MODEL PREDICTIVE CONTROL

STOCHASTIC MPC

Alberto Bemporad

http://cse.lab.imtlucca.it/~bemporad

©2019 A. Bemporad - "Model Predictive Control"

COURSE STRUCTURE

- ✔ Linear model predictive control (MPC)
- ✔ Linear time-varying and nonlinear MPC
- ✔ MPC computations: quadratic programming (QP), explicit MPC
- ✔ Hybrid MPC
- Stochastic MPC
- Data-driven MPC

MATLAB Toolboxes:

- **MPC Toolbox** (linear/explicit/parameter-varying MPC)
- **Hybrid Toolbox** (explicit MPC, hybrid systems)

Course page:

`http://cse.lab.imtlucca.it/~bemporad/mpc_course.html`

STOCHASTIC MODEL PREDICTIVE CONTROL

OPTIMIZE DECISIONS UNDER UNCERTAINTY

- In many control problems decisions must be taken under **uncertainty**

renewable power

prices

water

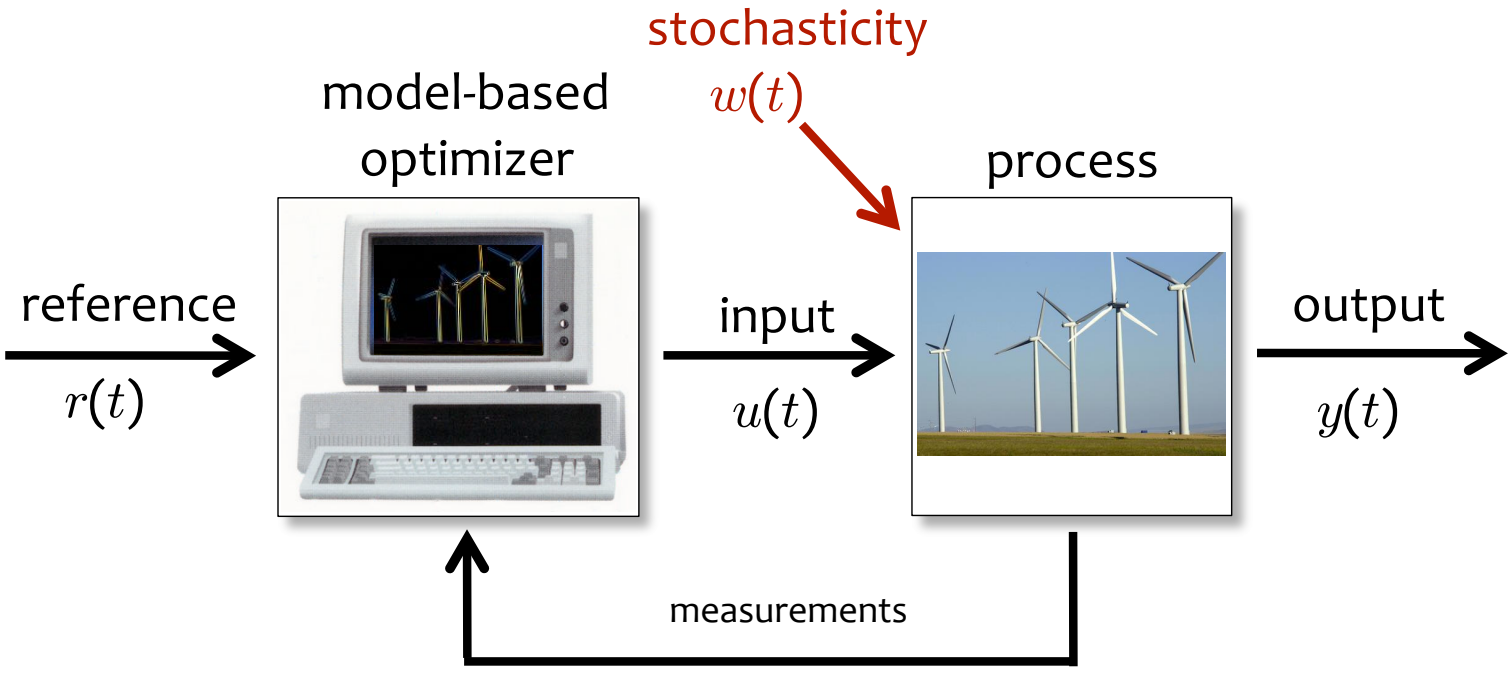
demand

human (inter)action

- **Robust** control approaches do not model uncertainty (only assume that is bounded) and pessimistically consider the worst case
- **Stochastic models** provide instead additional information about uncertainty
- **Optimality** often sought (ex: minimize expected economic cost)

©2019 A. Bemporad - "Model Predictive Control"

STOCHASTIC MODEL PREDICTIVE CONTROL (SMPC)



*Use a **stochastic** dynamical **model** of the process to **predict** its possible future evolutions and choose the "best" **control** action*

©2019 A. Bemporad - "Model Predictive Control"

STOCHASTIC MODEL PREDICTIVE CONTROL

- At time t: solve a **stochastic optimal control** problem over a finite future horizon of N steps:

$$\min_u \quad E_w\left[\sum_{k=0}^{N-1} \ell(y_k, u_k, w_k)\right]$$

s.t.

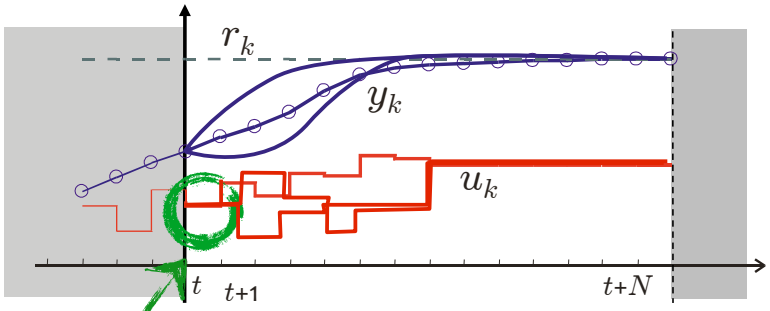
$$x_{k+1} = A(w_k)x_k + B(w_k)u_k + f(w_k)$$

$$y_k = C(w_k)x_k + D(w_k)u_k + g(w_k)$$

$$u_{\min} \le u_k \le u_{\max}$$

$$y_{\min} \le y_k \le y_{\max}, \forall w \quad \text{robustness}$$

$$x_0 = x(t) \quad \text{feedback}$$



$x(t)$ = process state

$u(t)$ = manipulated vars

$y(t)$ = controlled output

$w(t)$ = **stochastic disturbances**

- Solve stochastic optimal control problem w.r.t. future input sequence
- Apply the first optimal move $u(t)=u_0^*$, throw the rest of the sequence away
- At time $t+1$: Get new measurements, repeat the optimization. And so on …

©2019 A. Bemporad - "Model Predictive Control"

LINEAR STOCHASTIC MODEL W/ DISCRETE DISTURBANCE

- **Linear stochastic** prediction model

$$\begin{cases} x_{k+1} &= A(w_k)x_k + B(w_k)u_k + f(w_k) \\ y_k &= C(w_k)x_k + g(w_k) \end{cases}$$

- **Discrete disturbance**

$$w_k \in \{w^1, \dots, w^s\}$$

with discrete probabilities $p_j = \Pr\left[w_k = w^j\right], p_j \geq 0, \sum_{j=1}^{s} p_j = 1$

- (A, B, C) can be sparse matrices (ex: network of interacting subsystems), while often w_k is low-dimensional (ex: electricity price, weather, etc.)

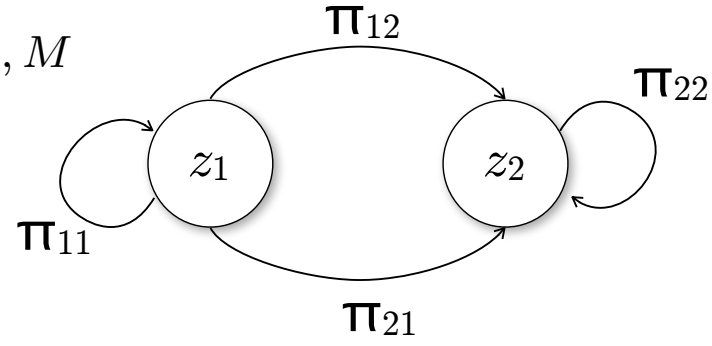
LINEAR STOCHASTIC MODEL W/ DISCRETE DISTURBANCE

- Probabilities p_j can be time varying, $p_j(t)$, and have their own dynamics

  Example: Markov chain

$$\pi_{ih} = \mathbf{Pr}[z(t+1) = z_h \mid z(t) = z_i], \;\; i,h = 1,\dots,M$$

$$p_j(t) = \begin{cases} e_{1j} & \text{if} \;\; z(t) = z_1 \\ \vdots & \vdots \\ e_{Mj} & \text{if} \;\; z(t) = z_M \end{cases}$$

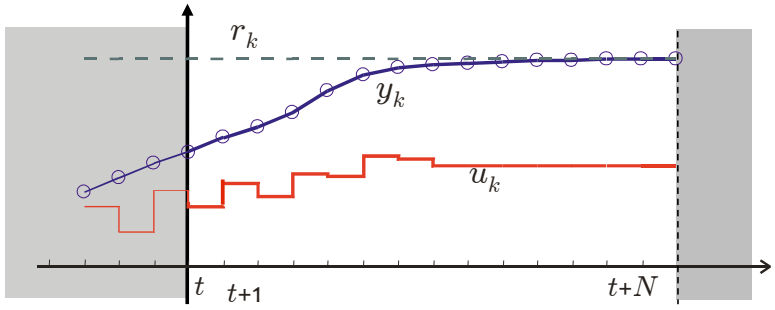


- Discrete distributions can be **estimated from historical data** (and adapted on-line)

COST FUNCTIONS FOR SMPC TO MINIMIZE

- **Expected performance**

$$\min_u \sum_{k=0}^{N-1} E_w\left[(y_k - r_k)^2\right]$$



- Tradeoff between **expectation & risk**

$$\min_u \sum_{k=0}^{N-1} (E_w\left[y_k - r_k\right])^2 + \alpha \mathrm{Var}_w\left[y_k - r_k\right] \qquad \alpha \geq 0$$

- Note that they coincide for α=1, since

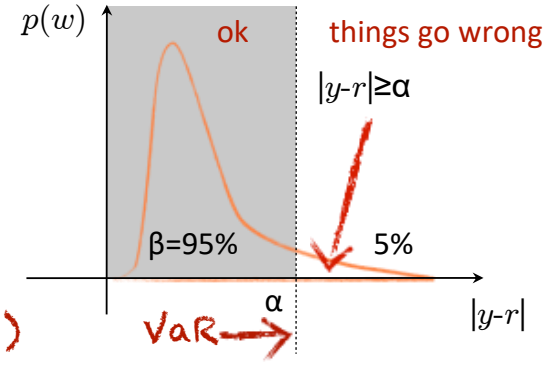
$$\mathrm{Var}_w\left[y_k - r_k\right] = E_w\left[(y_k - r_k)^2\right] - (E_w\left[y_k - r_k\right])^2$$

©2019 A. Bemporad - "Model Predictive Control"

COST FUNCTIONS FOR SMPC TO MINIMIZE

- Conditional Value-at-Risk (**CVaR**) (Rockafellar, Uryasev, 2000)

$$\min_{u,\alpha} \sum_{k=0}^{N-1} \alpha_k + \frac{1}{1-\beta} E_w[\max\{|y_k - r_k| - \alpha_k, 0\}]$$

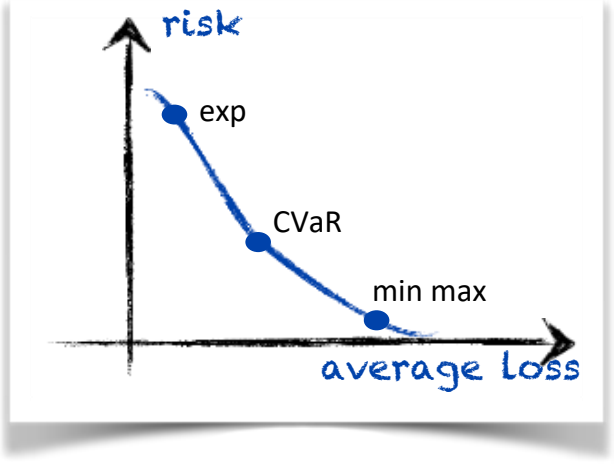


= minimize expected loss when things go wrong (convex !)

= expected shortfall

- **Min-max** = minimize worst case performance

$$\min_{u} \sum_{k=0}^{N-1} \max_{w} |y_k - r_k|$$



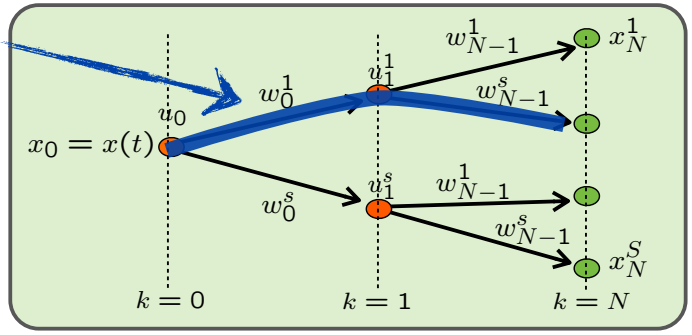
©2019 A. Bemporad - "Model Predictive Control"

STOCHASTIC OPTIMAL CONTROL PROBLEM

- Enumerate all possible scenarios $\{w_0^j, w_1^j, \dots, w_{N-1}^j\},\ j = 1, \dots, S$

  scenario = path on the tree

  number S of scenarios = number of leaf nodes



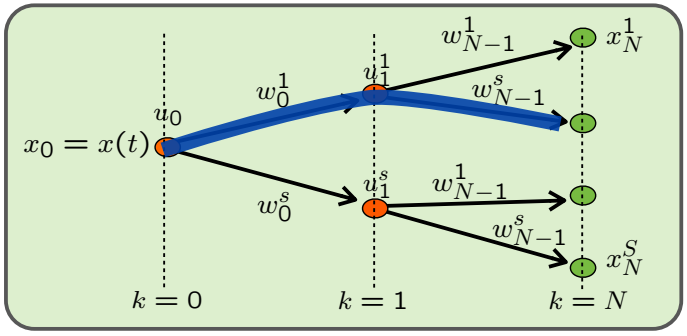
- Each scenario has probability $p^j = \prod_{k=0}^{N-1} \mathbf{Pr}[w_k = w_k^j]$

STOCHASTIC OPTIMAL CONTROL PROBLEM

- Each scenario has its own evolution

$$x_{k+1}^j = A(w_k^j)x_k^j + B(w_k^j)u_k^j + f(w_k^j)$$

(=linear time-varying system)



- Expectations become simple sums !

Ex: $\min E_w \left[x_N' P x_N + \sum_{k=0}^{N-1} x_k' Q x_k + u_k' R u_k \right]$

$$\min \sum_{j=1}^{S} p^j \left((x_N^j)' P x_N^j + \sum_{k=0}^{N-1} (x_k^j)' Q x_k^j + (u_k^j)' R u_k^j \right)$$

Expectations of quadratic costs remain quadratic costs

©2019 A. Bemporad - "Model Predictive Control"

STOCHASTIC OPTIMAL CONTROL PROBLEM

- CVaR optimization (Rockafellar, Uryasev, 2000)

$$\min_{u,\alpha} \sum_{k=0}^{N-1} \alpha_k + \frac{1}{1-\beta} E_w \left[\max\{|y_k - r_k| - \alpha_k, 0\}\right]$$

$$\begin{aligned} \min_{u,z,\alpha} \quad & \sum_{k=0}^{N-1} \alpha_k + \frac{1}{1-\beta} \sum_{j=1}^{S} p^j z_k^j \\ \text{s.t.} \quad & z_k^j \geq y_k^j - r_k^j - \alpha_k \\ & z_k^j \geq r_k^j - y_k^j - \alpha_k \\ & z_k^j \geq 0 \end{aligned}$$

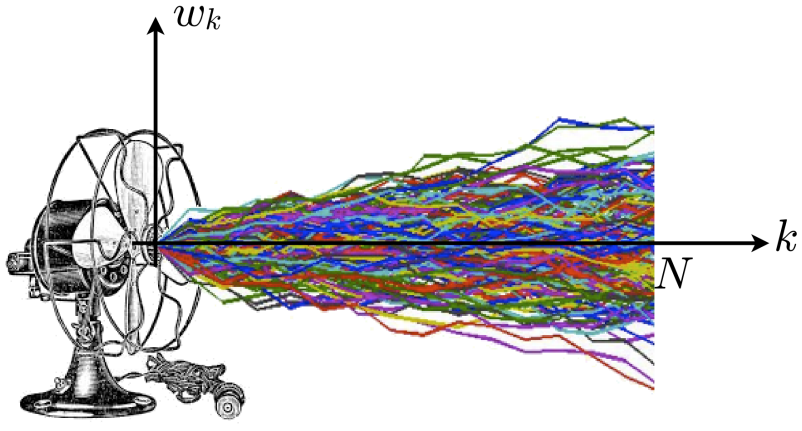
CVaR optimization becomes a linear programming problem

SCENARIO TREE GENERATION FROM DATA

- Scenario trees can be generated by **clustering** sample paths

- Paths can be obtained by **Monte Carlo simulation** of (estimated) models, or from **historical data**

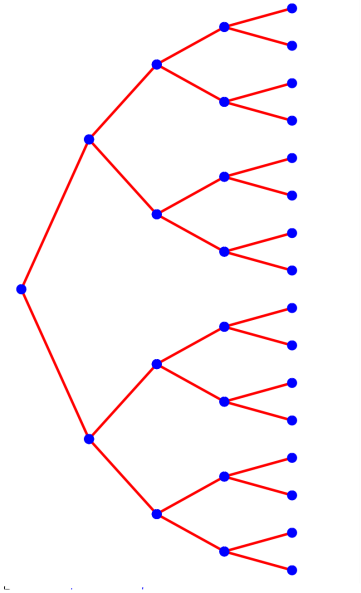
- The **number of nodes** can be decided a priori



scenario "fan" (collection of sample paths)

Heuristic Multilevel Clustering

(Heitsch, Römisch, 2009)

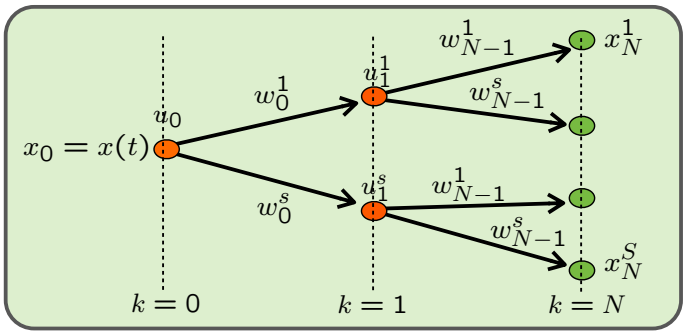


scenario tree

- Alternative (simpler/less accurate) approach: **k-means** clustering

©2019 A. Bemporad - "Model Predictive Control"

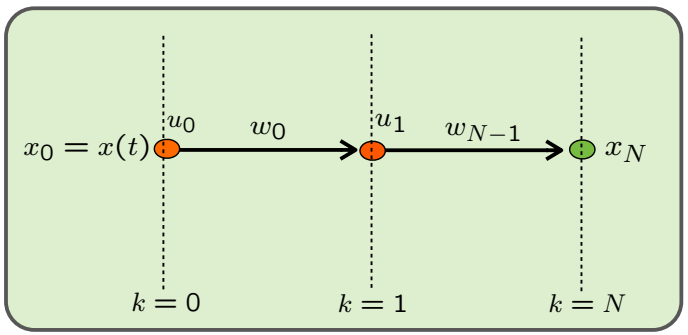
FREE CONTROL VARIABLES



Stochastic optimal control

Causality constraint: $u_k^j = u_k^h$ when scenarios j and h share the same node at prediction time k (for example: $u_0^j \equiv u_0^h$ at root node k=0)

Decision u_k only depends on past disturbance realizations $\{w_0, w_1, \ldots, w_{k-1}\}$



Deterministic control

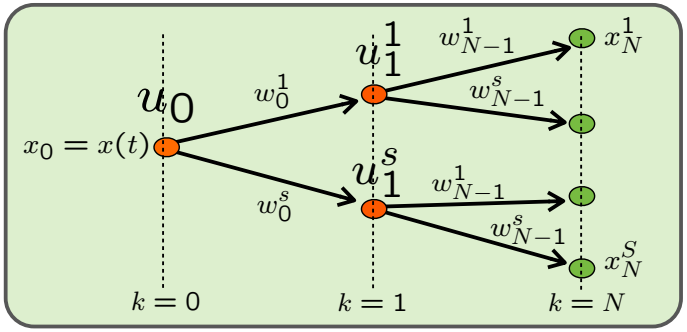
Only a sequence of disturbances is considered

- frozen-time: $w_k \equiv w(t),\ \forall k$ (causal prediction)
- prescient control: $w_k \equiv w(t+k)$ (non-causal)
- certainty equivalence: $w_k = E[w(t+k)|t]$ (causal)

We can trade off between **complexity** of optimization problem (=number of nodes) and **performance** (=accuracy of stochastic modeling)

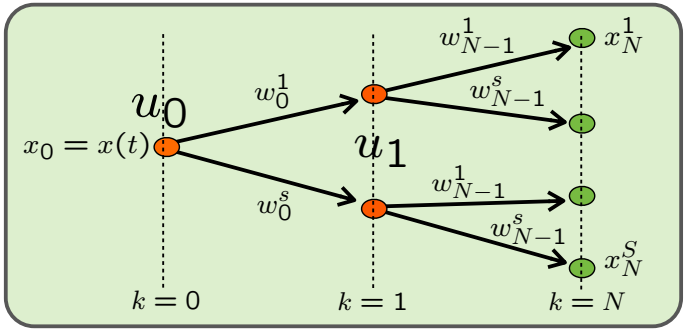
©2019 A. Bemporad - "Model Predictive Control"

OPEN-LOOP VS CLOSED-LOOP PREDICTION



closed-loop prediction

A proper move u is optimized to counteract each possible outcome of the disturbance w



open-loop prediction

Only a sequence of inputs $\{u_0, u_1, \ldots, u_{N-1}\}$ is optimized, the same u must be good for all possible disturbance w

- Intuitively: OL prediction is more conservative than CL in handling constraints
- OL problem = CL problem + additional constraints $u^j \equiv u, \ \forall j = 1, \ldots, S$ (=less degrees of freedom)

©2019 A. Bemporad - "Model Predictive Control"

LINEAR STOCHASTIC MPC FORMULATION

Existing literature on stochastic MPC

(Schwarme & Nikolaou, 1999) (Munoz de la Pena, Bemporad, Alamo, 2005) (Oldewurtel, Jones, Morari, 2008)
(Wendt & Wozny, 2000) (Couchman, Cannon, Kouvaritakis, 2006) (Ono, Williams, 2008)
(Batina, Stoorvogel, Weiland, 2002) (Primbs, 2007) (van Hessem & Bosgra 2002) (Bemporad, Di Cairano, 2005)
(Bernardini, Bemporad, 2012) Survey: (Mesbah, 2016)

- Performance index $$\min E_w \left[x_N' P x_N + \sum_{k=0}^{N-1} x_k' Q x_k + u_k' R u_k \right]$$

- Goal: ensure **mean-square convergence** $\lim_{t\to\infty} E[x'(t)x(t)] = 0$ (for $H(w(t))=0$)

- The existence of a **stochastic Lyapunov function** $V(x) = x'Px$

$$E_{w(t)}[V(x(t+1)] - V(x(t)) \le -x(t)'Lx(t), \quad \forall t \ge 0 \qquad L = L' > 0$$

(Morozan, 1983)

ensures mean-square stability

LINEAR STOCHASTIC STABILIZATION

Existing literature on stochastic MPC

(Schwarme & Nikolaou, 1999)
(Wendt & Wozny, 2000)
(Batina, Stoorvogel, Weiland, 2002)
(Bernardini, Bemporad, 2012)
(Munoz de la Pena, Bemporad, Alamo, 2005)
(Couchman, Cannon, Kouvaritakis, 2006)
(Primbs, 2007) (van Hessem & Bosgra 2002)
Survey: (Mesbah, 2016)
(Oldewurtel, Jones, Morari, 2008)
(Ono, Williams, 2008)
(Bemporad, Di Cairano, 2005)

- Performance index $\min E_w \left[x_N' P x_N + \sum_{k=0}^{N-1} x_k' Q x_k + u_k' R u_k \right]$

- Goal: ensure **mean-square convergence** $\lim_{t\to\infty} E[x'(t)x(t)] = 0$ (for $g(w(t))=0$)

- The existence of a **stochastic Lyapunov function** $V(x) = x'Px$

$$E_{w(t)}[V(x(t+1)] - V(x(t)) \le -x(t)'Lx(t), \quad \forall t \ge 0 \qquad L = L' > 0$$

