Iniciació a la Criptografia

Artur Travesa

(versió 2024-07)

Capítol 1. El criptosistema de Cèsar, i afins

1.0. Introducció

En aquest capítol tractarem d'un tipus de criptosistemes molt antics i molt elementals, que es poden encabir en una família de criptosistemes anomenats monoalfabètics de substitució.

La comprensió del significat d'aquests sintagmes és gairebé tautològica: els missatges, escrits en un alfabet, es xifren caràcter a caràcter per substitució d'un caràcter per un altre del mateix alfabet, sempre el mateix, i de manera bijectiva.

1.1. Alfabets

1.1.0. Definició

A fi de disposar d'eines matemàtiques precises, anomenarem *alfabet* qualsevol conjunt finit i no buit; els seus elements seran els *símbols* o les *lletres* de l'alfaget. I els *missatges* (relatius a aquest alfabet) són les successions finites de lletres de l'alfabet.

1.1.1. Exemples

Com a exemples, podem pensar en l'alfabet (grec), o l'abecedari (llatí), o en altres alfabets. Per exemple, durant molts anys, es consideraven "ch", "ll" i "ñ" com a caràcters (lletres) del 'abecedario' castellà; això feia un alfabet de 29 lletres.

Però podem pensar en altres llengües o alfabets: ciríl·lic, europeus orientals, escandinaus, hebreu, àrabs, xinesos, coreà, japonès, tais, bràmics, altres alfabets orientals, i molts altres africans, amerindis, malaisians, etcètera, dels quals ni en conec l'existència.

I, si es vol, encara caldria pensar en alfabets que continguin tots els caràcters accentuats o modificats d'alguna manera. Sense anar més lluny, com es consideren ny o ç en català? Com un caràcter o com un dígraf, l'un? I com un caràcter senzill o com un caràcter modificat (equivalent a accentuat), l'altre? El diccionari de l'IEC considera ny com un dígraf i ç com una c amb un accent (almenys, en algunes paraules que jo he consultat).

D'altra banda, els alfabets s'aprenen sovint en un cert ordre; és a dir, se suposa que els caràcters de l'alfabet en qüestió estan ordenats d'una certa manera estàndard; per exemple, l'ordre alfabètic, si es tracta de l'alfabet (grec) o l'abecedari (llatí), o altres ordres depenent de l'alfabet que es faci servir. Per exemple, quan es consideren ch, ll i ñ com a caràcters del 'abecedario', la ch es col·loca entre la c i la d, la ll, entre la l i la m, i la ñ, entre la n i la o; això determina un ordre en l'alfabet castellà de 29 lletres.

Però no tothom considera de la mateixa manera l'ordre dels caràcters. Segons quina aplicació o quin programari d'ordinador es fa servir (o fins i tot quina versió del mateix programari), ens podem trobar que la ch es pensa com dues lletres (i, per tant, se situa entre cg i ci), o bé com una de sola (i se situa entre c i d). I anàlogament amb les ll i ñ, o les lletres accentuades (i en quin ordre van els accents?).

1.1.2. El nostre alfabet

Caldrà tenir això present. Per tant, suposarem que treballem amb un alfabet fixat i amb un ordre determinat en aquest alfabet.

En aquest tutorial, utilitzarem com a alfabet el conjunt format pels $1114112 = 16^5 + 16^4$ caràcters (possibles) que fa servir *SageMath*, i l'ordre, el que es deriva de manera natural de l'ordre de la seva codificació $0 < 1 < 2 < 3 < \cdots < 1114111$.

1.2. El criptosistema de Cèsar

El criptosistema de Cèsar consisteix a fer córrer cíclicament els caràcters d'un missatge una quantitat fixa de caràcters; aquesta quantitat és la clau secreta. Per exemple, es pot usar la funció següent, que s'aplica a un missatge pla (sense codificar) i proporciona un missatge xifrat (també sense codificar). (Això de xifrar missatges sense codificar és una mala praxi, perquè podem intentar usar codis que corresponen a caràcters no imprimibles... però ja hi tornarem, a aquest punt.)

Òbviament, doncs, la clau secreta és un nombre enter entre 0 i 1114112; de fet, i per la propietat de ciclicitat que hem anomenat més amunt, la clau és un nombre enter qualsevol, del qual només compta la seva classe residual mòdul 1114112.

