A Course on Distributed Control

Anders Rantzer and Enrico Lovisari

Schedule

```
Mo April  8 at 1315-1430 lecture
Mo April 15 at 1315-1600 lecture and exercises
Fr April 26 at 0815-1100 lecture and exercises
We   May  8 at 0915-1200 lecture and exercises
We   May 15 at 0915-1200 lecture and exercises
Mo   May 20 at 1315-1600 lecture and exercises
Mo   May 27 at 1315-1500 exercises
```

A Course of Six Lectures

```
1. Introduction
   Fixed modes, Team theory, Witsenhausen's counterexample

2. Partial nestedness and quadratic invariance
   Control with information delays
   Example: Tele-operation

3. Dual decomposition
   The saddle algorithm
   Example: The Internet protocol

4. Distributed MPC
   Example: Water Supply Network

5. Distributed control of positive systems. Consensus algorithms

6. Spatially invariant systems.
```

Lecture 4

- More on dual decompostion
- Distributed MPC
- Gradient methods for large-scale systems

Last week: Dual decomposition

$$\min_{z_i}[V_1(z_1,z_2)+V_2(z_2)+V_3(z_3,z_2)]$$
$$=\max_{p_i}\min_{z_i,v_i}\Big[V_1(z_1,v_1)+V_2(z_2)+V_3(z_3,v_3)+p_1(z_2-v_1)+p_3(z_2-v_3)\Big]$$

The optimum is a Nash equilibrium of the following game:

The three computers try to minimize their respecive costs

Computer 1: $\min_{z_1,v_1}\left[V_1(z_1,v_1)-p_1v_1\right]$

Computer 2: $\min_{z_2}\left[V_2(z_2)+(p_1+p_3)z_2\right]$

Computer 3: $\min_{z_3,v_3}\left[V_3(z_3,v_3)-p_3v_3\right]$

while the "market makers" try to maximize their payoffs

Between computer 1 and 2: $\max_{p_1}\left[p_1(z_2-v_1)\right]$

Between computer 2 and 3: $\max_{p_3}\left[p_3(z_2-v_3)\right]$

The saddle point algorithm

Update in gradient direction:

Computer 1: $\begin{cases}\dot z_1 = -\partial V_1/\partial z_1\\ \dot v_1 = -\partial V_1/\partial z_2 + p_1\end{cases}$

Computer 1 and 2: $\dot p_1 = z_2 - v_1$

Computer 2: $\dot z_2 = -\partial V_2/\partial z_2 - p_1 - p_3$

Computer 2 and 3: $\dot p_3 = z_2 - v_3$

Computer 3: $\begin{cases}\dot z_3 = -\partial V_3/\partial z_3\\ \dot v_3 = -\partial V_3/\partial z_2 + p_3\end{cases}$

Globally convergent if V_i are convex!
[Arrow, Hurwicz, Usawa 1958]

A long history

The saddle algorithm:
Arrow, Hurwicz, Usawa 1958

Books on control and coordination in hierarchical systems:
Mesarovic, Macko, Takahara 1970
Singh, Titli 1978
Findeisen 1980

Major application to water supply network:
Carpentier and Cohen, Automatica 1993

Case study: A water supply network in Paris

[Carpentier and Cohen, Automatica 1993]

- Network serving about 1 million inhabitants
- 20 main water reservoirs
- 30 pumping stations
- 13 peripheral subnetworks

Challenges for control

- Minimize cost for pumping
- Bounds on reservoirs
- Bounds and delays in pumping power
- Prediction of consumption

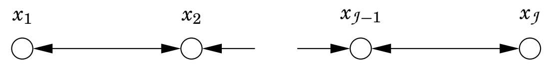
Optimal control using dual decomposition and saddle algorithm
Subnetworks separated by two variables: Water flow and price

Important Aspects of Dual Decomposition

- Very weak assumptions on graph
- No need for central coordination
- Natural learning procedure is globally convergent
- Unique Nash equilibrium corresponds to global optimum

