Computing an Optimal Control Policy for an Energy Storage

Pierre Haessig, Thibaut Kovaltchouk,
Bernard Multon, Hamid Ben Ahmed, and Stéphane Lascaud

EDF R&D LME, ENS Cachan SATIE
contact : pierre.haessig@ens-cachan.fr

EuroSciPy 2013, Brussels, August 24th 2013

Companion article: http://publications.pierreh.eu

Outline of the presentation

Outline of the presentation

My background

- Curriculum in Electrical Engineering and Control Theory
  $\rightarrow$ *Matlab/Simulink kingdom*
- PhD student on Electricity Storage in relation to Wind Energy
- Python for all my simulation and visualisation work
  (and a bit of `R` for time series analysis)

StoDynProg: a Dynamic Optimization problem solving code

Working on the management of Energy Storage with Wind Power, I've progressively discovered that:

- my problems fall in the class of *Dynamic Optimization* (a quite specific problem structure)
- the *Dynamic Programming* approach exists to solve them.
- basic DP algorithms are "too simple to be worth implementing" !!

StoDynProg: a Dynamic Optimization problem solving code

Working on the management of Energy Storage with Wind Power, I've progressively discovered that:

- my problems fall in the class of *Dynamic Optimization* (a quite specific problem structure)
- the *Dynamic Programming* approach exists to solve them.
- basic DP algorithms are "too simple to be worth implementing" !!

So I've started a generic code to solve all *my* problems and hopefully other Dynamic Optimization problems as well.

I wanted to challenge this "genericity claim" by trying it on a *different* problem: I took it from a topic of interest of my research group: Ocean Power Smoothing (with an Energy Storage).

Outline of the presentation

1. Intro
2. Example of Ocean Power Smoothing
3. solving Dynamic Optimization with Dynamic Programming

Ocean Wave Energy Harvesting

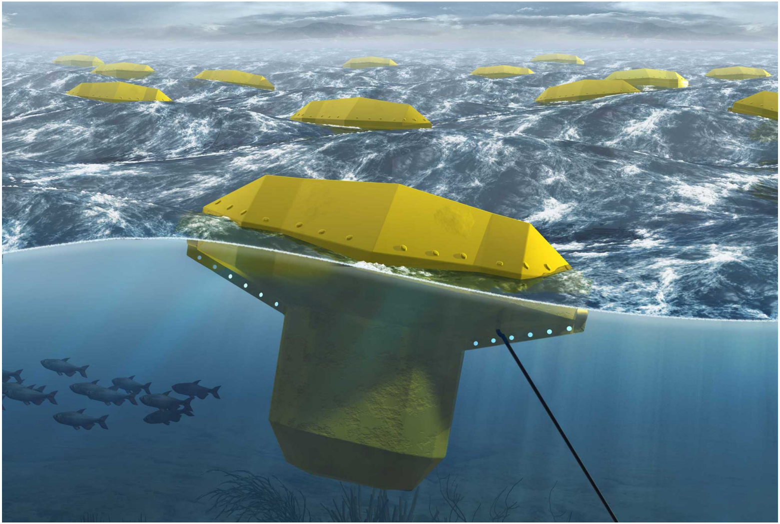
(CC-BY-NC picture by polandeze)
www.flickr.com/photos/polandeze/3151015577

Harvesting electric power from Ocean Waves with "big machines" is an active area of Research & Development.

There are no industrialized devices yet (unlike for wind & sun), but rather *a wide variety of prototypes machines:*
Wave Energy Converters

E.ON P2 Pelamis, July 2011 http://www.pelamiswave.com

Ocean Energy Converter: the SEAREV

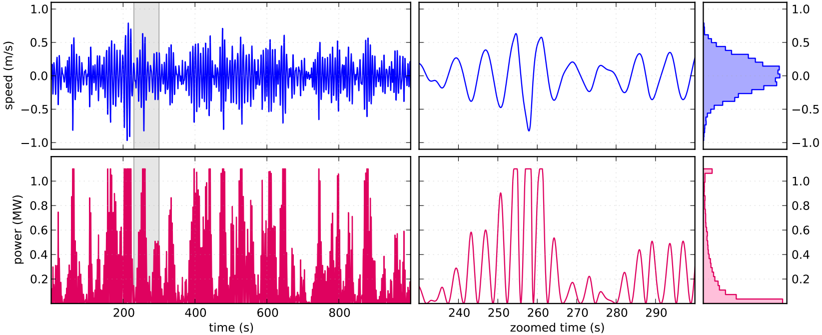


Hydro-mechanical design from Centrale Nantes.

