Parallel implementation of the deflated GMRES in the PETSc package

Désiré NUENTSA WAKAM*[1],

Joint work with Jocelyne ERHEL[1], William D. GROPP[2]

[1] INRIA Rennes, [2] NCSA-University of Illinois

Third Workshop of the INRIA-Illinois Joint Laboratory for Petascale Computation,

NCSA, November 22-24, 2010

* Work done while visiting the Jointlab @NCSA (10/26 - 11/24)

Outline

1. Deflated GMRES
2. Adaptive strategy
3. Practical Implementation
4. Early results

General purpose

Linear system

- The core computation of many scientific simulations is to solve

$$Ax = b$$

- we restrict to $A \in \mathbb{R}^{n \times n}$ nonsymmetric, $x, b \in \mathbb{R}^n$

The GMRES method [Saad and Schultz, 86]

- From x_0, iteratively build a sequence of approximate solutions $x_1, \ldots x_k \ldots$
- at step k, $x_k \in x_0 + \mathcal{K}_k$ s.t. $r_k = b - Ax_k \perp A\mathcal{K}_k$
- $\mathcal{K}_k$ is a Krylov subspace
- From the Arnoldi process : $AV_k = V_{k+1}\bar{H}_k$ and $x_k = x_0 + V_k y_k$
- From the minimal residual condition : y_k solves $min||\beta e_1 - \bar{H}_k y_k||$

The solution is found in at most n iterations.

GMRES(m)

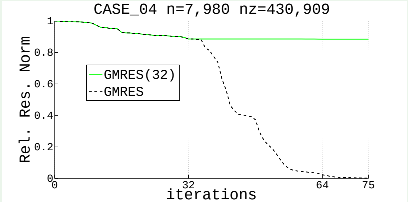
⇒ *Computational and memory requirements grow with k.*

Restarted GMRES(m)

- At step m, set $x_0 \leftarrow x_m$, and restart
- V_m and $\bar{H}_m$ are discarded ⇒ difficult to predict the convergence behaviour.
- ⇒ The iterative process can stall as well !!

An example

- Matrix: CASE_04
- Field : Fluid dynamics
- Origin: 2D linear cascade turbine
- Source : FLUOREM Matrix Collection



Deflated GMRES

GMRES and Eigenvalues

- Rate of convergence depends on the spectral distribution of A
- Removing or deflating eigenvalues (usually the smallest ones) improve the convergence rate
- Deflation occurs autommatically in the full version of GMRES when the subspace is large enough.
- In the restarted version, deflate an eigenvalue $\Rightarrow$ add the corresponding eigenvector [Morgan, J. Sci. Stat. Comput. 02]

In this work

- Deflation occurs by using a preconditioner corresponding to the invariant subspace associated to the selected eigenvalues [Erhel et al, JCAM, 1996; Burrage et al, NLAA, 1998]
- $U = [u_1 \ldots u_k]$, $T = U^T AU$
- $M^{-1} \equiv I_n + U(|\lambda_n| T^{-1} - I_k)U^T$,

Deflated GMRES with right preconditioning

$$\Rightarrow \text{Solve } AM^{-1}(Mx) = b \text{ for } x,$$

Algorithm: Input (m, k)[Erhel et al, JCAM, 1996]

Set x_0; $M = I$; $U = []$

1. Arnoldi process on AM^{-1} to get V_m and $\bar{H}_m$.
2. $x_m \leftarrow x_0 + V_m y_m$, y_m solution of $min||\beta e_1 - \tilde{H}_m y_m||$
3. If no convergence
    1. Find k Schur vectors S_k of H_m
    2. Orthogonalize $X = V_m S_r$ against U
    3. Increase U by X and Compute $T = U^T AU$
    4. Set $M^{-1} \equiv I_n + U(|\lambda_n| T^{-1} - I_{r_j})U^T$
    5. Set $x_0 \leftarrow x_m$ and Restart at step 1

Adaptive strategy

Main observation

- Deflated GMRES induces extra cost to compute and apply the preconditioner
- Should be applied only if necessary ; for instance, to prevent stagnation or insufficient reduction in the residual norm
- $\Rightarrow$ Adaptive strategy : *Detect stagnation at the end of a GMRES cycle and switch to deflation*

How to detect stagnation in GMRES(m) ??

