QÜESTIIÓ, Vol 20, 3 pp. 505-521, 1996

PRIVACY HOMOMORPHISMS FOR STATISTICAL CONFIDENTIALITY†

JOSEP DOMINGO I FERRER*

Universitat Rovira i Virgili

When publishing contingency tables which contain official statistics, a need to preserve statistical confidentiality arises. Statistical disclosure of individual units must be prevented. There is a wide choice of techniques to achieve this anonymization: cell suppression, cell perturbation, etc. In this paper, we tackle the problem of using anonymized data to compute exact statistics; our approach is based on privacy homomorphisms, which are encryption transformations such that the decryption of a function of ciphers is a (possibly different) function of the corresponding clear messages. A new privacy homomorphism is presented and combined with some anonymization techniques, in order for a classified level to retrieve exact statistics from statistics computed on disclosure-protected data at an unclassified level.

Keywords: Statistical confidentiality, privacy homomorphisms.

† This work is partly supported by the Spanish CICYT under grant no. TIC95-0903-C02-02.

* Estadística i Investigació Operativa, Dep. Enginyeria Química, Universitat Rovira i Virgili, Autovia de Salou s/n, 43006 Tarragona, **e-mail:** jdomingo@etse.urv.es

- Article rebut el juny de 1995.

- Acceptat l'abril de 1996.

1. INTRODUCTION

Statistical confidentiality attempts to keep individual information anonymous when releasing macrodata (contingency tables) and microdata (individual records). Such data can be released by publishing printed reports or following a set of queries to a statistical database. In the first case, off-line protection is enough, whereas in the second case on-line protection is required. While off-line statistical confidentiality techniques seem well developed, on-line techniques do not appear as very promising (see [Adam (1989)] for a survey) even if they have been long studied: as queries are done interactively, one can very often devise an adaptive query strategy to get a particular information out of the database. Another taxonomy of disclosure control methods [Schackis (1993)] is based on whether the data to be protected are macrodata or microdata; for macrodata, available methods are cell suppression, change of the classification scheme, rounding and random perturbation; for microdata, the choice is between data reduction and data modification methods. Rather than following one of the above mentioned references to extensively review all available methods, we prefer to recall here three operating principles underlying, if not all methods, at least a good deal of them:

① ***Random perturbations***. The basic idea is to distort microdata/macrodata by adding a small (often zero-mean) perturbation. This technique suffices for anonymizing off-line contingency tables, and can be improved by performing secondary compensations in order to keep marginal sums unchanged [Appel (1992)], [Turmo (1993)]. For databases which can be queried repeatedly, zero-mean random perturbations are far from secure: the true answer can be derived by computing the average of a sufficiently large number of perturbed answers to the same query. Error inoculation is a possible solution: the perturbation used for a value is not computed each time the record is used for a computation, but is determined by a fixed perturbation factor stored along with the value.

② ***Data suppression or query set size control***. Statistics affecting more than K individual records o less than $N-K$ are not supplied, where N is the total number of records in the database. This strategy is useful for anonymizing off-line contingency tables, where it amounts to eliminating cells with very low or high frequencies (disclosure cells). In this case, secondary suppressions will probably have to be performed; otherwise, marginal sums could be used to determine primary suppressions. It is desirable to minimize the total value of suppressed cells, which can be achieved through the use of linear programming [Cox (1992)]. For on-line statistical databases, data suppression amounts to controlling the query set size (queries having very small or very large query sets are not answered). Unfortunately, this technique is rather ineffective for on-line protection: Schlörer showed long ago that an individual record can be isolated by an iterative technique such as the «tracker» [Schlörer (1979)], even for K near $N/2$.

③ ***Random samples***. In this approach, statistics are not computed on the original data, but ~~on a random~~ sample of them —a random query subset in a database—. This is a rather secure technique, but can only provide estimates of marginal sums. If samples are small, then there is a clear loss of quality. A related alternative is using a re-sampling method, such as bootstrapping (see [Heer (1992)] and subsection 4.3).

Disclosure control methods yield data only usable for consultation and approximate computation at an unclassified level. In this paper, we will show a cryptographic way for a classified level (statistical institute) to take advantage of unclassified (*e. g.* subcontracted) computation on disclosure-protected data. The idea is to extract, with little classified effort, exact statistics from approximate statistics computed on disclosure-protected data at an unclassified level. Future research will be directed to exporting our results to on-line scenarios. In section 2, we introduce privacy homomorphisms (PH), which are the basic tool used in this paper. In section 3, we present a new PH which has better properties than the PHs known so far. In section 4, the use of PHs for multilevel computation on microdata and macrodata is discussed; more specifically, we illustrate the combination of PHs with disclosure control methods based on random perturbation, data suppression and resampling. Section 5 contains some final remarks.