Conclusion: Ideal for distributed control synthesis

A control problem with graph structure



$$\begin{bmatrix} x_1(\tau+1) \\ x_2(\tau+1) \\ \vdots \\ x_{\mathcal{J}}(\tau+1) \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & & 0 \\ A_{21} & \ddots & \ddots & \\ & \ddots & \ddots & A_{(\mathcal{J}-1)\mathcal{J}} \\ 0 & & A_{\mathcal{J}(\mathcal{J}-1)} & A_{\mathcal{J}\mathcal{J}} \end{bmatrix} \begin{bmatrix} x_1(\tau) \\ x_2(\tau) \\ \vdots \\ x_{\mathcal{J}}(\tau) \end{bmatrix} + \begin{bmatrix} u_1(\tau) \\ u_2(\tau) \\ \vdots \\ u_{\mathcal{J}}(\tau) \end{bmatrix}$$

Minimize the convex objective $\sum_{t=0}^{N} \underbrace{\sum_{i=1}^{\mathcal{J}} \ell_i(x_i(\tau), u_i(\tau))}_{\ell(x(\tau),u(\tau))}$

with convex constraints $x_i(\tau) \in X_i$, $u_i(\tau) \in U_i$ and $x(0) = \bar{x}$.

Decomposing the problem

Minimize $\sum_{t=0}^{N} \ell(x(\tau), u(\tau))$

subject to

$$\begin{bmatrix} x_1(\tau+1) \\ x_2(\tau+1) \\ \vdots \\ x_{\mathcal{J}}(\tau+1) \end{bmatrix} = \begin{bmatrix} A_{11}x_1(\tau) \\ A_{22}x_2(\tau) \\ \vdots \\ A_{\mathcal{J}\mathcal{J}}x_{\mathcal{J}}(\tau) \end{bmatrix} + \begin{bmatrix} v_1(\tau) \\ v_2(\tau) \\ \vdots \\ v_{\mathcal{J}}(\tau) \end{bmatrix} + \begin{bmatrix} u_1(\tau) \\ u_2(\tau) \\ \vdots \\ u_{\mathcal{J}}(\tau) \end{bmatrix}$$

where $x(0) = \bar{x}$ and

$$v_i = \sum_{j \neq i} A_{ij} x_j$$

holds for all i.

Decomposing the Cost Function

$$\max_p \min_{u,v,x} \sum_{\tau=0}^{N} \sum_{i=1}^{\mathcal{J}} \left[\ell_i(x_i, u_i) + p_i^T \left(v_i - \sum_{j \neq i} A_{ij} x_j \right) \right]$$

$$= \max_p \sum_i \min_{u_i, x_i} \sum_{\tau=0}^{N} \underbrace{\left[\ell_i(x_i, u_i) + p_i^T v_i - x_i^T \left(\sum_{j \neq i} A_{ji}^T p_j \right) \right]}_{\ell_i^p(x_i, u_i, v_i)}$$

so, given the sequences $\{p_j(t)\}_{t=0}^{N}$, agent i should minimize

$$\underbrace{\sum_{\tau=0}^{N} \ell_i(x_i, u_i)}_{\text{local cost}} + \overbrace{\sum_{\tau=0}^{N} p_i^T v_i}^{\text{what he expects others to charge him}} - \underbrace{\sum_{\tau=0}^{N} x_i^T \left(\sum_{j \neq i} A_{ji}^T p_j \right)}_{\text{what he is payed by others}}$$

subject to $x_i(t+1) = A_{ii} x_i(t) + v_i(t) + u_i(t)$ and $x_i(0) = \bar{x}_i$.

Distributed Optimization Procedure

Local optimizations in each node

$$V_i^{N,p}(\bar{x}_i) = \min_{u_i, x_i} \sum_{\tau=0}^{N} \ell_i^p(x_i(\tau), u_i(\tau), v_i(\tau))$$

can be coordinated by (local) gradient updates of the prices

$$p_i^{k+1}(\tau) = p_i^k(\tau) + \gamma_i^k \left[v_i^k(\tau) - \sum_{j \neq i} A_{ij} x_j^k(\tau) \right]$$

Future prices included in negotiation for first control input!