- Strictly : Rate of convergence $||r_m||/||r_0|| > \tau$, $0 < \tau \leq 1$
- Pratically : should have a more realistic experimental test

GMRES(m) is declared to have stagnated if at the averate rate of progress over the last restart cycle of m steps, the residual norm tolerance cannot be met in some large multiple of the remaining number of steps allowed (the number of steps permitted is bounded by itmax)

[Sosonkina et al., NLAA98]:

Pratical implementation

Outline of Deflated GMRES (DGMRES(m, l, rmax))

choose $x_0, tol, m, itmax, k$
Set $B \equiv A\bar{M}^{-1}$ where $\bar{M}^{-1}$ is any external preconditioner
$r_0 = b - Bx_0; \quad M = I; \quad k = 0; \quad it = 0; \quad l = 0;$

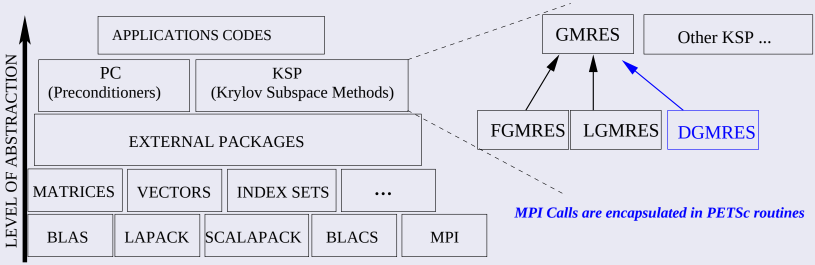
while $(||r_0|| > tol)$

Run a GMRES cycle of m max. iterations
with a minimization step to get x_m and r_m
$it+ = m;$

If $(||r_m|| > tol$ and $||it < itmax)$ **then**
$test = m * log(tol/||r_m||)/log(||r_m||/||r_0||);$
If (test $>$ smv * (itmax-it)) **then**
Estimate k smallest eigenvalues of BM^{-1} *and compute data for the preconditioner* M^{-1} *associated to the deflation*
End If
End If
$x_0 = x_m, \quad r_0 = r_m$
end while

Implementation in PETSc

New KSP type : DGMRES



Usage in Petsc

- In your executable, register dynamically the new KSP
- KSPRegisterDynamic(KSPDGMRES, "Path-to-libdgmres.a", "KSPCreate_DGMRES", KSPCreate_DGMRES);
- Use DGMRES just as any other KSP with the following options
  - *KSPSetType(ksp, KSPDGMRES)*; or *−ksp_type* dgmres
  - *−ksp_dgmres_eigen* <k>: Number of eigenvalues to deflate
  - *−ksp_dgmres_max_eigen* <kmax>: Maximum Number of eigenvalues to deflate
  - *−ksp_dgmres_smv* <smv> : relaxation parameter in the adaptive strategy
  - *−ksp_gmres_restart*<m>, *−ksp_max_it*<itmax>, *−ksp_rtol*<rtol> ...
  - Any option from GMRES holds;
  - Any preconditioner available for GMRES *−pc_type*<pc>

Environment for tests

Platform of tests

Cluster (Parapluie) @ GRID'5000

- SMP Nodes HP Proliant DL165
- Dual CPU/Node; 12 AMD cores/CPU @ 1.7GHZ
- Infiniband DDR network

Test Examples

Petsc Example *(Ex3)*

- Basic poisson problem
- 2D regular mesh on the unit square
- 2000x2000 mesh in these tests
- N:=4,000,000 NZ:=35,936,032
- easy to solve; given here just for large scale experimentation

FLUOREM Example *(CASE_017)*

- Parametrized Navier-Stokes equation
- Finite volume discretization of the steady equation
- Nonsymmetric jacobian matrix
- N:=381,689 NZ:=37,464,962
- strongly non-elliptic operator; need robust solver

Policy of tests and options

Policy of tests

- Run the iterative method until $||b - Ax||/||b|| \leq$ <rtol> or the maximum number of iterations <max_it> (not matrix-vectors) is reached.
- Domain decomposition preconditioner is used (Additive Schwarz, block jacobi); data assigned to processes in chunk of contiguous rows.
- For each test case and each number of subdomains #D
  - GMRES(m) : restarted GMRES; cycle of m iterations; right preconditioning;
  - DGMRES(m, k, max) : Deflated GMRES; k eigenvalues extracted at each restart; *max* eigenvalues extracted !!
  - DGMRES_A(m,k,max): Deflated GMRES with adaptive strategy.

