Gurobi 10.0 新亮点

顾宗浩 博士，Gurobi CTO 和联合创始人

GUROBI OPTIMIZATION

主讲人介绍

- **顾宗浩 博士**
- Gurobi 首席技术官 (CTO) 和联合创始人
- 上海同济大学机械工程学士和工业管理硕士，佐治亚理工学院工业工程博士
- 曾担任高级软件开发人员，后来担任ILOG CPLEX研发团队的杰出科学家和架构师
- 在线性、整数和二次规划计算方面世界最领先的专家之一

讲座大纲

性能提升新技术

易用功能和 API 接口改进

开源 Github 应用库

GUROBI OPTIMIZATION

GUROBI
OPTIMIZATION

性能提升新技术

GUROBI OPTIMIZATION

Gurobi 10.0

性能提升汇总

相对于 Gurobi 9.5 性能提升汇总

类型	整体提升	>100秒复杂模型
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

GUROBI OPTIMIZATION

Gurobi 10.0

性能提升汇总

相对于 Gurobi 9.5 性能提升汇总

类型	整体提升	>100秒复杂模型
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

LP 性能

- 新增网络单纯形算法 New network simplex algorithm
- LP 并发算法改进: 仅对预优化之后的最终模型进行并行计算
- Crossover 改进
  - 并行原始单纯形外推 Parallel primal pushes
  - 外推前内点法解调整 Barrier solution adjustment before pushes
- 新增和改进模型精简算法，提供更多选项
  - 把一些 MIP精简算法应用到 LP, 例如 PreSparsify
    - 处理对偶和基的反压缩 Handle dual and basis uncrush
  - Aggregate 合并参数新增数值 2

网络单纯形算法 (Network Simplex Algorithm)

- 针对问题
  - 最小成本流 Minimum cost flow
  - 可以建模为常规 LP 模型，用常规LP 求解器
- 开发动力
  - 众所周知: 经常在 OR, CS 和 Math 课程上讲到
  - 研究较多: 许多不同算法
    - 连续最短路径算法 Successive shortest path algorithm
    - 缩放算法，多项式运行时间 Scaling algorithms, polynomial
      - Cost scaling, capacity scaling, double scaling, etc.
    - 网络单纯形算法 Network simplex algorithms
      - Primal and dual network simplex
    - Reference: Network Flows, R. Ahuja, T. Magnanti and J. Orlin

网络单纯形算法 (Network Simplex Algorithm)

- Gurobi 网络单纯形算法
  - 只实现原始单纯形
  - 最主要的挑战部分
    - 生成树(spanning tree, i.e., basis)的数据结构
    - 维护/更新生成树
- 原始单纯形网络算法相比常规 LP 原始单纯形算法优势在于
  - 可以利用特殊结构加快速度: 大概在五倍速
  - 强可行生成树: 保证没有循环(cycling)
  - 较容易构造特殊算法，产生较好的初始生成树(basis crash)等
  - 在我们网络数据集的性能
    - Vs. 广义原始单纯形: 36X, 大约减少 50% 迭代
    - Vs. 广义对偶单纯形: 3.9X, 大约增加10X 迭代
- 对偶网络单纯形
  - 还未开发: 类似的挑战，也许更难一些
  - 还不清楚简单好用的方法来保证没有循环
  - 预期比原始网络单纯形算法要快

并发LP算法(Concurrent LP Algorithms)

- Gurobi LP 算法求解过程 (单纯形和内点法)
  - 决定求解原始还是对偶形式
    - 结论经常对于单纯形法或内点法而不同
  - 应用 LP 精简算法产生预优化模型
    - 对于原始和对偶单纯形相同，对于内点法稍有不同
  - 折叠(Fold)预优化模型，如果存在弱对称性
  - 再次应用 LP 精简算法产生更小的预优化模型
  - 用单纯形或者内点法求解最终的预优化模型
  - 反压缩(Uncrush) 折叠的预优化模型，应用单纯形法在折叠的模型上
  - 反折叠(Unfold) 并采用交叉转换(Crossover) 得到基解
  - 反压缩(Uncrush) 并采用单纯形法在初始模型上直到结束
- 开发动力
  - 计算机核数持续增加
  - 越来越多用户使用并发LP求解器
  - 并发LP 求解器通常使用更多的内存；一些用户建议设置软性内存限制参数，以便限制内点法内存占用

并发LP算法(Concurrent LP Algorithms)

- V9.5 及以前版本的并发LP算法
  - 每个并发任务都会保留一个模型拷贝
    - 通常占用很大内存
  - 每个任务，不论原始单纯形，对偶单纯形还是内点法独立运算所有步骤
    - 每个步骤都由并发任务各自独立并发完成
- 并发运行的任务可能会让计算速度显著降低
  - 取决于机器和模型规模，速度可能会降低 30% - 60%
- 新并发LP算法
  - 只在最终预优化模型上应用并发算法
  - 其他步骤顺序执行
  - 挑战在于
    - 决定求解形式是原始形式还是对偶形式
    - 管理单纯形和内点法的预优化差异
  - 对于规模较大模型的提速经常远超 10%
  - 如今使用更少的内存
    - 取决于预优化模型规模，而不是初始模型规模

