GPU implementation of multigrid solver of Stokes equation with strongly variable viscosity

Liang ZHENG[1,4] Taras Gerya[2] Matthew Knepley[3] David A. Yuen[4] Huai ZHANG[1] Yaolin SHI[1]

[1] Key laboratory of Computational Geodynamics, Chinese Academy of Sciences, China
[2] Institute of Geophysics, ETH-Zurich , Switzerland
[3] Computational Institute, University of Chicago, Chicago, Illinois, U.S.
[4] Minnesota Supercomputing Institute, University of Minnesota. Minnesota, U.S.

Berkeley, Jan 2011

Figure: I am Liang (Larry) ZHENG!

Outline

Why do geoscientists study the Stokes flow problem?

1. The solid earth deforms slowly and behaves as viscous fluid over geological time so geoscientists can study the earth using fluid dynamic methods with the basic principles of conservation.

Why do geoscientists study the Stokes flow problem?

1. The solid earth deforms slowly and behaves as viscous fluid over geological time so geoscientists can study the earth using fluid dynamic methods with the basic principles of conservation.
2. Stokes flow

Why do geoscientists study the Stokes flow problem?

1. The solid earth deforms slowly and behaves as viscous fluid over geological time so geoscientists can study the earth using fluid dynamic methods with the basic principles of conservation.
2. Stokes flow
   - Stokes flow is also named creeping flow which Reynolds number is close to Zero . . .

Why do geoscientists study the Stokes flow problem?

1. The solid earth deforms slowly and behaves as viscous fluid over geological time so geoscientists can study the earth using fluid dynamic methods with the basic principles of conservation.
2. Stokes flow
   - Stokes flow is also named creeping flow which Reynolds number is close to Zero . . .
   - Advective Inertial Forces are small compared with viscous forces

Why do geoscientists study the Stokes flow problem?

1. The solid earth deforms slowly and behaves as viscous fluid over geological time so geoscientists can study the earth using fluid dynamic methods with the basic principles of conservation.
2. Stokes flow
   - Stokes flow is also named creeping flow which Reynolds number is close to Zero . . .
   - Advective Inertial Forces are small compared with viscous forces
   - Geodynamic Processes: Mantle Convection , Lithospheric Deformation, Lava Flow . . .

Stokes equations

Stokes equations

Mass Conservation

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{1}$$

Stokes equations

Mass Conservation

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{1}$$

Momentum Conservation

$$\frac{\partial \sigma'_{ij}}{\partial x_j} - \frac{\partial P}{\partial x_i} + \rho g_i = 0 \tag{2}$$

Stokes equations

Mass Conservation

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{1}$$

Momentum Conservation

$$\frac{\partial \sigma'_{ij}}{\partial x_j} - \frac{\partial P}{\partial x_i} + \rho g_i = 0 \tag{2}$$

this makes it a saddle point system , which is difficult to solve.

Stokes equations

Mass Conservation

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{1}$$

Momentum Conservation

$$\frac{\partial \sigma'_{ij}}{\partial x_j} - \frac{\partial P}{\partial x_i} + \rho g_i = 0 \tag{2}$$

this makes it a saddle point system , which is difficult to solve.

Viscous Constitutive Relationship

$$\sigma'_{ij} = 2\mu\,\dot{\varepsilon}_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{3}$$

Strongly Variable-viscosity

Strongly Variable-viscosity

Effective viscosity depends on environmental parameters

$$\mu_{eff} \propto exp\left(\frac{E_a + V_a P}{nRT}\right) \tag{4}$$

Strongly Variable-viscosity

Effective viscosity depends on environmental parameters

$$\mu_{eff} \propto exp\left(\frac{E_a + V_a P}{nRT}\right) \tag{4}$$

1. These strongly variable viscosity phenomenons are commonplace in geodynamic processes. For example the contrast of viscosity of surface conditions of the earth and upper mantle reaches $\frac{\mu_1}{\mu_0} \sim 10^{10}$ for the huge difference of temperature.

Strongly Variable-viscosity

Effective viscosity depends on environmental parameters

$$\mu_{eff} \propto exp\left(\frac{E_a + V_a P}{nRT}\right) \tag{4}$$

1. These strongly variable viscosity phenomenons are commonplace in geodynamic processes. For example the contrast of viscosity of surface conditions of the earth and upper mantle reaches $\frac{\mu_1}{\mu_0} \sim 10^{10}$ for the huge difference of temperature.
2. Because coupled with continuum equation Stokes flow problem is hard to solve even with constant viscosity for the saddle-point problems. What's worse is with the strongly variable coefficients the matrix becomes very ill-conditioned and there are not many proven methods for solving it.

Methods used on CPU

Methods used on CPU

1. A lot of numerical methods have been applied to solve Stokes equations on CPU before the birth of GPU including finite difference, finite volume and finite element method.

