aposentadoria

March 8, 2021

1 Aposentadoria: simulando o futuro

Doherty Andrade – www.metodosnumericos.com.br

2 1. Introdução

Para planejar a sua aposentadoria, fora do INSS, você precisa fazer um aplicação de longo prazo. Podemos simular os pagamentos realizados mensalmente, possíveis taxas mensais de juros e o montante capitalizado ao longo dos anos. É isso que vamos tentar fazer aqui utilizando Python.

Os resultados são bem sensíveis à taxa de juros. Ao fazer simulações procure ser realista e use taxas reais. No longo prazo com a estabilização da economia as taxas de juros tendem a cair e este é um complicador, mas ainda assim podemos fazer boas simulações.

Antes vamos importar as bibliotecas que precisaremos.

```
In [1]: import numpy as np
        import sympy as sp
        import scipy as sc
        import pandas as pd
        import matplotlib.pyplot as plt
        from sympy import Symbol, sin, cos, exp
```

3 2. Acumulando capital

Se você quiser fazer sua própria poupança para garantir sua própria aposentadoria no futuro, vai precisar responder às seguintes perguntas.

(a) Quanto deveria poupar?

(b) Por quanto tempo devo poupar?

(c) Qual o valor a receber?

(d) Por quanto tempo vou poder usufruir dessa economia?

Para responder estas questões usaremos Python para determinar os valores e prazos.

Precisaremos de um pouco de matemática financeira como juros compostos e acumulação de capital.

O valor futuro (FV) de um valor presente (PV) que rende juros compostos a taxa (i) após um tempo n é dado por:

$$FV = PV(1+i)^n.$$

Suponha que já tenha um valor inicial PV (aporte inicial- uma conta poupança que estava esperando melhor uso) para iniciar o investimento e depósitos mensais fixos M para incrementar a poupança. Então, a equação deve ser acrescida de um termo M para representar estes depósitos, temos mês a mês:

$$FV_1 = PV(1+i)^1 + M(\text{valor futuro para o primeiro mês})$$

$$FV_2 = PV(1+i)^2 + M(1+i) + M(\text{valor futuro para o segundo mês})$$

$$FV_3 = PV(1+i)^3 + M(1+i)^2 + M(1+i) + M(\text{valor futuro para o terceiro mês})$$

Observando a sequência que está se formando, vemos que para o tempo n teremos a seguinte expressão:

$$FV_n = PV(1+i)^n + M(1+i)^{n-1} + M(1+i)^{n-2} + ... + M(1+i) + M$$

Colocando M em evidência, vemos que os termos $(1+i)$ formam a soma de um Progressão Geométrica finita com razão $(1+i)$:

$$FV_n = PV(1+i)^n + M\left[(1+i)^{n-1} + (1+i)^{n-2} + ... + (1+i) + 1\right]$$

A soma dos termos de um PG finita é dada pela fórmula:

$$S_n = a_1\left[\frac{q^n - 1}{q-1}\right].$$

Substituindo na expressão anterior e simplificando temos a fórmula para calcular o montante acumulado após certo tempo acrescido dos juros tanto sobre o capital inicial quanto sobre os depósitos mensais:

$$FV_n = PV(1+i)^n + M\left[\frac{(1+i)^n - 1}{i}\right].$$

Note que se $M = 0$, então recai-se na conhecida fórmula do montante a juros compostos.
Esta fórmula pode ser calculada pela função abaixo, em seguida um exemplo de como usá-la.

```
In [2]: def  montante_acumulado(aporte, depositomensal, juros, prazo):
            montante = aporte * (1 + juros)**prazo + deposito*(((1 + juros)**prazo - 1)/juros)
            print("O capital acumulado é: %d reais." %montante)
```

3.0.1 EXEMPLO

Uma pessoa possui em conta R\$ 20.000,00 e deseja iniciar o projeto de aposentadoria. Ela consegue depositar mensalmente a quantia de R$ 1.500,00 no seu projeto de aposentadoria. Supondo que a taxa ao longo dos próximos 20 anos seja de 0.6% ao mês. Quanto ela teria acumulado neste período?

```
In [3]: aporte   = 20000      # aporte inicial
        deposito = 1500       # valor do depósito mensal
        juros    = 0.006      # 0.6%  taxa mensal
        prazo    = 240        # tempo de capitalização em meses

        montante_acumulado(aporte, deposito, juros, prazo)
```

```
O capital acumulado é: 884694 reais.
```

4 Planejando retiradas

Tendo acumulado essa quantia, vamos investigar quanto tempo durarão essas economias se fizermos mensalmente uma retirada, tendo decidido o valor do saque e a taxa vigente na época.