My group involved in the electric generator design.

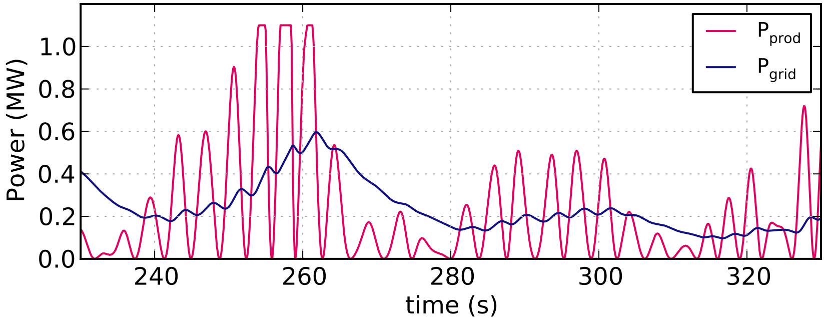
Ocean Energy Converter: the SEAREV

a highly fluctuating output



SEAREV is a giant double-pendulum that swings with the waves.
An electric generator “brakes” the inner wheel to generates power ($P_{prod} = T(\Omega) \times \Omega$).

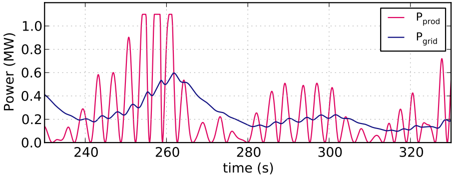
Power smoothing

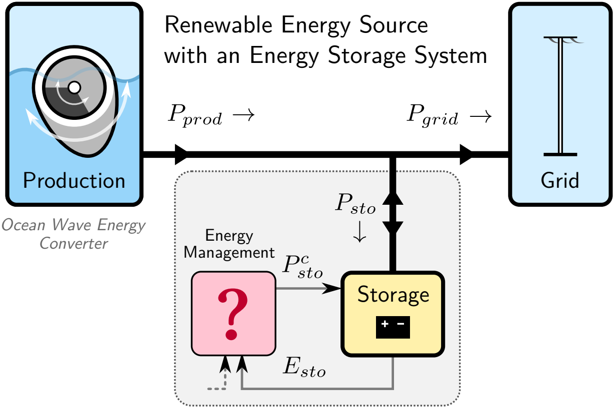


Objective of this application

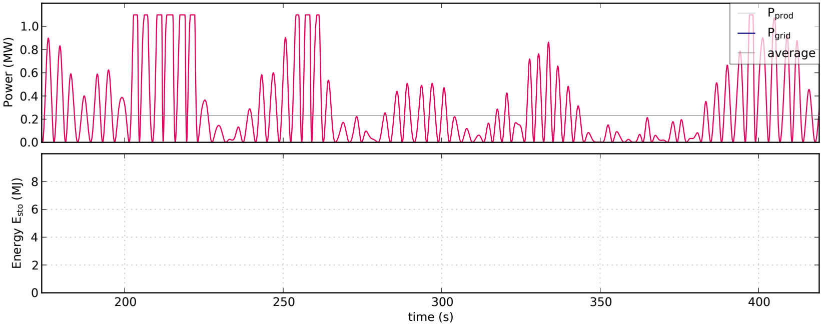
I want to *smooth out the variations* of the power production.
This requires an **energy buffer** to store the difference
$(P_{prod} - P_{grid})$.

