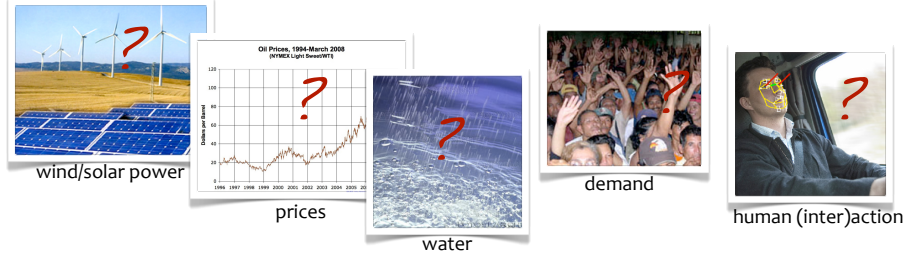


Stochastic MPC

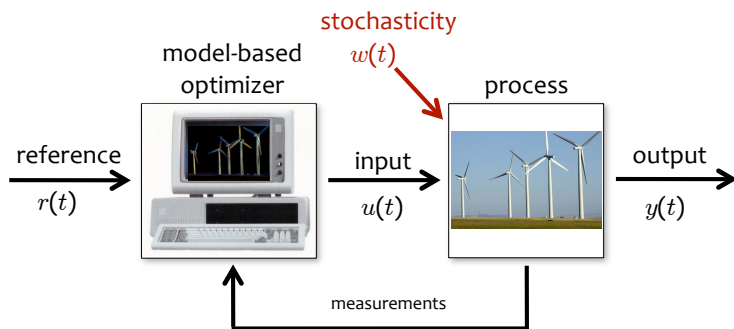
Stochastic systems

- In many control problems decisions must be taken under **uncertainty**
- **Robust** control approaches do not model uncertainty (only assume that is bounded) and pessimistically consider the worst case
- **Stochastic models** provide additional information about uncertainty



Need to include stochastic models in control problem formulation

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

Stochastic Model Predictive Control

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

$$\begin{aligned} \min_u \quad & E_w \left[\sum_{k=0}^{N-1} \ell(y_k - r(t+k), u_k) \right] \\ \text{s.t.} \quad & x_{k+1} = f(x_k, u_k, w_k) \\ & y_k = g(x_k, u_k, w_k) \\ & u_{\min} \leq u_{t+k} \leq u_{\max} \\ & y_{\min} \leq y_k \leq y_{\max}, \quad \forall w \\ & x_0 = x(t) \end{aligned}$$

$x(t)$ = process state
 $u(t)$ = manipulated vars
 $y(t)$ = controlled output
 $w(t)$ = **stochastic disturbances**

- Only apply the first optimal move $u^*(t)$, discard $u^*(t+1)$, $u^*(t+2)$, ...
- At time $t+1$: Get new measurement $x(t+1)$, repeat the optimization. And so on ...

Linear stochastic MPC w/ discrete disturbance

- Linear stochastic prediction model

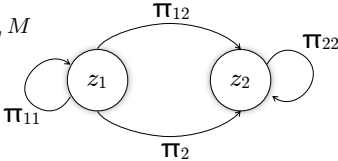
$$x(t+1) = A(w(t))x(t) + B(w(t))u(t) + H(w(t))$$

- Discrete disturbance

$$w(t) \in \{w_1, \dots, w_s\} \quad p_j(t) = \Pr[w(t) = w_j] \quad \sum_{j=1}^s p_j(t) = 1, \quad \forall t \geq 0$$

- Probabilities $p_j(t)$ can have their own dynamics. Example: Markov chain

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

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


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

Cost functions for SMPC to minimize

performance $\rightarrow J(u, w) \triangleq \sum_{k=0}^{N-1} \ell(y_k - r(t+k), u_k)$

