From Revenue Management to Higher Automation via Machine Learning and Optimization

Roberto Battiti

LION-lab

University of Trento, Italy

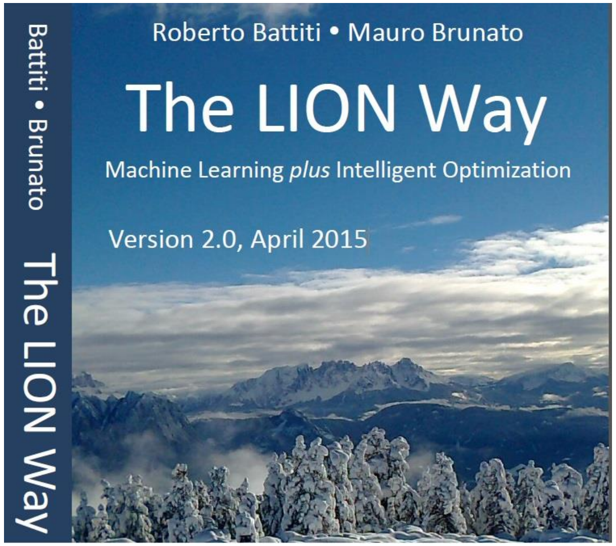
UNIVERSITY OF TRENTO - Italy

Slides of Keynote talk at REVME, EHL Lausanne, Dec 3, 2019

I will need to skip all references, if you want more details you can send me email:

Roberto.Battiti@unitn.it

https://intelligent-optimization.org/LIONbook/



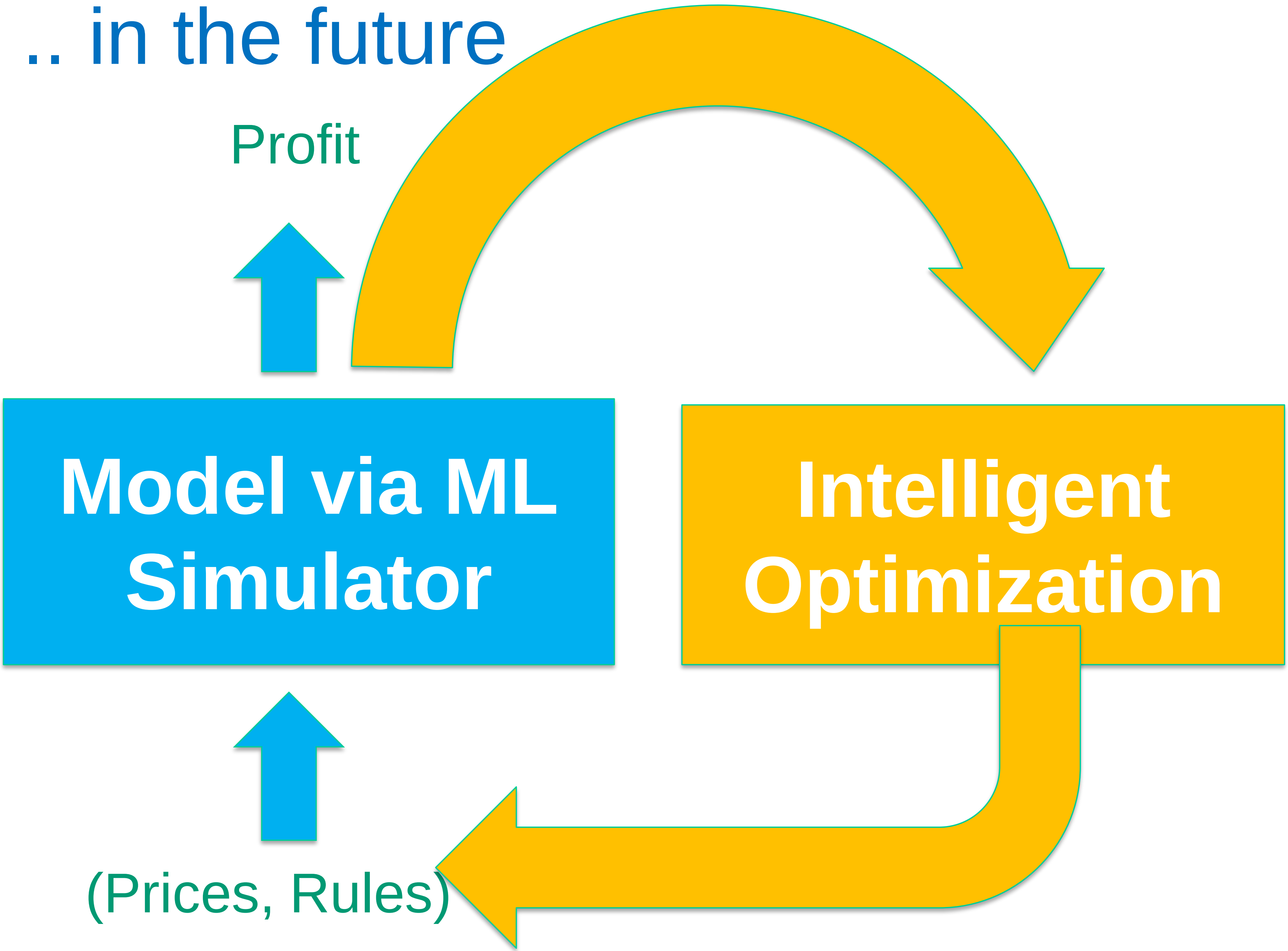
Now…

Profit

Hotel/ Customers

(Prices, Rules)

.. in the future



RM is an old field…BUT

If one uses traditional **Mathematical Programming/Optimization**…

- Specific (often unrealistic) **assumptions** (e.g. **linear** price elasticity of demand)
  - More flexibility
- **Exploding CPU times** (e.g. dynamic programming)
  - Intelligent optimization heuristics
- **Multiple-objectives** (e.g. short-term vs long-term)
  - Opportunities for (machine) learning the correct function to be optimized (not only short-term profit)

E.g., assumption that customers respond in a **linear** manner to prices not always true!

La Californienne

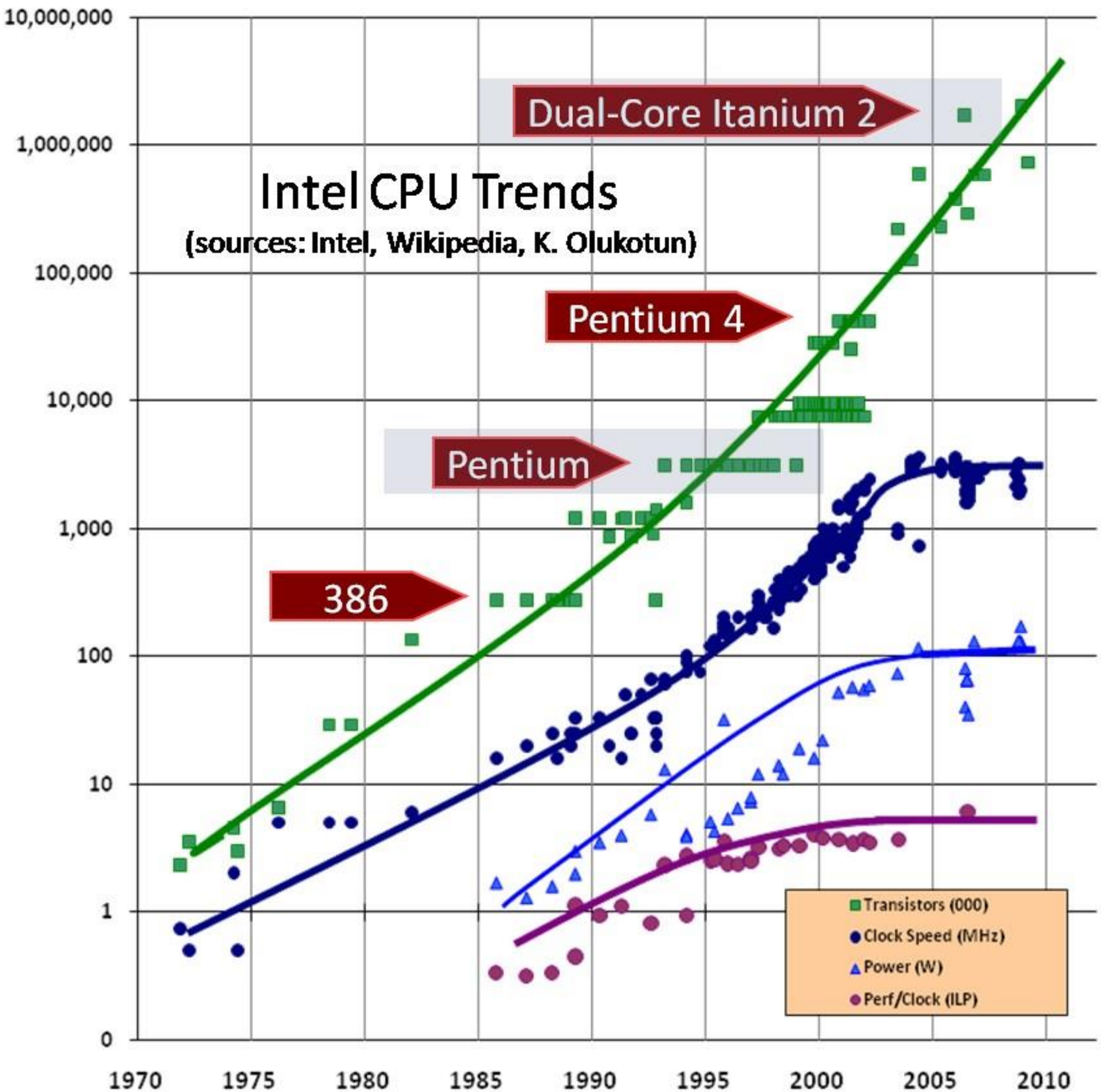
34mm Rolex Watch

CHF6,501.30

Color: Cerulean/Venice

Imagine you drop the price to 1,000 CHF
Probably customers think it is a fraud and do **not** buy!!

What changed from the seventies?



2x transistor per chip

Every 18 months

10,000,000 X

faster

in 30 years

CPU 2.0 GHz

2,000,000,000

cycles / second

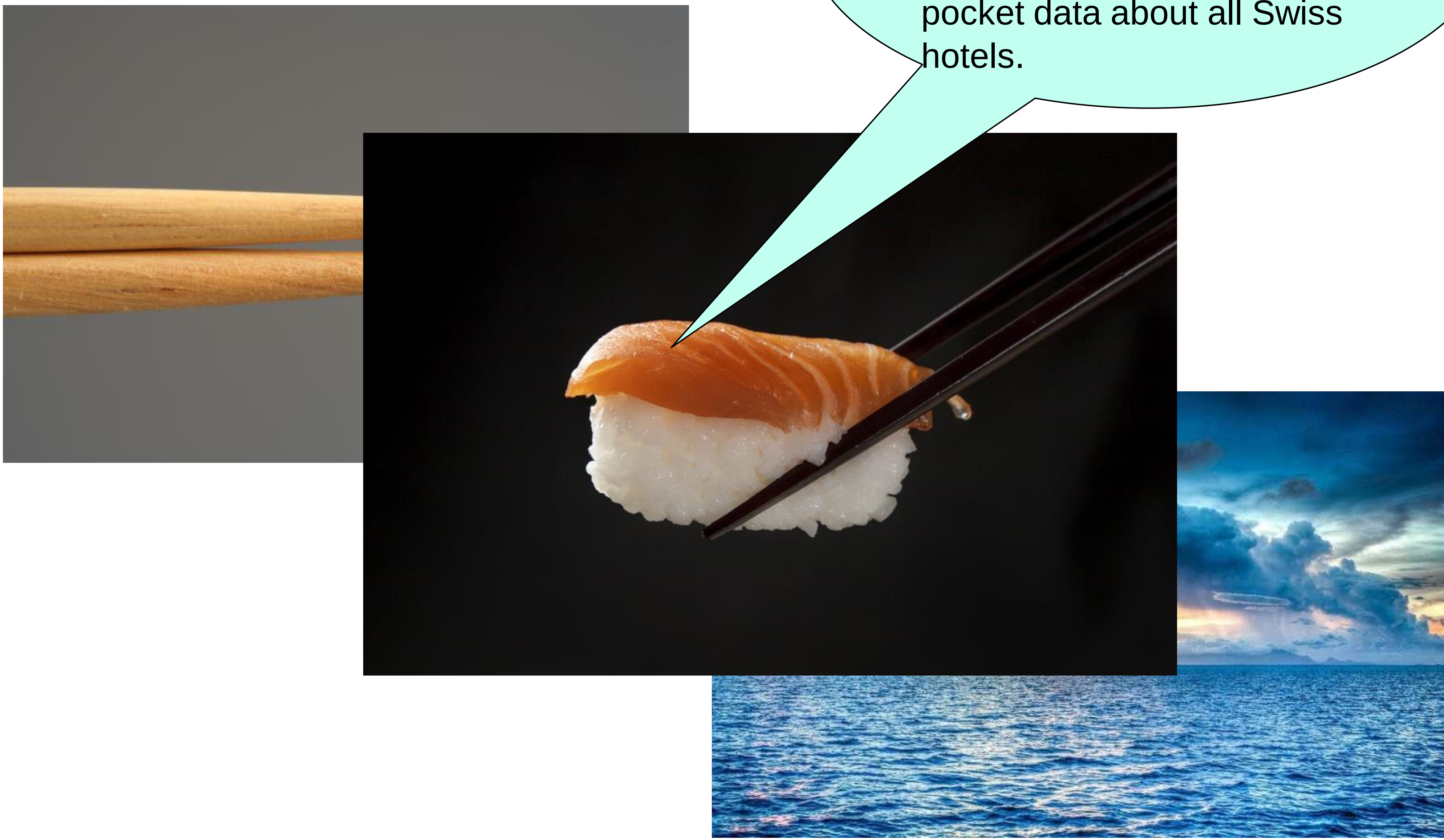
Big data or small data?

(consider a single medium-size hotel)

Big data or small data?

