Optimization Project in Energy ENT306

Elise Grosjean
Ensta-Paris

Ensta-Paris
Institut Polytechnique de Paris

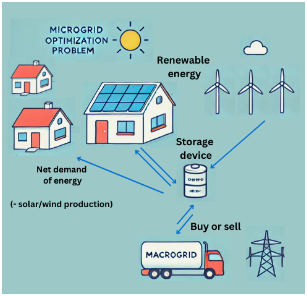
Optimization problem

Optimisation/decision variable :

- $x(s)$: state of charge of the battery at time s, $s = 1, ..., T + 1$.
- $a(s)$: amount of electricity bought on the network ($s = 1, ..., T$).
- $v(s)$: amount of energy sold on the network ($s = 1, ..., T$).

Parameters:

- $d(s)$: net demand of energy (load minus solar/wind production) at time s, $s = 1, ..., T$.
  If $d > 0$, more demand than production of energy from the microgrid
  If $d < 0$, more production of energy from the microgrid
- $P_a(s)$: unitary buying price of energy at time s
- $P_v(s)$: unitary selling price of energy at time s
- $x_{\max}$: storage capacity of the battery.



Horizon: 24 hours.
Stepsize: 1 hour.
Optimization over $T = 24$ intervals.

Optimization problem

Optimisation/decision variable :

- $x(s)$: state of charge of the battery at time s, $s = 1, ..., T+1$.
- $a(s)$: amount of electricity bought on the network ($s = 1, ..., T$).
- $v(s)$: amount of energy sold on the network ($s = 1, ..., T$).

Parameters:

- $d(s)$: net demand of energy (load minus solar/wind production) at time s, $s = 1, ..., T$.
  If $d > 0$, more demand than production of energy from the microgrid
  If $d < 0$, more production of energy from the microgrid
- $P_a(s)$: unitary buying price of energy at time s
- $P_v(s)$: unitary selling price of energy at time s
- $x_{\max}$: storage capacity of the battery.

- Contraints:
  - $\forall s = 1, ..., T$,

    $$x(s+1) = x(s) - d(s) + a(s) - v(s)$$

  - $x(1) = 0$
  - $a(s) \geq 0$, $\forall s = 1, ..., T$
  - $v(s) \geq 0$, $\forall s = 1, ..., T$
  - $0 \leq x(s) \leq x_{\max}$, $\forall s = 1, \dots T+1$.
- Cost function to be minimized:

$$J(x, a, v) = \sum_{s=1}^{T} \Big(P_a(s)a(s) - P_v(s)v(s) \Big)$$

Deterministic or random demand?

- A priori-known demand $\rightarrow$ TP1
- Random demand $\rightarrow$ TP2

One day optimization problem that need to be solved for many days:
Dynamical programming

Main idea behind **dynamic programming:**

- We **parametrize** the problem to be solved $\rightsquigarrow$ a sequence of problems of increasing complexity.
- We look for a **relation** ("dynamic programming principle") between the optimal values of the different problems.

Parameters :

- Initial time $t \in \{1, ..., T+1\}$.
- Initial state-of-charge of the battery $y \in [0, x_{\max}]$.

Initial problem with $t = 1$ and $y = 0$.

parametric problem

Parameterized problem:

$$V(t, y) = \inf_{\substack{x(t), x(t+1), \ldots, x(T+1) \\ a(t), a(t+1), \ldots, a(T) \\ v(t), v(t+1), \ldots, v(T)}} \sum_{s=t}^{T} P_a(s)a(s) - P_v(s)v(s) \qquad (P(t, y))$$

under the constraints:

- $x(s+1) = x(s) - d(s) + a(s) - v(s), \ \forall s = t, \ldots, T$
- $x(t) = y$
- $a(s) \geq 0, \ \forall s = t, \ldots, T$
- $v(s) \geq 0, \ \forall s = t, \ldots, T$
- $0 \leq x(s) \leq x_{\max}, \ \forall s = t, \ldots T + 1.$

The function V is called **value function**; it plays a crucial role, in particular in the treatment of the stochastic version of the problem.

