Latency hiding of global reductions in pipelined Krylov methods

On rounding error resilience, maximal attainable accuracy and parallel performance of the pipelined Conjugate Gradients method in PETSc

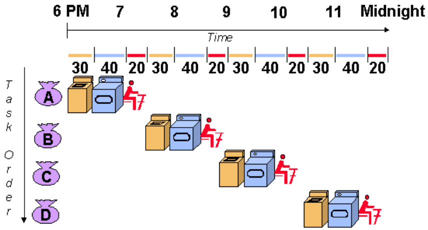
Exascale Applications & Software Conference, Stockholm, April 25-29, 2016

University of Antwerp* [BE] & INRIA Bordeaux† [FR]

Siegfried Cools*, Wim Vanroose*, Fatih Yetkin†, Emmanuel Agullo†, Luc Giraud†

Contact: siegfried.cools@uantwerp.be

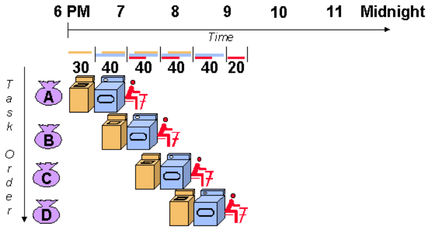
What are we working on?



Classical Krylov subspace method

Washing, drying and ironing in classical 'Laundry method'

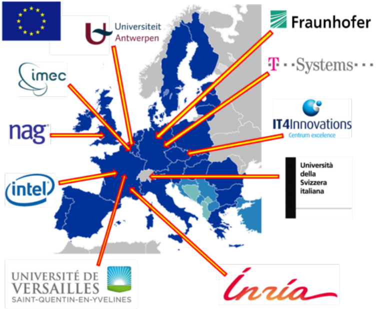
vs.



Pipelined Krylov subspace method

Latency hiding of global drying in pipelined 'Laundry method'

The EXA2CT European project



Increasing gap between computation and communication costs

- Floating point performance steadily increases
- Network latencies only go down marginally
- Memory latencies decline slowly
- Avoid communication: trade communication for computations
- Hide communication: overlap communication with computations

EXascale Algorithms and Advanced Computational Techniques
`http://exa2ct.eu/`

More info: see talk by Tom Vander Aa, Thu April 28, 13h40

Krylov subspace methods
General idea

Iteratively improve an approximate solution of linear system $Ax = b$,

$$x_i \in x_0 + \mathcal{K}_i(A, r_0) = x_0 + \mathrm{span}\{r_0, Ar_0, A^2 r_0, \ldots, A^{i-1} r_0\}$$

- minimize an error measure over expanding Krylov subspace $\mathcal{K}_i(A, r_0)$
- usually in combination with sparse linear algebra/stencil application
- three building blocks:
  - i. dot-product
  - ii. SpMV
  - iii. axpy

E.g. Conjugate Gradients

Algorithm 1 CG

```
procedure CG(A, b, x_0)
    r_0 := b − A x_0; p_0 = r_0
    for i = 0, . . . do
        s_i := A p_i
        α_i := (r_i, r_i) / (s_i, p_i)
        x_{i+1} := x_i + α_i p_i
        r_{i+1} := r_i − α_i s_i
        β_{i+1} := (r_{i+1}, r_{i+1}) / (r_i, r_i)
        p_{i+1} := r_{i+1} + β_{i+1} p_i
    end for
end procedure
```

dot-pr
SpMV
axpy

Classical CG

Algorithm 1 CG

```
procedure CG(A, b, x_0)
    r_0 := b − A x_0; p_0 = r_0
    for i = 0, ... do
        s_i := A p_i
        α_i := (r_i, r_i) / (s_i, p_i)
        x_{i+1} := x_i + α_i p_i
        r_{i+1} := r_i − α_i s_i
        β_{i+1} := (r_{i+1}, r_{i+1}) / (r_i, r_i)
        p_{i+1} := r_{i+1} + β_{i+1} p_i
    end for
end procedure
```

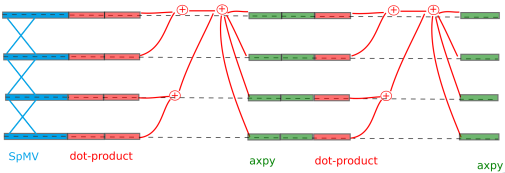
dot-pr
SpMV
axpy

Hestenes & Stiefel (1952)

i. 3 dot-products
   - 2 global reduction phases
   - latency dominated
   - scales as $log_2(\#partitions)$

ii. 1 SpMV
   - scales well (minor commun.)
   - non-overlapping (sequential to dot-product)

iii. 3 axpy's
   - recurrences to avoid SpMV's
   - perfectly scalable (no commun.)



