Global Optimization of continuous MinMax problem

Dominique Monnet, Jordan Ninin, Benoît Clément
LAB-STICC, UMR 6285 / ENSTA-Bretagne

Min Max problems

Min Max problems appear in:

- Robust control
- Game theory
- Risk management
- Every problem involving uncertainty

Plan

What is control?

- Dynamic system (Robot, Missile, Dam, Washing machine...).

$$\boxed{G}$$

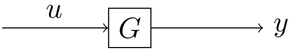
What is control?

- Dynamic system (Robot, Missile, Dam, Washing machine...).
- Actuators (Motor, Steering wheel, Flap, ...).

$$\xrightarrow{\quad u \quad} \boxed{G}$$

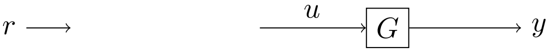
What is control?

- Dynamic system (Robot, Missile, Dam, Washing machine...).
- Actuators (Motor, Steering wheel, Flap, ...).
- Sensors (INS, Sonar, Temperature/Pressure sensor, ...).



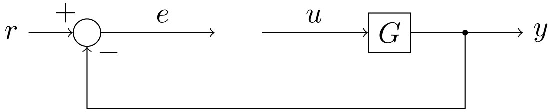
What is control?

- Dynamic system (Robot, Missile, Dam, Washing machine...).
- Actuators (Motor, Steering wheel, Flap, ...).
- Sensors (INS, Sonar, Temperature/Pressure sensor, ...).
- Reference to follow.



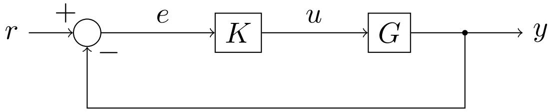
What is control?

- Dynamic system (Robot, Missile, Dam, Washing machine...).
- Actuators (Motor, Steering wheel, Flap, ...).
- Sensors (INS, Sonar, Temperature/Pressure sensor, ...).
- Reference to follow.



What is control?

- Dynamic system (Robot, Missile, Dam, Washing machine...).
- Actuators (Motor, Steering wheel, Flap, ...).
- Sensors (INS, Sonar, Temperature/Pressure sensor, ...).
- Reference to follow.
- Controller to close the loop.



Frequency constraint

Frequency constraint on $e(i\omega)$: we want $|\frac{e(i\omega)}{r(i\omega)}| = |T_{r\to e}(K, i\omega)|$ to be small

$$\forall \omega \geq 0,\ |T_{r\to e}(K, i\omega)| \leq |W(i\omega)| \iff \sup_{\omega}(|T_{r\to e}(K, i\omega)W^{-1}(i\omega)|) \leq 1$$

Min max problem formulation

Stability constraint:

- The closed loop system is stable $\iff R(K) \leq 0$ (Routh criterion).
- $R(K) \leq 0$ is a non-convex rational system.

Problem formulation

$$\begin{cases} \min_{K} \sup_{\omega} |T_{r\to e}(K, i\omega)W^{-1}(i\omega)|, \\ s.t. \quad R(K) \leq 0 \end{cases}$$

We want:

- an enclosure of the minimum.
- reliable computation.

$\rightarrow$ Interval Based Branch and Bound Algorithm

Plan

Min max problem formulation

We search $x^* \in \mathcal{X}$ such that $\sup_{\boldsymbol{y} \in \mathcal{Y}} f(x^*, y)$ is minimal.

Constrained Min max problem

$$\begin{cases} \min_{x \in \mathcal{X}} \sup_{y \in \mathcal{Y}} f(x, y), \\ \\ s.t. \quad C_x(x) \leq 0 \\ \qquad\;\; C_{xy}(x, y) \leq 0 \end{cases}$$

- $\mathcal{X}$ and $\mathcal{Y}$ are bounded.
- f, $\mathcal{C}_x$ and $\mathcal{C}_{xy}$ can be evaluated with interval computation.

Main Branch and bound algorithm: minimization

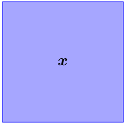
Interval Based Branch and Bound Algorithm

Init: push $\mathcal{X}$ in $\mathcal{L}$

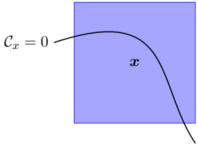
1. Choose a box $\boldsymbol{x}$ from $\mathcal{L}$.
2. Contract $\boldsymbol{x}$ w.r.t $\mathcal{C}_x(\boldsymbol{x}) \leq 0$ using CSP techniques.
3. Compute $[lb_{\boldsymbol{x}}, ub_{\boldsymbol{x}}]$ an enclosure of $\sup_{y \in \mathcal{Y}} \mathbf{f}(\boldsymbol{x}, y)$.
4. Try to find a good feasible solution in $\boldsymbol{x}$.
5. Update best current solution.
6. Bisect $\boldsymbol{x}$ into $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, push $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in $\mathcal{L}$.

Stop criterion: $width([\min_{\boldsymbol{x} \in \mathcal{L}} lb_{\boldsymbol{x}}, \min_{\boldsymbol{x} \in \mathcal{L}, \mathcal{C}(\boldsymbol{x}) \leq 0} ub_{\boldsymbol{x}}]) \leq \epsilon$.