GUROBI OPTIMIZATION

Gurobi 10.0

性能提升汇总

相对于 Gurobi 9.5 性能提升汇总

类型	整体提升	>100秒复杂模型
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

MIP 性能

- 各种增强分支(Strong branching)方法的改进
- 多种对称性改进
- 深度搜索时(diving), 关闭非有效切平面以便更好求解松弛问题
- 求解子MIP问题时, 用力度更大的设置
- 新预优化简化算法和改进
- 对于松弛问题的并发LP算法改进和调优
- 力度更大地融和聚类(cliques) 和VUB
- 基于优化的边界紧缩算法 Optimization-based bound tightening (OBBT)
  - 对 MIQP/MIQCP/MINLP 有帮助，后续还会介绍
- 对于机器学习模型的多种改进
  - 将在后续开源 Github 应用库中介绍

增强分支方法的改进

- 增强分支(Strong branching)
  - 选择取分数值的0-1变量/整数变量的集合
  - 对于每个变量，执行向上和向下分支的一定数量的对偶迭代
    - 利用目标值在二个分支上的变化来选择分支变量
- 尝试各种方法
  - 增强分支通常代价较高，需要事半功倍
    - 具有前瞻性, 定义终止数值
    - 利用对称性节省步骤
  - 利用从向上和向下分支获得的引导
    - 传播引申的边界值
    - 传播聚类(cliques)
    - 传播 SOS 约束
  - 应用增强分支的频率
  - 各种其他考虑

GUROBI OPTIMIZATION

Gurobi 10.0

性能提升汇总

相对于 Gurobi 9.5 性能提升汇总

类型	整体提升	>100秒复杂模型
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

MIQP/MIQCP 性能

- 新 QUBO 启发算法
- 远景强化 Perspective strengthening
- 把目标中的Q 项移到约束中
- 对于 QC 固定启发算法的工作限度(work limit)调整
- 加强二次约束中0-1变量的系数
- 在启发算法中以一定次序固定0-1变量
- 求解集覆盖问题来选择线性化方式
- 移除共同变量
- 基于优化的边界紧缩算法 OBBT
- 其他 MIP 改进同样适用

新 QUBO 启发算法

- 二种类型的启发算法
  - 构造: 发现新的可行解
  - 改进: 提升已有可行解
- 9.5 版本中的QUBO 算法
  - 禁忌搜索(Tabu search) – 改进算法
    - 从一个随机起点出发
  - 局部改进对于 QUBO 相对容易
    - 没有约束
- 10.0版本中的新 QUBO 算法
  - Rank-2 松弛启发算法 – 构造算法
    - Burer, Monteiro, and Zhang, *Rank-Two Relaxation Heuristics for MAX-Cut and Other Binary Quadratic Programs*
    - 行和列映射到圆上的点
    - 考虑所有可能切分圆的弦

新 QUBO 启发算法

- 我们计算结果显示
  - 对于 QUBO 问题可以快速找到好的解
  - 并且可以显著缩短优化的时间，对于启发算法来说很少见。

GUROBI OPTIMIZATION

Gurobi 10.0

性能提升汇总

相对于 Gurobi 9.5 性能提升汇总

类型	整体提升	>100秒复杂模型
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

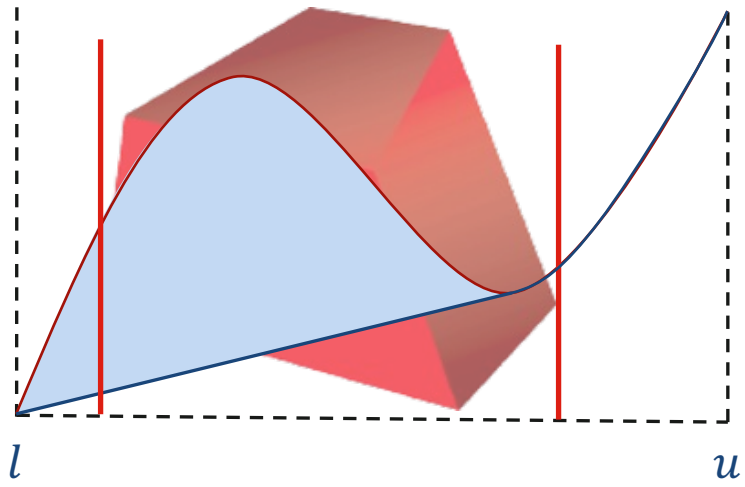
非凸 MIQCP 性能

- 基于优化的边界紧缩算法 OBBT
- 在乘积项覆盖中显性处理二分图 (Dealing explicitly with bipartite graphs in the product term covering)
- NLP 启发算法终止条件的改进
- NLP 启发算法的多次启动
- 其他 MIP 和 凸 MIQCP 改进同样适用