2. PRIVACY HOMOMORPHISMS

Privacy homomorphisms (PHs from now on) were formally introduced in [Rivest (1978b)] as a tool for processing encrypted data. Basically, they are encryption functions $E_k : T \longrightarrow T'$ which allow to perform a set F' of operations on encrypted data (in T') without knowledge of the decryption function D_k. Knowledge of D_k allows to recover the same outcome that would be obtained if the corresponding set F of operations had been used on clear data (in T). The security gain is obvious because classified data can be encrypted, processed by an unclassified computing facility, and the result decrypted by the classified level. Next, we include some simple examples of PHs

Example 1. An exponential cipher such as RSA [Rivest (1978a)] is a PH. Let $m = pq$, where p and q are two large secret primes (about 100 decimal digits each). In this case,

$$T = T' = Z/(m)$$

$$E_k(a) = a^e \bmod m$$

$$D_k(a') = (a')^d \bmod m$$

where $Z/(m)$ is the set of integers modulo m, d is secret and $ed \bmod \phi(m) = 1$, with $\phi(m) = (p-1)(q-1)$ being Euler's totient function. Clearly,

$$D_k(E_k(a)) = a^{ed} \bmod m = a^{1+t\phi(m)} \bmod m = a$$

where Euler's theorem is used in the last step. Now, let $F = F' = \{\star\}$, where $\star$ denotes the modular multiplication over $Z/(m)$. The following homomorphic property holds

$$E_k(a) \star E_k(b) = (a^e \bmod m)(b^e \bmod m) \bmod m = (a \star b)^e \bmod m = E_k(a \star b)$$

This homomorphism allows only one operation, but appears to be very secure. Finding D_k from E_k, *i. e.* finding d from e, seems to be equivalent to factoring a large modulus m —no polynomial-time algorithm for factoring has been published up-to-date—. An additional interesting property relates to the preservation of the equality predicate, because it holds that

$$E_k(a) = E_k(b) \text{ if and only if } a = b$$

□

Example 2. In [Rivest (1978b)], the following PH is given. Let p and q be two large secret primes (100 decimal digits each). Consider the set of cleartext data $T = Z/(m)$ and the set of cleartext operations $F = \{+_m, -_m, \times_m\}$ consisting, respectively, of the addition, subtraction and multiplication modulo m, with $m = pq$. Define the encryption key $k = (p,q)$ and $E_k(a) = [a \bmod p, a \bmod q]$. Now, given k, and given $[b,c] \in Z/(p) \times Z/(q)$, computation of $D_k([b,c])$ is as follows: decryption consists of finding an $x \in Z/(m)$ such that

$$b = x \bmod p \quad \text{and} \quad c = x \bmod q \tag{1}$$

To compute x, we have from the first equation 1 that

$$x = b + pt$$

for some t. Then, substitution in the second equation 1 yields

$$b + pt \bmod q = c$$

Now, if p^{-1} is the multiplicative inverse of p modulo q —Euclid's extended algorithm can be used to invert p over $Z/(q)$—, we have

$$t = p^{-1}(c-b) \bmod q = d + qr$$

for some r. Finally

$$x = b + pd + (pq)r = b + pd \bmod m$$

The Chinese remainder theorem guarantees the uniqueness of x.

Thus, the set of ciphertext data is $T' = Z/(p) \times Z/(q)$, and the set F' of ciphertext operations consists of the componentwise modular addition, subtraction and multiplication. Since $k = (p,q)$ is only known at a classified level, unclassified operations on encrypted data are actually carried out over $Z \times Z$ (reduction over $Z/(p) \times Z/(q)$ being restricted to the classified decryptor). This fact implies that, for a given number in $Z/(m)$, there is an infinity of different enciphered versions; therefore, test for equality is not possible at an unclassified level. Also, the authors of [Rivest (1978b)] do not consider division as a valid operation. Whereas there is no problem in including the componentwise modular division in F', this operation cannot be formally included in F, because $Z/(m)$ is not a field. However, the only additional requirement for division to be possible is that the divisor be relatively prime to m. This is not a very restrictive condition for statistical data processing, because there are $\phi(m) = pq - p - q + 1$ numbers relatively prime to m in $Z/(m)$; if p and q are 100-digit primes, then $\phi(m)$ and m have about the same order of magnitude. The probability of getting a divisor not relatively prime to m is

$$\frac{m - \phi(m)}{m} = \frac{p + q - 1}{pq} \approx O(10^{-100})$$

So, dividing at the unclassified level involves finding the multiplicative inverse of the divisor over $Z/(m) \times Z/(m)$ (almost always possible), and then multiplying. The resulting quotient can be reduced by the classified level into $Z/(p) \times Z/(q)$, and then decrypted to get a quotient in $Z/(m)$. However, *this quotient coincides with the standard quotient over Z only if the division is exact* (with remainder 0). Thus it is better to leave divisions in rational form (as fractions).