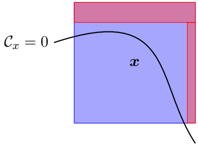
Main Branch and bound



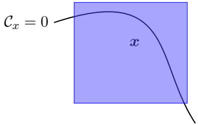
Main Branch and bound



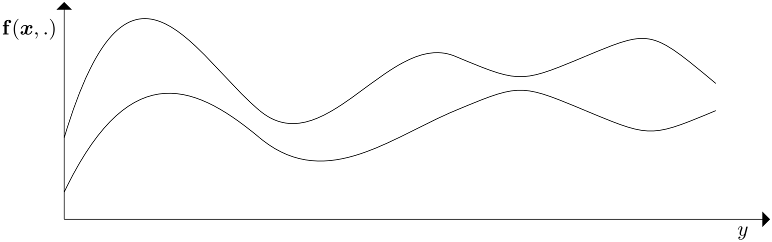
Main Branch and bound



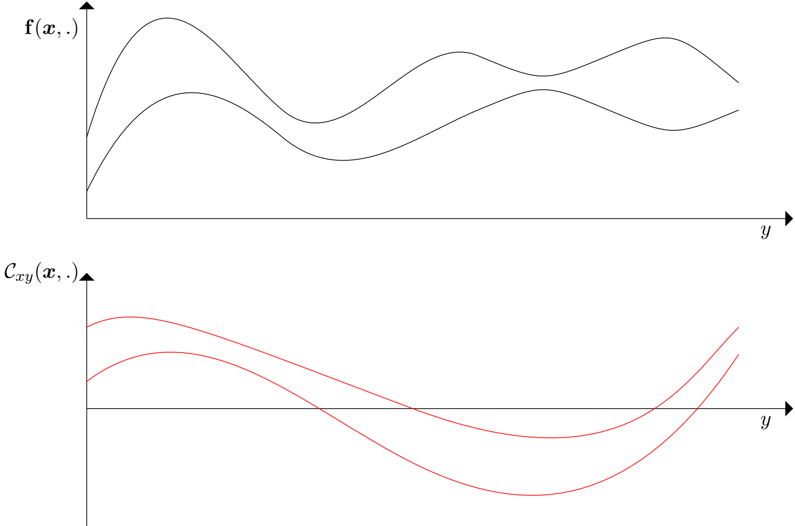
Main Branch and bound



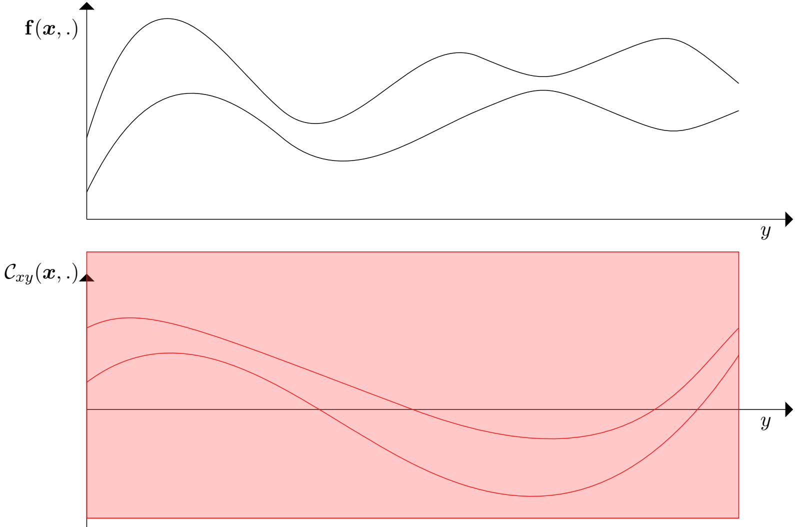
Secondary Branch and Bound algorithm: maximization



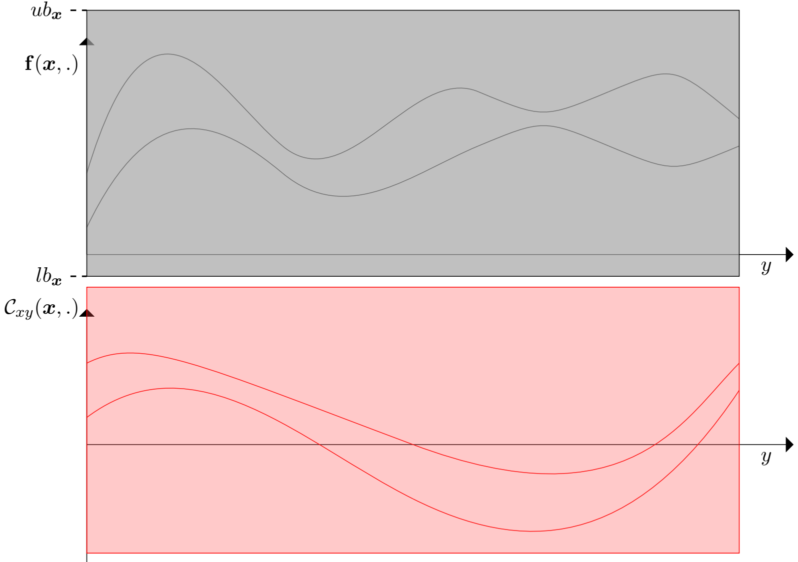
Secondary Branch and Bound algorithm: maximization



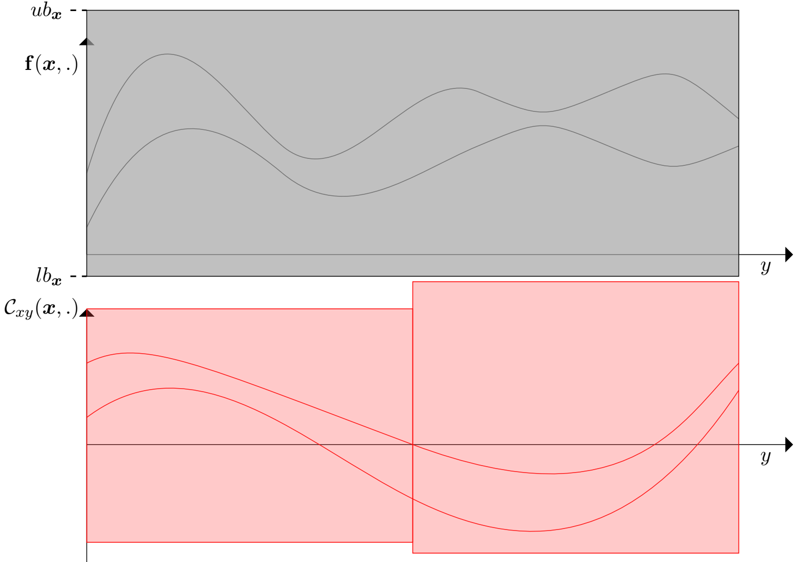
Secondary Branch and Bound algorithm: maximization



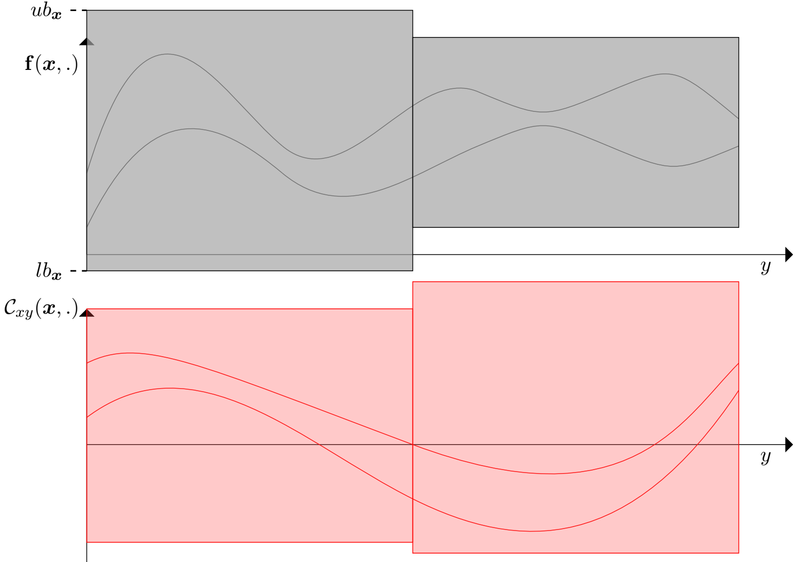
Secondary Branch and Bound algorithm: maximization



