A GPU Accelerated PCG Pressure Projection Solver on Dynamic Adaptive Grids

Austen Duffy and Mark Sussman

Department of Mathematics, Florida State University

February 26, 2010

Objectives

Goal: Accelerate the pressure projection step of a Coupled Level Set and Volume of Fluid (CLSVOF) code in order to reduce the overall flow solution time. To this end, we will examine

- Multigrid (MG) as a preconditioner for the Conjugate Gradient (CG) method
- The use of appropriate smoothers for the MG preconditioner
- GPU acceleration of the MG smoothers

Pressure Projection Step

Solve $\quad \nabla \cdot \frac{1}{\rho} \nabla p = f \quad \rightarrow \quad Ax = b$

$$
\mathrm{A} = \begin{vmatrix}
a & b & & & d & & & & \\
c & a & b & & & d & & & \\
 & c & a & b & & & . & & \\
 & & . & . & . & & & . & \\
e & & & . & . & . & & & d \\
 & e & & & . & . & . & & \\
 & & . & & & c & a & b & \\
 & & & . & & & c & a & b \\
 & & & & e & & & c & a
\end{vmatrix}
$$

$a = -(\beta_{i+1/2,j} + \beta_{i-1/2,j} + \beta_{i,j+1/2} + \beta_{i,j-1/2})$
$b = \beta_{i,j+1/2}$, $c = \beta_{i,j-1/2}$, $d = \beta_{i+1/2,j}$, $e = \beta_{i-1/2,j}$

The Preconditioned Conjugate Gradient Method

To iteratively solve the sparse linear system $Ax = b$ with preconditioning matrix M, we use the PCG algorithm

Given x^0, M
Compute $r^0 = Ax^0 - b$, Solve $Mz^0 = r^0$, Set $p^0 = -z^0$
for $k = 0, Maxiterates$ do
  $\alpha^k = (r^k, r^k)/(p^k, Ap^k)$
  $x^{k+1} = x^k + \alpha^k p^k$
  $r^{k+1} = r^k + \alpha^k Ap^k$
  Enfore solvability condition by projecting r^{k+1} into the range of A. $\sum_K r_K^{k+1} = 0$. (Neumann BC on all walls)
  Solve $Mz^{k+1} = r^{k+1}$
  $\beta^{k+1} = (r^{k+1}, z^{k+1})/(r^k, z^k)$
  $p^{k+1} = -z^{k+1} + \beta^{k+1} p^k$
end for

The Multigrid Preconditioned Conjugate Gradient Method (MGPCG)- History

The MGPCG method was first introduced by Tatebe*, who used a Gauss Seidel Red-Black smoother.

- Works well even when the problem is ill-conditioned
- Reported 5x speedup over ILU-PCG and 12x speedup over MG for discontinuous coefficient problems
- CG is able to overcome the downfalls experienced by MG alone → MGPCG becomes an efficient method

*O. Tatebe, "The multigrid preconditioned conjugate gradient method", 6th Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, April, 1993.

