Review. The Newton form of the polynomial interpolating $f(x)$ at $x = x_0, x_1, \ldots$ is

$$f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + f[x_0, x_1, x_2, x_3](x - x_0)(x - x_1)(x - x_2) + \ldots$$

Comparing this to the Taylor expansion of $f(x)$ at $x = x_0$, which is

$$f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2!}f''(x_0)(x - x_0)^2 + \frac{1}{3!}f'''(x_0)(x - x_0)^3 + \ldots,$$

it is not surprising that, as we showed, $f[x_0, x_1, \ldots, x_n] = \frac{1}{n!}f^{(n)}(\xi)$ for some ξ between the x_i.

Recall that, if $P_n(x)$ is the Taylor polynomial of order n, then the error term is $\frac{1}{(n+1)!}f^{(n+1)}(\xi)(x - x_0)^{n+1}$.

Likewise, if $P_n(x)$ is the interpolating polynomial for $f(x)$ at $x_0, x_1, \ldots, x_n$, then

$$f(x) = P_n(x) + \underbrace{\frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)(x - x_1)\cdots(x - x_n)}_{\text{error term}}$$

for some ξ between the x_i and x.

Example 85. Suppose we approximate a function $f(x)$ by the polynomial $P(x)$ interpolating it at $x = -1, -\frac{1}{3}, \frac{1}{3}, 1$. Suppose that we know that $|f^{(n)}(x)| \leqslant n$ for all $x \in [-1, 1]$.

(a) Give an upper bound for the error when $x = -\frac{2}{3}$.

(b) Give an upper bound for the error when $x = 0$.

(c) Give an upper bound for the error for all $x \in [-1, 1]$.

Solution. By Theorem 83, the error is

$$f(x) - P(x) = \frac{f^{(4)}(\xi)}{4!}(x + 1)\left(x + \frac{1}{3}\right)\left(x - \frac{1}{3}\right)(x - 1) = \frac{f^{(4)}(\xi)}{4!}(x^2 - 1)\left(x^2 - \frac{1}{9}\right),$$

where ξ is between -1 and 1 (provided that $x \in [-1, 1]$). Since $\frac{1}{4!}|f^{(4)}(\xi)| \leqslant \frac{4}{4!} = \frac{1}{6}$, the error is bounded by

$$|f(x) - P(x)| \leqslant \frac{1}{6}\left|(x^2 - 1)\left(x^2 - \frac{1}{9}\right)\right|.$$

(a) If $x = -\frac{2}{3}$, then this bound becomes $|f(x) - P(x)| \leqslant \frac{1}{6}\left|(x^2 - 1)\left(x^2 - \frac{1}{9}\right)\right| = \frac{1}{6} \cdot \frac{5}{27} \approx 0.0309$.

(b) If $x = 0$, then this bound becomes $|f(x) - P(x)| \leqslant \frac{1}{6}\left|(x^2 - 1)\left(x^2 - \frac{1}{9}\right)\right| = \frac{1}{6} \cdot \frac{1}{9} \approx 0.0185$.

Comment. It is not surprising that this error bound is better than the one for $x = -\frac{2}{3}$ since, roughly speaking, there are more interpolation nodes around 0.

(c) Consider $g(x) = (x^2 - 1)\left(x^2 - \frac{1}{9}\right) = x^4 - \frac{10}{9}x^2 + \frac{1}{9}$. We need to compute $\max_{x \in [-1, 1]} |g(x)|$.

Since $g(\pm 1) = 0$, the maximum value of $|g(x)|$ must be attained at a point where $g'(x) = 0$.

We compute $g'(x) = 4x^3 - \frac{20}{9}x$. Hence $g'(x) = 0$ if $x = 0$ or $x = \pm\frac{\sqrt{5}}{3}$.

Since $|g(0)| = \frac{1}{9}$ and $\left|g\left(\pm\frac{\sqrt{5}}{3}\right)\right| = \frac{16}{81} > \frac{1}{9}$, we conclude that $\max_{x \in [-1, 1]} |g(x)| = \frac{16}{81}$.

Therefore, our bound for the error is $|f(x) - P(x)| \leqslant \frac{1}{6} \max_{z \in [-1, 1]} \left|(z^2 - 1)\left(z^2 - \frac{1}{9}\right)\right| = \frac{1}{6} \cdot \frac{16}{81} \approx 0.0329$.

