Descripteurs de formes

Pour les mots

Introduction

- Objectif du cours
  - Présenter quelques techniques d'extraction de caractéristiques/primitives spécifiques aux mots manuscrits/imprimés

Extraction de primitives

- Principe
  - L'extraction de primitives consiste à transformer une image (caractère, graphème, mot, ...) en un vecteur de primitives de taille fixe
  - Cette transformation revient à changer l'espace de représentation des données, du plan de l'image vers un espace à N dimensions ($\Re^N$)
  - Le choix des primitives est critique et influence nettement le résultat
  - Ces primitives doivent avoir deux propriétés :
    - être discriminantes : permettre une bonne différentiation entre les classes de symboles à reconnaître
    - maintenir un nombre de dimensions limité, afin d'éviter le phénomène de malédiction de la dimensionnalité (curse of dimensionality)

Extraction de primitives

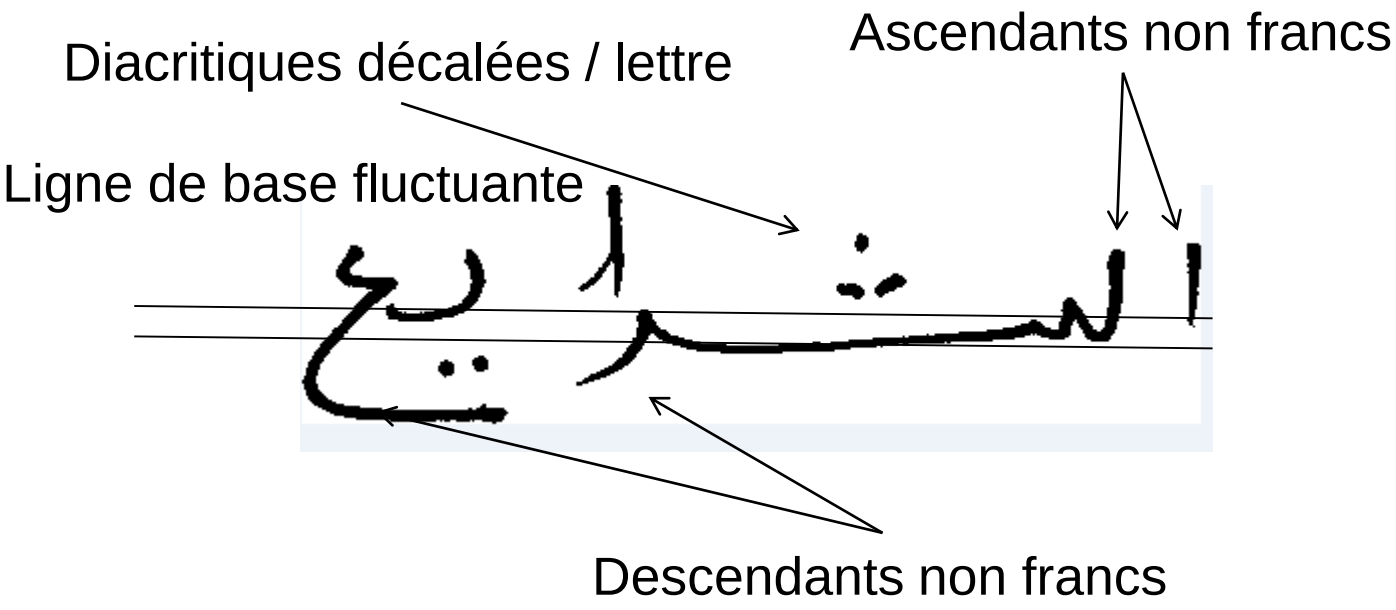
- Plan
  - Topologie du mot
    - Extraction des lignes d'écriture
  - Familles de primitives
    - Les primitives extraites à partir des profils et des contours
    - Les moments invariants
    - Les descripteurs de Fourier

Topologie du mot

- C’est une entité fondamentale du texte
- A une topologie : composition

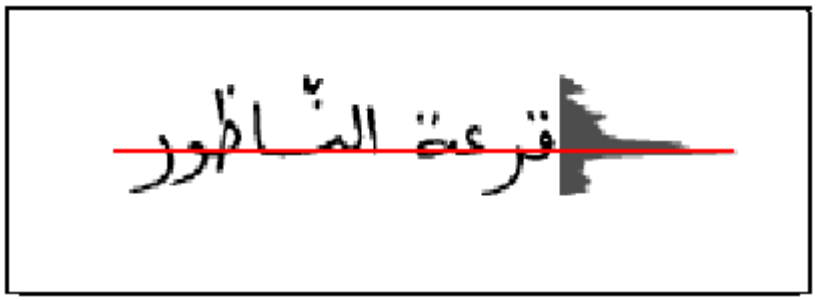
Topologie du mot manuscrit

- Variations courantes



Recherche de la ligne de base

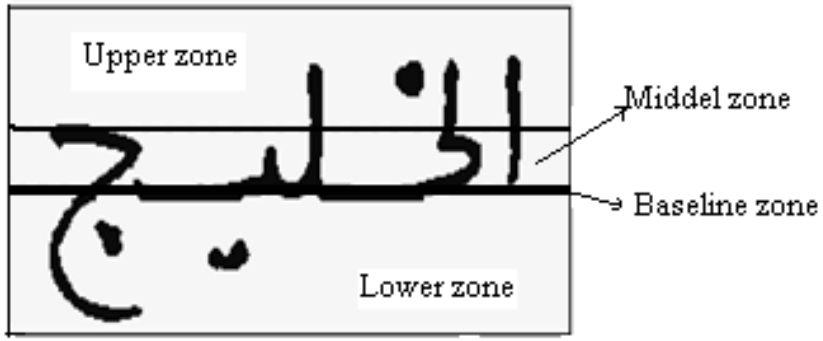
- Primordiale en reconnaissance de l'Arabe car il est cursif
- État de l'art assez fourni
  - A. AL-Shatnawi and K. Omar, Methods of Arabic Language Baseline Detection – The State of Art, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.10, October 2008
- Méthode par projection horizontale
  - Pechwitz et al. : recherche de pic de l'histogramme de projection



$$h(i) = \sum_{j} \mathrm{Im}age(i, j)$$

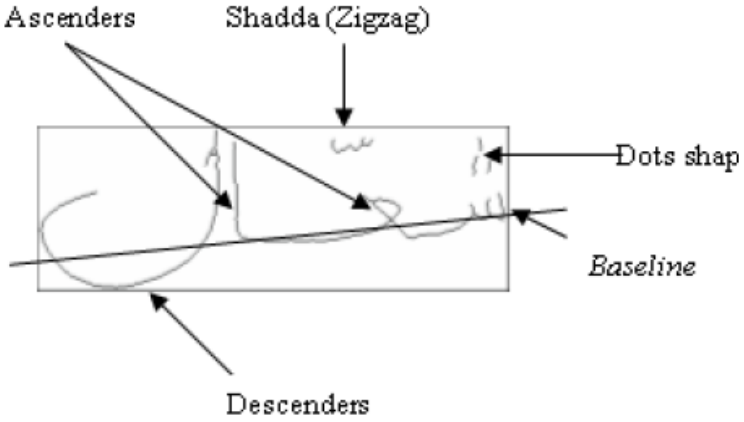
Recherche de la ligne de base

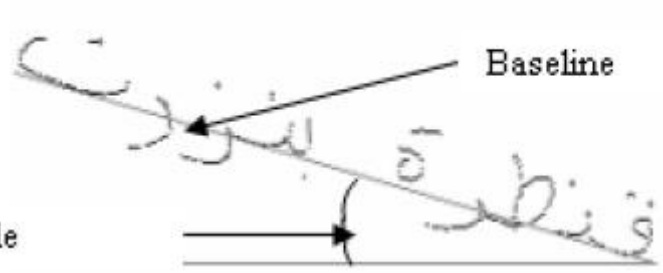
- Nawaz et al, Sarfraz et al utilisent la PH pour diviser le mot en quatre zones
  - baseline, middle, upper and lower zone
    - ➢ Ligne la plus dense
  - Middle : zone de forte concentration de pixels
    - ➢ Ligne juste au-dessus de la ligne de base et limite sup du pic de projection
  - I uts
  - I



Recherche de la ligne de base

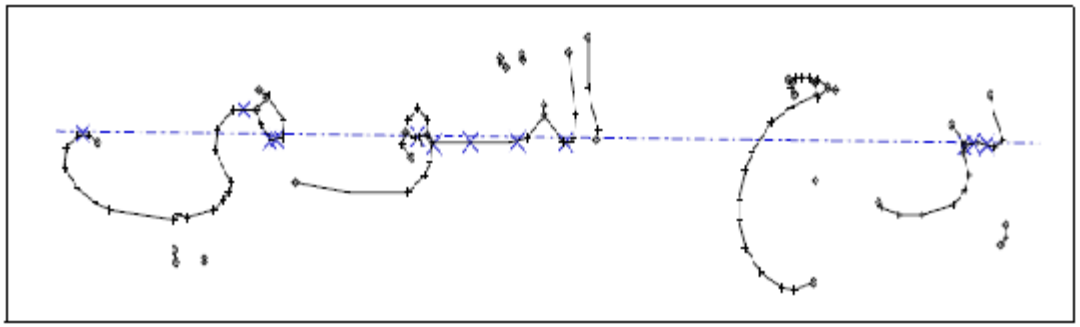
- H. Al Rashaideh : Preprocessing phase for Arabic word handwritten recognition. Russian Academy of Sciences, 6(1): 11-19
  - Fait l'hypothèse que la ligne de base est une vraie ligne de pixels
  - Applique à l'image deux rotations à +seuilangle° et -seuilangle°
  - Calcule l'histogramme de projection et regarde de quel côté pourrait être l'inclinaison (en comparant les deux pics)
  - si le deuxième pic est plus important, réapplique à nouveau la deuxième étape (avec un autre seuil d'angle), sinon, arrêt et détermination de la ligne de base





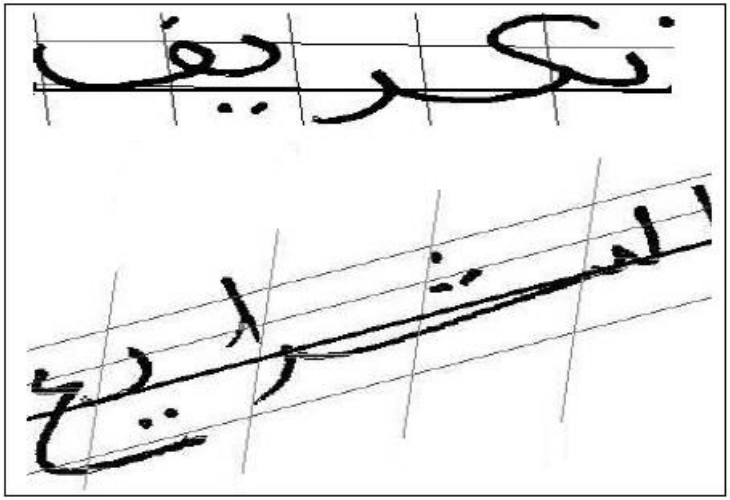
Recherche de la ligne de base

- Méthode basée sur le squelette
  - Pechwitz et Margner, ICFHR 2002
    - Utilisent la régression linéaire : droite des moindres carrés
    - La régression linéaire réagit bien aux endroits où l'accumulation linéaire des pixels existe



Recherche de la ligne de base

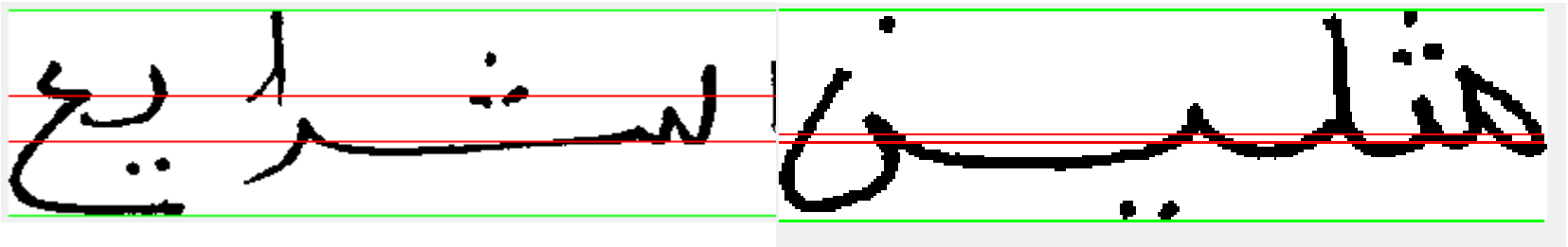
- Méthode basée sur le contour
  - F. Farooq, V. Govindaraju & M. Perrone, ICDAR 2005
    - Détecte les minima locaux des contours du mot
    - Les minima dépendent de l'orientation du contour : localisés là où le contour change de sens
    - Applique ensuite la régression linéaire pour détecter une ligne de base approximative



    - Recherche de nouveaux minimas locaux proches des anciens
    - Refait une régression linéaire pour trouver la vraie ligne de base

Recherche de la ligne de base

- Méthode basée sur le RunLength
  - Les run-length représentent des longueurs de segments noirs suivant une droite
  - L'histogramme des run-legth (par ajout ou multiplication des longueurs) peut renseigner sur les lignes
  - Voir Interface Matlab :
    - LignesDeBaseCompareesAKram
    - LigneBasePrimitives_Akram Khemiri

