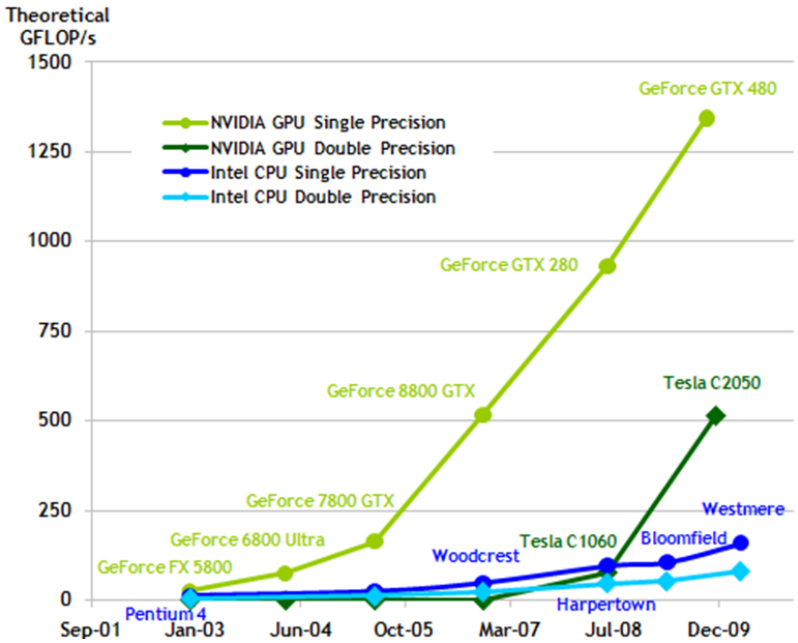
Accelerating Model Reduction of Large Linear Systems with Graphics Processors

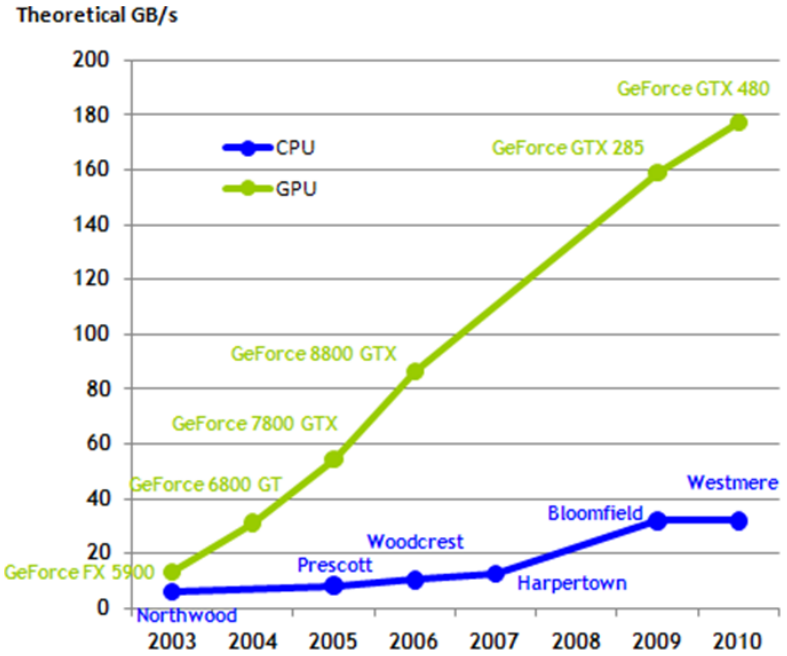
P. Benner[1], P. Ezzatti[2], D. Kressner[3],
E.S. Quintana-Ortí[4], **Alfredo Remón**[4]

[1]Max-Plank-Institute for Dynamics of Complex Technical Systems (Magdeburg, Germany).
[2]Centro de Cálculo-Inst. de la Computación,Univ. de la República (Montevideo, Uruguay).
[3]Seminar für Angewandte Mathematik, ETHZ (Zürich, Switzerland).
[4]Depto. de Ingeniería y Ciencia de Computadores, Universidad Jaume I (Castellón, Spain).

ModRed'10 - December 2010

Why GPUs?