Example 86. Python We can approximate $\frac{1}{6} \max_{z \in [-1,1]} \left| (z^2-1)\left(z^2-\frac{1}{9}\right) \right| = \frac{1}{6} \cdot \frac{16}{81} \approx 0.0329$ as follows using 100 points.

```
>>> from numpy import linspace
>>> max([1/6*abs((z**2-1)*(z**2-1/9)) for z in linspace(-1,1,100)])
    0.0328984640831
```

Example 87. Python The following code measures how well a function f is approximated by the polynomial interpolating f at the given points. It returns an approximation of the maximal error on the interval $[a, b]$.

```
>>> from numpy import linspace, pi, cos, sin
>>> from scipy import interpolate
>>> def max_interpolation_error(f, a, b, xpoints, nr_sample_points):
        ypoints = [f(x) for x in xpoints]
        poly = interpolate.lagrange(xpoints, ypoints)
        max_error = max([abs(f(x)-poly(x)) for x in linspace(a,b,nr_sample_points)])
        return max_error
```

Let us verify that this works using an example we have discussed before:

```
>>> max_interpolation_error(sin, 0, pi, [0,pi/2,pi], 100)
    0.0560067197786
```

This agrees with the maximal error that we observed at the end of Example 84. Let us look how the error develops as we add more points:

```
>>> [max_interpolation_error(sin, 0, pi, linspace(0,pi,n), 100) for n in range(2,9)]
    [0.99987412767387496, 0.056006719778558423, 0.043613266903306247,
    0.0018097268033398783, 0.0013114413108160916, 3.385907546618605e-05,
    2.4246231325325551e-05]
```

It is pleasing to see that the error decreases. However, as we will see in the next example, this does not have to be the case.

Comment. Note that the error seems to really decrease every second step (i.e. after adding two more points). Can you offer an explanation for what might be the cause of this?

Example 88. Python However, this is not the end of the story. It turns out that the interpolation error does not always go down if we add additional points.

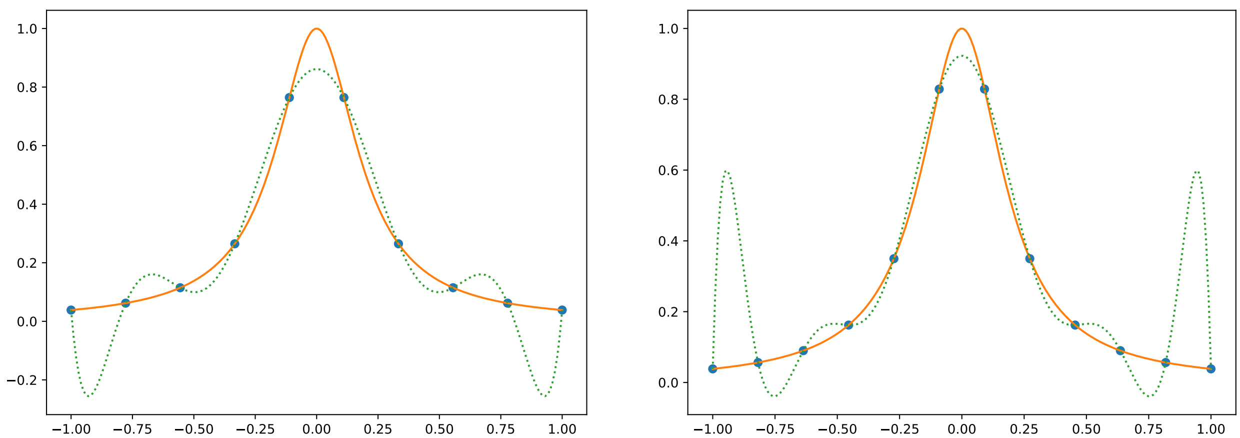
```
>>> def f(x):
        return 1/(1+25*x**2)

>>> [max_interpolation_error(f, -1, 1, linspace(-1,1,n), 100) for n in range(2,18)]

[0.9589941912351845, 0.6459699748665507, 0.7044952736346626, 0.4382728746134098,
0.43032461596244886, 0.6164015686420344, 0.24528527039305037, 1.0450782163781276,
0.297971540151836, 1.9154342696798625, 0.5538529081557272, 3.6117015978042333,
1.064460371610917, 7.189298472061041, 2.0967229089912, 14.013534491466531]
```

The function $f(x) = \frac{1}{1+25x^2}$ in this example is known as the **Runge function** and one can show that, by adding more points, the error grows without bound.

$$\lim_{n\to\infty} \max_{x\in[-,1,1]} |f(x) - P_n(x)| = \infty$$

https://en.wikipedia.org/wiki/Runge%27s_phenomenon

The following plots show the situation using 10 and 12 interpolation nodes.



While the approximation becomes better towards the center of the interval $[-1,1]$, the oscillations towards the ends of the interval become more violant (resulting in an increasing worst-case error).

Homework. Recreate the above plots following Example 78.

In the next section, we will see that we can avoid this issue if we don't choose equally spaced points but carefully chosen ones called **Chebyshev nodes**.