Fundament

Josef Brejcha

2018-04-24

Basis (Fundament) in the Package of the Methods Used

Basic: Step in the Chain of Power of Two

$M_1 = A$
$M_2 = (I + Q) * M_1 = A + Q * A$
$M_3 = A + Q * A + Q^2 * A$
$M_4 = A + Q * A + Q^2 * A + Q^3 * A = M_2 + (Q^2 + Q^3) * A$
$M_4 = M_2 + Q^2 * (I + Q) * A = M_2 + Q^2 * M_2$
$M_4 = (I + Q^2) * M_2$
$M_5 = M_4 + Q^4 * A$
$M_6 = M_5 + Q^5 * A$
$M_7 = M_6 + Q^6 * A$
$M_8 = M_7 + Q^7 * A = M_6 + (Q^6 + Q^7) * A$
$M_8 = M_5 + (Q^5 + Q^6 + Q^7) * A = M_4 + (Q^4 + Q^5 + Q^6 + Q^7) * A$
$M_8 = M_4 + Q^4 * (I + Q + Q^2 + Q^3) * A$
$M_8 = M_4 + Q^4 * M_4 = (I + Q^4) * A$
...
$M_{2n} = (I + Q^n) * M_n, n = 1, 2, 3, ..$

More Complex

$M_3 = A + Q * A + Q^2 * A$
$M_4 = A + Q * A + Q^2 * A + Q^3 * A$
$M_5 = A + Q * A + Q^2 * A + Q^3 * A + Q^4 * A$
$M_6 = A + Q * A + Q^2 * A + Q^3 * A + Q^4 * A + Q^5 * A$
$M_6 = (I + Q^3) * M_3$
...
$M_{3*2^n} = (I + Q^{3*2^{n-1}}) * M_{3*2^{n-1}},$
$n = 1, 2, 3, ..$

and

$M_{k2^n} = (I + Q^{k2^{n-1}}) * M_{k2^{n-1}},$
$k = 1, 2, 3, .., n = 1, 2, 3, ..$

Examples

For Basic

```
require(matrixcalc)
An = 2
matmult <- function(A, B){
    C = matrix(numeric(4), 2, 2)
    for (i in 1:2){
        for (j in 1:2){ C[i, j] = sum(A[i, ]*B[, j])}
    }
    return(C)
}
Q = array(c(0.58, 0.53, 0.42, 0.47), c(2, 2))
q = 0
for (i in 1:8){
  q = q + matrix.power(Q, i)
}
print(paste("i =", i))
```

```
## [1] "i = 8"
```

```
print(q)
```

```
##          [,1]     [,2]
## [1,] 4.486427 3.513573
## [2,] 4.433795 3.566205
```

```
M = Q
I = diag(1, 2, 2)
n = c(1, 2, 4, 8)
for (i in 2:length(n)){
  M = matmult((I + matrix.power(Q, n[i-1])), M)
}
print(paste("n[i] =", n[i]))
```

```
## [1] "n[i] = 8"
```

```
print(M)
```

```
##          [,1]     [,2]
## [1,] 4.486427 3.513573
## [2,] 4.433795 3.566205
```

For More

```
Q = array(c(0.58, 0.53, 0.42, 0.47), c(2, 2))
k = 3
q = 0
for (i in 1:48){
  q = q + matrix.power(Q, i)
  if (i == k) Qk = q
}
print(paste("i =", i))
```

```
## [1] "i = 48"
```

```
print(q)
```

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

```
## [1,] 26.80222 21.19778
## [2,] 26.74958 21.25042
```

```
M = Qk
I = diag(1, 2, 2)
n = integer(5)
for (i in 1:5){
  n[i] = k*2^(i-1)
}
print(n)
```

```
## [1]  3  6 12 24 48
```

```
for (i in 2:length(n)){
  M = matmult((I + matrix.power(Q, n[i-1])), M)
}
print(paste("n[i] =", n[i]))
```

```
## [1] "n[i] = 48"
```

```
print(M)
```

```
##          [,1]     [,2]
## [1,] 26.80222 21.19778
## [2,] 26.74958 21.25042
```