Extracted from: *CUDA C Programming Guide 3.1, NVIDIA Corporation*

GPUs for general purpose programming

GPUs where not developed for general programming

- Difficult and slow code generation

GPUs for general purpose programming

GPUs where not developed for general programming

- Difficult and slow code generation $\rightarrow$ CUDA

In 2006 appears **CUDA** (Computed Unified Device Architecture)

- Created by NVIDIA, is a HW-SW platform to facilitate the use of GPUs in general purpose programming
- Software: compilers (C, FORTRAN), libraries (CUFFT, CUBLAS, ...)
- Hardware: efficient thread management and memory access

GPUs for general purpose programming

GPUs where not developed for general programming

- Difficult and slow code generation $\rightarrow$ CUDA
- Gap between single and double precision performance

GPUs for general purpose programming

GPUs where not developed for general programming

- Difficult and slow code generation → CUDA
- Gap between single and double precision performance → Fermi

In 2010 appears the Fermi architecture

- The double precision computations are only two times slower than single precision computations

GPUs for general purpose programming

GPUs where not developed for general programming

- Difficult and slow code generation → CUDA
- Gap between single and double precision performance → Fermi

For a list of scientific CUDA applications visit
http://www.nvidia.com/object/cuda_apps_flash_new.html

Outline

Model reduction
- Model reduction via the BT method
- Matrix sign function method
- Numerical results

Model Reduction: Purpose

Given

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \qquad t > 0, \qquad x(0) = x^0, \\ y(t) &= Cx(t) + Du(t), \qquad t \geq 0, \end{aligned}$$

find a reduced model

$$\begin{aligned} E_r\dot{x}_r(t) &= A_r x_r(t) + B_r u(t), \qquad t > 0, \qquad x_r(0) = {x_r}^0, \\ y_r(t) &= C_r x_r(t) + D_r u(t), \qquad t \geq 0, \end{aligned}$$

of order $r \ll n$ and output error

$$y - y_r = Gu - G_r u = (G - G_r)u$$

such that

$$\|y - y_r\| \text{ and } \|G - G_r\| \text{ are "small"!}$$

Model Reduction: Example

Optimal cooling of steel profiles

Arises in a manufacturing method for steel profiles.

Objective: *reduce the temperature as fast as possible.*
Method: *spraying of cooling fluids on the surface.*
Goal: *Material properties (durability, porosity) have to satisfy quality standards.*
Problem dimensions:

$n = 5,177$, $m = 7$, and $p = 6$.

Math. model: STEEL_I from the Oberwolfach benchmark collection Oberwolfach benchmark collection: http://www.imtek.de/simulation/benchmark/
Model details: [Tröltzsch/Unger 1999/2001], [Penzl 1999] and [Saak 2003].

Outline

Model reduction
- Model reduction via the BT method
- Matrix sign function method
- Numerical results

Balanced Truncation (BT) method

Procedure composed of three steps:

1. Solve the "coupled" generalized Lyapunov matrix equations

$$\begin{aligned} AW_cE^T + EW_cA^T + BB^T &= 0, \\ A^TW_oE + E^TW_oA + C^TC &= 0, \end{aligned}$$

   with $W_0 = E^TW_oE$ for S, R such that $W_c = S^TS$, $W_o = R^TR$.

2. Compute

$$SR^T = U\Sigma V^T = [\, U_1 \; U_2 \,] \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix},$$

   with $\Sigma_1 \in \mathbb{R}^{rxr}$, $\Sigma_2 \in \mathbb{R}^{(n-r)x(n-r)}$.

Balanced Truncation method (Cont.)

3. In the last stage

$$T_l = \Sigma_1^{-1/2} V_1^T R \quad \text{and} \quad T_r = S^T U_1 \Sigma_1^{-1/2},$$

and $(A_r, B_r, C_r, D_r, E_r) = (T_l A T_r, T_l B, C T_r, D, T_l E T_r)$.

- The state-space dimension r of the reducer-order model can be chosen adaptatively as this method provides a realization $\hat{G}$ satisfying

$$\|G - G_r\|_\infty \leq 2 \sum_{j=r+1}^{n} \sigma_j.$$

Balanced Truncation method (Cont.)

3. In the last stage

$$T_l = \Sigma_1^{-1/2} V_1^T R \quad \text{and} \quad T_r = S^T U_1 \Sigma_1^{-1/2},$$

and $(A_r, B_r, C_r, D_r, E_r) = (T_l A T_r, T_l B, C T_r, D, T_l E T_r)$.

The most expensive computation is the solution of the generalized Lyapunov equations.