Deterministic model and dynamic programming

- Dynamic programming = Temporal decomposition with smaller sub-problems
- Dynamic programming principle = Recursive relation
  - process ensuring that each subproblem is solved optimally, taking into account future implications and not the previous decisions.
- Interests of the decomposition:
  - reducing complexity
  - optimality of the global solution thanks to Bellman optimality principle
  - clearer implementation
  - ...

Theorem [Dynamic programming principle]

The following holds true:

$$V(t,y) = \inf_{(z,a,v)\in\mathbb{R}^3} \quad P_a(t)a - P_v(t)v + V(t+1,z), \qquad \text{s.t.:} \quad \begin{cases} z,a,v \geq 0 \\ z = y - d(t) + a - v \\ z \leq x_{\max} \end{cases} \qquad (DP(t,y))$$

for all $t \in \{1, \dots T\}$ and $y \in [0, x_{\max}]$ and

$$V(T+1,y) = 0.$$

Deterministic model and dynamic programming

exercice 5: ***FORWARD loop***

*To find **V(t,y)** for any t in [0,T] and any y in [0,x_max] from Dynamic Programming (DP) principle*

$$V(t,y) = \inf_{(z,a,v)\in\mathbb{R}^3} P_a(t)a - P_v(t)v + V(t+1,z),$$

We are here (t,y)

Time

We want to take the good decision by taking into account next step V(t+1,z)

V(T+1,y)=0

Deterministic model and dynamic programming

exercice 5: *FORWARD loop*

To find ***V(t,y)*** *for any t in [0,T] and any y in [0,x_max] from Dynamic Programming (DP) principle*

$$V(t,y) = \inf_{(z,a,v)\in\mathbb{R}^3} P_a(t)a - P_v(t)v + V(t+1,z),$$

We are here (t,y)

Time

We want to take the good decision by taking into account next step V(t+1,z)

From V(T+1,z), we can calculate V(T,y) only if we know y

...

but we can calculate ***V(T,y_j)*** *for different y_j in [0,x_max]*

V(T+1,z)=0

Deterministic model and dynamic programming

exercice 5: ***FORWARD loop***

To find ***V(t,y)*** *for any t in [0,T] and any y in [0,x_max] from Dynamic Programming (DP) principle*

$$V(t,y) = \inf_{(z,a,v)\in\mathbb{R}^3} P_a(t)a - P_v(t)v + V(t+1,z),$$

From V(T+1,z), we can calculate V(T,y) only if we know y

We are here (t,y)

Time

...

V(T+1,z)=0

We want to take the good decision by taking into account next step V(t+1,z)

exercice 3:

Function DP_solve

- *INPUT:*
  t, y_j,
  interpolation of V(t+1,z)
- *OUTPUT: V(t,y)*

but we can calculate ***V(T,y_j)*** *for different y_j in [0,x_max]*

Interpolation!

exercice 2:

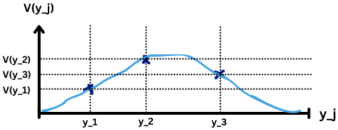
define an interpolation of order 2

$$f(y) \approx \bar{\alpha}_1 + \bar{\alpha}_2 y + \bar{\alpha}_3 y^2$$

exercice 4: ***BACKWARD loop***

To find interpolations of all the values V(t,y_j) for t in [0,T] and y_j in [0,x_max]

