RANDOMIZATION
in SYSTEMS and CONTROL:
a CHANGE of PERSPECTIVE

Marco C. Campi

with …

Simone Garatti

Giuseppe Calafiore

Erik Weyer

Maria Prandini

Algo Care’

outline:

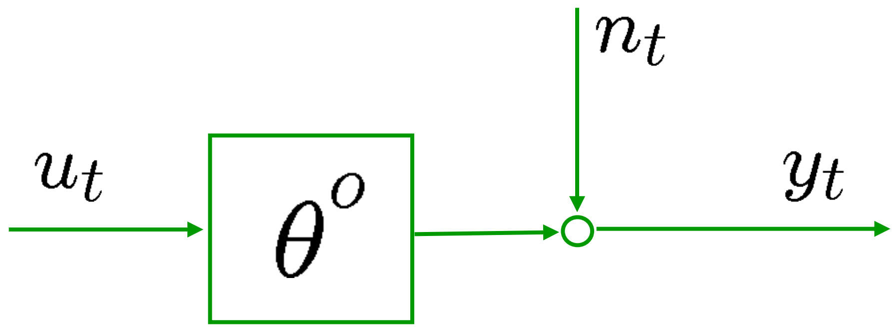
(i) focusing on randomization:
an example in estimation theory

(ii) what is a randomized algorithm?

(iii) where are we?

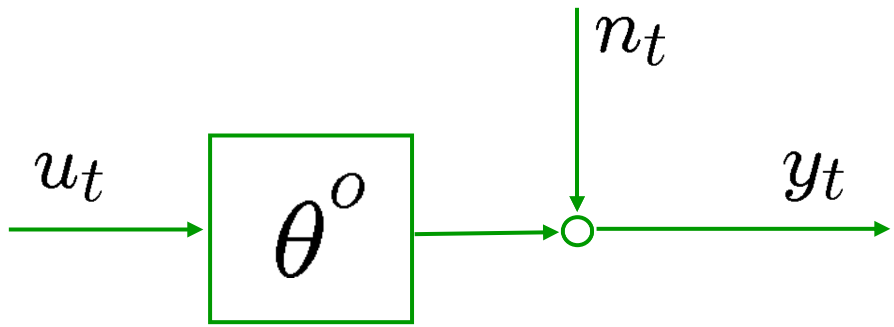
(iv) uncertain systems

an estimation problem



$$y_t = \theta^o u_t + n_t$$

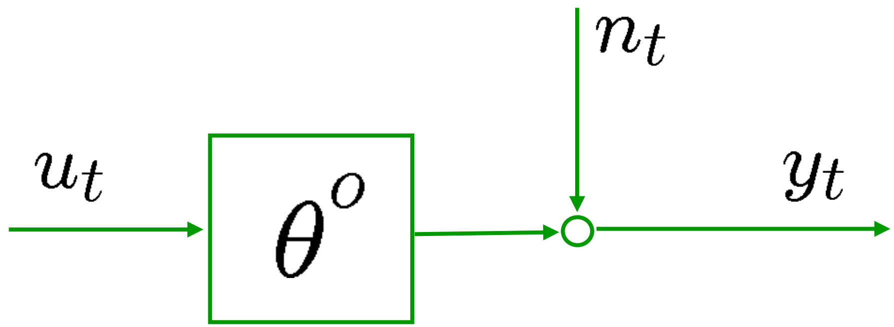
an estimation problem



$$y_t = \theta^o u_t + n_t$$

- select $u_t, \ t = 1, \ldots, N$
- measure y_t

an estimation problem



$$y_t = \theta^o u_t + n_t$$

- select $u_t, \ t = 1, \ldots, N$
- measure y_t

goal: provide an interval Θ for θ^o

$$M\colon \ (u_t, y_t) \ \rightarrow \ \Theta$$

an estimation problem

<u>requirements:</u>

an estimation problem

requirements:

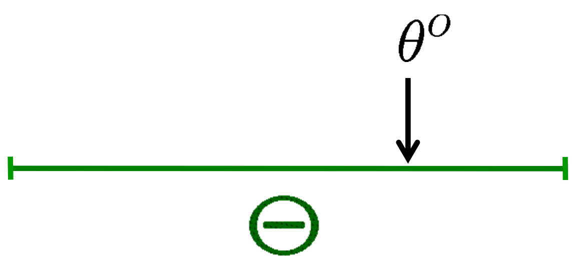
(i) no assumptions on n_t

an estimation problem

requirements:

(i) no assumptions on n_t

(ii) we want to issue a certificate of reliability on Θ valid $\forall n_t$



is this at all possible?

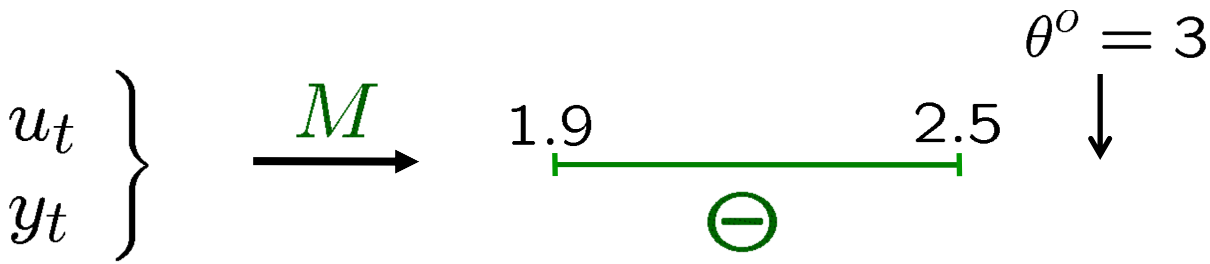
. . . no deterministic algorithm solves the problem

… no deterministic algorithm solves the problem

- take an input sequence u_t
- measure y_t

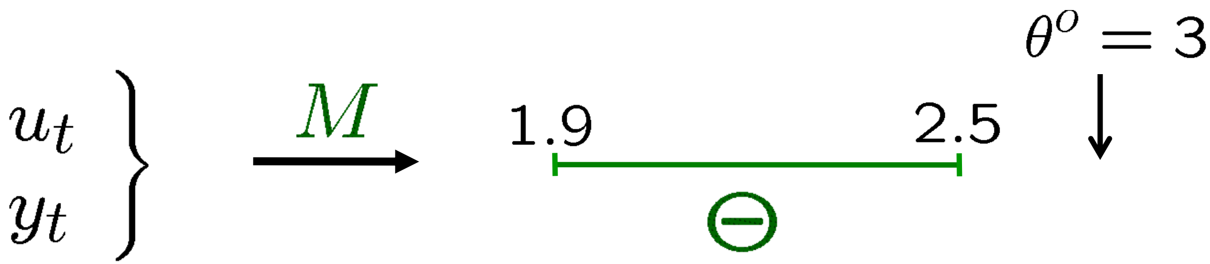
… no deterministic algorithm solves the problem

- take an input sequence u_t
- measure y_t



… no deterministic algorithm solves the problem

- take an input sequence u_t
- measure y_t



$$y_t = 3u_t + n_t$$

$$\Downarrow$$

$$n_t = -3u_t + y_t$$

- a deterministic algorithm that comes with a certificate of reliability does not exist!

- a deterministic algorithm that comes with a certificate of reliability does not exist!

- shall we give up?

moving one step ahead …
a change of perspective

moving one step ahead …
a change of perspective

Algorithm

1) $$u_t = \begin{cases} +1, & \text{with prob} = 0.5 \\ -1, & \text{with prob} = 0.5 \end{cases}$$

(R. Fisher)

moving one step ahead …
a change of perspective

Algorithm

1) $u_t = \begin{cases} +1, & \text{with prob} = 0.5 \\ -1, & \text{with prob} = 0.5 \end{cases}$ (R. Fisher)

2) M: $(u_t, y_t) \longrightarrow \ominus$ needs some explanation

(work done with E. Weyer)

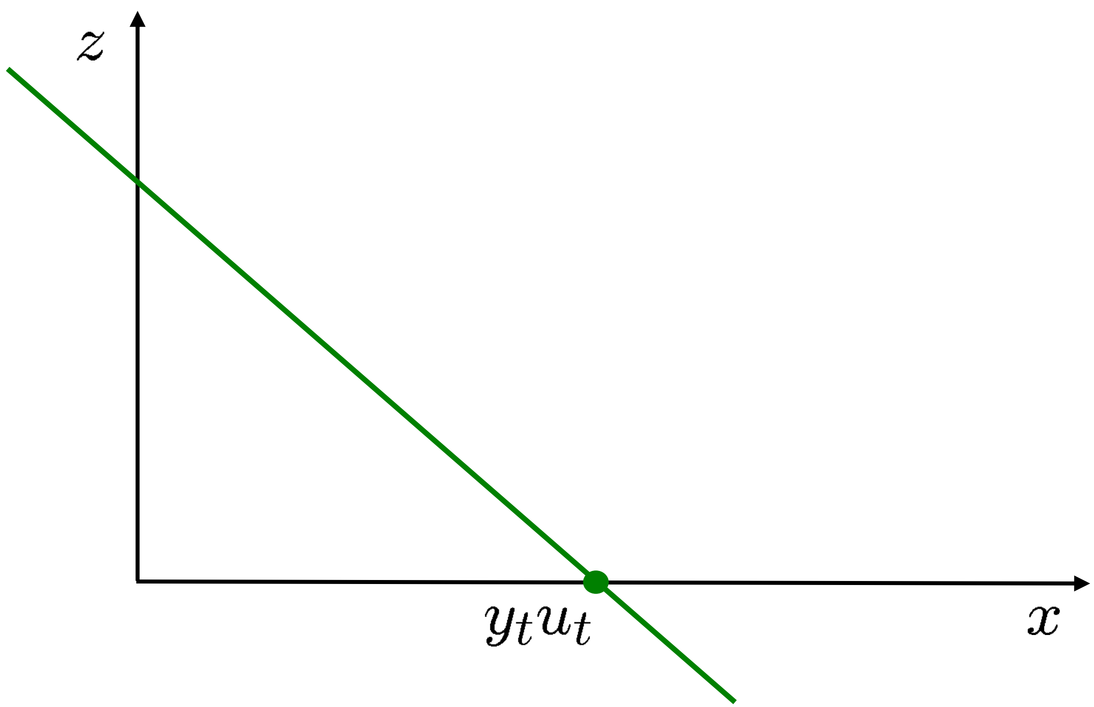
explaining M

number of observations $= 7$

explaining M

number of observations $= 7$

$z = y_t \cdot u_t - x$ -45 degrees line $t = 1, 2, \ldots, 7$



explaining M

number of observations $= 7$

$z = y_t \cdot u_t - x$ -45 degrees line $t = 1, 2, \ldots, 7$

explaining M

number of observations $= 7$

$z = y_t \cdot u_t - x$ -45 degrees line $t = 1, 2, \ldots, 7$

construct some averages:

$$\frac{1}{|T_1|} \sum_{t \in T_1} [y_t \cdot u_t - x]$$

$$\frac{1}{|T_2|} \sum_{t \in T_2} [y_t \cdot u_t - x]$$

$$\vdots$$

explaining M

numerical example:

u_t	-1	1	1	1	-1	1	-1
y_t	-0.45	0.97	1.74	1.80	0.1	1.92	-0.73

explaining M

numerical example:

u_t	-1	1	1	1	-1	1	-1
y_t	-0.45	0.97	1.74	1.80	0.1	1.92	-0.73

	1	2	3	4	5	6	7
T_1	●	●	○	●	●	○	○
T_2	●	○	●	●	○	●	○
T_3	○	●	●	○	●	●	○
T_4	●	●	○	○	○	●	●
T_5	●	○	●	○	●	○	●
T_6	○	●	●	●	○	○	●
T_7	○	○	○	●	●	●	●

$$\Longrightarrow \frac{1}{4} \sum_{t \in \{1,2,4,5\}} [y_t \cdot u_t - x]$$

$$\Longrightarrow \frac{1}{4} \sum_{t \in \{1,3,4,6\}} [y_t \cdot u_t - x]$$

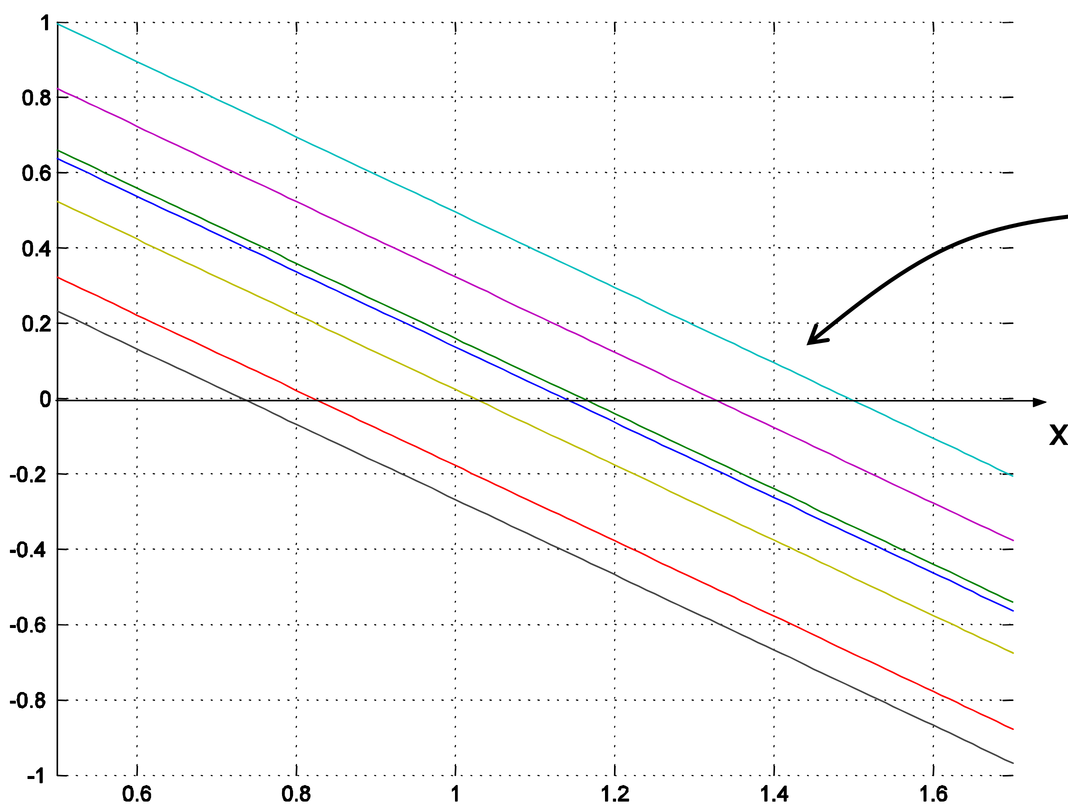
$$\Longrightarrow \frac{1}{4} \sum_{t \in \{2,3,5,6\}} [y_t \cdot u_t - x]$$

etc.

