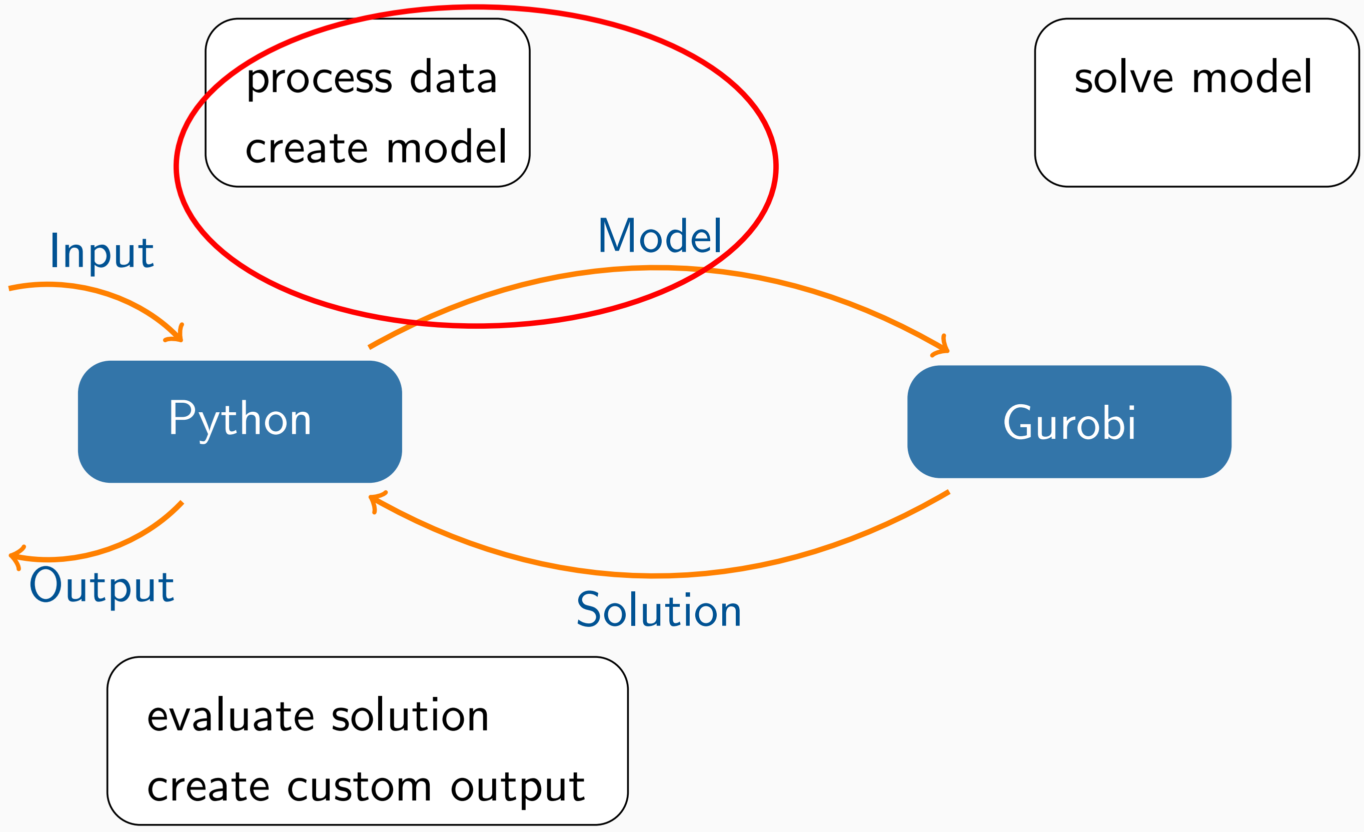
Advanced Gurobi Datatypes

Advanced Datatypes



Tuplelist

- subclass of python list
- list of tuples of same size
- easy notation to find specific subsets
- same is doable with list comprehension *but* tuplelist is faster

Tuplelist

Creation

```
l = tuplelist([(1, 2), (1, 3), (2, 3), (2, 4)])
```

Tuplelist

Creation

```
l = tuplelist([(1, 2), (1, 3), (2, 3), (2, 4)])
```

Queries ('*' is wildcard character)

```
l.select(1, '*')
l.select('*', [2, 4])
l.select('*', '*')
```

Tupledict

- subclass of python dictionary
- dictionary with tuplelist as keys
- usually used for variables of complex systems
- easy access via $select$
- easy constraint generation via sum and $prod$

Tupledict

Creation

```
l = tuplelist([(1, 2), (1, 3), (2, 3), (2, 4)])
d=model.addVars(l)
```

