Companion software for "Volker Ziemann, *Physics and Finance, Springer, 2021*" (https://link.springer.com/book/10.1007/978-3-030-63643-2)

RSA public-key cryptography (Section 12.6)

Volker Ziemann, 211116, CC-BY-SA-4.0

In this example we illustrate the workings of the public-key cryptographic system from Rivest, Shamir, and Adelman, commonly referred to by the acronym RSA. We assume that Alice, one of the participants in the secret communication first specifies two prime numbers p and q, as well as an encryption key e. From p and q she calculate $n = pq$ and the totient $\phi(n) = (p-1)(q-1)$. Here public key is the pair of (n, e), but she must absolutely keep p, q, and $\phi(n)$ secret. Please consult the discussion in Section 12.6 for more background.

```
p=5;                % Alice's prime, secret
q=11;               % also her prime, secret
e=3;                % encryption key, public
n=p*q               % public
```

```
n = 55
```

```
phin=(p-1)*(q-1)   % totient, secret
```

```
phin = 40
```

Following the discussion in Section 12.6, Alice needs to solve Equation 12.29, which is called Bezout's equation and can be solved using MATLAB's functon `gcd()`. It returns the decryption key d and the greatest common denominator `gcdval`, which must be 1, otherwise e and `phin` are not coprime, a prerequisite for the RSA algorithm to work. The call to `powermod()` only ensures that d is positive. Note that only Alice knows $p, q,$ and $\phi(n)$ and can calculate d.

```
[gcdval,d,~]=gcd(e,phin) % coprime -> gcdval must be 1!
```

```
gcdval = 1
d = -13
```

```
if gcdval ~= 1
  disp('Error: e and phin are not coprime');
  return;
end
d=powermod(d,1,phin)    % key to decode
```

```
d = 27
```

Bob, a friend Alice, now wants to send a secret `message`, say $m = 6$ to Alice. He therefore obtains Alice's public key (n, e) and encodes his `message` m by calculating the `ciphertext` $c = m^e (\mathrm{mod}\ n)$, which he then sends across an open communication channel to Alice.

```
message=6                       % Bob's message
```

```
message = 6
```

```
ciphertext=powermod(message,e,n)  % encrypted message
```

```
ciphertext = 51
```

But only Alice knows d and can therefore decrypt Bob's secret message by calculating $m' = c^d \pmod{n}$, which should be equal to Bob's original message.

```
decoded=powermod(ciphertext,d,n)
```

```
decoded = 6
```