Backward loop



$$V(T+1,z)=0 \xrightarrow{T,y_j} V(T,y_j)\ \forall y_j$$

$$= \inf_{z,a,v} P_a(T-1)a - P_v(T-1)v$$

$$s.t.\ z = y_j - d(T) + a - v \in [0, x_{max}]$$

$$\simeq \alpha_{0,T} + \alpha_{1,T} y_j + \alpha_{2,T} y_j^2$$

$$\downarrow$$

$$V(T-1,y_j)\ \forall y_j$$

$$= \inf_{z,a,v} P_a(T-1)a - P_v(T-1)v + \alpha_{0,T} + \alpha_{1,T} z + \alpha_{2,T} z^2$$

$$s.t.\ z = y_j - d(T) + a - v \in [0, x_{max}]$$

$$\simeq \alpha_{0,T-1} + \alpha_{1,T-1} y_j + \alpha_{2,T-1} y_j^2$$

$$\downarrow$$

$$\ldots$$

$$\downarrow$$

$$V(1,y_j)\ \forall y_j$$

$$\simeq \alpha_{0,1} + \alpha_{1,1} y_j + \alpha_{2,1} y_j^2$$

Forward loop

$$V(T+1,z)=0 \xrightarrow{T,y_j} V(T,y_j)\ \forall y_j$$

$$\simeq \alpha_{0,T}+\alpha_{1,T}y_j+\alpha_{2,T}y_j^2$$

$$\downarrow$$

$$V(T-1,y_j)\ \forall y_j$$

$$\simeq \alpha_{0,T-1}+\alpha_{1,T-1}y_j+\alpha_{2,T-1}y_j^2$$

$$\downarrow$$

$$\ldots$$

$$\downarrow$$

$$V(1,y_j)\ \forall y_j$$

$$\simeq \alpha_{0,1}+\alpha_{1,1}y_j+\alpha_{2,1}y_j^2$$

$t, y, (\alpha_{0,t}, \alpha_{1,t}, \alpha_{2,t})$

$\downarrow$

$V(t,y)$ Found while minimizing also the next costs $V(t+1,z)$

Non-deterministic demand

What changes when the demand is random?

Random demand

Random demand and decision process.

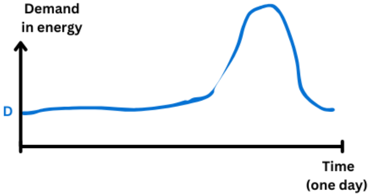
Two additional difficulties:

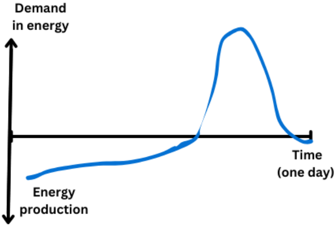
- The demand $d(t)$ is **random**.
- No available **mathematical model** for $d(t)$.

Adaptativity of the decision process.

- At the beginning of the time interval 1, $d(1)$ is revealed.
- Then: decision of the variables $a(1)$ and $v(1)$.
- At the beginning of the time interval 2, $d(2)$ is revealed.
- Then: decision of the variables $a(2)$ et $v(2)$.
- Etc.

$\rightarrow$ What is the probability to get the demand D for a time t = 10 (morning time) ?





Random demand

Random demand and decision process.

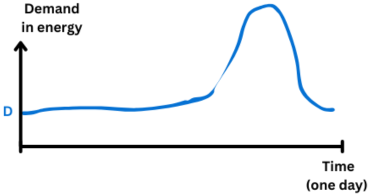
Two additional difficulties:

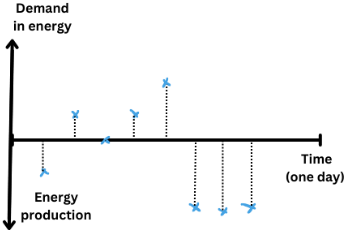
- The demand $d(t)$ is **random**.
- No available **mathematical model** for $d(t)$.

→ What is the probability to get the demand D for a time t = 10 (morning time) ?

We suppose that the demands $d(1)$, $d(2)$,...,$d(T)$, are T random variables. We can try to find **time correlations** between them if they have a temporal dependance in average, and try to identify the probability distribution of each random variable.

→ For that, we now consider that we have access to other demand scenarios from previous days.
Autoregressive (AR) Models: Models where the current value depends linearly on its past values.