Convergence guaranteed under different types of assumptions on the step size sequence γ_i^k.

Distributed Optimality Conditions

Suppose that $Q_i, R_i > 0$ for $i = 1, \ldots, \mathcal{J}$. Then the saddle point

$$\max_p \min_{u,v,x} \sum_{t=0}^{N} \sum_{i=1}^{\mathcal{J}} \left[|x_i|_{Q_i}^2 + |u_i|_{R_i}^2 + 2p_i^T \left(v_i - \sum_{j \neq i} A_{ij} x_j \right) \right]$$

with minimization over system dynamics with $x_i(0) = x_i^0$ and maximization with $p(N) = 0$, is uniquely defined by

$$u_i(t) = R_i^{-1} B_i^T p_i(t) \qquad v_i(t) = \sum_{j \neq i} A_{ij} x_j(t)$$

$$p_i(t-1) = \sum_j A_{ij}^T p_j(t) - Q_i x_i(t)$$

Notice:
Similarity with Pontryagin's maximum principle in discrete time
Future prices are relevant for consensus about todays control

Dynamics in vector form

$$\underbrace{\begin{bmatrix} x_i(N) \\ \vdots \\ x_j(1) \\ x_i(0) \end{bmatrix}}_{\mathbf{x}_i} = \sum_j \underbrace{\begin{bmatrix} 0 & A_{ij} & & 0 \\ & \ddots & \ddots & \\ & & \ddots & A_{ij} \\ 0 & & & 0 \end{bmatrix}}_{\mathbf{A}_i} \underbrace{\begin{bmatrix} x_j(N) \\ \vdots \\ x_j(1) \\ x_j(0) \end{bmatrix}}_{\mathbf{x}_i} + \underbrace{\begin{bmatrix} 0 & B_i & & 0 \\ & \ddots & \ddots & \\ & & \ddots & B_i \\ 0 & & & 0 \end{bmatrix}}_{\mathbf{B}_i} \underbrace{\begin{bmatrix} u_i(N) \\ \vdots \\ u_i(1) \\ u_i(0) \end{bmatrix}}_{\mathbf{u}_i} + \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ x_i^0 \end{bmatrix}}_{\mathbf{x}_i^0}$$

Optimality conditions in vector form

Let $\mathbf{p}_i$, $\mathbf{u}_i$ and $\mathbf{x}_i$ be the vectors of prices, inputs and states for $t = 0, 1, 2, \ldots, N$. Then

$$\min_{\mathbf{u},\mathbf{x}} \sum_{i=1}^{\mathcal{J}} \left(|\mathbf{x}_i|_{Q_i}^2 + |\mathbf{u}_i|_{R_i}^2 \right) \text{ subject to } \mathbf{x}_i = \sum_j \mathbf{A}_{ij} \mathbf{x}_j + \mathbf{B}_i \mathbf{u}_i + \mathbf{x}_i^0$$

$$= \max_{\mathbf{p}} \min_{\mathbf{u},\mathbf{x}} \sum_{i=1}^{\mathcal{J}} \left[|\mathbf{x}_i|_{Q_i}^2 + |\mathbf{u}_i|_{R_i}^2 - 2\mathbf{p}_i^T \left(\sum_j \mathbf{A}_{ij} \mathbf{x}_j + \mathbf{B}_i \mathbf{u}_i - \mathbf{x}_i \right) \right]$$

Differentiation with respect to $\mathbf{p}$, $\mathbf{u}$ and $\mathbf{x}$ gives the saddle point

$$\mathbf{x}_i = \sum_j \mathbf{A}_{ij} \mathbf{x}_j + \mathbf{B}_i \mathbf{u}_i + \mathbf{x}_i^0$$
$$\mathbf{u}_i = R_i^{-1} \mathbf{B}_i^* \mathbf{p}_i$$
$$\mathbf{p}_i = \sum_j \mathbf{A}_{ij}^* \mathbf{p}_j - Q_i \mathbf{x}_i$$

How do we reach this equilibrium by a distributed algorithm?