- Expected performance

$$\min_u E_w [J(u, w)]$$

- Tradeoff between expected performance & risk

$$\min_u E_w [J(u, w)] + \rho \text{Var} [J(u, w)]$$

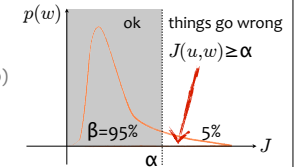
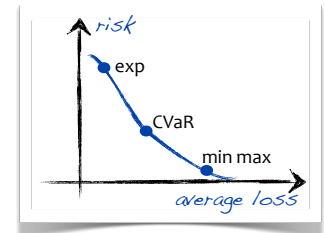
- Conditional Value-at-Risk (CVaR)

$$\min_{u, \alpha} \left\{ \alpha + \frac{1}{1-\beta} E[\max(J(u, w) - \alpha, 0)] \right\} \quad (\text{Rockafellar, Uryasev, 2000})$$

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

- Min-max

$$\min_u \{ \max_w J(u, w) \} = \text{minimize worst case performance}$$



Linear stochastic MPC formulation

- 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 $\lim_{t \rightarrow \infty} E[x'(t)x(t)] = 0$ (for $H(w(t))=0$)

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

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

(Morozan, 1983)

ensures mean-square stability

- Existing SMPC approaches:

| | | |
|-------------------------------------|---|-----------------------------------|
| (Schwarme & Nikolaou, 1999) | (Munoz de la Pena, Bemporad, Alamo, 2005) | (Ono, Williams, 2008) |
| (Wendt & Wozny, 2000) | (Couchman, Cannon, Kouvaritakis, 2006) | (Oldewurtel, Jones, Morari, 2008) |
| (Batina, Stoorvogel, Weiland, 2002) | (Primbs, 2007) | (Bernardini & Bemporad, 2009) |
| (van Hessem & Bosgra 2002) | (Bemporad, Di Cairano, 2005) | |

Stochastic program

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

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

- Each scenario has its own evolution $x_{k+1}^j = A(w_k^j)x_k^j + B(w_k^j)u_k^j$ (LTV system)

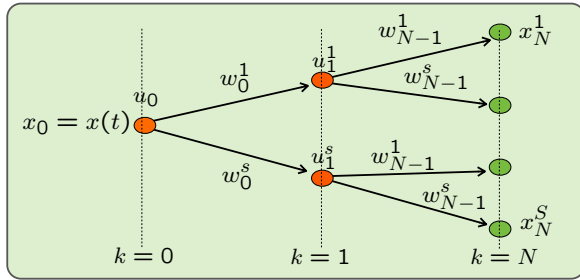
- Expectations become simple sums

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

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

This is again a quadratic function of the inputs

Scenario tree



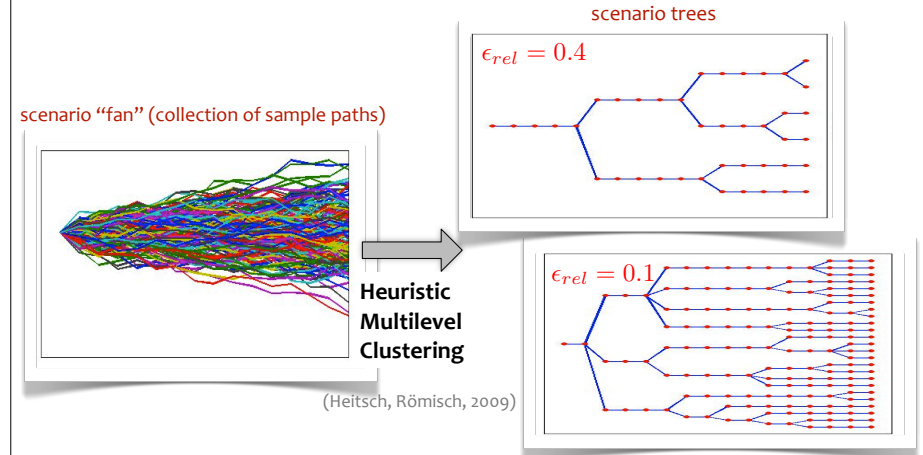
- Scenario = path on the tree
- Number S of scenarios = number of leaf nodes
- Some paths can be removed if their probability is very small (at your own risk)
- **Causality constraint:** $w_k^j = u_k^h$ when scenarios j and h share the same node at prediction time k (for example: $w_0^j = u_0^h$ at root node $k=0$)

$$\min \dots + p^j(x_k^j)' Q x_k^j + \dots$$

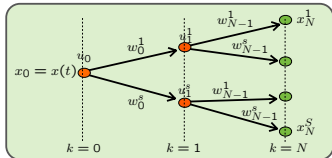
$$y_{\min} \leq y_k \leq y_{\max}, \forall w$$