1.2.0. La funció

In [1]:
```
def Cesar(mis,clau):
    m=1114112
    xif=""
    for i in range(len(mis)):
        xif=xif+chr((ord(mis[i])+clau) %m)
    return xif
```

1.2.1. Exemple

Per exemple, podem xifrar amb la clau que sovint s'atribueix a l'emperador romà Juli Cèsar

(clau=3) el missatge "VENIVIDIVINCI", que també se li atribueix a ell. D'aquí el nom del criptosistema.

In [2]:
```
1 mispla="VENIVIDIVINCI"
```

In [3]:
```
1 xifrat=Cesar(mispla,3)
```

In [4]:
```
1 xifrat
```

Out[4]: 'YHQLYLGLYLQFL'

Efectivament, el mètode xifra. Es tracta de veure que també desxifra, si es fa servir la clau oposada. Notem que, òbviament, és el mateix usar -3 que usar $1114112-3$.

In [5]:
```
1 Cesar(xifrat,-3)
```

Out[5]: 'VENIVIDIVINCI'

In [6]:
```
1 Cesar(xifrat,1114109)
```

Out[6]: 'VENIVIDIVINCI'

In [7]:
```
1 Cesar(xifrat,-3)==mispla
```

Out[7]: True

1.2.2. Altres exemples

Qui no ha jugat a intentar desxifrar missatges xifrats amb el criptosistema de Cèsar i l'alfabet llatí de 26 lletres? Sembla extraordinàriament "senzill"; de fet, només hi ha 26 claus possibles, de manera que s'acaba aviat. O no?

Algú ha intentat esbrinar perquè l'ordinador de la novel·la 2001: A space Oddysey, d'Arthur C. Clarke, es diu HAL i és de la sèrie 9000? O a qui fa correspondre les inicials JCN?

Hom pot pensar que serà molt difícil desxifrar un missatge xifrat amb el criptosistema de Cèsar i una clau desconeguda, ja que aquesta s'ha pogut triar entre les 1114112 possibles. Caldrà provar les 1114112 claus?

1.2.3. Exercici

Es demana intentar desxifrar el text següent, que ha estat xifrat amb la funció Cesar anterior i una clau menor que 1114112.

In [8]:
```
1 x='寶尅尉尅尒寀尃小對寀導小寀屻尔寀尔封導寀射尉将岍尃尉尅察'
```

1.2.3.0. Una solució

Comencem per conèixer els caràcters numèrics (que, de fet, és allò que compta)

asssociats al missatge xifrat.

In [9]:
```
lta=[ord(i) for i in x]
```

In [10]:
```
print(lta)
```

```
[23542, 23557, 23561, 23557, 23573, 23488, 23555, 23567, 23565, 234
88, 23566, 23567, 23488, 23689, 23571, 23488, 23572, 23553, 23566,
23488, 23556, 23561, 23558, 23693, 23555, 23561, 23564, 23519]
```

Ara, mirem com són els caràcters numèrics associats a un text de prova.

In [11]:
```
print([ord(i) for i in "A veure què succeeix."])
```

```
[65, 32, 118, 101, 117, 114, 101, 32, 113, 117, 232, 32, 115, 117,
99, 99, 101, 101, 105, 120, 46]
```

Notem que el menor caràcter és el 32, i correspon a l'espai en blanc.

Si el missatge xifrat és un text (com diu l'enunciat), deu tenir espais en blanc i, per tant, el menor dels codis hauria de correspondre a l'espai blanc. Això hauria de fer senzill proporcionar-nos la clau.

In [12]:
```
min(lta)-32
```

Out[12]:
```
23456
```

La clau podria ser 23456. Provem-ho.

In [13]:
```
Cesar(x,-23456)
```

Out[13]:
```
'Veieu com no és tan difícil?'
```

Observació. Cal notar que els missatges poden ser molt més complicats que simples textos; però els codis associats als caràcters que els componguin difícilment pertanyeran a intervals molt grans, de manera que es poden provar claus en intervals petits (i per a parts petites dels missatges). Més avall tornarem sobre això.

1.2.4. Versions millorades de la funció Cesar(mis,clau)

Més amunt hem comentat que no és una bona praxi treballar amb els missatges sense codificar. A fi de tractar aquesta qüestió, incorporem les funcions Codis( ) i Caracters( ) del capítol 0.

