

Mixed LMI/Randomized Methods for Static Output Feedback Control Design

Denis Arzelier[†], Elena N. Gryazina[‡], Dimitri Peaucelle[†] and Boris T. Polyak[‡]



LAAS-CNRS - Université de Toulouse - FRANCE



ICS-RAS - Moscow - RUSSIA

American Control Conference - Baltimore

June 30 - July 2 2010

- Work in the framework of a joint project funded by CNRS and RFBR
- "Robust and adaptive control of complex systems"

- Confrontation of LMI and Randomized techniques
- Get some insight on an LMI-based heuristic for SOF design
- Using hit-and-run methods to explore randomly the set of solutions.

- Useful algorithm
- Extensions to robust multi-objective problems
- Design sets of controllers

- ① LMI-based heuristic for SOF design
- ② Hit-and-run method
- ③ Proposed algorithms
- ④ Examples from the Compleib library
- ⑤ Conclusions & extensions

■ Static state and output feedback

$$\dot{x} = Ax + Bu, \quad y = Cx$$

$$u = K_s x \quad | \quad u = Ky$$

● Classical LMI results describing all SF gains

$$\mathcal{K}_{SF} = \{ K_s = SX^{-1} : AX + BS + XA^T + S^T B^T < \mathbf{0}, \quad X > \mathbf{0} \}$$

● [ECC01] LMI result describing a subset of SOF gains

$$\mathcal{K}_{SOF}^{K_s} = \{ K = -F^{-1}Z : \mathcal{L}^{K_s}(P, Z, F) < \mathbf{0}, \quad P > \mathbf{0} \}$$

where $\mathcal{L}^{K_s}(P, Z, F) =$

$$\begin{bmatrix} A^T P + PA & PB \\ B^T P & \mathbf{0} \end{bmatrix} + \begin{bmatrix} K_s^T \\ -\mathbf{1} \end{bmatrix} \begin{bmatrix} ZC & F \end{bmatrix} + \begin{bmatrix} C^T Z^T \\ F^T \end{bmatrix} \begin{bmatrix} K_s & -\mathbf{1} \end{bmatrix}$$

- [ECC01] LMI result describing a subset of SOF gains

$$\mathcal{K}_{SOF}^{K_s} = \{ K = -F^{-1}Z : \mathcal{L}^{K_s}(P, Z, F) < \mathbf{0}, P > \mathbf{0} \}$$

where $\mathcal{L}^{K_s}(P, Z, F) =$

$$\begin{bmatrix} A^T P + P A & P B \\ B^T P & \mathbf{0} \end{bmatrix} + \begin{bmatrix} K_s^T \\ -\mathbf{1} \end{bmatrix} \begin{bmatrix} Z C & F \end{bmatrix} + \begin{bmatrix} C^T Z^T \\ F^T \end{bmatrix} \begin{bmatrix} K_s & -\mathbf{1} \end{bmatrix}$$

▲ $K_s \notin \mathcal{K}_{SF} \Rightarrow \mathcal{K}_{SOF}^{K_s} = \emptyset$

$$\begin{bmatrix} \mathbf{1} & K_s^T \end{bmatrix} \mathcal{L}^{K_s}(P, Z, F) \begin{bmatrix} \mathbf{1} \\ K_s \end{bmatrix} = (A + B K_s)^T P + P (A + B K_s) < \mathbf{0}$$

▲ But $K_s \in \mathcal{K}_{SF} \not\Rightarrow \mathcal{K}_{SOF}^{K_s} \neq \emptyset$

(P has to prove stability of $A + B K_s$ and $A + B K C$ simultaneously)

1 LMI-based heuristic for SOF design

- [ECC01] LMI result describing a subset of SOF gains

$$\mathcal{K}_{SOF}^{K_s} = \{ K = -F^{-1}Z : \mathcal{L}^{K_s}(P, Z, F) < \mathbf{0}, P > \mathbf{0} \}$$

where $\mathcal{L}^{K_s}(P, Z, F) =$

$$\begin{bmatrix} A^T P + P A & P B \\ B^T P & \mathbf{0} \end{bmatrix} + \begin{bmatrix} K_s^T \\ -\mathbf{1} \end{bmatrix} \begin{bmatrix} Z C & F \end{bmatrix} + \begin{bmatrix} C^T Z^T \\ F^T \end{bmatrix} \begin{bmatrix} K_s & -\mathbf{1} \end{bmatrix}$$