How do we construct and evaluate our statistical model?
$\rightarrow$ Control strategies

Previous scenarios

Controls. Decision variables that we can adjust to minimize the cost function

- $a(s)$
- $v(s)$

We call **demand scenario** a vector $(D(s))_{s=1,\dots,T}$.
Two set of scenarios are available:

- **Training set** D_T: history of N_T demand scenarios.
  Used to **build** a probabilistic model for the demand and an appropriate *control strategy*.
- **Test (or Simulation) set** D_S: history of N_S demand scenarios.
  Used to **test** the probabilistic model and the controls choice. Avoid to build biased strategies.

Control strategies

Online and offline phases.

We compute the decision variables in two steps.

1. **Offline phase**. We compute a variable $\mathcal{I}$ which synthesizes all the available information, depending only on D_T and the global parameters ($x_{\max}$, P_a, P_v). For example, $\mathcal{I}$ can contain **statistical data for D_T** and coefficients describing some value function (see after autoregressive processes).

2. **Online phase**. Given a demand scenario $D \in \mathbb{R}^{T+T_0}$ (in D_T), the buying and selling decisions are taken at any time $s = 1, ..., T$ with the help of some function ϕ in the following way:

$$(a(s), v(s)) = \phi\Big(s, x(1), ..., x(s), D(1), ..., D(T_0 + s), \mathcal{I}\Big).$$

Control strategies

We call **control strategy** the pair $(\mathcal{I}, \phi)$.
For example, ϕ can be the function that returns $(a(s), v(s))$ while minimizing the cost $J(x,a,v) = \sum_{s=1}^{T} \Big(P_a(s)a(s) - P_v(s)v(s) \Big)$

Remarks.

- **Feasibility**. The function ϕ must be such that

$$x(s+1) = x(s) + a(s) - v(s) - D(T_0 + s) \in [0, x_{\max}],$$

  for any possible demand scenario.

- The mecanism is **non-anticipative**. At time s, we only use the revealed values of the demand (those until time s) and our a priori knowledge of the demand process, represented by the $\mathcal{I}$.

Control strategies

Cost and evaluation of a control strategy.

- Let us fix $\mathcal{I}$ and ϕ. Given a demand scenario $D \in \mathbb{R}^{T+T_0}$, we denote

$$J_{\mathcal{I},\phi}(D) = \sum_{s=1}^{T} \Big(P_a(s)a(s) - P_v(s)v(s) \Big),$$

where $(a(s))_{s=1,\dots,T}$ and $(v(s))_{s=1,\dots,T}$ are computed by

$$(a(s), v(s)) = \phi\Big(s, x(1), \dots, x(s), D(1), \dots, D(T_0 + s), \mathcal{I}\Big).$$

- We set

$$J_{\mathcal{I},\phi} = \frac{1}{N_S} \sum_{\ell=1}^{N_S} J_{\mathcal{I},\phi}(D_S(\ell, \cdot)).$$

This number measure the efficiency of the strategy. Remember that the history D_S is used only for evaluating the control strategy.

Control strategies

We program a control strategy in three steps:

- **Offline phase:** we program $\mathcal{I}$.
  We use D_T.
- **Online phase:** we program ϕ and $J_{\mathcal{I},\phi}$.
  We use $\mathcal{I}$.
- **Evaluation phase:** we evaluate $J_{\mathcal{I},\phi}$.
  We use $J_{\mathcal{I},\phi}$ and D_S.

Evaluation of control stategies with no apriori knowledge of D_T: $\mathcal{I} = \emptyset$

Exercice 6: Find optimal bound with the optimal cost
Exercice 7: Evaluate a naive strategy where the energy stock remains always the same
Exercice 8: Evaluate a more reasonable strategy

Control strategies

We program a control strategy in three steps:

- **Offline phase:** we program $\mathcal{I}$.
  We use D_T.
- **Online phase:** we program ϕ and $J_{\mathcal{I},\phi}$.
  We use $\mathcal{I}$.
- **Evaluation phase:** we evaluate $J_{\mathcal{I},\phi}$.
  We use $J_{\mathcal{I},\phi}$ and D_S.

