Gradient-based and derivative-free multiobjective optimization via iterative single-objective optimization: MO-BFGS and MO-SLSQP

Dimo Brockhoff
joint work with Anne Auger, Nikolaus Hansen, and Baptiste Plaquevent-Jourdain

Inria and CMAP, Ecole Polytechnique, IP Paris, France

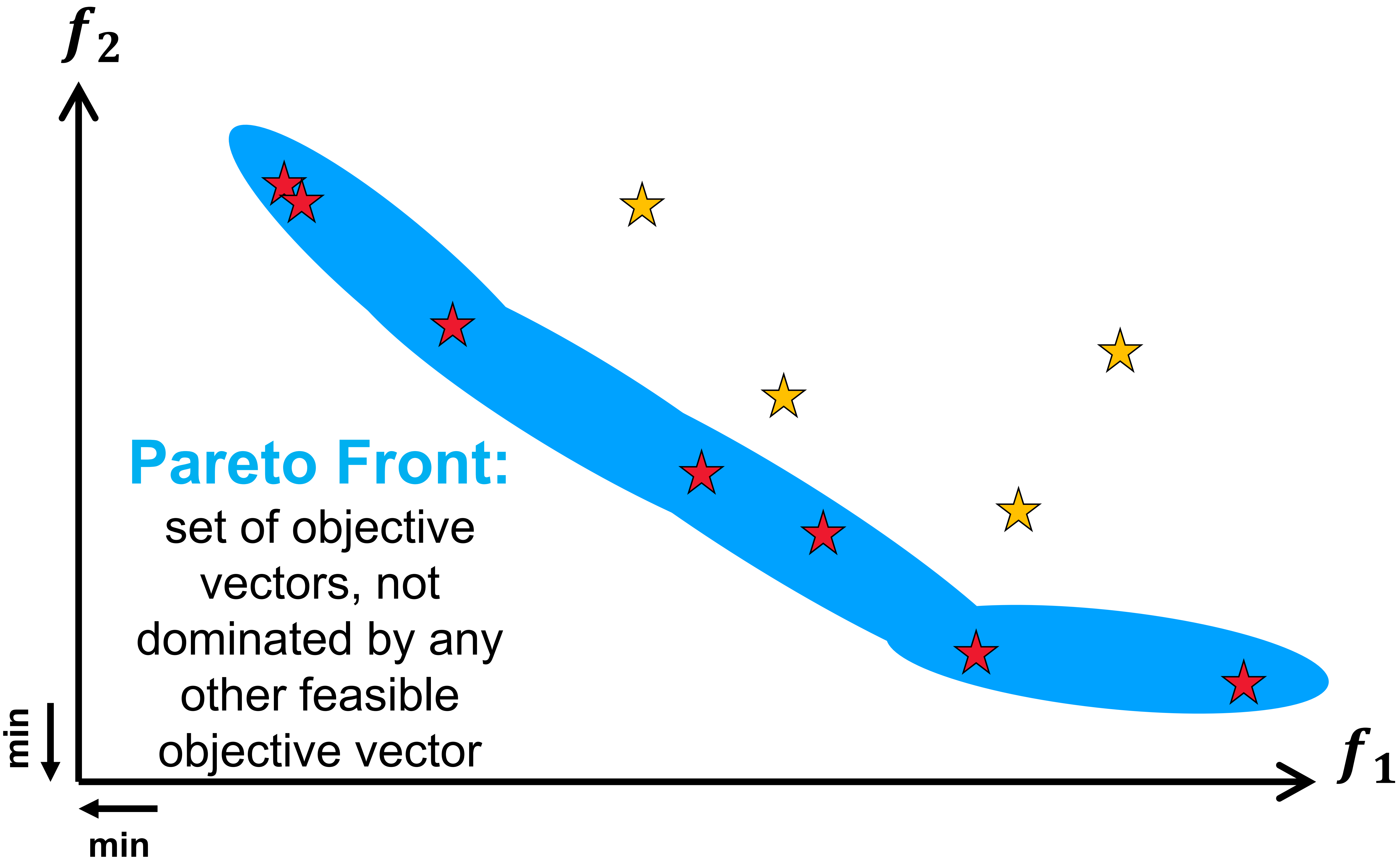
July 23, 2024 --- ISMP, Montreal, Canada

Context: Multiobjective Blackbox Optimization

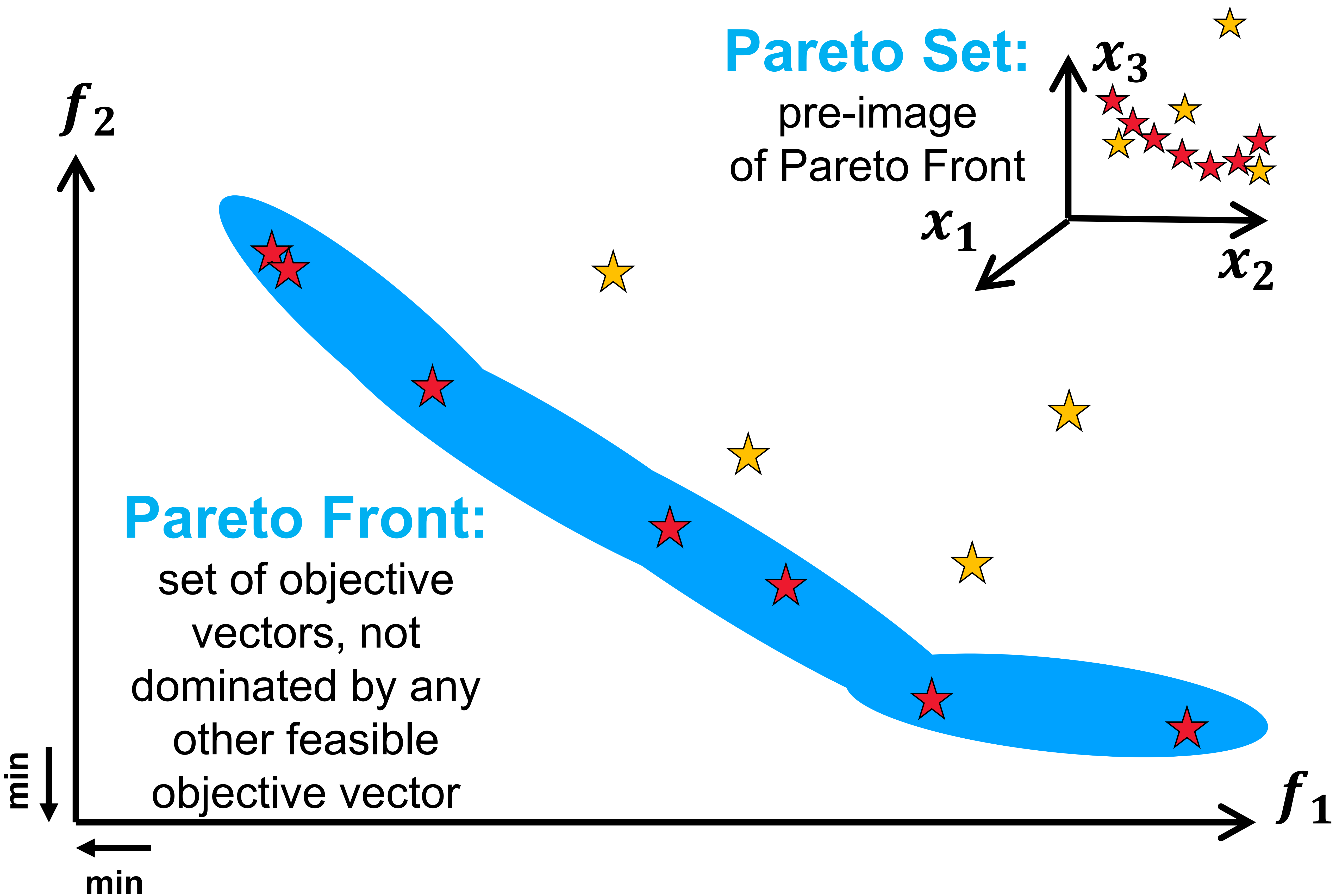
if $k > 1$:

- incomparable solutions (objective trade-offs)
- "optimum" is a set

Pareto Set and Pareto Front

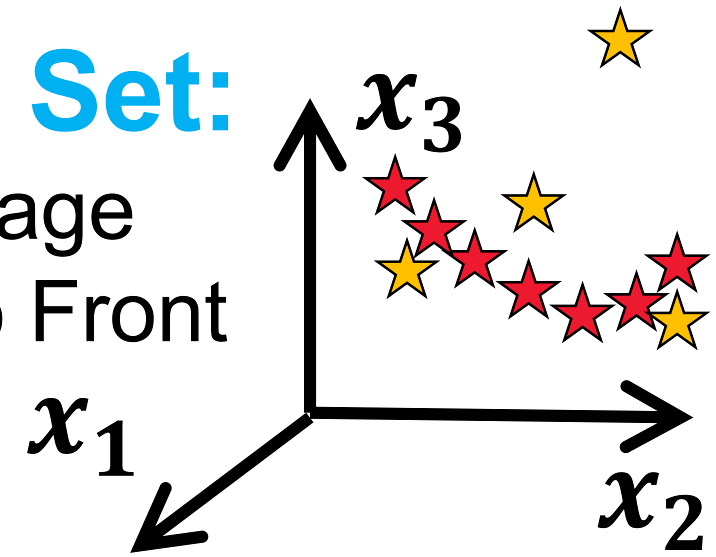


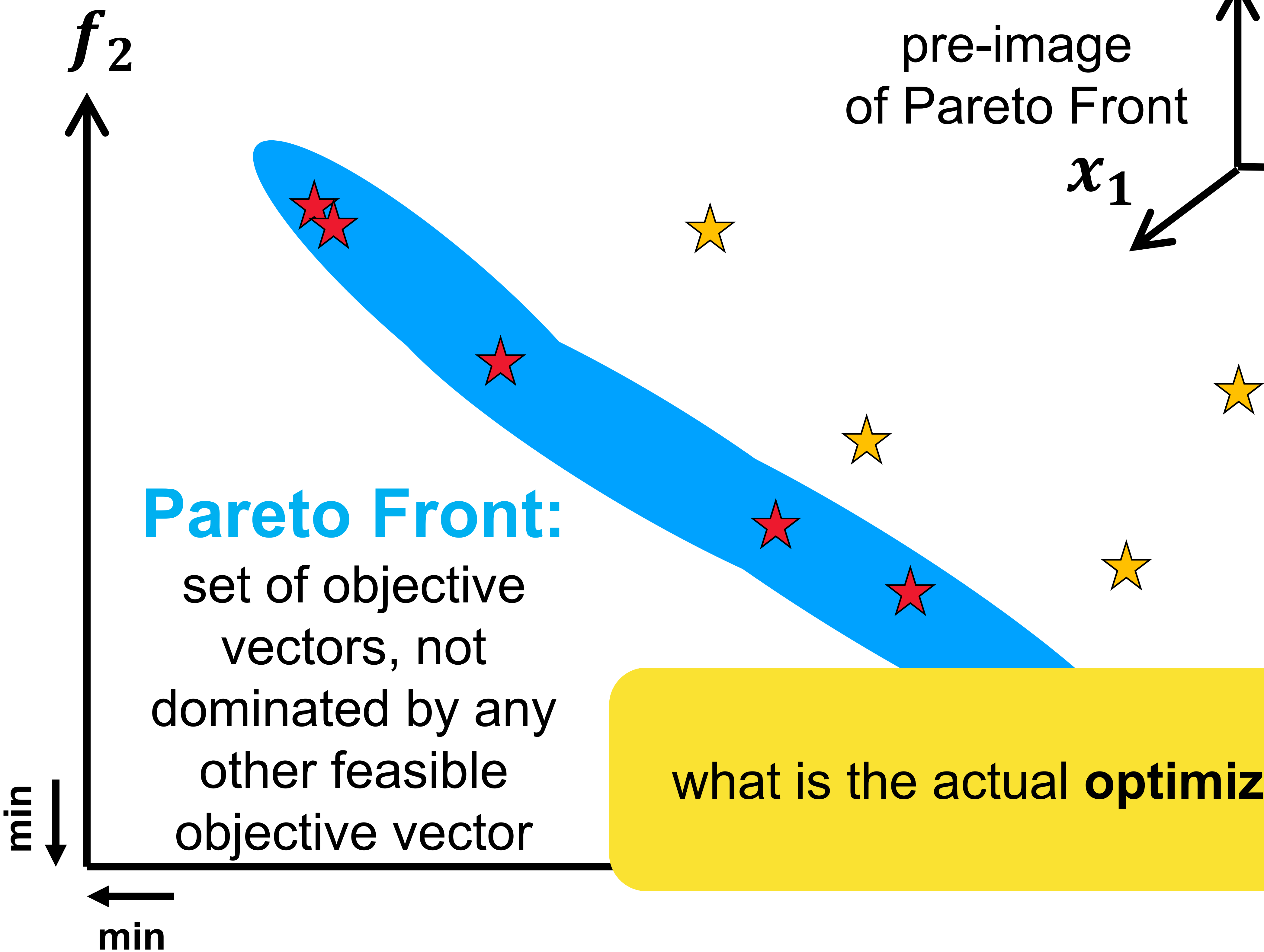
Pareto Set and Pareto Front



Pareto Set and Pareto Front

Pareto Set:
pre-image
of Pareto Front





Pareto Front:
set of objective
vectors, not
dominated by any
other feasible
objective vector

what is the actual **optimization goal**?

Optimization Goal

Here: optimize Hypervolume (HV) indicator

Optimization Goal

Here: optimize Hypervolume (HV) indicator

Even then, 2 optimization goals:

- fixed size approximation (maximizing HV with p points)

  given $p \in \mathbb{N}$, find p solutions $\{x_1, \ldots, x_p\} \subset \mathbb{R}^n$ with maximal hypervolume $HV(\{x_1, \ldots, x_p\})$

- unbounded approximation (= finding the “entire” Pareto Set)

Optimization Goal

Here: optimize Hypervolume (HV) indicator

Even then, 2 optimization goals:

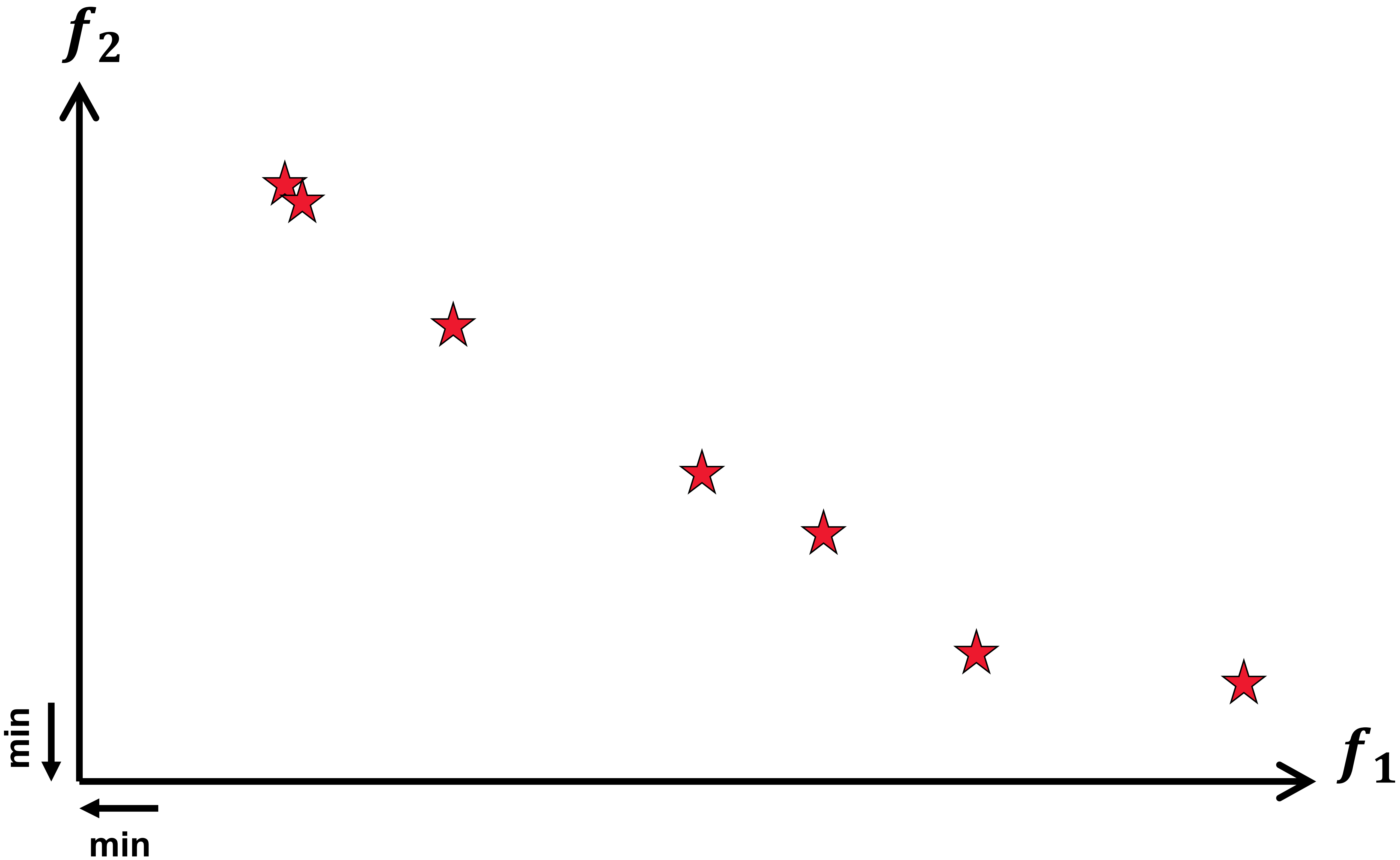
- fixed size approximation (maximizing HV with p points)

  given $p \in \mathbb{N}$, find p solutions $\{x_1, \dots, x_p\} \subset \mathbb{R}^n$ with maximal hypervolume $HV(\{x_1, \dots, x_p\})$

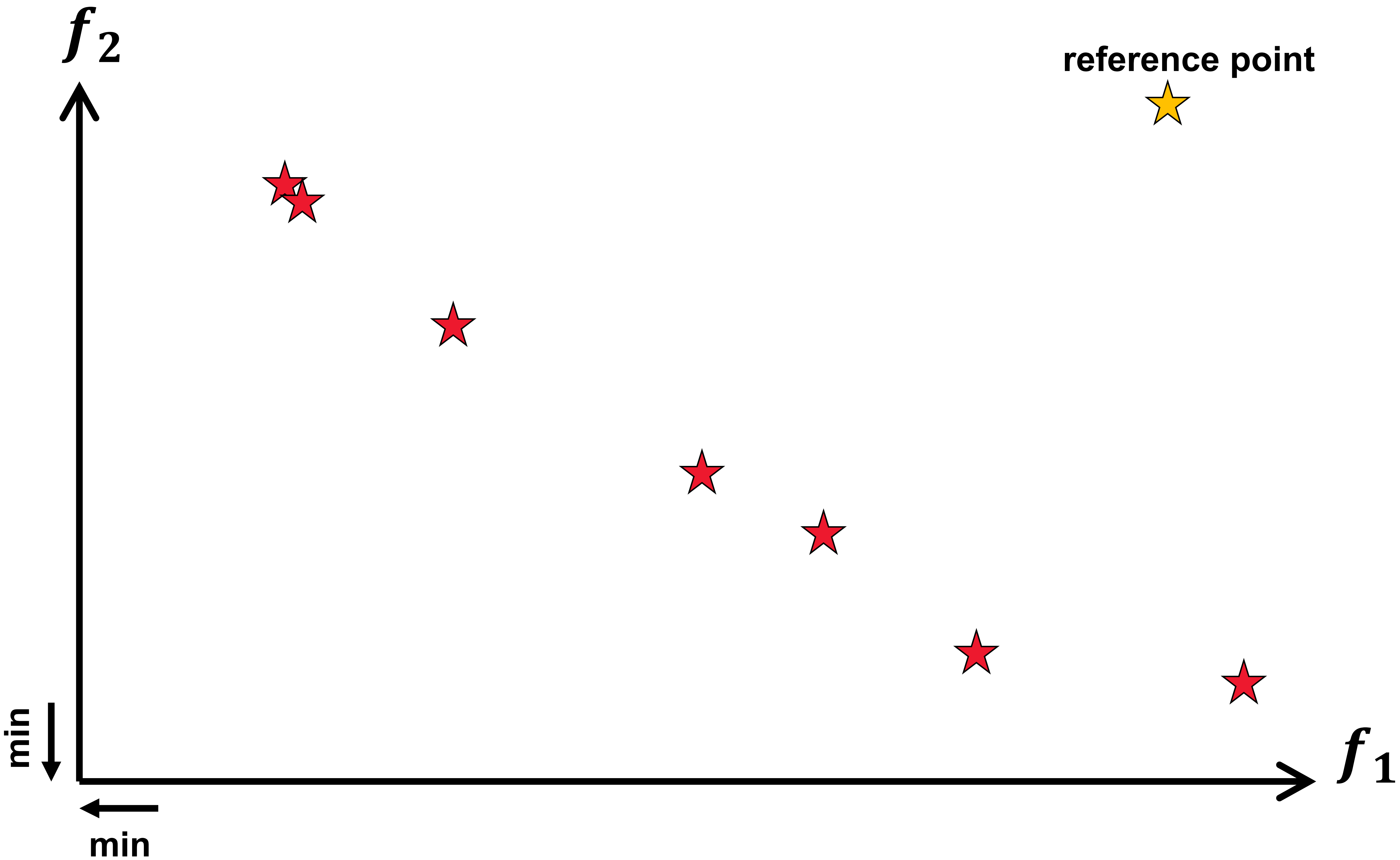
- unbounded approximation (= finding the "entire" Pareto Set)