▲ $K_s \notin \mathcal{K}_{SF} \Rightarrow \mathcal{K}_{SOF}^{K_s} = \emptyset$ but $K_s \in \mathcal{K}_{SF} \not\Rightarrow \mathcal{K}_{SOF}^{K_s} \neq \emptyset$

- All SOF gains are represented: $\bigcup_{K_s \in \mathcal{K}_{SF}} \mathcal{K}_{SOF}^{K_s} = \mathcal{K}_{SOF}$

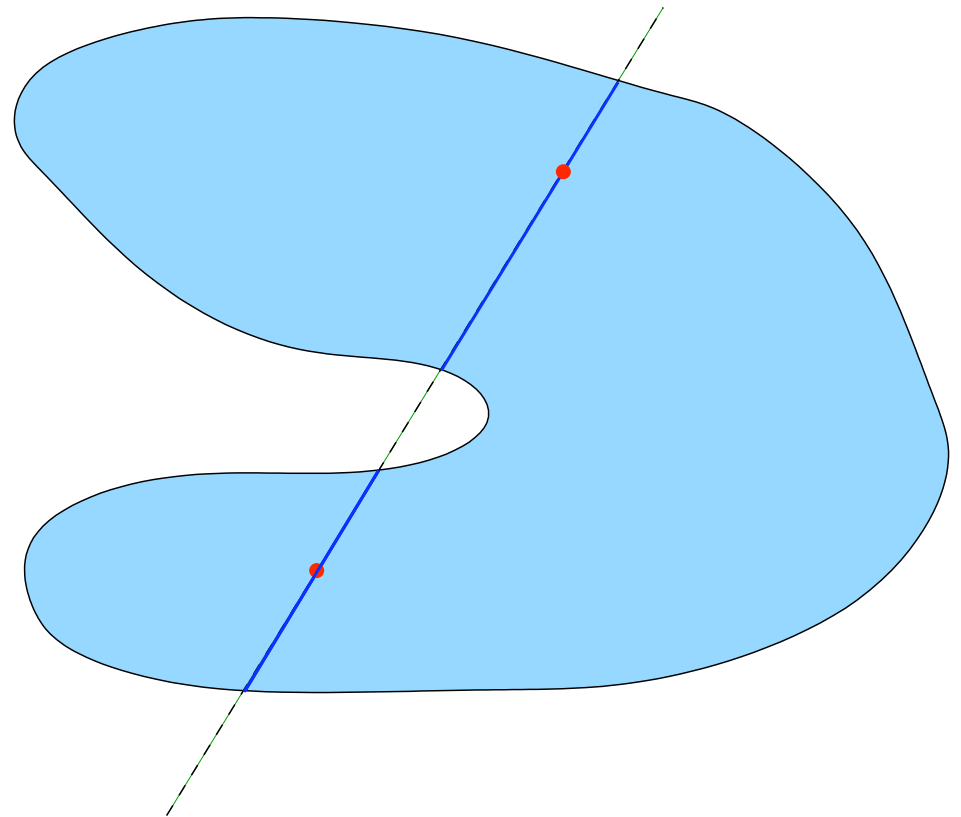
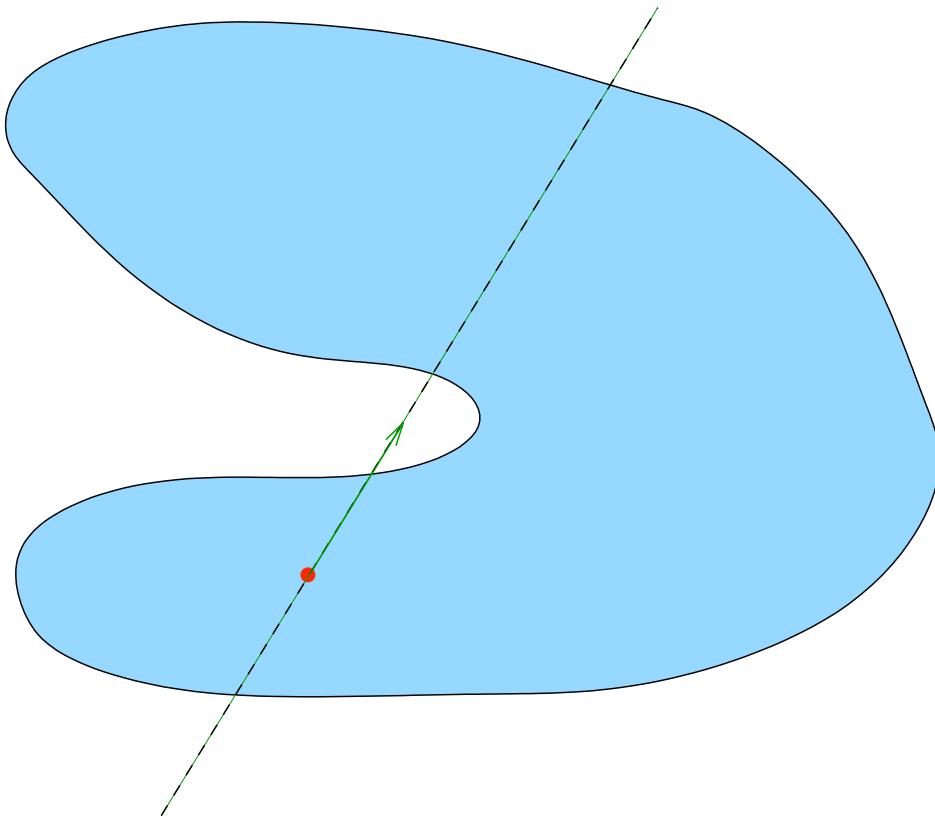
$$(K \in \mathcal{K}_{SOF} \Rightarrow KC \in \mathcal{K}_{SF} \text{ and } K \in \mathcal{K}_{SOF}^{KC})$$