(Morozan, 1983)

ensures mean-square stability

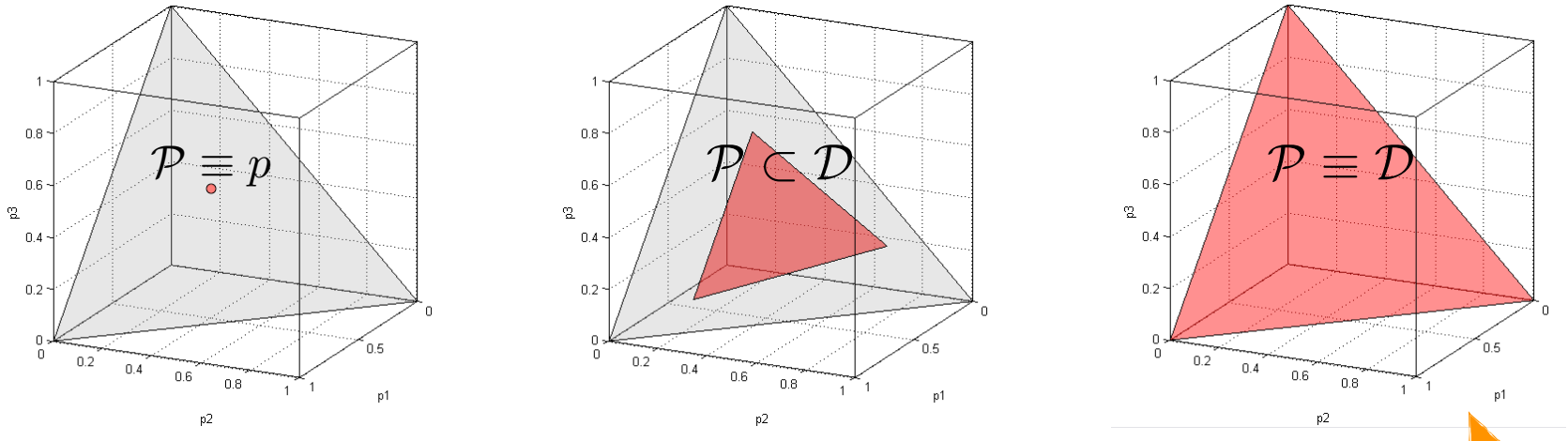
ROBUST LINEAR STOCHASTIC STABILIZATION

- The approach can be generalized to uncertain probabilities $p(t) \in \mathcal{P}$ (Example: time-varying probabilities)

probability distribution is known

uncertain or time-varying probability distribution

no probability distribution is known, $w(t)$ can vary arbitrarily



- If $\mathcal{P} \equiv \mathcal{D}$ we have a **robust** control problem *(robust convergence)*
- The more information we have about the probability distribution $p(t)$ of $w(t)$ the less conservative is the control action

©2019 A. Bemporad - "Model Predictive Control"

STABILIZING STOCHASTIC MPC

(Bernardini, Bemporad, 2012)

- Impose stochastic stability constraint in SMPC problem (=quadratic constraint w.r.t. u_0)

$$\begin{aligned}
\min_{u} \quad & E_w\left[\sum_{k=0}^{N-1} \ell(x_k, u_k)\right] \\
\text{s.t.} \quad & x_{k+1} = A(w_k)x_k + B(w_k)u_k \\
& E\left[V(A(w_0)x_0 + B(w_0)u_0)\right] \le x_0'(Q^{-1} - L)x_0 \\
& x_0 = x(t)
\end{aligned}$$

performance and stability are decoupled

- SMPC approach:
  1. Solve LMI problem off-line to find stochastic Lyapunov fcn $V(x) = x'Q^{-1}x$
  2. Optimize stochastic performance based on scenario tree

Theorem: The closed-loop system is as. stable in the mean-square sense

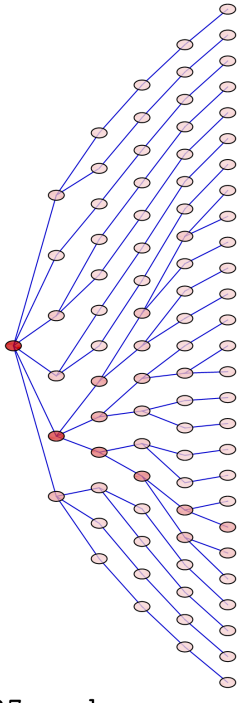
- SMPC can be generalized to handle **input and state constraints**

Note: *recursive feasibility* guaranteed by backup solution $u(k) = Kx(k)$

©2019 A. Bemporad - "Model Predictive Control"

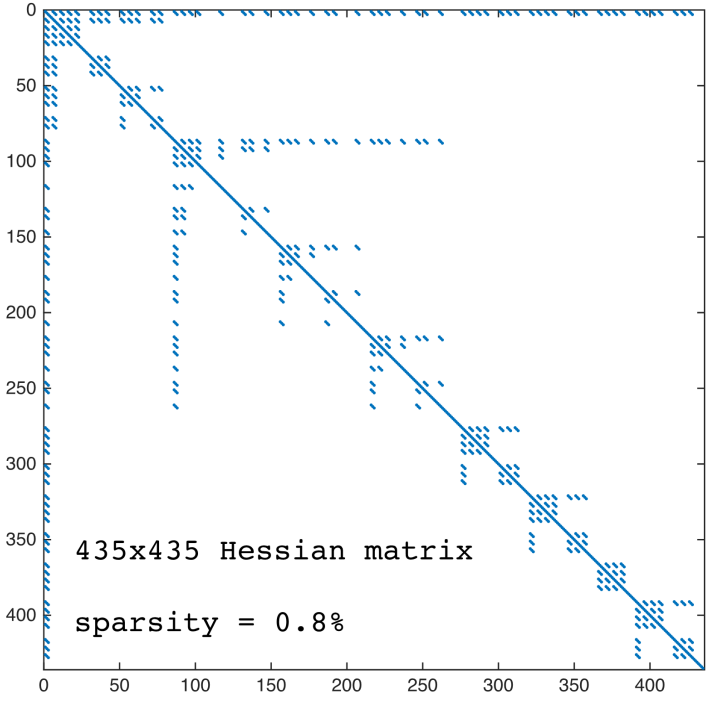
COMPLEXITY OF STOCHASTIC OPTIMIZATION PROBLEM

- #optimization variables = #nodes x #inputs (in condensed version)
- Problems are **very sparse** (well exploited by **interior point methods**)
- Example: SMPC with quadratic cost and linear constraints

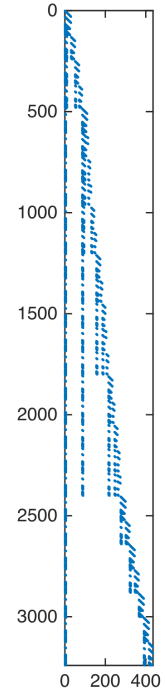


Tree=87 nodes

Branching factor M=[6 3 2 2 2]



435 free variables (5 inputs x node)



3240x435 constraint matrix
sparsity = 1.1%

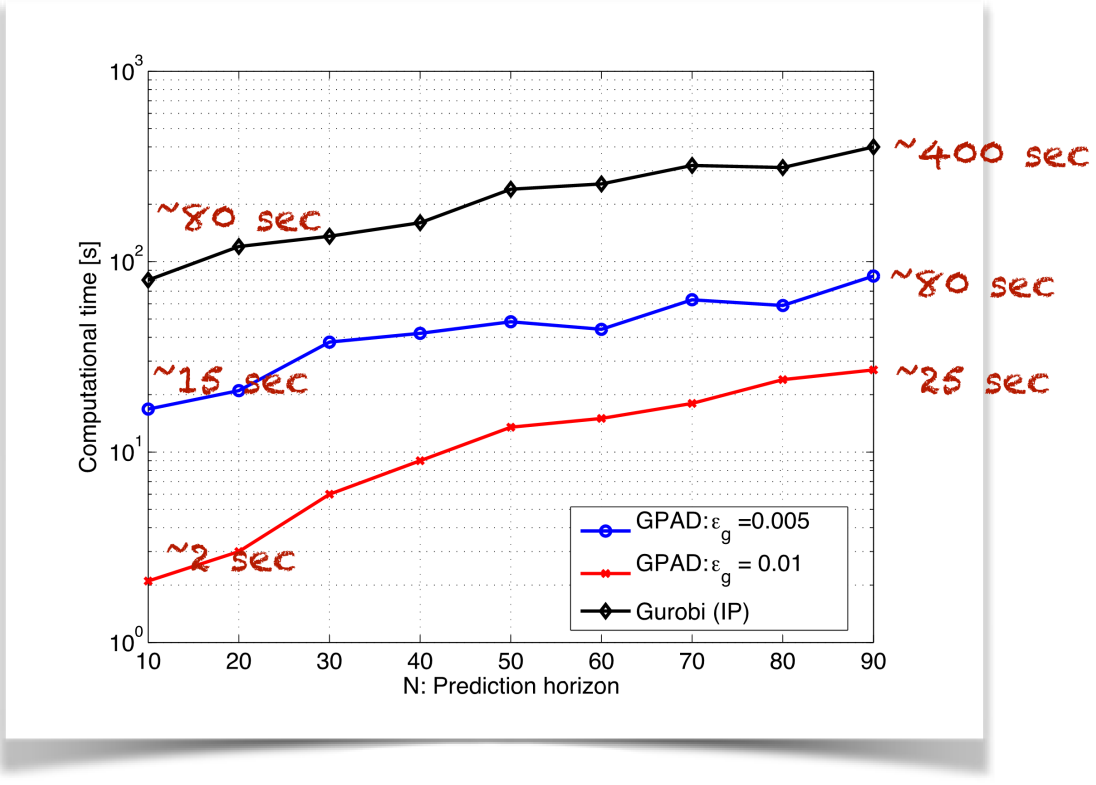
©2019 A. Bemporad - "Model Predictive Control"

DISTRIBUTED GPAD FOR STOCHASTIC MPC

- A distributed (parallelized) variant of the **A**ccelerated **G**radient **P**rojection applied to **D**ual (GPAD) for solving SMPC problems is available

(Sampathirao, Sopasakis, Bemporad, 2014)

Example: stochastic MPC with **60 states, 25 inputs, 256 scenarios**



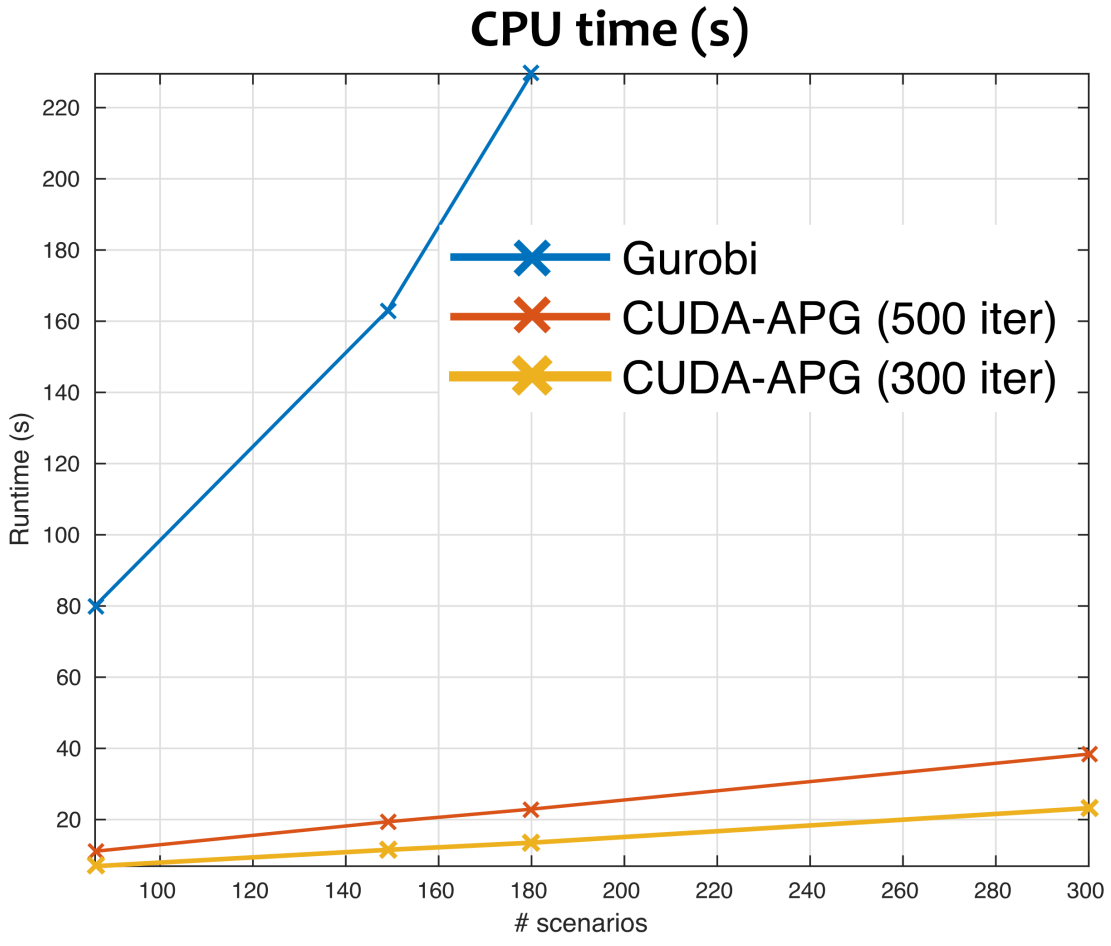
16x ÷ 40x faster
than commercial
state-of-the-art
Interior-Point method

Remark: For larger problems (*e.g.*, *50* states, *30* inputs, *9036* nodes) GUROBI gets stuck on a 4GB 4-core PC, while dGPAD can solve the problem

©2019 A. Bemporad - "Model Predictive Control"

DISTRIBUTED GPAD FOR STOCHASTIC MPC

(Sampathirao, Sopasakis, Bemporad, 2015)



APG = Accelerated Proximal Gradient, parallel implemented on NVIDIA Tesla 2075 CUDA platform

©2019 A. Bemporad - "Model Predictive Control"

A FEW SAMPLE APPLICATIONS OF SMPC

- **Energy systems**: power dispatch in smart grids, optimal bidding on electricity markets

  (Patrinos, Trimboli, Bemporad 2011)
  (Puglia, Bernardini, Bemporad 2011)

- **Financial engineering**: dynamic hedging of portfolios replicating synthetic options

  (Bemporad, Bellucci, Gabbriellini, 2009)
  (Bemporad, Gabbriellini, Puglia, Bellucci, 2010)
  (Bemporad,Puglia, Gabbriellini, 2011)

- **Water networks**: pumping control in urban drinking water networks, under uncertain demand & minimizing costs under varying electricity prices

  (Sampathirao, Sopasakis, Bemporad, 2014)

- **Automotive control**: energy management in HEVs, adaptive cruise control (human-machine interaction)

  (Di Cairano, Bernardini, Bemporad, Kolmanovsky, 2014)

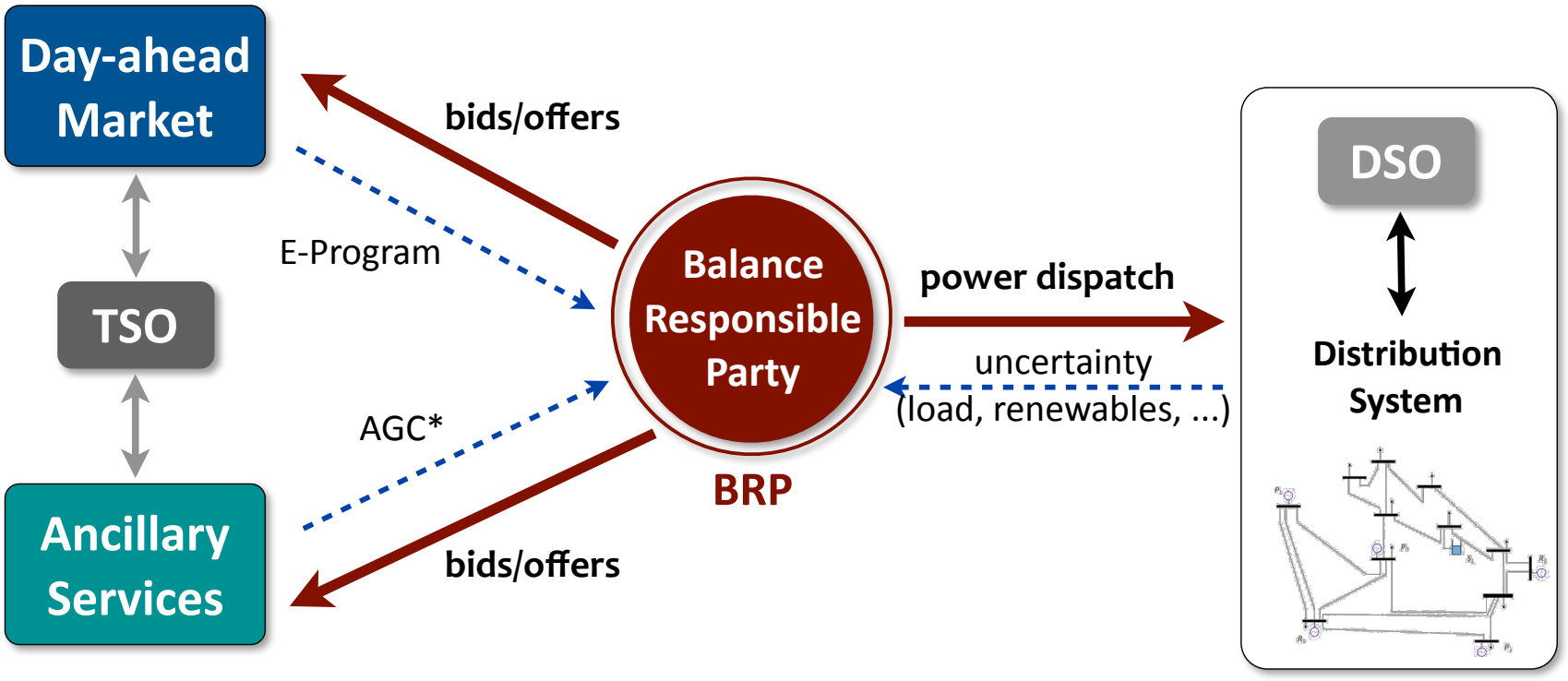
- **Networked control**: improve robustness against communication imperfections

  (Bernardini, Donkers, Bemporad, Heemels, NECSYS 2010)

©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR REAL-TIME OPTIMAL POWER DISPATCH

SMPC FOR OPERATING ON ENERGY MARKETS



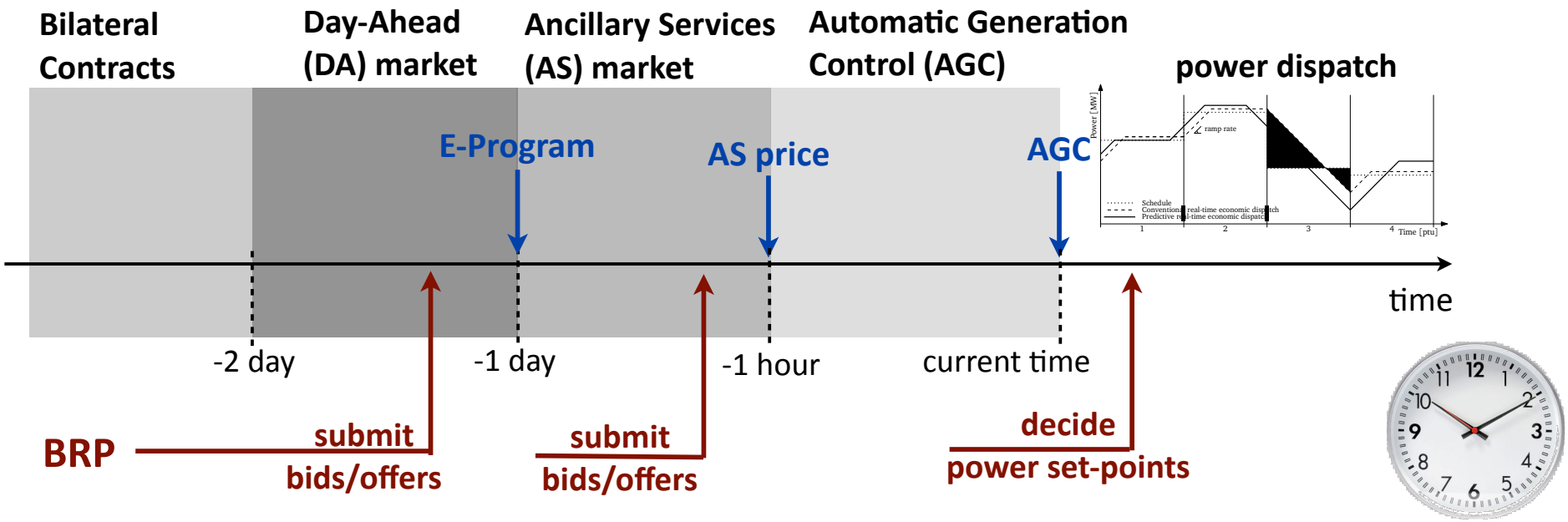
Objective: maximize BRP's profit !

while satisfying local power demand
and all grid constraints

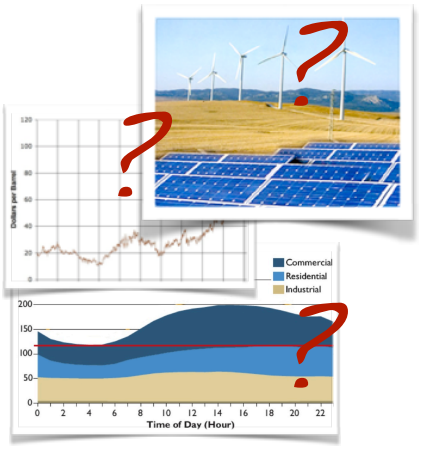
*AGC = Automatic Generation Control

©2019 A. Bemporad - "Model Predictive Control"

OPERATIONS OF ENERGY MARKET PLAYERS (BRPS)



- We can use SMPC strategies for solving:
  - The power dispatch problem
  - The two bidding problems



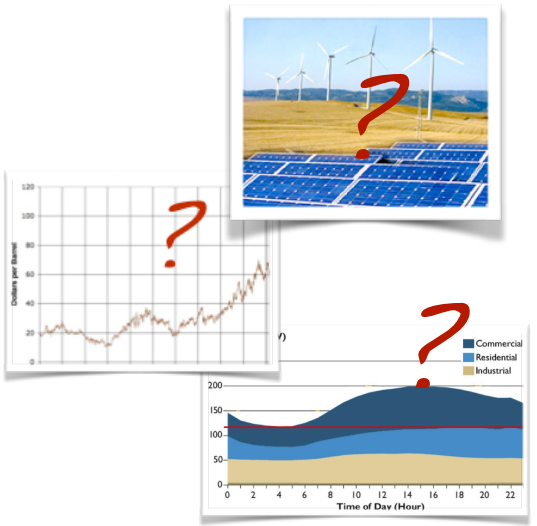
- The decision process is heavily affected by **uncertainty** (prices, load, renewables, ...)

©2019 A. Bemporad - "Model Predictive Control"

REAL-TIME POWER DISPATCH PROBLEM

- **Balance Responsible Parties (BRPs)** participate to the various energy markets and trade electricity to satisfy their loads and make profits

- Optimal control of BRPs is challenging, as in **real-time** a BRP must
  - **fulfill its E-Program**, despite perturbations induced by **uncertainties**
  - **intermittent generation** from renewable sources
  - **time-varying** internal loads
  - **react to signals** arriving from the TSO
  - **frequency deviations, AS bids** activated by the TSO
  - **minimize** generation and imbalance **costs**
  - **time-varying, stochastic** imbalance prices
  - consider plant **dynamics** and satisfy **constraints**
  - bounds on power output, **ramp-rate constraints**



©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR REAL-TIME MARKET-BASED POWER DISPATCH

(Patrinos, Trimboli, Bemporad, 2011)

- We are a legal entity (BRP) trading on the energy (PX) and ancillary service (AS) markets

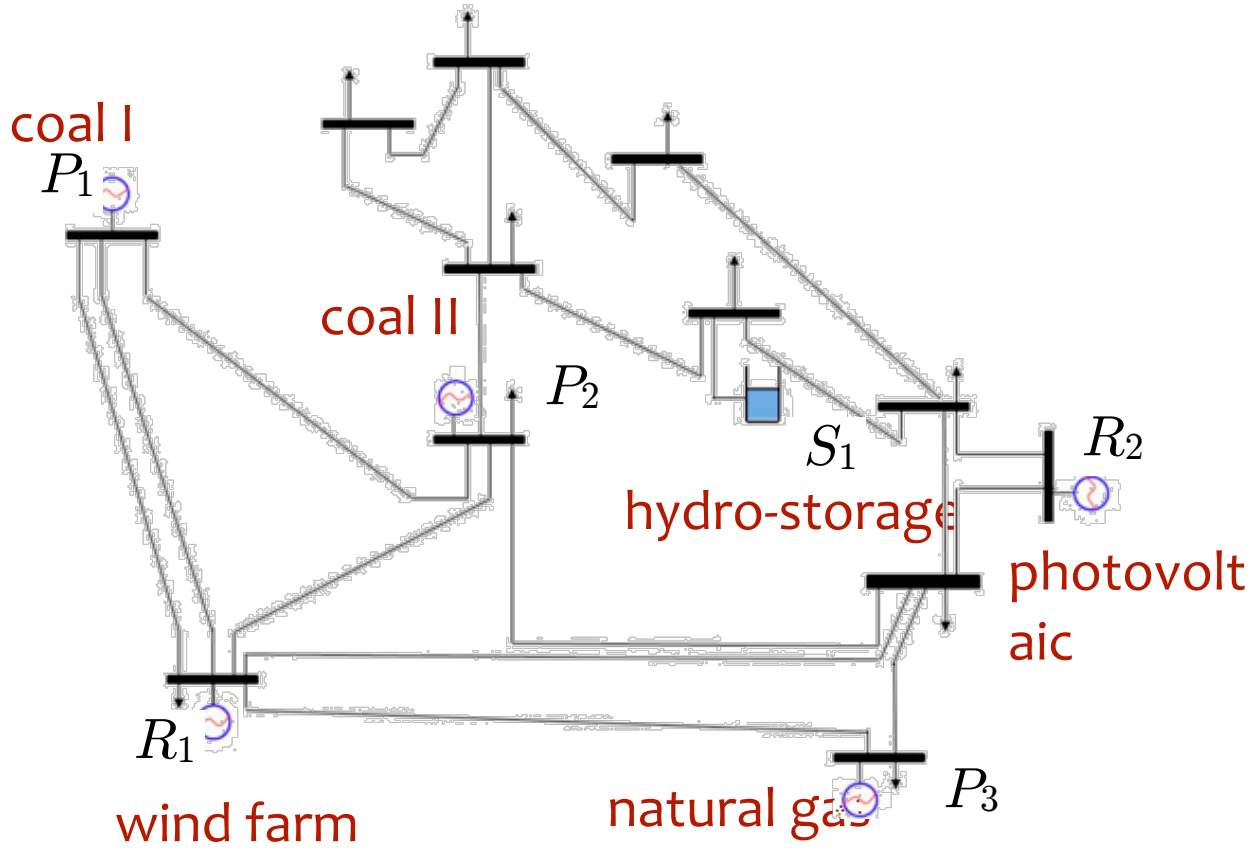
- **Objective:** Minimize costs via efficient use of intermittent resources, and maximize profits by trading on electricity (PX, AS) markets

