Butlletí de la Societat Catalana de Matemàtiques
Vol. 33, núm. 2, 2018. Pàg. 147–171. DOI: 10.2436/20.2002.01.83

Codis correctors d'errors i criptografia postquàntica

Narcís Sayols i Sebastià Xambó

Resum: Revisitem el sistema criptogràfic de clau pública de McEliece, introduït fa quaranta anys, amb l'ajuda de recursos desenvolupats recentment: una millora del descodificador de Peterson-Gorenstein-Zierler per als codis correctors d'errors alternants; un sistema de computació simbòlica i un paquet d'utilitats funcionals per als càlculs emprats en la definició, codificació i descodificació de codis correctors d'errors, tot programat en Python, i una pàgina web que dona accés lliure als materials generats pel projecte. L'interès principal del sistema de McEliece rau en el fet que és un candidat seriós per a un estàndard de criptografia postquàntica.

Paraules clau: criptografia postquàntica, codis alternants, computació simbòlica, programari lliure.

Classificació MSC2010: 11T71, 68-04, 68W30, 94A60.

Pròleg

Com que els autors d'aquest article no som experts en criptografia, abans de res ens sembla convenient comentar què ens ha portat a escriure'l.

El context ha estat el treball, iniciat la tardor del 2015, per dissenyar i programar PyECC, un entorn que permeti la construcció, codificació i descodificació de codis correctors d'errors i posar-lo a la lliure disposició del públic, tot pensant bàsicament en estudiants, docents i investigadors. El nucli d'aquest entorn és un paquet de classes i funcions de Python[1] que anomenem CC (vegeu [46]).

Inicialment la idea fou igualar la funcionalitat del paquet Wiris/cc descrit a [56], un entorn desenvolupat a fi de poder realitzar les tasques computacionals esmentades per als codis correctors tractats en aquell text. Però aviat ens van convèncer que podíem anar més enllà en diverses direccions. Una d'elles, que ha resultat ser un bon test per a PyECC, fou la implementació d'un prototip rudimentari del sistema criptogràfic de clau pública descrit a [32].

1 https://ca.wikipedia.org/wiki/Python.

Aquest estudi del sistema de McEliece ens portà a interessar-nos per la seva evolució des que es va publicar fa quaranta anys fins a l'actualitat. El resultat és que segueix sent, com veurem, una de les línies més prometedores en un horitzó de *criptografia postquàntica*, locució introduïda el 2003 per D. J. Bernstein (vegeu [41]) per designar paradigmes criptogràfics que garanteixin protecció contra qualsevol giny de computació, actual o futur, incloent-hi els sistemes de computació quàntica (per a una introducció general a la computació quàntica, vegeu, per exemple, [45]; per a una avaluació recent de la possible evolució de l'enginyeria de sistemes de computació quàntica i dels sistemes de *criptografia quàntica*, vegeu [17, 16] i la nota al final de la secció 5). Aquest fet ens sembla que té una gran importància, ja que els estàndards criptogràfics usats en l'actualitat no ofereixen aquestes garanties. Per exemple, els sistemes basats en la dificultat de factoritzar números enters, com ara el sistema RSA,[2] resulten extremament vulnerables a la computació quàntica a causa de la (baixa) complexitat polinòmica del q-algorisme de Shor per factoritzar números enters (vegeu [45, § 8]).

En gran mesura, el contingut d'aquest treball es correspon amb el de la conferència [47] programada en la Jornada de Teoria de Nombres celebrada el 7 d'octubre del 2017 al Departament de Matemàtica de la Universitat de Lleida.[3] Tanmateix, el desig d'augmentar-ne l'assequibilitat ens ha impulsat a incorporar força més detalls dels que es poden comunicar en una presentació oral, i a introduir variacions en l'ordre expositiu per aconseguir que el pendent de la lectura sigui tan suau com sigui possible.

A la primera secció es descriuen breument els ingredients d'un sistema criptogràfic de clau pública i amb més detall, a la segona secció, els que concorren en un sistema de McEliece. Tot seguit s'exposen, a la tercera secció, les generalitats que necessitem sobre codis correctors d'errors, incloent-hi els detalls específics sobre els codis d'un sistema de McEliece (codis de Goppa clàssics binaris). A les seccions quarta i cinquena es tracten, respectivament, l'evolució dels sistemes de McEliece i el grau de seguretat criptogràfica que ofereixen. L'apèndix A conté una breu notícia sobre PyECC, i en particular sobre el seu ús i instal·lació, i a l'apèndix B es donen els detalls més rellevants sobre la implementació usant PyECC d'un sistema de McEliece.

1 Criptografia de clau pública

En un *sistema criptogràfic*, un missatge M és *encriptat* en un *criptograma* C amb una funció E_k que depèn d'un paràmetre k anomenat *clau d'encriptació*:

$$C = E_k(M).$$

2 Vegeu `https://ca.wikipedia.org/wiki/RSA`.

3 Seminari «Cargols 2017», organitzat pel Grup de Recerca en Criptografia i Grafs, `http://www.cig.udl.cat/`.

La *descodificació* $D_{k'}(C)$ corresponent es realitza mitjançant una funció $D_{k'}$, també depenent d'un paràmetre k' (*clau de desencriptació*), i el primer requisit fonamental és que s'ha de complir la relació

$$D_{k'}(C) = D_{k'}(E_k(M)) = M. \tag{1}$$

A més, les funcions E i D s'han de poder calcular eficientment.

Per tal que es pugui verificar l'equació (1), k i k' han d'estar relacionades i, per tant, un usuari emissor i un usuari receptor han de compartir informació adient sobre k i k'. L'intercanvi d'aquesta informació de manera que altres usuaris no hi puguin accedir és l'anomenat *problema de distribució de claus*. Un pas cabdal en la solució d'aquest problema fou l'esquema de Diffie i Hellman [22] publicat el 1976. Segons afirmen en el resum els seus autors: «les necessitats de telecomunicació han donat lloc a la necessitat de nous tipus de sistemes criptogràfics que minimitzin la necessitat de disposar de canals de distribució de claus segures i que proporcionin l'equivalent a una signatura per escrit. Aquest article suggereix maneres de resoldre aquests problemes actualment oberts». Una de les maneres que proposen és la dels *sistemes criptogràfics de clau pública*, «que permeten la transmissió d'informació per canals insegurs sense comprometre la seguretat del sistema».

En un sistema criptogràfic de *clau pública*, com ara el de McEliece, cada usuari té una clau d'encriptació *pública* i una clau de desencriptació *privada* (o secreta). Abans d'entrar en més detalls, il·lustrem breument el funcionament en el cas del familiar sistema RSA. Publicat el 1978 per Rivest, Shamir i Adleman [44], en descrivim el funcionament a continuació (com a referència general, vegeu [20]).

Tot usuari disposa de la seva clau en una part privada i una part pública. La part privada està formada per dos números primers p i q, i la clau pública, pel producte $n = pq$ i per un número natural k primer amb $(p-1)(q-1)$. L'encriptació que fa un altre usuari del missatge M (que podem suposar que és un número enter no negatiu inferior a n) s'obté per la fórmula

$$C = M^k \bmod n, \tag{2}$$

i la descodificació amb

$$C^{k'} \bmod n, \tag{3}$$

on k' és l'invers de k mòdul $(p-1)(q-1)$. La relació (1) en aquest cas significa que

$$M^{kk'} = M \bmod n, \tag{4}$$

la qual és una conseqüència directa de resultats elementals d'aritmètica modular i per als quals el lector pot consultar [20, § 3.3.1]. Adonem-nos que un sistema RSA és segur sempre que la factorització de números enters grans sigui un problema difícil.

2 Ingredients d'un criptosistema de McEliece

Els ingredients *generals* d'aquest protocol d'encriptació són els següents:

- $F = F_q$, un cos finit de cardinal q (aquí el cas més important serà $F = \mathbb{Z}_2$).
- k, un número enter positiu. Dels elements de l'espai F^k en direm *missatges*.
- $n > k$, un número enter. Dels elements de l'espai F^n en direm *missatges de transmissió* o també *missatges de canal*. Si $\boldsymbol{x} \in F^n$, posarem $|\boldsymbol{x}|$ per denotar el nombre de components no nul·les de $\boldsymbol{x}$ i direm que és el *pes* de $\boldsymbol{x}$.

Aquests ingredients són públics i hom suposa que qualsevol usuari del protocol pot generar missatges i missatges de transmissió, i que pot enviar qualsevol missatge de transmissió a qualsevol altre usuari.

