



# Gurobi Optimizer – Get the Software

## Gurobi Optimizer

Gurobi Optimizer is the Gurobi optimization libraries. In addition to the software, the corresponding README file contains installation instructions. [Here](#) is the list of bug fixes for each release.

Current version		64-bit Windows	64-bit Linux	64-bit macOS	64-bit AIX
9.1.0	<a href="#">README</a>	<a href="#">Gurobi-9.1.0-win64.msi</a>	<a href="#">gurobi9.1.0_linux64.tar.gz</a>	<a href="#">gurobi9.1.0_mac64.pkg</a>	<a href="#">gurobi9.1.0_power64.tar.gz</a>
md5 Checksum		5394eff3d8f5d8c16190f9ea5bc70020	832040cce622ba7f267e26645fc0d200d,	758713ea51b0981928f85d9bd81e6b27	948768b299de3d6c69653c7c0a0ed3a5
9.0.3	<a href="#">README</a>	<a href="#">Gurobi-9.0.3-win64.msi</a>	<a href="#">gurobi9.0.3_linux64.tar.gz</a>	<a href="#">gurobi9.0.3_mac64.pkg</a>	<a href="#">gurobi9.0.3_power64.tar.gz</a>
md5 Checksum		5394eff3d8f5d8c16190f9ea5bc70020	832040cce622ba7f267e26645fc0d200d,	758713ea51b0981928f85d9bd81e6b27	948768b299de3d6c69653c7c0a0ed3a5

## Old versions

8.1.1	<a href="#">README</a>	<a href="#">Gurobi-8.1.1-win64.msi</a>	<a href="#">gurobi8.1.1_linux64.tar.gz</a>	<a href="#">gurobi8.1.1_mac64.pkg</a>	<a href="#">gurobi8.1.1_power64.tar.gz</a>
md5 Checksum		17dfc21f0ed64daaa4bdf7634eab705b	05ccb96072e393bd4ebb1d8b9526ce01	d05a73c0df6622851b4371dc1d292579	3d1a756695d52065eeefc15516d9aac6
8.0.1	<a href="#">README</a>	<a href="#">Gurobi-8.0.1-win64.msi</a>	<a href="#">gurobi8.0.1_linux64.tar.gz</a>	<a href="#">gurobi8.0.1_mac64.pkg</a>	<a href="#">gurobi8.0.1_power64.tar.gz</a>
md5 Checksum		d9363f13daa63b79c0cd0aa37ad92e8b6	cfc595ddf9482734bdc0268749093cc4	a02d04ef884e64e7091ef7a7439cfe68	877f94a02e602346ee767b9894df4030

# License Details

Information and installation instructions

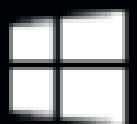
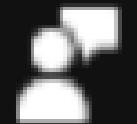
License ID	516013
Date issued	2020-10-28T22:25:41
Purpose	Trial
License Type	TRIAL
Key Type	TRIAL
Version	9
Expiration Date	2021-04-26
Distributed Limit	0
Host Name	
Host ID	

## Installation

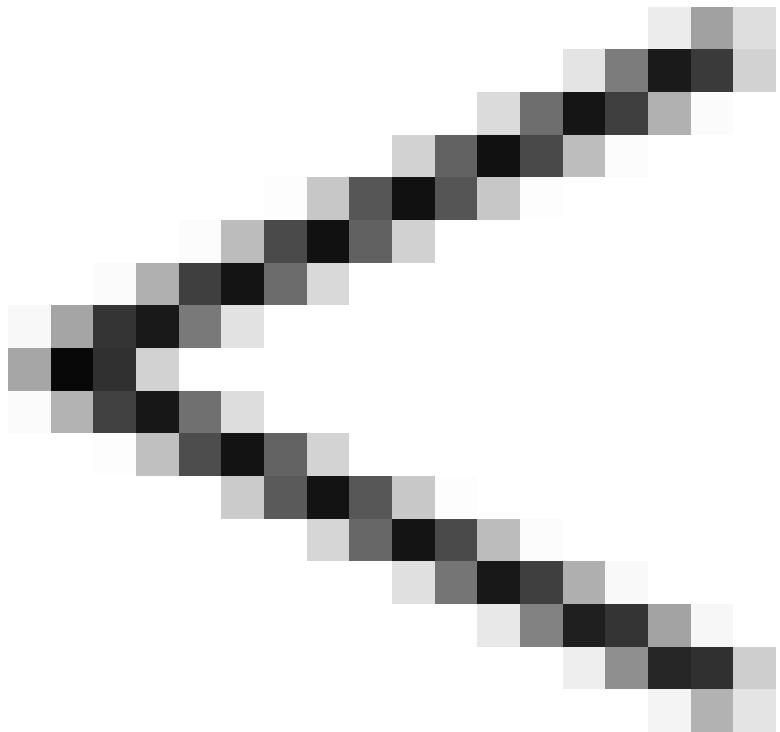
To install this license on a computer where Gurobi Optimizer is installed, copy and paste the following command to the Start/Run menu (Windows only) or a command/terminal prompt (any system):

