lagrange_interpolations

April 18, 2024

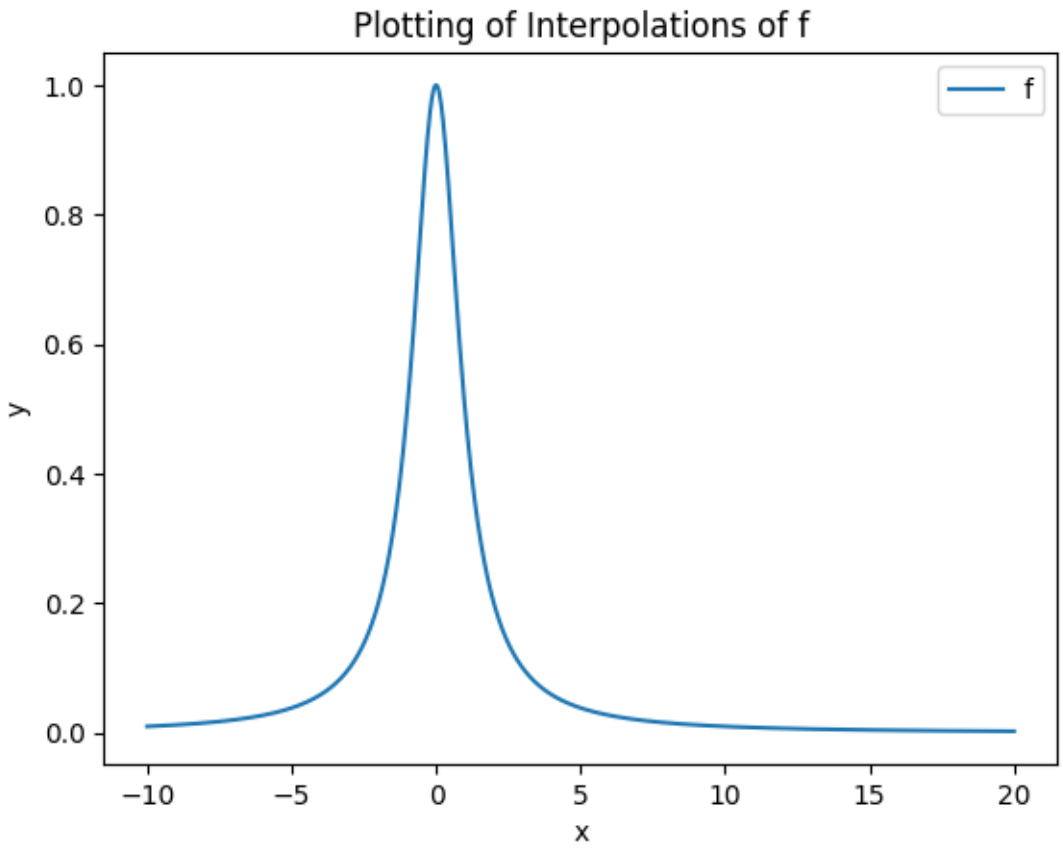
1 Lagrange Interpolations of a given analytic function

1.1 0. Definitions of Plotting parameters

[ ]:
```
import numpy as np

a, b, n_plot = -10, 20, 1000
x_plot = np.linspace(a, b, n_plot)
# print("x_plot =", x_plot)
```

1.2 1. Definition of f

[ ]:
```
from utils import *

# f_exp = "cos(x)" # "1/(1+x**2)"
# def f(x):
    # return eval(f_exp, {"x": x})

f = lambda x: 1/(1+x**2)

fig, ax = set_fig()
plot_f(ax, f, x_plot)
```