explaining M

numerical example:

u_t	-1	1	1	1	-1	1	-1
y_t	-0.45	0.97	1.74	1.80	0.1	1.92	-0.73

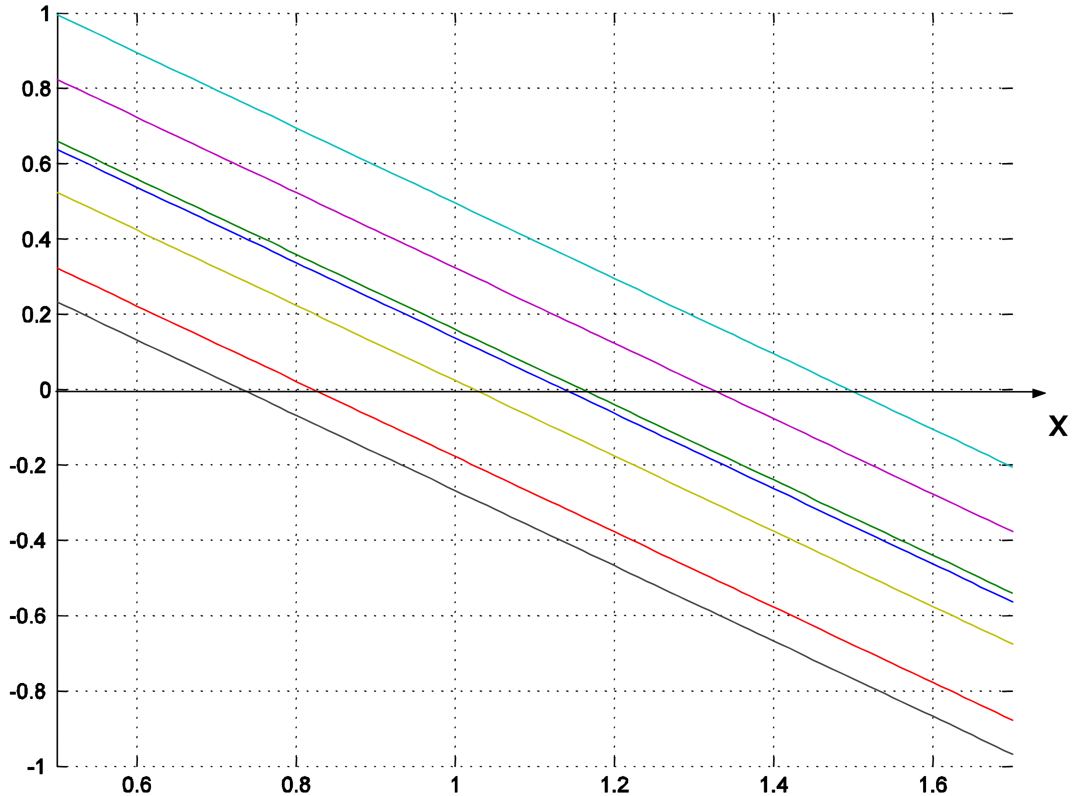


$\frac{1}{4}\sum_{t\in T}[y_t \cdot u_t - x]$

explaining M

$$\frac{1}{4}\sum_{t\in\{1,2,4,5\}}[y_t\cdot u_t - x] = [\theta^o - x] + \frac{1}{4}\sum_{t\in\{1,2,4,5\}} n_t\cdot u_t$$

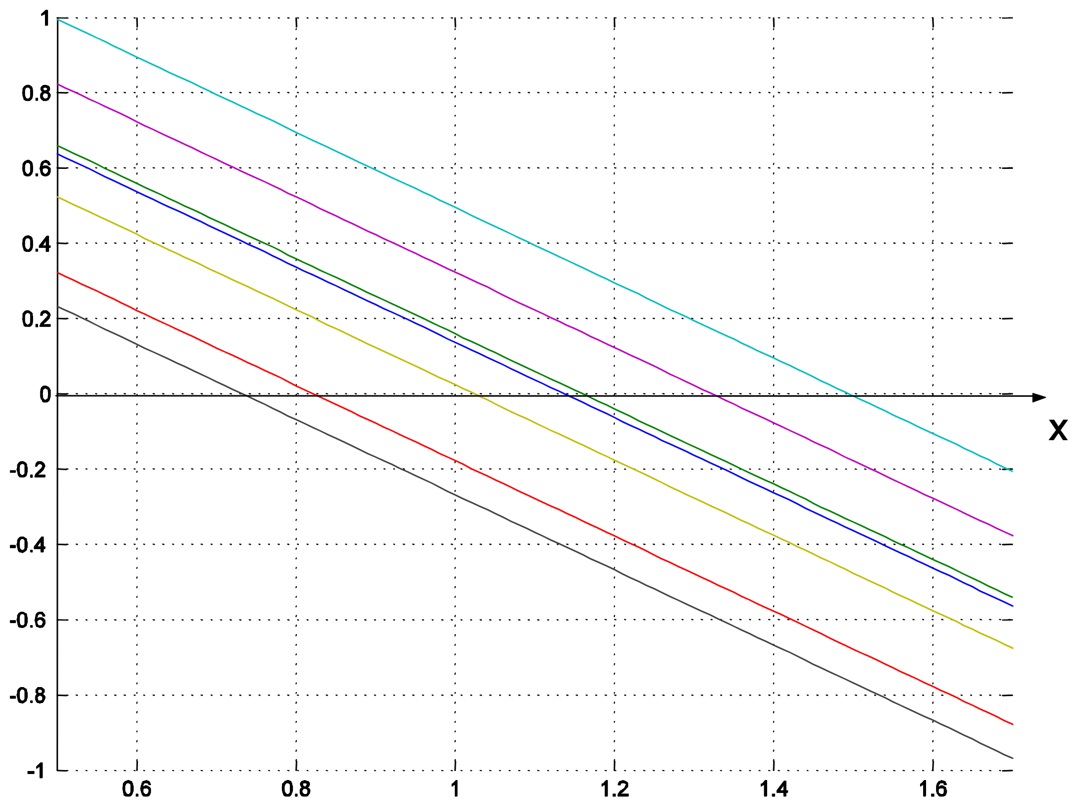
$y_t = \theta^o u_t + n_t$



explaining M

$$\frac{1}{4}\sum_{t\in\{1,2,4,5\}}[y_t\cdot u_t - x] = [\theta^o - x] + \frac{1}{4}\sum_{t\in\{1,2,4,5\}} n_t\cdot u_t$$

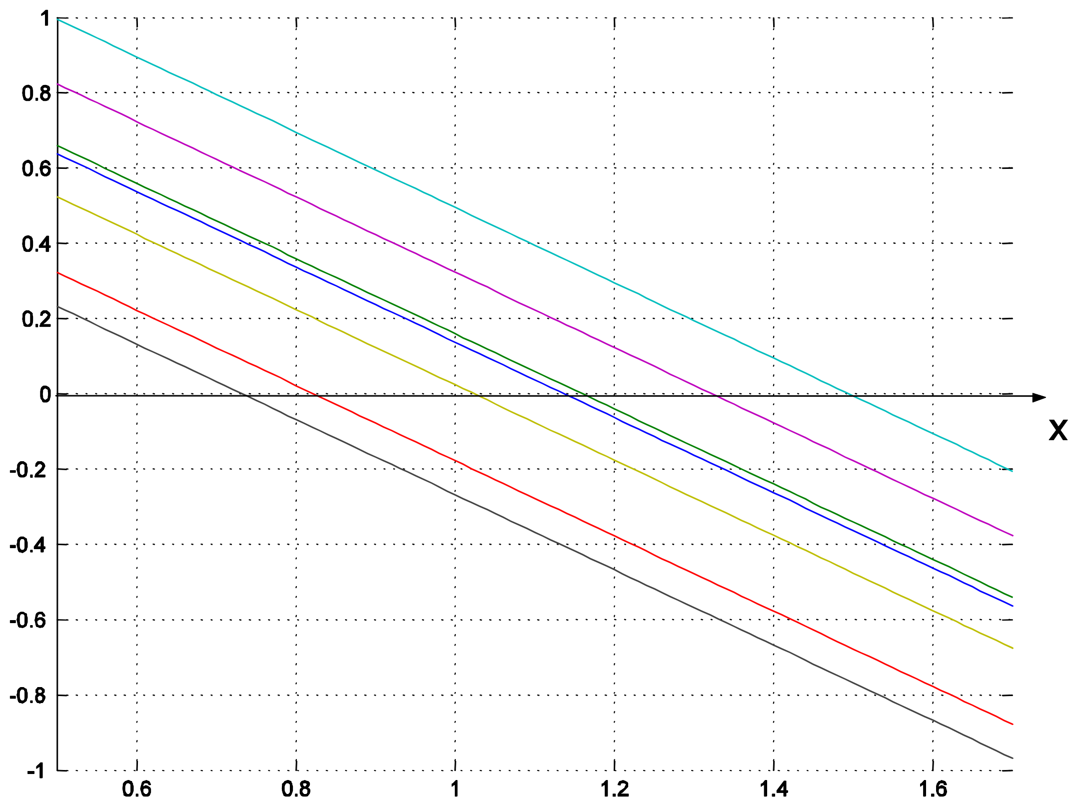
$y_t = \theta^o u_t + n_t$



explaining M

$$\frac{1}{4}\sum_{t\in\{1,2,4,5\}}[y_t \cdot u_t - x] = [\theta^o - x] + \frac{1}{4}\sum_{t\in\{1,2,4,5\}} n_t \cdot u_t$$

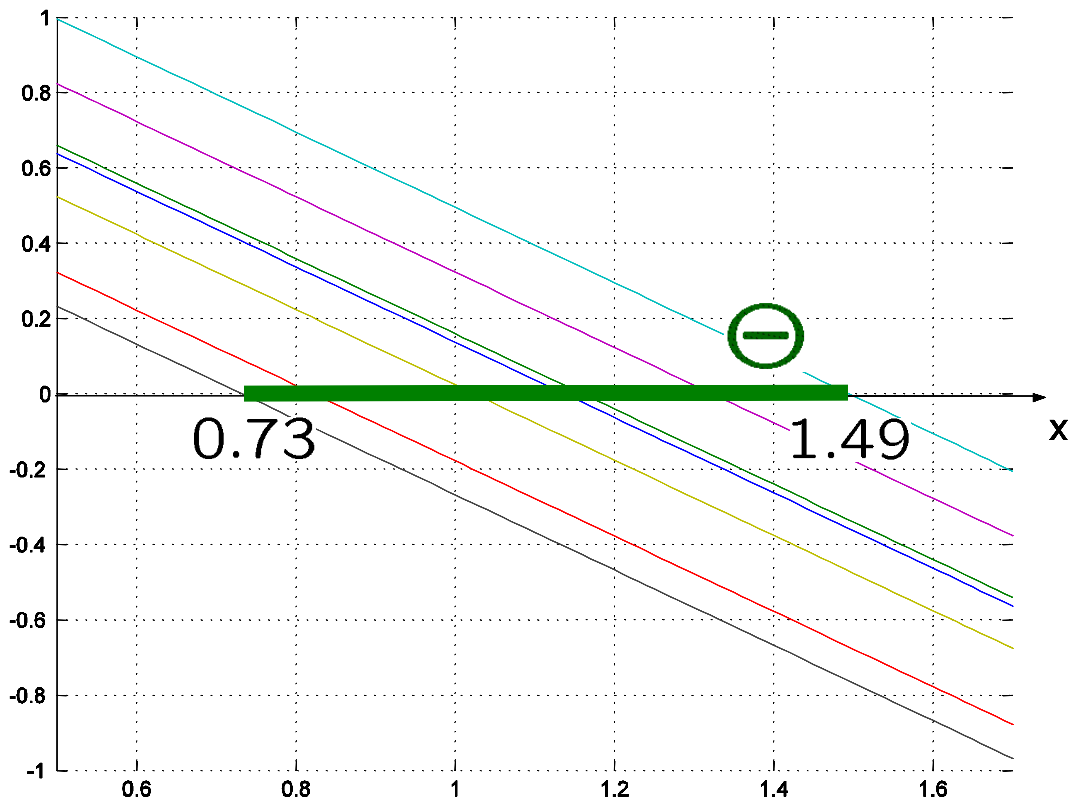
$\theta^o u_t + n_t$



explaining M

$$\frac{1}{4}\sum_{t\in\{1,2,4,5\}}[y_t{\cdot}u_t - x] = [\theta^o - x] + \frac{1}{4}\sum_{t\in\{1,2,4,5\}} n_t{\cdot}u_t$$

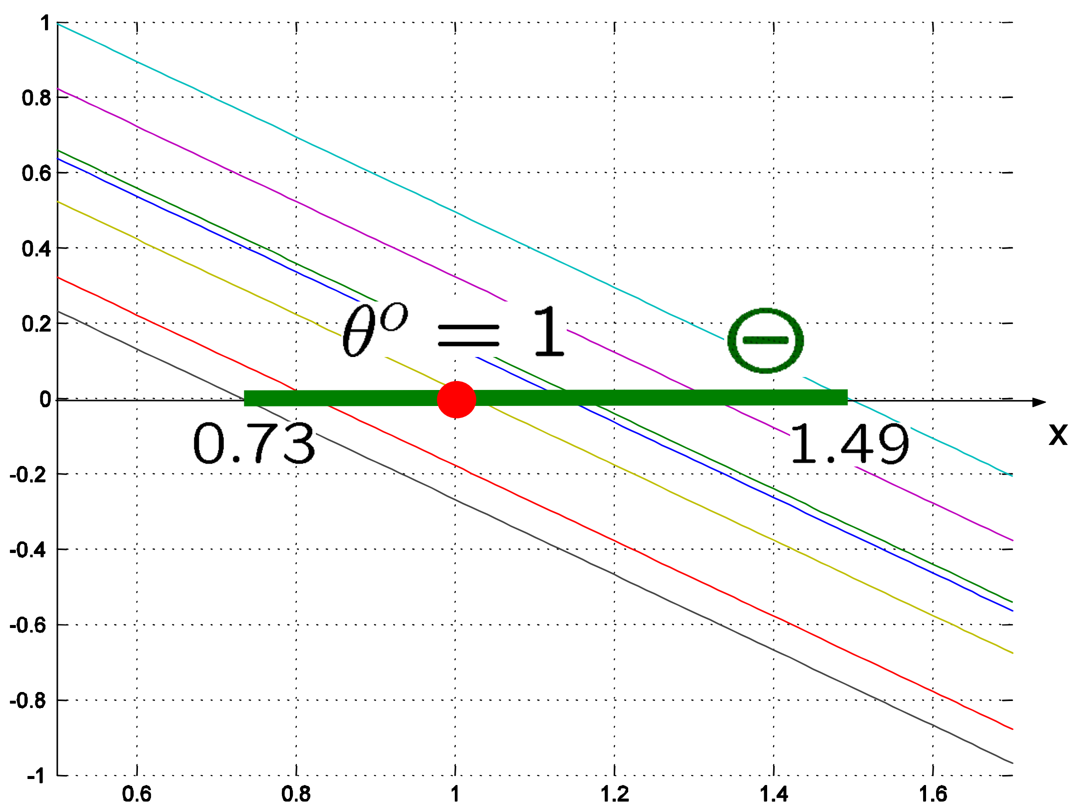
⊖ = not all functions are positive or all are negative



explaining M

$$\frac{1}{4}\sum_{t\in\{1,2,4,5\}}[y_t \cdot u_t - x] = [\theta^o - x] + \frac{1}{4}\sum_{t\in\{1,2,4,5\}} n_t \cdot u_t$$

⊖ = not all functions are positive or all are negative

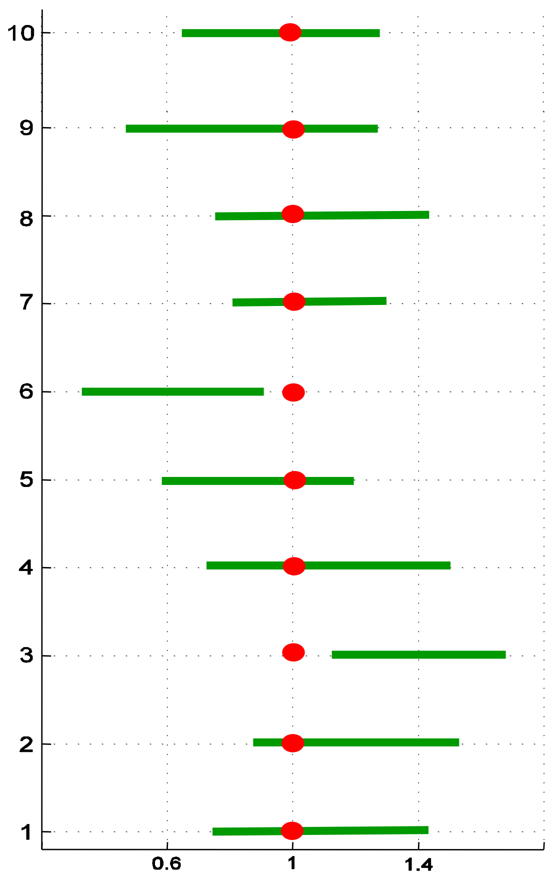


Theorem (with E. Weyer)

$\forall n_t, \quad \theta^o \in \Theta$ with probability 0.75.

Theorem (with E. Weyer)

$\forall n_t, \quad \theta^o \in \Theta$ with probability 0.75.



n_t is fixed

Theorem (with E. Weyer)

$\forall n_t, \quad \theta^o \in \Theta$ with probability 0.75.

Theorem (with E. Weyer)

$\forall n_t, \quad \theta^o \in \Theta$ with probability 0.75.

$\Theta = [0.73, 1.49]$ with probability $= 0.75$

$N = 7$

Theorem (with E. Weyer)

$\forall n_t, \quad \theta^o \in \Theta$ with probability 0.75.

$\Theta = [0.73, 1.49]$ with probability $= 0.75$

$N = 7$

$N = 1023$

$\Theta = [0.945, 1.033]$ with probability $= 0.975$

Theorem (with E. Weyer)

$\forall n_t, \quad \theta^o \in \Theta$ with probability 0.75.

