Distributed Control of Water Delivery Canal Networks

J. M. Lemos with ***J. M. Igreja***

INESC-ID/IST/UTL, ***INESC-ID/ISEL/IPL***

Controlo 2012, Funchal, Madeira

Performed in the framework of AQUANET project, financed by FCT

http://ramses.inesc.pt/AQUANET

Parts of this presentation uses work made in cooperation with

Luís Pinto

Filipe Cadete

Inês Sampaio

Fernando Machado

Nuno Nogueira

Luís Rato

The experimental results shown have been obtained in the canal of

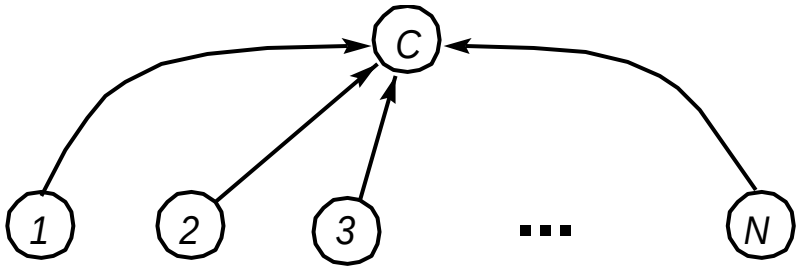
Núcleo de Hidráulica e Controlo de Canais

Universidade de Évora

An example in distributed estimation

Find the average of the money in the pockets of the members of the assistance in the first row.

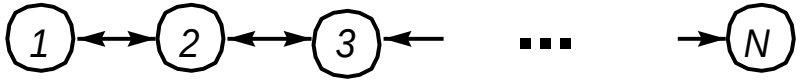
Trivial using a centralized communication solution



Each node (person) i communicates the amount of money y_i inside his pocket to the central node that simply computes the average as

$$S = \frac{1}{N}\sum_{i=1}^{N} y_i$$

Assume now that there is no central node and that each person can only talk to their immediate neighbors. The communication structure is then



One possibility consists in an iterative procedure in which each node (person)

1. Tells his neighbors about how much money he has
2. Computes the average of the money in his pocket and in his neighbors pocket.
3. Repeats 1) and 2) iteratively, telling the average he has computed.

Does this converges to the average of the money for everybody in the row?

One can represent this iteration as a discrete-time state equation

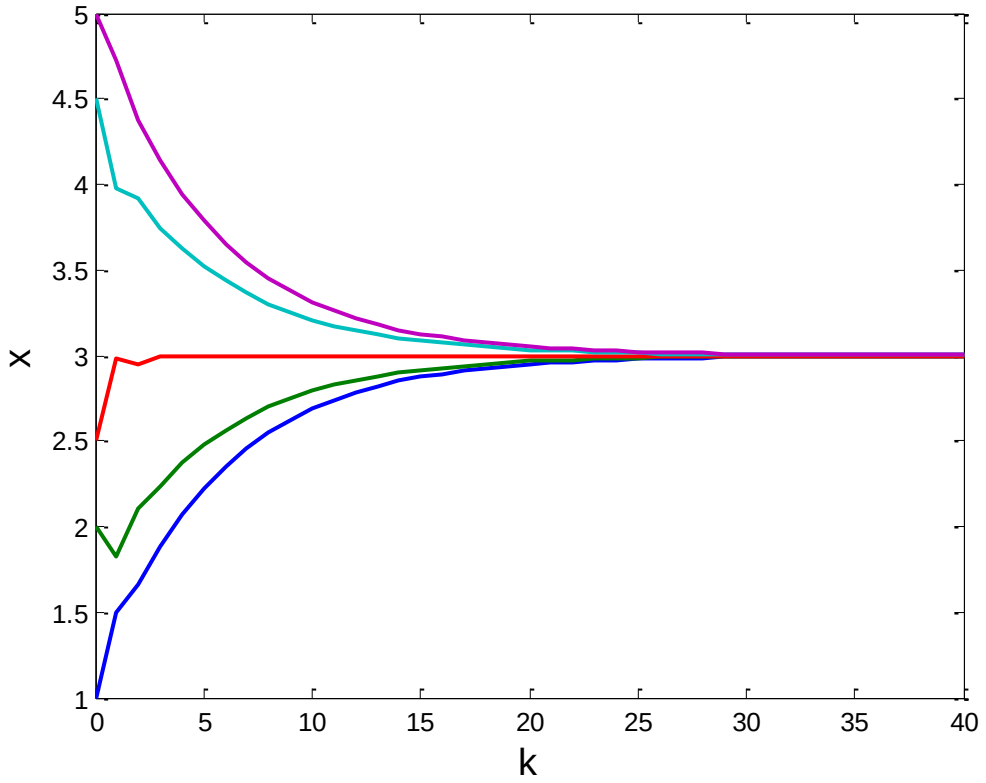
$$x(k+1) = P\,x(k)$$

Positive system

$\Longrightarrow$ Frobenius-Perron theorem implies that there is a dominant eigenvalue (actually equal to 1 with an eigenvector with all components equal)

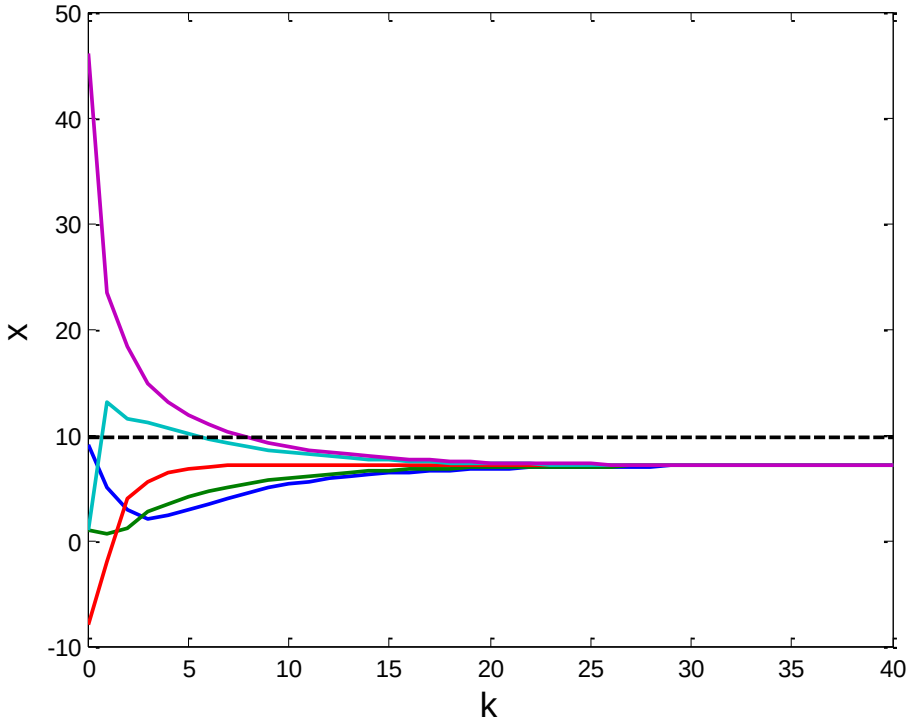
$\Longrightarrow$ The states will converge to a situation in which all states are equal

$x(0) = \begin{bmatrix} 1 & 2 & 2.5 & 4.5 & 5 \end{bmatrix}$ average = 3. It seems to work!?



Unfortunately this method does not always work.

The problem is that the sum of the elements of $x(k)$ is not constant along time.



What to retain from this example

- Distributed processing: multiple “agents” cooperate locally to reach a common result.
- How to do the coordination?
- Does it converge?
- Performance evaluation: How close is the result from the centralized one?

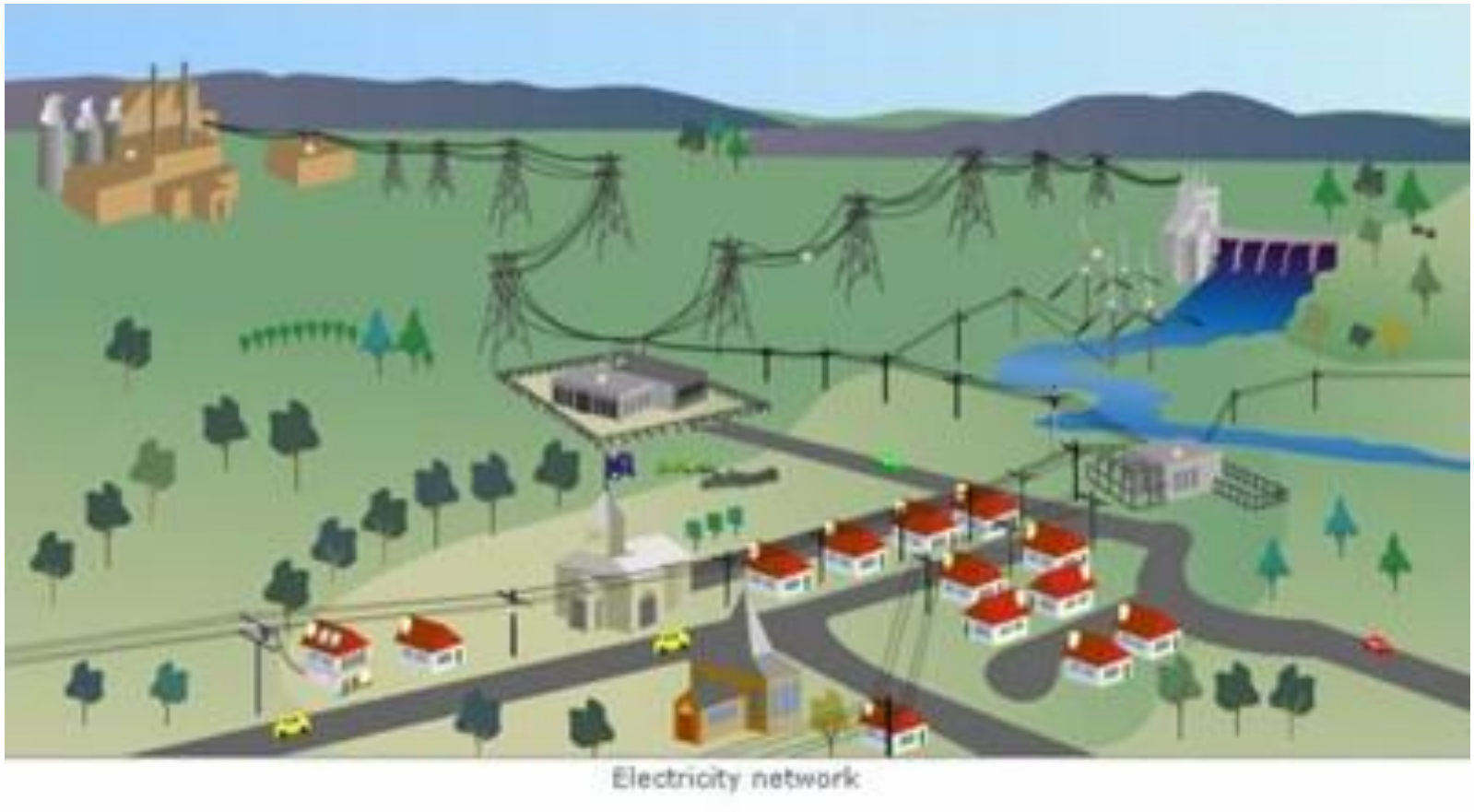
Other examples: Bird formation flying

A classic example in distributed systems: Each bird decides its trajectory based on the position of its neighbors.

Other examples: Large fields of wind-mills

Seemingly isolated systems are actually coupled.