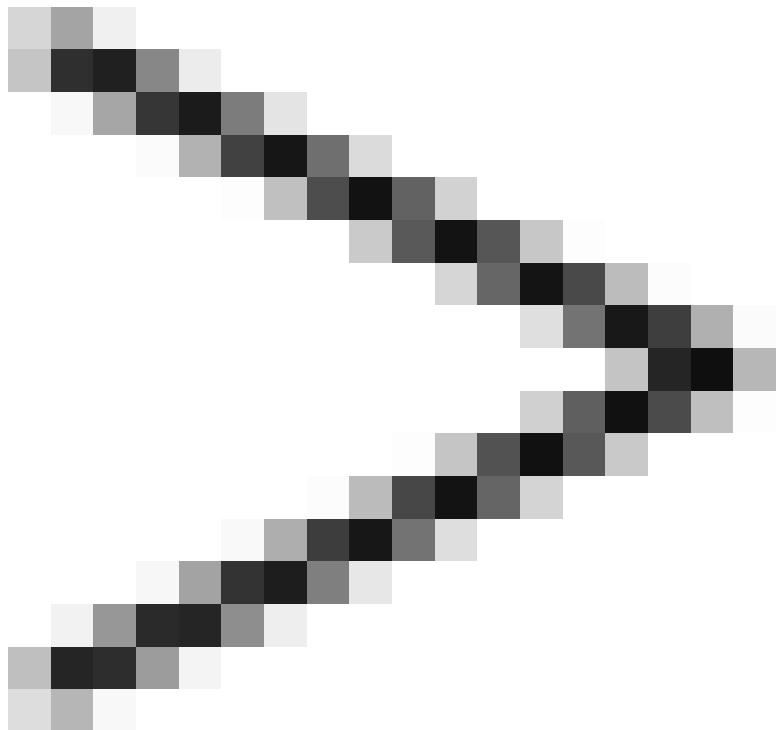
```
grbgetkey 83af988a-196c-11eb-865d-0a7c4f30bdbe
```

