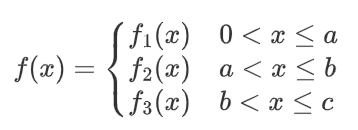
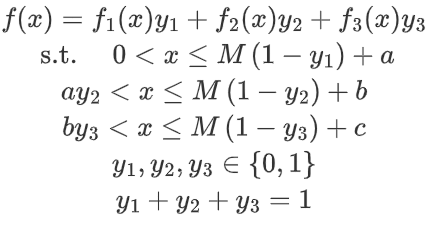
# ****分段函数****

分段函数f（x）的数学表达式如下：



如果要对分段函数f（x）进行线性化处理，则需要引入0-1辅助变量y和一个很大的正整数M。



【举例】将下列分段函数线性化

min f(x)



如果要对分段函数f（x）进行线性化处理，则需要引入0-1辅助变量y和一个很大的正整数M。化简后的模型如下：



现在我们用python+gurobi对上述模型进行求解，代码如下：

from gurobipy import \*

model = Model('non-linear model')

x = model.addVar(lb=0, ub=GRB.INFINITY, vtype=GRB.CONTINUOUS, name='x')

y1 = model.addVar(lb=0, ub=1, vtype=GRB.BINARY, name='y1')

y2 = model.addVar(lb=0, ub=1, vtype=GRB.BINARY, name='y2')

y3 = model.addVar(lb=0, ub=1, vtype=GRB.BINARY, name='y3')

bigM = 10000

epsilon = 1e-5

model.setObjective((-1 \* x + 2) \* y1 + (-0.5 \* x + 1.25) \* y2 + (x - 3) \* y3, GRB.MINIMIZE)

# 修正约束

model.addConstr(x <= bigM \* (1 - y1) + 1, name='c1')

model.addConstr(x >= y1 + epsilon, name='c1a') # 当y1=1时，x至少为1

model.addConstr(x <= bigM \* (1 - y2) + 3, name='c2')

model.addConstr(x >= y2 + epsilon, name='c2a') # 当y2=1时，x至少为3

model.addConstr(x <= bigM \* (1 - y3) + 5, name='c3')

model.addConstr(x >= 3 \* y3 + epsilon, name='c3a') # 当y3=1时，x至少为5

model.addConstr(y1 + y2 + y3 == 1, name='c4')

model.setParam(GRB.Param.LogFile, 'gurobi.log')

modelName = 'nlTol1.lp'

model.write(modelName)

model.optimize( )

solName = 'nlTol1.sol'

model.write(solName)

print('x:', x.x)

print('y1:', y1.x)

print('y2:', y2.x)

print('y3:', y3.x)

# ****绝对值函数****

绝对值函数的数学表达式如下：

引入ui和vi替代|xi|



因此原模型线性化为如下的模型

【举例】将下列分段函数线性化

min |x1|+|x2|

s.t. x1+x2=0

引入u1,v1,u2,v2分别替代和，则原模型转化为：

很明显，当x1=0且x2=0时，f（x）取最小值。

python+gurobi对上述线性模型进行求解，代码如下：

from gurobipy import \*  
  
model = Model('non-linear model')  
u1 = model.addVar(lb=0, ub=GRB.INFINITY, vtype=GRB.CONTINUOUS, name='u1')  
u2 = model.addVar(lb=0, ub=GRB.INFINITY, vtype=GRB.CONTINUOUS, name='u2')  
v1 = model.addVar(lb=0, ub=GRB.INFINITY, vtype=GRB.CONTINUOUS, name='v1')  
v2 = model.addVar(lb=0, ub=GRB.INFINITY, vtype=GRB.CONTINUOUS, name='v2')  
  
model.setObjective(u1+v1+u2+v2, GRB.MINIMIZE)  
  
model.addConstr(u1-v1+u2-v2 == 0, name='c1')  
  
model.setParam(GRB.Param.LogFile, 'gurobi.log')  
  
modelName = 'nlTol2.lp'  
model.write(modelName)  
model.optimize()  
solName = 'nlTol2.sol'  
model.write(solName)  
  
print('u1:', u1.x)  
print('v1:', v1.x)  
print('u2:', u2.x)  
print('v2:', v2.x)