Other examples: Hydro-power valley and electric power networks

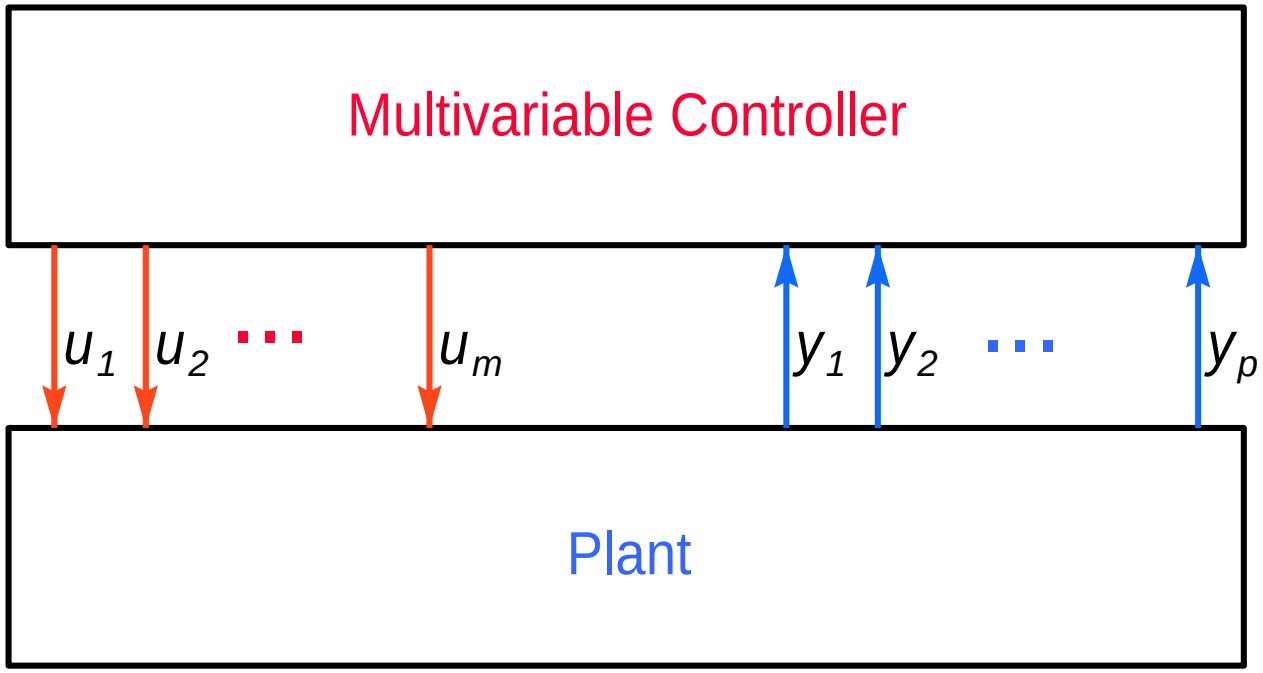


Controller structures

- Centralized (multivariable).
- Decentralized
- Distributed control agents

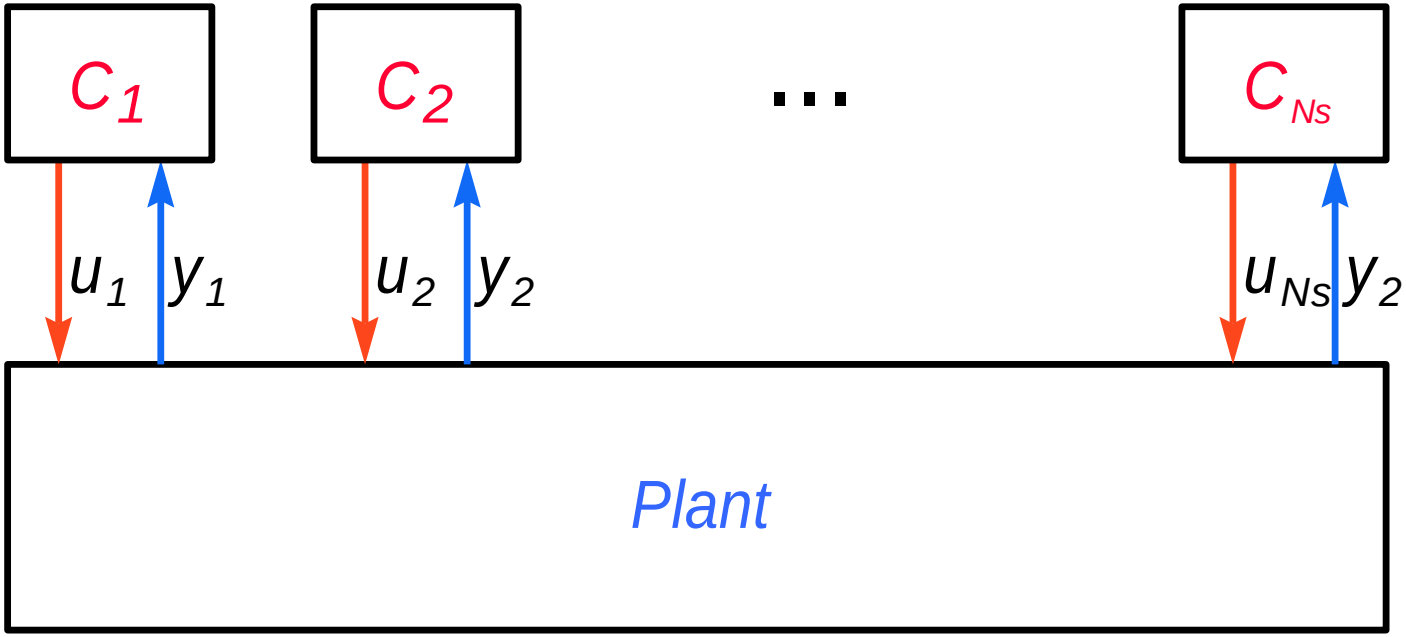
Controller structures: Centralized

(Multivariable control)



A single central controller receives all plant output signals and computes in a centralized way all the manipulated variables. Requires heavy communication.

Controller structures: Decentralized



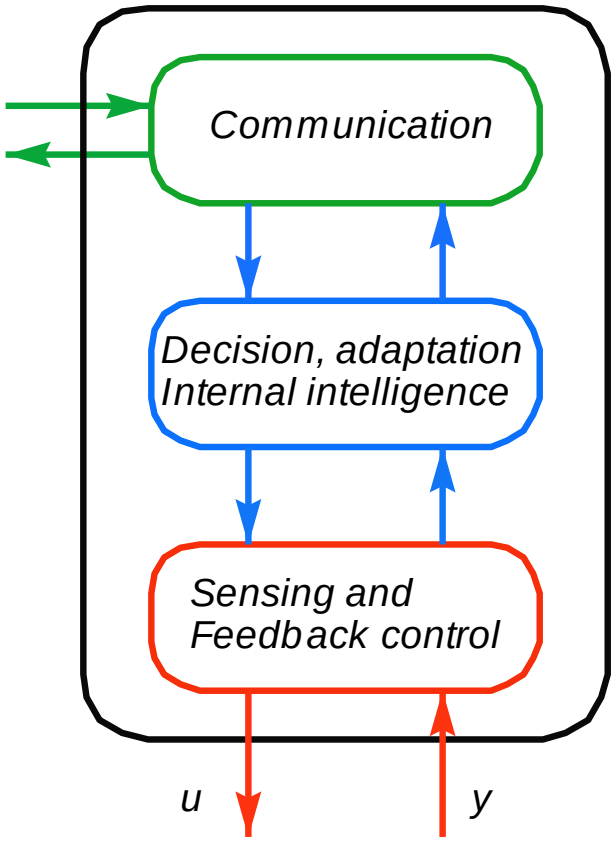
A set of independent controllers, each one closing one feedback loop without any concern to the others. Stability problems.

Decentralized control: A warning example

Although each single loop has a good stability margin, the system controlled with decentralized control may be made unstable by small uncertainty terms.

See an example in

Doyle and Stein (1081). Multivariable Feedback Design: Concepts for a classical modern synthesis. IEEE TAC AC-26(1):4-16.



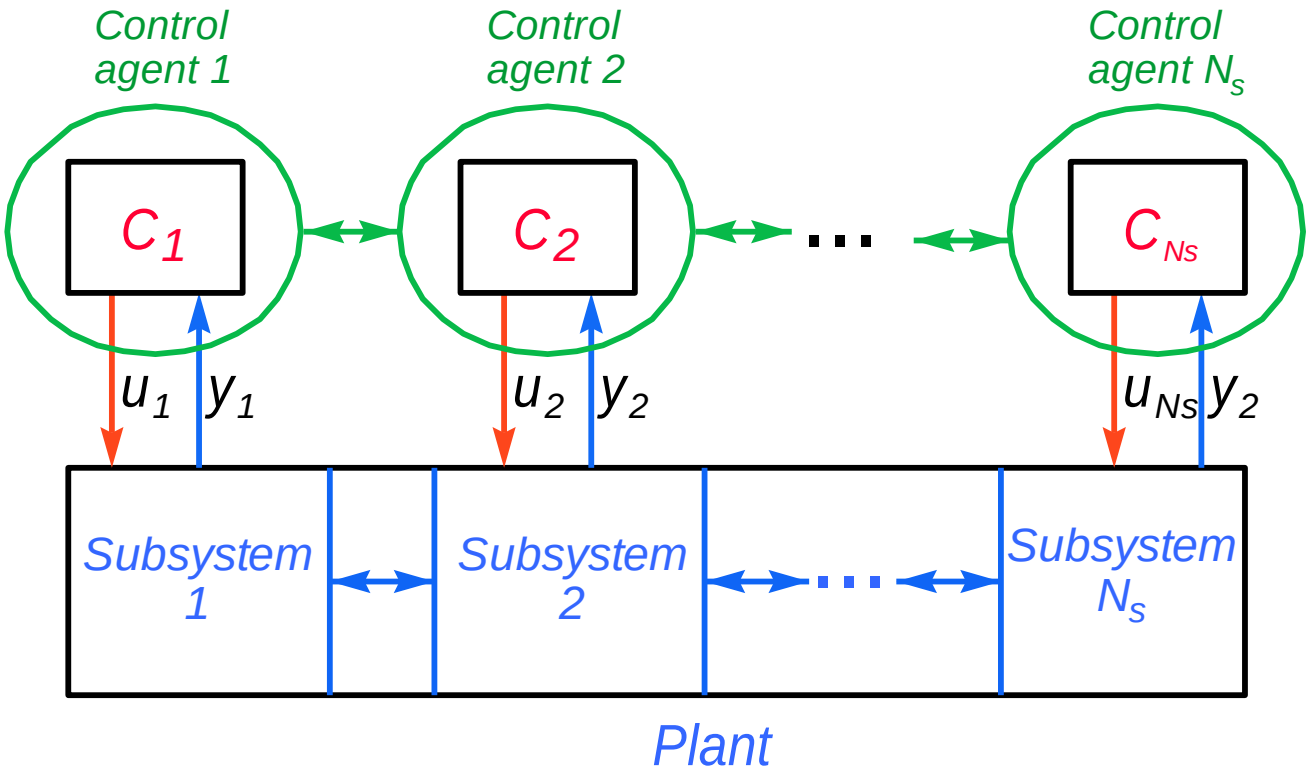
Control agents

A software entity capable of

- Communicating with other control agents
- Sensing and feedback control
- Changing the feedback control decisions depending on internal intelligence mechanism.

A step towards autonomous systems.

Controller structures: Distributed control agents



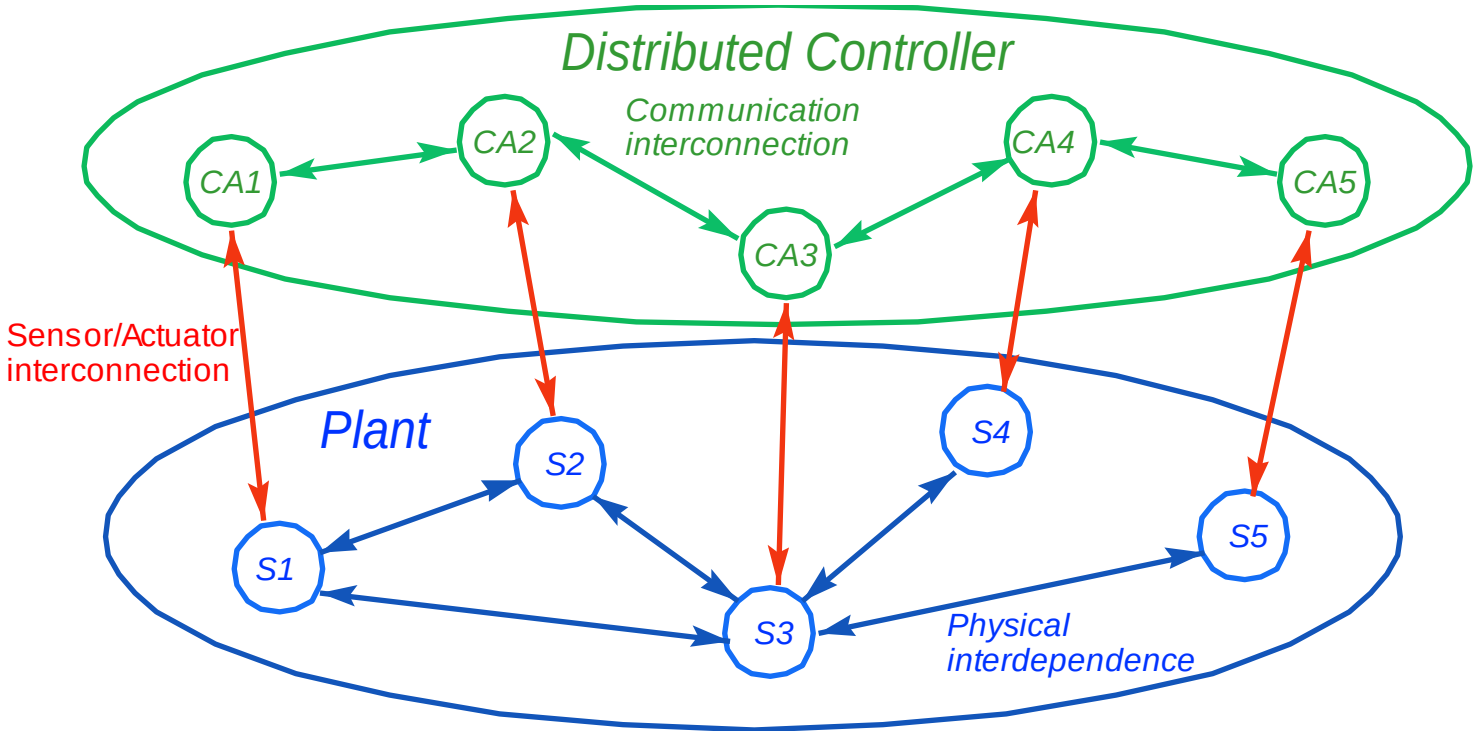
☞Decompose the plant in subsystems.

☞Encapsulate the local controllers in control agents.

☞Create communication links.

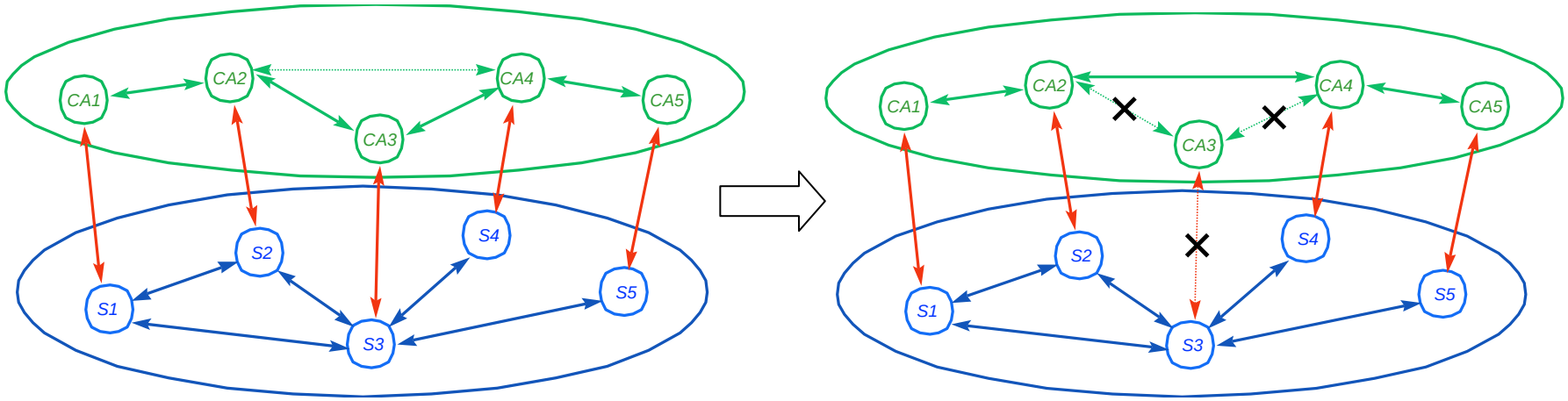
☞Define the algorithm that allows the control agents to act in a coordinated way.