Evaluation of control stategies with no apriori knowledge of D_T: $\mathcal{I} = \emptyset$

Exercice 6: Find optimal bound with the optimal cost

Exercice 7: Evaluate a naive strategy where the energy stock remains always the same

Exercice 8: Evaluate a more reasonable strategy

Control strategies

A lower bound for the cost

Given a demand scenario $D \in \mathbb{R}^{T+T_0}$, we denote $J_{\mathrm{anti}}(D)$ the optimal cost obtained, assuming that D is entirely known. We denote

$$J_{\mathrm{anti}} = \frac{1}{N_S} \sum_{\ell=1}^{N_S} J_{\mathrm{anti}(D_S(\ell,\cdot))}.$$

The number J_{anti} is a lower bound for the evaluation cost of any (feasible and non-anticipative) strategy.

Exercise 6

Write a function `lower_bound` which computes J_{anti}. To this purpose, use the functions already written in exercise 1. Pay attention to the shifiting of time indices.

Autoregressive process

Shifting of the time index.

The two available histories of demand scenarios contain T_0 values of the demand from the "previous day", corresponding to the time intervals 0, -1, $-2,\ldots,-(T_0-1)$. They can be used to approximate any other time t

On the computer: a demand scenario is a vector of size $T+T_0$. The training and simulation sets are matrices with $(T+T_0)$ columns and respectively N_T and N_S rows.

We "get access" to the demand at time t, for the scenario ℓ with

$$D_T(\ell, t+T_0) \qquad D_S(\ell, t+T_0).$$

Control strategies

1. **The naive strategy**.
   - Offline phase: $\mathcal{I} = \emptyset$. We do not exploit D_T.
   - Online phase: at time s, given the demand $d(s)$, we chose

$$(a(s), v(s)) = \begin{cases} (d(s), 0), & \text{si } d(s) \geq 0, \\ (0, -d(s)), & \text{si } d(s) \leq 0 \ . \end{cases}$$

Exercise 7

Verify that the naive strategy is non-anticipative and feasible. Write a function `naive_online` which computes the decision variables and the cost associated with a demand scenario (given in input). Write a function `naive_eval` which computes the cost of the cost of the strategy.

Control strategies

2. The reasonable strategy

- Offline phase: $\mathcal{I} = \emptyset$. Again, we do not exploit D_T.
- Online phase: at time s, given the demand $d(s)$ and the state of charge $x(s)$:
  - if $d(s) \geq 0$: we dip into the reserve $x(s)$ and we buy electricity if $d(s) \geq x(s)$.
  - If $d(s) \leq 0$: we stock energy in the battery as much as possible; if $d(s) \leq x(s) - x_{\max}$, the surplus is sold.

Exercise 8

Verify that the strategy is non-anticipative and feasible. Write two function `raisonnable_online` and `raisonnable_eval` implementing and testing this strategy.

Autoregressive processes

Generalities.
Compute $\mathcal{I}$ offline.

- We look for a stochastic model describing **faithfully** the evolution of the demand with respect to time.
- This model should be of **reasonable complexity**, so that it can be exploited numerically.
- We are interested in **autoregressive processes**, for which an approach by dynamic programming can be implemented.

For a time t, we do not have access to demand $d(t+1), \ldots, d(T)$ that we our going to approximate with $\mathcal{I}$.

Autoregressive processes

Generalities.

Offline phase of the control strategy: we program $\mathcal{I}$.
We use D_T.

