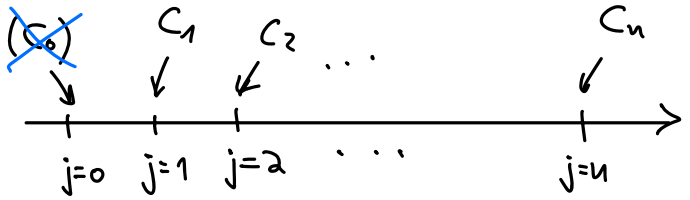

1.2 General Cash Flows

n = # of years, r = annual interest rate

Suppose at end of year j, there is a cash flow C_j:



(sometimes we write P = price for the present value PV)

$$PV = \frac{C_1}{1+r} + \frac{C_2}{(1+r)^2} + \dots + \frac{C_n}{(1+r)^n} = \sum_{j=1}^{n} \frac{C_j}{(1+r)^j}$$

↑ present value of "cash-flow" or price of this "financial instrument"

How do we implement such a fct. in python in the most efficient way.
Let's define $x = \frac{1}{1+r}$. Need to evaluate a polynomial $P(x) = \sum_{j=1}^{n} C_j x^j$

- explicit "for" loop: Discouraged, since very slow/inefficient
- instead: use vectorized operations

  vector in python is a numpy array
  - ↳ vector of C_j's: $(C_1, \dots, C_n)$
  - ↳ define vector $(1, \dots, n)$: $j = \text{arange}(1, n+1)$

↳ now: can define X**j = vector with x^k as k-th component

= (X**1, X**2, ..., X**n)

↳ now we can use dot/scalar product: P = dot(C, X**j)

- Horner's scheme:

$$\Big[\Big(\big(C_n x + C_{n-1}\big)x + C_{n-2}\Big)x + \dots \quad C_1\Big]x = P$$

(requires the fewest operations when evaluating a general polynomial)

E.g.: $\Big[\big((C_3 x + C_2)\big)x + C_1\Big]x$

- built-in python fct.: polval: uses optimized Horner's scheme

Next: special case where $C_j = C$ (all C_j's equal) is called <u>annuity</u>.

<u>Ordinary annuity</u>: pays at end of year

<u>Annuity due</u>: pays at beginning of year

Consider ordinary annuity: C at end of period with m payment periods per year:

$$PV = \sum_{j=1}^{nm} \frac{C}{(1+\frac{r}{m})^j} = C \sum_{j=1}^{nm} \frac{1}{(1+\frac{r}{m})^j} = C \sum_{j=1}^{nm} x^j \quad \text{with } x = \frac{1}{1+\frac{r}{m}}$$

$$= Cx \sum_{j=0}^{nm-1} x^j$$

$$= Cx \frac{x^{nm}-1}{x-1}$$

$$= C \frac{1}{1+\frac{r}{m}} \left(\frac{(1+\frac{r}{m})^{-nm}-1}{\frac{1}{1+\frac{r}{m}}-1} \right)$$

↑ recall: this is a geometric sum

$$\sum_{j=0}^{N} x^j = \frac{x^{N+1}-1}{x-1}$$

$$(1-x)\sum_{j=0}^{N} x^j = \sum_{j=0}^{N} x^j - \sum_{j=1}^{N+1} x^j = 1 - x^{N+1}$$

$$= C \frac{(1+\frac{r}{m})^{-nm} - 1}{1-(1+\frac{r}{m})}$$

$$PV = C\frac{m}{r}\left(1-(1+\frac{r}{m})^{-nm}\right)$$

Note: $FV = C\sum_{i=0}^{nm-1}(1+\frac{r}{m})^i = C\frac{m}{r}\left((1+\frac{r}{m})^{nm}-1\right)$

↓ at the end (of year n)

- perpetual annuity: $n \to \infty$

$$\Rightarrow PV_\infty = \lim_{n\to\infty} C\frac{m}{r}\left(1-(1+\frac{r}{m})^{-nm}\right) \qquad (r>0)$$

$$= C\frac{m}{r}$$

- Amortization: repay a loan P with regular payments C (e.g. mortgage)

Compute regular payments C by: $C = P\frac{r}{m}\left(1-(1+\frac{r}{m})^{-nm}\right)^{-1}$.

