Phys 512 - PS1

Fernanda C. Rodrigues Machado
ID# 260905170

September 18, 2020

P3

Use $\cos(x)$ from $-\pi/2$ to $\pi/2$.
Compare accuracy of the interpolations:
- polynomial
- cubic spline
- rational function
using the same number of points for each.
Repeat using a Lorentzian $1/(1+x^2)$ from. -1 to 1.

Accompanying Python code: Phys512_PS1_P3_FCRM.py

Interpolation for $\cos(x)$:

Cubic polynomial

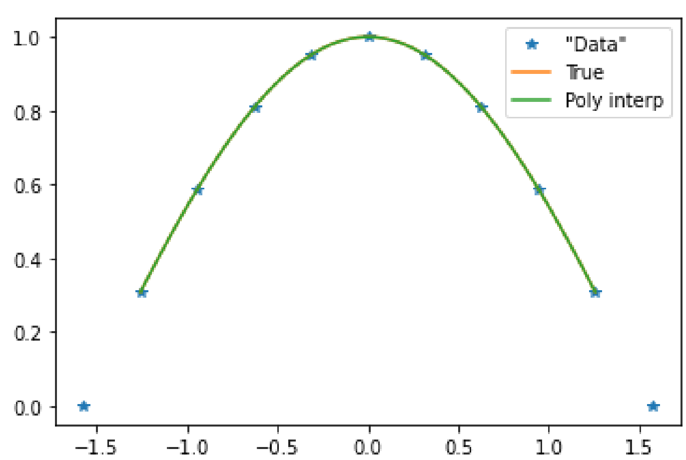

Figure 1: Cubic polynomial interpolation of $\cos(x)$.

Cubic polynomial error is 6.274279434148598e-05.

Cubic spline

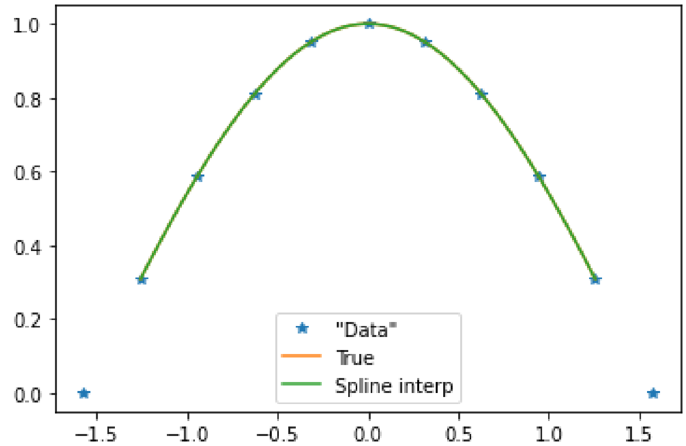

Figure 2: Cubic spline interpolation of $\cos(x)$.

Cubic spline error is 1.0525189115353985e-05.

Rational function

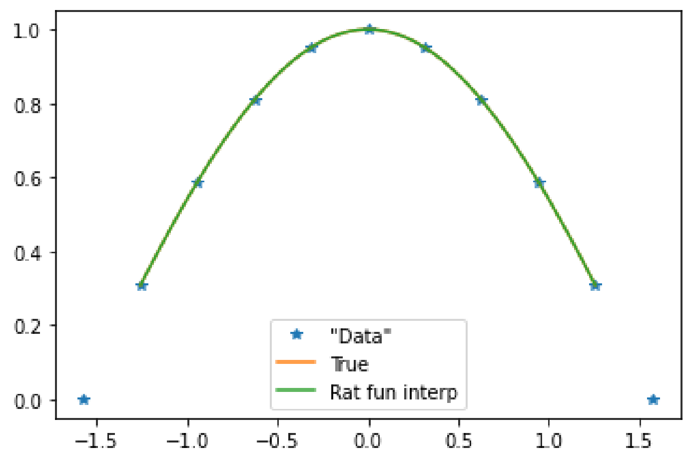

Figure 3: Rational function interpolation of $\cos(x)$.

Rational function error is 4.2809204874762446e-10.

Comparing interpolation methods for $\cos(x)$:

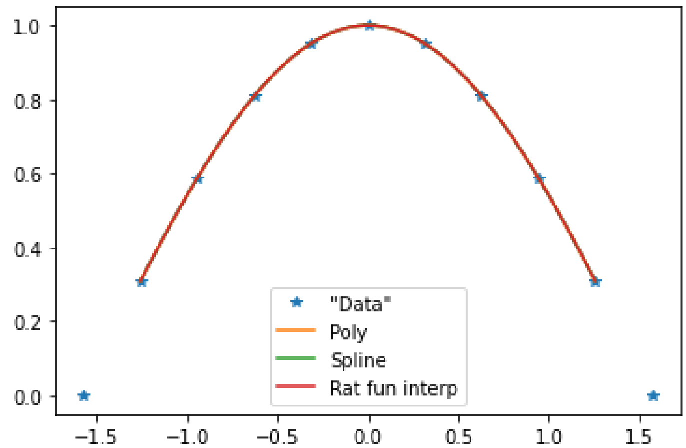

Figure 4: Comparing interpolation methods for $\cos(x)$.