Cada usuari també ha de disposar de mitjans per generar els objectes i efectuar els càlculs adients al seu paper d'emissor o receptor. Tot seguit descrivim aquests objectes i operacions segons que es tracti de generar claus (clau privada i clau pública) per tal d'esdevenir un usuari receptor, d'encriptar i enviar un missatge en el cas d'un usuari emissor, o de desencriptar un missatge rebut en el cas d'un receptor.

Generació de claus. Una *clau privada*, que denotarem $\{G, S, P\}$, està formada per tres matrius d'elements de F:

- G, matriu $k \times n$ de rang k.
- S, matriu $k \times k$ invertible seleccionada aleatòriament segons la distribució uniforme.
- P, matriu $n \times n$ de permutació seleccionada aleatòriament, també segons la distribució uniforme.

La propietat més decisiva de la matriu G per a un usuari receptor és que ha d'existir una funció

$$D\colon X \to F^n, \text{ on } X \subseteq F^n,$$

que gaudeixi de la propietat següent, on t és un número enter positiu prefixat:

$(*)$ Per a tot $\boldsymbol{u} \in F^k$ i tot $\boldsymbol{e} \in F^n$ amb $|\boldsymbol{e}| \leq t$, es compleix

$$\boldsymbol{x} = \boldsymbol{u}G + \boldsymbol{e} \in X \quad \text{i} \quad D(\boldsymbol{x}) = \boldsymbol{u}G.$$

D'aquesta funció D en direm *descodificador*, i ens referirem a la propietat anterior dient que *D pot corregir fins a t errors*. Dels elements de X es diu que són vectors *descodificables*.

En principi, cada usuari pot tenir el seu propi descodificador, en el sentit que l'algorisme emprat per calcular D, i la seva implementació, poden ser privats per als usuaris que ho desitgin. Tanmateix, cal dir que un sistema complet ha d'incorporar un descodificador que serveixi per a tothom (descodificador *per*

defecte), i que en principi és raonable pensar que tots els usuaris poden ser emissors i receptors.

La teoria fonamental que usarem per a la construcció de descodificadors apropiats és la teoria de codis correctors d'errors.

La *clau pública* corresponent a la clau privada $\{G, S, P\}$ és un parell $\{G', t\}$ definit com segueix:

- $G' = SGP$, que és una matriu $k \times n$.
- Un número enter positiu t que satisfà $(*)$.

Protocol d'encriptació. El protocol que ha de seguir un usuari que vol encriptar i enviar un missatge $\boldsymbol{u}$ a l'usuari amb clau pública $\{G', t\}$ consta de dos passos:

- Generació aleatòria d'un vector de transmissió $\boldsymbol{e}$ de pes t.
- Tramesa del vector $\boldsymbol{x} = \boldsymbol{u}G' + \boldsymbol{e} = \boldsymbol{u}SGP + \boldsymbol{e}$.

Protocol de desencriptació. Consta de quatre passos que només usen informació privada del receptor i el vector $\boldsymbol{x}$ enviat per l'emissor:

- Càlcul de $\boldsymbol{y} = \boldsymbol{x}P^{-1}$, de manera que $\boldsymbol{y} = (\boldsymbol{u}S)G + \boldsymbol{e}P^{-1}$.
- Càlcul de $\boldsymbol{x}' = D(\boldsymbol{y})$. Com que P és una matriu de permutació,

$$|\boldsymbol{e}P^{-1}| = |\boldsymbol{e}| = t$$

  i, per tant, aquesta operació està ben definida, ja que D corregeix fins a t errors. El resultat és $\boldsymbol{x}' = (\boldsymbol{u}S)G$. És a dir, $\boldsymbol{x}'$ és una combinació lineal de les files de G amb coeficients $\boldsymbol{u}' = \boldsymbol{u}S$.
- Com que G té rang k, $\boldsymbol{u}'$ queda unívocament determinat per $\boldsymbol{x}'$ i es pot obtenir resolent el sistema d'equacions lineals $\boldsymbol{x}' = \boldsymbol{u}'G$, on la incògnita és el vector $\boldsymbol{u}'$.
- Càlcul de $\boldsymbol{u} = \boldsymbol{u}'S^{-1}$. Missatge rebut correctament!

Per als aspectes computacionals d'aquests ingredients, vegeu: A.2 «Aritmètica modular», A.3 «Polinomis irreductibles», A.4 «Construcció de cossos finits», A.5 «Vectors i matrius», i A.6 «Utilitats per al sistema McEliece».

3 Codis de Goppa clàssics

En el que segueix, posem $F(k, n)$ per denotar les matrius d'ordre $k \times n$ formades amb elements d'un cos F.

Codis lineals: conceptes bàsics i notacions Un codi (lineal) de tipus $[n, k]$ sobre F_q és un subespai vectorial C de dimensió k de F_q^n. En particular s'ha de complir $k \leq n$. Diem que n i k són la *longitud* i la *dimensió* de C, respectivament, i escrivim $C \sim [n, k]$. La *taxa de transmissió* de C és el quocient $R = k/n$.

Una matriu G de tipus $k \times n$ es diu que és una *matriu generadora* de C si les seves files formen una base de C. Donada una matriu G de tipus $k \times n$, posem $\langle G \rangle$ per denotar el subespai vectorial generat per les files de G. És un codi de tipus $[k', n]$, on k' és el rang de G. En el cas que G sigui una matriu generadora de C, llavors $\langle G \rangle = C$.

Si G és una matriu generadora del codi $C \sim [n, k]$, l'aplicació lineal $f\colon F_q^k \to F_q^n$, $\boldsymbol{u} \mapsto \boldsymbol{u}G$, és injectiva i la seva imatge és C. De l'aplicació f es diu que és un *codificador* de C, i si $\boldsymbol{u} \in F_q^k$, $\boldsymbol{x} = f(\boldsymbol{u})$ és la *codificació* de $\boldsymbol{u}$.

EXEMPLE (EL CODI DE HAMMING BINARI $[7, 4]$). $\boldsymbol{x} = \boldsymbol{u}G$, on

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Els codis lineals també es poden descriure mitjançant el que podem anomenar *construcció dual*. Sigui H una matriu de tipus $r \times n$ i suposem que el seu rang és r. Llavors

$$C_H = \{\boldsymbol{x} \in \mathbb{F}^n : \boldsymbol{x}H^T = 0\} = \ker H^T$$

és un codi lineal $[n, n - r]$. De la matriu H es diu que és una *matriu de control* de C_H. És clar que podem formar una matriu generadora de C_H escollint una base de $\ker H^T$.

EXEMPLE. La matriu

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

és una matriu de control del codi de Hamming $[7, 4]$. En efecte, si posem R per denotar la submatriu formada per les tres darreres columnes de G, llavors $G = (I_4|R)$, $H = (R^T|I_3)$, i

$$GH^T = (I_4|R)\begin{pmatrix} R \\ I_3 \end{pmatrix} = R + R = 0.$$

Per tant, $C = \langle G \rangle \subseteq \ker H^T$, que de fet és una igualtat perquè els dos termes de la inclusió tenen dimensió 4.

REMARCA. Una matriu generadora de la forma $G = (I_k|R)$ es diu que és *sistemàtica* respecte de les components $1, \ldots, k$, ja que $\boldsymbol{x} = \boldsymbol{u}G = (\boldsymbol{u}|\boldsymbol{u}R)$ i això permet recuperar $\boldsymbol{u}$ com les primeres k components de $\boldsymbol{x}$. En aquest cas, un argument similar al de l'exemple anterior ens permet concloure que $H = (-R^T|I_{n-k})$ és una matriu de control de $\langle G \rangle$. Remarquem que $H = (R^T|I_{n-k})$ en el cas binari.

La construcció dual admet la generalització que descrivim a continuació i que usarem per construir els codis de Goppa clàssics. Sigui $F = F_q$ i $\bar{F} = F_{q^m}$, on m és un número enter positiu. Sigui $\bar{H}$ una matriu de tipus $r \times n$ amb

components de $\bar{F}$ i suposem que el seu rang és r. Llavors podem definir el codi lineal/F

$$C_{\bar{H},F} = C_{\bar{H}} \cap F^n = \{\boldsymbol{x} \in F^n : \boldsymbol{x}\bar{H}^T = 0\}.$$

Per trobar una matriu de control H (sobre F) d'aquest codi, podem procedir de la manera següent. Sigui H' la matriu sobre F obtinguda substituint cada element β de $\bar{H}$ per la columna de les m components de β relativament a una base de $\bar{F}$ sobre F, de manera que $H' \in F(rm, n)$. Finalment, sigui H el resultat de suprimir les files de H' que són combinació lineal de les anteriors, $H \in F(r', n)$. De la construcció es desprèn que

$$C_{\bar{H},F} = C_{H'} = C_H$$

i, per tant, $\dim C_H = n - r'$. Atès que $r' \leq mr$, resulta que $k \geq n - r'$. També es compleix que $r' \geq r$ o $k \leq n - r$ (vegeu, per exemple, [56, prop. 4.1]). Per a un exemple concret de càlcul de H', H, r' i k, vegeu A.7.