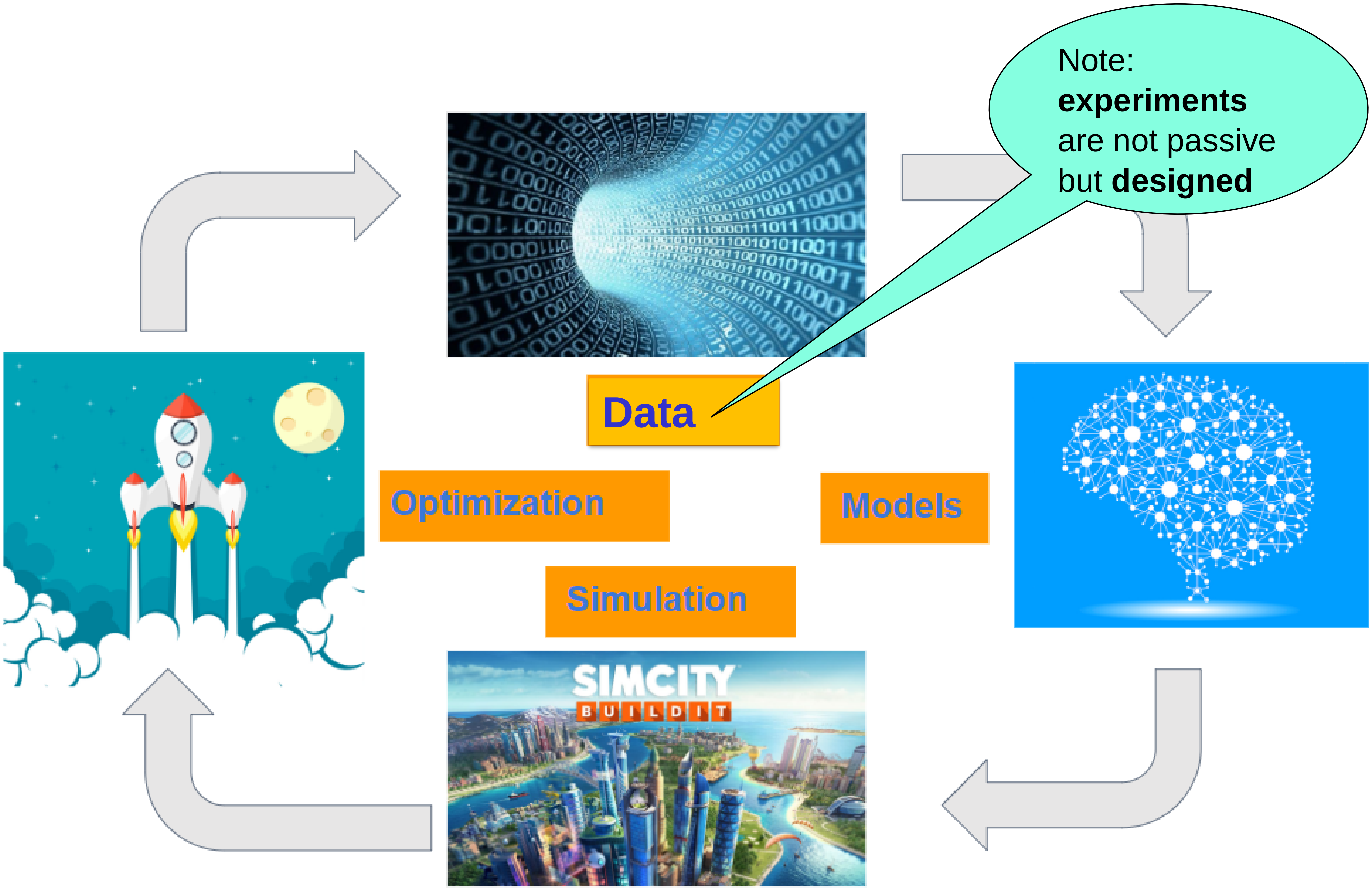
(single medium-size hotel)

Let’s not exaggerate…Data of a single hotel **are not big data** if properly filtered!
You can easily put in your pocket data about all Swiss hotels.



ML + Simulation + Optimization

The real power for innovation in Revenue Management comes from the combination


Note:
experiments are not passive but **designed**
Data
Optimization
Models
Simulation
SIMCITY
BUILDIT

What’s behind

- use **data** to build models and extract knowledge

Machine learning or learning from data

- exploit **knowledge** to **automate** the discovery of improving solutions

Optimization (automated problem solving)

- connect insight to **decisions and actions**.

Prescriptive analytics (much more than BI)

A “zip” of the history of AI - NN - ML

Symbolic AI
(up to 1985)

Syb-symbolic
Neural nets

Statistics/Machine learning. Deep learning…

Symbols
Logic
Expert systems
Explicit symbolic programming
Inference, search algorithms
AI programming languages
Rules, Ontologies, Plans, Goals…

Knowledge in parameters
Dynamical systems
Neural Nets / Backprop
Bayesian learning
Deep learning
Connectionism

Learning from Data *and* Machine Learning

If you show a picture to a three-year-old and ask if there is a tree in it, you will likely get the correct answer. If you ask a thirty-year-old what the definition of a tree is, you will likely get an inconclusive answer. We didn't learn what a tree is by studying the mathematical definition of trees. We learned it by looking at trees. In other words, we learned from 'data'.

Yaser Abu-Mostafa

A zip of the history of AI - NN - ML

Symbolic AI (up to 1985) → Syb-symbolic Neural nets → Statistics/Machine learning. Deep learning…

Easier to debug
Easier to ==explain==
Easier to control
Not so Data-based
More useful for **explaining** people's thought
Better for abstract problems
Fragile
Needs knowledge elicitation
Curse of dimensionality

More robust against noise
Faster (from inputs to outputs)
Less knowledge upfront
Easier to scale up
Data-based
More useful for connecting to neuroscience
Better for perceptual problems

Why do we need models? Why surrogates?

Three ways of building **models**

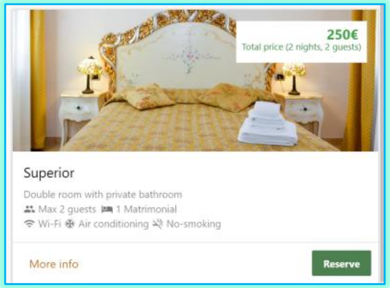
output

Buy or not

Model

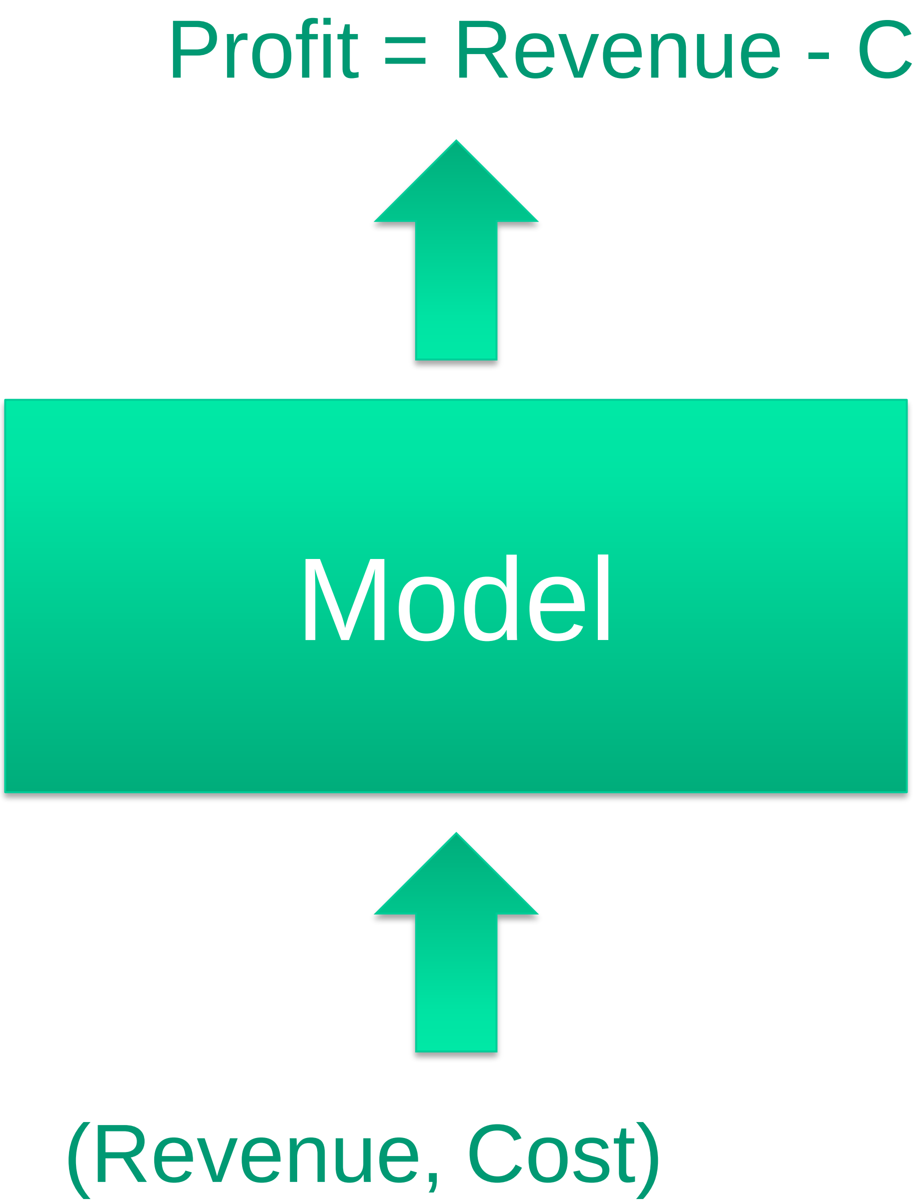
- what-if questions
- predictions
- Input optimization

Room type and characteristics
Price
Context
Advertisement



input

1) **Explicit exact and rigid** models



e.g., Physics: Boyles's law:

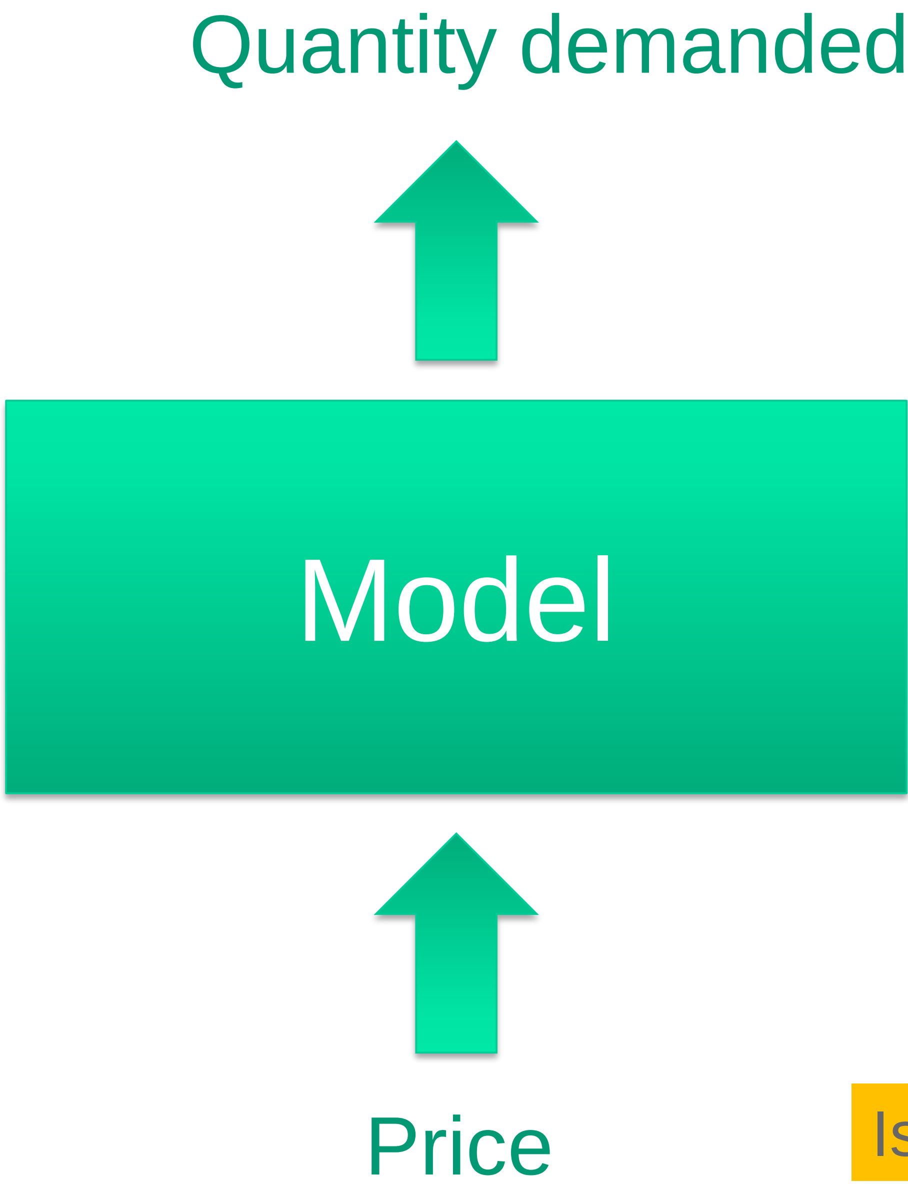
"For a fixed mass of gas, at a constant temperature, the product (pressure x volume) is a constant."

$PV = N\,k\,T$

Why do we need other models?

2) **Parametric**, with statistics

Ronald Fisher in 1913



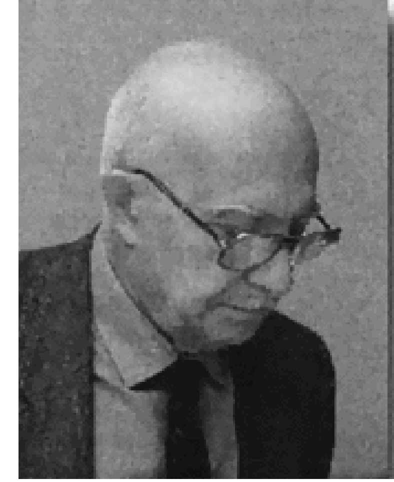
Price elasticity of demand =

$$\frac{\text{Proportionate change in quantity demanded}}{\text{Proportionate change in price}} = \frac{\frac{\Delta Q}{Q} \times 100\%}{\frac{\Delta P}{P} \times 100\%} = \frac{\frac{\Delta Q}{Q}}{\frac{\Delta P}{P}}$$

e.g., Maximum likelihood estimation

Is this related to Machine Learning?

3) ***Non*-parametric** models, neural nets, modern ML (1960++, 1985, 2010)



Eduardo Caianiello, 1961

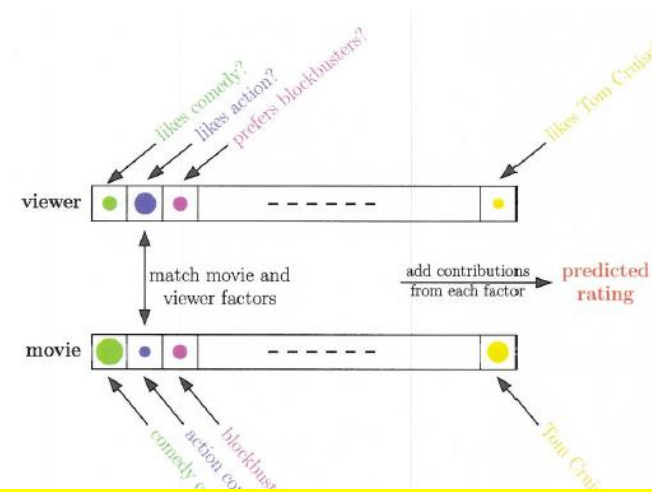
Recommendation

↑

Model

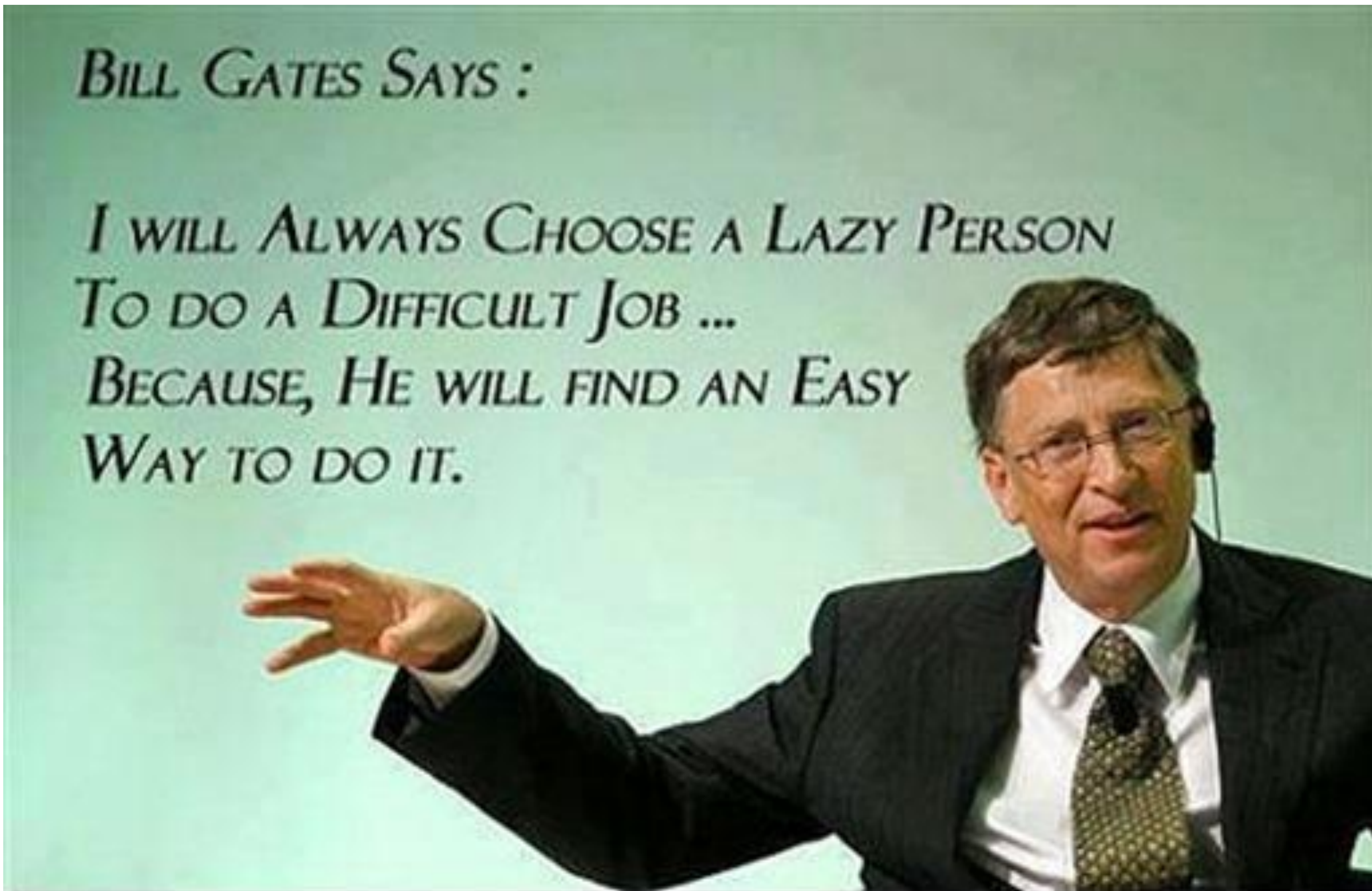
↑

(Movie, Viewer)