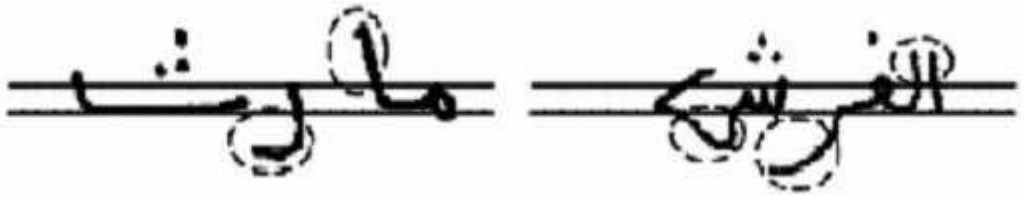


Extraction de caractéristiques

Les caractéristiques locales

1) Les diacritiques

2) Les ascendants et les descendants

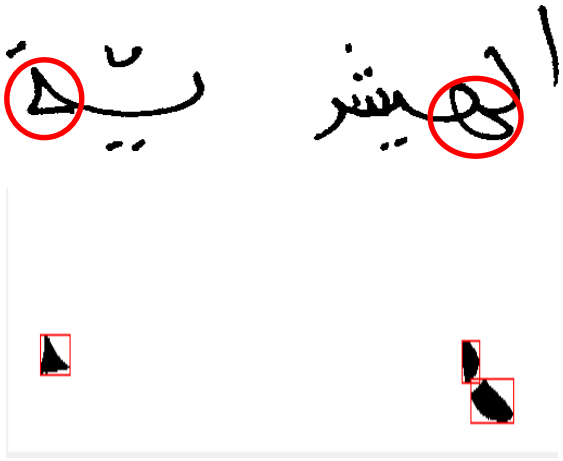


3) Les boucles

Extraction de caractéristiques

Les caractéristiques locales

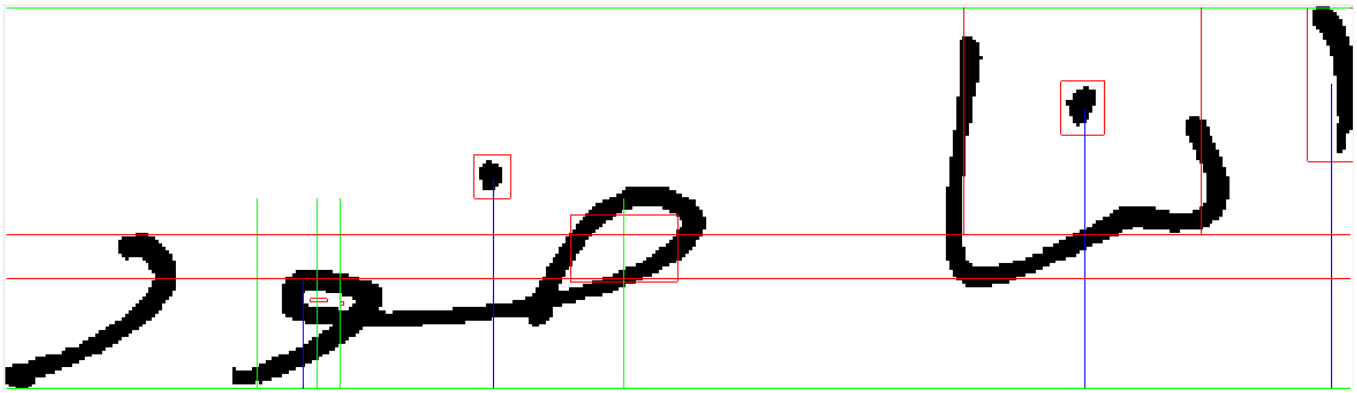
- Exemple : InterfaceMatlab4/Trous



Extraction de caractéristiques

Les caractéristiques locales

- Exemple : InterfaceMatLab4/PrimitivesStructurelles
  - Reprendre ce programme et faire mieux ressortir les hampes et les jambages
  - Donner une description linéaire des primitives de la droite vers la gauche



Extraction de caractéristiques

Les caractéristiques globales

- Différents types
  - Caractéristiques structurelles
  - Caractéristiques des composantes connexes
  - Répartition des pixels
  - Bottom lines
  - Vertical edge
  - Propriétés locales

Extraction de caractéristiques

Caractéristiques structurelles

- Rôle
  - Refléter les différences d'apparence visuelle et de structure
    - Largeur
      - Largeur (*w*) du bloc en pixels
    - Hauteur
      - Hauteur (*h*) du bloc en pixels
    - Ratio Hauteur/Largeur
      - Rapport entre la hauteur et la largeur
    - Aire
      - Surface du bloc

- Caractéristiques structurelles (suite 1)
  - Densité
    - Rapport entre le nombre de pixels noirs et le nombre total de pixels :

$$\text{densité} = \frac{\text{nb pixels noirs}}{\text{hauteur} * \text{largeur}}$$

    - Si I(x,y) est un pixel de l'image valant 0 (fond) ou 1 (pixel noir), la densité d s'écrit :

$$d = \frac{\sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y)}{w * h}$$

    - Étant donné que la hauteur, la largeur, le ratio et l'aire peuvent varier selon la police d'écriture, il est possible de normaliser par la police dominante en utilisant une méthode à histogramme

- Caractéristiques structurelles
  - Exemples

width = 449
height = 119
area = 53431
density = 0.08
ratio = 3.7

width = 213
height = 78
area = 16614
density = 0.20
ratio = 2.7

Caractéristiques globales

Descripteurs issus des composantes connexes

- Recouvrement (overlapping areas)
  - Yefeng Zheng, 2004
  - Pour l'écriture manuscrite, les blocs encadrant les composantes connexes ont tendance à se chevaucher
  - Cela ne se produit pas pour la plupart des polices imprimées
  - On calcule alors le recouvrement comme la somme des aires de recouvrement sur l'aire totale du bloc

– La formule de la caractéristique est donc :

$$recouvrement = \frac{\sum_{i \neq j} \text{Aire}\left(BB(i) \cap BB(j)\right)}{a}$$

– où BB(i) est le rectangle encadrant la composante connexe i, et *a* l'aire totale du bloc

Caractéristiques globales

Répartition des pixels

- Les moments (R. Kandan, 2007)
  - Les moments traduisent la distribution des pixels dans le plan
  - Connaissant l'objet par sa fonction caractéristique f(x,y), une représentation classique de sa forme consiste à en mesurer les divers moments :

$$M_{mn} = \iint x^m y^n f(x,y) dx dy$$

  - En particulier, les moments centrés traduisent l'influence qu'ils exercent en fonction de leur éloignement / au centre :

$$\bar{M}_{mn} = \iint (x - X_g)^m (y - Y_g)^n f(x,y) dx dy$$

– Sur une image discrète, ces moments s'écrivent :

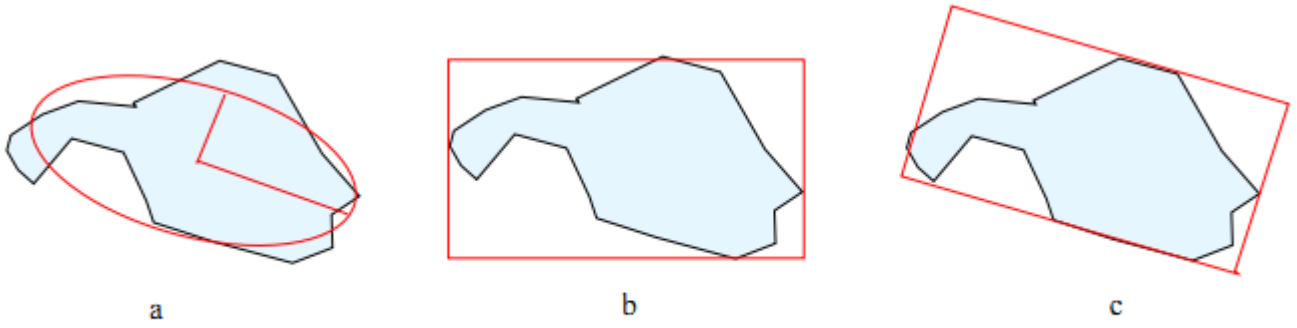
$$\bar{M}_{mn} = \frac{1}{J^m K^n} \sum_{j=1}^{J} \sum_{k=1}^{K} (j - X_g)^m (k - Y_g)^n f(j,k)$$

– avec le centre de masse :

$$X_g = \frac{M_{10}}{M_{00}} \text{ et } Y_g = \frac{M_{01}}{M_{00}}$$

- InterfaceMatlab4/Moments centraux

- Les moments d'inertie
  - Traduisent les concentrations de points le long des axes importants de la forme, et donc les allongements, l'épaisseur, etc.
  - Correspondent aux valeurs propres de la matrice d'inertie
  - Sont invariants par rotation, échelle…



Description de formes par : a - ellipse et axes d'inertie, b - boîte englobante, c - boîte minimale.

- Les moments invariants (de Hu)
  - Ce sont des moments issus des moments d'inertie (centraux)
    - nommés également moments de Hilbert
  - Ils sont invariants par translation, rotation et changement d'échelle [Hu, 1962]
  - Ils se construisent à partir du moment normé :

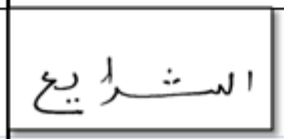
$$N_{nm} = \frac{J^m K^n}{\bar{M}_{00}^{\alpha}} \bar{M}_{mn}$$

où $\alpha = \frac{1}{2}(m+n)+1$, par les formules :

$$
\begin{aligned}
h_1 &= N_{20} + N_{02} \\
h_2 &= (N_{20} - N_{02})^2 + 4N_{11}^2 \\
h_3 &= (N_{30} - 3N_{12})^2 + (N_{03} - 3N_{21})^2 \\
h_4 &= (N_{30} + N_{12})^2 + (N_{03} - N_{21})^2 \\
h_5 &= (N_{30} - 3N_{12})(N_{30} + N_{12})[(N_{30} + N_{12})^2 - 3(N_{03} + N_{21})^2] \\
&\quad +(3N_{21} - N_{03})(N_{03} + N_{21})[3(N_{30} + N_{12})^2 - (N_{03} + N_{21})^2] \\
h_6 &= (N_{20} - N_{02})[(N_{30} + N_{12})^2 - (N_{03} + N_{21})^2] \\
&\quad +4N_{11}(N_{30} + N_{12})(N_{03} + N_{21}) \\
h_7 &= (3N_{12} - N_{30})(N_{03} + N_{21})[3(N_{30} + N_{12})^2 - (N_{03} + N_{21})^2] \\
&\quad +(3N_{21} - N_{03})(N_{30} + N_{12})[(N_{30} + N_{12})^2 - 3(N_{03} + N_{21})^2]
\end{aligned}
$$

- Interprétation
  - h_1 : analogue au moment d'inertie et fait intervenir des moments d'ordre 2
  - h_2 : fait intervenir des moments d'ordre 2
    - Certains l'interprètent comme : l'étroitesse
  - h_3 : n'ont pas d'interprétation physique et sont des moments d'ordre 3
  - h_7 :
    - est invariant par rotation
    - permet de détecter les images miroir :
      - si on applique une symétrie sur le motif, le signe de Φ_7 est inchangé

- Test des moments de Hu : InterfaceMatlab4/MomentsHu

		h1	h2	h3	h4	h5	h6	h7
Transformation	Image	Φ1	Φ2	Φ3	Φ4	Φ5	Φ6	Φ7
image de base		4,84	20,77	15,54	11,40	151,75	53,25	2,06
miroir horizontal		4,84	20,77	15,54	11,40	151,75	53,25	-2,06
		0%	0%	0%	0%	0%	0%	0%
miroir vertical		4,84	20,77	15,54	11,40	151,75	53,25	-2,06
		0%	0%	0%	0%	0%	0%	0%
rotation 180°		4,84	20,77	15,54	11,40	151,75	53,25	2,06
		0%	0%	0%	0%	0%	0%	0%
translation		4,84	20,77	15,54	11,40	151,75	53,25	2,06
		0%	0%	0%	0%	0%	0%	0%

rotation 90° droite		4,84	20,77	15,54	11,40	151,75	49,87	2,06
		0%	0%	0%	0%	0%	6%	0%
rotation 90° gauche		4,84	20,77	15,54	11,40	151,75	49,87	2,06
		0%	0%	0%	0%	0%	6%	0%
rotation 36° droite		4,87	21,06	15,58	11,42	152,24	49,64	2,05
		1%	1%	0%	0%	0%	7%	0%
grandissement échelle	140% de la taille d'origine	4,46	17,62	12,48	9,18	98,29	39,38	1,29
		8%	15%	20%	19%	35%	26%	37%
diminution échelle	80% taille d'origine	4,34	16,74	11,66	8,64	86,61	36,12	1,17
		10%	19%	25%	24%	43%	32%	43%

Les primitives extraites à partir des profils et des contours

Extraction de primitives

Moments invariants

- Moments de Zernike : InterfaceMatlab4/Zernike_code
  - Les polynômes de Zernike $V_{mn}(x,y)$ sont exprimés en coordonnées polaires :

$$V_{mn}(r,\theta) = R_{mn}(r)e^{-jn\theta}$$

  - où $R_{mn}(r)$ est le polynôme radial orthogonal :

$$R_{mn}(r) = \sum_{s=0}^{\frac{m-|n|}{2}} (-1)^s \frac{(m-s)!}{s!(\frac{m+|n|}{2}-s)!\ (\frac{m-|n|}{2}-s)!} r^{m-2s}$$

  - Les moments A_{mn} sont invariants par rotation et changements d'échelles

- Variance du profil de projection verticale

  (Lincoln Faria da Silva, 2009)

  – Cette caractéristique permet d'en savoir plus sur l'étalement horizontal du mot

  – Pour la calculer, on projette verticalement un bloc afin d'obtenir un histogramme

    • L'histogramme peut s'obtenir en calculant les couples :

$$(i, n_i) = \left(i \, , \sum_y I(i, y) \right), i \in [\![1, w]\!]$$

    • n_i est le nombre de pixels noirs présents sur la colonne i

    • On peut alors calculer la variance en remarquant que n_i représente les effectifs de la grandeur i.

$$Var_{Profil\ Vertical} = \frac{1}{n} * \sum_i (i - \bar{\imath})^2 * n_i$$

Où $\bar{\imath} = E[i] = \frac{1}{n} * \sum_i i * n_i$ est la moyenne des i.