- **Constraints**: Grid capacity, rate limits, load balancing, AS balancing

POWER DISPATCH MODEL

(Patrinos, Trimboli, Bemporad, 2011)

- Microgrid with three conventional power generators (P1,P2,P3), two renewables (R1,R2), one storage system (S1), satisfying local load and exchanging power with the grid



http://www.e-price-project.eu/

ENABLING THE FUTURE ENERGY SYSTEM

SEVENTH FRAMEWORK PROGRAMME

©2019 A. Bemporad - "Model Predictive Control"

POWER DISPATCH MODEL

- Conventional generator model (i=1,2,3)

power generated at next time unit

$$P_{i,k+1} = P_{i,k} + \Delta P_{i,k}$$

constraints on generated power: $P_{i,\min} \le P_{i,k} \le P_{i,\max}$

constraints on power variation: $\Delta P_{i,\min} \le \Delta P_{i,k} \le \Delta P_{i,\max}$

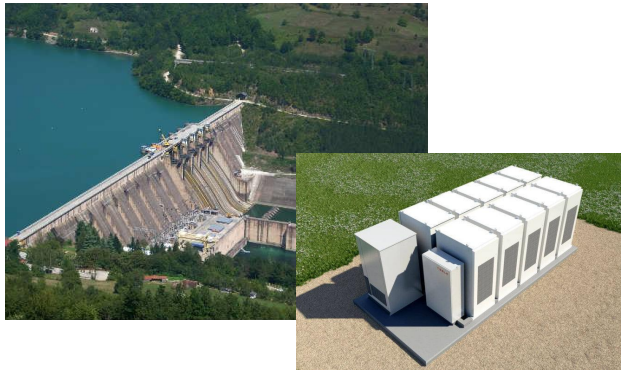
- Storage model

$$S_{k+1} = \alpha S_k + \alpha_c u_{c,k} - \frac{1}{\alpha_d} u_{d,k}$$

α = self discharge loss

α_c = charge efficiency

α_d = discharge efficiency



constraints on charge/discharge: $S_{\min} \le S_k \le S_{\max}$

constraints on charge/discharge rate: $\Delta_{S,\min} \le S_{k+1} - S_k \le \Delta_{S,\max}$

constraints on power flows: $0 \le u_{c,k} \le u_{c,\max}, \;\; 0 \le u_{d,k} \le u_{d,\max}$

©2019 A. Bemporad - "Model Predictive Control"

POWER DISPATCH MODEL

- Power exchanged with the rest of the grid (=balance)

$$P_{\text{ex},k} = P_{1,k} + P_{2,k} + P_{3,k} - u_{c,k} + u_{d,k} + (r_k - l_k)$$

r-l is STOCHASTIC !

conventional power

storage charge/ discharge

renewable power

load

- Overall linear model and constraints

$$x = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ S \end{bmatrix} \qquad u = \begin{bmatrix} \Delta P_1 \\ \Delta P_2 \\ \Delta P_3 \\ u_c \\ u_d \\ r - l \end{bmatrix} \qquad y = \begin{bmatrix} P_{\text{ex}} \\ P_1 \\ P_2 \\ P_3 \\ S \\ \Delta S \end{bmatrix}$$

uncontrolled input

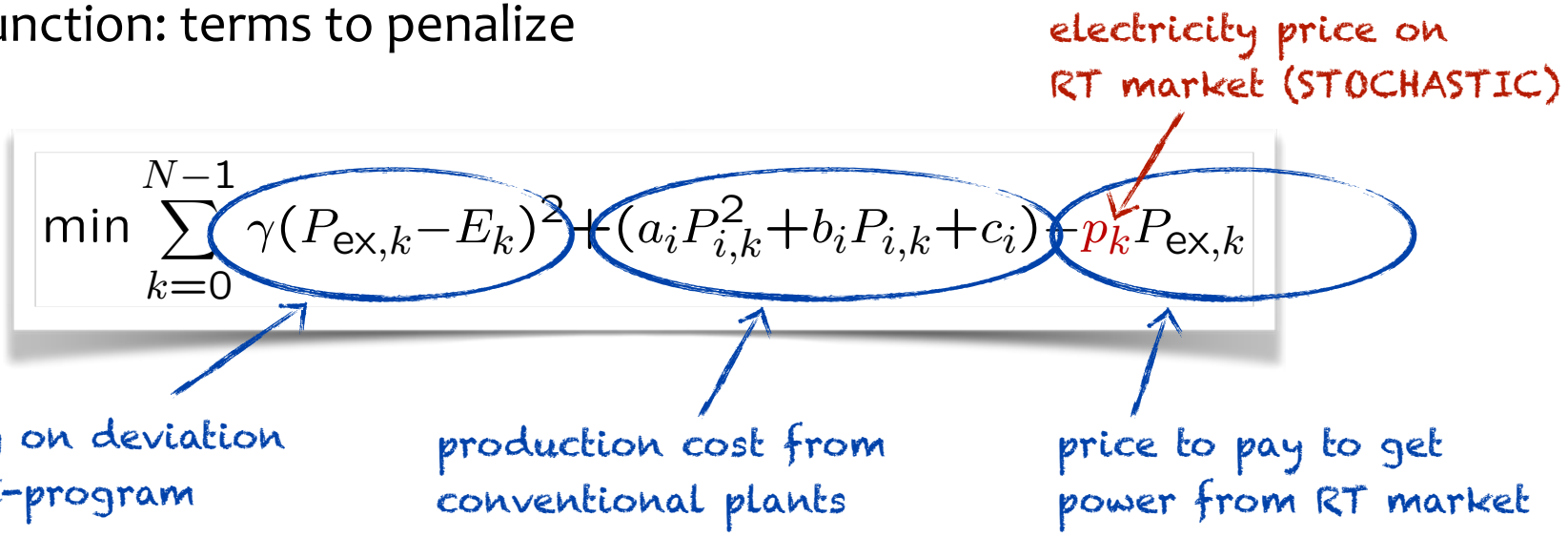
$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \alpha \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_c & -\frac{1}{\alpha_d} & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \alpha - 1 \end{bmatrix} \qquad D = \begin{bmatrix} 0 & 0 & 0 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_c & -\frac{1}{\alpha_d} & 0 \end{bmatrix}$$

©2019 A. Bemporad - "Model Predictive Control"

POWER DISPATCH COST FUNCTION

- Cost function: terms to penalize



$E_k = 0$, $\gamma = 0$ if no E-Program is agreed on the day-ahead market

- The overall linear MPC problem maps into a QP:

$$\min_z \quad \frac{1}{2}z'Hz + \left(F\begin{bmatrix} x_0 \\ r-l \\ E \end{bmatrix} + c\right)z + d$$

$$\text{s.t.} \quad Gz \le W + S\begin{bmatrix} x_0 \\ r-l \\ E \end{bmatrix}$$

x_0 = current state
$r-l$ = predicted renewable power - load
E = E-program

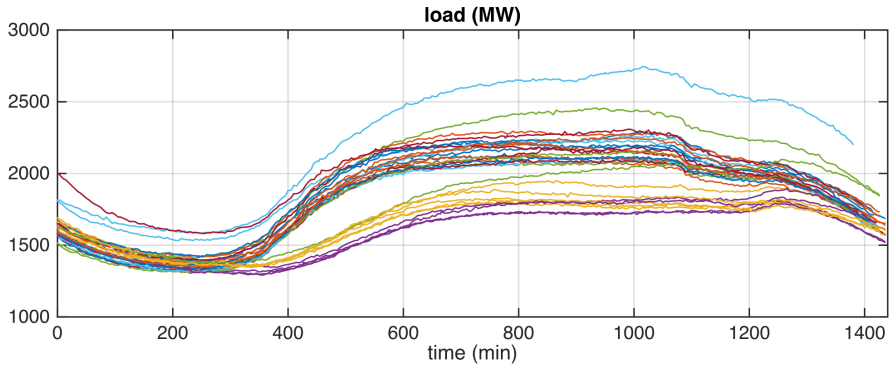
$$z = \left\{\Delta P_{i,k}, u_{c,k}, u_{d,k}\right\}_{k=0}^{N-1}$$

©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Historical data of load (MW)

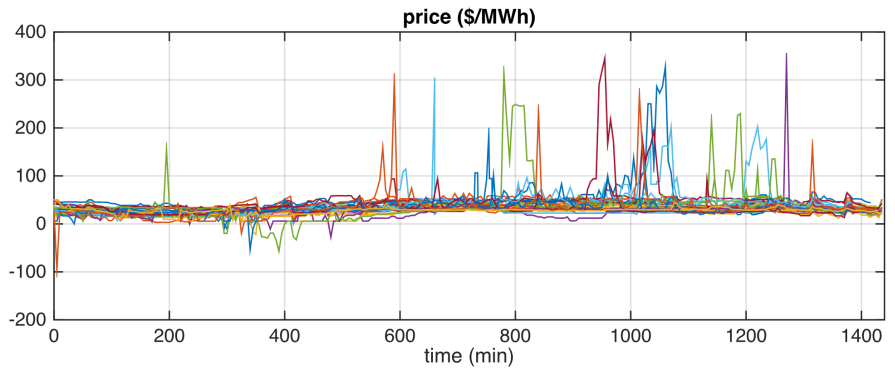
load = 1/3 load of N.Y.C. district
(daily data of 1-31 May 2014,
sampling time = 5 min)



http://www.nyiso.com/public/markets_operations/market_data/load_data/index.jsp

- Historical data of price (MW)

electricity price of N.Y.C. district
(daily data of 1-31 May 2014,
sampling time = 5 min)



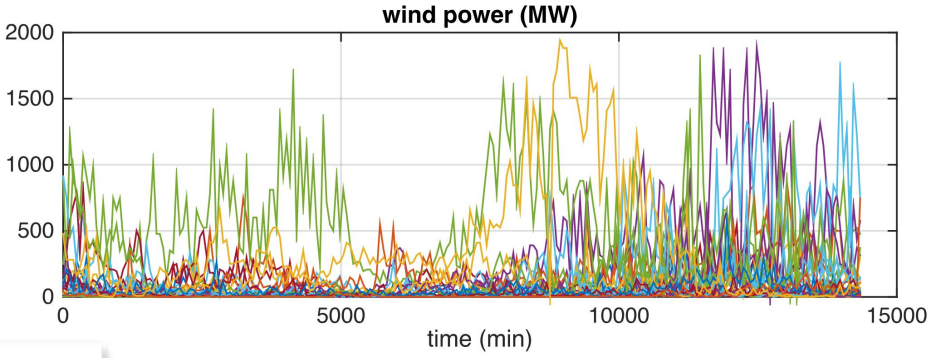
http://www.nyiso.com/public/markets_operations/market_data/pricing_data/index.jsp

©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Historical data of wind speed (m/s)

Station BGNN4 (NY)
(daily data of 1-31 May 2014,
sampling time = 6 min)



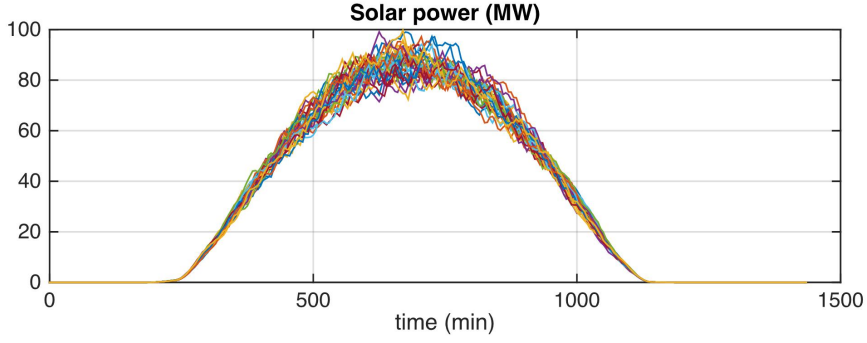
wind power proportional to cubic wind velocity → $P_w = k v_w^3$

http://www.ndbc.noaa.gov/station_history.php?station=bgnn4

- Historical data of solar irradiation (W/m²)

NY Central Park, daily data of
1-31 May 1991-2005, sampling time = 1 h

Data perturbed by noise to mimic
account cloud coefficient (unavailable)

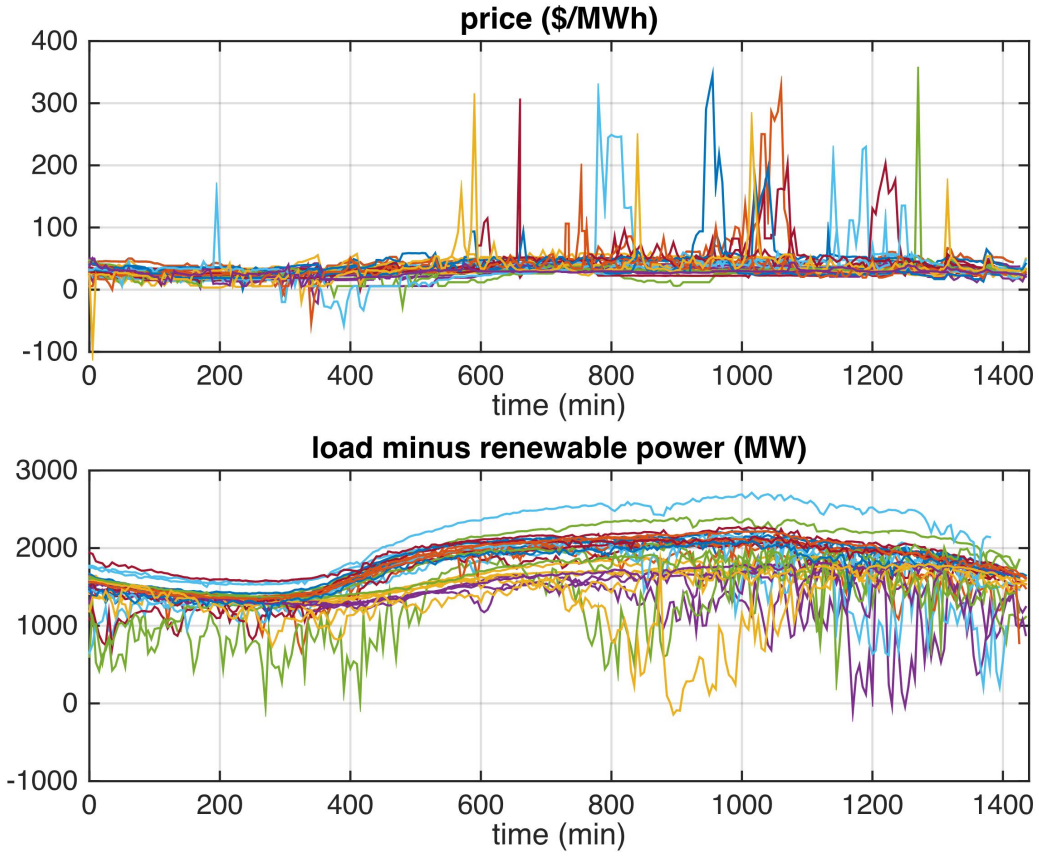


http://en.openei.org/datasets/files/39/pub/725033.tar.gz

©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Historical data of overall uncertainty



©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Data used for scenario generation (31 days):

value at time t+k in scenario #j

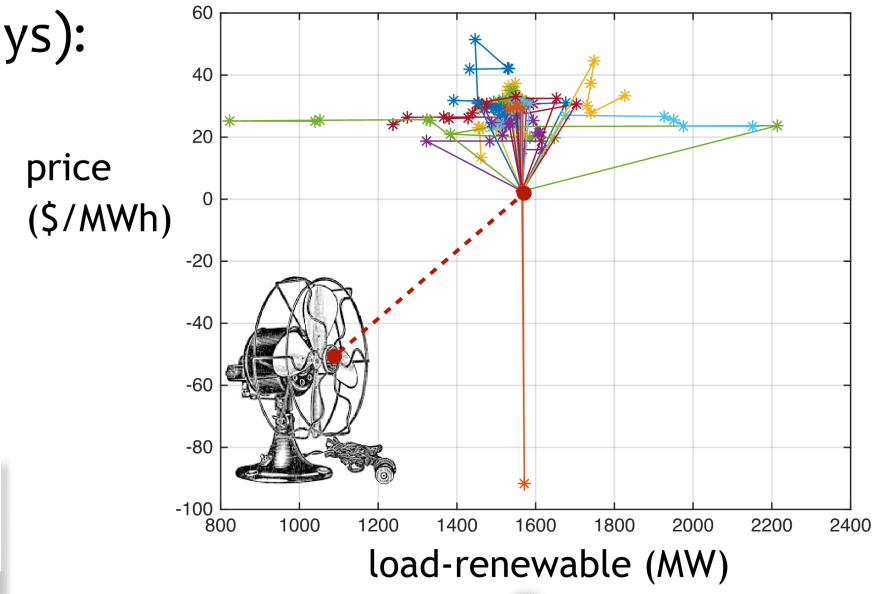
initial value at time t in scenario #j

$$w^j(t+k) = v^j(t+k) - v^j(t) + v(t)$$

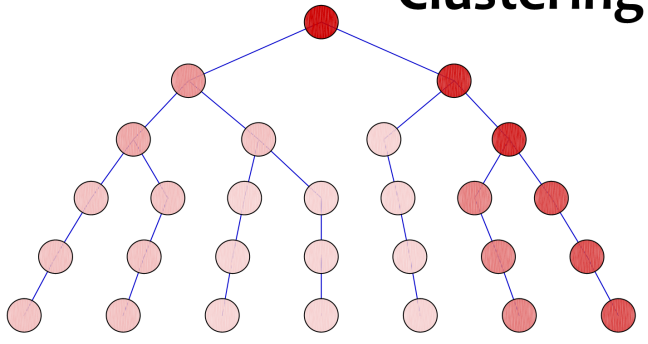
$$w^j(t) = v(t)$$

stochastic vector on scenario #j

actual value at time t



Heuristic Multilevel Clustering



28 nodes

- Tree obtained from 31 scenarios (branching factor M=[2 2 2 1 1])

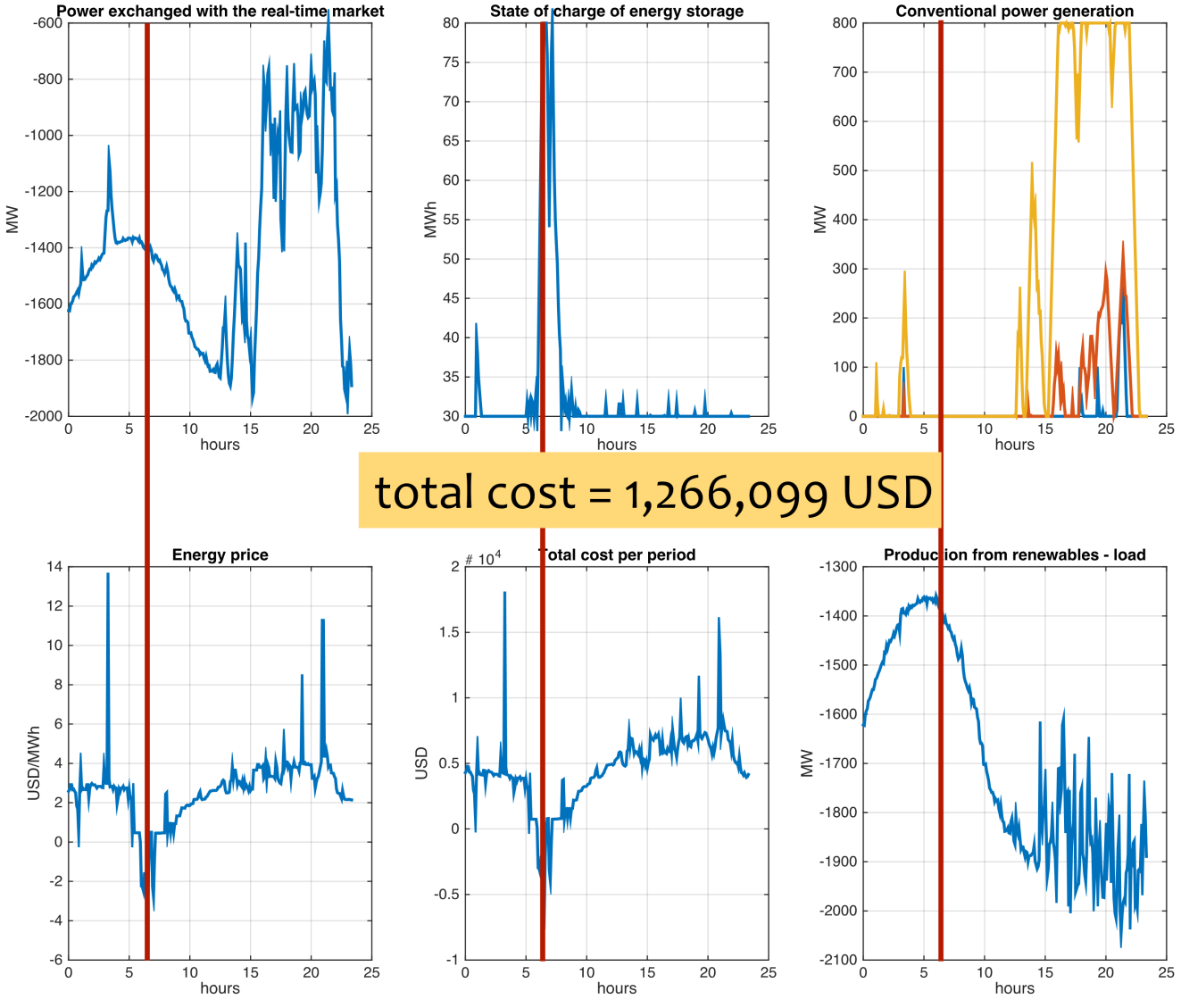
©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- MPC setup:
  - Sampling time: T_s=5 min
  - Prediction horizon: N=6 steps (=1/2 hour ahead)
  - Three controller options:
    - **Stochastic MPC**, with branching factor M (ex: M=[4 3 2 2 1])
    - **Average MPC**, that is deterministic MPC based on the **expected** (price, load-renewable) realization
    - **Prescient MPC**, that is deterministic MPC based on the **exact** future (price, load-renewable) realization

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Simulation results using SMPC, M=[2,2,2,1,1] (1 day, May 26, 2014)



©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Compare simulation results wrt different tree complexity, prescient, and deterministic (1 day, May 26, 2014)

exact knowledge of future uncertainty

stochastic formulation

```
Prescient:    Total cost=   1,247,909 [USD],                          nvar=  30, CPUTIME =  14 [ms]
Stochastic:   Total cost=   1,266,099 [USD], M=[2,2,2,1,1], nvar= 105, CPUTIME =  43 [ms]
Stochastic:   Total cost=   1,266,123 [USD], M=[3,3,1,1,1], nvar= 140, CPUTIME =  50 [ms]
Stochastic:   Total cost=   1,266,214 [USD], M=[2,2,1,1,1], nvar=  95, CPUTIME =  30 [ms]
Stochastic:   Total cost=   1,266,701 [USD], M=[3,1,1,1,1], nvar=  80, CPUTIME =  27 [ms]
Stochastic:   Total cost=   1,267,069 [USD], M=[2,1,1,1,1], nvar=  55, CPUTIME =  22 [ms]
Average:      Total cost=   1,267,113 [USD], M=[1,1,1,1,1]  nvar=  30, CPUTIME =  14 [ms]
Frozen-time:  Total cost=   1,267,401 [USD],                          nvar=  30, CPUTIME =  14 [ms]
```

deterministic: assume future disturbance = average of historical data

deterministic: assume future disturbance = current disturbance

nvar = number of variables in QP problem = 5*(# nodes), **CPUTIME** = time to build tree,build QP matrices, and solve

©2019 A. Bemporad - "Model Predictive Control"

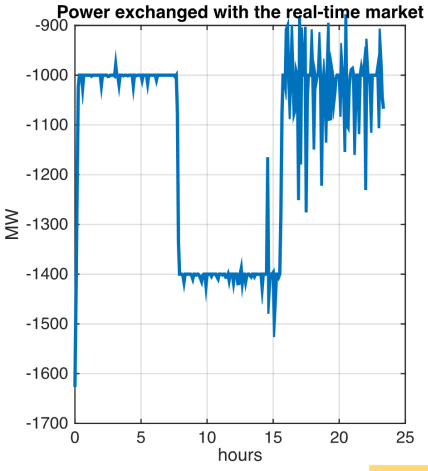
SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

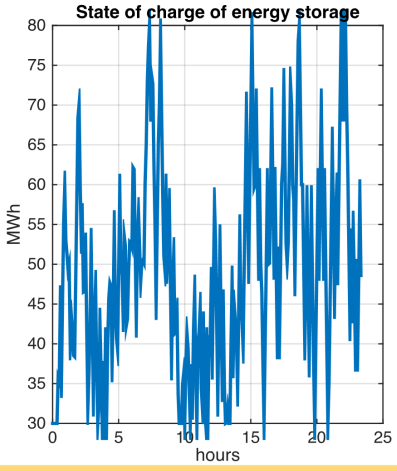
- Tracking an E-Program

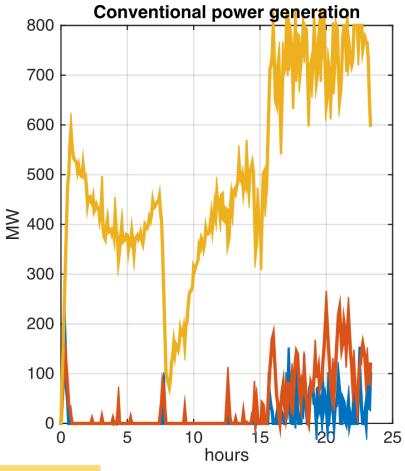
$$\min \sum_{k=0}^{N-1} \gamma(P_{\text{ex},k} - E_k)^2 + (a_i P_{i,k}^2 + b_i P_{i,k} + c_i) - p_k[P_{\text{ex},k} - E_k]$$