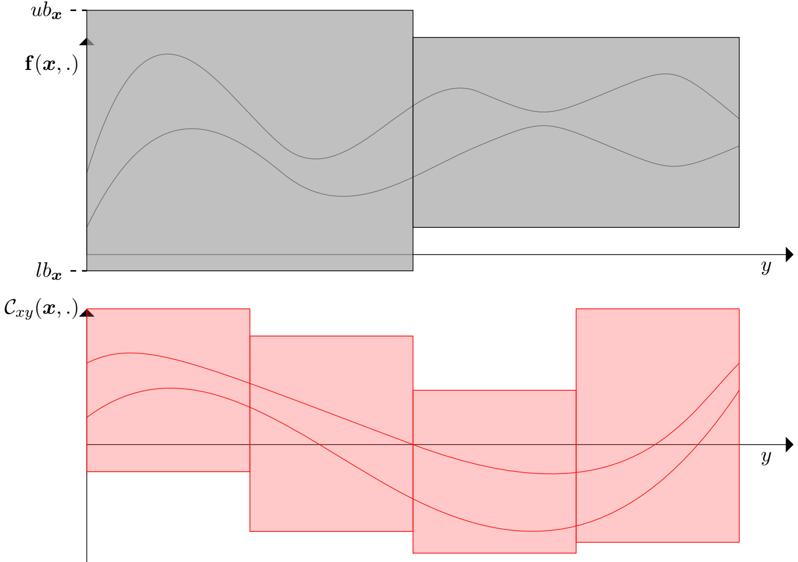
Secondary Branch and Bound algorithm: maximization



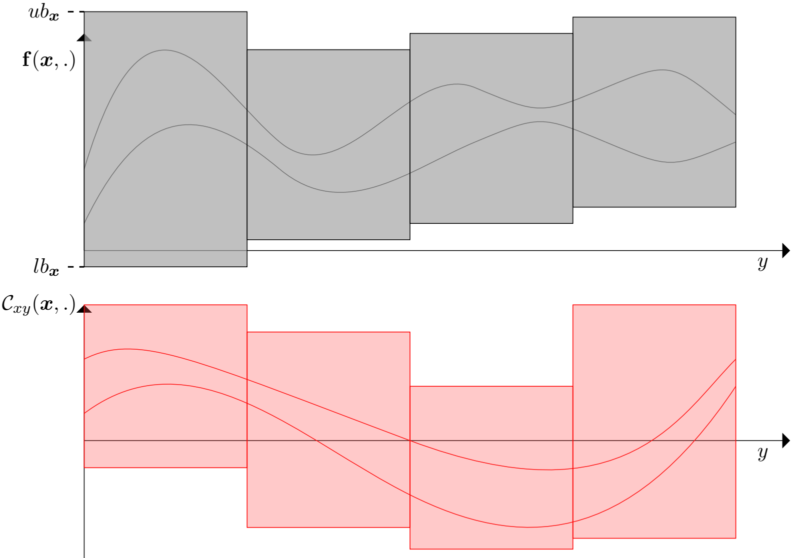
Secondary Branch and Bound algorithm: maximization



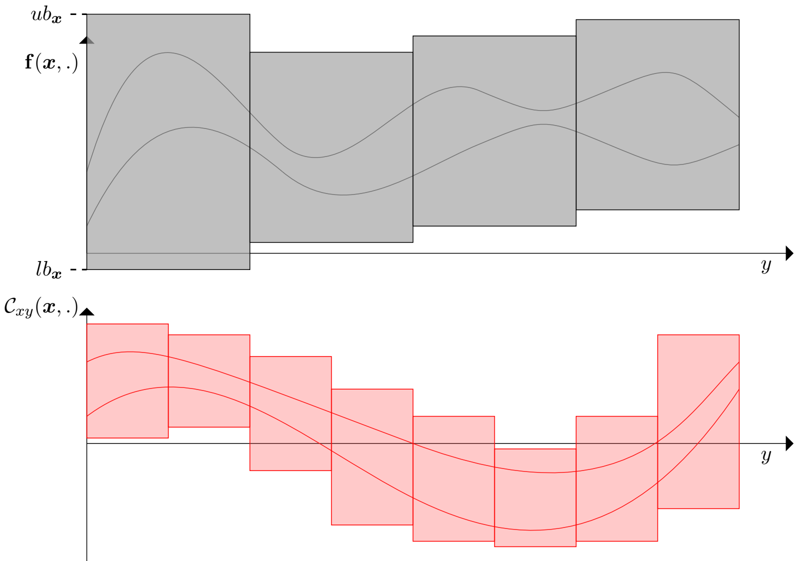
Secondary Branch and Bound algorithm: maximization



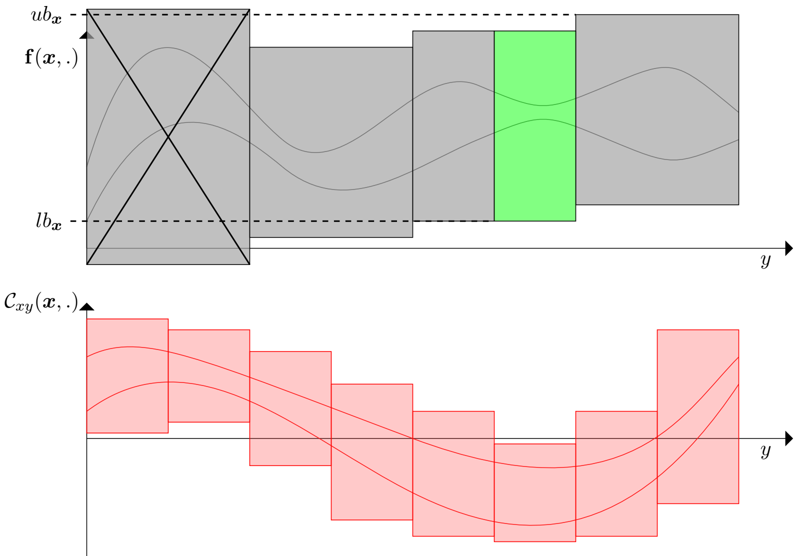
Secondary Branch and Bound algorithm: maximization



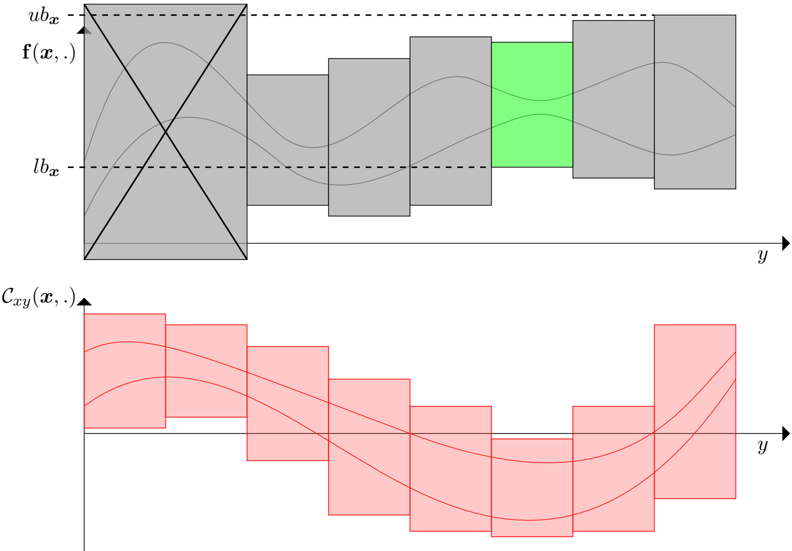
Secondary Branch and Bound algorithm: maximization