基于优化的边界紧缩算法
Optimization Based Bound Tightening

- 对于 MINLP 求解器的通常技术
- 给定一个（非凸）MINLP 的LP松弛模型
- 对于任何一个变量 x
  - 最小化/最大化 x 松弛数值
  - 用最优值当作 x 的下界/上界
  - 用新的上下界紧缩松弛系数
- OBBT 改进 (Gleixner et al. 2017)
  - 筛选变量
  - 利用热启动
  - 利用 OBBT LP的对偶解紧缩分支树的边界

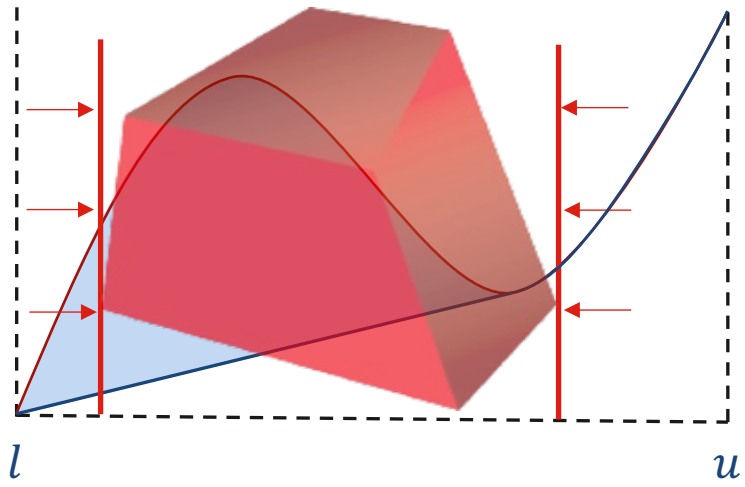
e.g.: $\mathrm{conv}(y \geq f(x): l \leq x \leq u) \cap X$



基于优化的边界紧缩算法
Optimization Based Bound Tightening

- 对于 MINLP 求解器的通常技术
- 给定一个（非凸）MINLP 的LP松弛模型
- 对于任何一个变量 x
  - 最小化/最大化 x 松弛数值
  - 用最优值当作 x 的下界/上界
  - 用新的上下界紧缩松弛系数
- OBBT 改进 (Gleixner et al. 2017)
  - 筛选变量
  - 利用热启动
  - 利用 OBBT LP的对偶解紧缩分支树的边界

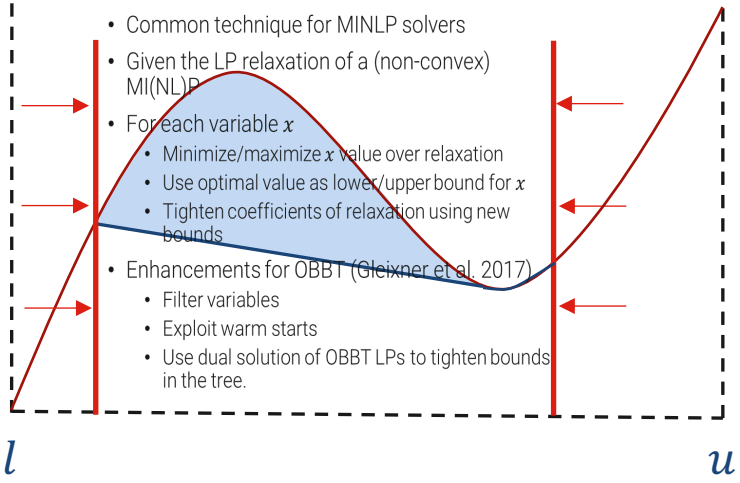
e.g.: $\mathrm{conv}(y \geq f(x): l \leq x \leq u) \cap X$



基于优化的边界紧缩算法
Optimization Based Bound Tightening

- 对于 MINLP 求解器的通常技术
- 给定一个（非凸）MINLP 的LP松弛模型
- 对于任何一个变量 x
  - 最小化/最大化 x 松弛数值
  - 用最优值当作 x 的下界/上界
  - 用新的上下界紧缩松弛系数
- OBBT 改进 (Gleixner et al. 2017)
  - 筛选变量
  - 利用热启动
  - 利用 OBBT LP的对偶解紧缩分支树的边界

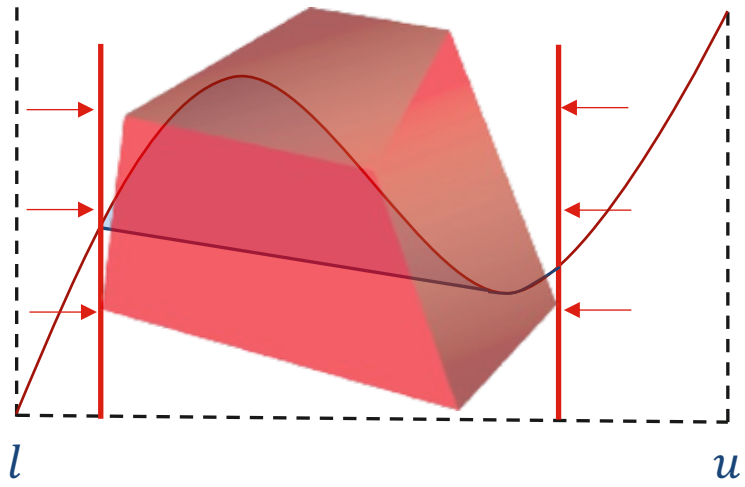
e.g.: $\mathrm{conv}(y \geq f(x): l \leq x \leq u) \cap X$