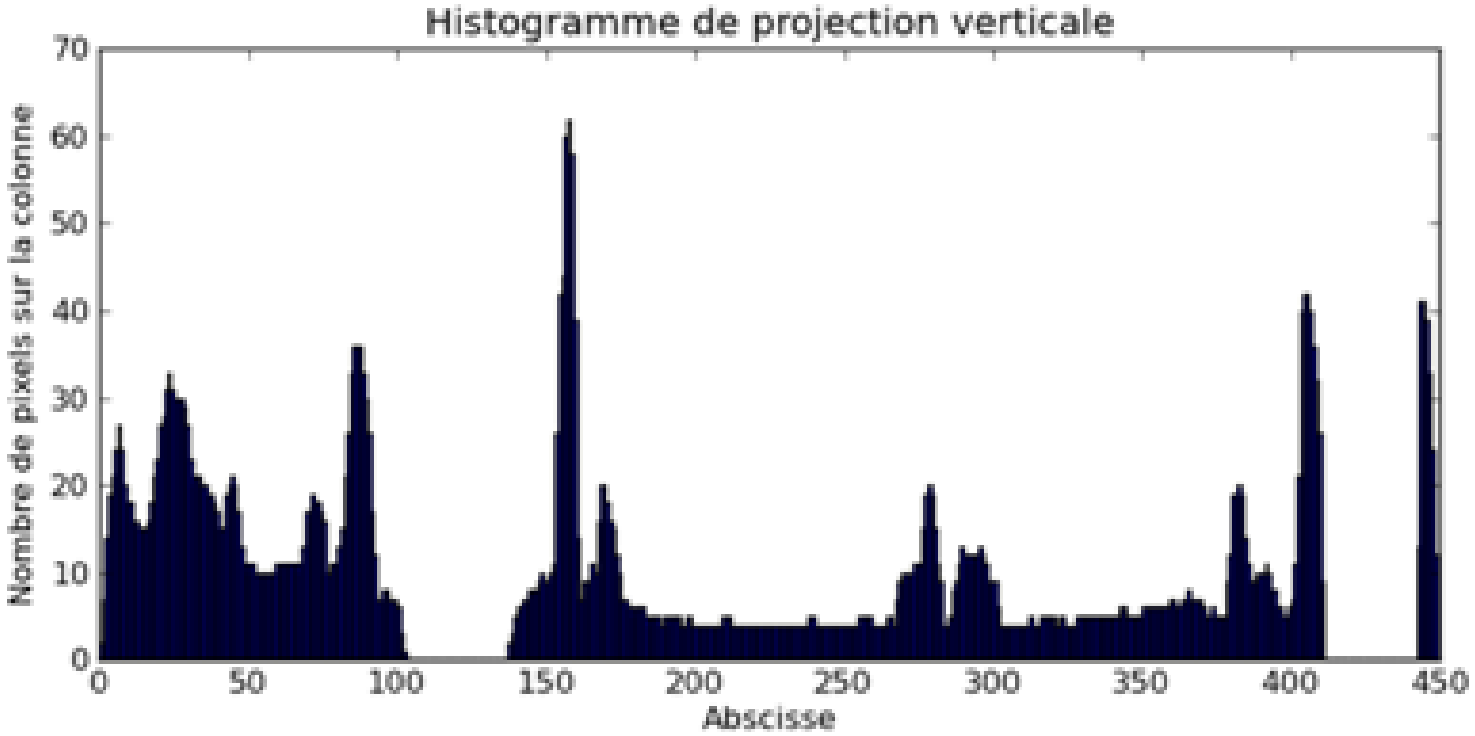
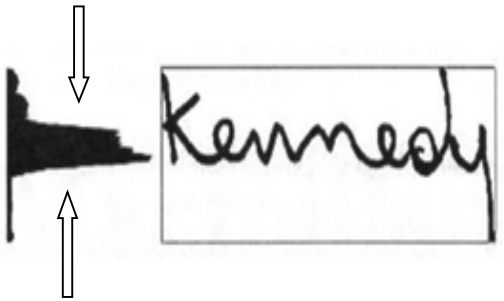


Figure 2 : Histogramme de projection verticale

- Pic de projection horizontale

  Lincoln Faria da Silva, 2009

  – On projette un bloc horizontalement afin d'obtenir un histogramme
  – Ensuite, on cherche à calculer la largeur du pic, sa hauteur, etc.



$$Profil_{horizontal}(y) = \sum_{x} I(x,y)$$

- Distribution des pixels (Lincoln Faria da Silva, 2009)
  - Cette caractéristique indique s'il y a une grande différence de répartition des pixels entre la moitié supérieure et la moitié inférieure du bloc
  - Le bloc est divisé en 2 par une ligne horizontale au milieu du bloc. Ensuite, on réduit la taille du bloc de 10 pixels en hauteur. On calcule ensuite la densité dans la moitié supérieure du bloc (UD), puis dans la moitié inférieure (LD). On garde alors le module des différences :

$$Pixels\ Distribution = |UD - LD|$$

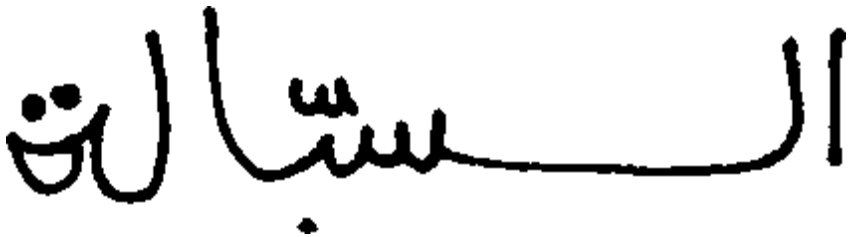
Caractéristiques globales

Bottom lines

- Principe
  - On compte le nombre de pixels noirs présents sur la dernière ligne du bloc
  - Le rapport entre ce nombre et la largeur du bloc est stockée comme caractéristique
    - Un mot imprimé aura plutôt tendance à avoir une grande valeur de bottom line
  - On note que cette caractéristique est sensible à l'angle d'inclinaison du document

$$\text{Bottom Line} = \frac{\sum_x I(x, y_{last})}{w}$$

Variance Profil = 20314.81
Bottom line = 0.00445
Pixel Distribution = 0.036



Variance Profil = 17226.22
Bottom line = 0.0071
Pixel Distribution = 0.079

Variance Profil = 11426.17
Bottom line = 0.0112
Pixel Distribution = 0.0656

Caractéristiques globales

Propriétés locales

- Objectif
  - Décrire globalement la forme en s'intéressant à quelques détails locaux
  - Plusieurs types
    - Run-length
    - Crossing Count Histogram
    - Bilevel Co-Occurrence
    - NxM Gram

Caractéristiques globales

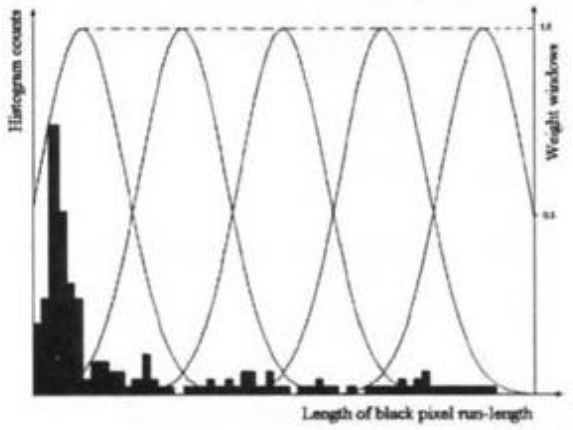
Propriétés locales

- Run-length Histogram (Yefeng Zheng, 2004)
  - La caractéristique Run-length tient compte du nombre de pixels noirs successifs qu'on peut trouver dans plusieurs directions
  - Il est possible de prendre par exemple l'axe horizontal, l'axe vertical et les deux diagonales
  - Pour une direction donnée (par exemple l'axe horizontal), on compte le nombre de pixels noirs successifs suivant l'axe (ici, pour chaque ligne)

– Afin d'obtenir une invariance d'échelle, on normalise les valeurs obtenues :
  - Soit C_k le nombre de segments de longueur k, la valeur normalisée vaut

$$C'_k = \frac{C_k}{\sum_{i=1}^{N} C_i}.$$

  - On construit ensuite l'histogramme des valeurs normalisées
  - On divise l'histogramme en 5 fenêtres de largeur égales
  - Le type de fenêtre (rectangulaire, gaussienne) servira à sommer les valeurs de l'histogramme tout en pondérant les valeurs par rapport à la position dans la fenêtre



– Zheng conclut que la fenêtre gaussienne donne de meilleurs résultats que la fenêtre rectangulaire. Pour des valeurs de i de 1 à 5, on calcule :

$$Rh_i = \sum_{k=k_{min}}^{k_{max}} G(k; u_i, \sigma).C'_k$$

avec $G(k;u_i;\sigma)$ la gaussienne numéro k, centrée en u_i et de variance σ

$$G(k; u_i, \sigma) = \exp\left(-\frac{(k-u_i)^2}{2\sigma^2}\right)$$

- La variance est choisie pour que les gaussiennes valent ½ aux extrémités de leur zone
- On calcule donc la largeur d'une gaussienne :

$$largeur = \frac{k_{max} - k_{min}}{n}$$

- Les centres des gaussiennes et la variance valent :

$$u_i = k_{min} + largeur * \left(\frac{1}{2} + i\right), i \in [\![0, n-1]\!]$$

$$\sigma = \frac{largeur}{2\sqrt{2\ln(2)}}$$

InterfaceMatlab4: RunLengthNBHVD

Caractéristiques globales

Propriétés locales

- Crossing Count Histogram : à faire en Matlab
  - Le principe du crossing count est de compter le nombre de fois que la valeur du pixel passe de 0 à 1 en suivant un axe donné
  - Les axes pris sont l'axe horizontal et l'axe vertical
  - La même technique est employée que pour le Run-Length Histogram pour en extraire des caractéristiques

$$Profil_{CCHorizontal}(y) = \sum_{x} \big(1 - I(x,y)\big) * I(x+1,y)$$

$$Profil_{CCVertical}(x) = \sum_{y} \big(1 - I(x,y)\big) * I(x,y+1)$$

Caractéristiques globales

Propriétés locales

- Bilevel Co-Occurrence : InterfaceMatlab4/ matCoccur
  - On compte le nombre de fois qu'un motif apparait
    - Un motif est une paire de pixels blanc-blanc, blanc-noir, noir-blanc, noir-noir distance de pixels, suivant un axe donné
  - Ici, on considère que ce sont les pixels noirs qui contiennent de l'information :

    ➔ On garde donc les couples noir-noir

Caractéristiques globales

Propriétés locales

- Bilevel Co-Occurrence (suite)
  - Les directions choisies sont l'axe horizontal, l'axe vertical et les deux diagonales
    - Les distances retenues par Zheng sont 1, 2, 4 et 8 pixels
    - Les caractéristiques se mettent donc sous la forme :

$$C_h(d) = \sum_x \sum_y I(x,y)I(x+d,y) \quad \text{avec} \quad d \in \{1,2,4,8\}$$

$$C_v(d) = \sum_x \sum_y I(x,y)I(x,y+d) \quad \text{avec} \quad d \in \{1,2,4,8\}$$

$$C_{d1}(d) = \sum_x \sum_y I(x,y)I(x+d,y-d) \quad \text{avec} \quad d \in \{1,2,4,8\}$$

$$C_{d2}(d) = \sum_x \sum_y I(x,y)I(x+d,y+d) \quad \text{avec} \quad d \in \{1,2,4,8\}$$

Caractéristiques globales

Propriétés locales

- Bilevel Co-Occurrence (suite 2)
  - Pour chaque distance, on normalise la valeur obtenue par la somme des occurrences dans toutes les directions
    - Ce qui nous intéresse sont en fait les directions privilégiées pour une distance donnée
  - Pour l'axe horizontal, la formule devient :

$$C'_h(d) = \frac{C_h(d)}{C_h(d) + C_v(d) + C_{d1}(d) + C_{d2}(d)}$$

    - où d1 et d2 sont les diagonales

Caractéristiques globales

Propriétés locales

- NxM Gram : à faire en Matlab
  - Cette méthode étend le principe du Bilevel Co-occurrence en deux dimensions
  - On retient le 2x2 Gram, qui concerne 4 cellules
  - Il y a donc $2^4 = 16$ motifs
  - On retire le motif ne comportant que des cases blanches :
    - 15 motifs nous intéressent
    - Les distances choisies restent les mêmes, celles-ci s'appliquant en hauteur et en largeur