Un *descodificador* d'un codi C és una aplicació exhaustiva $g\colon F^n \to C \sqcup \mathcal{E}$ (unió disjunta), on $\mathcal{E}$ és un conjunt de «missatges d'error», que compleix $g(\boldsymbol{x}) = \boldsymbol{x}$ per a tot $\boldsymbol{x} \in C$.

El conjunt $g^{-1}(C) \subseteq F^n$ és el dels *vectors descodificables*, mentre que el conjunt $F^n - g^{-1}(C) = g^{-1}\mathcal{E}$ és el conjunt dels *vectors erronis*. Un descodificador és *complet* si $g^{-1}(C) = F^n$ (cas en el qual $\mathcal{E} = \emptyset$).

Diem que el descodificador g té *capacitat correctora* t si $g(\boldsymbol{y}) = \boldsymbol{x}$ per a tot vector $\boldsymbol{y} \in F^n$ tal que $|\boldsymbol{y} - \boldsymbol{x}| \leq t$.

Si posem $B(\boldsymbol{x}, t) = \{\boldsymbol{y} \in \mathbb{F}^n : |\boldsymbol{y} - \boldsymbol{x}| \leq t\}$ (d'aquest conjunt es diu que és la *bola de Hamming* de centre $\boldsymbol{x}$ i radi t), aleshores tenim que $\cup_{\boldsymbol{x}\in C} B(\boldsymbol{x}, t) \subseteq g^{-1}(C)$ i $g(B(\boldsymbol{x}, t)) = \{\boldsymbol{x}\}$.

Codis de Goppa binaris Per completar la construcció d'un sistema de McEliece, l'ingredient més important que ens falta és una matriu generadora binària G que compleixi la condició $(*)$ de la pàgina 150. Dediquem el que resta d'aquesta secció a descriure com es resol aquest problema. Per a la construcció efectiva de les matrius S i P, vegeu A.6.

Si bé és cert que el sistema inicial de McEliece és binari, la descripció dels codis de Goppa clàssics la farem sobre un cos finit F qualsevol. Preparem així el terreny per poder considerar variacions del sistema original introduïdes posteriorment sense que, de fet, representi cap dificultat teòrica addicional.

Sigui, doncs, $F = F_q$ un cos finit ($F = \mathbb{Z}_2$ en el cas binari). Escollim un número enter positiu m i posem $\bar{F} = F_{q^m}$. Escollim elements distints $\boldsymbol{\alpha} = \alpha_1, \ldots, \alpha_n \in \bar{F}$, cosa que implica $n \leq q^m$, i un polinomi $p \in \bar{F}[T]$ de grau $r \geq 1$ tal que $p(\alpha_j) \neq 0$ $(j = 1, \ldots, n)$. Finalment posem $h_j = 1/p(\alpha_j)$ $(j = 1, \ldots, n)$ i $\Gamma = \Gamma(p, \boldsymbol{\alpha}) := C_{\bar{H},F}$, on

$$\bar{H} = \begin{pmatrix} h_1 & \ldots & h_n \\ h_1\alpha_1 & \ldots & h_n\alpha_n \\ \vdots & & \vdots \\ h_1\alpha_1^{r-1} & \ldots & h_n\alpha_n^{r-1} \end{pmatrix}.$$

Diem que Γ és el *codi de Goppa clàssic* associat a p i $\boldsymbol{\alpha}$. Com que el nombre de files de $\bar{H}$ és r, i $\bar{F}$ té grau m sobre F, $k = \dim\Gamma(p, \boldsymbol{\alpha})$ compleix, pel que ja hem dit, que $n - r \geq k \geq n - rm$. A més, resulta que Γ corregeix t o menys errors, on t és la part entera de $r/2$ [56, p. 191]. En el cas binari, Γ corregeix r o menys errors si p no té arrels múltiples [56, exercici P.4.7, p. 194]. De fet, en aquest cas es compleix que $\Gamma(p, \boldsymbol{\alpha}) = \Gamma(p^2, \boldsymbol{\alpha})$ i, per tant, aquest codi corregeix t errors si $t \leq r$. Per a un exemple detallat, vegeu A.8.

Descodificadors Els codis de Goppa clàssics són un cas especial dels anomenats *codis alternants*. L'interès d'aquests codis no rau només en el fet que els constructors de codis de Goppa clàssics són especialitzacions dels constructors de codis alternants, sinó que els codis alternants admeten descodificadors generals eficients que proporcionen bons descodificadors per als codis de Goppa clàssics i, en particular, per als codis de Goppa clàssics binaris.

El paquest PyECC inclou implementacions dels dos descodificadors principals per a codis alternants: el de Berlekamp-Massey-Sugiyama (BMS) i el de Petterson-Gorenstein-Zierler (PGZ). Vegeu l'exemple A.9.

4 Evolució dels sistemes de McEliece

El context en què McEliece va proposar el seu sistema va ser el creat pels treballs de Diffie i Hellman [22], en què van introduir la noció de sistema criptogràfic de clau pública; de Merkle i Hellman [33], en què van proposar el sistema basat en la dificultat de l'anomenat *problema de la motxilla*,[4] i sobretot de Rivest, Shamir i Adleman [44], en què van proposar el celebrat sistema RSA basat en la dificultat de factoritzar números enters (vegeu § 1). El sistema de McEliece fou el primer sistema de clau pública basat en la teoria dels codis algebraics que havia introduït Goppa el 1970 [28], i que aquí en diem *codis de Goppa clàssics* (en el marc dels codis de Goppa generals, es corresponen amb els codis de gènere 0).[5] Esmentem també que els paràmetres escollits per McEliece per il·lustrar diversos aspectes del seu sistema van ser $[n, k] = [1024, 524]$ i $t = 50$, que garantien una seguretat de més de 60 bits, considerable fa quaranta anys, però no, com veurem a la secció següent, en les circumstàncies actuals.

Una modificació del sistema de McEliece va ser proposada per Niederreiter el 1986 [37]. La clau privada d'aquest sistema és una terna $\{H, S, P\}$ de matrius binàries de tipus $n \times (n-k)$, $(n-k) \times (n-k)$ i $n \times n$, respectivament. Aquestes matrius han de satisfer les condicions següents: H és una matriu de control d'un codi de Goppa clàssic binari $\Gamma \sim [n, k]$ (és a dir, $\boldsymbol{y} \in \Gamma$ si i només si $\boldsymbol{y}H^T = 0$) que corregeix t errors, S és una matriu no singular escollida aleatòriament segons la distribució uniforme, i P és una matriu de permutació,

4 Vegeu `https://ca.wikipedia.org/wiki/Problema_de_la_motxilla`, on també es documenta la ruptura d'aquest sistema el 1982 per Shamir.

5 McEliece no cita l'article de Goppa i, en canvi, remet a les pàgines 179–180 i 193–194 de [31].

també escollida aleatòriament segons la distribució uniforme. La clau pública corresponent és el parell $\{H', t\}$, on $H' = PHS$. Els missatges $\boldsymbol{u}$ són vectors binaris de longitud n i pes t. El missatge encriptat és $\boldsymbol{x} = \boldsymbol{u}H' = \boldsymbol{u}PHS$. El receptor recupera $\boldsymbol{u}$ de la manera següent: (1) calcula $\boldsymbol{s} = \boldsymbol{x}S^{-1} = \boldsymbol{u}PH$; (2) com que coneix H, pot trobar $\boldsymbol{u}' = \boldsymbol{u}P = D(\boldsymbol{s}, \boldsymbol{x})$ amb un descodificador D apropiat (algorisme de Patterson [39], que avui convé veure com una variant dels descodificadors estàndard BMS o PGZ), i (3) com que coneix P, pot trobar $\boldsymbol{u} = \boldsymbol{u}'P^{-1}$, que és el missatge enviat. Els valors dels paràmetres en la il·lustració inicial eren $[n, k] = [1024, 644]$, $t = 38$. Per a la generació eficient de missatges de pes t i longitud n, vegeu [48].