The Hypervolume Indicator

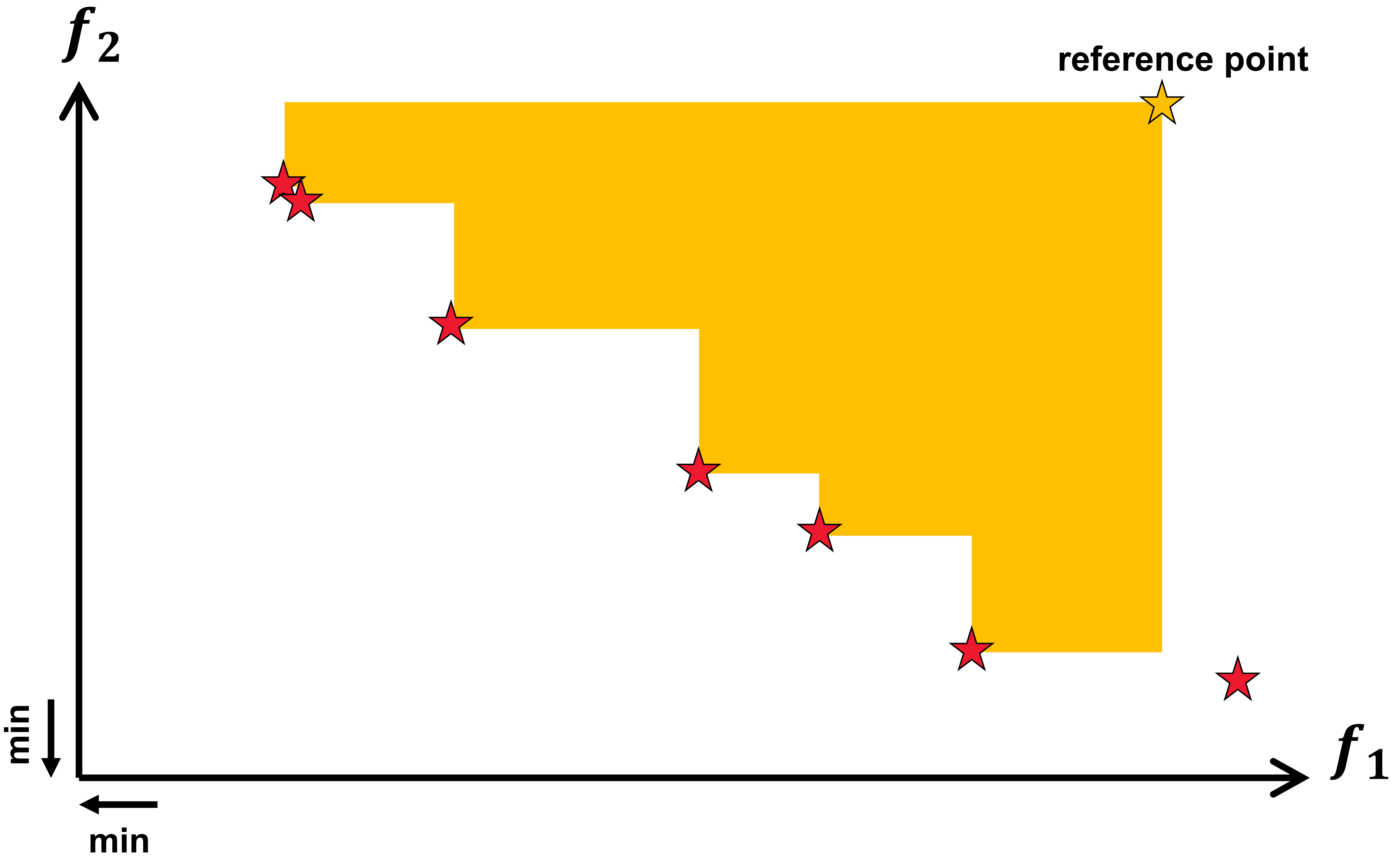
The Hypervolume Indicator



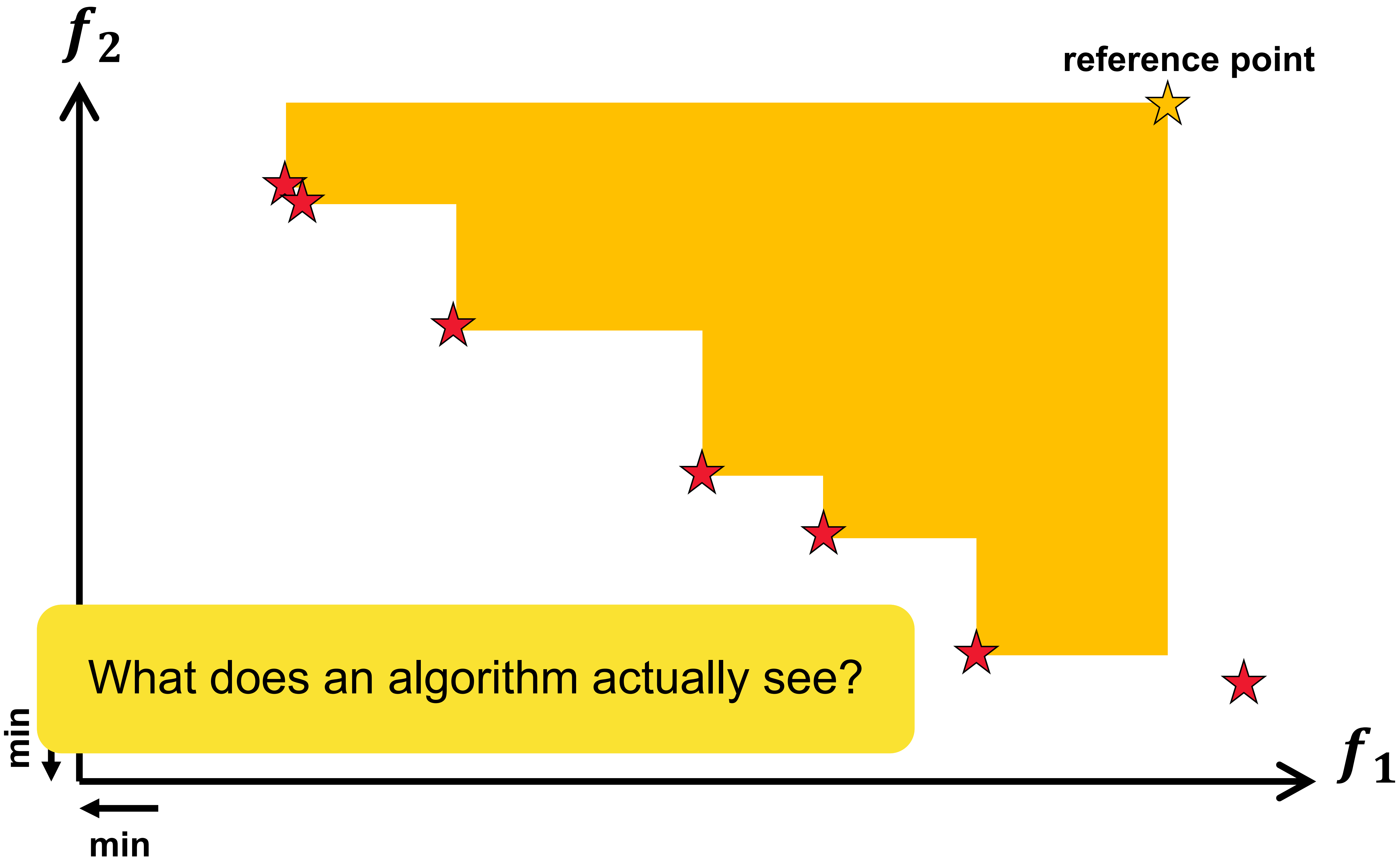
The Hypervolume Indicator



The Hypervolume Indicator



The Hypervolume Indicator

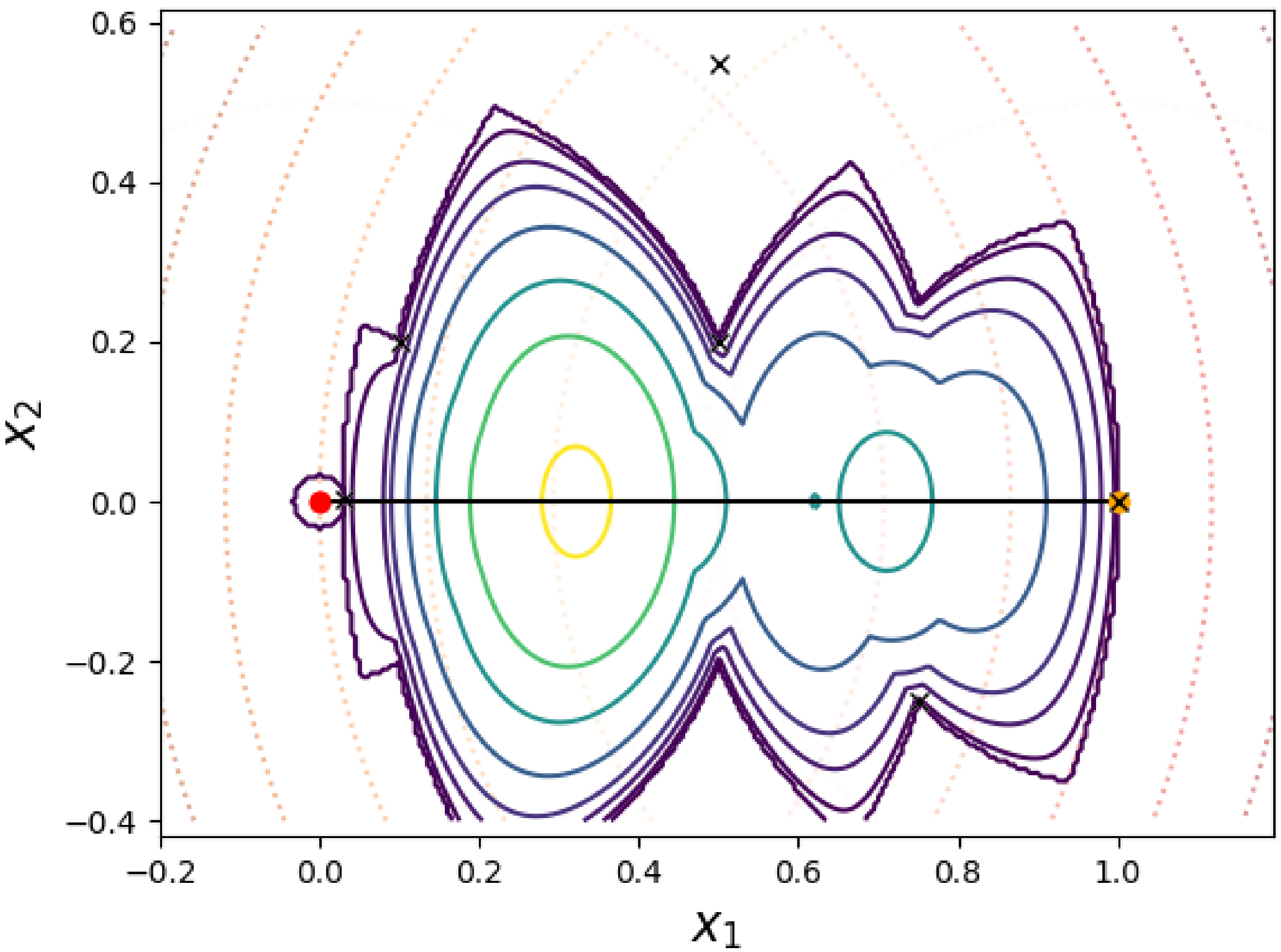


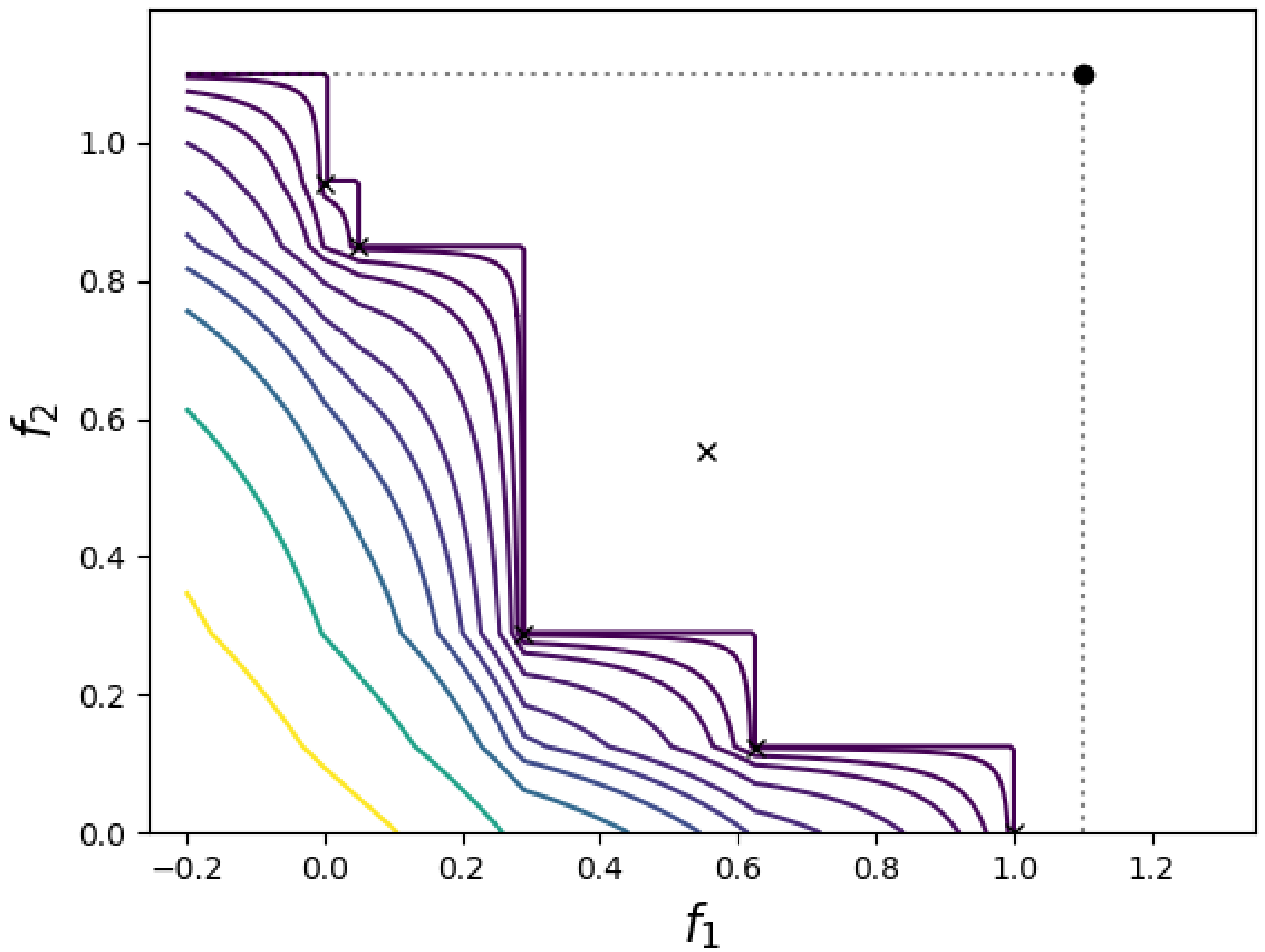
consider hypervolume maximization as $(n \cdot p)$-dimensional problem
then:
subspace optimization = optimize the hypervolume *improvement* (HVI)