Scenario tree generation from data

- Scenario trees can be generated by **clustering** sample paths
- Paths can be obtained by Monte Carlo simulation of (arbitrarily complex) models, or from historical data

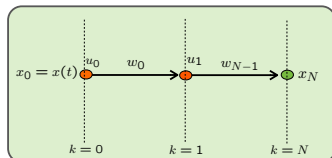


Scenario enumeration



scenario tree

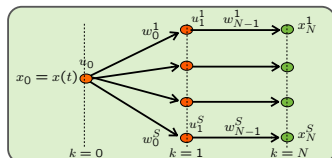
Causality is exploited: decision u_k only depends on past disturbance realizations $\{w_0, w_1, \dots, w_{k-1}\}$



deterministic

Only a sequence of disturbances is considered

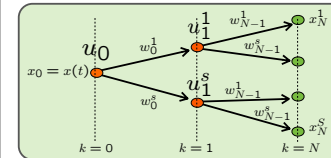
- frozen-time: $w_k \equiv w(t), \forall k$ (causal prediction)
- anticipative action: $w_k \equiv w(t+k)$ (non-causal)
- "expected" problem: $w_k = E[w(t+k)|t]$ (causal)



scenario "fan"

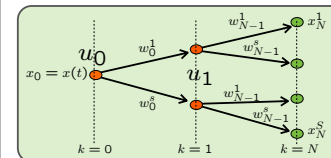
- generate a set of scenarios (Monte Carlo simulation)
- decision u_k also depends on future disturbance realizations $\{w_k, w_{k+1}, \dots, w_{N-1}\}$

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 $\{u_0, u_1, \dots, u_{N-1}\}$ 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 $w^j \equiv u, \forall j = 1, \dots, S$ (=less degrees of freedom)

Linear stochastic stabilization

- Assume $w(t) \in \{w_1, \dots, w_s\}$ and **constant** probability $p(t) \equiv p, \forall t$
- The **stochastic convergence** condition $E_{w(t)}[V(x(t+1)) - V(x(t))] \leq -x(t)'Lx(t)$ can be recast as the **LMI** condition

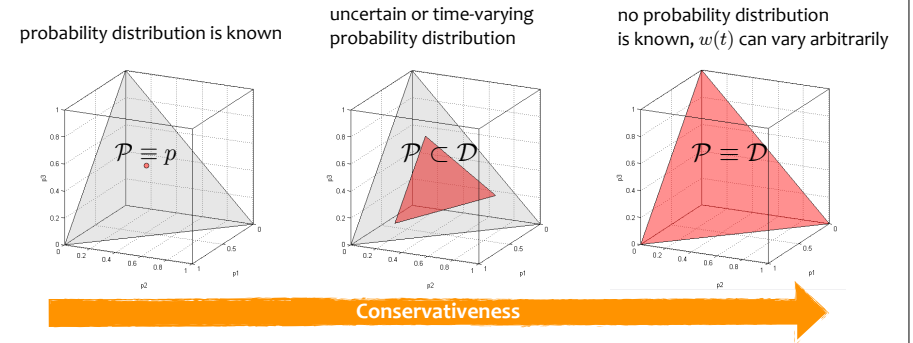
$$\begin{bmatrix} Q & Q & \sqrt{p_1}(A_1Q+B_1Y)' & \dots & \sqrt{p_s}(A_sQ+B_sY)' \\ \sqrt{p_1}(A_1Q+B_1Y) & W & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sqrt{p_s}(A_sQ+B_sY) & 0 & \dots & \dots & Q \end{bmatrix} \succeq 0$$

$$Q = Q' > 0, W = W' > 0$$

- The **Lyapunov function** is $V(x) = x'Q^{-1}x$
- Mean-square stability** guaranteed by linear feedback $u(k) = Kx(k)$, $K = YQ^{-1}$, $L = W^{-1}$
- A **minimum decrease rate** L can be imposed