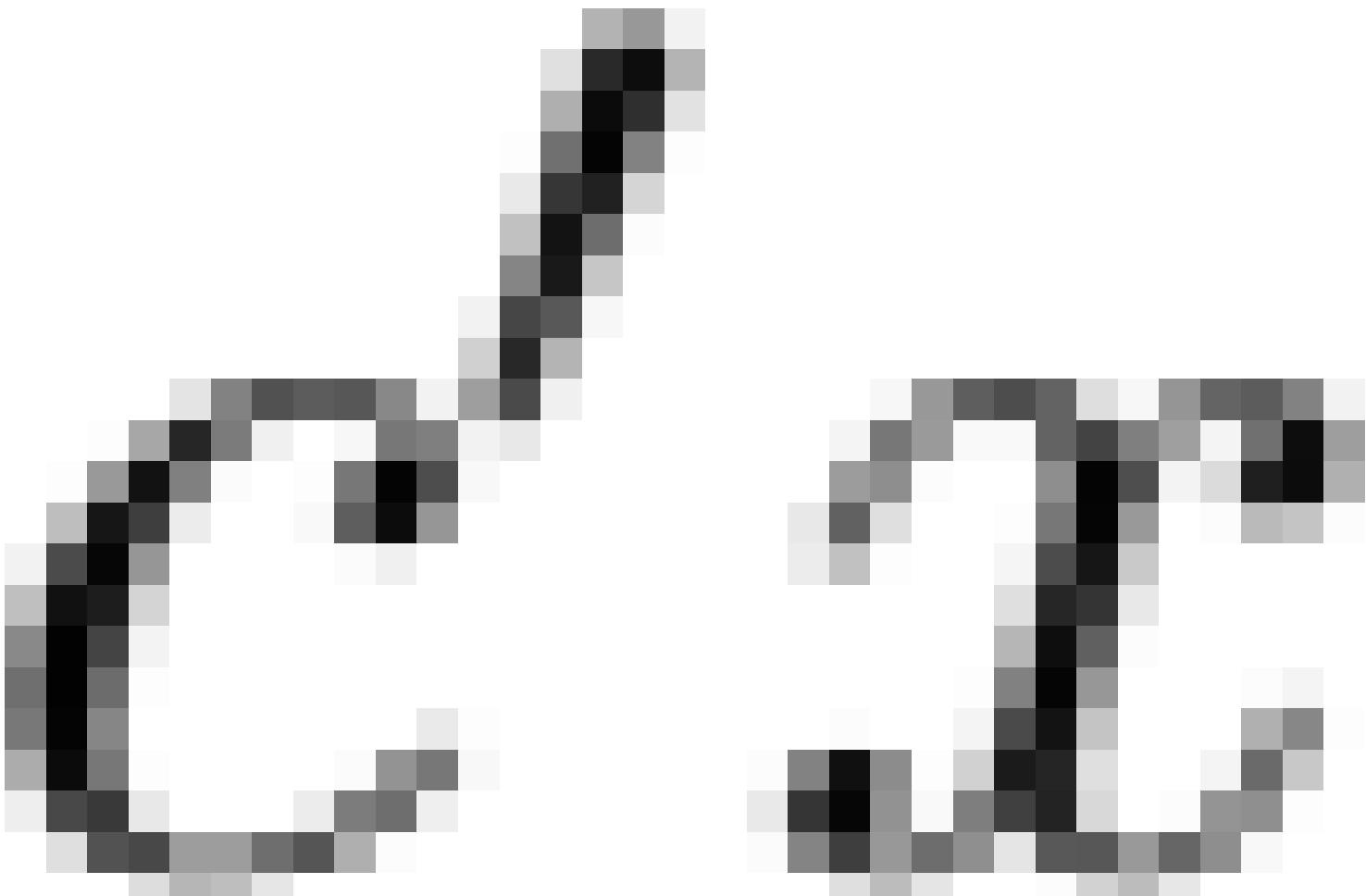
The `grbgetkey` command requires an active internet connection. If your computer has no internet access, or you get no response or an error message such as "Unable to contact key server", [Please click here for additional instructions](#).

