1 Introduction

Solving constrained non linear optimization problems begins with identification of the Karush-Kuhn-Tucker conditions. This is due to the fact that for a differentiable convex system where constraint qualification holds, the solution of NLP is the solution to the KKT system and vise versa. The KKT conditions enables solving the NLP model as an MCP model by representing the KKT conditions as complimentarily equations. Thus, it is often advantageous to explicitly model the KKT conditions and solve them as a Mixed Complementarity Problem (MCP), as described in the PATH Solver Manual (/link) using the example of a Linear Transportation model. The traditional model of fixed demand and price is made more realistic by making the variables endogenous, i.e. the price of commodity is market affects the demand of the commodity, and the model is solved as a complimentarily problem. At times the complexity of the model causes errors in formulation of KKT conditions, leading to incorrect or infeasible solutions. Finding the source of the error can be a particularly tedious task. Thus, in this article, we propose a systematic method for identification of errors in the explicit KKT conditions for solving an NLP. .

We begin by understanding the definition of complementarity. If a function $F(z) : R^n \mapsto R^n$, lower bounds $l \in R \cup -\infty^n$ and upper bounds $u \in R \cup \infty^n$ has a solution vector $z \in R$ such that for each $i \in 1, ..., n$, one of the following three conditions hold :

$$F_i(z) = 0 \text{ and } l_i \leq z_i \leq u_i \text{ or}$$
$$F_i(z) > 0 \text{ and } z_i = l_i \text{ or}$$
$$F_i(z) < 0 \text{ and } z_i = u_i$$

then function F is complementary to the variable z and its bounds. This is written in compact form as

$$F(z) \perp L \leq z \leq U$$

Where the symbol $\perp$ means "perpendicular to". From point of view of optimization, if $F(z)$ is non zero (non-binding constraint), changes in z may further optimize the objective function until the constraint becomes binding. If $F(z)$ is binding, no changes in z would further enhance the objective function, causing the marginal to be zero.
Consider the generalized NLP formulation below.

$$\begin{array}{lll} \min & f(x) & \\ \text{s.t.} & g_i(x) \leqslant 0 & i = 1, 2...m \\ & h_k(x) = 0 & k = 1, 2...p \\ & d_l(x) \geqslant = 0 & l = 1, 2...q \\ & L \leq x \leq Z & \\ & f(x) : R^n \mapsto R, g(x) : R^n \mapsto R^m & \\ & h(x) : R^n \mapsto R^p, d(x) : R^n \mapsto R^p & \end{array} \tag{1}$$

The Lagrange function and KKT conditions for the formulation above in the complementarity form are written as

$$\begin{array}{l} L(x, u, v, w) = f(x) - < u, g(x) > - < v, h(x) > - < w, d(x) > \\ \bigtriangledown_x L \perp L_x \leq x \leq U_x \\ - \bigtriangledown_u L \perp u \geq 0 \\ - \bigtriangledown_v L \perp v free \\ - \bigtriangledown_w L \perp w \leq 0 \end{array} \tag{2}$$

It should be noted that the gradient of Lagrangian w.r.t to the marginals from NLP result in the original inequality and equality constraints of the NLP.
Thus an NLP can be solved as an MCP using the KKT conditions. However, formulating the KKT conditions might prove difficult for complex systems, and require verification of their accuracy. In the following sections, we provide the framework for modeling the KKT conditions by using concept of dummy complementarity equations in the KKT formulation. The process includes the following steps:

1. Solve NLP formulation without explicit KKT conditions, and save the results

2. Restart the problem as an MCP with a system of dummy KKT conditions using solution from step 1 as initial point. The MCP should start at the solution itself

3. Replace one dummy equation at a time with one KKT condition. Resolve the NLP and save the results

4. Restart the problem as MCP. If MCP iteration count is > 0, there exists a problem with the KKT condition which was introduced.

Follow steps 3 and 4 until all KKT conditions have been successfully incorporated.