Distributed control as an interconnection of control agents



Considering the plant/controller interconnection as a graph as a number of advantages: Results from graph theory; reconfiguration.

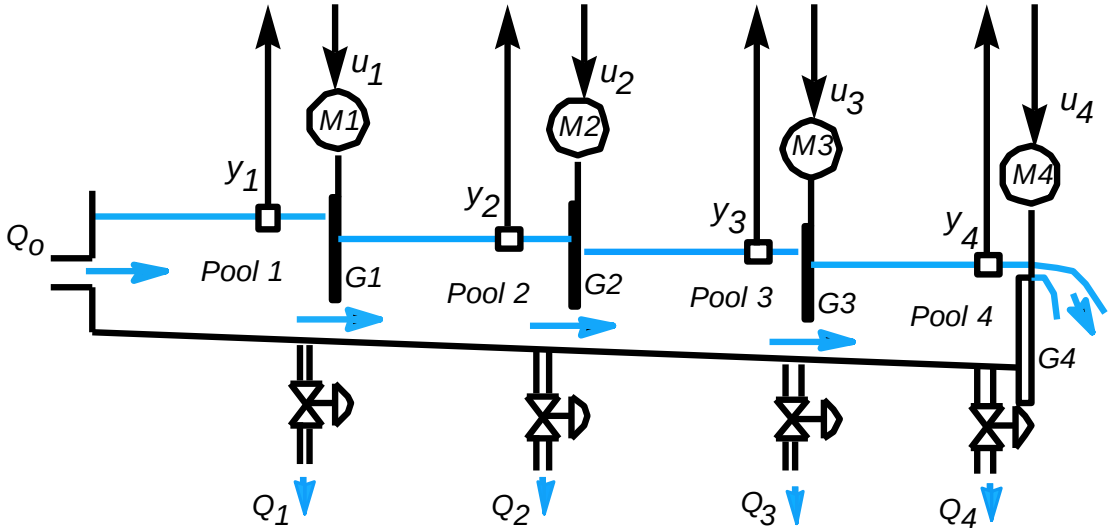
Controller reconfiguration



Controller reconfiguration in response to a partial fault.

See the presentation of Inês Sampaio in the session on Coordinated Control of Water Delivery Canals for further details of fault tolerant distributed control.

Example: Water delivery canal



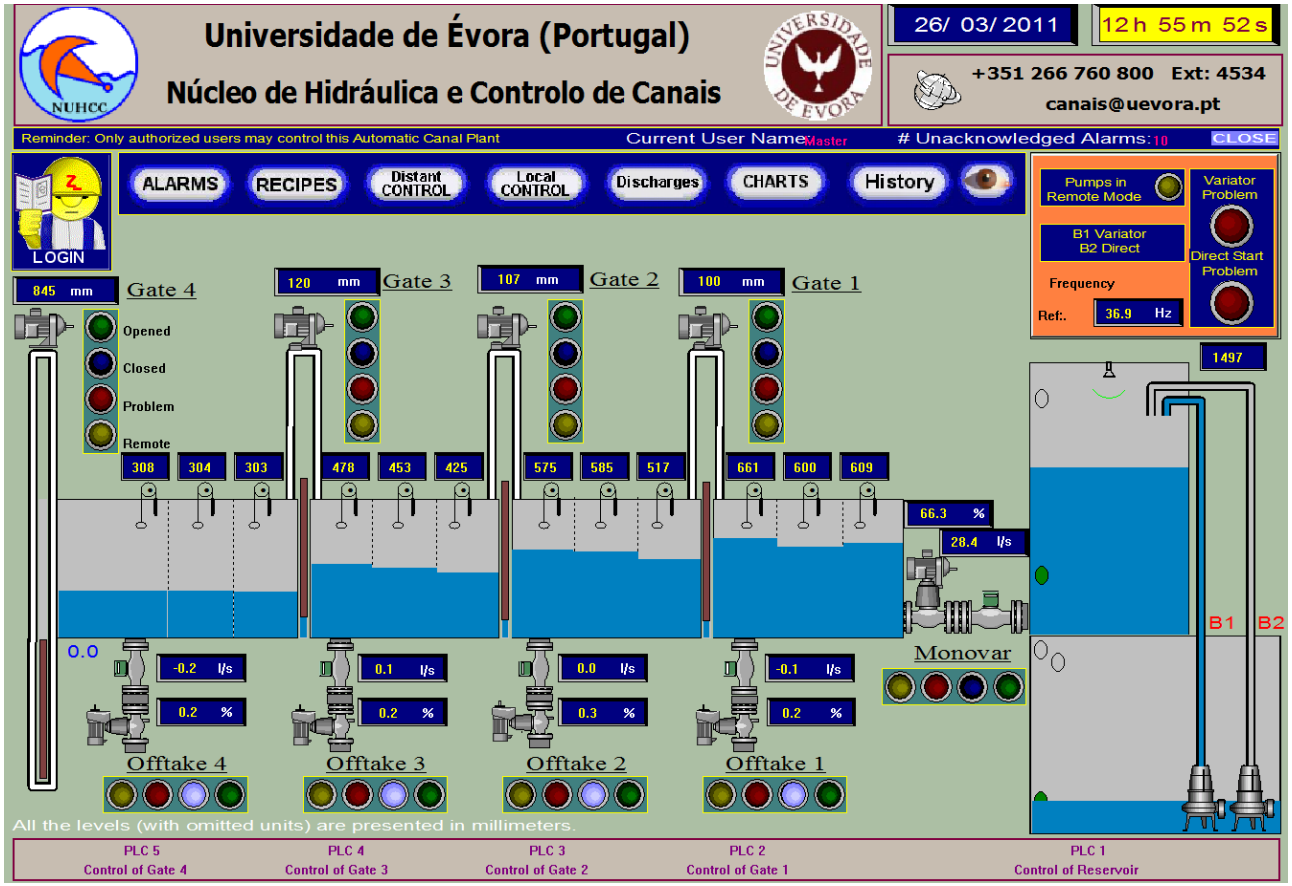
4 pools, each of about 45m long

90 L/s nominal flow

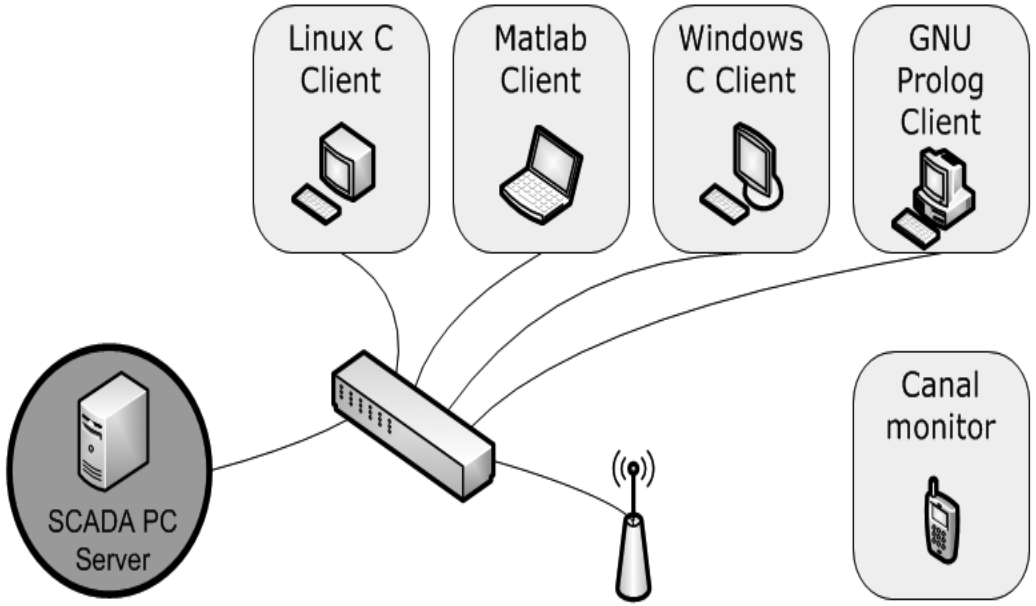
Water fed by gravity

Side takes simulate water usage

Manual operation: The SCADA system

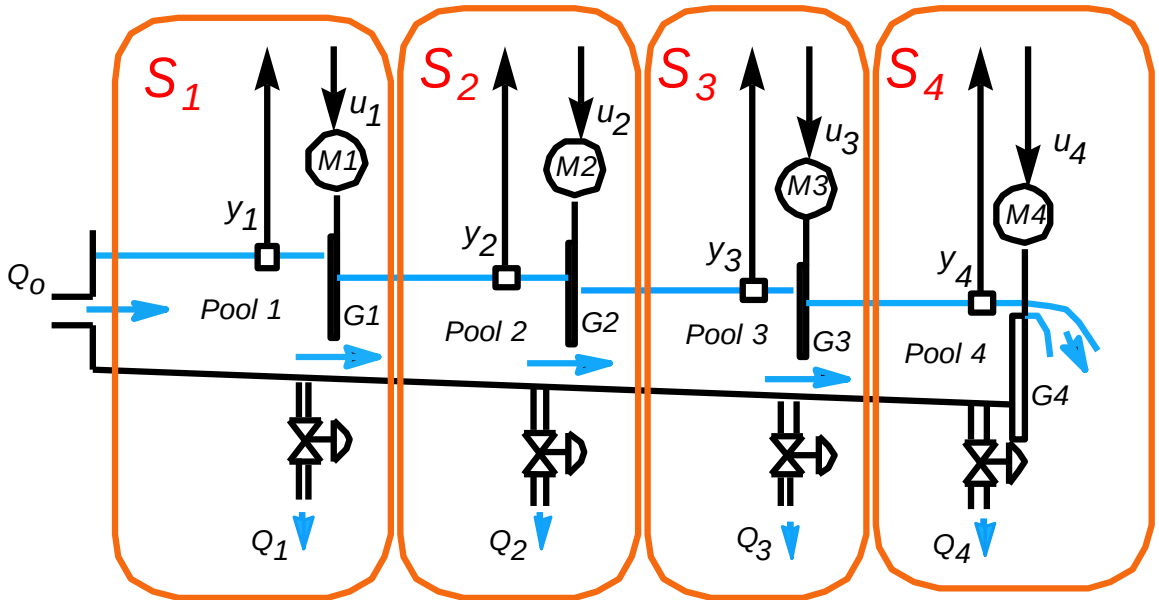


Mundane (but essential…) issues: Controller interface with SCADA



Easy connection to the router via cable or wireless of multiple PCs “MATLAB clients” that read data from the canal and send orders to gates and valves.

Decomposing the canal in subsystems



The canal is decomposed in a chain of subsystems.

Each subsystem comprises a pool, its downstream gate and an oftake.

Subsystems are physically coupled.

How to obtain canal models?

The Saint-Venant equations - PDE embedding mass and momentum conserv.

$$\frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = q_l$$

$$\frac{\partial Q}{\partial t} + \frac{\partial Q^2/A}{\partial x} + gA\frac{\partial h}{\partial x} + gA(S_f - S_0) = kq_lV$$

- t time [s]
- x space [m]
- $A(x,t)$ flow cross sectional area [m^2]
- $Q(x,t)$ discharge [m^3s^{-1}]
- $q_l(x,t)$ lateral discharge per unit length [m^2s^{-1}]
- k constant
  inflow: $q_l > 0 \rightarrow k = 0$;
  outflow: $q_l < 0 \rightarrow k = 1$
- g gravitational acceleration [ms^{-2}]
- $h(x,t)$ water level [m]
- $S_f(x,t)$ friction slope
- $V(x,t)$ water velocity [ms^{-1}]

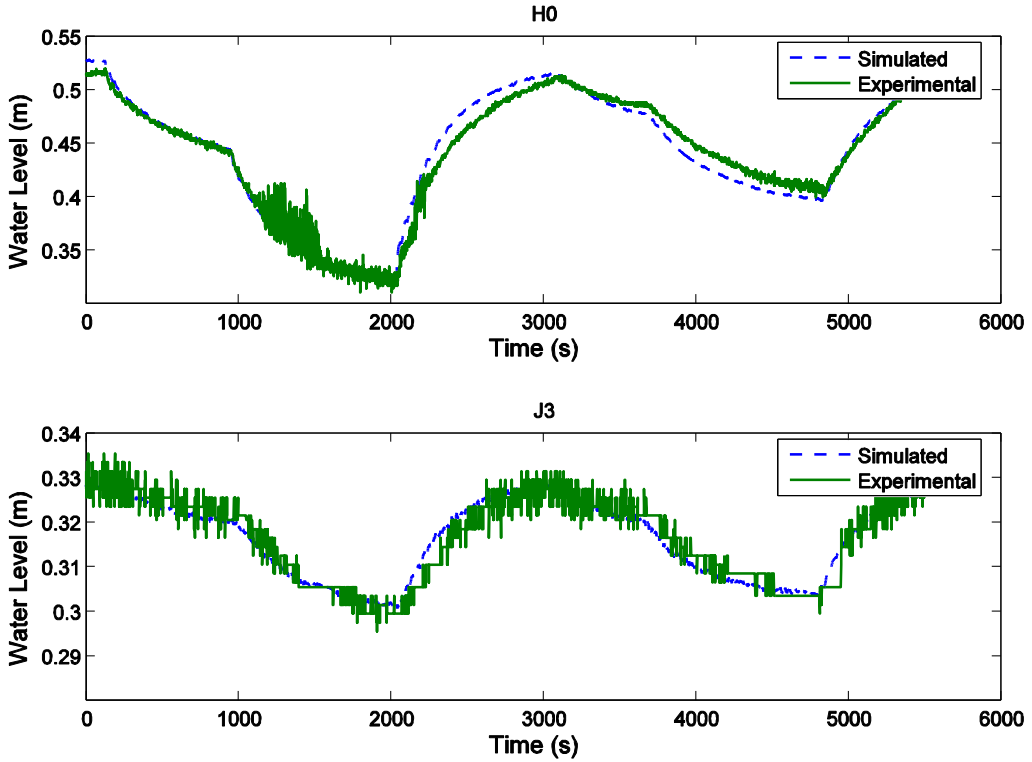
The friction is modeled using the Manning-Strickler formula:

$$S_f = \frac{Q|Q|n^2}{A^2R^{4/3}} \quad (3)$$

where:

- $R(x,t)$ hydraulic radius [m]
- n Manning coefficient [$m^{-1/3}s$]

Nonlinear model based on Saint-Venant equations



Numeric integration with Preissman method.