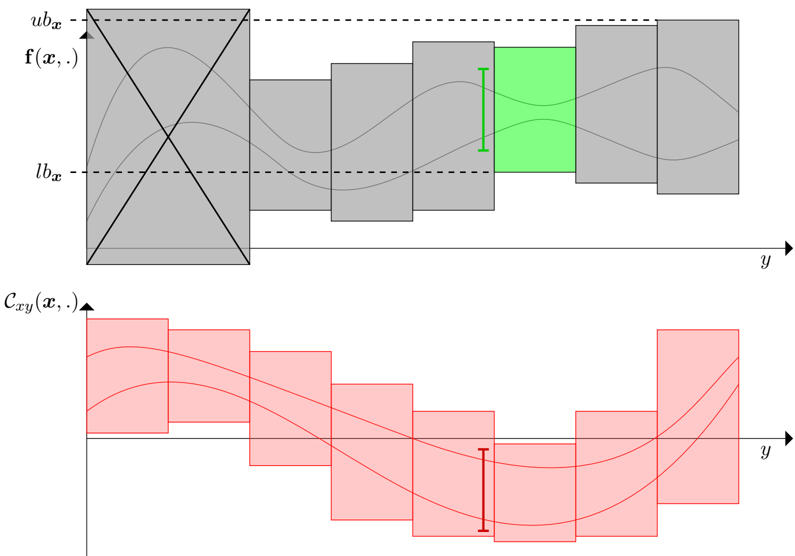
Secondary Branch and Bound algorithm: maximization



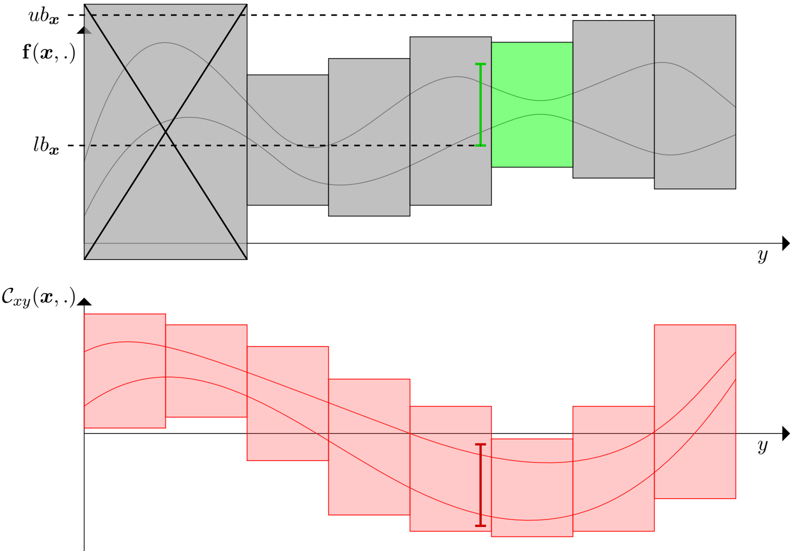
Secondary Branch and Bound algorithm: maximization



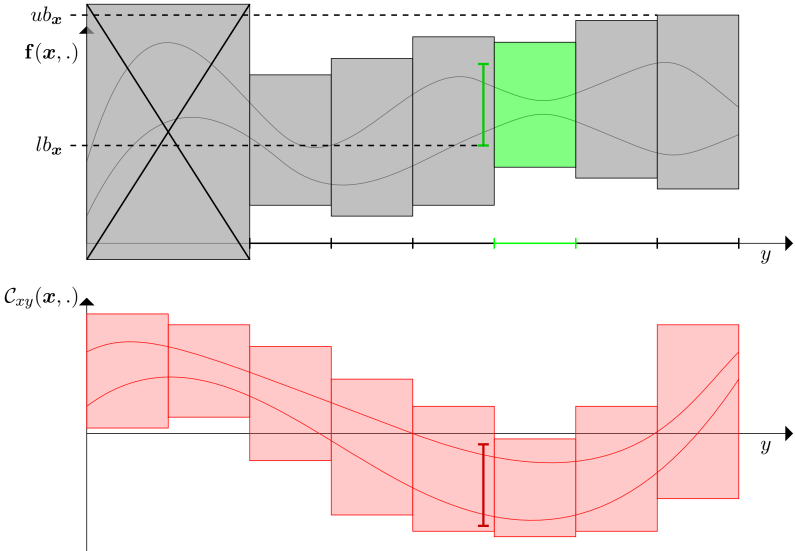
Secondary Branch and Bound algorithm: maximization



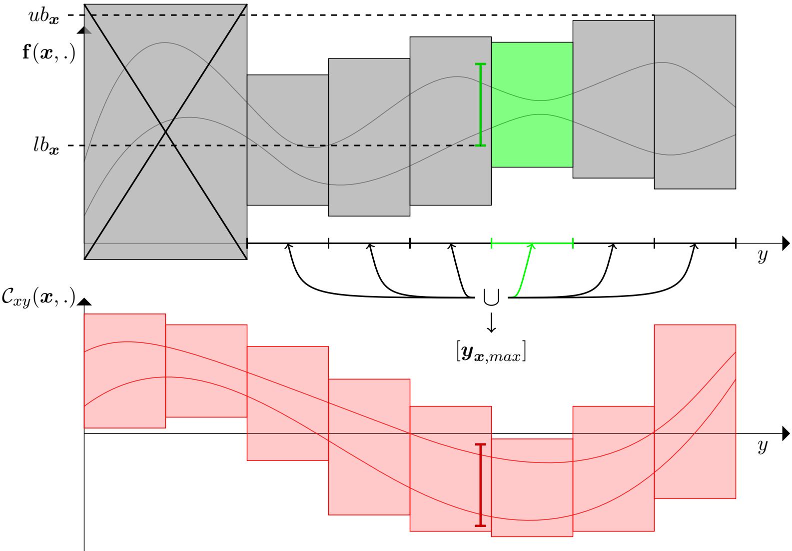
Secondary Branch and Bound algorithm: maximization



Secondary Branch and Bound algorithm: maximization



Secondary Branch and Bound algorithm: maximization



Inclusion properties

Let be $\boldsymbol{x} \subseteq \mathcal{X}$ and $\boldsymbol{y} \subseteq \mathcal{Y}$, we denote

$$\mathbf{f}_{max}(\boldsymbol{x}) = \{\sup_{y} f(x,y), x \in \boldsymbol{x}\}$$

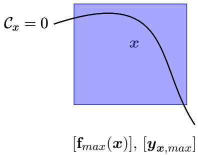
$$\boldsymbol{y}_{\boldsymbol{x},max} = \{y \in \mathcal{Y} | \exists x \in \boldsymbol{x},\ y \text{ maximizes } f(x,y)\}$$

Let be $\boldsymbol{x}_1 \subseteq \boldsymbol{x}$.

