possible entre les cellules et le fond de l'image (observer l'écran de contrôle).*

Présenter un champs vide (ne contenant pas de cellule ou de fragments de tissus cellulaire).

Choisir OK.

4 - Le champs vide est-il correct ?

Choisir OUI si le champs vide est correct (sans cellule ou fragments de cellule), sinon choisir NON et reprendre au point 3.

L'image de référence est sauvegardée dans le répertoire avec le nom "IMVIDE.BMP".

ATTENTION : Une fois cette étape effectuée, il ne faut plus toucher au réglage des diaphragmes et de l'intensité lumineuse du microscope.

5 - Acquérir les images

Choisir OK

6 - Premier indice de fichier

Saisir un indice (laisser 0 par défaut)

7 - Réglage du numériseur

Déplacer la lame sous l'objectif de la caméra, régler la mise au point sur les cellules et choisir OK (taper ENTREE).

Ne pas changer la luminosité.

Une image vient d'être sauvegardée sur le disque.

Cette opération sera répétée autant de fois qu'il y aura d'images à sauvegarder sur le disque.

8 - Après trois acquisitions, un contrôle peut être effectué : choisir OUI pour **effectuer un contrôle**. Les images acquises et les images binaires correspondantes sont affichées sur l'écran de contrôle. On doit pouvoir observer la forme entière des cellules dans les images binaires. Si les cellules sont très érodées dans les trois images, c'est soit parce que les réglages initiaux du microscope ne sont pas suffisamment bons (intensité lumineuse à revoir), soit parce que la coloration de la lame est trop faible. (Cette opération dure environ 3x30 secondes.)

9 - Continuer les acquisitions

Choisir OUI si les résultats du contrôle sont satisfaisants

Choisir NON sinon.

Si la réponse est OUI, continuer à acquérir les images comme précédemment (étape 7).

Pour arrêter l'acquisition, choisir ANNULER.

Si la réponse est NON, plusieurs possibilités se présentent.

a - Recommencer l'acquisition à partir du champs libre (étape 2) soit sur la même lame, en changeant l'intensité lumineuse pour augmenter le contraste, soit sur une autre lame.

b - Continuer pour contrôler trois autres images.

Dans ce cas, choisir NON pour l'acquisition d'un champs vide (les trois premières images sont conservées).

L'acquisition continue comme à l'étape 7, et après trois sauvegarde, le contrôle 8 peut de nouveau être effectué.

La trace d'exécution (fenêtre OUTPUT) permet de retenir les noms de répertoire ainsi que le nombre d'images acquises pour chaque lame. Ces renseignements seront nécessaires pour la partie analyse&classement.

IMPORTANT : Il faut créer un répertoire par lame. L'image de référence est sauvegardée sous le nom unique de IMVIDE.BMP quelque soit la lame. Il est donc impératif de la conserver.

PARTIE II - L'ANALYSE D'IMAGES ET LE CLASSEMENT

A _ L'imprimante doit toujours être allumée durant le traitement.

B _ Une fois les images des lames enregistrées, ouvrir l'application ***ai_param.sam*** du répertoire C:\ANGELI\SCRIPT\ avec la commande OUVRIR du menu FICHIER.

C _ Exécuter l'application avec la commande EXECUTER du menu PROGRAMME, ou en cliquant sur l'icône de raccourci.

D _ Déroulement du programme :

1 - Nombre de lames

Saisir le nombre de lames histologiques utilisées lors de la phase d'acquisition d'images et valider.

2 - Pour chaque lame :

2.1 - Chemin du répertoire de travail

Saisir le chemin du répertoire créé pour la première lame (par exemple : C:\ANGELI\LAME1

Ne pas oublier le \ final).

2.2 - Premier indice d'image

Saisir le numéro de la première image (par défaut 0) puis valider.

3- Dernier indice d'image

Saisir le numéro de la dernière image acquise puis valider.

4 - Nom du fichier de sauvegarde OUTPUT

Saisir le nom du fichier (avec son chemin) de sauvegarde des résultats de classement puis valider.

5 - Nom du fichier de sauvegarde des résultats d'analyse

Saisir le nom du fichier (avec son chemin) de sauvegarde des mesures effectuées sur les cellules.

Le programme se déroule seul.