Theorem (consistency)

Under general assumptions,

$\Theta \longrightarrow \{\theta^o\}$ as $N \rightarrow \infty$

in summary:

for any n_t:

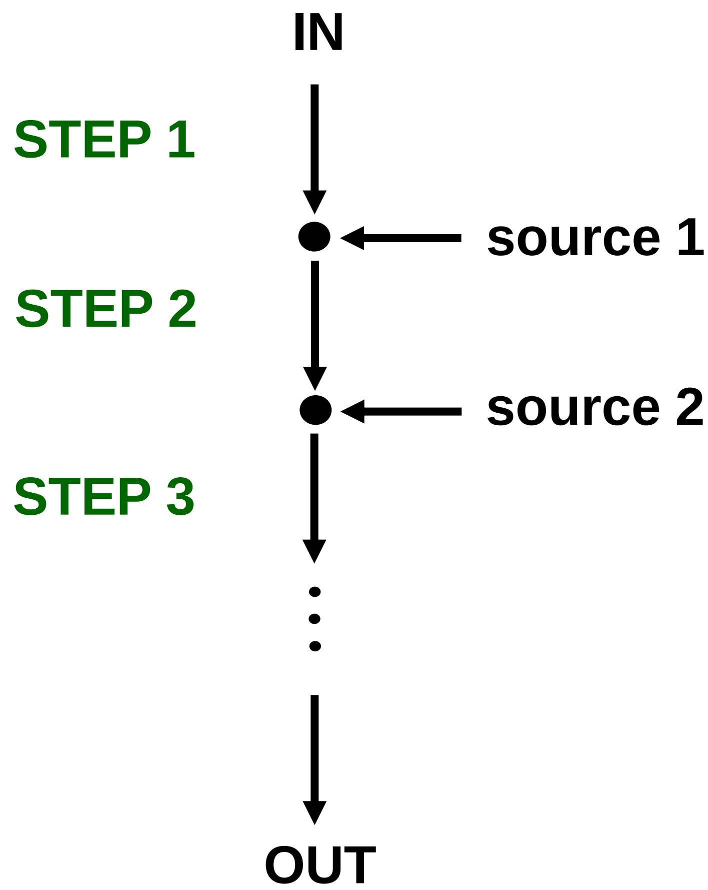
(i) $\theta^o \in \Theta$ with a precise probability

(ii) the size of Θ depends on the strength of the noise

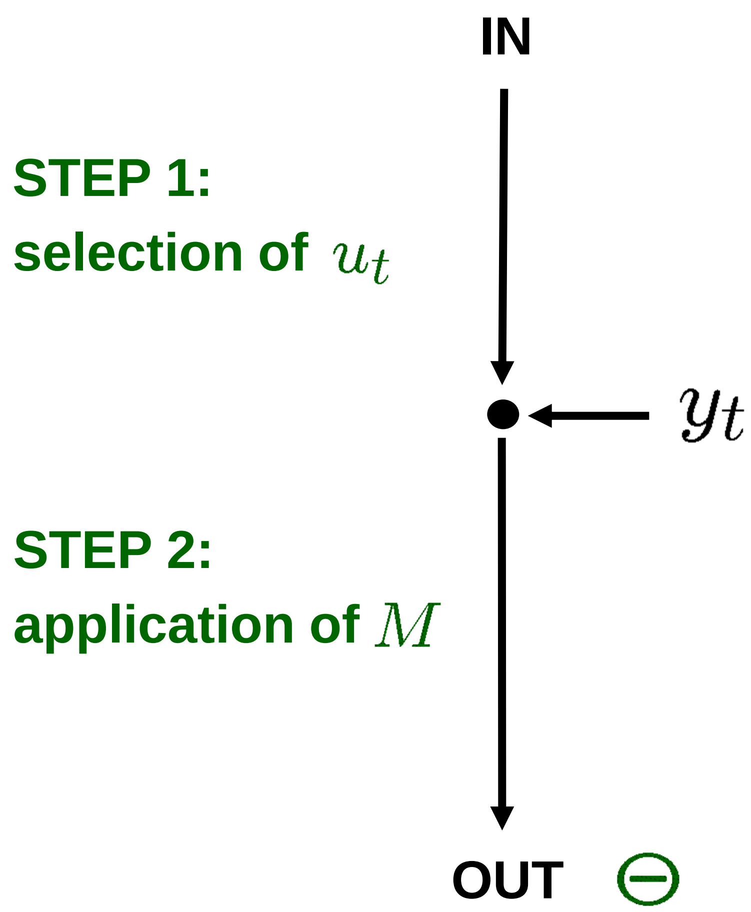
(iii) Θ shrinks around θ^o as *N* increases how is all this possible?

what is a randomized algorithm?

deterministic algorithm

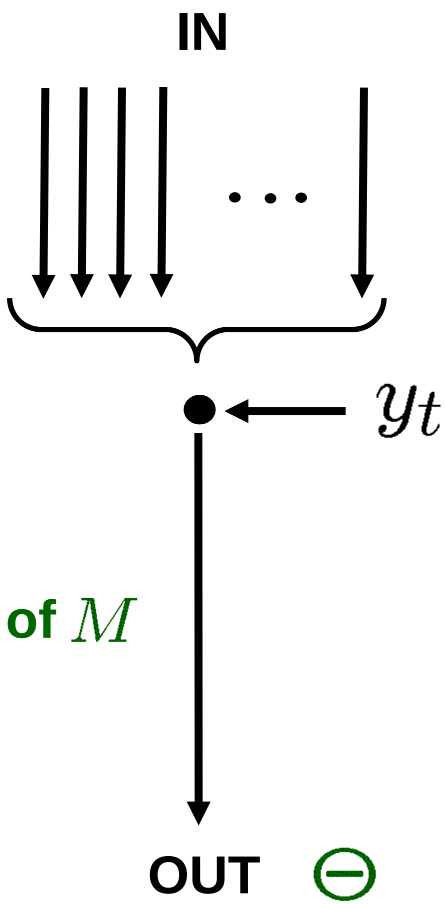


deterministic algorithm



randomized algorithm

STEP 1:
random selection of u_t



STEP 2:
application of M

randomized algorithm

a randomized algorithm is an algorithm where one or more steps are based on a random choice

that is – among many deterministic choices – one choice is selected at random according to a probability *P*

randomized algorithm

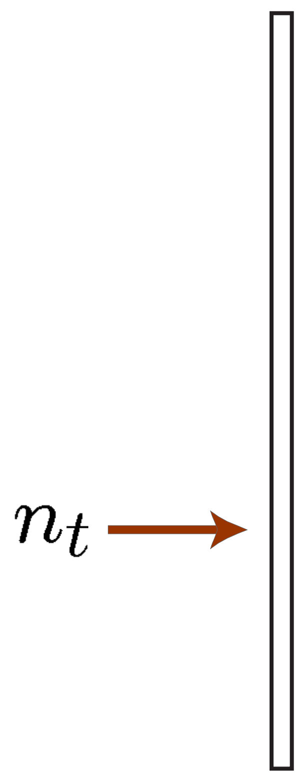
a randomized algorithm is an algorithm where one or more steps are based on a random choice

that is – among many deterministic choices – one choice is selected at random according to a probability P

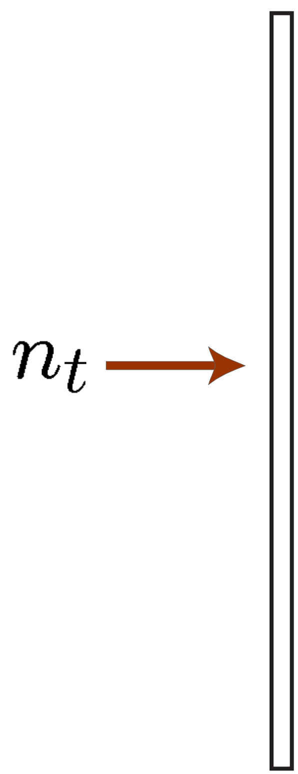
why is this useful?