$\gamma = 10^3$

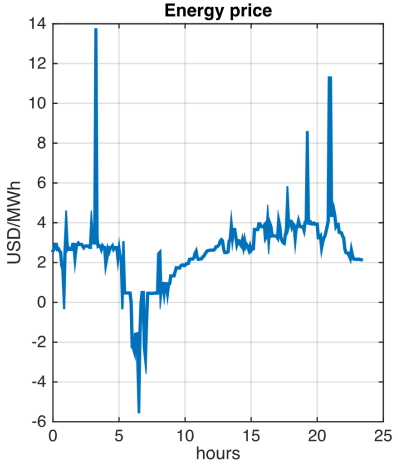
SMPC with branching factor M=[2 2 2 1 1]

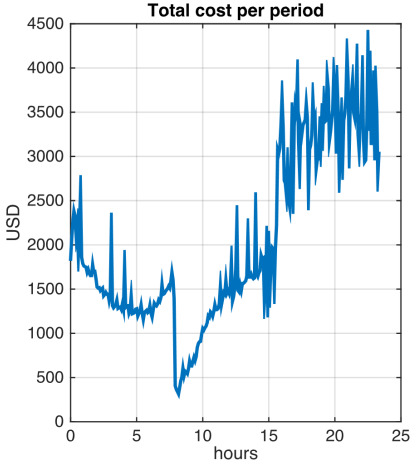


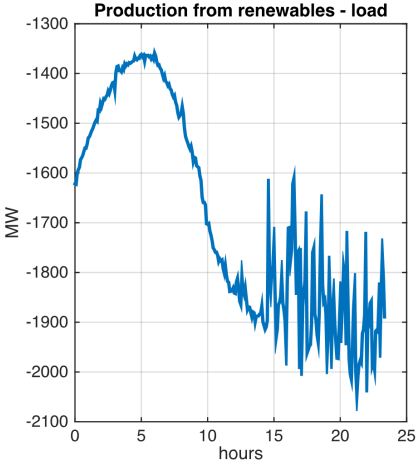




total cost = 574,388 USD







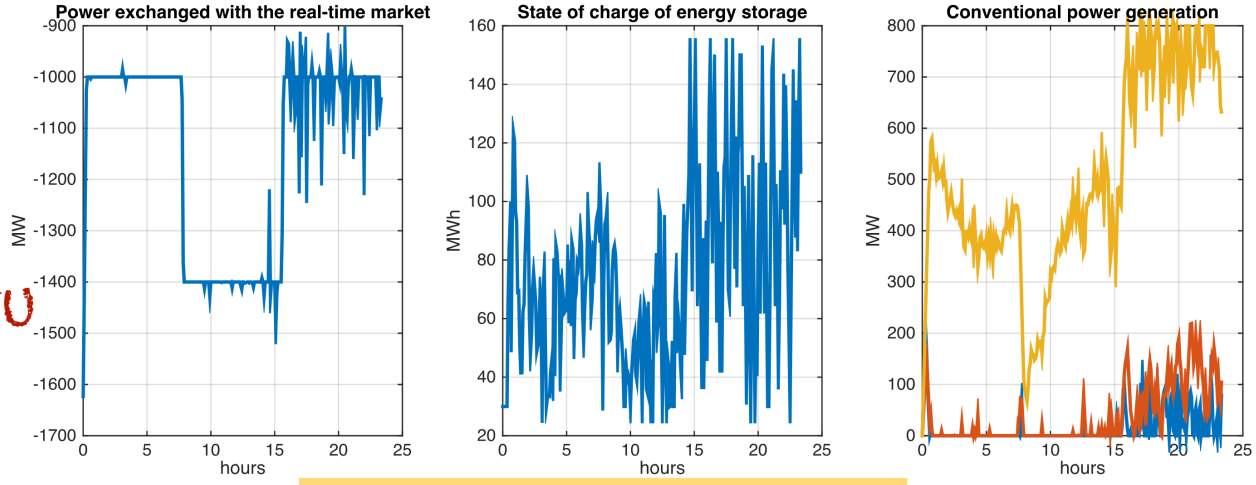
Revenues from day-ahead market are not counted

©2019 A. Bemporad - "Model Predictive Control"

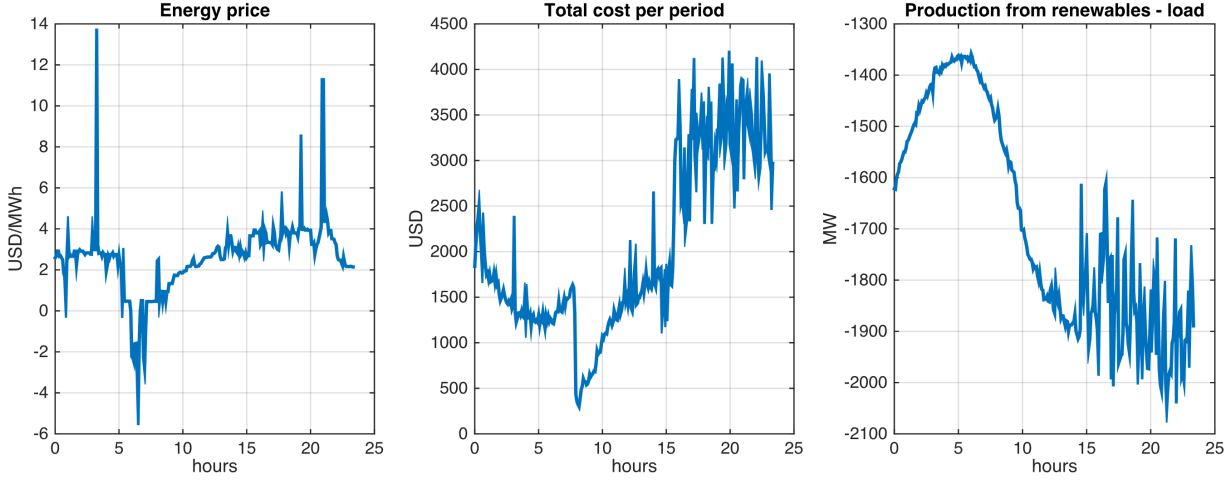
SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Change storage type: $S_{\min} \le S_k \le S_{\max}$ $\Delta_{S,\min} \le S_{k+1} - S_k \le \Delta_{S,\max}$

$S_{\min} = 30$
$S_{\max} = \cancel{80}$ 150 MWh
$\Delta_{S,\min} = \cancel{-10}$
$\Delta_{S,\max} = \cancel{10}$ ±40 MWh/PTU



total cost = 563,283 USD

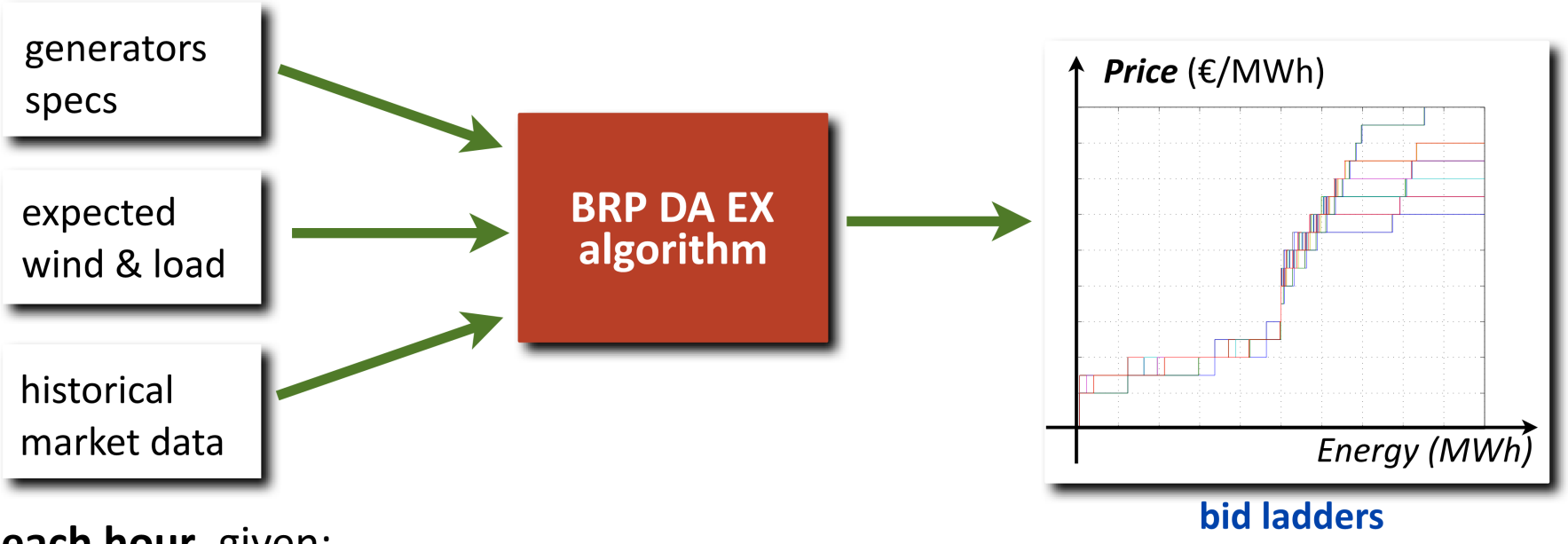


Revenues from day-ahead market are not counted

©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR OPTIMAL BIDDING ON ENERGY MARKETS

BIDDING ON THE DAY-AHEAD (EXCHANGE) MARKET



For each hour, given:

1. generator data (current power profile, min and max power, efficiency, etc.)
2. expected load and wind profiles for each hour of the following day
3. scenarios for AS prices, which are not disclosed yet (**stochastic disturbance**)

compute optimal allocation of energy on the EX market

that **maximizes profits** and **minimize imbalance risks**

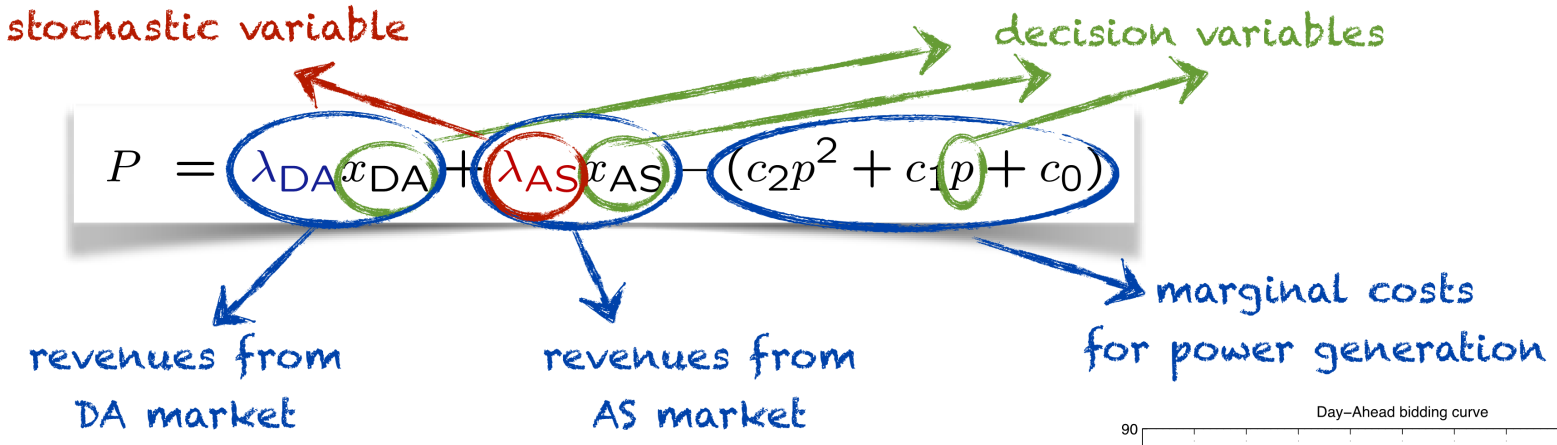
Output: 24 bid curves (one for each hour of the following day)

©2019 A. Bemporad - "Model Predictive Control"

BIDDING ON THE DAY-AHEAD (DA) MARKET

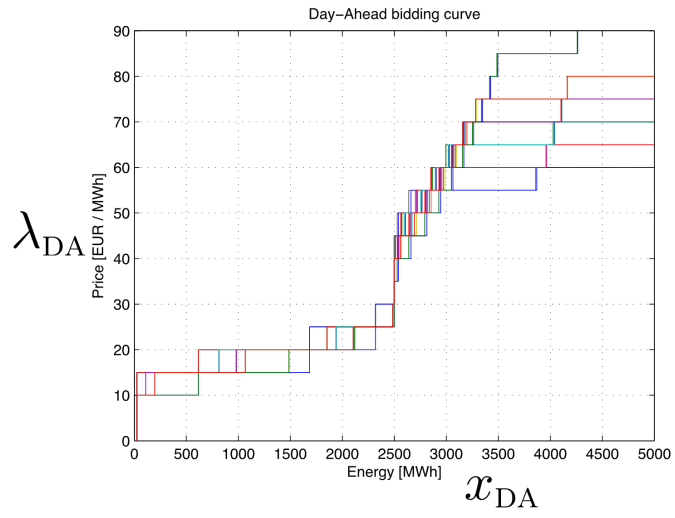
(Puglia, Bernardini, Bemporad, 2011)

- **Price-taking point of view**: the offer has no influence on the cleared price
- **Profit**: for each hour, **for each fixed price** λ_{DA}, the profit is



- **Objective**: minimize CVaR or maximize profit

- **Constraints:**
  - production limits, internal balancing

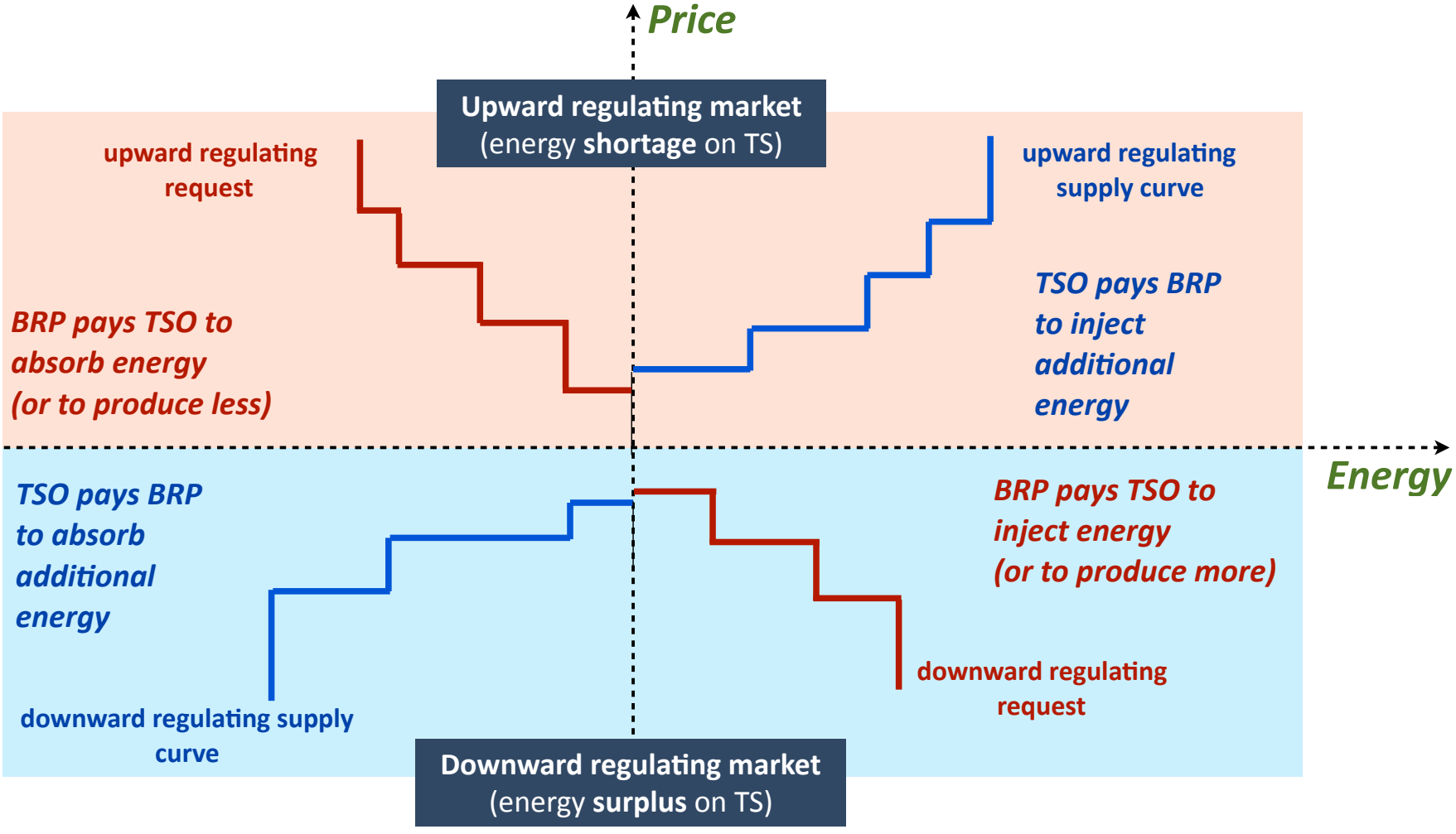


- **Uncertainty:** only AS price, average renewables/load considered (it's 1 day ahead!)

©2019 A. Bemporad - "Model Predictive Control"

BIDDING ON AS MARKET (DOUBLE-SIDED MARKET)

(Puglia, Bemporad, Virag, Jokic, 2011)



- **AS bidding algorithm:** for each (supply, request) price pair, **compute the best energy allocation** (considering stochastic scenarios of available wind power minus load)

©2019 A. Bemporad - "Model Predictive Control"

SIMULATION RESULTS

- Economic performance metrics calculated to test the effectiveness of the E-Price market structure and algorithms
- **Dutch TN**: 7 BRPs consisting of gas/coal plants and wind farms
- Wind power forecast in E-program = actual wind power:

	Production Costs (€)	Imbalance Costs (€)	AS Profit (€)
Single-sided market	1,821,846	36,602	285,294
Double-sided market	1,824,743	16,899	366,895

- Simulation = 1 day
- Sum of all BRPs

- Imprecise wind forecast (actual wind power > expected during DA bidding):

	Production Costs (€)	Imbalance Costs (€)	AS Profit (€)
Single-sided market	1,749,858	105,577	347,151
Double-sided market	1,750,263	98,024	452,491

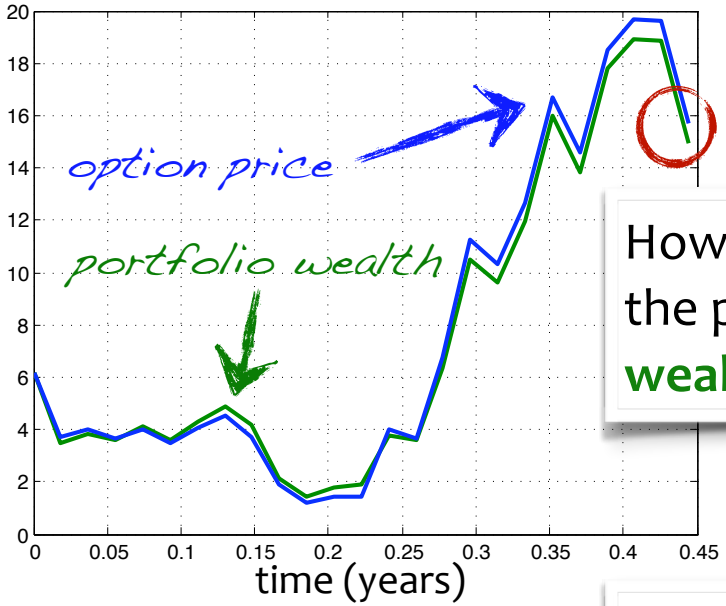
- As creating imbalance is more expensive in the double-sided market, BRPs have higher incentives to efficiently allocate resources

©2019 A. Bemporad - "Model Predictive Control"

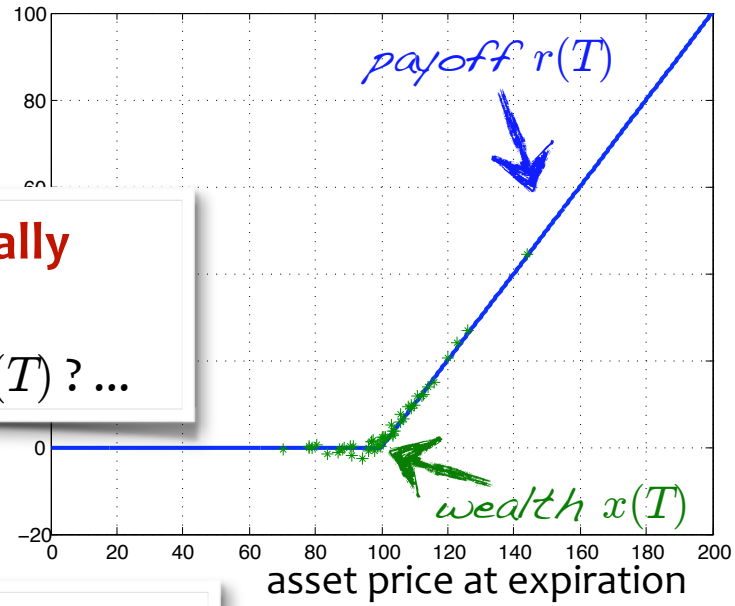
SMPC FOR DYNAMIC HEDGING OF FINANCIAL OPTIONS

DYNAMIC HEDGING PROBLEM FOR FINANCIAL OPTIONS

- The financial institution sells a synthetic option to a customer and gets $x(0)$ (€)
- Such money $x(0)$ is used to create a portfolio $x(t)$ of n underlying assets (e.g., stocks) whose prices at time t are $w_1(t)$, $w_2(t)$, ..., $w_n(t)$
- At the expiration date T, the option is worth the payoff $r(T)$ = wealth (€) to be returned to the customer



How to **adjust dynamically** the portfolio so that **wealth** $x(T)$ = **payoff** $r(T)$? ...



.. for any price realization $w_i(t)$?

©2019 A. Bemporad - "Model Predictive Control"

PORTFOLIO DYNAMICS

- Portfolio wealth at time t:

$$x(t) = u_0(t) + \sum_{i=1}^{n} w_i(t)u_i(t)$$

money in bank account (risk-free asset)

price of asset #i (stochastic process)

number of assets #i

Example: $w_i(t)$ = **log-normal** model (used in Black-Scholes' theory)

$$dw_i = (\mu dt + \sigma dz_i)w_i$$

geometric Brownian motion

- Assets traded at **discrete-time** intervals under the *self-balancing constraint*:

$$x(t+1) = (1+r)x(t) + \sum_{i=0}^{n} b_i(t)u_i(t)$$

r = interest rate

$$b_i(t) \triangleq w_i(t+1) - (1+r)w_i(t)$$

©2019 A. Bemporad - "Model Predictive Control"

PAYOFF FUNCTION

- Let $w(t) = \begin{bmatrix} w_1(t) \\ \vdots \\ w_n(t) \end{bmatrix}$ = vector of assets, and $y(t) = \begin{bmatrix} y_1(t) \\ \vdots \\ y_n(t) \end{bmatrix}$

- Option price $p(t)$: $p(t) = f(w(t), y(t))$

$$p(t) = f(w(0), w(1), \ldots, w(t), w(t))$$ path-dependent option

- Payoff $p(T)$: $p(T) = f(w(0), w(1), \ldots, w(T))$

- Examples (n=1)

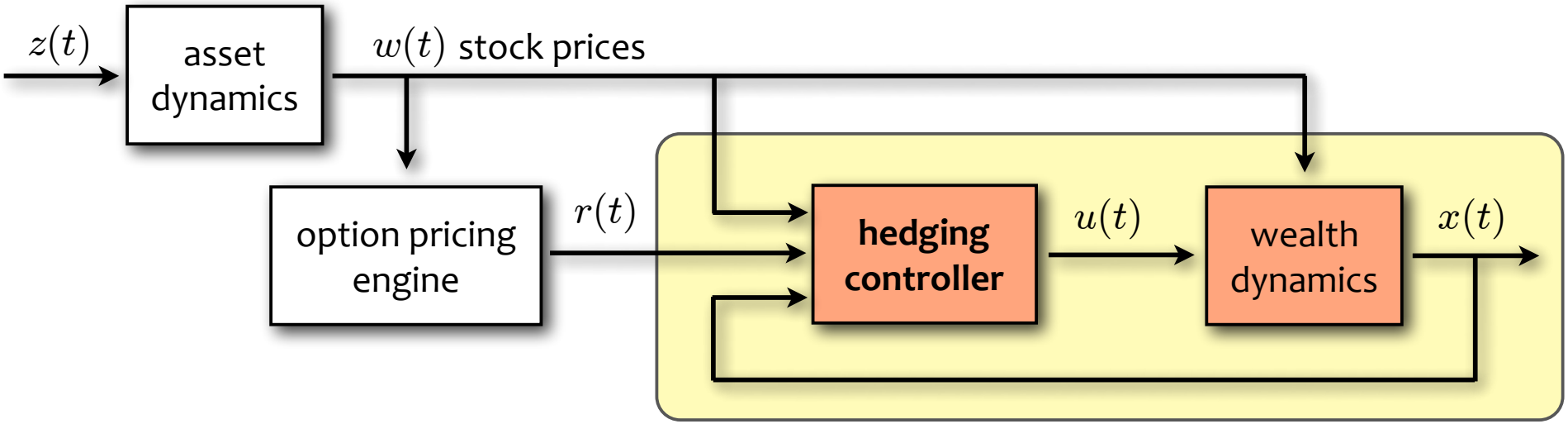
$$p(T) = \max\{w(T) - K, 0\}$$ **European call**

$$p(T) = \max\left\{0, C + \min_{i \in \{1,\ldots,N_{\text{fix}}\}} \frac{w(t_i) - w(t_{i-1})}{w(t_{i-1})}\right\}$$ **Napoleon cliquet** (t_i= fixing dates)

©2019 A. Bemporad - "Model Predictive Control"

OPTION HEDGING = LINEAR STOCHASTIC CONTROL

- Block diagram of dynamic option hedging problem:



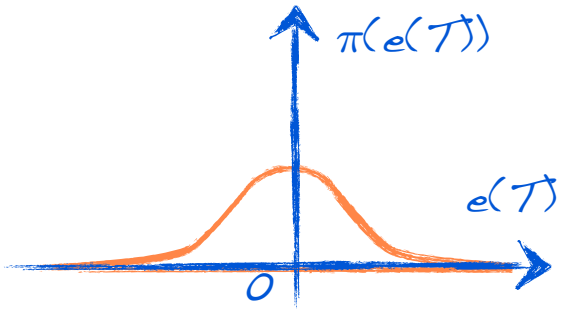
- Reference signal $r(t)$ = price $p(t)$ of hedged option

- **Control objective**: $x(T)$ should be as close as possible to $r(T)$, for any possible realization of the asset prices $w(t)$ *("tracking w/ disturbance rejection")*

©2019 A. Bemporad - "Model Predictive Control"

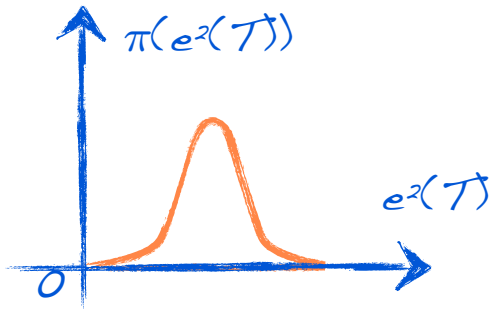
CONTROL OBJECTIVE

- Define **hedging error** $e(t) \triangleq w(t) - p(t)$ ← we want e(T) small !