# ****MaxMin/MinMax函数****

将函数线性化后的表达式如下：



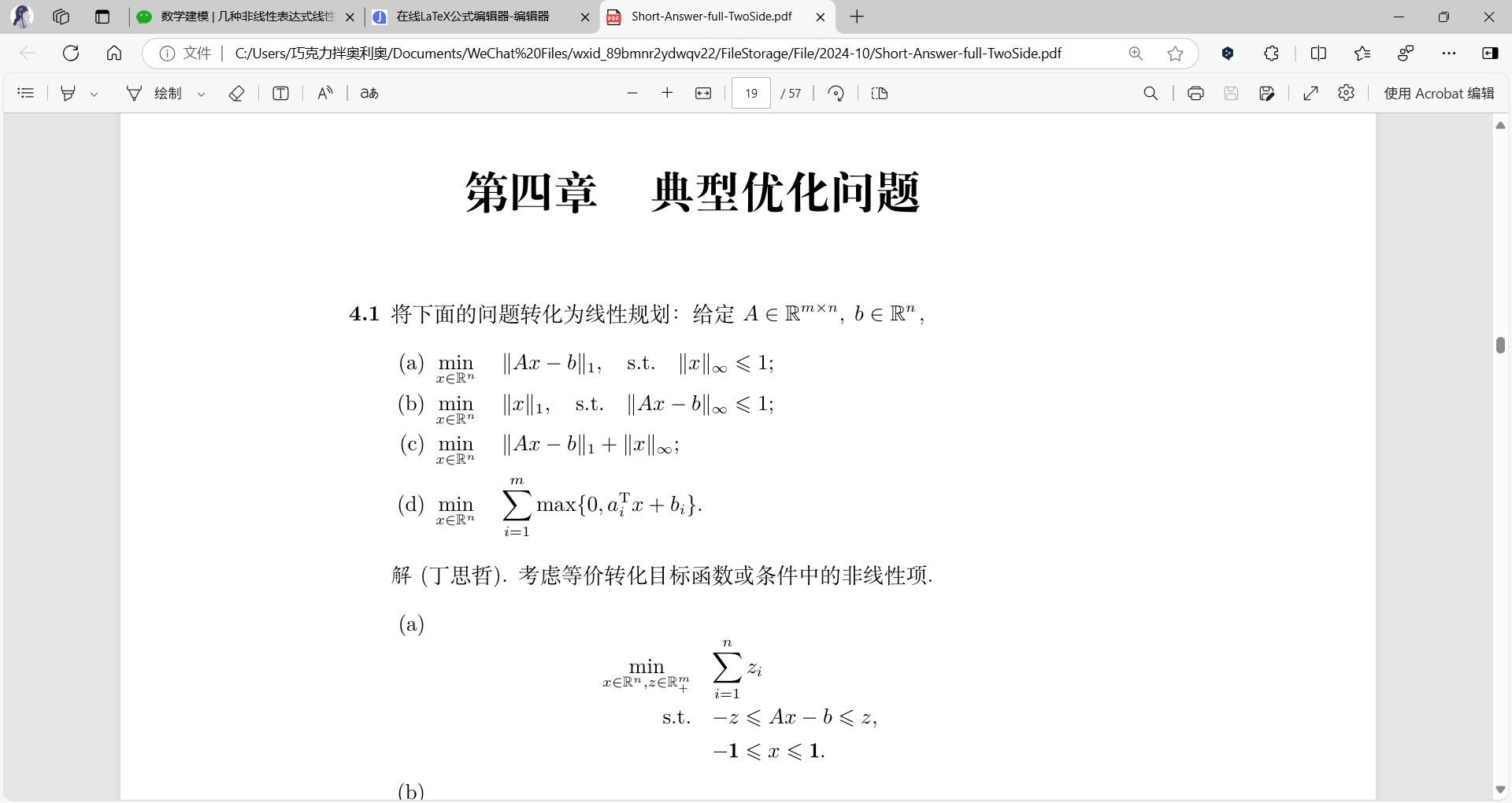
该形式下的线性化方法是：用z替代函数，这样，又由目标函数是取最大的z，从而限制住z会无限小下去，而只会取满足条件的z。