Very flexible, no rules elicitation,
Only need abundant (relevant) data

ML:Solving problems withour explicit programming and rules… is it about laziness?



It is about robustness!

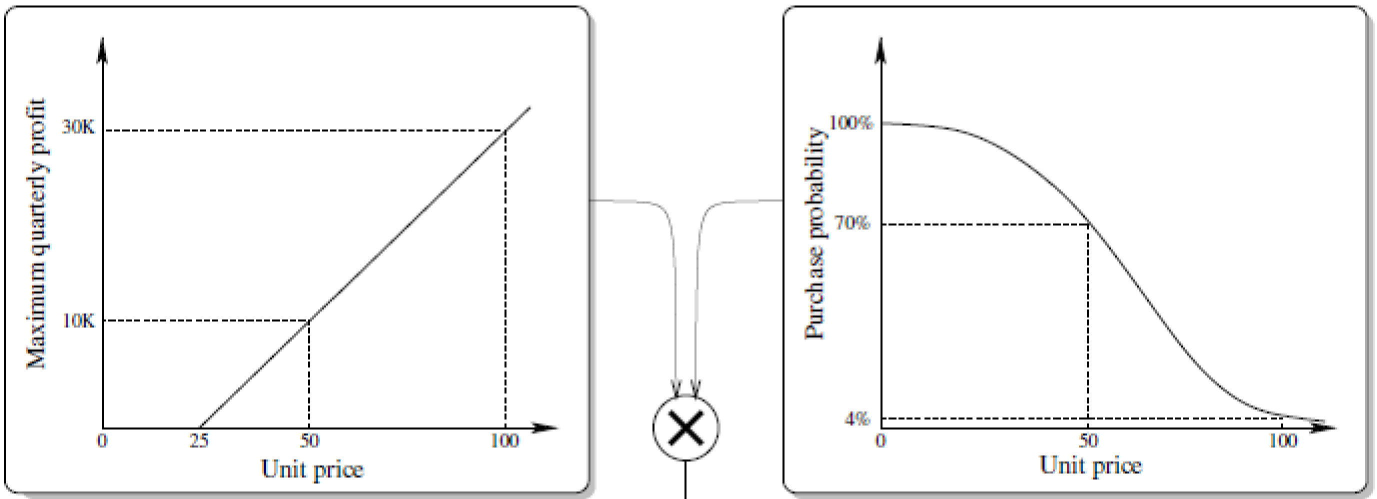
No self-driving mountain bike yet!

It is about flexibility!

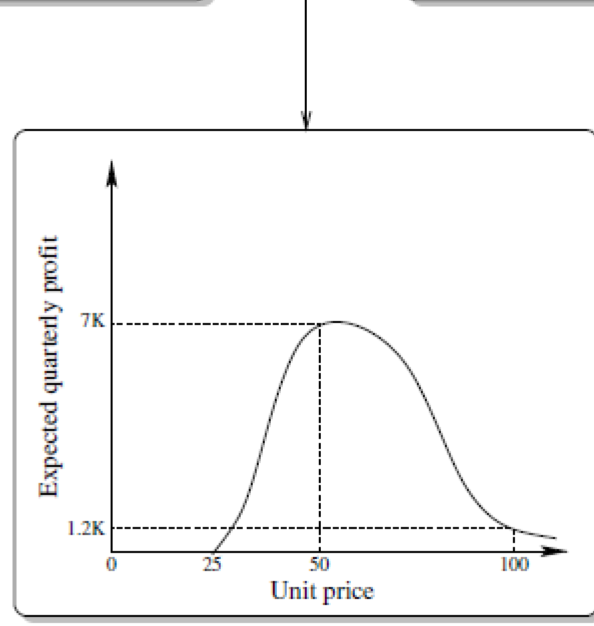
Different models are appropriate for different contexts

Which kind of model?

Cost model



Purchase probability



Profit

The dream

"give computers the ability to **learn** without being explicitly programmed" (Arthur Samuel, 1959).

The Tool

Weights of the flexible model are determined via **optimization**, but aiming at **generalization** (learning is *mean* not *end*)

No need to be an expert of the specific business to improve businesses

Is it possible? Neural networks!

…but airplanes do not flap their wings

Quegli che pigliavano per altore altro che la natura, maestra de' maestri, s'affaticavano invano.

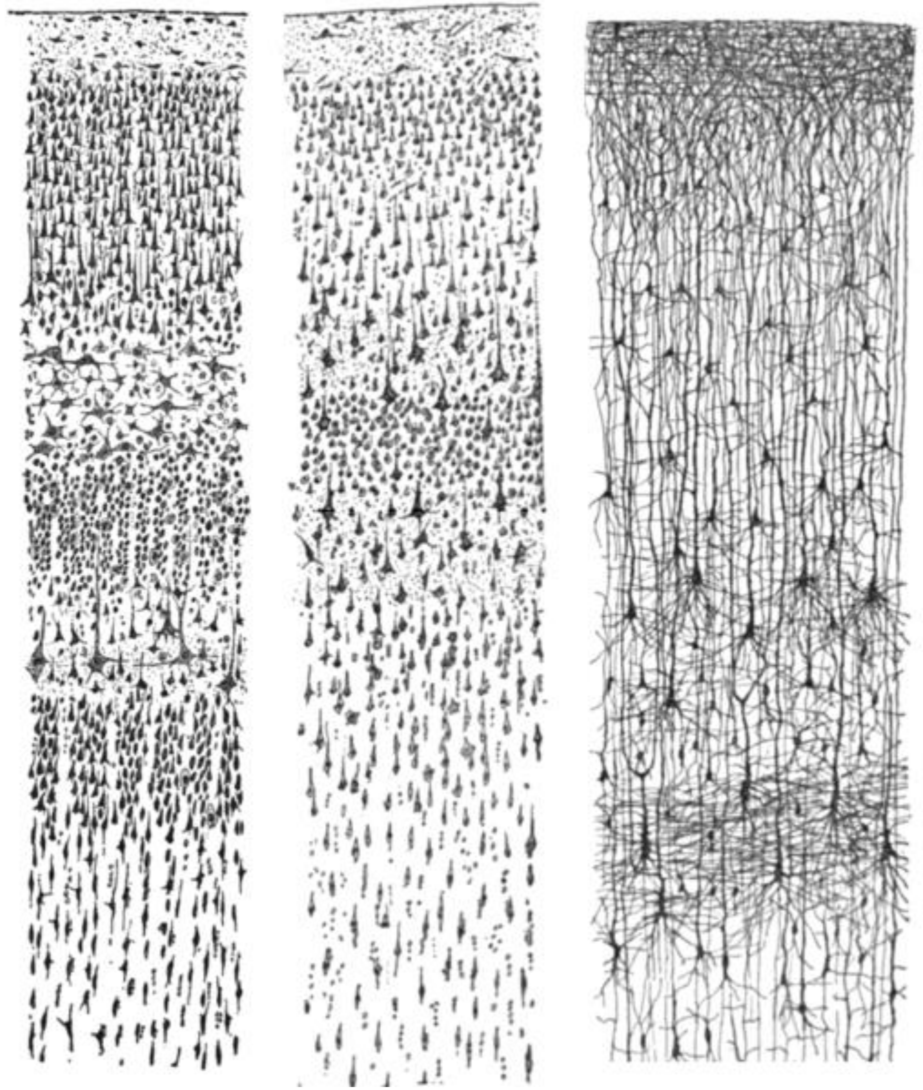
(Leonardo Da Vinci)

The biological metaphor

We are the living proof of learning from data

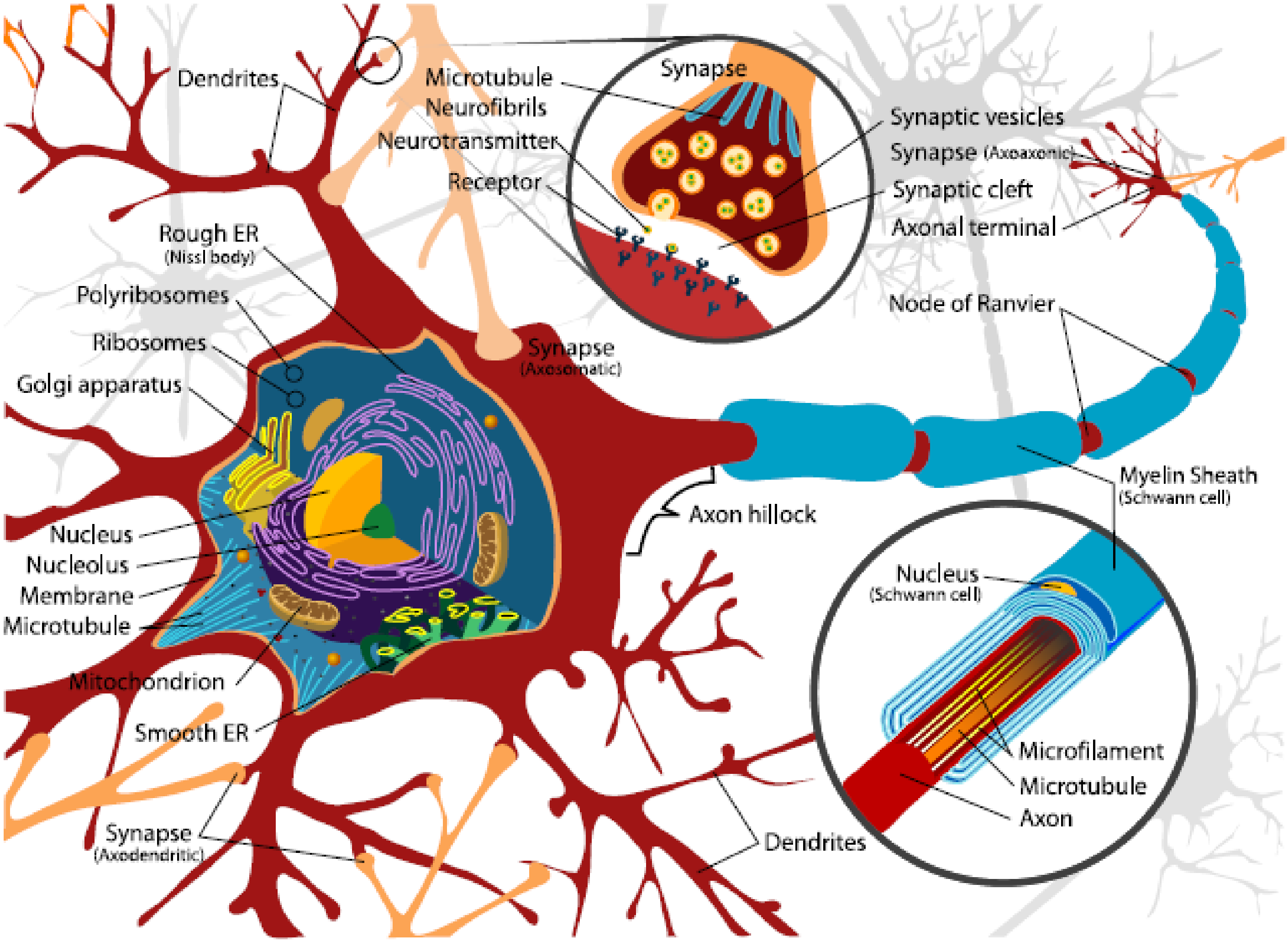
- Our neural system is composed of 100 billion computing units (neurons) and 10^{15} connections (synapses).
- How can a system composed of many simple interconnected units give rise to highly complex activities?
- **Emergence**: complex systems arise out of a multiplicity of relatively simple *interacting* units.

Emergence is very present in Physics!



Drawings of **cortical lamination** by Santiago Ramon y Cajal, each showing a vertical cross-section, with the surface of the **cortex** at the top. The different stains show the **cell bodies of neurons** and the **dendrites and axons** of a random subset of neurons.

Biological motivations



Neurons and synapses in the human brain

…but airplanes do not flap their wings, **we are welcome to use different algorithms and hardware**

Artificial Neural Networks

- A neuron is modeled as a simple computing unit, a scalar product **w x** (“pattern matching”) followed by a sigmoidal (“logistic”) function.