■ Aim: generate finite set $\{K_{s,i=1\dots N}\} \subset \mathcal{K}_{SF}$

and get approximation $\bigcup_{i=1\dots N} \mathcal{K}_{SOF}^{K_{si}} \subset \mathcal{K}_{SOF}$.

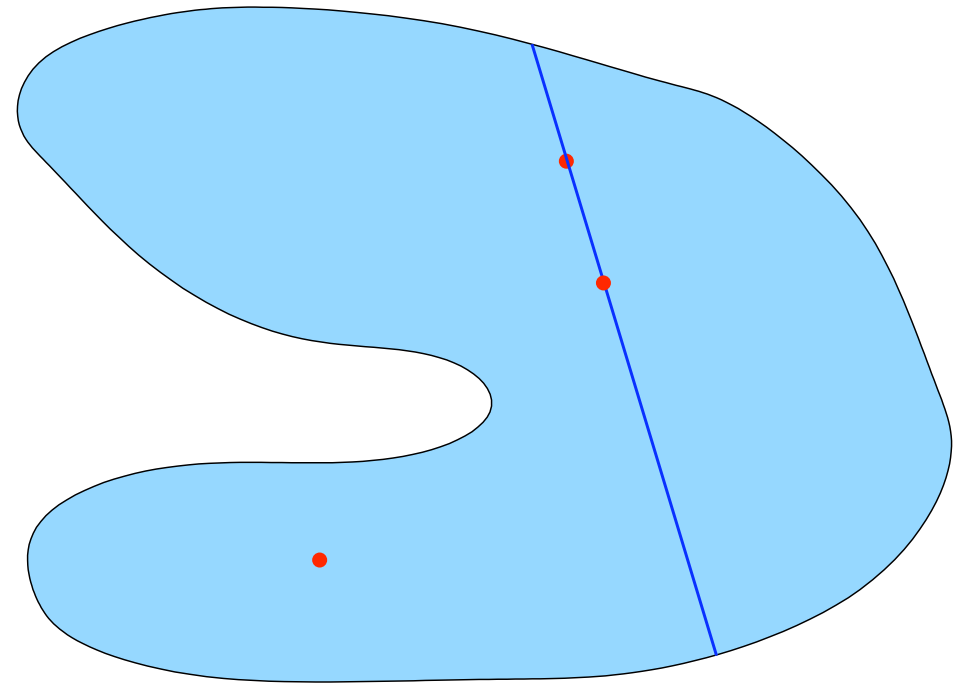
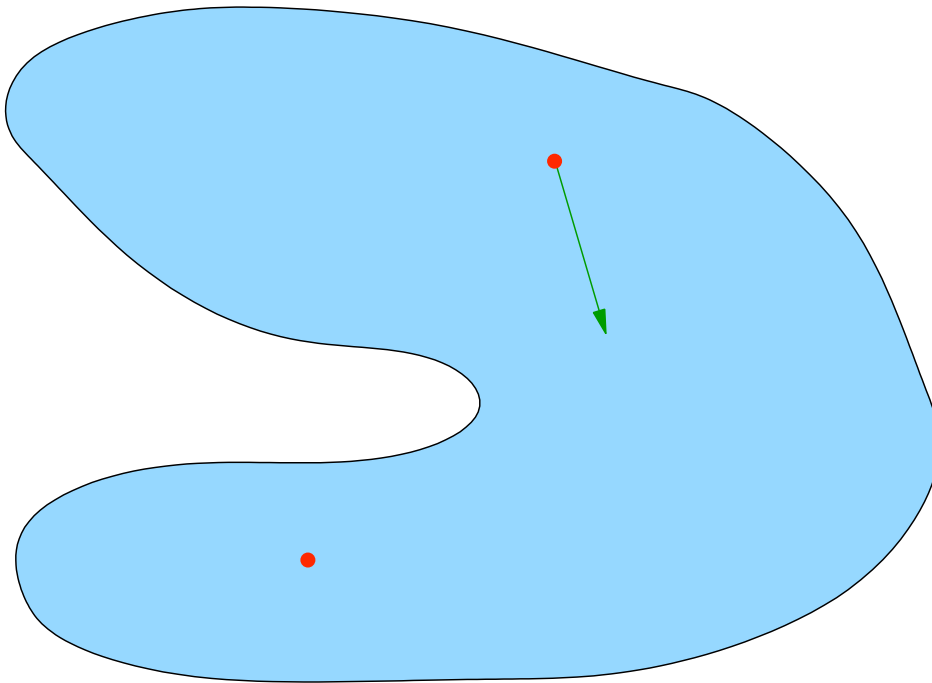
② Hit-and-run methods

- Starting from a feasible point, explore sets in random directions



② Hit-and-run methods

■ Iterate to get the needed number of points



▲ Intervals in the (random) direction obtained using a boundary oracle.

● New point is taken random in the interval.

■ Considered sets

$$\mathcal{K}_{SF} = \{K_s : A + BK_s \text{ Hurwitz}\}$$

$$\mathcal{K}_{SOF} = \{K : A + BKC \text{ Hurwitz}\}$$

- From a known gain K_i ,
- search in random direction $D = Y/\|Y\|$ ($Y = \text{randn}(m, p)$),
- find the set (for example using `fsolve`):

$$\Theta_{K_i} = \{\theta : f(\theta) = \max \text{R}\lambda(A + B(K_i + \theta D)C) < 0\}$$

- Take a random value $\theta_i \in \Theta_{K_i}$ (with `rand`) and get a new point

$$K_{i+1} = K_i + \theta_i D$$

▲ How efficient is the following procedure for SOF design ?

“Take $K_s \in \mathcal{K}_{SF}$ and solve $\mathcal{L}^{K_s} < \mathbf{0}$ ”

■ Algorithm 1:

- 1- Find one value $K_s \in \mathcal{K}_{SF}$ by solving the corresponding LMI problem.
 - 2- Using K_s as a starting point, generate $\{K_{s,i=1\dots N}\} \subset \mathcal{K}_{SF}$ using H&R
 - 3- Check if the LMIs $\mathcal{L}^{K_{s,i}} < \mathbf{0}$ is feasible.
- N_{sof} : number of cases when the LMIs are feasible at step ● 3
- N_{sof}/N : efficiency of the procedure to find and SOF gain

3 Proposed algorithms

■ Algorithm 1:

- 1- Find one value $K_s \in \mathcal{K}_{SF}$ by solving the corresponding LMI problem.
 - 2- Using K_s as a starting point, generate $\{K_{s,i=1\dots N}\} \subset \mathcal{K}_{SF}$ using H&R
 - 3- Check if the LMI $\mathcal{L}^{K_{s,i}} < \mathbf{0}$ is feasible.
- N_{sof} : number of cases when the LMIs are feasible at step ● 3
- Each solution of the LMIs at step ● 3 gives an SOF K_j .
- ▲ Is the set $\{K_{j=1\dots N_{sof}}\}$ a “good” subset of \mathcal{K}_{SF} ?

■ Algorithm 2:

- 1- Find one value $K_s \in \mathcal{K}_{SF}$ by solving the corresponding LMI problem.
 - 2- Using K_s as a starting point, generate $\{K_{s,i}\} \subset \mathcal{K}_{SF}$ using H&R
and stop as soon as one value is such that $\mathcal{L}^{K_{s,i}} < \mathbf{0}$ is feasible.
 - 3- Using K as a starting point to generate $\{K_{i=1\dots N}\} \subset \mathcal{K}_{SOF}$ using H&R
- $\{K_{j=1\dots N_{sof}}\}$ and $\{K_{i=1\dots N}\}$ are two subsets of \mathcal{K}_{SOF} for comparison.

4 Examples from the Compleib library

- Compleib `www.complib.de` examples of SOF design problems
- All 53 non open-loop stable problems were tested
- $N_{sof} = 0$ (algorithm 1 fails) only for one example (AC10, known to be hard)
- $N_{sof}/N < 5\%$ for 11 problems (bad conditioning = num pb in LMIs)

4 Examples from the Compleib library

Ex.	n_x	n_u	n_y	OLS	N_{sof}/N
AC1	5	3	3	OLMS	240/1000
AC2	5	3	3	OLMS	334/1000
AC5	4	2	2	OLNS	283/1000
AC9	10	4	5	OLNS	34/1000
AC10	55	2	2	OLNS	*
AC11	5	2	4	OLNS	996/1000
AC12	4	3	4	OLNS	997/1000
AC13	28	3	4	OLNS	39/1000
AC14	40	3	4	OLNS	13/1000
AC18	10	2	2	OLNS	17/1000

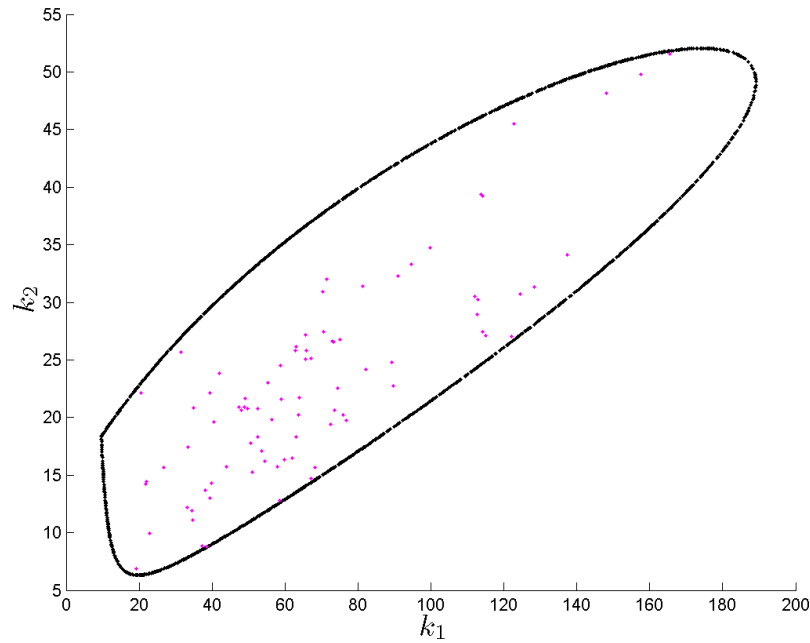
HE1	4	2	1	OLNS	93/1000
HE3	8	4	6	OLNS	12/1000
HE4	8	4	6	OLNS	1000/1000
HE5	4	2	2	OLNS	13/1000
HE6	20	4	6	OLNS	321/1000
HE7	20	4	6	OLNS	327/1000
DIS2	3	2	2	OLNS	842/1000
DIS4	6	4	6	OLNS	1000/1000
DIS5	4	2	2	OLNS	725/1000
JE2	21	3	3	OLMS	94/1000
JE3	24	3	6	OLMS	44/1000
REA1	4	2	3	OLNS	999/1000
REA2	4	2	2	OLNS	554/1000
REA3	12	1	3	OLNS	965/1000

