Physik auf dem Computer **SS 2016**

Worksheet 4: Interpolation

May 4th, 2016

General Remarks

- The deadline for handing in the worksheets is **Tuesday, May 10th, 2016, 10:00**.
- This is a short worksheet. You can achieve a maximum of 5 points.
- To hand in your solutions, send an email to `mkuron@icp.uni-stuttgart.de`.
- Please try to only hand in a single file that contains your program code for all tasks. If you are asked to answer questions, you should do so in a comment in your code file or a text block in your IPython notebook. If you need to include an equation or graph, you can do that in your IPython notebook, or you may hand in a separate PDF document with all your answers, graphs and equations.

Throughout the worksheet, the following functions are used on the specified domains:

Name	**Definition**	**Domain**
Sine Function	$f(x) = \sin x$	$[0, 2\pi]$
Runge Function	$g(x) = \frac{1}{1+x^2}$	$[-5, 5]$
Lennard-Jones Function	$h(x) = x^{-12} - x^{-6}$	$[1, 5]$

The Python program `ws4.py` and the IPython Notebook `ws4.ipynb` demonstrate how to create plots of the functions and their interpolating polynomials using SciPy's functions.

Task 4.1: Interpolating Polynomials (5 points)

In this task, you should implement the interpolating polynomials.

- **4.1.1** (3 points) Implement your own class `NewtonInterpolation` that does the interpolation using Newton's representation. The class shall define two methods:
  - `__init__(self,x,y)` does the same as `neville()` from the lecture script and stores the x and γ in class variables
  - `__call__(self,x)` does the same as `horner()` from the lecture script but does not require the parameters `gamma` and `x` but uses the stored class variables instead

  Redo the plots from the demo using this new class.
- **4.1.2** (2 points) Modify the Python program from the previous tasks such that it optionally computes the interpolating polynomials at the Chebyshev nodes. Add curves for the interpolating polynomials with Chebyshev nodes to the graphs you created above.