Power smoothing using an Energy Storage

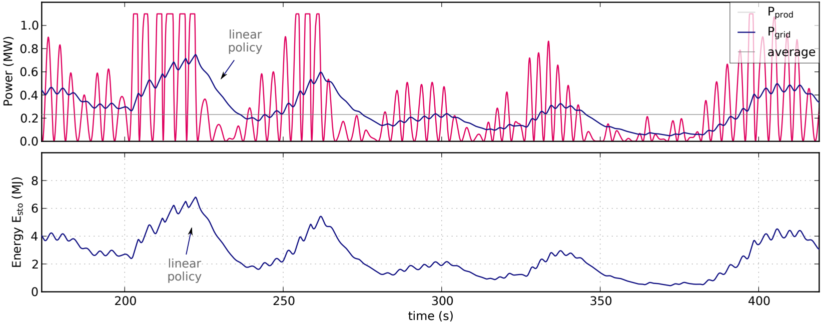




Power smoothing: control of the Energy Storage



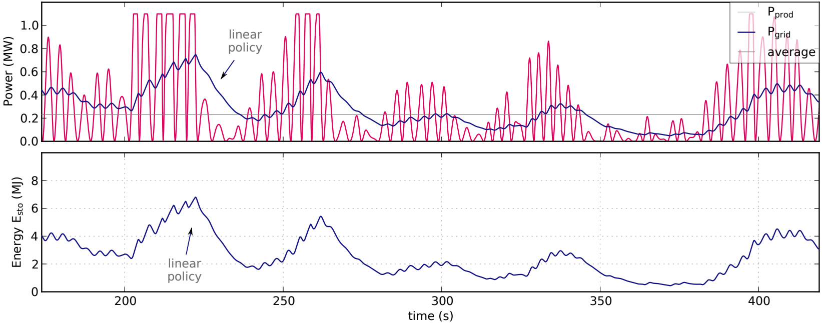
Power smoothing: control of the Energy Storage



First, using a simple control law (∼policy)

. . . quite good result but storage is underused → could do better.

Power smoothing: control of the Energy Storage

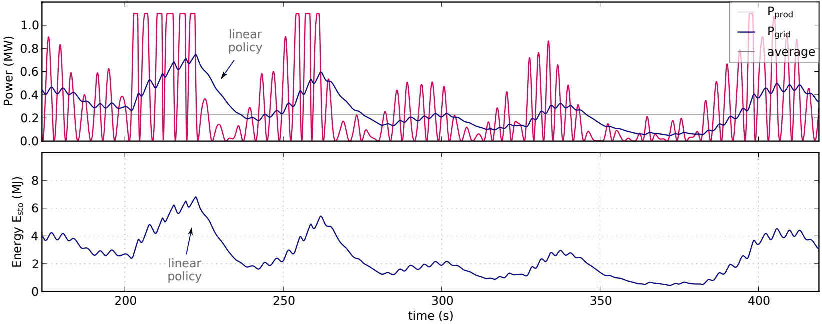


"Doing better" is defined with an additive cost function which penalizes P_{grid} variations:

$$J = \frac{1}{N}\mathbb{E}\left\{\sum_{k=0}^{N-1} cost(P_{grid}(k) - P_{avg})\right\} \quad \text{with } N \to \infty$$

cost J should be minized.

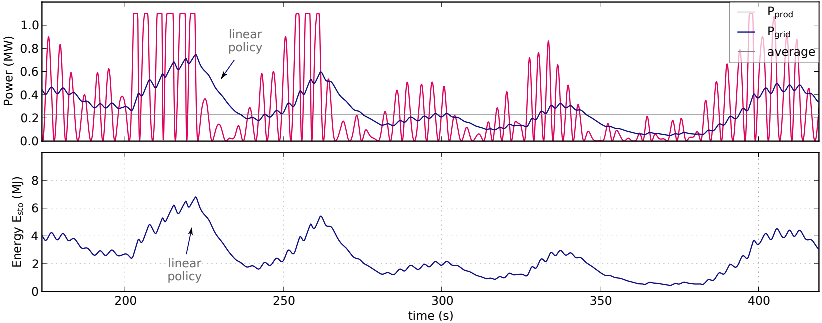
Power smoothing: control of the Energy Storage



Controlling the storage (choosing P_{grid} at each time step) in order to minimize a cost function is a ***Stochastic* Dynamic Optimization** problem

(also called Stochastic Optimal Control)

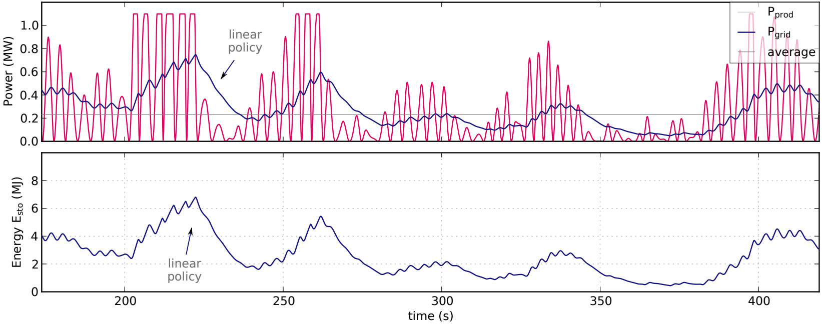
Power smoothing: control of the Energy Storage