Pipelined CG

Algorithm 3 Pipelined CG

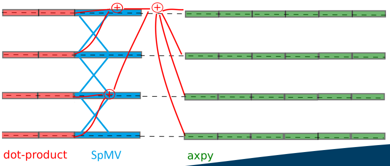
1: **procedure** PIPE-CG(A, b, x_0)
2: $r_0 := b - Ax_0$; $w_0 := Ar_0$
3: **for** $i = 0, \dots$ **do**
4: $\gamma_i := (r_i, r_i)$
5: $\delta := (w_i, r_i)$
6: $q_i := Aw_i$
7: **if** $i > 0$ **then**
8: $\beta_i := \gamma_i / \gamma_{i-1}$; $\alpha_i := (\delta/\gamma_i - \beta_i/\alpha_{i-1})^{-1}$
9: **else**
10: $\beta_i := 0$; $\alpha := \gamma_i / \delta$
11: **end if**
12: $z_i := q_i + \beta_i z_{i-1}$
13: $s_i := w_i + \beta_i s_{i-1}$
14: $p_i := r_i + \beta_i p_{i-1}$
15: $x_{i+1} := x_i + \alpha_i p_i$
16: $r_{i+1} := r_i - \alpha_i s_i$
17: $w_{i+1} := w_i - \alpha_i z_i$
18: **end for**
19: **end procedure**

dot-pr
SpMV
axpy

Ghysels & Vanroose (2013)

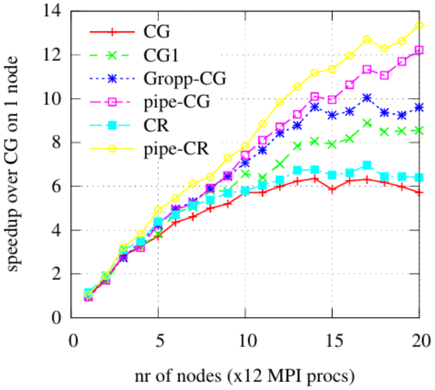
Re-organized version of classical CG for improved parallel performance

- equivalent to CG in exact arithmetic
- Communication avoiding: dot-products are grouped in one global reduction phase (line 4+5)
- Communication hiding: overlap global commun. (line 4+5) with computations (SpMV, line 6)
- 3 extra recurrences for $s_i = Ap_i$, $w_i = Ar_i$, $z_i = As_i$ (line 12+13+17)



Strong scaling experiment

- Hydrostatic ice sheet flow, $100 \times 100 \times 50$ Q1 finite elements, PETSc experiment
- Line search Newton method (rtol=10^{-8}, atol=10^{-15})
- CG preconditioned with block Jacobi with ICC(0) (rtol=10^{-5}, atol=10^{-50})



Ghysels & Vanroose (2013)

Performance breakdown

- max pipe-CG/CG speedup: $2.14\times$
- max pipe-CG/CG1 speedup: $1.43\times$
- max pipe-CR/CR speedup: $2.09\times$

(CG1 = Chronopoulos/Gear CG)
(CR = Conjugate Residuals)

Other pipelined Krylov methods

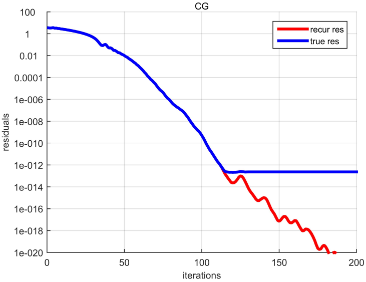
- Pipelined CG — Ghysels, Vanroose (2013)
- Pipelined GMRES — Ghysels, Ashby, Meerbergen, Vanroose (2012)

$$V_{i-\ell+1} = [v_0, v_1, \dots, v_{i-\ell}]$$
$$Z_{i+1} = [z_0, z_1, \dots, z_{i-\ell}, \underbrace{z_{i-\ell+1}, \dots, z_i}_{\ell}]$$