While this homomorphism allows more operations than the one in example 1, it is less secure. Brickell and Yacobi [Brickell (1988)] have shown that it can be broken by a known cleartext attack. If the ciphertexts $[b_i, c_i]$ corresponding to some cleartexts a_i, for $1 \leq i \leq r$, are known (the ciphertexts being nontrivial, *i. e.* $b_i \neq a_i$ or $c_i \neq a_i$), then p and q can be found with a high probability, and any ciphertext decrypted. The idea is that if $p' = \gcd\{a_i - b_i : 1 \leq i \leq r\}$ and $q' = \gcd\{a_i - c_i : 1 \leq i \leq r\}$, then $p | p'$ and $q | q'$. There is a high probability that $p = p'$ and $q = q'$, even for small r; anyway, as r increases, so does the probability of finding p and q. Still, this homomorphism can safely be used as long as no corresponding nontrivial ciphertext and cleartext are released.

□

3. A NEW PRIVACY HOMOMORPHISM

We propose in this section a new privacy homomorphism which is similar to the one of example 2 (it has the same sets T, T', F and F'), but entails two significant improvements

- Small values are nontrivially encrypted.
- The new PH is able to withstand a general known-cleartext attack, and specifically the [Brickell (1988)] attack.

3.1. The basic idea

When p, q, $m = pq$ are very large integers, a small value a is very likely to have the same representation over $Z/(m)$, $Z/(p)$ and $Z/(q)$, that is

$$a \bmod m = a \bmod p = a \bmod q \quad \text{if } a < \min(p,q)$$

This is an undesirable feature, because the homomorphism of example 2 leaves the cleartext unencrypted (trivial ciphertext). A possible solution is to multiply a by a pair of secret constants r_p and r_q such that $r_p < p$ and $r_q < q$ (the encryption key is now extended to $k = (p,q,r_p,r_q)$). A further improvement to deter ciphertext-only attacks based on frequency analysis is to secretly and randomly split a into $a_{.1},\cdots,a_{.n}$, such that $a_{.j} \in Z/(m)$ and $\sum_{j=1}^{n} a_{.j} \bmod m = a$. Thus the privacy homomorphism we propose is

Public parameters n, m (actually m can be made secret, to increase security).

Secret key p, q large primes, such that $pq = m$. Also, $r_p \in Z/(p)$, such that it generates a large multiplicative subgroup in $Z/(p) - \{0\}$. Also, r_q with similar properties with respect to $Z/(q)$.

Encryption Randomly split $a \in Z/(m)$ into secret $a_{.1},\cdots,a_{.n}$ such that

$$a = \sum_{j=1}^{n} a_{.j} \bmod m \quad \text{and} \quad a_{.j} \in Z/(m).$$

Compute

$$E_k(a) = ([a_{.1}r_p \bmod p, a_{.1}r_q \bmod q], [a_{.2}r_p^2 \bmod p, a_{.2}r_q^2 \bmod q], \cdots \quad (2)$$
$$\cdots, [a_{.n}r_p^n \bmod p, a_{.n}r_q^n \bmod q])$$

Decryption Compute the scalar product of the j-th [mod p, mod q] pair by $[r_p^{-j} \bmod p, r_q^{-j} \bmod q]$ to retrieve the $[a_{.j} \bmod p, a_{.j} \bmod q]$. Add up to get $[a \bmod p, a \bmod q]$. Use the Chinese remainder theorem to get $a \bmod m$.

3.2. The operations on encrypted data

As operations on encrypted values are carried out over $Z \times Z$ by the unclassified level, the use of r_p and r_q requires that the terms of the encrypted value having different r-degree be handled separately —the r-degree of a mod p, resp. mod q, term is the exponent of the power of r_p, resp. r_q, contained in the term—. This is necessary for the classified level to be able to multiply each term by r_p^{-1} (inverse of r_p over $Z/(p)$) and r_q^{-1} (inverse of r_q over $Z/(q)$) the right number of times, before adding all terms up, reducing the final result into $Z/(p) \times Z/(q)$, and decrypting into $Z/(m)$.

The only operation altering the r-degree is multiplication. If cleartext data x and y have been encrypted as $E_k(x)$ and $E_k(y)$, with r-degrees n_1 and n_2, then the product $E_k(x)E_k(y)$ has r-degree $n = n_1 + n_2$. The result may have terms $E_{k,j}(z)$ with degrees ranging from $j = 1$ to n and can be represented in vector notation as

$$(E_{k,1}(x), E_{k,2}(x), \cdots, E_{k,n}(x))$$

If we set $r = [r_p, r_q]$ then $E_k(x)$ can also be written as a polynomial of r

$$E_k(x)[r] = t_1 r + \cdots + t_n r^n$$

Although the coefficients t_j are in practice unknown to the unclassified level, the polynomial notation is useful to understand how algebraic operations should be carried out by this level using terms rather than coefficients

Addition and subtraction In vector notation, they are done componentwise over Z, which in polynomial notation means adding terms with the same degree.

Multiplication It works like in the case of polynomials: all terms are cross-multiplied in Z, with an j_1-th degree term by a j_2-th degree term yielding a $j_1 + j_2$-th degree term; finally, terms having the same degree are added up.

Division Cannot be carried out in general because the polynomials are a ring, but not a field. A good solution is to leave divisions in rational format by considering the field of rational functions, *i. e.* fractions whose numerator and denominator are polynomials. In this way, if a and b are two integers, we encrypt a/b as $\frac{E_k(a)}{E_k(b)}$.

