Polinomios

Imanol

9/3/2021

En R

```
library(polynom)
p = polynomial(coef = c(1,2,3,4,5))
p
```

```
## 1 + 2*x + 3*x^2 + 4*x^3 + 5*x^4
```

```
q = polynomial(coef = c(1,2,3))
q
```

```
## 1 + 2*x + 3*x^2
```

```
p==q
```

```
## [1] FALSE
```

```
# Calcular grado del polinomio
gradoP = length(p)-1
gradoQ = length(q)-1
gradoP
```

```
## [1] 4
```

```
gradoQ
```

```
## [1] 2
```

```
p+q
```

```
## 2 + 4*x + 6*x^2 + 4*x^3 + 5*x^4
```

```
p*q
```

```
## 1 + 4*x + 10*x^2 + 16*x^3 + 22*x^4 + 22*x^5 + 15*x^6
```

```
# Para dividir
cociente = p / q
resto = p%%q
cociente
```

```
## 0.2962963 + 0.2222222*x + 1.666667*x^2
```

```
resto
```

```
## 0.7037037 + 1.185185*x
```

```
q*cociente + resto == p
```

```
## [1] TRUE
```

```
# Para evaluar un polinomio
predict(p,1)
```

```
## [1] 15
```

```
predict(p,0)
```

```
## [1] 1
```

```
# Para hallar raices
polyroot(c(-4,0,1)) # x^2-4
```

```
## [1]  2+0i -2+0i
```

```
polyroot(p)
```

```
## [1]  0.1378323+0.6781544i -0.5378323+0.3582847i -0.5378323-0.3582847i
## [4]  0.1378323-0.6781544i
```

En PYthon

Primera manera sympy

```
import sympy

x = sympy.symbols('x') # A partir de ahora x es un simbolo
p = sympy.Poly(x**2)
p
```

```
## Poly(x**2, x, domain='ZZ')
```

```
q = sympy.Poly(1+x+x**3)
q
```

```
## Poly(x**3 + x + 1, x, domain='ZZ')
```

```
p == q

#Grado del polinomio
```

```
## False
```

```
p.degree()
```

```
## 2
```

```
q.degree()
```

```
## 3
```

```
p+q
```

```
## Poly(x**3 + x**2 + x + 1, x, domain='ZZ')
```

```
p*q
```

```
## Poly(x**5 + x**3 + x**2, x, domain='ZZ')
```

Segunda manera sympy

```
import numpy

r = numpy.poly1d([1,2,1]) # X^2 + 2x + 1
print(r)
```

```
##    2
## 1 x + 2 x + 1
```

```
s = numpy.poly1d([1,2,3,4,5]) # X^4 + 2X^3 + 3X^2 + 4x + 5
print(s)
```

```
##    4     3     2
## 1 x + 2 x + 3 x + 4 x + 5
```

```
r == s

#Grado del polinomio
```

```
## False
```

```
r.order
```

```
## 2
```

```
s.order
```

```
## 4
```

```
r+s
```

```
## poly1d([1, 2, 4, 6, 6])
```

```
r*s

# Division de polinomios
```

```
## poly1d([ 1,  4,  8, 12, 16, 14,  5])
```

```
s/r # Primera parte cociente, segunda parte
```

```
## (poly1d([1., 0., 2.]), poly1d([3.]))
```

```
r*numpy.poly1d([1,0,2]) + 3 == s

# Evaluar polinomios
```

```
## True
```

```
r(0)
```

```
## 1
```

```
s(2)

#Para hallar las raices
```

```
## 57
```

```
r.r
```

```
## array([-1., -1.])
```

```
s.r
```

```
## array([-1.28781548+0.85789676j, -1.28781548-0.85789676j,
##         0.28781548+1.41609308j,  0.28781548-1.41609308j])
```

Manejar polinomios

```
#Grado del polinomio
p.degree()
```

```
## 2
```

```
q.degree()
```

```
## 3
```

En Matlab

```
p = [1,2,3,4,5]
q = [1,0,0,1]

gradoP = length(p)-1
gradoQ = length(q)-1

% Para poder sumar los polinomios
p = [zeros(1,gradoQ-gradoP),p]
q = [zeros(1,gradoP-gradoQ),q]
p+q

% Producto de polinomios
p = [1,2,3,4,5]
q = [1,0,0,1]
producto = conv(p,q)

% Division de polinomios
p = [1,2,3,4,5]
q = [1,0,0,1]
[cociente, resto] = deconv(p,q)

p == conv(q,cociente) + resto

% Evaluar polinomios
p = [1,2,3,4,5]
q = [1,0,0,1]
polyval(p,0)
polyval(q,3)

% Raices
p = [1,2,3,4,5]
q = [1,0,0,1]
roots(p)
roots(q)
```