- The complexity comes from having more interconnected layers of neurons involved in a complex action
- The ”squashing” functions is essential to introduce nonlinearities in the system

A neuron is like a keyhole opened by a specific key (input signals)

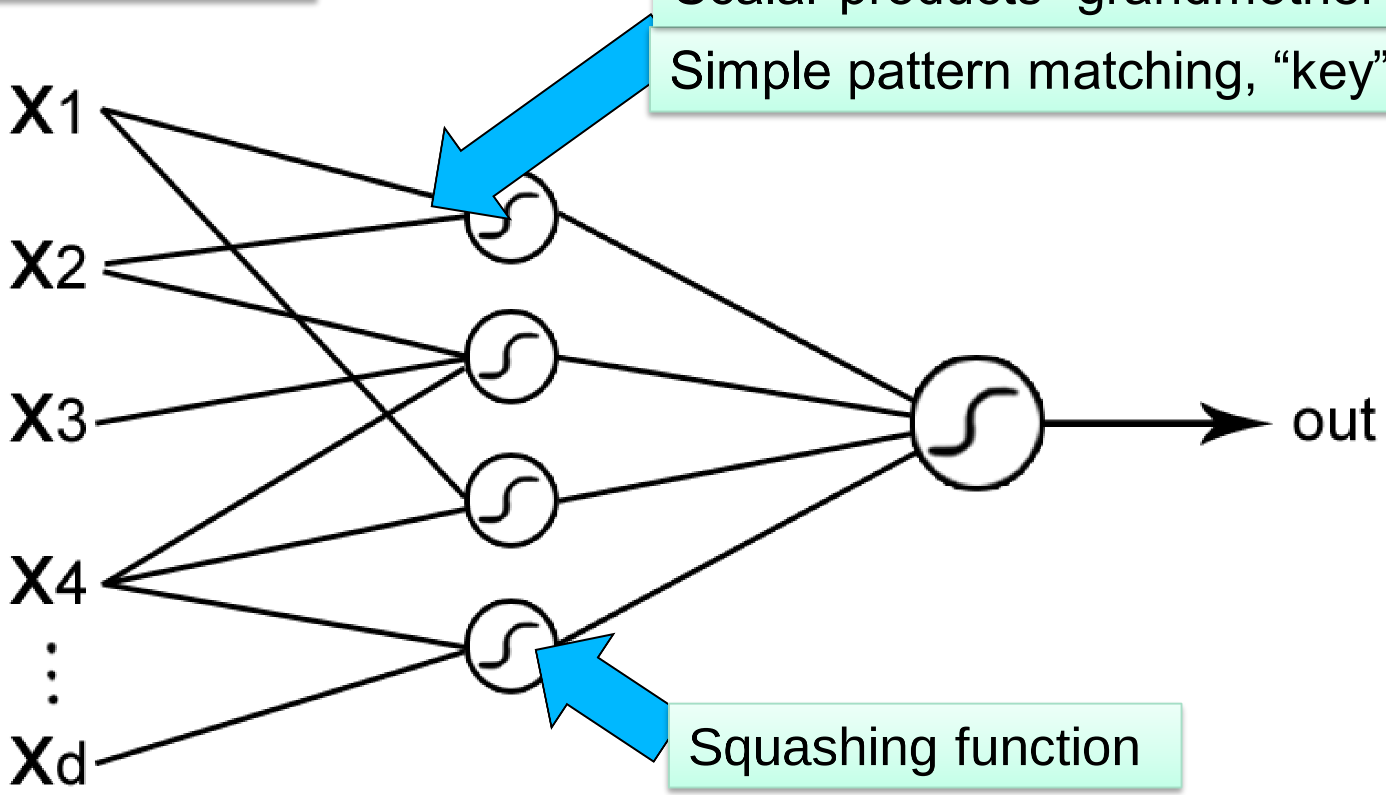
MLP architecture

- a large number of interconnected units working in parallel and organized in **layers** with a **feedforward** information flow.

fast “no reasoning”

Scalar products “grandmother neurons”

Simple pattern matching, “key” – “keyhole”



Squashing function

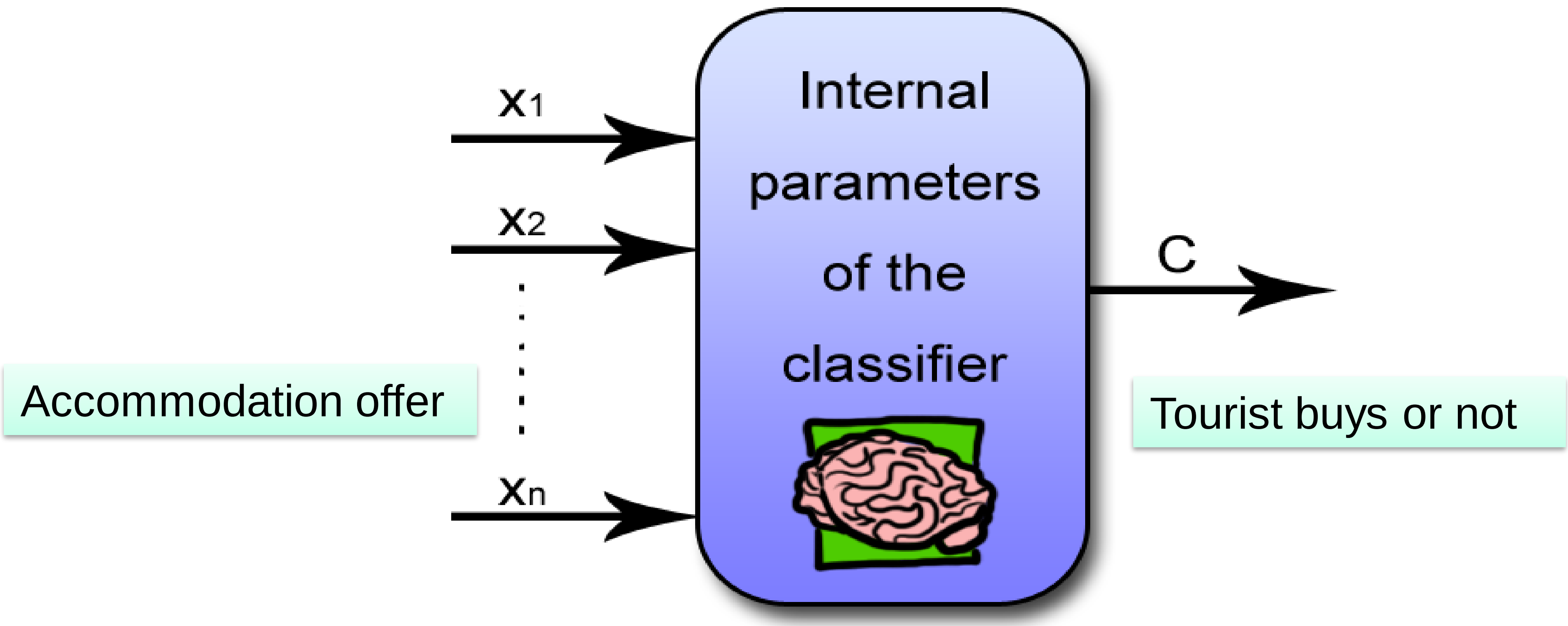
What is (machine) learning?

It's a kind of magic?

- Learning is *more than memorizing* *(«learning by heart»)*
- Unifying/compressing different cases by discovering the **underlying explanatory laws**.
- Learning from examples is only a **means** to reach the real goal: ***generalization***, the capability of explaining new cases

How to learn:
Supervised machine learning

a «teacher» is giving labeled examples



Learning from labeled examples: minimization and generalization

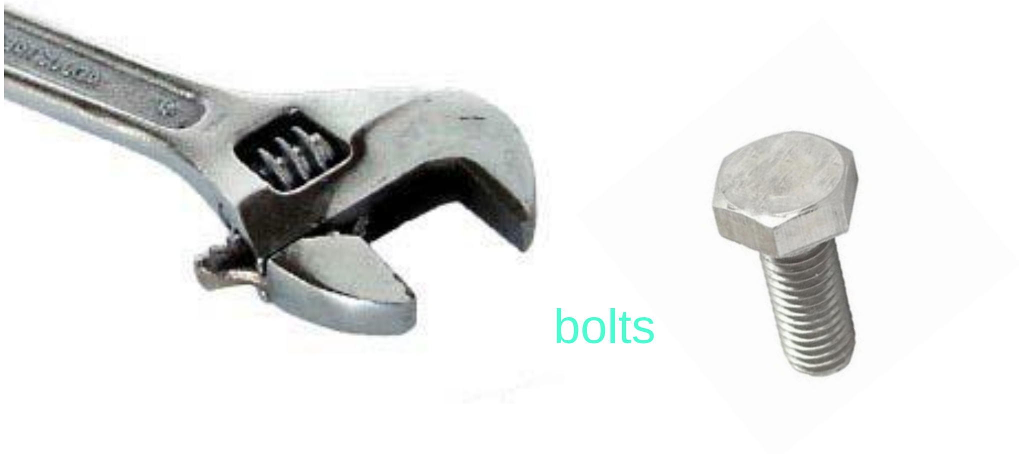
- A **flexible model f(x;w),** where the flexibility is given by some **tunable parameters** (or weights) **w**

wrench

- determination of the best parameters is fully **automated**, this is why the method is called *machine* learning after all

Learning from labeled examples: minimization and generalization (2)

- fix the free parameters by demanding that the **learned model works (approximately) correctly on the examples in the training set**.



- **power of** **optimization**:

full clarity about the objective

  - 1. define an **error** **measure** to be minimized,
  - 2. determine optimal parameters via (automated) **optimization**

Learning from labeled examples: minimization and generalization (3)

- suitable **error measure** is the **sum of the errors** between the correct answer (given by the example label) and the outcome predicted

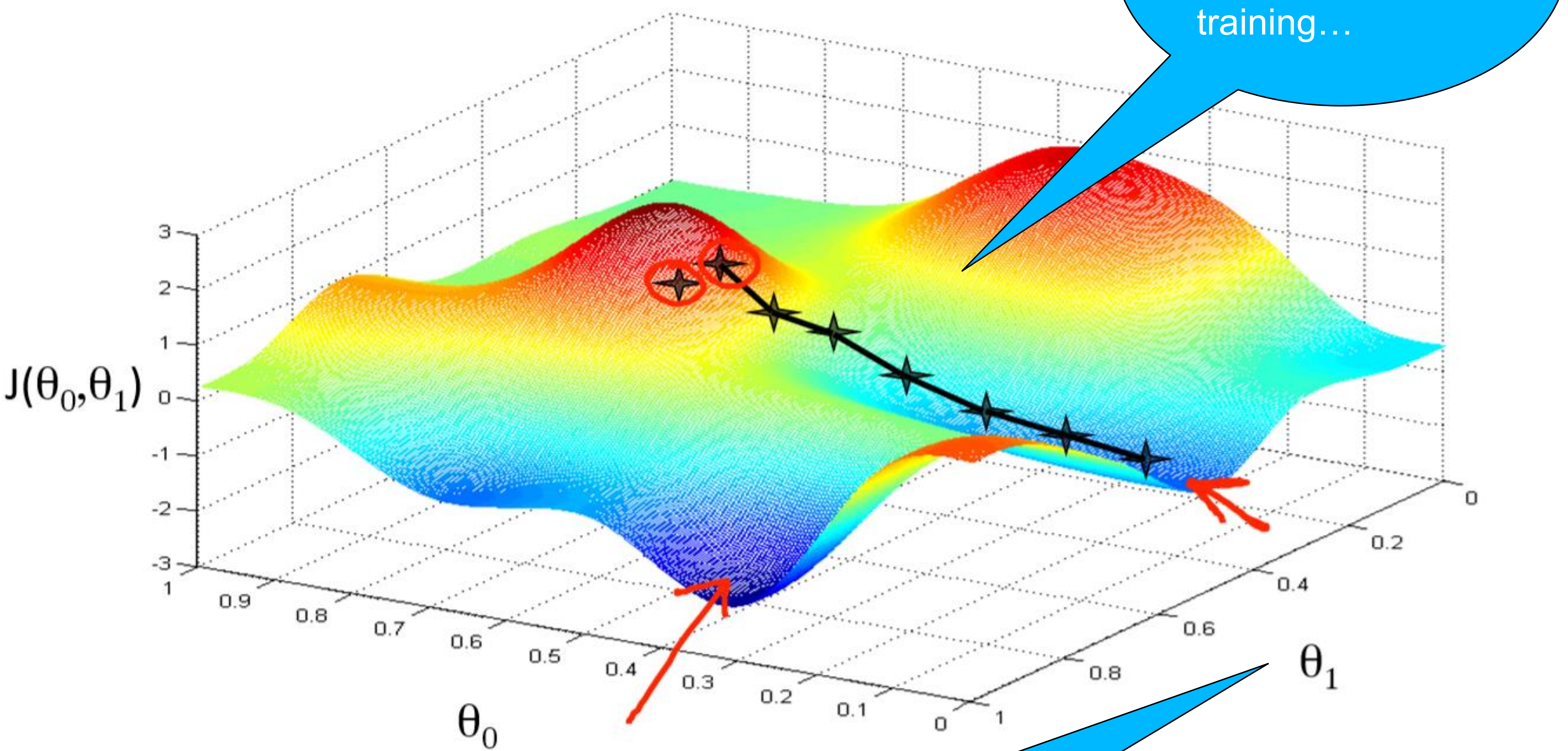
- if the function is smooth one can discover points of low altitude by being blindfolded and parachuted to a random initial point…

(gradient descent)


Gradient descent is like skiing

Gradient descent

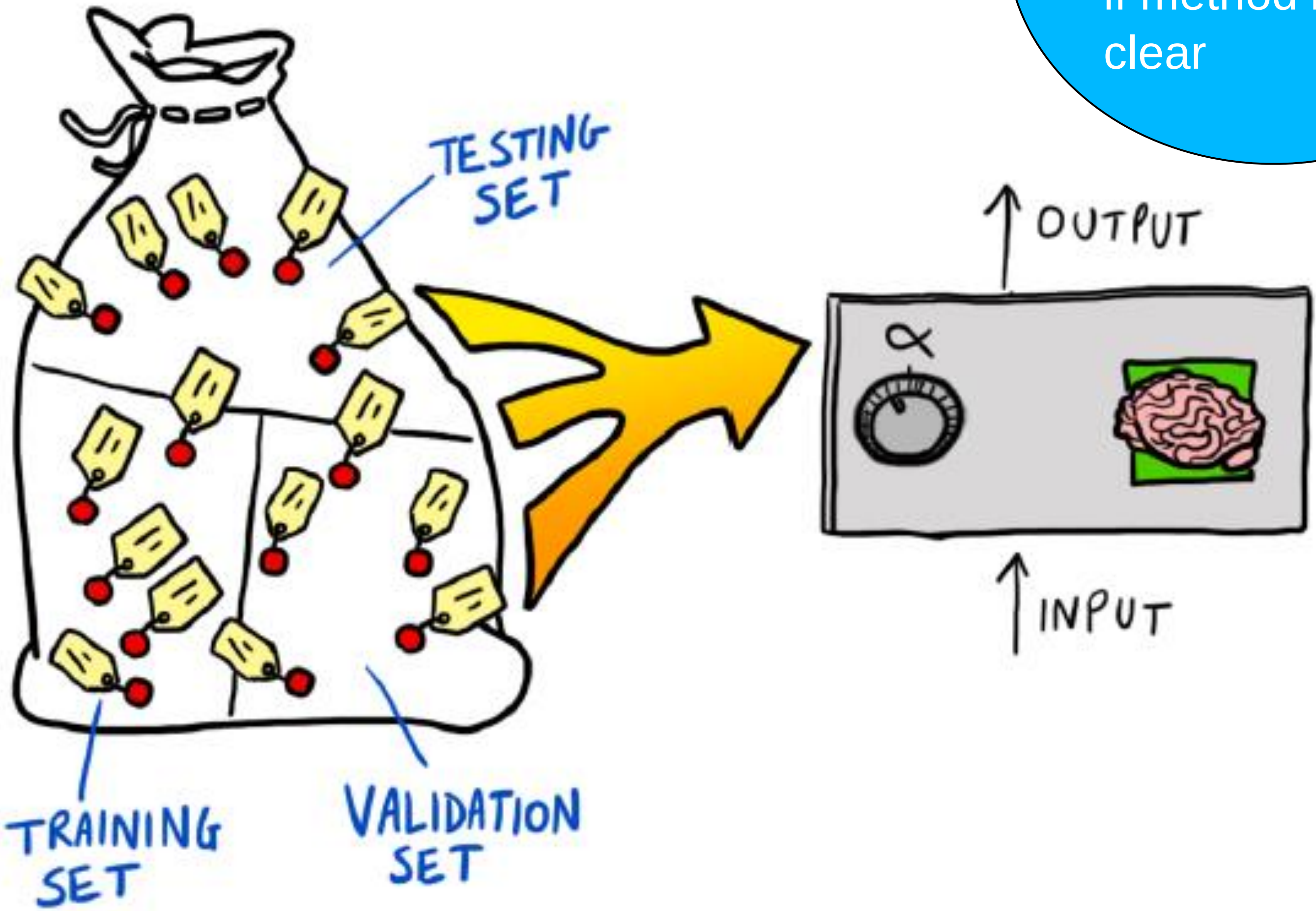
Imagine the altitude is your error during training…