Linear stochastic stabilization

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



- 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

Stabilizing SMPC

- Impose stochastic stability constraint in SMPC problem (=quadratic constraint w.r.t. u_0) (Bernardini, Bemporad, CDC'09) (Bernardini, Bemporad, IEEE TAC, 2012)

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

performance and stability are decoupled

- SMPC approach:
 - Solve LMI problem off-line to find stochastic Lyapunov fcn $V(x) = x'Q^{-1}x$
 - 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**

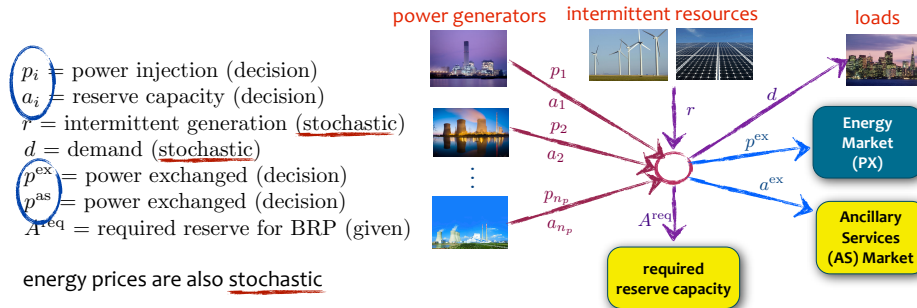
Note: recursive feasibility guaranteed by backup solution $u(k) = Kx(k)$

A few sample applications of SMPC

- Financial engineering:** dynamic hedging of portfolios replicating synthetic options (Bemporad, Bellucci, Gabbriellini, 2009) (Bemporad, Gabbriellini, Puglia, Bellucci, 2010) (Bemporad, Puglia, Gabbriellini, 2011)
- Energy systems:** power dispatch in smart grids, optimal bidding on electricity markets (Patrinos, Trimboli, Bemporad 2011) (Puglia, Bernardini, Bemporad 2011)
- Automotive control:** energy management in HEVs, adaptive cruise control (human-machine interaction) (Bichi, Ripaccioli, Di Cairano, Bernardini, Bemporad, Kolmanovsky, CDC 2010)
- Networked control:** improve robustness against communication imperfections (Bernardini, Donkers, Bemporad, Heemels, NECSYS 2010)

SMPC for real-time market-based power dispatch

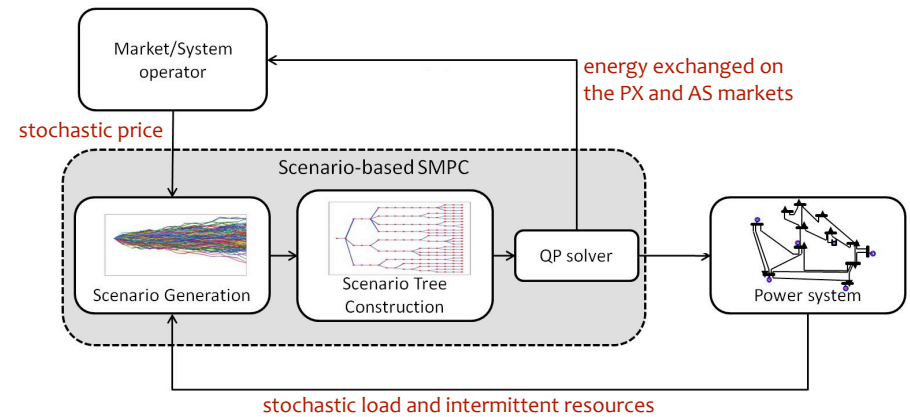
- A Balance Responsible Party (BRP) is the only legal entity trading on the **energy (PX)** and **ancillary service (AS)** markets
- **Objective:** Minimize (expected) costs via efficient use of intermittent resources, and maximize (expected) profits by trading on PX and AS markets
- **Constraints:** Grid capacity constraints, rate limits, load balancing, AS balancing



SMPC for real-time market-based power dispatch

(Patrinos, Trimboli, Bemporad 2011)