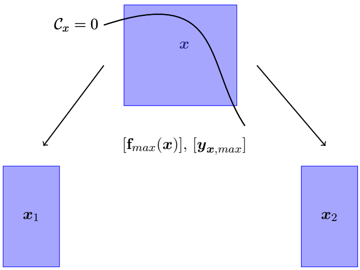
Proposition

- $\mathbf{f}_{max}(\boldsymbol{x}_1) \subseteq \mathbf{f}_{max}(\boldsymbol{x})$
- $\boldsymbol{y}_{\boldsymbol{x}_1,max} \subseteq \boldsymbol{y}_{\boldsymbol{x},max}$
- $\mathcal{C}_x(\boldsymbol{x}) \leq 0 \implies \mathcal{C}_x(\boldsymbol{x}_1) \leq 0$
- $\mathcal{C}_{xy}(\boldsymbol{x},\boldsymbol{y}) \leq 0 \implies \mathcal{C}_{xy}(\boldsymbol{x}_1,\boldsymbol{y}) \leq 0$

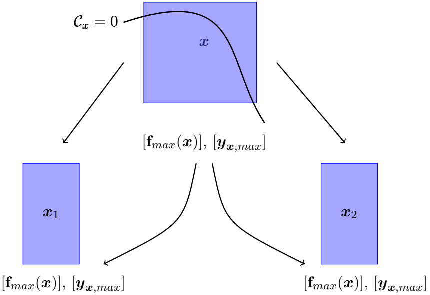
Main Branch and bound



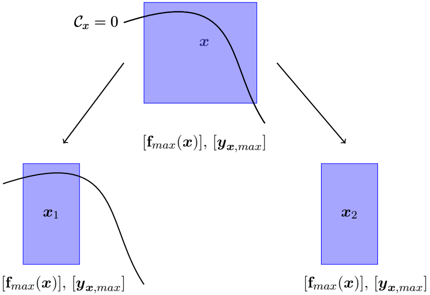
Main Branch and bound



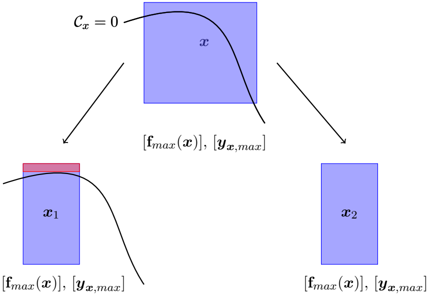
Main Branch and bound



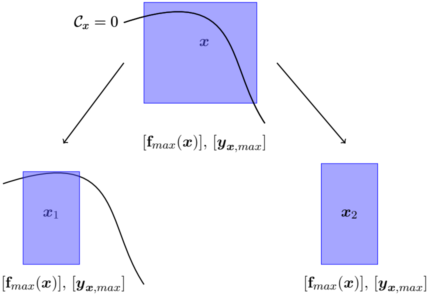
Main Branch and bound



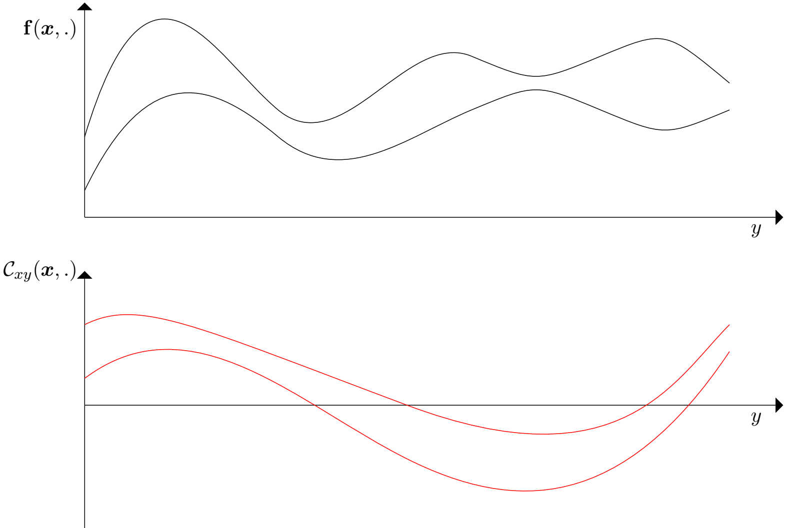