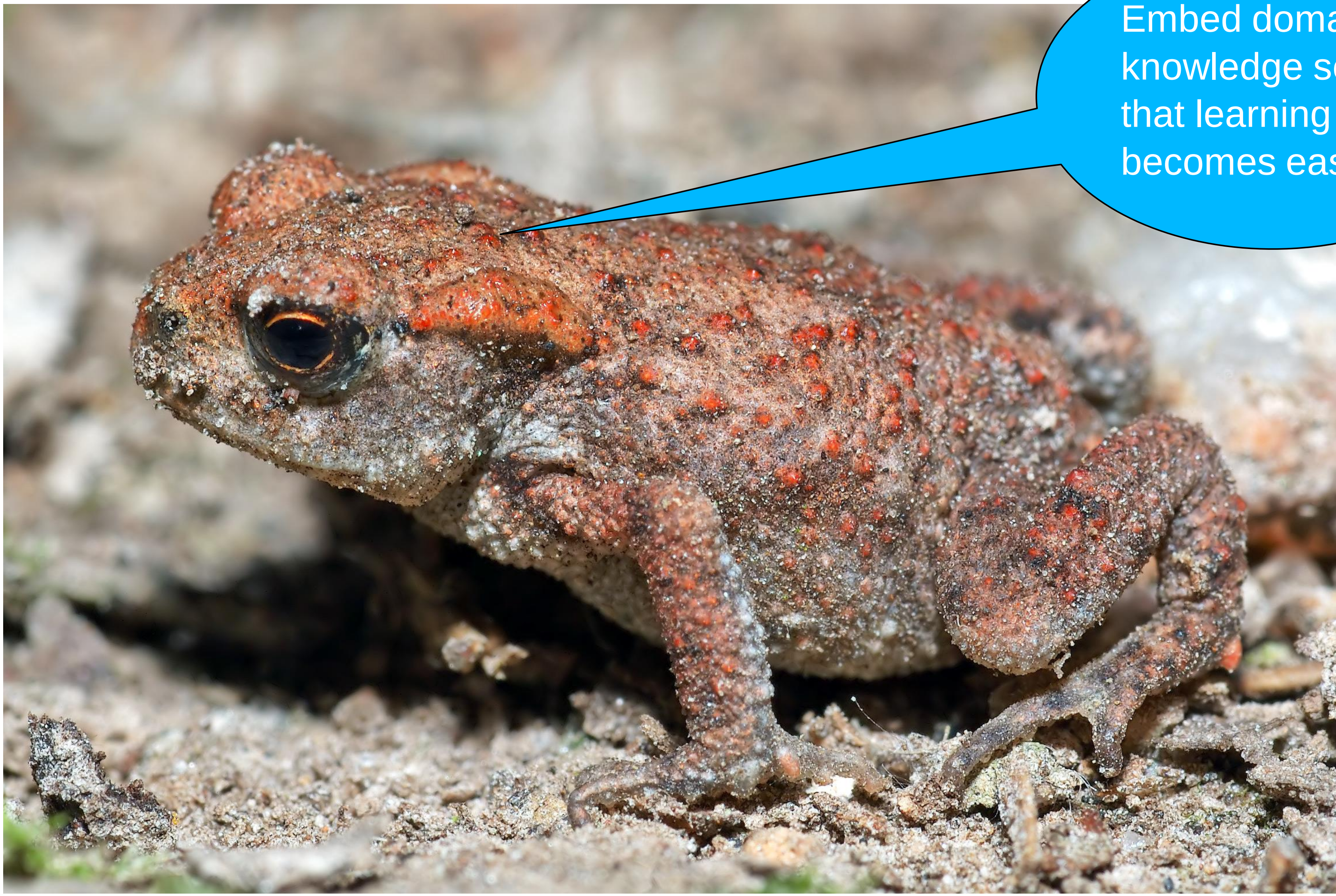
…and these are the parameters you can change

Learn, validate, test!

Warning: terrible mistakes ahead if method is not clear



Deep Learning



Feature detectors in a frog retina (*Bufo Bufo*) are hard-wired and **specialized to detect a fly at the distance that the frog could strike.**

Intelligent Optimization

Optimization

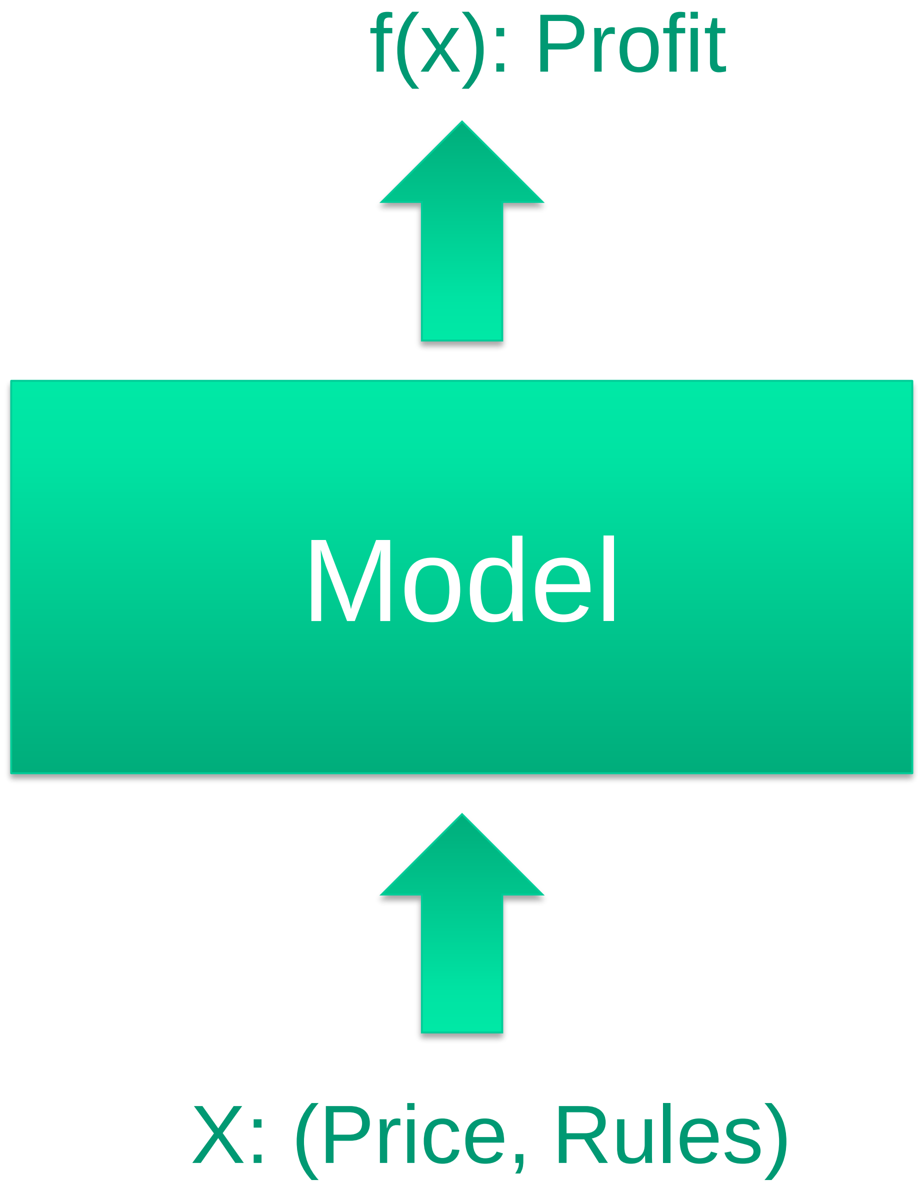
x are the parameters you can change (e.g., prices)
f(X) is the result (e.g., the profit)

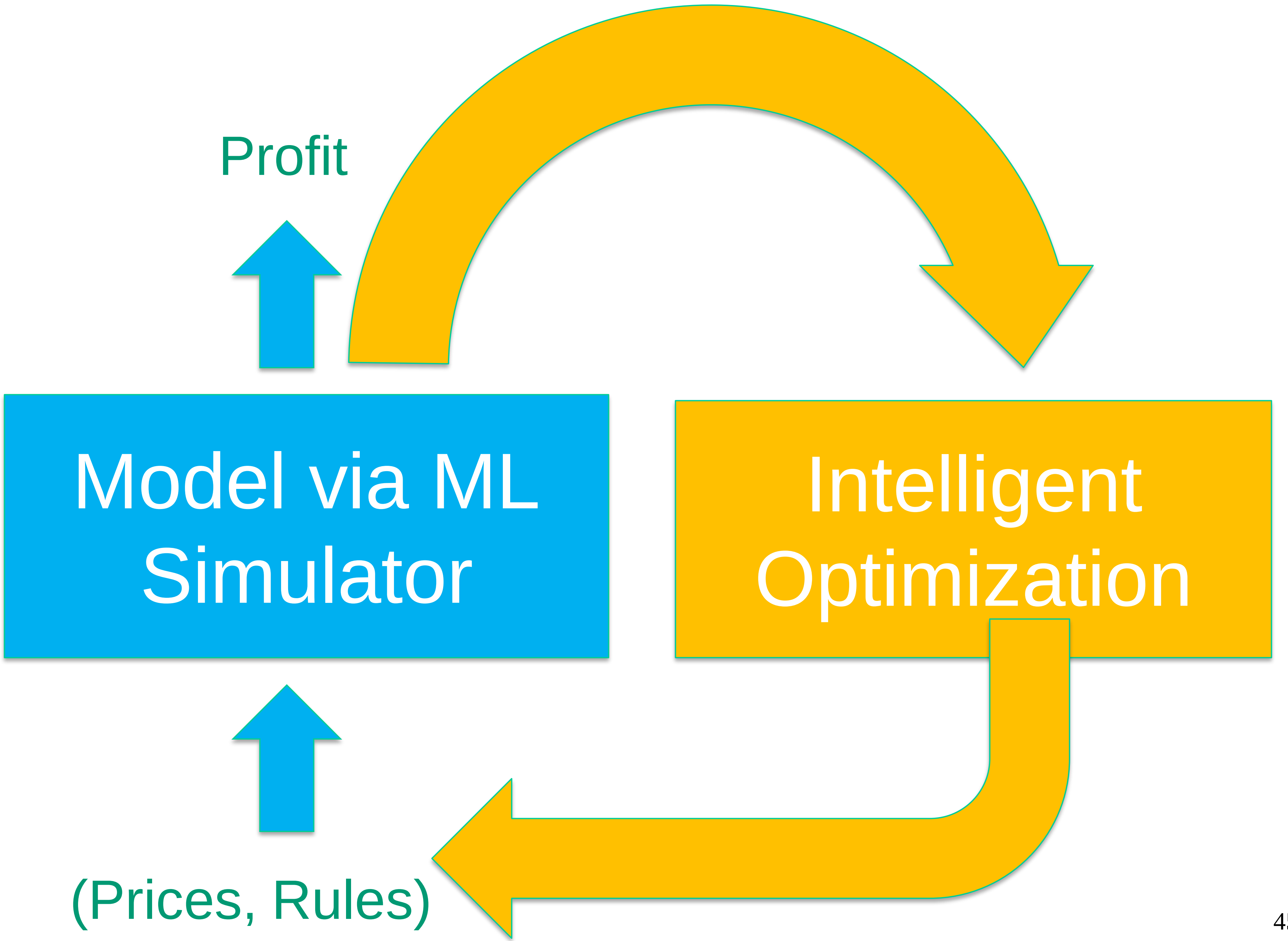
Minimum_x f(x)

(or maximum)

Optimization scares people. They think it is too math-oriented to be understood.
But its power is truly enormous…

A practical view of a «function»

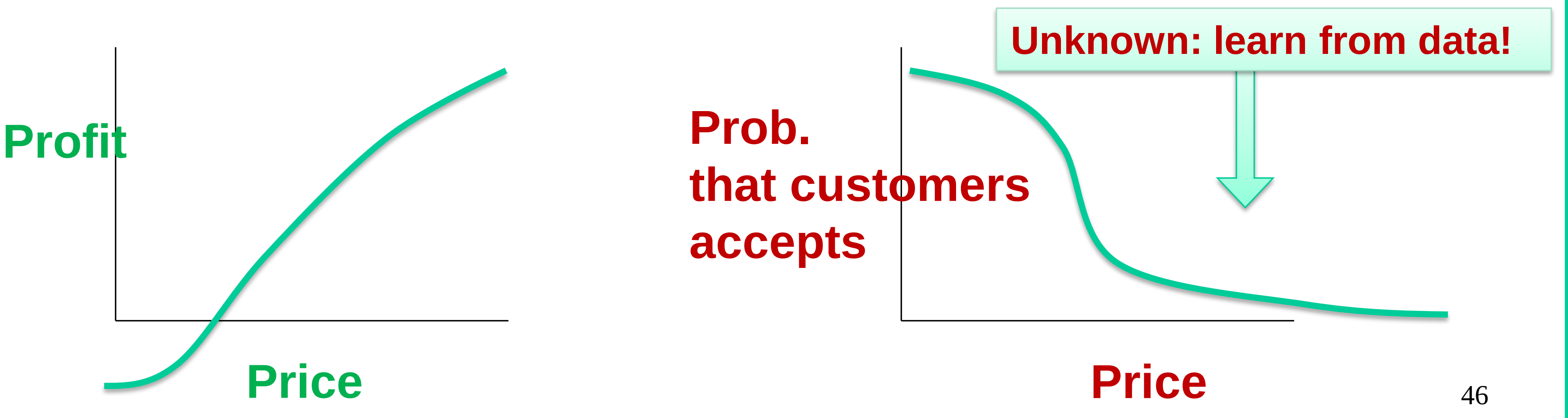



Profit
Model via ML Simulator
Intelligent Optimization
(Prices, Rules)

Example: determine the best price

- Profit = price paid – **costs**
- **Probability of accepting offer**

- Actual profit is **multiplication** of the two factors



- **After (machine) learning… optimize!**

How many problems can you solve **exactly** in reasonable computing times?