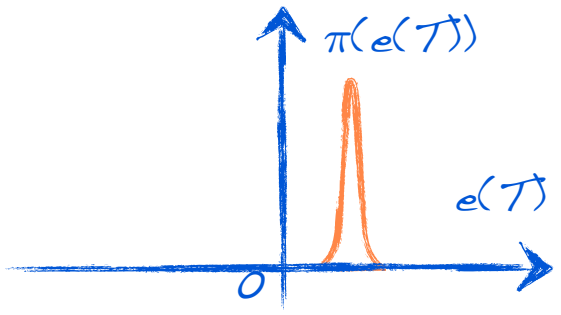
$$\min E\left[e(T)\right]^2$$

Very risky, variance of e(T) may be large !



$$\min E\left[e(T)^2\right]$$

If minimum is 0 then both mean and variance of hedging error are 0



$$\min E\left[(e(T) - E[e(T)])^2\right]$$

How about E[e(T)] ?

In fact: $E\left[e(T)^2\right] = E\left[(e(T) - E[e(T)])^2\right] + \alpha E\left[e(T)\right]^2$ for $\alpha = 1$

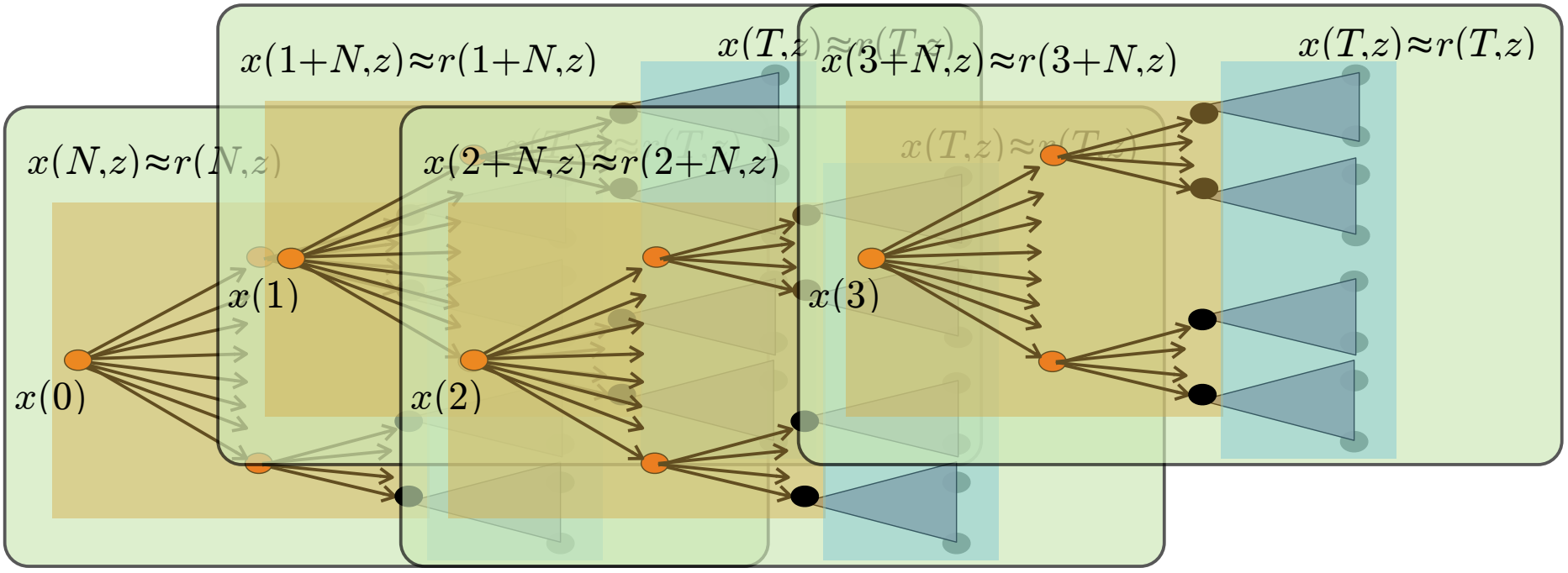
Var[e(T)]

©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR DYNAMIC OPTION HEDGING

- Stochastic finite-horizon optimal control problem:

$$\begin{aligned} \min_{\{u(k,z)\}} \quad & \mathrm{Var}_z\left[x(t+N,z) - r(t+N,z)\right] \\ \text{s.t.} \quad & x(k+1,z) = (1+r)x(k,z) + \sum_{i=0}^{n} b_i(k,z)u_i(k,z), \;\; k = t,\ldots,t+N \\ & x(t,z) = x(t) \end{aligned}$$



SMPC FOR DYNAMIC OPTION HEDGING

- Drawback: the longer the horizon N, the largest the number of scenarios !

- Special case: use $N=1$!

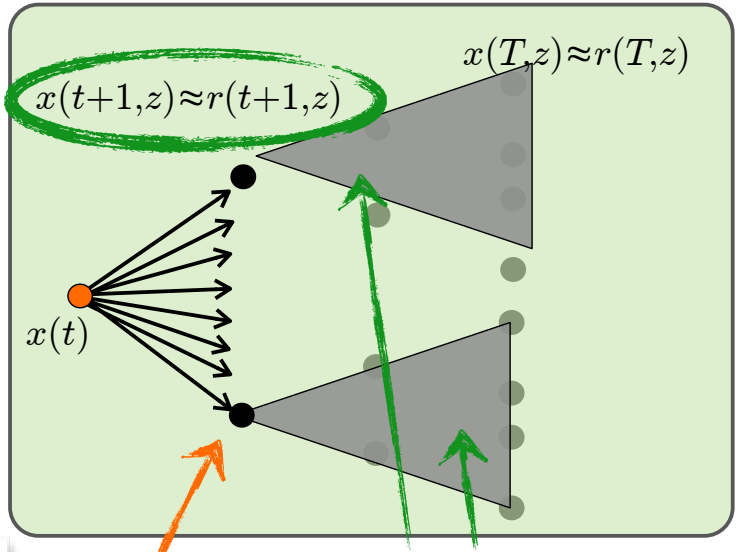
$$\begin{aligned} \min_{u(t)} \quad & \mathrm{Var}_z\left[x(t+1,z) - r(t+1,z)\right] \\ \text{s.t.} \quad & x(t+1,z) = (1+r)x(t) + \sum_{i=0}^{n} b_i(t,z)u_i(t) \end{aligned}$$

minimum variance control !

✓ **Only one** vector $u(t)$ to optimize

✓ No further branching, so we can generate **a lot** of scenarios for z! (example: 1000)

◉ Need to compute target wealth $r(t+1,z)$ for all z



On-line optimization: very simple least squares problem with n variables !

(n = number of traded assets)

SMPC HEDGING ALGORITHM

- Let t=current hedging date, $w(t)$=wealth of portfolio, $x(t)\in\mathbb{R}^n$= asset prices

- Use **Monte Carlo simulation** to generate M scenarios of future asset prices

$$x^1(t+1),\ x^2(t+1),\ \ldots,\ x^M(t+1)$$
$$y^1(t+1),\ y^2(t+1),\ \ldots,\ y^M(t+1)$$

- Use a **pricing engine** to generate the corresponding future option prices

$$p^1(t+1),\ p^2(t+1),\ \ldots,\ p^M(t+1)$$

- **Optimize** *sample variance*, get new asset quantities $u(t)\in\mathbb{R}^n$, rebalance portfolio:

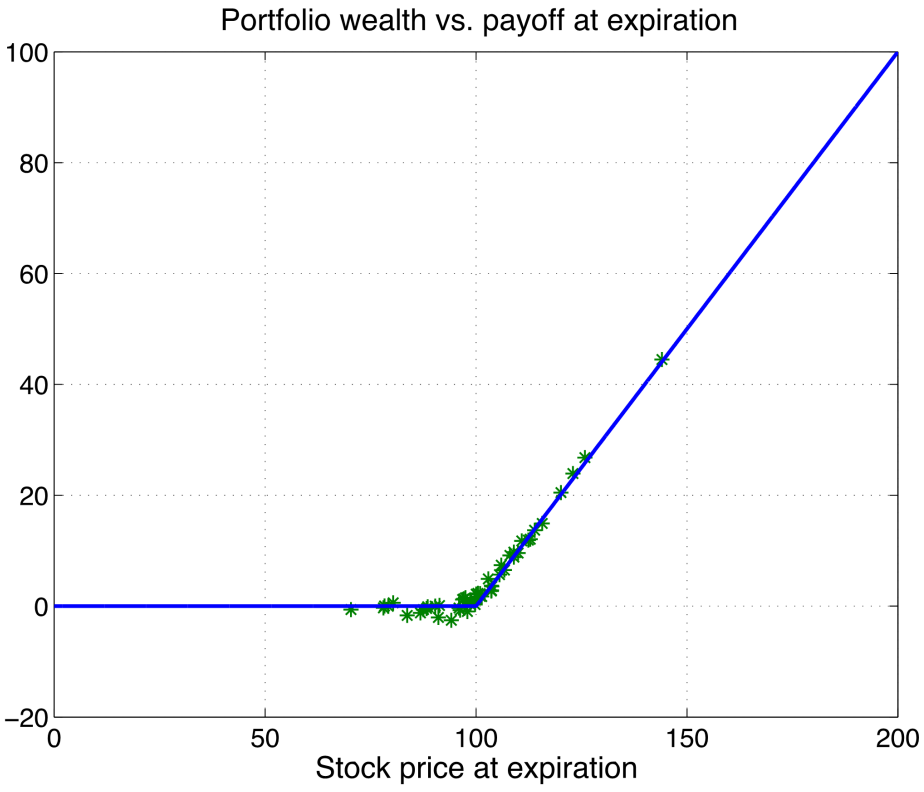
$$\min_{u(t)} \sum_{j=1}^{M} \left(w^j(t+1) - p^j(t+1) - \left(\frac{1}{M} \sum_{i=1}^{M} w^j(t+1) - p^j(t+1) \right) \right)^2$$

Least squares problem (n = number of traded assets)

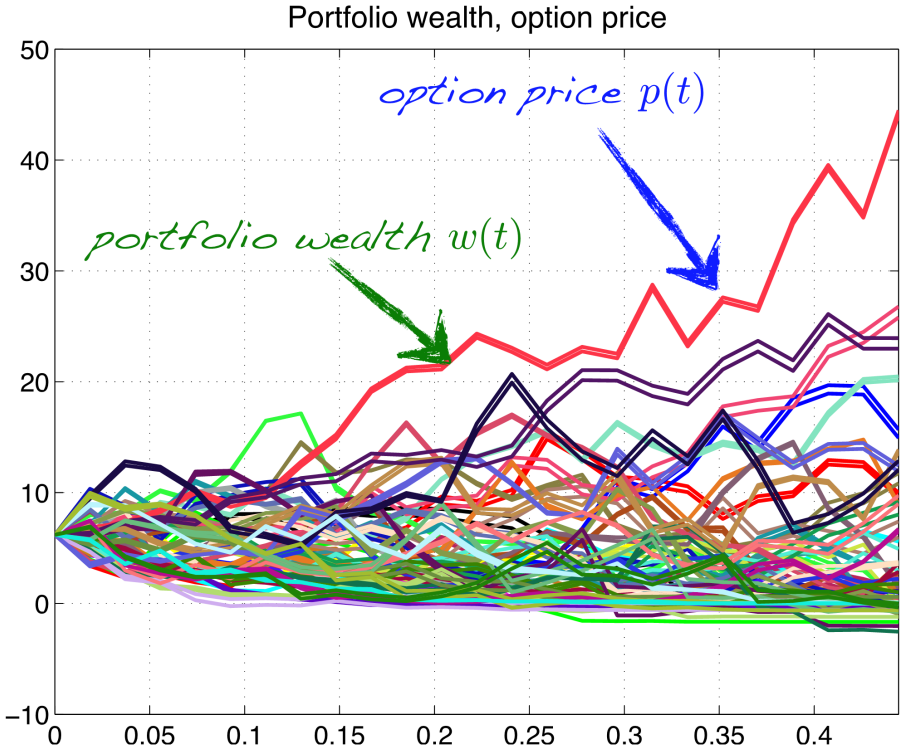
With transaction costs, the problem can be cast to a *quadratic program*

(Bemporad, Puglia, Gabbriellini, 2011)

EXAMPLE: BS MODEL, EUROPEAN CALL



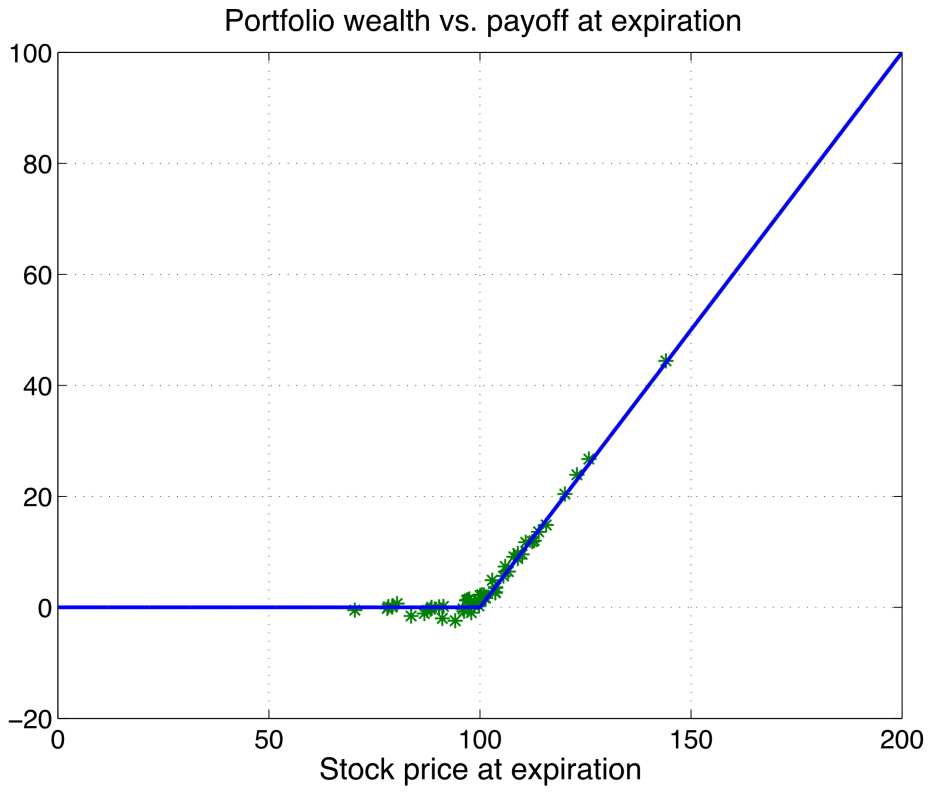
- Black-Scholes model (=log-normal)
- volatility$=0.2$, risk-free$=0.04$
- $T=24$ weeks ($\Delta t=1$ week)
- 50 simulations
- $M=100$ scenarios
- Pricing method: Monte Carlo sim.
- **SMPC**



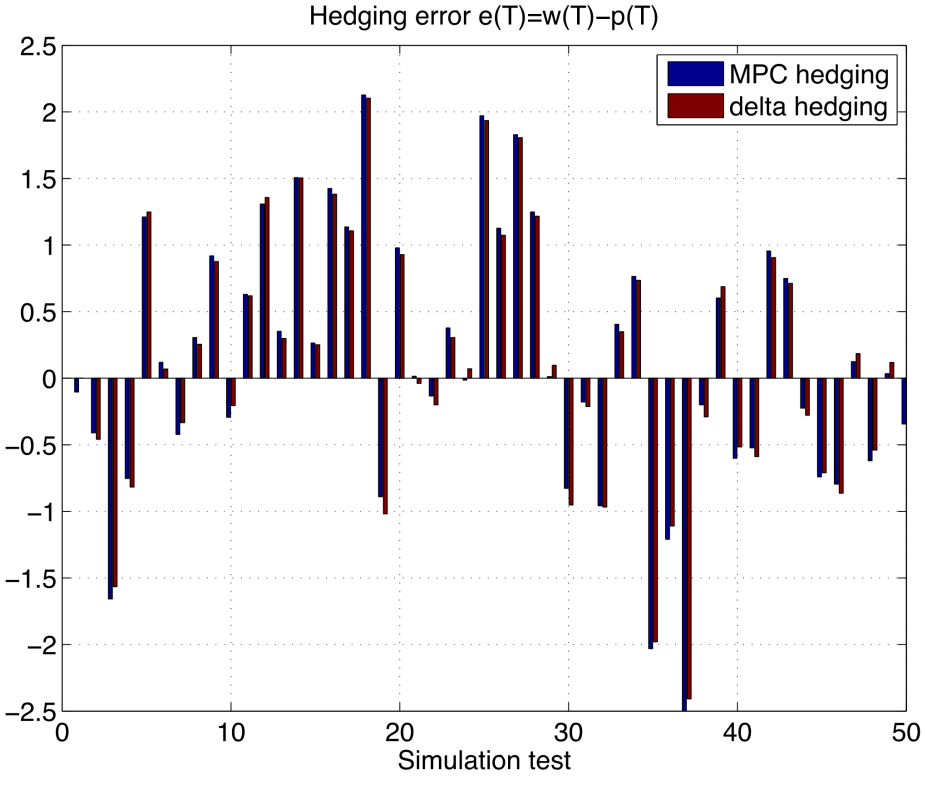
- CPU time $=$ 7.52 ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

©2019 A. Bemporad - "Model Predictive Control"

EXAMPLE: BS MODEL, EUROPEAN CALL



- Black-Scholes model (=log-normal)
- volatility$=0.2$, risk-free$=0.04$
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- **Delta-hedging**



- CPU time $=0.2$ ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

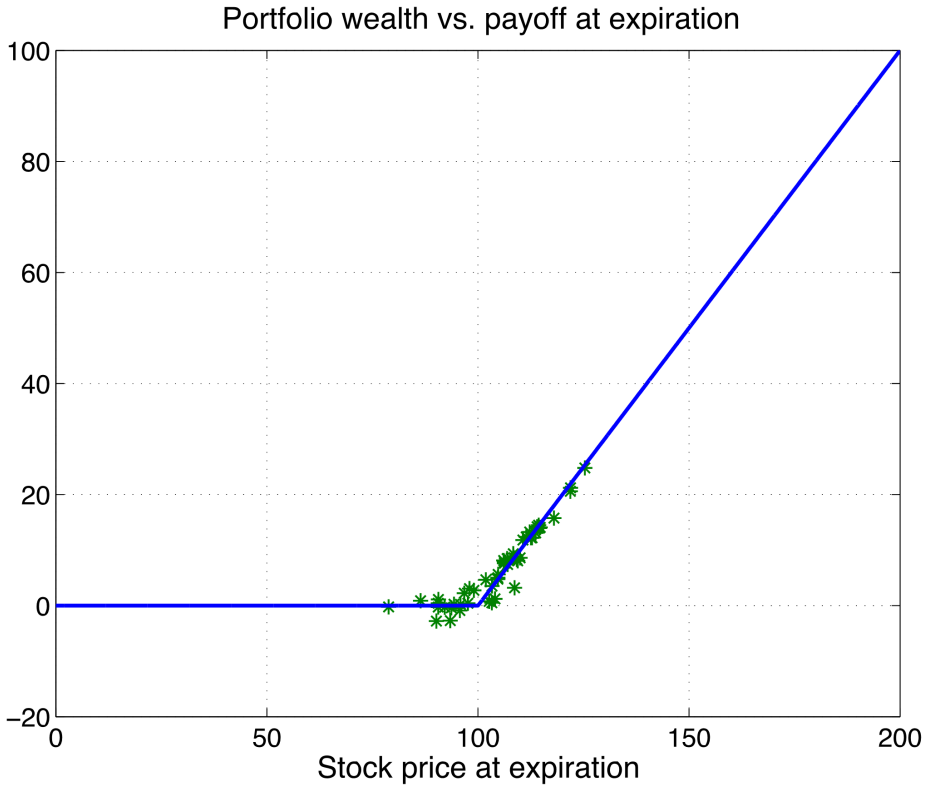
SMPC and delta-hedging are almost indistinguishable

©2019 A. Bemporad - "Model Predictive Control"

EXAMPLE: BS MODEL, EUROPEAN CALL

TIME	x(t)	w(t)	p(t)	u0(t)	x(t)*u1(t)	x(t)*dp/dx(t)
t=0.0000:	S=100.000,	P= 6.196,	O= 6.196,	P(B)=-52.15	P(S)= 58.348	(BS delta= 57.926)
t=0.0185:	S=101.367,	P= 6.955,	O= 6.865,	P(B)=-56.091,	P(S)= 63.046	(BS delta= 62.628)
t=0.0370:	S= 96.897,	P= 4.134,	O= 4.261,	P(B)=-42.629,	P(S)= 46.762	(BS delta= 46.307)
t=0.0556:	S= 94.582,	P= 2.985,	O= 3.108,	P(B)=-35.080,	P(S)= 38.065	(BS delta= 37.607)
t=0.0741:	S= 93.057,	P= 2.345,	O= 2.415,	P(B)=-29.877,	P(S)= 32.222	(BS delta= 31.771)
t=0.0926:	S= 93.371,	P= 2.431,	O= 2.395,	P(B)=-30.200,	P(S)= 32.632	(BS delta= 32.165)
t=0.1111:	S= 94.295,	P= 2.732,	O= 2.591,	P(B)=-32.518,	P(S)= 35.250	(BS delta= 34.760)
t=0.1296:	S= 88.192,	P= 0.426,	O= 0.859,	P(B)=-14.985,	P(S)= 15.411	(BS delta= 15.053)
t=0.1481:	S= 90.411,	P= 0.803,	O= 1.199,	P(B)=-19.776,	P(S)= 20.579	(BS delta= 20.147)
t=0.1667:	S= 88.586,	P= 0.373,	O= 0.754,	P(B)=-14.236,	P(S)= 14.609	(BS delta= 14.234)
t=0.1852:	S= 87.683,	P= 0.214,	O= 0.544,	P(B)=-11.312,	P(S)= 11.526	(BS delta= 11.186)
t=0.2037:	S= 90.998,	P= 0.641,	O= 1.000,	P(B)=-18.744,	P(S)= 19.385	(BS delta= 18.910)
t=0.2222:	S= 94.742,	P= 1.425,	O= 1.867,	P(B)=-30.734,	P(S)= 32.158	(BS delta= 31.555)
t=0.2407:	S= 99.890,	P= 3.149,	O= 3.945,	P(B)=-52.320,	P(S)= 55.469	(BS delta= 54.841)
t=0.2593:	S=102.720,	P= 4.682,	O= 5.466,	P(B)=-64.736,	P(S)= 69.418	(BS delta= 68.857)
t=0.2778:	S= 99.723,	P= 2.609,	O= 3.439,	P(B)=-51.468,	P(S)= 54.077	(BS delta= 53.379)
t=0.2963:	S= 99.591,	P= 2.499,	O= 3.147,	P(B)=-50.513,	P(S)= 53.012	(BS delta= 52.268)
t=0.3148:	S= 98.178,	P= 1.709,	O= 2.233,	P(B)=-42.460,	P(S)= 44.169	(BS delta= 43.336)
t=0.3333:	S=100.471,	P= 2.709,	O= 3.142,	P(B)=-55.135,	P(S)= 57.845	(BS delta= 57.034)
t=0.3519:	S=102.804,	P= 4.012,	O= 4.363,	P(B)=-69.359,	P(S)= 73.371	(BS delta= 72.719)
t=0.3704:	S= 97.457,	P= 0.144,	O= 1.202,	P(B)=-34.892,	P(S)= 35.037	(BS delta= 33.884)
t=0.3889:	S= 97.789,	P= 0.238,	O= 1.030,	P(B)=-34.692,	P(S)= 34.930	(BS delta= 33.564)
t=0.4074:	S= 98.881,	P= 0.602,	O= 1.089,	P(B)=-41.289,	P(S)= 41.891	(BS delta= 40.275)
t=0.4259:	S= 97.699,	P= 0.071,	O= 0.308,	P(B)=-22.850,	P(S)= 22.921	(BS delta= 20.300)
t=0.4444:	S= 96.002,	P= -0.344,	O= 0.000,	P(B)= -0.344,	P(S)= 0.000	(BS delta= 0.000)

