

Accelerators for the SABRE method with ELT scale AO systems

Elisabeth Brunner, Coen de Visser, Michel Verhaegen
Delft Center of Systems and Control

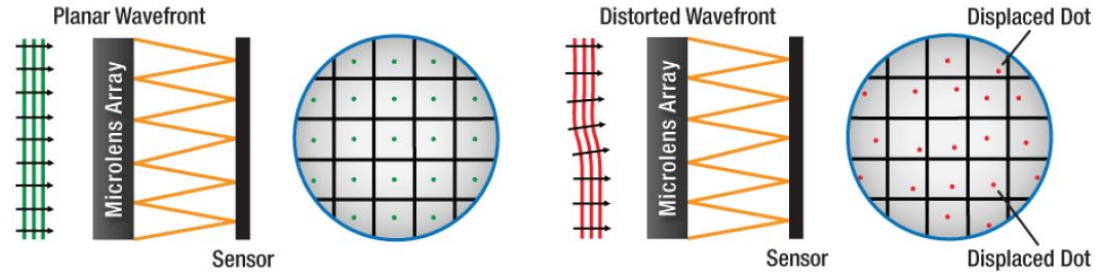
AO RTC workshop, Paris
27th January 2016

Content

- 1) SABRE – Spline Based Aberration reconstruction
 - Basics and Perks
- 2) The Distributed D-SABRE framework
- 3) First Benchmarking with CuRe-D
 - Comparison in YAO simulations
 - Open and closed loop results
- 4) Complexity, Speed, Latency estimates for D-SABRE
- 5) Accelerators – Hardware and Solvers (ongoing work)
- 6) Outlook and questions

The SABRE method

Basics



- SH slope measurements

$$s_x(x, y) = \frac{\partial \phi(x, y)}{\partial x}, \quad s_y(x, y) = \frac{\partial \phi(x, y)}{\partial y}$$

- B-spline wavefront model

$$\phi(x, y) \approx p^d(x, y) = \mathbf{B}^d(x, y)\mathbf{c}$$

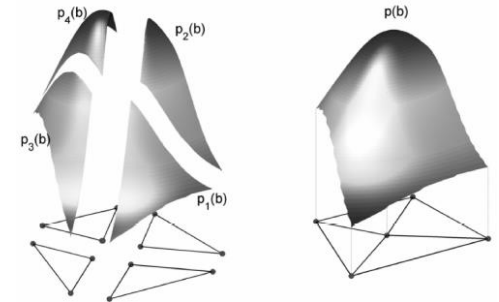
- Sparse regression matrix

$$\mathbf{y} = \mathbf{D}\mathbf{c} + \mathbf{n}$$

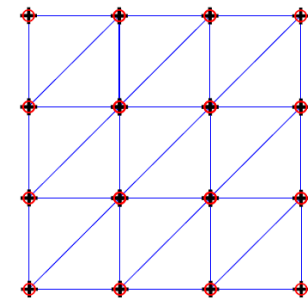
- Constrained least squares problem

$$\arg \min \|\mathbf{y} - \mathbf{D}\mathbf{c}\|_2^2 \quad \text{subject to } \mathbf{A}\mathbf{c} = 0$$

B-spline principle



Triangulation



The SABRE method

Perks

SH WFR problem

$$\arg \min \|\mathbf{y} - \mathbf{D}\mathbf{c}\|_2^2 \quad \text{subject to } \mathbf{A}\mathbf{c} = 0$$