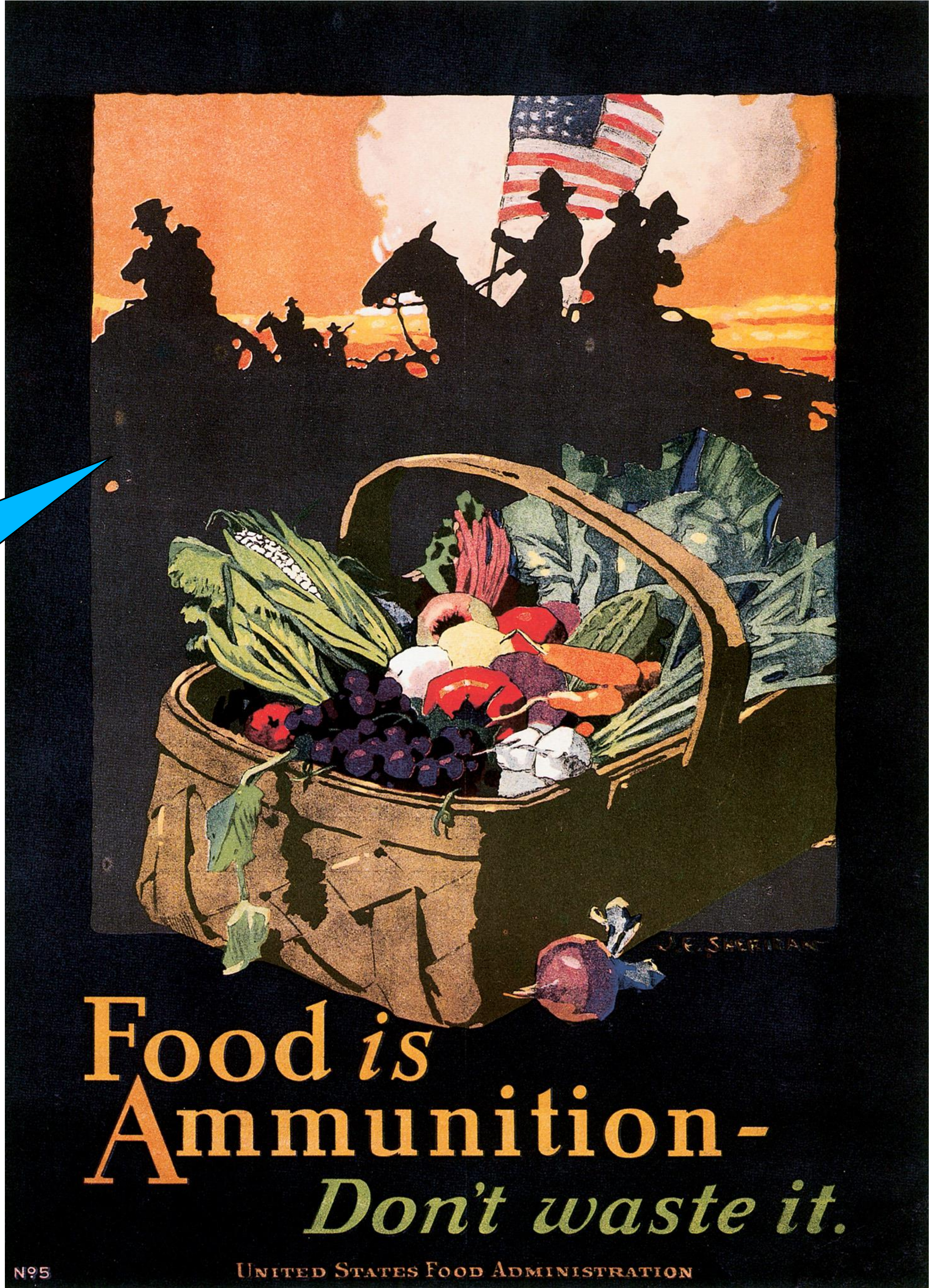
Not many ☹

How many solutions can you **improve** with intelligent optimization?

Most (all?) of them ☺

LP
Linear
Programming

Finding the optimal diet for soldiers was one of the first applications (objective: minimize cost of diet but keep soldiers healthy)



Quadratic Programming: how to find the minimum

Very relevant for RM:
Revenue = price x reservations
Reservations = c price
Revenue = c x price^2

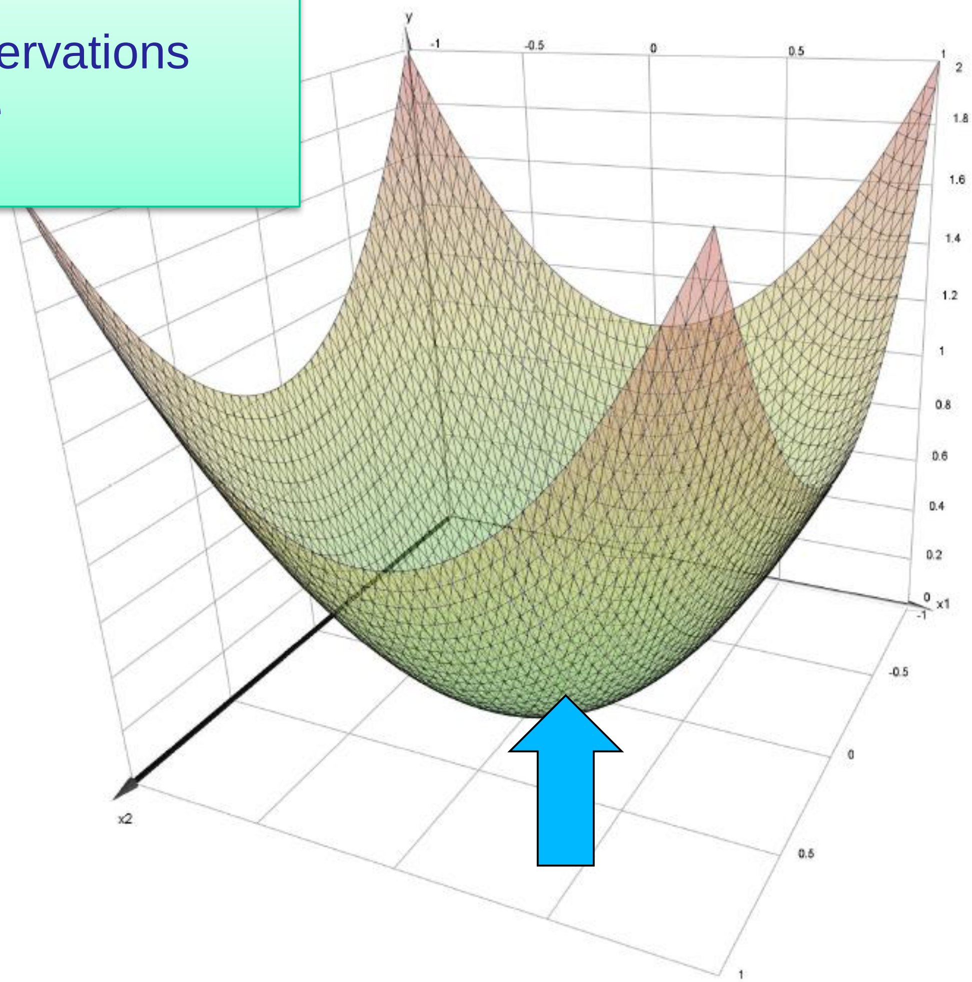


Figure 18.6: Quadratic positive definite f of two variables.

One sees it…

Try many (x,y) values…

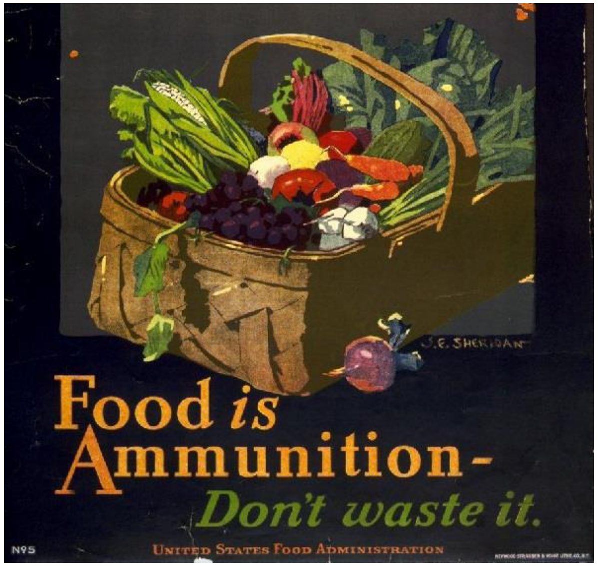
Which values?
All possible vals?

“Local steps”

Two (very different) paradigmatic methods

Optimization is a very old topic…

Operations research



Paradigms:

1 Stochastic global optimization (memory-less, "brute force", but very robust)

2 Local Search and **Reactive Search Optimization** (use learning while optimizing)

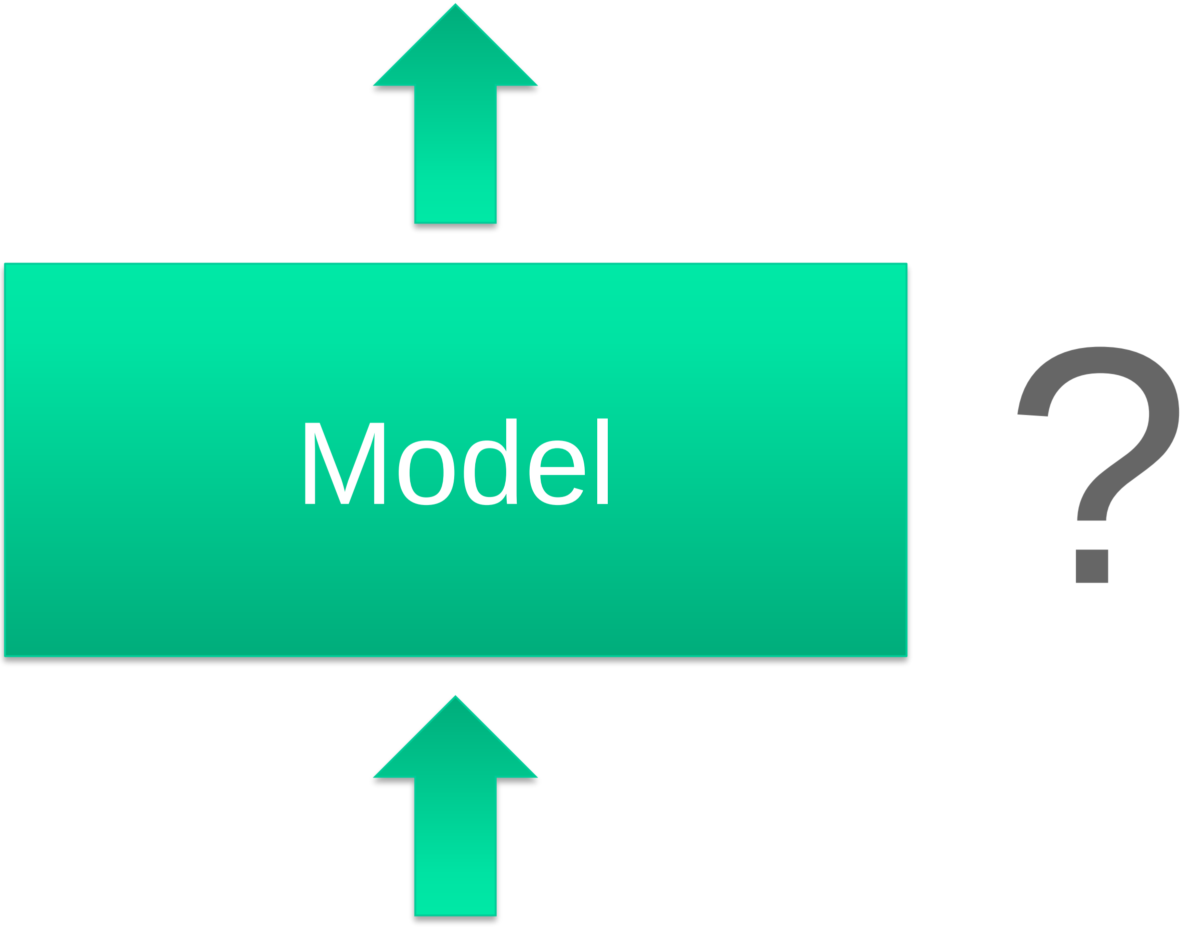
Paradigm1: Stochastic Global Optimization

Stochastic Global Optimization

- **black-box interface**: the algorithm can query the value f(x) for a sample point x, but it cannot “look inside” f
- **separation of concerns**: to be as generally applicable as possible, optimization routines do not need to know anything about the application domain;
- a computer scientist can improve profits for hotels or improve survivability of patients cured for cancer **without any knowledge** of economics or medicine.

Ignorance **can** bring value

Black-box optimization



Stochastic Global Optimization

- just function evaluations
- **function of continuous (real) variables**
- one can decide where to place sample points, and one can use the information obtained to build internal models of the function and tune its own meta-parameters.
- stochasticity in the generation of sample points helps to improve robustness and avoid that some false initial assumptions lead to low-quality local optima

Convergence Rate of **Pure Random Search**

- Success with probability (1 – γ)
- In the asymptotic behavior when *d* is fixed and $\epsilon \to 0$,number of iteration for success $n_* = O\left(\frac{1}{\epsilon^d}\right)$
- **Curse of dimensionality**

d	$\gamma = 0.1$			$\gamma = 0.05$		
	$\varepsilon = 0.5$	$\varepsilon = 0.2$	$\varepsilon = 0.1$	$\varepsilon = 0.5$	$\varepsilon = 0.2$	$\varepsilon = 0.1$
1	0	5	11	0	6	14
2	2	18	73	2	23	94
3	4	68	549	5	88	714
4	7	291	4665	9	378	6070
5	13	1366	43743	17	1788	56911
7	62	38073	$4.9 \cdot 10^6$	80	49534	$6.3 \cdot 10^6$
10	924	$8.8 \cdot 10^6$	$9.0 \cdot 10^9$	1202	$1.1 \cdot 10^7$	$1.2 \cdot 10^{10}$
20	$9.4 \cdot 10^7$	$8.5 \cdot 10^{15}$	$8.9 \cdot 10^{21}$	$1.2 \cdot 10^8$	$1.1 \cdot 10^{16}$	$1.2 \cdot 10^{22}$
50	$1.5 \cdot 10^{28}$	$1.2 \cdot 10^{48}$	$1.3 \cdot 10^{63}$	$1.9 \cdot 10^{28}$	$1.5 \cdot 10^{48}$	$1.7 \cdot 10^{63}$
100	$1.2 \cdot 10^{70}$	$7.7 \cdot 10^{109}$	$9.7 \cdot 10^{139}$	$1.6 \cdot 10^{70}$	$1.0 \cdot 10^{110}$	$1.3 \cdot 10^{140}$

Table 2.1. Values of $n_* = n_*(\gamma, \varepsilon, d)$, see (2.22), for vol$(A) = 1$, $\gamma = 0.1$ and 0.05, $\varepsilon = 0.5, 0.2$ and 0.1, for various d.

Curse of dimensionality

- ”Abandon all hope, you who enter here”. If dimension is large there is **no magic algorithm to rapidly approximate the global optimum for a generic function in less than exponential number of iterations**.
- There are just too many places to hide in d dimensions.
- Hope is related to **functions with special forms, so that regularities can be learnt** from an initial sampling, albeit in approximated form, and used to identify shortcuts leading rapidly to close approximations of the optimal solution (learning x optimization)
- Chance that we encounter highly-structured functions in real applications? Not negligible. **Nature does not play dice…**

Problem structure is helping us

Paradigm2: Local Search and Reactive Search Optimization (RSO)

Reactive Search Optimization

More details in:

Battiti, R., Brunato, M. and Mascia, F., 2008. *Reactive search and intelligent optimization* (Vol. 45). Springer Science & Business Media.

Local search based on perturbations

- brute force is not always the solution

1) start from an initial tentative solution

2) try to improve it through repeated small changes

3) stop when no improving local change exists

(local optimum, or locally optimal point)

Modifications of local search based on perturbations

- local search by small perturbations is an effective technique but additional ingredients are in certain cases needed to obtain superior results

"It is a good morning exercise for a research scientist to discard a pet hypothesis every day before breakfast: it keeps him young" (Konrad Lorenz, 1903-1989).

Reactive Search Optimization (RSO): **Learning while searching**

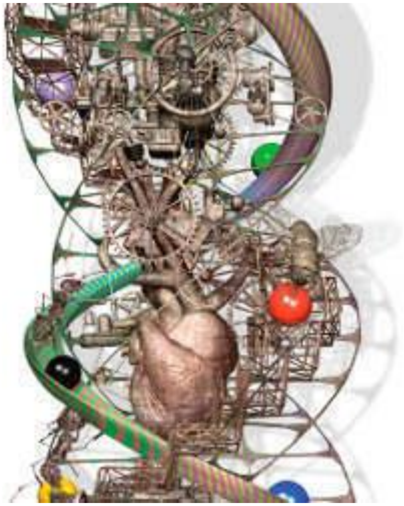
- Many problem-solving methods are characterized by a certain number of choices and free parameters, usually manually tuned.
- Parameter tuning can be automated as a part of the optimization algorithm
- This leads to self-contained, fully automated algorithms, independent from human intervention

Reactive Search Optimization (RSO) integrates online machine learning techniques and search heuristics for solving complex optimization problems.