Main Branch and bound



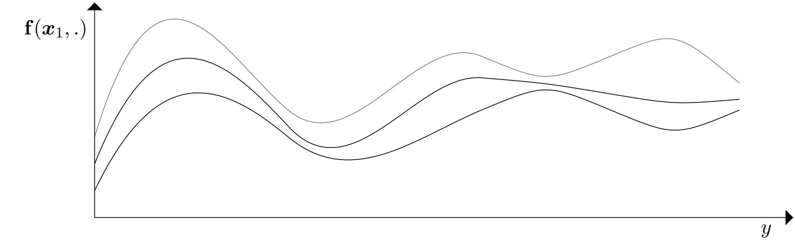
Main Branch and bound

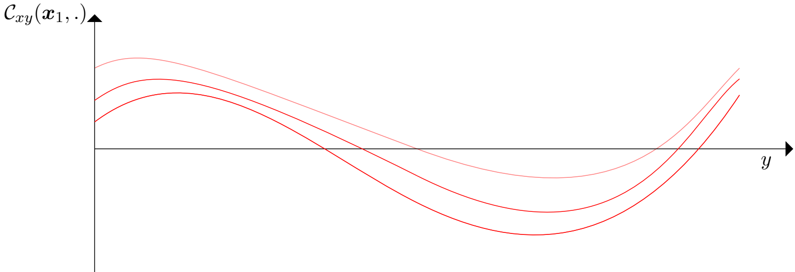


Main Branch and bound with Inclusion properties

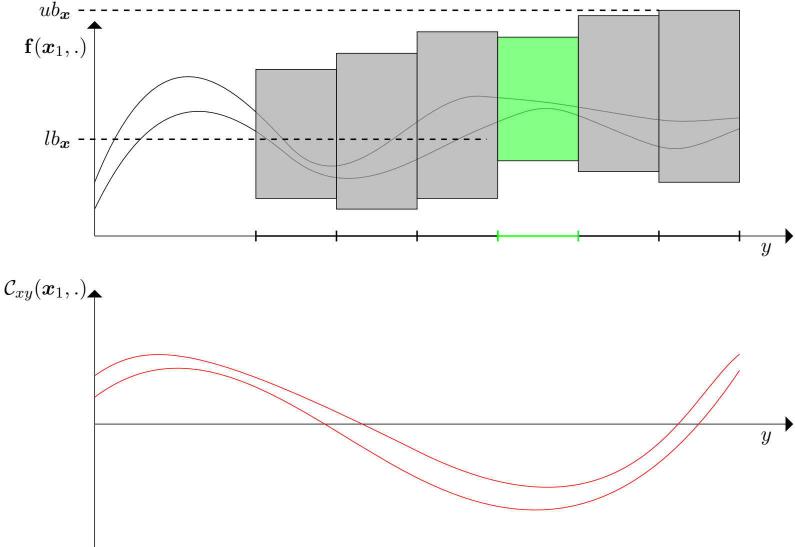


Main Branch and bound with Inclusion properties

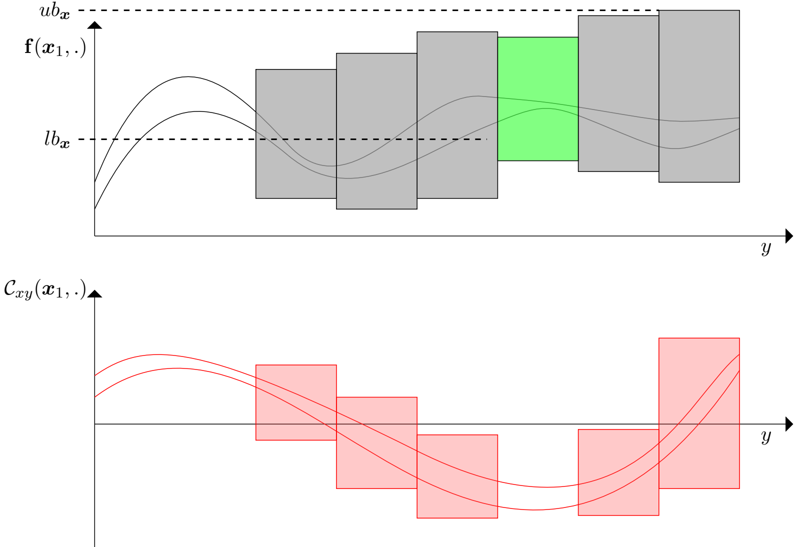




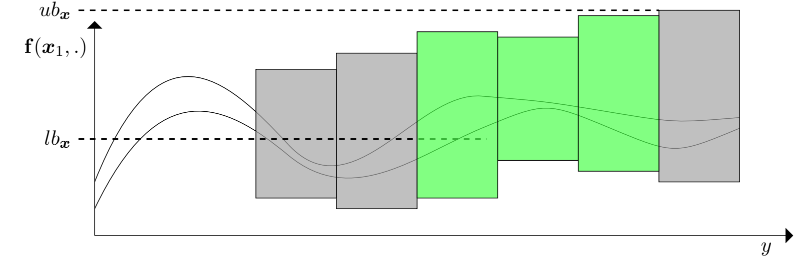
Main Branch and bound with Inclusion properties

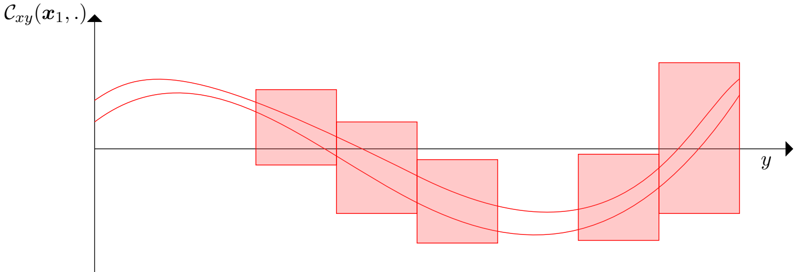


Main Branch and bound with Inclusion properties

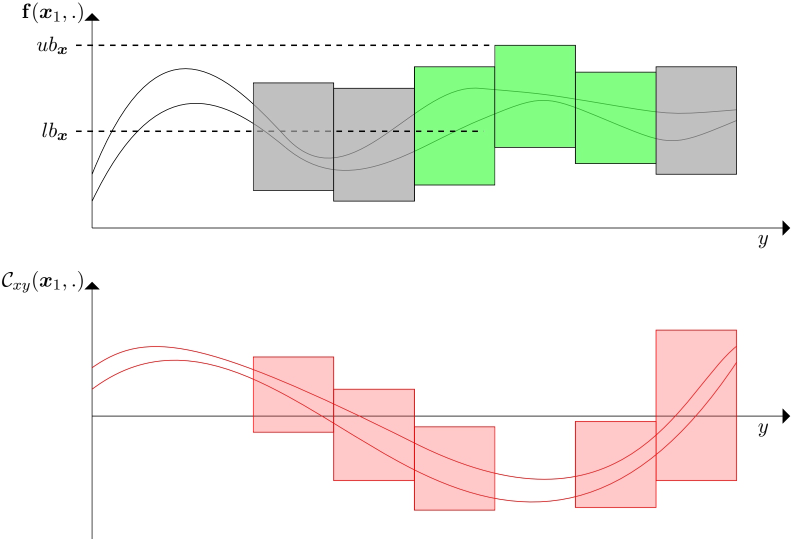


Main Branch and bound with Inclusion properties

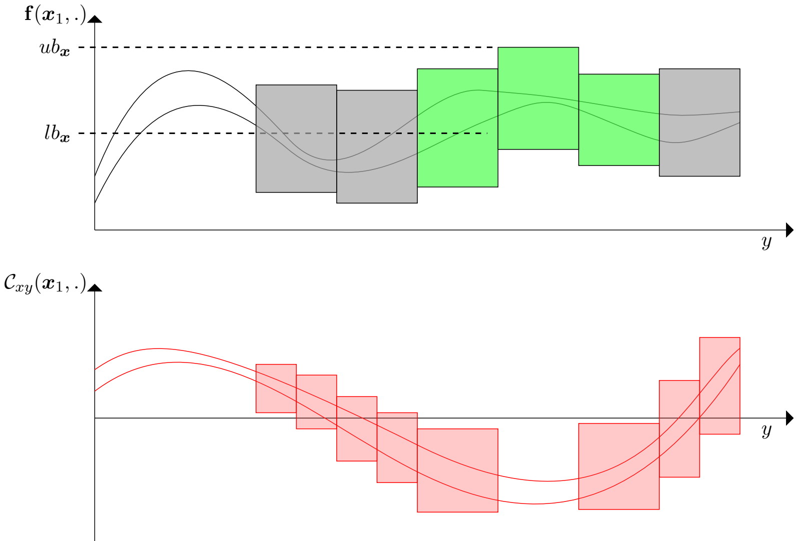




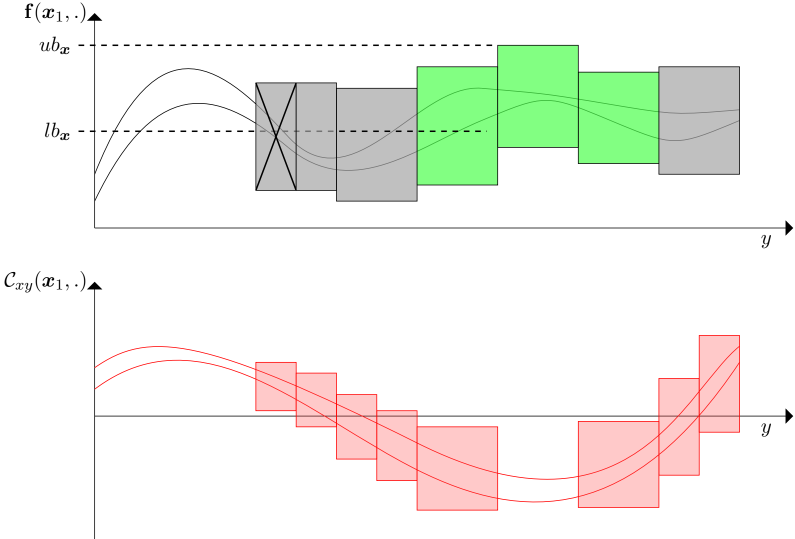
Main Branch and bound with Inclusion properties