- Stochastic MPC architecture



SMPC for market-based optimal power dispatch

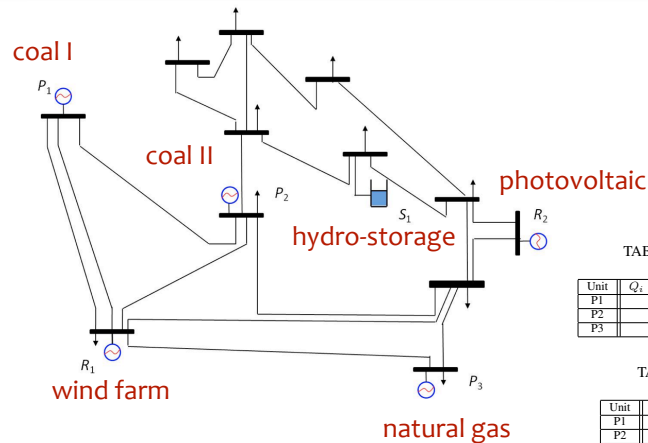


TABLE I: Generator Cost Data

| Unit | Q_i (\$/MWh ²) | q_i (\$/MWh) | c_i (\$) |
|------|------------------------------|----------------|------------|
| P1 | 0.009 | 30.375 | 398.025 |
| P2 | 0.0225 | 73.35 | 292.275 |
| P3 | 0.0488 | 61.488 | 489.952 |

TABLE II: Generator Data

| Unit | p_i^{\min} | p_i^{\max} | Δp_i^{\min} | Δp_i^{\max} |
|------|--------------|--------------|---------------------|---------------------|
| P1 | 450 | 1100 | -250 | 250 |
| P2 | 50 | 500 | -200 | 200 |
| P3 | 50 | 100 | -75 | 75 |

TABLE III: Storage Data

| Unit | x_i^{\min} | x_i^{\max} | Δx_i^{\min} | Δx_i^{\max} | α_i | α_i^c | α_i^d |
|------|---------------------------|--------------|---------------------------|---------------------|---------------------------|--------------|--------------|
| SI | 15 | 300 | -120 | 120 | 0.95 | 0.85 | 0.90 |
| | $u_i^{\min} = u_i^{\max}$ | | $u_i^{\min} = u_i^{\max}$ | | $u_i^{\min} = u_i^{\max}$ | | |
| | 0 | | 0 | | 300 | | |

SMPC for market-based optimal power dispatch

- Numerical results

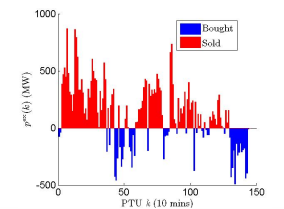
Exact knowledge future uncertainty

Time-varying expectations used for future uncertainty

SMPC: as tree density increases, performance gets better, although numerical complexity gets larger

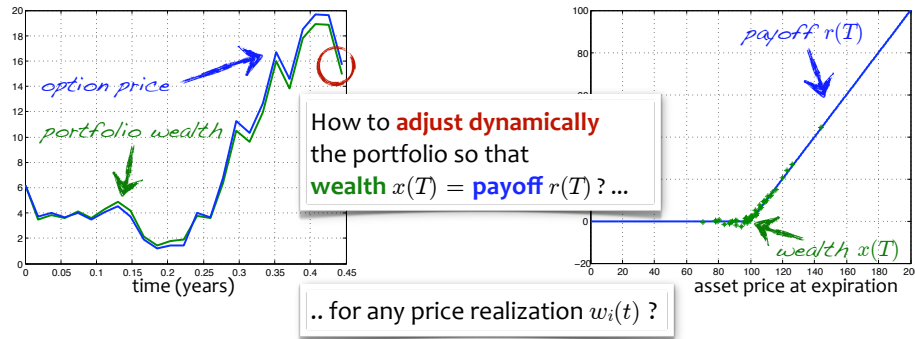
| Algorithm | Storage Cost | No Storage Cost | Avg # of nodes |
|----------------------------------|--------------|-----------------|----------------|
| Prescient-OC | 6427979 | 6879741 | |
| CE-MPC | 9778750 | 9819518 | |
| SSMPC ($\epsilon_{rel} = 0.1$) | 7134582 | 7245962 | 350 |
| SSMPC ($\epsilon_{rel} = 0.2$) | 7144011 | 7249401 | 335 |
| SSMPC ($\epsilon_{rel} = 0.3$) | 7148494 | 7250207 | 172 |
| SSMPC ($\epsilon_{rel} = 0.4$) | 7179848 | 7264505 | 87 |
| SSMPC ($\epsilon_{rel} = 0.5$) | 7224912 | 7267497 | 50 |
| SSMPC ($\epsilon_{rel} = 0.6$) | 7239985 | 7277410 | 38 |
| SSMPC ($\epsilon_{rel} = 0.7$) | 7259491 | 7298023 | 31 |
| SSMPC ($\epsilon_{rel} = 0.8$) | 7255246 | 7312092 | 26 |
| SSMPC ($\epsilon_{rel} = 0.9$) | 7260424 | 7318643 | 22 |
| SSMPC ($\epsilon_{rel} = 1.0$) | 7260424 | 7318642 | 20 |