同理，将函数线性化后的表达式如下：

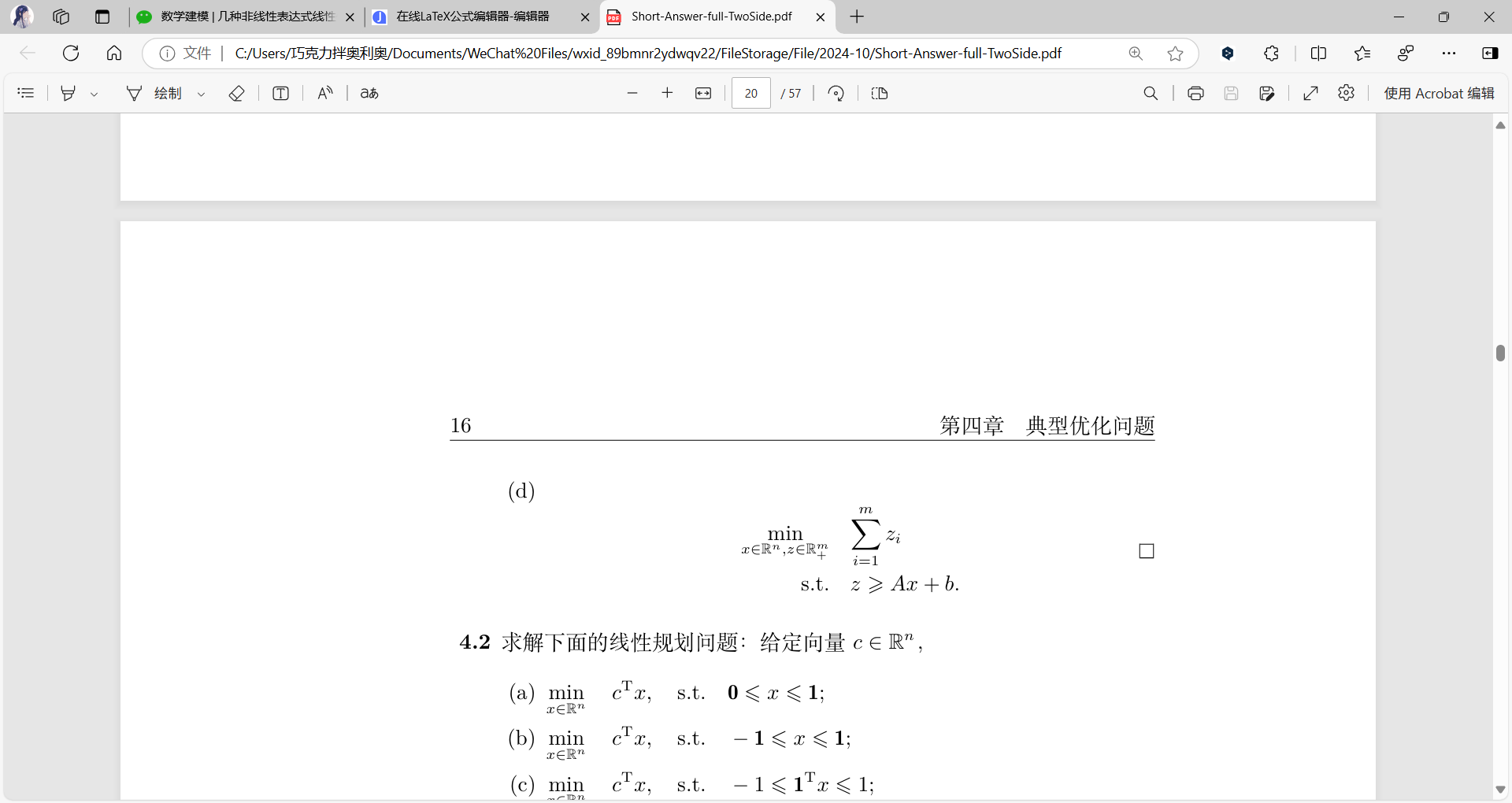


【举例】将max（min x+2y+10,3x+y+1）函数线性化后的表达式如下：