EXAMPLE: HESTON MODEL, EUROPEAN CALL



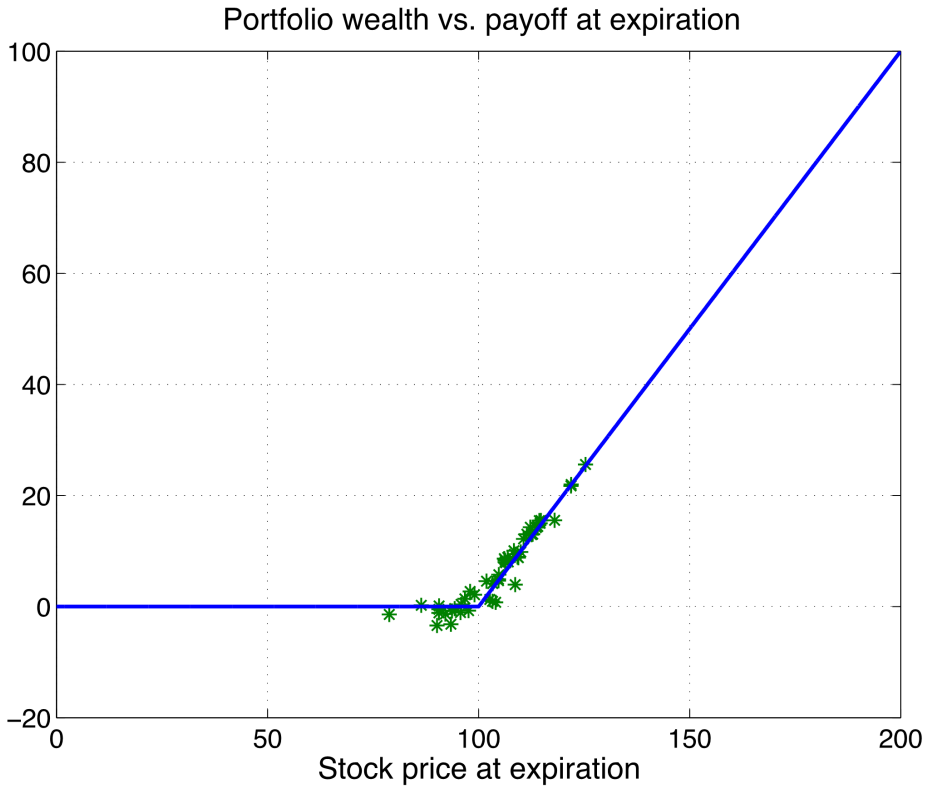
- **Heston's model**
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free$=0.04$
- Pricing method: Monte Carlo sim.
- **SMPC**

- CPU time $= 85.5$ ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

Heston's model

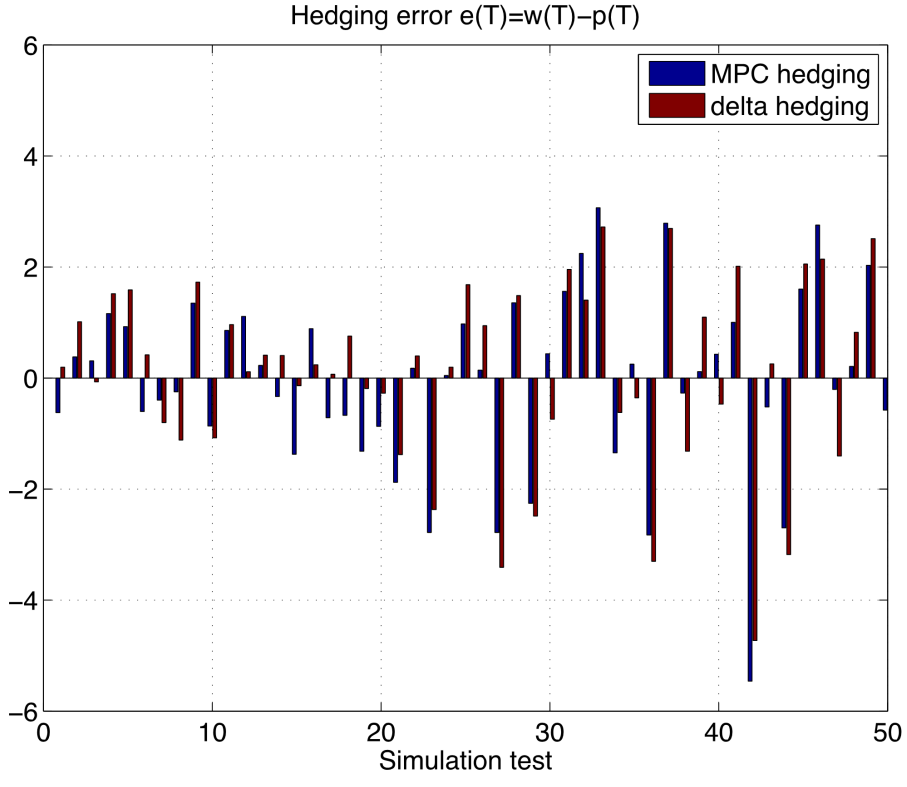
$$\begin{aligned} dx_i(\tau) &= (\mu_i^x d\tau + \sqrt{y_i(\tau)} dz_i^x) x_i(\tau) \\ dy_i(\tau) &= \theta_i(k_i - y_i(\tau))d\tau + \omega_i \sqrt{y_i(\tau)} dz_i^y \end{aligned}$$

EXAMPLE: HESTON MODEL, EUROPEAN CALL



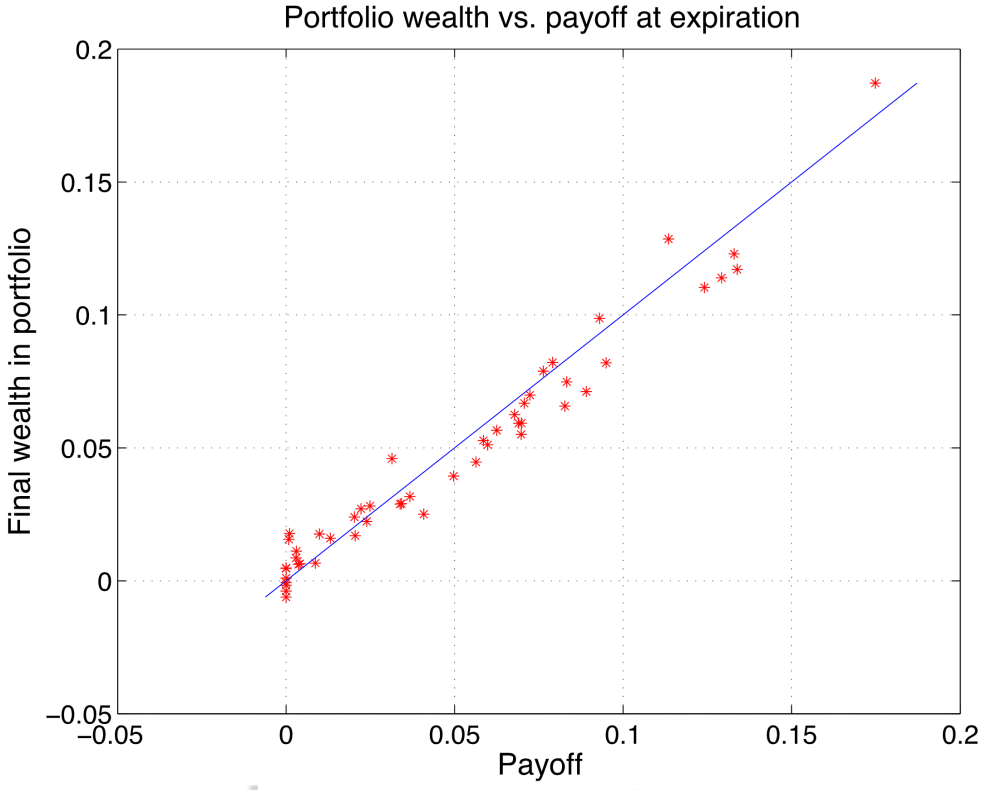
- CPU time $=$ 1.85 ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

- Heston's model
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free$=0.04$
- **Delta hedging**



©2019 A. Bemporad - "Model Predictive Control"

EXAMPLE: BS MODEL, NAPOLEON CLIQUET



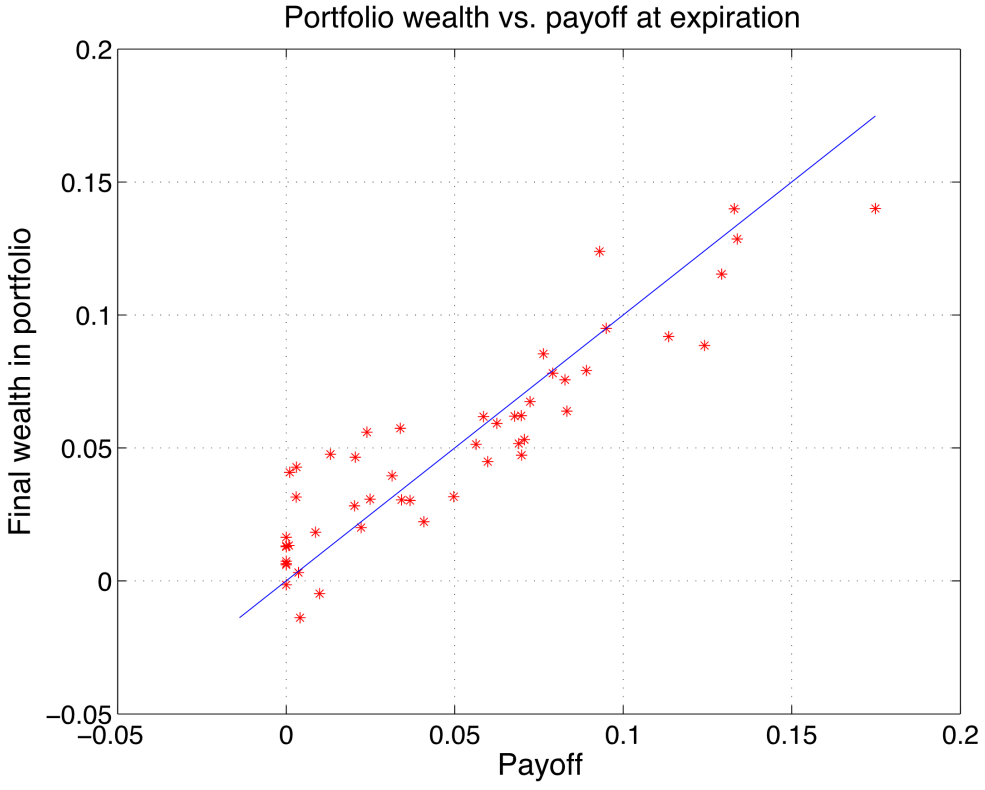
- Black-Scholes model (=log-normal)
- volatility$=0.2$
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free$=0.04$
- Pricing method: Monte Carlo sim.
- **SMPC: only trade underlying stock**

$$p(T) = \max\left\{0, C + \min_{i\in\{1,\dots,N_{\text{fix}}\}} \frac{x(t_i) - x(t_{i-1})}{x(t_{i-1})}\right\}$$

t_i=0,8,16,24 weeks

- CPU time $=$ 1400 ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

EXAMPLE: BS MODEL, NAPOLEON CLIQUET



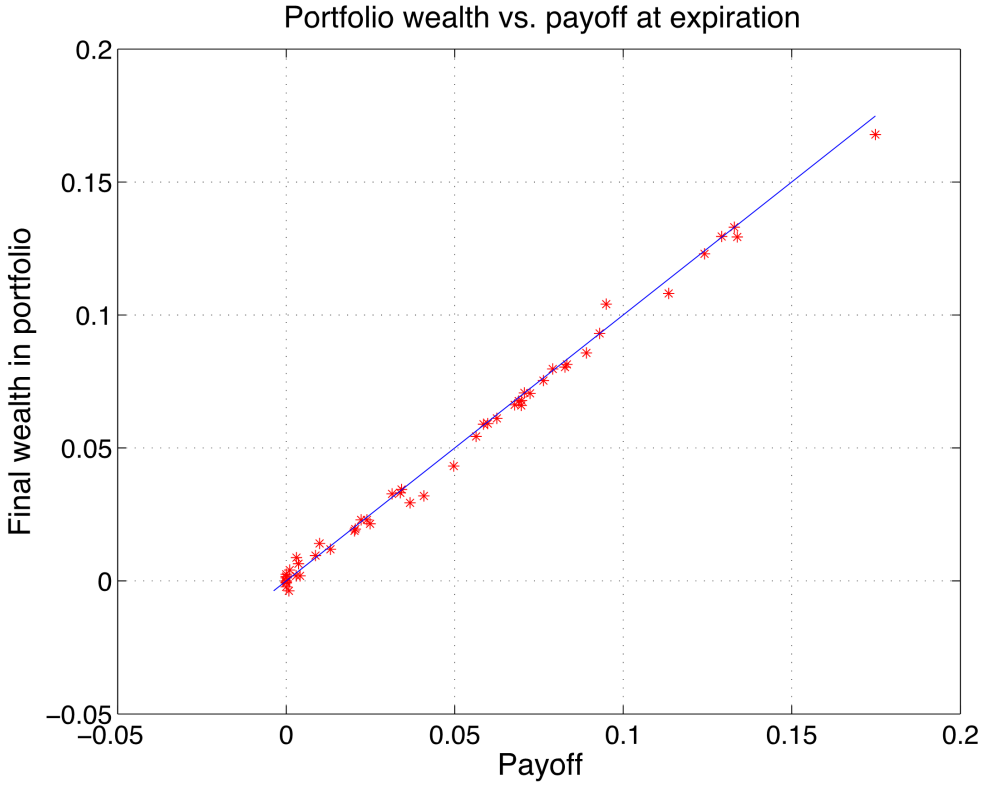
- Black-Scholes model (=log-normal)
- volatility$=0.2$
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free$=0.04$
- **Delta hedging, only trade underlying stock**

$$p(T) = \max\left\{0, C + \min_{i\in\{1,\dots,N_{\text{fix}}\}} \frac{x(t_i) - x(t_{i-1})}{x(t_{i-1})}\right\}$$

t_i=0,8,16,24 weeks

- CPU time $=$ 2.41 ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

EXAMPLE: BS MODEL, NAPOLEON CLIQUET



- Black-Scholes model (=log-normal)
- volatility$=0.2$
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free$=0.04$
- Pricing method: Monte Carlo sim.
- **SMPC: Trade underlying stock & European call with maturity** $t+T$

$$p(T) = \max\left\{0, C + \min_{i\in\{1,\dots,N_{\text{fix}}\}} \frac{x(t_i) - x(t_{i-1})}{x(t_{i-1})}\right\}$$

t_i=0,8,16,24 weeks

- CPU time $=$ 1625 ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

APPROXIMATE OPTION PRICING

- **Bottleneck** of the approach for exotic options: price M future option values

$$p^1(t+1),\ p^2(t+1),\ \ldots,\ p^M(t+1)$$

- **Monte Carlo pricing** can be time consuming: say L scenarios to evaluate a single option value ⇒ need to simulate ML paths to build optimization problem

  (e.g.: M=100, L=10000, ML=10^6)

- Use off-line **function approximation** techniques to estimate $p(t)$ as a function of current asset parameters and other option-related parameters

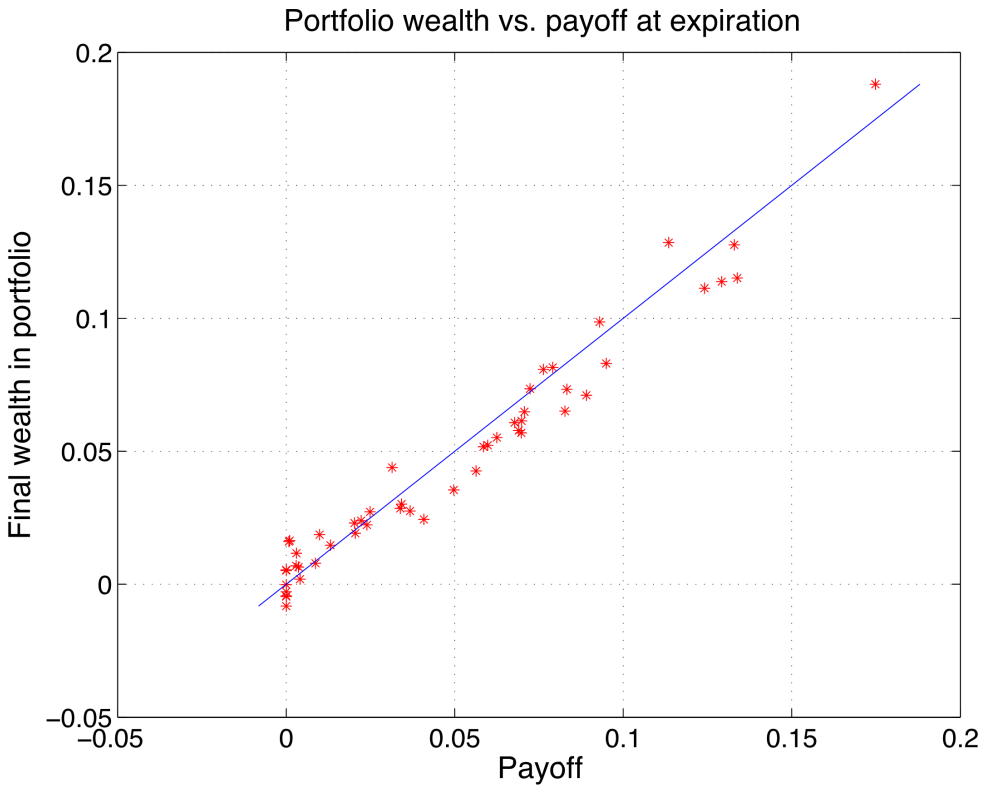
  Example: Napoleon cliquet, Heston model

$$p(t) = f(x(t), \sigma(t), x(t_1), \ldots, x(t_{N_{\text{fix}}}))$$

- Most suitable method for estimating pricing function f:
  least-squares Monte Carlo approach based on polynomial approximations

(Longstaff, Schwartz, 2001)

EXAMPLE: BS MODEL, NAPOLEON CLIQUET



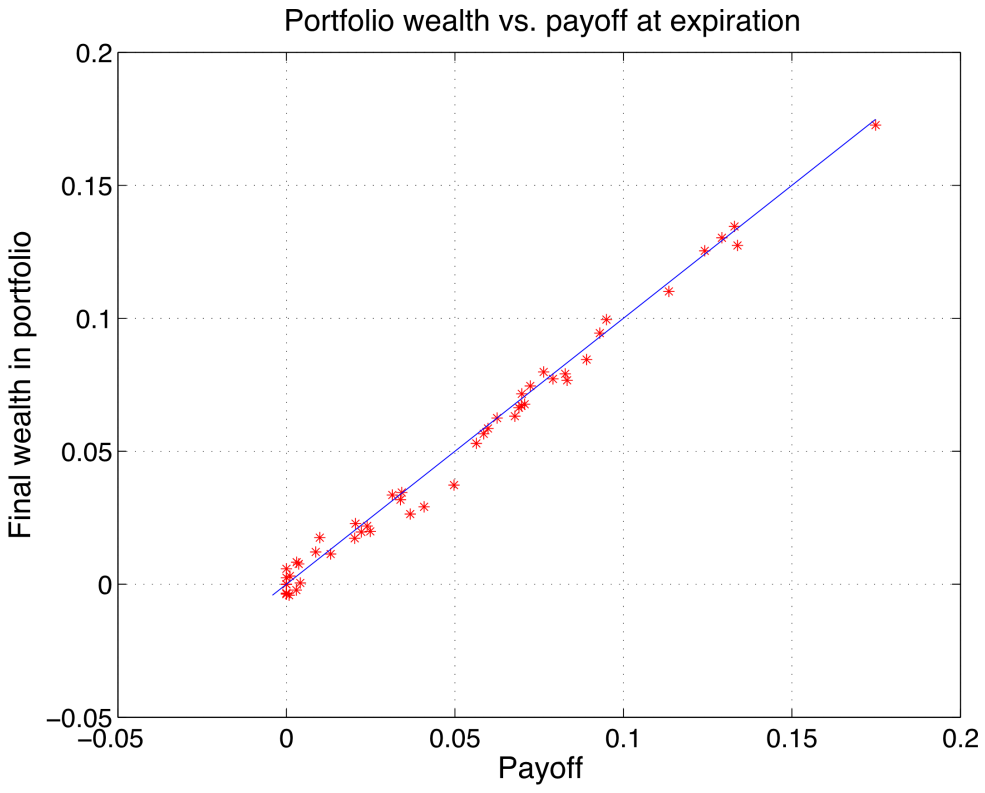
- Black-Scholes model (=log-normal)
- volatility$=0.2$, risk-free$=0.04$
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- Pricing method: LS approximation
- **SMPC: only trade underlying stock**

- ~~CPU time $= 1400$ ms per SMPC step~~
  ~~(Matlab R2009 on 1.86GHz Intel Core 2 Duo)~~

- CPU time $= 50.5$ ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)
- CPU time $= 76.7$ s to compute LS approximation (off-line)

Hedging quality is very similar !

EXAMPLE: BS MODEL, NAPOLEON CLIQUET



- Black-Scholes model (=log-normal)
- volatility$=0.2$, risk-free$=0.04$
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- Pricing method: LS approximation
- **SMPC: Trade underlying stock & European call with maturity** $t+T$

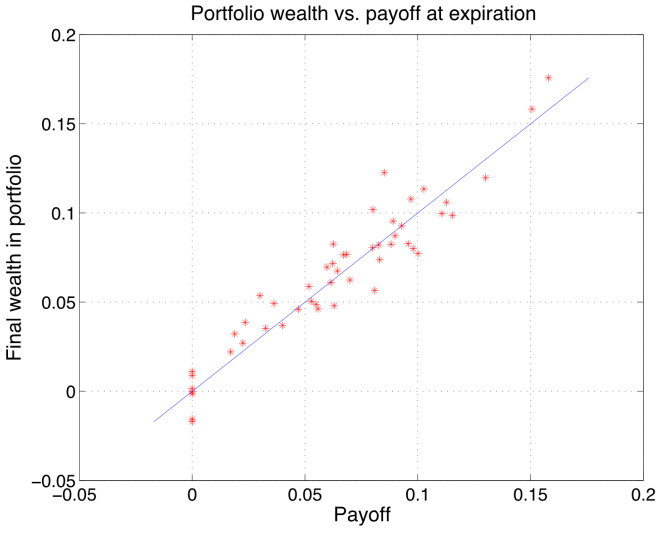
- ~~CPU time $= 1625$ ms per SMPC step~~
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

- CPU time $= 59.2$ ms per SMPC step
  (Matlab R2009 on 1.86GHz Intel Core 2 Duo)

- CPU time $= 76.7$ s to compute LS approximation (off-line)

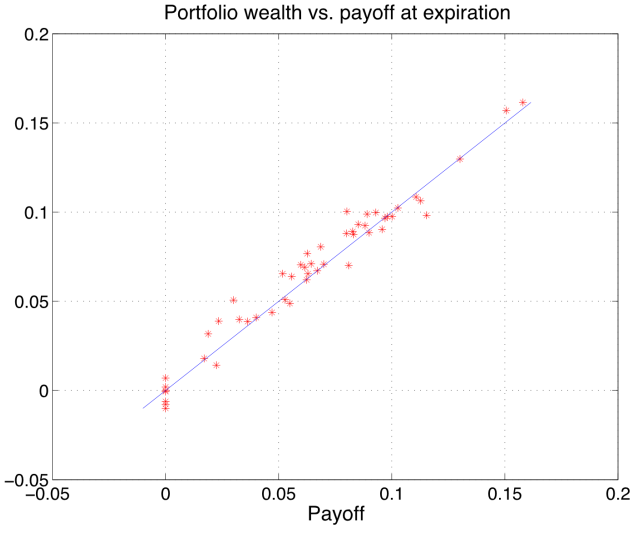
Hedging quality is very similar !

