Übungsserie 4

Erstellen Sie für ihre manuelle Lösung für die Aufgabe 1 eine PDF-Datei *Name_S4_Aufg1.pdf* und fassen Sie diese mit Ihren Python-Skripts zu den Aufgaben 2 und 3 in einer ZIP-Datei *Name_S4.zip* zusammen. Laden Sie dieses File vor der Übungsstunde nächste Woche auf Moodle hoch.

Aufgabe 1 (ca. 40 Minuten):

Bei der Übermittlung von Daten eines Wettersatelliten über den Atmosphärendruck als Funktion der Höhe sind Fehler aufgetreten, so dass die in der folgenden Messreihe mit *NaN* ("Not a Number") bezeichneten Werte geschätzt werden müssen. Verwenden Sie hierzu die Lagrange-Interpolation und geben Sie Ihre manuelle Lösung als *Name_S4_Aufg1.pdf* ab.

Höhe über Meer [m]	0	2500	3750	5000	10000
Atmosphärendruck [hPa]	1013	747	***NaN***	540	226

Aufgabe 2 (ca. 40 Minuten):

Implementieren sie die Lagrange-Interpolation in Python in eine Funktion `y_int = lagrange_int(x,y,x_int)`. Die Vektoren `x` und `y` beschreiben die vorgegebenen Koordinaten (x_i, y_i) der Datenpunkte, `x_int` ist die $x-$Koordinate des Punktes (x_{int}, y_{int}), dessen $y-$Koordinate mittels Lagrange-Interpolation interpoliert werden soll, `y_int` wird als Resultat zurückgegeben. Überprüfen Sie mit Ihrere Funktion die berechneten Resultate aus Aufgabe 1.

Aufgabe 3 (ca. 40 Minuten):

Die Funktion `numpy.polyfit()` (siehe Onlinedokumentation) kann verwendet werden, um die Koeffizienten des Interpolations-Polynoms vom Grad n durch $n+1$ Datenpunkte zu berechnen. Das verwendete Verfahren ist die Methode der kleinsten Quadrate der linearen Ausgleichsrechnung, die wir demnächst kennen lernen werden. Die Funktion `numpy.polyval()` (siehe Onlinedokumentation) wird dazu verwendet, die Funktionswerte eines Polynoms zu berechnen, dessen Koeffizienten bekannt sind.

Betrachten Sie nun den Anteil der Haushalte in den USA, die einen Computer besitzen, über den Verlauf der Zeit:

Jahr	1981	1984	1989	1993	1997	2000	2001	2003	2004	2010
Haushalte mit Computer [%]	0.5	8.2	15	22.9	36.6	51	56.3	61.8	65	76.7

a) Bestimmen Sie mit `polyfit()` die Koeffizienten des Interpolationspolynoms (welches exakt durch jeden Datenpunkt gehen sollte) und plotten Sie die Zeitreihe sowie die Werte des Polynoms unter Verwendung von `numpy.polyval()` in einer geeigneten Auflösung der $x-$Achse (z.B. 0.1 Jahre) auf dem Intervall $x \in [1975, 2020]$ und $y \in [-100, 250]$. Geht das so berechnete Polynom wirklich exakt durch alle Datenpunkte? Schreiben Sie die Antwort als Kommentar in Ihr Skript.

b) Wiederholen Sie die Aufgabe a), diesmal subtrahieren Sie aber von den Jahreszahlen zuerst den Mittelwert (also `x-x.mean()`), bevor Sie `polyfit()` ausführen, und plotten das Polynom dann für die ursprünglichen Jahreszahlen $x \in [1975, 2020]$ und $y \in [-100, 250]$ mit der gleichen Auflösung wie bei a). Was stellen Sie im Vergleich zu a) fest?

c) Was ist der Schätzwert für 2020, basierend auf Ihren Resultaten aus b)? Ist das realistisch, und können solche Polynome hoher Ordnung für Schätzwerte ausserhalb des Intervalls der vorhandenen Datenwerte benutzt werden? Schreiben Sie die Antwort als Kommentar in Ihr Skript.

d) Benutzen Sie Ihre Funktion aus Aufgabe 2 und erweitern Sie sie so, dass als `x_int` ein Vektor (nicht nur ein Wert) akzeptiert wird. Erzeugen Sie eine neue Abbildung, indem Sie die mit der Lagrange-Interpolation berechneten Werte für $x \in [1981, 2010]$ plotten, mit der gleichen Auflösung der $x-$Achse wie bei b), und $y \in [-100, 250]$. Plotten Sie in diese Abbildung auch noch mal das Interpolationspolynom aus Aufgabe b). Was stellen Sie im Vergleich der beiden Methoden fest? Schreiben Sie die Antwort als Kommentar in Ihr Skrip.