- Résultats

Crossing count hor = [0.29293911 0.20147792 0.19401156 0.20262877 0.10894265]
Run length hor = [0.80022152 0.15538848 0.02264449 0.01561083 0.00613468]

Crossing count vert = [0.06875975 0.25584121 0.35208784 0.2316849 0.09162629]
Run length vert = [0.84966116 0.10397628 0.01516866 0.0171797 0.01401421]

Run length diag 1 = [0.50874198 0.41819312 0.05365507 0.00990799 0.00950184]

Run length diag 2 = [0.80929627 0.16698513 0.01337975 0.00388301 0.00645584]

bilevel cooccurence horizontal (d=1) = 26.0
bilevel cooccurence vertical (d=1) = 25.0
bilevel cooccurence diag1 (d=1) = 24.0
bilevel cooccurence diag2 (d=1) = 24.0

bilevel cooccurence horizontal (d=2) = 28.0
bilevel cooccurence vertical (d=2) = 26.0
bilevel cooccurence diag1 (d=2) = 22.0
bilevel cooccurence diag2 (d=2) = 24.0

bilevel cooccurence horizontal (d=4) = 36.0
bilevel cooccurence vertical (d=4) = 27.0
bilevel cooccurence diag1 (d=4) = 16.0
bilevel cooccurence diag2 (d=4) = 21.0

bilevel cooccurence horizontal (d=8) = 45.0
bilevel cooccurence vertical (d=8) = 28.0
bilevel cooccurence diag1 (d=8) = 10.0
bilevel cooccurence diag2 (d=8) = 17.0

Crossing count hor = [0.12318122 0.18431467 0.2213045 0.25314288 0.21805674]
Run length hor = [0.68725231 0.23465071 0.05175605 0.02127178 0.00506914]

Crossing count vert = [0.09142458 0.14295313 0.30557778 0.28822855 0.17181595]
Run length vert = [0.78433626 0.17280301 0.02551422 0.01087498 0.00647153]

Run length diag 1 = [0.28906096 0.53451739 0.10751059 0.0484915 0.02041956]

Run length diag 2 = [0.62897413 0.2798586 0.05765077 0.0190905 0.01442599]

bilevel cooccurence horizontal (d=1) = 26.0
bilevel cooccurence vertical (d=1) = 25.0
bilevel cooccurence diag1 (d=1) = 24.0
bilevel cooccurence diag2 (d=1) = 24.0

bilevel cooccurence horizontal (d=2) = 27.0
bilevel cooccurence vertical (d=2) = 26.0
bilevel cooccurence diag1 (d=2) = 23.0
bilevel cooccurence diag2 (d=2) = 24.0

bilevel cooccurence horizontal (d=4) = 32.0
bilevel cooccurence vertical (d=4) = 27.0
bilevel cooccurence diag1 (d=4) = 19.0
bilevel cooccurence diag2 (d=4) = 22.0

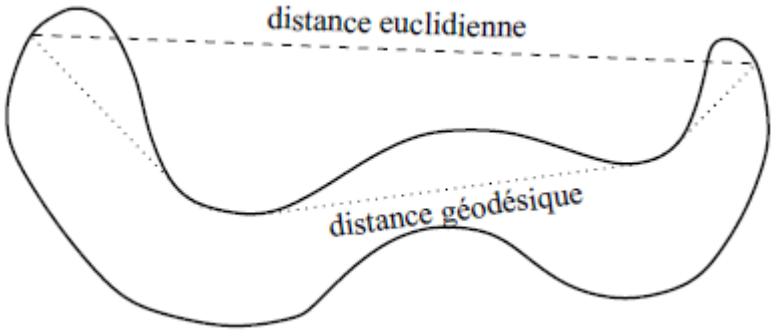
bilevel cooccurence horizontal (d=8) = 38.0
bilevel cooccurence vertical (d=8) = 25.0
bilevel cooccurence diag1 (d=8) = 14.0
bilevel cooccurence diag2 (d=8) = 24.0

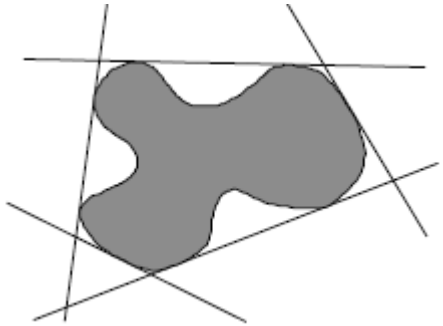
Descripteurs de contours et de formes

Composantes connexes

Composantes connexes

- Description de formes
  - Diamètre : plus grande distance entre deux points de l'objet
    - ➔ choix de la distance : euclidienne ou géodésique
  - Enveloppe convexe : intersection de tous les demi-plans contenant l'objet





Composantes connexes

- Indices de forme : à faire en matlab
  - Rapport isopérimétrique : >= 1, =1 pour un disque

$$p = \frac{carré\ \ du\ \ périmètre}{4\pi\ \ surface}$$

  - Allongement : =0 pour une courbe, =1 pour un cercle

$$p = \frac{rayon\ \ du\ \ plus\ \ grand\ \ cercle\ \ inscrit}{rayon\ \ du\ \ plus\ \ petit\ \ cercle\ \ circonscrit}$$

  - Concavité :

$$p = \frac{périmètre\ \ de\ \ l'enveloppe\ \ convexe}{périmètre\ \ de\ \ l'objet}$$

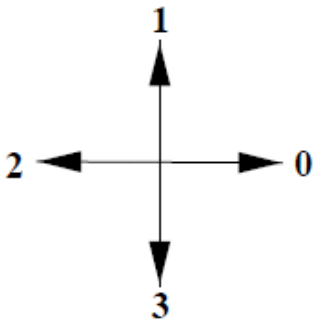
Description des contours

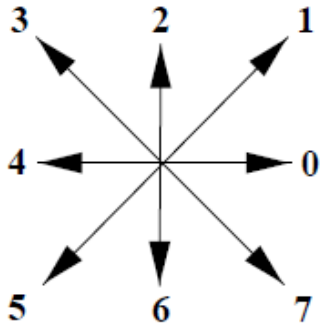
- Caractérisation de la forme par ses contours
  - Représentation des contours
    - Codage de Freeman
    - Signature
    - Descripteur de Fourier
  - Approximation polygonale des contours

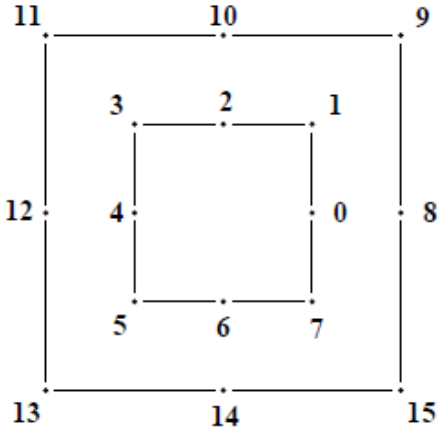
Description des contours

- Codage de Freeman
  - Méthode la plus ancienne de description des contours
  - Principe
    - Codage des directions du contour dans un repère absolu à partir d'une origine donnée
      - Coordonnées cartésiennes du premier point
      - Liste des déplacements (4-connexité sur 2 bits, 8 connexité sur 3 bits)

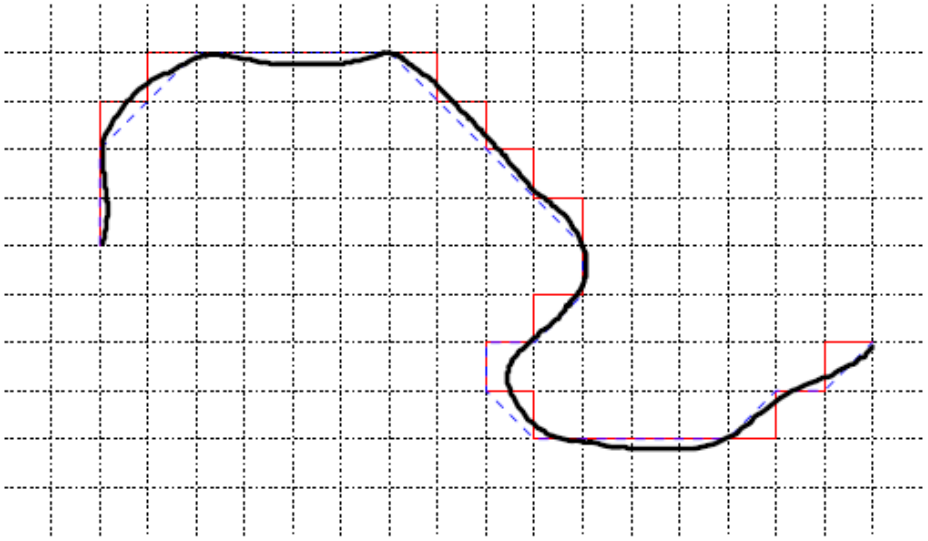
- Exemple







4 directions : 2 bits **8 directions : 3 bits** **16 directions : 4 bits**

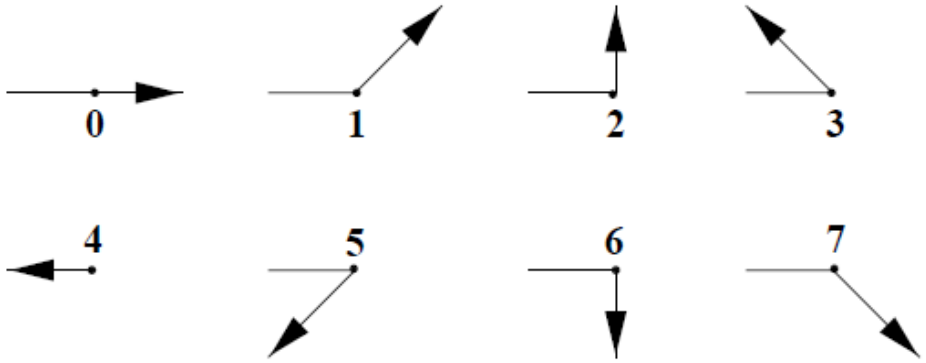


1 1 1 0 1 0 0 0 0 0 0 3 0 3 0 3 0 3 3 2 3 2 3 0 3 0 0 0 0 0 1 0 1 0 **34 x 2 = 68 bits**

2 2 1 1 0 0 0 0 7 7 7 7 6 5 4 6 7 0 0 0 0 1 0 1 **24 x 3 = 72 bits**

10 9 8 8 15 15 6 5 4 6 7 8 8 1 0 1 **16 x 4 = 64 bits**

- Codage de Freeman relatif
  - Codage de façon différentielle
    - Coordonnées cartésiennes du premier point + 1er déplacement
    - Liste des changements de direction



0 1 1 1 6 7 2 1 7 2 2 7 6

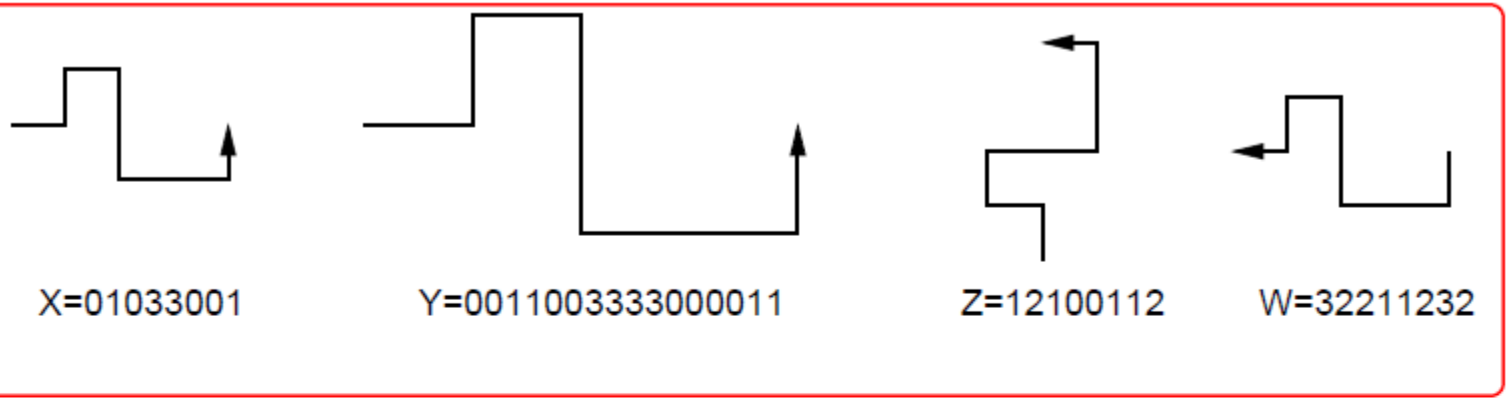
  - Propriétés
    - Invariant par translation
    - Invariant par rotation d'un multiple de 45°

