Package 'HomomorphicEncryption'

January 9, 2024

Title BFV, BGV, CKKS Schema for Fully Homomorphic Encryption

Version 0.9.0

Description Implements the Brakerski-Fan-Vercauteren (BFV, 2012) <https://eprint.iacr.org/2012/144>, Brakerski-Gentry-Vaikuntanathan (BGV, 2014) <doi:10.1145/2633600>, and Cheon-Kim-Kim-Song (CKKS, 2016) <https://eprint.iacr.org/2016/421.pdf> schema for Fully Homomorphic Encryption. The included vignettes demonstrate the encryption procedures.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Depends polynom, stats, HEtools

Suggests covr, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Bastiaan Quast [aut, cre] (<https://orcid.org/0000-0002-2951-3577>)

Maintainer Bastiaan Quast <bquast@gmail.com>

Repository CRAN

Date/Publication 2024-01-09 13:40:05 UTC

R topics documented:

`BFV_encrypt` *BFV encryption*

Description

BFV encryption

Usage

```
BFV_encrypt(m, pk)
```

Arguments

`m`	message
`pk`	public key

Value

polynomial

BFV_KeyGen *Brakerski / Fan-Vercauteren*

Description

Brakerski / Fan-Vercauteren

Usage

```
BFV_KeyGen(d = 4, q = 424242)
```

Arguments

d	the d input
q	the q input

Value

polynomial

Examples

```
d  =      4
n  =      2^d
p  =     (n/2)-1
q  = 424242
GenPolyMod(n)
```

compute_basis_coordinates *compute basis coordinates*

Description

Compute basis coordinates

Usage

```
compute_basis_coordinates(sigma_R_basis, z)
```

Arguments

sigma_R_basis	sigma_R_basis
z	z

Value

basis coordinates

coordinate_wise_random_rounding *coordinate-wise random rounding*

Description

Coordinate-wise random rounding

Usage

```
coordinate_wise_random_rounding(coordinates)
```

Arguments

coordinates	coordinates

Value

rounded coordinates

decode *decode*

Description

Decode

Usage

```
decode(xi, M, scale, p)
```

Arguments

xi	xi
M	M
scale	scale
p	p

Value

decoded xi

encode *encode*

Description

Encode

Usage

```
encode(xi, M, scale, z)
```

Arguments

xi	xi
M	M
scale	scale
z	z

Value

encode polynomial

EncryptPoly0 *Encrypt Polynomial Message Part 0*

Description

Encrypt Polynomial Message Part 0

Usage

```
EncryptPoly0(m, pk0, u, e1, p, pm, q)
```

Arguments

m	message
pk0	public key part 0
u	u
e1	e1
p	p
pm	pm
q	q

Value

polynomial which contains the message in ciphertext

EncryptPoly1 *Encrypt Polynomial Message Part 1*

Description

Encrypt Polynomial Message Part 1

Usage

```
EncryptPoly1(pk1, u, e2, pm, q)
```

Arguments

pk1	public key part 1
u	u
e2	e2
pm	pm
q	q

Value

polynomial which contains the message in ciphertext

GenA *Generate a*

Description

Generate a

Usage

```
GenA(n, q)
```

Arguments

n	the order
q	the ciphermod of coefficients

Value

polynomial of order x^^n with coefficients 0,..,q

Examples

```
n = 16
q =  7
GenA(n, q)
```

GenError *Generate a*

Description

Generate a

Usage

```
GenError(n)
```

Arguments

n	the order

Value

polynomial of order x^^n with discrete Gaussian distribution, bounded (not strictly true) by -n,n

Examples

```
n = 16
GenError(n)
```

GenEvalKey0 *Generate the Evaluation Key*

Description

Generate the Evaluation Key

Usage

```
GenEvalKey0(a, s, e)
```

Arguments

a	a
s	s
e	e

Value

polynomial

GenPubKey *Generate the Public Key*

Description

Generate the Public Key

Usage

```
GenPubKey(a, n, e, pm, q)
```

Arguments

a	a
n	n
e	e
pm	pm
q	q

Value

list with the two polynomials that form the public key

GenPubKey0 *Generate part 0 of the Public Key*

Description

Generate part 0 of the Public Key

Usage

```
GenPubKey0(a, s, e, pm, q)
```

Arguments

a	a
s	s
e	e
pm	pm
q	q

Value

polynomial

GenPubKey1 *Generate part 1 of the Public Key*

Description

Generate part 1 of the Public Key

Usage

```
GenPubKey1(a)
```

Arguments

a	a

Value

polynomial

GenSecretkey *Generate Secret key*

Description

Generate Secret key

Usage

```
GenSecretKey(n)
```

Arguments

n	the order

Value

polynomial of order x^^n with coefficients (-1,-,1)

Examples

```
n = 16
GenSecretKey(n)
```

GenU *Generate u*

Description

Generate u

Usage

```
GenU(n)
```

Arguments

n	the order

Value

polynomial of order x^^n-1 with coefficients (-1,-,1)

Examples

```
n = 16
GenU
```

pi_function *pi function*

Description

Pi function

Usage

```
pi_function(M, z)
```

Arguments

M	M
z	z

Value

Pi of M

pi_inverse *pi inverse function*

Description

Pi inverse function

Usage

```
pi_inverse(z)
```

Arguments

z	z

Value

inverse of z

round_coordinates *round coordinates*

Description

Round coordinates

Usage

```
round_coordinates(coordinates)
```

Arguments

coordinates	coordinates

Value

rounded coordinates

sigma_function *sigma function*

Description

Sigma

Usage

```
sigma_function(xi, M, p)
```

Arguments

xi	xi
M	M
p	p

Value

sigma of xi

sigma_inverse *sigma inverse*

Description

Sigma inverse

Usage

```
sigma_inverse(xi, M, b)
```

Arguments

xi	xi
M	M
b	b

Value

sigma inverse of xi

sigma_R_discretization *sigma discretization*

Description

Sigma discretization

Usage

```
sigma_R_discretization(xi, M, z)
```

Arguments

xi	xi
M	M
z	z

Value

sigma R discretization

vandermonde *vandermonde*

Description

Vandermonde

Usage

```
vandermonde(xi, M)
```

Arguments

xi	xi
M	M

Value

The Vandermonde matrix

Index