Methods used on CPU

1. A lot of numerical methods have been applied to solve Stokes equations on CPU before the birth of GPU including finite difference, finite volume and finite element method.
2. Coupled with mass equation the Stokes equation becomes the saddle problem that some special iterative methods are there for solving it such as the multigrid method and preconditioned Krylov subspace methods, etc.

Methods used on CPU

1. A lot of numerical methods have been applied to solve Stokes equations on CPU before the birth of GPU including finite difference, finite volume and finite element method.
2. Coupled with mass equation the Stokes equation becomes the saddle problem that some special iterative methods are there for solving it such as the multigrid method and preconditioned Krylov subspace methods, etc.
3. Multigrid method can speed up the iteration because of the fast convergence of the part of high frequency residuals. Generally multigrid method can be used as a preconditioner of Krylov subspace method which we want to implement later.

Strategies on GPU

Strategies on GPU

1. On GPU we used geometric multigrid (GMG) coupled with Red-Black updating method to solve the Stokes equations. The V-cycle GMG is showing as figure 2 which has two part:
   - '–' represents the restriction;
   - '-.' represents the prolongation.

Strategies on GPU

1. On GPU we used geometric multigrid (GMG) coupled with Red-Black updating method to solve the Stokes equations. The V-cycle GMG is showing as figure 2 which has two part:
   - ’–’ represents the restriction;
   - ’-.’ represents the prolongation.
2. Using Red-Black Gauss-Seidel (RBGS) iteration technology for the smoother can avoid the disordered threads when executing the GPU kernels. Figure 3 shows the RBGS technology which can be divided into two parts:
   - Set the boundary condition and ghost points around the nodal points (blue points);
   - Update the red and black points in different kernels (red and blue points).

GMG and RBGS

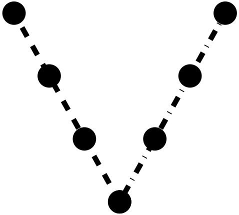


Figure: V-cycle mulgrid

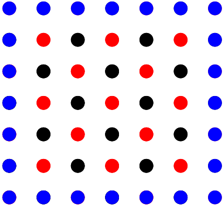


Figure: Red-Black Gauss-Seidel

Testing Model

We set a testing model showing as figure 4 to check the solver's ability of dealing with variable viscosity in which the viscosity has a contrast of 10^6. All the boundary nodes are set with free slip velocity so the force of this model is only gravity. 2D and 3D codes are implemented on GPU to compute the result of the testing model.

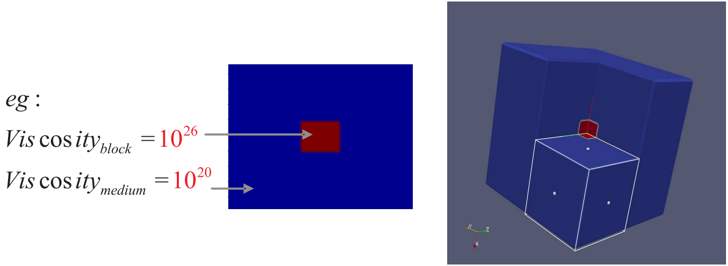


Figure: Testing model

2D version

1. Conservative finite difference methods and staggered stencil are used.
2. A disordered smoother's performance is actually between the Jacobi and Gauss-Seidel iteration.
3. We compare the codes without multigrid and the 2-level grids under 128*128 resolution.

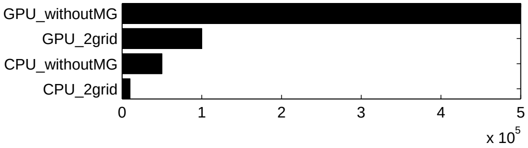


Figure: Comparison of iterations on CPU and GPU

2D version

We also compared the running time on different GPUs including Tesla 1060c (GT200) and GTX 480 (Fermi) architecture with the same cycles (10^5 iterations).

Table: Comparison of different platforms with different precision

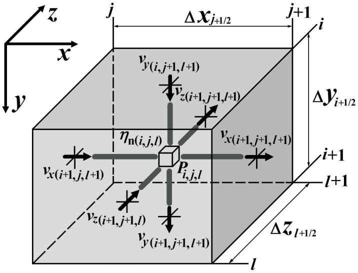
Platform	Single Precision	Double Precision
GPU(Tesla 1060C)	303 sec	619 sec
GPU(GTX 480)	160 sec	245 sec