Le temps de traitement est long, environ 2 minutes par image.

ANNEXE

Comment créer un répertoire ?

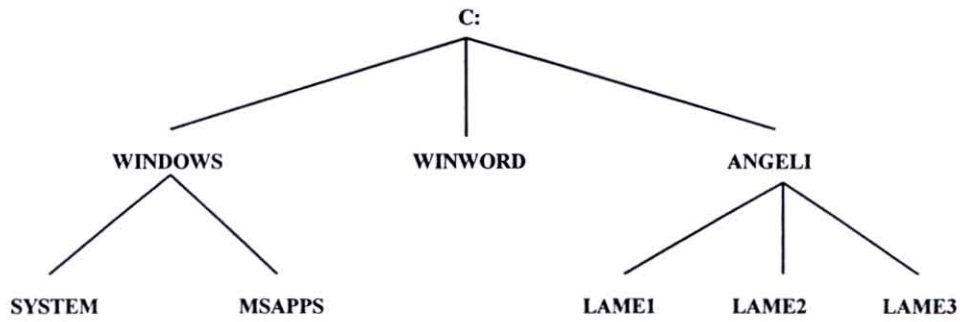
Dans le gestionnaire de fichier sous Windows, sélectionner un répertoire quelconque dans l'arborescence (C:\, C:\WINDOWS, ...).

Dans le menu FICHIER, choisir CREER UN REPERTOIRE. Saisir un nom, par exemple "LAME1" et valider (taper ENTREE). Le répertoire LAME1 est créé dans le répertoire sélectionné.

Si le répertoire sélectionné au départ est C:\, on dit que le nouveau répertoire se situe à la racine.

Qu'est-ce que le chemin d'un répertoire ?

Comme pour se rendre chez quelqu'un il faut emprunter un chemin, pour accéder à un répertoire, il faut parcourir toute l'arborescence de répertoires située "en amont" du répertoire demandé.



Ici, le chemin du répertoire LAME1 est : C:\ANGELI\LAME1\.

5.2 - Programme d'acquisition

```
Text     chemin, vide, image, titre, indice;
Int      premier, i, j, ressaisir, tempint, seuil;
Condition yesno,redebut;

redebut:=TRUE;
chemin:="C:\ANGELI\LAME\";
PromptText("chemin","repertoire d'acquisition",chemin);
WriteText(chemin);Newline;

vide:=Concat(chemin,"IMVIDE.BMP");

{/*/*/*/ image vide /*/*/*/}
YesNo("Acqu,rir un champ vide.","Champs Vide",yesno);
if yesno then
   ressaisir:=1;
else
   ressaisir:=0;
endif;
while ressaisir=1 do
   user;
   VideoIn(0,42,0);
   auto;
   YesNo("Champs vide correct ?","Champs vide",yesno);
   if yesno then
      ressaisir:=2;
   endif;
endwhile;
if ressaisir=2 then
   SaveImg(0,vide,3,FALSE,FALSE);
endif;

{/*/*/*/ les images /*/*/*/}
image:=Concat(chemin,"IM");
Pause("Acqu,rir les images.");
PromptInt("Premier indice de fichier","Indice : ",premier);
WriteText("depart : ");WriteInt(premier);Newline;
for i:=premier to premier+1000 do
   user;
   VideoIn(0,42,0);
   auto;
   indice:=i;
   titre:=Concat(image,indice);
   titre:=Concat(titre,".BMP");
   SaveImg(0,titre,3,FALSE,FALSE);
   tempint:=i-premier+1;
   WriteInt(tempint);WriteText(" images");Newline;

   {/*/*/*/ controle /*/*/*/}
   if( ((i-premier+1)%3=0) and redebut ) then
      YesNo("Effectuer un contr^le.","Contr^le",yesno);
      if yesno then
         for j:=i-2 to i do
            indice:=j;
            titre:=Concat(image,indice);
            titre:=Concat(titre,".BMP");
            ClearAll; {efface toutes les images}
            OpenImg(0,vide,-1,0,0); {ouvre l'image de fond}
            Copy(0,1);
            OpenImg(0,titre,-1,0,0); {ouvre l'image a traiter}
            AbsDiff(1,0,2);
            ImgHisto(2,3); {histogramme de l'image}
            Fisher(seuil); {determination du seuil de binarisation}
            Thresh(2,3,seuil,1); {binarisation}
            NotImg(3,2);
            Label(2,3,8,0,0,1,50000.0,262144.0,tempint);
            Thresh(3,2,1,0); {seuil bas sur le fond : on recupere les cellules pleines}
            Clear(3);
            user;
            DispAll;
            auto;
         endfor;

         YesNo("Continuer les acquisitions","Suite",YesNo);
         if yesno=FALSE then
            YesNo("Acqu,rir un champ vide.","Champs Vide",yesno);
            if yesno then
               ressaisir:=1;
               redebut:=TRUE;
               WriteText("Mauvaise acquisition");Newline;
            else
               ressaisir:=0;
               redebut:=FALSE;
               WriteText("Revue de l'acquisition");Newline;
            endif;
            while ressaisir=1 do
               user;
               VideoIn(0,42,0);
```

```
            auto;
            YesNo("Champs vide correct ?","Champs vide",yesno);
            if yesno then
               ressaisir:=2;
            endif;
         endwhile;
         if ressaisir=2 then
            vide:=Concat(chemin,"IMVIDE.BMP");
            SaveImg(0,vide,3,FALSE,FALSE);
         endif;

         Pause("Acqu,rir les images.");
         if redebut then
            PromptInt("Premier indice de fichier","Indice : ",premier);
            WriteText("depart : ");WriteInt(premier);Newline;
            i:=premier-1;
         else
            redebut:=TRUE;
         endif;
      else
         redebut:=FALSE;
      endif;{continuer acquisition}
   else
      redebut:=FALSE;
   endif;{controle}
 endif; {i=premier+3}
 {/*/*/*/ fin controle /*/*/*/}

endfor;{1000}
```

