Guide d'utilisation du projet python 2019-20120

Auteur : Cornier Alexandre et Cuhadar Ali

1° Introduction :

Les nouvelles technologies de séquençage nous permettent aujourd'hui de lire le génome d'un individu rapidement et à moindre cout. Le but de ce projet est de reproduire de manière simplifiée les différentes étapes du processus de séquençage. Pour cela, nous avons développé un code en utilisant :

- Python 3.7.0 comme langage de programmation
- PyCharm qui est un environnement de développement (IDE) utilisé pour programmer en python

2° Les différentes fonctions :

Le code est composé de 10 fonctions :

- **Alphabet(type)** qui permet de définir les lettres qui compose une séquence d'ADN
- **Randseq(num, alpha)** qui permet de créer une séquence d'ADN de 1 000 nucléotides, ajoutés aléatoirement à partir de la première fonction.
- **Cut(seq)** permet de couper la séquence d'ADN en donnant le choix à l'utilisateur de la taille des reads. Cette fonction permet aussi de retourner à l'utilisateur la liste des reads ainsi que la qualité de chaque nucléotide associé.
- **Liste_adaptateur()** qui demande à l'utilisateur le nombre d'adaptateurs, et de rentrer le(s) motif(s) que l'on stocke dans une variable.
- **Nombre_adaptateur(listcutseq, adaptateur)** permet de repérer dans chaque read le motif d'un adaptateur dans le sens forward et reverse et de compter le nombre de motifs au total.
- **Filtre_adaptateurs(listcutseq, listqual, adaptateur)** permet de supprimer les motifs correspondant aux adaptateurs présents dans chaque read dans le sens forward et reverse, mais également de supprimer les qualités associées.
- **Filtre_extrémité(listcutseq, listqual)** permet de supprimer, à chaque extrémité, les qualités inférieures à un seuil donné par l'utilisateur et les nucléotides associés.
- **Filtre_moyenne(listcutseq, listqual)** demande un seuil moyen à l'utilisateur, si la qualité moyenne d'un read est inférieure à ce seuil alors le read est supprimé.
- **Filtre_read(listcutseq)** demande à l'utilisateur un motif, le compare à l'ensemble des reads et retourne le ou les reads qui produisent le chevauchement maximal. Ce chevauchement doit impliquer l'une des deux extrémités de chacun des deux reads.
- **Flatten(listcutseq)** permet de réaliser l'assemblage de reads obtenus après les étapes de filtrage. Ce qui correspondra à la séquence génomique qui avait généré l'ensemble des reads de départ.

3° Utilisation :

Quand on lance le programme, tout d'abord le code affiche la séquence d'ADN d'une taille de 1 000 nucléotides :

```
(base) macbook-pro:Desktop Alex$ python3 projetOBP-final.py
GCGAGAATGCAAAACGCGCAAGGACCATTGACTGCCAGAAAGTTCATCTTTCTAATTAATCCCTCAGAAGAGTGTTACCCCCCATTCAGTGCAAACCTTCCGATGGGCTACGGGATAGCACAGCACGACAGCGAAGCTCAGGA
CCACATAAGAAGAACGGGTTTGGCCCGCCTCGACCTAAAGCGGAATCAGTGGCGCAGTCAATGGTATCCTCTCACCGCACGGCCAATGATGGCGACTATGTGGGCCCGTGGTCCCGACTGGTTCGCCACGTGGACCTATTTGC
CCGGGTTGATATAGTGGCACTGTGAAATTGCAGAGTCCGGAGTCCCGACATAGATTGTCGTAATCGGAGGCTACGACTCTTGGTCGTGAATTACAAGTCCGGAACTTATATTATTCCTCTCTTATTTTTTCTGACATAATTGT
GGTCATGCGAGGATGCCTCTTAGGCTATCAAGGCCGGCCTCAGACGTGACTATGCAACTGATTGCGCACTCGGACCTAGCTTATTTAAACAATCTCACGAACACTCACTGGTGGATGATTAAGGTCACGTGATCCGCACCATA
CTGCAATCATATGTCCAAAGGTGCGCATTGATTTGTGCTTTCACTAAGGCCTAGATACATAGCGTAACACGGGGGAGCCCGAAACTCGACTGTCAATGTGACTTTTGGCTACAGAGGAGCATTGTACCATTCCCAGATTGAAG
GGTCTAAATACTTTAAATATACCCCCATAGAGCGACCAGCGTCGGTTGACTCAGGGTTCCTGTAGGGGTTAGAGCAAATTCCCCATCCAAGTTCTCGTTGGGGTCCCTGAGGGTTCGGCCGGTTAGTTTTACCTAGGTAATGG
CACTAACCTCCTCCGATCCGGGCTAGCGCCATTTTCGAGGAATCAAGTTTTAACCCTTCTTAGTTGTGGACCTAGCAGCCAGTGTTGACTGTTCTGGTCCACGCTGCCAAGATCAATTTAAACAGAGAAAGGGTACGGGCGT
Selectionner une taille de read entre 20 et 100 :
```