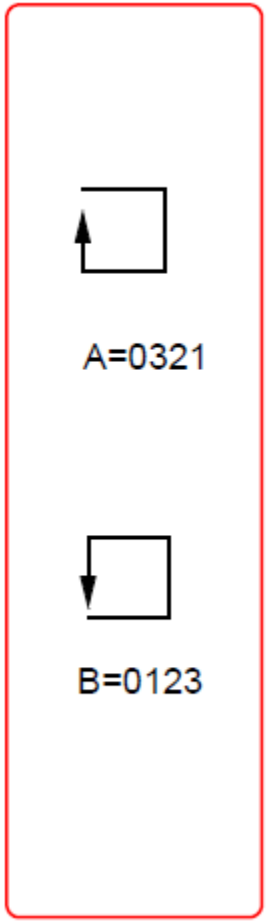
- Propriétés des chaînes de Freeman
  - Les chaînes de Freeman se prêtent à un certain nombre de manipulations commodes

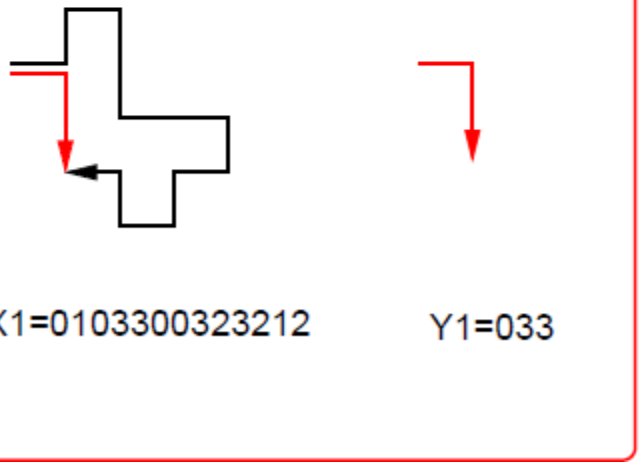
1. On obtient une dilatation de la courbe d'un facteur k en répétant k fois chaque descripteur
2. On ne peut généralement pas réduire une courbe sans distorsion
3. On fait tourner une courbe de kx2π/n (dans le cas d'une chaîne de Freeman en n connexité) en ajoutant ou retranchant k modulo n à la chaîne initiale
4. On mesure la longueur de la chaîne par les formules suivantes :
   - En 4-connexité : L = nombre de descripteurs
   - En 8-connexité : L = nombre de descripteurs pairs + $\sqrt{2}$ nombre de descripteurs impairs
5. Inversion d'un chemin : on inverse tous les descripteurs et on inverse la séquence. L'inverse d'un descripteur j est j*=n/2+ j mod(n)

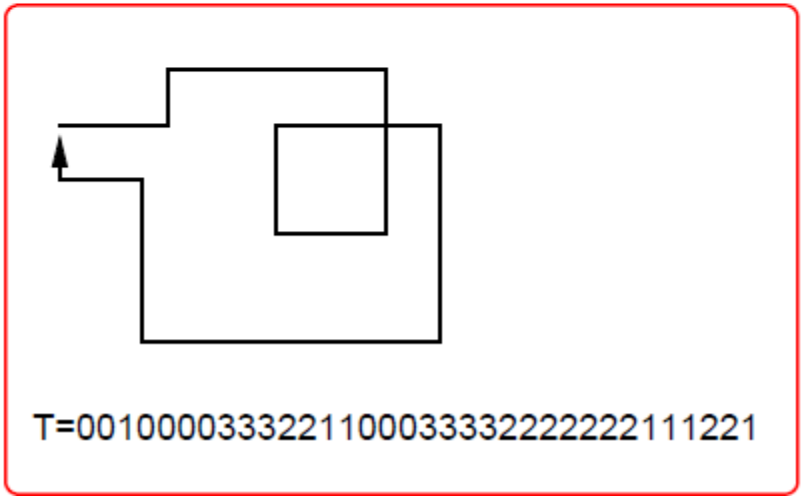
6. Simplification d'un chemin : c'est un chemin dont on a supprimé des détails sans changer globalement la forme. Cela s'obtient en remplaçant des séquences de p descripteurs par des descripteurs équivalents reliant les mêmes points
   - Exemple en 4-connexité : {012} ➔ {1}
   - Exemple en 8-connexité : {03} ➔ {2}
7. Réduction d'un chemin : c'est l'un des chemins de longueur minimale reliant les 2 extrémités de la courbe initiale
   - On associe 2 par 2 des descripteurs inverses de la chaîne et on les supprime
   - Exemple en 4-connexité : X={00132122} ➔ X*={21}
   - On obtient tous les chemins réduits en changeant l'ordre des associations
   - En 8-connexité : c'est plus complexe

8. Fermeture d'un contour :
   - on teste la fermeture d'un contour en vérifiant que la chaine réduite est nulle
9. Courbe qui s'intercepte :
   - pour savoir si une courbe décrite par sa chaîne se recoupe, on procède à une réduction de chemin systématique en partant de son origine et en testant si chaque nouveau descripteur possède un inverse dans la chaine déjà parcourue
   - Si à un instant la chaine déjà parcourue se réduit à une chaine nulle, on a trouvé un point double
10. Changement d'origine d'une chaîne de longueur L :
   - revient à une permutation circulaire des descripteurs modulo L







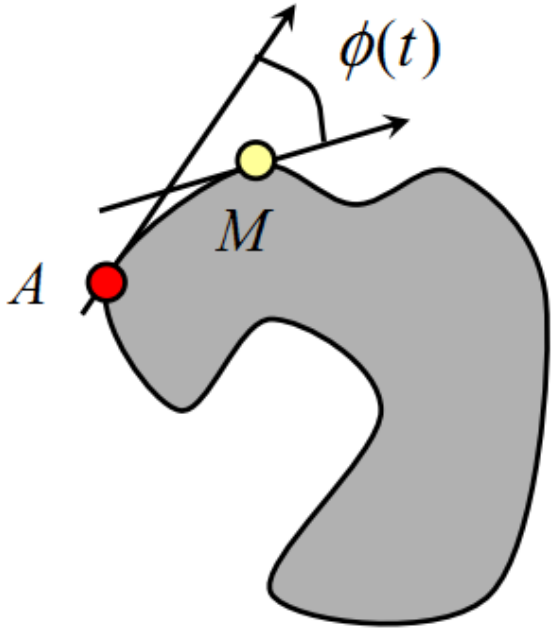


X = chaine originale, Y = contour double, Z = rotation de $\pi/2$, W = contour inverse

X1 = chaine originale, Y1 = chaîne réduite, T = exemple de boucle dans une chaîne

A et B sont les deux contours élémentaires (inverse et direct) en 4-connexité.

- Descripteurs de Fourier
  - Descripteurs par tangente
    - Dans cette approche, on considère le contour comme une courbe continue qui peut être décrite par son abscisse curviligne s à partir d'une origine A
    - On paramètre la courbe par l'angle fait par le vecteur tangent en chaque point et celui au point d'origine : $\phi(s)$
    - et on crée la variable réduite t (on la réduit en l'obligeant à prendre des valeurs entre 0 et 2 π) : $t= 2\pi s/L$, où L est la longueur complète du contour



- On construit alors la fonction $\phi(t)$:

$$\phi(t) = \phi[\frac{2\pi.s}{L}] - \frac{2\pi.s}{L}$$

- Le dernier terme est un terme correctif pour qu'on n'aille pas au-delà d'un tour du contour
- La fonction $\phi(t)$ est une fonction périodique sur $[0, 2\pi[$ qui admet donc une série de Fourier

$$\phi(t) = \sum_{k=0}^{\infty} a_k e^{(-ikt)}$$

- On appelle descripteur de Fourier l'ensemble des modules des a_k : $|a_k|$
- Ils bénéficient des propriétés suivantes :
  - invariants par translation de la forme
  - invariants par changement d'échelle (puisque t est normalisé)
  - invariants par rotation (puisque l'on a choisi la différence d'angles entre 2 tangentes)
  - invariants par changement d'origine
- Pour comparer des formes, on compare leurs descripteurs par ordre croissant
- Idée :
  - Création d'une autre représentation permettant une interprétation géométrique plus simple

- Descripteurs de Fourier
  - Représentation complexe
    - Ici, la forme est décrite par un ensemble M_j de points de contours
    - Et on représente la forme dans le plan complexe
    - On attache donc à chaque M_j un nombre complexe $z_j=x_j+iy_j$
    - On appelle alors descripteurs de Fourier, les coefficients de la transformée de Fourier discrète Z des z_j :

$$Z_k = \sum_{i=0}^{N-1} z_i \exp(-j2\pi ik)$$

    - avec k $\in$[-N/2+1;N/2]
- InterfaceMatlab4 : foirier

Exemple : représentation de fontes

- Problème :
  - On cherche à classer des textes imprimés dans des fontes et styles différents à partir de la représentation de leurs mots

Doc-times / Doc-Arial

The density of the text depends on the font family which usu ally rep resents its class. The com puter can takes several dec isions dep ending on the boun dary of their characters if this is

Times Roman 10

The density of the text depends on the font family which usually represents its class. The computer can ta kes several decisions depen ding on the boundary

Arial 10

The density of the text depends on the font fa mily which usually re presents its class. The computer can takes seve ral decisions depending on the boundary of the ir characters

Courier 9 Italic Bold

  - Voici des exemples de mots écrits dans différents styles et fontes

MotsImprimes

boundary	**boundary**	*boundary*	***boundary***	boundary	**boundary**	*boundary*	***boundary***
class	**class**	*class*	***class***	class	**class**	*class*	***class***
computer	**computer**	*computer*	***computer***	computer	**computer**	*computer*	***computer***
decision	**decision**	*decision*	***decision***	decision	**decision**	*decision*	***decision***
density	**density**	*density*	***density***	density	**density**	*density*	***density***

Les fontes choisies

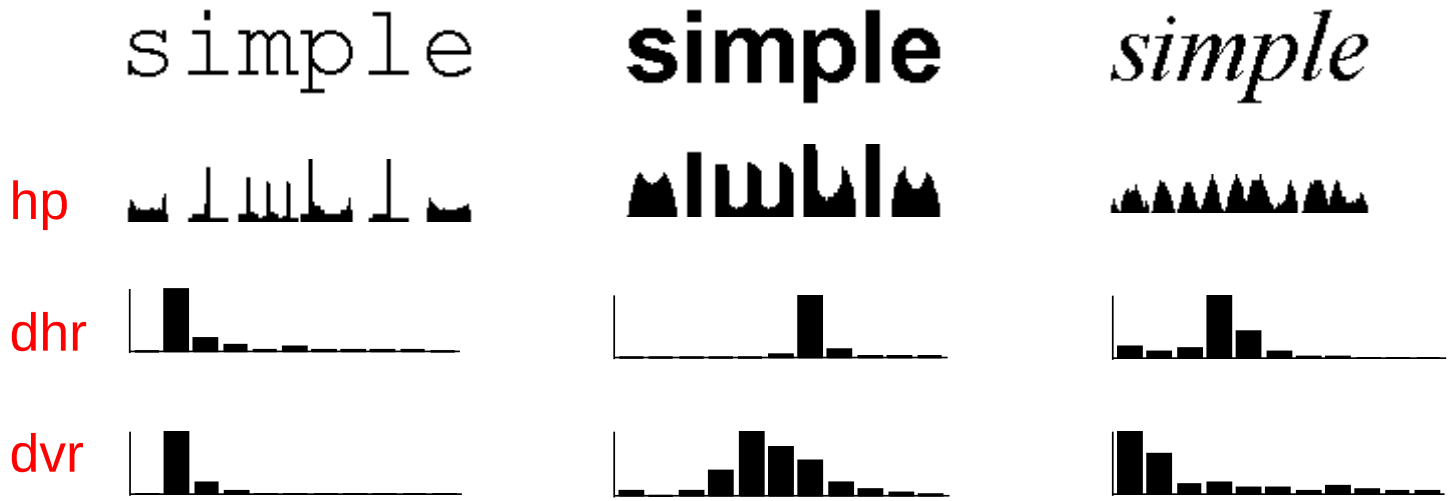
- Un ensemble de 12 fontes standards :
  - 3 polices (Arial, Courier, Times),
  - 4 styles
    - Plain
    - Italic
    - Bold
    - Bold Italic
  - même taille : 12 pt
- Les images sont :
  - binaires
  - numérisées à 300 dpi

La base

- Fontes : répertoire : « base/MotsImprimes »
  - contient 8 fichiers, chacun a 13 images de mots
  - font1 : Times Normal
  - font2 : Times Gras
  - font3 : Times Italique
  - font4 : Times Gras/Italique
  - font5 : Arial Normal
  - font6 : Arial Gras
  - font7 : Arial Italique
  - font8 : Arial Gras/Italique
- Paramètres : répertoire : « Fontes/Fonte1.txt », « Fontes/Fonte2.txt »…
  1. hp-mean (or density) : moyenne de hp (horizontal projection)
  2. hr-mean : moyenne des horizontal runs (up to length 12)
  3. hpd-stdev (or slanting) : écart type de hpd (horizontal projection derivation)
  4. hr-stdev : écart type des horizontal runs (up to length 12)
  5. vr-stdev : écart type des vertical runs (up to length 12)
  6. vr-mean : moyenne of vertical runs (up to length 12)
- Chaque fichier : $Fonte_i$.txt contient : 6 x 13 valeurs

