Hill ciphers

Hill cipher and affine cipher are alike. Both of them employ a function $f(x) = ax + b$ to encrypt. the difference is the dimension. Affine cipher are one-dimensional, take one character each time. Hill ciphers act on blocks of k characters. It forces to consider $x = \vec{x}$ and $b = \vec{b}$ with $\vec{x}$ and $\vec{b}$ two k-dimensional vectors and $a = A$ a $k \times k$ matrix.

In the following program we take

$$A = \begin{pmatrix} \texttt{key11} & \texttt{key12} \\ \texttt{key21} & \texttt{key22} \end{pmatrix} = \begin{pmatrix} 9 & 5 \\ 7 & 4 \end{pmatrix} \qquad \text{and} \qquad \vec{b} = \begin{pmatrix} \texttt{key1} \\ \texttt{key2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

The blocks are of dimension 2. The variable `digraph` runs over the block of two consecutive characters.

```
def encrypt_hill(message, key11, key12, key21, key22, key1, key2):
    alph = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

    encrypted = ''
    for i in range(0,len(message),2):
        digraph = message[i:i+2]
        encrypted += alph[Mod(key11*alph.find(digraph[0])
                     +key12*alph.find(digraph[1]) + key1,26)]
        encrypted += alph[Mod(key21*alph.find(digraph[0])
                     +key22*alph.find(digraph[1]) + key2,26)]
    print message,'->',encrypted
```

Encrypting `WHATEVER`

```
# encrypt using the matrix [9, 5; 7, 4] and b=0
encrypt_hill('WHATEVER', 9,5,7,4, 0,0)
```

we get `ZARYLIRS`.

To decrypt this message we have to employ the inverse function $A^{-1}(\vec{x}-\vec{b}) = A^{-1}\vec{x} - A^{-1}\vec{b}$. The inverse should be computed $\pmod{26}$. In this case it does not matter because A^{-1} is an integral matrix.

$$A^{-1} = \begin{pmatrix} 9 & 5 \\ 7 & 4 \end{pmatrix}^{-1} = \begin{pmatrix} 4 & -5 \\ -7 & 9 \end{pmatrix} \qquad \text{and} \qquad A^{-1}\vec{b} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Consequently

```
# decrypt using its inverse [4, -5; -7, 9] and b=0
encrypt_hill('ZARYLIRS', 4,-5,-7,9, 0,0)
```

gives `WHATEVER`.

Let us practice with a non-zero vector $\vec{b}$ take for instance

$$A = \begin{pmatrix} \texttt{key11} & \texttt{key12} \\ \texttt{key21} & \texttt{key22} \end{pmatrix} = \begin{pmatrix} 3 & 5 \\ 7 & 2 \end{pmatrix} \qquad \text{and} \qquad \vec{b} = \begin{pmatrix} \texttt{key1} \\ \texttt{key2} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

```
# encrypt using the matrix A=[3, 5; 7, 2] and b=[1,1]
encrypt_hill('SECRET', 3,5,7,2, 1,1)
```

We obtain XFOXEP.

To decrypt we have to compute

$$A^{-1} = \begin{pmatrix} 3 & 5 \\ 7 & 2 \end{pmatrix}^{-1} = -\frac{1}{29}\begin{pmatrix} 2 & -5 \\ -7 & 3 \end{pmatrix} \equiv \begin{pmatrix} 8 & 19 \\ 11 & 25 \end{pmatrix} \qquad \text{and} \qquad A^{-1}\vec{b} \equiv \begin{pmatrix} 25 \\ 16 \end{pmatrix}.$$

Then we recover SECRET with

```
# decrypt using  A^-1=[8, 19; 11, 25] and -A^-1b=[25,16]  (26)
encrypt_hill('XFOXEP', 8,19,11,25, 25,16)
```