2 Example : Maximum Revenue- NLP formulation

Consider a simple case of a steel factory trying to maximize the revenue under budget constraints, with man-hours(h) and raw materials steel (s) as the decision variables. The revenue is a function of decision variables given by

$$R(h, s) = 200h^{(2/3)}s^{(1/3)}$$

where, budget = $ 20,000 cost of manpower = $ 20 /hr cost of raw material = $ 170 / tonn
In it's standard form, the model can be written as :

$$\begin{aligned} \min \quad & R(h, s) = -200h^{2/3}s^{1/3} \\ \text{s.t.} \quad & 20h + 170s \leq 20000 \\ & L < h, s < U \end{aligned} \tag{3}$$

The GAMS program below gives the optimum values of

```
Maximum Revenue, R -51854.82
Man hours, h 666.67
Tons of raw material, s 39.22
```

```
$ontext
The model below showcases using GAMS to identify / modify the KKT conditions for a given
NLP formulation. Our starting point is a basic minimization NLP model.
$offtext

scalar
    labor 'cost of labor dollar per hour' /20/
    steel 'cost of steel per ton' /170/
    budget 'total budget for production' /20000/
    ;
variables
    h 'man hours in production' /lo 1, up 50000/
    s 'tons of raw material' /lo 1, up 50000/
    R 'Revenue'
    ;
Equations
    con1 'constraint on budget'
    obj 'objective function'
    ;
obj.. R =e= - 200 * h**(2/3) * s**(1/3) ;
con1.. 20*h + 170*s =l= budget;

h.l=10;
s.l=10;
```

```
model khan /con1,obj/;
solve khan using NLP minimizing R;
```

The above model is then written in form of an MCP, by explicitly adding the KKT conditions to the model. For the given system, the KKT conditions are given as

$$\begin{aligned} L =& -200h^{2/3}s^{1/3} - con1_m[20h + 170s - 20000] \\ \bigtriangledown_h L =& -200 * (2/3)h^{(-1/3)}s^{(1/3)} - con1\_m * (20) \\ \bigtriangledown_s L :& -200 * (1/3)h^{(2/3)} * (1/3) * s^{(-2/3)} - con1\_m * (170) \\ \bigtriangledown_{con1_m} L :& 20 * h + 170 * s - 20000 = 0 \end{aligned} \tag{4}$$