5.3 - Programme d'analyse et classement

```
Int debut[100],fin[100],i,i1,j1,j2,nblame,contlame;
Int seuil,nbcel,vrainbcel,nbnoy[256],totalcellame;
Int indice,indicefic,ecrase;
Int gauchecel[256],hautcel[256],droitecel[256],bascel[256];
Int gauche,haut,droite,bas;
Int homogene[256],tempint,tableau[512],celluleOK[256];
Int scoregranu[256],scorepetit[256],appartenance[256],nbpoints[4],nbpara[4];
Real pourcent,pourcentpara;
Real airecel[256],integrcel[256],meancel[256],stdevcel[256],skewcel[256],kurtcel[256];
Real surfacecel[256],fond,tmpsurfacenoy,surfacenoy[256];
Real histocel[256],tmpreal,miniboni,maxiboni,max,boni[256];
Real rapportsurface;
Real Moylocale[256],Energie[256],Entropie[256],Inertie[256];
Real SRE[256],LRE[256],GLD[256],RLD[256],RLP[256];

Int retirercellule,tmpint,taillenoyaumin;

Text bmp,txt,temptext,saveparam,outputxt;
Text tempcheminimage,cheminimage[100],cheminsauve,vide,image,encourim,encourbi;
Condition yesno;

{/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/}
{/*/*/*/*/*/*/ INITIALISATIONS /*/*/*/*/*/*/*/*/}
{/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/}

tempcheminimage:="C:\ANGELI\im_26_0";
cheminsauve:="C:\ANGELI\SCRIPTS\";
saveparam:="SAV.TXT";
outputxt:="OUT.TXT";
encourim:="C:\ANGELI\ENCOURIM.BMP";
encourbi:="C:\ANGELI\ENCOURBI.BMP"


bmp:=".BMP";
txt:=".TXT";
taillenoyaumin:=500;    {***** A VOIR *****}
miniboni:=30;
maxiboni:=250;


PromptInt("Indice","nombre de lames ",nblame);

{**** LES LAMES ****}
for contlame:=1 to nblame do
   temptext:=contlame;
   cheminimage[contlame]:=concat(tempcheminimage,temptext);
   cheminimage[contlame]:=concat(cheminimage[contlame],"\");
   PromptText("Repertoire de travail","Chemin du repertoire de travail ",cheminimage[contlame]);
   PromptInt("debut de traitement","Premier indice d'image ? ",debut[contlame]);
   PromptInt("fin de traitement","Dernier indice d'image ? ",fin[contlame]);
endfor;

{**** fichiers de sauvegarde ****}
temptext:=Concat(cheminsauve,outputxt); {le fichier output existe-t-il ?}
ecrase:=-1;
while ecrase=-1 do
   PromptText("Fichier de sauvegarde OUTPUT","Nom du fichier",temptext);
   ecrase:=0;
   OpenPar(ecrase,temptext);
```

```
endwhile;
ClosePar(ecrase);
outputxt:=temptext;

saveparam:=Concat(cheminsauve,saveparam); {le fichier resultat existe-t-il ?}
ecrase:=-1;
while ecrase=-1 do
   PromptText("Fichier de sauvegarde RESULTATS","Nom du fichier",saveparam);
   ecrase:=0;
   OpenPar(ecrase,saveparam);
endwhile;


{/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/}
{/*/*/*/*/*/*/ DEBUT DU PROGRAMME */*/*/*/*/*/*/}
{/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/}


tmpreal:=nblame;
WritePar(0,tmpreal);

for contlame:=1 to nblame do

  totalcellame:=0;

  Newline;
  tmpreal:=fin[contlame]-debut[contlame]+1;
  WritePar(0,tmpreal);

  cheminimage[contlame]:=Concat(cheminimage[contlame],"IM");
  vide:=Concat(cheminimage[contlame],"VIDE.BMP"); {fichier contenant l'image de fond}