Interpolation	Error
Cubic poly	6.274279434148598e-05
Cubic spline	1.0525189115353985e-05
Rational fun	4.2809204874762446e-10

Interpolation for a Lorentzian function $1/(1+x^2)$:

Cubic polynomial

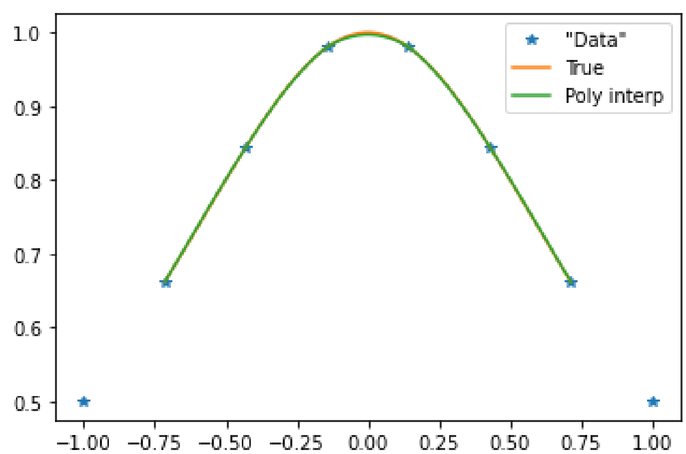

Figure 5: Cubic polynomial interpolation of a Lorentzian function.

Cubic polynomial error is 0.0011796404101852547.

Cubic spline

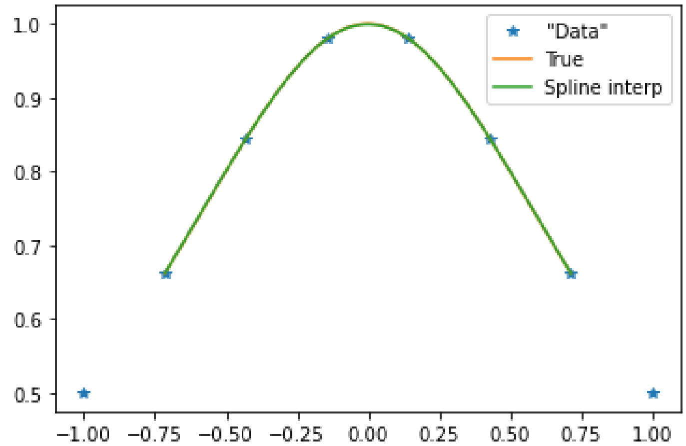

Figure 6: Cubic spline interpolation of a Lorentzian function.

Cubic spline error is 0.00037571667068939894.

Rational function

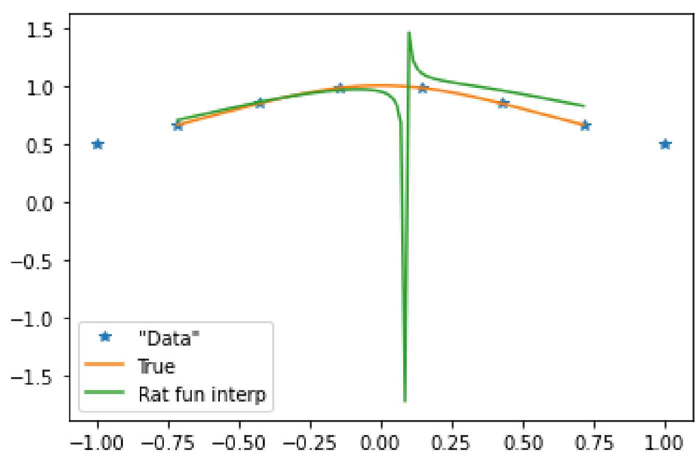

Figure 7: Rational function interpolation of a Lorentzian function.