Ex3

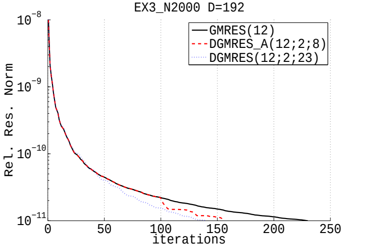
- ksp <rtol> 1e-12
- gmres restart <m> 12
- gmres <max_it> 500
- right preconditioning
- PC block jacobi
- Subdomains <D> [96 192 394 512 1024]
- sub_solver ILU(0) (Hypre-Euclid)
- dgmres eigen <k> 2

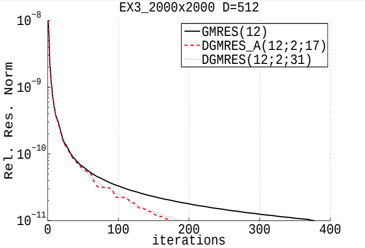
CASE_017

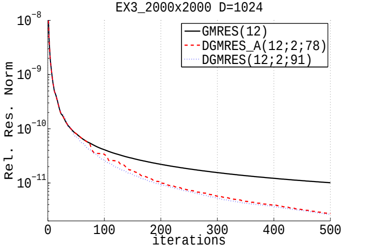
- ksp <rtol> 1e-08
- gmres restart <m> 64
- gmres <max_it> 1500
- right preconditioning
- PC ASM <overlap> 1
- Subdomains <D> [16 32 64]
- sub_solver LU (MUMPS)
- dgmres eigen <k> 5

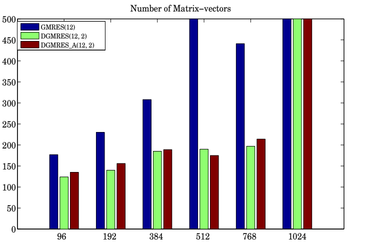
Ex3: Poisson problem; 2000x2000 mesh; $\approx 36M$ entries