In [14]:
```
def Codis(text):
    return [ord(i) for i in text]
```

In [15]:
```
def Caracters(codis):
    m=1114112
    cars=""
    for i in range(len(codis)):
        cars=cars+chr(codis[i]%m)
    return cars
```

1.2.4.0. Millorem l'entrada i la sortida de la funció

La funció Cesar(mis,clau) s'aplica a una tira de caràcters mis i un nombre enter clau i proporciona una altra tira de caràcters. Es tracta de refer la funció en una funció CesarX(mis,clau) en què mis sigui indistintament una tira de caràcters o bé a una llista de codis i clau sigui un nombre enter, i proporcioni el missatge xifrat (o desxifrat, segons la clau que s'utilitzi) en el mateix format que el missatge d'entrada.

I encara, si tenim en compte que sovint les claus poden xifrar un text senzill en una successió il·legible de caràcters del codi, molt probablement sigui més útil una funció, CesarC(mis,clau), que proporcioni el missatge xifrat/desxifrat en forma codificada (independentment del tipus d'entrada); i una altra funció, CesarD(mis,clau), que el proporcioni en forma de tira de caràcters (també independentment del tipus d'entrada), per als missatges descodificats.

Notem que en qualsevol cas, sempre podrem aplicar les funcions Codis() o Caracters() per a obtenir-ne (o veure'n) l'altra versió.

1.2.4.1. Les funcions

In [16]:
```
def CesarX(mis,clau):
    m=1114112
    if type(mis)==str:
        b=1
        lta=Codis(mis)
    else:
        lta=mis
        b=0
    xif=[(lta[i]+clau)%m for i in range(len(lta))]
    if b==1:
        xif=Caracters(xif)
    return xif

```

In [17]:
```
def CesarC(mis,clau):
    m=1114112
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    xif=[(lta[i]+clau)%m for i in range(len(lta))]
    return xif

```

In [18]:
```
def CesarD(mis,clau):
    m=1114112
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    xif=Caracters([(lta[i]+clau)%m for i in range(len(lta))])
    return xif
```

1.2.4.2. Exemples

In [19]:
```
mispla
```

Out[19]: 'VENIVIDIVINCI'

In [20]:
```
CesarX(mispla,3)
```

Out[20]: 'YHQLYLGLYLQFL'

In [21]:
```
CesarC(mispla,3)
```

Out[21]: [89, 72, 81, 76, 89, 76, 71, 76, 89, 76, 81, 70, 76]

In [22]:
```
CesarD(mispla,3)
```

Out[22]: 'YHQLYLGLYLQFL'

In [23]:
```
cds=Codis(mispla)
```

In [24]:
```
cds
```

Out[24]: [86, 69, 78, 73, 86, 73, 68, 73, 86, 73, 78, 67, 73]

In [25]:
```
CesarX(cds,3)
```

Out[25]: [89, 72, 81, 76, 89, 76, 71, 76, 89, 76, 81, 70, 76]

In [26]:
```
CesarC(cds,3)
```

Out[26]: [89, 72, 81, 76, 89, 76, 71, 76, 89, 76, 81, 70, 76]

In [27]:
```
CesarD(cds,3)
```

Out[27]: 'YHQLYLGLYLQFL'

1.3. Observacions sobre la seguretat

A fi de poder desxifrar un missatge xifrat amb un criptosistema de Cèsar sense conèixer la clau, cal fer-ne una estimació. I això pot ser (potser) impossible.

Imaginem-nos que els missatges que es volen transmetre xifrats siguin nombres de, per exemple, 8 xifres decimals (nombres de DNI?).

Podem utilitzar un alfabet de 10 símbols, 0, 1, 2, 3, 4, 5, 6, 7, 8, i 9. Però quina de les 10

claus possibles s'ha fet servir per a xifrar un missatge determinat? En principi, totes 10 són igualment probables, i qualsevol produeix un missatge xifrat (o desxifrat) vàlid. Per tant, *sembla* que el criptosistema és prou segur, perquè no sabem desxifrar el missatge xifrat, llevat que coneguem la clau.

Ara bé, tot i que només una clau desxifra correctament el missatge, entre els 10^8 missatges que es poden enviar, només n'hi ha 10 que puguin correspondre al missatge xifrat amb el criptosistema de Cèsar.

És a dir, no tots els missatges plans possibles tenen la mateixa probabilitat de ser el missatge original: n'hi ha 99999990 amb probabilitat zero i només 10 amb probabilitat no nul·la, $\frac{1}{10} = 0,1$.