Rational function error is 0.28802883676840935.

p_i	0.94866263	-12.	5.75	0.44277028
q_i	-12.	7.5	-8.	4.5

Comparing interpolation methods for a Lorentzian function:

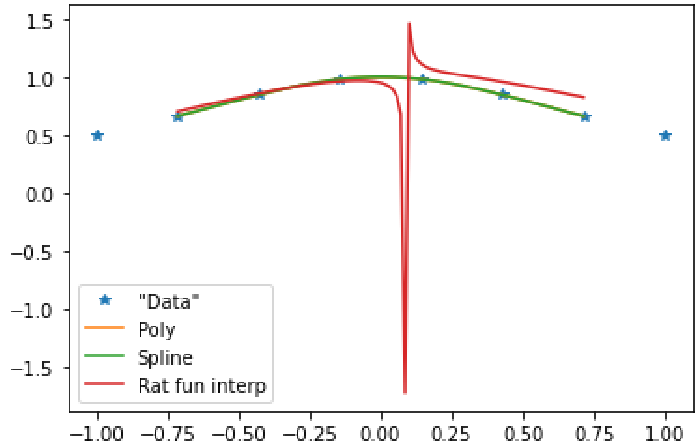

Figure 8: Comparing interpolation methods for a Lorentzian function.

Interpolation	Error
Cubic poly	0.0011796404101852547
Cubic spline	0.00037571667068939894
Rational fun	0.28802883676840935

We would expect a very small error for the rational fit of a Lorentzian since we are using a rational fit to fit a rational function.

So I will now switch to linalg.pinv.

Rational function with "linalg.pinv"

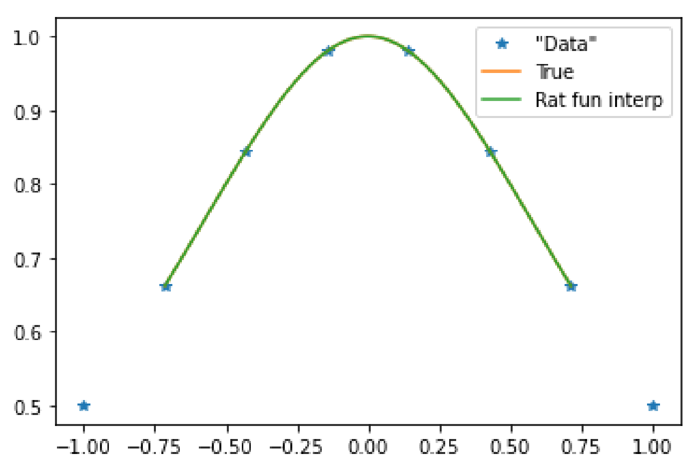

Figure 9: Rational function interpolation of a Lorentzian function using linalg.pinv.

Rational function (pinv) error is 2.261852513443805e-16.

p_i	1.	0.	-0.33333333	0.
q_i	4.44089210e-16	6.66666667e-01	-8.88178420e-16	-3.33333333e-01

Our matrix has two rows that are not independent. linalg.pinv solves the inversion of a Det=0 matrix (singular matrix).

The polynomial fit is given by

$$\frac{p(x)}{q(x)} = y(x) \ .$$

$$p(x) = 1p_0 + p_1 x + p_2 x^2 + ... \ ,$$

$$q(x) = 1 + q_1 x + q_2 x^2 + ...$$

To fit a Lorentzian $1/(1+x^2)$, we need a zeroth-order polynomial in the numerator, so p[0] should be 1 and all other p's=0

We can see that the fit done with linalg.pinv gives p[0]=1,p[1]=p[3]0 whereas linalg.inv gives high values for p[1] and p[2].

Comparing interpolation methods for a Lorentzian function (again):

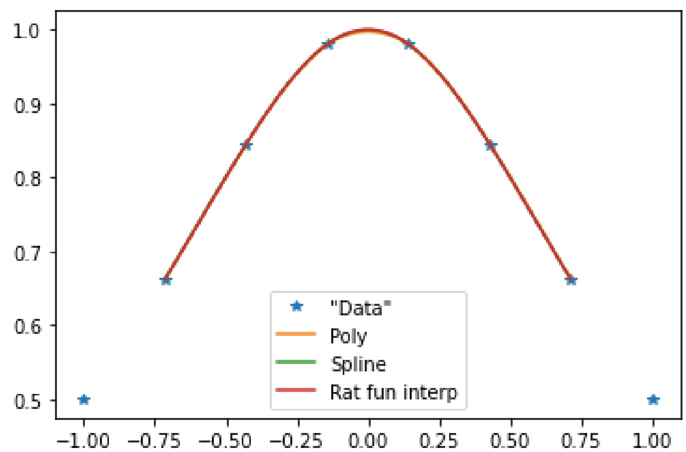

Figure 10: Comparing interpolation methods for a Lorentzian function.

Interpolation	**Error**
Cubic poly	0.0011796404101852547
Cubic spline	0.00037571667068939894
Rational fun (.pinv)	2.261852513443805e-16