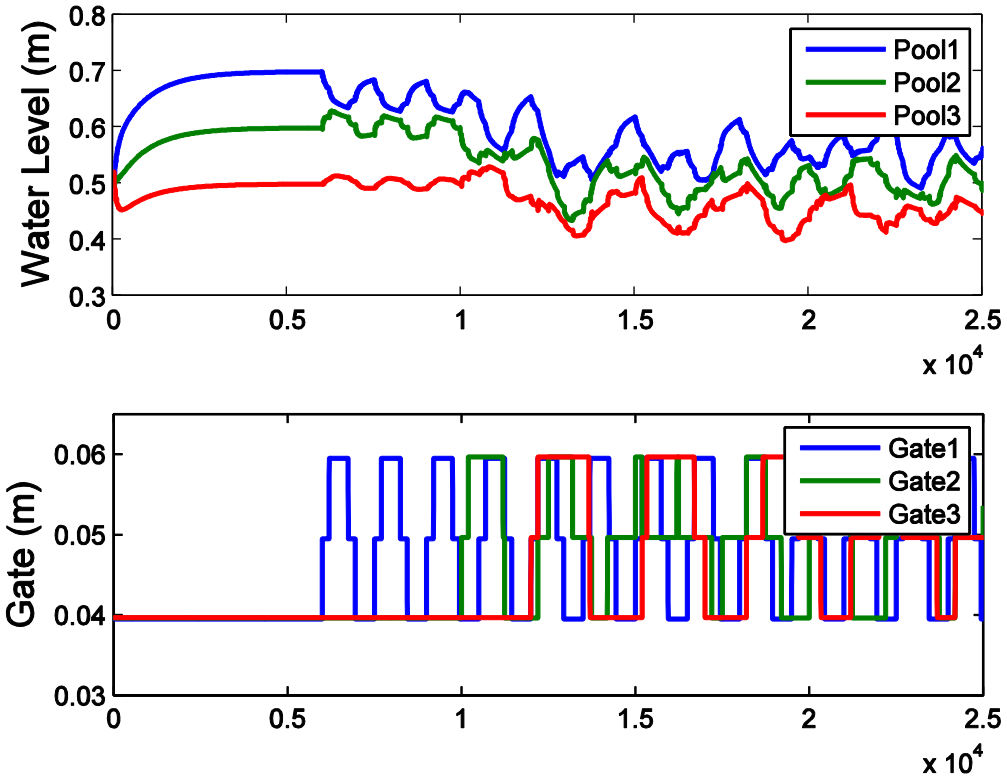
Linear models: From the Saint-Venat equations

Obtain a pencil of linear state-space models for different operating regimes

defined by flow and level by linearizing the Saint-Venant equations.

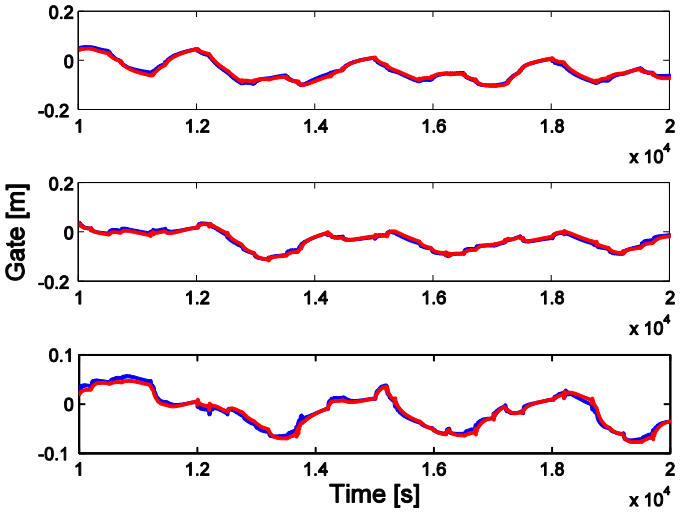
Parameters of the SV equations obtained from plant data.

Linear Models: Identification of plant data



Identify MISO ARX models imposing the desired structure;

Convert to state-space model.



Canal decomposition

Describe the canal by the linear state-space model with accessible disturb.:

$$x(k+1) = Ax(k) + Bu(k) + \Gamma d(k) \qquad y(k) = Cx(k)$$

Decompose the model by restraining its structure to be

$$A = \begin{bmatrix} A_{11} & A_{12} & 0 & 0 \\ A_{21} & A_{22} & A_{23} & 0 \\ 0 & A_{32} & A_{33} & A_{34} \\ 0 & 0 & A_{43} & A_{44} \end{bmatrix} \quad B = \begin{bmatrix} B_{11} & B_{12} & 0 & 0 \\ B_{21} & B_{22} & B_{23} & 0 \\ 0 & B_{32} & B_{33} & B_{34} \\ 0 & 0 & B_{43} & B_{44} \end{bmatrix}$$

Local sub-system dynamics

$$x_i(k+1) = A_i x_i(k) + B_i u_i(k) + \Gamma_i \delta_i(k)$$

$$\delta_i(k) = \begin{bmatrix} d_i(k) & u_{i-1}(k) & u_{i+1}(k) & x_{i-1}(k) & x_{i+1}(k) \end{bmatrix}^T$$

$$y_i(k) = C_i x_i(k)$$

An important special case

Sub-systems interact only through their manipulated inputs

$$x(k+1) = Ax(k) + Bu(k) + \Gamma d(k) \qquad y(k) = Cx(k)$$

Matrix A is imposed to be block diagonal:

$$A = \begin{bmatrix} A_{11} & 0 & 0 & 0 \\ 0 & A_{22} & 0 & 0 \\ 0 & 0 & A_{33} & 0 \\ 0 & 0 & 0 & A_{44} \end{bmatrix} \quad B = \begin{bmatrix} B_{11} & B_{12} & 0 & 0 \\ B_{21} & B_{22} & B_{23} & 0 \\ 0 & B_{32} & B_{33} & B_{34} \\ 0 & 0 & B_{43} & B_{44} \end{bmatrix}$$

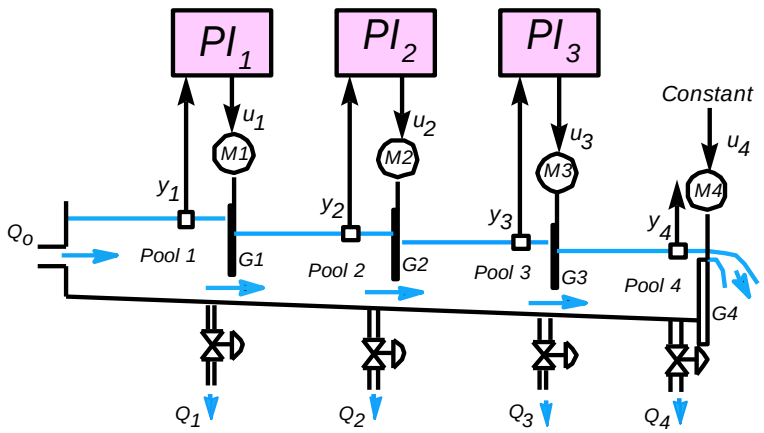
Local sub-system dynamics with interaction only through the inputs

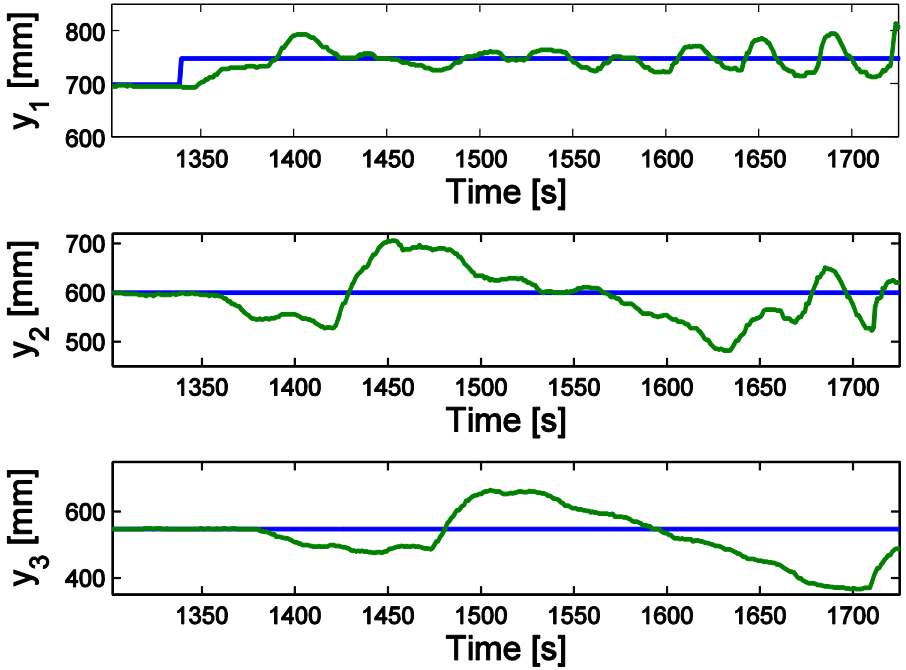
$$x_i(k+1) = A_i x_i(k) + B_i u_i(k) + \Gamma_i \delta_i(k)$$

$$\delta_i(k) = \begin{bmatrix} d_i(k) & u_{i-1}(k) & u_{i+1}(k) \end{bmatrix}^T$$

$$y_i(k) = C_i x_i(k)$$

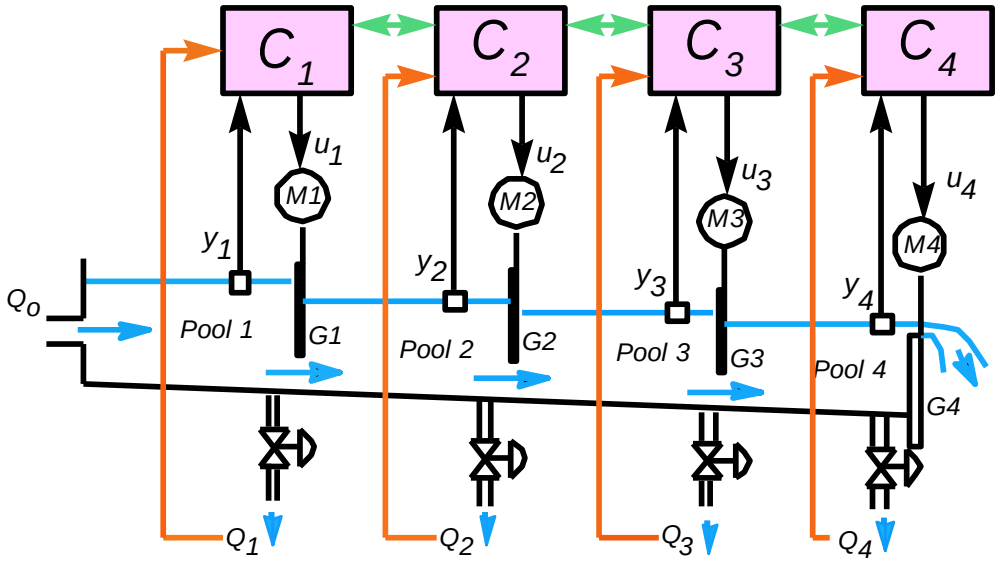
Does it work with decentralized PI controllers?





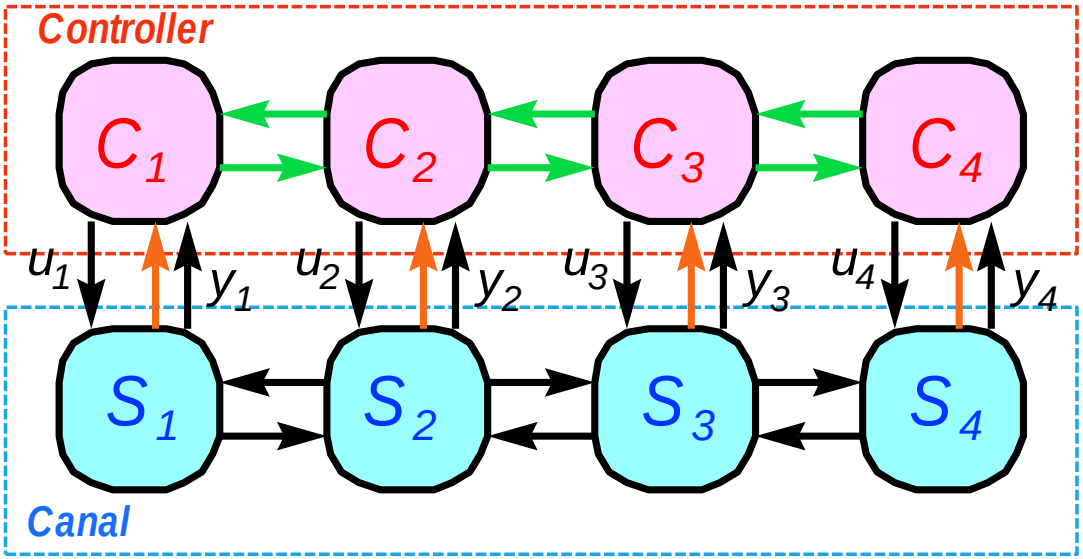
May easily become unstable!

Distributed controller structure



A chained network of upstream local level controllers, each one communicating only with their neighbors, exchanging information about their decisions on the manipulated and output variables.