Ensuite on demande à l'utilisateur la taille des reads, compris entre 20 et 100 nucléotides. Une fois la taille choisie, le code retourne la liste de reads avec la qualité pour chaque nucléotide associé de manière aléatoire :

```
AGGCATGAGTGTACTCCTCTACGCACGGGGCCGAATGCCAATCCTGGACA [0.48, 0.1, 0.76, 0.28, 0.36, 0.71, 0.63, 0.53, 0.6, 0.8, 0.95, 0.19, 0.79, 0.73, 0.95, 1.0,
 0.36, 0.96, 0.28, 0.5, 0.48, 0.28, 0.58, 0.14, 0.2, 0.86, 0.18, 0.21, 0.62, 0.6, 0.33, 0.14, 0.24, 0.9, 0.38, 0.26, 0.5, 0.33, 0.13, 0.67, 0.6
2, 0.83, 0.5, 0.97, 0.18, 0.55, 0.78, 0.14, 0.69, 0.96]
TGGTTCCTTAGGTACACTGTCATCTTGTCAGATTGAACTGATTCTAGCGA [0.78, 0.79, 0.71, 0.44, 0.56, 0.19, 0.32, 0.31, 0.69, 0.33, 0.92, 0.41, 0.46, 0.52, 0.58, 0
.52, 0.78, 0.22, 0.59, 0.8, 0.54, 0.4, 0.61, 0.52, 0.32, 0.94, 0.29, 0.79, 0.57, 0.91, 0.49, 0.78, 0.24, 0.31, 0.37, 0.34, 0.72, 0.58, 0.55, 0.
49, 0.75, 0.55, 0.53, 0.62, 0.85, 0.23, 0.56, 0.83, 0.99, 0.9]
TCGGGGCGACTCTACGAGCTCGCAGGGGGCGGGGAGCGCCGGTTCAAAAC [0.63, 0.66, 0.67, 0.87, 0.74, 0.11, 0.92, 0.79, 0.53, 0.11, 0.97, 0.73, 0.52, 0.95, 0.78, 0
.45, 0.51, 0.36, 0.75, 0.94, 0.12, 0.46, 0.37, 0.95, 0.64, 0.16, 0.86, 0.5, 0.63, 0.84, 0.17, 0.52, 0.61, 0.44, 0.64, 0.82, 0.11, 0.13, 0.83, 0
.67, 0.5, 0.38, 0.59, 0.68, 0.28, 0.18, 0.64, 0.27, 0.15, 0.7]
TGTGGCTATTGATCATGCTACTGTGCCTACCAAGCCAGACTCCACTGTTT [0.94, 0.55, 0.96, 0.83, 0.44, 0.28, 0.47, 0.67, 0.38, 0.99, 0.22, 0.19, 0.12, 0.91, 0.69, 0
.15, 0.76, 0.28, 0.54, 0.53, 0.1, 0.77, 0.43, 0.71, 0.76, 0.35, 0.62, 0.44, 0.13, 0.64, 0.2, 0.82, 0.11, 0.15, 0.29, 0.75, 0.17, 0.54, 0.79, 0.
72, 0.96, 0.2, 0.57, 0.32, 0.97, 0.5, 0.53, 0.28, 0.15, 0.18]
AAGGACGGGCATGATCATACGAAGACCCGATAATGGTCAACGTCAGAGTC [0.52, 0.54, 0.23, 0.58, 0.4, 0.23, 0.1, 0.15, 0.14, 0.8, 0.62, 0.47, 0.85, 0.13, 0.22, 0.69
, 0.24, 0.91, 0.32, 0.77, 0.4, 0.68, 0.2, 0.24, 0.79, 0.18, 0.71, 0.66, 0.19, 0.2, 0.22, 0.4, 0.89, 0.22, 0.21, 0.18, 0.13, 0.68, 0.53, 0.28, 0
.4, 0.54, 0.63, 0.11, 0.37, 0.96, 0.84, 0.41, 0.35, 0.57]
TCCTGCCCCAGCGATGGAATACGCCAGCTAGTCAAACGTGCTCATAAGTA [0.68, 0.49, 0.62, 0.43, 0.82, 0.88, 0.31, 0.26, 0.71, 0.69, 0.44, 0.73, 0.95, 0.46, 0.15, 0
.3, 0.75, 0.41, 0.43, 0.69, 0.21, 0.97, 0.65, 0.59, 0.86, 0.91, 0.18, 0.16, 0.79, 0.93, 0.65, 0.41, 0.95, 0.32, 0.73, 0.21, 0.92, 0.32, 0.53, 0
.69, 0.56, 0.73, 0.19, 0.16, 0.9, 0.43, 0.41, 0.1, 0.17, 0.49]
