# PMAA 2008

Neuchâtel, June 22th, 2008

# Chebyshev methods for the matrix exponential


Konrad Waldherr

Technische Universität München, Germany

# Overview

- Gradient Flow Algorithm
- The structure of the matrices
- Methods of computing the exponential of a matrix
  - Taylor series
  - Padé approximation
  - eigendecomposition
  - Chebyshev series expansion
- Numerical results
  - accuracy
  - computation time
  - Speedup and efficiency
- Conclusions

# Gradient Flow Algorithm

One iteration step in the Gradient Flow Algorithm

- Calculate the forward-propagation for all $t_1, t_2, ..., t_k$:

$$\mathbf{U}(t_k) = e^{-i\Delta t \mathbf{H}_k} \cdot e^{-i\Delta t \mathbf{H}_{k-1}} \cdots e^{-i\Delta t \mathbf{H}_1}$$

- Compute the backward-propagation for all $t_M, t_{M-1}, \ldots, t_k$

$$\mathbf{\Lambda}^*(t_k) = e^{-i\Delta t \mathbf{H}_k} \cdot e^{-i\Delta t \mathbf{H}_{k+1}} \cdots e^{-i\Delta t \mathbf{H}_M}$$

- Calculate the update

$$\frac{\partial h(\mathbf{U}(t_k))}{\partial u_j} = \mathrm{Re}\left\{\mathrm{tr}\left[\mathbf{\Lambda}^*(t_k)(-i\mathbf{H}_j)\mathbf{U}(t_k)\right]\right\}$$

# Numerical tasks

- Computation of the matrix exponentials

$$\mathbf{U}_k := e^{-i\Delta t \mathbf{H}_k}$$

- Computation of all intermediate products

$$\begin{aligned}
&\mathbf{U}_0 \\
&\mathbf{U}_1 \cdot \mathbf{U}_0 \\
&\mathbf{U}_2 \cdot \mathbf{U}_1 \cdot \mathbf{U}_0 \\
&\vdots \\
\mathbf{U}_M \cdot \mathbf{U}_{M-1} \cdot \mathbf{U}_{M-2} &\cdots \mathbf{U}_2 \cdot \mathbf{U}_1 \cdot \mathbf{U}_0
\end{aligned}$$

# Structure of the matrices $\mathbf{H}$

- Pauli matrices

$$\mathbf{P}_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \mathbf{P}_y = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}, \quad \mathbf{P}_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

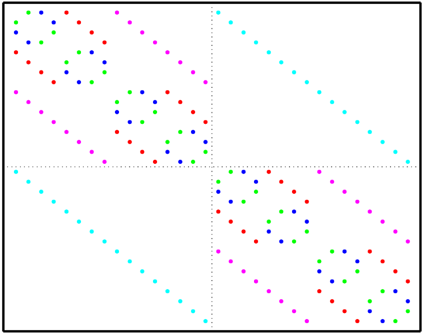
- $\mathbf{H} = \mathbf{H}_d + \mathbf{H}_c$, where

$$\mathbf{H}_d = \sum_{k_1,k_2=0}^{q-1} \gamma_{k_1,k_2} \mathbf{I}_{2^{k_1}} \otimes \mathbf{P}_z \otimes \mathbf{I}_{2^{k_2}} \otimes \mathbf{P}_z \otimes \mathbf{I}_{2^{q-k_1-k_2-2}}$$

$$\mathbf{H}_c = \sum_{k=0}^{q-1} \mathbf{I}_{2^k} \otimes \left(\alpha_k \mathbf{P}_x + \beta_k \mathbf{P}_y\right) \otimes \mathbf{I}_{2^{q-k-1}}$$

# Properties of $\mathbf{H}$

- $\mathbf{H}$ is sparse, hermitian and persymmetric
- $\mathbf{H}$ has the following sparsity pattern