deterministic vs. randomized

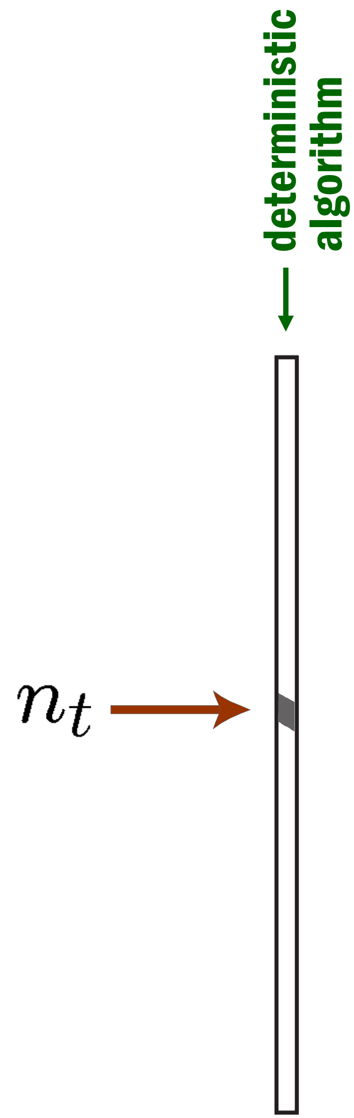
deterministic vs. randomized



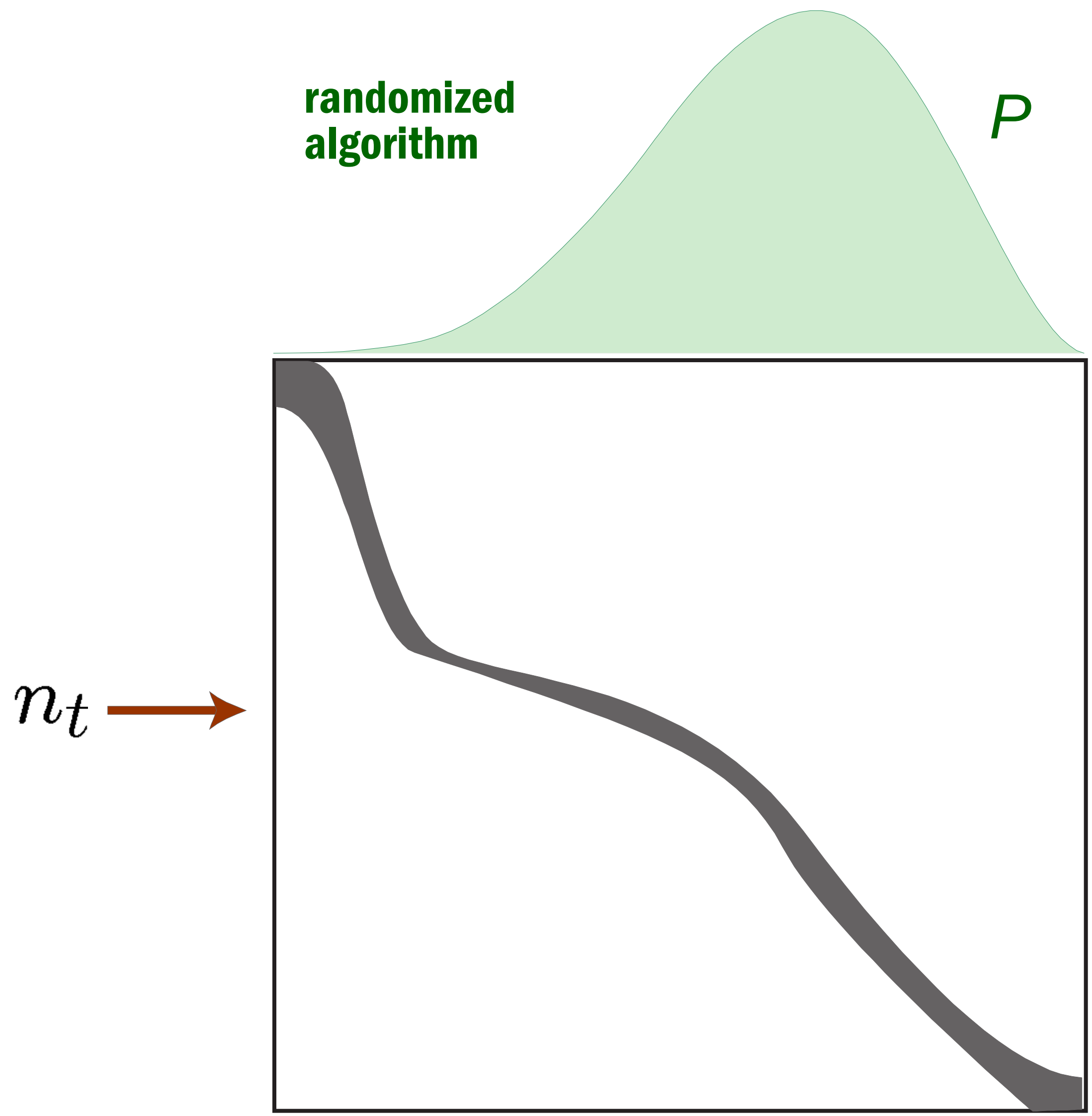
deterministic vs. randomized



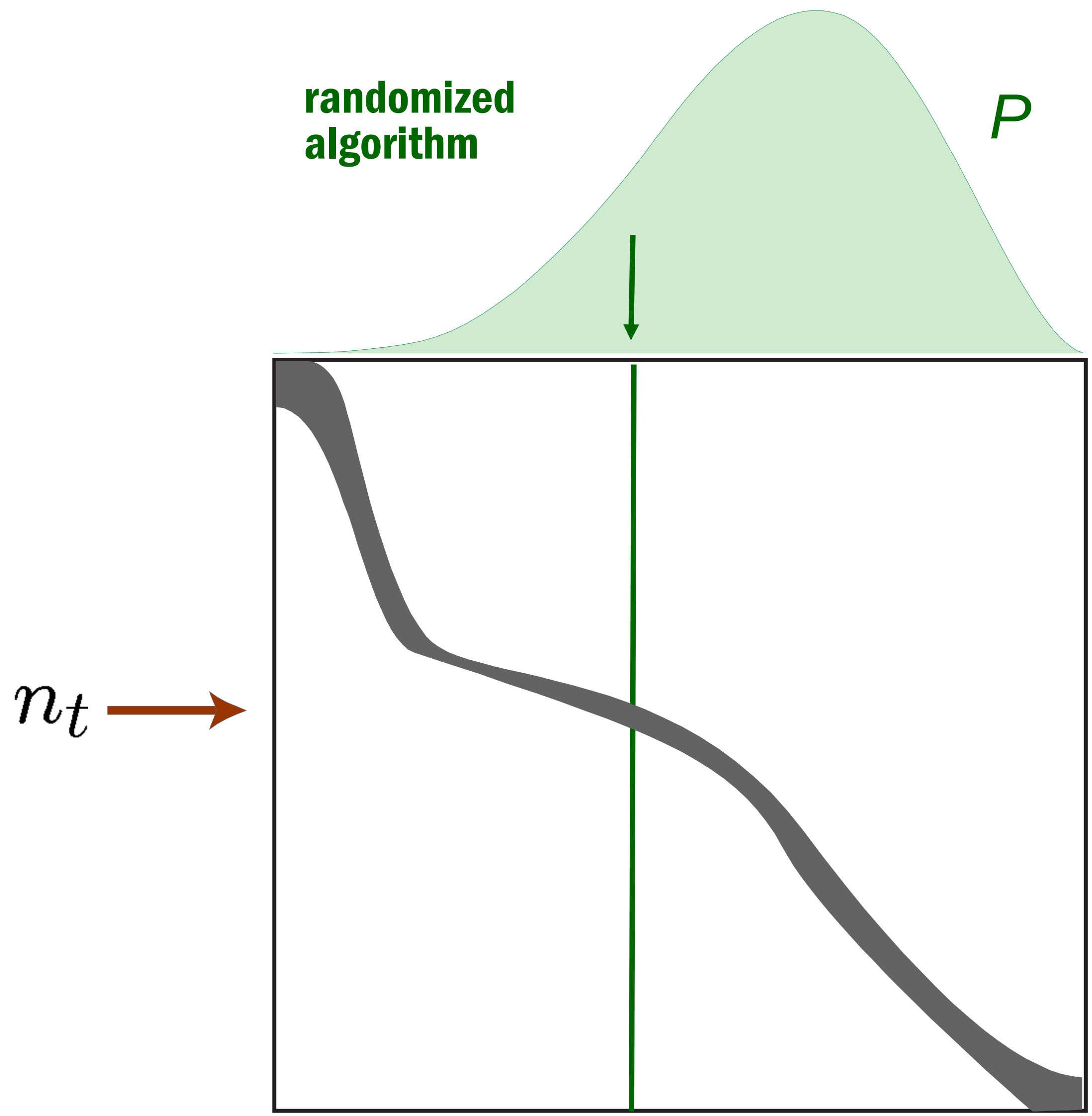
deterministic vs. randomized



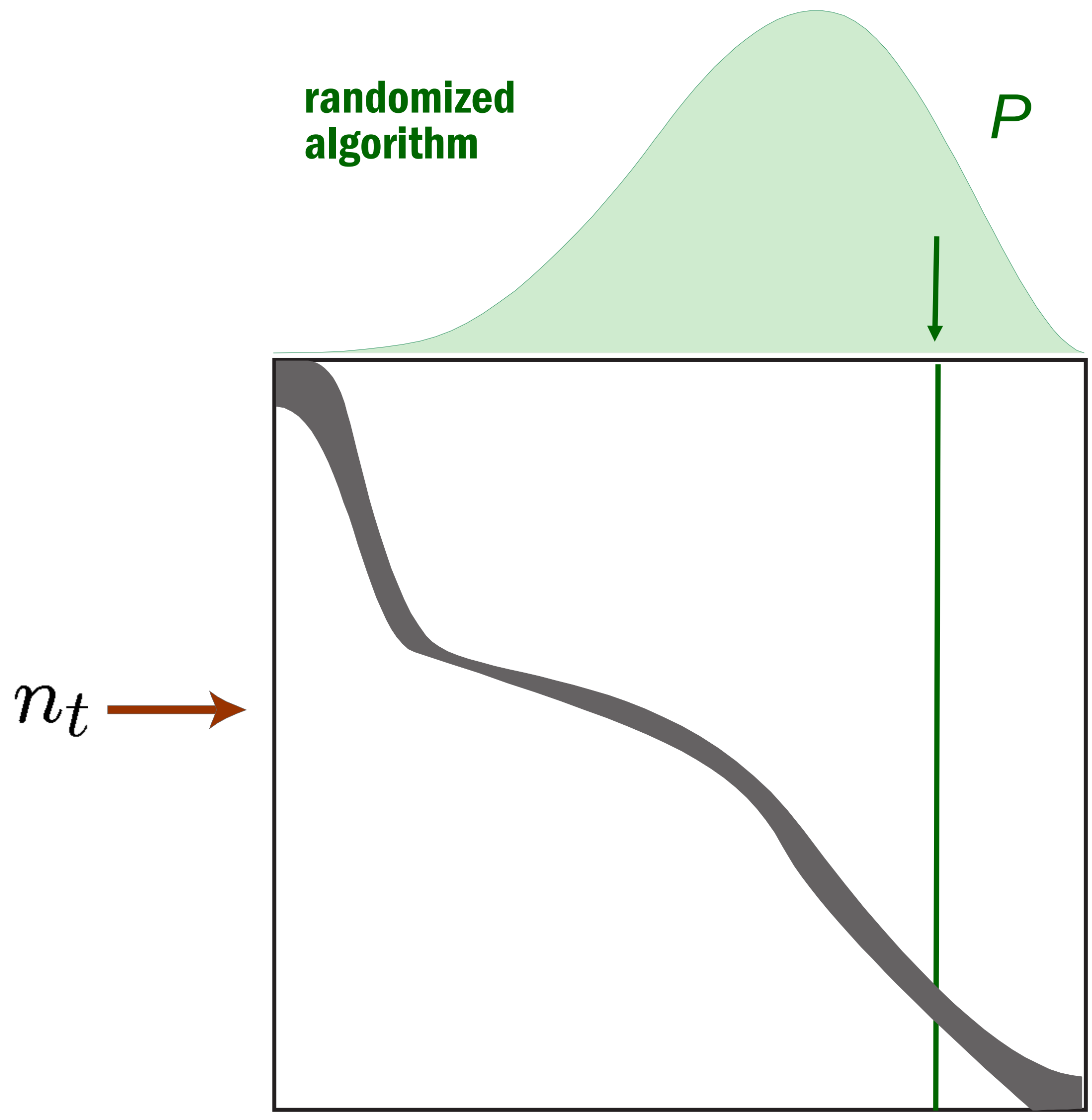
deterministic vs. randomized



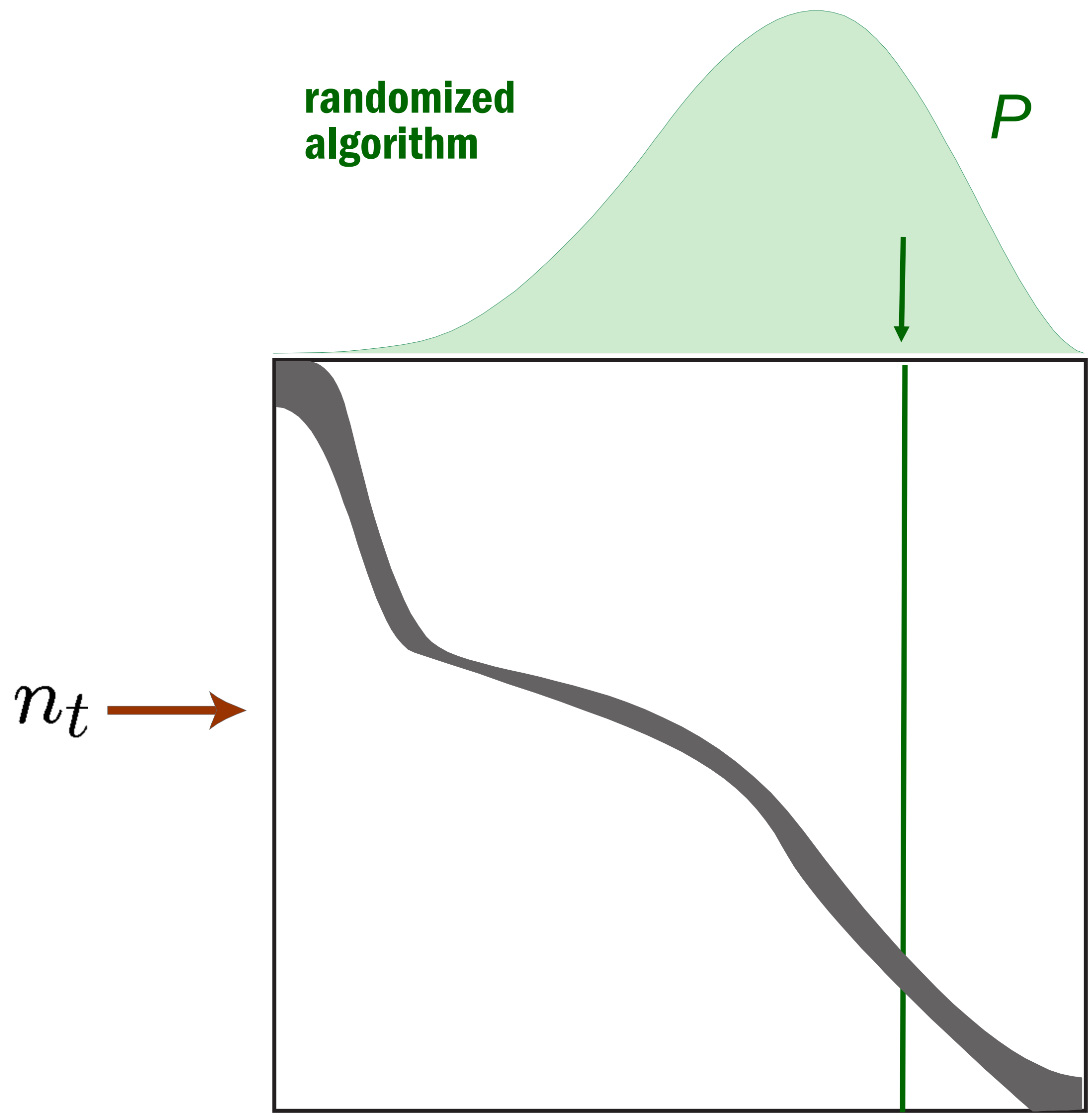
deterministic vs. randomized



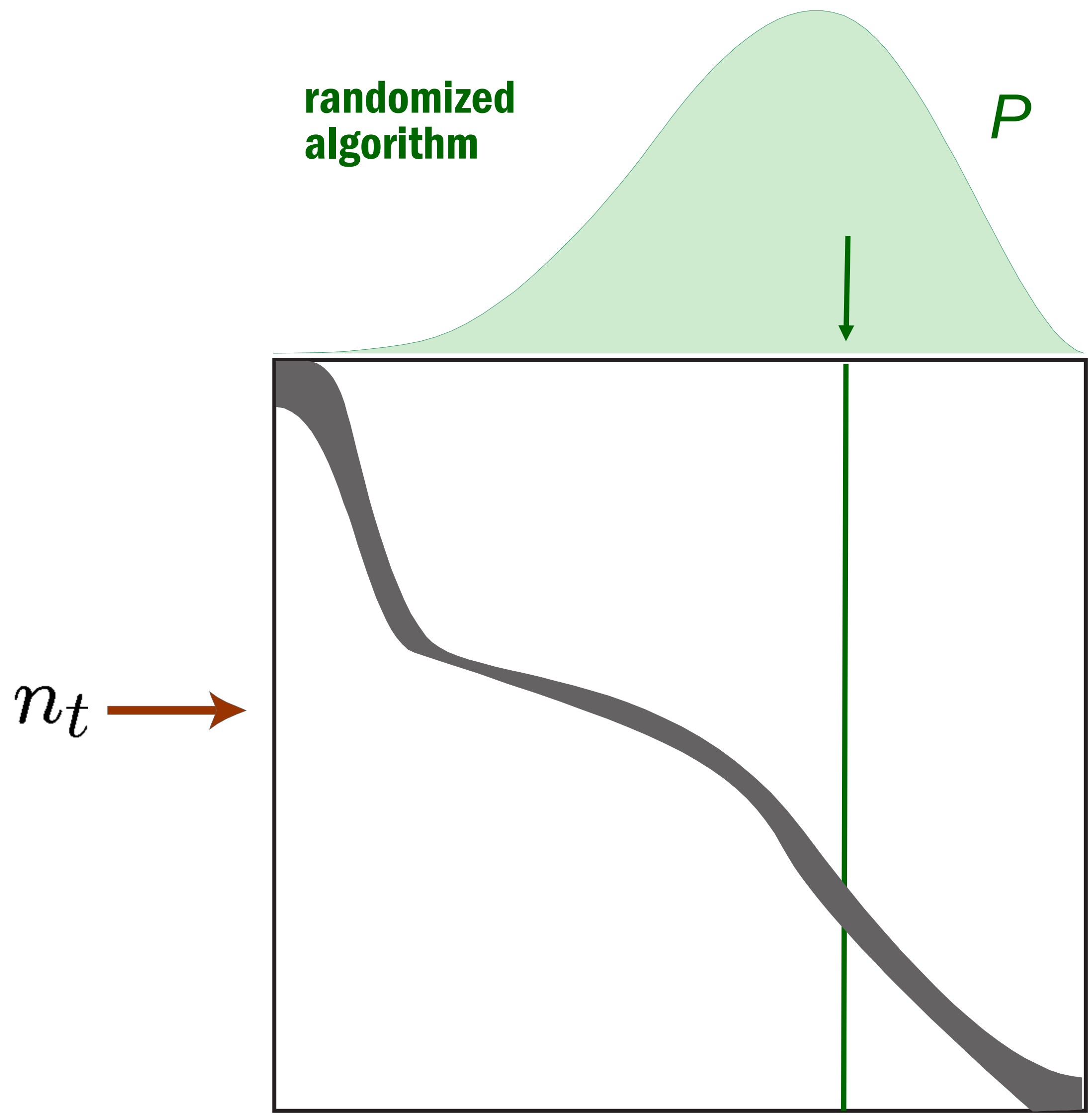
deterministic vs. randomized



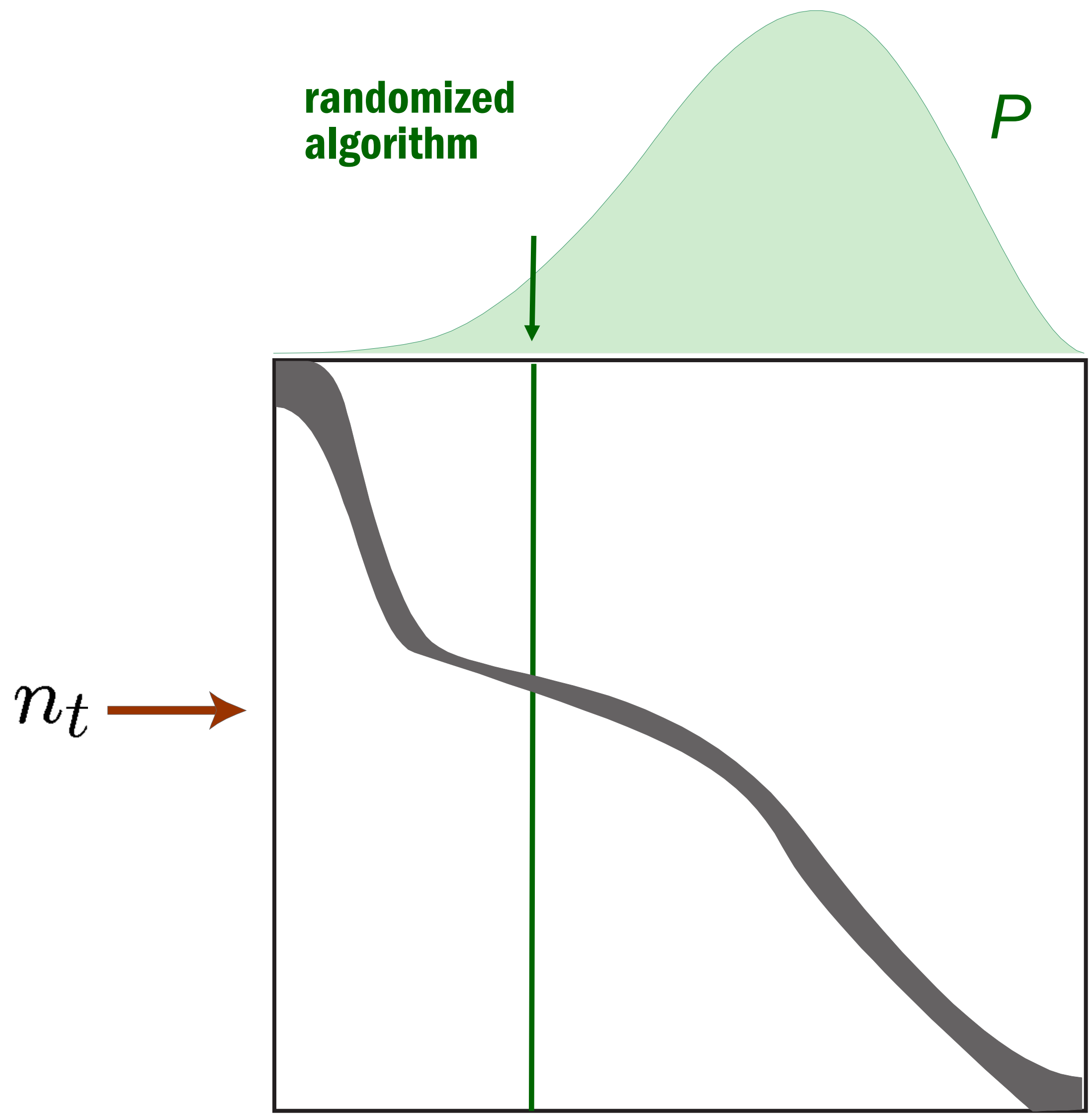
deterministic vs. randomized



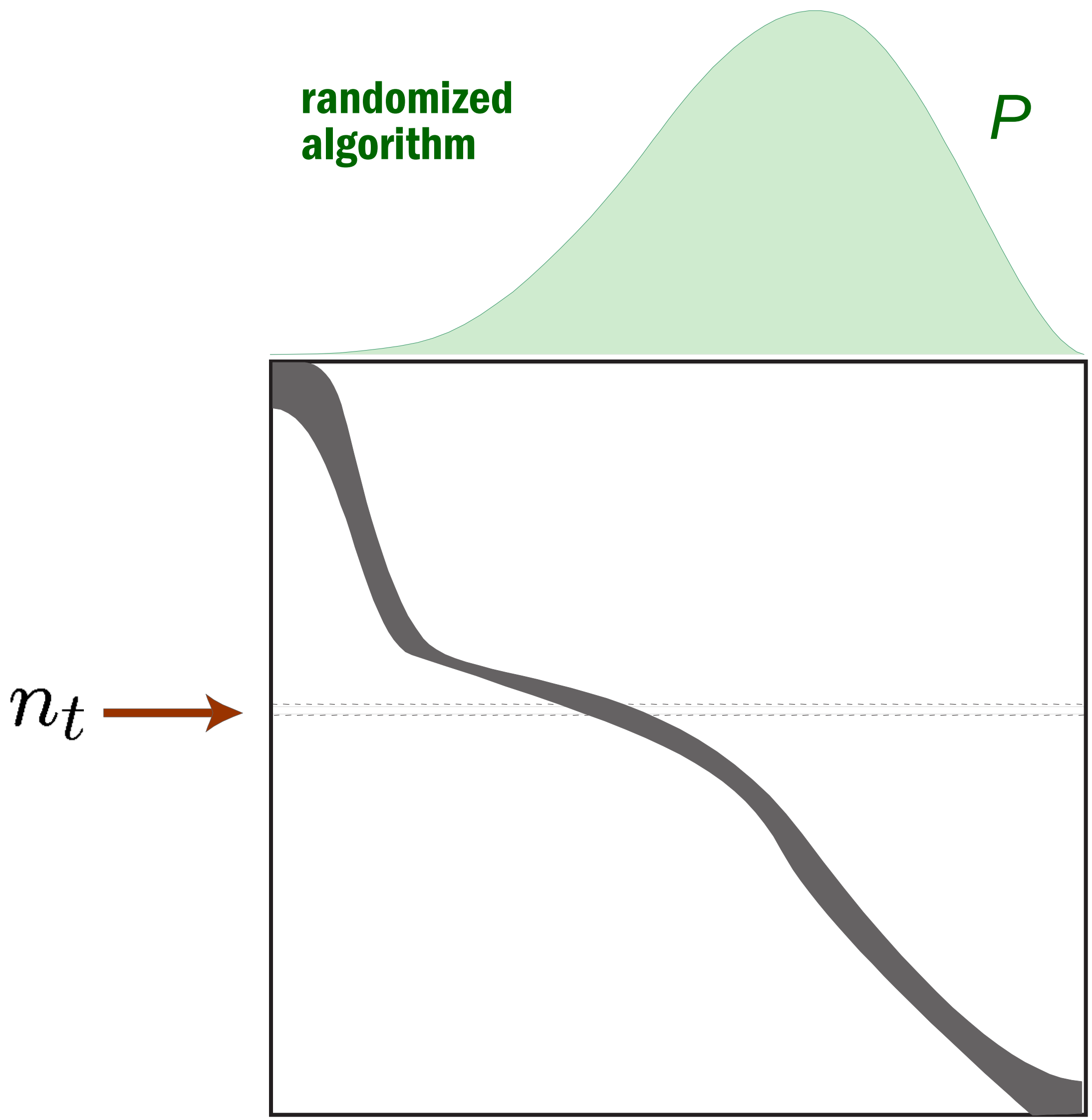
deterministic vs. randomized



deterministic vs. randomized



deterministic vs. randomized



successful algorithm

successful algorithm

Definition 1 (successful deterministic algorithm)
An algorithm is successful if, in all situations, it provides a correct answer.

successful algorithm

<u>Definition 1</u> (successful deterministic algorithm)
An algorithm is successful if, in all situations, it provides a correct answer.

… a change of perspective:

<u>Definition 2</u> (probabilistically successful algorithm)
An algorithm is successful with probability p if, in all situations, its probability to provide a correct answer is at least p.

successful algorithm

Definition 1 (successful deterministic algorithm)
An algorithm is successful if, in all situations, it provides a correct answer.

… a change of perspective:

Definition 2 (probabilistically successful algorithm)
An algorithm is successful with probability *p* if, in all situations, its probability to provide a correct answer is at least *p*.

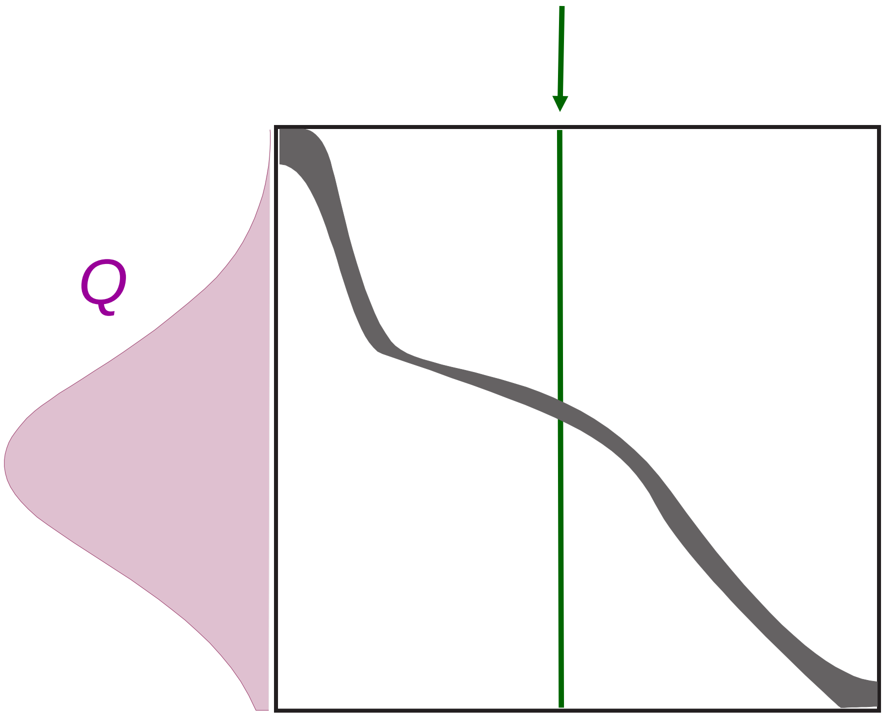
this offers an extraordinary opportunity to satisfactorily solve “hopeless” problems

comments

- *P* exists in the algorithm

comments

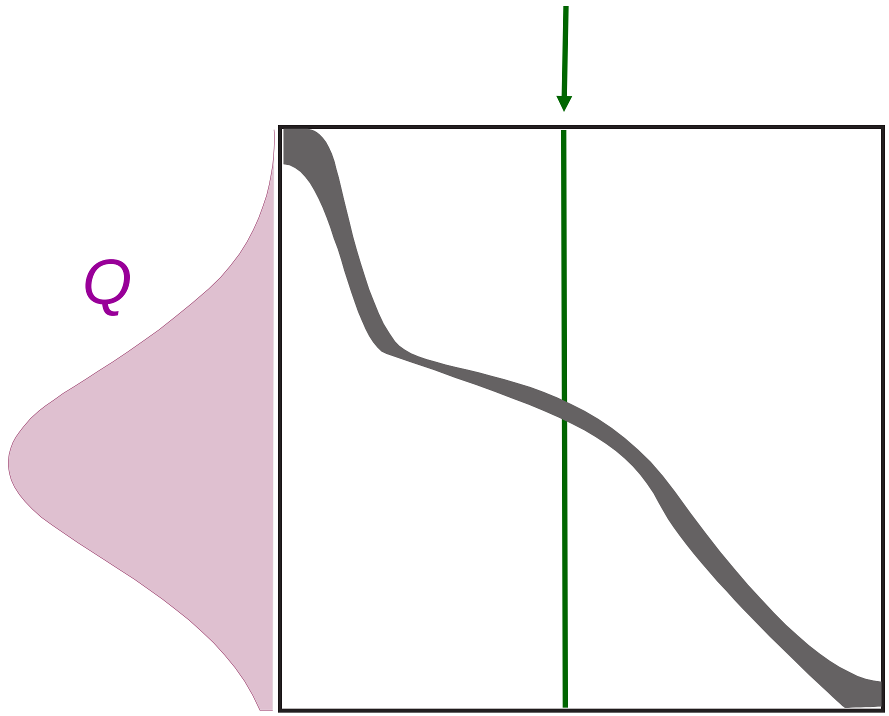
- *P* exists in the algorithm



Bayesian perspective

comments

- *P* exists in the algorithm

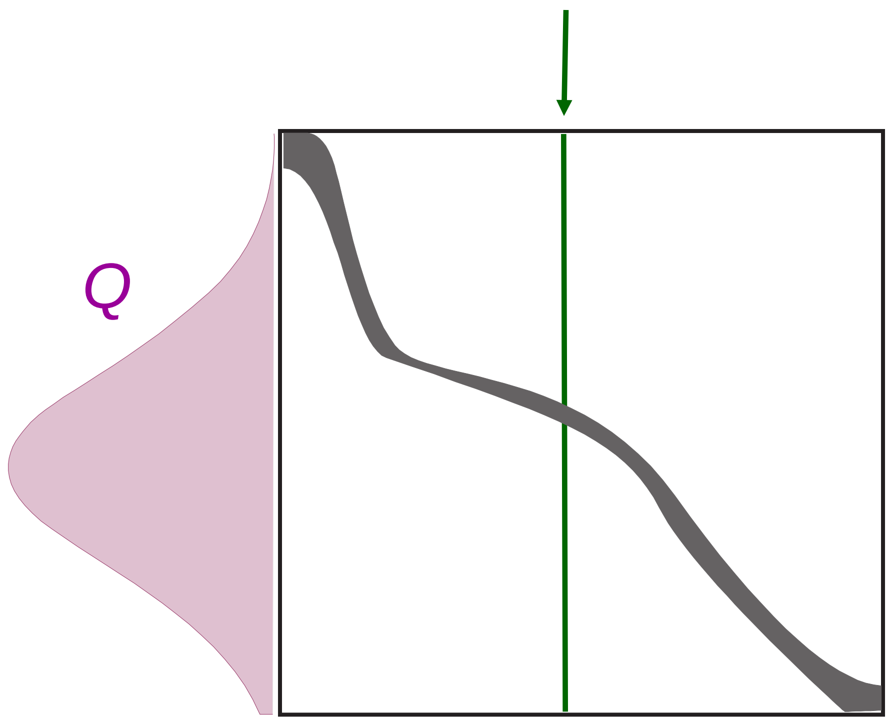


Bayesian perspective

(1) the result holds with high probability with respect to the situation of application

comments

- *P* exists in the algorithm



Bayesian perspective

(1) the result holds with high probability with respect to the situation of application

(2) *Q* describes reality → poor modeling

comments

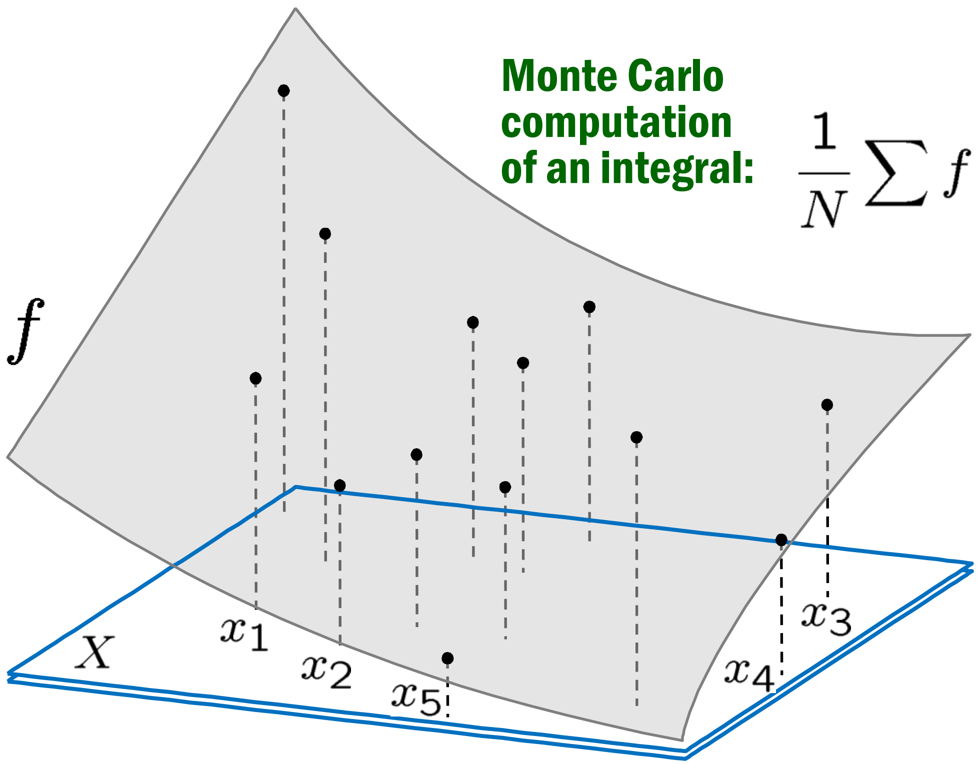
- *P* exists in the algorithm

comments

- *P* exists in the algorithm
- often the chance of failure can be made very small (concentration inequalities)

comments

- *P* exists in the algorithm
- often the chance of failure can be made very small (concentration inequalities)

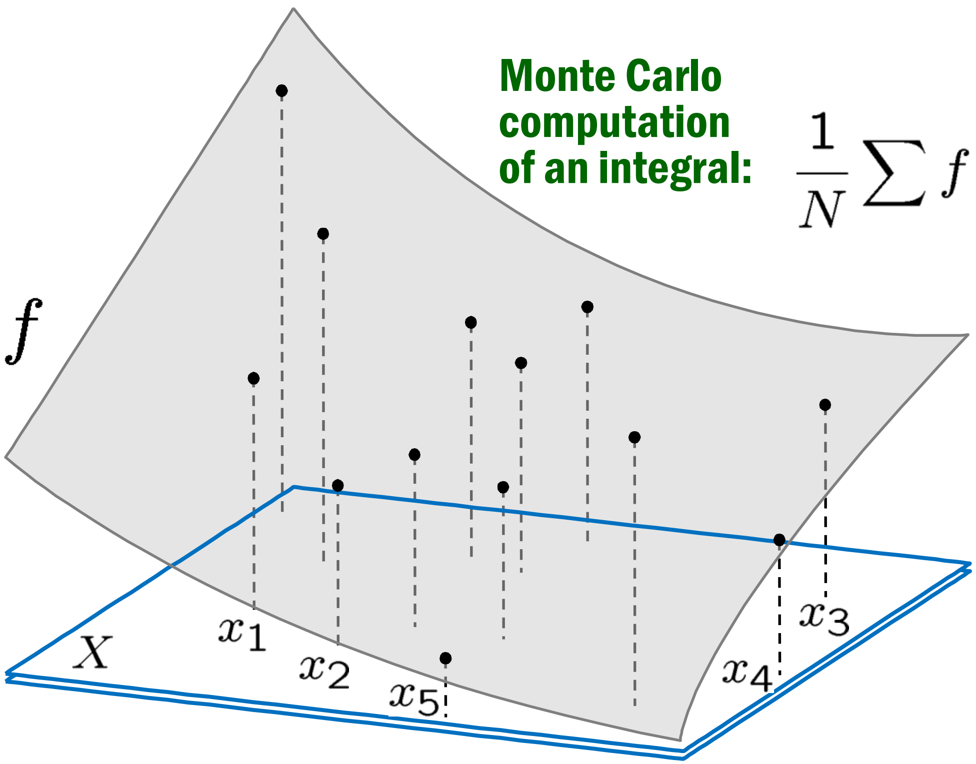


Monte Carlo computation of an integral: $\frac{1}{N}\sum f(x_i)$

comments

- *P* exists in the algorithm
- often the chance of failure can be made very small (concentration inequalities)

