

Rappeler les initialisations pour ce cours :

```
MAT16 = "http://iml.univ-mrs.fr/~kohel/tch/MAT16/"
exec(open(get_remote_file(MAT16 + "MAT16.sage")).read())
S = AlphabeticStrings()
M = S.encoding(open(get_remote_file(MAT16 + "chapitre.1.txt")).read())
```

1. Comparer les fréquences des lettres pour le chiffrement à substitution :

```
Subst = SubstitutionCryptosystem(S)
EK = Subst(S("ISCQUNJVPGZEORLFMDAKHTYWXB"))
```

en mode ECB, CBC, CFB, et OFB.

```
ECB : ct_ECB = EK(M,mode="ECB")
      ct_ECB.frequency_distribution().plot()
CBC : ct_CBC = EK(M,mode="CBC",IV=S("XYZ"))
      ct_CBC.frequency_distribution().plot()
CFB : ct_CFB = EK(M,mode="CFB",IV=S("XYZ"),mode_block_length=1)
      ct_CFB.frequency_distribution().plot()
OFB : ct_OFB = EK(M,mode="OFB",IV=S("XYZ"),mode_block_length=1)
      ct_OFB.frequency_distribution().plot()
```

2. Comparer les fréquences des lettres pour le chiffrement par transposition :

```
Trans = TranspositionCryptosystem(S,6)
EK = Trans([3, 1, 6, 5, 2, 4])
```

en mode ECB, CBC, CFB, et OFB.

```
ECB : ct_ECB = EK(M,mode="ECB")
      ct_ECB.frequency_distribution().plot()
CBC : ct_CBC = EK(M,mode="CBC",IV=S("ABCXYZ"))
      ct_CBC.frequency_distribution().plot()
CFB : ct_CFB = EK(M,mode="CFB",IV=S("ABCXYZ"),mode_block_length=1)
      ct_CFB.frequency_distribution().plot()
OFB : ct_OFB = EK(M,mode="OFB",IV=S("ABCXYZ"),mode_block_length=1)
      ct_OFB.frequency_distribution().plot()
```

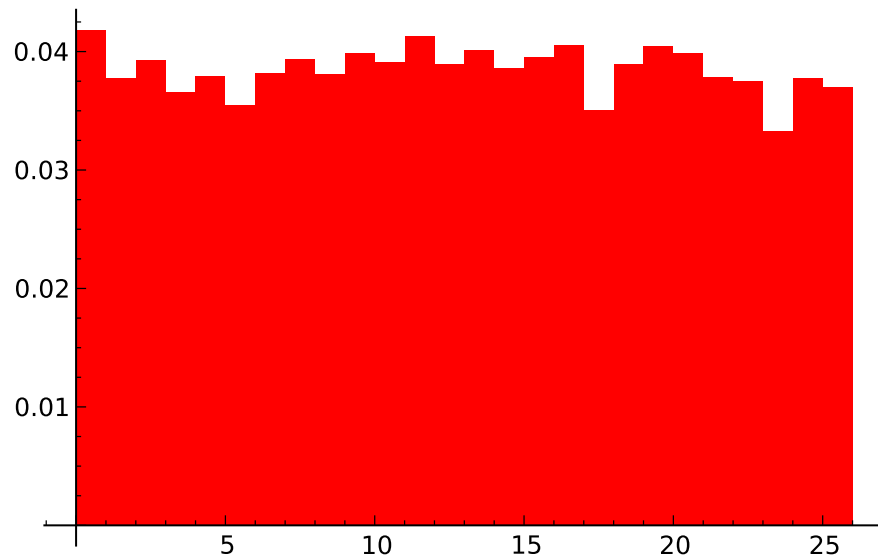
3. Est-ce qu'il y a des différences évidentes entre les modes d'opérations avec les chiffrements par transposition par rapport aux chiffrements à substitution ?

*Solution.* Pour ces exemples, pas vraiment, mais voir ci-dessous.