1.3 2. Definition of Interpolation parameters

[ ]:
```
from polynomial.utils import *

n = 40

# Defintion of Uniforms points
x_uniform = np.linspace(a, b, n)
y_uniform = [f(x) for x in x_uniform]
print("Uniforms points")
print("x_uniform =", x_uniform)
print("\ny_uniform =", y_uniform)

#Definition of Tchebychev points
x_tchebychev = tchebychev_points(a, b, n)
y_tchebychev = [f(x) for x in x_tchebychev]
print("\nTchebychev points")
print("x_tchebychev =", x_tchebychev)
print("\ny_tchebychev =", y_tchebychev)
```

```
Uniforms points
x_uniform = [-10.          -9.23076923  -8.46153846  -7.69230769  -6.92307692
  -6.15384615  -5.38461538  -4.61538462  -3.84615385  -3.07692308
  -2.30769231  -1.53846154  -0.76923077   0.           0.76923077
   1.53846154   2.30769231   3.07692308   3.84615385   4.61538462
   5.38461538   6.15384615   6.92307692   7.69230769   8.46153846
   9.23076923  10.          10.76923077  11.53846154  12.30769231
  13.07692308  13.84615385  14.61538462  15.38461538  16.15384615
  16.92307692  17.69230769  18.46153846  19.23076923  20.        ]

y_uniform = [0.009900990099009901, 0.011599972544443685, 0.013774553753362133,
0.01661913659160193, 0.020437779658967224, 0.025726899071395953,
0.033339909252318015, 0.044839479968161323, 0.0633195953540652,
0.09553420011305824, 0.15809167446211417, 0.29701230228471004,
0.6282527881040898, 1.0, 0.6282527881040887, 0.29701230228471004,
0.15809167446211406, 0.09553420011305815, 0.06331959535406517,
0.04483947996816129, 0.03333990925231801, 0.025726899071395953,
0.020437779658967224, 0.016619136591601923, 0.013774553753362128,
0.011599972544443674, 0.009900990099009901, 0.008548737923010773,
0.007455114914641137, 0.006558267685979275, 0.005813753483091954,
0.005188983389112346, 0.004659626678430615, 0.004207224476586421,
0.003817569857010548, 0.0034795857439930823, 0.0031845333433831425,
0.002925444442521074 6, 0.0026967081012941, 0.0024937655860349127]

Tchebychev points
x_tchebychev = [-9.98843554 -9.89602685 -9.71177921 -9.43682855 -9.07287004
-8.62214761
 -8.08744011 -7.47204418 -6.77975396 -6.01483764 -5.18201118 -4.28640924
 -3.3335535  -2.32931862 -1.27989606 -0.19175586  0.92839325  2.07364517
  3.23693904  4.41110276  5.58889724  6.76306096  7.92635483  9.07160675
 10.19175586 11.27989606 12.32931862 13.3335535  14.28640924 15.18201118
 16.01483764 16.77975396 17.47204418 18.08744011 18.62214761 19.07287004
 19.43682855 19.71177921 19.89602685 19.98843554]

y_tchebychev = [0.009923702151011782, 0.010108019604047558,
0.010491125754546266, 0.011104481362899777, 0.01200235667594107,
0.013272909851190158, 0.015058724325880785, 0.017595892773673957,
0.021292422900195854, 0.026897430949082982, 0.035902466052996684,
0.0516174211295445, 0.08255880170206485, 0.1556244566777482, 0.3790558120568454,
0.9645337949248454, 0.5370817254023783, 0.18867909255758195, 0.0871249051144412,
0.04888104543573196, 0.03102143761619173, 0.021395411486943704,
0.015667326041435663, 0.01200565961346107, 0.00953544390675747,
0.007798118855818746, 0.006535430956576048, 0.005593352673164703,
0.004875635009075586, 0.004319776397221688, 0.003883871848247988,
0.0035390695557751435, 0.003265068073809879, 0.0030473358871580887,
0.002875344762912063, 0.002714206646083506, 0.002639983097643359,
0.0025670366246447543, 0.0025198318033394936, 0.002496644787275692]
```