EXAMPLE: HESTON MODEL, NAPOLEON CLIQUET



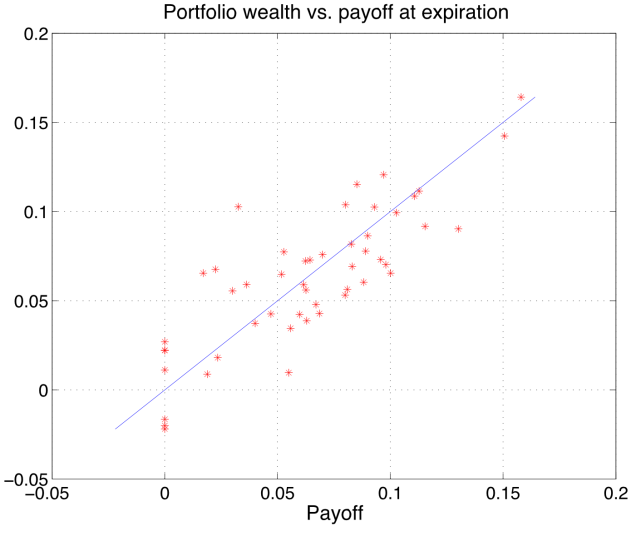
SMPC: only trade underlying stock

CPU time $=$ 220 ms per SMPC step



SMPC: Trade underlying stock & European call with maturity $t+T$

CPU time $=$ 277 ms per SMPC step



Delta hedging only trade underlying stock

CPU time $=$ 156 ms per SMPC step

CPU time $=$ 156 s to compute LS approximation (off-line)

SMPC FOR AUTOMOTIVE CONTROL

SMPC FOR HYBRID ELECTRIC VEHICLES (HEVS)

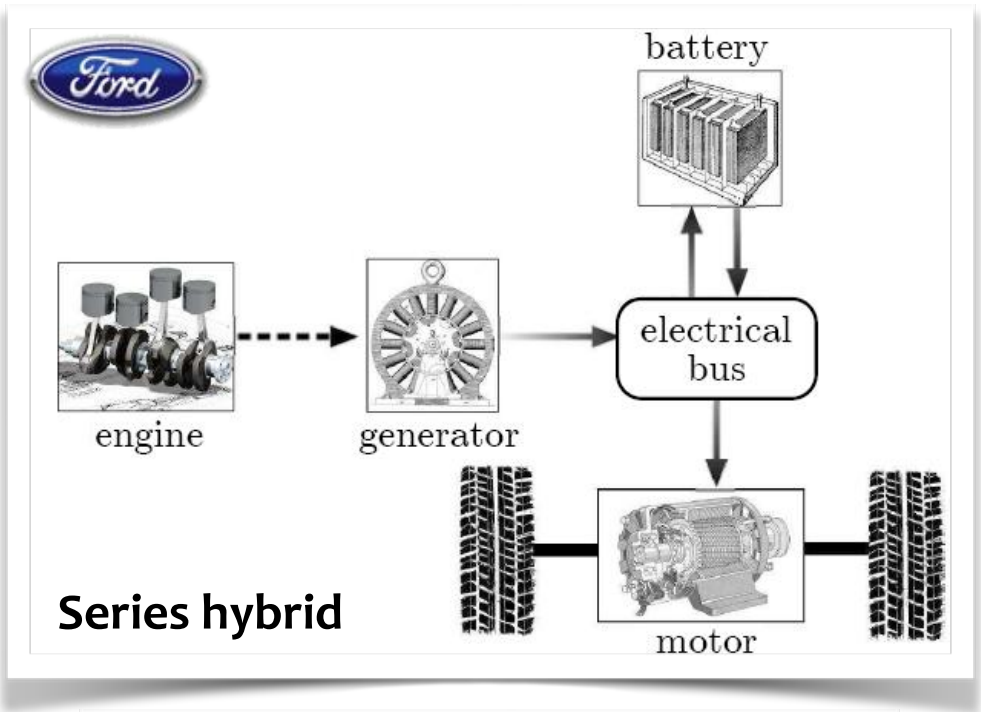
Control problem:

(Bichi, Ripaccioli, Di Cairano, Bernardini, Bemporad, Kolmanovsky, CDC 2010)

Decide optimal generation of **mechanical power** (from engine) and **electrical power** (from battery) to satisfy **driver's power request**

What will the future power request from the driver be ?

$P_{req}(w(t))$ = driver's power request



$$P_{req}(k) = P_{el}(k) + P_{mec}(k) - P_{br}(k)$$

©2019 A. Bemporad - "Model Predictive Control"

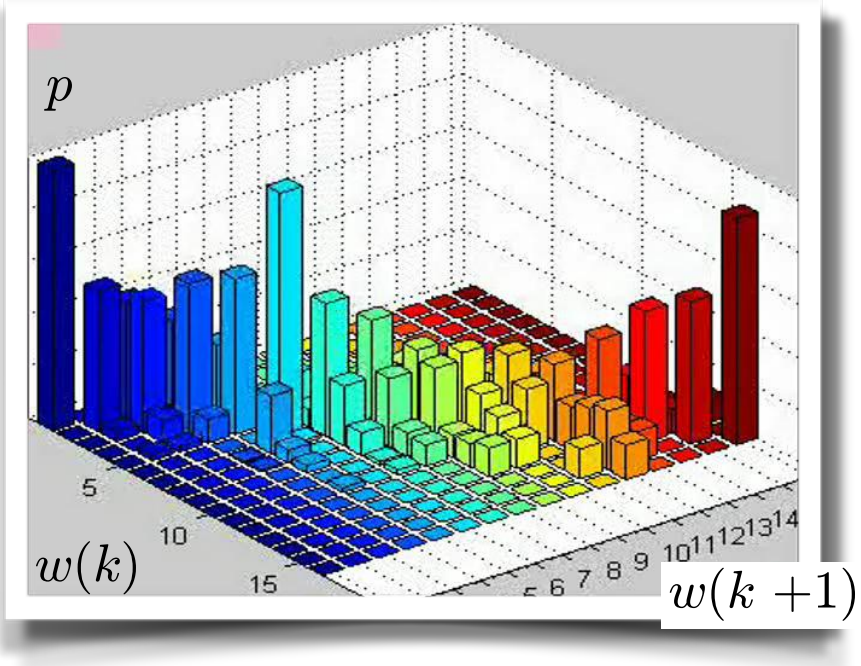
LEARNING A STOCHASTIC MODEL OF THE DRIVER

- The driver action on the vehicle is modeled by the **stochastic** process $w(k)$
- Assume that the realization $w(k)$ can be **measured** at every time step k
- Depending on the **application**, $w(k)$ may represent different quantities (e.g., power request in an HEV, acceleration, velocity, steering wheel angle, …)

Good model for control purposes: $w(k)$ = **Markov chain**

$$[T]_{ij} = \mathbf{P}[w(k+1) = w_j | w(k) = w_i]$$

Number of states in Markov chain determines the **trade-off** between complexity *and* accuracy



Transition probability matrix T is easily estimated from driver's data

Several model improvements are possible (e.g., multiple Markov chains)

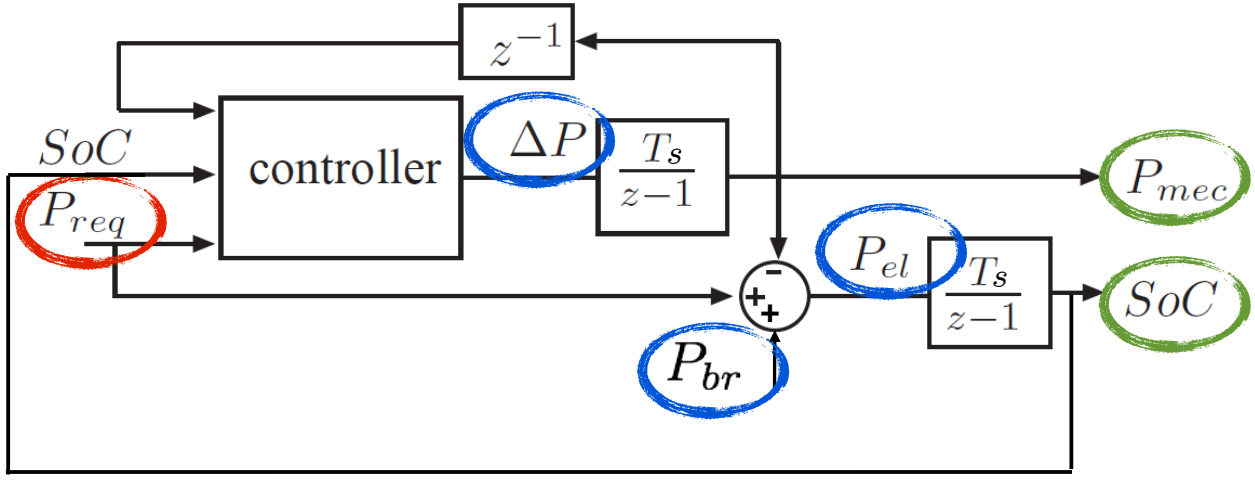
SMPC PROBLEM FOR HEV POWER MANAGEMENT

Manipulates inputs

$$\Delta P(k),\ P_{el}(k),\ P_{br}(k)$$

Uncertainty

$$P_{req}(w(k))$$



Controlled output

$$P_{req}(k) = P_{el}(k) + P_{mec}(k) - P_{br}(k)$$

sample time T_s=1 s

State-space equations

$$SoC(k+1) = SoC(k) - KT_sP_{el}(k)$$

$$P_{mec}(k+1) = P_{mec}(k) + \Delta P(k)$$

Constraints

$$\begin{array}{rcccl} SoC_{min} & \le & SoC(k) & \le & SoC_{max} \\ 0 & \le & P_{mec}(k-1) & \le & P_{mec,max} \\ P_{el,min} & \le & P_{el}(k) & \le & P_{el,max} \\ \Delta P_{min} & \le & \Delta P & \le & \Delta P_{max} \\ 0 & \le & P_{br}(k) & & \end{array}$$

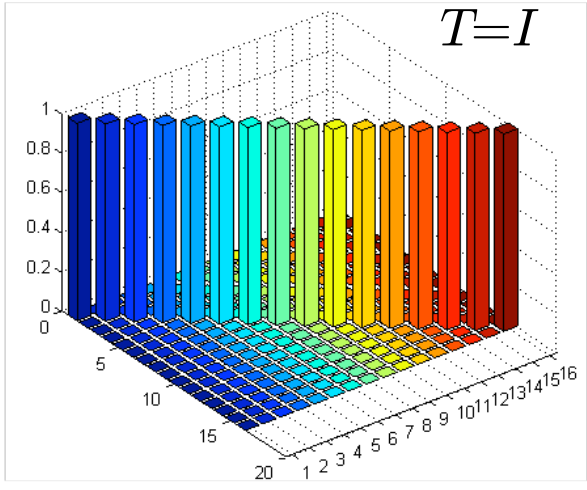
©2019 A. Bemporad - "Model Predictive Control"

COMPARISON WITH DETERMINISTIC MPC

"Frozen-time" MPC (FTMPC)

No stochastic disturbance model,
simply ZOH along prediction horizon

$$P_{req}(w(t+k|k))=P_{req}(w(k))$$



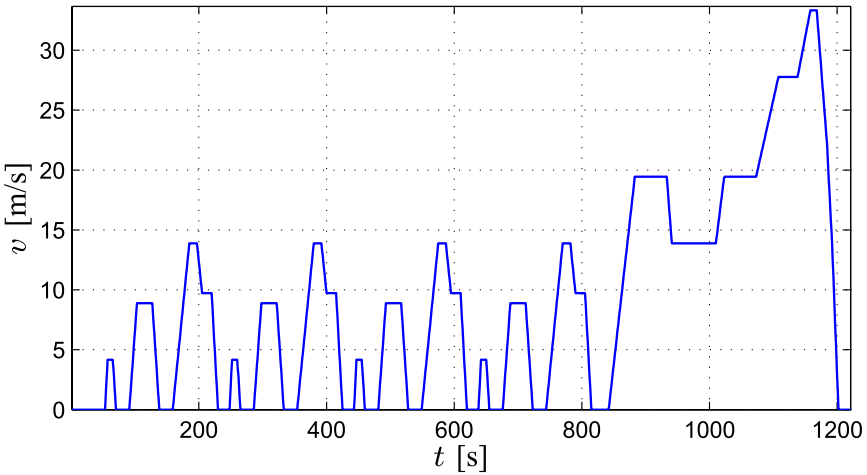
"Prescient" MPC (PMPC)

Future disturbance sequence $P_{req}(w(t+k|k))$
known in advance

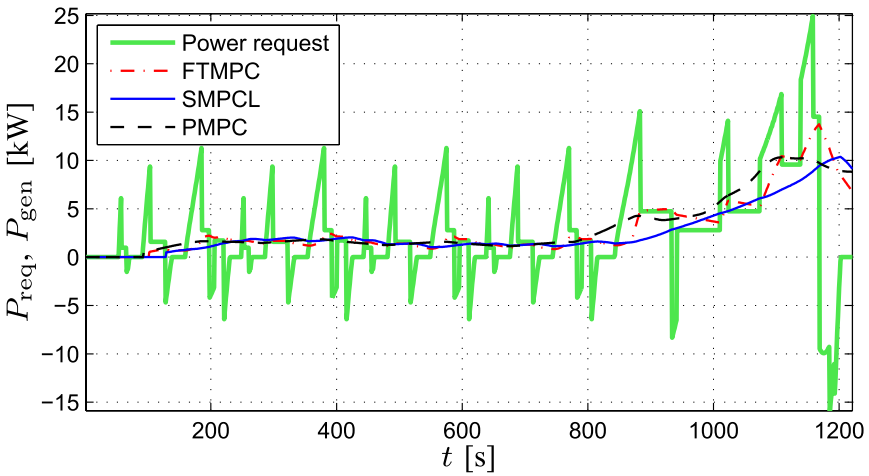
©2019 A. Bemporad - "Model Predictive Control"

SIMULATION RESULTS

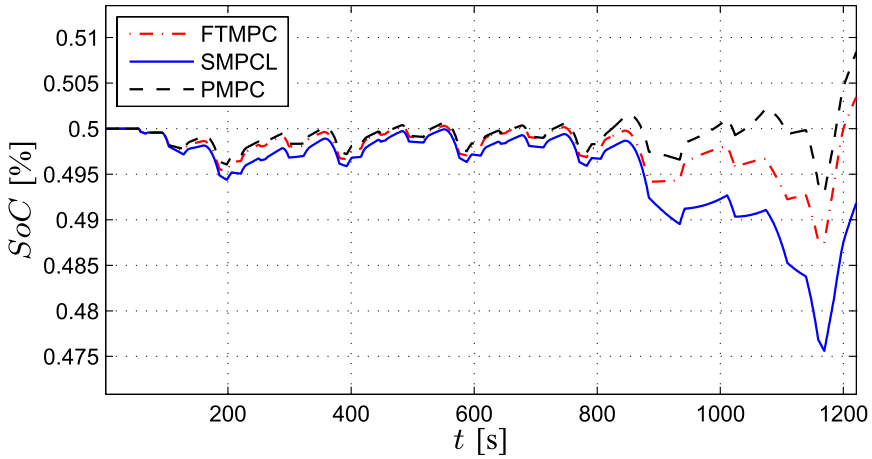
velocity



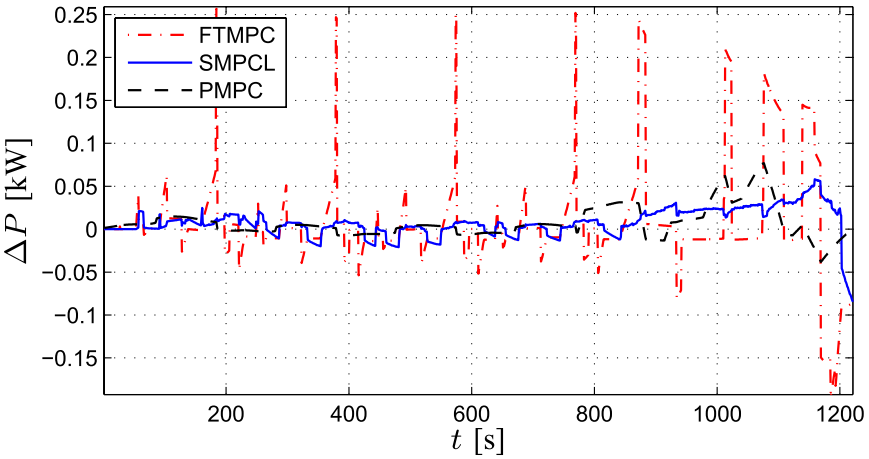
power request/generator power



state of charge



generator power variations



Results obtained on New European Driving Cycle (NEDC)

©2019 A. Bemporad - "Model Predictive Control"

SIMULATION RESULTS: CONTROLLER COMPARISON

Comparison on different driving cycles

pretty close to having
the crystal ball.
But we don't, we just
model uncertainty carefully

SHEV Energy Management Simulation Results on Standard Driving Cycles

	$\|\Delta P\|$	Fuel cons.	ΔSoC gain/loss	Equiv. fuel cons.	impr. wrt FTMPC
			NEDC		
FTMPC	37.57kW	204g	0.35%	197g	–
→ SMPCL	16.28kW	166g	-0.82%	184g	6.45%
PMPC	15.25kW	196g	0.84%	177g	9.97%
			FTP-75		
FTMPC	89.28kW	348g	0.64%	334g	–
→ SMPCL	26.07kW	292g	0.08%	290g	13.10%
PMPC	32.30kW	307g	0.89%	286g	14.20%
			FTP-Highway		
FTMPC	39.33kW	267g	0.64%	253g	–
→ SMPCL	16.84kW	281g	2.12%	235g	7.26%
PMPC	16.33kW	254g	0.91%	234g	7.32%

©2019 A. Bemporad - "Model Predictive Control"

SIMULATION RESULTS: CONTROLLER COMPARISON

Comparison on different driving cycles - Real driving data

TABLE II

SIMULATION RESULTS ON REAL-WORLD DRIVING CYCLES

	$\|\Delta P\|$	Fuel cons.	SoC gain/loss	Equiv. fuel cons.	impr. wrt FTMPC
Trace #1 - smooth accelerations					
FTMPC	37.84kW	243g	-0.05%	244g	–
→ SMPCL	14.32kW	244g	0.90%	225g	8.04%
PMPC	14.08kW	223g	-0.08%	224g	8.19%
Trace #2 - steep accelerations					
FTMPC	80.61kW	327g	0.11%	323g	–
→ SMPCL	35.74kW	320g	1.16%	287g	11.34%
PMPC	30.67kW	287g	0.17%	282g	12.73%

TABLE III

PERCENTAGE IMPROVEMENT OF SMPCL STRATEGY DUE TO ONLINE LEARNING OF THE MARKOV CHAIN

Standard cycle	Learning Improvement	Real-word Driving	Learning Improvement
NEDC	12.7%	Trace #1	1.3%
FTP-75	16.5%	Trace #2	13.4%
FTP-H.	1.1%		

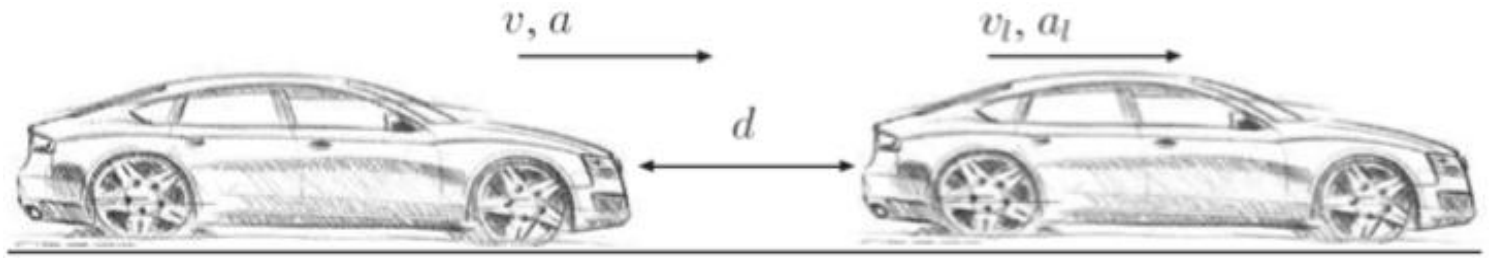
©2019 A. Bemporad - "Model Predictive Control"

STOCHASTIC MPC FOR ADAPTIVE CRUISE CONTROL

Problem setup

Follower
(equipped with ACC)

Leader
vehicle



Goals

Control the follower acceleration variation (jerk) in order to:

- Improve safety (constraint on minimum distance)
- Improve comfort (reduce acceleration / deceleration)
- Track reference velocity

©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR ACC: PREDICTION MODEL

States

$a(k)$ acceleration

$v(k)$ velocity

$d(k)$ distance

$v_l(k)$ leader velocity

Inputs

$u(k)$ jerk

$a_l(k)$ leader acceleration

Dynamical Model

$$\begin{aligned} a(k+1) &= a(k) + T_s u(k) \\ v(k+1) &= v(k) + T_s a(k) \\ v_l(k+1) &= v_l(k) + T_s a_l(k) \end{aligned}$$

Uncertainty

Stochastic leader acceleration

$a_l(k) = w(k)$

Constraints

$d(k) \geq d_{min}(k) = \delta + \gamma v(k)$ safety

$u_{min} \leq u \leq u_{max}$ comfort

References

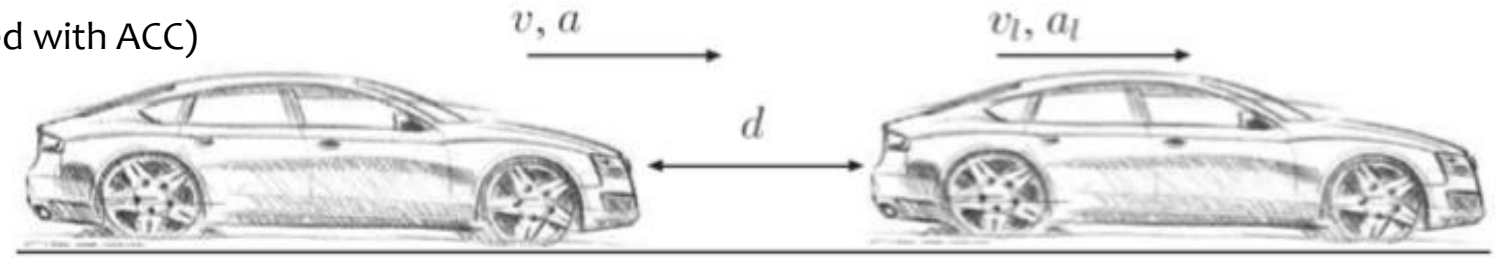
$d_{ref}(k) = \delta_{ref} + \gamma_{ref} v(k)$

$v_{ref} = 26\text{m/s}$

©2019 A. Bemporad - "Model Predictive Control"

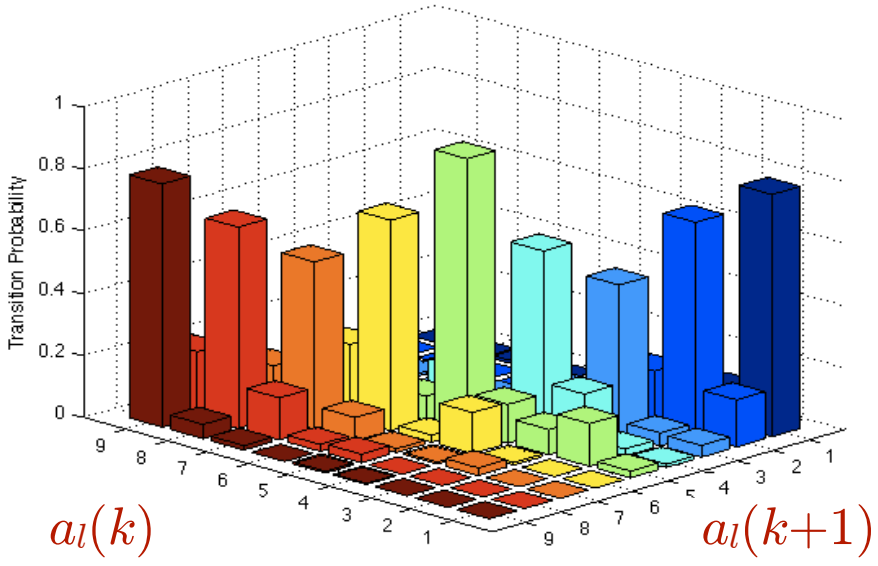
SMPC FOR ACC: STOCHASTIC LEADER MODEL

Follower vehicle
(equipped with ACC)



Leader vehicle

Leader acceleration a_l modeled by a Markov Chain (quantized in 9 states)



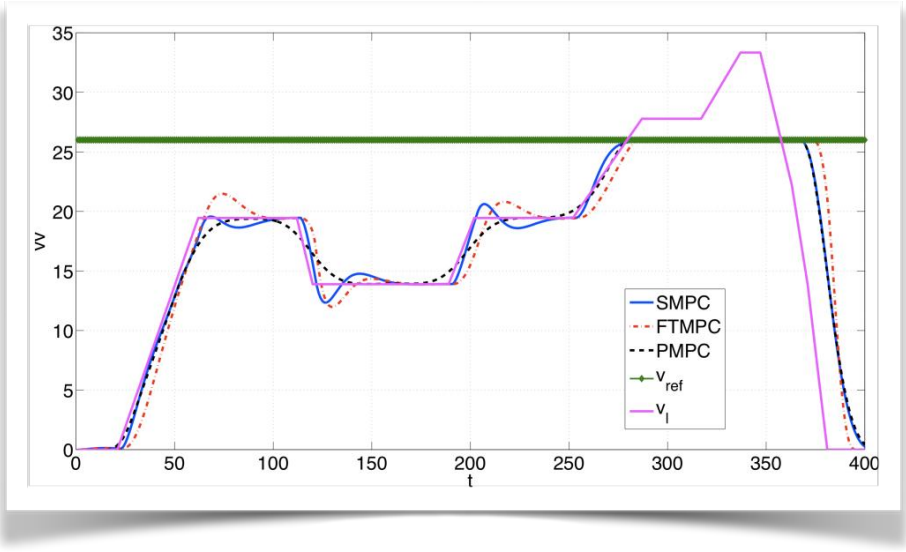
The Markov Chain is:

- Trained off-line on a collection of driving cycles (FTP, NEDC, 10-15 Mode)
- Adapted on-line by means of the learning algorithm

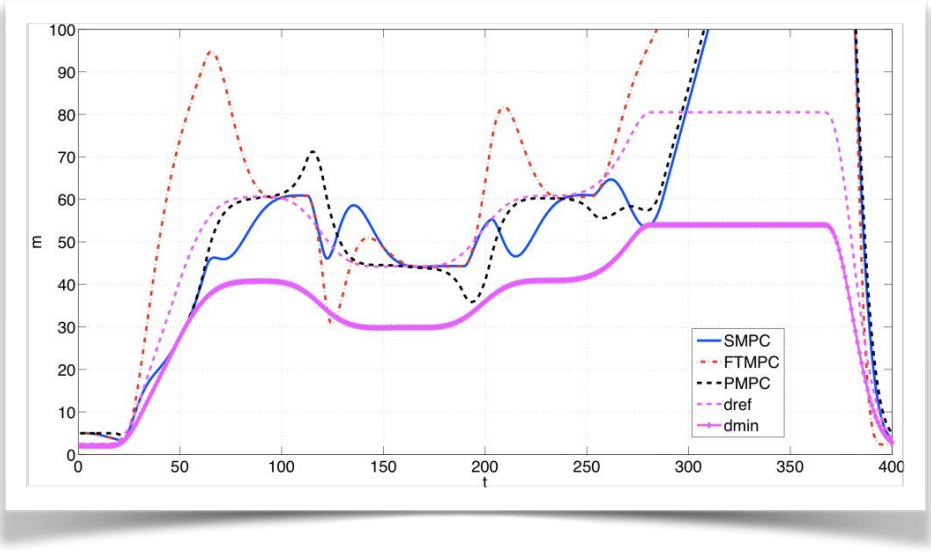
©2019 A. Bemporad - "Model Predictive Control"