Level Sets

consider hypervolume maximization as $(n \cdot p)$-dimensional problem
then:
subspace optimization = optimize the hypervolume *improvement* (HVI)

level sets of hypervolume improvement



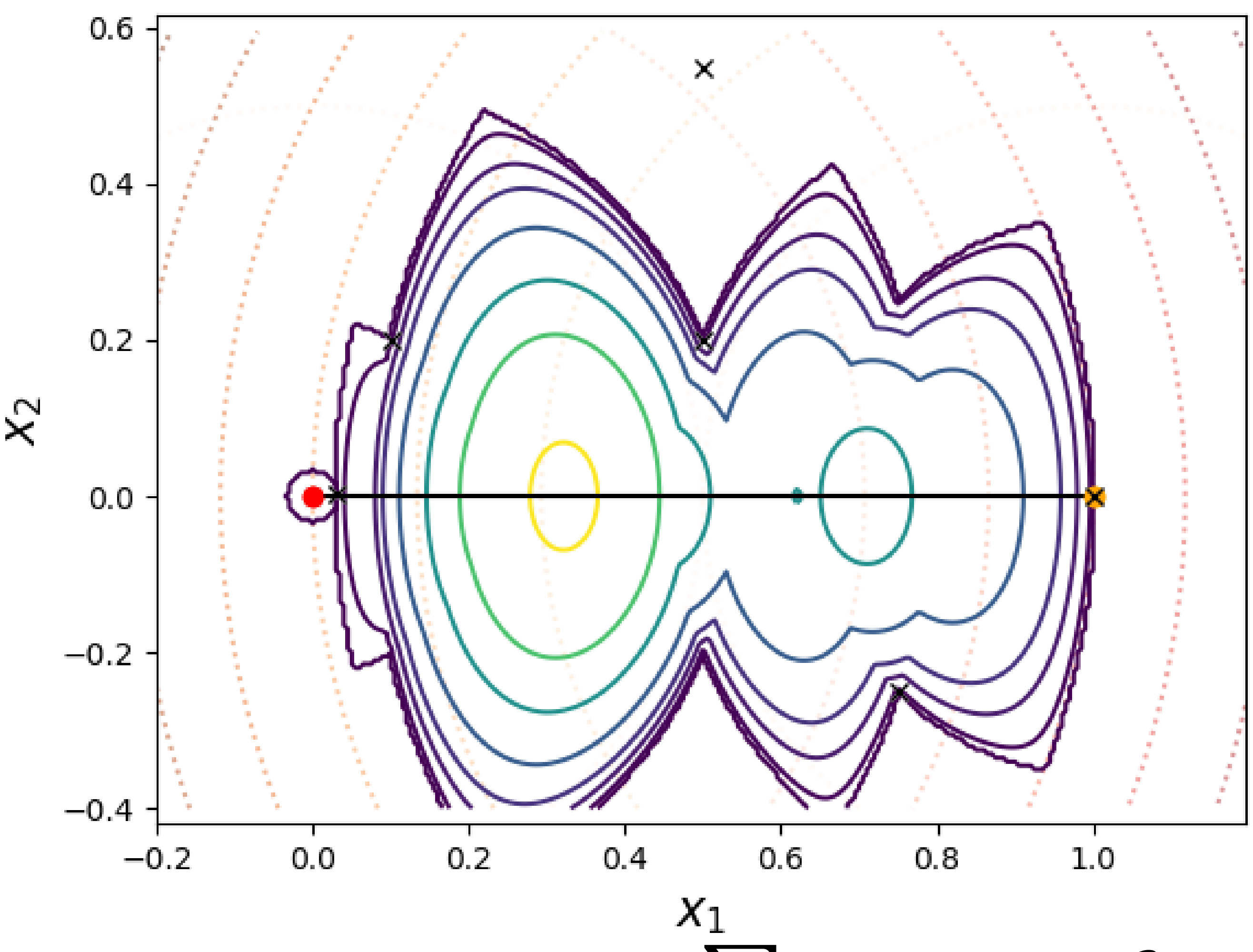


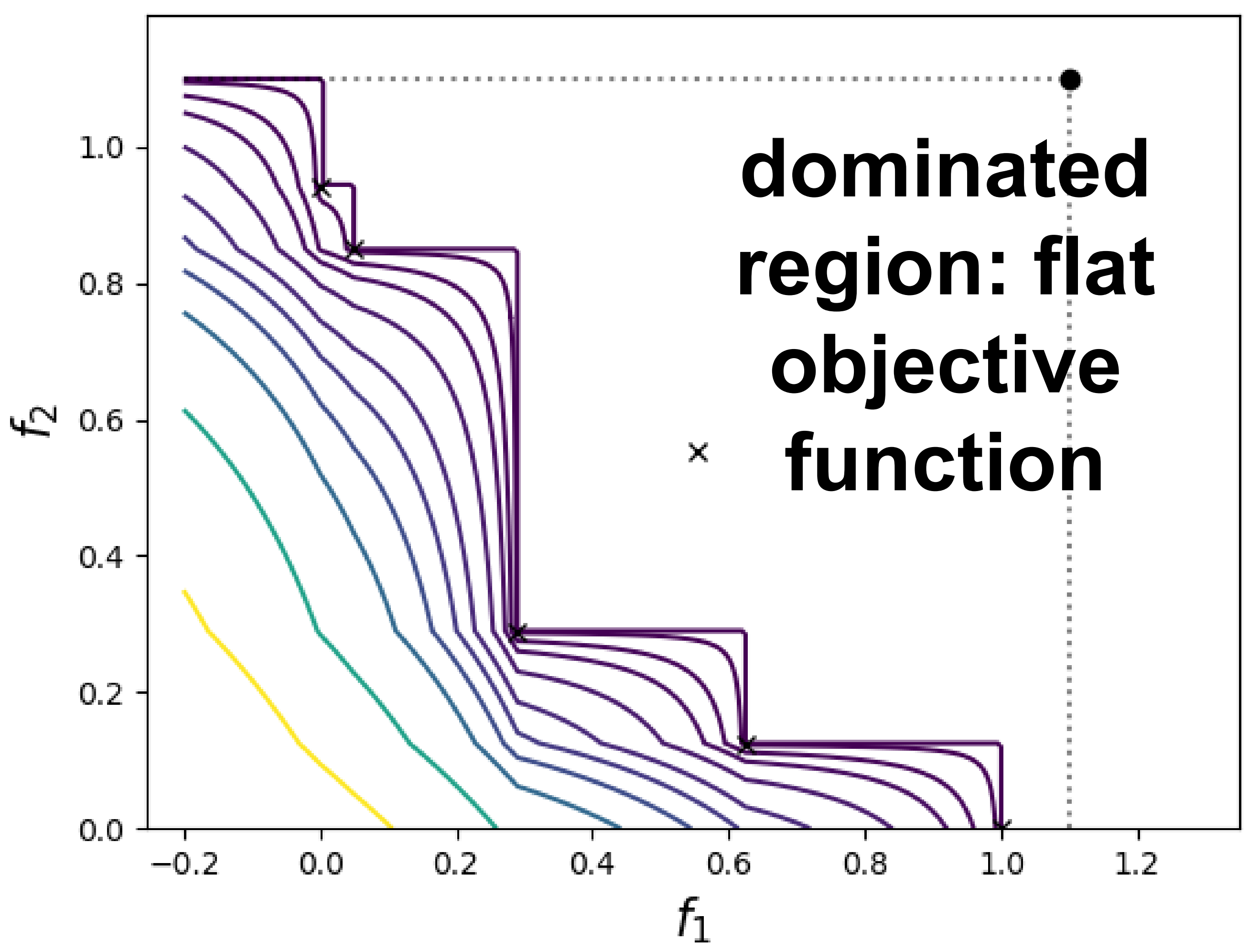
$$\boldsymbol{f_1(x)} = \sum_{\boldsymbol{i}} \left(\boldsymbol{x_i} - \boldsymbol{x_{opt,f1}}\right)^2 \qquad \boldsymbol{f_2(x)} = \sum_{\boldsymbol{i}} \left(\boldsymbol{x_i} - \boldsymbol{x_{opt,f2}}\right)^2$$

Level Sets

consider hypervolume maximization as $(n \cdot p)$-dimensional problem
then:
subspace optimization = optimize the hypervolume *improvement* (HVI)

level sets of hypervolume improvement



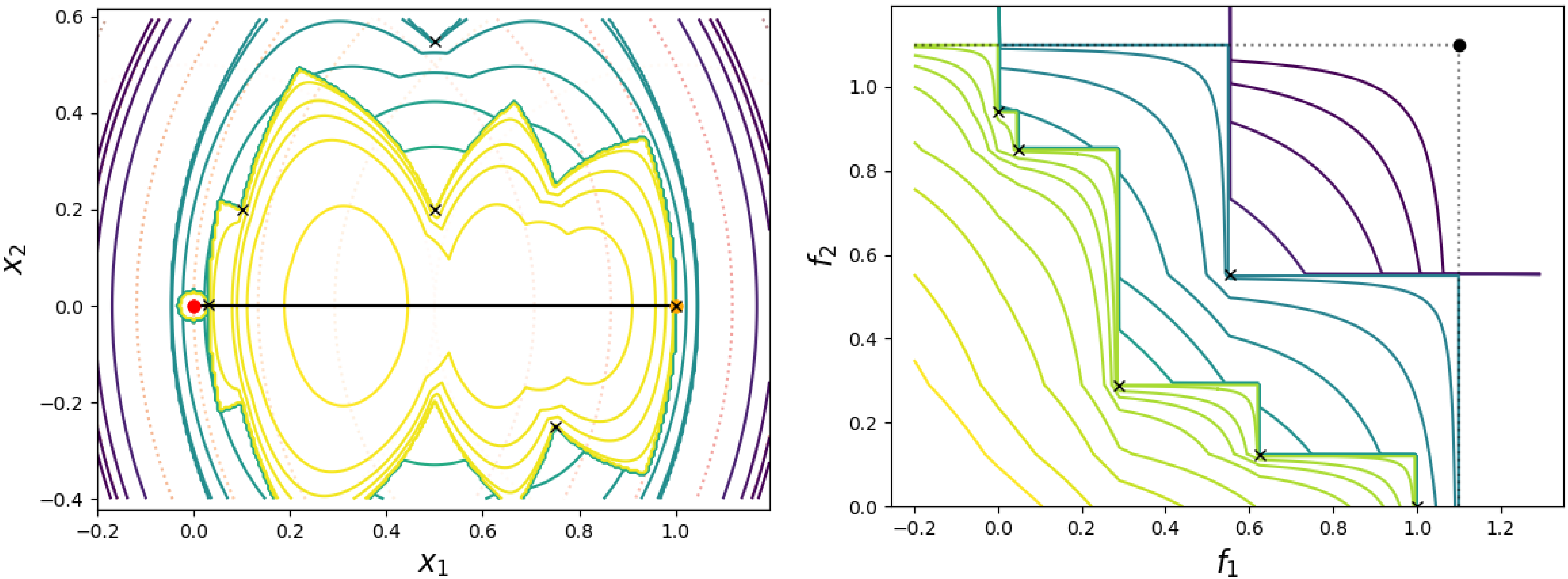


$$\boldsymbol{f_1(x) = \sum_i (x_i - x_{opt,f1})^2} \qquad \boldsymbol{f_2(x) = \sum_i (x_i - x_{opt,f2})^2}$$

Level Sets

additional use of non-dominated sorting

level sets of hypervolume improvement + non-dom. sorting

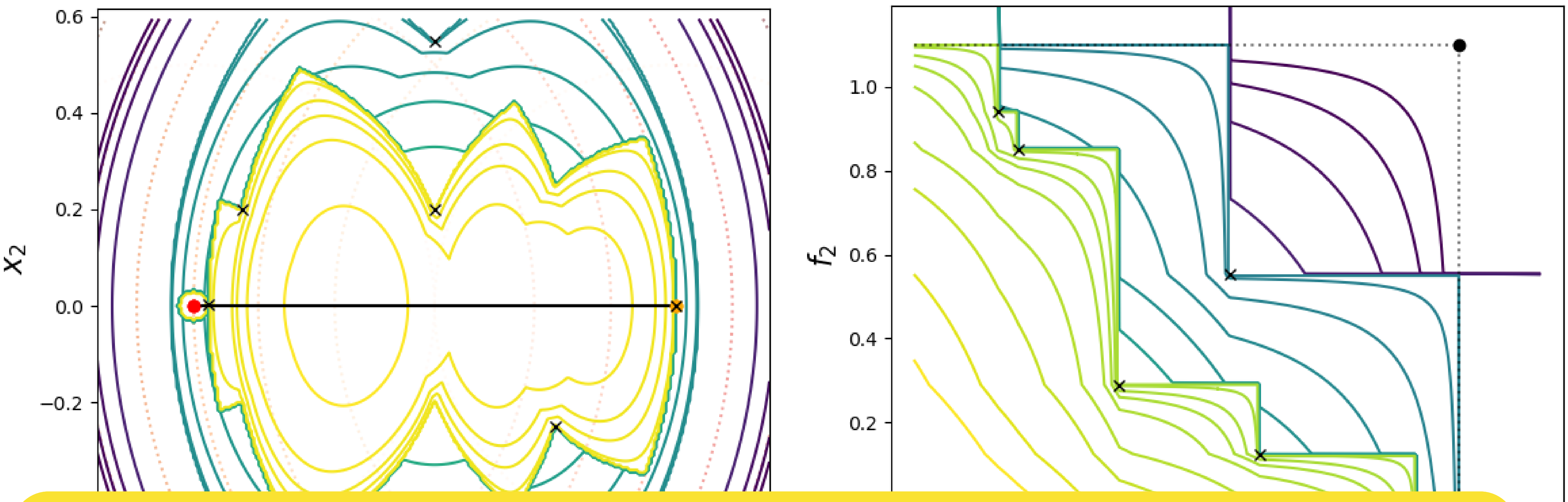


$f_1(x) = \sum_i (x_i - x_{opt,f1})^2$ $f_2(x) = \sum_i (x_i - x_{opt,f2})^2$

Level Sets

additional use of non-dominated sorting

level sets of hypervolume improvement + non-dom. sorting



Problems:

❶ **still flat regions without gradient**

❷ **search directed towards points, we already know**

“upgrading” the HVI

Cheikh Touré et al. 2019

“Uncrowded” Hypervolume Improvement (UHVI):

1) HVI if non-dominated
2) negative distance to empirical Pareto front otherwise

“upgrading” the HVI

“Uncrowded” Hypervolume Improvement (UHVI):

1) HVI if non-dominated
2) negative distance to empirical Pareto front otherwise

Quality of solution $\mathbf{s}$ wrt. current Pareto set approximation S:

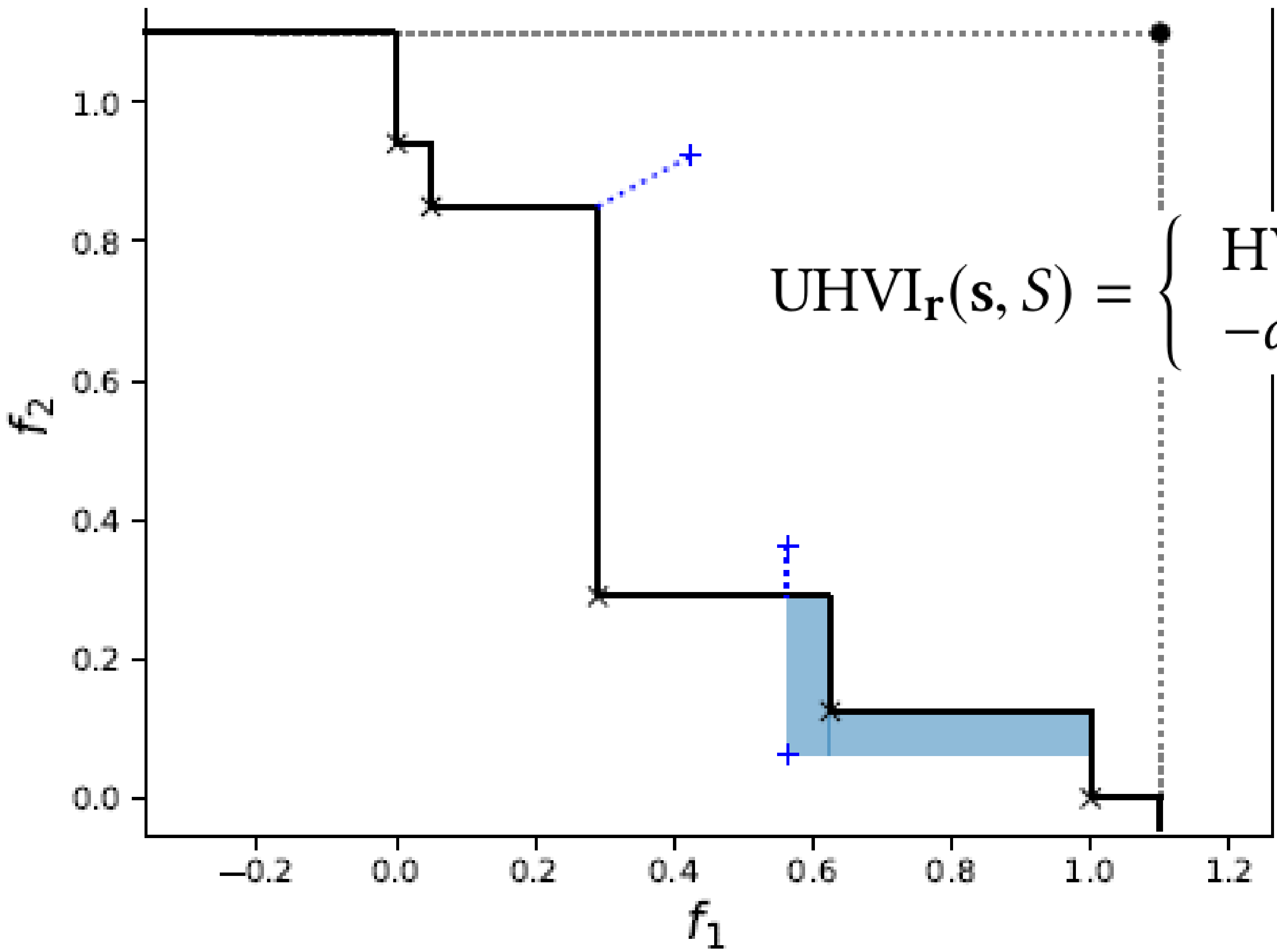
$$\mathrm{UHVI}_{\mathbf{r}}(\mathbf{s}, S) = \begin{cases} \mathrm{HVI}_{\mathbf{r}}(\mathbf{s}, S) & \text{if } \mathbf{EPF}_{S,\mathbf{r}} \nprec \mathbf{f}(\mathbf{s}) \\ -d_{\mathbf{r}}(\mathbf{s}, S) & \text{if } \mathbf{EPF}_{S,\mathbf{r}} \prec \mathbf{f}(\mathbf{s}) \end{cases}$$

"upgrading" the HVI

"Uncrowded" Hypervolume Improvement (UHVI):