Solution by saddle algorithm

The equilibrium equations

$$\mathbf{x}_i = \sum_j \mathbf{A}_{ij}\mathbf{x}_j + \mathbf{B}_i\mathbf{u}_i + \mathbf{x}_i^0$$
$$\mathbf{u}_i = R_i^{-1}\mathbf{B}_i^*\mathbf{p}_i$$
$$\mathbf{p}_i = \sum_j \mathbf{A}_{ij}^*\mathbf{p}_j - Q_i\mathbf{x}_i$$

can be solved distributively by iteration:

$$\mathbf{x}_i := Q_i^{-1}\left(\mathbf{p}_i - \sum_j \mathbf{A}_{ij}^*\mathbf{p}_j\right)$$
$$\mathbf{u}_i = R_i^{-1}\mathbf{B}_i^*\mathbf{p}_i$$
$$\mathbf{p}_i^+ := \mathbf{p}_i - \gamma_i\left(\sum_j \mathbf{A}_{ij}\mathbf{x}_j + \mathbf{B}_i\mathbf{u}_i + \mathbf{x}_i^0 - \mathbf{x}_i\right)$$

Negotiation of future prices needed to decide first control input!

Lecture 4

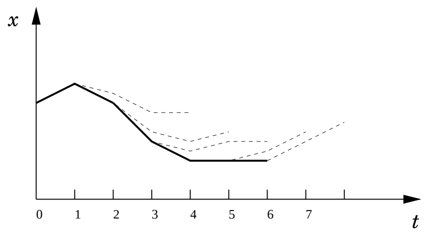
- More on dual decompostion
- **Distributed MPC**
- Gradient methods for large-scale systems

Idea of Distributed Model Predicitve Control

Replace the original problem by iterative online solutions of the decentralized finite horizon problem

$$\min_{x_i,u_i} \sum_{t=0}^{N} l_i^p(x_i(t), u_i(t), v_i(t))$$

Two sources of error: Finite horizon and non-optimal prices



A Distributed MPC Algorithm

At time t:

1. Measure the states $x_i(t)$ locally.
2. Use gradient iterations to generate
   - price prediction sequences $\{p_i(t,\tau)\}_{\tau=0}^N$
   - state prediction sequences $\{x_i(t,\tau)\}_{\tau=1}^N$
   - input prediction sequences $\{u_i(t,\tau)\}_{\tau=1}^N$

   warm-starting from predictions at time $t-1$.
3. Apply the inputs $u_i(t) = u_i(t,0)$.

Important parameters: Prediction horizons N_i, number of gradient iterations K_i and gradient step sizes γ_i.

Fixed or flexible parameters N_i, K_i, γ_i?

Fixed parameters

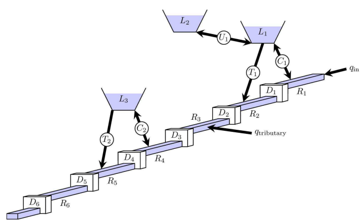
- Simpler implementation
- Linear plant, quadratic cost gives distributed LTI controllers
- Can be analyzed off-line or on-line

Flexible parameters

- Useful to handle hard state constraints
- Can speed up on-line computations
- Can slow down on-line computations

Hydro Power Valley

Benchmark in EU-project HD-MPC coordinated from Delft



Equipped with
10 turbines $(T_1, T_2, D_1, \ldots, D_6, C_{1t}, C_{2t})$ and 2 pumps (C_{1p}, C_{2p})
3 reservoirs (lakes)
6 dams and reaches

Objectives: Follow power-reference. Avoid flooding.