1.4 3. Test of Newton Lagrange Polynomial Representation

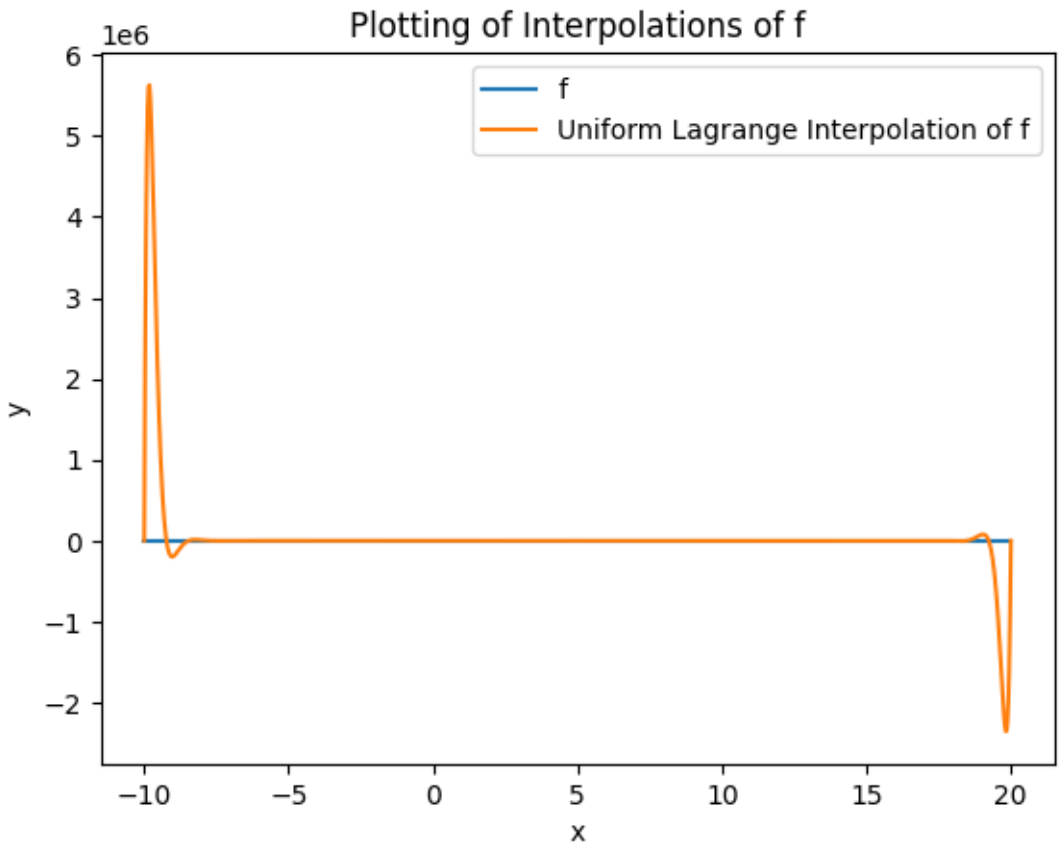
[ ]:
```
from polynomial.newton_poly import *

x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
print("x =", x)
print("y =", y)
polynomial = NewtonInterpolPoly(x, y)
print(polynomial)

x = 6
value = polynomial.horner_eval(x)
print(f"P{x}) = {value}")
```

```
x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
P(x) = 1.0 + 3.0 * (x - 1.0) + 1.0 * (x - 1.0)(x - 2.0)
P6) = 36.0
```

1.5 4. Uniform Lagrange Interpolation of f

[ ]:
```
uni_lagrange_poly = NewtonInterpolPoly(x_uniform, y_uniform,␣
  ↪"Uni_lagrange_poly")

print(uni_lagrange_poly)

x0 = 1
print(f"\nUni_lagrange_poly({x0}) =", uni_lagrange_poly.horner_eval(x0))

# print("\nx_uniform =", x_uniform)
# print("\ny_uniform =", y_uniform)

fig, ax = set_fig()
plot_f(ax, f, x_plot)
uni_lagrange_poly.plot(ax, x_plot, "Uniform Lagrange Interpolation of f")
```

```
Uni_lagrange_poly(x) = 0.009901 + 0.002209 * (x + 10.0) + 0.000402 * (x +
10.0)(x + 9.230769) + 7.1e-05 * (x + 10.0)(x + 9.230769)(x + 8.461538) + 1.3e-05
* (x + 10.0)(x + 9.230769)(x + 8.461538)(x + 7.692308) + 3e-06 * (x + 10.0)(x +
9.230769)(x + 8.461538)(x + 7.692308)(x + 6.923077) + 1e-06 * (x + 10.0)(x +
9.230769)(x + 8.461538)(x + 7.692308)(x + 6.923077)(x + 6.153846)

Uni_lagrange_poly(1) = 0.5008665305931007
```