1) HVI if non-dominated
2) negative distance to empirical Pareto front otherwise

level sets of UHVI

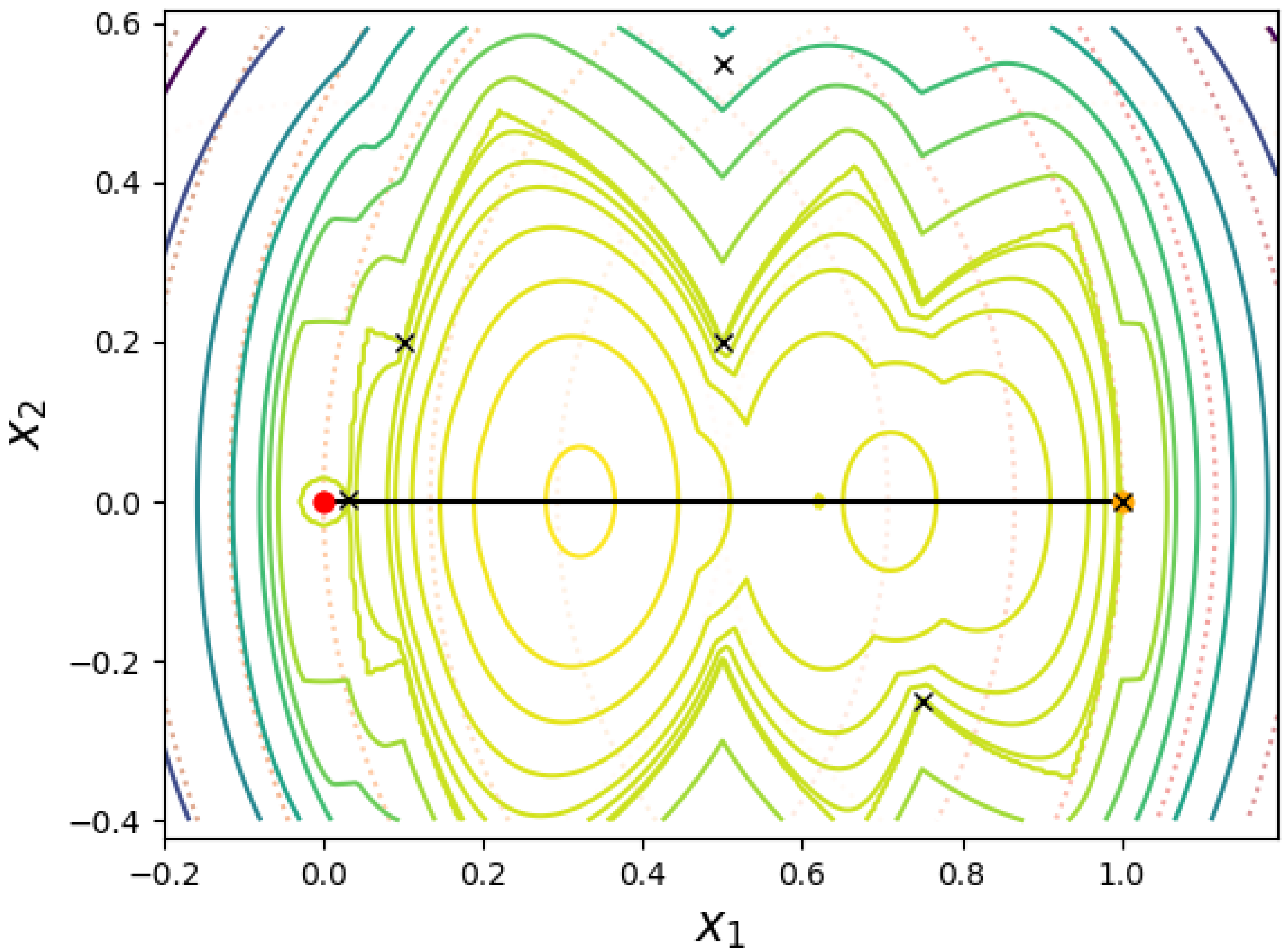


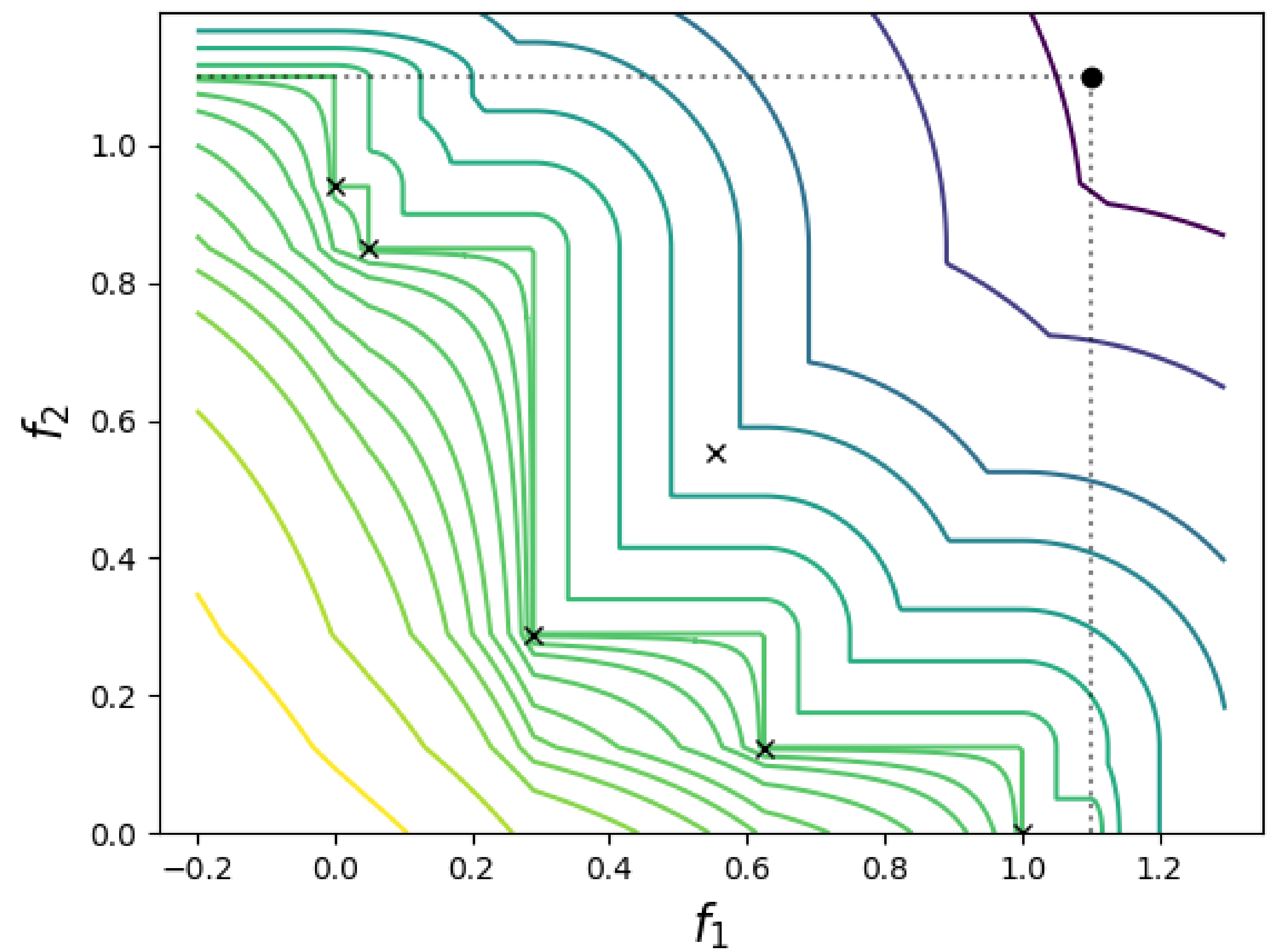
$$\mathrm{UHVI}_{\mathbf{r}}(\mathbf{s}, S) = \begin{cases} \mathrm{HVI}_{\mathbf{r}}(\mathbf{s}, S) & \text{if } \mathbf{EPF}_{S,\mathbf{r}} \nprec \mathbf{f}(\mathbf{s}) \\ -d_{\mathbf{r}}(\mathbf{s}, S) & \text{if } \mathbf{EPF}_{S,\mathbf{r}} \prec \mathbf{f}(\mathbf{s}) \end{cases}$$

Level Sets of the UHVI

"Uncrowded" Hypervolume Improvement (UHVI):
1) HVI if non-dominated
2) negative distance to empirical Pareto front otherwise

level sets of UHVI





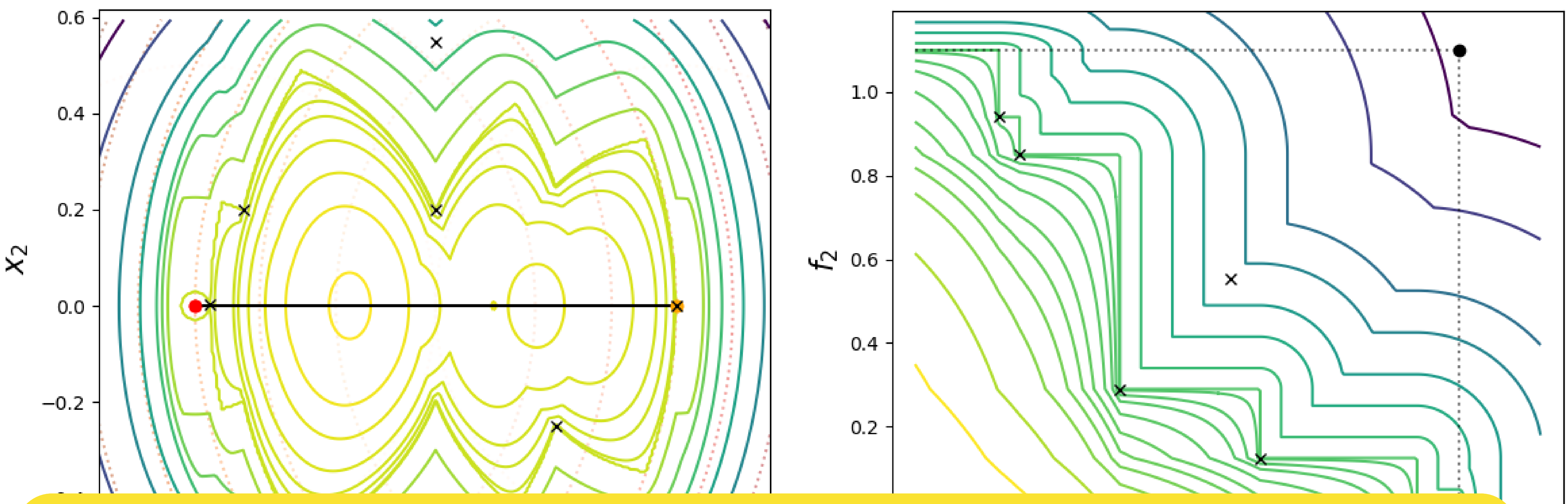
$$\boldsymbol{f_1(x) = \sum_i (x_i - x_{opt,f1})^2} \qquad \boldsymbol{f_2(x) = \sum_i (x_i - x_{opt,f2})^2}$$

Level Sets of the UHVI

"Uncrowded" Hypervolume Improvement (UHVI):

1) HVI if non-dominated
2) negative distance to empirical Pareto front otherwise

level sets of UHVI



❶ **non-zero indicator (+gradient) almost everywhere ☺**
❷ **and: search directed towards "empty" regions ☺**

Gradient available almost everywhere

- we can write $\mathrm{UHVI}_{r,S}(x) = \overline{\mathrm{UHVI}}_{r,f(S)}(f(x))$

Gradient available almost everywhere

- we can write $\mathrm{UHVI}_{r,S}(x) = \overline{\mathrm{UHVI}}_{r,f(S)}(f(x))$
- gradient of UHVI then computable via single-objective gradients:

PROPOSITION 2.1. *Let $x \in \mathbb{R}^n$. Assume f is differentiable in x and $\overline{UHVI}$ is differentiable in $f(x)$. Then the gradient of UHVI equals*

$$\nabla_x UHVI_{r,S}(x) = \partial_1 \overline{UHVI}_{r,f(S)}(f(x)) \nabla f_1(x) + \partial_2 \overline{UHVI}_{r,f(S)}(f(x)) \nabla f_2(x) \tag{2.3}$$

with ∂_i being the partial derivative with respect to the ith coordinate.

Gradient available almost everywhere

- we can write $\mathrm{UHVI}_{r,S}(x) = \overline{\mathrm{UHVI}}_{r,f(S)}(f(x))$
- gradient of UHVI then computable via single-objective gradients:

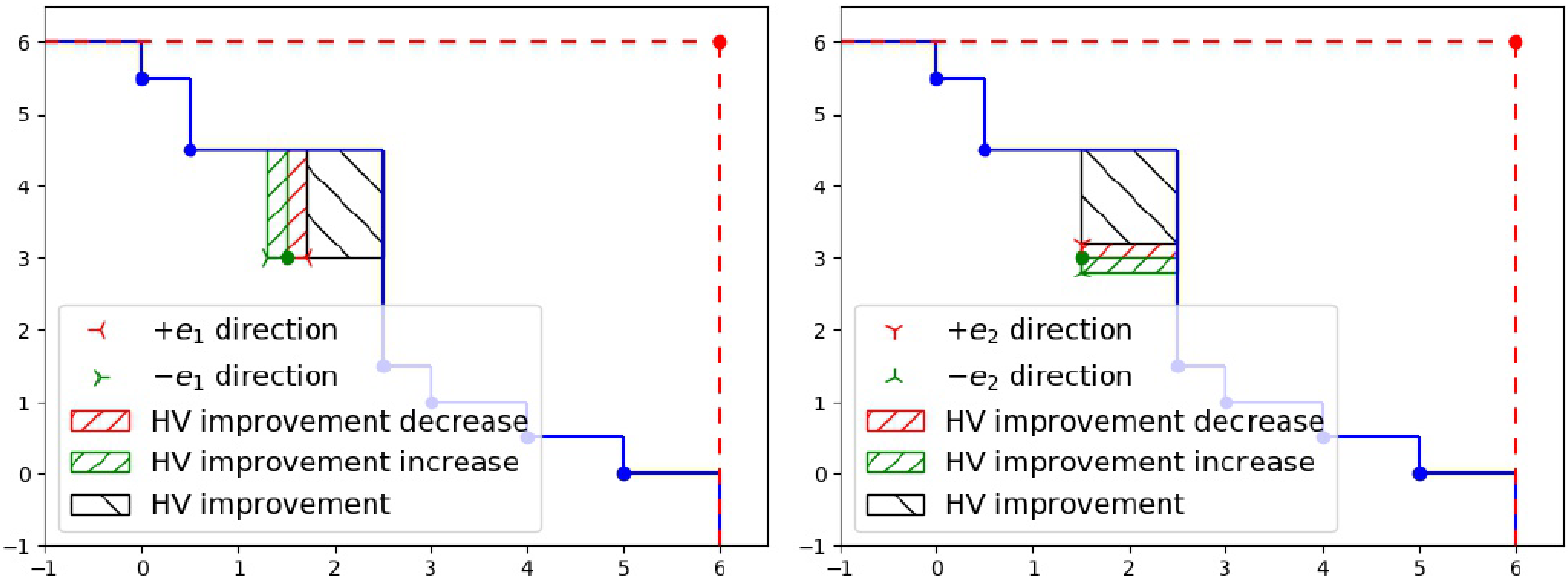
PROPOSITION 2.1. *Let $x \in \mathbb{R}^n$. Assume f is differentiable in x and $\overline{UHVI}$ is differentiable in $f(x)$. Then the gradient of UHVI equals*

$$\nabla_x UHVI_{r,S}(x) = \partial_1 \overline{UHVI}_{r,f(S)}(f(x)) \nabla f_1(x) + \partial_2 \overline{UHVI}_{r,f(S)}(f(x)) \nabla f_2(x) \tag{2.3}$$

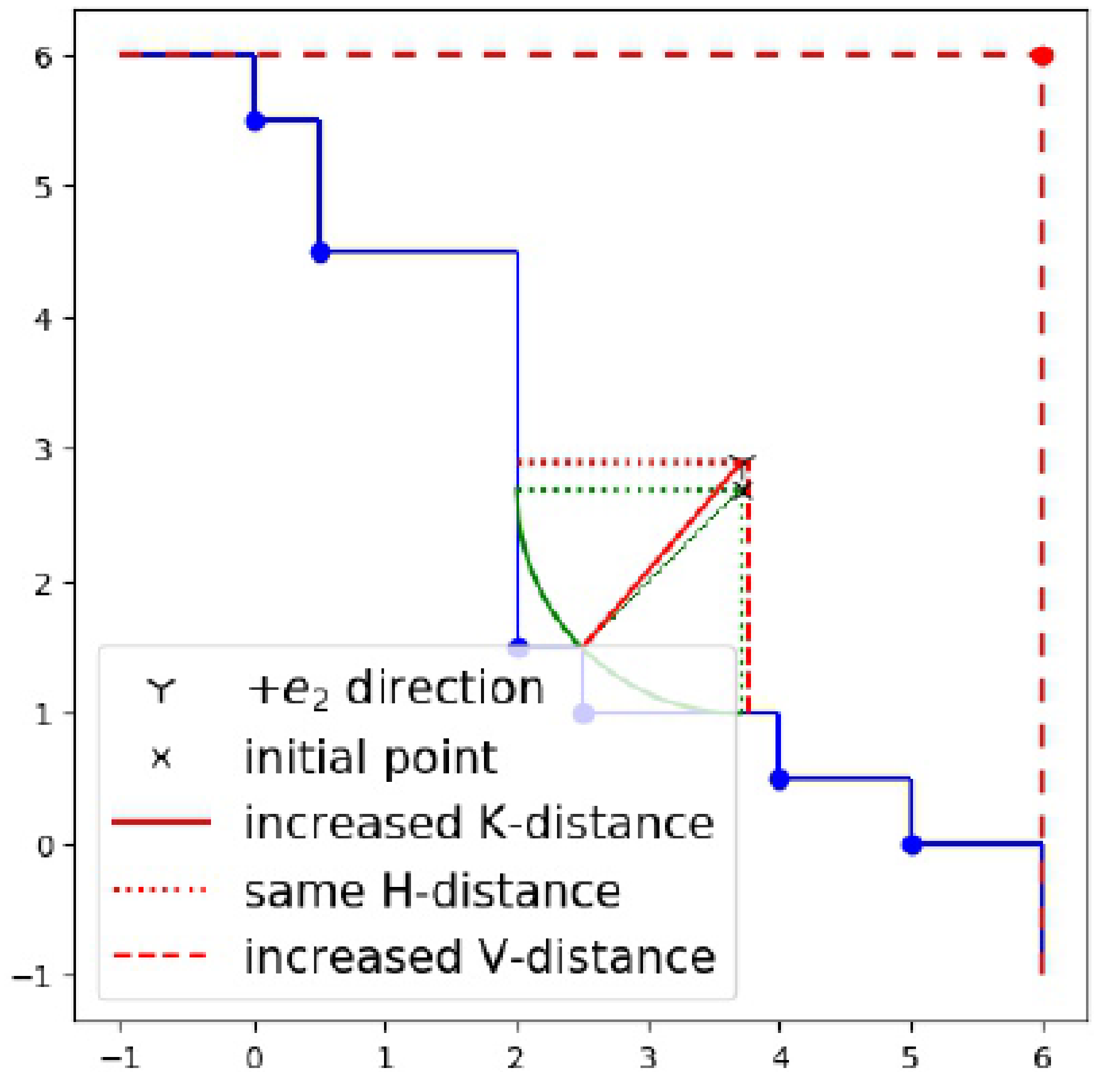
with ∂_i being the partial derivative with respect to the ith coordinate.

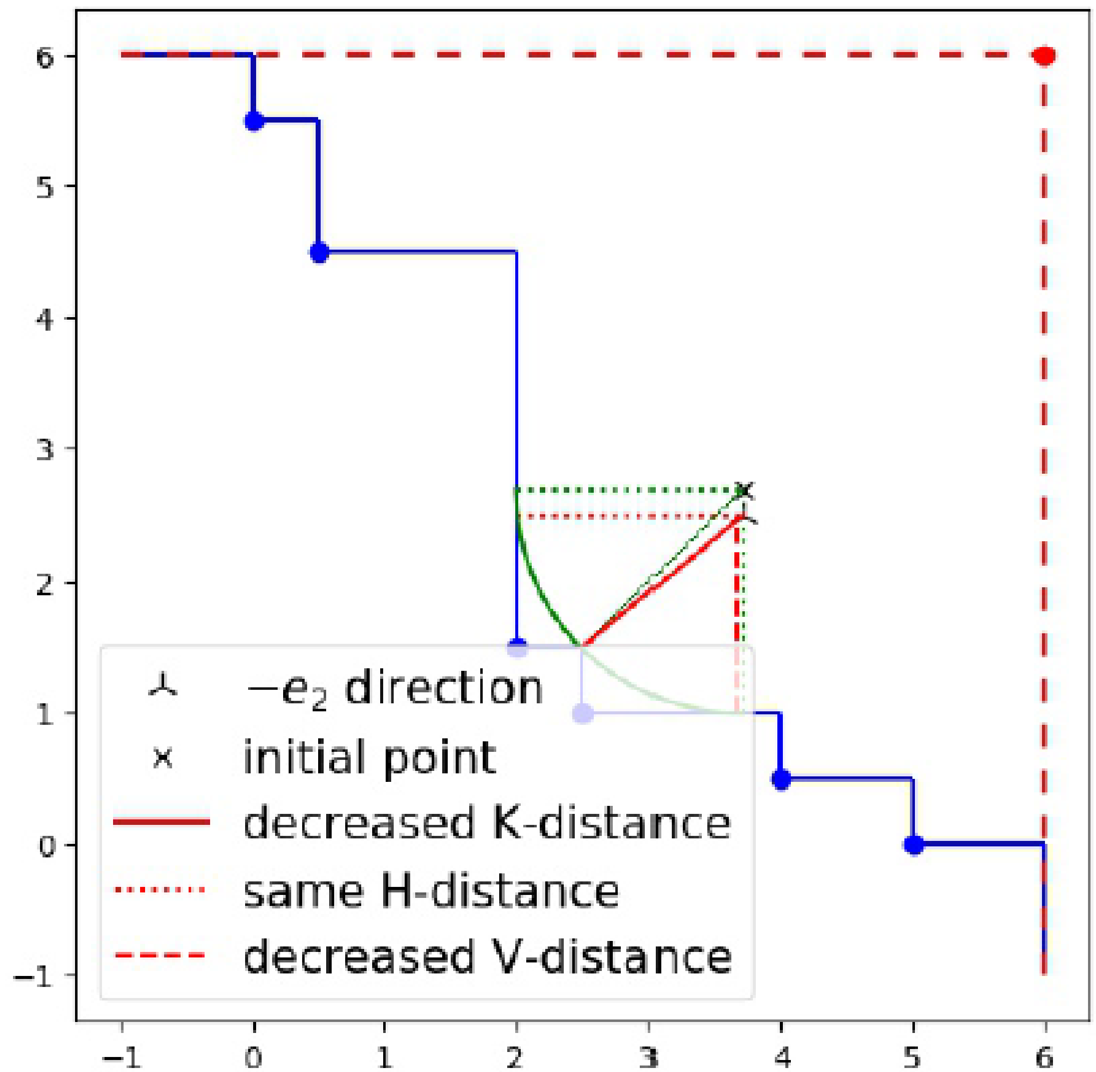
- $\partial_1 \overline{UHVI}_{r,f(S)}$ and $\partial_2 \overline{UHVI}_{r,f(S)}$ independent of f and relatively simple to compute

