Geophysical Research Abstracts
Vol. 15, EGU2013-12637, 2013
EGU General Assembly 2013

Influence of intrinsic and extrinsic forces on 3D stress distribution using CUDA programming

Ludovic Räss, Samuel Omlin, and Yuri Podladchikov
Unil, Switzerland (ludovic.raess@unil.ch)

In order to have a better understanding of the influence of buoyancy (intrinsic) and boundary (extrinsic) forces in a nonlinear rheology due to a power law fluid, some basics needs to be explored through 3D numerical calculation. As first approach, the already studied Stokes setup of a rising sphere will be used to calibrate the 3D model. Far field horizontal tectonic stress is applied to the sphere, which generates a vertical acceleration, buoyancy driven. This simple and known setup allows some benchmarking performed through systematic runs. The relative importance of intrinsic and extrinsic forces producing the wide variety of rates and styles of deformation, including absence of deformation and generating 3D stress patterns, will be determined. Relation between vertical motion and power law exponent will also be explored.

The goal of these investigations will be to run models having topography and density structure from geophysical imaging as input, and 3D stress field as output. The stress distribution in Swiss Alps and Plateau and its implication for risk analysis is one of the perspective for this research. In fact, proximity of the stress to the failure is fundamental for risk assessment. Sensitivity of this to the accurate topography representation can then be evaluated.

The developed 3D numerical codes, tuned for mid-sized cluster, need to be optimized, especially while running good resolution in full 3D. Therefor, two largely used computing platforms, MATLAB and FORTRAN 90 are explored. Starting with an easy adaptable and as short as possible MATLAB code, which is then upgraded in order to reach higher performance in simulation times and resolution. A significant speedup using the rising NVIDIA CUDA technology and resources is also possible. Programming in C-CUDA, creating some synchronization feature, and comparing the results with previous runs, helps us to investigate the new speedup possibilities allowed through GPU parallel computing. These codes justify then to build-up low-cost GPU clusters, using a large number of gaming cards in parallel, so Cost - Performances ratio will be very interesting.