Reactive Search Optimization

integration of online machine learning techniques for local search heuristics.

The word ***reactive*** hints at a ready response to events *during* the search through an internal online feedback loop for the *self-tuning* of critical parameters.



Biological systems are highly adaptive; they use signals coming in from receptors and sensors to fine-tune their functioning. Adaptivity is a facet of the **reactivity** of such systems.

Disruptive innovation by

combining ML + simulation + IO

Optimization: a tremendous power

Tapping and musik

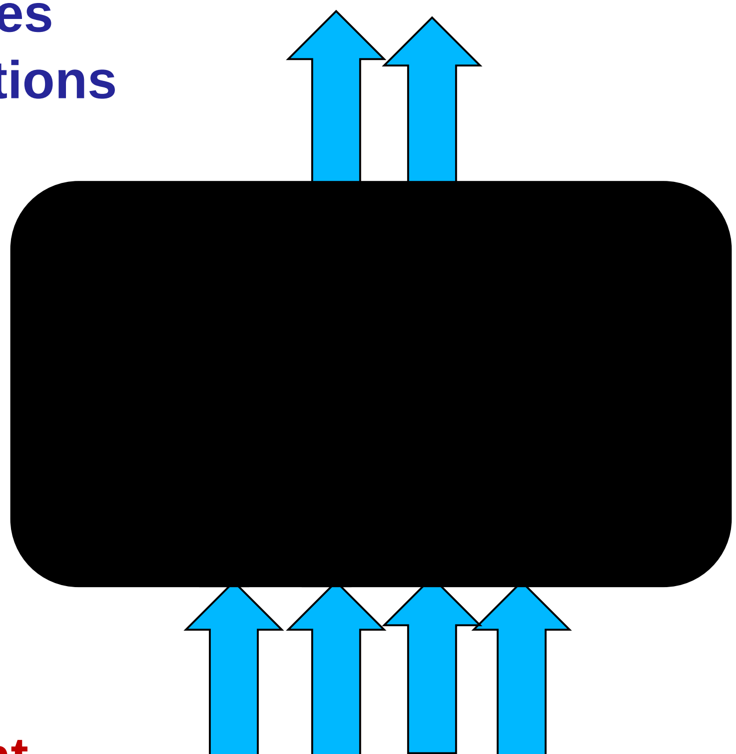
- Still largely unexploited in most real-world contexts: standard optimization assumes a **function f(x)** to be minimized, …and **math** knowledge.
- function f(x) (a.k.a “model”) helps people to **concentrate on goals/objectives**, not on algorithms (on policies not on processes)
- BUT static f(x) does not exist in explicit form or is extremely difficult and costly to build by hand, and math knowledge is scarce.

Try asking an hotel manager

Real word is dirty (black?)

Some posivite objectives (MOOP)
Combination not clear
Hidden objectives
Dynamic aspirations

Learn !



No math formula
Maybe some high-level knowledge and intuition

Learn !

Many inputs, noisy, some irrelevant

Learn !

Machine Learning !

If f(x) not given? Learn *what* to optimize

Example: MOP: Finding a partner: *intelligence* versus *beauty*

How many IQ points for one less beauty point?

Is beauty more important than intelligence for you? By how much?

Effective optimization
as iterative process with learning

Pareto-optimality

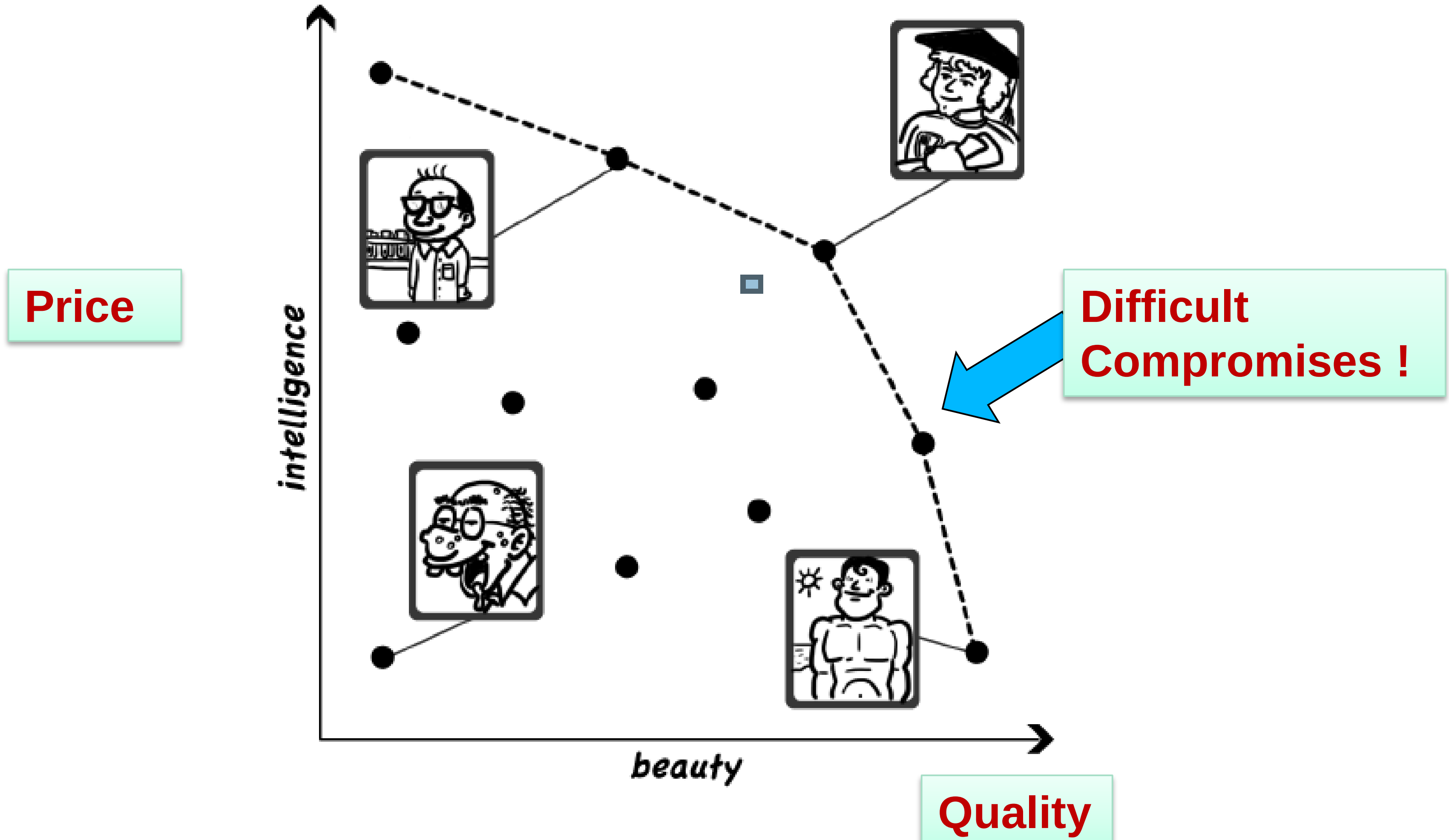


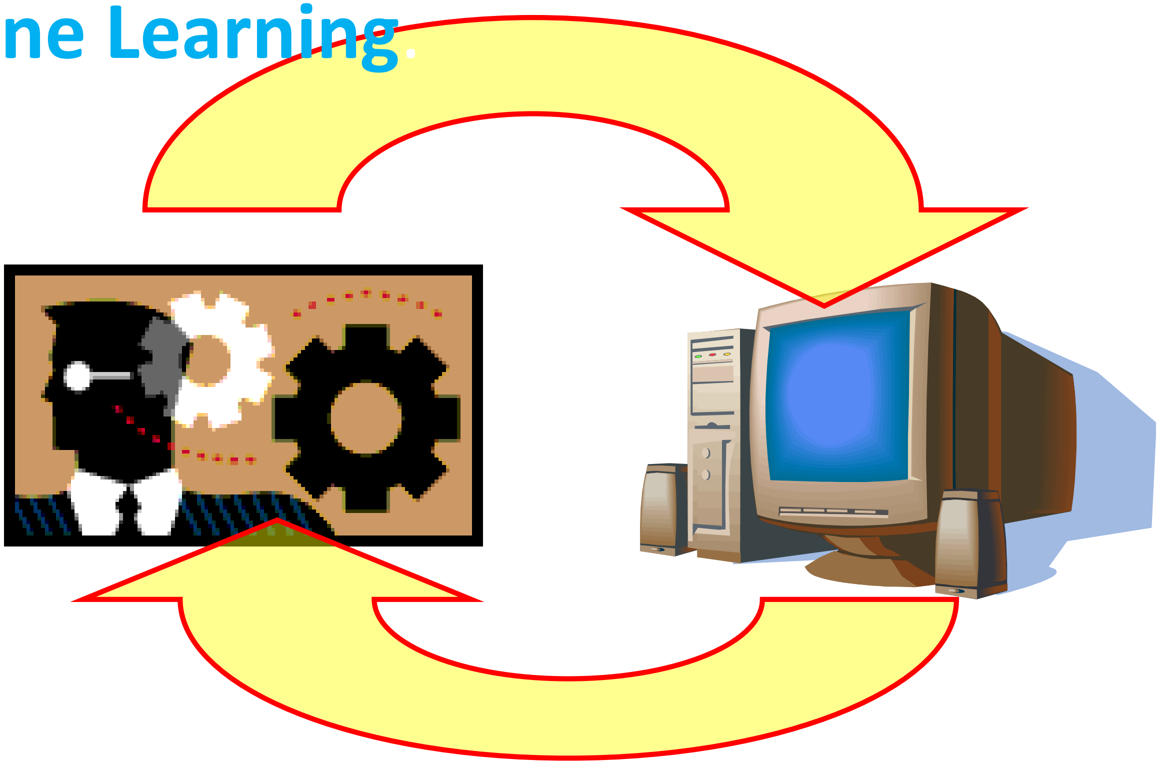
Figure 41.3: Pareto optimality. All dominated points like the persons in the middle are not considered as potential candidates for the final choice. On the Pareto frontier, shown with a dashed line, tradeoffs need to be considered.

Flexible and interactive decision support and problem solving

Crucial decisions depend on factors and priorities which are not always easy to describe before.

Feedback from the user in the exploration phase!

+ **Machine Learning**.



An example: Combining Intelligent Optimization with Simulators in Hotel RM

More details in:

Brunato, Mauro and Battiti, Roberto

"Combining intelligent heuristics with simulators in hotel revenue management"

Annals of Mathematics and Artificial Intelligence", 2019",

issn="1573-7470",doi="10.1007/s10472-019-09651-9",

url="https://doi.org/10.1007/s10472-019-09651-9"}

https://link.springer.com/article/10.1007/s10472-019-09651-9

Combining Intelligent Optimization with Simulators in Hotel RM

- resorting to heuristics does not imply abandoning **experimental science** (e.g., training vs validation vs test)
- Real experiments in hotels can be difficult (and slow)
- Massive experiments are now made possible by **fast hotel simulators** which can **be trained on the hotel data** to simulate the hotel reservation process

There is another area in which experimentation with the real thing is difficult….

Monte Carlo method

(invented in the late 1940s by Stanislaw Ulam)

HotelSimu: a general-purpose simulator for hotels

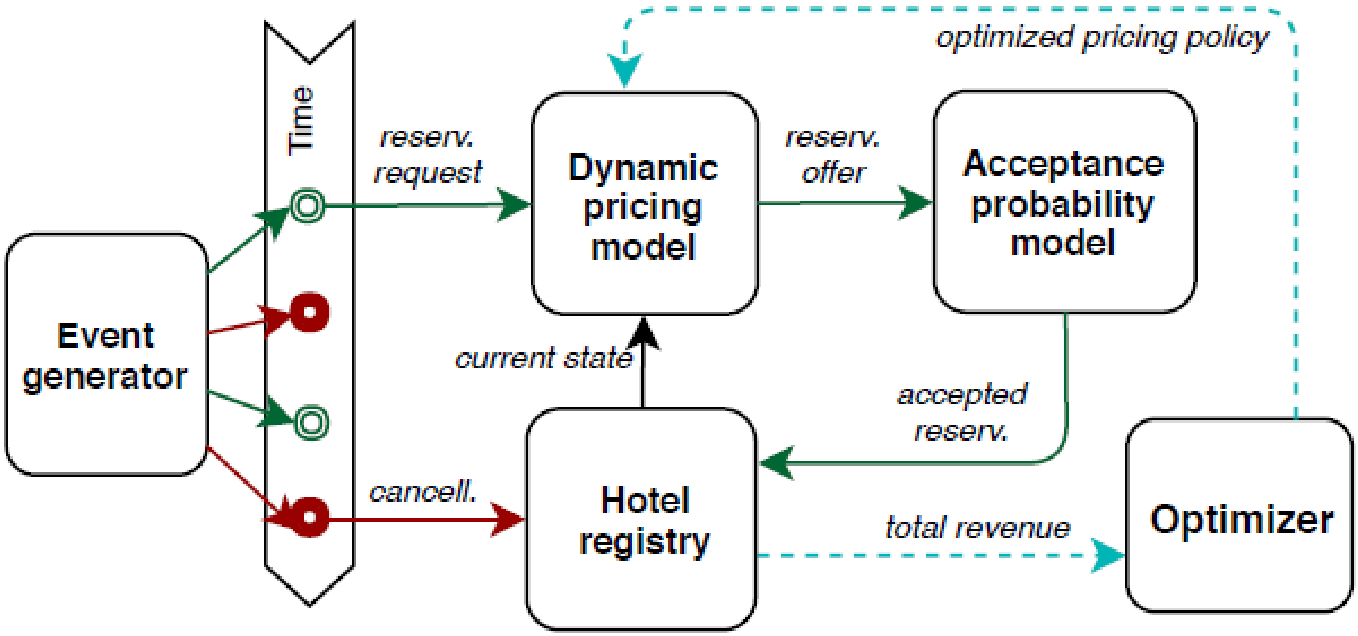


Figure 2: HotelSimu overview. Reservation requests and cancellations are interspersed. The state of the hotel after one complete simulation is used by the optimizer to compute the total revenue and adjust the pricing policy.