4 Examples from the Compleib library

Ex.	n_x	n_u	n_y	OLS	N_{sof}/N
WEC1	10	3	4	OLNS	775/1000
BDT2	82	4	4	OLMS	43/1000
IH	21	11	10	OLMS	63/1000
CSE2	60	2	30	OLNS	10/10
PAS	5	1	3	OLMS	236/1000
TF1	7	2	4	OLMS	81/1000
TF2	7	2	3	OLMS	189/1000
TF3	7	2	3	OLMS	6/1000
NN1	3	1	2	OLNS	629/1000
NN2	2	1	1	OLMS	1000/1000
NN5	7	1	2	OLNS	84/1000
NN6	9	1	4	OLNS	983/1000
NN7	9	1	4	OLNS	620/1000
NN9	5	3	2	OLNS	7/1000

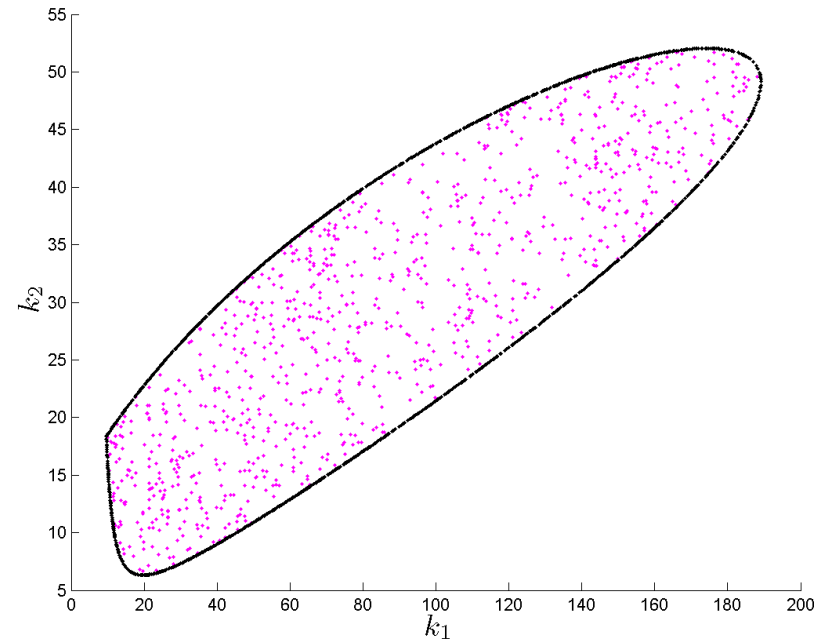
NN12	6	2	2	OLNS	28/1000
NN13	6	2	2	OLNS	77/1000
NN14	6	2	2	OLNS	44/1000
NN15	3	2	2	OLMS	821/1000
NN16	8	4	4	OLMS	61/1000
NN17	3	2	1	OLNS	125/1000
HF2D10	5	2	3	OLNS	991/1000
HF2D11	5	2	3	OLNS	993/1000
HF2D14	5	2	4	OLNS	1000/1000
HF2D15	5	2	4	OLNS	1000/1000
HF2D16	5	2	4	OLNS	998/1000
HF2D17	5	2	4	OLNS	1000/1000
HF2D18	5	2	2	OLNS	755/1000
TMD	6	2	4	OLNS	654/1000
FS	5	1	3	OLNS	977/1000

4 Examples from the Compleib library



NN5 SOF gains with Algo 1.