Gradient available almost everywhere

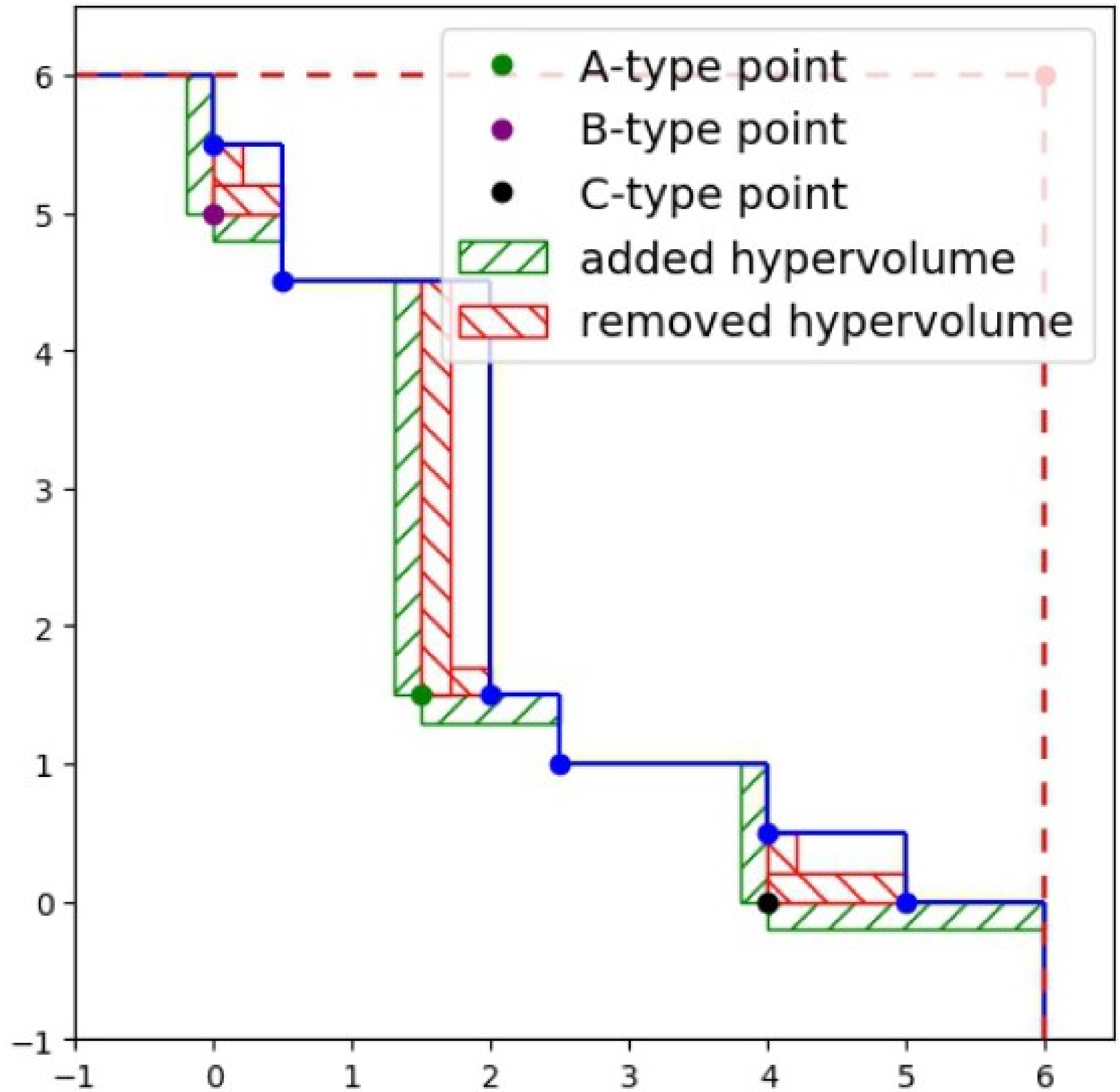


Gradient available almost everywhere





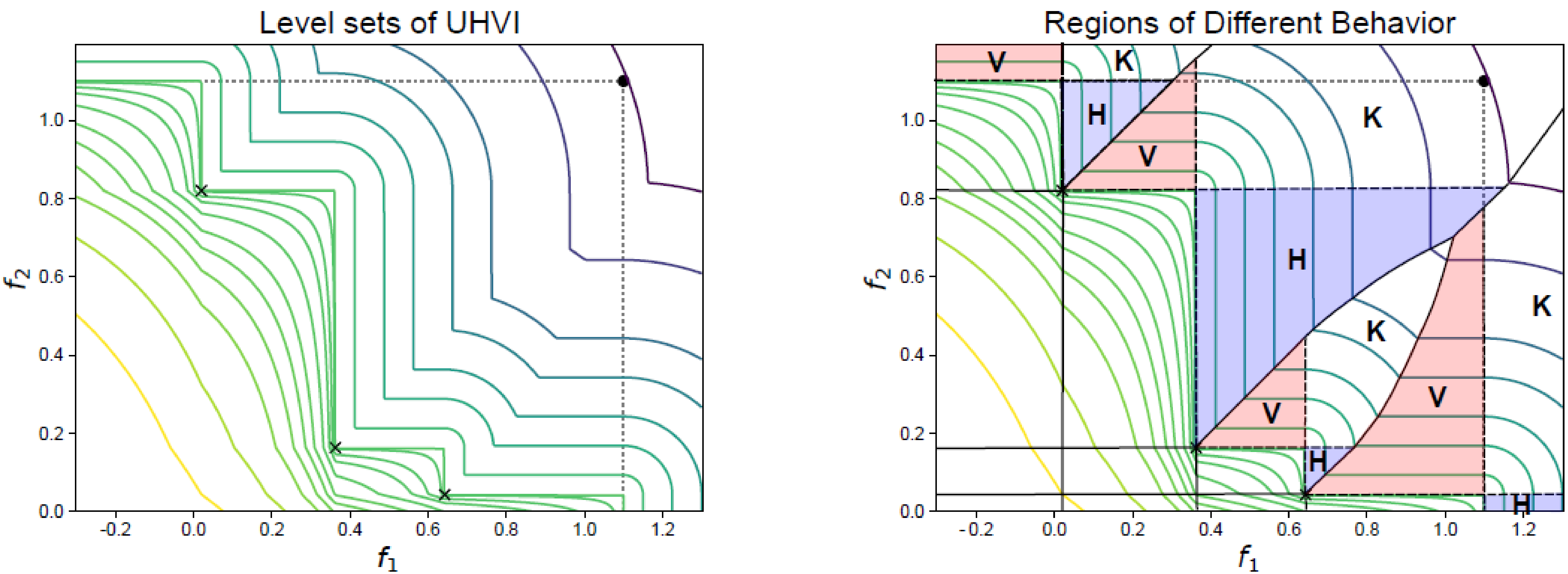
Cases where Gradient not defined



Gradient available almost everywhere

all those cases are analytically characterizable

when set S is given



hence: non-differentiable regions are null-sets

assuming that single-objective gradients are differentiable

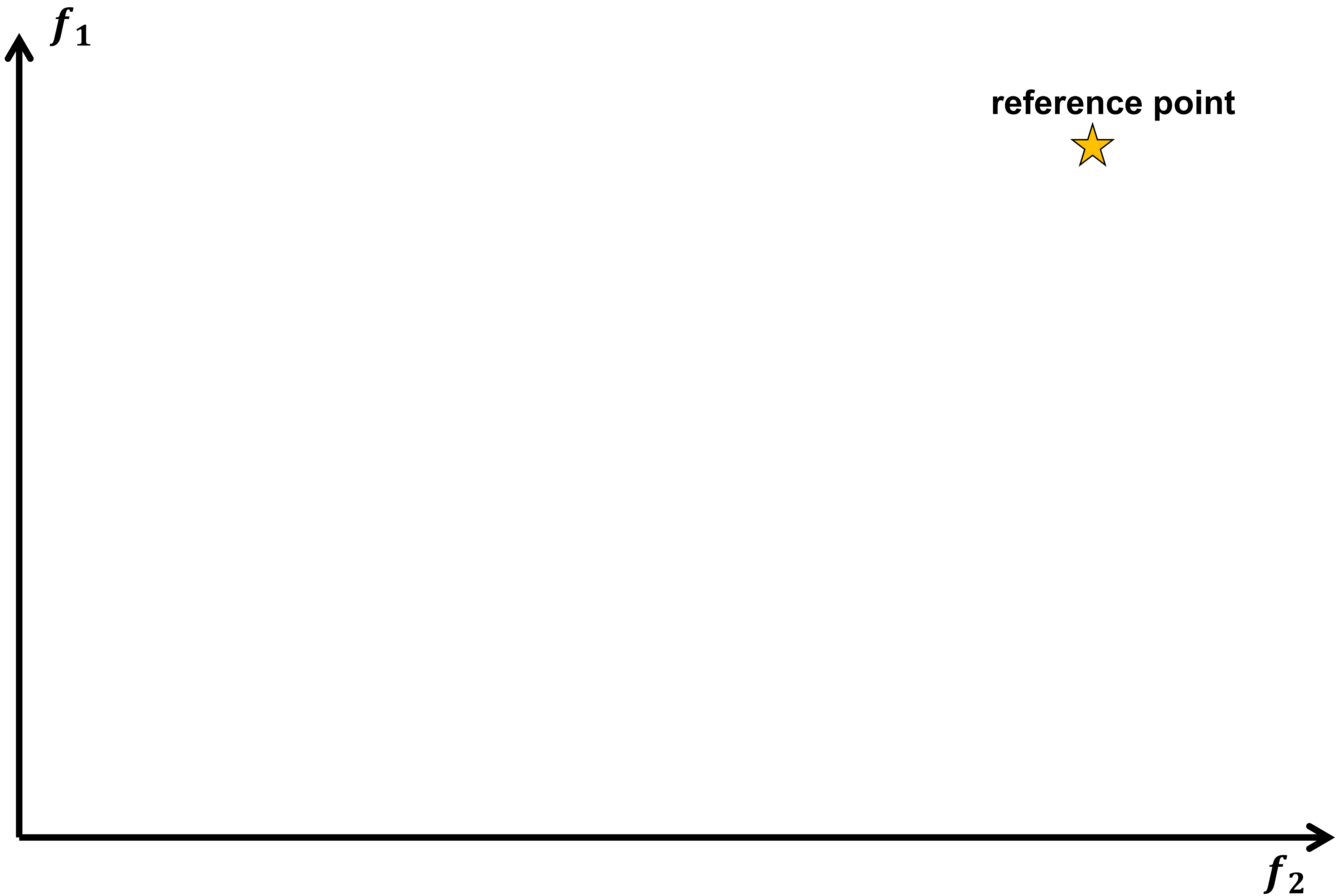
Unbounded UHVI optimization

Idea:

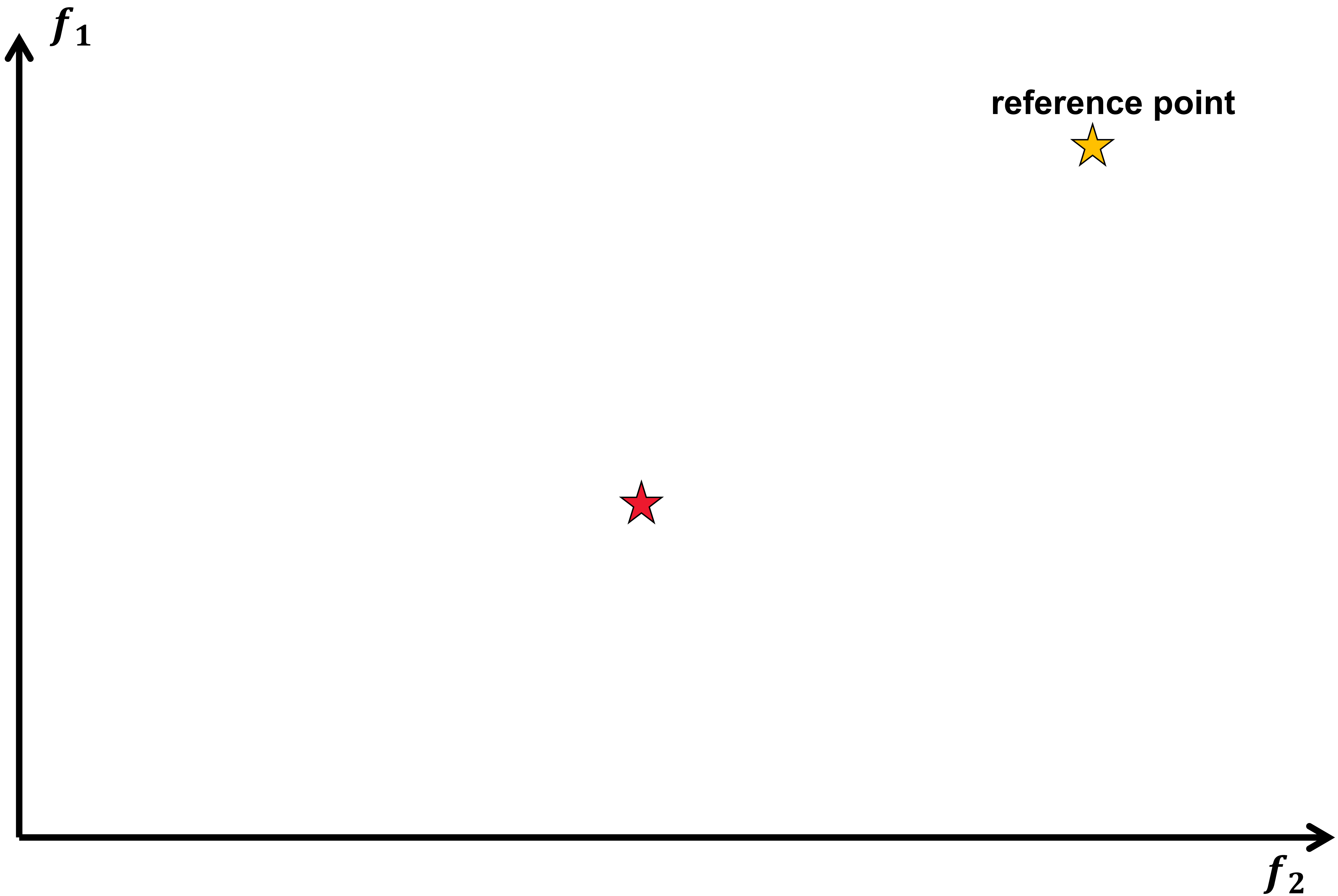
- start with $S = \emptyset$
- while not happy:
  - use single-objective solver to optimize $UHVI_{\mathbf{r}}(x, S)$
  - $S \ = S \cup$ incumbent of last run when stopped