  for indicefic:=debut[contlame] to fin[contlame] do {pour chaque fichier}

     temptext:=indicefic;
     image:=Concat(cheminimage[contlame],temptext);
     image:=Concat(image,bmp); {fichier contenant l'image des cellules}

     ClearAll; {efface toutes les images}
     OpenImg(0,vide,-1,0,0); {ouvre l'image de fond}
     Copy(0,1);
     OpenImg(0,image,-1,0,0); {ouvre l'image a traiter}
     AbsDiff(1,0,2);
     SaveImg(2,encourim,3,FALSE,FALSE); {sauvegarde de l'image pre-traitee}

{***** MESURES MORPHOMETRIQUES SUR LES CELLULES *****}

    ImgHisto(2,3); {histogramme de l'image}
    Fisher(seuil); {determination du seuil de binarisation}
    Thresh(2,3,seuil,1); {binarisation}

    NotImg(3,0);
    Label(0,1,8,0,0,1,50000.0,262144.0,nbcel);
    Thresh(1,0,1,0); {seuil bas sur le fond : on recupere les cellules pleines}

    SaveImg(0,encourbi,1,TRUE,TRUE); {sauvegarde de l'image binaire}

    Label(0,1,8,1,1,1,1700.0,50000.0,nbcel); {etiquetage des cellules}
    vrainbcel:=nbcel;

    for i:=1 to nbcel do {pour chaque cellule}

      ClearAll(); {efface tout}
      OpenImg(0,encourbi,-1,0,0); {ouverture de l'image binaire}

      Label(0,1,8,1,1,1,1700.0,50000.0,nbcel); {etiquetage}
      Extract(1,0,i,gauchecel[i],hautcel[i],droitecel[i],bascel[i],surfacecel[i]);
                                {extraction de la cellule}
      SetWindow(gauchecel[i],hautcel[i],droitecel[i],bascel[i]);

        OpenImg(1,encourim,-1,0,0); {ouverture de l'image des cellules}

        AndImg(0,1,2); {cellule seule}

        GetRegHisto(2,0,histocel); {histogramme sur la cellule seule}
        Fisher(seuil);
        seuil:=seuil+20; {ajustement du seuil de binarisation}

        Thresh(2,3,seuil,1); {binarisation pour localiser le noyau}

        densito(airecel[i],integrcel[i],meancel[i],stdevcel[i],skewcel[i],kurtcel[i]);

          {parametres de texture}

        TextureCooc(2,0,1,16,Moylocale[i],Energie[i],Entropie[i],Inertie[i]);

        TextureRLS(2,0,16,SRE[i],LRE[i],GLD[i],RLD[i],RLP[i]);

        Label(3,2,8,1,1,1,miniboni,2000.0,nbnoy[i]);

          {** caracterisation des noyaux **}
        surfacenoy[i]:=taillenoyaumin;
        max:=0;
        boni[i]:=0;

        for i1:=1 to nbnoy[i] do {pour chaque noyau etiquete}
```

```
        Extract(0,1,i1,gauche,haut,droite,bas,tmpsurfacenoy);

        if (tmpsurfacenoy>miniboni)&&(tmpsurfacenoy<maxiboni) then
          boni[i]:=boni[i]+1;
        else
          if (tmpsurfacenoy>surfacenoy[i]) then
            max:=surfacenoy[i];
            surfacenoy[i]:=tmpsurfacenoy;
          else
            if (tmpsurfacenoy>max) then
              max:=tmpsurfacenoy;
            endif;
          endif;
        endif;

      endfor; {pour chaque noyau}


      if ((max>taillenoyaumin)||(surfacenoy[i]<=taillenoyaumin)) then
                celluleOK[i]:=0;
                vrainbcel:=vrainbcel-1;
      else
                celluleOK[i]:=1;
      endif;

    RestWindow;

endfor; {fin pour chaque cellule i}


tmpreal:=vrainbcel;