Wavefront model

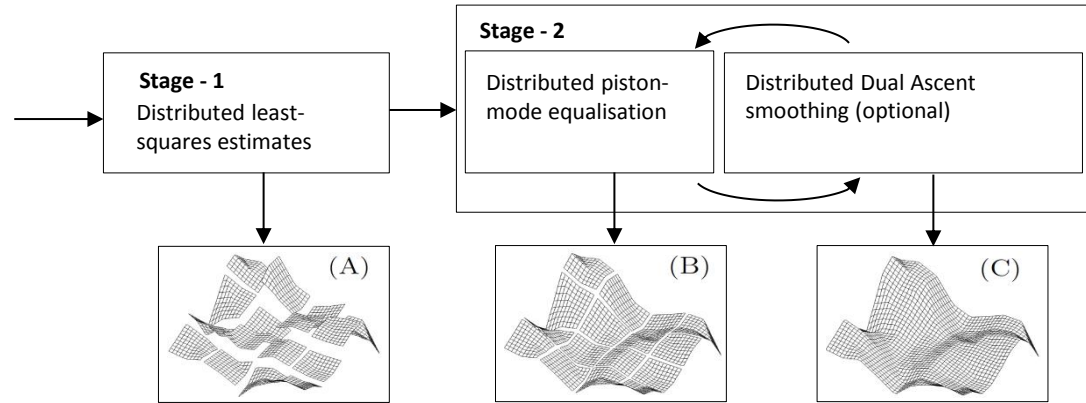
$$\phi(x, y) \approx p^d(x, y) = \mathbf{B}^d(x, y)\mathbf{c}$$

- Sparse regression matrix \mathbf{D}
- Sparse constraint matrix \mathbf{A}
- Basis functions \mathbf{B}^d with local support
- B-coefficients \mathbf{c} give analytical WF estimate

The B-spline framework

- 1) Analytical solution optimal in a least squares sense
- 2) Triangulations provide geometric flexibility
- 3) Local basis functions allow distributed solution

The Distributed SABRE Framework



- Stage – 1: Distributed least squares estimation

$$\arg \min \|\mathbf{y}_i - \mathbf{D}_i \mathbf{c}_i\|_2^2 \quad \text{subject to } \mathbf{A}_i \mathbf{c}_i = 0$$

$$\hat{\mathbf{c}}_i = (\mathbf{D}_{i,\mathbf{A}}^T \mathbf{D}_{i,\mathbf{A}})^{-1} \mathbf{D}_{i,\mathbf{A}}^T \mathbf{y}_i, \quad 1 \leq i \leq G.$$

- Stage – 2: Distributed, iterative piston mode equalisation (+ optional smoothing)

$$k_i(l+1) = \max\{\mathbf{c}_{i,m} - \mathbf{c}_{m,i}\}, \quad m \in \mathcal{M}_i$$

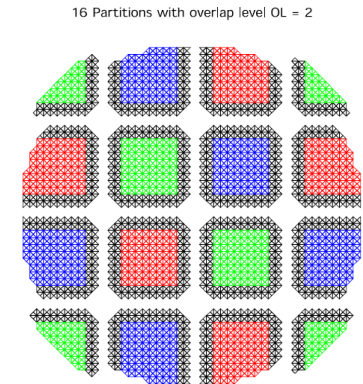
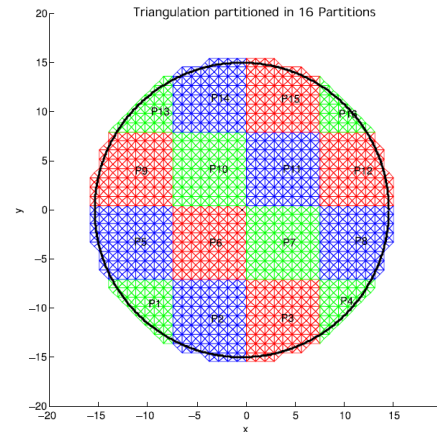
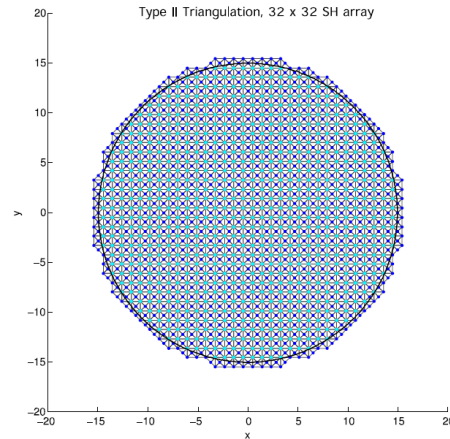
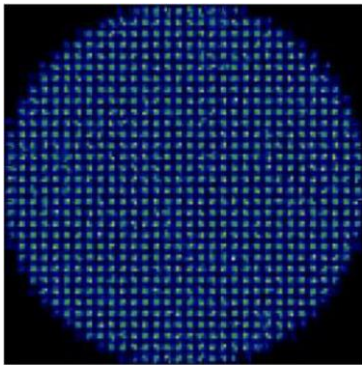
$$\mathbf{c}_i(l+1) = \mathbf{c}_i(l) + k_i(l+1)$$

$$\mathbf{c}_i(l+1) = \hat{\mathbf{c}}_i(0) + \mathbf{z}_i^T(l) \mathbf{A}_i$$

$$\mathbf{z}_{i,m}(l+1) = \mathbf{z}_{i,m}(l) + \alpha(l) \mathbf{A}_{i,m} \mathbf{c}_{i,m}(l)$$