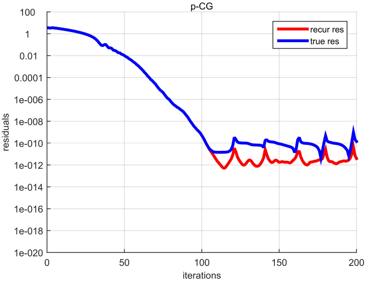
  - compute ℓ new basis vectors for Krylov subspace (SpMVs) during global communication (dot-products).
  - more technical, but deeper and variable pipelining possible p(ℓ)-GMRES
- Pipelined BiCGStab
  - non-symmetric operators
  - work-in-progress
- Preconditioned pipelined Krylov methods are available
  - p-pipe-CG
  - p-pipe-GMRES
- Augmented and deflated pipelined Krylov methods are (almost) available
  - work-in-progress

Rounding error propagation

Classical CG



Pipelined CG



Motivation: *pipe-CG loses max. attainable accuracy compared to classical CG*

- Model problem (figures): small 2D Laplacian problem, 2500 unknowns, single-core
- Loss of attainable accuracy is even more pronounced for larger systems

Rounding error propagation in CG

Rounding errors due to recursive definition of residual (and auxiliary variables)

$$\begin{cases} x_{i+1} = x_i + \alpha_i p_i & + \quad \delta_i^x, \\ r_{i+1} = r_i - \alpha_i s_i & + \quad \delta_i^r, \end{cases}$$

which deviates from the true residual $r_i = b - Ax_i$ in finite precision arithmetics

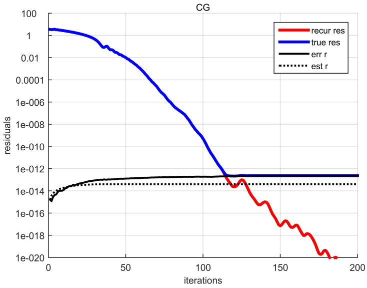
$$\begin{aligned} \Delta_{i+1}^r \quad &:= \quad r_{i+1} - (b - Ax_{i+1}) \\ &= \quad r_i - \alpha_i s_i + \delta_i^r - b + A(x_i + \alpha_i p_i + \delta_i^x) \\ &= \quad r_i - (b - Ax_i) - \alpha_i(Ap_i) + A(\alpha_i p_i) + A\delta_i^x + \delta_i^r \\ &= \quad \Delta_i^r + \underbrace{A(\alpha_i p_i) - \alpha_i(Ap_i)}_{\text{non-commutativity error}} + \underbrace{A\delta_i^x + \delta_i^r}_{\text{local rounding error}}, \end{aligned}$$

Rounding error model: $\|A(\alpha_i p_i) - \alpha_i(Ap_i)\| \approx 2\alpha_i \|s_i\| \psi, \qquad \psi = \text{eps.mach.}$

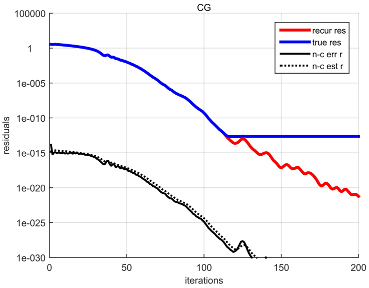
C. & Vanroose (2016)

Rounding error propagation in CG

Accumulated rounding error



Per-iter non-comm error



- Rounding errors accumulate and cause the true residuals to stagnate
- Maximal attainable accuracy for CG is still good (1e-13)
- Rounding error model: runtime tracking of residual rounding errors (estimate)

Rounding errors in pipe-CG

Observation: *rounding error propagation in pipe-CG is much more dramatic due to additional recurrence relations that all build up rounding errors.*

$$\begin{cases} z_i = q_i + \beta_i z_{i-1}, \\ s_i = w_i + \beta_i s_{i-1}, \\ p_i = r_i + \beta_i p_{i-1}, \end{cases} \qquad \begin{cases} x_{i+1} = x_i + \alpha_i p_i, \\ r_{i+1} = r_i - \alpha_i s_i, \\ w_{i+1} = w_i - \alpha_i z_i. \end{cases}$$

Residual rounding error is **coupled** with the error on the auxiliary variables:

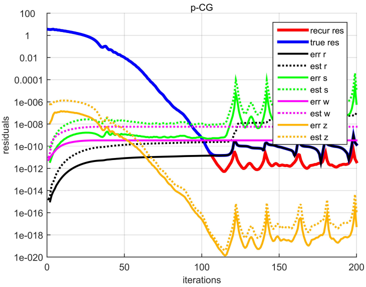
$$\begin{pmatrix} \|\Delta^r_{i+1}\| \\ \|\Delta^s_{i+1}\| \\ \|\Delta^w_{i+1}\| \\ \|\Delta^z_{i+1}\| \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & \alpha_i & 0 & 0 \\ 0 & \beta_{i+1} & 1 & \alpha_i \\ 0 & 0 & 1 & \alpha_i \\ 0 & 0 & 0 & \beta_{i+1} \end{pmatrix}}_{\text{rounding error propagation matrix}} \begin{pmatrix} \|\Delta^r_{i}\| \\ \|\Delta^s_{i}\| \\ \|\Delta^w_{i}\| \\ \|\Delta^z_{i}\| \end{pmatrix} + \begin{pmatrix} e^r_i \\ e^s_i \\ e^w_i \\ e^z_i \end{pmatrix},$$

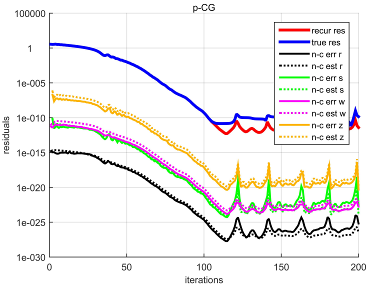
with $e^r_i = \|\alpha_i(Ap_i) - A(\alpha_i p_i)\|$, e^s_i, e^w_i, e^z_i the non-comm error on each variable.

Rounding errors in pipe-CG

Accumulated rounding error

Per-iter non-comm error





- Loss of maximal attainable accuracy (1e-11) compared to classical CG
- Rounding error model: runtime tracking of residual rounding error (estimate)
  Cost? two additional dot-pr per iteration to compute $\|s_i\|$ and $\|z_i\|$
  But: can be included in existing global reduction phase
  $\rightarrow$ no additional overhead

Pipe-CG with automated residual replacement

Replace r_i, s_i, w_i and z_i with their true values

$$\begin{cases} s_i = Ap_i, \\ z_i = As_i, \\ r_{i+1} = b - Ax_{i+1}, \\ w_{i+1} = Ar_{i+1}. \end{cases} \tag{1}$$

Replacement criterion:

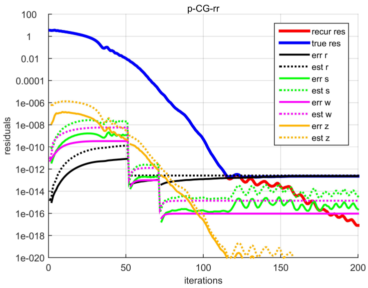
$$\|\Delta_{i-1}^r\| \leq \tau \|r_{i-1}\| \quad \text{and} \quad \|\Delta_i^r\| > \tau \|r_i\|, \tag{2}$$

with $\tau = \sqrt{\psi}$.

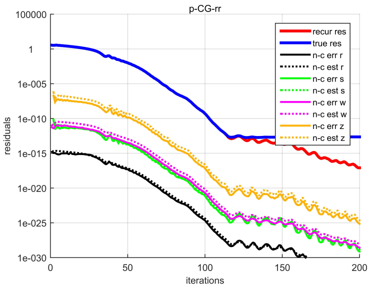
- Van der Vorst & Ye, *Residual Replacement strategies for Krylov Subspace iterative methods for Convergence of the True Residuals*, SISC, 2000.
- Tong & Ye, *Analysis of the finite precision bi-conjugate gradient algorithm for nonsymmetric linear systems*, Mathematics and Computations, 1999.

Pipe-CG with automated residual replacement

Accumulated rounding error



Per-iter non-comm error



- Pipe-CG-rr = pipe-CG with residual replacement based on rounding error model

  Cost? 4 additional SpMV's per replacement step
- Replacement criterion ensures:

  (1) a limited number of replacements,

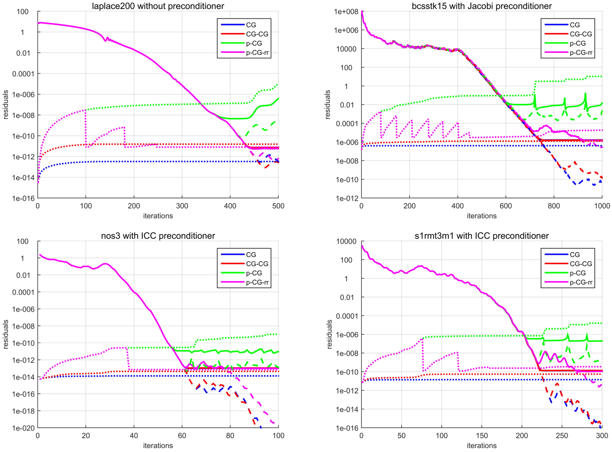
  (2) only when $\|r_i\|$ is sufficiently large (Krylov convergence is not affected).

Maximal attainable accuracy

MatrixMarket collection: convergence tests on all SPD matrices

Matrix	Prec	$\kappa(A)$	N	$\#nnz$	$\|\|b - Ax_0\|\|$	CG		CG-CG		p-CG		p-CG-rr		
						iter	res	iter	res	iter	res	iter	res	repl
bcsstk14	JAC	1.3e+10	1806	63,454	2.1e+09	631	2.0e-06	630	2.1e-06	411	2.3e-02	570	4.6e-05	6
bcsstk15	JAC	8.0e+09	3948	117,816	4.3e+08	758	1.7e-06	743	1.9e-06	575	8.7e-02	793	1.1e-05	8
bcsstk16	JAC	65	4884	290,378	1.5e+08	288	6.3e-07	281	1.6e-06	238	6.6e-03	285	1.1e-06	4
bcsstk17	JAC	65	10,974	428,650	9.0e+07	3445	1.4e-06	3317	3.6e-06	2478	5.8e+00	3237	3.8e-05	25
bcsstk18	JAC	65	11,948	149,090	2.6e+09	2271	5.7e-06	2248	6.1e-06	1270	4.6e-01	2056	7.6e-05	11
bcsstk27	JAC	7.7e+04	1224	56,126	1.1e+05	332	4.3e-10	322	1.0e-09	268	5.7e-06	352	8.9e-10	3
bcsstm19	-	2.3e+05	817	817	3.8e+06	766	4.8e-09	800	1.1e-07	345	3.9e-01	942	6.5e-06	34
bcsstm20	-	2.6e+05	485	485	4.6e+06	465	3.8e-09	484	9.9e-08	272	2.5e-01	627	4.3e-06	24
bcsstm21	-	24	3600	3600	1.0e-04	4	1.8e-20	4	3.5e-20	4	8.9e-20	4	8.9e-20	1
bcsstm22	-	9.4e+02	138	138	3.1e-03	68	2.1e-18	67	3.8e-18	65	1.6e-14	71	7.2e-18	2
bcsstm23	-	9.5e+08	3134	3134	6.7e+05	-	9.6e-03	-	1.4e-03	1749	2.4e-02	3700	2.1e-03	38
bcsstm24	-	1.8e+13	3562	3562	1.1e+05	-	1.0e-07	-	1.1e-05	1301	3.0e-03	2445	4.6e-04	39
bcsstm25	-	6.1e+09	15,439	15,439	6.9e+07	-	1.7e+01	-	1.6e+02	2195	6.7e+01	-	1.7e+02	309
bcsstm26	-	2.6e+05	1922	1922	2.6e-01	3615	7.6e-16	3484	2.0e-15	1720	1.8e-09	3063	4.6e-13	37
gr_30_30	-	3.8e+02	900	7744	1.1e+00	54	3.7e-15	53	7.9e-15	49	6.8e-13	54	6.7e-15	2
nos1	ICC*	2.5e+07	237	1017	5.7e+07	342	8.4e-07	334	8.0e-07	270	1.9e-01	631	1.7e-06	9
nos2	ICC*	6.3e+09	957	4137	1.8e+09	3501	1.5e-04	3683	1.9e-04	1923	1.1e+04	3867	2.3e-01	56
nos3	ICC	7.3e+04	960	15,844	1.0e+01	63	1.0e-13	62	1.1e-13	56	2.3e-11	68	9.2e-14	2
nos4	ICC	2.7e+03	100	594	5.2e-02	31	9.4e-17	30	1.2e-16	29	2.6e-15	30	1.3e-16	2
nos5	ICC	2.9e+04	468	5172	2.8e+05	60	1.2e-10	60	1.2e-10	56	1.4e-08	59	5.4e-10	2
nos6	ICC	8.0e+06	675	3255	8.6e+04	40	4.2e-10	34	5.7e-10	28	4.3e-06	47	5.3e-10	2
nos7	ICC	4.1e+09	729	4617	8.6e+03	47	2.5e-10	44	3.6e-10	33	3.7e-10	56	3.4e-10	4
s1rmq4m1	ICC	1.8e+06	5489	262,411	1.5e+04	122	6.5e-11	121	7.1e-11	110	9.6e-08	142	2.9e-10	3
s1rmt3m1	ICC	2.5e+06	5489	217,651	1.5e+04	227	1.3e-10	225	1.4e-10	204	6.2e-07	258	3.2e-10	3
s2rmq4m1	ICC*	1.8e+08	5489	263,351	1.5e+03	366	1.0e-11	362	1.2e-11	309	1.2e-06	449	4.3e-10	8
s2rmt3m1	ICC	2.5e+08	5489	217,681	1.5e+03	273	1.3e-11	270	1.6e-11	240	2.1e-06	363	3.8e-11	6
s3dkq4m2	ICC*	1.9e+11	90,449	2,455,670	6.8e+01	-	1.3e-06	-	1.4e-06	2658	3.6e-06	2460	9.2e-06	37
s3dkt3m2	ICC*	3.6e+11	90,449	1,921,955	6.8e+01	-	2.0e-05	-	2.0e-05	3409	1.3e-05	3370	1.5e-05	36
s3rmq4m1	ICC*	1.8e+10	5489	262,943	1.5e+02	1692	2.2e-12	1779	2.6e-12	1396	2.4e-05	1343	6.8e-06	32
s3rmt3m1	ICC*	2.5e+10	5489	217,669	1.5e+02	2399	4.2e-12	2255	4.7e-12	1821	1.1e-04	1878	9.9e-06	28
s3rmt3m3	ICC*	2.4e+10	5357	207,123	1.3e+02	2906	4.2e-12	2706	4.8e-12	2068	1.7e-04	2477	2.1e-07	60