Ara, canviem una mica el paradigma. En comptes de pensar en missatges de 8 lletres d'un alfabet de 10 símbols, pensem en missatges d'una sola lletra d'un alfabet de 10^8 símbols. Ara, hi ha 10^8 claus possibles i tots els missatges possibles són equiprobables. Només podrem desxifrar el missatge si encertem la clau?

Tot i que això sembla molt més segur, encara hi ha un altre fet que incideix en la seguretat: són tots els missatges igualment probables?

La freqüència amb què apareix cada caràcter en el text pla pot ser determinant per a desxifrar un missatge xifrat amb el criptosistema de Cèsar. Per exemple, ja hem utilitzat més amunt que el caràcter blanc és el més utilitzat en textos escrits en català, i també en moltes altres llengües de paraules no especialment llargues o que tenen moltes paraules curtes (per exemple, articles, preposicions, conjuncions), i que aquestes paraules curtes apareixen molt sovint en un escrit llarg.

Això fa que en escrits llargs (o sigui, de molts caràcters de l'alfabet), la probabilitat dels caràcters no sigui la mateixa i una *anàlisi de freqüències* pot permetre obtenir la clau i desxifrar el missatge.

Deixo oberta una reflexió més profunda sobre aquest fet. Només comentaré que, a fi d'evitar els problemes derivats de la diferent probabilitat dels missatges plans/sifrats, convé que tots els missatges que es xifren siguin (pseudo)aleatoris o, com a mínim, tant com sigui possible. Tornarem a aquest punt en un capítol posterior (farciment de missatges).

1.3.0. Exercici

Intenteu esbrinar la clau del missatge xifrat "YHQLYLGLYLQFL" a partir d'una anàlisi de freqüències.

1.3.0.1. Una solució

El caràcter més freqüent és "L"; potser correspon a una vocal (i tot són majúscules...). Provem les cinc possibilitats.

```
In [28]:  1 prova="YHQLYLGLYLQFL"
```

```
In [29]:  1 CesarX(prova,ord("A")-ord("L"))
```

```
Out[29]: 'N=FANA<ANAF;A'
```

```
In [30]:  1 CesarX(prova,ord("E")-ord("L"))
```

```
Out[30]: 'RAJERE@EREJ?E'
```

```
In [31]:  1 CesarX(prova,ord("O")-ord("L"))
```

```
Out[31]: '\\KTO\\OJO\\OTIO'
```

```
In [32]:  1 CesarX(prova,ord("U")-ord("L"))
```

```
Out[32]: 'bQZUbUPUbUZOU'
```

```
In [33]:  1 CesarX(prova,ord("I")-ord("L"))
```

```
Out[33]: 'VENIVIDIVINCI'
```

Sembla clar.

1.3.1. Exercici proposat

Intenteu desxifrar el missatge xifrat 'JZWJPF'.

1.4. Criptosistemes afins

Els criptosistemes afins (en dimensió 1) són una evolució natural del criptosistema de Cèsar. Es tracta de donar-ne una descripció senzilla.

Recordem que el criptosistema de Cèsar és un criptosistema monoalfabètic de substitució. Això vol dir que els caràcters de l'alfabet es xifren un a un (monoalfabètic) fent-ne una substitució del caràcter del missatge pla per un caràcter del missatge xifrat, de manera que aquest procés de substitució sigui una aplicació bijectiva.

Formalment, es considera una bijecció entre el conjunt dels N caràcters de l'alfabet i $\mathbb{Z}/N\mathbb{Z}$. Recordem que en el nostre cas fem servir $N = 1114112$. Fins aquí, només es tracta de la codificació dels missatges: successions d'elements de $\mathbb{Z}/N\mathbb{Z}$.

Una clau qualsevol del criptosistema de Cèsar és un element $c \in \mathbb{Z}/N\mathbb{Z}$. El procés de xifratge consisteix a aplicar la translació (bijecció) $x(m) :\equiv m + c \pmod{N}$; i el procés de desxifratge és l'aplicació de la translació inversa, $x'(m') :\equiv m' + (N - c) \pmod{N}$.

1.4.0. Descripció dels criptosistemes afins

No cal limitar-nos a aplicacions tan senzilles com les translacions; podem utilitzar

transformacions lineals $x(m) :\equiv cm \pmod{N}$, o bé afins $x(m) :\equiv cx + d \pmod{N}$, on ara les claus són c, per a les lineals, o la parella (c, d), per a les afins.