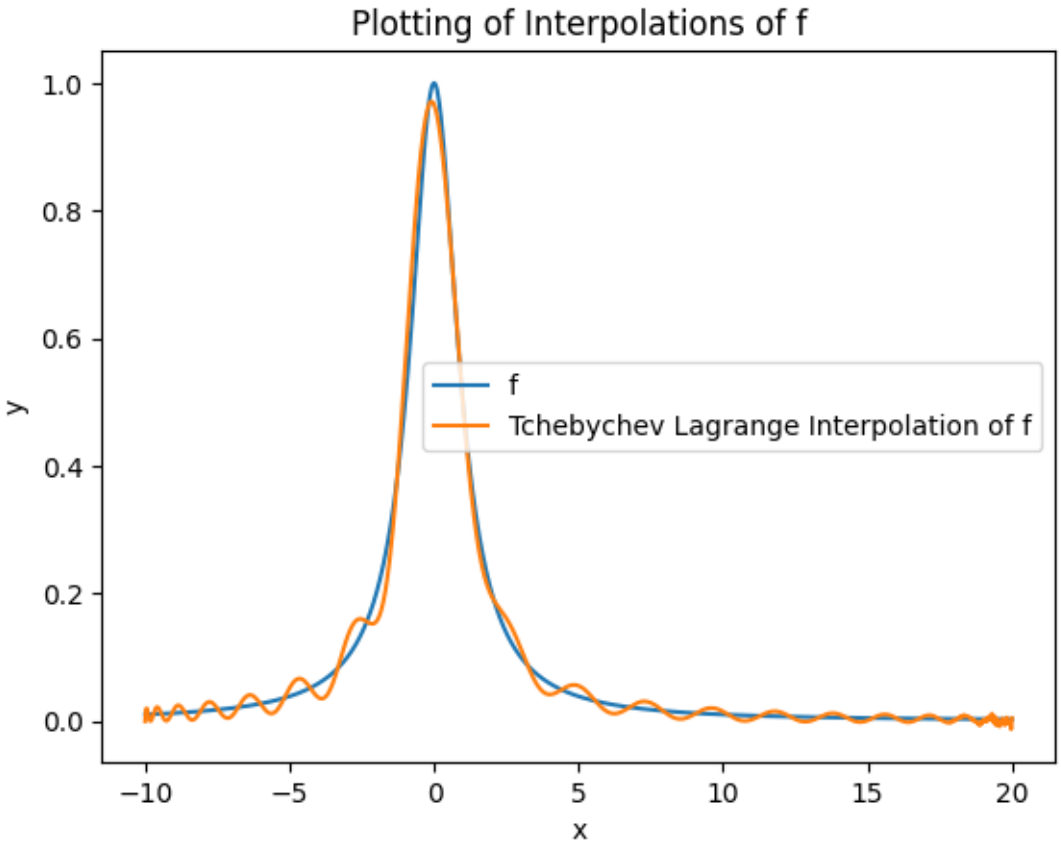


1.6 5. Tchebychev Lagrange Interpolation of f

[ ]:
```
tchebychev_lagrange_poly = NewtonInterpolPoly(x_tchebychev, y_tchebychev,␣
 ↪"Tchebychev_lagrange_poly")

print(tchebychev_lagrange_poly)

x0 = 1
print(f"\nTchebychev_lagrange_poly({x0}) =", tchebychev_lagrange_poly.
 ↪horner_eval(x0))

# print("\nx_tchebychev =", x_tchebychev)
# print("\ny_tchebychev =", y_tchebychev)

fig, ax = set_fig()
plot_f(ax, f, x_plot)

# uni_lagrange_poly.plot(ax, x_plot, "Uniform Lagrange Interpolation of f")
```

```
tchebychev_lagrange_poly.plot(ax, x_plot, "Tchebychev Lagrange Interpolation of␣
↪f")
```

```
Tchebychev_lagrange_poly(x) = 0.009924 + 0.001995 * (x + 9.988436) + 0.000306 *
(x + 9.988436)(x + 9.896027) + 4.3e-05 * (x + 9.988436)(x + 9.896027)(x +
9.711779) + 6e-06 * (x + 9.988436)(x + 9.896027)(x + 9.711779)(x + 9.436829) +
1e-06 * (x + 9.988436)(x + 9.896027)(x + 9.711779)(x + 9.436829)(x + 9.07287)

Tchebychev_lagrange_poly(1) = 0.496660645025543
```



1.7 6. Test of Gauss Integration

[ ]:
```
from integration import *

f_ = lambda x: x**2
start, end = -2, 5
gaus_int_f_ = gauss_integration(f_, -2, 5)
print(f"Gauss Integration of f_ from {start} to {end} =", gaus_int_f_)
```