Outline

Model reduction
- Model reduction via the BT method
- Matrix sign function method
- Numerical results

Matrix sign function method

Remarks

- It is an efficient tool to solve stable Lyapunov equations.
- There are different schemes to solve the matrix sign function, like the Newton iteration method.
- The Newton iteration method for the matrix sign function.

$$\begin{aligned} A_0 &= A, \\ A_{k+1} &= \frac{1}{2}(A_k + A_k^{-1}), \end{aligned}$$

  - Main features:
    - Simple.
    - Efficient on parallel implementation.
    - Asymptotic quadratic convergence.

Matrix sign function method

Low-rank factors version of the algorithm

- On convergence after j iterations, $W_c \approx \tilde{S}^T \tilde{S}$ and $W_o \approx \tilde{R}^T \tilde{R}$.
- Convergence can be accelerated using a scaling factor, in our case:

$$c_k = \sqrt{\|A\|_F / \|E A_k^{-1} E\|_F}.$$

- Even if A is sparse, $\{A_k\}_{k=1,2,\ldots}$ in general are full dense matrices.

Requires $\mathcal{O}(n^3)$ floating-point operations per iteration.

- The most computationally expensive step is the matrix inversion.

Algorithm 1 CGCLNC

1: $A_0 = A$, $\hat{S}_0 = B^T$, $\hat{R}_0 = C$.
2: $k = 0$.
3: **repeat**
4: $\quad A_{k+1} = \frac{1}{\sqrt{2}} \left(A_k/c_k + c_k(E{A_k}^{-1})E \right)$.
5: $\quad$ Compute the rank-revealing QR (RRQR) decomposition

$$\frac{1}{\sqrt{2c_k}} \left[\tilde{S}_k, \;\; c_k \tilde{S}_k (E A_k^{-1})^T \right] = Q_s \begin{bmatrix} U_s \\ 0 \end{bmatrix} \Pi_s$$

6: $\quad \tilde{S}_{k+1} \leftarrow U_s \Pi_s$
7: $\quad$ Compute the rank-revealing QR (RRQR) decomposition

$$\frac{1}{\sqrt{2c_k}} \left[\tilde{R}_k, \;\; c_k (\tilde{R}_k A_k^{-1}) E \right] = Q_r \begin{bmatrix} U_r \\ 0 \end{bmatrix} \Pi_r$$

8: $\quad \tilde{R}_{k+1} \leftarrow U_r \Pi_r$
9: $\quad k = k + 1$.
10: **until** $\|A_k - E\|_1 < \tau \|A_k\|_1$

Hybrid implementation

Computations performed at iteration j

CPU **GPU**

1. $PA_j = LU$ *
2. EA_j^{-1} ; $\tilde{R}_k A_j^{-1}$
3. $(EA_j^{-1})E$
4. Compute the rank-revealing QR (RRQR) decomposition

$$\frac{1}{\sqrt{2c_j}} \left[\tilde{S}_j, \;\; c_j \tilde{S}_j (EA_j^{-1})^T \right] = Q_s \begin{bmatrix} U_s \\ 0 \end{bmatrix} \Pi_s$$

5. $\tilde{S}_{j+1} \leftarrow U_s \Pi_s$
6. Compute the rank-revealing QR (RRQR) decomposition

$$\frac{1}{\sqrt{2c_j}} \left[\tilde{R}_j, \;\; c_j (\tilde{R}_j A_j^{-1}) E \right] = Q_r \begin{bmatrix} U_r \\ 0 \end{bmatrix} \Pi_r$$

7. $\tilde{R}_{j+1} \leftarrow U_r \Pi_r$
8. $A_{j+1} = \frac{1}{\sqrt{2}} \left(A_j / c_j + c_j (EA_j^{-1}) E \right)$

* CPU and GPU cooperate during this operation

Outline

Model reduction
- Model reduction via the BT method
- Matrix sign function method
- Numerical results

- **Hardware**
  - Platform consisting of two INTEL Xeon QuadCore E5410 processors at 2.33GHz, connected to an NVIDIA Tesla C1060 via a PCI-e bus.

- **Software**
  - LAPACK(CPU): all the computations are performed on the CPU using LAPACK and BLAS kernels (MKL v.10.2).
  - Hybrid(CPU+GPU): computations are executed on the most convenient architecture minimizing the communications. (MKL(v.10.2)+CUBLAS(v.2.1))

Model **STEEL_I** from the **Oberwolfach benchmark collection**

- Arises in a manufacturing method for steel profiles.
- The objective is to design a control that yields moderate temperature gradients when the rail is cooled down.
- The model corresponds to a 2-D heat equation.