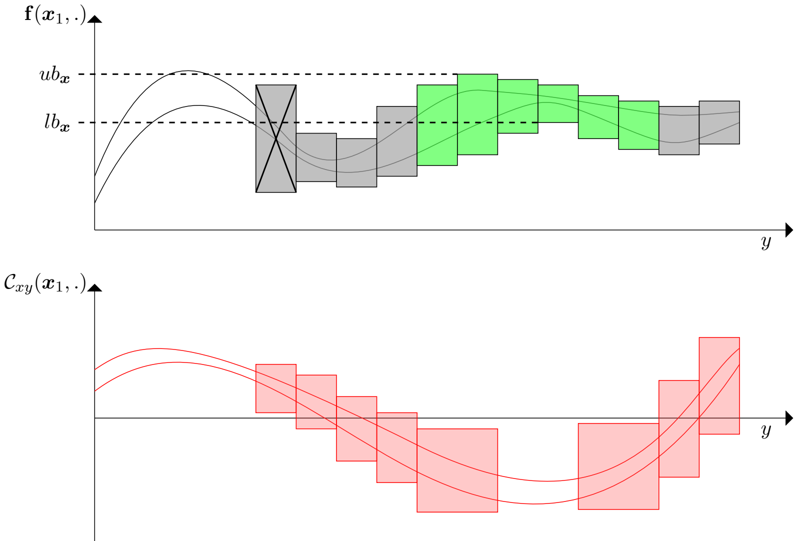
Main Branch and bound with Inclusion properties



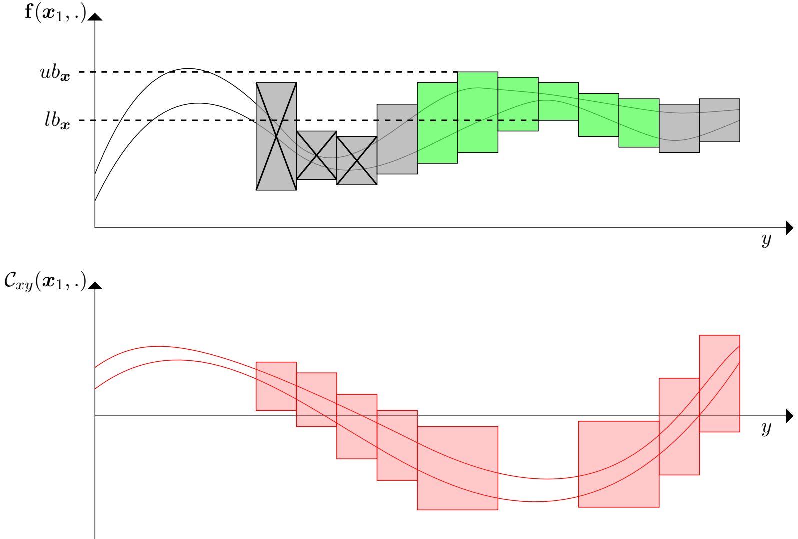
Main Branch and bound with Inclusion properties



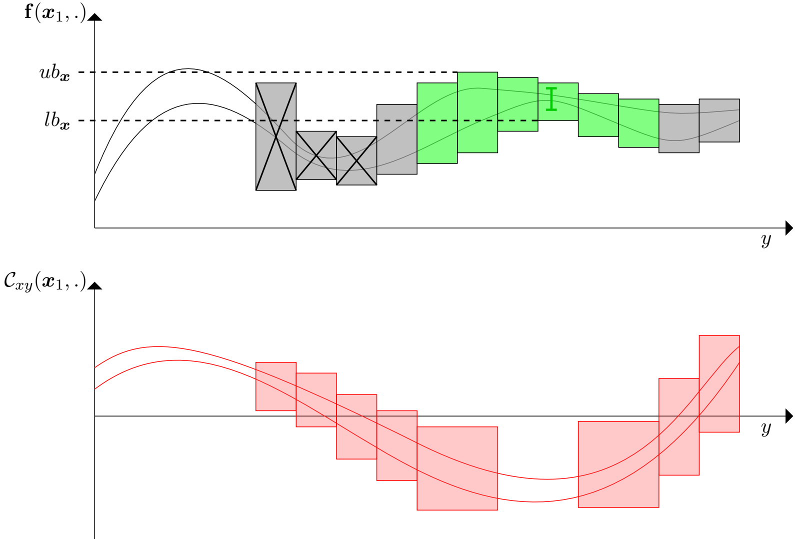
Main Branch and bound with Inclusion properties



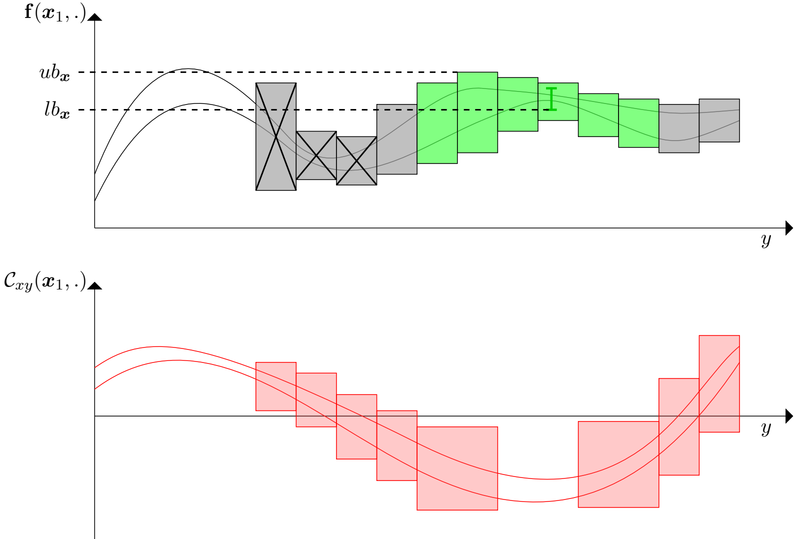
Main Branch and bound with Inclusion properties



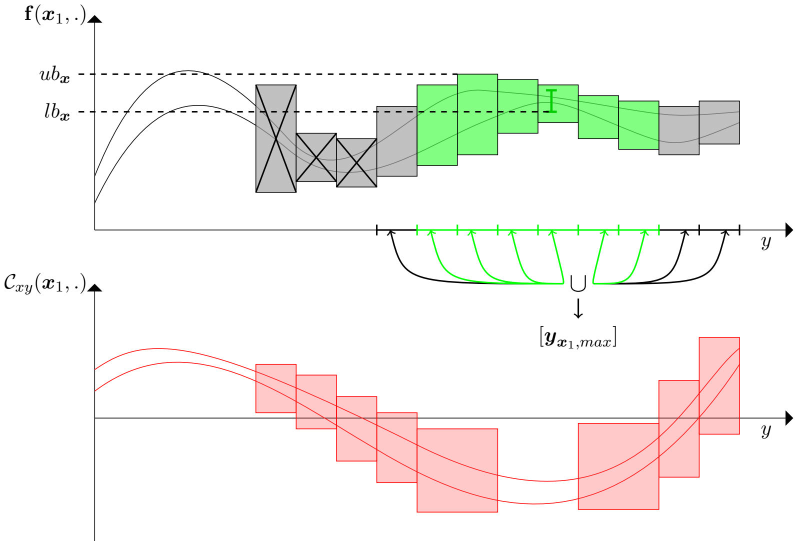
Main Branch and bound with Inclusion properties