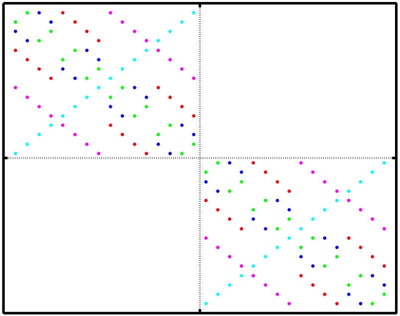
- Transformation to real symmetric matrix $\tilde{\mathbf{H}}$ possible:

$$\mathbf{H} = \mathbf{V}\tilde{\mathbf{H}}\mathbf{V}^*$$

# Transformation of $\mathbf{H}$

- $\tilde{\mathbf{H}}$ can be transformed to two real blocks of half size:

$$\begin{pmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{I} & -\mathbf{J} \end{pmatrix} \cdot \tilde{\mathbf{H}} \cdot \begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{J} & -\mathbf{J} \end{pmatrix} = \begin{pmatrix} \mathbf{H}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_2 \end{pmatrix} =$$



- Result: 2 problems of half size

# The exponential of a matrix

- **Definition:** The exponential of a matrix $\mathbf{H} \in \mathbb{C}^{n\times n}$ is defined by the infinite Taylor series

$$e^{\mathbf{H}} := \sum_{k=0}^{\infty} \frac{\mathbf{H}^k}{k!}$$

- **Method 1:** Computation of the matrix exponential by using a partial sum of the Taylor series

$$e^{\mathbf{H}} \approx S_m(\mathbf{H}) := \sum_{k=0}^{m} \frac{\mathbf{H}^k}{k!}$$

- **Error Estimate:**

$$\| e^{\mathbf{H}} - S_m(\mathbf{H}) \| \leq \left( \frac{\| \mathbf{H} \|^{m+1}}{(m+1)!} \right) \left( \frac{1}{1 - \| \mathbf{H} \| / (m+2)} \right) \leq \delta$$

# Properties of the matrix exponential

- The functional equation $e^{x+y} = e^x e^y$ does in general not hold for matrices:

  $$e^{A+B} = e^A \cdot e^B \Longleftrightarrow A \cdot B = B \cdot A$$

  *"The great matrix exponential tragedy"*
  (Moler, van Loan, 1978)

- **Trotter Product Formula:**

  $$\lim_{m \to \infty} \left( e^{\mathbf{A}/m} e^{\mathbf{B}/m} \right)^m = e^{\mathbf{A}+\mathbf{B}}$$

- $\left( e^{\mathbf{A}/m} \right)^m = e^{\mathbf{A}}$

- **Scaling & Squaring**

  $$e^{\mathbf{A}} = \left( e^{\mathbf{A}/2^k} \right)^{2^k}$$

# Nineteen dubious ways

Moler, van Loan: *Nineteen dubious ways to compute the exponential of a matrix*, 1978

- Series methods (Taylor, Padé)
- ODE-methods (Single step methods, Multistep methods)
- Polynomial methods(Cayley-Hamilton, Newton interpolation,...)
- Matrix decomposition methods(Eigen decomposition, Jordan canonical form, Schur)
- Splitting methods

# Nineteen dubious ways

Moler, van Loan: *Nineteen dubious ways to compute the exponential of a matrix,* ***25 years later*** , 2003

- Series methods (Taylor, Padé)
- ODE-methods (Single step methods, Multistep methods)
- Polynomial methods(Cayley-Hamilton, Newton interpolation,...)
- Matrix decomposition methods(Eigen decomposition, Jordan canonical form, Schur)
- Splitting methods
- **Krylov methods**

# Nineteen dubious ways

Moler, van Loan: *Nineteen dubious ways to compute the exponential of a matrix, 25 years later*, 2003

- Series methods (Taylor, **Padé**)
- ODE-methods (Single step methods, Multistep methods)
- Polynomial methods(Cayley-Hamilton, Newton interpolation,...)
- Matrix decomposition methods(**Eigen decomposition**, Jordan canonical form, Schur)
- Splitting methods
- Krylov methods