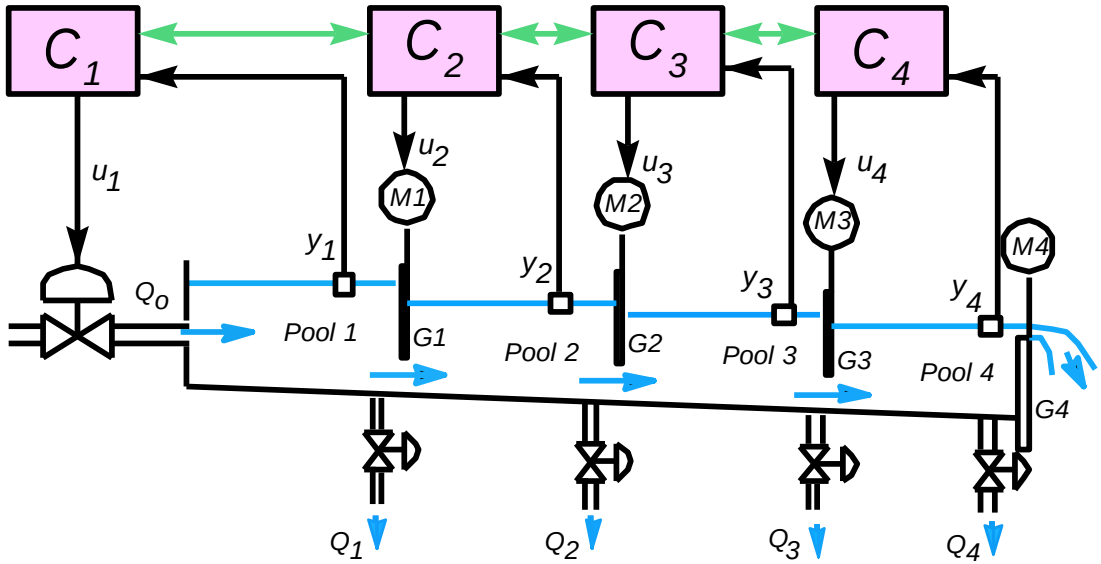
An equivalent graph view of the distributed controller



How to design:

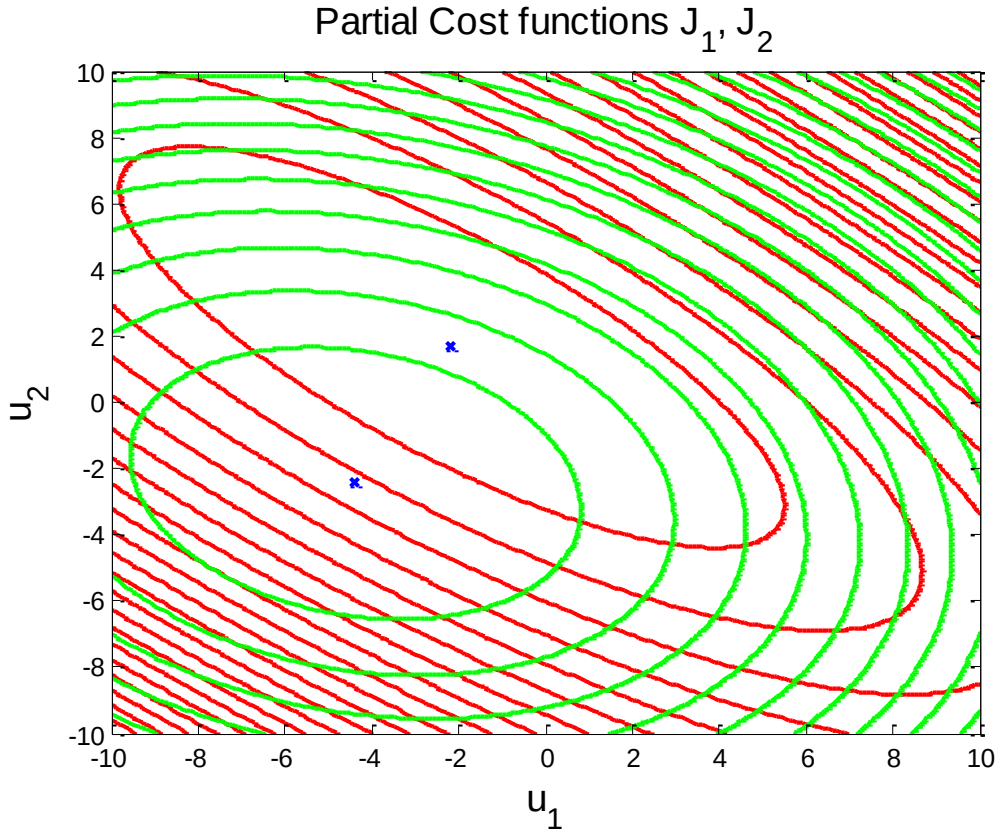
- The controllers?
- The coordination algorithm?

An alternative: Distant upstream control



- Saves more water (water comes “on demand”)
- Worst rejection of disturbances (extra delay in action, fast disturbances)
- Feedforward can also be included.

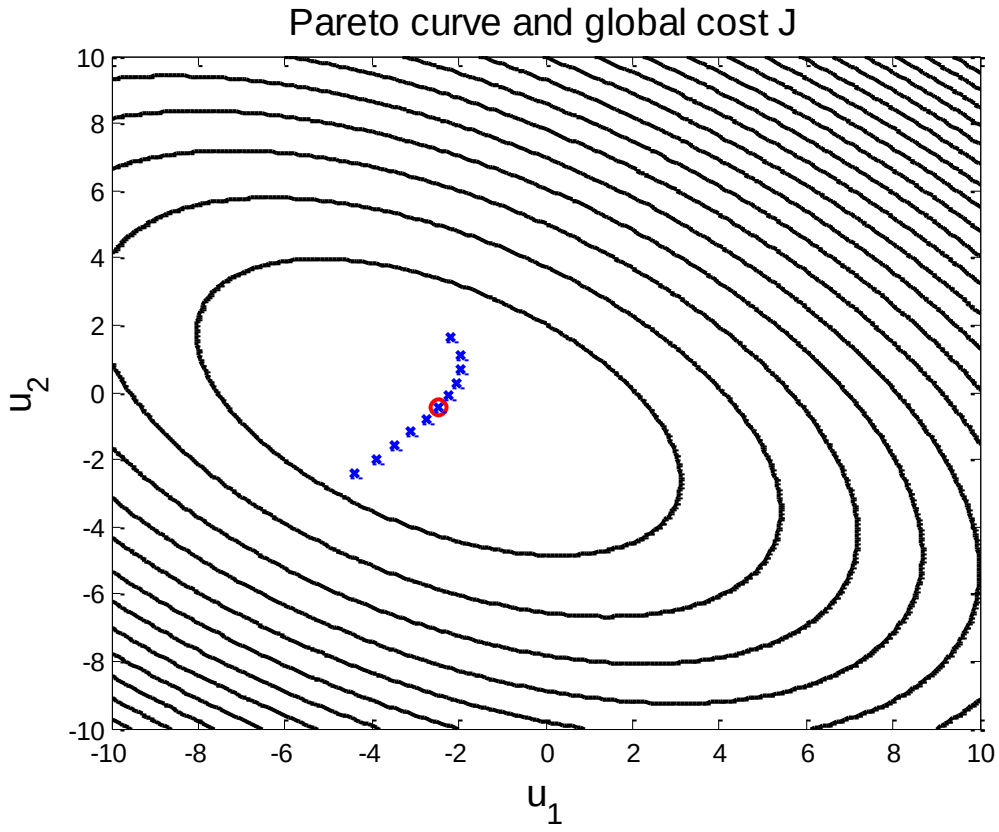
Multiobjective optimization



Find the minimum of a global cost function made by the addition of local cost functions:

$$J(u_1,u_2) = J_1(u_1,u_2) + J_2(u_1,u_2)$$

Multiobjective optimization: Pareto optimal locus

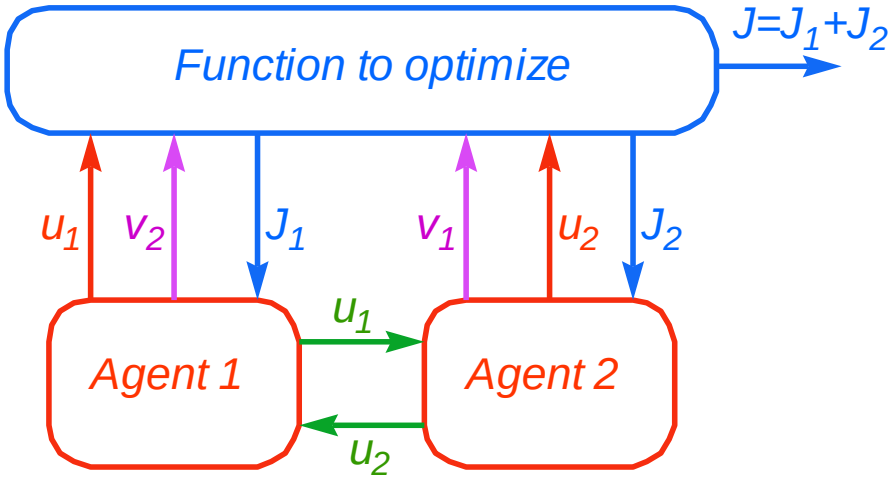


The locus of Pareto optimal decisions on the independent variables is given by the minimum of

$$\alpha J_1(u_1,u_2) + (1-\alpha) J_2(u_1,u_2)$$

for $0 \le \alpha \le 1$.

Distributed optimization



$$\min_{u_1,u_2}\left[J_1(u_1,u_2)+J_2(u_1,u_2)\right]$$

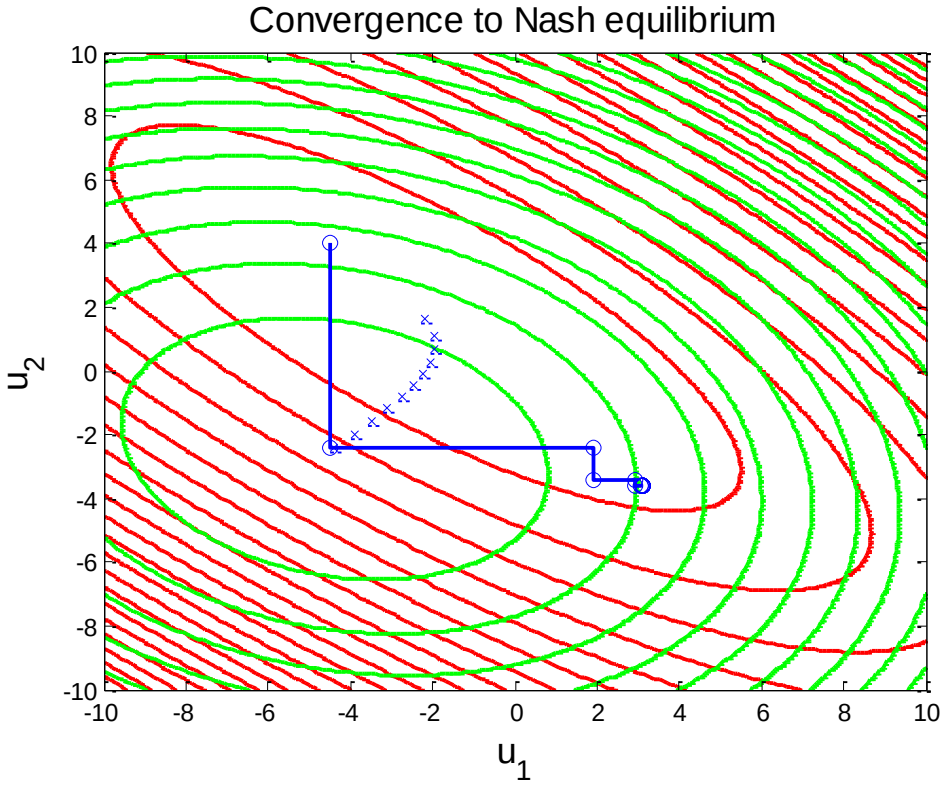
Replicate the independent variables for each agent.

Agent 1 minimizes $J_1(u_1,v_2)$ where v_2 is a replica of u_2.

Agent 2 minimizes $J_2(v_1,u_2)$ where v_1 is a replica of u_1.

A mechanism to reconcile v_1 with u_1 and v_2 with u_2 is needed.

Distributed optimization: Convergence to the Nash equilibrium of alternating variable minimization



Iterate the steps:

1) Optimize $J_1(u_1, u_2)$ with respect to u_1, keeping u_2 constant;

2) Optimize $J_2(u_1, u_2)$ with respect to u_2, keeping u_1 constant.

A detour in Convex Optimization: The dual problem, a way of incorporating constraints

Primal problem: $\min_u J(u)$ subject to $Au - b = 0$

Lagrangian: $L(u, \lambda) = J(u) + \lambda^T (Au - b)$

Dual function: $g(\lambda) = \inf_u L(u, \lambda)$

Dual problem: $\lambda^* = \arg\max g(\lambda)$

$$u^* = \arg\min_u L(u, \lambda^*)$$

Dual ascent method

Iterate the following steps:

$$u(k+1) = \arg\min_{u} L(u, \lambda(k))$$

$$\lambda(k+1) = \lambda(k) + \alpha(Au(k+1) - b)$$

Requires strong assumptions to work.

Back to distributed optimization

$$\min_{u_1,u_2}\left[J_1(u_1,u_2)+J_2(u_1,u_2)\right]$$

Decomposition

Agent 1

$$\min_{u_1,v_2} J_1(u_1,v_2)$$

Subject to $v_2 = u_2$

Agent 2

$$\min_{v_1,u_2} J_2(v_1,u_2)$$

Subject to $v_1 = u_1$

$$\max_{\lambda_i}\min_{v_i,u_i}\left[J_1(u_1,v_2)+J_2(v_1,u_2)+\lambda_1(u_2-v_2)+\lambda_2(u_1-v_1)\right]$$

$$\max_{\lambda_i} \min_{v_i, u_i} \left[J_1(u_1, v_2) + J_2(v_1, u_2) + \lambda_1 (u_2 - v_2) + \lambda_2 (u_1 - v_1) \right]$$

Agent 1

$$\min_{u_1, v_2} \left[J_1(u_1, v_2) - \lambda_1 v_2 + \lambda_2 u_1 \right]$$

$$\max_{\lambda_1} \left[\lambda_1 (u_2 - v_2) \right]$$

Agent 2

$$\min_{v_1, z_2} \left[J_2(v_1, u_2) - \lambda_{21} v_1 + \lambda_1 u_2 \right]$$

$$\max_{\lambda_2} \left[\lambda_2 (u_1 - v_1) \right]$$

Adjustment of multipliers ensure coordination.