En l'estudi dels sistemes de McEliece i Niederreiter presentat a la memòria d'habilitació de Sendrier de 2002 [49], s'afirma: «Després de vint anys d'esforços [i cita una dotzena llarga de treballs], cap criptoanàlisi ha pogut rompre aquests sistemes». Com veurem, aquesta apreciació segueix essent vàlida.

El problema de modificar els sistemes de McEliece i Niederreiter per poder funcionar amb claus públiques substancialment menors és considerat per Gaborit a [26]. Serveixin, com a il·lustració del resultat, les dades següents: per a longituds 2047 i 4095, són suficients claus públiques de 12 KB i 20 KB, respectivament. Per a més detalls de com estaven les coses fa una dècada, és recomanable l'article [24]. Per als autors, «la criptografia basada en teoria de codis és una alternativa interessant a la criptografia basada en teoria de nombres», ja que «moltes de les funcions criptogràfiques bàsiques [...] es poden realitzar amb conceptes de teoria de codis».

També cal esmentar l'article [12], que entre d'altres qüestions proposa nous paràmetres per als sistemes de McEliece i Niederreiter que assoleixen bons nivells de seguretat contra tots els atacs coneguts i amb longituds de les claus considerablement menors. A més, s'afirma que «els computadors quàntics no aporten millores significatives a la potència dels atacs més enllà de possibles millores genèriques derivades de l'algorisme de Grover i, per tant, l'esquema criptogràfic de McEliece és un dels candidats interessants per a la criptografia postquàntica» (per a l'algorisme de Grover, podeu consultar per exemple [45, § 5]).

Si bé la noció de criptografia postquàntica va ser introduïda, com ja s'ha dit, el 2003, al nostre entendre l'inici més visible es produeix quan a la publicació del magnífic manual de criptografia convencional [29] va seguir, un any més tard, la del volum [10]. En aquest volum, el primer d'aquest nou enfocament de la criptografia, és important l'article introductori de D. J. Bernstein [7], que en particular presenta els quatre paradigmes considerats més prometedors. A més de la criptografia basada en codis (article d'Overbeck i Sendrier [38]), es consideren també sistemes basats en polinomis de diverses variables, en reticles de $\mathbb{R}^n$ (vegeu [53] per a un article recent sobre la qüestió), i en funcions *hash*. En l'article de Bernstein tot just esmentat, trobem el que podria ser el *leitmotiv* de la criptografia postquàntica: «No hi ha justificació per saltar de l'afirmació "els computadors quàntics destrueixen RSA, DSA i ECDSA" a l'afirmació "els computadors quàntics destrueixen la criptografia". Hi ha moltes

classes importants de sistemes criptogràfics més enllà de RSA, DSA i ECDSA». De fet, en l'article d'Overbeck i Sendrier llegim: «Passades tres dècades, s'han hagut d'ajustar una mica els paràmetres, però no es coneix cap atac que representi una amenaça seriosa [als sistemes de McEliece], ni tan sols els d'un computador quàntic».

Aquestes apreciacions s'han repetit, i fins i tot reforçat, en els darrers anys. Esmentem una mostra de treballs en què s'aprofundeix en diverses direccions: [15], [13] (aquest article proposa nous paràmetres per aconseguir un bon nivell de seguretat davant de tots els atacs coneguts), [3], [34], [36] (tesi doctoral), [40] (una altra tesi doctoral, en la qual es revitalitzen els codis de Srivastava generalitzats, i amb implementacions efectives), [43], [54] i [21].

En tota aquesta dècada postquàntica, un pioner a tenir en compte és D. J. Bernstein, que ja hem citat i del qual en recomanem la pàgina web: [6]. També val la pena consultar la pàgina que publica amb T. Lange, [41], i seguir la conferència PQC 2018, enfocada a aconseguir una estandardització dels protocols criptogràfics postquàntics i de la qual podeu trobar informació a [19].

5 Seguretat criptogràfica d'un sistema de McEliece

El mateix McEliece a [32] considerà la qüestió de la seguretat que podia oferir el seu sistema. En un planteig general, hom suposa que un espia, diguem-ne E, ha aconseguit el vector encriptat $\boldsymbol{x}$ enviat per un usuari i que coneix la clau pública del destinatari, és a dir, $G' = SGP$ i t. Quines possibilitats té d'obtenir el missatge inicial $\boldsymbol{u}$?

Intentar descodificar $\boldsymbol{x} = \boldsymbol{u}G' + \boldsymbol{e}$ usant la matriu coneguda G' no es pot considerar prometedor, si k és suficientment gran, ja que el problema de descodificar codis lineals $[n, k]$ és NP-complet (cf. [5]).

L'altre atac que va considerar McEliece al seu sistema és més sorprenent. L'espia pot seleccionar k coordenades de $\boldsymbol{x}$ aleatòriament. Si té sort, cap d'aquestes coordenades serà errònia i podrà determinar $\boldsymbol{u}$ resolent el sistema d'equacions lineals $\bar{\boldsymbol{x}} = \boldsymbol{u}\bar{G}'$, on $\bar{\boldsymbol{x}}$ és el vector format amb les coordenades de $\boldsymbol{x}$ seleccionades i $\bar{G}'$ és la submatriu de G' formada amb les columnes corresponents als índexs de $\bar{\boldsymbol{x}}$. Si no té sort, alguna de les entrades seleccionades serà errònia i el sistema no tindrà solució. Aquest atac és més seriós del que podria semblar a primera vista. En efecte, decidir si el sistema $\bar{\boldsymbol{x}} = \boldsymbol{u}\bar{G}'$ és compatible, i trobar la solució si en té, comporta un nombre d'operacions de l'ordre de k^3, mentre que la probabilitat d'escollir k coordenades no errònies és

$$\binom{n-t}{k} \Big/ \binom{n}{k} = \left(1 - \frac{t}{n}\right)\left(1 - \frac{t}{n-1}\right) \cdots \left(1 - \frac{t}{n-(k-1)}\right) < (1 - t/n)^k,$$

de manera que el nombre d'operacions que l'espia ha d'esperar fer per tenir èxit és superior a $k^3/(1 - t/n)^k$. Amb els paràmetres de la il·lustració de McEliece, això comporta un esforç no inferior a 2^{64} operacions, un fet que podem expressar dient que assoleix una seguretat de 64 bits (relativament a l'atac en qüestió).

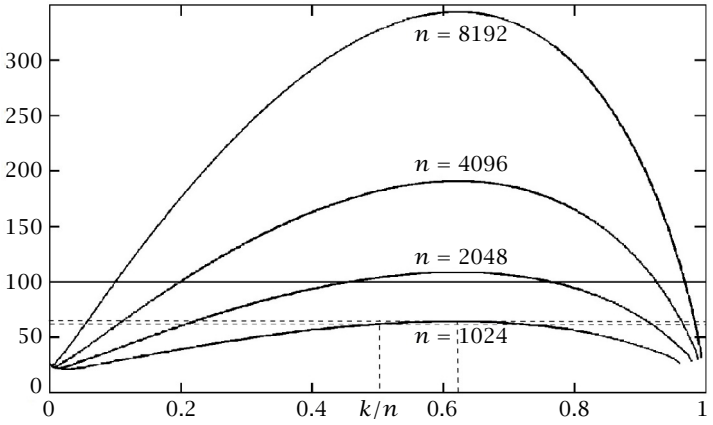

FIGURA 1: Grau de seguretat en bits del sistema de McEliece en funció de la taxa de transmissió $R = k/n$ i per a longituds $n = 2^l$ ($l = 10, \ldots, 13$) del codi. Com a grau de seguretat es pren el logaritme en base 2 del nombre d'unitats de computació (factor de treball) de l'atac més fort conegut.

La figura 1, adaptada de la figura 6.2 de [49], resumeix el grau de seguretat del sistema de McEliece a principis d'aquest segle. Hi hem afegit les línies de punts per tal de poder-lo comparar amb el grau de seguretat que acabem de comentar. Veiem que, tenint en compte tots els atacs, els paràmetres originals garanteixen 63 bits de seguretat, i que amb una taxa una mica superior (0.625 en lloc de $524/1024 = 0.512$) es té, per la mateixa longitud, una seguretat de 64 bits, en remarcable coincidència amb l'anàlisi inicial.

Sendrier també remarca que, si bé els dos sistemes tenen avantatges pel que fa a la facilitat d'implementació, la seva clau pública és relativament gran. També anuncia un resultat que cal considerar important, ja que fins aleshores no s'havia aconseguit trobar: un esquema de signatura digital basat en McEliece.

