

Übung 7b: Python für aufwendige Berechnungen (I)

Ziel der heutigen Übung ist das Erlernen der Grundlegenden Vorgehensweise bei der Performance-Analyse durch Laufzeitmessung und Profiling. Auf Basis der Messergebnisse werden die kritischen Stellen identifiziert und die Laufzeit durch die Anwendung der im Vortrag beschriebenen Strategien verbessert. Als Beispiel dient die Lösung der Laplace-Gleichung $\nabla^2 u = 0$ mittels eines expliziten iterativen Lösungsansatzes bei dem ausgehend von einer vorherigen Lösung eine neue Lösung durch die direkte Diskretisierung der elliptischen DGL berechnet wird.

Übungsaufgabe 1: Laufzeitmessung und Profiling (aufgabe1.py)

1. Fügen Sie vor und nach Stellen im Code die Ihnen geeignet erscheinen Befehle zur Zeitmessung ein (`time.time()`), um die benötigte Laufzeit für eine Iteration und die Gesamtlaufzeit für 100 Iterationen zu messen.
2. Erstellen Sie ein Profil der ursprünglichen `aufgabe1.py` mittels des Befehls `python -m cProfile aufgabe1.py`.
3. Filtern Sie die Ausgabe nach den relevanten Daten der `aufgabe1.py`.

Übungsaufgabe 2: Numpy und Weave (aufgabe2.py)

1. Schreiben Sie in der Klasse `NumpyLaplaceSolver` die Funktion `step` mittels der in `numpy` verfügbaren Operationen
2. Messen Sie mittels `time` und `cProfile` die Laufzeit der reimplementierten `step`-Funktion und Vergleichen die Ergebnisse mit denen aus Aufgabe 1.
3. Wenden Sie `weave.blitz` auf den `numpy`-Ausdruck der `step`-Funktion an.
4. Messen Sie die Laufzeiten für verschiedene Iterationszahlen und Gebietsgrößen (`n_iter`, `nx`, `ny`) und Vergleichen Sie die Ergebnisse.

Übungsaufgabe 3: Cython (aufgabe3.py)

1. Was wird in der Datei `cython_update.pyx` gemacht?
2. Messen Sie die Laufzeit von `aufgabe3.py`.