Note. When addition or subtraction are performed on fractions with different denominators, numerators cannot be added or subtracted directly. The rules for ordinary fractions should be followed

$$\frac{E_k(a)}{E_k(b)} \pm \frac{E_k(c)}{E_k(d)} = \frac{E_k(a)E_k(d) \pm E_k(b)E_k(c)}{E_k(b)E_k(d)}$$

3.3. Security

We shall next quote two security results related to the proposed PH; for a detailed security analysis of the new PH from the cryptographic point of view, refer to [Domingo (1996)]. The first result is that our proposal appears to withstand any known-cleartext attack. In other words, even if an enemy has access to some *random* cleartexts and corresponding ciphertexts, she is not able to retrieve the secret key $k = (p, q, r_p, r_q)$ that would allow her to decipher arbitrary ciphertext data. A second result relates to plaintext splitting, which turns out to be essential to the security of the PH. This means that one should take $n \geq 2$. Without plaintext splitting ($n = 1$), the PH is shown to be insecure and can be broken by a known-plaintext attack, specifically an extended Brickell-Yacobi gcd attack.

4. MULTILEVEL COMPUTATION USING PRIVACY HOMOMORPHISMS

PHs enable multilevel processing of sensitive data. The idea is to have just a «small core» of resources for classified tasks at a statistical office. PHs make possible to subcontract external computing facilities without compromising statistical secrecy —external service providers being special instances of unclassified levels.

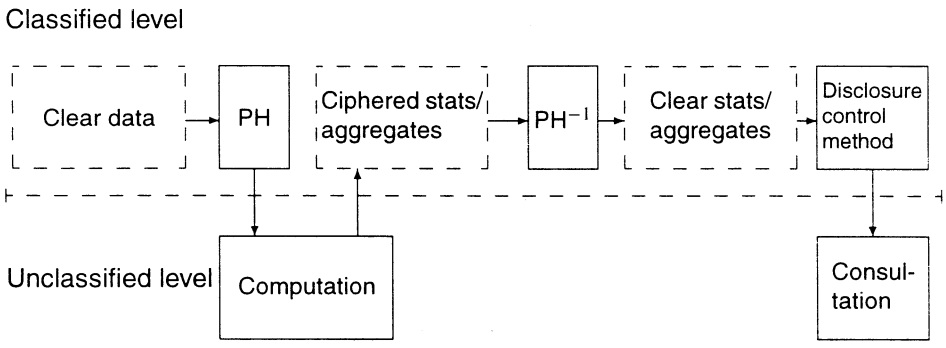

Figure 1.

Multilevel computation (scenario A).

At least two scenarios for using PHs are conceivable. In scenario A (see 1), sensitive micro/macrodata can be encrypted under a PH at a classified level and then be forwarded to an unclassified facility for computation of encrypted statistics or aggregate data; such results can later be decrypted at the classified level. Scenario A is quite straightforward and needs no further discussion. On the other hand, scenario B makes use of PHs in more subtle ways (see figure 2). After using a disclosure con-

trol method to protect clear exact micro/macrodata, some information generated by the method can be encrypted and forwarded to the unclassified facility. This facility takes such encrypted information and the disclosure-protected data as inputs to its computation. The outputs are forwarded to the classified level: this level *alone* can extract (with little effort) exact results from the outputs received from the unclassified level. Scenario B is the most relevant to this paper and will be developed in the following subsections for disclosure control methods based on random perturbation, data suppression and resampling.

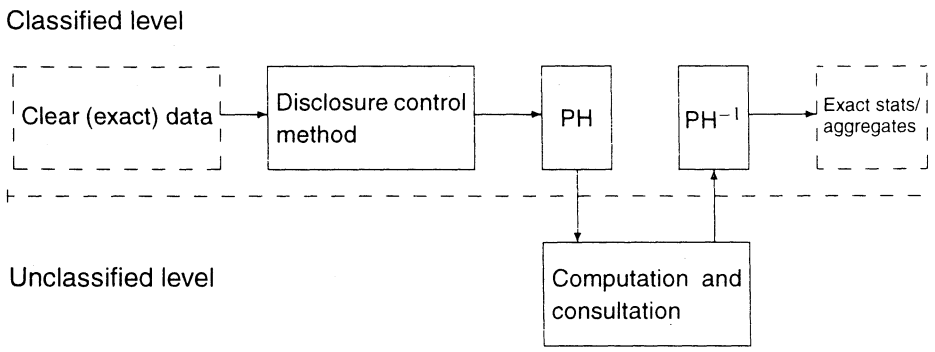

Figure 2.

Multilevel computation (scenario B).

4.1. Multilevel processing of randomly perturbed data

Assume that, at a classified level, a statistical office uses a random perturbation method on data $x_1, x_2, \cdots, x_n$ (which may be microdata or frequencies in a contingency table). This gives

$$x_i^* = x_i + \varepsilon_i \quad \forall i = 1, \cdots, n$$

where ε_i is a random value. Let E_k be the encrypting transformation of a PH with a set F of cleartext operations and a corresponding set of F' of ciphertext operations. Now, the classified level releases the pairs $(x_i^*, E_k(-\varepsilon_i))$ to the unclassified level for further processing. This level is able to perform on the x_i^* the operations in F and compute the encrypted perturbation of the result by using the operations in F' on the $E_k(-\varepsilon_i)$.