The Distributed SABRE

YAO Simulations



Yao simulation specs

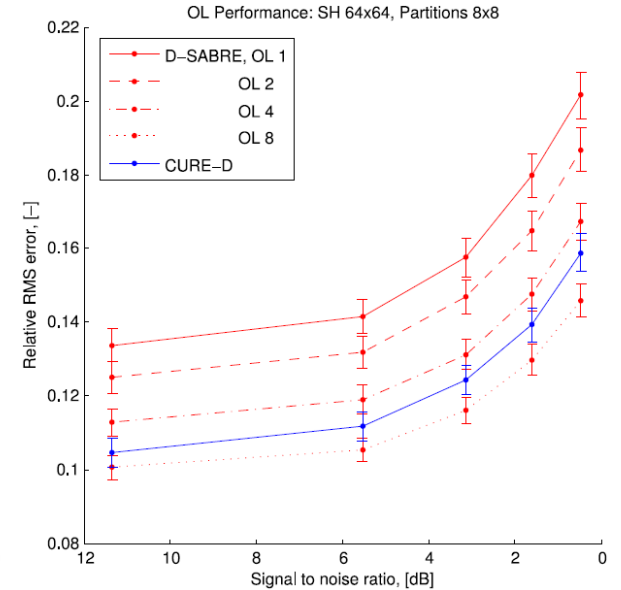
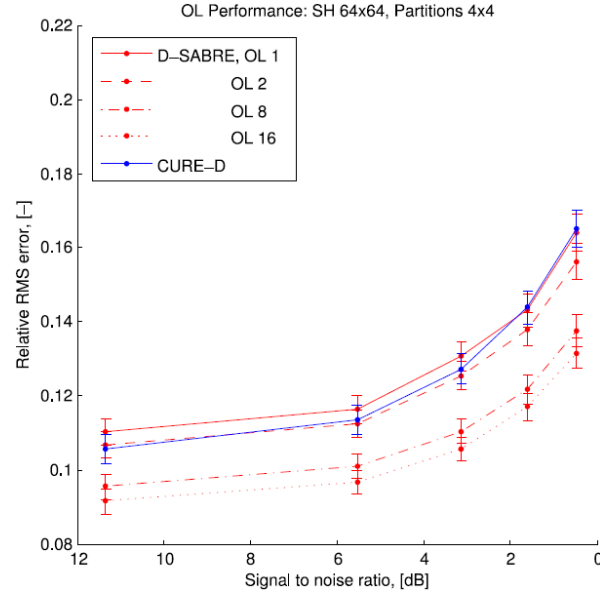
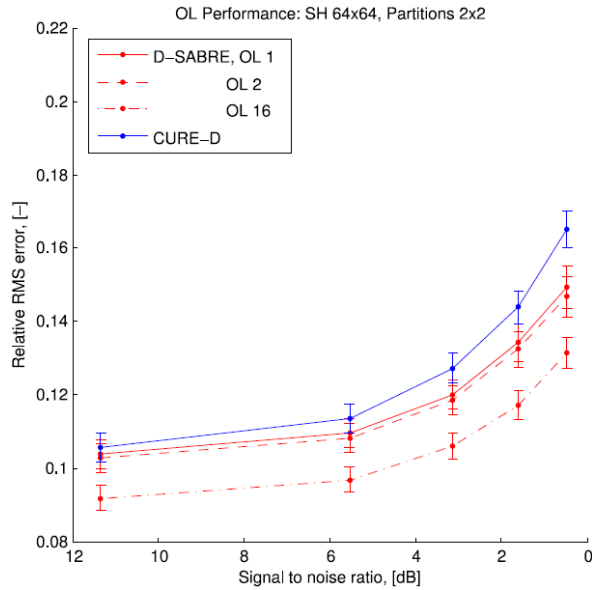
- 64 x 64 SH sensor, minimum illumination 50%
- 33 x 33 actuator stackarray DM, actuator coupling 20%