Hydro Power Valley - modeling

Modeling:

1. Saint Venant PDE (mass and volume balance)
2. Spatial discretization (MOL) ⇒ non-linear ODE:s (249 states, 12 inputs, used as simulation model)
3. Linearization, discretization (h=30 min) and model reduction ⇒ MPC-model (32 states, 12 inputs)

Hydro Power Valley - control

Difficulties:

- Non-linear power-production $p(t) = u(t)^T S_i x(t)$
  - Linearize around nominal working point (x_0, u_0), $\Delta p = u_0^T S\Delta x + x_0^T S^T \Delta u$
- Non-linear constraints; $u_{C_{it}} u_{C_{ip}} = 0,\ i = 1,2$
  - We have $u_{C_{it}} \geq 0$ and $u_{C_{ip}} \leq 0$, penalize $-u_{C_{it}} u_{C_{ip}}$

Cost function:

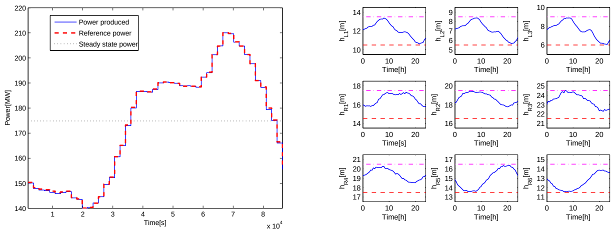
$$\sum_{t=0}^{N-1}\left(\frac{1}{2}\begin{bmatrix}\Delta x(t)\\ \Delta u(t)\end{bmatrix}^T H \begin{bmatrix}\Delta x(t)\\ \Delta u(t)\end{bmatrix} + \gamma\left\| P\begin{bmatrix}\Delta x(t)\\ \Delta u(t)\end{bmatrix} - \Delta p_{ref}(t)\right\|_1\right)$$

with $P = [u_0^T S \;\; x_0^T S^T]$

Simulation results

Control horizon: $N = 10$ (5 hours)

Figure : Power reference tracking (left) and Dam water levels (right)



Notation

For a distributed accuracy test, let $\bar{V}_i^p(x_i)$ be an upper bound on

$$\min_{u_i, v_i, x_i} \sum_{\tau=0}^{\infty} \ell_i^p(x_i(\tau), u_i(\tau), v_i(\tau))$$

Such an upper bound can for example be computed by minimization over a finite time horizon with a terminal constraint at the origin.

Theorem on accuracy of distributed MPC

Suppose $x(t+1) = Ax(t) + Bu(t)$ for $t \geq 0$ and for some p that

$$\begin{aligned} &V_i^{N_i(t),p(t,\cdot)}(x_i(t)) + (1-\alpha)\ell_i(x_i(t), u_i(t)) \\ \geq\, &\bar{V}_i^{p(t,\cdot)}(x_i(t+1)) + \ell_i^{p(t,\cdot)}(x_i(t), u_i(t), \textstyle\sum_{j \neq i} A_{ij} x_j(t)) \end{aligned}$$

for all i and t. Then

$$\alpha \sum_{t=0}^{\infty} \ell(x(t), u(t)) \leq V^{\infty}(\bar{x})$$

Notice: Failure of inequality hints on update of N_i or K_i!

Conclusions on Distributed MPC

We have synthesized a game that solves optimal control problems via independent decision-makers in every node, acting in their own interest!

- Optimal strategies independent of global graph structure!
- States are measured only locally
- Linearly complexity (given horizon and iteration scheme)
- Distributed bounds on distance to optimality

Lecture 4

- More on dual decomposition
- Distributed MPC
- **Gradient methods for large-scale systems**

Controller Tuning for Large Tri-diagonal Plant

Minimize $V = \mathbf{E} \sum_{i=1}^{n} \left(|x_i|^2 + |u_i|^2\right)$

$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ \vdots \\ x_n(t+1) \end{bmatrix} = \begin{bmatrix} 0.6 & 0.1 & & 0 \\ 0.3 & \ddots & \ddots & \\ & \ddots & \ddots & 0.1 \\ 0 & & 0.3 & 0.6 \end{bmatrix} \begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ \vdots \\ x_n(t+1) \end{bmatrix} + \begin{bmatrix} u_1(t) + w_1(t) \\ u_2(t) + w_2(t) \\ \vdots \\ u_n(t) + w_n(t) \end{bmatrix}$$