At any time, the classified level can take advantage of computations performed by the unclassified level on perturbed data, since restoration of the exact result involves

only decrypting the perturbation of the result. Remark that the unclassified level must content itself with perturbed statistics (on perturbed data), unless the classified level *decides* to reveal the clear perturbation corresponding to some perturbed statistic.

The homomorphism of section 3 can be used to have the unclassified level compute the encrypted perturbations corresponding to any kind of perturbed statistics. It is useful here to consider non-integer perturbations and non-integer data. Remark that integer perturbations are often too coarse, since in most applications perturbed statistics are required to reflect original statistics to some extent. So, if perturbations have at most u decimal positions, we multiply them by 10^u. In this case, a perturbation ε_i is encrypted as $\frac{E_k(\varepsilon_i \times 10^u)}{10^u}$; similarly, a perturbed value x_i^* is represented as $\frac{x_i^* \times 10^u}{10^u}$. Remark that, being public, the denominator 10^u need not be encrypted. Combining cleartext values with encrypted values in computations poses no arithmetical problems, as will be shown below. In this way, without loss of generality, we can assume in what follows that perturbations ε_i and perturbed values x_i^* have been converted to integers (the numerators of their corresponding fractions).

4.1.1. Encrypted perturbation algebra

Table 1 summarizes the computations on encrypted perturbations generated by elementary operations. Given that $x = x^* - \varepsilon_x$ and $y = y^* - \varepsilon_y$, deriving the clear perturbations for addition, subtraction and multiplication is straightforward. Division is not explicitly considered, because it follows from subsection 3.2 that it can be avoided if rational numbers are represented and handled as fractions.

Table 1

Encrypted perturbations corresponding to elementary operations

Perturbed operation	Clear perturbation	Encrypted perturbation
$x^* + y^*$	$-(\varepsilon_x + \varepsilon_y)$	$E_k(-\varepsilon_x) + E_k(-\varepsilon_y)$
$x^* - y^*$	$-(\varepsilon_x - \varepsilon_y)$	$E_k(-\varepsilon_x) - E_k(-\varepsilon_y)$
$x^* y^*$	$-x^*\varepsilon_y - y^*\varepsilon_x + \varepsilon_x\varepsilon_y$	$x^* E_k(-\varepsilon_y) + y^* E_k(-\varepsilon_x) + E_k(-\varepsilon_x)E_k(-\varepsilon_y)$

The new homomorphism has two remarkable properties which justify the formulae for encrypted perturbations in table 1.

- The fact that it is based on finite fields greatly facilitates computation at the unclassified level, which can operate over $Z \times Z$. Then the classified level, which knows

the underlying $Z/(p) \times Z/(q)$ vector space, is able to perform a suitable reduction on the result received from the unclassified level.

- Thanks to the previous property and to cleartext addition and multiplication being mapped to ciphertext addition and multiplication, respectively, the unclassified level can compute $E_k(-x^*\varepsilon_y - y^*\varepsilon_x)$ as $x^*E_k(-\varepsilon_y) + y^*E_k(-\varepsilon_x)$ over Z. This greatly facilitates unclassified computation of the encrypted perturbations.

4.1.2. Undoing the integer conversion

When the classified level receives the result of a computation from the unclassified level, it retrieves the clear perturbation and adds it to the perturbed result to obtain the exact result. If initial data and perturbations were converted to integers as discussed above, every result received from the unclassified level is a fraction; the numerator of the perturbation must be decrypted and thereafter divided over the real numbers by the decrypted denominator (be it a power of 10 or not), in order to get the right number of decimal positions.

4.1.3. Numerical examples

We next give two examples to illustrate computations with the proposed PH.

Example 3. (Multiplication) The following example is unrealistic because of the small size of p and q, and also because the amount of computation at the unclassified level is smaller than at the classified level, even if the latter could precompute the encrypted perturbations. Actually, the overhead introduced to perform a single multiplication is rather comical. For brevity and clarity, take $n = 2$, that is, perturbations are split into two parts during encryption.