基于优化的边界紧缩算法
Optimization Based Bound Tightening

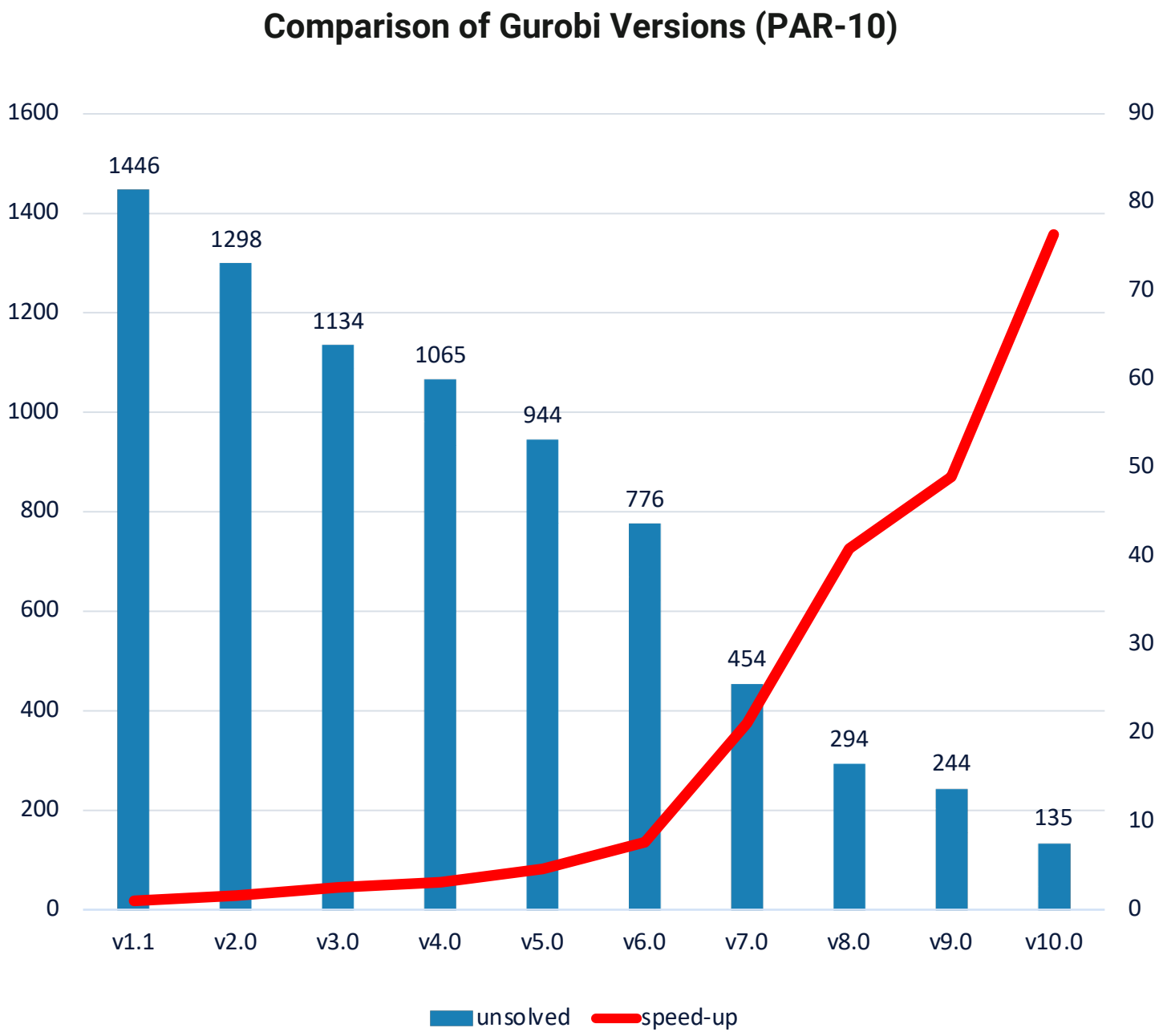
- 对于非凸 MIQCP:
  - 14% 整体提升
  - 33% 大于100s 复杂模型的提升
- 对于 MIP, 额外的提升包括:
  - 检测到对于大M系数有影响的变量
  - 对那些变量进行聚集
  - 在每个聚类中应用 OBBT, 并且传播
  - 着眼于带有 ReLU 结构的神经网络 (inspired by Fischetti, Jo 2017)
  - 对于MIP/MIQP/MIQCP的适中提升: 1%
  - 但对于特定模型 (NN with ReLU) 有巨大提升

e.g.: $\mathrm{conv}(y \geq f(x): l \leq x \leq u) \cap X$



GUROBI OPTIMIZATION

MILP
性能演变



GUROBI
OPTIMIZATION

易用功能和 API 接口改进

Gurobi 10.0 优化引擎

易用功能 – 产品特色

- 极大提升了Gurobi Python 矩阵 API，使用更友好
  - 所有矩阵对象都支持多维度
  - 维度处理规则和 Numpy 规则保持一致，包括传播机制(broadcasting)
- 新增逻辑(logistic)函数广义约束
  - 更容易在MIP模型中增加逻辑函数做为约束
  - 逻辑函数有多种用途，包括生态学、统计学、机器学习、医药化工等领域
- .NET 的 NuGet 支持
  - 允许 .NET 用户直接从 NuGet 服务器下载 Gurobi
- 内存限制终止参数，允许优雅退出
  - 用户可以设定一个内存使用上限，一旦达到，可以优雅退出并获得最好的结果