Autoregressive processes

Numerical approximation. We propose the following method to approximate an autoregressive process $d(t)$ of order I. We proceed in two steps:

- For all $t = 1, \ldots, T$, compute the solution $(\bar{\gamma}, \bar{\beta}_1, \ldots, \bar{\beta}_I)$ to

$$\inf_{\gamma,\beta_1,\ldots,\beta_p\in\mathbb{R}} \sum_{\ell=1}^{N_T} \Big(D_T(\ell, t+T_0) - (\gamma + \sum_{i=1}^{I} \beta_i D_T(\ell, t+T_0-i)) \Big)^2$$

  We set $\gamma(t) = \bar{\gamma}$, $\beta_1(t) = \bar{\beta}_1$,...,$\beta_I(t) = \bar{\beta}_I$.

- We sample the variable $\varepsilon(t, \ell)$, given by

$$\varepsilon(\ell, t) = D_T(\ell, t+T_0) - \Big(\gamma(t) + \sum_{i=1}^{I} \beta_i(t) D_T(\ell, t+T_0-i) \Big).$$

Autoregressive processes

Definition

We call white noise a sequence of independent random variables $(\varepsilon(t))_{t=1,\dots}$ with null expectation.

Definition

We call the process $d(t)$ an autoregressive process of order $l \in \mathbb{N}$ if there exist deterministic coefficients $\gamma(t)$, $\beta_1(t)$,...,$\beta_l(t)$ and a white noise $(\varepsilon(t))_t$ such that:

$$d(t) = \gamma(t) + \beta_1(t)d(t-1) + \dots + \beta_l(t)d(t-l) + \varepsilon(t).$$

$\rightarrow$ Allows us to approximate any demand from the previous ones.

Autoregressive processes

Processes of order 0.

- Construction of $\mathcal{I}$ =the coefficients

  We suppose that the demands $d(1)$, $d(2)$,...,$d(T)$, are T **independent** random variables. Thus we do not need to identify any correlation between them, but we need to identify the probability distribution of each random variable.

  Given t, we approximate $d(t)$ with a random variable which can take N_E different values with probability $p := 1/N_E$. This values are obtained by **sampling**.

$$d(t) = \gamma(t) + \varepsilon(t).$$

- Evaluation of $\mathcal{I}$

  We test the quality of γ by computing ε with D_S or the averaged cost.

Autoregressive processes

Sampling.

Let $d(t) \in \mathbb{R}^{N_T}$ be a given vector, that we need to sample with N_E values. The result of the procedure is a vector $\gamma(t) \in \mathbb{R}^{N_E}$.

- To simplify, we will assume that $q := N_T/N_E$ is an integer.
- Let $\tilde{d}(t)$ be the vector obtained by sorting the values of $d(t)$, from the smallest value to the largest one.
- We define $\gamma(t)$ as follows:

$$\gamma(t,1) = \frac{1}{q}\sum_{\ell=1}^{q} \tilde{d}(t,\ell), \quad \gamma(t,2) = \frac{1}{q}\sum_{\ell=q+1}^{2q} \tilde{d}(t,\ell), \; \ldots$$

$$\gamma(t,N_E) = \frac{1}{q}\sum_{\ell=N_T-q+1}^{N_T} \tilde{d}(t,\ell).$$

Autoregressive processes

Exercise 9

- Write a fonction `sample` realising the sampling of an arbitrary vector h in N_E values. Use the function `sort` of Python.
- Write a function `sample_training_set` with output a matrix $E \in \mathbb{R}^{N_E \times T}$ such that each column contains the sampled values of the vectors

$$D_T(:, T_0 + 1), \quad D_T(:, T_0 + 2), \ldots \quad D_T(:, T_0 + T).$$

Autoregressive processes

Exercise 10

Write a function `auto_reg_1` realizing the approximation of $d(t)$ as an autoregressive process of order 1
Output variables: $\gamma \in \mathbb{R}^T$, $\beta_1 \in \mathbb{R}^T$, $E \in \mathbb{R}^{N_E \times T}$.

Optional. Write a function `auto_reg` which realizes the approximation of $d(t)$ by an autoregressive process of arbitrary order (given as input variable).