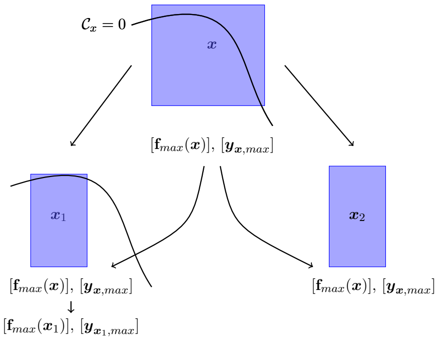
Main Branch and bound with Inclusion properties



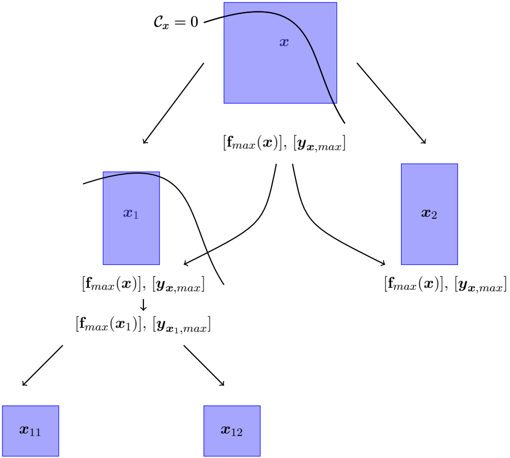
Main Branch and bound with Inclusion properties



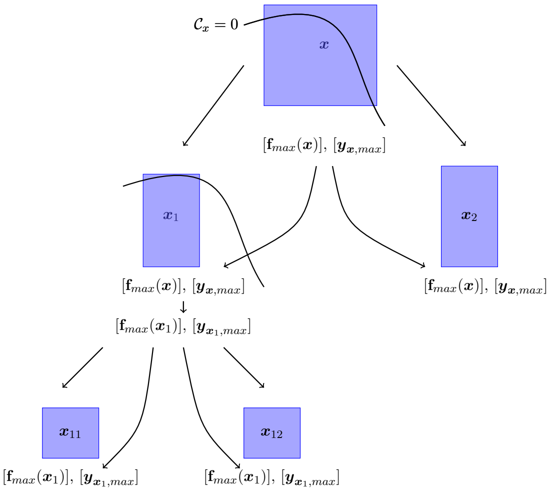
Main Branch and bound



Main Branch and bound



Main Branch and bound



Plan

Examples

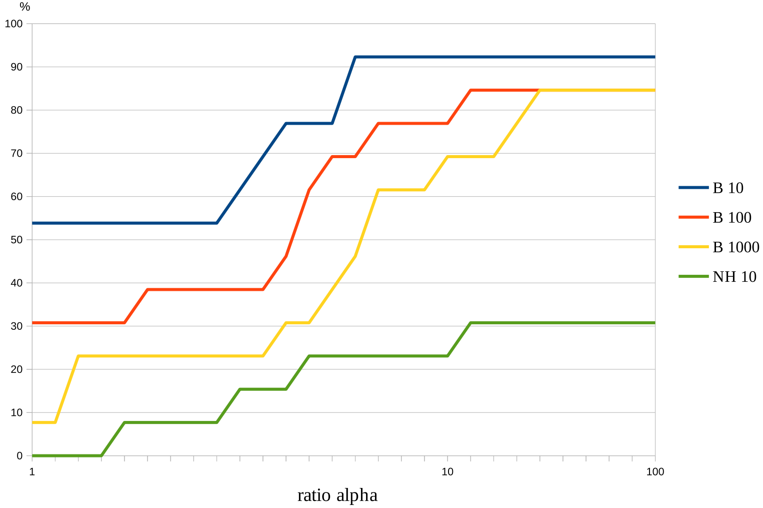
Problems	Obj. func.	x dim	y dim	$\mathcal{C}_x$	$\mathcal{C}_{xy}$
Article example[1]	other	2	2	no	no
Article example[3]	polynomial	1	1	no	yes
Article example[3]	trigonometric	1	1	no	yes
Control	rational	3	1	yes	no
Control	rational	4	1	yes	no
Control	rational	2	1	yes	no
Control	rational	4	1	yes	no
Control	rational	4	1	yes	no
Risk Management[2]	polynomial	2	2	no	no
Risk Management[2]	polynomial	2	2	no	no
Risk Management[2]	polynomial	2	2	no	no
Risk Management[2]	polynomial	2	3	no	no
Risk Management[2]	polynomial	3	3	no	no

Algorithm features

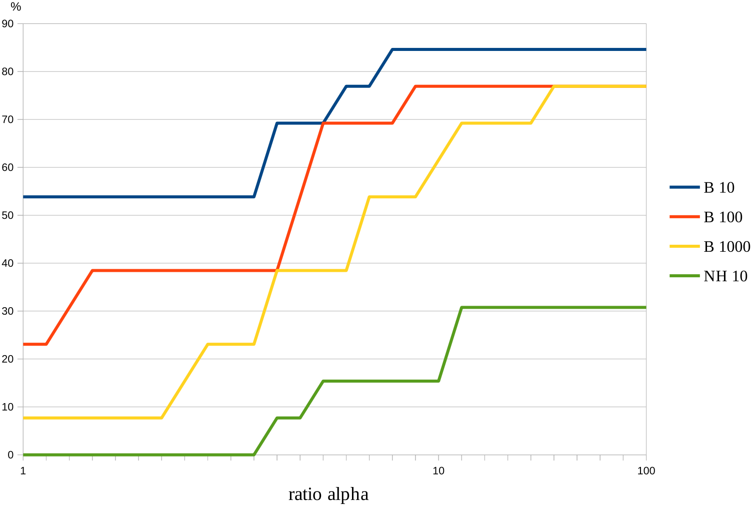
Algorithm is tested with four features:

- 10 bisections are performed in the maximization problem, inclusion properties used → B 10.
- 100 bisections are performed in the maximization problem, inclusion properties used → B 100.
- 1000 bisections are performed in the maximization problem, inclusion properties used → B 1000.
- 10 bisections are performed in the maximization problem, inheritance properties not used → NH 10.

Performance profile: cpu time



Performance profile: number of function evaluation



Plan

Conclusion

- Solver for non-convex problems (non-convex objective function and non-convex constraints).
- Taking advantage of Inclusion properties save computation time.
- Finding the best number of bisection is difficult.

Next steps:

- Test the algorithm on more examples.
- Improve convergence time (monotonicity tests, affine arithmetic, ...).
- How to find the number of bisection?

E. Carrizosa and F. Messine. A branch and bound method for global robust optimization. *Proc. 12th global optimization workshop (Málaga, Spain, September 2014)*, 2014.

B. Rustem and M. Howe. *Algorithms for Worst-Case Design and Applications to Risk Management*. Princeton University Press, 2002.

M. Sainz, P. Herrero, J. Armengol, and J. Vehí. Continuous minimax optimization using modal intervals. *Journal of Mathematical Analysis and Applications*, 339(1):18–30, 2008.