Les primitives

- Illustration des primitives utilisées
  - horizontal projection profile
  - distribution of horizontal runs (from 1 to 11)
  - distribution of vertical runs (from 1 to 11)



Interface RDF

- Rubrique : File
  - Choisir un mot : base/font$_k$/mot$_i$.png
- Rubrique : Primitives
  - Choisir : Descripteurs_mot_font
    - Imprime
      - l'image du mot
      - et 3 primitives : Hrmean, VrMean, hpd

Exemple 1

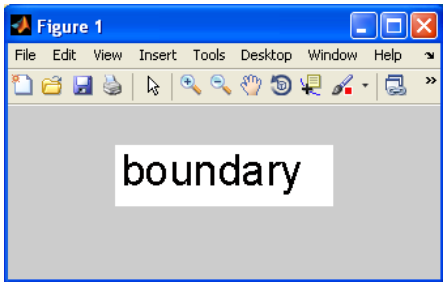


Figure 1
File Edit View Insert Tools Desktop Window Help
boundary

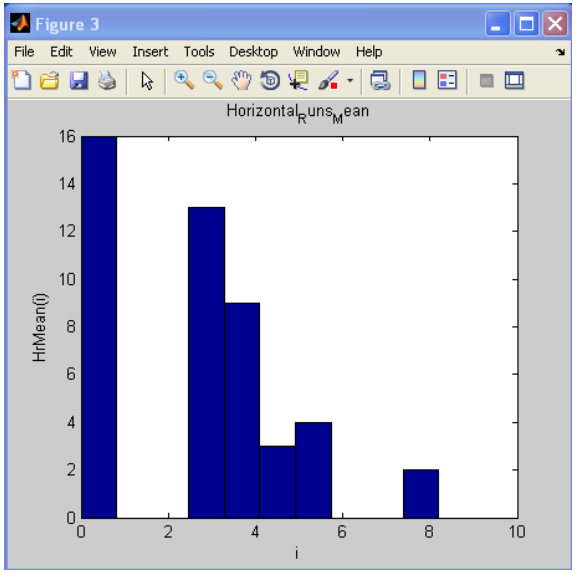


Figure 3
File Edit View Insert Tools Desktop Window Help
HrMean(i)
i

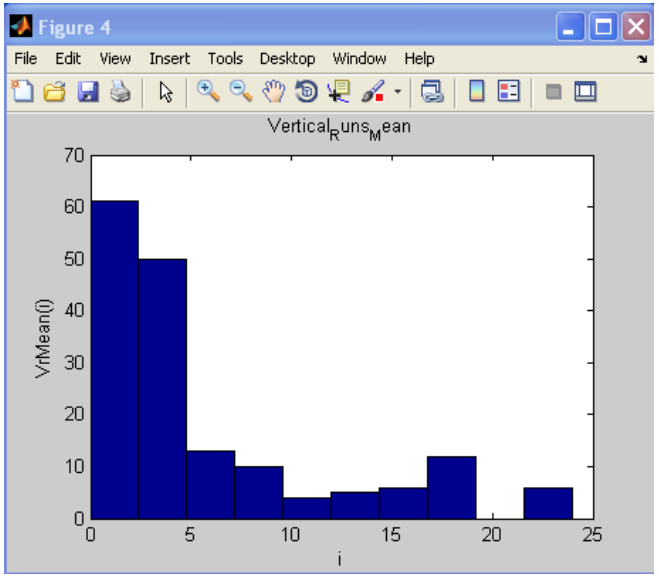


Figure 4
File Edit View Insert Tools Desktop Window Help
VrMean(i)
i

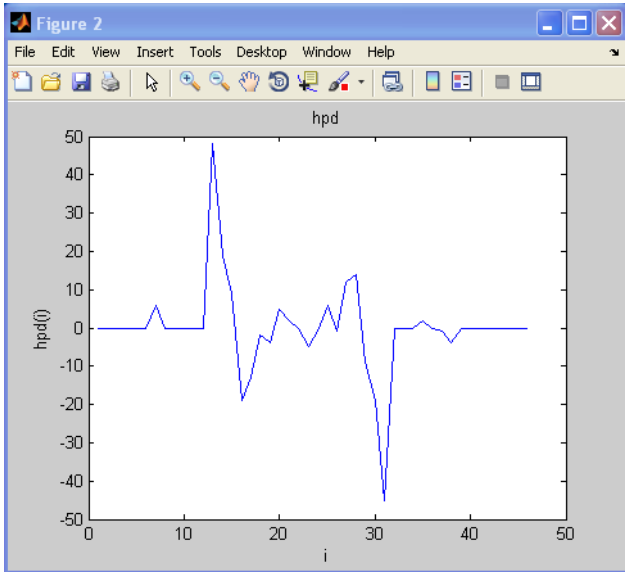


Figure 2
File Edit View Insert Tools Desktop Window Help
hpd
hpd(i)
i

Exemple 2

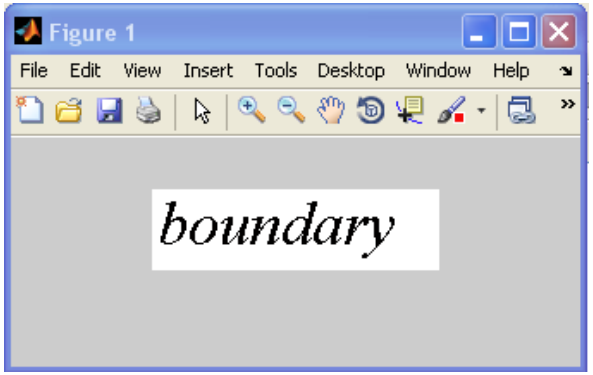


Figure 1
File Edit View Insert Tools Desktop Window Help
boundary

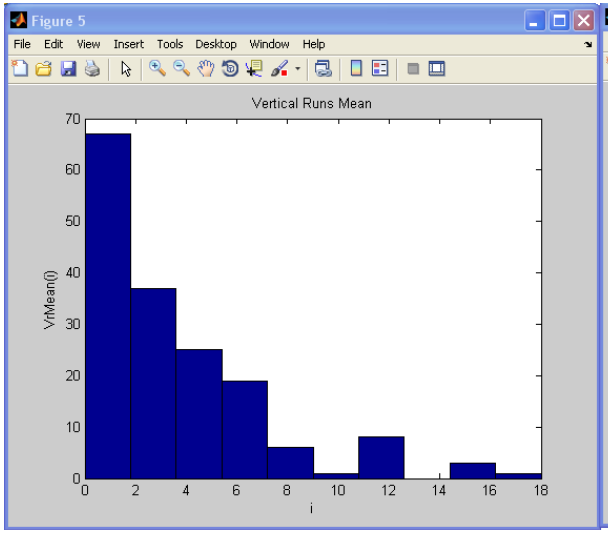


Figure 5
File Edit View Insert Tools Desktop Window Help
Vertical Runs Mean
VrMean(i)
i

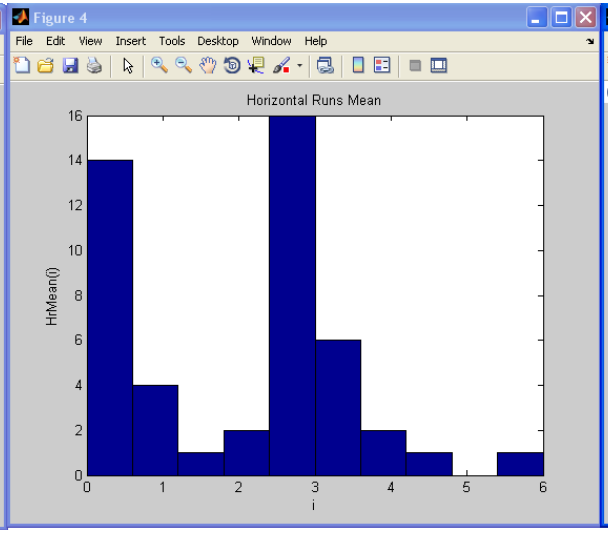


Figure 4
File Edit View Insert Tools Desktop Window Help
Horizontal Runs Mean
HrMean(i)
i

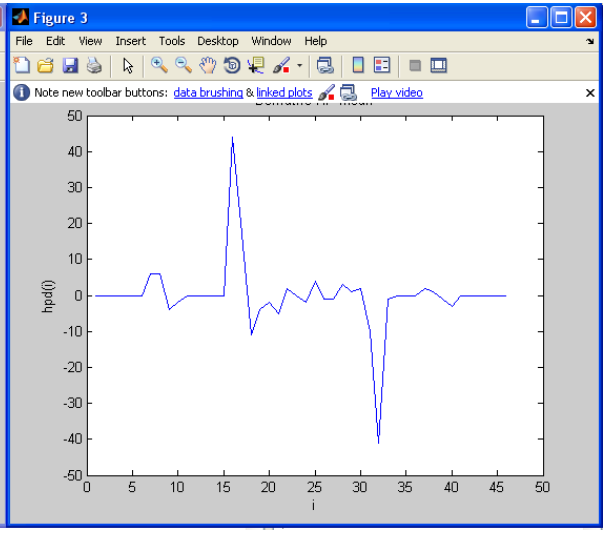


Figure 3
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots Play video
hpd(i)
i

Extraction de paramètres

- Sous Matlab
  - Interface Matlab :
    - Programme : Data_generator.m
      - Calcule les paramètres pour tous les mots d'une fonte (13 x 6) et range le résultat dans : Fonte8.txt
    - Gener_fontes
      - Initialise en dur des vecteurs de paramètres aux primitives des fontes

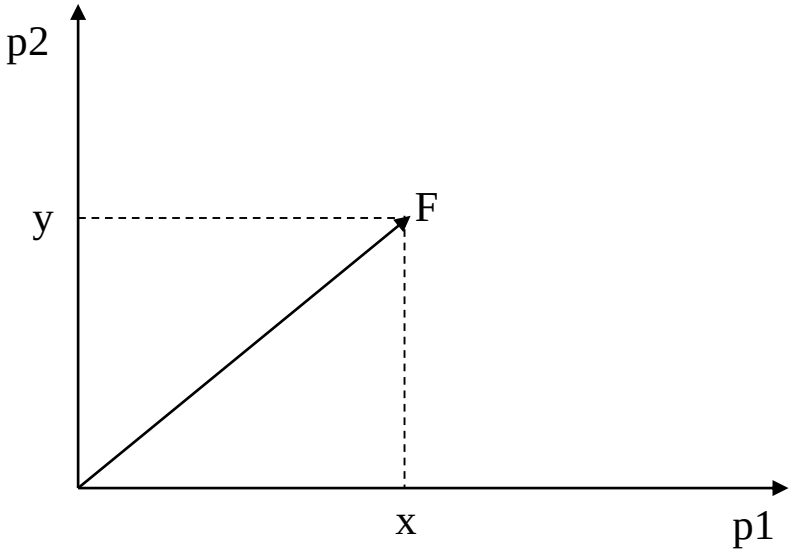
La base

- Exemple de Fonte8.txt : 13 mots, 6 paramètres (fonte1)

1.441915e+002	2.555927e+000	1.178700e+001	2.128454e+000	6.549255e+000	5.334331e+000
1.535319e+002	2.555015e+000	8.387822e+000	3.049856e+000	4.230460e+000	2.135729e+000
1.446596e+002	2.435558e+000	1.299915e+001	2.081720e+000	6.528934e+000	5.185629e+000
1.465532e+002	2.177212e+000	1.162946e+001	2.477387e+000	5.258731e+000	3.827345e+000
9.725000e+001	3.108523e+000	1.053565e+001	2.069492e+000	6.142959e+000	5.203081e+000
1.472128e+002	1.938970e+000	1.520088e+001	2.111690e+000	7.022661e+000	5.818095e+000
8.805263e+001	2.096898e+000	7.731609e+000	2.480529e+000	5.526396e+000	3.851485e+000
1.290278e+002	2.691390e+000	1.628966e+001	2.279730e+000	5.985792e+000	5.105208e+000
9.163889e+001	3.054623e+000	1.330635e+001	2.590170e+000	6.550155e+000	5.289855e+000
8.352778e+001	2.352767e+000	9.949874e+000	2.792387e+000	6.057398e+000	4.859736e+000
9.627027e+001	2.634521e+000	1.159803e+001	2.445712e+000	7.160642e+000	6.133333e+000
98.0	3.085580e+000	1.222094e+001	2.124242e+000	7.192165e+000	5.594444e+000
1.344474e+002	3.038822e+000	1.474600e+001	1.800296e+000	7.220974e+000	5.900794e+000

Représentation de formes

- Notion de vecteur de forme
  - Une forme F représentée par deux paramètres x et y peut être représentée par un point dans le plan (extrémité du vecteur)