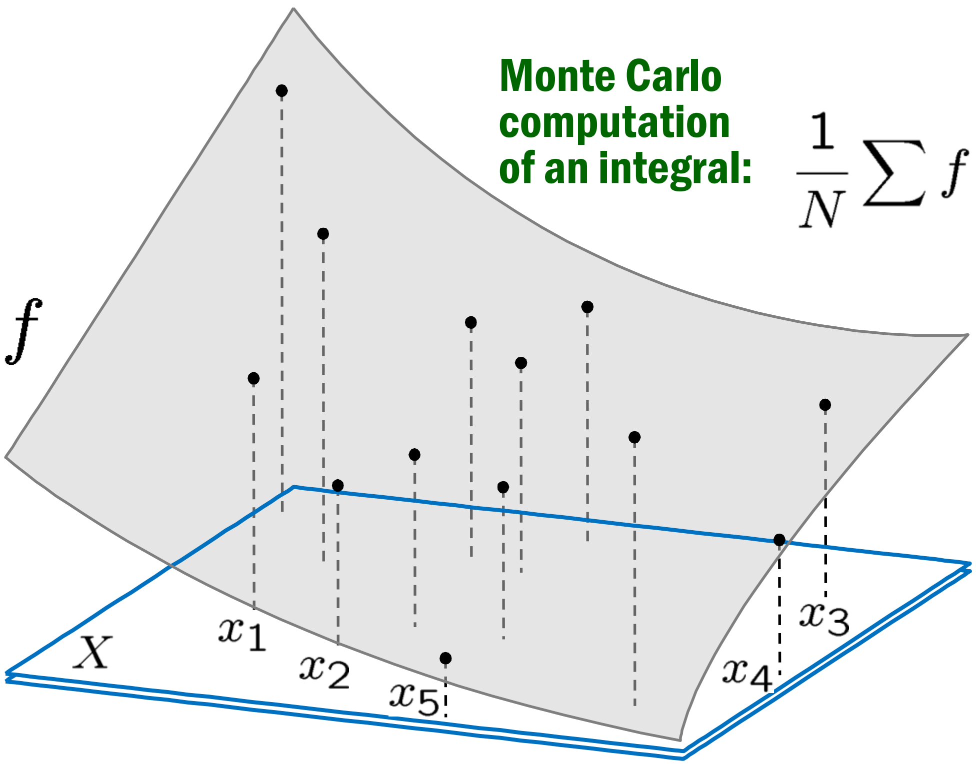
Monte Carlo computation of an integral: $\frac{1}{N}\sum f(x_i)$



$$\left|\int f - \frac{1}{N}\sum f(x_i)\right| \leq 0.05$$

comments

- *P* exists in the algorithm
- often the chance of failure can be made very small (concentration inequalities)



Monte Carlo computation of an integral: $\frac{1}{N}\sum f(x_i)$

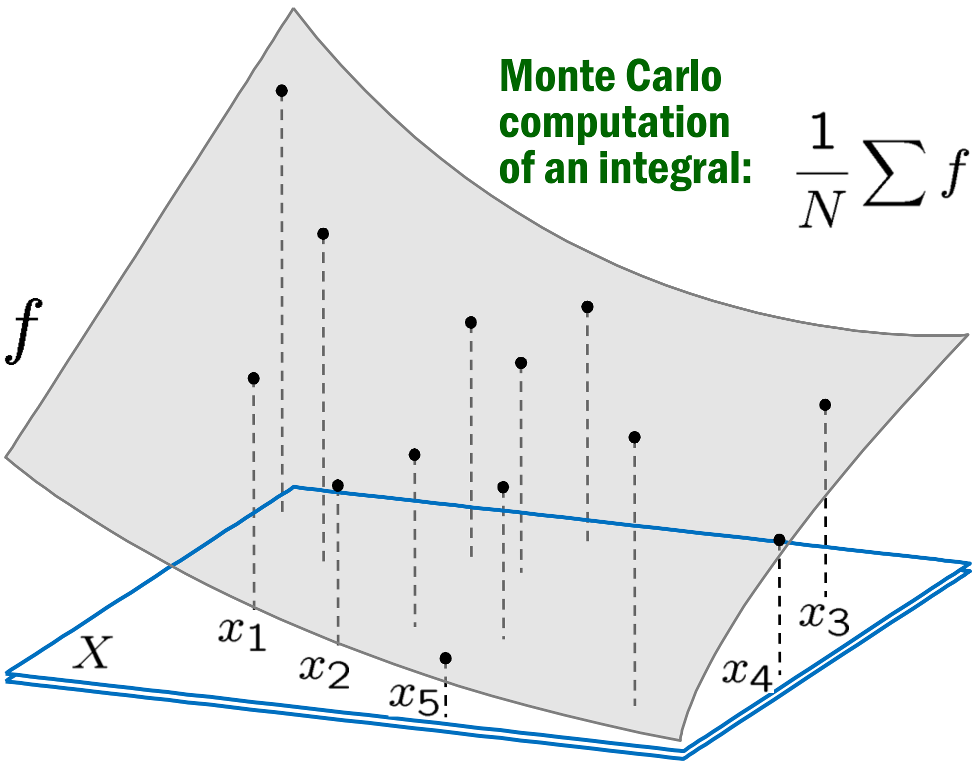
$$\left|\int f - \frac{1}{N}\sum f(x_i)\right| \leq 0.05$$

$$\rightarrow \quad N = \frac{\ln[2/(1-p)]}{2(0.05)^2}$$

(Hoeffding's)

comments

- P exists in the algorithm
- often the chance of failure can be made very small (concentration inequalities)



Monte Carlo computation of an integral: $\frac{1}{N}\sum f(x_i)$

$$\left|\int f - \frac{1}{N}\sum f(x_i)\right| \leq 0.05$$

$$\rightarrow \quad N = \frac{\ln[2/(1-p)]}{2(0.05)^2}$$

(Hoeffding's)

$$p = 1 - 10^{-7} \quad \rightarrow \quad N = 3362$$

comments

- *P* exists in the algorithm
- often the chance of failure can be made very small (concentration inequalities)

comments

- P exists in the algorithm
- often the chance of failure can be made very small (concentration inequalities)
- amazingly powerful results from probability theory can be used to assess the probability of success

where are we?

where are we?

- Monte Carlo method (1949)
- computer science
  - sorting
  - counting
  - incremental geometric constructions
  - etc.
- optimization
  - simulating annealing
  - genetic methods
  - large-scale convex optimization
  - etc.

→ little in systems and control

where are we in systems and control?

- uncertain systems
  - aerospace
  - adaptive control
  - network control
  - etc.
- many contributors:

  T. Alamo, E.W. Bai, B.R. Barmish, T. Basar, G. Calafiore, A. Chaouki, F. Dabbene, B. De Shutter, L. El Ghaoui, M. Fu, Y. Fujisaki, S. Garatti, H. Ishii, S. Kanev, H. Kimura, C. Lagoa, S.P. Meyn, Y. Oishi, B. Polyak, M. Prandini, P. Shcherbakov, J. Spall, R.F. Stengel, M. Sznaier, V.B. Tadic, R. Tempo, B. Van Roy, M. Verhagen, M. Vidyasagar, K. Zhou

- R. Tempo, G. Calafiore, and F. Dabbene (2005). “Randomized algorithms for analysis and control of uncertain systems”. Springer-Verlag.

uncertain systems: design

robust min-max design (convex)

uncertain systems: design

robust min-max design (convex)

(non-convex $\longrightarrow$ Alamo, Camacho, Tempo)

a successful story of randomization: robust min-max convex design

a successful story of randomization: robust min-max convex design

$$\min_{\theta} \left[\max_{\delta \in \Delta} \ell(\theta, \delta) \right]$$

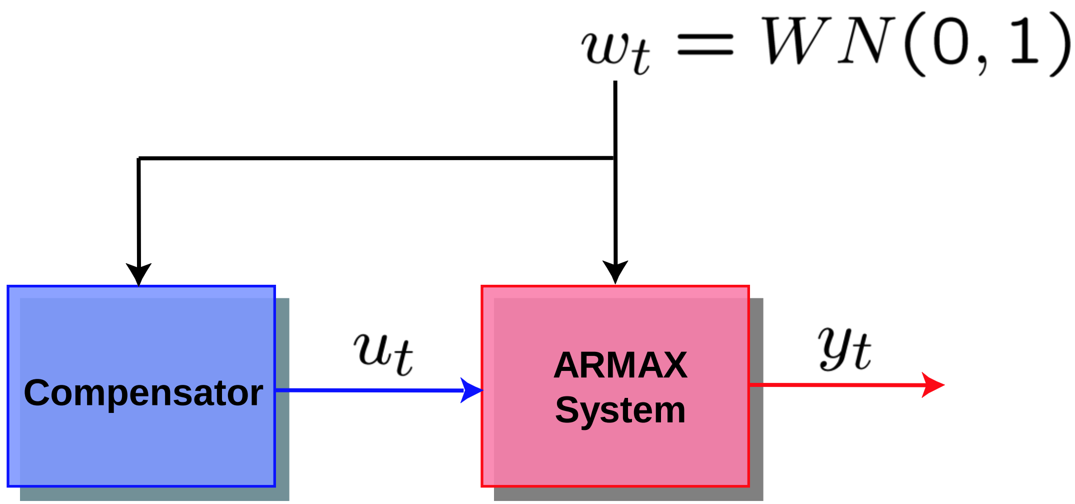
a successful story of randomization: robust min-max convex design

$$\min_{\theta} \left[\max_{\delta \in \Delta} \ell(\theta, \delta) \right]$$

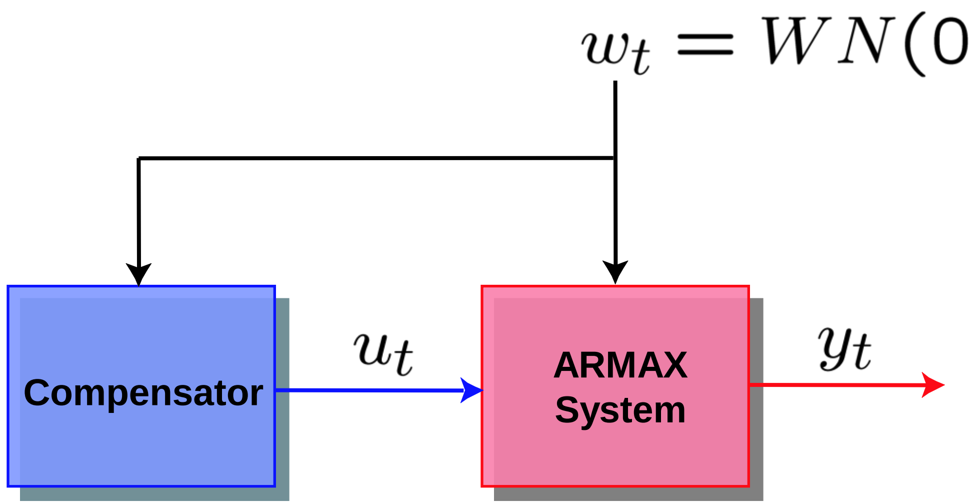
robust min-max design is hard!

Example: feedforward noise compensation

Example: feedforward noise compensation

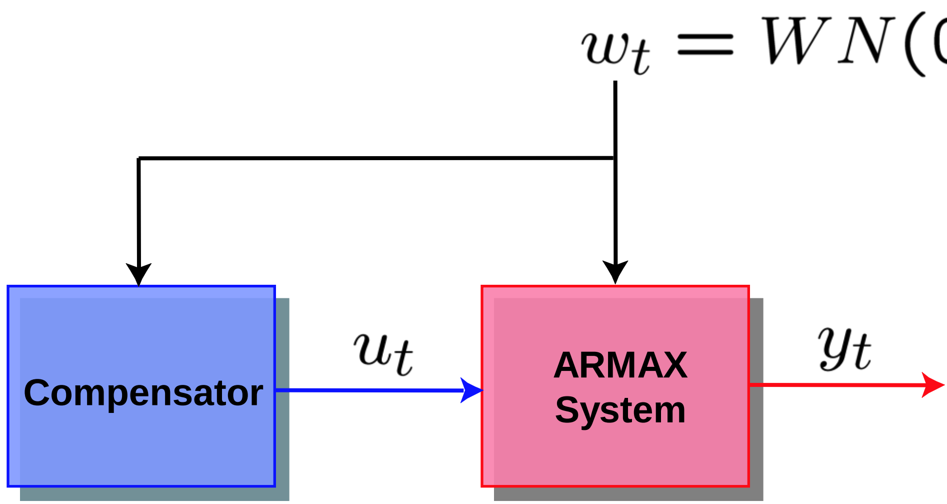


Example: feedforward noise compensation



Objective: reduce the effect of noise on y

Example: feedforward noise compensation



ARMAX System:

$$y_{t+1} = ay_t + bu_t + cw_t + dw_{t-1}$$

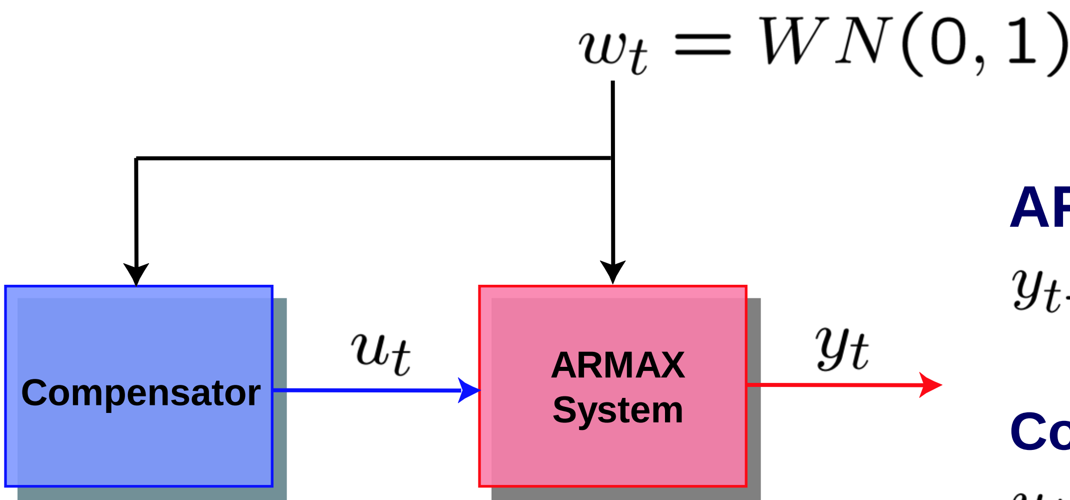
Compensator:

$$u_t = k_1 w_t + k_2 w_{t-1}$$

Goal:

$$\min var[y_t]$$

Example: feedforward noise compensation



ARMAX System:

$$y_{t+1} = ay_t + bu_t + cw_t + dw_{t-1}$$

Compensator:

$$u_t = k_1 w_t + k_2 w_{t-1}$$

Goal:

$$\min var[y_t]$$

$$var[y_t] = \frac{(c + bk_1)^2 + (d + bk_2)^2 + 2a(c + bk_1)(d + bk_2)}{1 - a^2}$$

Example: feedforward noise compensation

system parameters unknown: $a, b, c, d \in \Delta$

Example: feedforward noise compensation

system parameters unknown: $a, b, c, d \in \Delta$

robust min-max design:

$$\min_{k_1,k_2} \left[\max_{a,b,c,d \in \Delta} \frac{(c + bk_1)^2 + (d + bk_2)^2 + 2a(c + bk_1)(d + bk_2)}{1 - a^2} \right]$$

Example: feedforward noise compensation

system parameters unknown: $a, b, c, d \in \Delta$

robust min-max design:

$$\min_{k_1, k_2} \left[\max_{a,b,c,d \in \Delta} \frac{(c + bk_1)^2 + (d + bk_2)^2 + 2a(c + bk_1)(d + bk_2)}{1 - a^2} \right]$$

even a problem as simple as this is difficult for a generic Δ

other problems in robust control

- state-feedback stabilization
- H_∞ control
- H_2 control
- LPV control

$\vdots$

other problems in robust control

- state-feedback stabilization
- H_∞ control
- H_2 control
- LPV control

  ⋮

… and systems theory

- model reduction
- prediction

  ⋮

- robust min-max design is hard!

- robust min-max design is hard!
- what can we do?

- robust min-max design is hard!

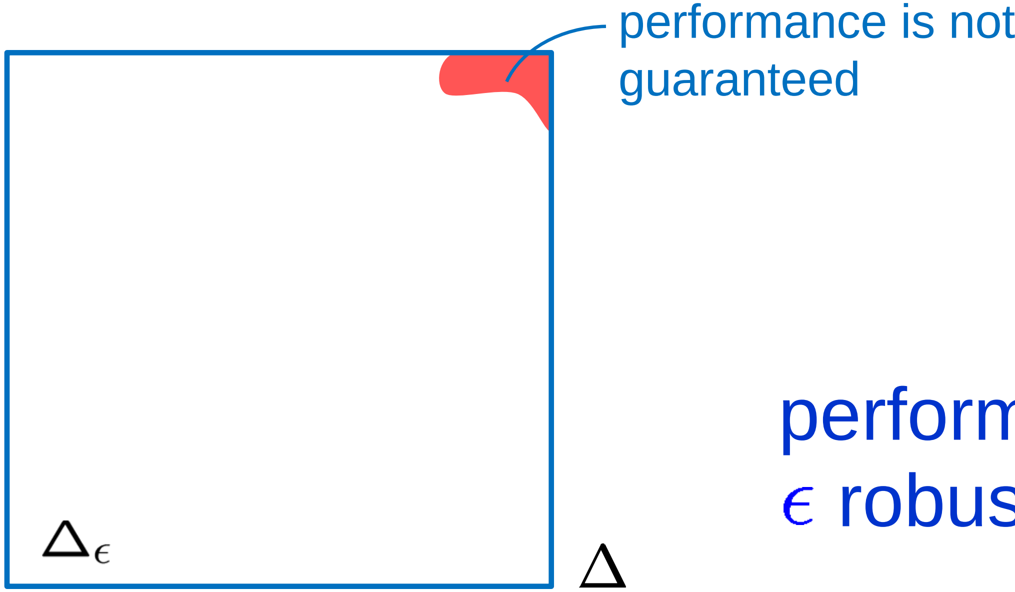
- what can we do?