Classified level

Let $p = 17$, $q = 13$, $r_p = 2$ and $r_q = 3$ be the secret key. One wants to have $x = -0.5$ and $y = 0.6$ multiplied by the unclassified level. Then random perturbations $\varepsilon_x = 0.3$ and $\varepsilon_y = -0.1$ are generated and the perturbed values $x^* = x + \varepsilon_x = -0.2$ and $y^* = y + \varepsilon_y = 0.5$ are obtained. In order to suppress decimal positions, perturbed data and perturbations are multiplied by 10, thus yielding the fractions

$$(x^*, -\varepsilon_x) = \left(\frac{\tilde{x}^*}{10}, \frac{-\tilde{\varepsilon}_x}{10}\right) = \left(\frac{-2}{10}, \frac{-3}{10}\right)$$

$$(y^*, -\varepsilon_y) = \left(\frac{\tilde{y}^*}{10}, \frac{-\tilde{\varepsilon}_y}{10}\right) = \left(\frac{5}{10}, \frac{1}{10}\right)$$

Numerators of perturbations are secretly and randomly split and then transformed according to the proposed PH, thus obtaining first and second r-degree terms

$$E_k(-\tilde{\varepsilon}_x) = E_k(-3) = E_k(1,-4) = ([2,3],[1,3])$$

$$E_k(-\tilde{\varepsilon}_y) = E_k(1) = E_k(-1,2) = ([15,10],[8,5])$$

Next, these encrypted perturbation numerators and perturbed data numerators $\tilde{x}^*, \tilde{y}^*$ are forwarded to the unclassified level, with a denominator 10 in all cases. Remark that encrypted perturbations could have been precomputed and/or reused from previous data.

Unclassified level

Compute $\tilde{x}^*\tilde{y}^* = -2 \times 5 = -10$ and the corresponding encrypted perturbation after table 1

$$\tilde{x}^* E_k(-\tilde{\varepsilon}_y) + \tilde{y}^* E_k(-\tilde{\varepsilon}_x) + E_k(-\tilde{\varepsilon}_x) E_k(-\tilde{\varepsilon}_y)$$

$$= -2 \times ([15,10],[8,5]) + 5 \times ([2,3],[1,3]) + ([2,3],[1,3]) \times ([15,10],[8,5])$$

$$= ([-2 \times 15 + 5 \times 2, -2 \times 10 + 5 \times 3],$$

$$[-2 \times 8 + 5 \times 1, -2 \times 5 + 5 \times 3] + [2 \times 15, 3 \times 10],$$

$$[2 \times 8 + 1 \times 15, 3 \times 5 + 3 \times 10], [1 \times 8, 3 \times 5])$$

$$= ([-20,-5],[19,35],[31,45],[8,15])$$

Notice that terms with different r-degree are dealt with as separate components. Return to the classified level with a denominator $10 \times 10 = 10^2$: i) the perturbed product; ii) its encrypted perturbation.

Classified level

Retrieve the clear perturbation $\tilde{\varepsilon}_{xy}$ of the product by computing

$$([-20 \times r_p^{-1} \bmod p, -5 \times r_q^{-1} \bmod q], [19 \times r_p^{-2} \bmod p, 35 \times r_q^{-2} \bmod q],$$

$$[31 \times r_p^{-3} \bmod p, 45 \times r_q^{-3} \bmod q], [8 \times r_p^{-4} \bmod p, 15 \times r_q^{-4} \bmod q])$$

$$= ([-20 \times 9 \bmod 17, -5 \times 9 \bmod 13], [19 \times 9^2 \bmod 17, 35 \times 9^2 \bmod 13],$$

$$[31 \times 9^3 \bmod 17, 45 \times 9^3 \bmod 13], [8 \times 9^4 \bmod 17, 15 \times 9^4 \bmod 13])$$

$$= ([7,7],[9,1],[6,6],[9,5]) = ([14,6])$$

where, in the last step, all terms have been added up over $\mathbb{Z}/(p) \times \mathbb{Z}/(q)$. Bearing in mind that $m = pq = 221$, use the Chinese remainder theorem on the pair $[14,6]$ to recover the perturbation $-\tilde{\varepsilon}_{xy} = 201 \bmod 221 = -20$ corresponding to $\tilde{x}^*\tilde{y}^*$. In the last equivalence, it has been implicity assumed that the upper half of $\mathbb{Z}/(221)$ represents negative integers: this assumption is valid if perturbations have an

absolute value much smaller than pq, which always occurs in real cases because pq is very large. Adding this perturbation to the perturbed product yields

$$\tilde{x}^*\tilde{y}^* - \tilde{\varepsilon}_{xy} = -10 - 20 = -30$$

Finally, divide -30 by the denominator 10^2 returned by the unclassified level. Thus the final result is $xy = -0.3$.

□

Example 4. (Average computation) This example is even more unrealistic as the previous one, but it illustrates a basic statistical operation such as an average computation —averages are the building blocks of most statistics, such as variance, skewness, kurtosis, etc. An average consists of an addition and a division. Again, take $n = 2$ (perturbations are split into two parts).