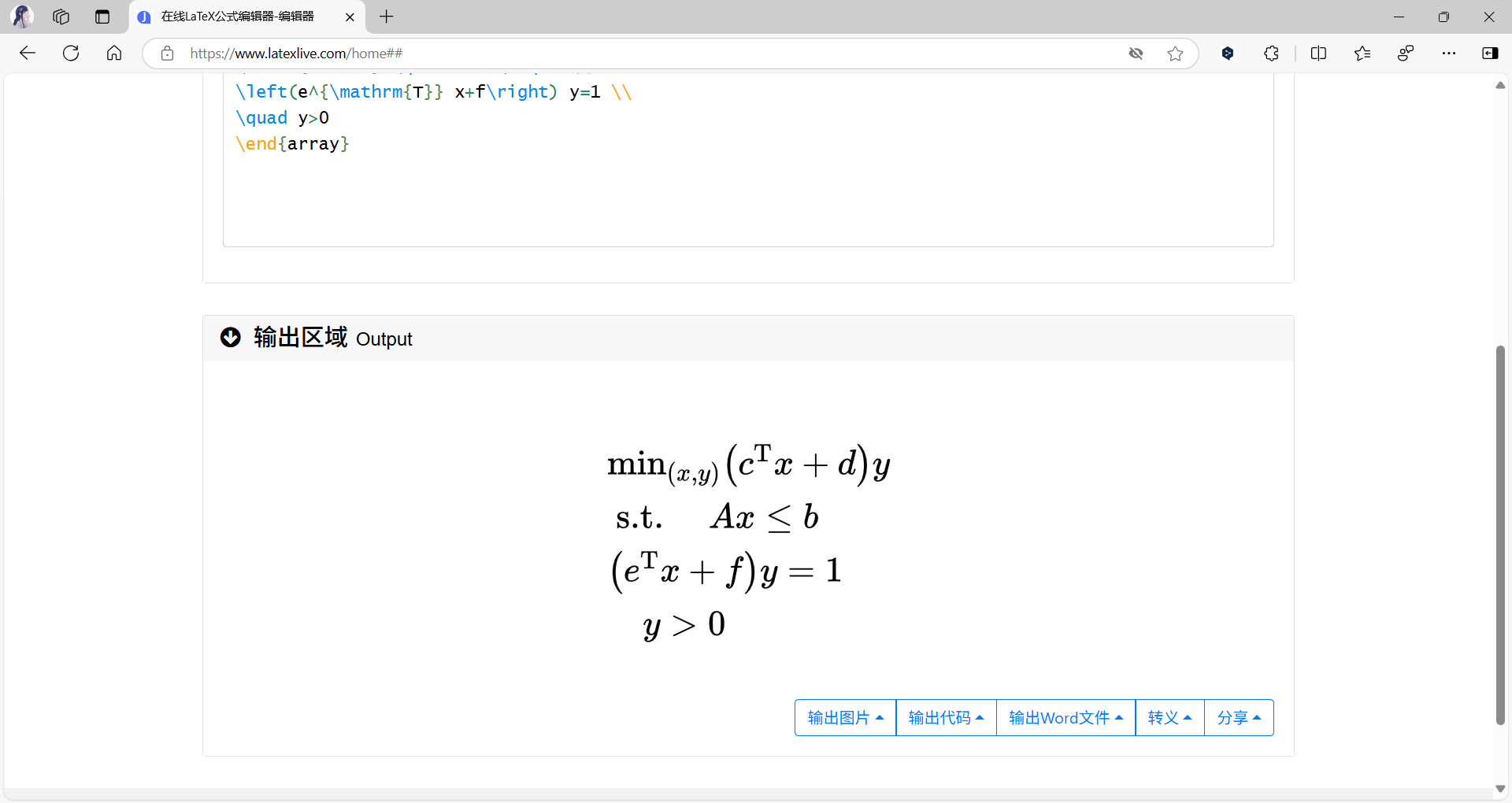
对于（d），令z=max{0，aiTx+bi}，这样zaiTx+bi,又由目标函数是取最大的z，从而限制住z会无限小下去，而只会取满足条件的z, 解得结果为