- **Dimensions of the problem:** $n = 5,177,\ m = 7,\ p = 6$

Math. model: [Tröltzsch/Unger 1999/2001], [Penzl 1999] and [Saak 2003].
Oberwolfach benchmark collection: http://www.imtek.de/simulation/benchmark/

Model **FLOW_METER** from the **Oberwolfach benchmark collection**

- Is a 2-D model of an anemometer-like structure.
- Mainly consists of a tube and a small heat source.
- The model is given by a spatially semi-discretized instationary convection difussion equation.
- The reference temperature is set to 300K and Dirichlet boundary conditions as well as initial conditions are set to 0 with respect to the reference.

- **Dimensions of the problem:** $n = 9,669,\ m = 1,\ p = 5$

Math. model: [Harper 1997], [Ernst 2001] and [Mossmann 2004]

Oberwolfach benchmark collection: http://www.imtek.de/simulation/benchmark/

Model reduction

Results for benchmark STEEL_I

#Iter. k	Time (s)		Conv. criterion $\|A_k + E\|_F$
	Hybrid Impl.	LAPACK	
1	2.958	5.337	8.153664e-02
2	2.618	5.286	6.157084e-03
3	2.650	5.354	1.103795e-03
4	2.732	5.465	3.400846e-04
5	2.955	5.638	1.088081e-04
6	3.486	6.219	2.369416e-05
7	3.946	6.553	2.551781e-06
8	4.442	6.909	1.702591e-07
TOTAL:	25.787	46.761	

TIME IS REDUCED ON A 45%

Problem dimensions: $n = 5,177$, $m = 7$, $p = 6$.

Model reduction

Results for benchmark STEEL_I (hybrid implementation)

#Iter. k	Time (s)				
	$PA_k = LU$	$EA_k{}^{-1}$, $\tilde{R}_k A_k{}^{-1}$	$(EA_k{}^{-1})E$	$\tilde{S}_k(EA_k{}^{-1})$,$(\tilde{R}_k A_k{}^{-1})E$, Compress	Iteration
1	0.698	1.041	0.807	0.121	2.958
2	0.544	1.023	0.788	0.047	2.618
3	0.544	1.023	0.788	0.079	2.650
4	0.544	1.023	0.788	0.159	2.732
5	0.543	1.023	0.789	0.381	2.955
6	0.545	1.023	0.788	0.909	3.486
7	0.546	1.022	0.789	1.366	3.946
8	0.543	1.023	0.788	1.866	4.442
				Accumulated time (s)	25.787

Problem dimensions: $n = 5,177$, $m = 7$, $p = 6$.

Model reduction

Results for benchmark FLOW_METER

#Iter. k	Time (s)		Conv. criterion $\|A_k + E\|_F$
	Hybrid Impl.	LAPACK	
1	17.254	31.516	7.586531e+01
2	16.861	31.580	7.447659e+00
3	16.883	31.725	1.747226e+00
4	16.961	31.970	5.521871e-01
5	17.140	32.126	1.741928e-01
6	17.454	32.329	5.558618e-01
7	17.726	32.525	1.368278e-02
8	17.831	32.842	1.876876e-03
9	17.953	32.896	1.274213e-04
10	18.016	32.997	1.592051e-06
11	17.994	32.881	2.632143e-07
TOTAL:	192.217	355.387	

TIME IS REDUCED ON A 46%

Problem dimensions: $n = 9,669$, $m = 1$, $p = 5$.

Model reduction

Results for benchmark Flow_Meter (hybrid implementation)

#Iter. k	Time (s)				
	$PA_k = LU$	$EA_k{}^{-1}$, $\tilde{R}_k A_k{}^{-1}$	$(EA_k{}^{-1})E$	$\tilde{S}_k(EA_k{}^{-1}),(\tilde{R}_k A_k{}^{-1})E$, Compress	Iteration
1	3.380	7.741	5.183	0.289	17.359
2	2.906	7.673	5.116	0.109	16.512
3	2.918	7.673	5.116	0.137	16.553
4	2.888	7.673	5.116	0.202	16.592
5	3.007	7.673	5.115	0.359	16.871
6	2.893	7.674	5.116	0.702	17.099
7	2.886	7.673	5.116	0.971	17.365
8	2.890	7.674	5.116	1.066	17.462
9	2.893	7.673	5.117	1.191	17.591
				Accumulated time (s)	192.217

Problem dimensions: $n = 9,669$, $m = 1$, $p = 5$.

Thanks...

Any question?