- we need to accept a compromise

chance-constrained optimization

$$\min_{\theta} \left[\max_{\delta \in \Delta_\epsilon} \ell(\theta, \delta) \right]$$

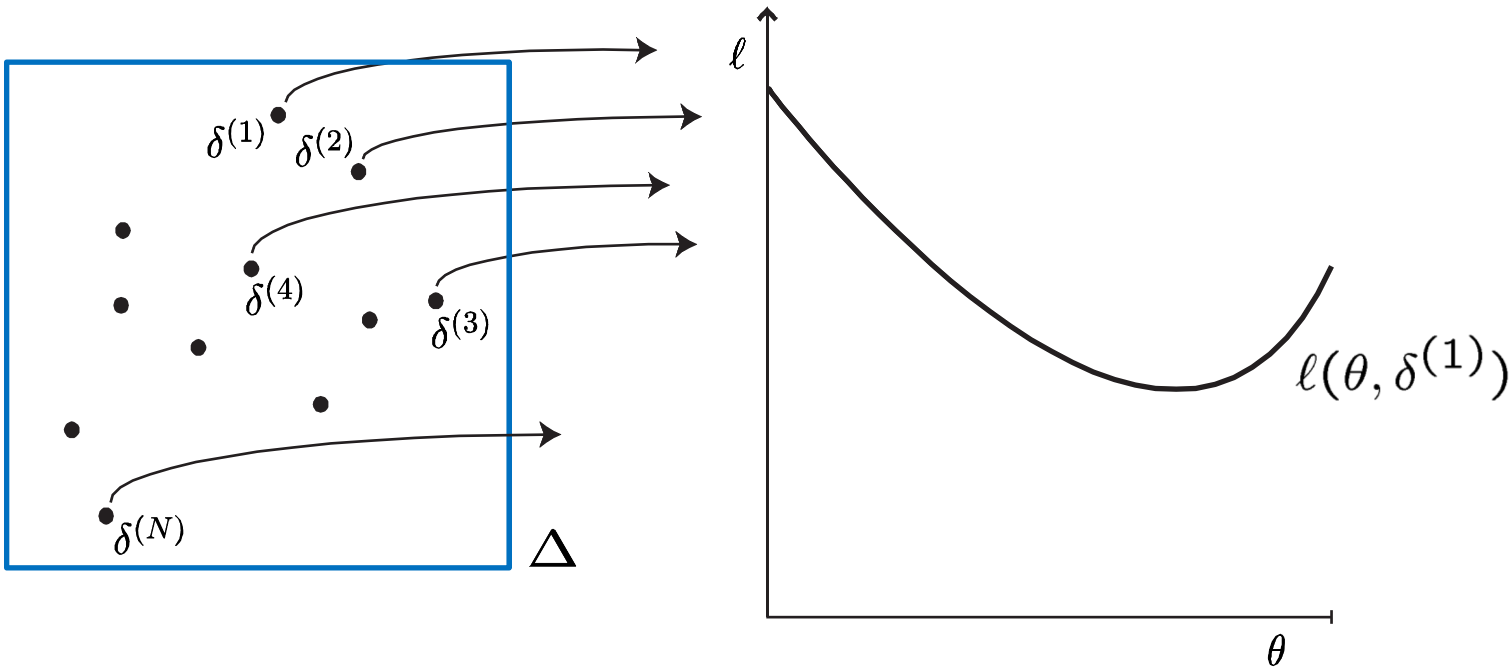
$$\frac{Vol(\Delta_\epsilon)}{Vol(\Delta)} \geq 1 - \epsilon$$



performance ℓ^* is ϵ robust

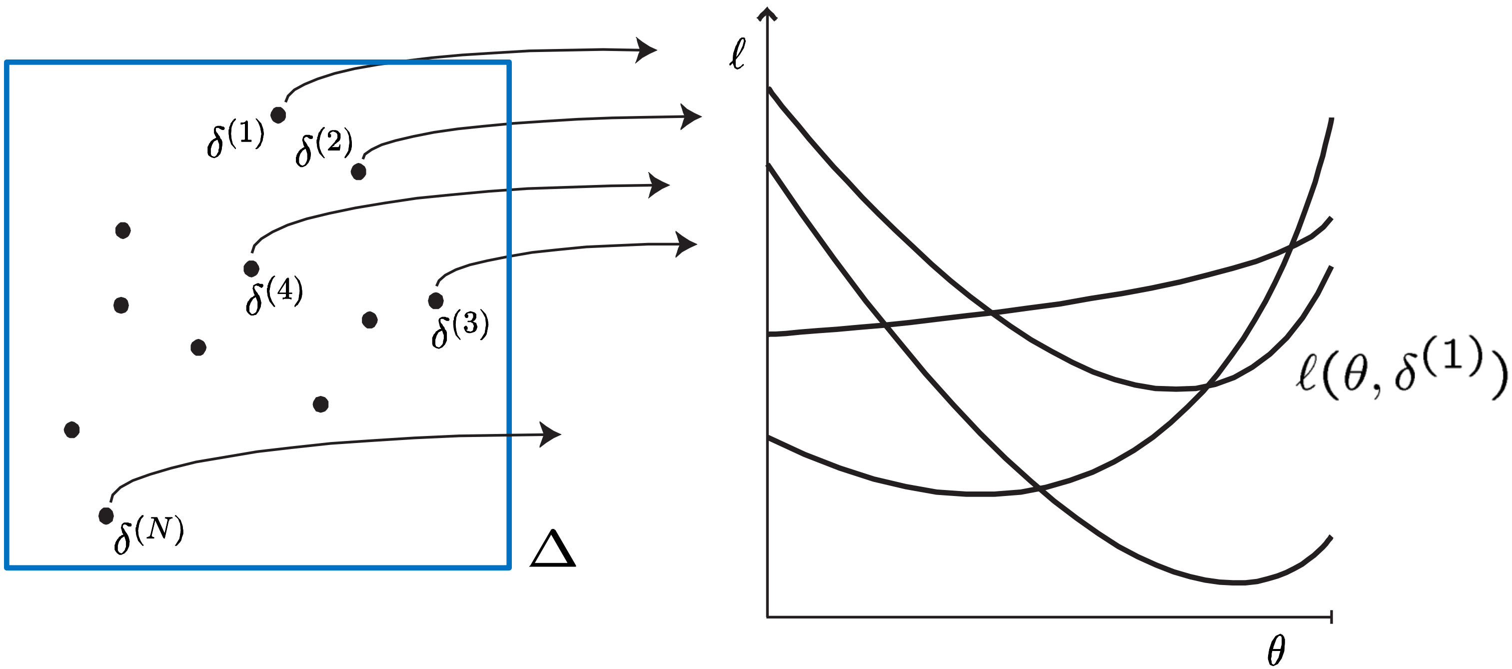
The “scenario” paradigm

(work done with G. Calafiore, S. Garatti)



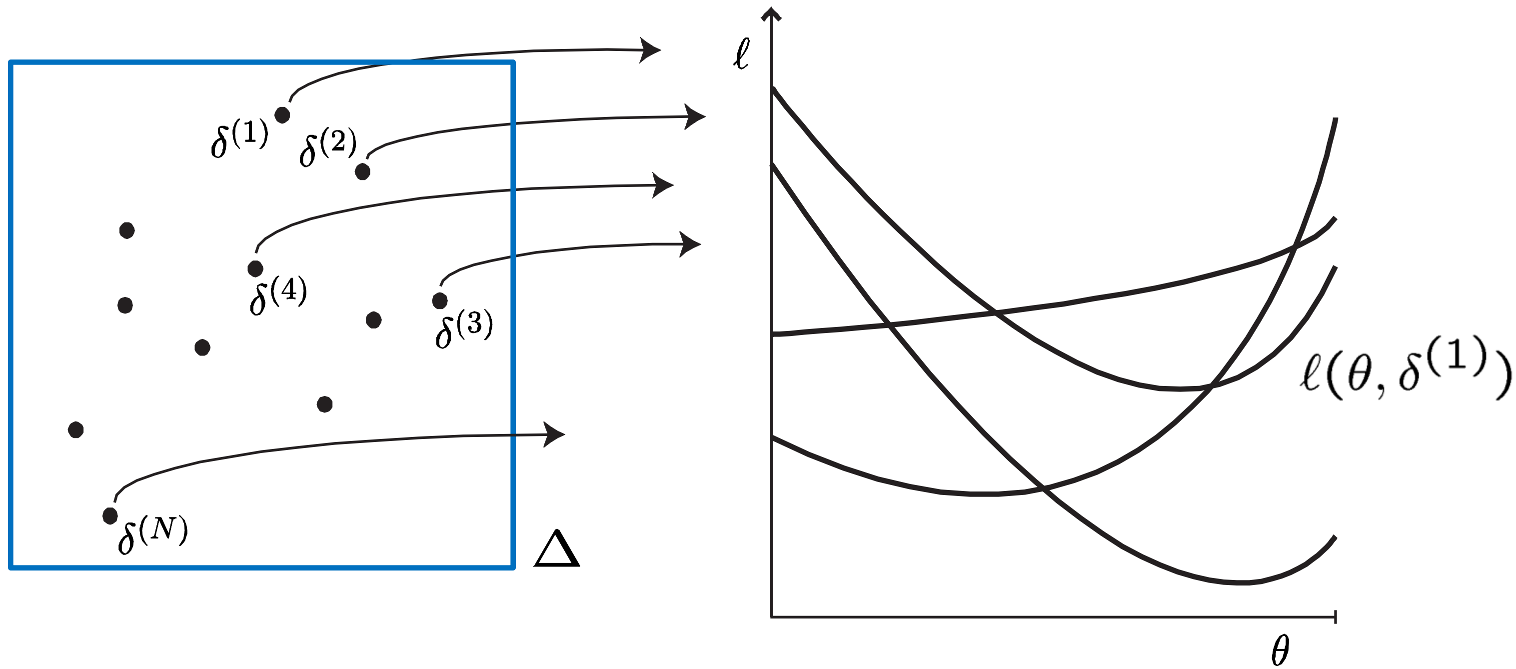
The "scenario" paradigm

(work done with G. Calafiore, S. Garatti)



The "scenario" paradigm

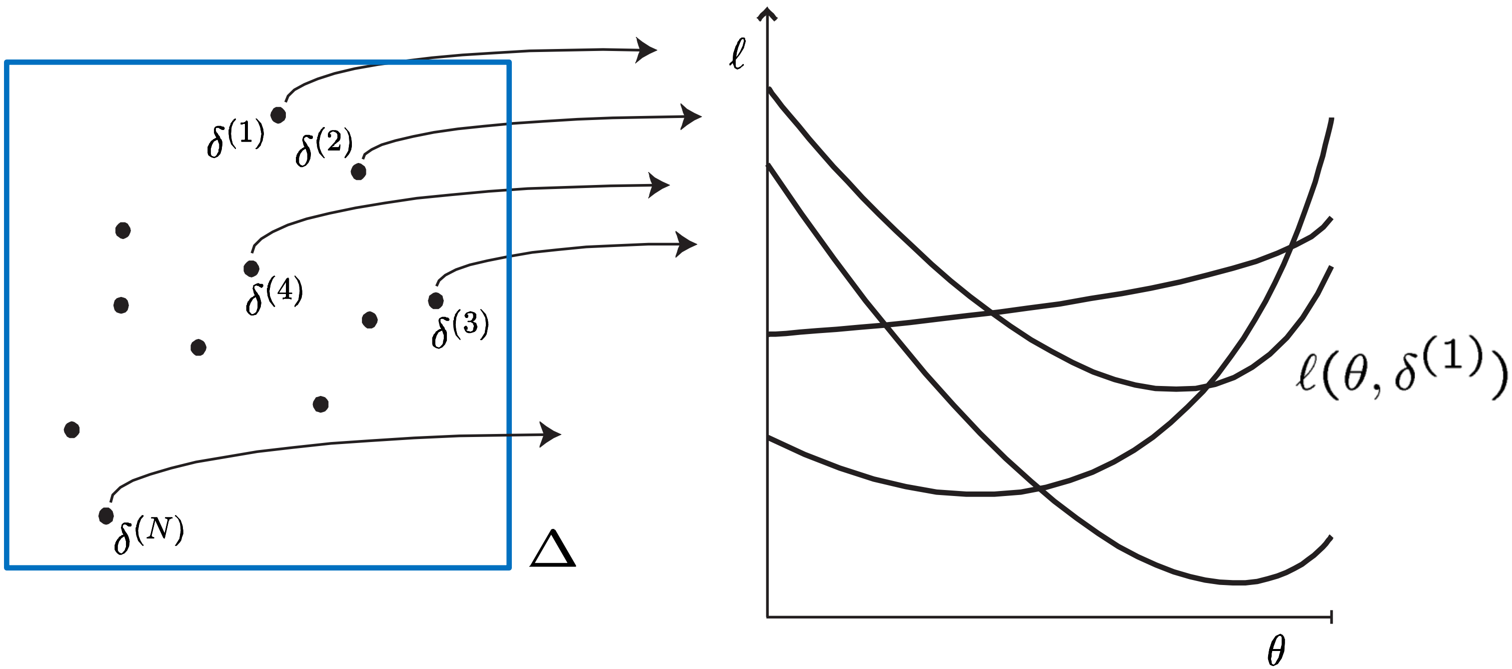
(work done with G. Calafiore, S. Garatti)



$$\min_{\theta} \left[\max_{i} \ell(\theta, \delta^{(i)}) \right]$$

The "scenario" paradigm

(work done with G. Calafiore, S. Garatti)

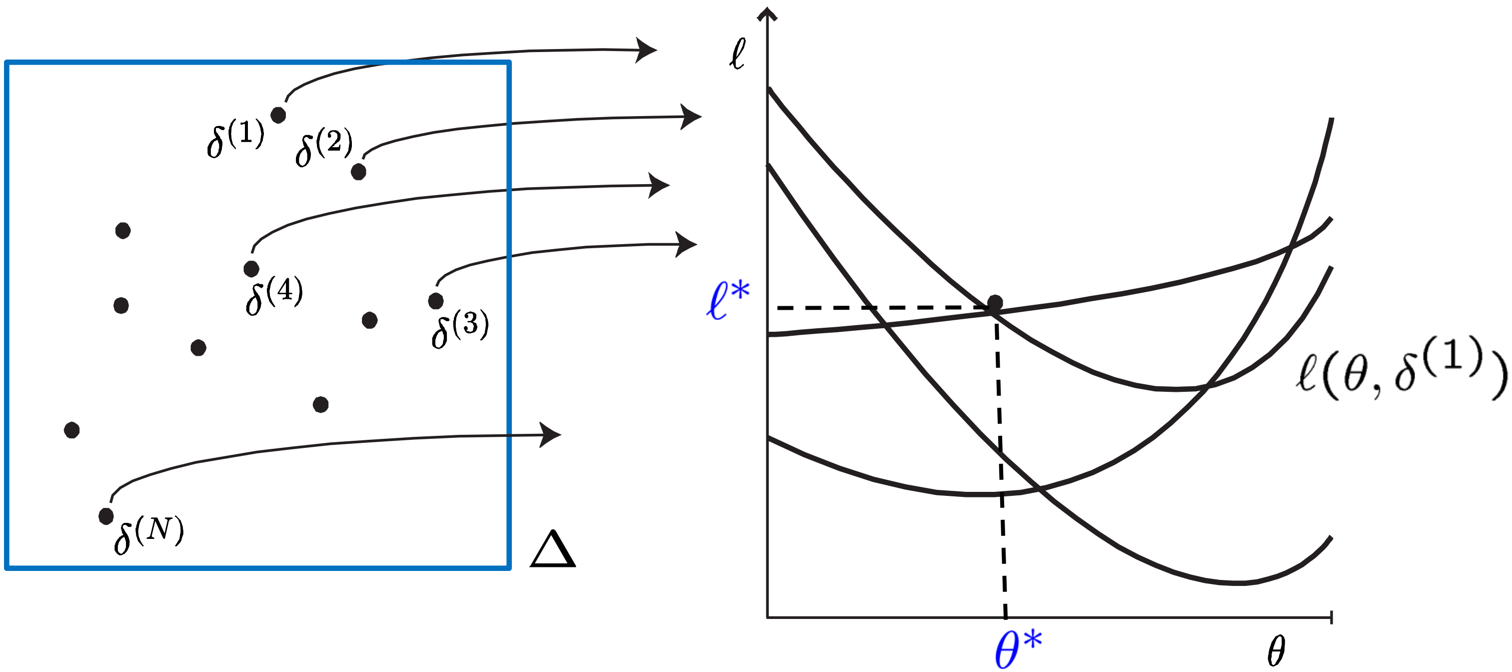


SP$_N$ = scenario program

$$\min_\theta \left[\max_i \ell(\theta, \delta^{(i)})\right]$$

The "scenario" paradigm

(work done with G. Calafiore, S. Garatti)