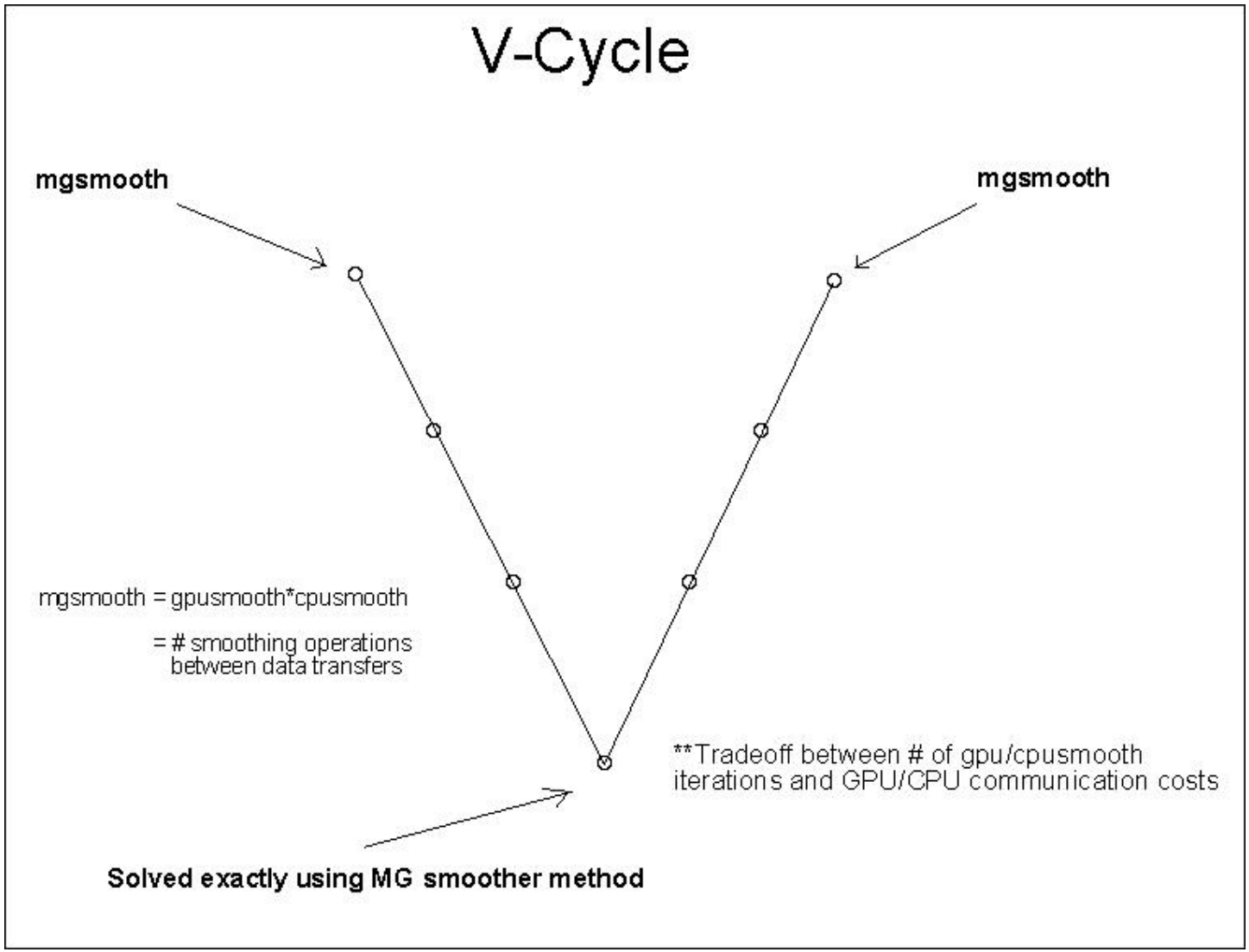
Promising Multigrid Approaches for Incompressible Two Phase Flow

- Klaus Stuben, Patrick Delaney, Serguei Chmakov, Algebraic Multigrid (AMG) for Ground Water Flow and Oil Reservoir Simulation
- Ruge, J.W., Stuben, K., 1986. Algebraic Multigrid (AMG), in .Multigrid Methods. (S. McCormick, ed.), Frontiers in Applied Mathematics, Vol 5, SIAM, Philadelphia.
- The Black Box Multigrid Numerical Homogenization Algorithm J. David Moulton, Joel E. Dendy Jr., and James M. Hyman JCP 142, (1998)
- Wan and Liu, "A boundary condition capturing multigrid approach to irregular boundary problems," SIAM J. Sci. Comput., 2004.