Comparison with **Chebyshev series** method

# Computation of the matrix exponential: eigendecomposition

- In the case of a diagonal matrix

$$\mathbf{H} = \mathrm{diag}(d_1, \ldots, d_n) = \begin{pmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{pmatrix}$$

it holds

$$e^{\mathbf{H}} = \mathrm{diag}(e^{d_1}, \ldots, e^{d_n}) = \begin{pmatrix} e^{d_1} & & \\ & \ddots & \\ & & e^{d_n} \end{pmatrix}$$

- If $\mathbf{H} = \mathbf{S}\mathbf{D}\mathbf{S}^{-1} = \mathbf{S}\left(\mathrm{diag}(d_1, \ldots, d_n)\right)\mathbf{S}^{-1}$ it follows

$$e^{\mathbf{H}} = S\left(\mathrm{diag}(e^{d_1}, \ldots, e^{d_n})\right)S^{-1}$$

- **Expensive part:** Computation of the eigen decomposition

# Computation of the matrix exponential: Padé approximation

- For $x \in \mathbb{C}$ the Padé approximation $r_m(x)$ of $e^x$ is given by

$$r_m(x) = \frac{p_m(x)}{q_m(x)}$$

  - $p_m(x) = \sum_{j=0}^{m} \frac{(2m-j)!m!x^j}{(2m)!(m-j)!j!}$
  - $q_m(x) = \sum_{j=0}^{m} \frac{(2m-j)!m!(-x)^j}{(2m)!(m-j)!j!}$

- **Generalization to matrices:**

$$e^{\mathbf{H}} \approx r_m(\mathbf{H}) = (q_m(\mathbf{H}))^{-1} p_m(\mathbf{H})$$

- Combination with Scaling & Squaring:

$$e^{\mathbf{H}} \approx \left[ \left( q_m \left( \frac{\mathbf{H}}{2^k} \right) \right)^{-1} p_m \left( \frac{\mathbf{H}}{2^k} \right) \right]^{2^k}$$

# Computation of the matrix exponential: Padé approximation

- Padé approximation:

$$e^{\mathbf{H}} \approx r_m(\mathbf{H}), \text{ where } q_m(\mathbf{H}) r_m(\mathbf{H}) = p_m(\mathbf{H})$$

- Suppose $m = 2l + 1$ odd:

$$p_{2l+1}(\mathbf{H}) = \underbrace{b_{2l}\mathbf{H}^{2l} + \cdots + b_2\mathbf{H}^2 + b_0\mathbf{I}}_{=:\mathbf{U}} + \underbrace{\mathbf{H}\left(b_{2l+1}\mathbf{H}^{2l} + \cdots + b_3\mathbf{H}^2 + b_1\mathbf{I}\right)}_{=:\mathbf{V}}$$

- Because of $p_m(-x) = q_m(x)$ we obtain $q_m(\mathbf{H}) = \mathbf{U} - \mathbf{V}$
- **Expensive part:** Linear system $(\mathbf{U} - \mathbf{V}) \cdot r_m(\mathbf{H}) = \mathbf{U} + \mathbf{V}$

# Computation of the matrix exponential: Chebyshev series expansion

- For $|x| \leq 1$ we have

$$e^x = J_0(i) + 2\sum_{k=1}^{\infty} i^k J_k(-i) T_k(x)$$

- $J_k$: Bessel function

$$J_k(z) = \sum_{l=0}^{\infty} \frac{(-1)^l}{2^{2l+k} l!(k+l)!} z^{2l+k}$$

- $T_k$: Chebyshev polynomial

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ T_{k+1}(x) &= 2xT_k(x) - T_{k-1}(x). \end{aligned}$$

# Computation of the matrix exponential: Chebyshev series expansion

- **Generalization to matrices:** For $\|\mathbf{H}\| \leq 1$

$$e^{\mathbf{H}} = J_0(i)\mathbf{I} + 2\sum_{k=1}^{\infty} i^k J_k(-i) T_k(\mathbf{H})$$

- In the case of $\mathbf{H}$ with arbitrary norm:
  Scaling & Squaring technique:

$$e^{\mathbf{H}} \approx \left[ J_0(i)\mathbf{I} + 2\sum_{k=1}^{m} i^k J_k(-i) T_k\left(\frac{\mathbf{H}}{2^k}\right)\right]^{2^k}$$

- $\Longrightarrow$ Only elementary matrix operations required!