Autoregressive processes

Predictive model: Φ

Phase offline. Approximation of $d(t)$ with an autoregressive process of order 1, with the help of coefficients γ and β_1.

Phase online. Let t be the current time step. Let x_t denote the current state-of-charge of the battery and let d_t denote the demand at time t (e.g. $d_t = D_S(t)$).

1. Prediction. Compute $(D_p(s))_{s=t,\dots T}$ as follows:

$$
\begin{aligned}
D_p(t) &= d_t, \\
D_p(t+1) &= \gamma(t+1) + \beta_1(t+1)D_p(t), \\
D_p(t+2) &= \gamma(t+2) + \beta_1(t+2)D_p(t+1), \\
&\dots \\
D_p(T) &= \gamma(T) + \beta_1(T)D_p(T-1).
\end{aligned}
$$

Predictive method

2. Optimization. We solve $V(t, x_t)$:

$$\inf_{\substack{x(t),\dots,x(T+1)\\ a(t),\dots,a(T)\\ v(t),\dots,v(T)}} \sum_{s=t}^{T} P_a(s)a(s) - P_v(s)v(s)$$

$$\text{s.t.} \begin{cases} x(s+1) = x(s) + a(s) - v(s) - D_p(s), & s = t, \dots T \\ x(t) = x_t \\ a(s) \geq 0, & s = t, \dots, T \\ v(s) \geq 0, & s = t, \dots, T \\ 0 \leq x(s) \leq x_{\max}, & s = t, \dots, T \end{cases}$$

Let $\bar{x}(t), \dots, \bar{x}(T+1)$, $\bar{a}(t), \dots, \bar{a}(T)$, $\bar{v}(t), \dots, \bar{v}(T)$ be a solution. We take:

$$a(t) = \bar{a}(t), \quad v(t) = \bar{v}(t).$$

Predictive method

Exercise 11

Implement the predictive method described above.

Remarks

Overfitting! Increase the order does not mean that the approximation will necessarily be more precise. For noisy data, a small order is often preferred to avoid modelling the noise.

Instead of time-dependent coefficients, we could have taken $\gamma, \beta_1 ... \beta_l \in \mathbb{R}$.

Dynamic programming

Case of an autoregressive process of order 0.

We suppose that the demande $d(t)$ is described by an autoregressive process of order 0, that is, all the random variables $d(1),...,d(T)$ are independent.

We suppose that a matrix $(D(j,t))_{\substack{j=1,...,N_E \\ t=1,...,T}}$ is given and that

$$\mathbb{P}\left[d(t) = D(j,t)\right] = \frac{1}{N_E},$$

for all $j = 1, ..., N_E$ and for all $t = 1, ..., T$.

Dynamic programming

From now on, we need to work with two value functions:

- $V(t,x)$: the expectation of the optimal cost (from t to T), with initial state-of-charge x at time t, before the demand $d(t)$ is revealed.
- $\tilde{V}(t,x,d_t)$: the expectation of the optimal cost (from t to T), with initial state-of-charge x at time t, conditionally to $d(t) = d_t$.

Dynamic programming

Theorem

The following holds true.

- For all $x \in [0, x_{\max}]$, $V(T+1, x) = 0$.
- For all $t = 1, ..., T$, for all $x \in [0, x_{\max}]$,

$$V(t,x) = \frac{1}{N_E} \sum_{j=1}^{N_E} \tilde{V}(t, x, D(j,t)).$$

- For all $t = 1, ..., T$, for all $x \in [0, x_{\max}]$,

$$\tilde{V}(t,x,d) = \inf_{(z,a,v) \in \mathbb{R}^3} P_a(t)a - P_v(t)v + V(t+1, z), \quad (DP(t,x,d))$$

$$\text{sous la contrainte : } \begin{cases} z = x + a - v - d, \\ 0 \leq z \leq x_{\max}, \\ a \geq 0, \ \ v \geq 0. \end{cases}$$

Dynamic programming

Phase offline: numerical approximation of $V(\cdot,\cdot)$.