Les referències que segueixen posen de manifest que aquests resultats de Sendrier sobre el grau de seguretat continuen essencialment vigents, sovint amb variacions pel que fa als codis emprats: [50, 52, 23, 2, 27, 11, 1].

En l'article [2] es consignen les recomanacions inicials del projecte PQCRYPTO, ja esmentat, pel que fa a la confiança en la seguretat dels sistemes, més que no pas en l'eficiència. Per exemple, per als sistemes de McEliece basats en codis de Goppa clàssics binaris, els paràmetres que es proposen són $n = 6960$, $k = 5413$ i $t = 119$ per assolir 128 bits de seguretat. La seguretat de 128 bits és inferior als 172 bits que dona el recompte de McEliece que hem comentat abans, la qual cosa s'explica pel fet que en les estimacions es tenen en compte tots els atacs coneguts que s'havien imaginat fins a aquell moment. Per a més detalls sobre el sistema de McEliece proposat per Bernstein *et al.*, i també sobre una variant del McEliece-Niederreiter, vegeu [18].

Acabem la secció amb unes referències sobre progressos en computació quàntica que poden ajudar a entendre per què aquest recurs, fins i tot amb tota la potència imaginable en el futur, potser no pot arribar a ser tan poderós com s'ha estimat en moments passats i que molt probablement mai serà suficient per trencar alguns dels sistemes que com el de McEliece tenen tot el potencial de ser realment postquàntics: [17, 9, 8, 14].

Tanmateix, cal esmentar que el treball pioner de Shor [51] va contribuir a promoure el camp de la criptografia quàntica, l'objetiu de la qual és usar les propietats quàntiques dels sistemes per dissenyar sistemes criptogràfics. Per a una descripció elemental d'aquests sistemes, vegeu [16] i la bibliografia corresponent. Per a una visió general, que inclou la relació amb la criptografia postquàntica, vegeu [55, 18] i els ja citats [10, 8, 11].

A Introducció a PyECC

L'arquitectura de PyECC es basa en una jerarquia de classes de Python gestionada mitjançant un paquet de funcions que li serveix d'interfície. Per als detalls sobre aquesta arquitectura, així com sobre la instal·lació i ús, vegeu [46]. En la resta d'aquest apèndix s'exposen unes consideracions generals destinades a il·lustrar-ne el funcionament.

A.1 Valors i noms

Tot objecte creat amb PyECC té un nom per defecte. Per exemple, `Zn(6)` crea l'anell $\mathbb{Z}_6$ i el seu nom és `Z6`. Podem optar per donar-li un nom, posem `'A'`, amb la sintaxi `Zn(6,'A')`. La llista següent mostra la seva flexibilitat (emprem els símbols `-->` per indicar que el valor que els segueix és el de l'expressió anterior):

```
Zn(6)
--> Z6 :: Ring
A = Zn(6)
A
--> Z6 :: Ring
Z6 = Zn(6,'A')
Z6
--> A :: Ring
A = Zn(6,'A')
A
--> A :: Ring
```

Un altre exemple és que l'anell de polinomis $A[X]$ sobre l'anell anomenat A es pot crear amb `polynomial_ring(A)`, i es pot optar per donar altres noms a la indeterminada i al mateix anell de polinomis:

```
A = Zn(6)
polynomial_ring(A)
--> [Z6[X] :: Ring, X :: Z6[X]]
```

```
A = Zn(6,'A')
polynomial_ring(A)
--> [A[X] :: Ring, X :: A[X]]
polynomial_ring(A,'t')
--> [A[t] :: Ring, t :: A[t]]
polynomial_ring(A,'t','P')
--> [P :: Ring, t :: P]
polynomial_ring(A,name='P')
--> [P :: Ring, X :: P]
```

Convencions similars són vàlides per a la creació d'extensions de cossos o d'anells, com es veurà en els exemples que segueixen.

A.2 Aritmètica modular

En general, `Zn(n)` crea l'anell $\mathbb{Z}_n = \mathbb{Z}/(n)$, i en particular el cos de n elements quan n és primer. Si A i B són anells construïts i existeix un homomorfisme «canònic» de A a B, la imatge de $a \in A$ a B per aquest homomorfisme es denota `a»B`.

```
A = Zn(17)
# Projecció de 3 a A
a = 3>>A
--> 3 :: Z17
# Ordre de l'element a
order(a)
--> 16
# Ordre de 3 mòdul 17
order(3,17)
--> 16
# Potències de a, vistes com enters
[lift(a**j) for j in range(17)]
--> [1, 3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1]
# Potències de 3 mòdul 17
[3**j % 17 for j in range(17)]
--> [1, 3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1]
# Potències de a
vector([a**j for j in range(17)])
--> [1,3,9,10,13,5,15,11,16,14,8,7,4,12,2,6,1] :: Vector[Z17]
```

A.3 Polinomis irreductibles

Per a la construcció de la matriu G d'una clau privada de McEliece és necessària l'elecció d'un polinomi mònic irreductible p de grau t sobre $F = F_q$ (t és el segon element de la clau pública). A tal fi tenim la funció

```
p = get_irreducible_polynomial(F,t)
```

Com que aquesta funció itera la selecció aleatòria d'un polinomi p mònic de grau t mentre p no sigui irreductible, és important conèixer la probabilitat que p sigui irreductible. El resultat és que aquesta probabilitat és $\sim 1/t$ com a

conseqüència de la *fórmula de Gauss* que dona el nombre $I_q(t)$ de polinomis mònics irreductibles de grau t sobre $\mathbb{F}_q$:

$$I_q(t) = \tfrac{1}{t}\sum_{d|t}\mu(t/d)q^d = \tfrac{q^t}{t} + \cdots$$

De fet, la probabilitat esmentada és $I_q(t)/q^t = \frac{1}{t} + \cdots$, ja que q^t és el nombre de polinomis mònics de grau t. Vegeu la figura 2 en la qual es mostra la primera cel·la d'un *notebook* de Jupyter (Nb) que conté codi PyECC per calcular $I_q(t)$ i tabular-ne uns quants valors.

A.4 Construcció de cossos finits

La creació de cossos (i anells) finits es pot fer com en l'exemple següent.

```
# Creació del cos binari
K = Zn(2)
# Creació del cos F = F8 com a K[X]/(f=X^3+X+1), amb a = X mod f
[F,a] = extension(K,[1,0,1,1],'a','F')
```

Si A és un anell ja construït i $C = [1, a_1, \ldots, a_r]$ és una llista d'elements de A amb primer element 1, PyECC crea l'anell

$$B = A[X]/(f = X^r + a_1X^{r-1} + \cdots + a_{r-1}X + a_r),$$

i ensems fa l'assignació $x = X \bmod f$, amb l'expressió

```
[B,x] = extension(A,C,'x','B')
```

Com ja s'ha dit abans, l'anell de polinomis $P = A[X]$ es pot crear amb l'expressió

```
[P,X] = polynomial_ring(A,'X')
```

Un cop creat P, si $f = X^r + a_1X^{r-1} + \cdots + a_{r-1}X + a_r \in P$, l'expressió

```
[B,x] = extension(A,f,'x','B')
```

té el mateix valor que l'expressió `[B,x] = extension(A,C,'x','B')` d'abans.

A.5 Vectors i matrius

Sigui A un anell i n un número enter positiu. La funció `v = vector(A,n)` crea el vector nul de longitud n amb coeficients de l'anell A. Si $a \in A$ i $0 \le j < n$, l'expressió `v[j] = a` assigna a a la component j-èsima de v, i si a és una llista d'elements de A, `vector(a)` la trasforma en un vector. Anàlogament, l'expressió `M = matrix(A,k,n)` crea la matriu nul·la de k files i n columnes, i es pot assignar l'element $a \in A$ a la casella (i, j) amb l'expressió `M[i,j] = a`. Si $a_1, \ldots, a_k$ són llistes de longitud n d'elements de A, l'expressió `matrix([a1,...,ak])` les transforma en la corresponent matriu $k \times n$.

A.6 Utilitats per al sistema McEliece

Dels ingredients que entren en un sistema de McEliece, hem vist com es pot construir G a la secció 3. A continuació es mostra com podem construir les matrius P i S.