Dynamic Programming (Richard Bellman, ~1950) teaches us that the optimal control is a *state feedback* policy:

$$P_{grid}(t) = \mu(x(t)) \quad \text{with } x = (E_{sto}, speed, accel)$$

Power smoothing: control of the Energy Storage

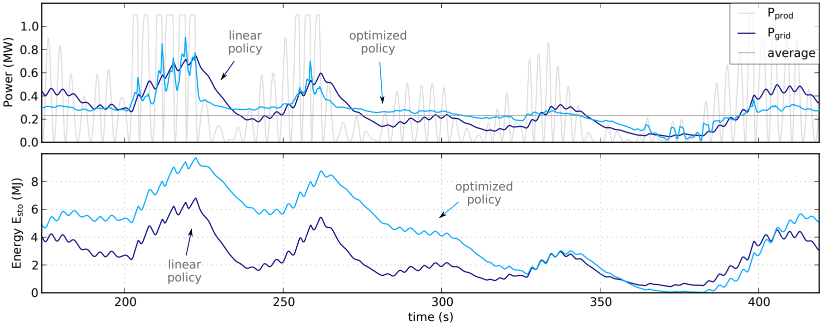


Dynamic Programming (Richard Bellman, ~1950) teaches us that the optimal control is a *state feedback* policy:

$$P_{grid}(t) = \mu(x(t)) \quad \text{with } x = (E_{sto}, speed, accel)$$

And DP gives us *methods to compute* this policy function μ...

Power smoothing: control of the Energy Storage



And now applying the optimal feedback policy μ^*, the standard deviation of the power injected to the grid is reduced by ∼20 % compared to the heuristic policy.

This improvement just comes from a smarter use of the stored energy.

Outline of the presentation

Dynamic Programming equation

In the end, the optimization problem turns into solving the DP equation:

$$J + \tilde{J}(x) = \min_{u \in U(x)} \mathbb{E}_w \Big\{ \underbrace{cost(x, u, w)}_{\text{instant cost}} + \underbrace{\tilde{J}(f(x, u, w))}_{\text{cost of the future}} \Big\}$$

u is control and w is random perturbation, using generic notations

- It is a *functional* equation: should be solved for all x
- The optimal policy $\mu : x \mapsto u$ appears as the `argmin`.

The DP equation is solved on a **discrete grid** over the state space. With $x \in \mathbb{R}^n$, $\tilde{J}$ and μ are computed as n-d numpy arrays.

Equation solving, Multilinear interpolator

The resolution is done purely in Python. Basically a giant `for` loop with an `argmin` inside.

- `numpy` for handling arrays, with a good amount of vectorization
- `itertools` to iterate over the state space grid (of arbitrary dimension)
- (`introspect` for some signature analysis magic)

Equation solving, Multilinear interpolator

The resolution is done purely in Python. Basically a giant `for` loop with an `argmin` inside.

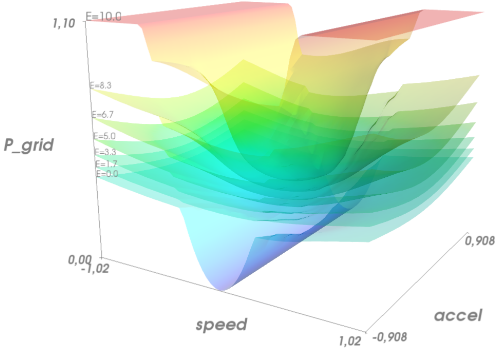
- `numpy` for handling arrays, with a good amount of vectorization
- `itertools` to iterate over the state space grid (of arbitrary dimension)
- (`introspect` for some signature analysis magic)

Extremely useful code reuse: a multilinear interpolator by *Pablo Winant* (within its `dolo` project: `github.com/albop/dolo`). Uses `Jinja` templates to generate Cython code for dimension 1-5.

Learning of this project (on scipy ML) saved me weeks, if not months !

Conclusion

`mayavi.surf` plot of the optimal storage policy



Code should be soon on GitHub (`github.com/pierre-haessig`). Decent Sphinx doc with examples (and complete code for SEAREV example), but ridiculous test coverage.