Però cal que les aplicacions de xifratge i les de desxifratge siguin bijectives (de fet, cal que l'aplicació de xifratge seguida de l'aplicació de desxigratge sigui la identitat; per tant, l'aplicació de desxifratge ha de ser exhaustiva i l'aplicació de xifratge ha de ser injectiva; i en un conjunt finit, això significa que les dues han de ser bijectives i l'una inversa de l'altra).

Per tant, cal que c sigui invertible mòdul N. En aquest cas, si c' és l'invers de c mòdul N, les transformacions inverses (o sigui, les de desxifratge), són les $x'(m') :\equiv c'm' \pmod{N}$, i $x'(m') :\equiv c'(m' - d) \equiv c'm' - c'd \pmod{N}$, respectivament. Notem que, en particular, la transformació inversa d'una transformació lineal també és lineal; i la transformació inversa d'una transformació afí també és afí.

No totes les claus proporcionaran missatges xifrats diferents del missatge pla. Per exemple, per a $c = 1$, l'aplicació lineal de xifratge és la identitat, de manera que els missatges xifrats són exactament els missatgs plans. I això mateix succeeix per a la clau afí $(1, 0)$.

D'altra banda, el xifratge de Cèsar amb clau d és exactament el missatge afí amb clau $(1, d)$. Doncs, el criptosistemes de Cèsar són casos particulars de criptosistemes afins.

1.4.1. Les claus

Així com per al criptosistema de Cèsar, hi ha exactament N claus possibles (tantes com caràcters té l'alfabet), per al criptosistema lineal amb $N = 1114112$ el nombre de claus possibles és exactament $\varphi(N) = 2^{15} \cdot 16 = 2^{19} = 524288$. I per al criptosistema afí amb $N = 1114112$, és $n \cdot \varphi(N) = 2^{35} \cdot 17 = 584115552256$.

1.4.2. Les funcions de xifratge / desxifratge

Es tracta de definir una funció per a xifrar amb el criptosistema afí de 1114112 caràcters. La funció ha d'admetre com a paràmetres el missatge, mis, i la clau de xifratge, clau. Se'n poden fer versions AfiX(mis,clau), AfiXC(mis,clau), AfiXD(mis,clau) de manera que la sortida sigui del mateix tipus que l'entrada, o bé sempre codificada, o bé sempre descodificada.

In [34]:
```
def AfiX(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
        b=1
    else:
        lta=mis
        b=0
    m=1114112
    c1=clau[0]
    c2=clau[1]
    xif=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    if b==1:
        xif=Caracters(xif)
    return xif

```

In [35]:
```
def AfiXC(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=clau[0]
    c2=clau[1]
    xif=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    return xif

```

In [36]:
```
def AfiXD(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=clau[0]
    c2=clau[1]
    xif=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    xif=Caracters(xif)
    return xif

```

Ara cal definir una funció AfiDX(mis,clau) per a desxifrar amb el criptosistema afí de 1114112 caràcters. La funció ha d'admetre com a paràmetres el missatge xifrat, mis, i la clau de xifratge, clau. Convé fer, també, les altres dues versions per a la sortida sempre codificada o sempre descodificada

In [37]:
```
def AfiDX(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
        b=1
    else:
        lta=mis
        b=0
    m=1114112
    c1=((clau[0]%m)^(-1))%m
    c2=-c1*clau[1]%m
    pla=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    if b==1:
        pla =Caracters(pla)
    return pla

```

In [38]:
```
def AfiDXC(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=((clau[0]%m)^(-1))%m
    c2=-c1*clau[1]%m
    pla=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    return pla

```

In [39]:
```
def AfiDXD(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=((clau[0]%m)^(-1))%m
    c2=-c1*clau[1]%m
    pla=Caracters([(c1*lta[i]+c2)%m for i in range(len(lta))])
    return pla

```

1.4.3. Proves

Comencem per definir un missage de prova.

In [40]:
```
missatge="Un missatge de prova per al criptosistema afí."
```

1.4.3.0.

La clau [1,0] xifra/desxifra com la identitat.

In [41]:
```
xifrat=AfiX(missatge,[1,0])
```

In [42]:
```
print(xifrat)
```

```
Un missatge de prova per al criptosistema afí.
```

1.4.3.1.