Procdemos de modo similar ao caso de acumular renda para a formação da poupança. Mas com a diferença é que agora temos de subtrair o valor da retirada mensal M:

$$FV_1 = PV(1+i)^1 - M(\text{valor após a primeira retirada})$$

$$FV_2 = PVV(1+i)^2 - M(1+i) - M(\text{valor após a segunda retirada})$$

$$FV_3 = PVV(1+i)^3 - M(1+i)^2 - M(1+i) - M(\text{valor após a terceira retirada})$$

Note que caímos na mesma PG anteriormene, com sinal negativo que modela o saque. A expressão então fica assim:

$$FV_n = PV(1+i)^n - M\left[\frac{(1+i)^n - 1}{i}\right]$$

Como queremos saber quanto tempo dura este capital após todas as retiradas, então o valor \$FV_n \$ tem que ser igual a zero:

$$0 = PV(1+i)^n - M\left[\frac{(1+i)^n - 1}{i}\right].$$

Como queremos isolar n na expressão anterior, teremos que aplicar logaritmo:

$$0 = PV(1+i)^n - M\left[\frac{(1+i)^n - 1}{i}\right]$$

$$PV(1+i)^n = M\left[\frac{(1+i)^n - 1}{i}\right]$$

$$i\left[\frac{PV}{M}\right](1+i)^n = (1+i)^n - 1$$

$$(1+i)^n - i\left[\frac{PV}{M}\right](1+i)^n = 1$$

Colocando $(1+i)^n$ em evidência:

$$(1+i)^n\left(1 - i\left[\frac{PV}{M}\right]\right) = 1$$

$$(1+i)^n = \frac{1}{(1 - i\left[\frac{PV}{M}\right])}$$

Simplificando a fração do lado direito da equação:

$$(1+i)^n = \frac{M}{M - iPV}$$

Aplicando logaritmo e suas propriedades dos dois lados da equação, temos:

$$\ln(1+i)^n = \ln\left(\frac{M}{M - iPV}\right)$$

$$n\ln(1+i) = \ln\left(\frac{M}{M - iPV}\right)$$

$$n = \frac{\ln\left(\frac{M}{M-iPV}\right)}{\ln(1+i)}.$$

Observe que as retiradas mensais devem ser superiores ao rendimento mensal, isto é, $M - iPV > 0$. Caso contrario, a fórmula vai acusar erro, pois o logaritmo está definido apenas para números maiores do que zero.

Com esta fórmula é possível saber quanto tempo irá durar a sua poupança depois de definir quanto irá sacar por mês. Em Python pode ser calculado assim:

```
In [4]: import math
        def duracao_capital(juros, saque, capital):
            duracao = math.log((saque)/(saque - juros * capital))/math.log(1 + juros)/12
            print("Tempo de retiradas: %d anos." %duracao)
```

4.0.1 Exemplo

Suponha que a pessoa tem acumulado R\$884.694,00 e acabou de se aposentar. Agora fará retiradas mensais de R$10.000,00. Sabendo que a taxa mensal é de 0.08% ao mês

```
In [5]: juros = 0.008 # 0.8% ao mês
        saque = 10000
        capital = 884694
        duracao_capital(juros, saque, capital)
```

```
Tempo de retiradas: 12 anos.
```

4.0.2 O pacote financial do Python

O Python possui o pacote financial com muitas funções financeiras que nos ajudam em inúmeros problemas.

Principais funções do pacote Financial:

1. fv(rate, nper, pmt, pv[, when])– calcula o valor futuro fv.
2. pv(rate, nper, pmt[, fv, when]) – calculo o valor presente pv.
3. npv(rate, values) – Retorna NPV (Net Present Value = valor presente liquido) de um fluxo de pagamentos.
4. pmt(rate, nper, pv[, fv, when]) – Calcula o pagamento contra o principal do empréstimo mais juros.
5. ppmt(rate, per, nper, pv[, fv, when])– Calcula o pagamento contra o principal do empréstimo ppmt.
6. ipmt(rate, per, nper, pv[, fv, when])– Calcula a parte dos juros de um pagamento.
7. irr(values) – Retorna a Taxa Interna de Retorno (IRR).
8. mirr(values, finance_rate, reinvest_rate) – Taxa interna de retorno modificada.
9. nper(rate, pmt, pv[, fv, when]) – Calcula o número de pagamentos periódicos nper.
10. rate(nper, pmt, pv, fv[, when, guess, tol, . . .]) – Calcula a taxa de juros por período rate.

Vejamos um exemplo.

```
In [6]: import numpy as np

        aporte  = 20000
        deposito = 1900
        juros = 0.006 # 0.6% ao mês
        prazo = 240 # 20 anos
        poupanca = np.fv(juros, prazo, -deposito, -aporte)
        print("Montante de capital acumulado: %d reais."  %poupanca)

        print("Valor mínimo para saque na aposentadoria: %d reais."  %(poupanca*juros))

        juros = 0.008# 0.8% ao mês
        saque = 10000
        capital = poupanca
        periodo = np.round(np.nper(juros, -saque, capital),3)
        print("Prazo de duracao do capital após retiradas: %d meses."  %periodo)
```

```
Montante de capital acumulado: 1098199 reais.
Valor mínimo para saque na aposentadoria: 6589 reais.
Prazo de duracao do capital após retiradas: 264 meses.
```

Fazendo diretamente usando a função nper do finance,temos:

```
In [7]: import numpy as np
        print("O tempo em meses é:", np.round(np.nper(0.008, -11000, 1098199), 2))
```

```
O tempo em meses é: 201.16
```

```
In [ ]:
```