Taylor Expansion

In this handout we will create, and experience with, a code that uses Taylor series to approximate a mathematical function in an interval around a point x_0. It can be shown that any function that is infinitely differentiable ($f(x) \in C^\infty$) is approximated by the power series

$$f(x) = \sum_{i=0}^{\infty} \frac{f^n(x_0)}{n!}(x - x_0)^n \tag{1}$$

where $f^n(x_0)$ is the n^{th} derivative of f at $x = x_0$. For most functions of interest in physics, the infinite sum is equal to the function itself. It is most common, however, to approximate the function with the first n terms of the sum when the researcher is interested in the local behavior of $f(x)$.

We will explore visually and quantitatively the accuracy of the approximation with the truncated sum for $f(x) = e^x$.

Step by step activity

1. Write a python script that plots the mathematical function $f(x) = e^x$, tabulated in the interval $x \in [-1, 3]$.

2. Add a python function that takes an array of x values in input and returns the Taylor series up to the third order (included, the one with x^3) for a small interval around $x_0 = 0$.

3. Plot the 3^{rd} order Taylor series in the graph, add a legend, and include labels for the axes.

Step by step activity: somewhat more challenging

1. Modyfy the python function to take the Taylor order as an integer input parameter and to return the Taylor series up to the requested term (included).

2. Add to the graph several Taylor series of different order (e.g., 1, 2, 3, 4, and 5). Make sure the legend clearly indicates which line correspond to each order.

3. Add a second panel to your figure where you plot the so-called residual (i.e., the difference between the mathematical function and the Taylor series approximation) for the degrees that you calcualted.

4. Modify your python function again to allow for x_0 to take any desired value.

Step by step activity: even more challenging

1. Try to repeat the somewhat challenging activity above with a function of your choice. $f(x) = \sin(x)$ and $f(x) = \cos(x)$ are doable because their n^{th} derivative can easily be calculated.

2. Alternatively, pick any mathematical function and repeat the initial activity. The idea here is that if you have a pre-decided order of the series, you can differentiate manually your selected mathematical function up to the desired degree.

Step by step activity: really challenging

1. Writing a general Taylor code would require a python function that can calculate the n^{th} derivative of an arbitrary function. Try to accomplish that.
   Full disclosure: I tried for a half day and was not able to write something that is not disrupted by numeric noise after the 5^{th} *derivative. The challenge is on!*