La clau [1,c] xifra/desxifra com el criptosistema de Cèsar.

In [43]:
```
xifrat=AfiX('VENIVIDIVINCI',[1,3])
```

In [44]:
```
xifrat
```

Out[44]:
```
'YHQLYLGLYLQFL'
```

In [45]:
```
AfiDX(xifrat,[1,3])
```

Out[45]:
```
'VENIVIDIVINCI'
```

1.4.3.2.

Xifrem i desxifrem el **missatge** amb la clau **[37,59]**, d'una banda, com a tira de caràcters, i de l'altra, com a llista de codis.

Primerament, com a llista de codis:

In [46]:
```
mis1=Codis(missatge)
print(mis1)
```

```
[85, 110, 32, 109, 105, 115, 115, 97, 116, 103, 101, 32, 100, 101,
32, 112, 114, 111, 118, 97, 32, 112, 101, 114, 32, 97, 108, 32, 99,
114, 105, 112, 116, 111, 115, 105, 115, 116, 101, 109, 97, 32, 97,
102, 237, 46]
```

In [47]:
```
xifrat1=AfiX(mis1,[37,59])
print(xifrat1)
```

```
[3204, 4129, 1243, 4092, 3944, 4314, 4314, 3648, 4351, 3870, 3796,
1243, 3759, 3796, 1243, 4203, 4277, 4166, 4425, 3648, 1243, 4203, 3
796, 4277, 1243, 3648, 4055, 1243, 3722, 4277, 3944, 4203, 4351, 41
66, 4314, 3944, 4314, 4351, 3796, 4092, 3648, 1243, 3648, 3833, 882
8, 1761]
```

In [48]:
```
pla1=AfiDX(xifrat1,[37,59])
print(pla1)
pla1==mis1
```

```
[85, 110, 32, 109, 105, 115, 115, 97, 116, 103, 101, 32, 100, 101,
32, 112, 114, 111, 118, 97, 32, 112, 101, 114, 32, 97, 108, 32, 99,
114, 105, 112, 116, 111, 115, 105, 115, 116, 101, 109, 97, 32, 97,
102, 237, 46]
```

Out[48]:
```
True
```

```
In [49]:  Caracters(pla1)
```

```
Out[49]: 'Un missatge de prova per al criptosistema afí.'
```

Ara, com a tira de caràcters:

```
In [50]:  xxifrat1=AfiX(missatge,[37,59])
```

```
In [51]:  print(xxifrat1)
```

```
಄အӛ࿼ཨლლเჿ༞໔ӛຯ໔ӛၫႵ၆ᅉเӛၫ໔Ⴕӛเ࿗ӛຊႵཨၫჿ၆ლཨლჿ໔࿼เӛเ໹≼ۡ
```

```
In [52]:  Codis(xxifrat1)==xifrat1
```

```
Out[52]: True
```

```
In [53]:  Caracters(xifrat1)==xxifrat1
```

```
Out[53]: True
```

```
In [54]:  xifrat2=AfiX(missatge,[37,59])
```

```
In [55]:  xifrat2
```

```
Out[55]: '಄အӛ\u0ffcཨლლเჿ༞໔ӛຯ໔ӛၫႵ၆ᅉเӛၫ໔Ⴕӛเ࿗ӛຊႵཨၫჿ၆ლཨლჿ໔\u0ffcเӛเ\u0ef9
≼ۡ'
```

```
In [56]:  xxifrat2=Codis(xifrat2)
```

```
In [57]:  print(xxifrat2)
```

```
[3204, 4129, 1243, 4092, 3944, 4314, 4314, 3648, 4351, 3870, 3796,
1243, 3759, 3796, 1243, 4203, 4277, 4166, 4425, 3648, 1243, 4203, 3
796, 4277, 1243, 3648, 4055, 1243, 3722, 4277, 3944, 4203, 4351, 41
66, 4314, 3944, 4314, 4351, 3796, 4092, 3648, 1243, 3648, 3833, 882
8, 1761]
```

```
In [58]:  mis2=AfiDX(xxifrat2,[37,59])
```

```
In [59]:  print(mis2)
```

```
[85, 110, 32, 109, 105, 115, 115, 97, 116, 103, 101, 32, 100, 101,
32, 112, 114, 111, 118, 97, 32, 112, 101, 114, 32, 97, 108, 32, 99,
114, 105, 112, 116, 111, 115, 105, 115, 116, 101, 109, 97, 32, 97,
102, 237, 46]
```

```
In [60]:  Caracters(mis2)
```

```
Out[60]: 'Un missatge de prova per al criptosistema afí.'
```

1.4.4. Anàlisi de freqüències

Però una anàlisi de freqüències pot trencar la clau de manera prou senzilla. De fet, només necessitem "endevinar" a quins caràcters corresponen els dos o tres més freqüents a fi de

tenir un sistema d'equacions lineals, mòdul N, la solució del qual sigui la clau.

