In this course, we will apply numerical methods to solve constrained optimization problems. Mathematically speaking, in a constrained optimization problem we are asked to find values for \( n \) parameters or design variables, \( \mathbf{x} = [x_1 \ x_2 \ x_3 \ \cdots \ x_n]^T \), that minimizes a specified objective function \( J = f(\mathbf{x}) \) such that a set of inequalities are satisfied.

In many design problems, the objective is to simply minimize the total cost (or price) of the product you are designing. In other design problems, the objective could be to maximize an aspect of the performance, to maximize profits, etc. The suitability of the optimized design depends on more than the cost (or performance, profits, etc). Most systems must also meet a range of other criteria (e.g., safe enough, reliable enough, strong enough). These criteria are expressed as inequality constraints, and depend upon the set of design variables. As long as the inequalities are met the criteria are satisfied. The inequalities constrain the best value of the objective. In a structural design, for example, some inequality constraints might be that: (i) the stress remains less than a yield stress value, (ii) the compression force in a column remains less than the buckling load of the column, and (iii) the deformation of the system remains within certain limits. Note that any inequality \( a \leq b \) can be written \( a - b \leq 0 \) or \( a/b - 1 \leq 0 \). In general, we will write a set of \( m \) inequality constraints as \( g_1(\mathbf{x}) \leq 0 \), \( g_2(\mathbf{x}) \leq 0 \), \( \cdots \), \( g_m(\mathbf{x}) \leq 0 \).

With this notation, a constrained optimization problem may be written:

\[
\begin{align*}
\text{minimize} & \quad J = f(x_1, x_2, \ldots, x_n), \\
\text{such that} & \quad g_1(x_1, x_2, \ldots, x_n) \leq 0 \\
& \quad g_2(x_1, x_2, \ldots, x_n) \leq 0 \\
& \quad \vdots \\
& \quad g_m(x_1, x_2, \ldots, x_n) \leq 0
\end{align*}
\] (1)

In certain rare cases, we can write actual equations for \( J = f(\mathbf{x}) \) and \( g(\mathbf{x}) \leq 0 \), and use calculus to find an equation for the constrained optimum. In the vast majority of practical problems, however, the equations are simply much too complicated for this approach. In such cases computer-aided analysis can automate the evaluation of the cost and feasibility of a particular trial design. Further, computer-aided optimization allows designers to rapidly determine feasible and possibly “optimal” designs.

In this homework, you will examine this process by solving optimization problems. Problem 2 involves writing about twenty lines of MATLAB. The TA will help you familiarize yourself with MATLAB in the recitation on Wednesday. Links to some tutorials are here . . . http://www.duke.edu/~hpgavin/m-files/ . . .
1. MLK day (10 points):

Attend an MLK day event or read something by MLK. (Links to some MLK writings are on the course webpage.) Write a paragraph about the event you attended or the material you read.

Note that Nelder and Mead published their now-famous algorithm in 1965, the same year that Lyndon B. Johnson signed the voting rights act.

2. The Nelder-Mead method (20 points):

Consider iterations of the Nelder-Mead algorithm for minimizing a function of several variables.

(a) Using the figure below, trace out five iterations of the Nelder-Mead algorithm. Each iteration starts with a reflection. Use a straight-edge to draw your triangle-shaped simplexes. Within each triangle write “R” for reflection; “RE” for reflection+extension; “Ci” for inside contraction; “Co” for outside contraction; and “S” for shrink.

(b) Within an iteration, is the best point of a “simplex” ever updated? (yes/no)

(c) If a “reflection” point is selected, does the “volume” of the “simplex” remain unchanged? (In 2D, the “simplex” is the triangle and the “volume” of the “simplex” is the area of the triangle.) (yes/no)

(d) When enforcing constraint equations via a penalty function, if the optimization algorithm converges to an infeasible point, should you re-try the optimization with a larger or a smaller penalty factor?

(e) If the optimization routine reaches its maximum iteration limit before converging, what changes to the optimization algorithm options:

```matlab
options = [ 2 0.01 0.01 0.01 500 1.0 1.0 ];
```

would you try, and why? The meaning of the terms in this vector are explained in the document “Examples of Running Constrained Optimization Codes.”
3. Convergence criteria (15 points)

In an iteration of the Nelder-Mead method for \( n = 5 \) has a simplex \( \mathbf{x} \) represented by

\[
\begin{array}{cccccccc}
84.80 & 84.93 & 84.88 & 84.92 & 85.01 & 85.10 \\
14.89 & 15.05 & 14.98 & 15.12 & 14.96 & 15.07 \\
9.97 & 10.00 & 9.99 & 10.09 & 9.84 & 9.94 \\
55.91 & 55.91 & 56.28 & 56.16 & 56.02 & 56.04 \\
72.98 & 72.79 & 72.89 & 72.95 & 72.86 & 72.96 \\
\end{array}
\]

with corresponding objective function values \( f \)

\[
\begin{array}{cccccccc}
99.81 & 99.93 & 100.04 & 100.06 & 100.10 & 100.18 \\
\end{array}
\]

If the convergence criteria used are \( \epsilon_x = 0.01 \) and \( \epsilon_f = 0.01 \), does this simplex and set of objectives represent a converged solution?
4. Optimize your 3-point shot (40 points):