Numerical convergence, 192-1024 subdomains



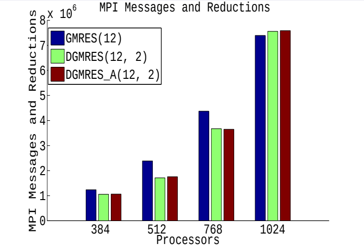


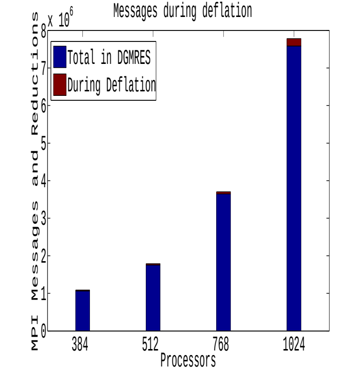




EX3: Overhead due to the deflation

MPI Messages on

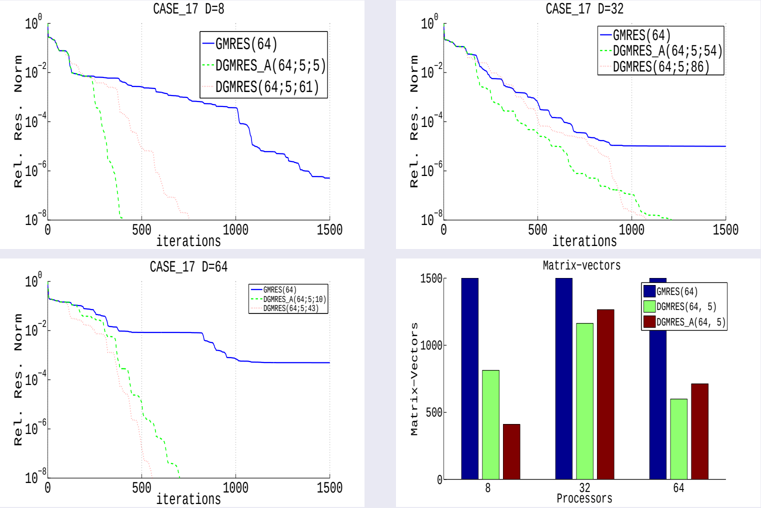




- Insignificant overhead of messages in the deflation phase
- More investigation on real test cases for the Flops and CPU time

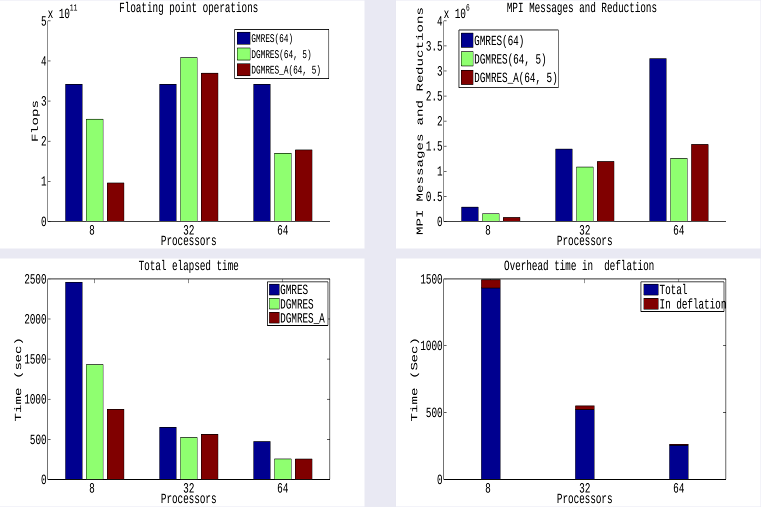
CASE_17; 3D CFD case; $\approx 37.5M$ entries

Numerical convergence; [8 32 64] subdomains,



CASE_17; 3D CFD case; $\approx 37.5M$ entries

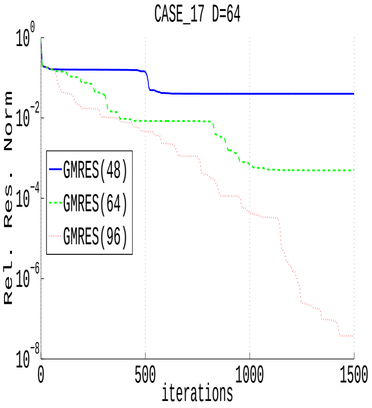
Flops, MPI messages and CPU Time; Linux cluster @GRID'5000

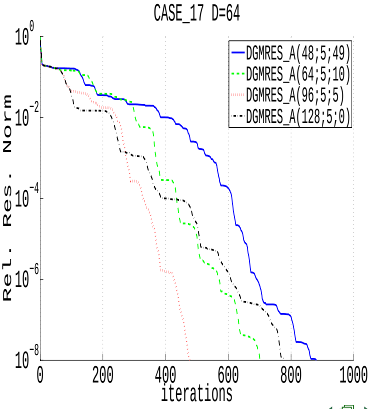


Real contribution of the adaptive strategy

The fact is:

- Sometimes, an increase of the restart parameter in GMRES may prevent stagnation
- Difficult to know how much it should be increase; more difficult to know if the method will converge either way after the increase.
- The deflated GMRES with adaptive strategy will provide more robustness with any restart parameter.





Conclusion

The strategy should be tested on more real applications and platforms

Nevertheless the code exists as a PETSc KSP module. . .
for real artihmetics.

Conclusion

The strategy should be tested on more real applications and platforms

Nevertheless the code exists as a PETSc KSP module...
for real artihmetics.

THANKS... QUESTIONS ???