WritePar(0,tmpreal);
retirercellule:=0;
for i:=1 to nbcel do {pour chaque cellule i}
  if (celluleOK[i]=0) then
    retirercellule:=retirercellule+1;
  else

    scoregranu[i]:=0;
    scorepetit[i]:=0;

    rapportsurface:=surfacenoy[i]/surfacecel[i];

    tmpreal:=i-retirercellule;
    WritePar(0,tmpreal);

    WritePar(0,surfacecel[i]);
      if surfacecel[i]>4500 then
        scoregranu[i]:=scoregranu[i]+1;
      endif;
      if surfacecel[i]<3500 then
        scorepetit[i]:=scorepetit[i]+1;
      endif;

    WritePar(0,surfacenoy[i]);
      if surfacenoy[i]>700 then
        scoregranu[i]:=scoregranu[i]+1;
      endif;

    WritePar(0,rapportsurface);
      if rapportsurface<0.26 then
        scoregranu[i]:=scoregranu[i]+1;
      endif;
      if rapportsurface>0.3 then
        scorepetit[i]:=scorepetit[i]+1;
      endif;

    WritePar(0,meancel[i]);
      if meancel[i]>80 then
        scoregranu[i]:=scoregranu[i]+1;
      endif;
      if meancel[i]>100 then
        scorepetit[i]:=scorepetit[i]+1;
      endif;

    WritePar(0,skewcel[i]);

    WritePar(0,kurtcel[i]);
      if kurtcel[i]<0 then
        scorepetit[i]:=scorepetit[i]+1;
      endif;

    WritePar(0,boni[i]);

    WritePar(0,Moylocale[i]);
      if Moylocale[i]>0.35 then
        scoregranu[i]:=scoregranu[i]+1;
      endif;
      if Moylocale[i]>0.4 then
        scorepetit[i]:=scorepetit[i]+1;
      endif;

    WritePar(0,Energie[i]);
      if Energie[i]<0.1 then
        scoregranu[i]:=scoregranu[i]+1;
```

```
        endif;
        if Energie[i]>0.05 then
          scorepetit[i]:=scorepetit[i]+1;
        endif;

      WritePar(0,Entropie[i]);
        if Entropie[i]>2.8 then
          scoregranu[i]:=scoregranu[i]+1;
        endif;

      WritePar(0,Inertie[i]);

      WritePar(0,SRE[i]);
        if SRE[i]>0.45 then
          scoregranu[i]:=scoregranu[i]+1;
        endif;
        if SRE[i]>0.5 then
          scorepetit[i]:=scorepetit[i]+1;
        endif;

      WritePar(0,LRE[i]);
        if LRE[i]<20 then
          scoregranu[i]:=scoregranu[i]+1;
        endif;
        if LRE[i]>10 then
          scorepetit[i]:=scorepetit[i]+1;
        endif;

      WritePar(0,GLD[i]);

      WritePar(0,RLD[i]);
        if RLD[i]>0.25 then
          scoregranu[i]:=scoregranu[i]+1;
        endif;

      WritePar(0,RLP[i]);
        if RLP[i]>70 then
          scoregranu[i]:=scoregranu[i]+1;
        endif;

  { les scores }

      if scoregranu[i]>9 then
        appartenance[i]:=2; { granulocyte }
        if boni[i]<5 then
          boni[i]:=0;
        endif;
      else
        if scorepetit[i]>6 then
          appartenance[i]:=4; { petit hyalinocyte }
        else
          if (scoregranu[i]>6)&&(scorepetit[i]>4) then
            appartenance[i]:=1; { non class, }
          else
            appartenance[i]:=3; { grand hyalinocyte }
          endif;
        endif;
      endif;

      nbpoints[appartenance[i]]:=nbpoints[appartenance[i]]+1;
      if boni[i]>1 then