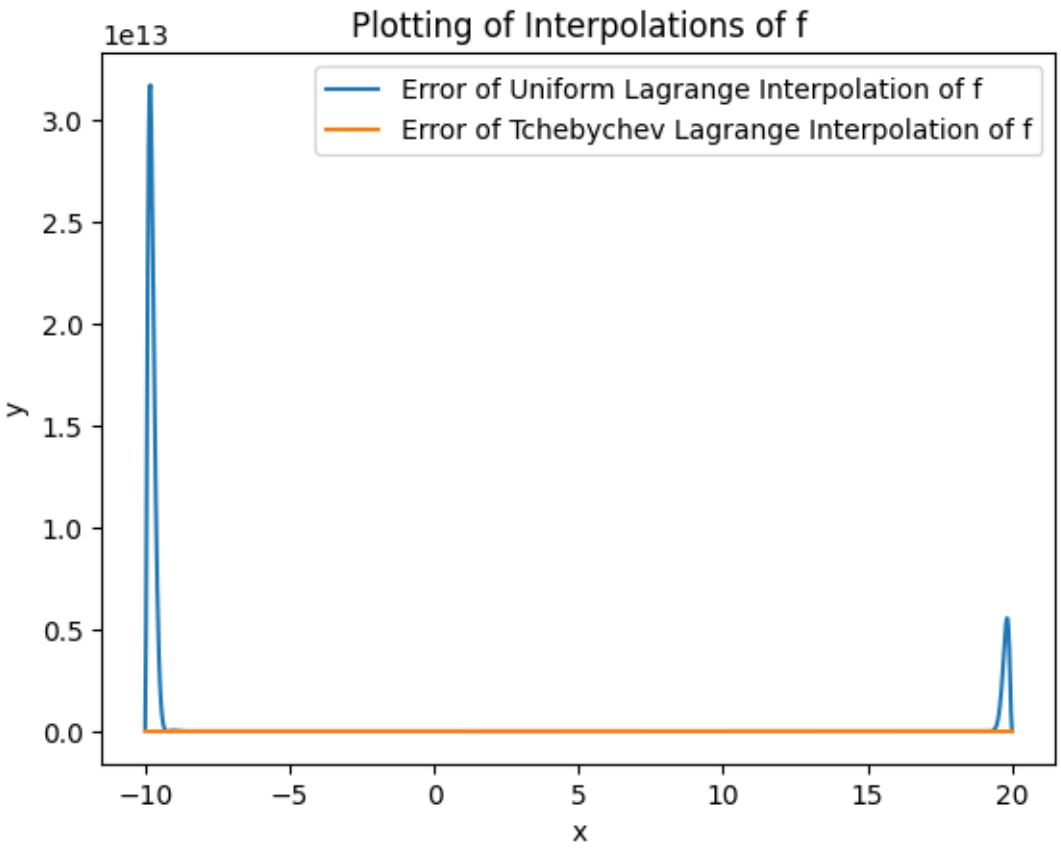
```
Gauss Integration of f_ from -2 to 5 = 44.33333333333334
```

1.8 7. Errors of Lagrange Interpolations of f

[ ]:
```
func_err_uniform = lambda x: (f(x) - uni_lagrange_poly.horner_eval(x))**2
func_err_tchebychev = lambda x: (f(x) - tchebychev_lagrange_poly.
 ↪horner_eval(x))**2
err_uniform = sqrt(gauss_integration(func_err_uniform, a, b))
err_tchebychev = sqrt(gauss_integration(func_err_tchebychev, a, b))
print("err_uniform =", err_uniform)
print("err_tchebychev =", err_tchebychev)

fig, ax = set_fig()

# plot_f(ax, f, x_plot)
# uni_lagrange_poly.plot(ax, x_plot, "Uniform Lagrange Interpolation of f")
# tchebychev_lagrange_poly.plot(ax, x_plot, "Tchebychev Lagrange Interpolation␣
 ↪of f")

y_uni_plot = [func_err_uniform(x) for x  in x_plot]
ax.plot(x_plot, y_uni_plot, label="Error of Uniform Lagrange Interpolation of␣
 ↪f")

y_tche_plot = [func_err_tchebychev(x) for x  in x_plot]
ax.plot(x_plot, y_tche_plot, label="Error of Tchebychev Lagrange Interpolation␣
 ↪of f")

ax.legend()
```

```
err_uniform = 4316983.984556286
err_tchebychev = 0.09804635359780245
```

[ ]:
```
<matplotlib.legend.Legend at 0x7ba5d55d2a10>
```


Plotting of Interpolations of f
1e13
Error of Uniform Lagrange Interpolation of f
Error of Tchebychev Lagrange Interpolation of f
3.0
2.5
2.0
1.5
1.0
0.5
0.0
y
−10
−5
0
5
10
15
20
x