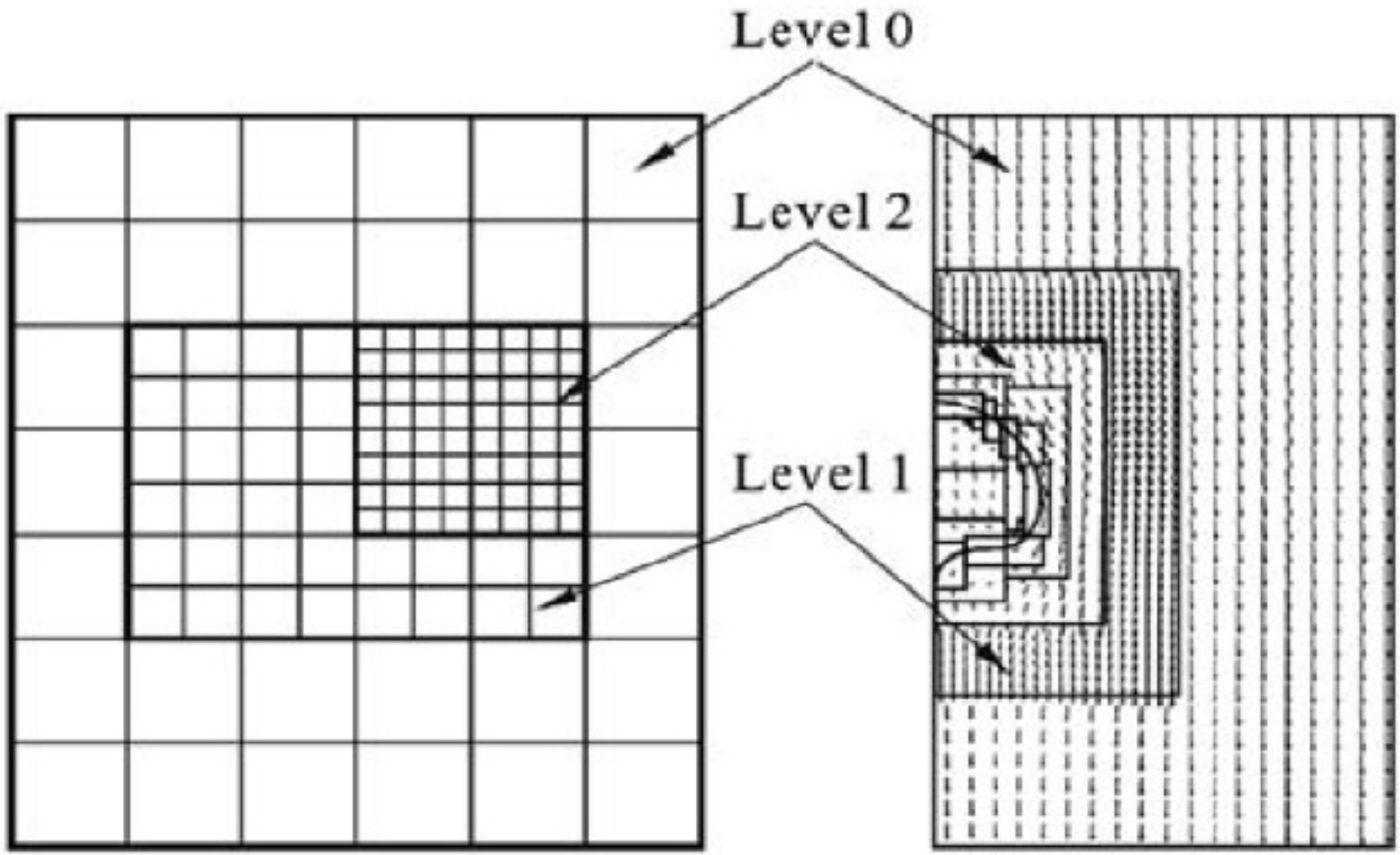
Promising Multigrid Approaches for Incompressible Two Phase Flow (cont)

- Mayo, "The fast solution of poisson's and the biharmonic equations in irregular domains," SIAM J. Num. Anal., 1984.
- Howell and Bell, "An adaptive-mesh projection method for viscous incomopressible flow," SIAM J. Sci. Comp., 1997.
- Schaffer, A Semicoarsening Multigrid Method for Elliptic PDE'S with Highly Discontinuous and Anisotropic Coefficients. SIAM J. SCI. COMP., 1998.