It should be noted that the third KKT equation, result of differentiation w.r.t to the marginal, is the inequality budget constraint from original NLP model. The above equations are solved as an MCP model as shown below using the results saved in the initial run. In the model below, an error has been made in one of the KKT conditions. Execution of the model terminates due to the abort statement, with declaration *'we did not start at the solution'* as shown in the subsequent log.

```
variables
    con1_m 'marginal value for con1' /up 0/
    ;
Equations
    dLdh 'gradL wrt h'
    dLds 'gradL wrt s'
    ;

con1_m.l=con1.m;
dLdh.. - 200*(2/3) * ([h**(2/3)]/h) * s**(1/3) - con1_m*(200) =n=0;
dLds.. - 200 * h**(2/3) * (1/3)*s**(-2/3) - con1_m*(170) =n=0;

model kkt /dLdh.h,dLds.s,con1.con1_m/;
kkt.iterlim=0;
solve kkt using MCP;

abort $(kkt.objval >1e-5) 'We should start at a solution';
```

```
--- Job sd_kkt.gms Start 07/18/18 10:11:54 25.1.1 r66732 WEX-WEI x86 64bit/MS Windows
GAMS 25.1.1 Copyright (C) 1987-2018 GAMS Development. All rights reserved
Licensee: Chintan Bhomia, Single User License G180612/0001CN-GEN
          GAMS Development, Fairfax DC14199
          cbhomia@gams.com
--- Starting continued compilation
--- Workfile was generated under GAMS version WEX251-251
--- sd_kkt.gms(17) 2 Mb
--- Starting execution: elapsed 0:00:00.008[LST:27]
--- sd_kkt.gms(42) 3 Mb
--- Generating MCP model kkt[LST:27]
--- sd_kkt.gms(43) 5 Mb
--- 3 rows 3 columns 8 non-zeroes
--- 25 nl-code 4 nl-non-zeroes
--- sd_kkt.gms(43) 3 Mb
--- Executing PATH: elapsed 0:00:00.026[LST:79]
Reading dictionary...
Reading row data...
Evaluating functions...
```

```
Checking model...
Calculating Jacobian...

PATH 25.1.1 r66732 Released May 19, 2018 WEI x86 64bit/MS Windows

3 row/cols, 8 non-zeros, 88.89% dense.

Path 4.7.04 (Sat May 19 15:07:52 2018)
Written by Todd Munson, Steven Dirkse, and Michael Ferris

Major Iteration Log
major minor func grad residual step type prox inorm (label)
    0 0 1 1 3.2033e+02 I 0.0e+00 3.2e+02 (dLdh)

FINAL STATISTICS
Inf-Norm of Complementarity . . 1.5533e+05 eqn: (dLdh)
Inf-Norm of Normal Map. . . . . 4.6669e+02 eqn: (dLdh)
Inf-Norm of Minimum Map . . . . 4.6669e+02 eqn: (dLdh)
Inf-Norm of Fischer Function. . 3.2033e+02 eqn: (dLdh)
Inf-Norm of Grad Fischer Fcn. . 2.7429e+04 eqn: (con1)
Two-Norm of Grad Fischer Fcn. . 2.7429e+04

 ** EXIT - iteration limit.

Major Iterations. . . . 0
Minor Iterations. . . . 0
Restarts. . . . . 0
Crash Iterations. . . . 0
Gradient Steps. . . . . 0
Function Evaluations. . 1
Gradient Evaluations. . 1
Basis Time. . . . . 0.000000
Total Time. . . . . 0.000000
Residual. . . . . 3.203316e+02

--- Restarting execution
--- sd_kkt.gms(43) 2 Mb
--- Reading solution for model kkt[LST:94]
--- Executing after solve: elapsed 0:00:00.123[LST:155]
--- sd_kkt.gms(45) 3 Mb
*** Error at line 45: Execution halted: abort$1 'We should start at a solution'[LST:160]
--- sd_kkt.gms(45) 3 Mb 1 Error
*** Status: Execution error(s)[LST:177]
--- Job sd_kkt.gms Stop 07/18/18 10:11:54 elapsed 0:00:00.125
```

Absence of *abort* statement results in a new solution to the MCP where the results do not match the desired values from the original NLP formulation, also indicating error in one of the KKT conditions. From simple observation, we can see that an error was made in typing out the multiplier for variable *con1_m* in equation *dLdh*, which when corrected provides results consistent with the NLP formulation.

However, mere observation does not help with larger, more complex systems. A strategy, as described in Section 1 is required to effectively correct the model. In the given case, the MCP formulation can be first solved using dummy KKT conditions described below. These conditions can then be replaced one at a time with the real KKT equations derived earlier.

```
dLdh.. 37 =n=0 ; h.fx=h.L;
```

```
dLds.. 37 + con1_m=n=0; s.fx=s.L; dummy

model kkt /dLdh.h,dLds.s,con1.con1_m/;
```

2.1 Rules for writing dummy KKT conditions

Writing dummy condition, as straightforward as it may seem requires consideration of few salient points to avoid errors from the GAMS compiler.

1. Number of dummy equations must equal number of marginal variables from the NLP formulation
2. Each dummy KKT condition must be accompanied by its differentiating variable (of Lagrangian)
3. The value to differentiating variable is fixed at the marginal values provided by NLP solution
4. Each KKT condition must be modeled as 'complimentary to' the differentiation variable for the condition. i.e. dLdh is $\perp$ to h, and is modeled as dLdh.h in the model statement
5. Each variable representing the original constraint marginal (e.g con1_m) in the MCP model must appear in one of the KKT conditions.

Thus, all variables representing the marginals of NLP constraints can be incorporated in any one of the dummy equation. This equation should be replaced by KKT condition at the very end. However, in the current case, variable 'con1_m' , representing marginal of constraint 'con1' from NLP formulation is part of both the KKT conditions. Thus, though incorporated in equation dLds , the order of replacing the dummy equation with KKT condition is irrelevant in the given case, but can be better understood in the example described in subsequent section