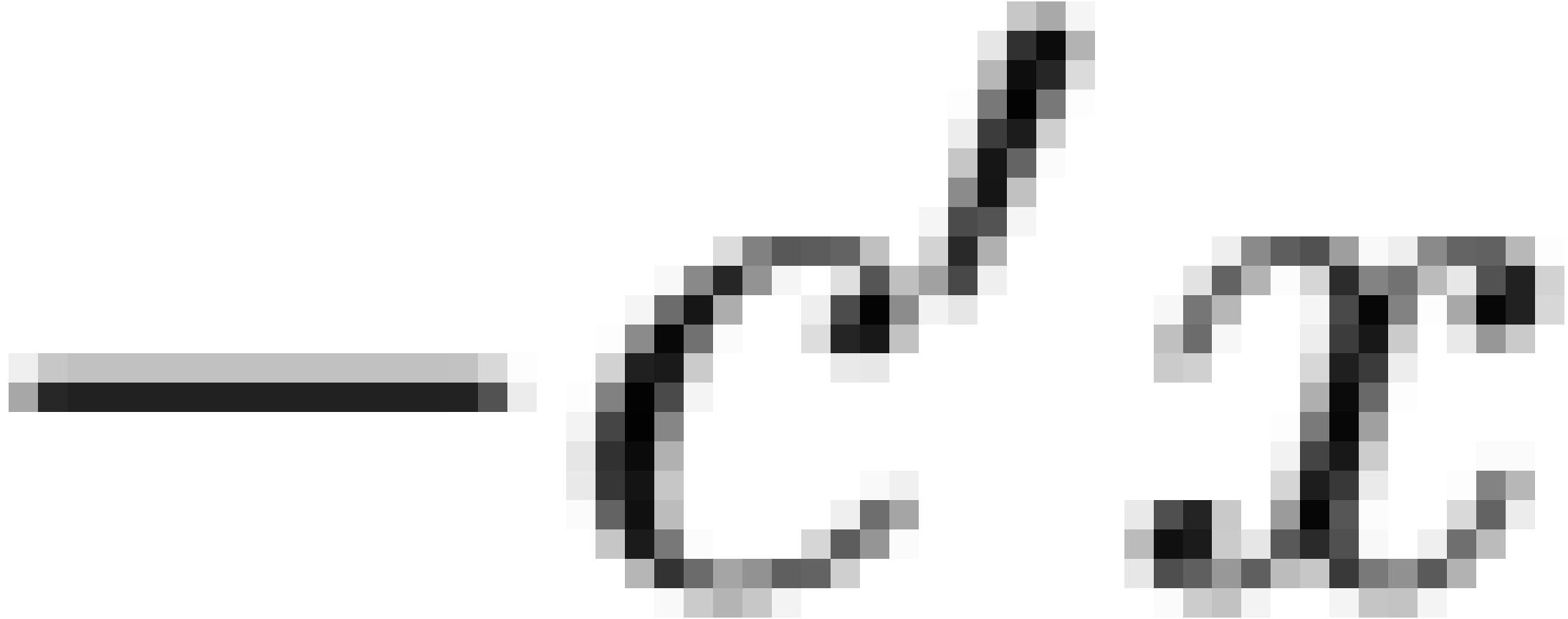


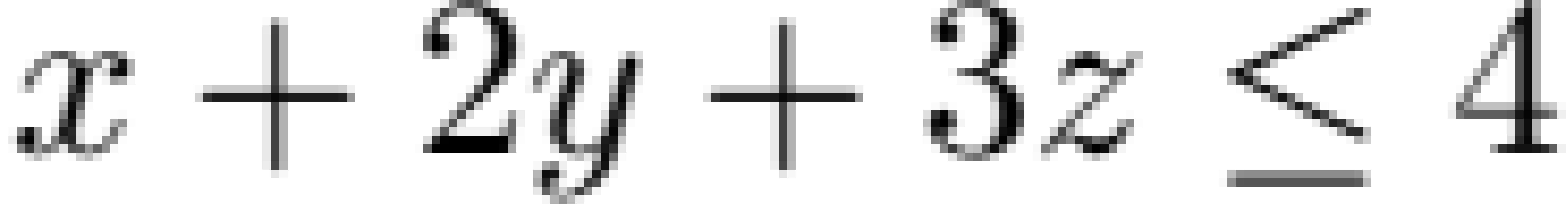
grbgetkey bba60259-a126-e14f-dab2-580a56ac4d2e|