Gradient base dual ascent algorithm

Agent 1

Receives $\lambda_2(k)$ and $u_2(k)$ from agent 2 and updates as

$$u_1(k+1) = u_1(k) - \gamma\left(\frac{\partial J_1}{\partial u_1} + \lambda_2(k)\right)$$

$$v_2(k+1) = v_2(k) - \gamma\left(\frac{\partial J_1}{\partial v_2} - \lambda_1(k)\right)$$

$$\lambda_1(k+1) = \lambda_1(k) + \gamma\left(u_2(k) - v_2(k)\right)$$

Agent 2

Receives $\lambda_1(k)$ and $u_1(k)$ from agent 1 and updates as

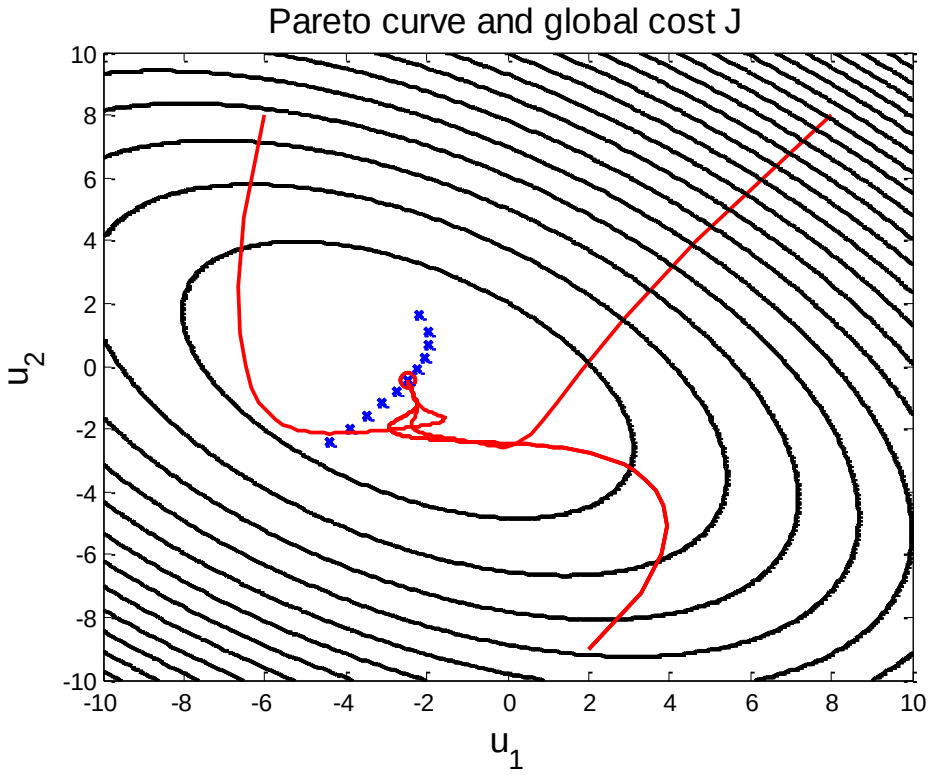
$$v_1(k+1) = v_1(k) - \gamma\left(\frac{\partial J_2}{\partial v_1} - \lambda_2(k)\right)$$

$$u_2(k+1) = u_2(k) - \gamma\left(\frac{\partial J_2}{\partial u_2} + \lambda_1(k)\right)$$

$$\lambda_2(k+1) = \lambda_2(k) + \gamma\left(u_1(k) - v_1(k)\right)$$

Going back to the toy example:

Distributed optimization with dual ascent



ADMM – Alternating directions method of multipliers

Minimize $f(x)+g(z)$ subject to $x=z$

Augmented Lagrangian $L(x,z,\lambda)=f(x)+g(x)+\lambda^T(x-z)+\frac{\rho}{2}\|x-z\|^2$

Recursively execute the steps:

1) $x(k+1)=\arg\min_x L(x,\lambda(k),z(k))$

2) $\lambda(k+1)=\lambda(k)+\rho(x(k+1)-z(k))$

3) $z(k+1)=\arg\min_z L(x(k+1),\lambda(k+1),z)$

LQG control agents

Each local control agent minimizes a quadratic cost

$$J_i(u) = \frac{1}{2}\sum_{k=1}^{\infty}(x_i^T(k)Q_i x_i(k) + \rho_i u_i^2(k))$$

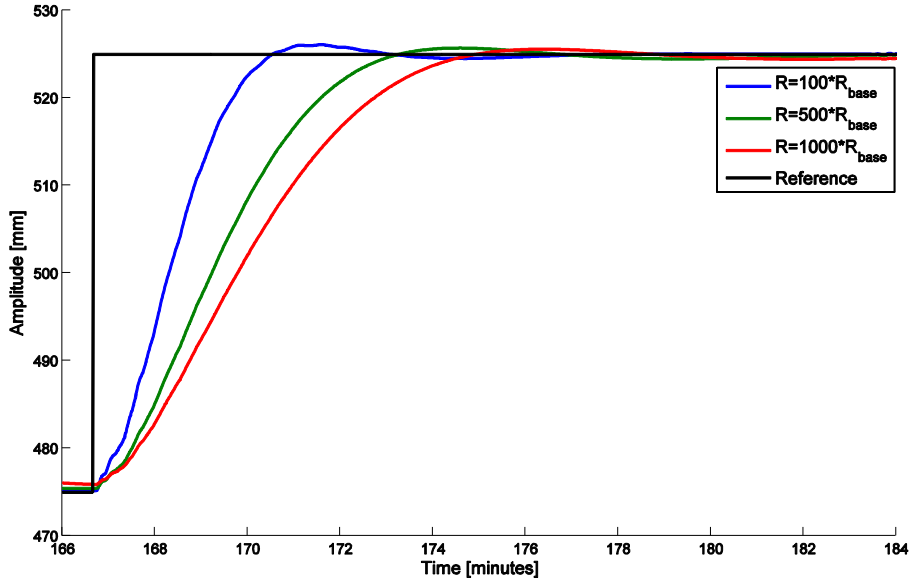
Resulting LQG controller with feedforward from accessible disturbances:

$$u_{opt,i}(k) = -K_{LQ,i}x_i(k) + u_{ff,i}(k)$$

The gain depends on a solution of a Riccati equation.

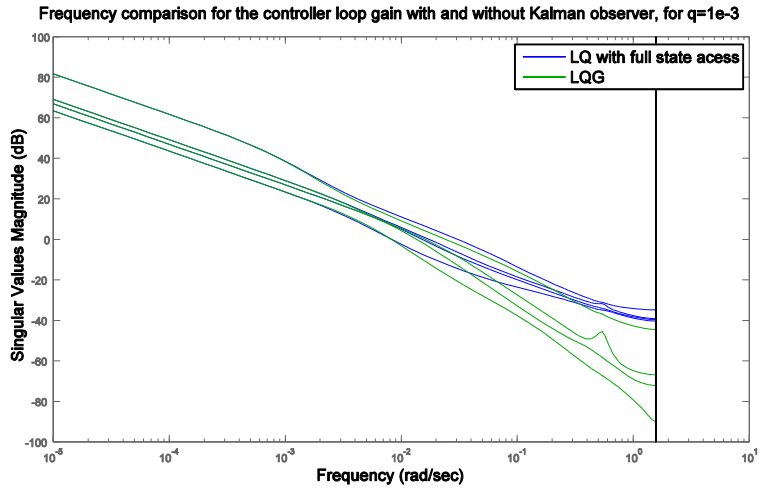
Designing local control agents: MV penalty weight

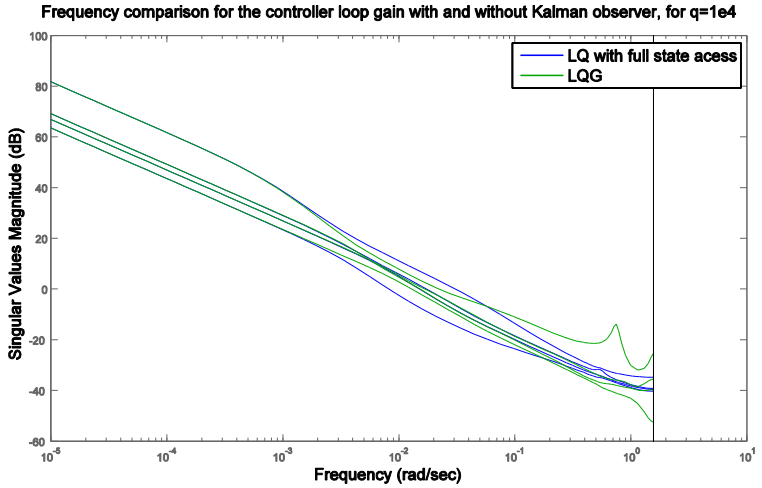
Increasing the weight on the manipulated variable effort reduces the loop gain, implying: reduces overshoot, response slower, cuts the influence to high frequency unmodeled dynamics.



Designing local control agents: State estimator

Design the Kalman filter state estimator by selecting the process noise variance so as to recover the LQ loop gain:





$q = 10^{-3}$

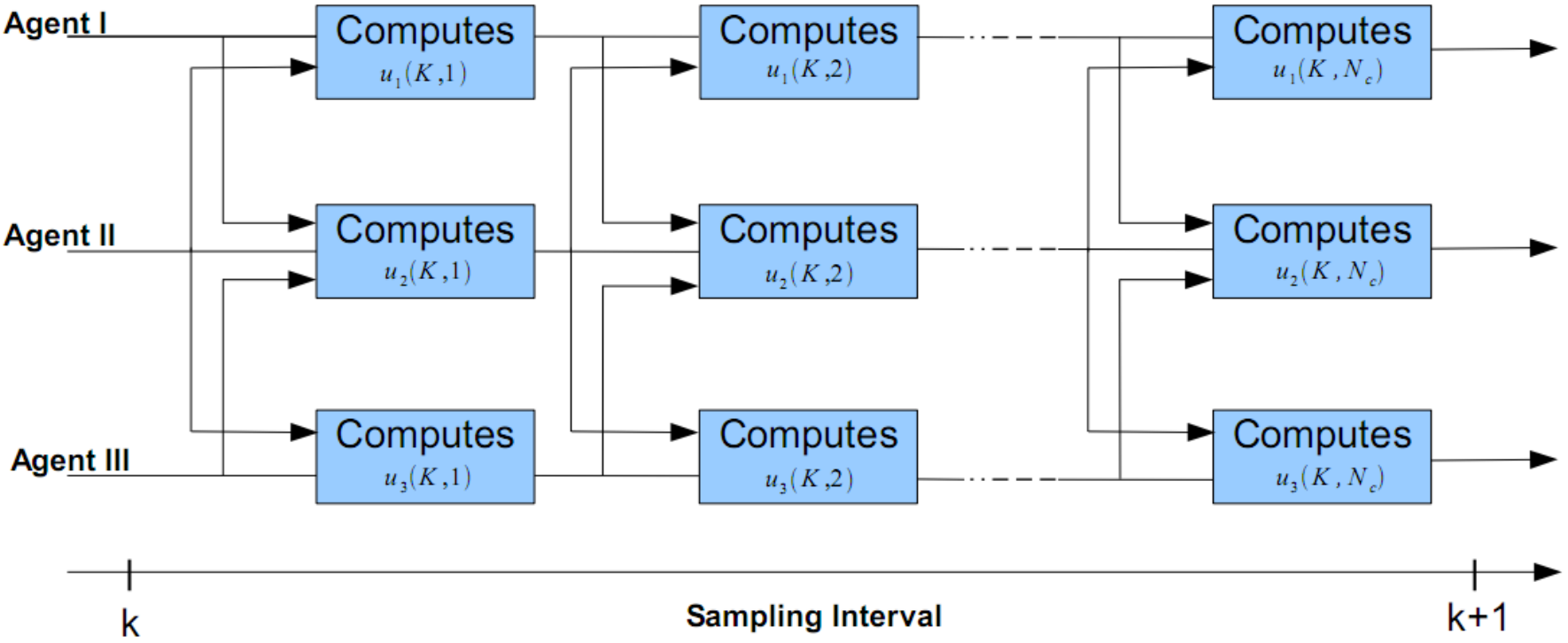
$q = 10^{4}$

Coordination among control agents (procedure)

At the beginning of each sampling interval execute the coordination recursive procedure

1. Initialize the manipulated variable for each control agent;
2. For each control agent optimize its local cost, given knowledge of the manipulated variable of its neighbors in the preceding iteration, that appear as feedforward variables;
3. If a number N_c of iterations that ensures convergence is performed, then stop. Otherwise, go to step 1.

Coordination among control agents (time evolution)



Coordination among control agents (convergence)

From iteration $l-1$ to iteration l, in each sampling interval k, the coordination procedure progresses as the state of a linear system:

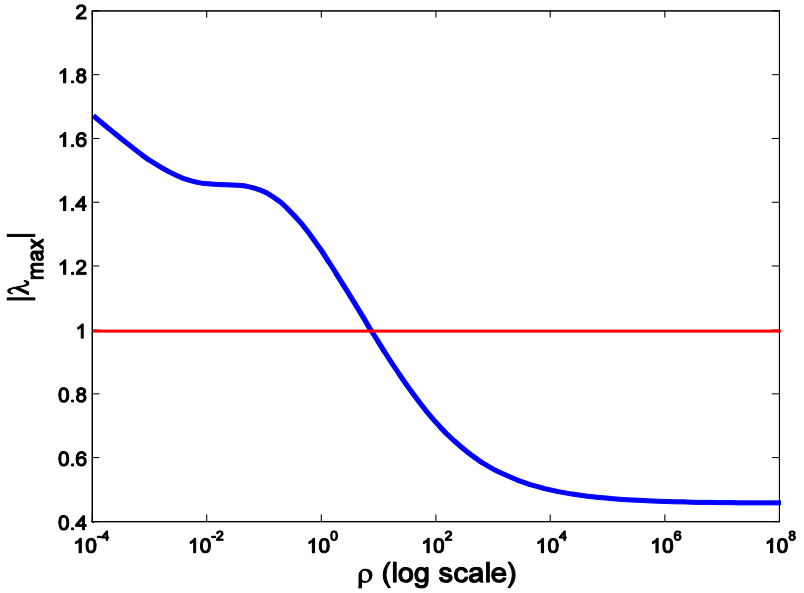
$$u(k,l) = \Xi u(k,l-1) + \Psi$$

where Ξ is a matrix and Ψ is a vector.