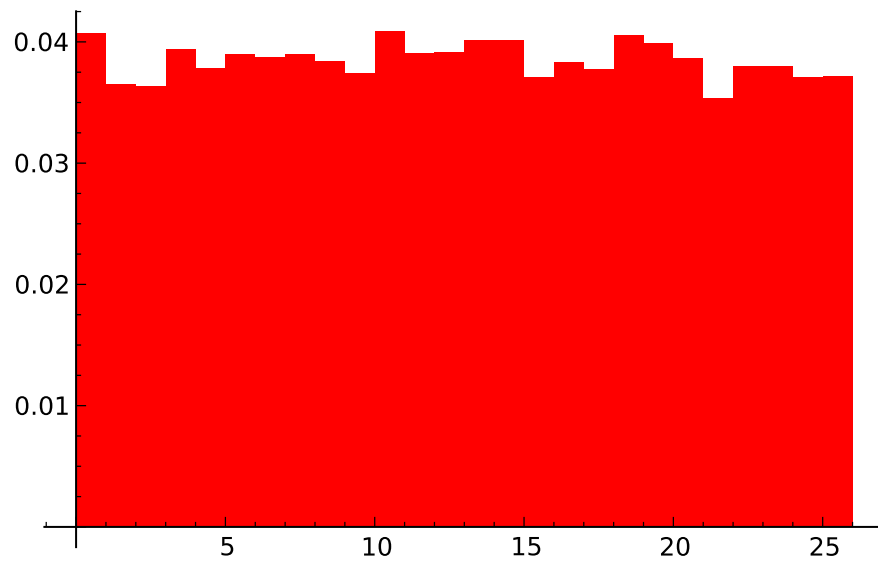
4. Qu'est-ce que se change si on utilise la transposition clé  $K = [1, 3, 4, 5, 6, 2]$  ?

*Solution.* Pour les modes CBC et CFB, il n'y a pas de grand changement.

**CBC :**

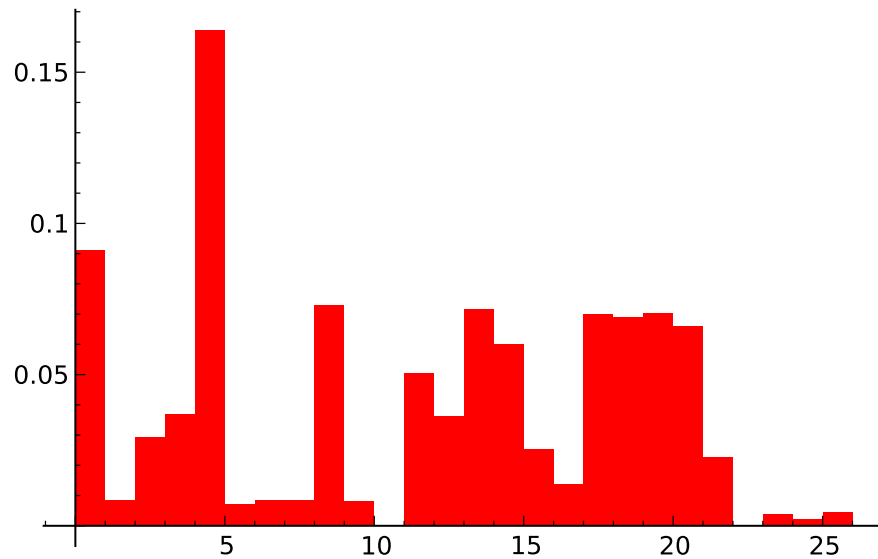


**CFB :**



Or, pour OFB, la distribution des fréquences est celui du texte clair.

**OFB :**



En effet, le texte chiffré est égal au texte clair, pourquoi ?

- Rappeler les étapes pour déchiffrement d'un texte chiffré en mode d'opération ECB, CBC, CFB, ou OFB.

*Solution.* Voir le cours.