SMPC FOR ACC: SIMULATION RESULTS

Speed



Distance



Stochastic MPC (blue solid line)

Frozen Time MPC (red dashed line)

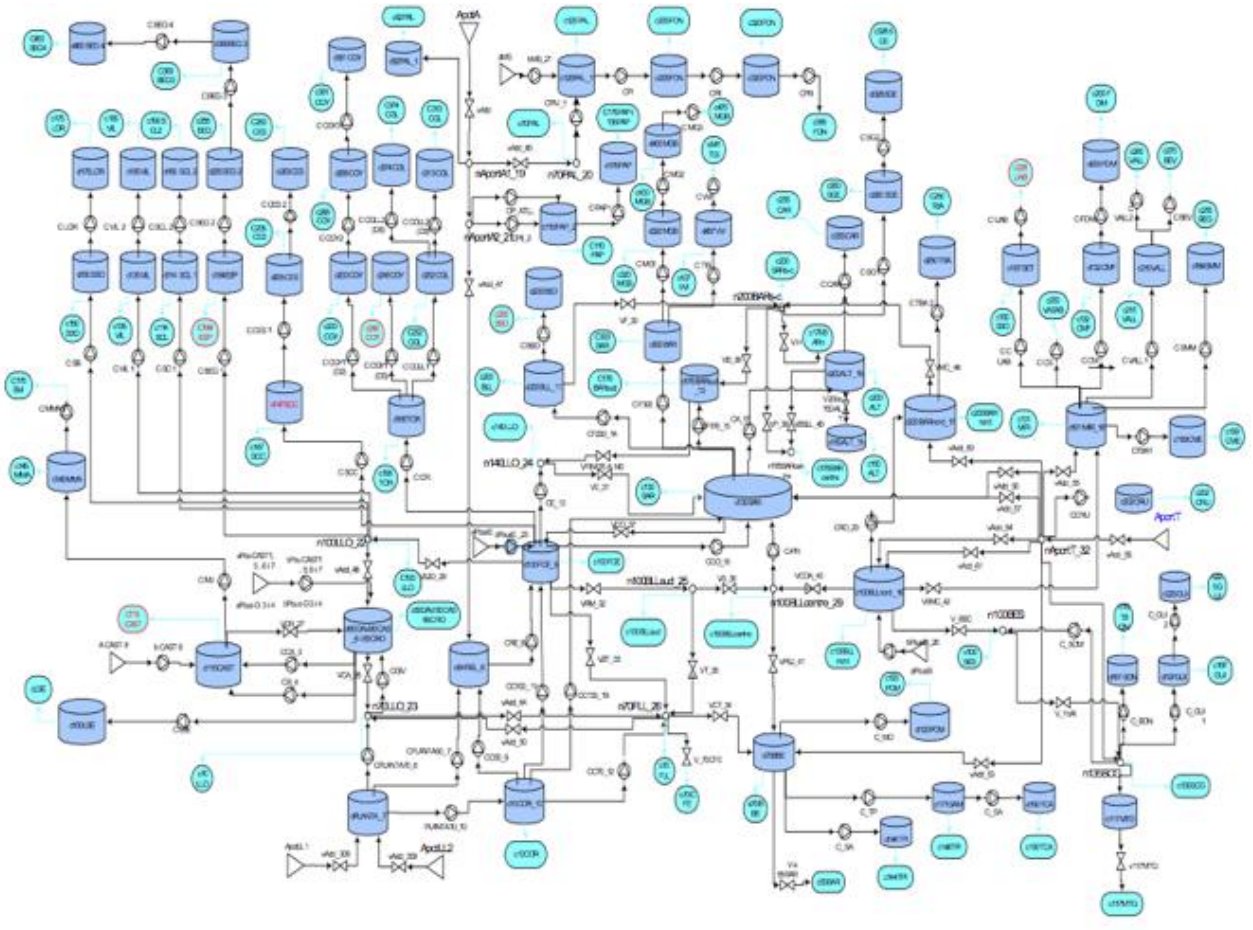
Prescient MPC (black dashed line)

Simulation results on European Urban Driving Cycle (EUDC)

©2019 A. Bemporad - "Model Predictive Control"

SMPC OF DRINKING WATER NETWORKS

DRINKING WATER NETWORK OF BARCELONA (SPAIN)



• General overview:

Municipalities supplied	23
Supply area	424 km^2
Population supplied	2.922.773
Average demand	7 m^3/s

• Network parameters:

Pipes length	4.645 km
Pressure floors	113
Sectors	218

• Facilities

Remote stations	98
Water storage tanks	81
Valves	64
Flow meters	92
Pumps / Pumping stations	180 / 84
Chlorine dosing devices	23
Chlorine analyzers	74

European FP7-ICT project WIDE
"DEcentralized and WIreless Control of Large-Scale Systems"

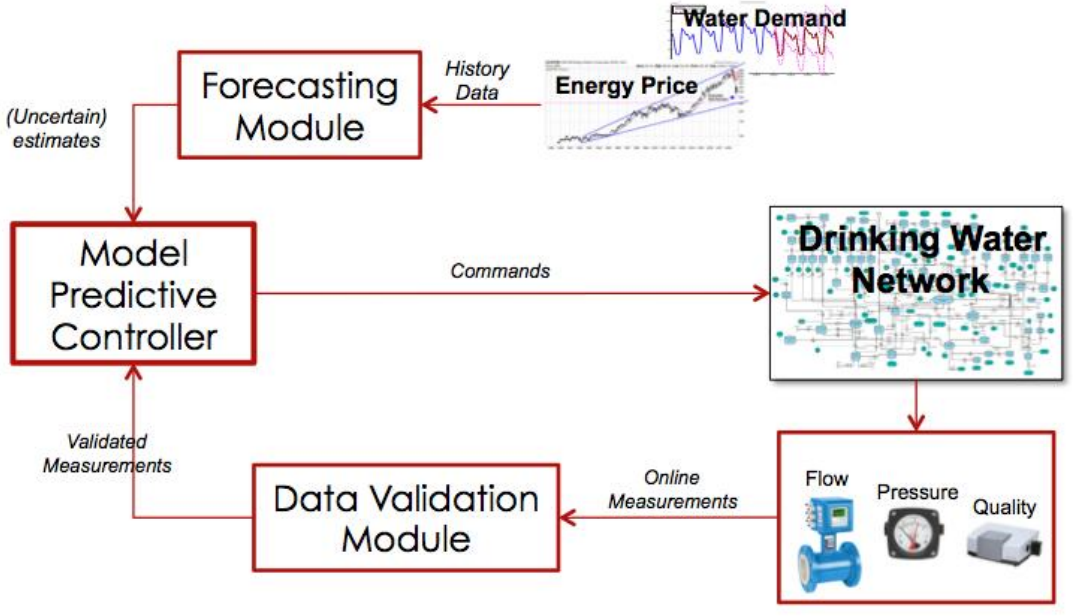
European FP7-ICT project EFFINET
"EFFIcient Integrated Real-time Monitoring and Control of Drinking Water NETworks"

©2019 A. Bemporad - "Model Predictive Control"

CONTROL OF THE DRINKING WATER NETWORK OF BCN

Main Goals:

- Reduce **electricity consumption** for pumping (€€€)
- Meet **demand** requirements
- Deliver **smooth** control actions
- Keep storage tanks above safety **limits**
- Respect the technical **limitations:** pressure limits, overflow limits & pumping capabilities



EFFINET

©2019 A. Bemporad - "Model Predictive Control"

CONTROL OF THE DRINKING WATER NETWORK OF BCN

- The control objectives are translated into cost functions:

Expected total squared water production cost (ETSWPC) = **economic** cost

$$J^{ws} = W_\alpha^2 \sum_{l=1}^{K} \sum_{i=0}^{H_p-1} p^l (\alpha_1 + \alpha_{2,k})^2 (u^l_{k+i|k})^2$$

Expected total smooth **operation** cost (ETSOC)

$$J^{\Delta} = \sum_{l=1}^{K} \sum_{i=1}^{H_p-1} p^l \ell^{\Delta}(\Delta u^l_{k+i|k})$$

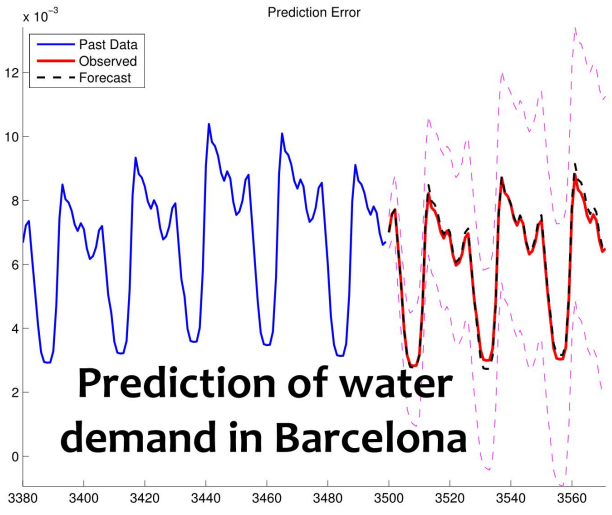
expected total **safety** storage cost (ETSSC)

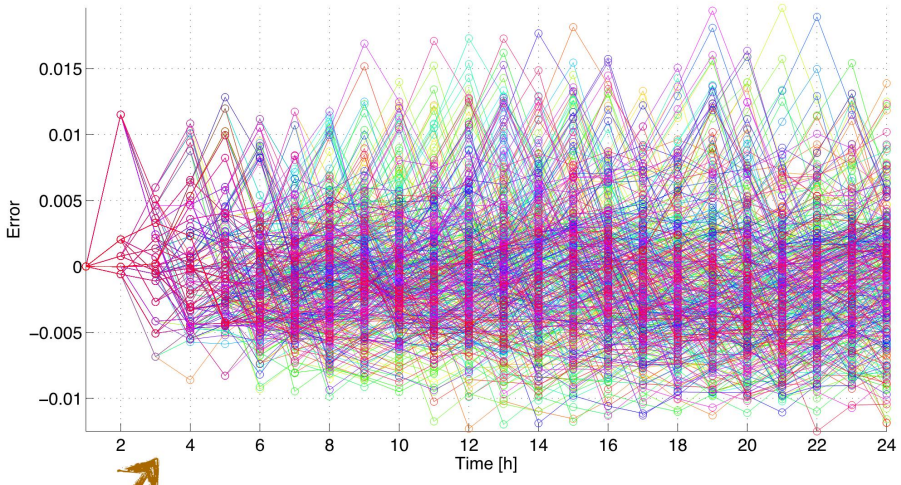
$$J^{S} = \sum_{l=1}^{K} \sum_{i=1}^{H_p} p^l \ell^{s}(x^l_{k+i|k})$$

Need to minimize the total operating cost $V = J^{ws} + J^{\Delta} + J^{S}$

©2019 A. Bemporad - "Model Predictive Control"

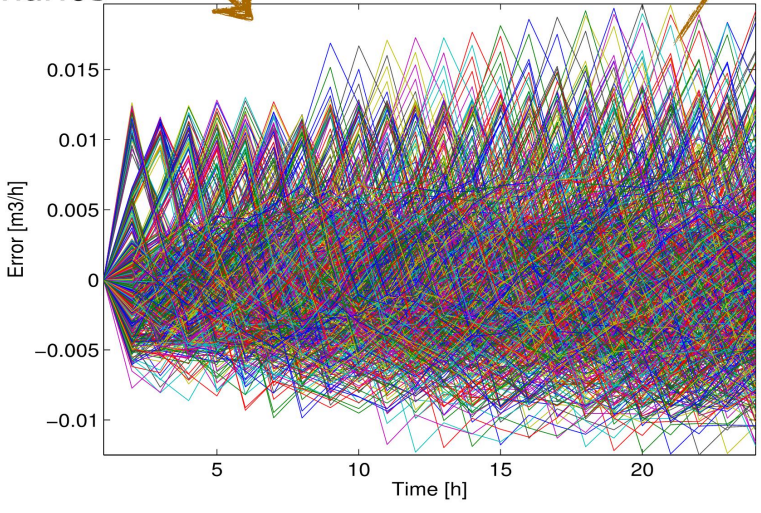
CONTROL OF THE DRINKING WATER NETWORK OF BCN





Uncertainty represented as a fan of scenarios

Reduction to a scenario tree



$$d_{k+i|k} = \hat{d}_{k+i|k} + \epsilon_{k+i|k}$$

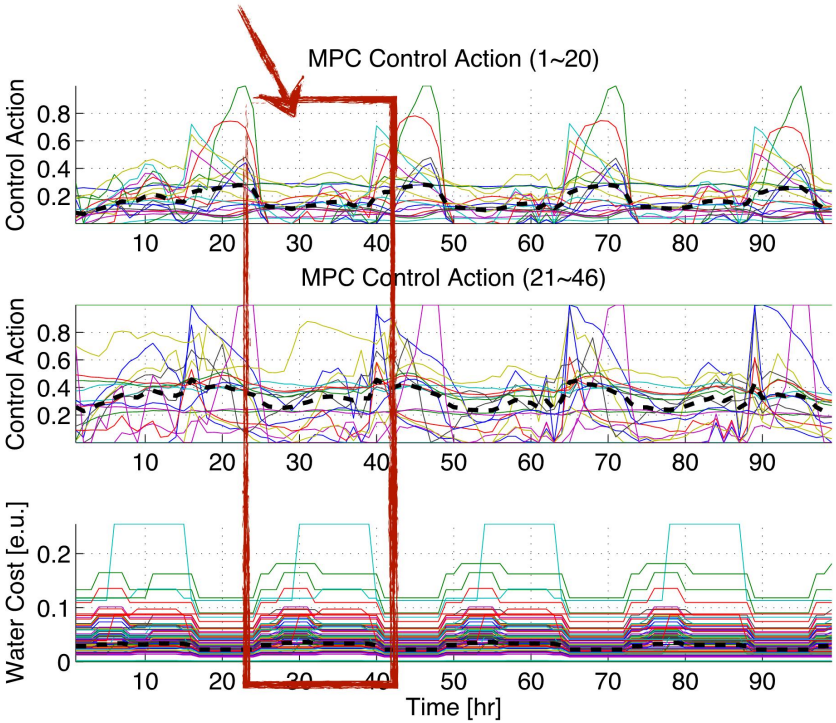
Uncertainty:
demand prediction error

Agbar

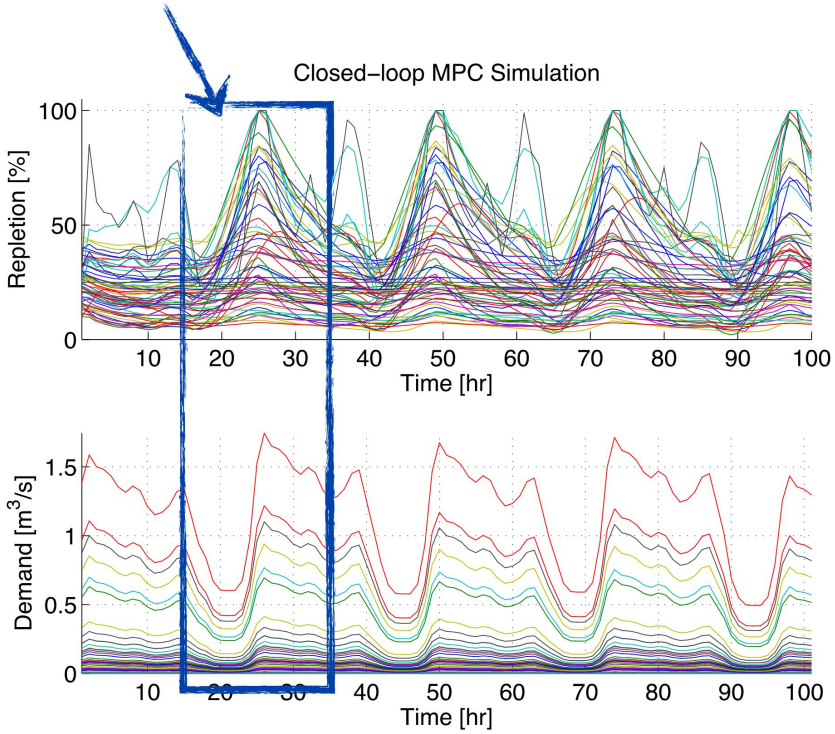
©2019 A. Bemporad - "Model Predictive Control"

CONTROL OF THE DRINKING WATER NETWORK OF BCN

Economic: *Avoid pumping when the price of electricity is high*

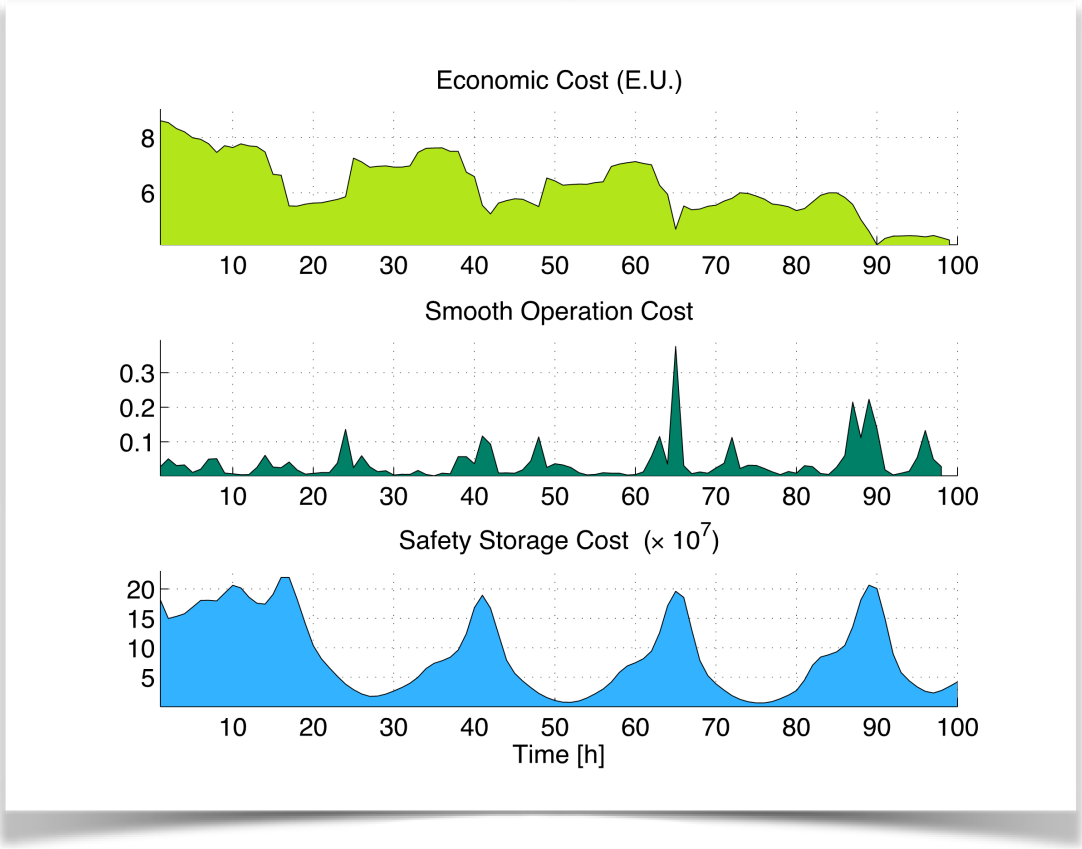


Foresight: *tanks start loading up before the consumers ask for water*

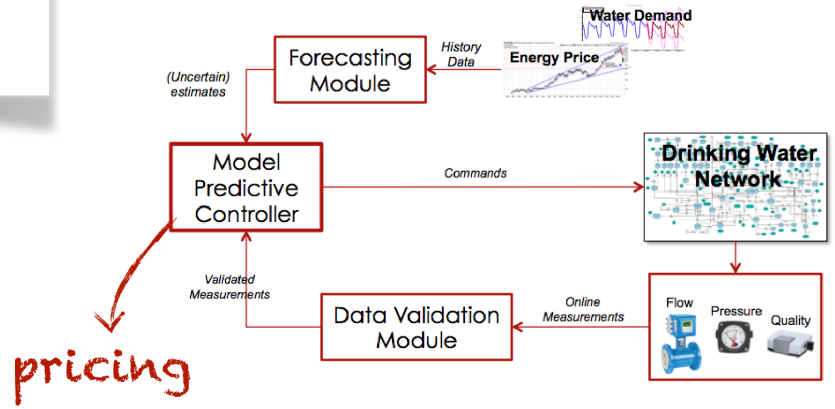


©2019 A. Bemporad - "Model Predictive Control"

CONTROL OF THE DRINKING WATER NETWORK OF BCN



SMPC: The network operator has online information about the current and predicted operating cost in real time



©2019 A. Bemporad - "Model Predictive Control"

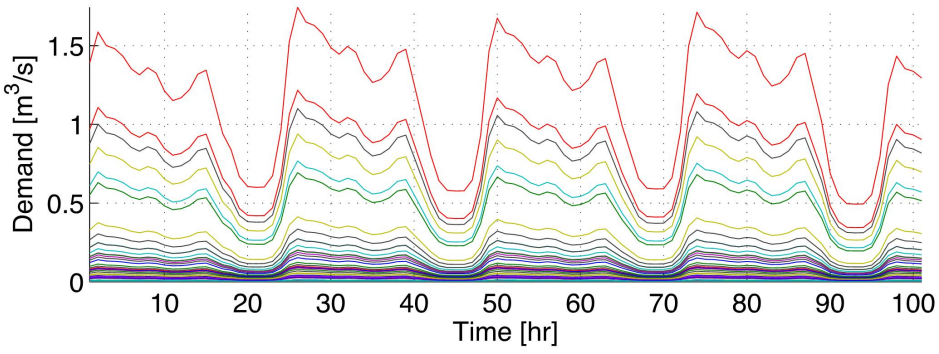
SCALABILITY OF THE SMPC APPROACH

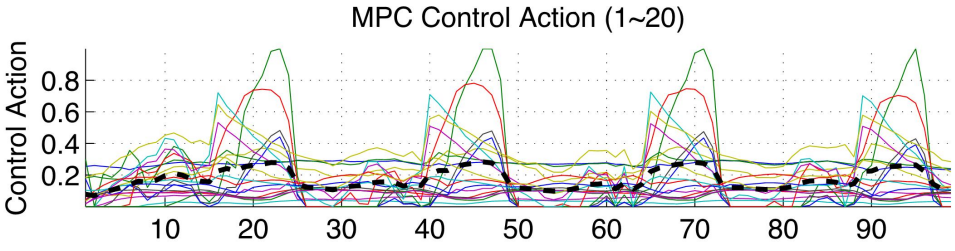
How does the approach **scale** with the dimension of the system ?

- The dGPAD algorithm scales-up well with the size of the scenario tree (thanks to heavy parallelization)
- Scalable alternatives:
  - **Decentralized SMPC**: divide into subsystems and control each of them in parallel, exchanging some decisions **after** computations (others' decisions = measured disturbances) (Bemporad, Barcelli, 2010)
  - **Distributed SMPC**: exchange some global variables **during** computations (Negenborn, Maestre, IEEE CSM, 2014)
- The same dGPAD algorithm can be used for decentralized SMPC (immediately), or for distributed SMPC by relaxing also the constraints that (weakly) couple the subsystems

©2019 A. Bemporad - "Model Predictive Control"

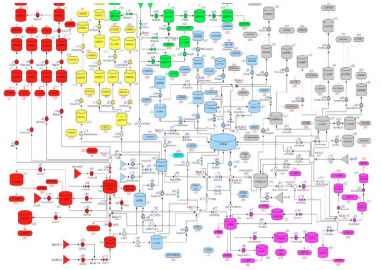
STOCHASTIC MPC AND PARALLEL COMPUTATIONS ON GPU

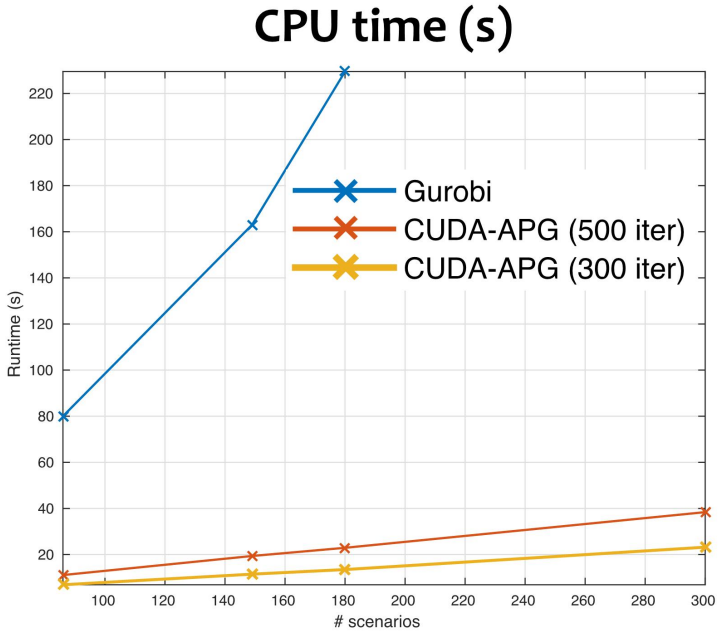




Drinking water network of Barcelona:

63 tanks
114 controlled flows
17 mixing nodes





APG = Accelerated Proximal Gradient, parallel implemented on NVIDIA Tesla 2075 CUDA platform

MPC-controlled network:

- Minimum pressure requirement hardly violated
- ~5% savings on energy cost w.r.t. current practice
- Smooth control actions
- sampling time = 1 hour

FP7-ICT project "**EFFINET - Efficient Integrated Real-time Monitoring and Control of Drinking Water Networks**" (2012-2015)

(Sampathirao, Sopasakis, Bemporad, 2015)

©2019 A. Bemporad - "Model Predictive Control"