1 taumain.py

Die Klasse "dataThread" dient zum Auslesen der Daten aus der tauhost.c Datei. Weiterhin benutzen wir die "animThread"-Klasse, um die Werte, die wir aus "dataThread" bekommen, zu plotten. Zur Datenverarbeitung haben wir einen Eventhandler zwischen "animThread" und "dataThread" benutzt. Mit den presets übergeben wir der tauhost-Datei, die für die verschiedenen Potentiale spezifischen Werte. Dann kommen die benutzten Variablen:

- n: Anzahl der Punkte
- deltat: Zeitunterschied zwischen zwei Punkten
- deltatau: Entwicklung der Liste über die fiktive Zeit tau
- h: Parisi-h
- parisi: Wird der Parisi-Trick benutzt oder nicht
- entw: Anzahl der Entwicklungsschritte
- c: eine momentan nicht benutzte Konstante
- device: ID des Geräts, das benutzt werden soll
- rpf: Mit rpf=1 werden in jedem Loop die Daten übergeben
- intime: Wie viele Loops laufen, bevor in fsum und fhsum geschrieben wird
- loops: Loops pro Entwicklungsschritt
- inputf: Möglichkeit der Dateieingabe
- outputf: In welche Datei wird geschrieben
- acco: Genauigkeit der geschrieben Daten

Das Unterprogramm mit den zuvor definierten Parametern wird ausgeführt in Zeile 154.

2 tauhost.c

- bis Zeile 41: Parameter werden eingelesen
- Zeilen 47-50: Speicherplatz wird reserviert
- Zeile 56: Wenn deltatau sich ändert, wird deltatau nicht überschrieben, sondern in deltautmp geschrieben
- Zeilen 77-79: es werden mit Hilfe der Box-Müller-Transformation zwei gaußverteilte Zufallszahlen zwischen 0 und 1 erzeugt, damit wird omega zum ersten Mal berechnet

- Zeilen 82-172: Inputverarbeitung, falls Datei als Input genutzt wird
- Zeilen 174-192: Daten werden initialisiert (Seeds für Zufallszahlengeneration werden erstellt)
- Zeilen 205-255: Infos über das benutzte Gerät
- Zeilen 264-319: Speicherbuffer auf dem Gerät wird kreiert
- Zeilen 324-377: Werte werden das erste Mal in den Buffer geschrieben
- Zeile 381: Programm wird kreiert
- Zeile 403: Kernel erstellt
- Zeilen 417-433: Argumente werden an Kernel übergeben
- Zeile 481: Beginn des Hauptloops
- Zeile 483: Kernel ausführen
- Zeile 485: Synchronisierung
- Zeile 487: Beginn zweiter Loop (Wertausgabe)
- Zeile 534: Wert für stable wird ausgelesen und der Kernel meldet, ob die Berechnung stabil verlief oder nicht
- Zeilen 547f: Wert wird fsum und fhsum hinzugefügt
- Zeile 563: Wenn die Berechnung nicht stabil verlief, dann wird deltatau aus dem Kernel ausgelesen, in deltatautmp geschrieben und das neu berechnete deltatau wird wieder an den Kernel zurückgegeben
- Zeile 600f: f-Werte aus dem host-Programm werden zurück in den Kernel geschrieben
- Zeilen 614-630: Möglichkeit in Output-Datei zu schreiben
- Zeilen 636-666: Speicherplatz wird freigegeben