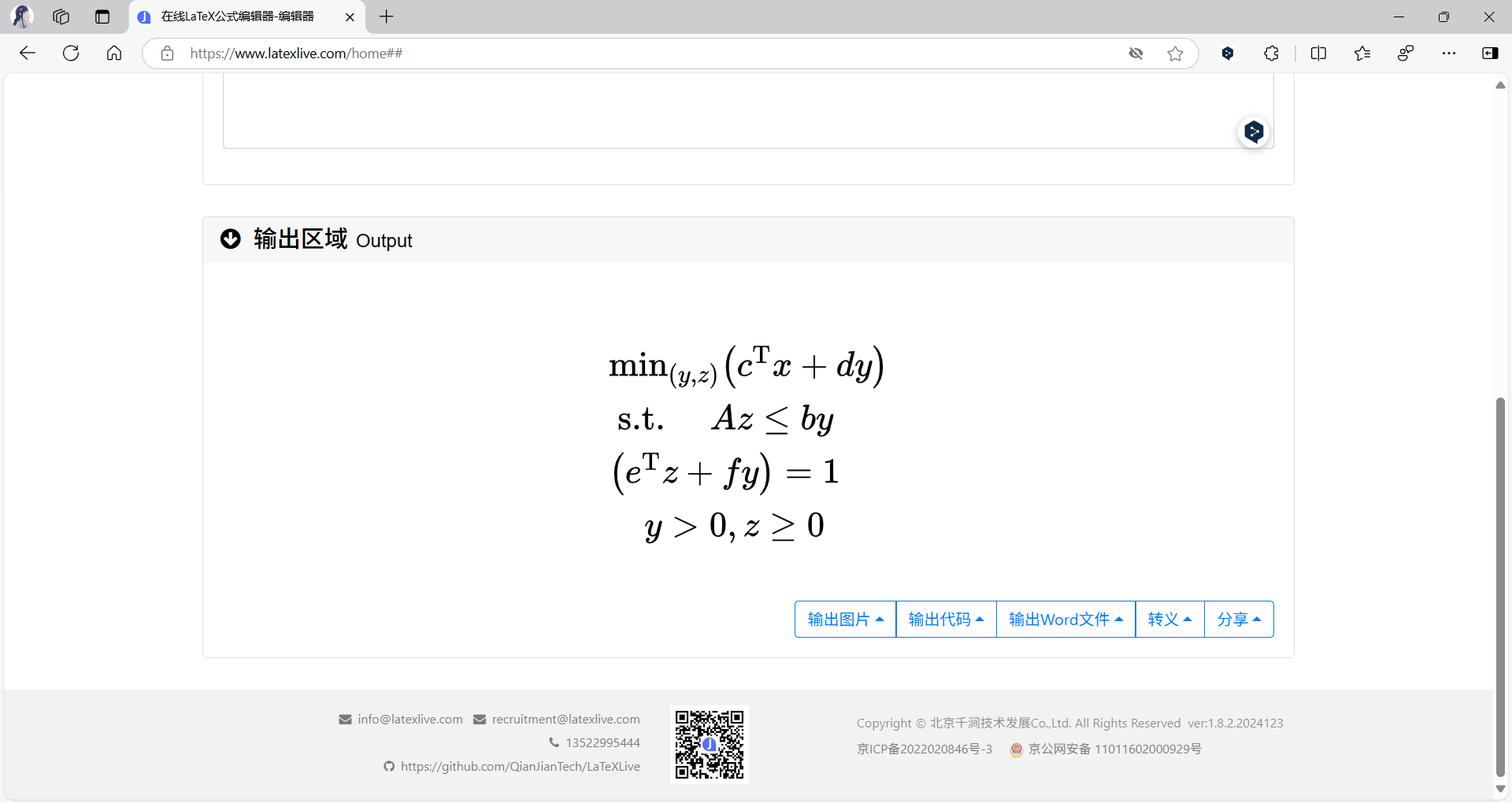
# ****分式目标函数****

分式目标函数表达式形式：

令带入上面可得：



再令z=xy代入上式可得：



令，则目标函数转化为

观察如上形式，又出现新的非线性函数：xz和yz。此时若令xz=u，yz=v，则约束条件也需要做出相应的改变（约束两边同乘z）：

可见，分式目标函数的线性化有两个步骤。一是将分母的倒数设置为新的线性参数；二是令非线性的xz 和 yz 等于新的线性参数。