Scope of the analysis

- Test baseline D-SABRE ($d = 1$, $r = 0$, only D-PME, partition overlap OL)
- Compare with CuRe-D as truly distributed method (i.e. only local data used)
- Focus on equal conditions rather simulating XAO systems

Benchmark CuRe-D

Open-loop results



Settings:

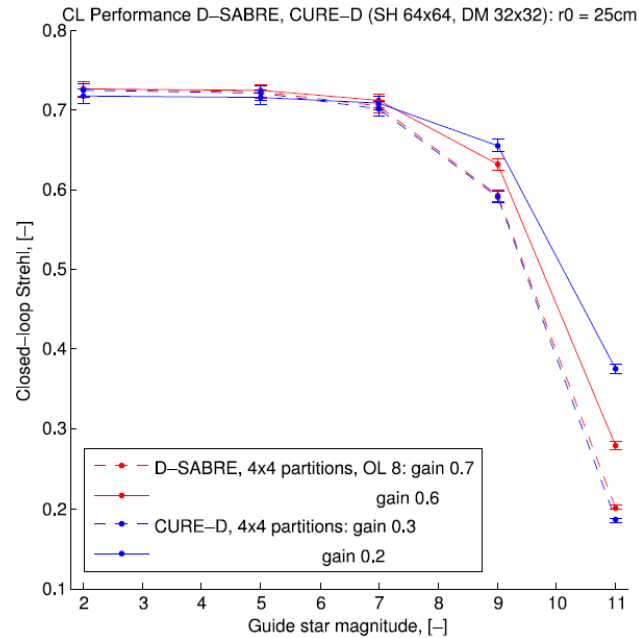
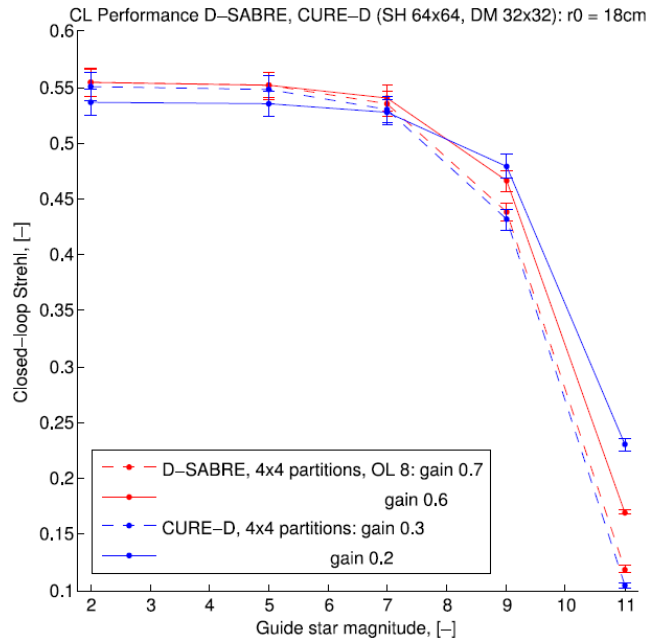
- Additive white noise on YAO slope measurements
- Turbulence of $r_0 = 18\text{cm}$, 100 realizations
- Tested for 4, 16 and 64 partitions

Findings:

- No noise propagation within partitions
- Performance loss for strong decomposition
- Counter act through increased overlap

Benchmark CuRe-D

Closed-loop results



Settings:

- Photon shot noise included
- Least-squares map: phase to actuator space
- Loop evaluated for 200 iterations
- Turbulence of $r_0 = 18\text{cm}$, 15cm ; 10 realizations

Findings:

- Open -loop results confirmed
- D-SABRE more robust to loop-gain variations

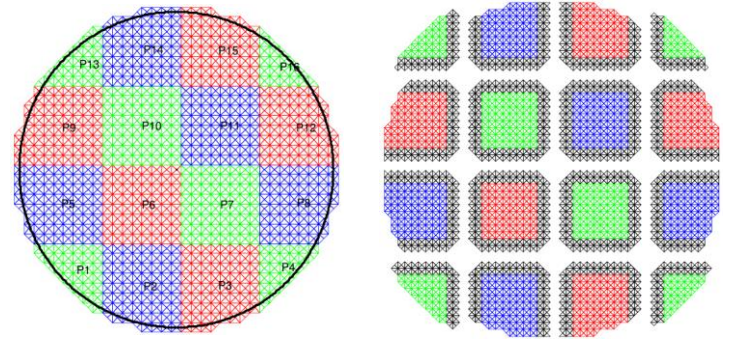
The Distributed SABRE

Potential & To do list

- 1) Improvements on the D-SABRE
 - Adapt sensor model to measurements of averaged local gradients
 - Revise D-PME procedure for approximation error propagation
- 2) **Suitable Hardware/Software accelerators**
 - **GPU implementation**
 - **Sparse iterative SABRE**
- 3) Distributed phase-DM mapping
 - Aim: 1 Step slope to actuator method
 - B-spline model of the actuator influence function space

The Distributed SABRE

Theoretical Speed-up



N subapertures
 G partitions

- Centralised SABRE

$$\begin{aligned} \mathcal{C}_{SABRE} &= \mathcal{O}(2NJ\hat{d}) && \text{for } J \text{ simplices, } \hat{d} \text{ coefficients per simplex} \\ &= \mathcal{O}(12N^2) && \text{for } J = 2N, \hat{d} = 3 \end{aligned}$$

- Speed-up D-SABRE

$$\tilde{\mathcal{S}}_{DSABRE} = \mathcal{O}\left(\frac{\rho^2 + R/J_{\Omega_i}}{G^2}\right) \text{ for } J = GJ_{\Omega_i}, \rho = \frac{J_i}{J_{\Omega_i}}, \rho \geq 1$$

- Complexity D-SABRE

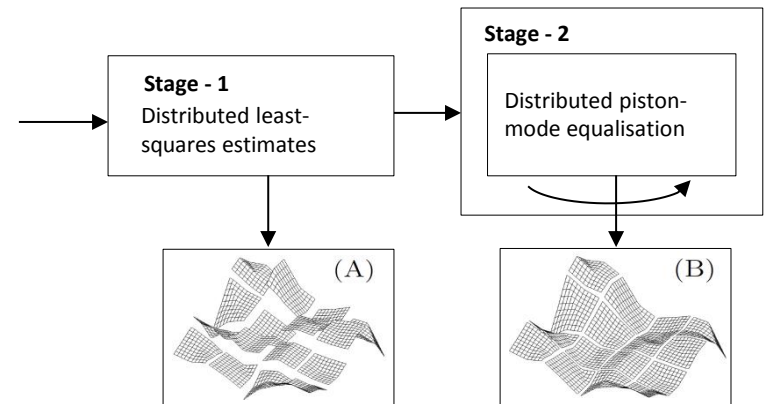
$$\mathcal{C}_{DSABRE} = \tilde{\mathcal{S}}_{DSABRE} \mathcal{C}_{SABRE}$$

ELT scale example:

$$\left. \begin{array}{l} N = 200 \times 200 \\ G = 20 \times 20 \\ OL = 2 \end{array} \right\} \mathcal{C}_{DSABRE} = \mathcal{O}(11N) \text{ for } R/J_i \ll \rho^2$$

Accelerators

Suitable for GPU implementation ?



– Speed and latency baseline (sequential C implementation)

($N = 200 \times 200$, $G = 20 \times 20$, $OL = 2$)

1) **Computation** (i5-4690 CPU)

Local reconstruction: 150 ms > single partition 0.39 ms

D-PME (18 iterations): 51 ms

2) **Data**

Spline data, Rec. Matrices, etc.: 1 GByte (very conservative)

– Conclusions, Challenges, Plans

1) Heaviest computation is MVM in local reconstruction

2) Latency for D-PME on GPU

– No communication over PCI-express necessary, all on GPU

– Issue might lay in start up times

– Require dedicated CUDA implementation

3) Rolling shutter to hide communication with CPU for slope data update

Accelerators

Sparse iterative SABRE

N subapertures
 J Simplices
 \hat{d} coefficients
($J = 2N$, $\hat{d} = 3$)