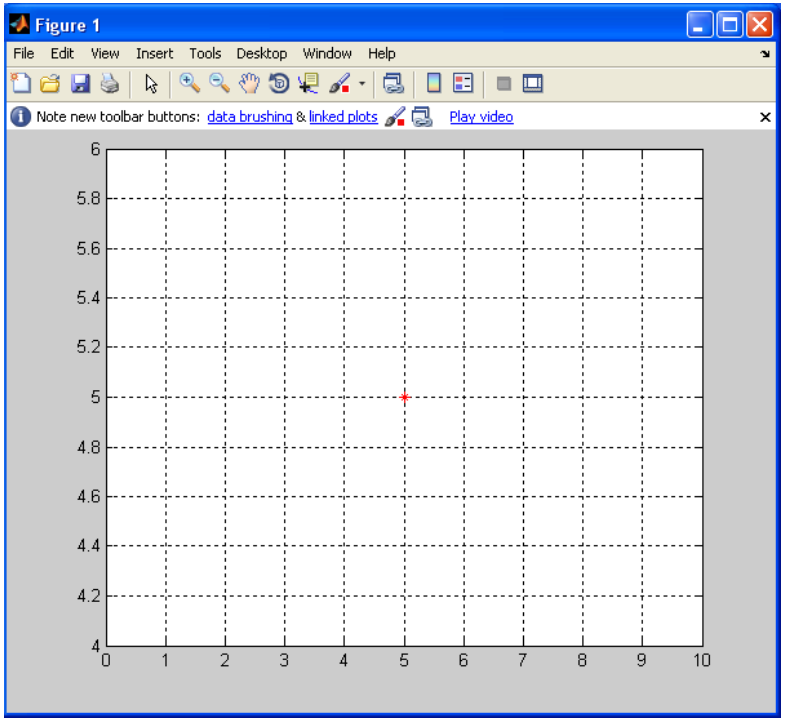


Représentation de formes

Matlab : Distributions >> Vecteur

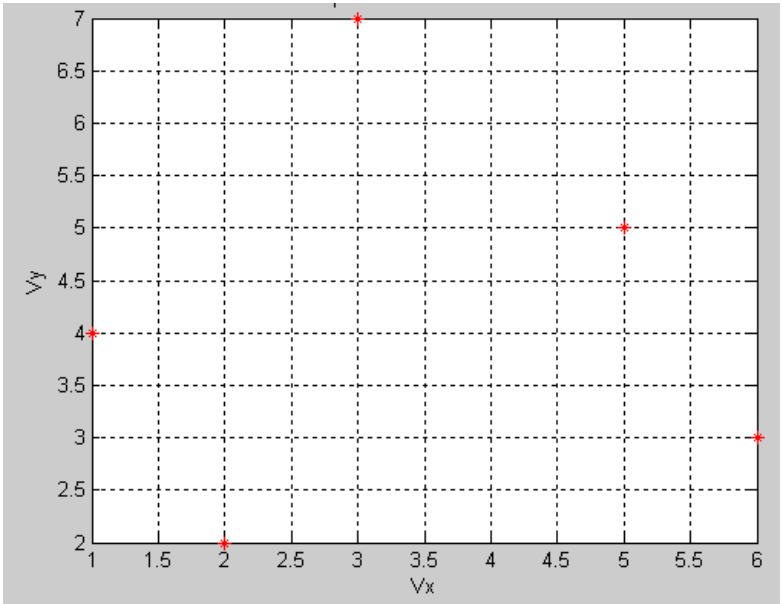
- Représentation en matlab

```
Vx=[5];
Vy=[5];
figure,plot(Vx,Vy,'r*');
grid('on');
axis([0 10 10 0]);
title('Représentation de
    vecteur');
xlabel('Vx'), ylabel('Vy');
```



- Distribution de plusieurs formes : Distributions >> PlusVecteurs

```
V1x=[5];V1y=[5];
V2x=[1];V2y=[4];
V3x=[6];V3y=[3];
V4x=[3];V4y=[7];
V5x=[2];V5y=[2];
figure,plot(V1x,V1y,'r*');
grid('on');
xlabel('Vx'), ylabel('Vy');
hold on
plot(V2x,V2y,'r*');
hold on
plot(V3x,V3y,'r*');
hold on
plot(V4x,V4y,'r*');
hold on
plot(V5x,V5y,'r*');
```



Représentation

- Début de classification
  - Nous allons voir quels sont les paramètres qui permettent déjà à ce niveau de séparer les fontes
  - Pour cela nous allons sélectionner des couples et des triplets de paramètres et projeter les vecteurs de ces paramètres dans un espace 2D
  - Exemple 1 :
    - Distribution de fonte 1 et fonte 2 par les paramètres :
      - Hp-mean et Hr-stdev
    - Construction de deux vecteurs par fonte
      - un donnant les abscisses et l'autre les ordonnées, ainsi chaque fonte sera distribuée dans le plan par des points 2D
    - Fonte 1 :
      - X1=Hp-mean(fonte1), Y1= Hr-stdev(fonte1)
    - Fonte 2 :
      - X2=Hp-mean(fonte2), Y2= Hr-stdev(fonte2)

Représentation

- Code Matlab : DistrF1F2

```
X1=param(:,1,1)';X2=param(:,1,2)';
Y1=param(:,2,1)';Y2=param(:,2,2)';
figure,plot(X1,Y1,'o');
xlabel('Moyenne du profil horizontal');ylabel('Ecart type(Horizontal
    Runs)');
hold on;
plot(X2,Y2,'r*');
legend('fonte1','fonte2');
```

Résultats

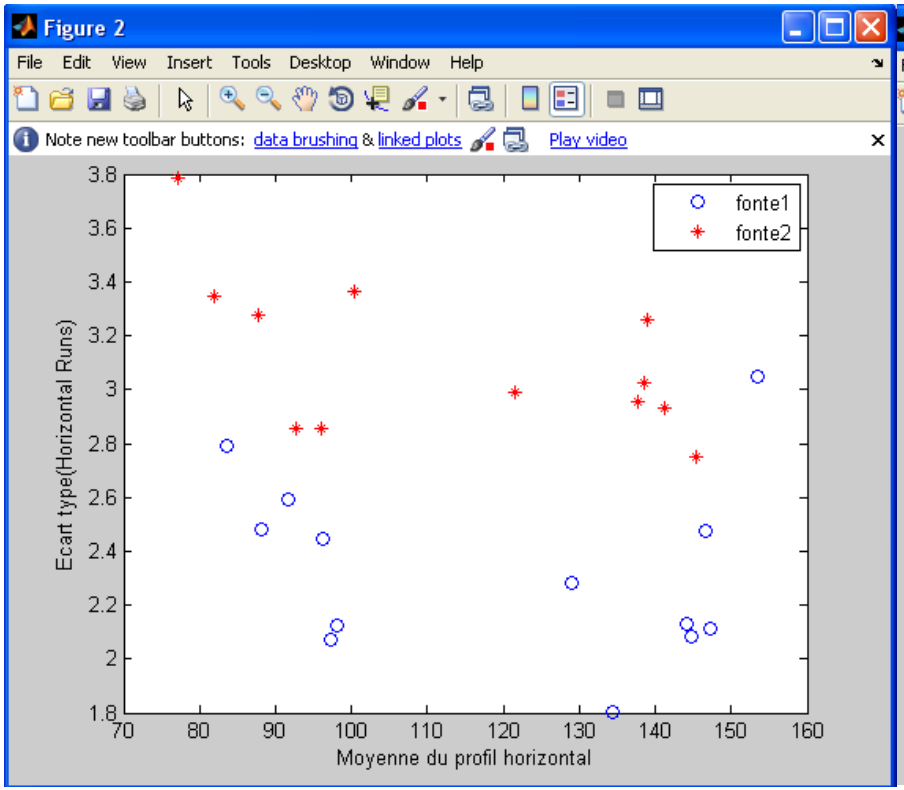


Figure 2
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots Play video
fonte1
fonte2
Ecart type(Horizontal Runs)
Moyenne du profil horizontal

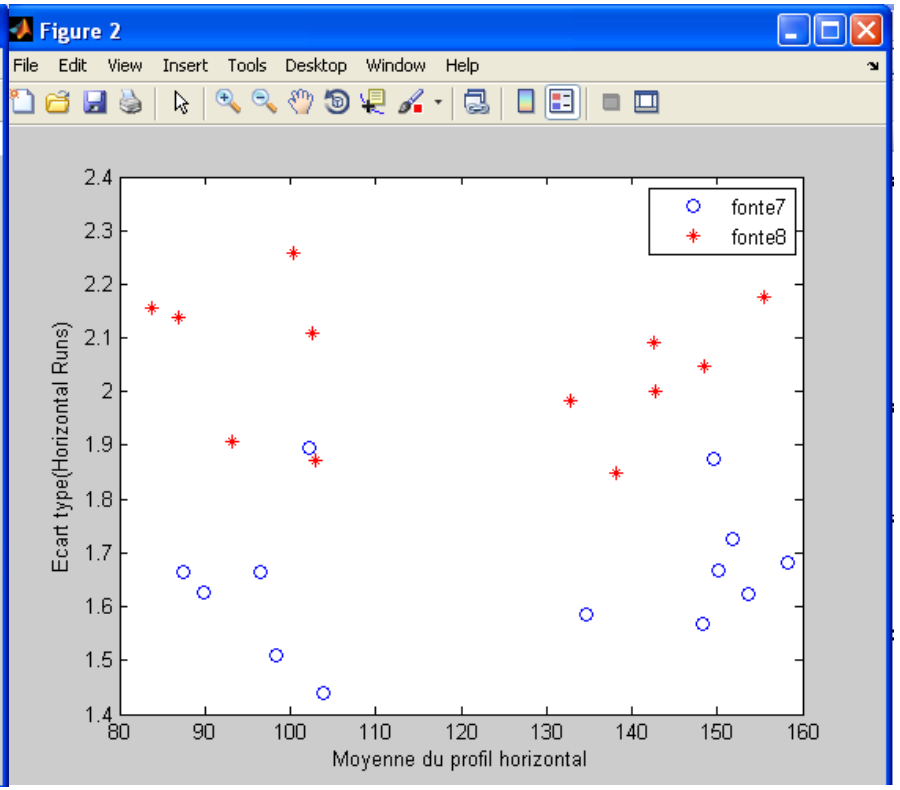
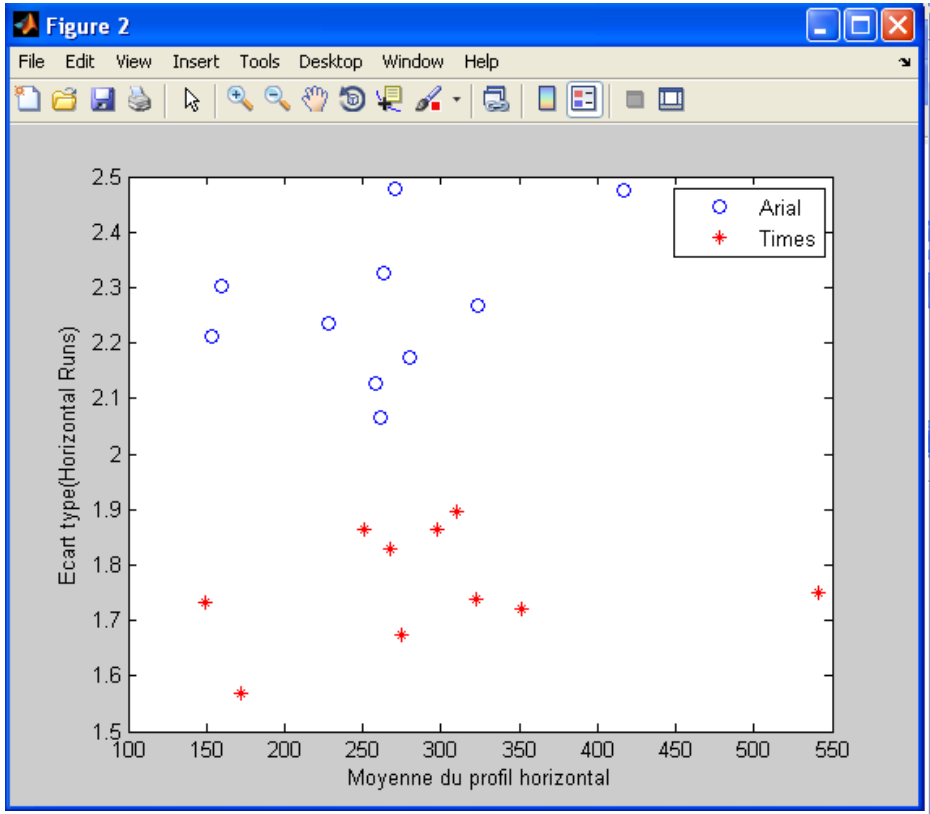