The mechanism is similar to the one seen in the deterministic framework.
Let $t \in \{1, ..., T\}$. Let us suppose $V(t+1,\cdot)$ that is known and represented as a polynomial function.

- We calculate $\tilde{V}(t, x_j, D(k,t))$ for all $j = 1, ..., J$ and for all $k = 1, ..., N_E$, by solving $(DP(t, x_j, D(k,t)))$.
- We calculate $V(t, x_j)$ for all $j = 1, ..., J$.
- We approximate the full function $V(t,\cdot)$ by approximation.

Phase online : at time t, when the demand $d(t)$ has been revealed, we solve $(DP(t, x, d))$, with x the current state-of-charge at time t and $d = d(t)$.

Dynamic programming

Exercise 12

Implement the control strategy induced by the dynamic programming principle with the auto-regressive model of order zero.

Dynamic programming

Case of a first-order autoregressive process.
We suppose that the demand $d(t)$ is described by a first-order autoregressive process, that is:

$$d(t) = \gamma(t) + \beta_1(t)d(t-1) + \varepsilon(t),$$

where $(\varepsilon(t))_{t=1,\dots,T}$ is a white noise.

We suppose that a matrix $(E(k,t))_{\substack{k=1,\dots,N_E \\ t=1,\dots,T}}$ is given and

$$\mathbb{P}\Big[\varepsilon(t) = E(k,t)\Big] = \frac{1}{N_E},$$

for all $k = 1, \dots, N_E$, and for all $t = 1, \dots, T$.

Dynamic programming

We consider two value functions:

- $V(t, x, d_{t-1})$: the optimal expected cost (from t to T), with state-of-charge x at time t, knowing that $d(t-1) = d_{t-1}$, before that $d(t)$ is revealed.
- $\tilde{V}(t, x, d_t)$: the optimal expected cost, with state-of-charge x at time t, knowing that $d(t) = d_t$.

Dynamic programming

Theorem

The following holds true.

- For all $x \in [0, x_{\max}]$, $V(T+1, x, d_T) = 0$.
- For all $t = 1, ..., T$, for all $x \in [0, x_{\max}]$,

$$V(t, x, d_{t-1}) = \frac{1}{N_E} \sum_{k=1}^{N_E} \tilde{V}(t, x, \gamma(t) + \beta_1(t) d_{t-1} + E(k, t)).$$

- For all $t = 1, ..., T$, for all $x \in [0, x_{\max}]$,

$$\tilde{V}(t, x, d_t) = \inf_{(z,a,v) \in \mathbb{R}^3} P_a(t)a - P_v(t)v + V(t+1, z, d_t),$$

$$\text{subject to: } \begin{cases} z = x + a - v - d_t, \\ 0 \leq z \leq x_{\max}, \\ a \geq 0, \;\; v \geq 0. \end{cases} \qquad (DP(t, x, d_t))$$

Dynamic programming

Remark. The value function (at time t) depends on two variables. We can seek for an approximation with a second-order polynomial of the form:

$$\begin{aligned} V(t, x, d_{t-1}) = &\alpha_1(t) + \alpha_2(t)x + \alpha_3(t)d_{t-1} \\ &+ \alpha_4(t)x^2 + \alpha_5(t)xd_{t-1} + \alpha_6(t)d_{t-1}^2. \end{aligned}$$

Dynamic programming

Autoregressive Processus of order 0 vs 1

	AR 0	AR 1
Formula	$d(t) = \gamma(t)$	$d(t) = \gamma(t) + \beta(t)d(t-1) + \varepsilon(t)$
Variability	No time relation	Linear dependance in time + white noise
Sampling	Size $N_E \times T$, $P = 1/N_E$	Generation of N_E trajectories with noise
Computing $V(t, y)$	Average over N_E values of $d(t)$	Interpolation over all the N_E trajectories
Interpolation	Grid of y_j	Grid of y_j and $d_k(t)$