- Projected least squares problem

$$\arg \min \|\mathbf{y} - \mathbf{D}_A \mathbf{c}\|_2^2$$

1) Pseudo inverse solution (MVM)

$$\hat{\mathbf{c}} = (\mathbf{D}_A^T \mathbf{D}_A)^{-1} \mathbf{D}_A^T \mathbf{y}$$

- Complexity

$$\begin{aligned} C_{SABRE} &= \mathcal{O}(2NJ\hat{d}) \\ &= \mathcal{O}(12N^2) \end{aligned}$$

- Fine grid parallelization: CUBLAS

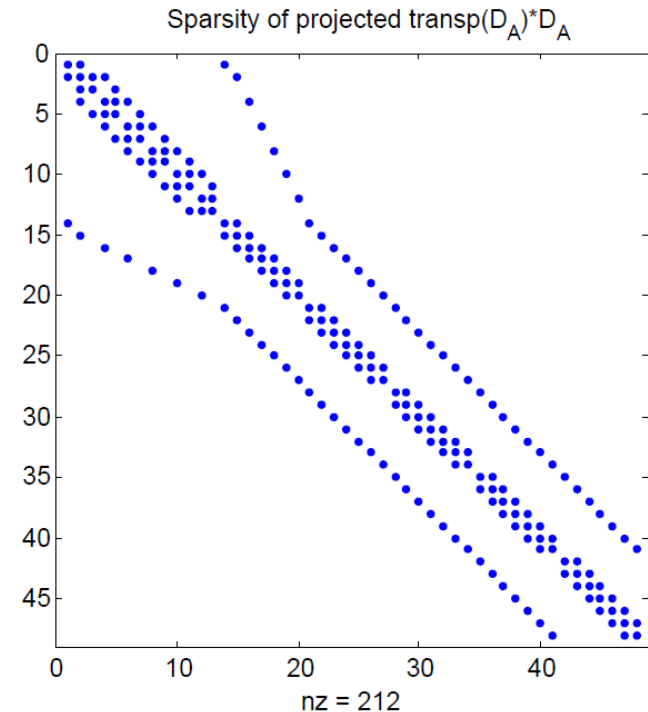
2) Sparse iterative solution (CG type)

$$(\mathbf{D}_A^T \mathbf{D}_A) \hat{\mathbf{c}} = \mathbf{D}_A^T \mathbf{y}$$

- Complexity

$$\begin{aligned} C_{SABRE}^{sparse} &= \mathcal{O}(4J\hat{d}) + \mathcal{O}(kJ\hat{d}), \\ &= \mathcal{O}((24 + 6k)N) \end{aligned}$$

- Fine grid parallelization: Paralution



Summary, Outlook, Questions

1) Distributed SABRE framework

- Comparison of baseline D-SABRE with CuRe-D

2) Complexity, Speed, Latency

- Local reconstruction MVM computationally most expensive
- No communication over PCI-e needed for D-PME
- Start up times of GPU issue in iterative D-PME ?

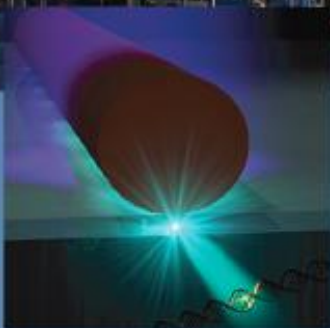
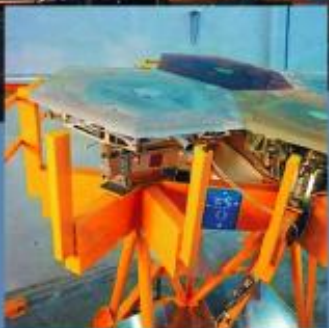
3) Solutions / Outlook

- Hide latency of data transfer in „rolling shutter“
- Sparse iterative solutions to local reconstruction

Open Window on the Future of Smart Optics Systems

March 3-4, 2016,
Aula Congress Centre, TU Delft, the Netherlands

 **TU Delft**



- 4 excellent guest speakers
- Opening of the Smart Optics Lab
- Inaugural speech Prof. dr. Gleb Vdovin
- 1 day school on smart optics technologies *