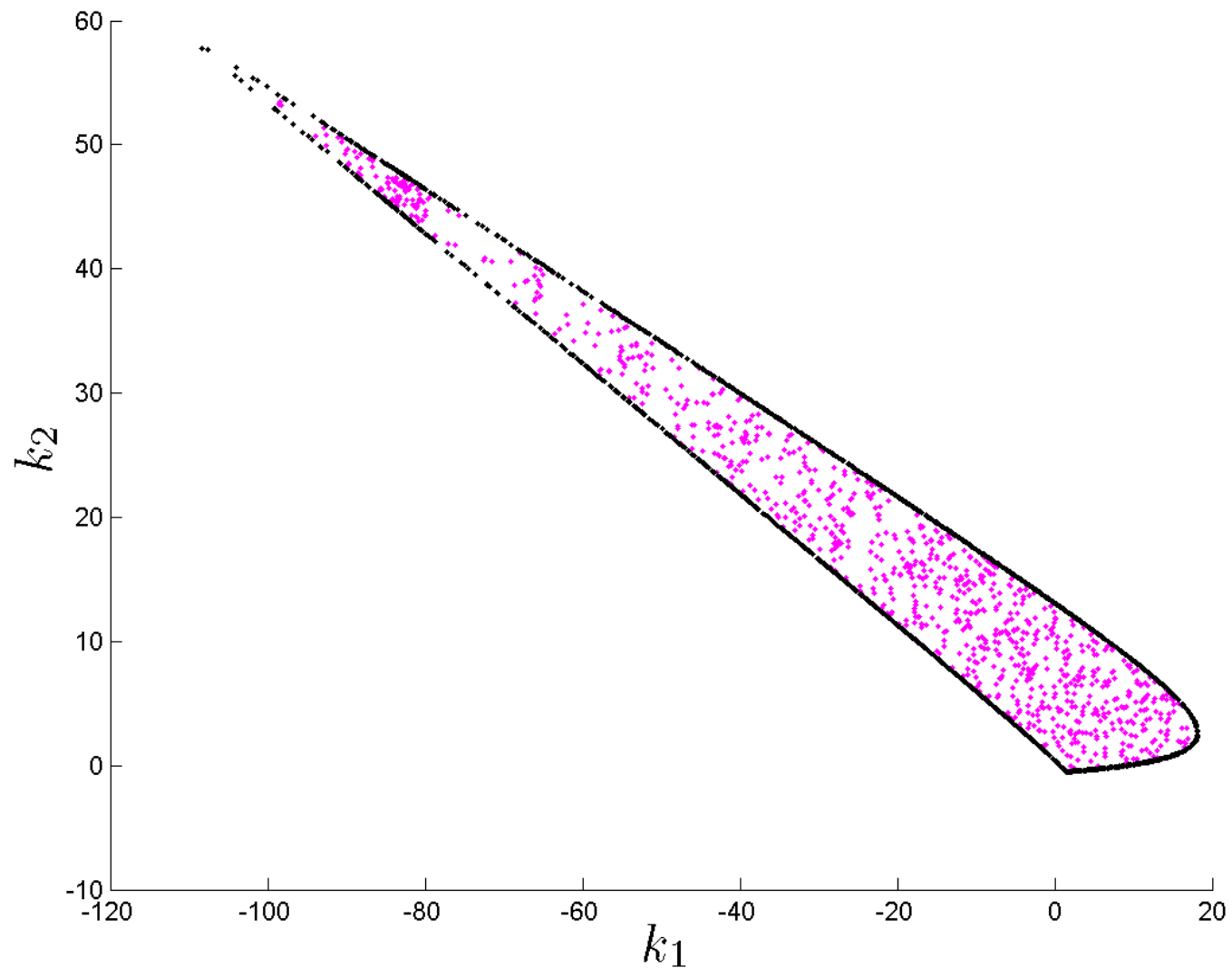
rather good distribution



NN5 SOF gains with Algo 2.

very good distribution

4 Examples from the Compleib library



AC7 SOF gains with Algo 2.

Conclusions

- Rather good properties of the studied LMI condition
- Combining LMI with H&R gives efficient algorithms for SOF design
(attested on Compleib examples)
- Algorithms allow to describe as a finite set the interior of \mathcal{K}_{SOF}
(one may choose the “best” one by inspection)
- LMI based technique: extends to robust/multi-objective problems easily
(results to be submitted soon)