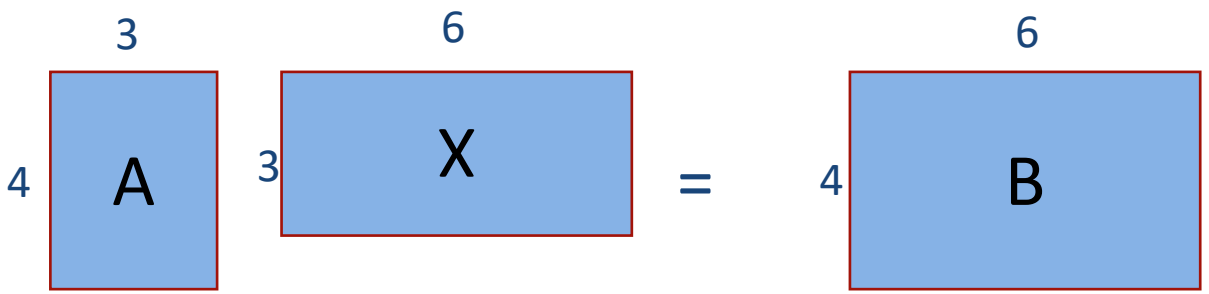
Gurobipy

多维度建模

- 到 9.5 为止，对于多维建模的支持是有限的
- 对于 10.0:
  - 所有 MVar, MLinExpr 和 MQuadExpr 支持任意维度
  - 从这些表达式添加到约束中会产生多维度 MConstr/MQConstr

2-D 线性约束

```
A = np.random.rand(4, 3)
B = np.random.rand(4, 6)
X = model.addMVar((3, 6))
model.setObjective(X.sum())
# Add 4*6=24 linear constraints
mc = model.addConstr(A @ X >= B)
```

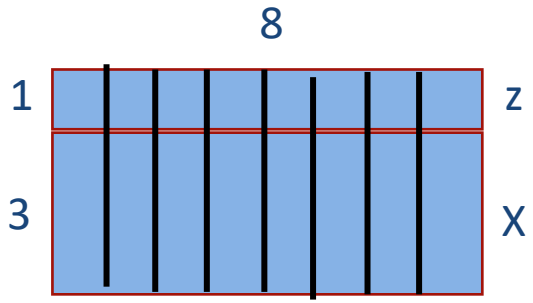


Gurobipy
多维度建模

- 到 9.5 为止，对于多维建模的支持是有限的
- 对于 10.0:
  - 所有 MVar, MLinExpr 和 MQuadExpr 支持任意维度
  - 从这些表达式添加到约束中会产生多维度 MConstr/MQConstr

1-D 二次约束

```
z = model.addMVar(8)
X = model.addMVar((8, 3), lb=-np.inf)
# Add eight standard cones
model.addConstr(z**2 >= (X**2).sum(axis=1))
```

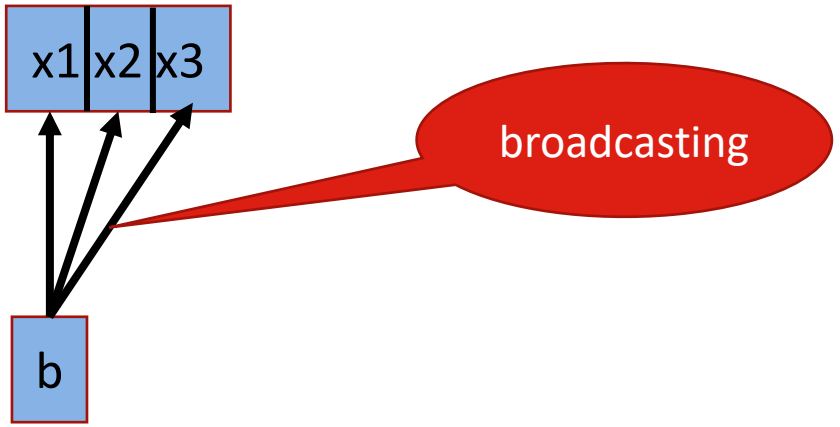


Gurobipy
传播 Broadcasting

- 到 9.5 为止：对于大多数操作, 维度要匹配
- 对于 10.0:
  - 所有 MVar, MLinExpr 和 MQuadExpr 支持传播
  - 支持 scalar, ndarray, SciPy sparse matrices, 和所有支持矩阵接口的对象

向量化的 VUB 约束

```
x = model.addMVar(3, ub=1.0)
b = model.addMVar((), vtype='B')
# three VUB constraints x[i] - b <= 0
model.addConstr(x - b <= 0)
```