La funció `permutation_matrix(n)` crea una matriu de permutació d'ordre n escollida uniformement. En la implementació s'usen les funcions `ZZ()`, que crea l'anell dels números enters, i `permutation(n)`, que subministra una permutació aleatòria, uniformement distribuïda, del conjunt `N` = $[0, 1, \ldots, n-1]$.

```
def permutation_matrix(n):
    N = list(range(n))
    p = permutation(n)
    P = matrix(ZZ(),n,n)
    for j in range(n):
        P[j,p[j]] = 1
    return P
```

Adonem-nos que l'expressió `P[j,p[j]]=1` posa un 1 a la fila `j`, columna `p[j]`.

La funció `rd_GL(n,F)` crea una matriu de $F(n)$ invertible escollida uniformement. Està definida de la manera següent:

```
def rd_GL(n,F=Zn(2)):
    a = rd_nonzero(F)
    A = matrix([[a]])
    for _ in range(2,n+1):
        A = rd_extend(A)
    return A
```

La part central és la funció `rd_extend(A)`, que crea, donada una matriu invertible $A \in F(k)$, una matriu invertible $B \in F(k+1)$ que segueix la distribució uniforme si A té la mateixa propietat. Com que el valor inicial de A és clarament una selecció uniforme d'una matriu invertible de $F(1)$, la *iteració* produeix una matriu invertible de $F(n)$ distribuïda uniformement.

La definició de `rd_extend(A)` està inspirada en el senzill algorisme *recursiu* de D. Randall publicat a [42]. Primer se selecciona un vector no nul de longitud $k+1$ i es posa r per indicar l'índex de la seva primera component no nul·la. La línia `rd_insert(A,r)` crea primer una matriu de $F(k, k+1)$ inserint una columna aleatòria d'elements de F entre les columnes d'indexs $r-1$ i r de A, després en fa una matriu invertible de $F(k+1)$ afegint una primera fila de 0 amb un 1 en la posició d'índex r, i finalment multiplica el resultat per la matriu obtinguda de la matriu identitat I_{k+1} per substitució de la seva r-èsima fila pel vector v. Tot comptat, a la fi resulta la definició següent:

```
def rd_extend(A):
    k = ncols(A); F = K_(A)
    v = rd_nonzero_vector(F,k+1)
    r = 0
    for j in range(k+1):
        if v[j]!=0:
            r = j; break
    A = rd_insert(A,r)
    x = v[r]
```

```
for j in range(r+1,k+1):
    A[:,j] = A[:,j]+ A[:,r]*v[j]
A[:,r] = x*A[:,r]
return A
```

En lloc de donar aquí una explicació més detallada, que podeu trobar a [46], ens sembla més adient il·lustrar la construcció en el cas de matrius invertibles de $F(2)$. Inicialment tenim un element aleatori no nul $a \in F$ i v pot tenir una de les dues formes següents: $v = [b, y]$ o $v = [0, b]$ ($b, y \in F$, $b \neq 0$). En el primer cas, l'explicació que hem donat es concreta (posant $x \in F$) en l'expressió

$$\begin{pmatrix} 1 & 0 \\ x & a \end{pmatrix} \begin{pmatrix} b & y \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} b & y \\ xb & xy + a \end{pmatrix}.$$

En el segon cas, resulta l'expressió (posant $x \in F$)

$$\begin{pmatrix} 0 & 1 \\ a & x \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & b \end{pmatrix} = \begin{pmatrix} 0 & b \\ a & xb \end{pmatrix}.$$

La primera expressió dona totes les matrius invertibles de $F(2)$, la primera fila de les quals té un element no nul en la primera posició. El nombre de matrius així obtingudes és $m_0 = q^2(q-1)^2$. La segona expressió dona les matrius invertibles de $F(2)$, la primera fila de les quals comença amb 0. El nombre d'aquestes matrius és $m_1 = q(q-1)^2$, i és immediat comprovar que $m_0 + m_1 = (q^2-1)(q^2-q)$, que és el cardinal del grup lineal $\mathrm{GL}(2, F)$. Això prova que els elements d'aquest grup apareixen uniformement quan es generen pel procediment explicat.

A.7 Càlcul de matrius generadores

La funció `blow(H̃,F)` calcula H' i la funció `prune(H')` elimina les files que són combinació lineal de les anteriors, de manera que al final del procés tenim una matriu de control H.

```
n = 7; r = 2
K = Zn(2)
[F,a] = extension(K,[1,0,1,1],'a','F')
R1 = [1,1,1,1,1,1,1]
R2 = [1,a,a**2,a**3,a**4,a**5,a**6]
HH = matrix(F,[R1,R2])
prune(blow(HH,F))
-->
[[1 1 1 1 1 1 1]
 [0 0 1 0 1 1 1]
 [0 1 0 1 1 1 0]
 [1 0 0 1 0 1 1]]
```

Això mostra que la dimensió del codi és $7 - 4 = 3$. Adonem-nos que les components de $1 \in F$ respecte de la base $\{1, a, a^2\}$ de F sobre K són $[1, 0, 0]$, de manera que la matriu `blow(HH,F)`, que és de tipus 6×7, té dues files de zeros i per tant l'efecte de `prune(blow(HH,F))` és simplement eliminar aquestes dues files.

A.8 Exemple de codi de Goppa

```
# Un codi de Goppa [19,12] sobre F5
F5 = Zn(5)
# Creació del cos F25, amb generador x tal que x**2-2=0
[F25,x] = extension(F5,[1,0,-2],'x','F25')
# Creació de l'anell de polinomis F25[T]
[A,T] = polynomial_ring(F25,'T')
# Conjunt dels elements no nuls del cos F25
# [1:] exclou l'element d'índex 0, que és 0>>F25
a = Set(F25)[1:]
# Creació d'un polinomi p
p = T**6 + T**3 + T + 1
# Llista d'elements de a que no són arrels de p
a = [t for t in a if evaluate(p,t)!=0]
n = len(a)
--> 19
# Creació del codi C de Goppa associat a p i a
C = Goppa(p,a)
# La matriu de control de C s'obté amb l'expressió H_(C)
H = blow(H_(C),F5)
n - rank(H)
--> 12
```

En l'exemple anterior, p té sis arrels (2, 2, 3, 4, $a + 2$, $4a + 2$) i, per tant, la longitud del codi és $25 - 1 - 5 = 19$. El rang de H resulta ser 7 i, per tant, $k = 12$.

A.9 Descodificació

L'exemple que segueix és una continuació de l'exemple anterior i il·lustra una descodificació amb PGZ.

```
# Generació d'un vector d'error aleatori de pes 3
e = rd_error_vector(Z5,n,3)
--> e = [0,1,0,0,0,3,0,4,0,0,0,0,0,0,0,0,0,0,0]
# Descodificació de e amb PGZ
PGZ(e,C)
--> PGZ: Error positions [1,5,7], error values [1,3,4]
         [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] :: Vector[Z5]
```

A.10 Entorns Jupyter

Acabem aquesta secció veient un exemple del funcionament de `Jupyter`.[6]

6 `http://jupyter.org/`.

Referint-nos a la figura 2, la primera línia és un comentari que conté el títol del *notebook*. Amb la comanda de la segona línia, `from PyECC import *`, es carrega el paquet PyECC. En la definició de la funció `irr(q,m)`, `divisors` i `mu_moebius` són funcions de PyECC que calculen la llista ordenada de divisors de m i el valor $\mu(m//d)$ de la funció de Möbius per al quocient enter de m per d, quocient que en Python s'expressa $m//d$ (cal remarcar que el valor de l'expressió m/d és un número decimal). El resultat de `show(M)` és equivalent a la taula següent.

```
# Irr_q(t)
from PyECC import *

def irr(q,t):
    N = 0
    D = divisors(t)
    for d in D:
        N += mu_moebius(d)*q**(t//d)
    return N//t

M=[[irr(q,t) for t in range(1,8)] for q in [2,3,4,5,7,8,9]]

show(M)
```

FIGURA 2: Imatge de la primera cel·la d'un *notebook* de Jupyter que mostra la funció irr(q,t) de PyECC que calcula el nombre de polinomis irreductibles mònics de grau t sobre F_q, així com les expressions per obtenir els valors de la taula 1.

1	2	3	4	5	6	7
2	1	2	3	6	9	18
3	3	8	18	48	116	312
4	6	20	60	204	670	2340
5	10	40	150	624	2580	11160
7	21	112	588	3360	19544	117648
8	28	168	1008	6552	43596	299592
9	36	240	1620	11808	88440	683280

TAULA 1: Valors de irr(q,t) per a $1 \le t \le 7$ i $q \le 9$.