We will optimize a tri-diagonal control structure

$$\bar{L} = \begin{bmatrix} * & * & & 0 \\ * & & \ddots & \\ & \ddots & & * \\ 0 & & * & * \end{bmatrix}$$

Computing the closed loop control performance

We are applying the control law $u = -Lx$ to the system

$$x(t+1) = Ax(t) + Bu(t) + w(t)$$

where w is white noise with variance W. Define

$$J(L) = \mathbf{E}\left(|x|_Q^2 + |u|_R^2\right)$$

Then the gradient with respect to a particular element L_{ij} is

$$(\nabla_L J)_{ij} = 2RL\mathbf{E}\,[x_i x_j^T] + 2B^T\mathbf{E}\,[p_i x_j^T]$$

where $p(t)$ is the stationary solution of the adjoint equation

$$p(t-1) = (A - BL)^T p(t) - (Q + L^T RL)x(t)$$

A distributed synthesis procedure

1. Measure the states $x_i(t)$ for $t = t_k, \dots, t_k + N$
2. Simulate the adjoint equation
$$p_i(t-1) = \sum_{j \in E_i} (A - BL)_{ji}^T p_j(t) - 2(Q_i x_i(t) - \sum_{j \in E_i} L_{ji}^T R_j u_j(t))$$
for $t = t_k, \dots, t_k + N$ by communicating states between nodes.
3. Calculate the estimates of $\mathbf{E}\, u_i x_j^T$ and $\mathbf{E}\, p_i x_j^T$ by
$$(\mathbf{E}\, u_i x_j^T)_{\text{est}} = \frac{1}{N+1} \sum_{t=t_k}^{t_k+N} u_i(t) x_j(t)^T \quad (\mathbf{E}\, p_i x_j^T)_{\text{est}} = \frac{1}{N+1} \sum_{t=t_k}^{t_k+N} p_i(t) x_j(t)^T$$
4. For fixed step length $\gamma > 0$, update
$L_{ij}^{(k+1)} = L_{ij}^{(k)} + 2\gamma R_i \left(\mathbf{E}\, u_i x_j^T\right)_{\text{est}} + B_i^T \left(\mathbf{E}\, p_i x_j^T\right)_{\text{est}}.$
Let $t_{k+1} = t_k + N$ and start over.

Gradient iteration for the wind park

```
cost =

   14.9887


L =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

Gradient iteration for the wind park

```
cost =

   10.5429


L =

    0.0327    0.0400         0         0         0
   -0.0007    0.0560    0.0527         0         0
         0   -0.0069    0.0434    0.0315         0
         0         0   -0.0207    0.0131    0.0437
         0         0         0   -0.0033    0.0373
```

Gradient iteration for the wind park

```
cost =

    7.8184


L =

    0.0310    0.0595         0         0         0
   -0.0168    0.1002    0.1151         0         0
         0    0.0345    0.1357    0.0986         0
         0         0    0.0636    0.0831    0.1351
         0         0         0    0.0102    0.1295
```

Gradient iteration for the wind park

```
cost =

    7.6192


L =

    0.0404    0.0685         0         0         0
   -0.0086    0.1076    0.1193         0         0
         0    0.0382    0.1421    0.1094         0
         0         0    0.0593    0.0991    0.1449
         0         0         0    0.0131    0.1348
```

Gradient iteration for the wind park

```
cost =

    7.4004


L =

    0.0576    0.0583         0         0         0
    0.0115    0.1224    0.1381         0         0
         0    0.0373    0.1500    0.1153         0
         0         0    0.0546    0.1068    0.1566
         0         0         0    0.0168    0.1594
```

Gradient iteration for the wind park

```
cost =

    7.2493


L =

    0.0712    0.0654         0         0         0
    0.0061    0.1224    0.1443         0         0
         0    0.0341    0.1550    0.1166         0
         0         0    0.0773    0.1409    0.1580
         0         0         0    0.0418    0.1601
```

Gradient iteration for the wind park

