1. Comparing different methods of interpolation

→ $f = \cos(x)$ (for 100 points)

cubic polynomial error: ~ 6.7689×10^{-9}
cubic spline error: ~ 1.629×10^{-7}
rational function error: 3.4849×10^{-5}

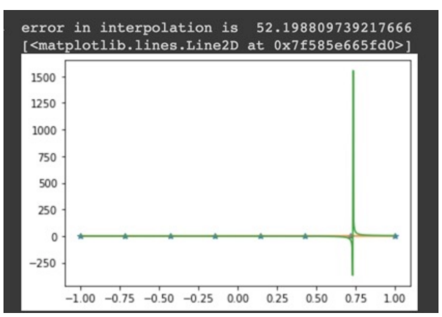
→ $f = \frac{1}{(1+x^2)}$

cubic polynomial error: ~ 2.7966×10^{-8}
cubic spline error: ~ 3.936×10^{-7}
rational function error: ~ 1.008×10^{-16}

The error of the Lorentzian should be near zero, since it is not well described by a Taylor series. The rational function interpolation takes care of this and thus provides an excellent fit for the Lorentzian.

As for the cos(x) function, the rational function worked poorly compared to the two other methods. In this case, the cubic polynomial interpolation yielded the smallest error.

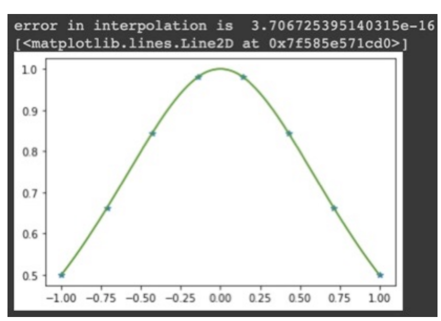
An error occurred when I changed n and m to 4 and 5 respectively.



The error jumped to 52.1988
+ visually, the rational function interpolation did not at all fit the Lorentzian properly.

However, upon changing np.linalg.inv to np.linalg.pinv, higher order n, m made the interpolation method work.

Notice the error dropped back down to the near-zero value, as it should.
error: 3.707×10^{-16}



SIDE NOTE

Playing around a little bit:
Set n = 49, m = 50, with np.linalg.pinv
error: 0.0001926 → much higher!

So what caused the error to change?

Well, np.linalg.pinv uses a pseudo inverse of a matrix when it isn't actually invertible.

For np.linalg.pinv:

$p = [1,\ 2.6645 \times 10^{-15},\ -3.33 \times 10^{-1},\ -3.108 \times 10^{-15}]$

$q = [3.108 \times 10^{-15},\ 6.67 \times 10^{-1},\ -5.329 \times 10^{-15},\ -3.33 \times 10^{-1}]$

For np.linalg.inv:

$p = [-0.382,\ 0,\ 2,\ -1.302]$ → looks very weird.

$q = [0,\ 0,\ -4,\ 2]$

The constant term in the denominator is set to 1, most likely to avoid any divisions by 0 (undefined).

Non-invertible ⇒ determinant is 0.

The values of p and q seemed to have occupied non-zero values when implementing np.linalg.pinv.

This allows us to do a pseudo inverse & properly use rational function interpolation with minimal error.