Nous allons étudier les propriétés des chiffrements à flot induit par un chiffrement par bloc en mode OFB et CFB. Nous recommençons avec le texte clair :

```
M = S.encoding(
"""
    Twas brillig, and the slithy toves
    Did gyre and gimble in the wabe;
    All mimsy were the borogoves,
    And the mome raths outgrabe.
""")
```

Pour les exercices suivants, nous créons un chiffrement Hill :

```
H = HillCryptosystem(S,10); A = H.random_key(); E = H(A)
```

- Rappeler les définitions des chiffrements à flot synchronisés et asynchronisés (ou autosynchronisants), et identifier les chiffrements en modes d'opérations CFB et OFB (avec  $r = 1$ ) avec ces chiffrements.

*Solution.* Voir le cours pour les définitions. Les modes d'opération CFB et OFB produisent des chiffrements à flot autosynchronisants et synchronisés, respectivement.

- Créer les textes chiffrés  $C_1$  et  $C_2$  en mode d'opération OFB et CFB :

```
I0 = S("IACQZPKDL")
C1 = E(M,mode="CFB",IV=I0,mode_block_length=1)
C2 = E(M,mode="OFB",IV=I0,mode_block_length=1)
```

et vérifier leurs déchiffrements :

```
"CFB:", M == E.deciphering(C1,mode="CFB",mode_block_length=1)
"OFB:", M == E.deciphering(C2,mode="OFB",mode_block_length=1)
```

*Solution.* Les déchiffrements sont bien égaux au texte clair :

CFB : True

OFB : True

8. Changer un caractère de  $C_1$  et  $C_2$  :

```
i, c = (19, S("T"))
X1 = C1[:i] * c * C1[i+1:]
X2 = C2[:i] * c * C2[i+1:]
```

et vérifier les changements :

```
print "C1: %s -> %s" % (C1[i], c)
print "C2: %s -> %s" % (C2[i], c)
```

Quels sont les propriétés évidentes de ces déchiffrements :

```
M1 = E.deciphering(X1,mode="CFB",mode_block_length=1)
M2 = E.deciphering(X2,mode="OFB",mode_block_length=1)
```

des textes chiffrés modifiés ?

*Solution.* Les textes chiffrés sont bien changés :

C1 : Q -> T

C2 : E -> T

En comparant le texte clair :

```
M :      TWASBRILLIGANDTHESLITHYTOVESDIDGYREANDGIMBLEINTHEW
        ABEALLMIMSYWERETHEBOROGOVESANDTHEMOMERATHSOUTGRABE
```

avec les déchiffrements des ces textes chiffrés modifiés :

```
M1 :      TWASBRILLGQHMGHGKZVTHYTOVESDIDGYREANDGIMBLEINTHEW
        ABEALLMIMSYWERETHEBOROGOVESANDTHEMOMERATHSOUTGRABE
```

```
M2 :      TWASBRILLXGANDTHESLITHYTOVESDIDGYREANDGIMBLEINTHEW
        ABEALLMIMSYWERETHEBOROGOVESANDTHEMOMERATHSOUTGRABE
```

nous voyons qu'une suite de 11 caractères de M1 ont changé, tandis que M2 est changé à un seul caractère.

9. Supprimer un caractère de  $C_1$  et  $C_2$  :

```
Y1 = C1[:i] * C1[i+1:]
Y2 = C2[:i] * C2[i+1:]
```

et déterminer les propriétés des déchiffrements de ces textes chiffrés modifiés :

```
M3 = E.deciphering(Y1,mode="CFB",mode_block_length=1)
M4 = E.deciphering(Y2,mode="OFB",mode_block_length=1)
```

*Solution.* Les messages sont :

M3 :      TWASBRILL \_UCVWRYNTMVTHYTOVESDIDGYREANDGIMBLEINTHEW  
            ABEALLMIMSYWERETHEBOROGOVESANDTHEMOMERATHSOUTGRABE

M4 :      TWASBRILL \_DAGOUAKMCWEURBLHAXNSZIPJSRATVPPZVDMZKINJ  
            HXSRKOWKSAVOGBGOLXVYXENGMPHDOZPFNIAHEPXPKPUBOPCDQ

En mode CFB on voit qu'un caractère est supprimé, après lequel 10 caractères sont faux, mais en mode OFB tous les caractères sont faux dès le caractère supprimé.