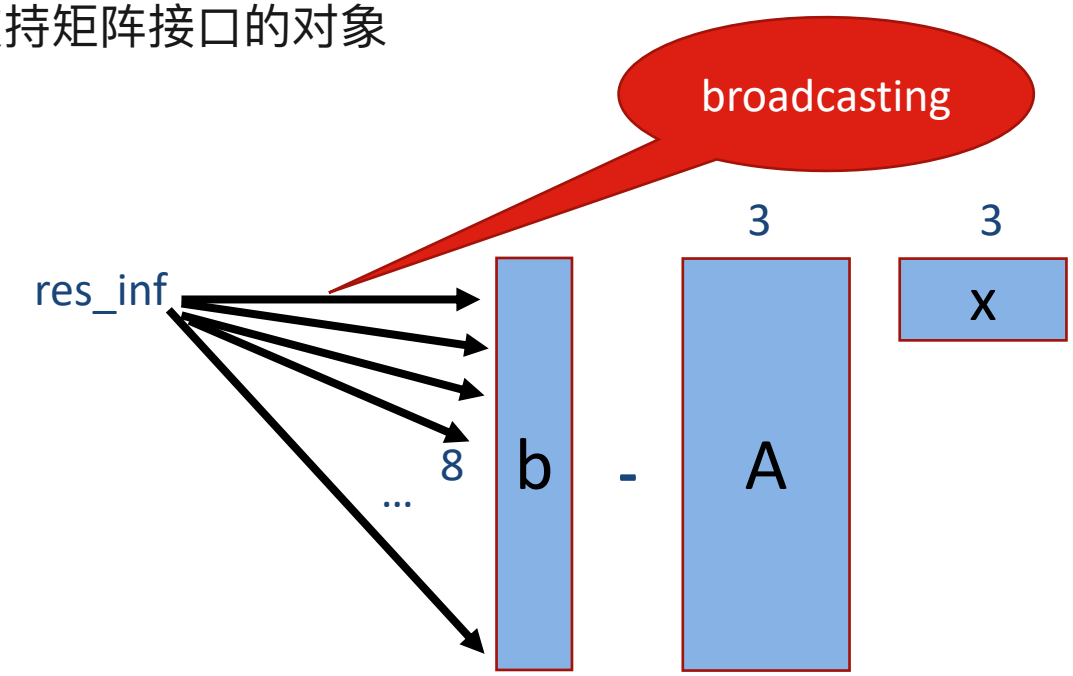
Gurobipy

传播 Broadcasting

- 到 9.5 为止：对于大多数操作, 维度要匹配
- 对于 10.0:
  - 所有 MVar, MLinExpr 和 MQuadExpr 支持传播
  - 支持 scalar, ndarray, SciPy sparse matrices, 和所有支持矩阵接口的对象

l_∞ regression

```
# l_inf regression
A = np.random.rand(8,3)
b = np.random.rand(8)
x = model.addMVar(3)
res_inf = model.addVar(obj=1)
# res_inf is broadcast upon adding constraint!
model.addConstr(res_inf >= b - A @ x)
model.addConstr(res_inf >= A @ x - b)
```



Gurobipy

其他矩阵接口功能和方法

- 概要
  - 对于经验丰富的 Numpy 用户操作结果的意外会大大减少
  - 支持矩阵相乘和按元素相乘二种乘法
- MVar
  - 从 MVar 类型变量 X, 提取对角线 : X.diagonal(offset).
  - 从一列常规变量 Var 转换成 MVar: x = MVar.fromlist(varlist)
  - 对 MVar 按照轴向进行加和: X.sum(axis=…)
  - 按照元素对 Mvar 变量进行平方: pow(X, 2), X**2
- MLinExpr
  - 全零表达式: MLinExpr.zeros(shape)
  - 对 MLinExpr 按照轴向进行加和: mle.sum(axis=…)
- 新类型 MQuadExpr
  - 用于多维二次约束建模
  - 与 MLinExpr 类似的方法和属性
- 新类型 MQConstr
  - 构造多维度二次约束时，调用 model.addConstr(…) 的返回类型
  - 与 MConstr 类似的方法和属性

