

- ✓ 1. Write a function to compute the polynomial of degree up to n that is the best fit for a given set of points. It should take as its arguments an array of tuples and an integer n , and return an object of type `Polynomial` (developed in the last coding assignment). The function is also expected to produce a plot of the input points along with the best-fit polynomial together.
- ✓ 2. Write a function to compute the polynomial of degree up to n that best approximates the function $\sin(x) + \cos(x)$ in the interval $[0, \pi]$. It should take as its argument an integer n , and return an object of type `Polynomial` (developed in the last coding assignment). The function is also expected to plot the actual and approximate function together.
- ✓ 3. Write a function that uses the enhanced `Polynomial` class (developed in the last coding assignment) to compute the n^{th} *Legendre polynomial*. It should take as its argument an integer n , and return an object of type `Polynomial` that represents the n^{th} *Legendre polynomial*.
- ✓ 4. Write a function to compute the least-square approximation of e^x in the interval $[-1, 1]$ using the first n *Legendre polynomials*. It should take as its argument an integer n , and return an object of type `Polynomial`. Also, plot the actual and approximate function together.
- ✓ 5. Write a function that uses the enhanced `Polynomial` class (developed in the last coding assignment) to compute the n^{th} *Chebyshev polynomial*. It should take as its argument an integer n , and return an object of type `Polynomial` that represents the n^{th} *Chebyshev polynomial*.
- ✓ 6. Write a function to compute the first 5 *Chebyshev polynomials* and numerically demonstrate that they are *orthogonal* with respect the weight function $w(x) = 1/\sqrt{1-x^2}$ in the interval $[-1, 1]$.
- ✓ 7. Write a function to compute coefficients of the best-fit *Fourier approximation* $S_n(x)$ of function e^x in the interval $[-\pi, \pi]$. It should take as its arguments an integer n , and print the coefficients of $S_n(x)$. The function is also expected to generate a plot with the actual and approximate function together.
8. Using Python's `scipy.fft` package, write a function, with time complexity of $O(n \log n)$, to multiply two large n -digit integers.
plot fft, traditional and the difference