# Evaluation of the matrix polynomial

- **Sparse case:** $T_{k+1}(\mathbf{H}) = 2\mathbf{H}T_k(\mathbf{H}) - T_{k-1}(\mathbf{H})$
  $\Longrightarrow$ Only products of the form Sparse * Dense
- **General (dense) case:** Paterson, Stockmeyer, 1973
  Efficient evaluation of matrix polynomials

# Evaluation of the matrix polynomial

- **Sparse case:** $T_{k+1}(\mathbf{H}) = 2\mathbf{H}T_k(\mathbf{H}) - T_{k-1}(\mathbf{H})$
  $\Longrightarrow$ Only products of the form Sparse * Dense
- **General (dense) case:** Paterson, Stockmeyer, 1973
  Efficient evaluation of matrix polynomials

**Example:**

$$p(\mathbf{A}) = \alpha_6\mathbf{A}^6 + \alpha_5\mathbf{A}^5 + \alpha_4\mathbf{A}^4 + \alpha_3\mathbf{A}^3 + \alpha_2\mathbf{A}^2 + \alpha_1\mathbf{A} + \alpha_0\mathbf{I}$$

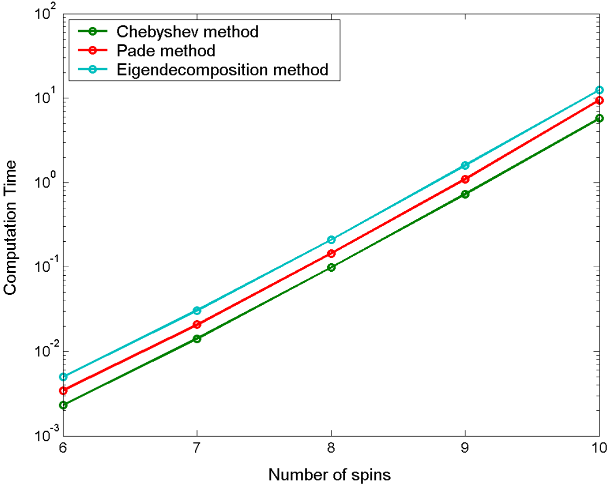
- **Horner scheme** requires **5** non-scalar multiplications:

$$(((((\alpha_6\mathbf{A} + \alpha_5\mathbf{I}) \cdot \mathbf{A} + \alpha_4\mathbf{I}) \cdot \mathbf{A} + \alpha_3\mathbf{I}) \cdot \mathbf{A} + \alpha_2\mathbf{I}) \cdot \mathbf{A} + \alpha_1\mathbf{I}) \cdot \mathbf{A} + \alpha_0\mathbf{I}$$

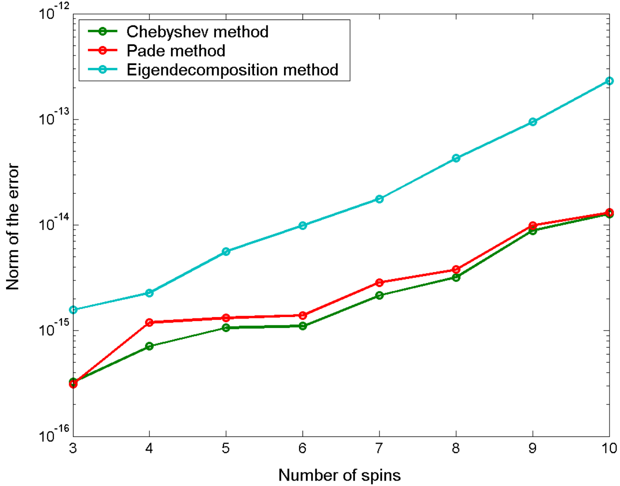
- Optimal method: (Only **3** matrix-matrix-products required)