power exchanged with grid



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 $w_1(t), w_2(t), \dots, w_n(t)$
- At the **expiration date** T , the option is worth the **payoff** $r(T)$ = wealth (€) to be returned to the customer



Portfolio dynamics

- Portfolio wealth at time t :

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

$u_0(t)$: money in bank account (risk-free asset)
 $w_i(t)$: number of assets #i
 $u_i(t)$: price of asset #i (stochastic process)

- Example: $w_i(t) = \mathbf{log-normal}$ model (used in Black-Scholes' theory)

$$dw_i = (\mu dt + \sigma dz_i)w_i \quad \text{geometric Brownian motion}$$

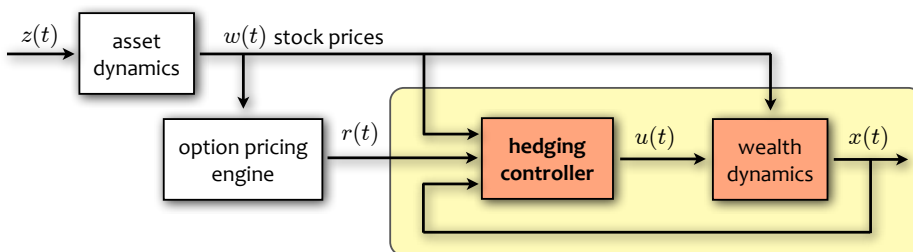
- Assets traded at **discrete-time** intervals under the **self-balancing constraint**:

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

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

Option hedging = linear stochastic control

- Block diagram of dynamic option hedging problem:



- Payoff function example: $r(T) = \max\{w(T) - K, 0\}$ **European call**

- 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")

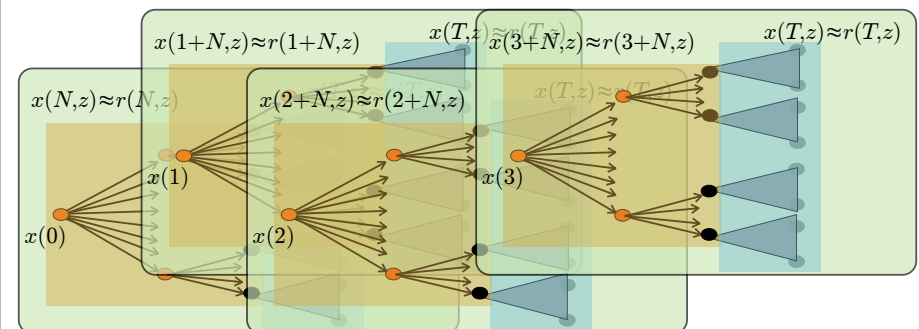
SMPC for dynamic option hedging

- Stochastic finite-horizon optimal control problem:

$$\min_{\{u(k; z)\}} \text{Var}_z [x(t+N, z) - r(t+N, z)]$$

$$\text{s.t. } x(k+1, z) = (1+r)x(k, z) + \sum_{i=0}^n b_i(k, z)u_i(k, z), \quad k = t, \dots, t+N$$

$$x(t, z) = x(t)$$



SMPC for dynamic option hedging

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

• Special case: use $N=1$!