Tupledict

Creation

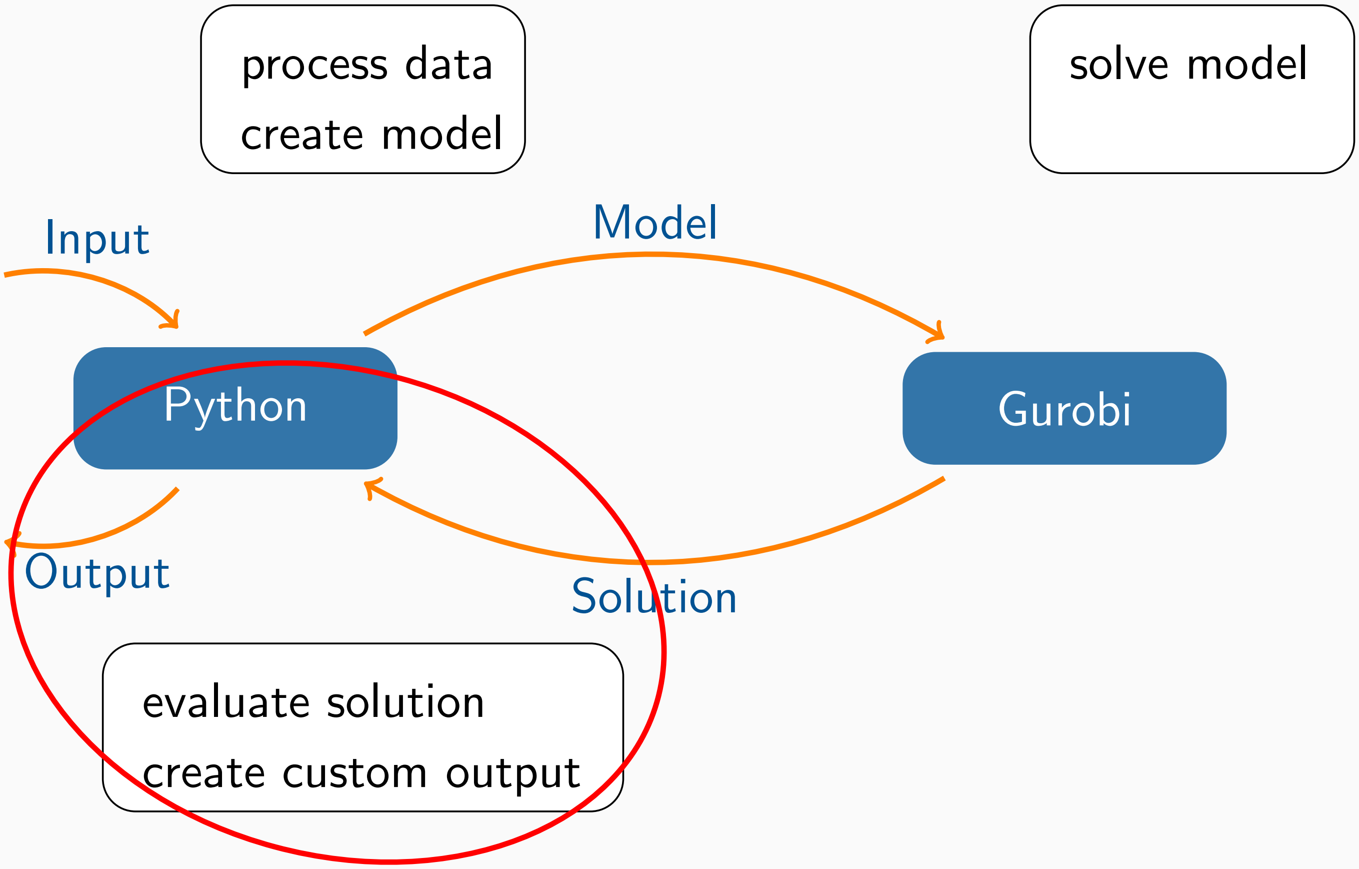
```
l = tuplelist([(1, 2), (1, 3), (2, 3), (2, 4)])
d=model.addVars(l)
```

Queries and creation of expressions

```
d.select(1, '*')
d.sum(1, '*')
coeff=[2,5]
d.prod(coeff, 1, '*')
```

Visualization

Result Visualization



process data

create model

solve model

Input

Model

Python

Gurobi

Output

Solution

evaluate solution

create custom output

Visualization of Graphs

- natural connection between many LPs and graphs
- visualizing graphs improves understanding
- matplotlib

Visualizing graphs

```
import networkx as nx
from random import randint
n = 10
position = [[randint(0,100),randint(0,100)]
    for i in range(n)]
edges = [(randint(0,n-1),randint(0,n-1))
    for i in range(3*n)]
G = nx.Graph()
G.add_nodes_from([(i, {'x': coord[0], 'y': coord[1]})
    for i, coord in enumerate(position)])
G.add_edges_from([(e[0], e[1]) for e in edges])
nx.draw_networkx_nodes(G, position, node_color="black")
nx.draw_networkx_edges(G, position,edge_color="black")
```