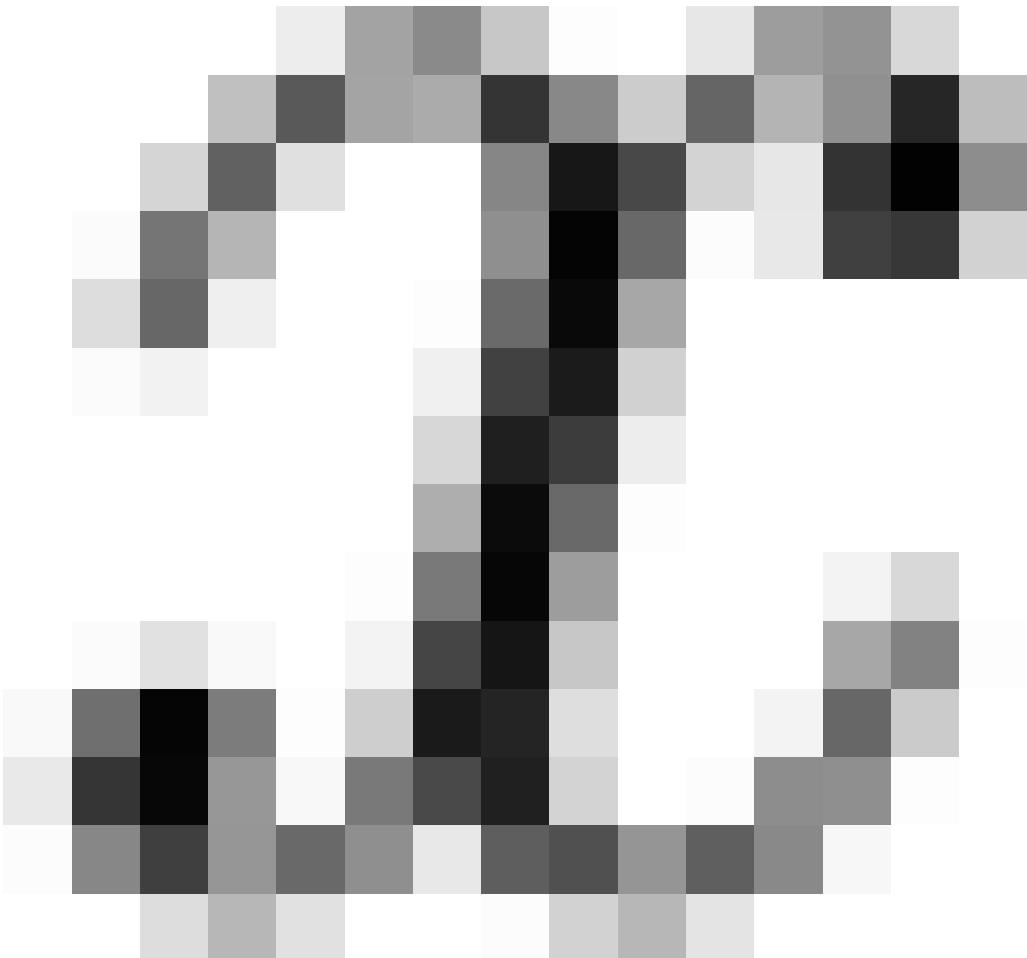


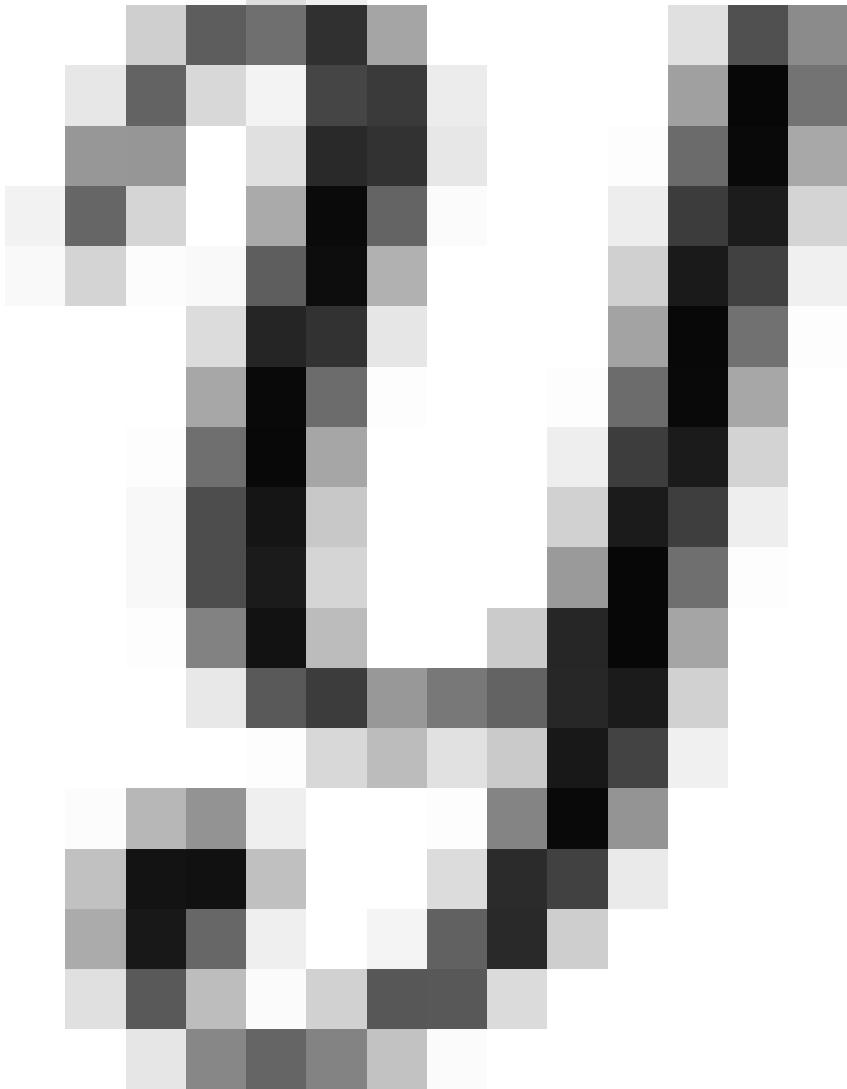


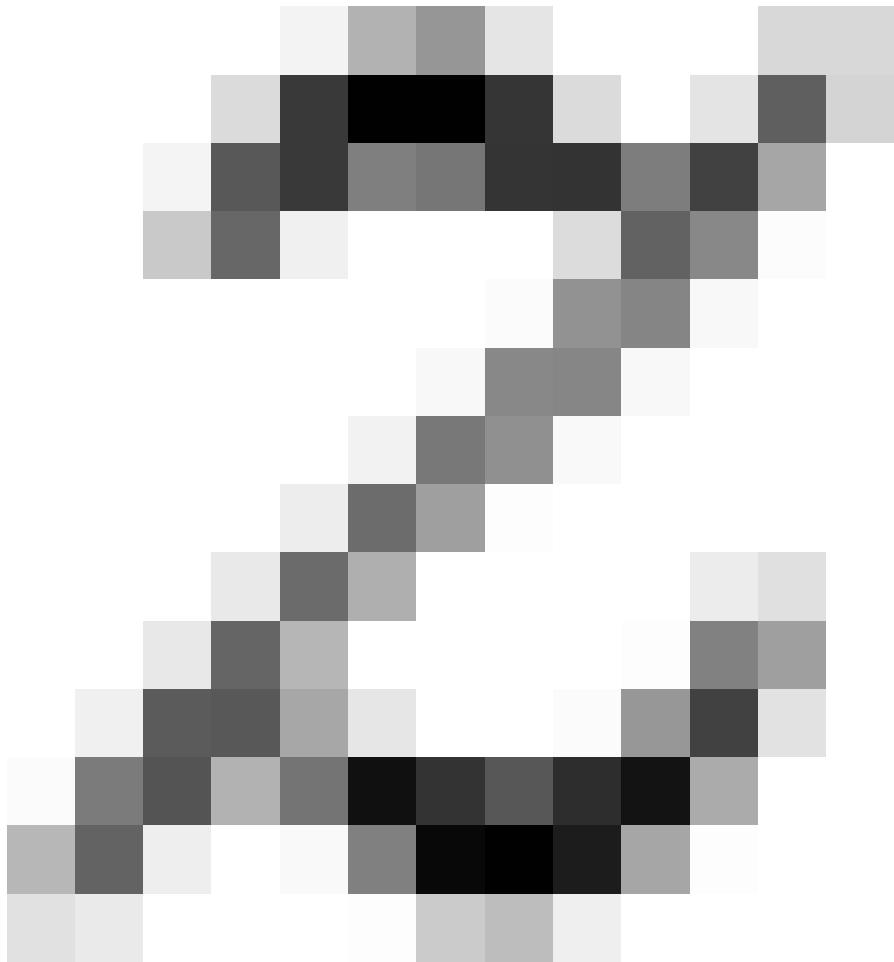












# Anaconda Installers

Windows 

Python 3.8

[64-Bit Graphical Installer \(466 MB\)](#)

[32-Bit Graphical Installer \(397 MB\)](#)

MacOS 

Python 3.8

[64-Bit Graphical Installer \(462 MB\)](#)

[64-Bit Command Line Installer \(454 MB\)](#)

Linux 

Python 3.8

[64-Bit \(x86\) Installer \(550 MB\)](#)

[64-Bit \(Power8 and Power9\) Installer \(290 MB\)](#)

Supercharge your data science  
efforts with **Anaconda**.

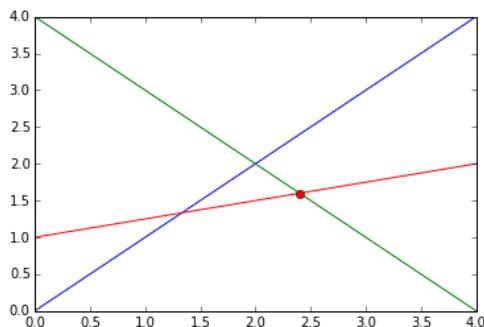
[Get Started](#)



```
In [45]: from gurobipy import *
m = Model()
v0 = m.addVar()
v1 = m.addVar()
m.update()
m.addConstr(v0 - v1 <= 4) # Constraint 1
m.addConstr(v0 + v1 <= 4) # Constraint 2
m.addConstr(.25*v0 + v1 <= 1) # Constraint 3
m.setObjective(v1, GRB.MAXIMIZE) # Objective: maximize v1
m.params.outputflag = 0
m.optimize()
```

```
Plot the optimal solution...
```

```
In [46]: import matplotlib.pyplot as pyplot
pyplot.plot([0,4], [0,4]) # Constraint 1
pyplot.plot([4,0], [0,4]) # Constraint 2
pyplot.plot([0,4], [1,2]) # Constraint 3
pyplot.plot([v0.x], [v1.x], 'ro') # Plot the optimal vertex
pyplot.show()
```



```
In [ ]:
```