*minimum
variance
control !*

$$\min_{u(t)} \text{Var}_z [x(t+1, z) - r(t+1, z)]$$

$$\text{s.t. } x(t+1, z) = (1+r)x(t) + \sum_{i=0}^n b_i(t, z)u_i(t)$$

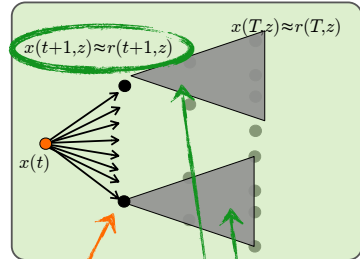
✓ Only one vector $u(t)$ to optimize

✓ 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 !

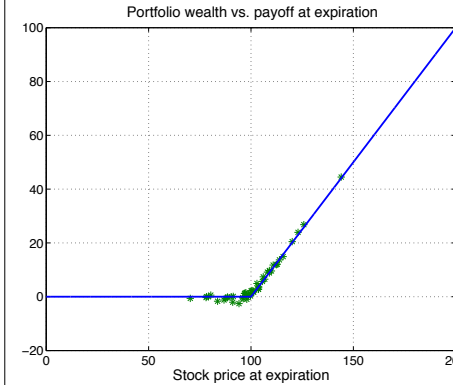
(n = number of traded assets)



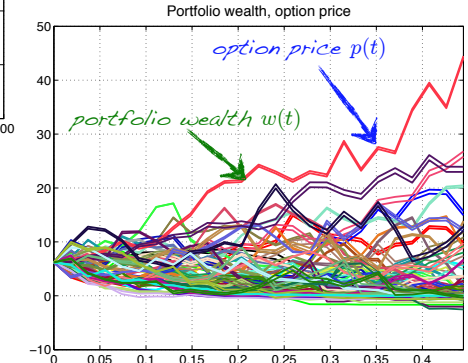
Perfect hedging assumption from time t+1 to T
Optimize up to time t+1

Example: Hedging an European call

(Bemporad, Gabriellini, Puglia, Bellucci, 2010)

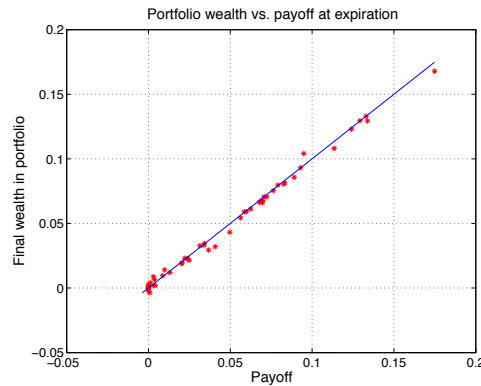


- Black-Scholes model (=log-normal)
- volatility=0.2, risk-free=0.04
- $T=24$ weeks ($\Delta t=1$ week)
- 50 simulations
- $M=100$ scenarios
- Pricing method: Monte Carlo sim.
- **SMPC**



- CPU time = 7.52 ms per SMPC step (Matlab R2009 on this mac)

Example: Hedging an exotic option



- 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 $t+T$**

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

“Napoleon cliquet” option

$t_i=0, 8, 16, 24$ weeks

- CPU time = 1625 ms per SMPC step (Matlab R2009 on this mac)

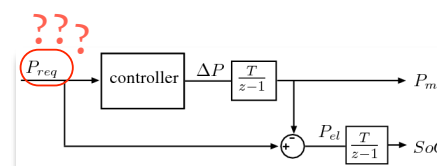
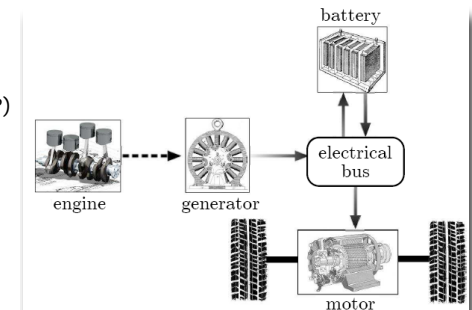
SMPC for hybrid electric vehicles

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

How to split **power** optimally among different power sources in **HEVs** (hybrid electric vehicles) to match power demanded by driver ?