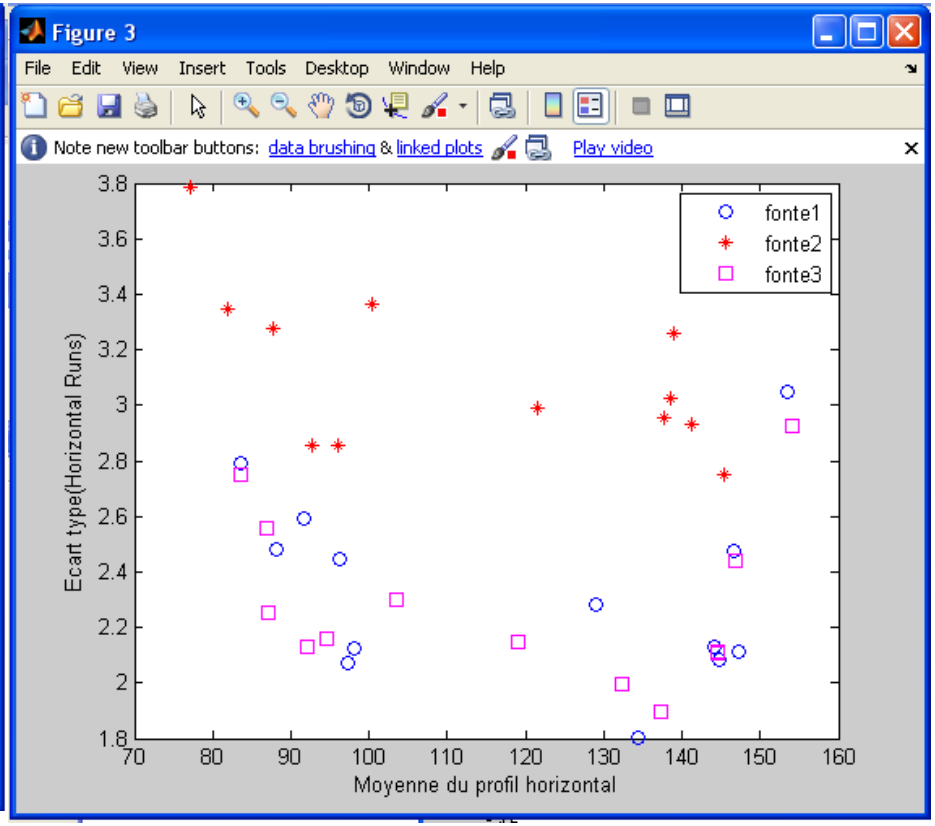
Figure 2
File Edit View Insert Tools Desktop Window Help
fonte7
fonte8
Ecart type(Horizontal Runs)
Moyenne du profil horizontal

Autres exemples

Distr_Arial_Times.m



DistrF1F2F3.m



Autre représentation

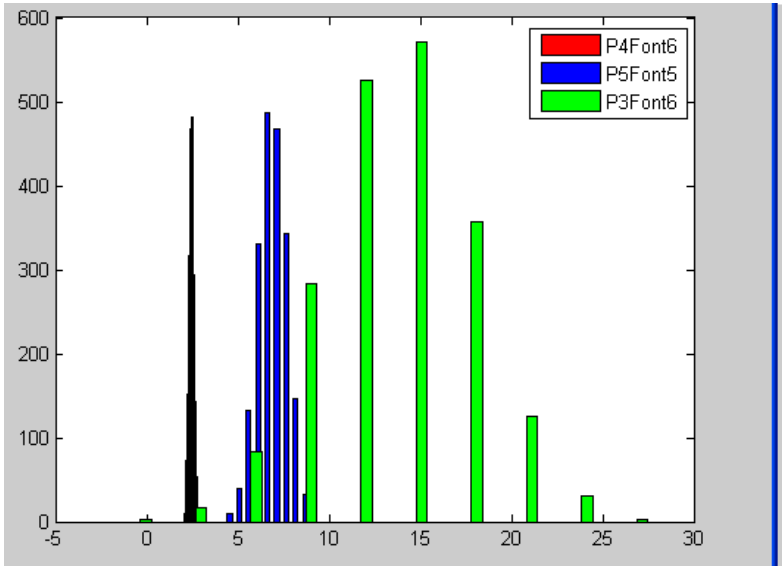
Distributions >>DistHistEcart

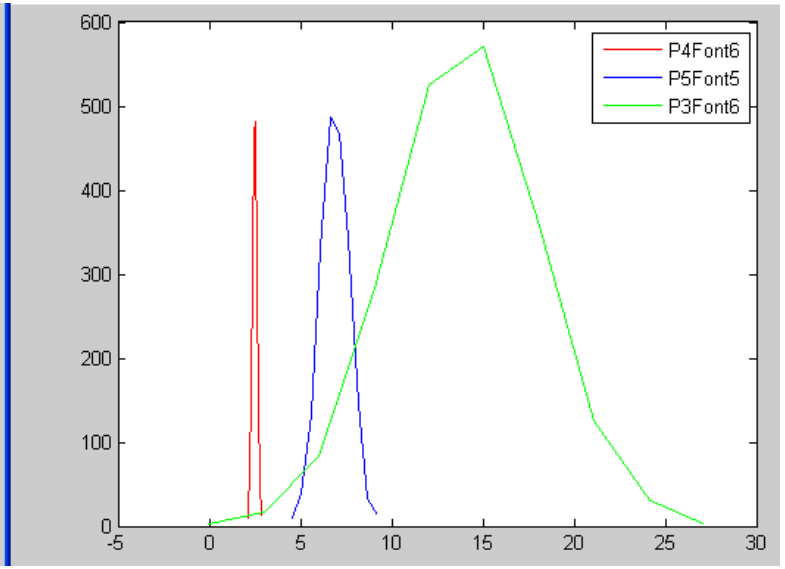
- On calcule l'histogramme des écarts par rapport à la moyenne de leur classe
  - Soit la classe P5fonte6
    - P5fonte6=[7.2796 5.1047 7.1598 6.1971 6.7633 7.3506 6.0305 6.2874 7.0858 6.7242 7.9133 7.5838 7.8342];
  - Calcul de la moyenne et de l'écart type
    - [moyenne,et] = normfit(P5fonte6);
  - Calcul de l'histogramme des écarts par rapport à la moyenne et compte tenu de l'écart type
    - R=normrnd(moyenne,et,2000,1);
    - [X,Y]=hist(R2);

Autre représentation

Distributions >>DistHistEcart

- On voit le chevauchement entre les classes
  - Les paramètres P3, P4 et P5 discriminent bien
    - P3 : hpd-stdev (or slanting) : écart type de hpd (horizontal projection derivation)
    - P4 : hr-stdev : écart type des horizontal runs (up to length 12)
    - P5 : vr-stdev : écart type des vertical runs (up to length 12)

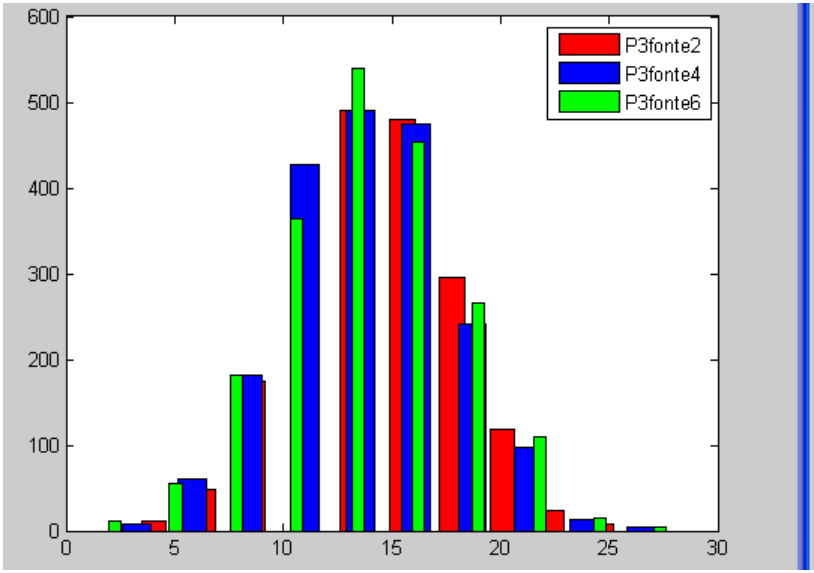


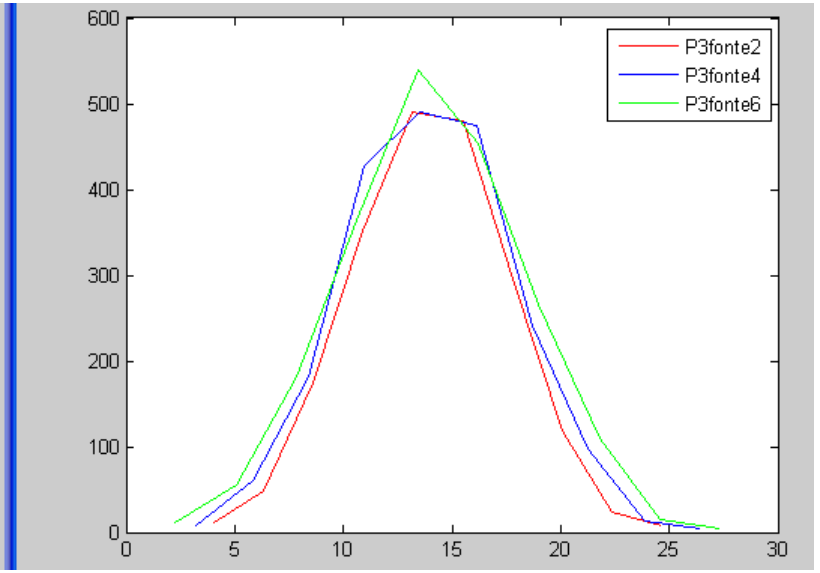


Autre distribution

Distributions >>DistHistEcart

- Par contre
  - Les fontes ne sont pas discriminées par le paramètre 3
  - Au contraire, elles se ressemblent beaucoup

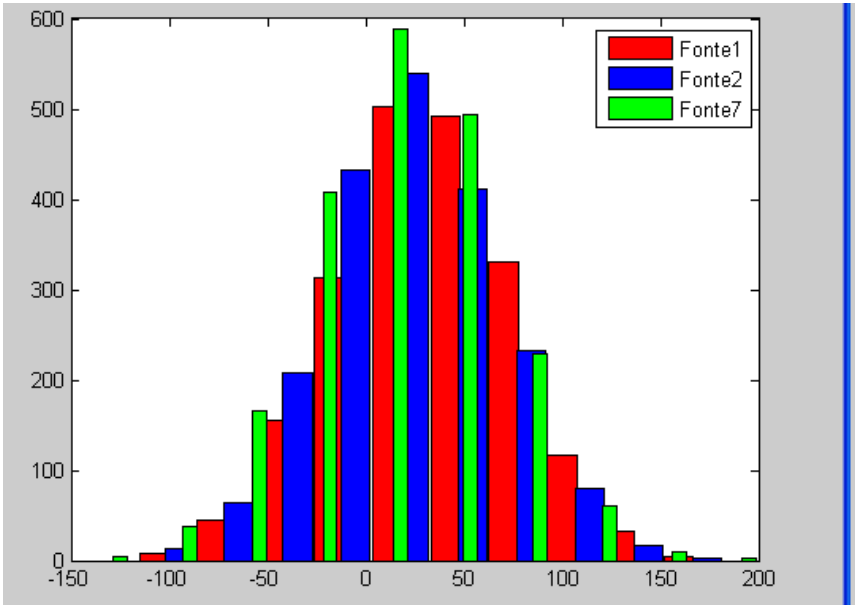


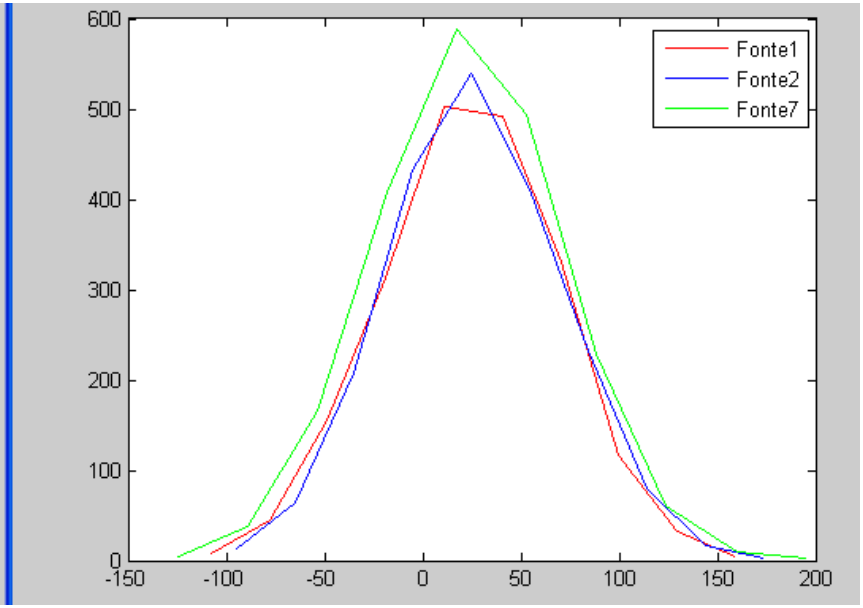


Autre distribution

Distributions >>DistHistEcart

- De plus
  - Les fontes ne sont pas discriminées par la globalité des 6 paramètres





Réduction de la dimensionnalité

- Problème
  - Certaines formes (vecteurs) sont représentées par plusieurs paramètres (composantes). Comment réduire ces composantes pour pouvoir les afficher dans le plan ?
- Techniques classiques
  - Analyse en composantes principales (ACP)
    - préserve la variance
  - Échelonnage multidimensionnel (Multidimensional Scaling)
    - préserve la distance inter-points
  - Locally Linear Embedding
    - préserve le voisinage

Analyse en composantes principales

- Trouver
  1. La principale composante (PC1)
     - La direction le long de laquelle se trouve la plus grande variation
  2. La principale composante (PC2)
     - La direction avec le maximum de variation restante dans la donnée, orthogonale à la direction (*i.e.* vecteur) de PC1
  3. La principale composante (PC3)
     - La direction avec variation maximale restante dans la donnée, orthogonale au plan de PC1 et PC2
     - (rarement utilisée)
  - etc.