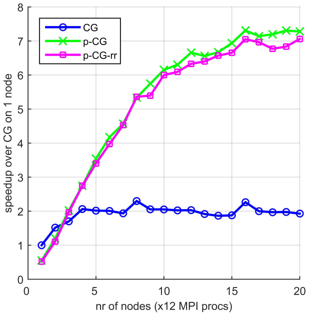
MatrixMarket collection: convergence tests on selected SPD matrices



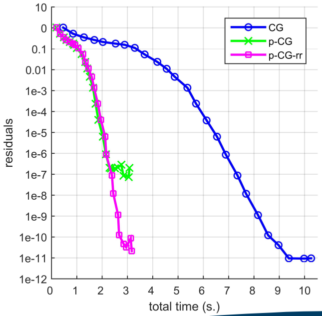
Strong scaling experiment

- PETSc implementation using MPICH-3.1.3 communication
- Benchmark problem: 2D Laplacian model, 1.000.000 unknowns
- System specs: 20 nodes, two 6-core Intel Xeon X5660 Nehalem 2.8GHz CPUs/node

Speedup over single-node CG
(12-240 cores)

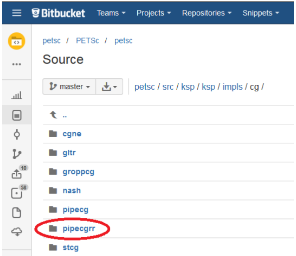


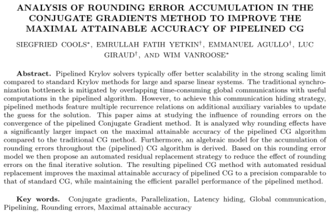
Accuracy i.f.o. total time spent
(144 cores)



Summary

- Pipe-CG-rr is robust & resilient version of pipe-CG:
  - improved maximal attainable accuracy w.r.t. pipe-CG (see Results: attainable accuracy),
  - good parallel performance, identical to pipe-CG (see Results: strong scaling).
- Pipe-CG-rr incorporated in *PETSc development version*.
- Pipe-CG-rr paper submitted to *SIAM Journal on Scientific Computing*.





