Gurobi 11.0

Every Solution, Globally Optimized

Latest Enhancement in Gurobi 11.0

Chung-Kyun Han
Technical Account Manager

April 2024

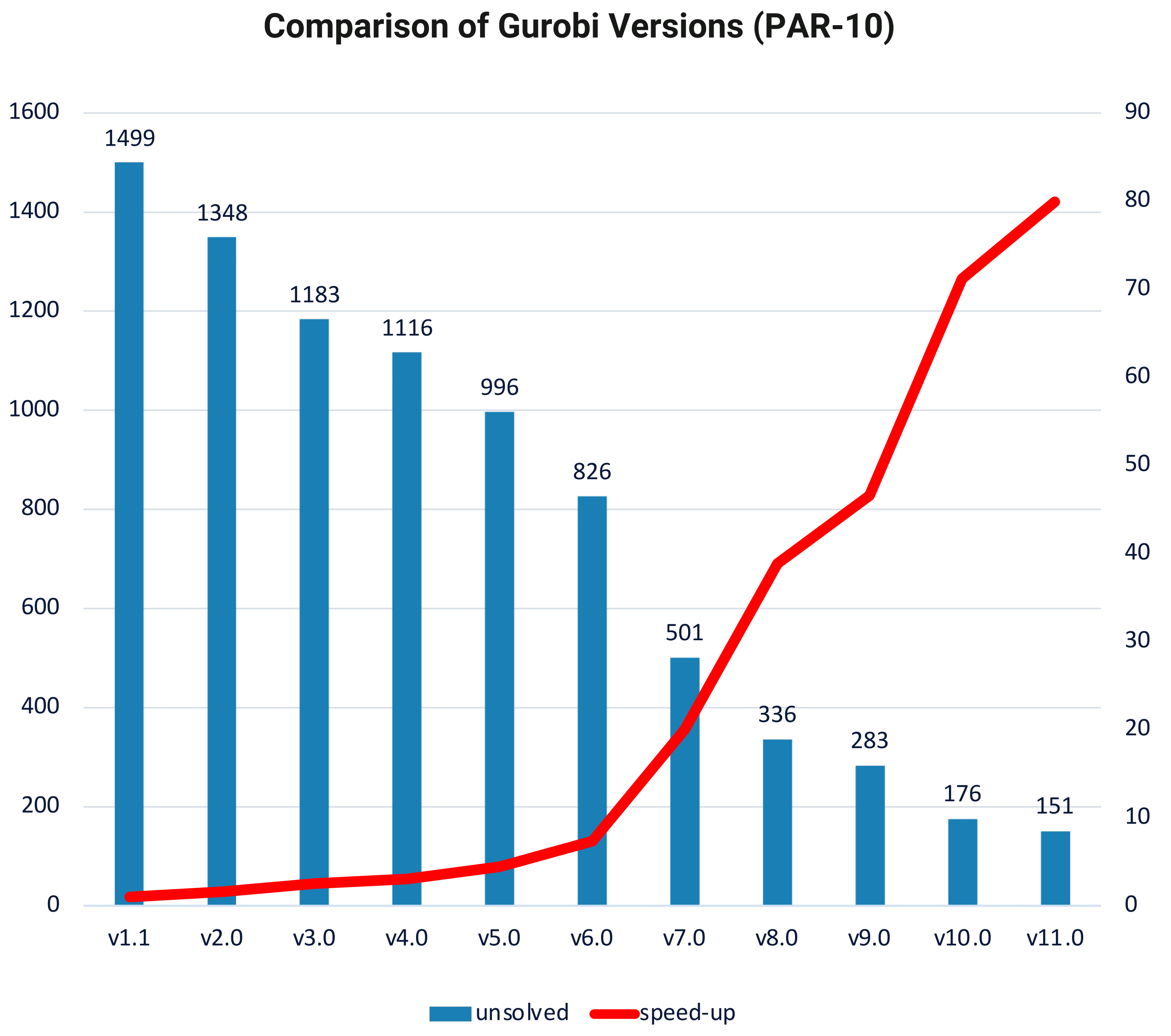
GUROBI OPTIMIZATION

Performance Improvements

GUROBI OPTIMIZATION

MILP

Performance Evolution



Time limit: 10000 sec.
Intel Xeon CPU E3-1240 v5 @ 3.50GHz
4 cores, 8 hyper-threads
32 GB RAM

Test set has 7000 models:
- 647 discarded due to inconsistent answers
- 1854 discarded that none of the versions can solve
- speed-up measured on >100s bracket: 2673 models

LP

Performance Evolution

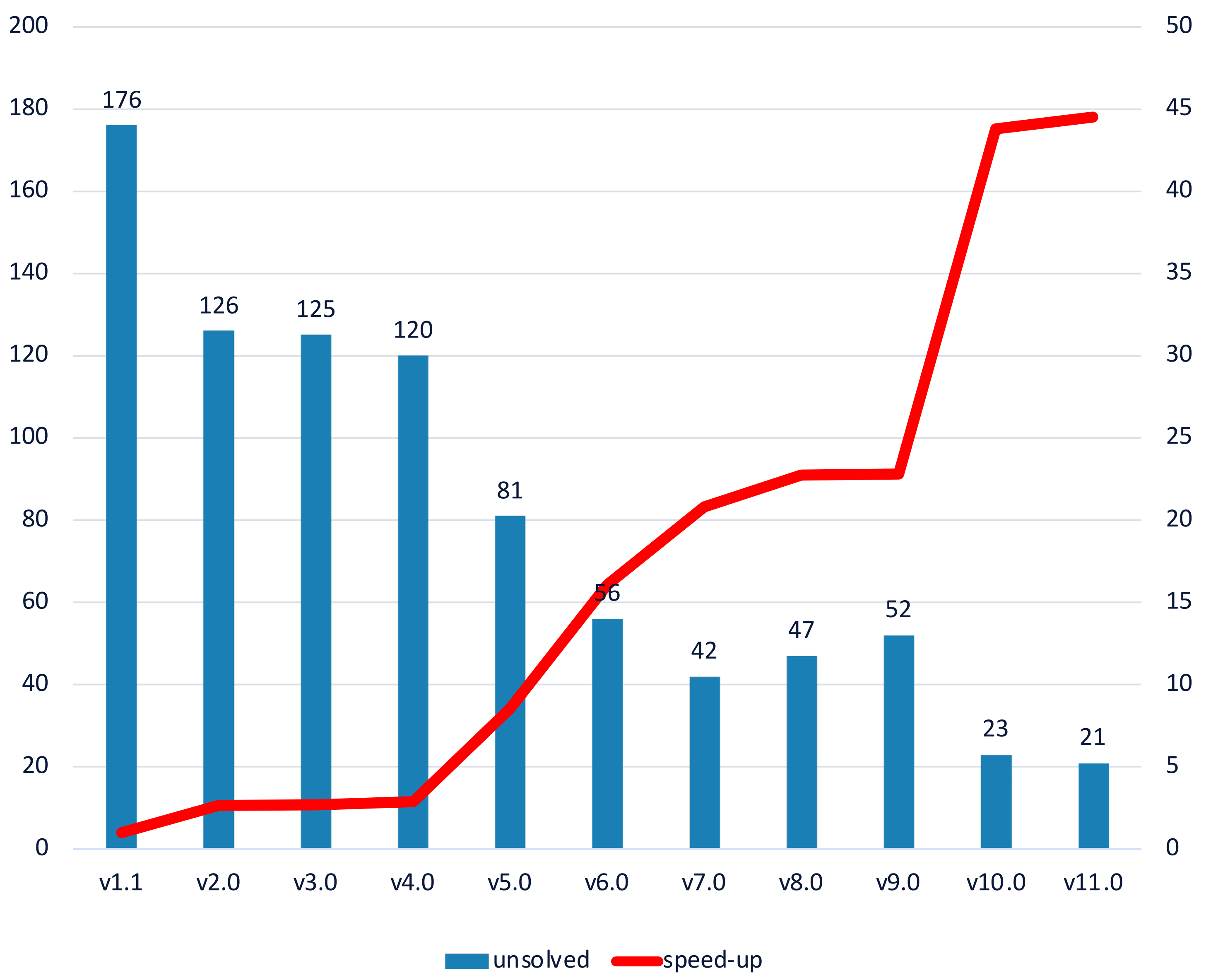
Default settings:

- Gurobi 1 – 4: dual simplex
- Gurobi 5+: concurrent LP

Time limit: 10000 sec.
Intel Xeon CPU E3-1240 v5 @ 3.50GHz
4 cores, 8 hyper-threads
32 GB RAM

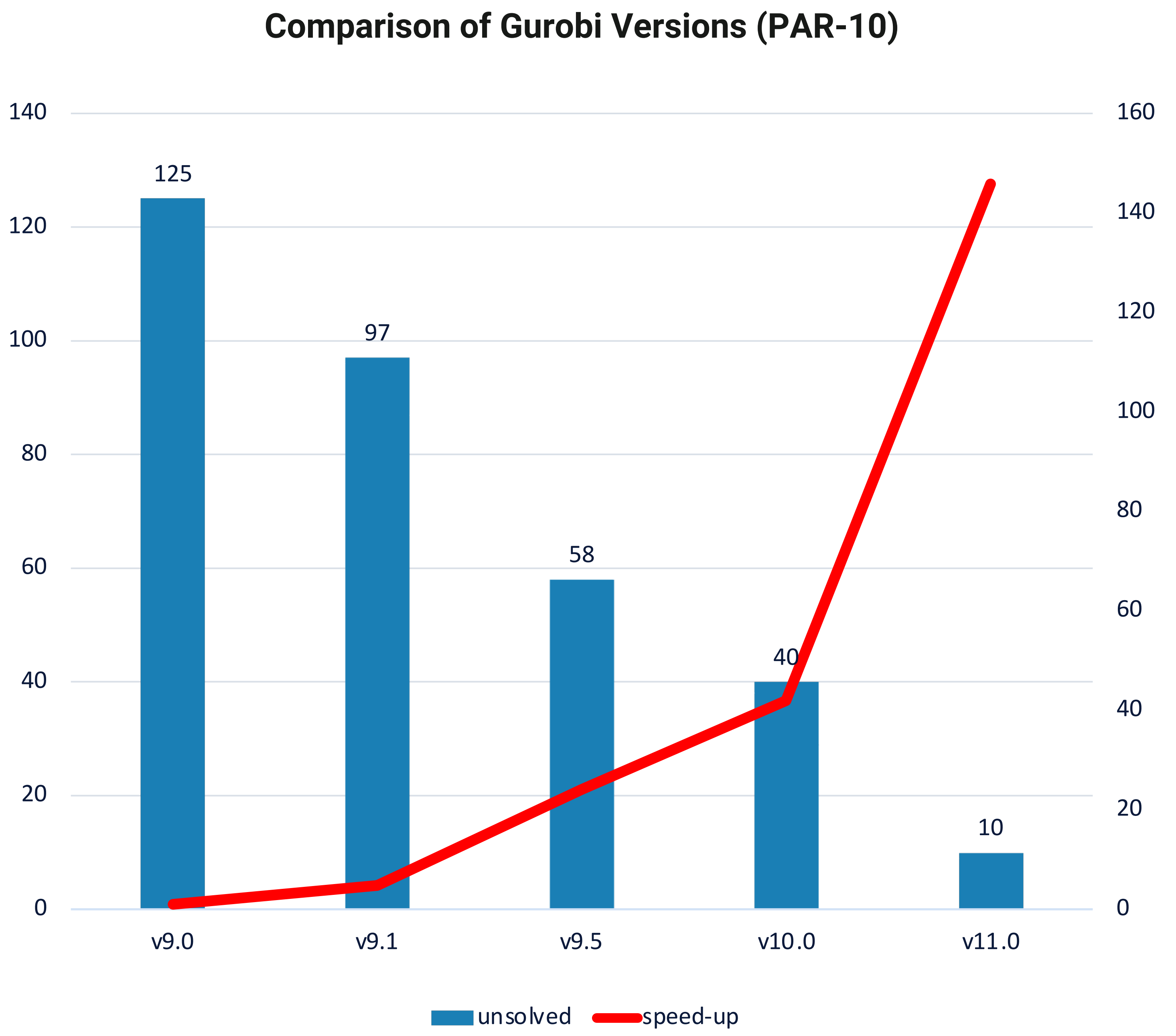
Test set has 2272 models:
- 184 discarded due to inconsistent answers
- 170 discarded that none of the versions can solve
- speed-up measured on >100s bracket: 477 models

Comparison of Gurobi Versions (PAR-10)



Non-convex MIQCP

Performance Evolution



Time limit: 10000 sec.
Intel Xeon CPU E3-1240 v5 @ 3.50GHz
4 cores, 8 hyper-threads
32 GB RAM

Test set has 847 models:
- 34 discarded due to inconsistent answers
- 269 discarded that none of the versions can solve
- speed-up measured on >100s bracket: 229 models

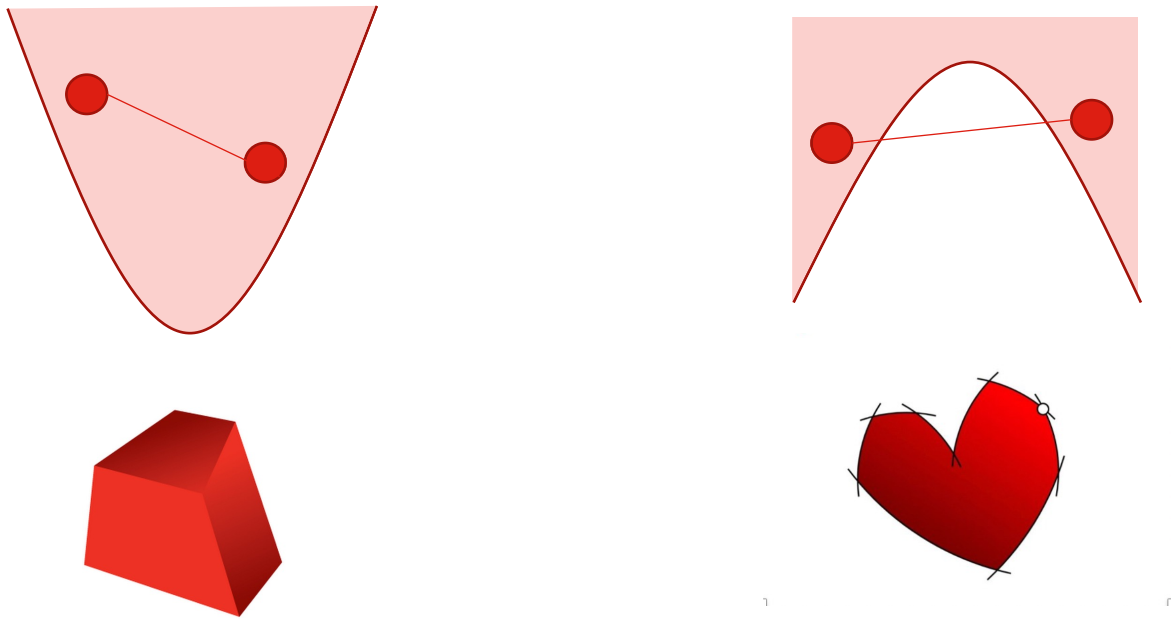
What is non-convex?
What are functional constraints?