Editor - /home/daesp... IPython console

temp.py\*

Console 1/A

```
In [1]: from gurobipy import *
In [2]: m = read('/opt/gurobi900/linux64/examples/data/p0033.mps')
Using license file /opt/gurobi900/gurobi.lic
Read MPS format model from file /opt/gurobi900/linux64/examples/data/p0033.mps
Reading time = 0.01 seconds
P0033: 16 rows, 33 columns, 98 nonzeros
```

```
In [3]: m.optimize()
Gurobi Optimizer version 9.0.0 build v9.0.0rc0 (linux64)
Optimize a model with 16 rows, 33 columns and 98 nonzeros
Model fingerprint: 0x0adb1647
Variable types: 0 continuous, 33 integer (0 binary)
Coefficient statistics:
    Matrix range      [1e+00, 4e+02]
    Objective range   [5e+01, 5e+02]
    Bounds range     [1e+00, 1e+00]
    RHS range        [1e+00, 3e+03]
Found heuristic solution: objective 3828.0000000
Presolve removed 5 rows and 14 columns
Presolve time: 0.00s
Presolved: 11 rows, 19 columns, 71 nonzeros
Found heuristic solution: objective 3089.0000000
Variable types: 0 continuous, 19 integer (16 binary)
```

Root relaxation: objective 2.839492e+03, 10 iterations, 0.00 seconds

Nodes		Current Node		Objective Bounds			Work	
Expl	Unexpl	Obj	Depth	IntInf	Incumbent	BestBd	Gap	It/Node Time
0	0	2839.49184	0	3	3089.00000	2839.49184	8.08%	- 0s
0	0	2941.40000	0	1	3089.00000	2941.40000	4.78%	- 0s
0	0	2952.00000	0	1	3089.00000	2952.00000	4.44%	- 0s
0	0	3045.27500	0	5	3089.00000	3045.27500	1.42%	- 0s
0	0	3089.00000	0	7	3089.00000	3089.00000	0.00%	- 0s

Cutting planes:

Gomory: 3  
MIR: 1

Explored 1 nodes (24 simplex iterations) in 0.04 seconds

IPython console History log

## Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /opt/gurobi900/linux64/examples/python/mip1.py

(mip1.py) X

```

1 #!/usr/bin/env python3.7
2
3 # Copyright 2019, Gurobi Optimization, LLC
4
5 # This example formulates and solves the following simple MIP model
6 # maximize
7 #     x + y + 2 z
8 # subject to
9 #     x + 2 y + 3 z <= 4
10 #    x + y           >= 1
11 #    x, y, z binary
12
13 import gurobipy as gp
14 from gurobipy import GRB
15
16 try:
17
18     # Create a new model
19     m = gp.Model("mip1")
20
21     # Create variables
22     x = m.addVar(vtype=GRB.BINARY, name="x")
23     y = m.addVar(vtype=GRB.BINARY, name="y")
24     z = m.addVar(vtype=GRB.BINARY, name="z")
25
26     # Set objective
27     m.setObjective(x + y + 2 * z, GRB.MAXIMIZE)
28
29     # Add constraint: x + 2 y + 3 z <= 4
30     m.addConstr(x + 2 * y + 3 * z <= 4, "c0")
31
32     # Add constraint: x + y >= 1
33     m.addConstr(x + y >= 1, "c1")
34
35     # Optimize model
36     m.optimize()
37
38     for v in m.getVars():

```

IPython console

Console 1/A X

In [4]: runfile('/opt/gurobi900/linux64/examples/python/mip1.py', wdir='/opt/gurobi900/linux64/examples/python')  
Gurobi Optimizer version 9.0.0 build v9.0.0rc0 (linux64)  
Optimize a model with 2 rows, 3 columns and 5 nonzeros  
Model fingerprint: 0xb2adf8c4  
Variable types: 0 continuous, 3 integer (3 binary)  
Coefficient statistics:  
Matrix range [1e+00, 3e+00]  
Objective range [1e+00, 2e+00]  
Bounds range [1e+00, 1e+00]  
RHS range [1e+00, 4e+00]  
Found heuristic solution: objective 2.0000000  
Presolve removed 2 rows and 3 columns  
Presolve time: 0.00s  
Presolve: All rows and columns removed

