Solving linear systems with multiple right-hand sides: Kernel for scientific computing and parallel computing challenges

Vasilis Kalantzis

Department of Computer Engineering & Informatics
University of Patras, Greece

LinkSCEEM-2, Athens, 13 July 2011

Acknowledgments

- CEID collaborators: Prof. Efstratios Gallopoulos, Jiannis Kalofolias, Maria Predari
- Dr. Costas Bekas, IBM Research, Zurich
- Prof. Ahmed Sameh, Purdue University

Applications vs. Comp. kernels (Sameh+'84)

	1	2	3	4	5	6	7	8	9	
Lattice Gauge (QCD)	*	.	.	*	.	.	.	.	*	.
quantum mechanics	.	.	.	*	.	.	*	*	*	
weather	.	.	.	.	*	*	.	.	.	
CFD	*	.	*	.	*	*	.	.	.	
geodesy	*	*	.	.	.	.	.	.	.	
inverse problems	.	*	.	.	*	.	.	.	.	.
structures	*	.	*	*	.	.	.	.	.	
circuit	*	.	*	.	.	*	*	.	*	.
circuit simulation	*	.	*	.	.	.	*	.	.	
electromagnetics	*	*	*	*	*	*	.	.	.	

1. linear systems 2. least squares 3.nonlinear systems
4. eigenvalues/SVD's 5. fast transforms 6. rapid elliptic solvers
7. stiff DE 8. Monte Carlo 9. integral transforms

New Applications vs. Computational Kernels

	1	2	3	4	5	6	7	8	9
financial	*	*	*	.	.	.	*	*	*
IR	*	*	.	*	.	.	*	.	.
DS& Image P.	*	*	*	*	*	.	*	.	*
Internet Algorithmics	*	.	.	*	.	.	*	*	.

1. linear systems 2. least squares 3.nonlinear systems
4. eigenvalues/SVD's 5. fast transforms 6. rapid elliptic solvers
7. *Optimization* 8. Monte Carlo 9. integral transforms

Short Introduction

- Numerical linear algebra computations are fundamental kernels of scientific computing (table 1) and optimization targets in HPC.
- Fundamental problems: $A \leftarrow A + BC$; Solve $Ax = b$, $\min_x \|Ax - b\|$; compute $PA = LU$; $A = LL^\top$; $A = V\Lambda V^{-1}$; $A = U\Sigma V^\top$, etc.
- A ``new'' fundamental problem:
  - $f(A)$, e.g. $\exp(A)$, where $f(.)$ is a function of A
  - $f(A)B$, e.g. $\exp(A)B$
  - Need for solving problems with multiple right-hand sides

Solving systems with mrhs

- Several applications demand the solution of linear systems with mrhs
  - Lattice QCD
  - Computational Electromagnetics
  - Uncertainty Quantification
  - Data Handling
  - Domain Decomposition
  - Time dependent problems (holy grail!)

Some applications solving mrhs

COMPUTING AND DEFLATING EIGENVALUES WHILE SOLVING MULTIPLE RIGHT HAND SIDE LINEAR SYSTEMS WITH AN APPLICATION TO QUANTUM CHROMODYNAMICS *

ANDREAS STATHOPOULOS † AND KONSTANTINOS ORGINOS ‡

Parallel hybrid solver for multiple right-hand sides for the wave propagation simulation in the frequency domain for 3D domains with heterogeneity and topography

Proposers: Henri Calandra (TOTAL), Luc Giraud and Jean ROMAN (INRIA).

Linear systems with mrhs

Main objective (assume dense matrices):

- Solve $AX = B$, where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times s}$ with $s > 1$

Direct methods

- Factorize and solve, e.g. $L(UX) = PB$
  - Cost: solve at a rate of $O(n^3/s + 2n^2)$ per rhs
  - $\Rightarrow$ cubic cost amortized as s increases

Iterative methods

- Cost: $O(\#iter * \mathrm{cost}(MV))$ per rhs in ``standard approaches'',
- ... e.g. applying CG separately per rhs.

Challenges

Direct methods

- Factorize $A \rightarrow$ high cost even for moderate size n

Iterative methods

- What is the analogue of the ``factorize once'' advantage of direct methods

Iterative methods for mrhs

Seed methods: Exploit Krylov subspace for ``other'' rhs

Saad'87, PapadrakakisSmerou, vdVorst, SmithPetersonMittra'89, Fisher, SimonciniG, ChanWan'97, GuennouniJbilou, Gu, LotstedtNilsson, MorganWilcoxAbdel-Rehim, ...

Block methods: Generate block Krylov subspace

O'Leary, Vital, NikishinYeremin, SimonciniG, CalvettiReichel, FreundMalhotra, Jbilou, JbilouMessaoudiSadok, JbilouSadok, GuennouniJbilou, BakerDennisJessup, Gutknecht, ...

Hybrid approaches: Block seeds, deflation, multiple matrices

SimonciniG, ChanWan, SaadErhel, ErhelGyomar'ch, ChanNg, deSturler, KilmerMillerRappoport, Morgan, GolubRuizTouhami, OrginosStathopoulos...

Iterative methods for mrhs

- Not all of the above methods are suitable for every problem
- Example: Let A be a SPD matrix and B random
  - Compare: standard CG vs. recent *seed* CG solver
  - $n = 500 : 500 : 5000$, stopping when $\|r\| \leq 1e-8$

Runtimes

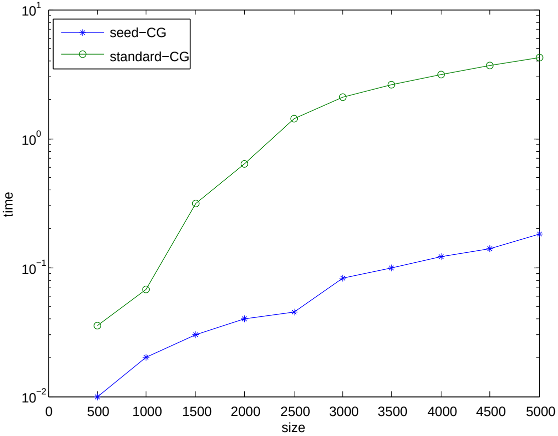

Figure: time per rhs

Why parallel computing?

- Need to solve much larger problems
- Resolve memory and computational cost bottlenecks

Architecture	$n = 10^3$	5×10^3	2×10^4	5×10^4	10^6
`32-bit`	4MB	100MB	1.6GB	10GB	4TB
`64-bit`	8MB	200MB	3.2GB	20GB	8TB

Table: memory requirements for different n

Some approaches

Direct approach

- Replicate A in each processor and factorize
- Factorization is needed only once, no matter what is s
- Even parallel factorization can be very costly

Iterative approach

- It is known today that iterative mrhs solvers can be effective
- ... on a single processor
- Challenge is to preserve the same advantage when going parallel,
- ... can we beat the ``embarassingly parallel'' approach?
- Information sharing between systems:
  - Overhead
  - Scalability
  - Granularity

Current work

- Build from the start parallel iterative methods for $AX = B$
- Use these as kernels for solving problems in different applications
- Combine with mixed precision arithmetic

$\boxed{\text{THANK YOU! QUESTIONS?}}$