# Correction de la durées des notes

Cet algorithme s’occupe de la correction de la durée des notes de manière à ce qu’elles puissent être en accord avec un découpage en mesure 4 :4.

## Explications :

Prenons la série de notes suivante :

durees = [4.3 4.2 5 4.6 4.4 4.7 2.2 2.3 2.2 2.1]

On rappelle que la durée d’une note est égale au nombre de doubles-croches qui la compose (2 = croche, 4 = noire, 5 = noire + double-croche) et qu’une mesure 4 :4 comporte 16 doubles-croches.

Le but ici est donc de corriger les durées pour que cette séquence puisse être découpée en multiple de 16.

## Description par étape de l’algorithme

Cet algorithme est divisé en deux grandes parties :

* La détermination des durées et de leurs probabilités
* La correction par multiple de 16

### La détermination des durées et de leurs probabilités

Nous avons décidé de baser les probabilités d’apparition de chaque durée sur des gaussiennes. En effet, ce modèle s’accorde bien avec notre système. Nous avons donc générer un peigne de gaussiennes (une pour chaque durée) afin de pouvoir établir la matrice "out" suivante :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Durées en Entrée (DE) | Durée Tronquée (DT) | Proba DT | Durée Supérieure (DS) | Proba DS |
| 4 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
|  |  |  |  |  |
| 2 |  |  |  |  |
| 2 |  |  |  |  |

Pour remplir cette matrice, on a créé la fonction

Out = determinationDurees(durees, peigneGaussienne, abscisse)

durees : DE

peigneGaussienne : probailité d’apparition de chaque durée

abscisse : Attribue un indice en fonction de la durée en entrée. Cet indice permet ensuite de trouver la probabilité de la DE sur le peigne de gaussienne.

Cette fonction opère de manière itérative sur chaque ligne (représenté par m) comme suit :

#### Première colonne :



Simple recopie de la valeur en entrée.

#### Deuxième colonne :



Afin de gérer les durées qui ne seraient pas des valeurs entières, on tronque les durées en entrée.

#### Troisième colonne :



Pour out(m, 3), nous procédons ainsi :

On cherche l’indice de la valeur DT(m), ce qui nous donne index(m).

On cherche ainsi la probabilité (différente de 0) sur la gaussienne la plus proche. Si cette probabilité est 0, on descend DT de 1, et on recommence. Si DT est égale à 0, on arrête la boucle de telle sorte que out(m, 2) = 0 avec out(m, 3) = sa probabilité.

Cette méthode est écrite de manière à pouvoir gérer des durées non-entières. En effet, si chaque durée entière possède une probabilité non nulle, ce n’est pas le cas des durées décimales.

#### Colonne 4 :



On cherche la valeur supérieure de DT.

#### Colonne 5 :



Out(m, 5) est calculé de la même manière que out(m, 3) mais cette fois-ci pour des valeurs supérieures.

On cherche une probabilité différente de 0, et si out(m, 4) = 16, on arrête la boucle et out(m, 5) = 0.

Dans notre exemple avec la série suivante :

durees = [4.3 4.2 5 4.6 4.4 4.7 2.2 2.3 2.2 2.1]

Cela nous donne une matrice "out" comme ci-dessous :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Durées en Entrée (DE) | Durée Tronquée (DT) | Proba DT | Durée Supérieure (DS) | Proba DS |
| 4.3 | 4 | 0.8538 | 16 | 0 |
| 4.2 | 4 | 0.9322 | 16 | 0 |
| 5 | 5 | 1 | 6 | 0.0795 |
| 4.6 | 4 | 0.5312 | 16 | 0 |
| 4.4 | 4 | 0.7550 | 16 | 0 |
| 5.1 | 5 | 0.9041 | 6 | 0.1286 |
| 2.2 | 2 | 0.9407 | 16 | 0 |
| 2.3 | 2 | 0.8714 | 16 | 0 |
| 2.2 | 2 | 0.9407 | 16 | 0 |
| 2.1 | 2 | 0.9849 | 16 | 0 |

Cette matrice va ensuite pouvoir être traitée par la deuxième partie de l’algorithme.

### La correction par multiple de 16