Classified level

Let $p = 17$, $q = 13$, $r_p = 2$ and $r_q = 3$ be the secret key. One wants to have the average of the sample $(x_1, x_2, x_3) = (0.3, 1.5, 1.0)$ computed by the unclassified level. Then three perturbations are generated to obtain the perturbed values (0.4, -0.1) for x_1, (1.2, 0.3) for x_2 and (0.9, 0.1) for x_3. In order to suppress decimal positions, these perturbed data are multiplied by 10, thus yielding the fractions

$$(x_1^*, -\varepsilon_1) = \left(\frac{\tilde{x}_1^*}{10}, \frac{-\tilde{\varepsilon}_1}{10}\right) = \left(\frac{4}{10}, \frac{-1}{10}\right)$$

$$(x_1^*, -\varepsilon_2) = \left(\frac{\tilde{x}_2^*}{10}, \frac{-\tilde{\varepsilon}_2}{10}\right) = \left(\frac{12}{10}, \frac{3}{10}\right)$$

$$(x_3^*, -\varepsilon_3) = \left(\frac{\tilde{x}_3^*}{10}, \frac{-\tilde{\varepsilon}_3}{10}\right) = \left(\frac{9}{10}, \frac{1}{10}\right)$$

Numerators of perturbations are secretly and randomly split and then transformed according to the new PH, thus obtaining

$$E_k(-\tilde{\varepsilon}_1) = E_k(-1) = E_k(1, -2) = ([2,3],[9,8])$$

$$E_k(-\tilde{\varepsilon}_2) = E_k(3) = E_k(-1,4) = ([15,10],[16,10])$$

$$E_k(-\tilde{\varepsilon}_3) = E_k(1) = E_k(-1,2) = ([15,10],[8,5])$$

Perturbed data and encrypted perturbation numerators are forwarded to the unclassified level, along with their denominators (10 in all cases). Remark that encrypted perturbations could have been precomputed or reused from a previous sample.

Unclassified level

Compute the perturbed average $\bar{x}^* = \frac{(\sum_{i=1}^{3} \tilde{x}_i^*)/10}{3} = \frac{(4+12+9)/10}{3} = \frac{25}{30}$. The encrypted perturbation of the numerator 25 of the average can be found by directly adding the numerators of encrypted perturbations (see table 1), because the denominator is 10 in all of them

$$\sum_{i=1}^{3} E_k(-\tilde{\varepsilon}_i)) = ([2,3],[9,8]) + ([15,10],[16,10]) + ([15,10],[8,5])$$

$$= ([2+15+15, 3+10+10], [9+16+8, 8+10+5]) = ([32,23],[33,23])$$

The denominator 30 of the average has perturbation 0, as it is the product of exact values 10 and 3. Return to the classified level: i) the perturbed average $\bar{x}^* = \frac{25}{30}$; ii) for the numerator 25, return the encrypted perturbation $([32,23],[33,23])$; iii) for the denominator 30, return perturbation 0.

Classified level

Retrieve the clear perturbation for the numerator of the average by computing

$$([32 \times r_p^{-1} \bmod p, 23 \times r_q^{-1} \bmod q], [33 \times r_p^{-2} \bmod p, 23 \times r_q^{-2} \bmod q])$$

$$= ([32 \times 9 \bmod 17, 23 \times 9 \bmod 13], [33 \times 9^2 \bmod 17, 23 \times 9^2 \bmod 13])$$

$$= ([16,12],[4,4]) = ([3,3])$$

where, in the last step, the first and the second r-degree terms have been added over $Z/(p) \times Z/(q)$. Bearing in mind that $m = pq = 221$, use the Chinese remainder theorem on the pair $[3,3]$ to recover the clear perturbation 3. The perturbation is taken as positive, because it belongs to the lower half of $Z/(221)$. Adding the perturbation 3 to the numerator of the perturbed average and the perturbation 0 to its denominator yields

$$\bar{x} = \frac{25+3}{30+0} = \frac{28}{30} (= 0.9\hat{3})$$

Remark that, in this last step, the amount of computation *is fixed and does not depend on the sample size.*

□

4.2. Multilevel processing with data suppression

The following disclosure-protected table is taken from [Cox (1992)]. Disclosure —sensitive— cells have been suppressed (primary suppressions); other cells have been secondarily suppressed to prevent inferences. A D-cell stands for a suppressed cell.

D	10	D	D	20	80
D	10	D	5	15	60
40	10	D	D	10	90
5	5	D	D	5	40
75	35	65	45	50	270

Now, an alternative approach to cell suppression is to use a PH to encrypt the D-cell values

$E_k(10)$	10	$E_k(25)$	$E_k(15)$	20	80
$E_k(20)$	10	$E_k(10)$	5	15	60
40	10	$E_k(20)$	$E_k(10)$	10	90
5	5	$E_k(10)$	$E_k(15)$	5	40
75	35	65	45	50	270

The unclassified level views the encrypted cells as if they had been suppressed, but it can perform on them the operations in F'. The classified level can take advantage of this work by decrypting the result. When using a PH breakable by a known-cleartext attack (such as the one in example 2), the unclassified level must separately operate on encrypted cells (operations in F') and unencrypted cells (operations in F); merging the results of both computational streams is up to the classified level. The use of a PH resistant to a known-cleartext attack (such as the ones in example 1 and section 3) allows the classified level to distribute the encrypted version of the whole table, along with the non-disclosure cells as cleartext for consultation purposes; thus the unclassified level can operate on the whole encrypted table using the operations in F', so that the only job left to the classified level is decryption of the final result.