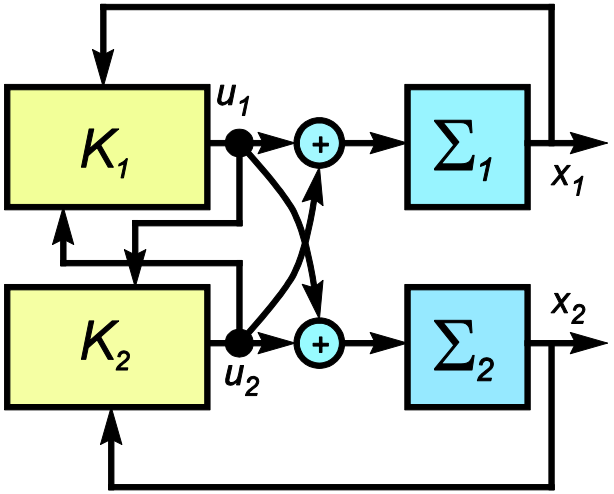
Converges if $\max\left(eig\left(\Xi\right)\right) < 1$.

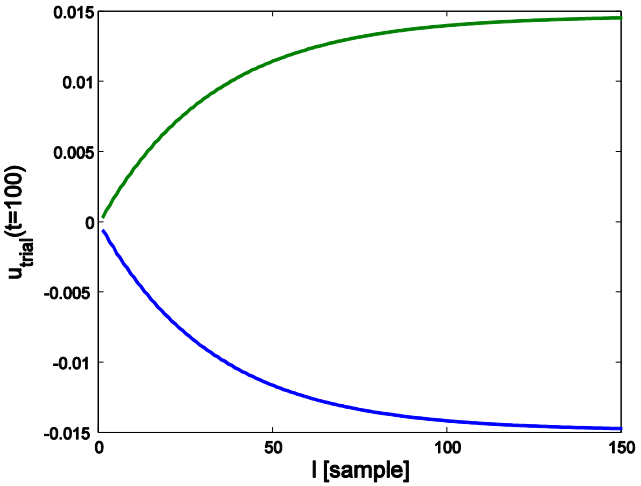
This spectral radius is affected by the control penalty weights ρ.

If all the ρ are equal:



A "synthetic" example: Distributed control of the double integrator



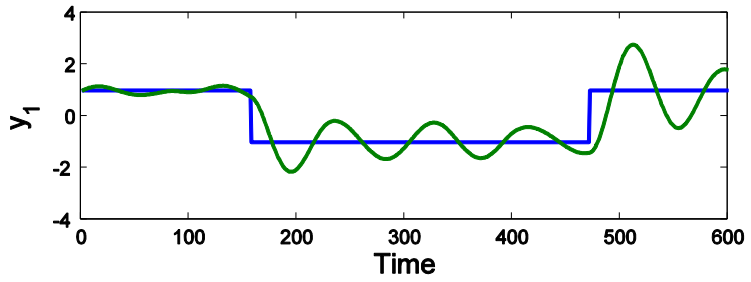


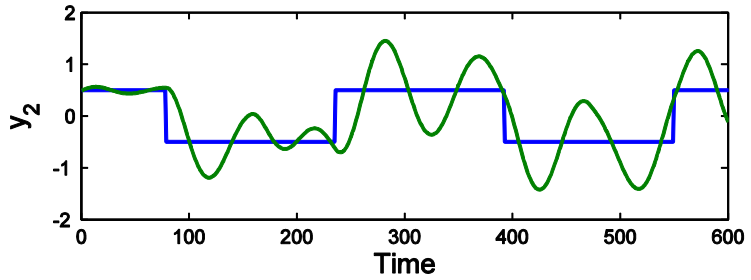
The input of each integrator is a disturbance to the other.

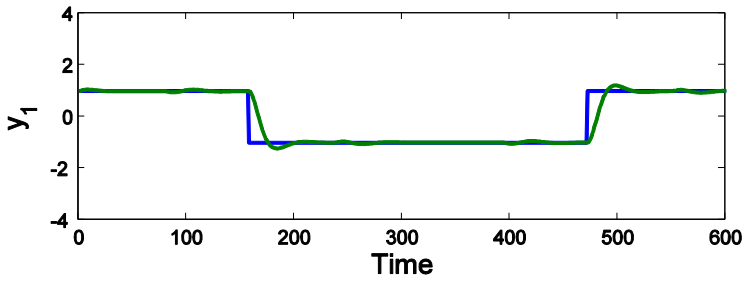
Each local control agent controls one of the integrators.

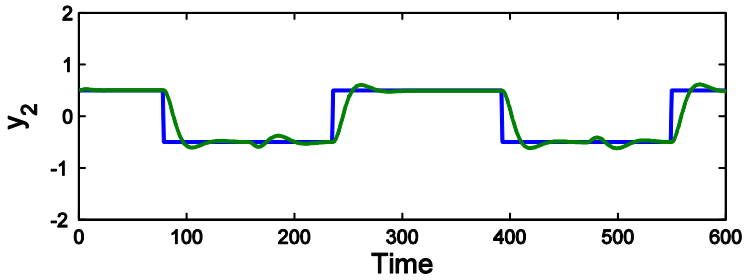
During one sampling interval, the coordination procedure adjusts the manipulated variables as indicated.

Distributed control of the double integrator (conclusion)





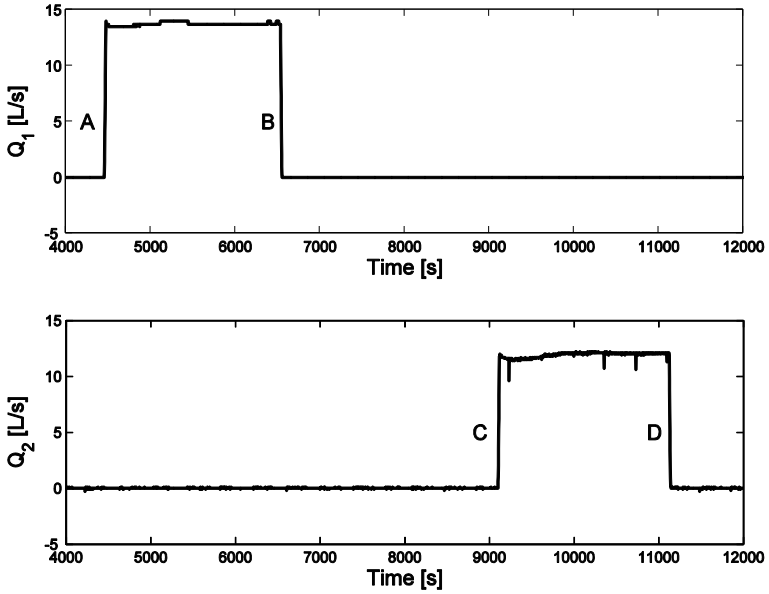


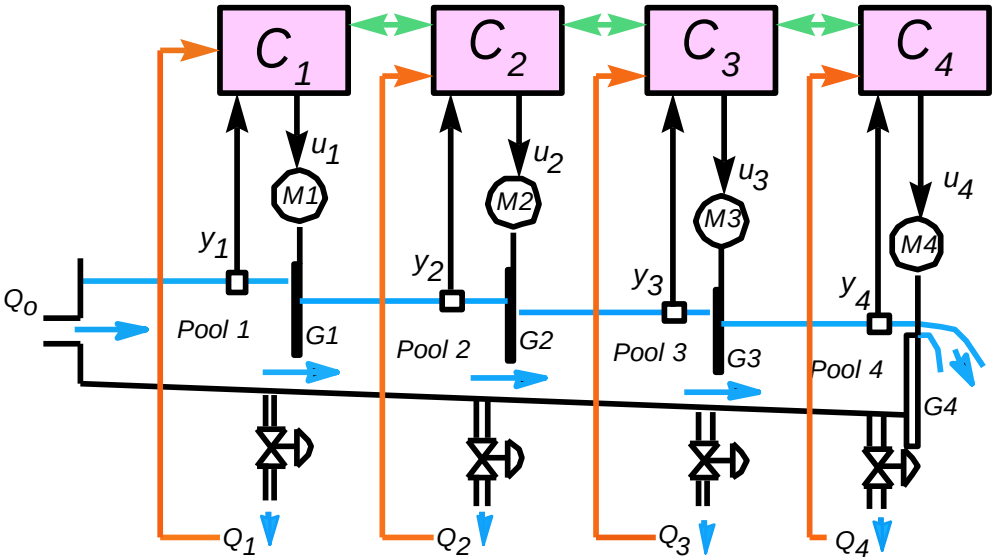


$N_c = 10$ $N_c = 100$

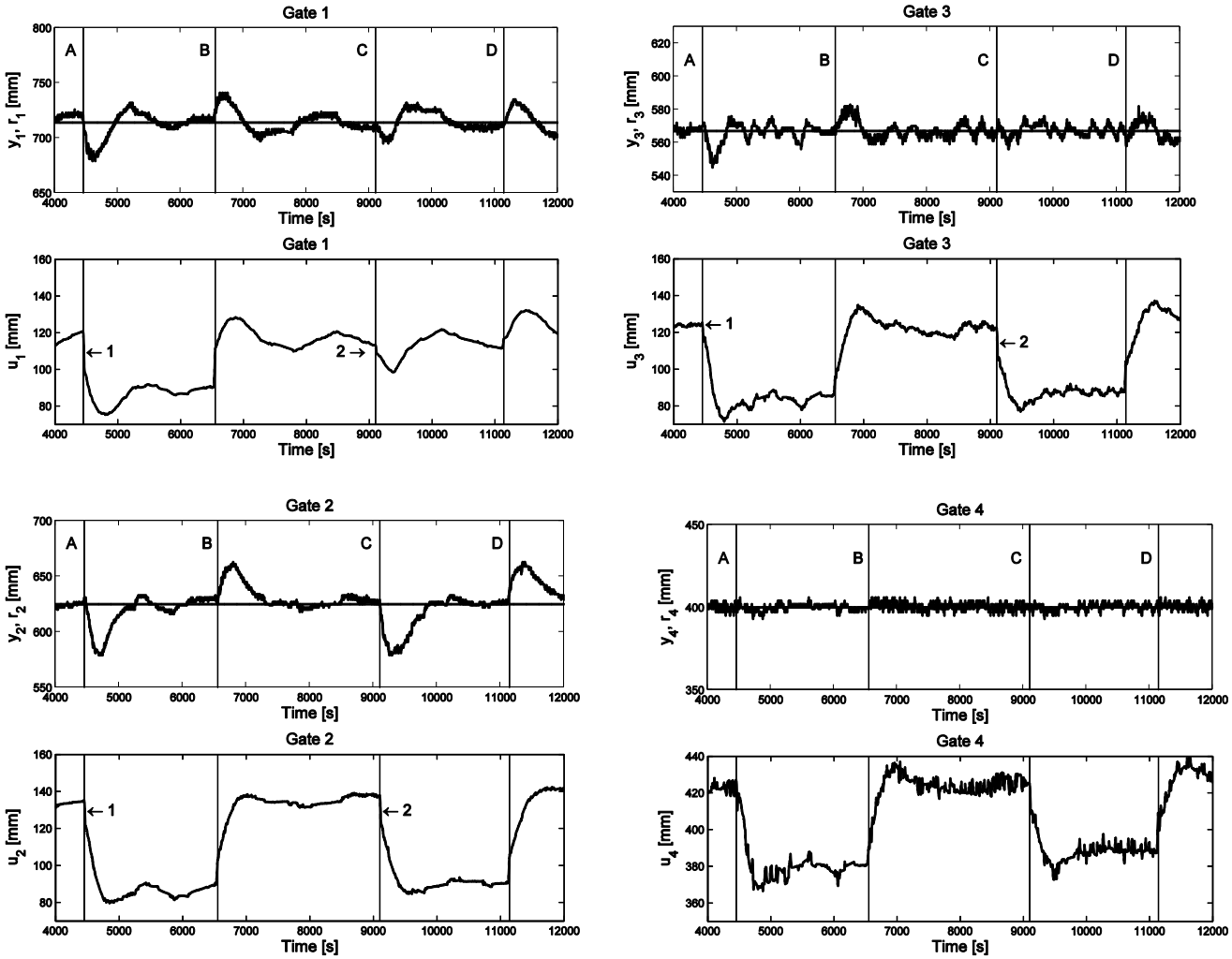
Increasing the number of coordination steps N_c, the performance improves.