Global MINLP

Solve non-convex functional constraints exactly

Convexity

Any point in the feasible region to any other point in the feasible region stays within the feasible region



Have support non-convex MIQCP since version 9.0

Non-Convex QP, QCP, MIQP, and MIQCP

• Pooling problem	(blending problem is LP)
• Petrochemical industry	(oil refinery: constraints on ratio of components in tanks)
• Wastewater treatment	
• Emissions regulation	
• Agricultural / food industry	(blending based on pre-mix products)
• Mining	
• Energy	
• Production planning	(constraints on ratio between internal and external workforce)
• Logistics	(restrictions from free trade agreements)
• Water distribution	(Darcy-Weisbach equation for volumetric flow)
• Engineering design	
• Finance	(constraints on exchange rates)

General and Function Constraints

General Constraints

- Max
- Min
- And
- Or
- Abs
- Indicator

SOS1 Constraints

PreSOS1BigM

Big-M Constraints

Functional Constraints

- Polynomial
- Exponential
- Natural Exponential
- Logarithm
- Natural Logarithm
- Power
- Sine
- Cosine
- Tangent

Can now solve exactly!

Piecewise-Linear Constraints

SOS2 Constraints

PreSOS2BigM

Big-M Constraints

Nonlinear Constraints

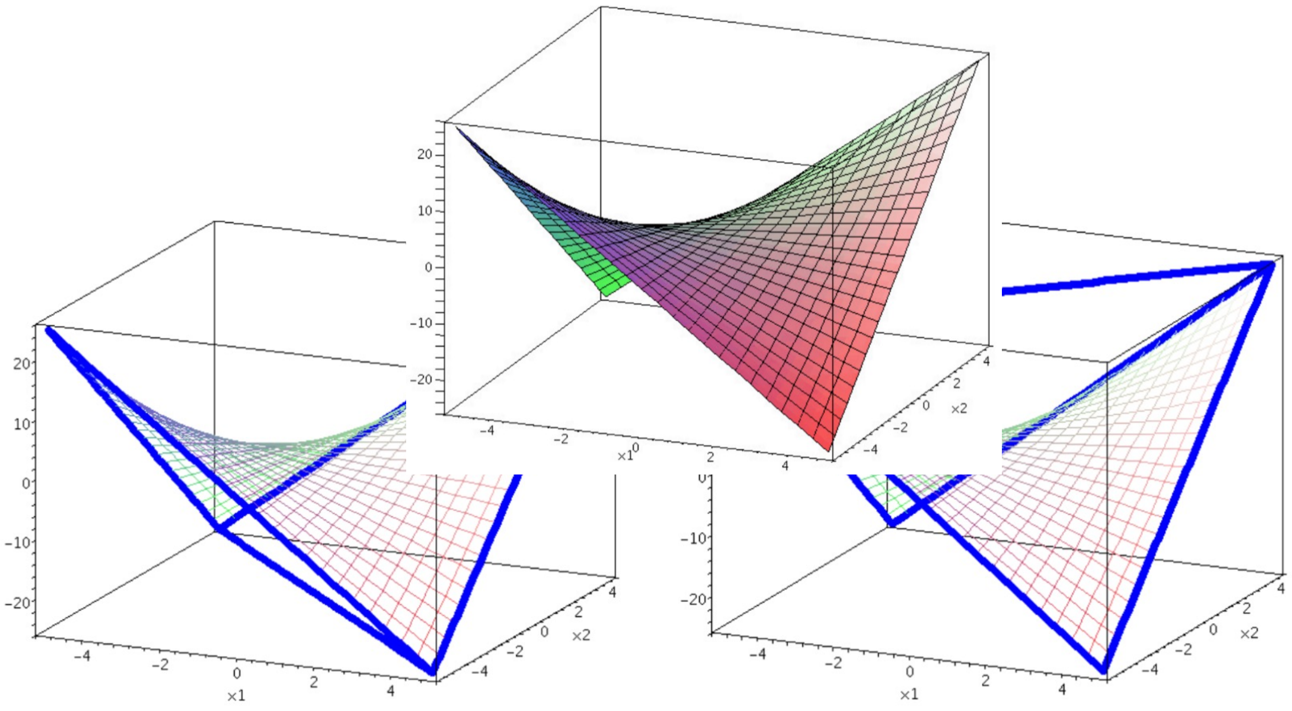
- Gurobi 9.0 and later provide API to define nonlinear functions
  - e^x, a^x
  - $\ln(x), \log_a(x)$
  - $\sin(x), \cos(x), \tan(x)$
  - x^a
  - $ax^3 + bx^2 + cx + d$

```
addGenConstrExp(), addGenConstrExpA()
addGenConstrLog(), addGenConstrLogA()
addGenConstrSin(), addGenConstrCos(), addGenConstrTan()
addGenConstrPow()
addGenConstrPoly()
```

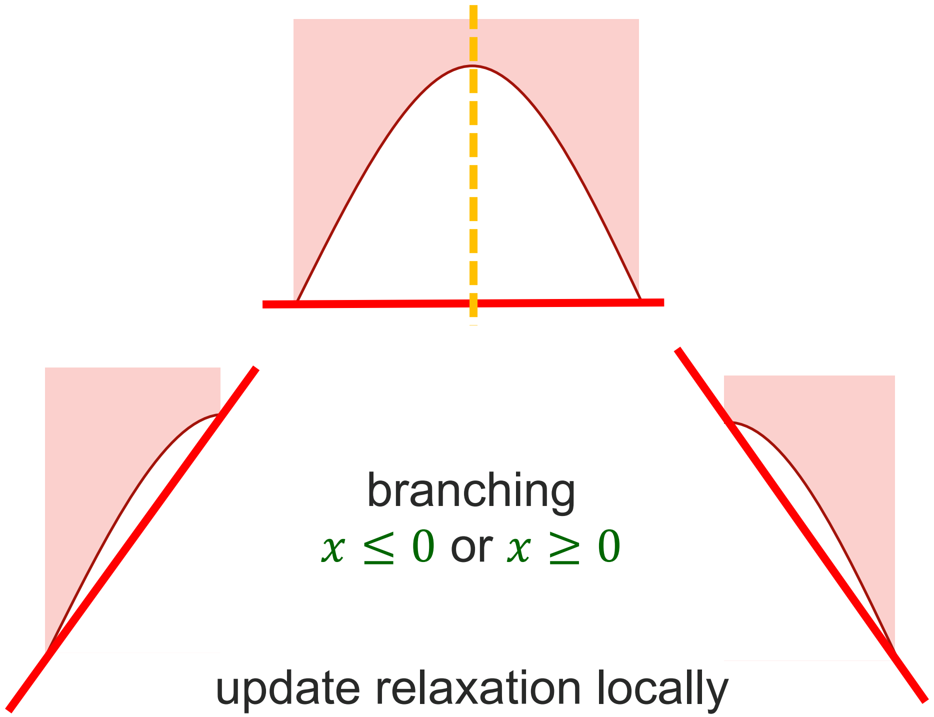
- Gurobi 9.0 – 10.0:
  - Nonlinear functions are replaced during presolve by a piecewise-linear approximation
- Gurobi 11.0:
  - Can choose to treat nonlinear constraints exactly