4.3. Multilevel processing of resampled data

The basic resampling scheme dealt with in this subsection is from [Heer (1992)]. Assume that microdata $z_1, \cdots, z_n$ are aggregated to elaborate macrodata in the form of a contingency table x with I rows and J columns, which is produced according to certain specifications. Let x_{ij} be the original frequency in the i-th row and j-th column. In order to produce an anonymized table x^*, a bootstrap sample $z_1^*, \cdots, z_n^*$ is obtained by drawing from the original data $z_1, \cdots, z_n$, n times and with replacement. The bootstrap table x^* thus obtained is an estimation of the original table x and does not allow to get any precise information of x, due to its random error.

The main features of a bootstrap table are

- The overall frequency is preserved, since $\sum x_{ij} = \sum x^*_{ij} = n$.
- Each individual bootstrap frequency x^*_{ij} is a value drawn from a variable having a binomial distribution with mean x_{ij} and variance $x_{ij}(1 - x_{ij}/n)$. Therefore, x^* is an unbiased estimation of x.
- An original frequency $x_{ij} = 0$ is preserved, *i. e.* $x_{ij} = 0$ implies $x^*_{ij} = 0$. If this is undesirable, then a compensated perturbation method could be used on the original table before bootstrapping, in order to replace zero frequencies with small frequencies.

It can be seen from the previous paragraphs that x^* is actually a perturbed image of x, *i. e*

$$x^* = x + (x^* - x) = x + \varepsilon$$

where $\varepsilon = (\varepsilon_{ij})$ is a matrix of random zero-mean estimation errors. Therefore, privacy homomorphisms can be used in a way analogous to the one described in subsection 4.1. The classified level computes the matrix ε and releases the following pairs for unclassified processing

$$(x^*_{ij}, E_k(-\varepsilon_{ij})) \quad \forall i = 1, \cdots, I, \forall j = 1, \cdots, J$$

5. CONCLUSION

The use of privacy homomorphisms for multilevel processing of classified statistical data has been motivated. A new privacy homomorphism has been presented which exhibits good cryptographic and algebraic properties for statistical computation. It has been shown how to combine privacy homomorphisms and several well-known techniques for statistical confidentiality, so that the classified level can recover exact statistics from statistics obtained at an unclassified level on disclosure-protected data. Application of PHs for providing statistical confidentiality in on-line scenarios is currently being investigated. The key issue is that PHs useful for statistical data protection are not necessarily cryptographically unbreakable PHs: they only need be as strong as the statistical confidentiality techniques with which they are combined.

ACKNOWLEDGMENT

Thanks go to two anonymous referees for helpful suggestions that contributed to improve the presentation of this work.

REFERENCES

[1] **N.R. Adam** and **J.C. Wortmann** (1989). «Security-control methods for statistical databases: a comparative study». *ACM Computing Surveys*, **21, 4**, Dec. 1989, 515–556.

[2] **G. Appel** and **D.J. Hoffman** (1992). «Perturbation by compensation». Preproceedings of the *International Seminar on Statistical Confidentiality*, ISI-Eurostat, Dublin, Sep. 1992. Final version in the Proceedings published by Eurostat in 1993.

[3] **E. Brickell** and **Y. Yacobi** (1988). «On privacy homomorphisms», in *Advances in Cryptology-Eurocrypt'87*, eds. W. L. Price and D. Chaum, Springer-Verlag, 117–125.

[4] **L.H. Cox** (1992). «Solving confidentiality protection problems in tabulations using network optimization: a model for cell suppression in U. S. economic censuses». Preproceedings of the *International Seminar on Statistical Confidentiality*, ISI-Eurostat, Dublin, Sep. 1992. Final version in the Proceedings published by Eurostat in 1993.

[5] **J. Domingo** (1996). «A new privacy homomorphism and applications», working paper, Universitat Rovira i Virgili.

[6] **D. Schackis** (1993). *Manual on Disclosure Control Methods*, internal document, Eurostat (unit D3 - Research and development and statistical methods).

[7] **G. Heer** (1992). «A bootstrap procedure to preserve statistical confidentiality in contingency tables». Preproceedings of the *International Seminar on Statistical Confidentiality*, ISI-Eurostat, Dublin, Sep. 1992. Final version in the Proceedings published by Eurostat in 1993.

[8] **R.L. Rivest, A. Shamir** and **L. Adleman** (1978a). «A method for obtaining digital signatures and public-key cryptosystems». *Communications of the ACM*, **21**, Feb. 1978, pp. 120–126.

[9] **R.L. Rivest, L. Adleman** and **M.L. Dertouzos** (1978b). «On data banks and privacy homomorphisms», in *Foundations of Secure Computation*, eds. R. A. DeMillo et al., Academic Press, 169–179.

[10] **J. Schlörer** (1979). *Disclosure from Statistical Databases: Quantitative Aspects of Trackers*, Institut für Medizinische Statistik und Dokumentation, Universität Giessen, Mar. 1979.

[11] **J. Turmo** (1993). «Pertorbacions aleatòries amb compensació: una tècnica per a la protecció d'informació estadística confidencial». *Qüestiió*, **17, 3**, 413–435.