B Implementació d'un prototip de sistema de McEliece

Seguirem pas a pas un exemple molt senzill que hauria de ser suficient per il·lustrar els diversos passos del procediment.

```
# 0. Inici. Importem PyECC i definim Z2 amb nom 'K'
from PyECC import *
K = Zn(2,'K')
```

```
# 1. Construïm F32 amb generador a
[F,a] = extension(K,get_irreducible_polynomial(K,5),'a','F')

# 2. Generem aleatòriament un polinomi mònic irreductible
#    de grau 3 de F[T]
p = get_irreducible_polynomial(F,3,'T')
show(p) -->
T**3+(a**4+a**3+a**2+a)*T**2+(a**4+a**3)*T+a**2+a :: F[T]

# 3. Creem el codi de Goppa de polinomi p**2 sobre
#    els elements no nuls de F (expressió X[1:])
#    Corregeix deg(p) = 3 errors
X = Set(F)
C = Goppa(p**2,X[1:])

# 4. Matrius de control/F i /K
HH = H_(C)
H1 = blow(HH,F)
show(shape(HH))
--> (6, 31)
show(shape(H1))
--> (30, 31)
show(rank(H1))
--> 15

# 5. Matriu generadora/K, G. Veiem que k = 16
G = transpose(kernel(H1))
show(shape(G))
--> (16, 31)

# 6. Matriu S de la clau privada; K_(G) és K, ja que C/K.
#    S és una matriu binària 16x16.
S=rd_GL(K_(G),nrows(G))
show(det(S))
--> 1

# 7. Matriu P (entera) de la clau privada. det(P) pot ser 1 o -1
P = permutation_matrix(ncols(G))
show(det(P))
--> 1

# 8. Clau pública (G1,t)
G1 = S*G*P
show(shape(G1))
--> (16, 31)
t = nrows(H_(C))//2
--> 3
```

```
# 9. Encriptació d'un vector aleatori u
u = rd_vector(F,nrows(G1))
--> [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1]
x = u*G1 -->
[0,0,0,0,1,1,1,1,1,1,0,0,0,0,1,0,0,1,0,0,0,0,1,1,0,1,0,0,1,0,1]
e = rd_error_vector(F,len(x),t) -->
[0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0]
x1 = x+e -->
[0,0,0,0,1,1,1,1,1,1,0,1,0,0,1,1,0,1,0,0,0,0,1,1,0,1,0,0,0,0,1]

# 10. Desencriptació. La inversa de P és la seva transposada
y = x1*transpose(P) -->
[0,0,1,1,1,0,1,1,0,0,0,1,0,0,1,1,0,0,1,0,1,1,1,1,0,0,0,0,1,0,0]
x2 = PGZ(y,C) -->    # uSG
[0,0,0,1,1,0,1,1,0,0,0,1,0,0,1,1,0,0,1,0,0,1,1,1,0,1,0,0,1,0,0]
u1 = solve_linear_system(transpose(G),transpose(x2))
u1 = transpose(u1)
--> [1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0]
u1/S
--> [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1]
u
--> [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1]
```

Sigles

AKE	Authenticated Key Exchange[7]
BMS	Berlekamp-Massey-Sugiyama decoding algorithm
CAKE	Code-based Algorithm for Key Encapsulation
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
HQC	Hamming Quasi-Cyclic Codes
ICT	Information and Communications Technology
IEEE	Institute of Electrical and Electronics Engineers
IND-CCA	Indistinguishability under Chosen Ciphertext Attack
IND-CPA	Indistinguishability under Chosen Plaintext Attack
INRIA	Institut National de Recherche en Informatique et en Automatique
ISIT	International Symposium on Information Theory
KEM	Key Encapsulation Mechanism
LDPC	Low-density Parity-check Codes
MDPC	Moderate-density Parity-Check Codes
NIST	National Institute of Standards and Technology
PGZ	Peterson-Gorenstein-Zierler decoding algorithm
RLCE	Random Linear Code Encryption
WCC	Workshop on Codes and Cryptography

7 Vegeu [30] per a molts altres acrònims relacionats.

Agraïments

La Jornada de Teoria de Nombres esmentada més amunt fou un important estímul per als autors, els quals es complauen a agrair la invitació dels organizadors (Ana Río, Josep M. Miret i Jordi Guàrdia) i la benvolença dels participants a escoltar la seva conferència. Gràcies a Rafel Farré i Santiago Molina, coautors dels articles [25] i [35], respectivament, ja que la programació dels algorismes que s'hi tracten va representar un pas important en el desenvolupament de PyECC. El nostre agraïment també al revisor de l'article pels seus suggeriments, que han comportat una millora de la versió original.

Referències

[1] Aragon, N.; Gaborit, P.; Hauteville, A.; Tillich, J.-P. «Improvement of generic attacks on the rank syndrome decoding problem». 2017. https://hal.archi ves-ouvertes.fr/hal-01618464.

[2] Augot, D.; Batina, L.; Bernstein, D. J.; Bos, J.; Buchmann, J.; Castryck, W.; Dunkelman, O.; Güneysu, T.; Gueron, S.; Hülsing, A.; Lange, T.; Mohamed, M. S. E.; Rechberger, C.; Schwabe, P.; Sendrier, N.; Vercauteren, F.; Yang, B. Y. «Initial recommendations of long-term secure post-quantum systems». PQCRYPTO ICT-645622 - Horizon 2020, 2015. https://pqcrypto.eu.org/ docs/initial-recommendations.pdf.

[3] Barbier, M.; Barreto, P. S. L. M. «Key reduction of McEliece's cryptosystem using list decoding». A: Kuleshov, A.; Blinovsky, V. M.; Ephremides, A. (ed.). *International Symposium of Information Theory (ISIT)*. Sant Petersburg: IEEE, 2011, 2657–2661.

[4] Barreto, P. S. L. M.; Gueron, S.; Güneysu, T.; Misoczki, R.; Persichetti, E.; Sendrier, N.; Tillich, J.-P. «CAKE: Code-based algorithm for key encapsulation». A: *Cryptography and Coding*. Cham: Springer, 2017, 207–226. (Lecture Notes in Comput. Sci.; 10655)

[5] Berlekamp, E. R.; McEliece, R. J.; van Tilborg, H. C. A. «On the inherent intractability of certain coding problems». *IEEE Trans. Information Theory*, IT-24 (3) (1978), 384–386.

[6] Bernstein, D. J. «Index of formal scientific papers». https://cr.yp.to/papers.html.

[7] Bernstein, D. J. «Introduction to post-quantum cryptography». A: *Post-Quantum Cryptography*. Berlín: Springer, 2009, 1–14.

[8] Bernstein, D. J. «Is the security of quantum cryptography guaranteed by the laws of physics?». 2017. https://sidechannels.cr.yp.to/qkd/holographic-20160326.pdf.

[9] Bernstein, D. J.; Biasse, J.-F.; Mosca, M. «A low-resource quantum factoring algorithm». A: *Post-Quantum Cryptography*. Cham: Springer, 2017, 330–346. (Lecture Notes in Comput. Sci.; 10346)

[10] Bernstein, D. J.; Buchmann, J.; Dahmen, E. (ed). *Post-Quantum Cryptography*. Berlín: Springer, 2009.

[11] Bernstein, D. J.; Lange, T. «Post-quantum cryptography». *Nature*, 549 (2017), 188–194.

[12] Bernstein, D. J.; Lange, T.; Peters, C. «Attacking and defending the McEliece cryptosystem». A: *Post-Quantum Cryptography*. Berlín: Springer, 2008, 31–46. (Lecture Notes in Comput. Sci.; 5299)

[13] Bernstein, D. J.; Lange, T.; Peters, C. «Wild McEliece incognito». A: *Post-Quantum Cryptography*. Heidelberg: Springer, 2011, 244–254. (Lecture Notes in Comput. Sci.; 7071)

[14] Bernstein, D. J.; Yang, B. Y. «Asymptotically faster quantum algorithms to solve multivariate quadratic equations». 2017. https://cr.yp.to/papers/groverxl-20171215.pdf.

[15] Biswas, B. «Implementational aspects of code-based cryptography». Tesi doctoral. École Polytechnique i INRIA, 2010.

[16] Cameron, P. J. «Notes on cryptography». http://www.maths.qmul.ac.uk/~pjc/notes/crypt.pdf.

[17] Campbell, E. T.; Terhal, B. M.; Vuillot, C. «The steep road towards robust and universal quantum computation». Preprint, 2016. https://arxiv.org/abs/1612.07330.

[18] Chen, L. [*et al.*] «Report on Post-Quantum Cryptography». NISTIR 8105. National Institute of Standards and Technology, U.S. Department of Commerce, 2016. https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8105.pdf.