$$\mathbf{A}_2 = \mathbf{A} \cdot \mathbf{A}, \quad \mathbf{A}_3 = \mathbf{A}_2 \cdot \mathbf{A}$$
$$p(\mathbf{A}) = \alpha_0\mathbf{I} + \alpha_1\mathbf{A} + \alpha_2\mathbf{A}_2 + \alpha_3\mathbf{A}_3 + \mathbf{A}_3 \cdot (\alpha_4\mathbf{A} + \alpha_5\mathbf{A}_2 + \alpha_6\mathbf{A}_3)$$

# Comparison of the methods: Computation time



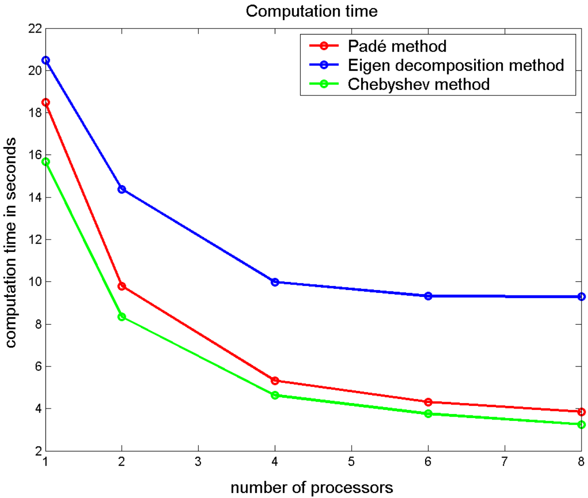
# Comparison of the methods: accuracy



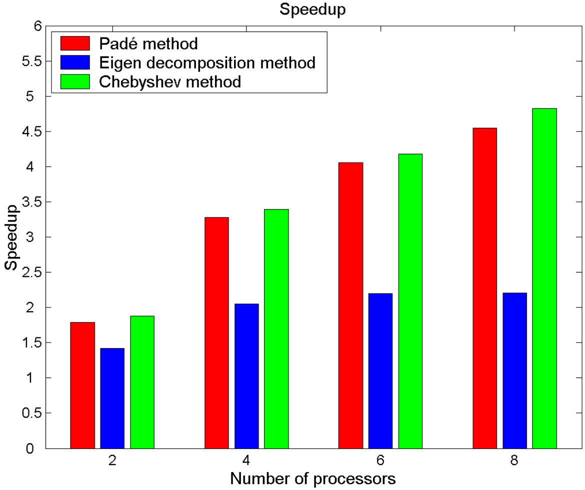
# Numerical results on parallel platform

- System size: 2048-by-2048
- Using Intel MKL BLAS and LAPACK libraries
- Platform:
  - Dual-Socket Intel Xeon
  - X5355 processors (Quad-Cores, 2,66 GHz and 2x4 MB Level Cache, Core 2 Duo)
  - 6 GB RAM

# Computation time on parallel architecture



# Speedup


Speedup
Padé method
Eigen decomposition method
Chebyshev method
Speedup
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
2
4
6
8
Number of processors

# Final conclusions

- Computation of the matrix exponential an intrinsically hard problem
- Chebyshev method very efficient (accuracy and computation time)
  - Good convergence properties
  - Only elementary matrix operations required
  - Efficient evaluation of matrix polynomials possible
  - Good in parallel

# Joint work

## Joint work with

- Thomas Huckle (Computer science department, TU München)
- Andreas Spörl, Thomas Schulte-Herbrüggen (Chemistry department, TU München)

**Thank you very much for your attention!**