Experimental results (Rejecting disturbances)



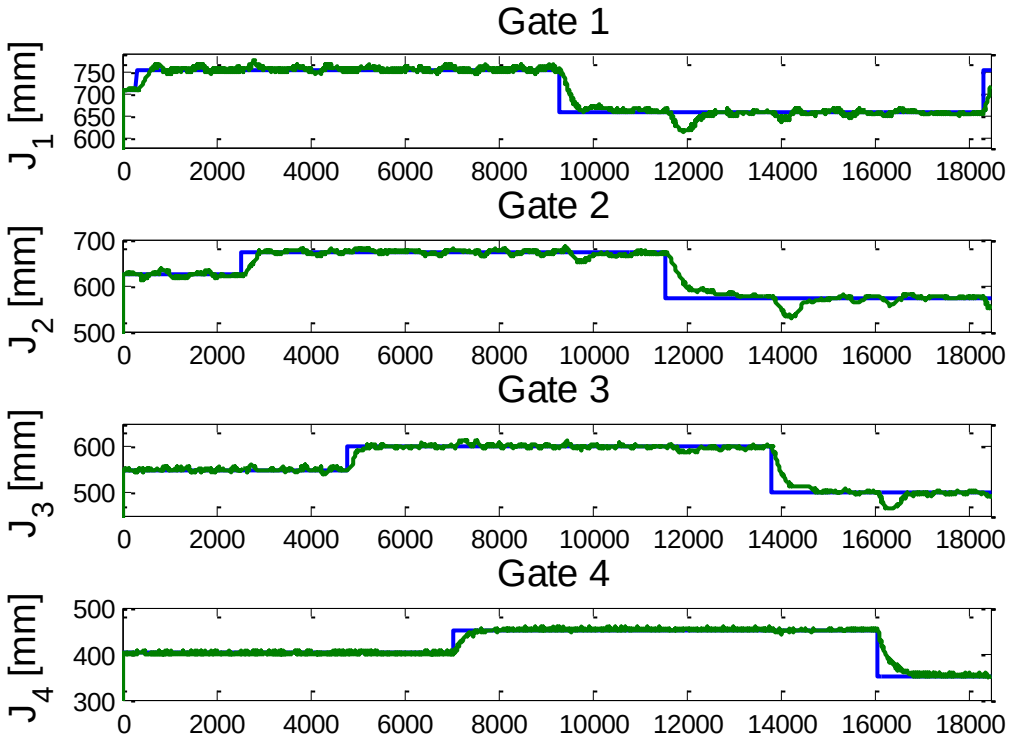


Control objective: Keep the downstream level of each pool close to a reference level when an oftake valve is open.

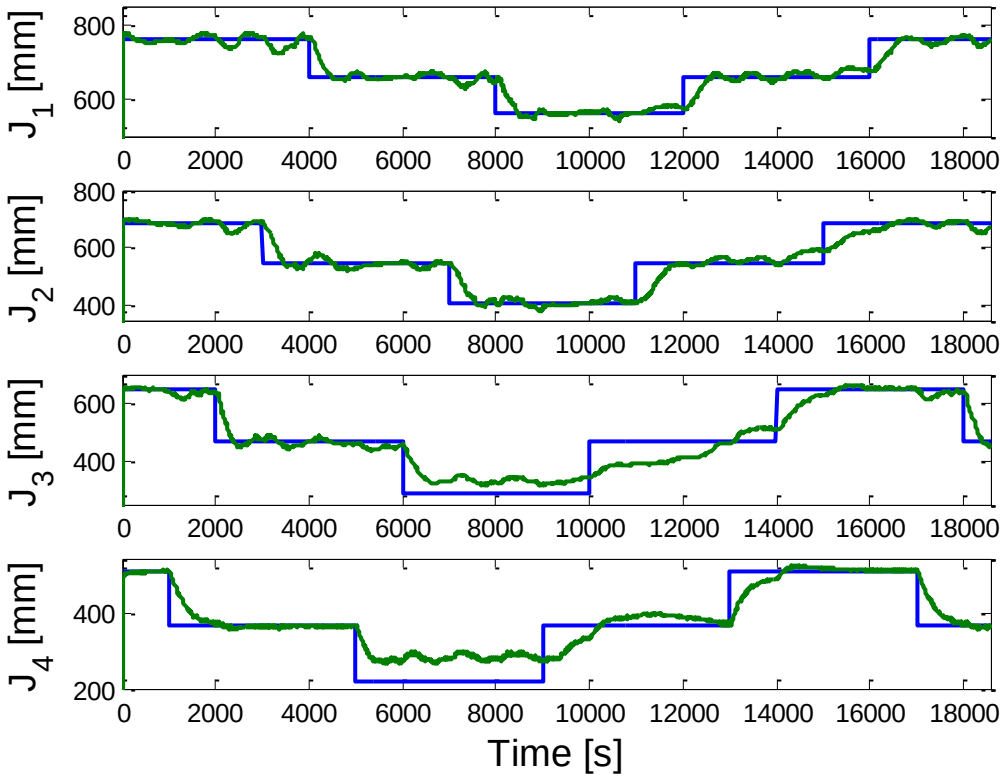


Remark: The propagation from local controller to local controller of the feedforward action.

Experimental results (reference tracking)



Experimental results with LQG – Changing the operating point



Distributed MPC

A precursor: Prof. Gyro Gearloose and the problem of firefly catching

Gyro Gearlosse: a MPC precursor that does not often receive due credit.

An alternative: "Altruistic" control

Each local control agent minimizes a quadratic cost that considers not only its tracking error but also the tracking errors of the neighbors:

$$J_1 = \sum_{i=1}^{N} \left[e_1^2(k+i) + e_2^2(k+i) + \rho_1 \left(\Delta u_1(k+i-1) \right)^2 \right]$$

$$J_2 = \sum_{i=1}^{N} \left[e_1^2(k+i) + e_2^2(k+i) + e_3^2(k+i) + \rho_2 \left(\Delta u_2(k+i-1) \right)^2 \right]$$

$$J_2 = \sum_{i=1}^{N} \left[e_2^2(k+i) + e_3^2(k+i) + \rho_1 \left(\Delta u_3(k+i-1) \right)^2 \right]$$

$e_i(k) = r_i(k) - y_i(k)$ is the tracking error for the level in pool i at time k.

Optimization under stability constraints

Each local cost *i* is optimized under the following terminal constraints:

$$y_i(k+N+j) = r_i(k+N+j) \quad \text{for} \quad j = 1, \ldots, P$$

$N+P$ is the prediction horizon and P is the number of coincidence points.

These constraints ensure stability of the local control loops taken in isolation if $P \geq \dim(x)$ but do not ensure per se the stability of the overall system.

With additional assumptions it is possible to ensure stability of the overall system.

Selecting N and P determines the type of response, as will be shown.

Coordination among control agents (procedure)

At the beginning of each sampling interval execute the coordination recursive procedure

4. Initialize the manipulated variable for each control agent;
5. For each control agent optimize its local cost, given knowledge of the manipulated variable of its neighbors in the preceding iteration, that appear as feedforward variables;
6. If a number N_c of iterations that ensures convergence is performed, then stop. Otherwise, go to step 1.

Coordination among control agents (convergence)

From iteration $l-1$ to iteration l, in each sampling interval k, the coordination procedure progresses as the state of a linear system:

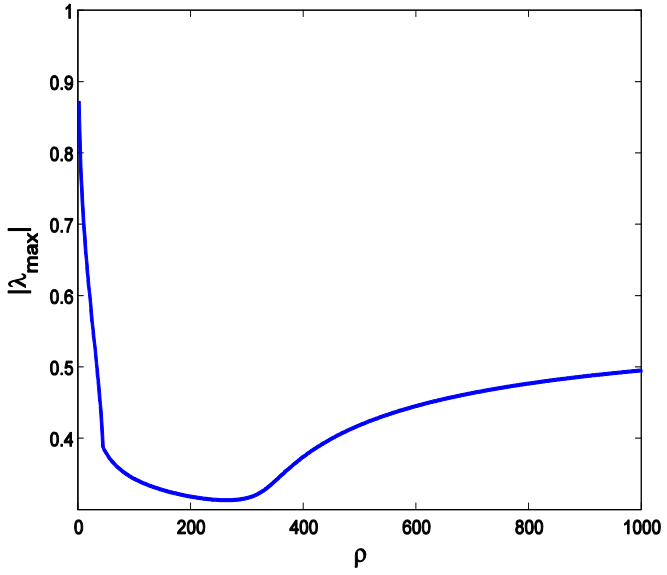
$$u(k,l) = \Xi u(k,l-1) + \Psi$$

where Ξ is a matrix and Ψ is a vector.

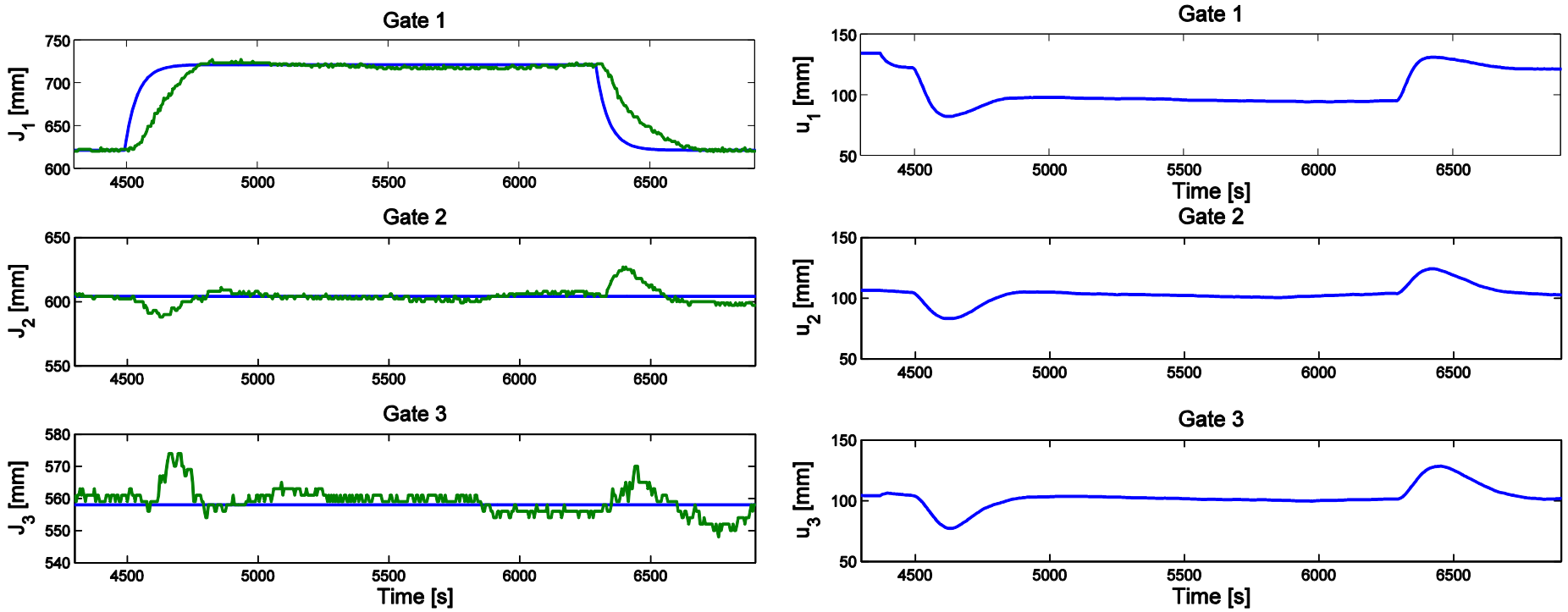
Converges if $\max\left(eig\left(\Xi\right)\right) < 1$.

This spectral radius is affected by the control penalty weights ρ.

If all the ρ are equal:

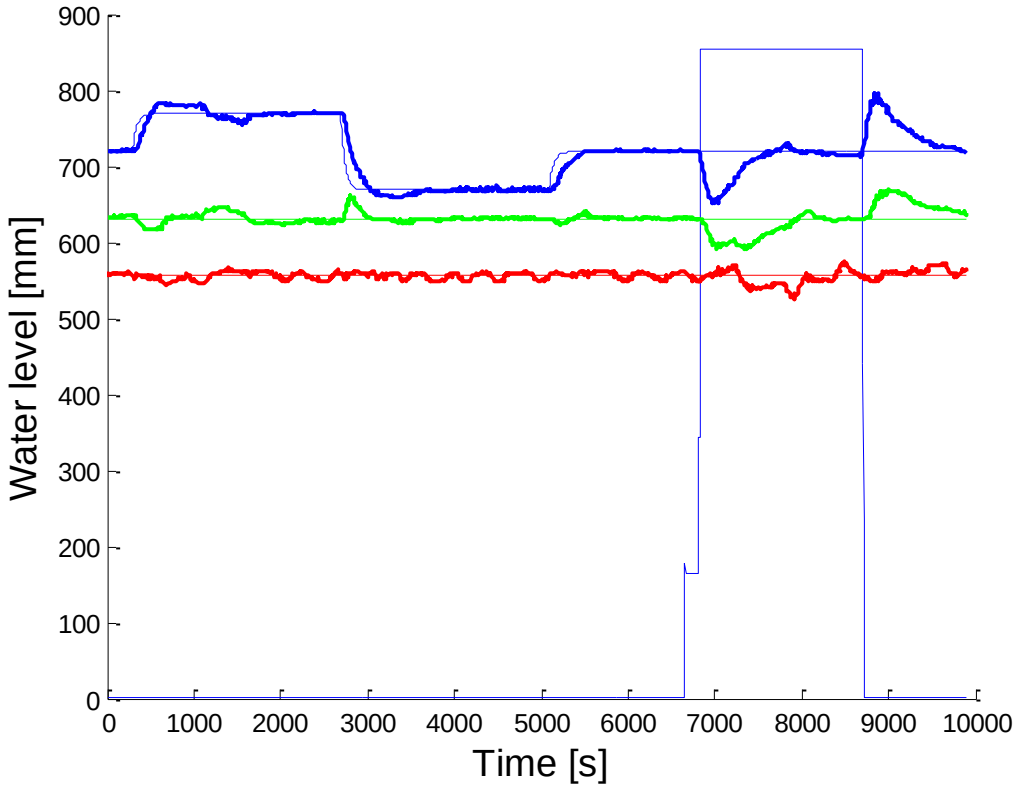


Experimental results with D-SIORHC (1)



Response to a step in the reference of the level of pool 1.

Experimental results with D-SIORHC (2)



Conclusions

- A distributed framework for water delivery canal networks based on multiple control agents
  - Decompose the plant
  - Specify the communication structure
  - Specify the local control laws
  - Specify the coordination algorithm
- Examples (MPC, LQG) of coordination algorithms with low computational load.
- Many ideas are applicable to plant networks other than water delivery

What next?

- Coordination algorithms with low computational load and low communication requirements that yield good approximations to the global optimum.
- Distributed minimum attention control (minimize the rate of change of the manipulated variable – Brocket).
- Adaptive distributed control.
- Autonomous systems based on multiple cooperating control agents.
- Fault tolerant control and reconfigurable control
- Classes of applications (e. g. transport systems).