Two important concepts for non-convex constraints

McCormick Envelopes:
Linear outer approximations

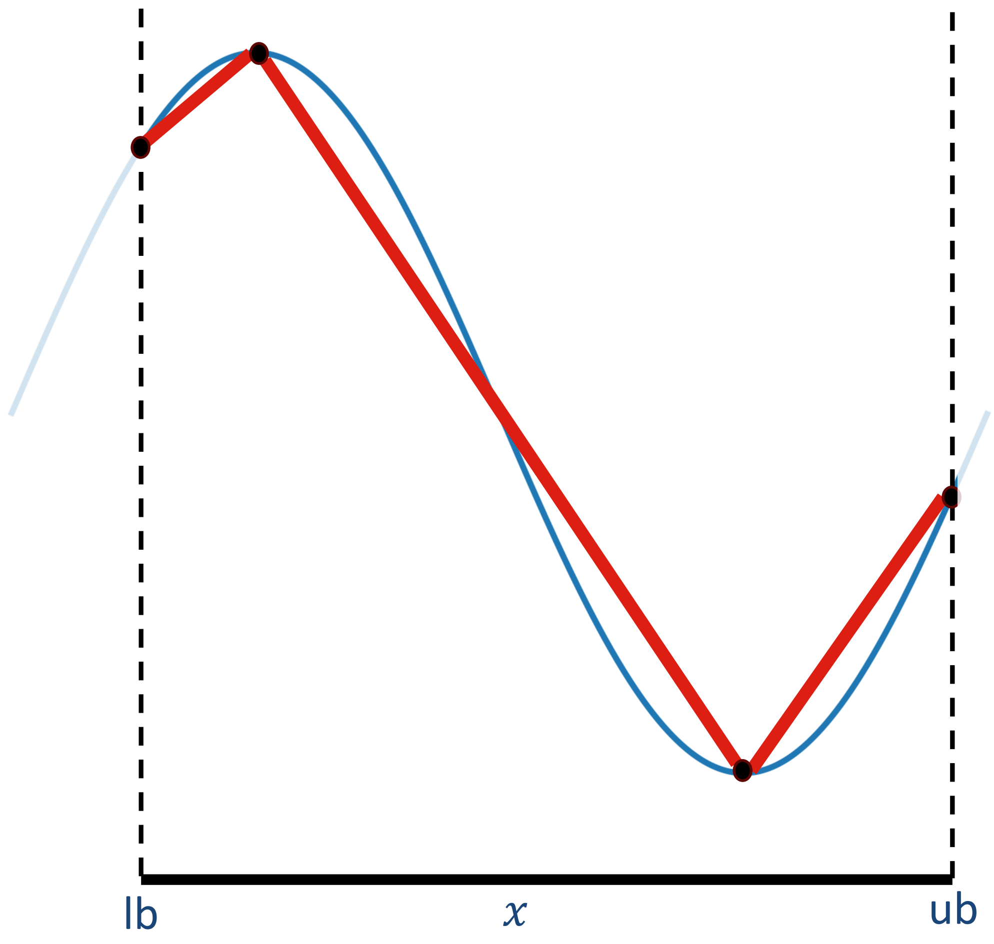


Spatial Branching:
Branch on cont. variables

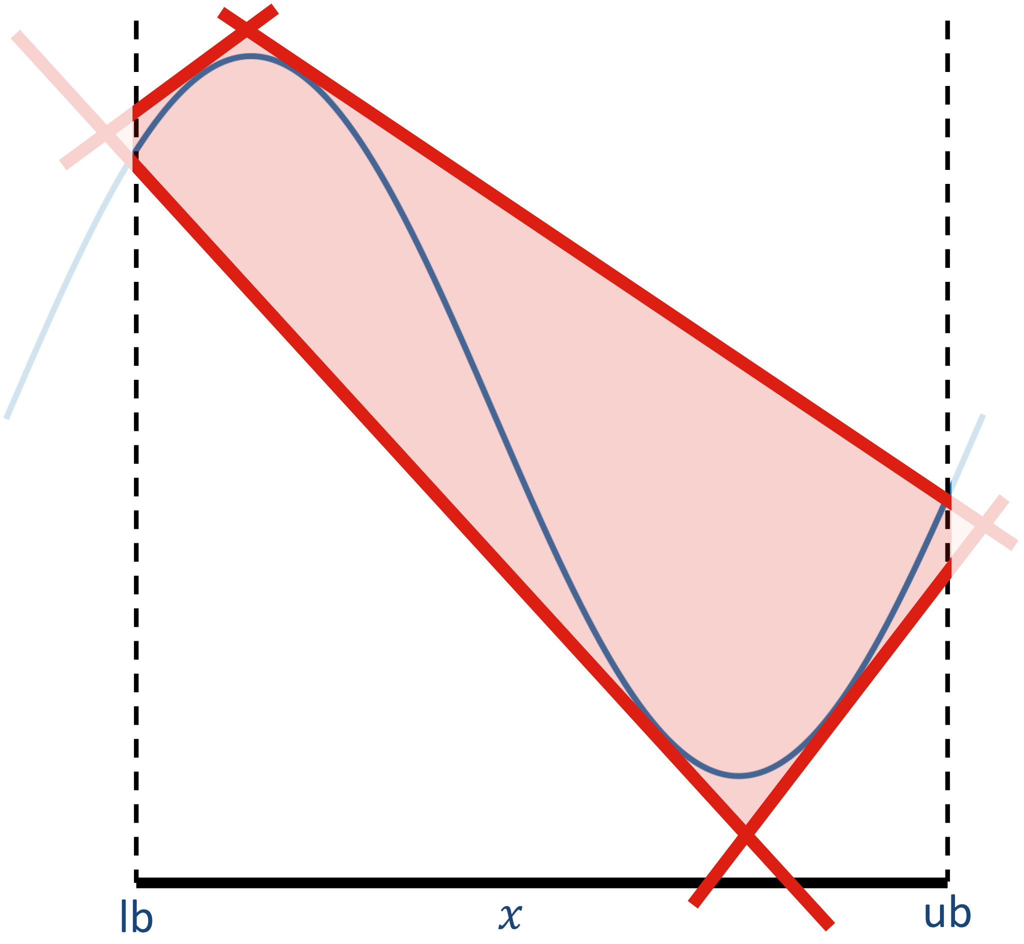


pictures from Costa and Liberti: "Relaxations of multilinear convex envelopes: dual is better than primal"