Now: simplify to $m=1$:

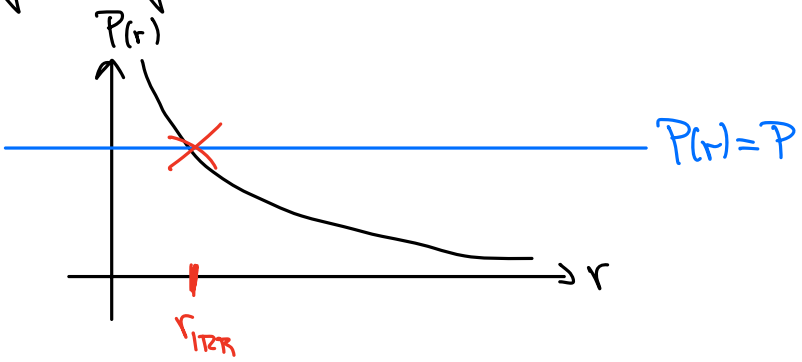
$$\frac{C}{r}\left(1-(1+r)^{-n}\right) = P(r) \text{ can be seen as fct. of } r$$

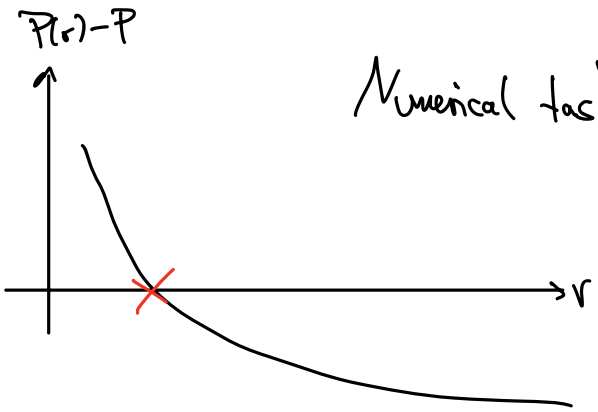
Sometimes the price of an instrument is fixed as P.

For which r is $P(r) = P$? The solution r_{return} is called IRR (internal rate of return).

Sometimes one also defines "net-present-value" $NPV(r) = P - P(r)$

(Again: P given, P(r) as above: Want to find r s.t. $P(r) = P_0$)



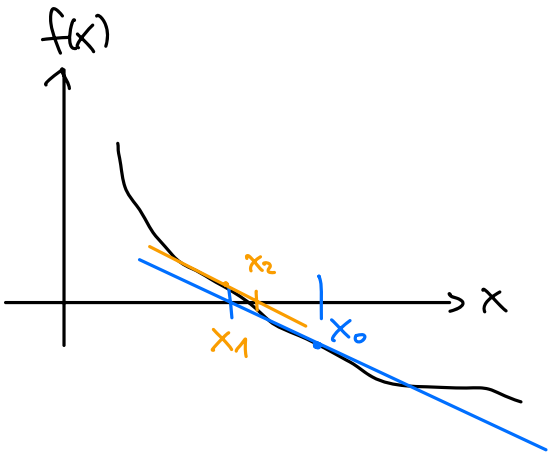


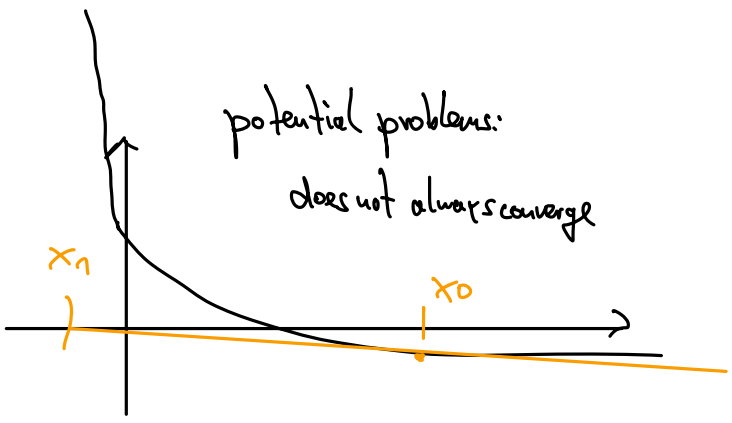
Numerical task: find zeroes of fct.s

(more specifically here: roots of polynomials)

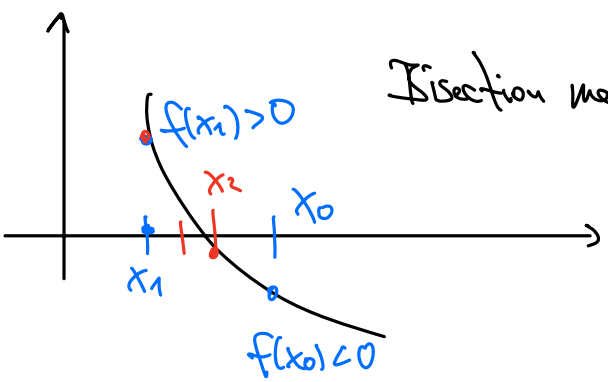
What numerical algorithms are there to find zeroes of fct.s?

- Newton's method





- Bisection method



- Secant method