3D version

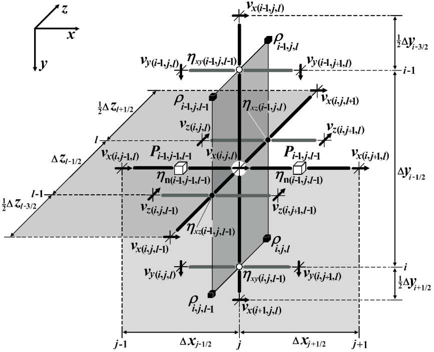
Stokes equation in 3D

$$\begin{cases} \dfrac{\partial\left(2\mu\frac{\partial u_x}{\partial x}\right)}{\partial x} + \dfrac{\partial\left(\mu\left(\frac{\partial u_x}{\partial y}+\frac{\partial u_y}{\partial x}\right)\right)}{\partial y} + \dfrac{\partial\left(\mu\left(\frac{\partial u_x}{\partial z}+\frac{\partial u_z}{\partial x}\right)\right)}{\partial z} - \dfrac{\partial P}{\partial x} + \rho g_x = 0 \\ \dfrac{\partial\left(\mu\left(\frac{\partial u_x}{\partial y}+\frac{\partial u_y}{\partial x}\right)\right)}{\partial x} + \dfrac{\partial\left(2\mu\frac{\partial u_y}{\partial y}\right)}{\partial y} + \dfrac{\partial\left(\mu\left(\frac{\partial u_y}{\partial z}+\frac{\partial u_z}{\partial y}\right)\right)}{\partial z} - \dfrac{\partial P}{\partial y} + \rho g_y = 0 \\ \dfrac{\partial\left(\mu\left(\frac{\partial u_x}{\partial z}+\frac{\partial u_z}{\partial x}\right)\right)}{\partial x} + \dfrac{\partial\left(\mu\left(\frac{\partial u_z}{\partial y}+\frac{\partial u_y}{\partial z}\right)\right)}{\partial y} + \dfrac{\partial\left(2\mu\frac{\partial u_x}{\partial x}\right)}{\partial z} - \dfrac{\partial P}{\partial z} + \rho g_z = 0 \\ \dfrac{\partial u_x}{\partial x} + \dfrac{\partial u_y}{\partial y} + \dfrac{\partial u_z}{\partial z} = 0 \end{cases} \tag{5}$$

Discrete scheme with staggered grid



Mass conservation equation



X-stokes equation

Figure: Discrete scheme with staggered grid

Index Macros for staggered grid

It's difficult to use multi-dimensional array in global memory on GPU. Instead we defined a series of macros to translate 3D to 1D array:

```
#define vx(i,j,k) vx[(i-1)+(j-1)*(ynum+1)+(k-1)*(xnum)*(ynum+1)]
#define RX(i,j,k) RX[(i-1)+(j-1)*(ynum+1)+(k-1)*(xnum)*(ynum+1)]
#define vy(i,j,k) vy[(i-1)+(j-1)*(ynum)+(k-1)*(xnum+1)*(ynum)]
#define RY(i,j,k) RY[(i-1)+(j-1)*(ynum)+(k-1)*(xnum+1)*(ynum)]
#define vz(i,j,k) vz[(i-1)+(j-1)*(ynum+1)+(k-1)*(xnum+1)*(ynum+1)]
#define RZ(i,j,k) RZ[(i-1)+(j-1)*(ynum+1)+(k-1)*(xnum+1)*(ynum+1)]
#define pr(i,j,k) pr[(i-1)+(j-1)*(ynum-1)+(k-1)*(xnum-1)*(ynum-1)]
#define RC(i,j,k) RC[(i-1)+(j-1)*(ynum-1)+(k-1)*(xnum-1)*(ynum-1)]
```