Unbounded UHVI optimization

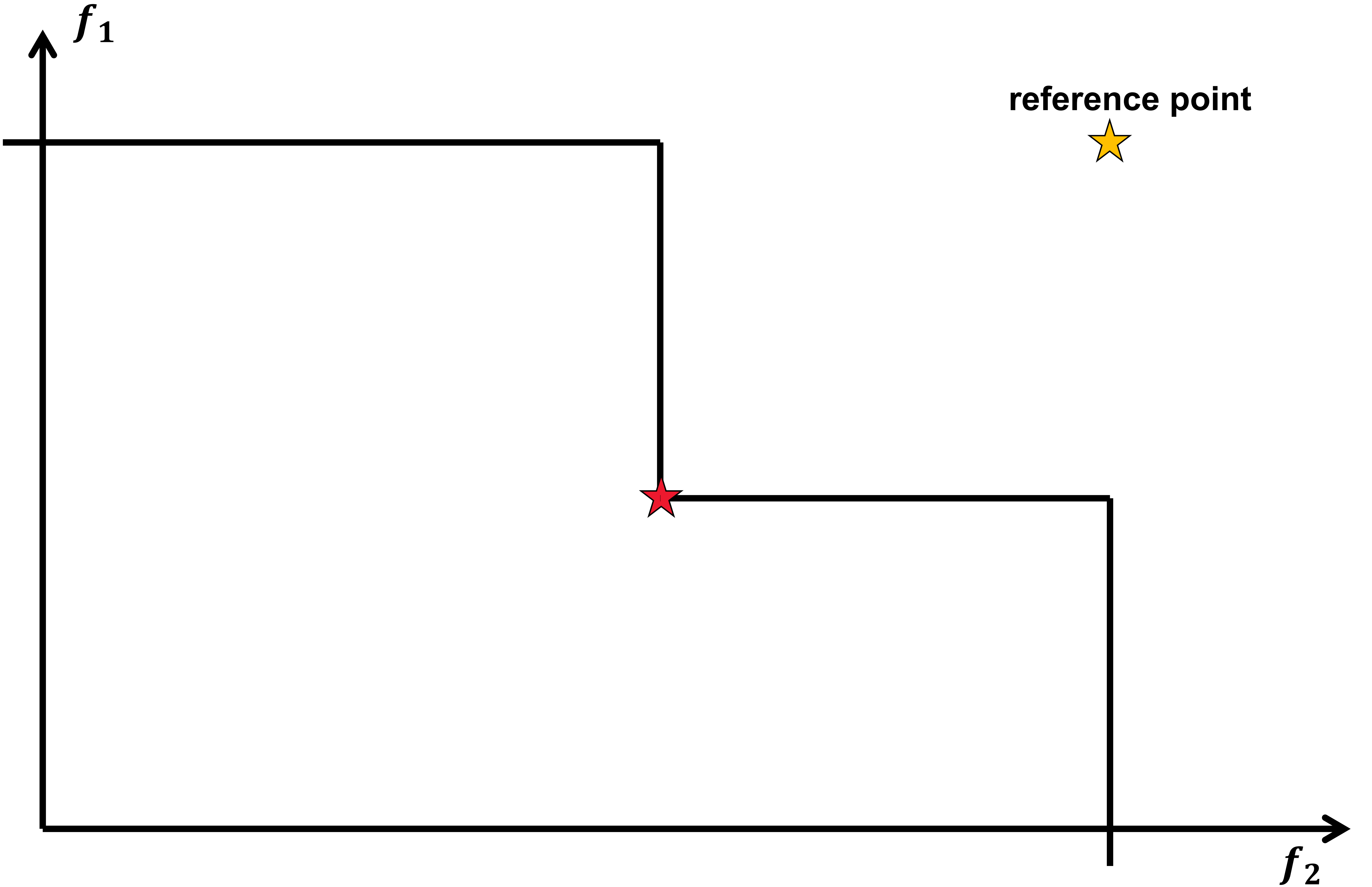
Unbounded UHVI optimization



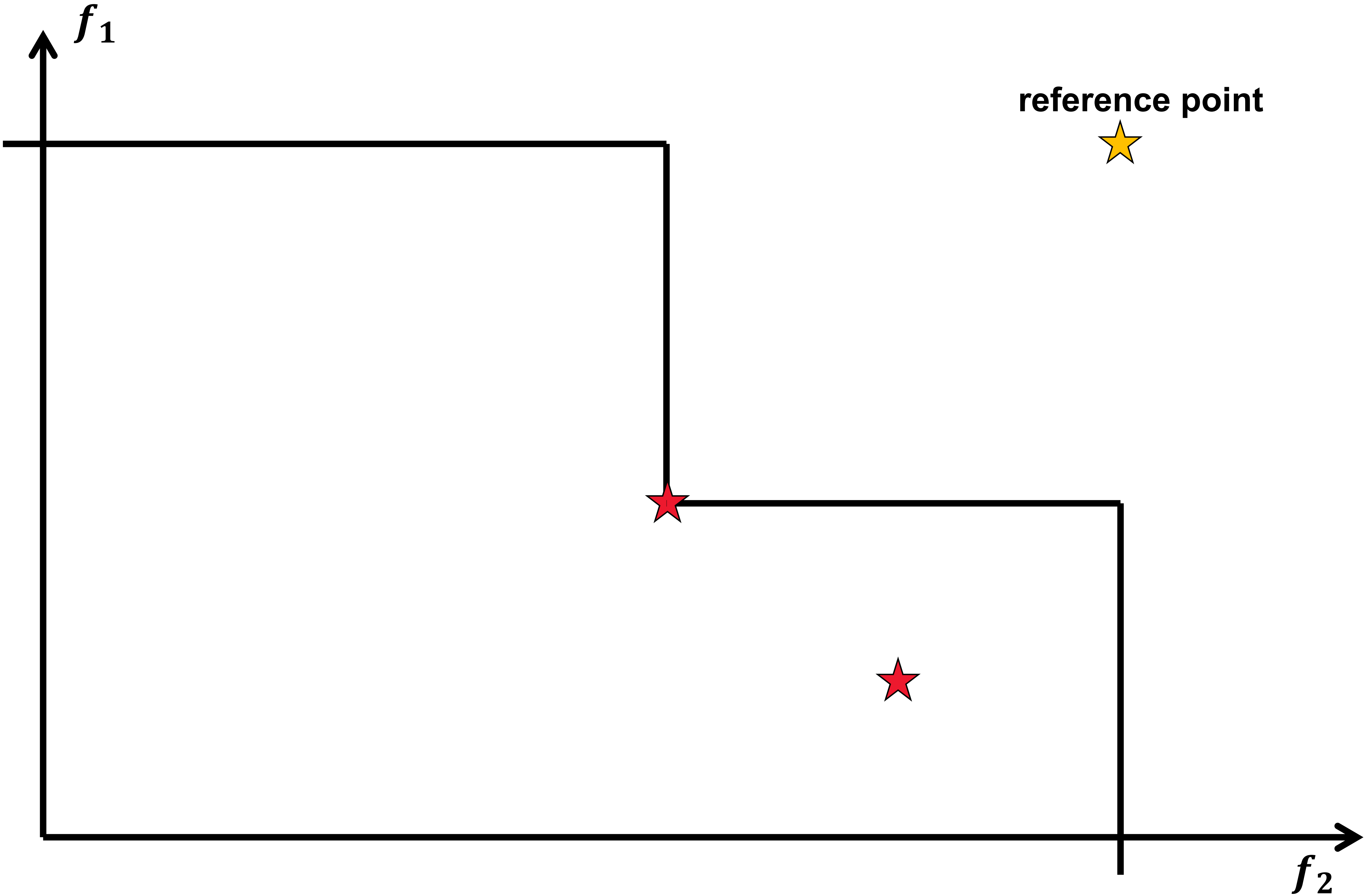
Unbounded UHVI optimization



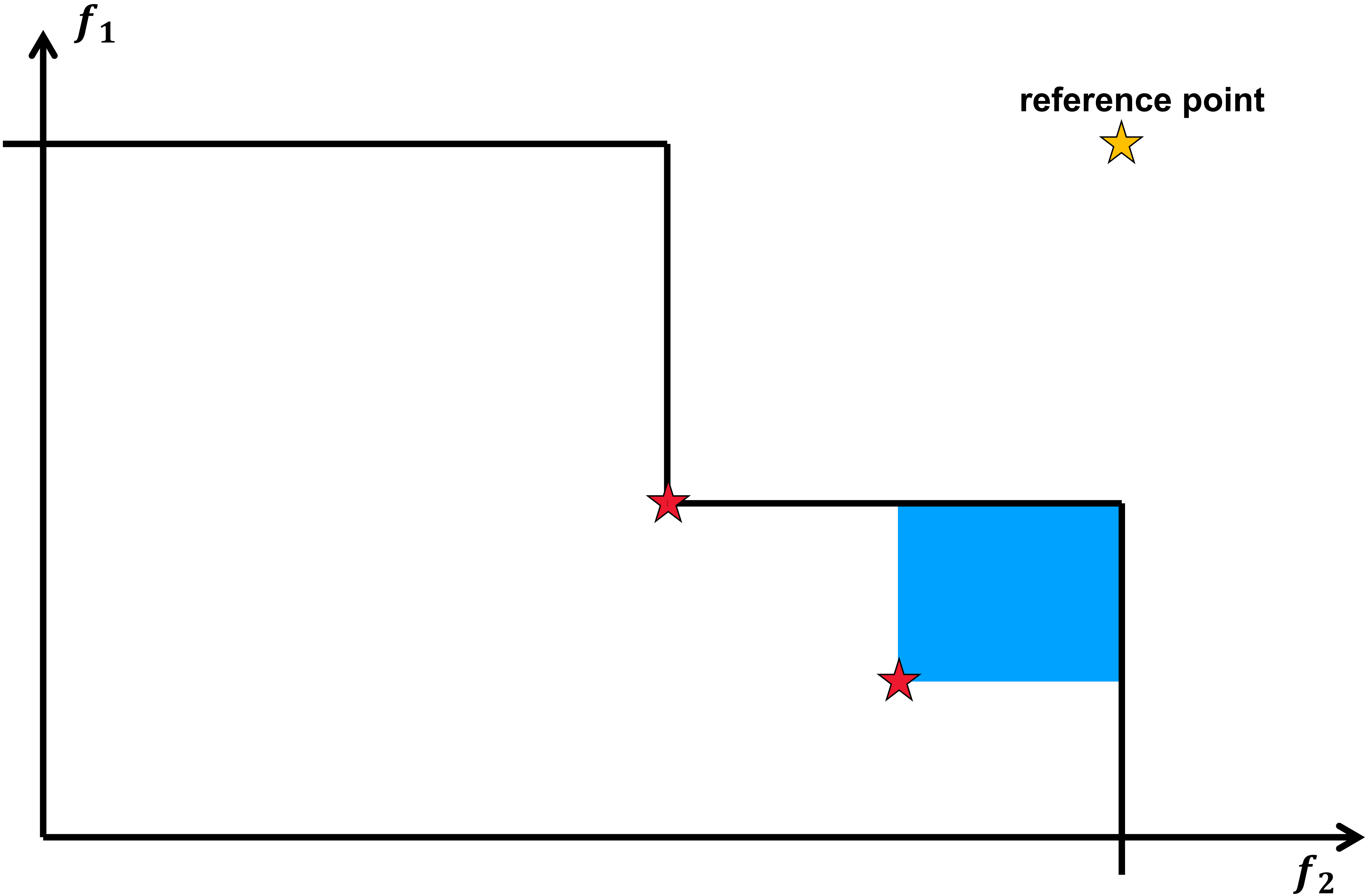
Unbounded UHVI optimization



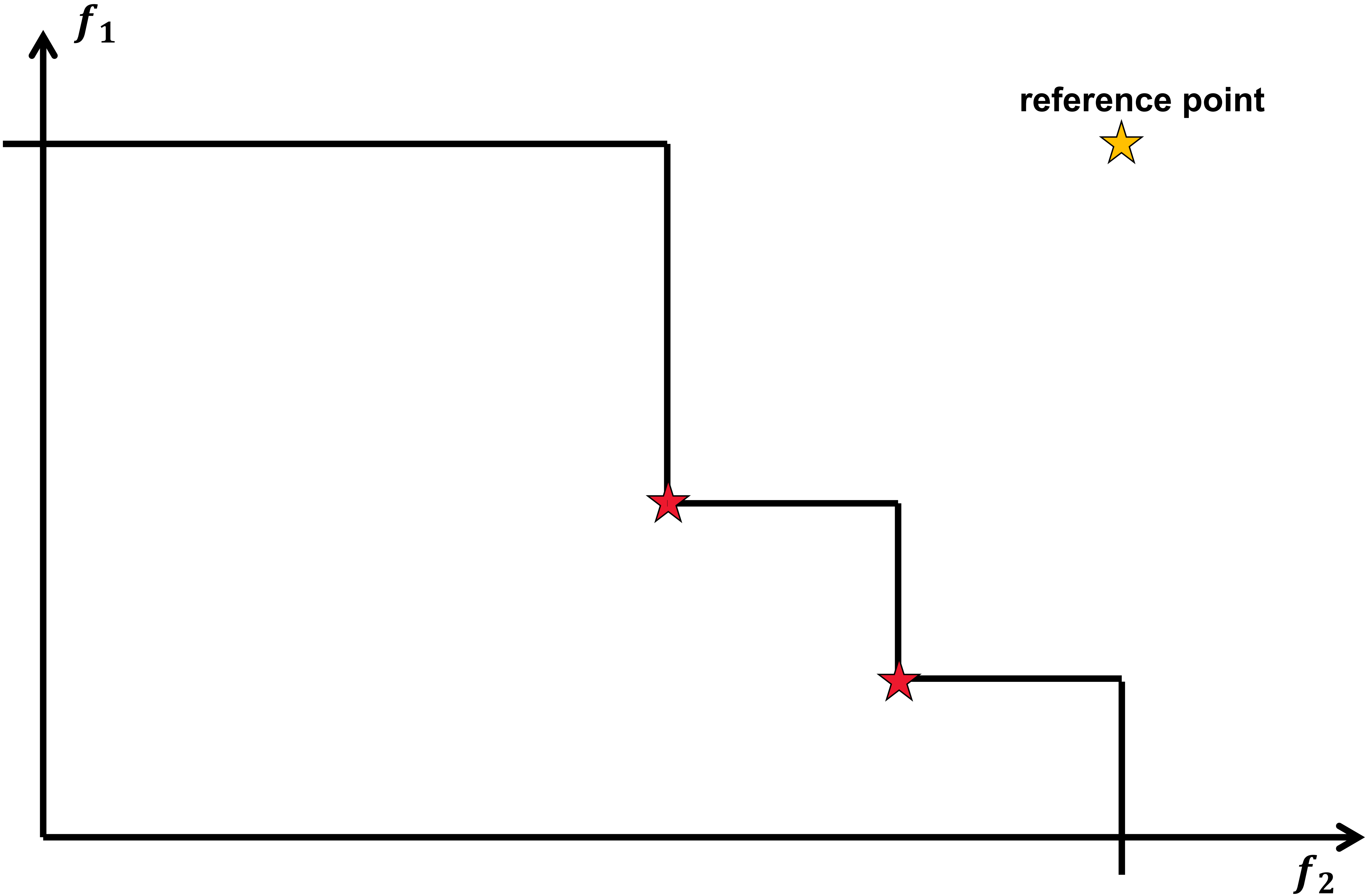
Unbounded UHVI optimization



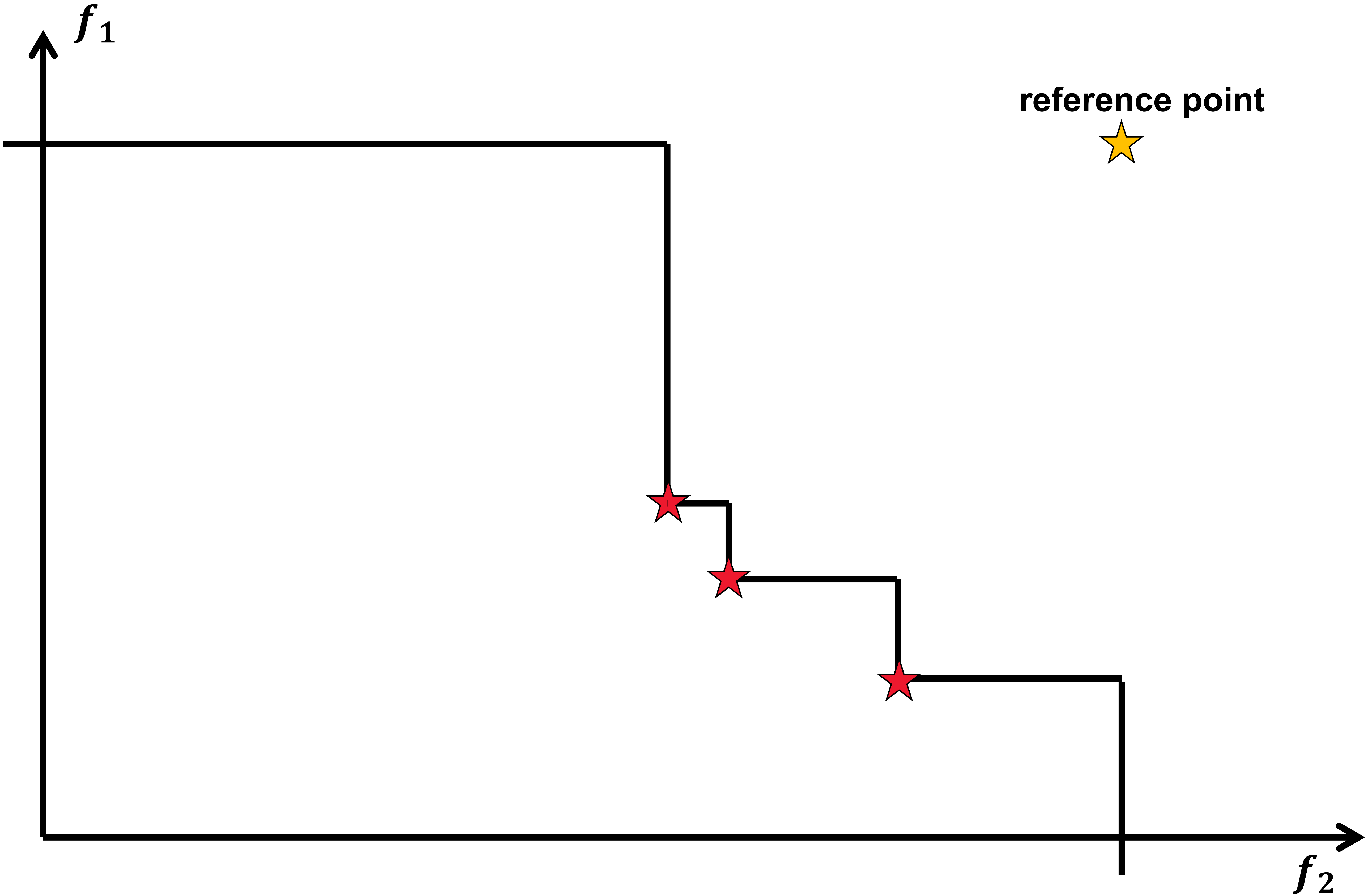
Unbounded UHVI optimization



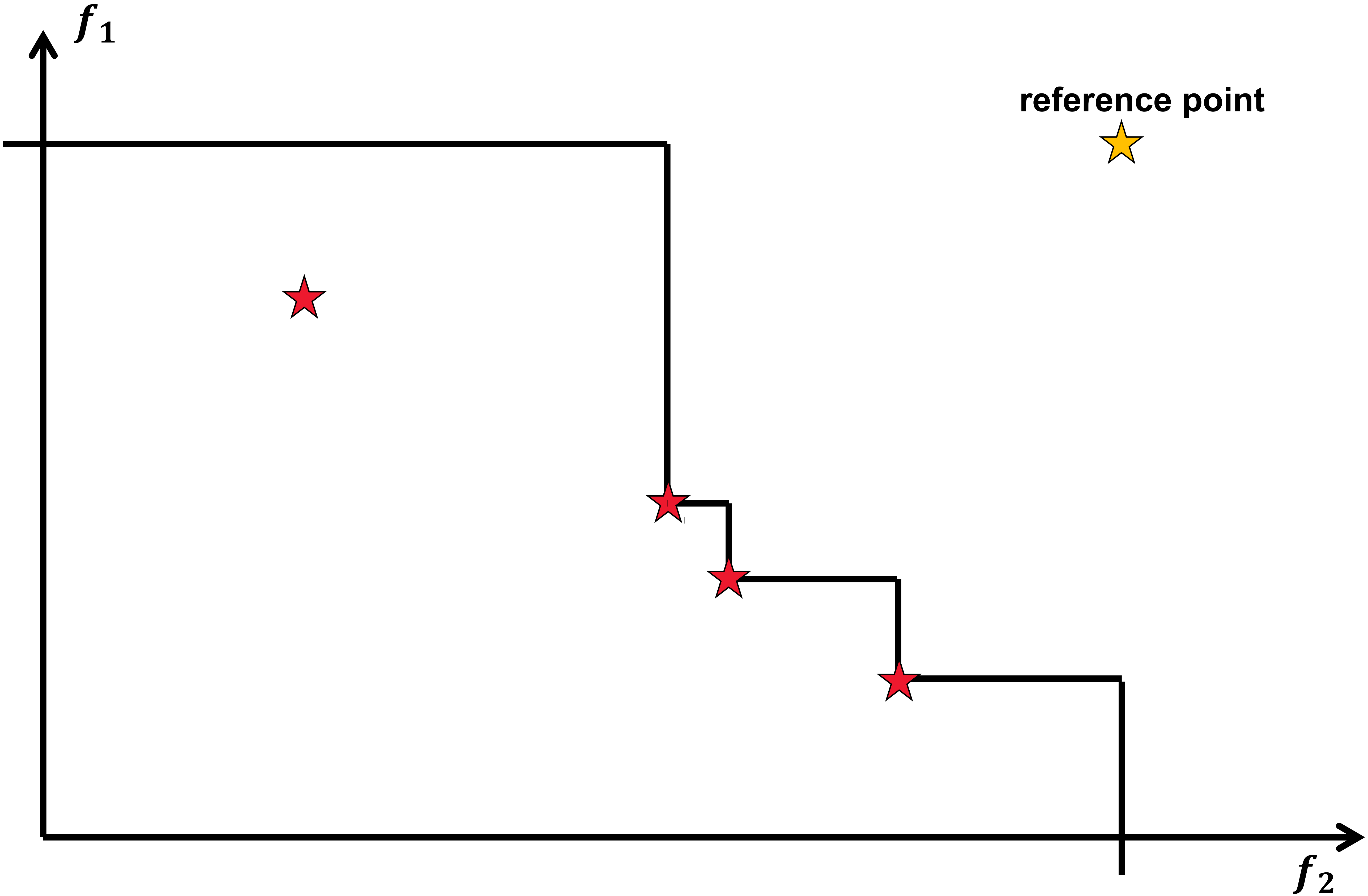
Unbounded UHVI optimization



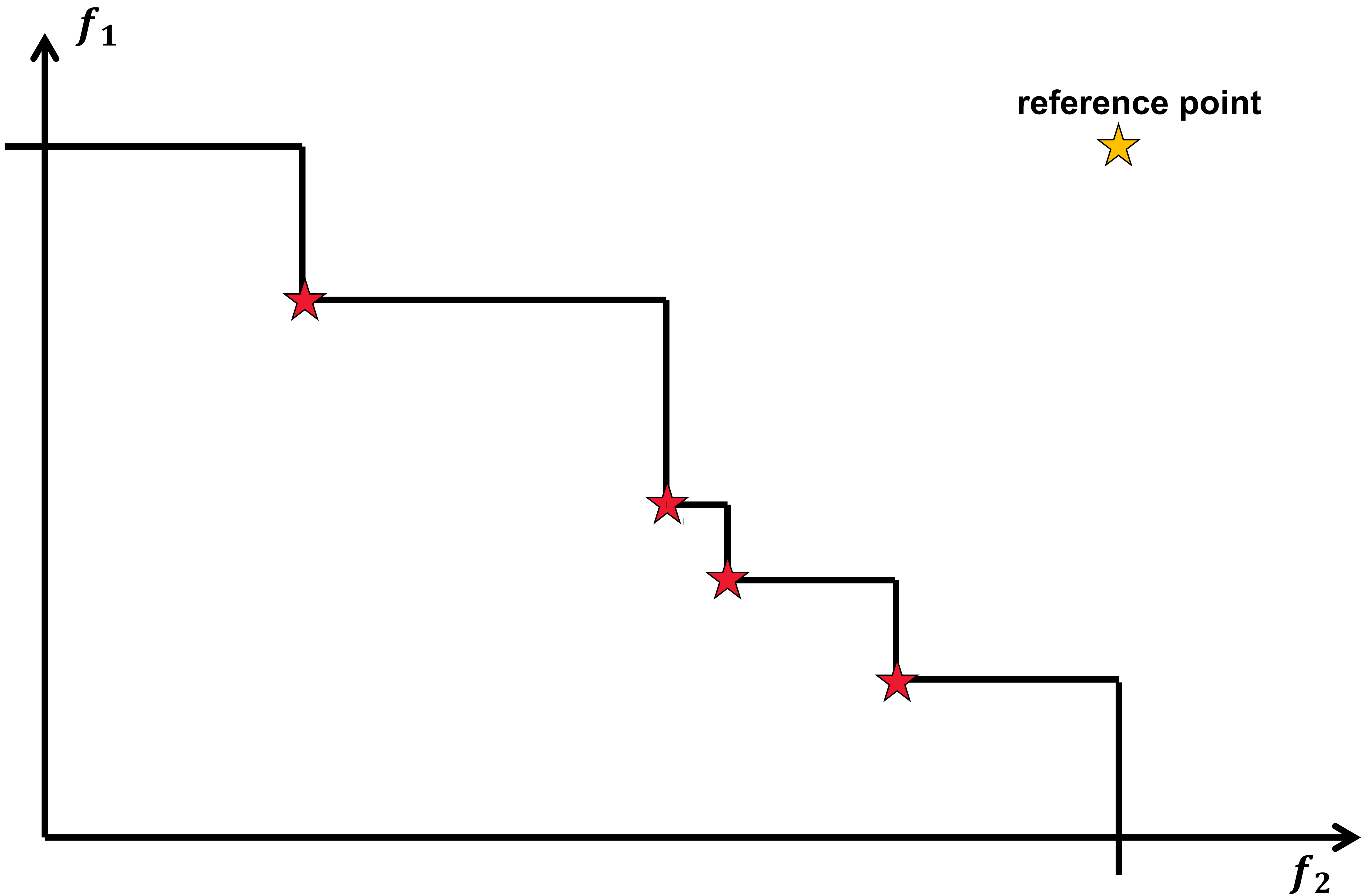
Unbounded UHVI optimization



Unbounded UHVI optimization



Unbounded UHVI optimization



Unbounded UHVI optimization

Idea:

- start with $S = \emptyset$
- While not happy:
  - use single-objective solver to optimize $UHVI_{\mathbf{r}}(x, S)$
  - $S \; = S \cup$ incumbent of last run when stopped

Unbounded UHVI optimization

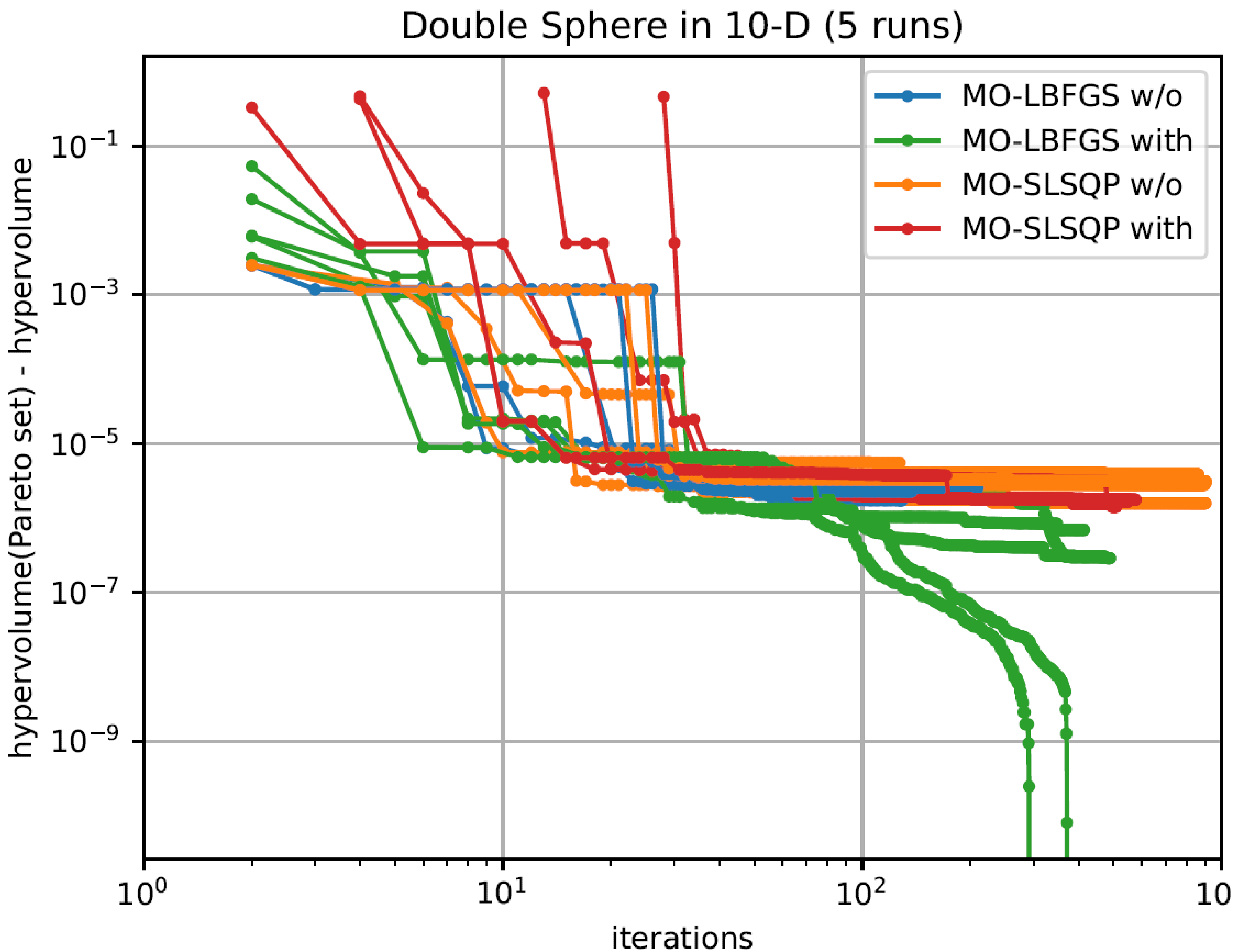
Idea:

- start with $S = \emptyset$
- While not happy:
  - use single-objective solver to optimize $UHVI_{\mathbf{r}}(x, S)$
  - $S \ = S \cup$ incumbent of last run when stopped

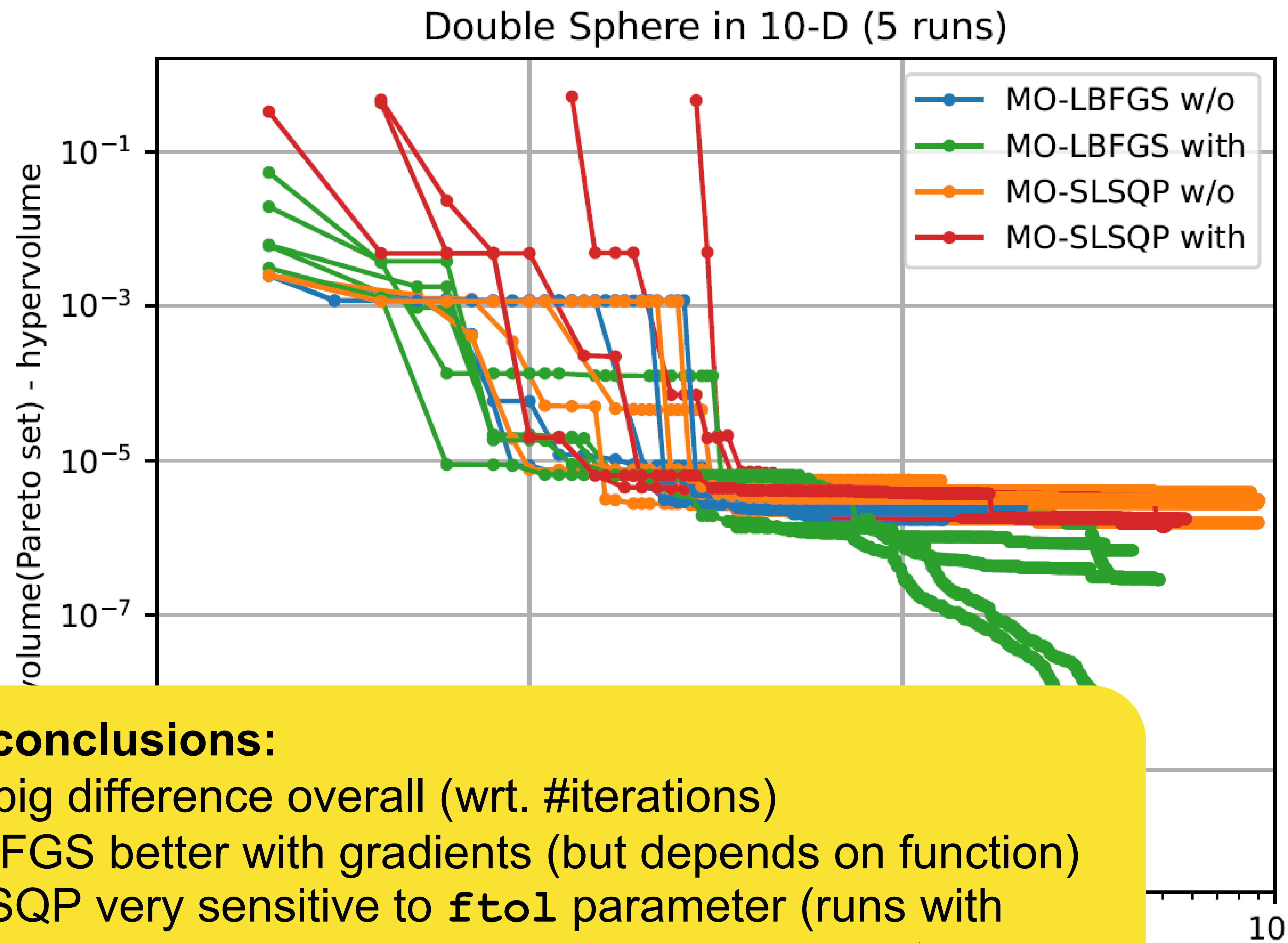
Note: the solver can be initialized fully random or at the previously found solution with largest hypervolume contribution

we typically choose the strategy that used less function evaluations in the past

Gradient-Based vs. Derivative-Free



Gradient-Based vs. Derivative-Free



main conclusions:

- no big difference overall (wrt. #iterations)
- L-BFGS better with gradients (but depends on function)
- SLSQP very sensitive to `ftol` parameter (runs with gradient are not stopping early enough if $< 10^{-1}$)

Commercial Break ;-)

COCO (Comparing Continous Optimizers)

`http://numbbo.github.io/coco/`

- allows for an (almost) automated benchmarking of solvers

Commercial Break ;-)

COCO (Comparing Continous Optimizers)

`http://numbbo.github.io/coco/`

- allows for an (almost) automated benchmarking of solvers
  - bbob (24 noiseless, unconstrained functions, $n = 2,3,5,10,20,40$)
  - bbob-noisy (30 functions with 3 noise types)
  - bbob-biobj (55 bi-objective functions)
  - bbob-largescale (like bbob but with $n \in \{20, 40, \ldots, 640\}$)
  - bbob-constrained (48 functions with $1 \ldots 9 + \lfloor 9n/2 \rfloor$ constraints, of which $1 \ldots 6 + 3n$ are active)
  - bbob-mixint & bbob-biobj-mixint (versions with 20% continuous variables, the rest discrete with varying arity from 2 to 16)
  - sbox-cost (bound-constrained version of bbob)

Commercial Break ;-)

COCO (Comparing Continous Optimizers)

http://numbbo.github.io/coco/

- allows for an (almost) automated benchmarking of solvers
  - bbob (24 noiseless, unconstrained functions, $n = 2,3,5,10,20,40$)
  - bbob-noisy (30 functions with 3 noise types)
  - bbob-biobj (55 bi-objective functions)
  - bbob-largescale (like bbob but with $n \in \{20, 40, \ldots, 640\}$)
  - bbob-constrained (48 functions with $1 \ldots 9 + \lfloor 9n/2 \rfloor$ constraints, of which $1 \ldots 6 + 3n$ are active)
  - bbob-mixint & bbob-biobj-mixint (versions with 20% continuous variables, the rest discrete with varying arity from 2 to 16)
  - sbox-cost (bound-constrained version of bbob)
- all in (predefined) 6 dimensions (in principle, any)

Commercial Break ;-)

COCO (Comparing Continous Optimizers)

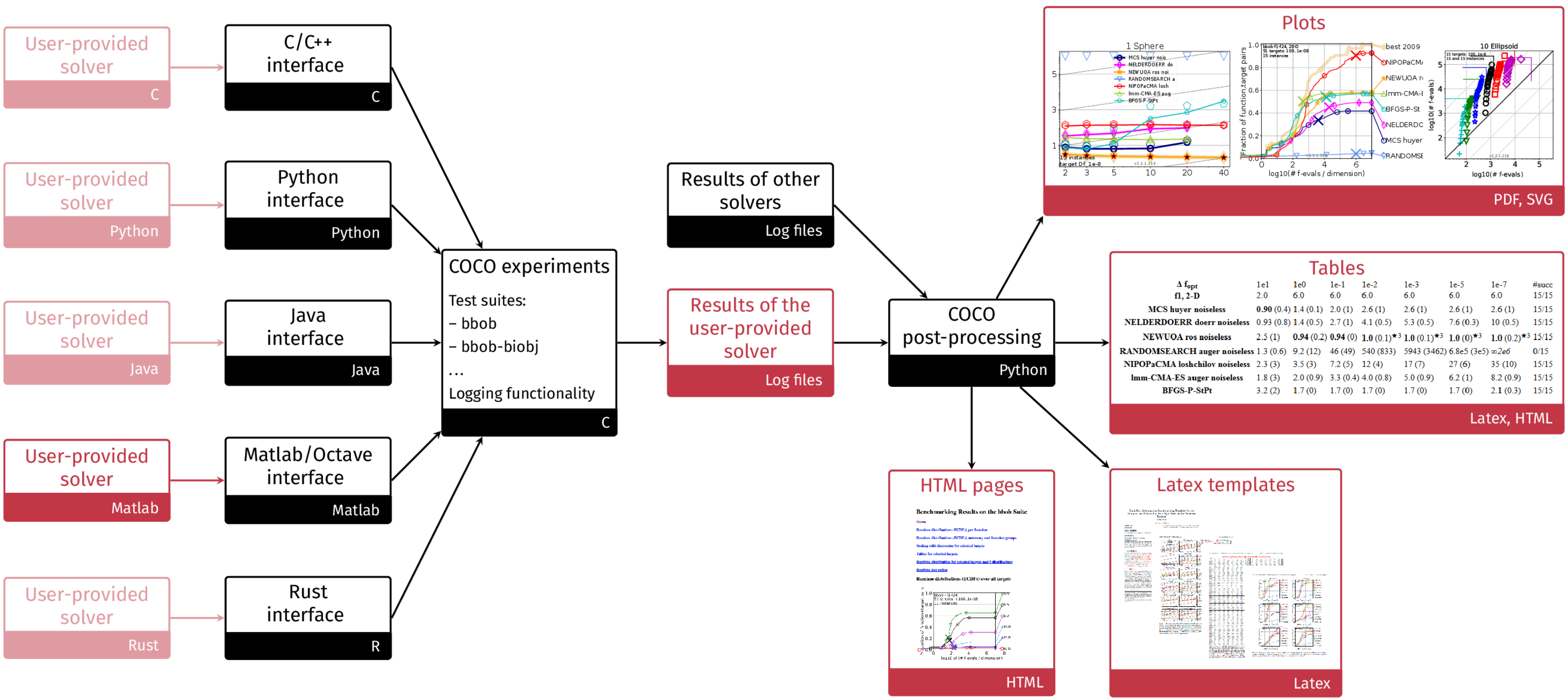
http://numbbo.github.io/coco/

- allows for an (almost) automated benchmarking of solvers
  - bbob (24 noiseless, unconstrained functions, $n = 2,3,5,10,20,40$)
  - bbob-noisy (30 functions with 3 noise types)
  - bbob-biobj (55 bi-objective functions)
  - bbob-largescale (like bbob but with $n \in \{20, 40, \dots, 640\}$)
  - bbob-constrained (48 functions with $1 \dots 9 + \lfloor 9n/2 \rfloor$ constraints, of which $1 \dots 6 + 3n$ are active)
  - bbob-mixint & bbob-biobj-mixint (versions with 20% continuous variables, the rest discrete with varying arity from 2 to 16)
  - sbox-cost (bound-constrained version of bbob)
- all in (predefined) 6 dimensions (in principle, any)
- available in Python, C/C++, Java, Matlab/Octave, Rust

Commercial Break ;-)

COCO (Comparing Continous Optimizers)

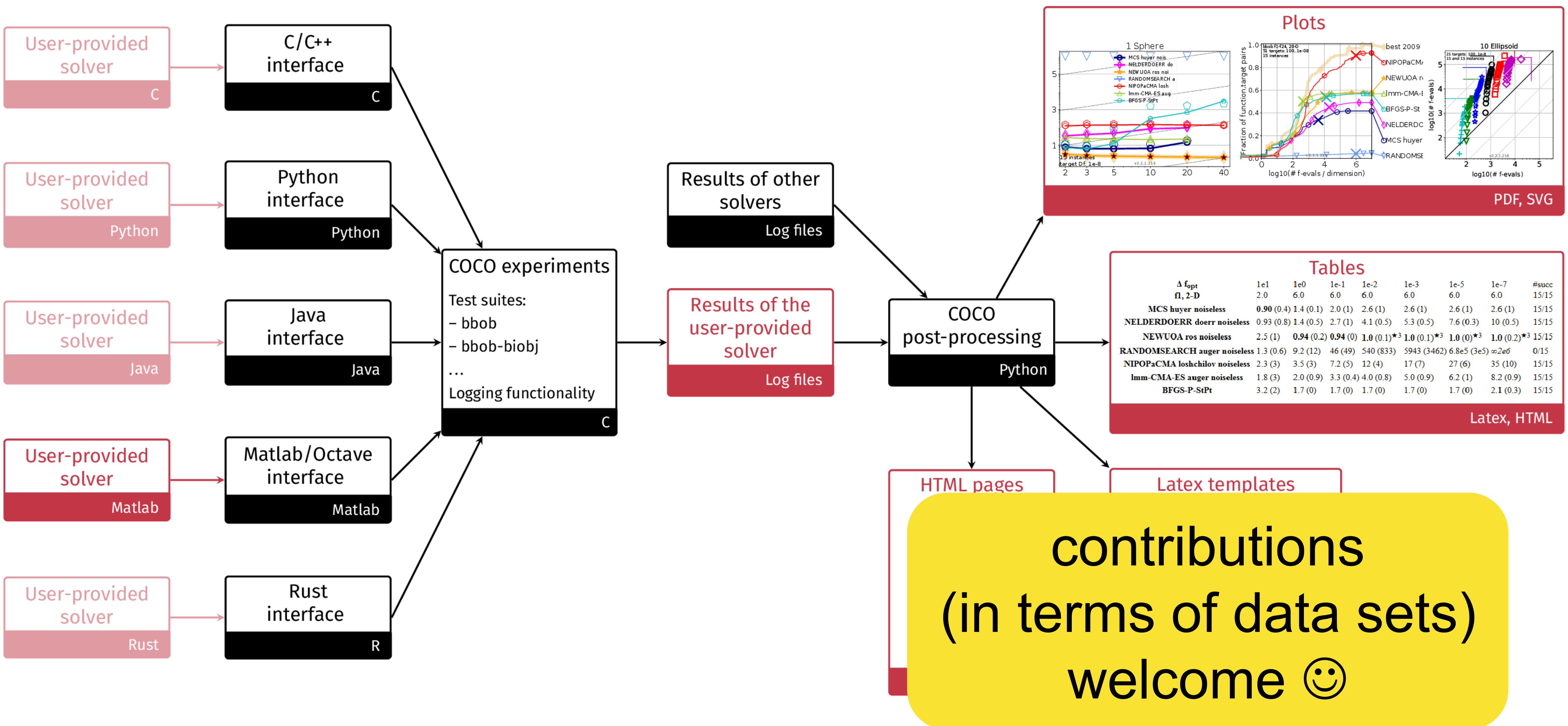
http://numbbo.github.io/coco/



Commercial Break ;-)