ANALYSIS OF ROUNDING ERROR ACCUMULATION IN THE CONJUGATE GRADIENTS METHOD TO IMPROVE THE MAXIMAL ATTAINABLE ACCURACY OF PIPELINED CG

SIEGFRIED COOLS*, EMRULLAH FATIH YETKIN†, EMMANUEL AGULLO†, LUC GIRAUD†, AND WIM VANROOSE*

Abstract. Pipelined Krylov solvers typically offer better scalability in the strong scaling limit compared to standard Krylov methods for large and sparse linear systems. The traditional synchronization bottleneck is mitigated by overlapping time-consuming global communications with useful computations in the pipelined algorithm. However, to achieve this communication hiding strategy, pipelined methods feature multiple recurrence relations on additional auxiliary variables to update the guess for the solution. This paper aims at studying the influence of rounding errors on the convergence of the pipelined Conjugate Gradient method. It is analyzed why rounding effects have a significantly larger impact on the maximal attainable accuracy of the pipelined CG algorithm compared to the traditional CG method. Furthermore, an algebraic model for the accumulation of rounding errors throughout the (pipelined) CG algorithm is derived. Based on this rounding error model we then propose an automated residual replacement strategy to reduce the effect of rounding errors on the final iterative solution. The resulting pipelined CG method with automated residual replacement improves the maximal attainable accuracy of pipelined CG to a precision comparable to that of standard CG, while maintaining the efficient parallel performance of the pipelined method.

Key words. Conjugate gradients, Parallelization, Latency hiding, Global communication, Pipelining, Rounding errors, Maximal attainable accuracy

Work in progress

Shifted pipelined CG

- introduce shift σ in auxiliary variables: $s_i = (A-\sigma)p_i$, $w_i = (A-\sigma)s_i$
- do **not** shift the solution: $r_i = b - Ax_i$
- stabilizing effect on rounding error propagation for the right shift choice

$$\begin{pmatrix} \|\Delta^r_{i+1}\| \\ \|\Delta^s_{i+1}\| \\ \|\Delta^w_{i+1}\| \\ \|\Delta^z_{i+1}\| \end{pmatrix} = \begin{pmatrix} 1 & \alpha_i & 0 & 0 \\ 0 & \beta_{i+1} - \sigma\alpha_i & 1 & \alpha_i \\ 0 & -\sigma\alpha_i & 1 & \alpha_i \\ 0 & 0 & 0 & \beta_{i+1} \end{pmatrix} \begin{pmatrix} \|\Delta^r_i\| \\ \|\Delta^s_i\| \\ \|\Delta^w_i\| \\ \|\Delta^z_i\| \end{pmatrix} + \begin{pmatrix} e^r_i \\ e^s_i \\ e^w_i \\ e^z_i \end{pmatrix}.$$

Rounding error model in soft error detection

- model allows for separation between rounding errors and soft errors
- introduce detect-and-correct strategy to improve soft error resilience

Pipelined BiCGStab

- development of stable pipe-BiCGStab for non-symmetric operators
- implement countermeasures to improve rounding error resilience

References and FAQs

- P. Ghysels, T.J. Ashby, K. Meerbergen, W. Vanroose, *Hiding Global Communication Latency in the GMRES Algorithm on Massively Parallel Machines*, SIAM J. Sci. Comput., 35(1), 2013.
- P. Ghysels, W. Vanroose, *Hiding global synchronization latency in the preconditioned Conjugate Gradient algorithm*, Parallel Computing, 40, 2013.
- H.A. Van der Vorst, Q. Ye, *Residual Replacement strategies for Krylov Subspace iterative methods for Convergence of the True Residuals*, 22(3), SIAM J. Sci. Comput., 2000.
- E. Carson, J. Demmel, *A Residual Replacement Strategy for Improving the Maximum Attainable Accuracy of s-Step Krylov Methods*, SIAM J. Mat. Anal. Appl., 35(1), 2014.
- S. Cools, W. Vanroose, *Analysis of rounding error accumulation in the Conjugate Gradients method to improve the maximal attainable accuracy of pipelined CG*, submitted to SIAM J. Sci. Comput., 2016.

Q: *What is the difference between pipelined and s-step Krylov methods?*

A: Global communication is hidden vs. avoided

A: Off-the-shelf preconditioning possible vs. specialized preconditioning

Q: *Is the code available online?*

A: Yes: pipe-CG, pipe-CG-rr, gropp-CG, pipe-CR and p(ℓ)-GMRES are all in the current PETSc library, available at `https://www.mcs.anl.gov/petsc/`.