fast_fibonacci

October 4, 2021

[1]:
```python
from sympy import *
```

Consider Fibonacci numbers F_n: $F_0 = 0, F_1 = 1, F_{n+2} = F_n + F_{n+1}$.

[2]:
```python
def F(n):
    if n == 0:
        return 0
    if n <= 2:
        return 1
    return F(n-1) + F(n-2)

[F(n) for n in range(11)]
```

[2]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

Denote $\Phi_{n+1} := \begin{bmatrix} F_{n+1} \\ F_n \end{bmatrix}$

Then $\Phi_{n+2} = \begin{bmatrix} F_{n+2} \\ F_{n+1} \end{bmatrix} = \begin{bmatrix} F_{n+1} + F_n \\ F_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} F_{n+1} \\ F_n \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \Phi_{n+1}$

So $\Phi_{n+1} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \Phi_n$

Denote $A := \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$

Then $\Phi_{n+1} = A \cdot \Phi_n = A^n \cdot \Phi_1 = A^n \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Notice, that A^n has form $\begin{bmatrix} p+q & q \\ q & p \end{bmatrix}$ for all natural n

Let's check it by induction

[3]:
```python
p = IndexedBase('p')
q = IndexedBase('q')
m, n = symbols('m, n')
```

[4]:
```python
def A(n):
    return Matrix([[p[n] + q[n], q[n]], [q[n], p[n]]])
```

[5]:
```
A(n)
```

[5]:

$$\begin{bmatrix} p_n + q_n & q_n \\ q_n & p_n \end{bmatrix}$$

Base case:

[6]:
```
A1 = A(1).subs({p[1]: 0, q[1]: 1}); A1
```

[6]:

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

Induction step:

If $A^n = \begin{bmatrix} p_i + q_i & q_i \\ q_i & p_i \end{bmatrix}$ for all natural $i \leqslant n$
then $A^n \cdot A = A^{n+1}$ with substitution $p_{n+1} = q_n, q_{n+1} = p_n + q_n$

[7]:
```
A(n) * A1
```

[7]:

$$\begin{bmatrix} p_n + 2q_n & p_n + q_n \\ p_n + q_n & q_n \end{bmatrix}$$

[8]:
```
A(n+1).subs({p[n+1]: q[n], q[n+1]: p[n] + q[n]})
```

[8]:

$$\begin{bmatrix} p_n + 2q_n & p_n + q_n \\ p_n + q_n & q_n \end{bmatrix}$$

Now we can get $\Phi_{n+1} = \begin{bmatrix} F_{n+1} \\ F_n \end{bmatrix} = A^n \cdot \Phi_1 = A^n \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} p_n + q_n & q_n \\ q_n & p_n \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

So $F_n = q_n = p_{n+1}, \quad F_{n+1} = p_n + q_n = q_{n+1}$

Let's compute A^{m+n} by coefficients of A^m and A^n

[9]:
```
Amn = A(n) * A(m); Amn
```

[9]:

$$\begin{bmatrix} (p_m + q_m)(p_n + q_n) + q_m q_n & (p_n + q_n) q_m + p_m q_n \\ (p_m + q_m) q_n + p_n q_m & p_m p_n + q_m q_n \end{bmatrix}$$

[10]:
```
p_mn = Amn[1, 1]; p_mn
```

[10]:

$$p_m p_n + q_m q_n$$

[11]:
```
q_mn = Amn[0, 1]; q_mn
```

[11]:

$$(p_n + q_n) q_m + p_m q_n$$

Now we can use these formulas to fast compute of Fibonacci numbers (see **pmtests.lesson1.Fibonacci.fast()** method):

$$\begin{cases} p_{m+n} = p_m p_n + q_m q_n \\ q_{m+n} = (p_n + q_n) q_m + p_m q_n \end{cases}$$

Let's check its

[12]:
```python
pn = [0] # pn[i] = p_mn[i+1]
qn = [1] # qn[i] = q_mn[i+1]

for i in range(1, 10):
    pn.append(p_mn.subs({m: i, n: 1}))
    qn.append(q_mn.subs({m: i, n: 1}))
    for j in range(i):
        pn[i] = pn[i].subs({p[j+1]: pn[j], q[j+1]: qn[j]})
        qn[i] = qn[i].subs({p[j+1]: pn[j], q[j+1]: qn[j]})

pn, qn
```

[12]:
```
([0, 1, 1, 2, 3, 5, 8, 13, 21, 34], [1, 1, 2, 3, 5, 8, 13, 21, 34, 55])
```

[13]:
```python
display(A1**2)
Amn.subs({m: 1, n: 1, p[1]: pn[0], q[1]: qn[0]})
```

$$\begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

[13]: $$\begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

[14]:
```python
display(A1**3)
Amn.subs({m: 2, n: 1,
          p[1]: pn[0], q[1]: qn[0],
          p[2]: pn[1], q[2]: qn[1]})
```

$$\begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix}$$

[14]: $$\begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix}$$

[15]:
```python
display(A1**4)
Amn.subs({m: 2, n: 2,
          p[1]: pn[0], q[1]: qn[0],
          p[2]: pn[1], q[2]: qn[1]})
```

$$\begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix}$$

[15]: $$\begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix}$$