COCO (Comparing Continous Optimizers)

`http://numbbo.github.io/coco/`



Unbounded UHVI optimization: Results

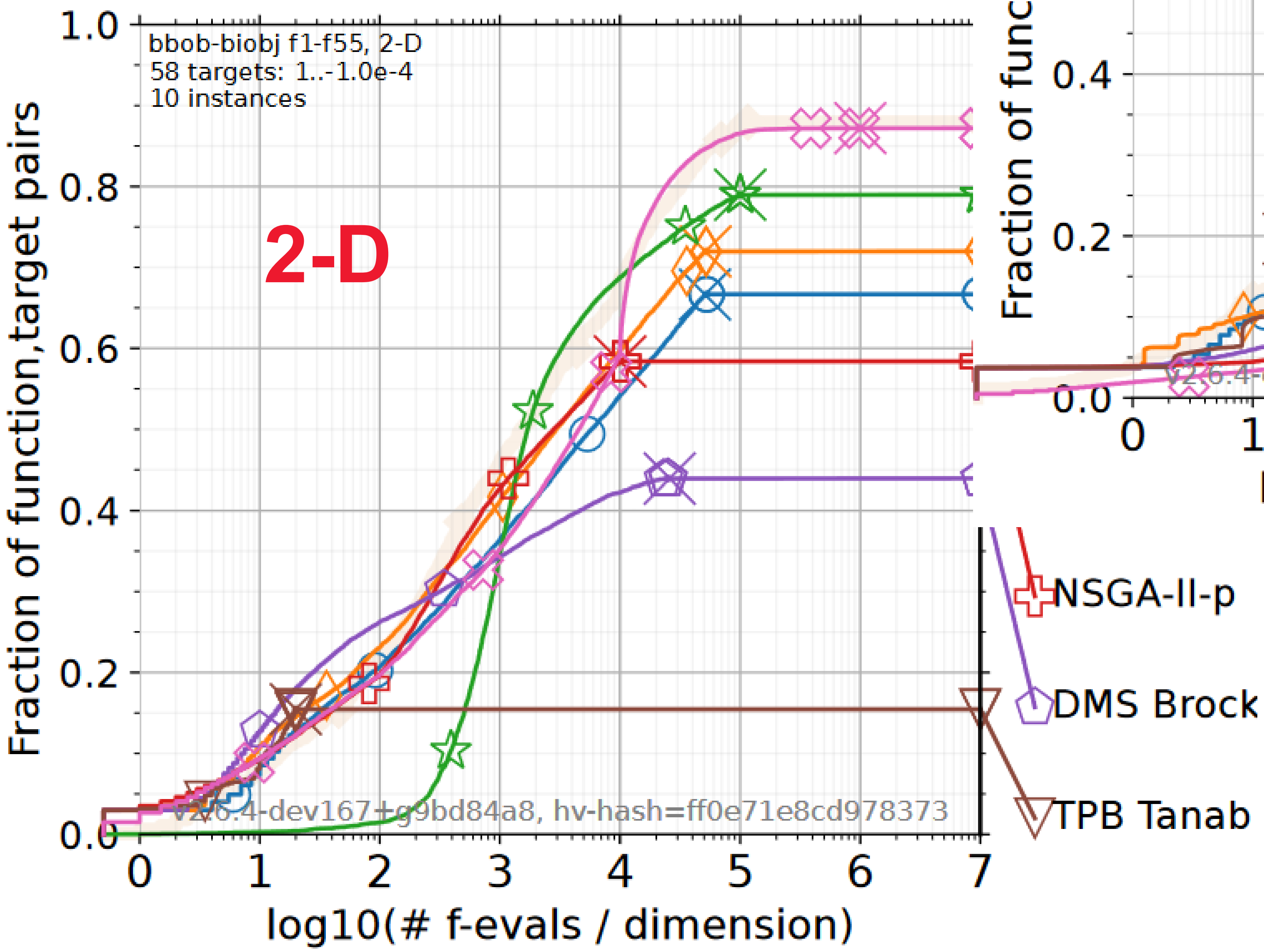
```
pip install cocopp

python -m cocopp MO-BFGS/ MO-SLSQP/
```

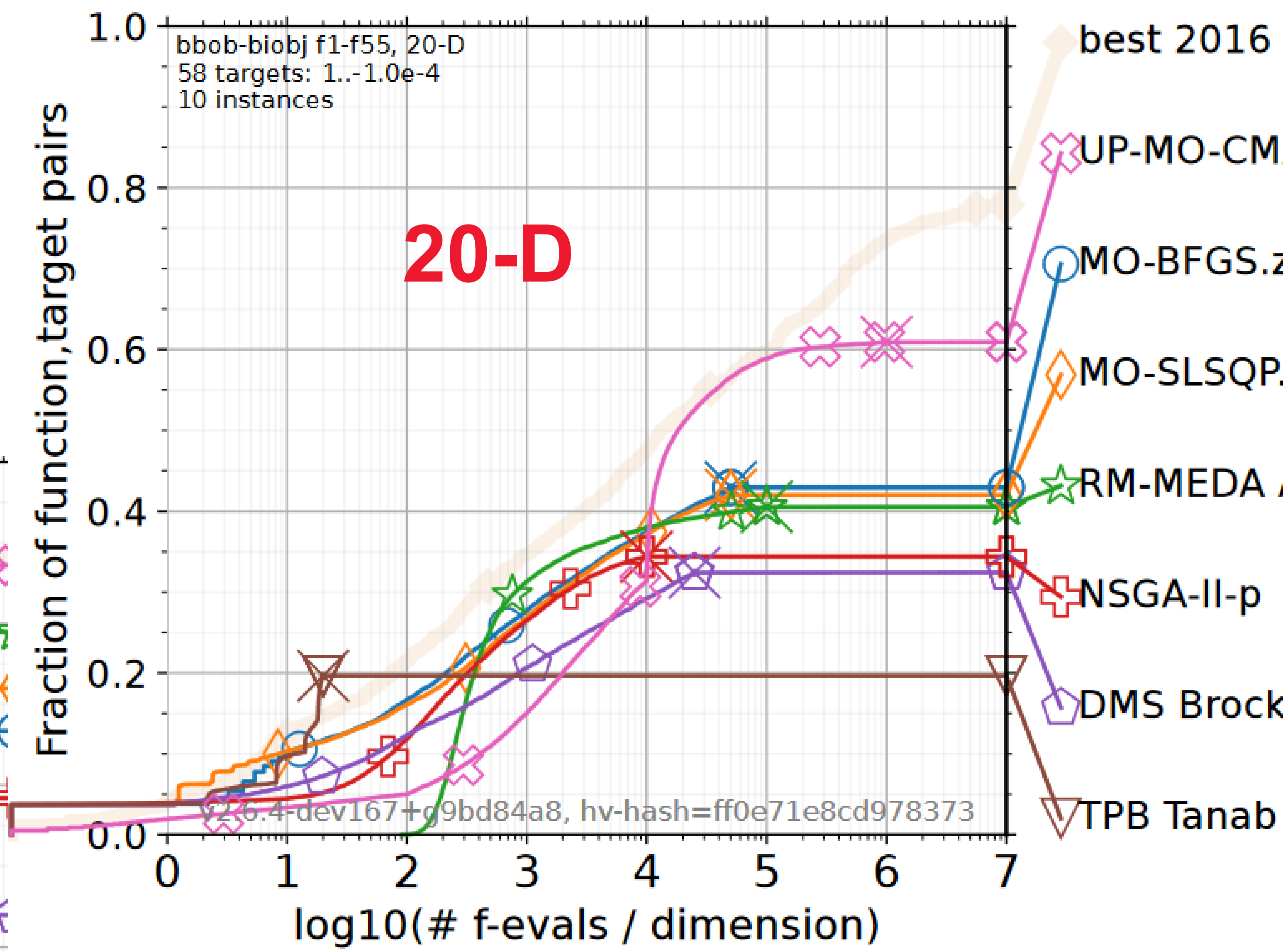
Unbounded UHVI optimization: Results

```
pip install cocopp
```

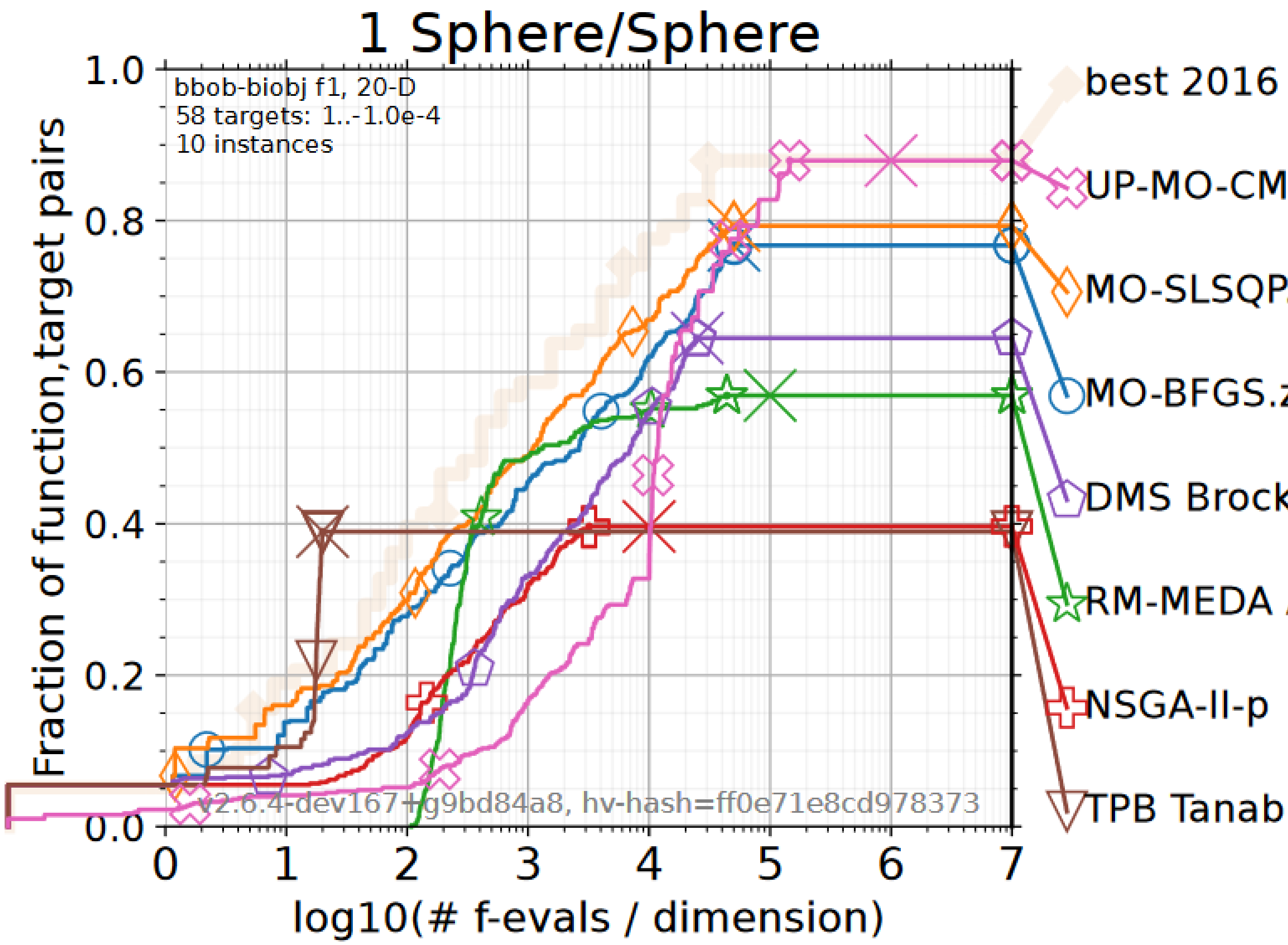
```
python -m cocopp MO-BFGS/ MO-SLSQP/ RM-MEDA NSGA-
II-platypus DMS TPB_Tanabe UP-MO-CMA
```

Unbounded UHVI optimization: Results

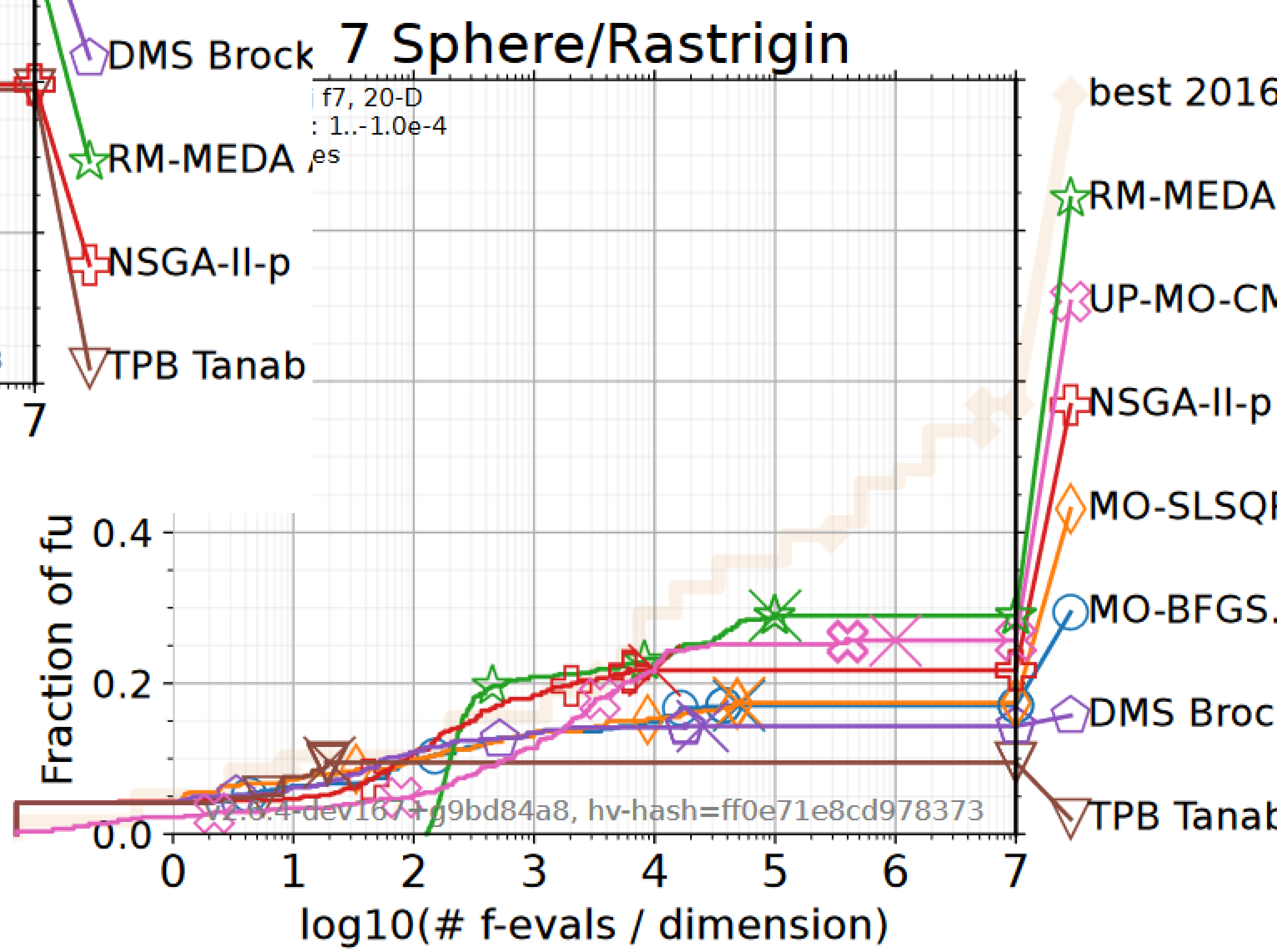




Unbounded UHVI optimization: Results



working well when Pareto set is connected



slow on multimodal functions

Commercial Break II

International Conference on Evolutionary Multi-Criterion Optimization

4-7 March, 2025
Canberra, Australia

EMO '25 CANBERRA

About

EMO is a biennial conference series devoted to studying various aspects of multiobjective optimization such as:

- evolutionary and population-based algorithms;
- surrogate-based or Bayesian optimization methods;
- machine learning/artificial intelligence-based hybrid methods;
- theoretical foundations of multiobjective optimization;
- multiple-criteria decision-making;
- benchmarking, performance indicators, and visualization;
- Practical/industry applications.

Plenary Speakers

- **Professor Bernhard Sendhoff**, CEO of Global Network Honda Research Institutes, Germany
- **Professor Qingfu Zhang**, Chair Professor of Computational Intelligence, City University of Hong Kong
- **Professor Kaisa Miettinen**, Professor of Industrial Optimization, University of Jyvaskyla, Finland

EMO 2025 will be held at the Shine Dome, Canberra, Australia (https://shinedome.org.au/).

Submission types

- **EMO Main track:** Full papers (12 pages)
- **MCDM Special track:** Full papers or abstracts (2 pages)
- **Industry Special track:** Full papers or abstracts

All accepted papers will be included in Springer Lecture Notes in Computer Science (LNCS) proceedings.

Key dates

- Paper submission deadline: 15 September 2024
- Notification of outcomes: 15 November 2024
- Camera-ready papers due: 30 November 2024
- Conference dates: 4-7 March 2025

Previous EMO conference proceedings:
http://www.emo-online.org/

Submissions open!

More information: https://emo2025.org/

Organizing committee

General chairs:
- Hemant Kumar Singh (UNSW, Australia)
- Tapabrata Ray (UNSW, Australia)
- Joshua Knowles (SLB Cambridge, UK)

Industry track chairs:
- Frank Neumann (University of Adelaide, Australia)
- Kate-Smith Miles (University of Melbourne, Australia)

Local arrangement chairs:
- Saber Elsayed (UNSW, Australia)
- Ali Ahrari (UNSW, Australia)

Conflict of Interest chairs:
- Hisao Ishibuchi (SUSTech, China)
- Lie Meng Pang (SUSTech, China)

Proceedings chairs:
- Bing Wang (UNSW Australia)
- Akira Oyama (JAXA, Japan)

Program chairs:
- Xiaodong Li, (RMIT, Australia)
- Juergen Branke (University of Warwick, UK)

MCDM track chairs:
- Ruhul Sarker (UNSW, Australia)
- Sanaz Mostaghim (OvGU, Germany)

Tutorial chair:
- Hussein Abbass (UNSW, Australia)

Web chair:
- Angus Kenny (UNSW, Australia)

Publicity chairs:
- Aneta Neumann (University of Adelaide, Australia)
- Yi Mei (VUW, New Zealand)
- Deepak Sharma (IITG, India)
- Handing Wang (Xidian Uni, China)
- Dimo Brockhoff (Inria Saclay, France)
- Carlos Coello-Coello (CINVESTAV-IPN, Mexico)

Springer

Multidisciplinary Design Optimization Lab
www.mdolab.net

Conclusions

How to do multiobj. optimization with single-obj. solvers?

- when an unbounded representation of the Pareto set is sought via iterative UHVI optimization
- start with one solution, then increase the approximation size
- experiments with solvers L-BFGS and SLSQP from `scipy`
- basic version works well on simple functions but problems still with multimodality
  - idea: add another stopping criterion, specific for the case of UHVI optimization

Conclusions

How to do multiobj. optimization with single-obj. solvers?

- when an unbounded representation of the Pareto set is sought via iterative UHVI optimization
- start with one solution, then increase the approximation size
- experiments with solvers L-BFGS and SLSQP from `scipy`
- basic version works well on simple functions but problems still with multimodality
  - idea: add another stopping criterion, specific for the case of UHVI optimization

Questions?