AAGCCTGTTTCCGCCCCCGTCGCTAAAGATATGACGGAGGCGGGAAATTCG [0.15, 0.3, 0.97, 0.13, 0.98, 0.33, 1.0, 0.8, 0.68, 0.33, 0.26, 0.95, 0.56, 0.19, 0.31, 0.98
, 0.5, 0.7, 0.54, 0.29, 0.57, 0.39, 0.3, 0.71, 0.2, 0.23, 0.18, 0.33, 0.14, 0.93, 0.27, 0.66, 0.71, 0.56, 0.31, 0.41, 0.17, 0.23, 0.88, 0.33, 0
.72, 0.22, 0.62, 0.7, 0.74, 0.44, 0.11, 0.61, 0.6, 0.23]
TGGGTACGGAATTATCGCTCGGTTGGCGGTTGGAAACACCAAAGAAAGTC [0.3, 0.58, 0.49, 0.73, 0.11, 0.46, 0.29, 0.7, 0.94, 0.41, 0.92, 0.11, 0.9, 0.98, 0.31, 0.53
, 0.8, 0.18, 0.68, 0.56, 0.27, 0.13, 0.92, 0.55, 0.34, 0.68, 0.5, 0.26, 0.52, 0.5, 0.16, 0.13, 0.14, 0.21, 0.16, 0.22, 0.92, 0.23, 0.34, 0.57,
0.6, 0.16, 0.57, 0.69, 0.35, 0.77, 0.3, 0.7, 0.26, 0.98]
CCTCCTCAGGCTTTCAGGGGTAGCTTTCGTAAAGCCGTATAAGGCAGAAA [0.75, 0.67, 0.44, 0.46, 0.13, 0.76, 0.79, 0.81, 0.68, 0.14, 0.8, 0.69, 0.42, 0.89, 0.76, 0.
6, 0.41, 0.49, 0.34, 0.72, 0.1, 0.71, 0.64, 0.33, 0.69, 0.55, 0.96, 0.64, 0.13, 0.56, 0.1, 0.76, 0.74, 0.73, 0.58, 0.58, 0.12, 0.3, 0.42, 0.28,
 0.34, 0.99, 0.56, 0.47, 0.35, 0.14, 0.56, 0.2, 0.81, 0.14]
CCCATCTGGCCAACGATGCGCGACTGCCACCTCGTAGTTCATATTATCAG [0.92, 0.96, 0.32, 0.83, 0.18, 0.54, 0.4, 0.15, 0.8, 0.41, 0.29, 0.36, 0.94, 0.58, 0.64, 0.3
5, 0.39, 0.2, 0.61, 0.68, 1.0, 0.43, 0.49, 0.43, 0.62, 0.91, 0.59, 0.5, 0.37, 0.13, 0.34, 0.9, 0.28, 0.11, 0.5, 0.83, 0.98, 0.44, 0.37, 0.43, 0
.25, 0.47, 0.72, 0.41, 0.56, 0.79, 1.0, 0.13, 0.22, 0.63]
AGCGGCGTCGGTAACTGGTGCAATGGATACATGATACCGAGTAGTTAGGC [0.47, 0.28, 0.36, 0.97, 0.89, 0.48, 0.56, 0.7, 0.18, 0.75, 0.96, 0.92, 0.81, 0.72, 0.67, 0.
28, 0.66, 0.16, 0.47, 0.56, 0.29, 0.67, 0.41, 0.43, 0.95, 0.51, 0.14, 0.55, 0.26, 0.52, 0.35, 0.33, 0.58, 0.65, 0.74, 0.47, 0.3, 0.26, 0.44, 0.
21, 0.94, 0.9, 0.86, 0.12, 0.14, 0.93, 0.72, 0.48, 0.5, 0.81]
CGAGTAGGCGCGGTCGCAAATAATCGCGAACTCATACACCGCTACTTCGT [0.52, 0.75, 0.91, 0.33, 0.73, 0.34, 0.72, 0.57, 0.23, 0.99, 0.77, 0.2, 0.9, 0.9, 0.51, 0.28
, 0.76, 0.91, 0.58, 0.27, 0.19, 0.21, 0.72, 0.83, 0.25, 0.27, 0.14, 0.75, 0.33, 0.88, 0.9, 0.18, 0.47, 0.63, 0.31, 0.19, 0.48, 0.42, 0.86, 0.18
, 0.69, 0.53, 0.64, 0.37, 0.26, 0.17, 0.18, 0.67, 0.81, 0.42]
CCGGACGGGCGCTTAATGGTCCAGGAGCAGTTAGAGTCGGAACCTTTTCA [0.82, 0.52, 0.18, 0.79, 0.2, 0.36, 0.88, 0.38, 0.62, 0.53, 0.2, 0.46, 0.86, 0.7, 0.75, 0.87
, 0.34, 0.32, 0.21, 0.47, 0.27, 0.36, 0.13, 0.32, 0.77, 0.45, 0.27, 0.23, 0.75, 0.17, 0.54, 0.81, 0.13, 0.51, 0.84, 0.37, 0.22, 0.15, 0.42, 0.9
3, 0.45, 0.53, 0.91, 0.24, 0.56, 0.6, 0.28, 0.81, 0.48, 0.96]

Combien d'adaptateur ?
```