You are to swish a shot from the 3-point line. Your shot is parameterized by the speed $V_o$ and the angle $\theta_o$ that the ball leaves your hands. The ball leaves your hands at a height $h$, which is assumed to not be a parameter in the design of the perfect 3-point shot. The rim is at a height $H$ from the floor and is a horizontal distance $L$ from the 3-point line. The ball has mass $m$ and diameter $d$ and the hoop has diameter $D$. To simplify your analysis, neglect air-resistance, the effect of any back-spin, and assume that your left-right aim is perfect. Derive equations for the horizontal velocity $V_x$ (which is constant in the absence of air resistance), and the vertical velocity $V_y(t)$. Derive an equation for the time $t_L$ that it takes for the ball to travel the horizontal distance $L$ in terms of the shot parameters $(V_o, \theta_o)$ and $L$. This is the time it takes for the center of the ball to get right above (or below) the center of the rim. Next, derive an equation for the height of the ball (from the floor) at time $t_L$, $y_L = y(t_L)$ in terms of the shot parameters $(V_o, \theta_o)$, $t_L$, and other constants ($h, g$). The slope of the ball-center trajectory when the center of the ball is right over the center of the hoop is $y'_L = V_y(t = t_L)/V_x$. Also, derive an equation for the power $P$ required to shoot the ball in terms of its mass initial velocity and the time spent throwing the ball, before it leaves your hands (assume this time to be 0.07 s).

The criterion for the shot passing through the rim depends on the relationships you have derived and some geometry, shown below.

\[
L + \frac{d}{2} s - \left( y_L + \frac{d}{2} c - H \right) \frac{1}{y'_L} \leq L + \frac{D}{2} \tag{2}
\]

\[
L - \frac{d}{2} s - \left( y_L - \frac{d}{2} c - H \right) \frac{1}{y'_L} \geq L - \frac{D}{2} \tag{3}
\]

where $\tan \beta = -y'_L$, $s = \sin \beta$, and $c = \cos \beta$.

With these equations, consider the following two optimization problems,

(a) Make the cleanest shot. Minimize the cost $J$, where

\[
J = f(V_o, \theta_o) = (y_L - H)^2 \tag{4}
\]

such that equations (2) and (3) are satisfied. ... but regardless of how powerful you need to be.

(b) Just make the shot but with the least amount of effort. Minimize $J$ where

\[
J = f(V_o) = P(V_o) \tag{5}
\]

such that equations (2) and (3) are satisfied.

Note that optimization problem (a) may have an infinite number of equally-optimal solutions. The solution to problem (b) should be unique.

Write a MATLAB function function $[J, g] = \text{three_point_analysis}(p, c)$ that computes the "cost function" $J$ and the constraints $g$ given the vector of shot parameters $x = [V_o; \theta_o]$, and and a vector of the other data required for the calculation, $c = [g, d, m, h, L, H, D]$. Use units of degrees for $\theta_o$ (and the $\sin$ and $\cos$ functions in MATLAB). You can write a single function file that has different lines to compute $J$ for problems (a) and (b). (Use comments to change the file for the desired case.) Use your function $\text{three_point_analysis.m}$, the optimization script on the next page (and the CEE 201 optimization codes: $\text{ORSopt.m}$, $\text{NMAopt.m}$, $\text{Optimizations.m}$.}
avg_cov_func.m, box_constraint.m, optim_options.m, plot_surface.m, plot_cvg_hst.m) to find the “best” $V_o$ and $\theta_o$. This script will display a surface plot of the objective function in terms of the two design parameters. You will see the penalized surface plotted over the domain bounded by $p_{\min}$ and $p_{\max}$. Solve problems (a) and (b) using NMAopt and ORSopt using different values of the initial guess for $V_o$ and $\theta_o$. It is totally ok if the initial guesses (your first tries at a 3-point shot) are infeasible (an air-ball). For problem (a), do different initial guesses converge to different solutions that are equally optimal (same value of $J$, and all $g$ values $\leq 0$)? For problem (b), do different initial guesses converge to the same solution? At the optimal solution, which constraints are closest to zero? What to the values of the constraint equations at the optimal solution tell you about the nature of the solution?

To hand in:

- Your derivation of the equations for $t_L$, $y_L$, and $P$.
- Your well-commented MATLAB function `three_point_analysis.m`
- The values of $d$, $m$, $h$, $L$, $H$, and $D$ you used, and the sources of this information.
- The table below, filled in.
- A brief discussion of the nature of the optimal solution in terms of costs and constraint values (just a few sentences).
\begin{verbatim}
Homework 2

\% three_point_opt

\% gravity, m/s^2
\d = 9.81;  \% diameter of the basketball, m
m = 0.07; \% mass of a basketball, kg
h = 1.09; \% height of player, m
L = 0.25; \% distance to 3-point line, m
H = 3.05; \% height of rim, m
D = 0.30; \% diameter of the rim, m

c = [g d m h L H D]; \% constants in a vector
Vo = 12.0; \% initial guess for speed of throw, m/s
To = 25.0; \% initial guess for angle of throw, deg.

x_init = [Vo ; To ];
x_min = [ 7.0 ; 30.0 ];
x_max = [ 11.0 ; 75.0 ];

options = [3 1e-2 1e-2 1e-4 1000 ??? ??? ];

[ x_opt , f_opt , g_opt , cvg_hst ] = ...
NMAopt ( 'three_point_analysis' , x_init , x_min , x_max , options , c);
\% ORSopt ( 'three_point_analysis' , x_init , x_min , x_max , options , c);
plot_cvg_hst ( cvg_hst , x_opt , 11 );
\end{verbatim}