逻辑函数广义约束 Logistic General Constraint

$$p(x) = \frac{1}{1 + e^{-(x)}}$$

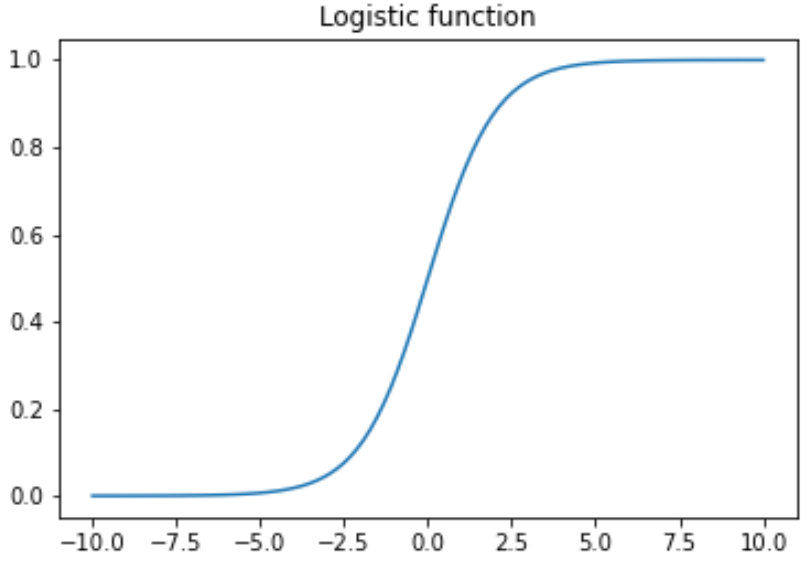
Gurobi 中的广义函数约束

允许定义 $y = f(x)$

- f 是预先定义的函数
- y 和 x 是一维变量

Gurobi 会自动在x 取值范围内对 f 进行分段线性化近似

将逻辑函数添加 f 函数集中



GUROBI
OPTIMIZATION

易学易用 – 开源 Github 应用库

Gurobi 10.0 – 开源 GitHub 应用库

易学易用

- Gurobi 机器学习
  - 允许用户将一个训练好的 机器学习 模型做为约束添加到 MIP 中
    - 比采用机器学习预测数据后输入优化模型更精巧
- Gurobipy 与 pandas 接口
  - 允许 gurobipy 模型更方便地利用 pandas 数据构造模型
- Gurobi 模型集锦*
  - 目标是让 不熟悉数学建模的用户 可以快速获得针对他们问题的结果
- 数值问题评估工具*
  - 允许用户分析有数值问题的模型并找到问题根源

*最终名字和发布时间仍需确认

Gurobipy 和 pandas

更方便用户使用受欢迎的数据分析包直接建立模型

- 创建以Gurobi 变量为基础的 pandas.DataFrame 结构
- 使用 pandas的操作运算符, 合并变量和数据, 组成约束
- 使用方法链; 所有输入和输出都是 pandas 对象
- 不需要在 pandas 和 gurobipy 之间转换

```
>>> x = (
...     pd.MultiIndex.from_product([items, knapsacks])
...     .grb.pd_add_vars(model, name='x', vtype=gp.GRB.BINARY)
... )
>>> x
item  knapsack
1     1           <gurobi.Var x[1,1]>
      2           <gurobi.Var x[1,2]>
2     1           <gurobi.Var x[2,1]>
      2           <gurobi.Var x[2,2]>
3     1           <gurobi.Var x[3,1]>
      2           <gurobi.Var x[3,2]>
```

```
>>> constr = (
...     x.groupby('item').sum()
...     .grb.pd_add_constrs(
           model, GRB.LESS_EQUAL, 1, name="c”
        )
... )
>>> constr
item
1    <gurobi.Constr c[1]>
2    <gurobi.Constr c[2]>
3    <gurobi.Constr c[3]>
```

Gurobi 机器学习

- 去年期间开发
- 在Gurobi 10.0 发布时向公众开放
- 开源 (Apache 2.0 License)
- 发布后会不断扩展 (基于用户反馈)
- 设计目标:
  - 轻巧
  - 表达简单
  - 容易扩展

我们目标

开发一个 python 软件包，能够:

1. 简化流程，可以让用流行ML软件训练的机器学习模型轻松嵌入到优化模型中
2. 提升算法性能，让优化模型可以在巨大解空间中搜索，找到能满足机器学习模型中建立的变量关系的解
3. 让优化模型更容易混合显性和隐性约束

灵感案例

- 美国牛油果销售
  - 我们的供应总量 S
  - 市场分割为不同区域集合为 R
  - 希望决定
    - 给每个区域的供货量 x_r
  - 最大化收益: 销售额 – 运输成本 – 未售出产品惩罚， 给定
    - 每个区域售价 p_r
    - 运输成本 c_r
    - 浪费惩罚 w
  - 每个区域的需求d_r
- 每个区域的需求 d_r 事先通过机器学习模型计算 (固定).

定价优化

- 定价 p_r 和需求 d_r 相关联
- 机器学习模型预测 d_r 可以建立这样的函数关系: $d_r = g(p_r)$
- 更现实的场景中，我们也希望能优化 p_r
- 为了能做到这点，我们需要找到能够把 $d_r = g(p_r)$ 插入到模型中的做法
- Notebook developed by J. Yurchisin and R. Swamy

能被识别的回归模型

GUROBI OPTIMIZATION

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
  - *One Hot encoder*
- pipelines to combine them

Keras

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

PyTorch

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

使用示范

- 假设我们利用scikit-learn 训练了如下回归模型:

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

- 嵌入到 Gurobi 模型中

```
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)
```


标杆测试

GUROBI OPTIMIZATION

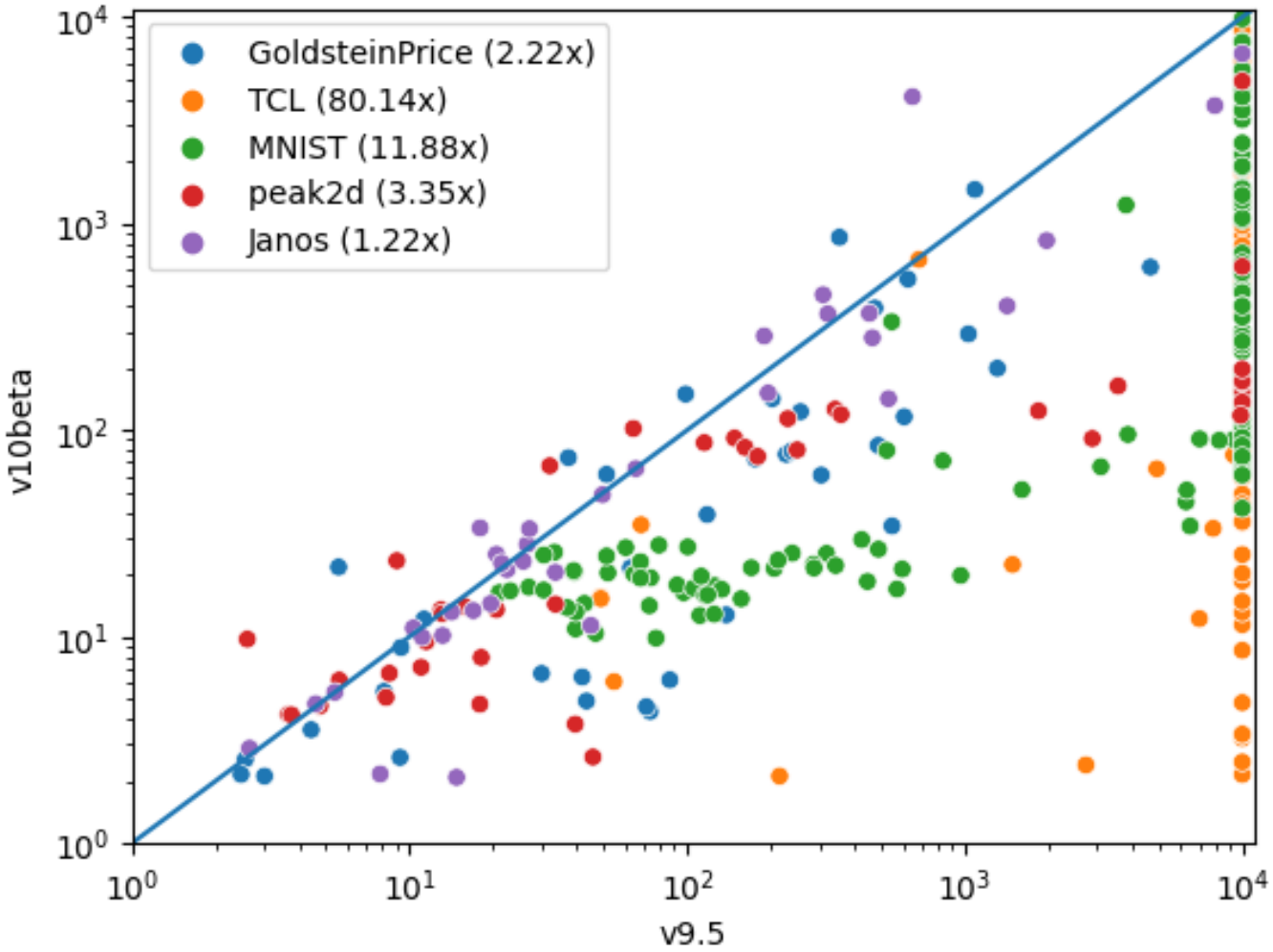
测试集

- 函数近似 Function approximation:
  - Goldstein-Price function (60 instances)
  - Peak function (60 instances)
- Janos (Bergman et.al. 2019): 500 predictor constraints with 3 features
- TCL (Amasyali et.al. 2022): 电气工程应用领域寻找界限范围内的输入/输出来最小化成本
- 对抗性机器学习 (Adversarial machine learning) MNIST: 119 instance trained by tensorflow and 90 trained by scikit-learn

运行环境

- Models solved on Intel(R) Xeon(R) CPU E3-1240 CPUs, 4 cores, 4 threads
- 时间限制在 10,000 秒
- 有逻辑回归的模型被排除
- 时间限制内没有被任何版本求解出的模型被排除

Gurobi 9.5 vs Gurobi 10.0



GUROBI
OPTIMIZATION

总结

总结和归纳

- Gurobi 对于性能提升的追求从来没有停止
  - 在历经50年发展后，LP 和 MILP 仍持续不断提升

Photo by Steve Harvey on Unsplash

  - MIQ(C)P 提升更为显著，潜力更大

总结和归纳

- 性能提升的同时，软件更加易学易用，促进了更多落地应用
  - 许可管理
  - Gurobi Python API 和其他 Python 模块库的融合
    - Pandas
    - Numpy, Scipy, etc.
    - SciKit-Learn, Keras, etc.
  - 诊断分析具有数值挑战性的模型

GUROBI
OPTIMIZATION

欢迎提问