Intentem fer-ho explícitament.

In [61]:
```
print(xifrat1)
```

```
[3204, 4129, 1243, 4092, 3944, 4314, 4314, 3648, 4351, 3870, 3796,
1243, 3759, 3796, 1243, 4203, 4277, 4166, 4425, 3648, 1243, 4203, 3
796, 4277, 1243, 3648, 4055, 1243, 3722, 4277, 3944, 4203, 4351, 41
66, 4314, 3944, 4314, 4351, 3796, 4092, 3648, 1243, 3648, 3833, 882
8, 1761]
```

In [62]:
```
fr=[[i,xifrat1.count(i)] for i in set(xifrat1)]
```

In [63]:
```
print(fr)
```

```
[[3204, 1], [3722, 1], [3870, 1], [4129, 1], [3759, 1], [4277, 3],
[3648, 5], [4166, 2], [4425, 1], [3796, 4], [4055, 1], [4314, 4],
[1243, 7], [1761, 1], [3944, 3], [4203, 3], [4092, 2], [3833, 1],
[8828, 1], [4351, 3]]
```

In [64]:
```
w=sorted((u:=transpose(matrix(fr)))[1],reverse=True)
```

In [65]:
```
u
```

Out[65]:
```
[3204 3722 3870 4129 3759 4277 3648 4166 4425 3796 4055 4314 1243 1
761 3944 4203 4092 3833 8828 4351]
[   1    1    1    1    1    3    5    2    1    4    1    4    7
1    3    3    2    1    1    3]
```

In [66]:
```
w
```

Out[66]:
```
[7, 5, 4, 4, 3, 3, 3, 3, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

In [67]:
```
cfr=[u[0][list(u[1]).index(i)] for i in w[0:5]]
```

In [68]:
```
print(cfr)
```

```
[1243, 3648, 3796, 3796, 4277]
```

Acabem de veure quins són els caràcters més freqüents. Ens podem imaginar que el més freqüent correspon a l'espai en blanc i el següent a la lletra 'e'. Això produeix un sistema d'equacions lineals, que intentem resoldre.

In [69]:
```
m=1114112
```

In [70]:
```
var('x,y')
```

Out[70]:
```
(x, y)
```

In [71]:
```
eq1=[ord(' ')*x+y==cfr[0],ord('e')*x+y==cfr[1]]
```

In [72]:
```
1 solve_mod(eq1,m)
```

Out[72]:
```
[(1001121, 274619)]
```

In [73]:
```
1 AfiDX(xxifrat1,[1001121, 274619])
```

Out[73]:
```
U000a33a6 \U000ff4a1\U0004f88d\U000f6ebf\U000f6ebfe\U0009adc4\U0\ϰ'
0107a83\U000afc79 \U0010bd74\U000afc79 \U000fb1b0\U00042fba\U000472
ab\U000f2bcee \U000fb1b0\U000afc79\U00042fba e\U0004b59c \U00057e6f
\U00042fba\U0004f88d\U000fb1b0\U0009adc4\U000472ab\U000f6ebf\U0004f
88d\U000f6ebf\U0009adc4\U000afc79\U000ff4a1e e\U00053b7e\U000a7721
'\U00047266
```

En cas que no haguem "endevinat" els caràcters més freqüents, podem intentar fer algunes proves similars; per exemple, amb altres vocals, o amb l'ordre invers, o...

Aquesta no és la clau. Provem amb 'a' en lloc de 'e'.

In [74]:
```
1 eq2=[ord(' ')*x+y==cfr[0],ord('a')*x+y==cfr[1]]
```

In [75]:
```
1 solve_mod(eq2,m)
```

Out[75]:
```
[(37, 59)]
```

No ha costat gaire!

Fi del capítol 1