[19] CSRC (Computer Security Resource Center). «Post-Quantum Cryptography 2018». https://csrc.nist.gov/Projects/Post-Quantum-Cryptography. [Primera conferència de normalització del PQC organitzada pel Centre de Recursos per a la Seguretat Informàtica - NIST]

[20] Delfs, H.; Knebl, H. *Introduction to Cryptography. Principles and Applications*. 2a ed. Berlín: Springer, 2007. (Information Security and Cryptography)

[21] Deneuville, J.-C.; Gaborit, P.; Zémor, G. «Ouroboros: a simple, secure and efficient key exchange protocol based on coding theory». A: *Post-Quantum Cryptography*. Cham: Springer, 2017, 18–34. (Lecture Notes in Comput. Sci.; 10346)

[22] Diffie, W.; Hellman, M. E. «New directions in cryptography». *IEEE Trans. Information Theory*, IT-22 (6) (1976), 644–654.

[23] Dinh, H.; Moore, C.; Russell, A. «McEliece and Niederreiter cryptosystems that resist quantum Fourier sampling attacks». A: *Advances in Cryptology—CRYPTO 2011*. Heidelberg: Springer, 2011, 761–779. (Lecture Notes in Comput. Sci.; 6841)

[24] Engelbert, D.; Overbeck, R.; Schmidt, A. «A summary of McEliece-type cryptosystems and their security». *J. Math. Cryptol.*, 1 (2) (2007), 151–199.

[25] FARRÉ, R.; SAYOLS, N.; XAMBÓ-DESCAMPS, S. «On PGZ decoding of alternant codes». Preprint, 2017. https://arxiv.org/abs/1704.05259.

[26] GABORIT, P. «Shorter keys for code-based cryptography». A: *Proceedings of Workshop on Codes and Cryptography*. França: WCC 2005, 2005, 81–90.

[27] GABORIT, P.; RUATTA, O.; SCHREK, J. «On the complexity of the rank syndrome decoding problem». *IEEE Trans. Inform. Theory*, 62 (2) (2016), 1006–1019.

[28] GOPPA, V. D. «A new class of linear correcting codes». *Problemy Peredači Informacii*, 6 (3) (1970), 24–30. [En rus]

[29] KATZ, J.; LINDELL, Y. *Introduction to Modern Cryptography*. Boca Raton, FL: Chapman & Hall/CRC, 2008. (Chapman & Hall/CRC Cryptography and Network Security)

[30] LEE, J.; PARK, C. S. «An efficient authenticated key exchange protocol with a tight security reduction». 2008. https://eprint.iacr.org/ 2008/345.pdf.

[31] MCELIECE, R. J. *The Theory of Information and Coding: A Mathematical Framework for Communication*. Mass.; Londres; Amsterdam: Addison-Wesley Publishing Co., Reading, 1977. (Encyclopedia of Mathematics and its Applications; 3)

[32] MCELIECE, R. J. «A public-key cryptosystem based on algebraic coding theory». *The Deep Space Network Progress Report* [Jet Propulsion Laboratory]. 42–44 (1978), NASA Code 310.10.67-11, 114–116.

[33] MERKLE, R. C.; HELLMAN, M. E. «Hiding information and signatures in trapdoor knapsacks». *IEEE Trans. Information Theory*, IT-24 (5) (1978), 525–530.

[34] MISOCZKI, R.; TILLICH, J.-P.; SENDRIER, N.; BARRETO, P. S. L. M. «MDPC-McEliece: New McEliece variants from moderate density parity-check codes». 2012. https://eprint.iacr.org/2012/409.pdf.

[35] MOLINA, S.; SAYOLS, N.; XAMBÓ-DESCAMPS, S. «A bootstrap for the number of $\mathbb{F}_{q^r}$-rational points on a curve over $\mathbb{F}_q$». Preprint, 2017. https://arxiv.org/abs/1704.04661.

[36] NIEBUHR, R. «Attacking and defending code-based cryptosystems». Tesi doctoral, Fachbereich Informatik der Technischen Universität Darmstadt, 2012.

[37] NIEDERREITER, H. «Knapsack-type cryptosystems and algebraic coding theory». *Problems Control Inform. Theory/Problemy Upravlen. Teor. Inform.*, 15 (2) (1986), 159–166.

[38] OVERBECK, R.; SENDRIER, N. «Code-based cryptography». A: *Post-Quantum Cryptography*. Berlín: Springer, 2009, 95–145.

[39] PATTERSON, N. J. «The algebraic decoding of Goppa codes». *IEEE Trans. Information Theory*, IT-21 (1975), 203–207.

[40] Persichetti, E. «Improving the efficiency of code-based cryptography». Tesi doctoral. Nova Zelanda: University of Auckland, Department of Mathematics, 2012.

[41] *Post-quantum cryptography*. https://pqcrypto.org/.

[42] Randall, D. «Efficient generation of random nonsingular matrices». Informe tècnic núm. UCB/CSD-91-658. Berkeley (EUA): University of California at Berkeley, 1991.

[43] Repka, M.; Zajac, P. «Overview of the McEliece cryptosystem and its security». *Tatra Mt. Math. Publ.*, 60 (2014), 57–83.

[44] Rivest, R. L.; Shamir, A.; Adleman, L. «A method for obtaining digital signatures and public-key cryptosystems». *Comm. ACM*, 21 (2) (1978), 120–126.

[45] Rué, J.; Xambó, S. «Introducció matemàtica a la computació quàntica». *Butlletí de la Societat Catalana de Matemàtiques*, 28 (2) (2013), 183–231.

[46] Sayols, N.; Xambó-Descamps, S. «PyECC: A Python package for the construction, coding and decoding of error-correcting codes». 2015–2018. https://mat-web.upc.edu/people/sebastia.xambo/PyECC.html.

[47] Sayols, N.; Xambó-Descamps, S. «Alternant codes and the McEliece cryptosystem». Conferència pronunciada a la Jornada de Teoria de Nombres (Universitat de Lleida, 2017). https://mat-web.upc.edu/people/sebastia.xambo/PyECC/s-CryptoLleida-7-10-2017.pdf.

[48] Sendrier, N. «Efficient generation of binary words of given weight». A: *Proceedings of the 5th IMA Conference on Cryptography and Coding*. Londres: Springer-Verlag, 1995, 184–187.

[49] Sendrier, N. «Cryptosystèmes à clé publique basés sur les codes correcteurs d'erreurs». Mémoire d'habilitation à diriger des recherches. París: Université Pierre et Marie Curie, Paris 6; Rocquencourt: Institut National de Recherche en Informatique et Automatique, 2002.

[50] Sendrier, N. «On the use of structured codes in code based cryptography». A: *Coding Theory and Cryptography III. Contactforum*. Koninklijke Vlaamse Academie van België voor Wetenschaeppen en Kunsten, 2009, 59–68.

[51] Shor, P. W. «Algorithms for quantum computation: discrete logarithms and factoring». A: *35th Annual Symposium on Foundations of Computer Science*. Los Alamitos, CA: IEEE Comput. Soc. Press, 1994, 124–134. [Per obtenir una versió revisada i ampliada d'aquest article, vegeu: «Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer». *SIAM J. Comput.*, 26 (5) (1997), 1484–1509]

[52] Tillich, J.-P. «The decoding failure probability of MDPC codes». Preprint, 2018. https://arxiv.org/abs/1801.04668.

[53] Wang, J. «Quantum resistant random linear code based public key encryption scheme RLCE». A: *Proceedings 2016 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2016, 2519–2523.

[54] WANG, J. «Decoding generalized Reed-Solomon codes and its application to RLCE encryption schemes». Preprint, 2017. https://arxiv.org/abs/1702.07737.

[55] «Quantum cryptography». A: Wikipedia. https://en.wikipedia.org/wiki/Quantum_cryptography.

[56] XAMBÓ-DESCAMPS, S. *Block Error-correcting Codes. A Computational Primer*. Berlín: Springer-Verlag, 2003. (Universitext)

NARCÍS SAYOLS
DEPARTAMENT ESAII
UPC, EDIFICI K2M
C. JORDI GIRONA, 1–3
08034 BARCELONA, SPAIN
narcis.sayols@upc.edu

SEBASTIÀ XAMBÓ
DEPARTAMENT DE MATEMÀTIQUES
UPC, EDIFICI OMEGA
C. JORDI GIRONA, 1–3
08034 BARCELONA, SPAIN
sebastia.xambo@upc.edu