PWL Approximation vs. Outer Approximation

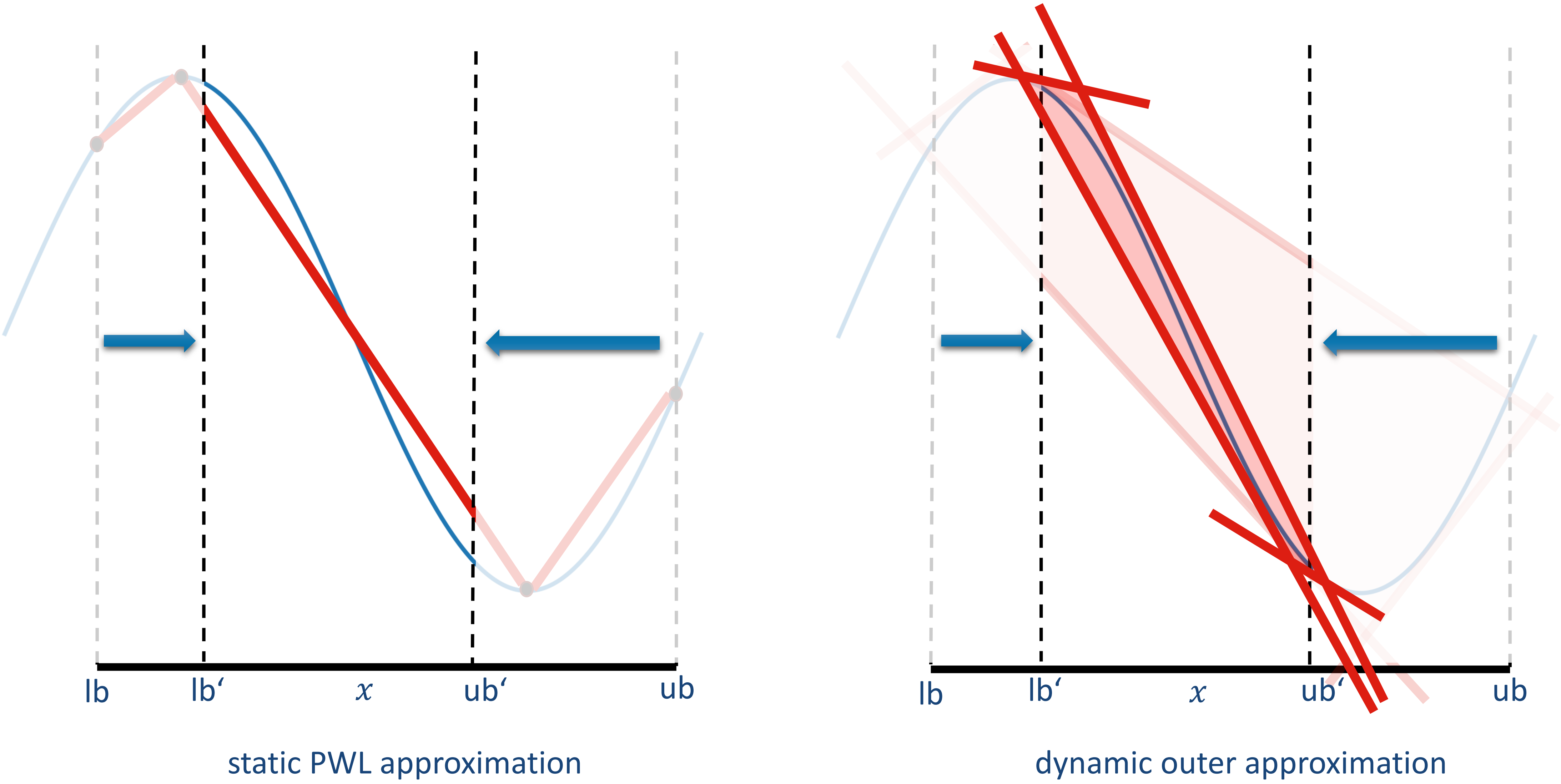


static PWL approximation



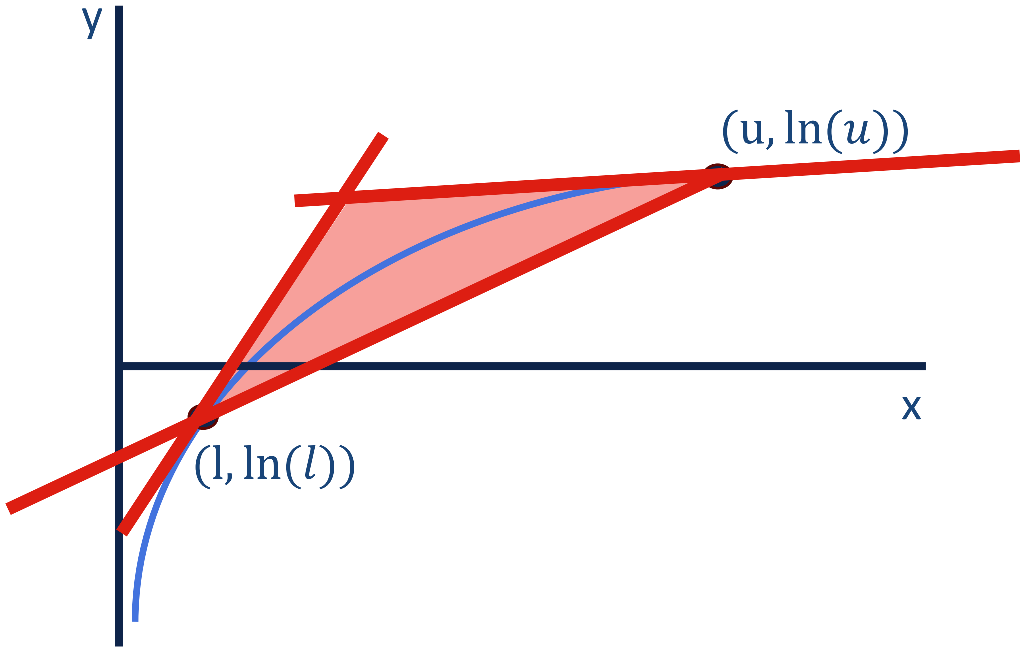
dynamic outer approximation

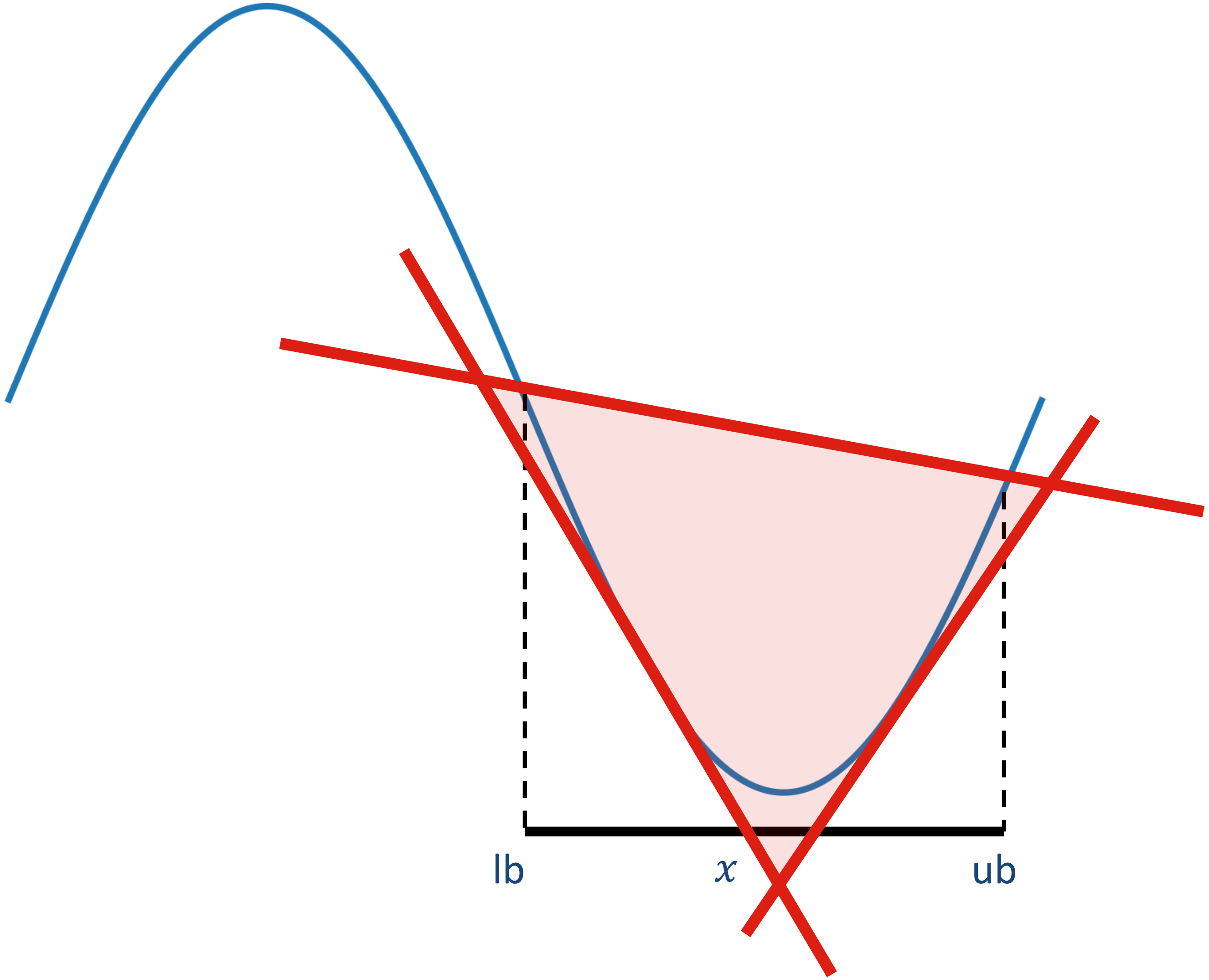
PWL Approximation vs. Outer Approximation



Relaxation of Nonlinear Functions

- Some convex envelopes are easier to approximate than others
  - Example: $y = \ln(x)$, a concave function, $l \leq x \leq u$
  - Lower envelope is given by secant through $\ln(l)$ and $\ln(u)$
  - Upper envelope is constructed by tangents