        nbpara[appartenance[i]]:=nbpara[appartenance[i]]+1;
      endif;


    endif; {if celluleOK}

  endfor; {for nbcel}
totalcellame:=totalcellame+vrainbcel;
endfor; {for fic}

WriteInt(totalcellame);
WriteText(" total");Newline;
      if totalcellame>0 then
        Newline;WriteText("Lame :");WriteInt(contlame);Newline;Newline;
        WriteInt(totalcellame);WriteText(" cellules");Newline;

        WriteText("Granuleuses : ");WriteInt(nbpoints[2]);WriteText(" dont ");
        WriteInt(nbpara[2]);WriteText(" parasit,es");Newline;

        WriteText("Agranuleuses : ");WriteInt(nbpoints[3]);WriteText(" dont ");
        WriteInt(nbpara[3]);WriteText(" parasit,es");Newline;

        WriteText("Petits : ");WriteInt(nbpoints[4]);WriteText(" dont ");
        WriteInt(nbpara[4]);WriteText(" parasit,es");Newline;

        WriteText("non class,es : ");WriteInt(nbpoints[1]);WriteText(" dont ");
        WriteInt(nbpara[1]);WriteText(" parasit,es");Newline;

        Newline;

  { en pourcentage }

        if nbpoints[2]>0 then
          pourcent:=100*nbpoints[2]/(totalcellame-nbpoints[1]);
          pourcentpara:=100*nbpara[2]/nbpoints[2];
```

```
            WriteReal(pourcent);WriteText(" % de granulocytes dont ");
            WriteReal(pourcentpara);WriteText(" % de parasit,es");Newline;
          else
            WriteText("0 granulocytes");Newline;
          endif;

          if nbpoints[3]>0 then
            pourcent:=100*nbpoints[3]/(totalcellame-nbpoints[1]);
            pourcentpara:=100*nbpara[3]/nbpoints[3];
            WriteReal(pourcent);WriteText(" % d'agranuleuses dont ");
            WriteReal(pourcentpara);WriteText(" % de parasit,es");Newline;
          else
            WriteText("0 agranuleuses");Newline;
          endif;

          if nbpoints[4]>0 then
            pourcent:=100*nbpoints[4]/(totalcellame-nbpoints[1]);
            pourcentpara:=100*nbpara[4]/nbpoints[4];
            WriteReal(pourcent);WriteText(" % de petits dont ");
            WriteReal(pourcentpara);WriteText(" % de parasit,es");Newline;
          else
            WriteText("0 petits");Newline;
          endif;
          Newline;

          if nbpoints[1]>0 then
            pourcent:=100*nbpoints[1]/totalcellame;
            pourcentpara:=100*nbpara[1]/nbpoints[1];
            WriteReal(pourcent);WriteText(" % de non classes dont ");
            WriteReal(pourcentpara);WriteText(" % de parasitees");Newline;
          endif;
          Newline;

          pourcent:=nbpara[1]+nbpara[2]+nbpara[3]+nbpara[4];
          pourcent:=pourcent*100/totalcellame;
          WriteReal(pourcent);WriteText(" % de cellules parasit,es au total");
          Newline;

        endif; { nbvraicel>0 }

for i:=1 to 4 do
  nbpoints[i]:=0;
  nbpara[i]:=0;
endfor;

endfor; {for contlame}

ClosePar(0);

SaveOutput(outputxt);
Delete(encourim);
Delete(encourbi);


SaveOutput(outputxt);

Pause("Imprimer les r,sultats ?");
PrintOutput;

Pause("Effacer la fenetre Output ?");
RestOutput;
ClearAll;
```