Generative model of requests

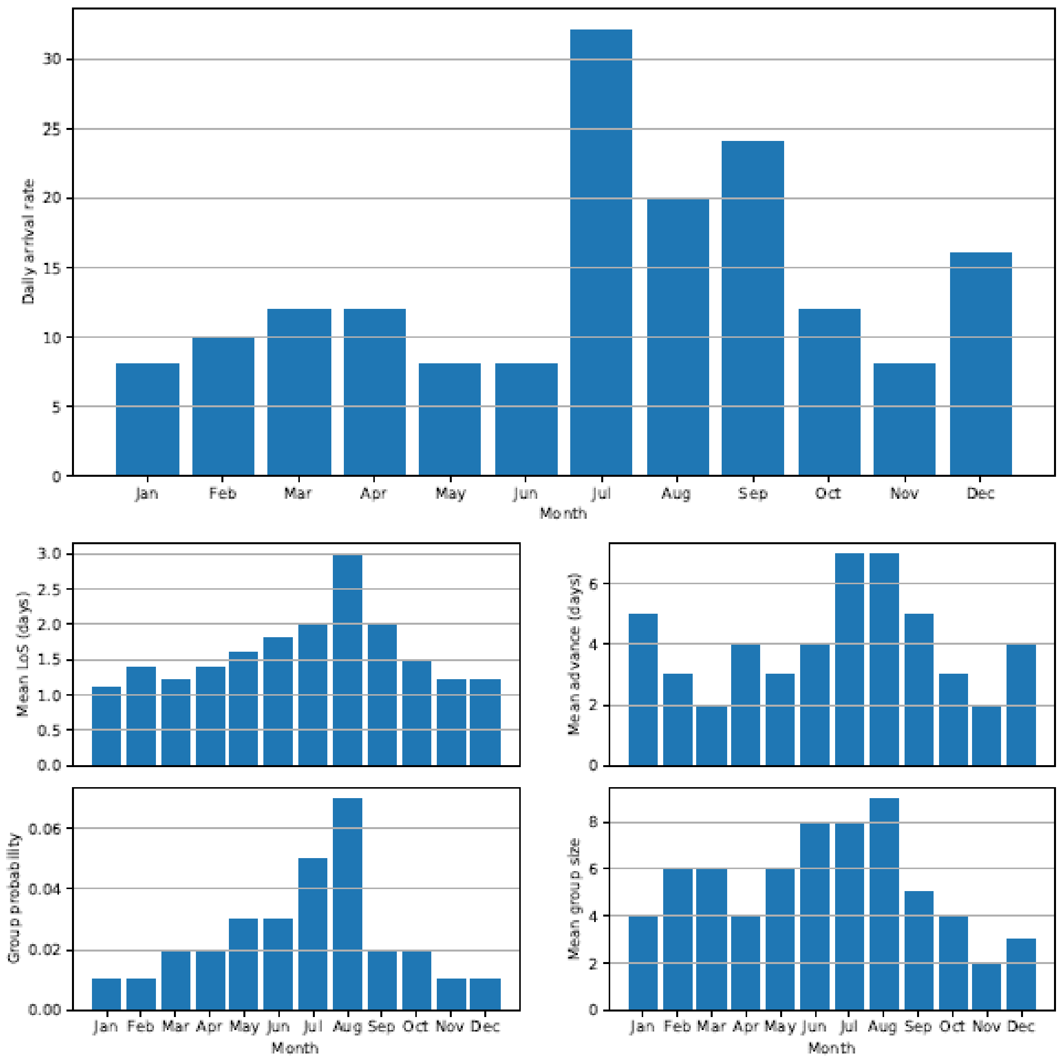


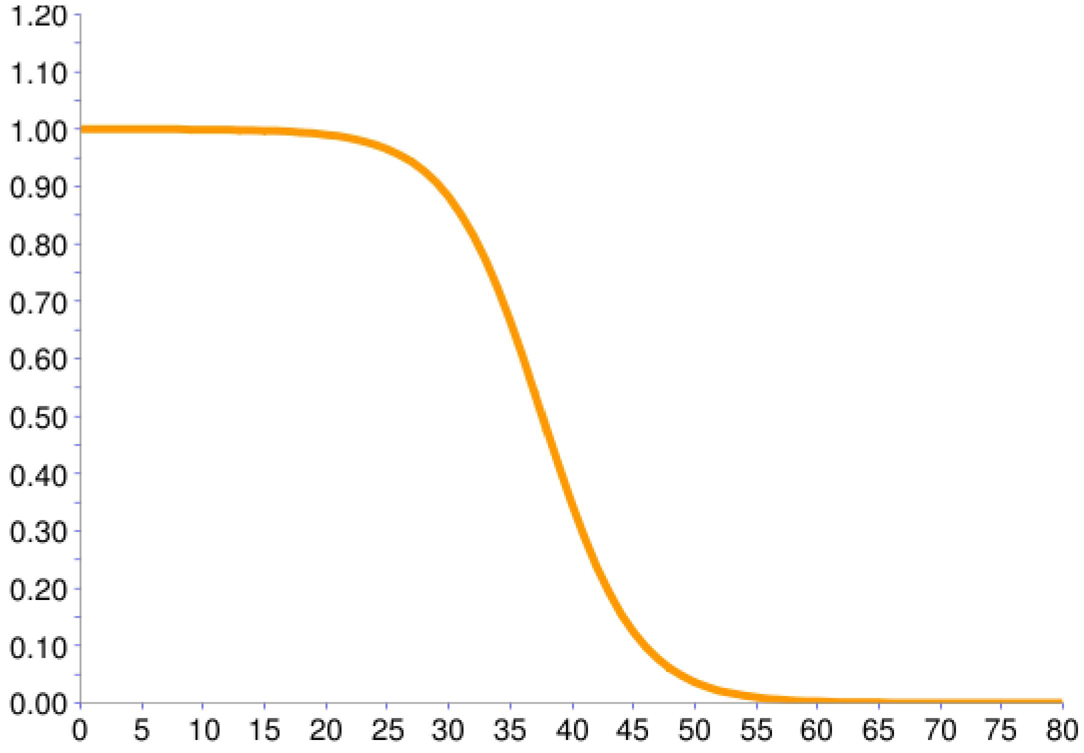
Fig. 2 Reservation generator time-varying parameters. Top: expected requests per day. Below, clockwise: expected length of stay, mean advance with respect to check-in date, expected size for group reservations (more than one room), probability of group reservations. Parameters adapted from historical data from some Northern Italian hotels.

Price acceptance model

$$p_a(u) = 1 - \sigma\left(\frac{u - \mu}{\eta}\right)$$

50% probability of acceptance,

elasticity.



Pricing policies

- **Median acceptance price** — “Const median”

$u = \mu$: acceptance probability is always 0.5

- **Unsaturated equilibrium price** — “Const equilibrium”

u maximizes the expected revenue on the hypothesis that there is an infinite supply of rooms

$$u^* = \arg\max_{u \in \mathbb{R}} u \cdot p_a(u) = \eta \left(1 + W_0\left(e^{\frac{\mu}{\eta} - 1}\right)\right),$$

where $W_0(\cdot)$ is the main branch of Lambert’s W function.

Pricing policies (2)

- **Pickup-based dynamic price** — "Dynamic"
  - Res one-day and independent, competing for the same check-in date, time discretization
- **Best constant price** — "Const Grid search"
  - determined by a grid search on a training set of reservations for the price that maximizes the hotel's revenue.
- **Factored pricing** — "Factored"

f1 is a 2-piecewise linear function of the time to arrival which can accommodate for independent early and last-minute discounts or penalties, while the next three linearly depend on group size, length of stay and residual capacity at the time of reservation.

Intelligent optimization heuristcs

- CMA-ES — An evolutionary optimization algorithm based on covariance matrix estimation
- Affine Shaker —Local search-based optimization method founded on the Collaborative Reactive
- **Cooperative Reactive Search Optimization (CoRSO)** framework particularly fit to low-dimensional search spaces.
- Inertial Shaker — An alternative to the latter; less computation-intensive and therefore fit to higher-dimensional search spaces

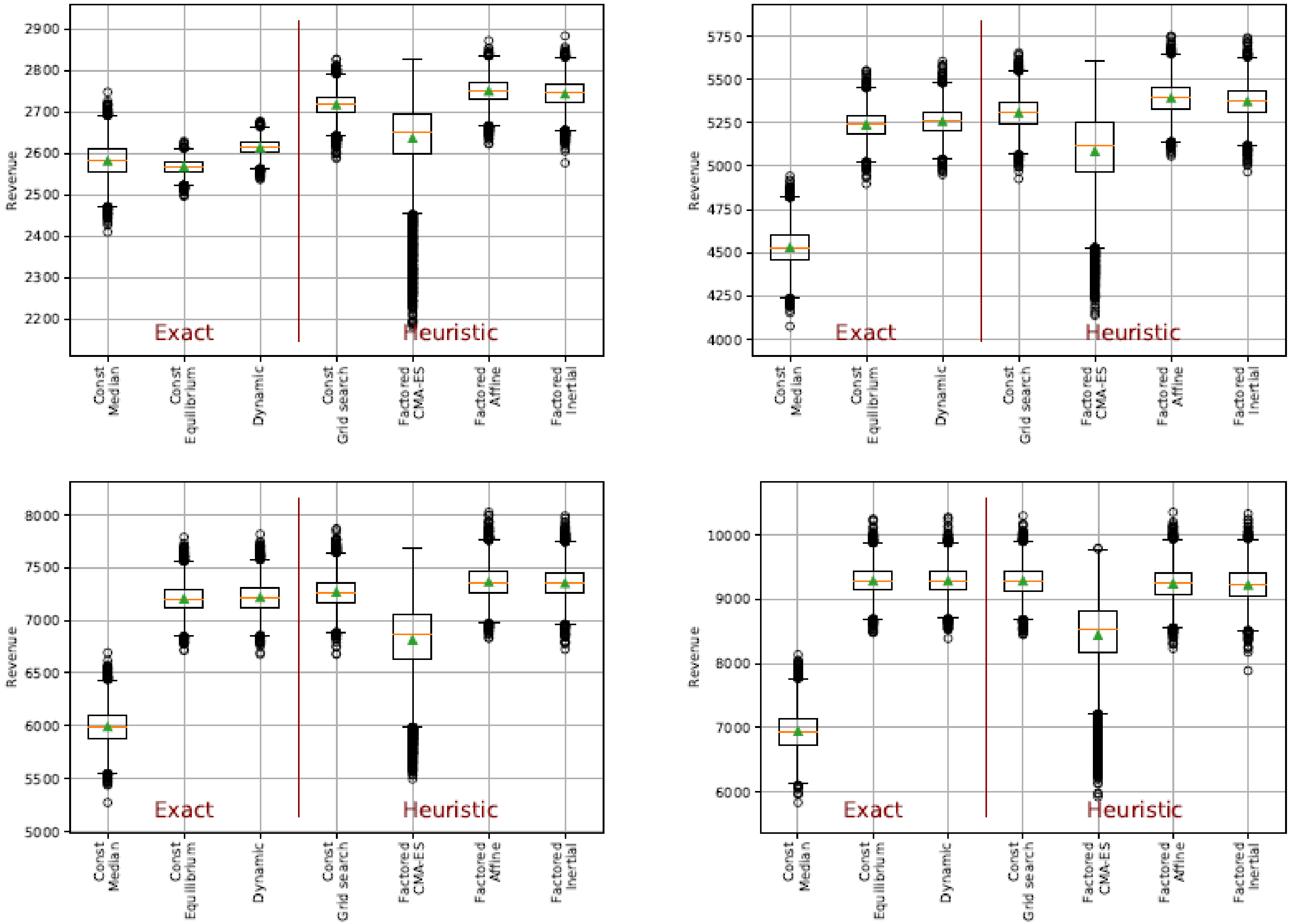
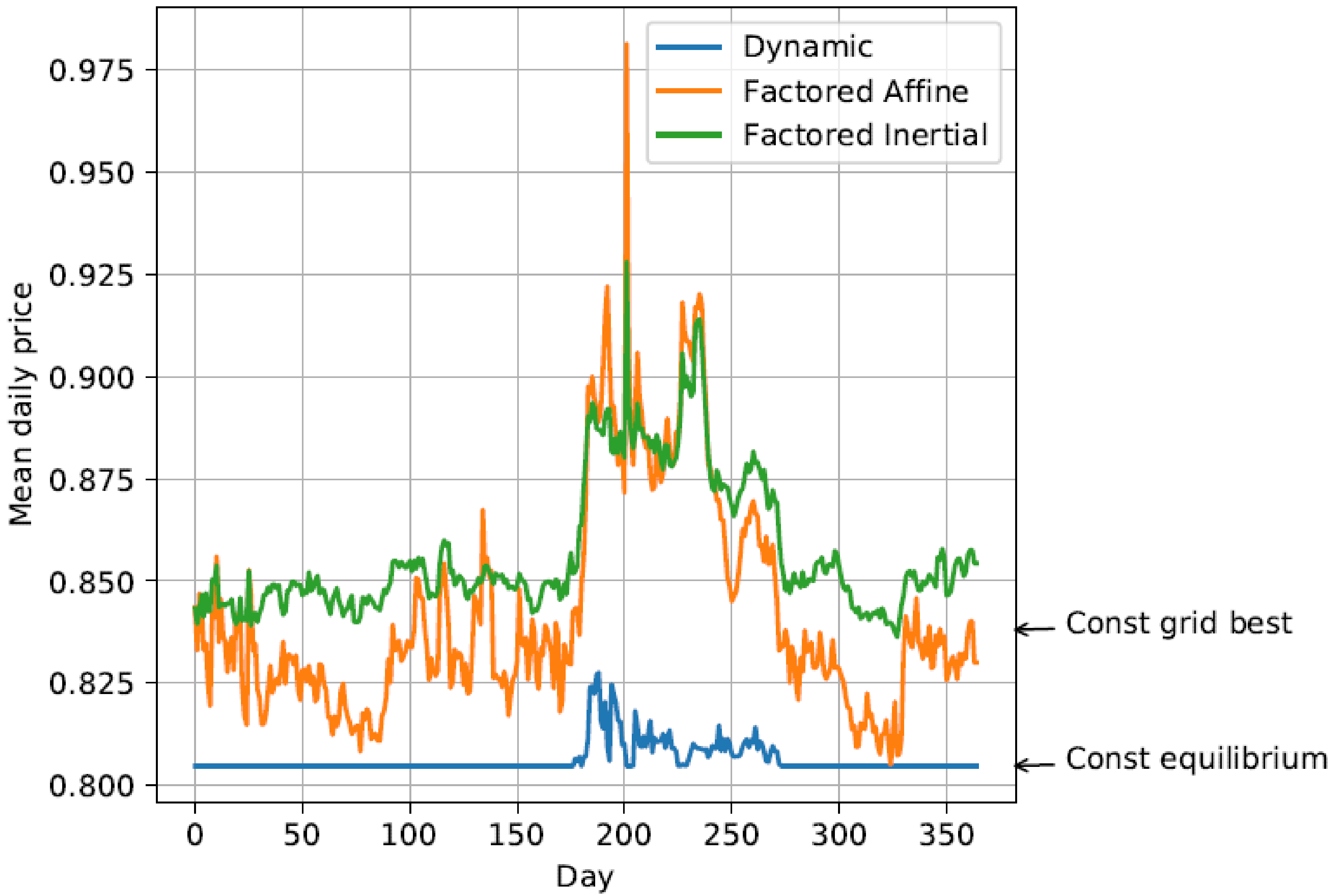


Fig. 4 Performance of the five pricing policies described in the text, combined when suitable with the three optimization techniques, for hotels with a varying number of rooms. From top left to bottom right: a 10-room hotel, often saturated, 30 rooms, 50 rooms, and 100 rooms. Boxes represent quartiles, the mean is represented by triangles, circles represent outliers (outside the $\frac{3}{2}$IQR-sized whiskers).



Dynamic
Factored Affine
Factored Inertial
Mean daily price
0.800
0.825
0.850
0.875
0.900
0.925
0.950
0.975
Day
0
50
100
150
200
250
300
350
Const grid best
Const equilibrium

Conclusions

- In complex contexts, the **simplifying assumptions that make the Dynamic Programming policy solvable** (e.g., reservations in different days do not interfere) **are too restrictive**, and the policy does not achieve good results.
- **Parametric pricing policies meaningfully improve the revenue**, particularly in the saturated case. The reactive optimizers show a consistently good performance

Conclusions

- **Flexibility**: **by combining ML models with optimization, one can make arbitrary changes in the model** of demand and customer behavior without impacting the way the optimization algorithm functions.

- **We are in 2020, not in 1950.**

We can solve/improve problems like complex Revenue Management situations which were impossible in the last century ☺


WENGER

Thanx! Roberto.Battiti@unitn.it

Ciaomanager *is ready with a RM system (Sinapsi) following the presented ideas*