L'utilisateur va ensuite pouvoir fournir le nombre d'adaptateurs souhaités, ainsi que les motifs correspondants :

```
Combien d'adaptateur ?
2
Taper vos adapteurs :
ATGC
GCGG

Le nombre total d'adaptateur est : 9
```

Le code retourne à l'utilisateur le readset complet en supprimant les motifs ainsi que la qualité de chaque nucléotide associé. Maintenant il peut choisir un seuil de qualité compris entre 0 et 1 pour supprimer les qualités trop basses à chaque extrémité :

```
Entrez le seuil minimal de qualité compris entre 0 et 1 :
0.8
```

Il peut définir un seuil moyen, de qualité comprise entre 0 et 1 également, pour filtrer les reads de faibles qualités sur toute la longueur. A chaque question, le programme retourne à l'utilisateur le readset avec les modifications, ainsi que les qualités associées.

```
Entrez le seuil moyen de qualité compris entre 0 et 1 :
0.6
CTCACATGCAGTGCGCTTCGGGCACCCTGTTTAAGTTCT  [0.89, 0.91, 0.45, 0.32, 0.97, 0.66, 0.26, 0.67, 0.77, 0.23, 0.96, 0.43, 0.67, 0.5, 0.26, 0.34, 0.14, 0.39, 0.86, 0.3
2, 0.82, 0.21, 0.92, 0.73, 0.94, 0.93, 0.82, 0.96, 0.23, 0.94, 0.32, 0.94, 0.11, 0.65, 0.76, 0.44, 0.45, 0.44, 0.94]
CACAGCATTAACAGGAGACTGCGACAGAAATAC  [0.96, 0.82, 0.81, 0.43, 0.84, 0.3, 0.73, 0.4, 0.85, 0.53, 0.44, 0.36, 0.79, 0.92, 0.92, 0.75, 0.85, 0.51, 0.45, 0.23, 0.42
, 0.26, 0.17, 0.7, 0.71, 0.12, 0.94, 0.95, 0.88, 0.94, 0.66, 0.34, 0.86]
ATACACCACTCCTAAGCCCCAGTGAGTGCAAATTC  [0.95, 0.96, 0.51, 0.48, 0.34, 0.83, 0.8, 0.23, 0.8, 0.7, 0.74, 0.32, 0.23, 0.98, 0.51, 0.48, 0.62, 0.67, 0.15, 0.63, 0.3
5, 0.88, 0.69, 0.87, 0.81, 0.86, 0.11, 0.89, 0.17, 0.61, 0.63, 0.18, 0.66, 0.45, 0.91]
GTCAAATTTTCAGTCAAATCCAGGATATTGTTTTACAAGATG  [0.98, 0.94, 0.92, 0.69, 0.54, 0.97, 0.71, 0.35, 0.51, 0.87, 0.99, 0.4, 0.99, 0.19, 0.75, 0.29, 0.44, 0.25, 0.86,
0.46, 0.87, 0.32, 0.83, 0.99, 0.74, 0.82, 0.28, 0.26, 0.46, 0.71, 0.58, 0.22, 0.32, 0.6, 0.25, 0.61, 0.89, 0.98, 0.35, 0.59, 0.94, 0.83]

Entrez le read a comparer :
```

Enfin on demande à l'utilisateur de rentrer un motif qui va être comparé avec l'ensemble du readset pour renvoyer le read avec lequel il produit le chevauchement maximal. Le code se termine en retournant la séquence génomique après filtrations ainsi que sa taille.

```
Entrez le read a comparer :
ATGCGG
Le(s) read(s) qui produit/produisent le meilleur chevauchement avec 4 ressemblance(s) :

ATACACCACTCCTAAGCCCCAGTGAGTGCAAATTC

La séquence génomique généré avec le readset filtré est :
CTCACATGCAGTGCGCTTCGGGCACCCTGTTTAAGTTCTCACAGCATTAACAGGAGACTGCGACAGAAATACATACACCACTCCTAAGCCCCAGTGAGTGCAAATTCGTCAAATTTTCAGTCAAATCCAGGATATTGTTTTACAAGATG
Elle a une taille de 149
```