Explored 0 nodes (0 simplex iterations) in 0.02 seconds  
Thread count was 1 (of 4 available processors)

Solution count 2: 3

Optimal solution found (tolerance 1.00e-04)  
Best objective 3.00000000000e+00, best bound 3.00000000000e+00,  
gap 0.0000%  
x 1  
y 0  
z 1  
Obj: 3

In [5]:

IPython console

History log

Permissions: R

End-of-lines: LF

Encoding: ASCII

Line: 1 Column: 1 Memory: 50 %

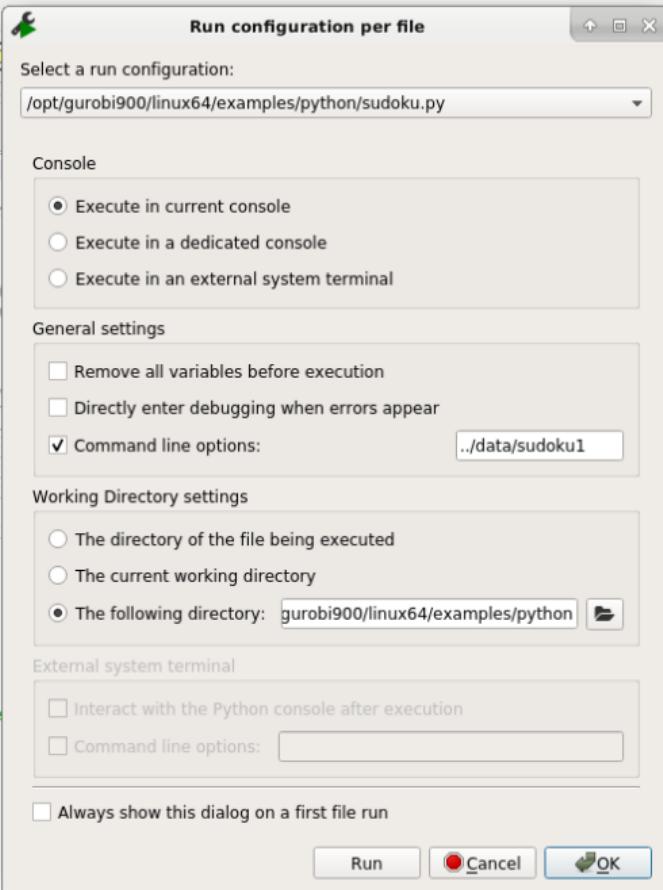
Editor - /opt/gurobi900/linux64/examples/pyt

(sudoku.py) X

```

1 #!/usr/bin/env python3.7
2
3 # Copyright 2019, Gurobi Optimization
4
5 # Sudoku example.
6
7 # The Sudoku board is a 9x9 grid
8 # of 3x3 grids. Each cell in the
9 # No two grid cells in the same
10 # same value.
11 #
12 # In the MIP formulation, binary
13 # cell <i,j> takes value 'v'.
14 #   1. Each cell must take exactly
15 #   2. Each value is used exactly
16 #   3. Each value is used exactly
17 #   4. Each value is used exactly
18 #
19 # Input datasets for this example
20
21 import sys
22 import math
23 import gurobiipy as gp
24 from gurobiipy import GRB
25
26
27 if len(sys.argv) < 2:
28     print('Usage: sudoku.py file')
29     quit()
30
31 f = open(sys.argv[1])
32
33 grid = f.read().split()
34
35 n = len(grid[0])
36 s = int(math.sqrt(n))
37
38

```



xamples/python  
@ build v9.0.0rc0 (linux64)  
3 columns and 5 nonzeros  
3 integer (3 binary)  
00]  
00]  
00]  
00]  
ctive 2.0000000  
columns  
removed  
erations) in 0.02 seconds  
able processors)  
nace 1.00e-04)  
+00, best bound 3.000000000000e+00,

IPython console History log

Permissions: R

End-of-lines: LF

Encoding: ASCII

Line: 1 Column: 1 Memory: 51 %