```
cost =

    6.9736


L =

    0.0936    0.1056         0         0         0
    0.0331    0.1775    0.1341         0         0
         0    0.0563    0.1500    0.1215         0
         0         0    0.0700    0.1564    0.1567
         0         0         0    0.0567    0.1646
```

Gradient iteration for the wind park

```
cost =

    6.8211


L =

    0.1390    0.1070         0         0         0
    0.0357    0.1821    0.1549         0         0
         0    0.0668    0.1797    0.1098         0
         0         0    0.0633    0.1685    0.1413
         0         0         0    0.0589    0.1754
```

Gradient iteration for the wind park

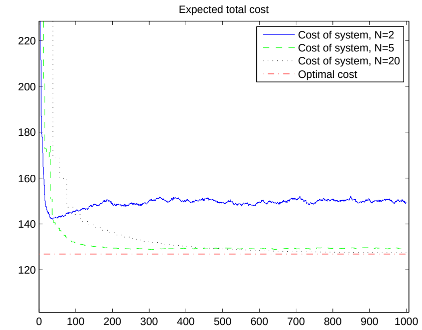
```
cost =

    6.7464


L =

    0.1438    0.1208         0         0         0
    0.0470    0.2031    0.1632         0         0
         0    0.0749    0.1909    0.1046         0
         0         0    0.0779    0.1843    0.1388
         0         0         0    0.0445    0.1732
```

Performance Versus Number of Gradient Iterations



A distributed controller with 100 agents, using only local data. Fewer gradient iterations gives faster convergence, but worse stationary performance.

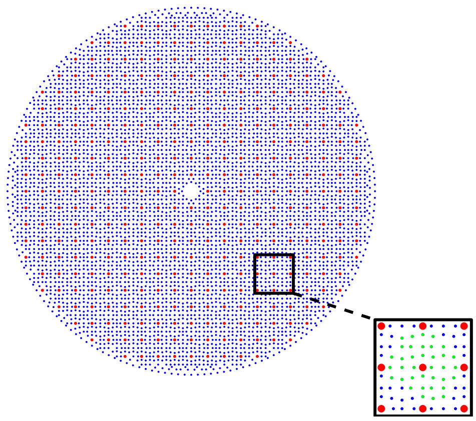
Control of a Large Deformable Mirror

Case study of a 1 m diameter deformable mirror, for adaptive optics in large telescopes. Used to correct for aberrations introduced by the atmosphere.

Using finite element method a spatially discretized model.

$$\mathcal{M}\ddot{\boldsymbol{\xi}} + \mathcal{C}\dot{\boldsymbol{\xi}} + \mathcal{K}\boldsymbol{\xi} = \boldsymbol{F}$$

- 6128 discretization points, each with 6 degrees of freedom.
- 372 force actuators.
- 1136 position sensors.



Method data and performance

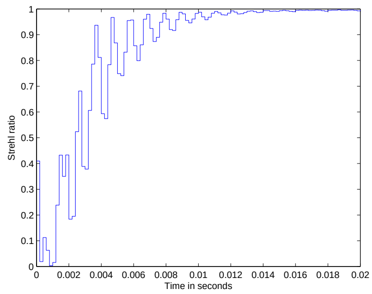
Method data

- The sparsity of feedback matrix L is $0.63\,\%$.
- Time horizon in gradient computation is 1000 time samples.
- 1000 update iterations are performed.

The computation time for the method becomes 16.6 hours. $70\,\%$ of this time is spent on calculating matrix inversions in the system simulation.

Control Performance

The controller is used on the mirror when using a simulated atmosphere. Strehl ratio is a common measure in adaptive optics. Defined by $S = e^{-(2\pi\epsilon(t)/\lambda)^2}$ where $\epsilon(t)$ is the RMS error at time t.



Lecture 4

- More on dual decompostion
- Distributed MPC
- Gradient methods for large-scale systems