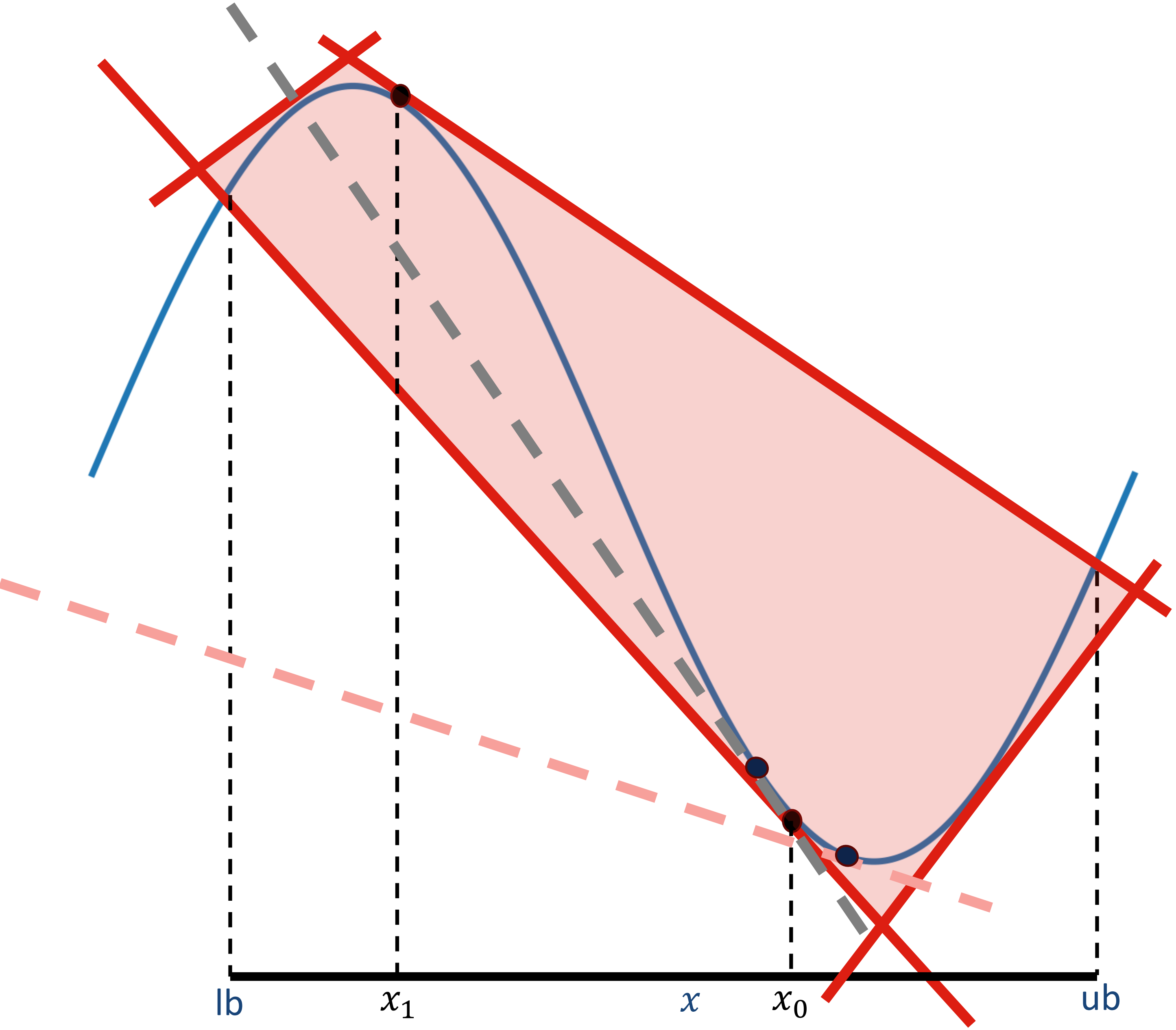


Extend to More Complex Functions

If sin is convex within the bounds of x …

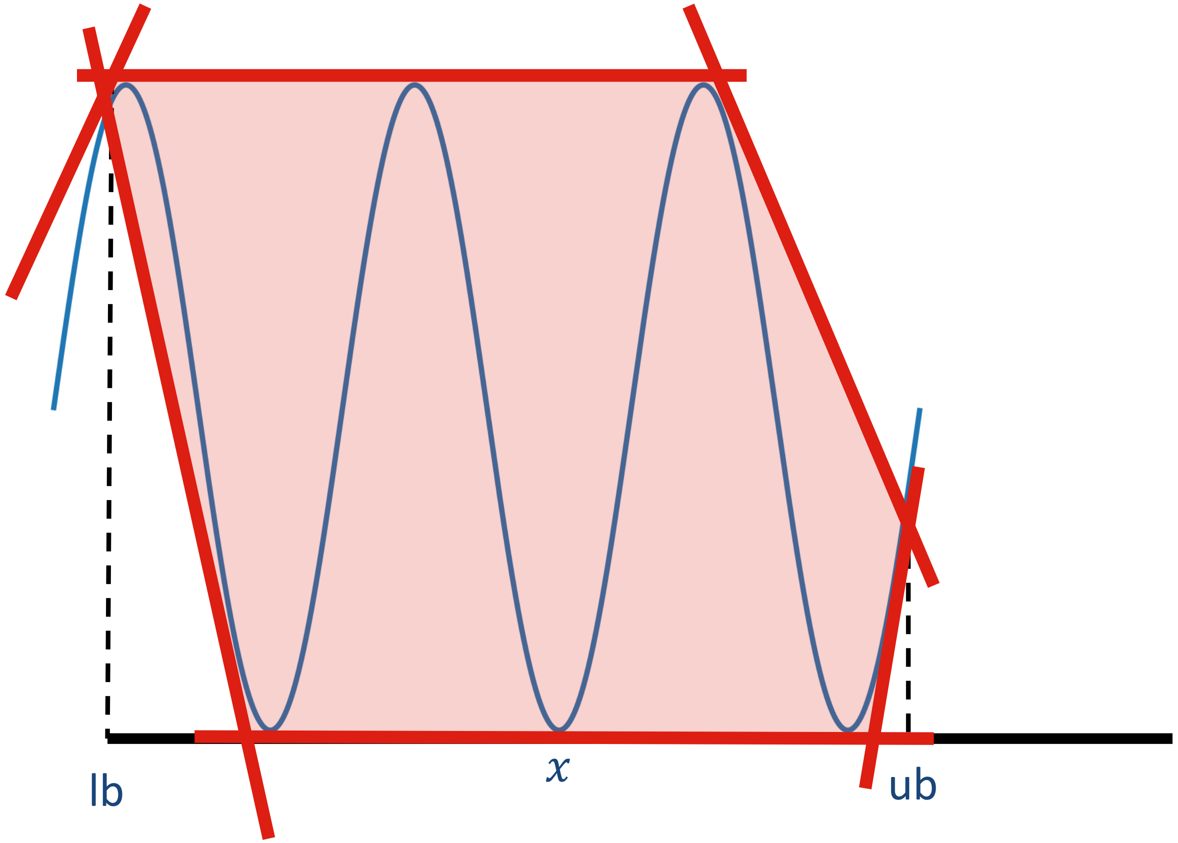
- Upper envelope is given by secant through $f(\mathrm{lb})$ and $f(\mathrm{ub})$
- Lower envelope constructed by tangents to $\sin$
- Resulting hyperplanes added to LP
- Shaded in red: Relaxation of $\mathrm{y} = f(x)$

- Similar if $\sin$ is *concave* on the domain of x
- Adding more tangents at various points improves the relaxation



Neither Convex Nor Concave

- If sin is neither convex nor concave on the domain of x…
- Lower envelope
  - Compute leftmost solution x_0 to
    $$\frac{d}{dx}\sin(x) = \frac{\sin(x) - \sin(\mathrm{lb})}{x - \mathrm{lb}}$$
  - Computed x_0 defines one tangent
  - Remaining part is convex, use some tangent(s)
- Upper envelope
  - Compute rightmost solution x_1 to
    $$\frac{d}{dx}\sin(x) = \frac{\sin(\mathrm{ub}) - \sin(x)}{\mathrm{ub} - x}$$
  - Computed x_1 defines one tangent
  - Remaining part is concave, use some tangent(s)