The Multigrid Preconditioned Conjugate Gradient Method (MGPCG)



Adaptive Grids



Multigrid Smoothers

We want to solve $Mz = r$ where $Q = M^{-1}$ represents our symmetric smoother:

$$z^{n+1} = z^n + Q(r - Az^n)$$

MG Smoothers - Jacobi

Slowest MG smoother, but very easy to vectorize

$M = D$, $Q = D^{-1}$

$$\begin{aligned} z^{n+1} &= z^n + D^{-1}(r - Az^n) \\ &= D^{-1}Dz^n + D^{-1}r - D^{-1}Az^n \\ &= D^{-1}(D - A)z^n + D^{-1}r \end{aligned}$$

MG Smoothers - Jacobi Code

PGI Fortran Code for Simple 1-D problem

```
!$acc region
do iterates=1,maxiterates
    do i=is,ie
       x(i)=(b(i)-L(i-1)*x_old(i-1)-U(i+1)*x_old(i+1))/D(i)
   end do
    do i=is,ie
       x_old(i)=x(i)
    end do
end do
!$acc end region
```

MG Smoothers - Symmetric-Gauss-Seidel

- Symmetric Gauss Seidel smoothers lead to faster convergence compared to Jacobi smoothers, but are not vectorizable in their natural form.
- A Red-Black ordering allows Gauss Seidel to be vectorized, simple ex.

$$\left|\begin{array}{cccc||c|}
2 & -1 & 0 & 0 & x_1 \\
-1 & 2 & -1 & 0 & x_2 \\
0 & -1 & 2 & -1 & x_3 \\
0 & 0 & -1 & 2 & x_4
\end{array}\right| \rightarrow \left|\begin{array}{cccc||c|}
2 & 0 & -1 & 0 & x_1 \\
0 & 2 & -1 & -1 & x_3 \\
-1 & -1 & 2 & 0 & x_2 \\
0 & -1 & 0 & 2 & x_4
\end{array}\right|$$

MG Smoothers - Symmetric Red-Black GS

$$\left| \begin{array}{cc} D_R & C^T \\ C & D_B \end{array} \right\| \left| \begin{array}{c} z_R \\ z_B \end{array} \right| = \left| \begin{array}{c} r_R \\ r_B \end{array} \right|$$

$$z_R^* = D_R^{-1} r_R$$

$$z_B = D_B^{-1}(r_B - C z_R^*)$$

$$z_R = D_R^{-1}(r_R - C^T z_B)$$

MG Smoothers - Symmetric Red-Black GS

$$\mathrm{Q} = \begin{vmatrix} D_R^{-1} + D_R^{-1} C^T D_B^{-1} C D_R^{-1} & -D_R^{-1} C^T D_B^{-1} \\ -D_B^{-1} C D_R^{-1} & D_B^{-1} \end{vmatrix}$$

$$z^{n+1} = z^n + Q(r - Az^n)$$

1. $r^* = r - Az^n$
2. $z^{n+1} = z^n + Qr^*$

MG Smoothers - Incomplete Cholesky (IC)

IC factorizations are often used as the preconditioner themselves, but here we use IC as a smoother for multigrid.

- Fastest MG smoother
- Factorization ensures M maintains same sparse structure as A
- The standard IC preconditioner cannot be vectorized.

MG Smoothers - ICRB

We can again use a Red-Black ordering just as in the GSRB case here to vectorize the IC algorithm following the method of Ortega*, then the algorithm is the same as for the GSRB case with D_B replaced by D_B^*.

$$\left|\begin{array}{cc} D_R & C^T \\ C & D_B \end{array}\right| \rightarrow \left|\begin{array}{cc} I & 0 \\ CD_R^{-1} & I \end{array}\right| \left\|\begin{array}{cc} D_R & 0 \\ 0 & D_B^* \end{array}\right\| \left|\begin{array}{cc} I & D_R^{-1}C^T \\ 0 & I \end{array}\right|$$

$$D_B^* = diagonal(D_B - CD_R^{-1}C^T)$$

*James Ortega, "Introduction to Parallel and Vector Solution of Linear Systems", Springer, 1988.