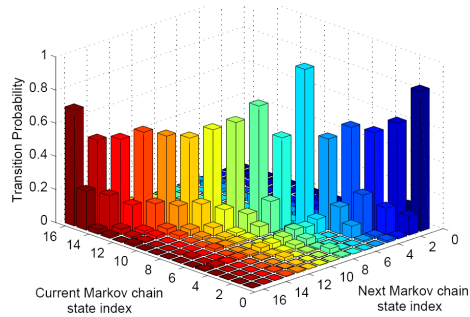
Existing approaches

- Deterministic Dynamic Programming (DDP)
- Stochastic Dynamic Programming (SDP)
- Rule-based
- Game Theory
- Deterministic (hybrid) MPC



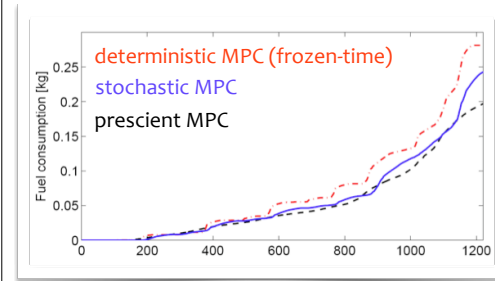
Stochastic model of power demand

- Power demanded by driver modeled as a **Markov chain**

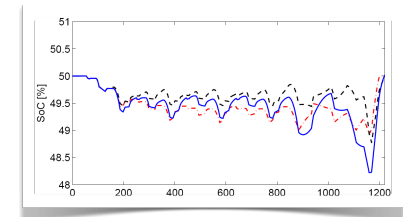
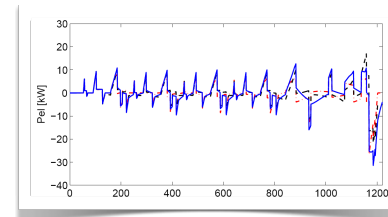


- requested power quantized in 16 levels
- Markov chain is modeling the probabilities of transition from one level to another
- transition probabilities estimated off-line on a collection of driving cycles (FTP, NEDC, 10-15 Mode)

Stochastic vs. prescient and deterministic MPC

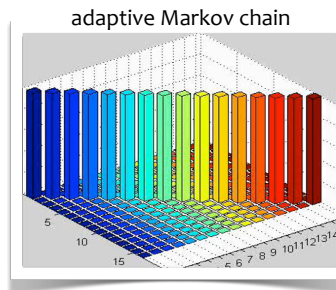
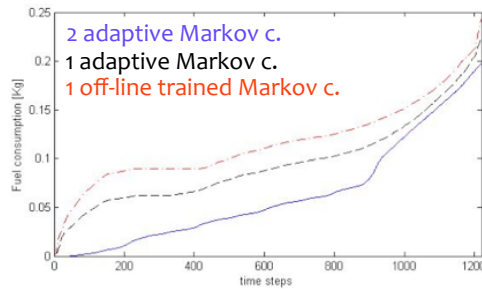


| driver model | fuel cons. [kg] | fuel improv. [%] |
|-----------------|-----------------|------------------|
| frozen-time MPC | 0.281 | - |
| stochastic MPC | 0.243 | 13.5 |
| prescient MPC | 0.197 | 29.8 |



A single Markov chain is tuned off-line

SMPC results (NEDC driving cycle)



pretty close to having the crystal ball !

| driver model | NEDC | FTP | 10-15 mode |
|----------------|-------|-------|------------|
| deterministic | 0.281 | 0.533 | 0.125 |
| 1 Markov chain | 0.244 | 0.323 | 0.091 |
| 2 Markov chain | 0.224 | 0.323 | 0.089 |
| 2 MC adaptive | 0.198 | 0.325 | 0.088 |
| prescient | 0.197 | 0.320 | 0.071 |