“Large” Domains

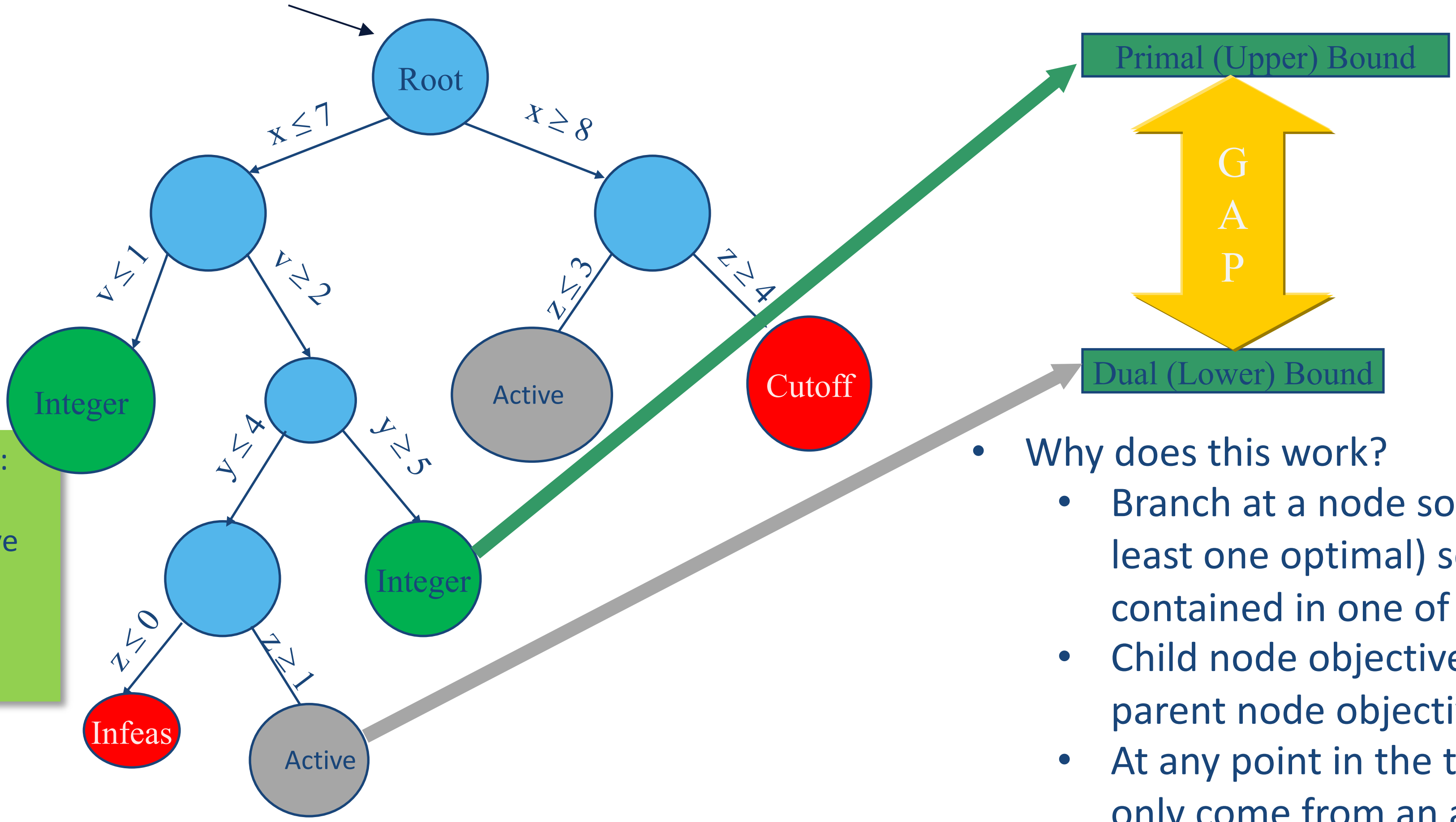
- Not much to get from the relaxation if domain of x is large
- Branching on x tightens the relaxation quickly!
- Tighter initial bounds will speed up performance

Spatial Branch and Bound Overview

- Extend MILP branch and bound to more general nonconvexities
- Branch to eliminate nonconvexity violations in convex relaxations
  - But branching must be done differently
- Add cuts to tighten the relaxations
  - But cuts must be done differently
- Introduce more general presolve reductions

Basic MILP branch and bound

Solve LP relaxation: x=7.55
fractional; branch on violated integrality restriction



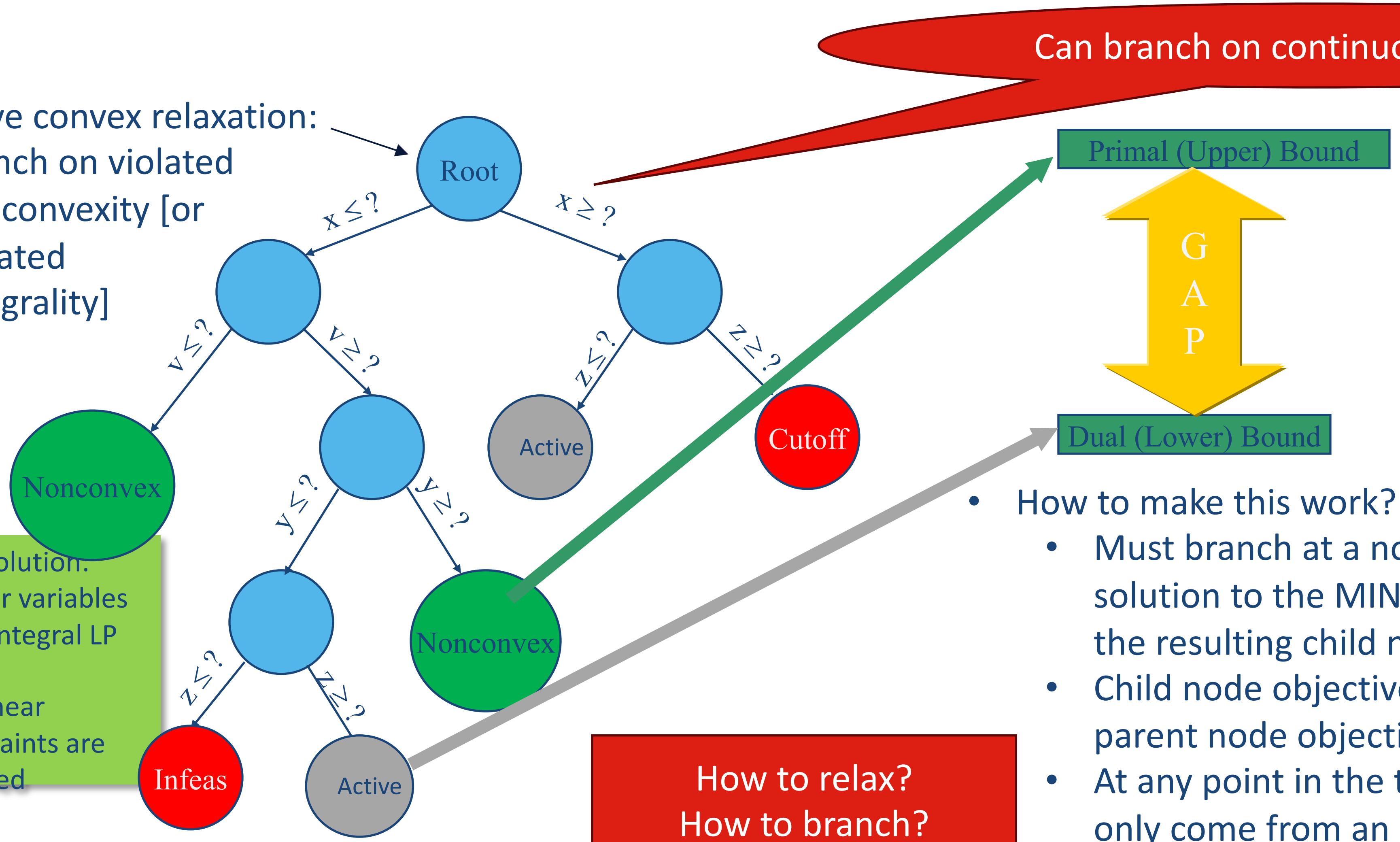
Feasible solution:

- Integer variables have integral LP value

- Why does this work?
  - Branch at a node so that any feasible (or at least one optimal) solution to the MILP is contained in one of the resulting child nodes
  - Child node objective values are no better than parent node objective value
  - At any point in the tree, a better solution can only come from an active node or its descendants

Basic nonconvex MINLP branch and bound

Solve convex relaxation: branch on violated nonconvexity [or violated integrality]



Feasible solution:

- Integer variables have integral LP value
- Nonlinear constraints are satisfied

How to relax?
How to branch?

- How to make this work?
    - Must branch at a node so that any feasible solution to the MINLP is contained in one of the resulting child nodes
    - Child node objective values are no better than parent node objective value
    - At any point in the tree, a better solution can only come from an active node or its descendants

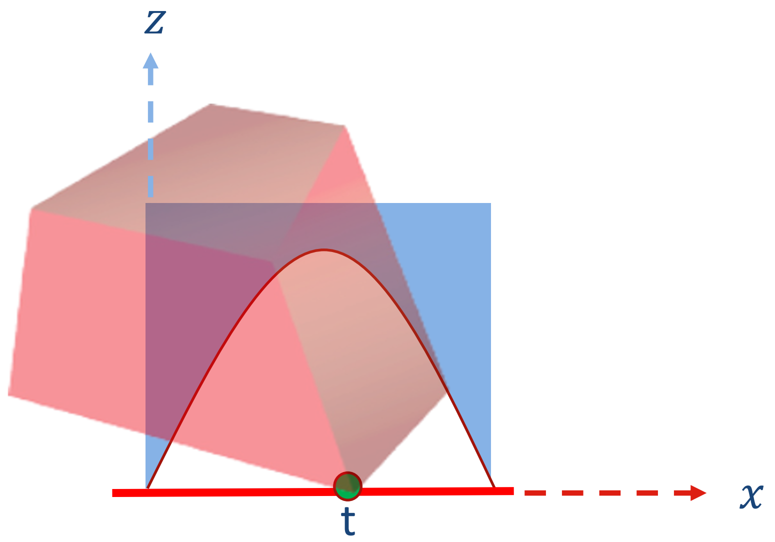
Extensions: Branching

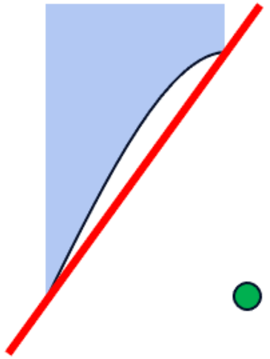
- After solving the convex relaxation, how do we branch on the violated nonconvexities?

How to relax?
How to branch?

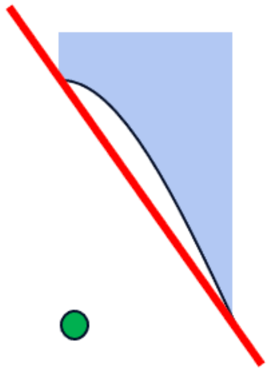
$$
\begin{array}{llllll}
\min & c^T x & + & d^T z & & \\
\text{s.t.} & Ax & + & Dz & \leq & b \\
 & -x_i x_j & + & z_{ij} & = & 0 \quad \text{for all } (i,j) \in S \\
 & l & \leq & x & \leq & u \\
 & & & x_j & \in & \mathbb{Z} \quad \text{for all } j \in I
\end{array}
$$

non-convex
$-z - x^2 \leq 0$





branching
$x \leq t$ or $x \geq t$



update relaxation bounds and the associated McCormick envelopes locally

Example of Benefits

- GurobiML – automatically embed trained machine learning model into MIP model
- Default neural network activation function…
  - ReLU - two-piece piecewise-linear function
- Another commonly used activation function
  - Softmax
  - Typically used in the last layer to translate scores to probabilities
  - Complex non-linear function
- Moved from PWL approximation in 10.0 to dynamic outer approximation in 11.0:
  - 13X performance increase
  - Significantly less error in the results

Gurobi Machine Learning

Regression Models Understood by Gurobi (and which has controllable errors)

scikit learn

- Linear/Logistic regression
- Decision trees
- Neural network with ReLU activation
- Random Forests
- Gradient Boosting trees
- Transformations:
  - Simple scaling of features
  - Polynomial features of degree 2
- Pipelines to combine them

dmlc XGBoost

- “gbtree” booster

Keras

- Dense layers
- ReLU layers
- Object Oriented, functional or sequential

PyTorch

- Dense layers
- ReLU layers
- Only torch.nn.Sequential models

LightGBM

Gurobi Machine Learning

Usage

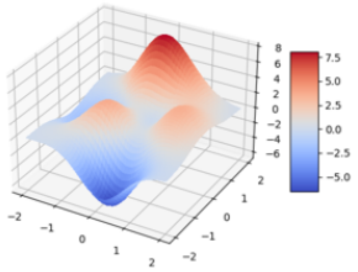
GUROBI OPTIMIZATION

- Say have trained the following regression with scikit-learn:

```
pipeline = make_pipeline(StandardScaler(), MLPRegressor([10]*2))
pipeline.fit(X_train, y_train)
```

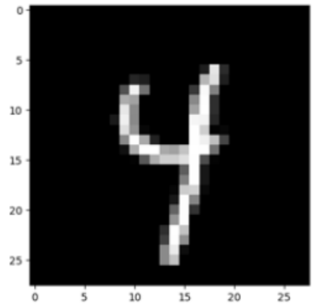
- Embedding into a Gurobi model

```
m = gp.Model()
# Add matrix variables for the regression
input = m.addMVar((n_constr, X_train.shape[1]), lb=-GRB.INFINITY)
output = m.addMVar(n_constr, lb=-gp.GRB.INFINITY)
# Add predictor constraint
pred_constr = add_predictor_constr(m, pipeline, input, output)
```

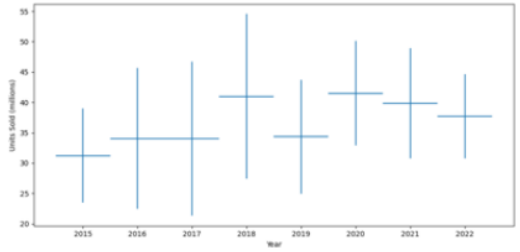



Surrogate Models

Student Enrollment



Adversarial Machine Learning



Price Optimization

Use Cases & Examples

[Examples — Gurobi Machine Learning documentation (gurobi-machinelearning.readthedocs.io)]

MINLP Roadmap for Gurobi 12

- Add API to state composite nonlinear functions directly
  - Use composite nonlinear functions for feasibility checks
  - Use composite nonlinear functions for interior point NLP solver
  - Exploit knowledge about composite nonlinear in presolve and for outer approximation
- Improve global MINLP performance
  - Presolve reductions
  - Cutting planes
  - Improve heuristics to better work with nonlinear constraints
  - Better branching variable and split point selection
- Interior point local NLP solver
  - Expose our internal local NLP solver to the user
    - Provides a locally optimal solution
  - Improve performance and robustness of local NLP solver
- Improve numerics

Name: Chung-Kyun Han
Email: chung-kyun.han@gurobi.com

Thank You

For more information: gurobi.com