RBGS on GPU

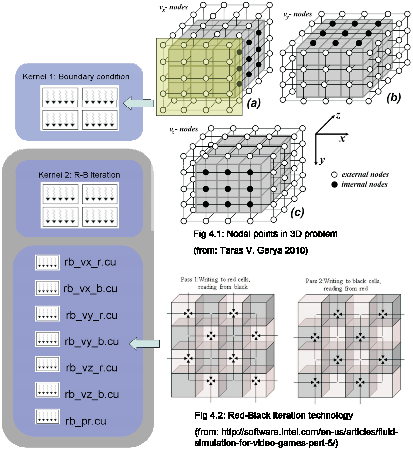


Figure: RBGS on GPU

kernel codes: taking x-direction velocity in red points for example

```
#include "Index.h"

__global__ void rb_vx_r( ... )
{
  int i=blockIdx.x;
  int j=blockIdx.y;
  int k=threadIdx.x;

  i+=2;j+=2;k+=2;
  //+2 means start from 1 and skip the boundary points
  if((i+j+k)%2!=0) return;
  //decide if it's the red nodes

  double resxcur,kfxcur;

  resxcur=RX(i,j,k)+(pr(i-1,j,k-1)-pr(i-1,j-1,k-1))/xstp;
  resxcur=resxcur-(xkf2*(etan(i-1,j,k-1)*(vx(i,j+1,k)-vx(i,j,k))-etan(i-1,j-1,k-1)*(
        vx(i,j,k)-vx(i,j-1,k))));
  resxcur=resxcur-(etaxy(i,j,k-1)*(ykf*(vx(i+1,j,k)-vx(i,j,k))+xykf*(vy(i,j+1,k)-vy(i
        ,j,k)))-etaxy(i-1,j,k-1)*(ykf*(vx(i,j,k)-vx(i-1,j,k))+xykf*(vy(i-1,j+1,k)-vy(
        i-1,j,k))));
  resxcur=resxcur-(etaxz(i-1,j,k)*(zkf*(vx(i,j,k+1)-vx(i,j,k))+xzkf*(vz(i,j+1,k)-vz(i
        ,j,k)))-etaxz(i-1,j,k-1)*(zkf*(vx(i,j,k)-vx(i,j,k-1))+xzkf*(vz(i,j+1,k-1)-vz(
        i,j,k-1))));
  kfxcur=-xkf2*(etan(i-1,j,k-1)+etan(i-1,j-1,k-1))-ykf*(etaxy(i,j,k-1)+etaxy(i-1,j,k
        -1))-zkf*(etaxz(i-1,j,k)+etaxz(i-1,j,k-1));
  vx(i,j,k)=vx(i,j,k)+resxcur/kfxcur*krelaxs;
}
```

Result of 3D codes

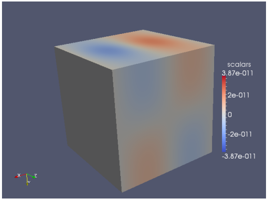


Figure: Velocity-X Unit: m/s

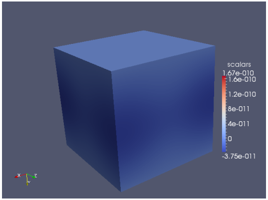


Figure: Velocity-Y Unit: m/s

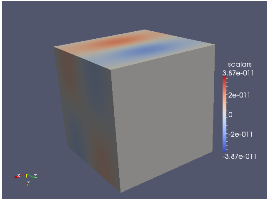


Figure: Velocity-Z Unit: m/s

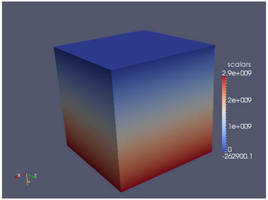


Figure: Pressure Unit: Pa

Comparison of time and iterations on different platforms

Table: Comparison of time and iterations on different platforms

Platform	Model: 32*32*32	Model: 64*64*64
CPU iterations	85	141
GPU iterations	148	106
CPU time	10min	96 min
GPU time	8min	31 min
Speedup	1.2x	3.1x

Comparison of speedup on different models

Table: Comparison of speedup on different models

Model	CPU Time	GPU time	Speedup
32*32*32	1.2s	0.3s	4.0x
64*64*64	10.5s	2.2s	4.8x
128*128*128	101.6s	21.4s	4.7x
256*256*256	787.3s	202.8s	3.9x

Conclusion and future

Conclusion and future

1. We have finished the preliminary implementation of GPU based multigrid solver for Stokes equation with strongly variable viscosity.It has increased the performance of the Matlab codes on CPU. But it's only a start and we still need to apply some technologies to improve the codes.

Conclusion and future

1. We have finished the preliminary implementation of GPU based multigrid solver for Stokes equation with strongly variable viscosity.It has increased the performance of the Matlab codes on CPU. But it's only a start and we still need to apply some technologies to improve the codes.
2. Future

Conclusion and future

1. We have finished the preliminary implementation of GPU based multigrid solver for Stokes equation with strongly variable viscosity.It has increased the performance of the Matlab codes on CPU. But it's only a start and we still need to apply some technologies to improve the codes.
2. Future
   - We are considering deploy it as a preconditioner using DA class in PETSc framework . . .

Conclusion and future

1. We have finished the preliminary implementation of GPU based multigrid solver for Stokes equation with strongly variable viscosity.It has increased the performance of the Matlab codes on CPU. But it's only a start and we still need to apply some technologies to improve the codes.
2. Future
   - We are considering deploy it as a preconditioner using DA class in PETSc framework . . .
   - Try to run it with MPI on GPU cluster . . .

Conclusion and future

1. We have finished the preliminary implementation of GPU based multigrid solver for Stokes equation with strongly variable viscosity.It has increased the performance of the Matlab codes on CPU. But it's only a start and we still need to apply some technologies to improve the codes.
2. Future
   - We are considering deploy it as a preconditioner using DA class in PETSc framework . . .
   - Try to run it with MPI on GPU cluster . . .
   - Use it to model the real large-scale geodynamic problems is our object.

The End

Thank you! Any Questions?

Figure: I am Liang (Larry) ZHENG!