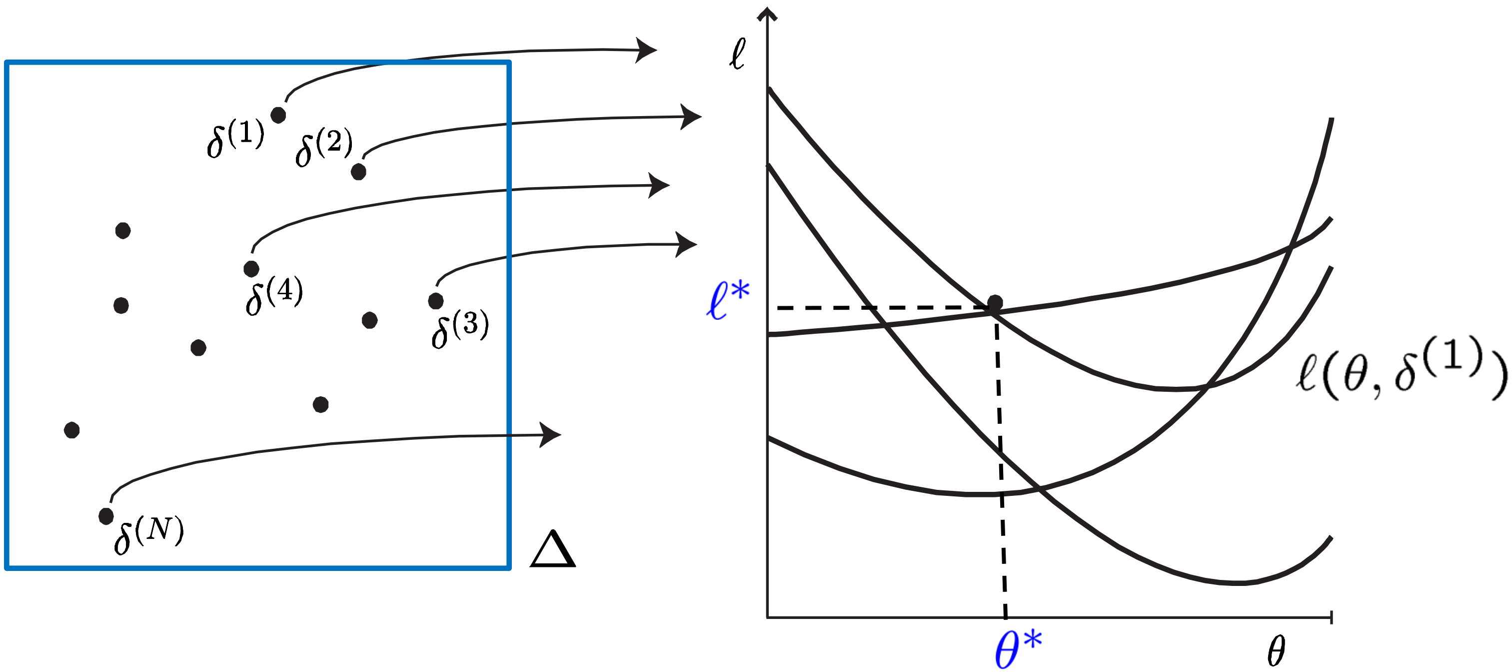


SP$_N$ = scenario program $$\min_{\theta} \left[\max_{i} \ell(\theta, \delta^{(i)}) \right]$$

- SP$_N$ is a standard finite convex optimization problem

The "scenario" paradigm

(work done with G. Calafiore, S. Garatti)



SPN = scenario program $\min_{\theta} \left[\max_{i} \ell(\theta, \delta^{(i)}) \right]$

- SPN is a standard finite convex optimization problem
- ℓ^* is superoptimal

Fundamental question:

How robust is ℓ^*?

Fundamental question:

How robust is ℓ^*?

is it ϵ robust?

Fundamental question:

How robust is ℓ^*?

is it ϵ robust?

from the "visible" to the "invisible"

Theorem (with S. Garatti – G. Calafiore)

Fix $p \in (0,1)$ (probability of success)

If $N = \frac{2}{\epsilon}\left(\ln \frac{1}{1-p} + n_\theta\right)$,
then,
with probability of success p,
ℓ^* is ϵ robust.

Theorem (with S. Garatti – G. Calafiore)

Fix $p \in (0,1)$ (probability of success)

If $N = \frac{2}{\epsilon}\left(\ln\frac{1}{1-p} + n_\theta\right)$,
then,
with probability of success p,
ℓ^* is ϵ robust.

$$p = 1 - 10^{-7} \quad \rightarrow \quad \ln\frac{1}{1-p} \sim 16$$

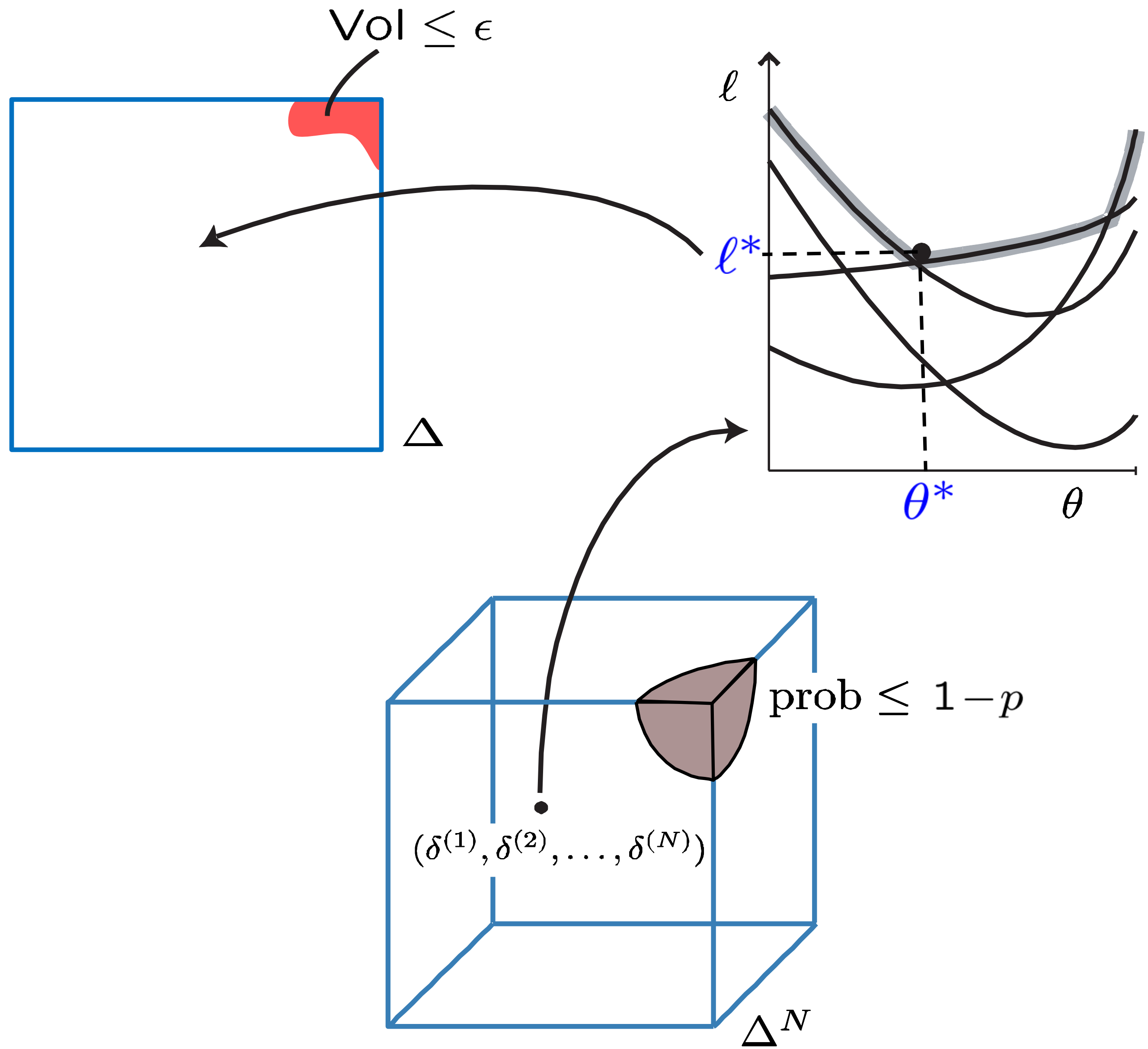
Theorem (with S. Garatti – G. Calafiore)

Fix $p \in (0,1)$ (probability of success)

If $N = \frac{2}{\epsilon}\left(\ln \frac{1}{1-p} + n_\theta\right)$,
then,
with probability of success p,
ℓ^* is ϵ robust.

$$p = 1 - 10^{-7} \quad \rightarrow \quad \ln \frac{1}{1-p} \sim 16$$

applicable to all convex problems!



Vol ≤ ϵ
Δ
ℓ
ℓ*
θ*
θ
prob ≤ 1−p
(δ(1), δ(2), . . . , δ(N))
ΔN

Example: feedforward noise compensation

$$var[y_t] = \frac{(c+bk_1)^2+(d+bk_2)^2+2a(c+bk_1)(d+bk_2)}{1-a^2}$$

Example: feedforward noise compensation

$$var[y_t] = \frac{(c+bk_1)^2+(d+bk_2)^2+2a(c+bk_1)(d+bk_2)}{1-a^2}$$

$$\Delta = \{a, b, c, d \ : \ \begin{aligned} a &= 0.45 + 0.5 * (1 - \mathrm{e}^{-8\cdot 10^3(\sigma_1^2+\sigma_2^2)}), \\ b &= 1 + \sigma_2^2, \\ c &= 0.2 + (\sigma_2 + \sin(\sigma_2) + 0.1) \cdot \sin(2\pi\sigma_2), \\ d &= 0.5 + \sigma_1^2 \cos(\sigma_2), \\ \\ & (\sigma_1, \sigma_2) \in [-1/3, 1/3]^2\}. \end{aligned}$$

Example: feedforward noise compensation

$$\varepsilon = 0.01 \quad p = 1 - 10^{-7} \quad \Rightarrow \quad N = 2158$$

sample: $a_i, b_i, c_i, d_i \in \Delta, \quad i = 1, 2, \ldots, 2158;$

solve:

$$\min_{k_1,k_2} \left[\max_i \frac{(c_i + b_i k_1)^2 + (d_i + b_i k_2)^2 + 2a_i(c_i + b_i k_1)(d_i + b_i k_2)}{1 - a_i^2} \right]$$

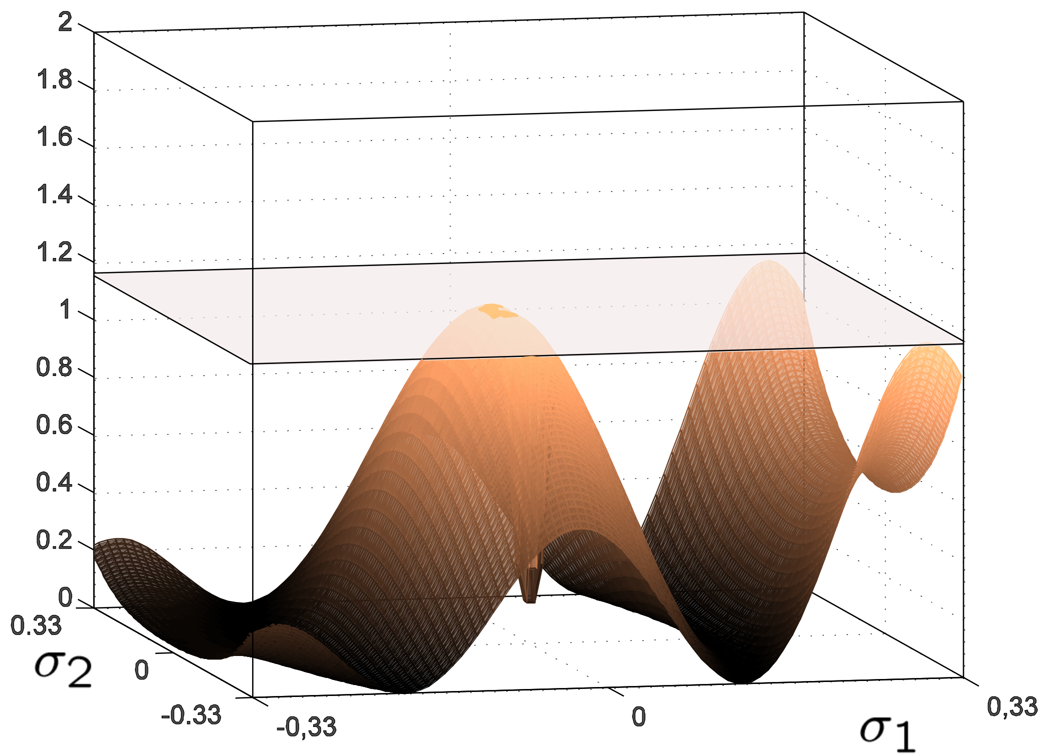
$$\Rightarrow \quad k_1^* = -0.50, \quad k_2^* = -0.53, \quad \ell^* = 1.16$$

Example: feedforward noise compensation

$\ell^* = 1.16$ ⇨ **Output variance below 1.16 for all plants but a small fraction (1%)**

Example: feedforward noise compensation

$\ell^* = 1.16$ ⇨ **Output variance below 1.16 for all plants but a small fraction (1%)**



once more, a tough problem has turned into a solvable one through randomization, … provided we accept an ϵ risk

… in conclusion

… in conclusion

- randomization changes our perspective of problem solvability

… in conclusion

- randomization changes our perspective of problem solvability

- the probability of success depends on an artificial *P* and can be assessed with extraordinarily powerful probabilistic tools

… in conclusion

- randomization changes our perspective of problem solvability

- the probability of success depends on an artificial *P* and can be assessed with extraordinarily powerful probabilistic tools

- it’s just a paradigm;
each single problem has to be studied separately

… in conclusion

- randomization changes our perspective of problem solvability

- the probability of success depends on an artificial *P* and can be assessed with extraordinarily powerful probabilistic tools

- it's just a paradigm; each single problem has to be studied separately

- can prove useful in many more problems in systems and control, especially at the boundary of control, communication and computation

THANK YOU

THANKS TO:

T. Alamo, S. Bittanti, G. Calafiore, F. Dabbene, Y. Fujisaki, S. Garatti, M. Gevers, C. Lagoa, Y. Oishi, S. Mitter, A. Nemirovski, B. Polyak, M. Prandini, M. Sznaier, R. Tempo, M. Vidyasagar, J. Willems

REFERENCES

M.C. Campi.
Why is resorting to fate wise? A critical look at randomized algorithms in systems and control
European J. of Control, 16, no.5: 419-430, 2010.

M.C. Campi and E. Weyer.
Non-asymptotic confidence sets for the parameters of linear transfer functions.
IEEE Trans. on Automatic Control, December 2010.

M.C. Campi and S. Garatti.
The Exact Feasibility of Randomized Solutions of Uncertain Convex Programs.
SIAM J. on Optimization, 19, no.3: 1211-1230, 2008.

G. Calafiore and M.C. Campi.
Uncertain Convex Programs: randomized Solutions and Confidence Levels.
Mathematical Programming, 102: 25-46, 2005.

G. Calafiore and M.C. Campi.
The Scenario Approach to Robust Control Design.
IEEE Trans. on Automatic Control, AC-51: 742-753, 2006.