Test Problems

2D Dam Break

- Large Aspect Ratio
- Physical Dimensions → 120x10
- Grid Points → 1920x192
- "mgsmooth"="gpusmooth"

3D Bubble

- Normal Aspect Ratio
- Grid Points → 32x32x64
- From Olsson and Kreiss "A conservative level set method for two phase flow"', JCP 2005
- "mgsmooth"="gpusmooth"

2D Dam Break

(Loading 2Ddambreak.avi)

2D Dam Break Problem - Single Precision (4 orders reduction)

CPU Only	Time	cycles	gpusmooth
ICPCG	45.2	10	300
IC-MGPCG	1.4	4	4
GSRB-MGPCG	2.1	6	4
GSRB-MGPCG	4.1	4	20
ICRB-MGPCG	2.1	5	4
ICRB-MGPCG	4.5	4	20

GPU accel.	Time	cycles	gpusmooth
GSRB-MGPCG	1.5	6	4
GSRB-MGPCG	1.7	4	20
ICRB-MGPCG	1.7	5	4
ICRB-MGPCG	1.9	4	20

2D Dam Break Problem - Double Precision (9 orders reduction)

CPU Only	Time	cycles	gpusmooth
ICPCG	451.1	35	1000
IC-MGPCG	3.8	8	4
GSRB-MGPCG	5.6	10	4
GSRB-MGPCG	14.0	7	20
ICRB-MGPCG	6.4	10	4
ICRB-MGPCG	14.3	7	20

GPU accel.	Time	cycles	gpusmooth
GSRB-MGPCG	3.6	10	4
GSRB-MGPCG	4.6	7	20
ICRB-MGPCG	4.4	10	4
ICRB-MGPCG	5.1	7	20

Rising 3D Bubble

(Loading 3Dbubble.avi)

3D Bubble Problem - Single Precision (4 order reduction)

CPU Only	Time	cycles	gpusmooth
ICPCG	0.70	13	8
IC-MGPCG	0.24	3	4
GSRB-MGPCG	0.26	3	4
GSRB-MGPCG	0.79	3	20
ICRB-MGPCG	0.31	3	4
ICRB-MGPCG	0.83	3	20

GPU accel.	Time	cycles	gpusmooth
GSRB-MGPCG	0.38	3	4
GSRB-MGPCG	0.73	3	20
ICRB-MGPCG	0.44	3	4
ICRB-MGPCG	0.80	3	20

3D Bubble Problem - Double Precision (7 order reduction)

CPU Only	Time	cycles	gpusmooth
ICPCG	1.65	24	8
IC-MGPCG	0.52	5	4
GSRB-MGPCG	0.72	6	4
GSRB-MGPCG	2.09	5	20
ICRB-MGPCG	0.83	6	4
ICRB-MGPCG	2.18	5	20

GPU Accel.	Time	cycles	gpusmooth
GSRB-MGPCG	0.91	6	4
GSRB-MGPCG	1.63	5	20
ICRB-MGPCG	1.05	6	4
ICRB-MGPCG	1.75	5	20

Conclusions

- The benefit of the GPU increases as "gpusmooth" increases. The effectiveness of the GPU can be significantly enhanced, even for relatively small values of "gpusmooth," if the matrix coefficients can be permanently stored on the GPU.
- The performance on the GPU does not degrade when using double precision.
- The IC smoother is the most effective smoother; albeit the GSRB and ICRB smoothers exhibit comparable performance. The performance of MGPCG is insensitive to the aspect ratio for the tests we report here.
- Multigrid as a preconditioner outperforms all other preconditioners for our test problems; the difference is most dramatic when the aspect ratio is large.