List of exercises 01
Fundamentals of optimization

Augusto Mathias Adams*
*augusto.adams@ufpr.br

***Abstract*—This document is submitted as a partial requirement for evaluation of the discipline of *Evolutionary Computation and Inteligence of Swarms(TE-943)* of the Electrical Engineering Course, with emphasis in Embedded Systems, of the Electrical Engineering Department at the Federal University of Paraná.**

***Index Terms*—Optimization Problem; Linear Programming; Non-linear Programming;**

I. Question 01

Given the design vector: $\vec{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots \mathbf{x}_n\}^T$, match the following terms and descriptions:

(a) Free feasible point	(e) $g_j(\vec{X}) = 0$
(b) Free infeasible point	(c) Some $g_j(\vec{X}) = 0$ and other $g_j(\vec{X}) < 0$
(c) Bound feasible point	(d) Some $g_j(\vec{X}) = 0$ and other $g_j(\vec{X}) \geq 0$
(d) Bound infeasible point	(b) Some $g_j(\vec{X}) > 0$ and other $g_j(\vec{X}) < 0$
(e) Active constraints	(a) All $g_j(\vec{X}) < 0$

From the book *"Engineering Optimization - Theory and Practice"* (Rao, 2019):

> Consider an optimization problem with only inequality constraints $g_j(\vec{\mathbf{X}}) \leq 0$. The set of values of $\vec{\mathbf{X}}$ that satisfy the equation $g_j(\vec{\mathbf{X}}) = 0$ forms a hypersurface in the design space and is called a constraint surface. Note that this is an $(n-1)$-dimensional subspace, where n is the number of design variables. The constraint surface divides the design space into two regions: one in which $g_j(\vec{\mathbf{X}}) \leq 0$ and the other in which $g_j(\vec{\mathbf{X}}) > 0$. Thus, the points lying on the hypersurface will satisfy the constraint $g_j(\vec{\mathbf{X}})$ critically, whereas the points lying in the region where $g_j(\vec{\mathbf{X}}) > 0$ are infeasible or unacceptable, and the points lying in the region where $g_j(\vec{\mathbf{X}}) < 0$ are feasible or acceptable. The collection of all the constraint surfaces $g_j(\vec{\mathbf{X}}) = 0, j = 1, 2, ..., m$ which separates the acceptable region is called the *composite constraint surface*.
>
> A design point that lies on one or more than one constraint surface is called a *bound point*, and the associated constraint is called an *active constraint*. Design points that do not lie on any constraint surface are known as *free points*. Depending on whether a particular design point belongs to the *acceptable* or *unacceptable* region, it can be identified as one of the following four types:
>
> 1) *Free and acceptable point*
> 2) *Free and unacceptable point*
> 3) *Bound and acceptable point*
> 4) *Bound and unacceptable point*

II. Question 02

How do you solve a maximization problem as a minimization problem?

Answer: The point $\hat{\mathbf{x}}$ that maximizes $f(\mathbf{x})$ minimizes $-f(\mathbf{x})$, thus to solve a maximization problem as a minimization problem is to solve for the negative objective function.

From the book *"Engineering Optimization - Theory and Practice"* (Rao, 2019):

> It can be seen from *Figure 1.1* that if a point $\hat{\mathbf{x}}$ corresponds to the minimum value of function $f(\mathbf{x})$, the same point also corresponds to the maximum value of the negative of the function, $-f(\mathbf{x})$. Thus, without loss of generality, optimization can be taken to mean minimization, since the maximum of a function can be found by seeking the minimum of the negative of the same function.

From the Book *"Practical Mathematical Optimization"* (Synman and Wilke, 2018):

> The maximization problem: $max\{f(\mathbf{x})\}$ can be cast in the standard form by observing that $max\{f(\mathbf{x})\} = -min\{-f(\mathbf{x})\}$. Therefore in applying a minimization algorithm set $F(\mathbf{x}) = -f(\mathbf{x})$. Also if the inequality constraints are given in the non-standard form: $g_j(\mathbf{x}) \geq 0$, then set $\hat{g}_j(\mathbf{x}) = -g_j(\mathbf{x})$. In standard form the problem then becomes:
>
> $$\text{minimize } F(\mathbf{x}) \text{ such that } \hat{g}_j(\mathbf{x}) \leq 0.$$
>
> Once the minimizer $\hat{\mathbf{x}}$ is obtained, the maximum value of the original maximization problem is given by $-F(\hat{\mathbf{x}})$.

III. Question 03

What is the difference between linear and nonlinear programming problems?

Answer: The major difference between linear and nonlinear programming problems is the function type of objective function and constraints. In the *Linear Programming Problem* (***LP***), both objective function and constraints are linear, that is, the objective function and constraints are of zero or first order. In the case of *Nonlinear Programming Problem* (***NLP***), any of objective function or constraints are nonlinear functions, that is, are of second or higher order or have exotic functions (trigonometrical functions, exponential functions, etc).

From the book *"Engineering Optimization - Theory and Practice"* (Rao, 2019):

> ***Linear Programming Problem*** If the objective function and all the constraints are linear functions of the design variables, the mathematical programming problem is called a *Linear Programming Problem* (***LP***).
>
> ***Nonlinear Programming Problem*** If any of the functions among the objective and constraint functions in Eq. (1.1) is nonlinear, the problem is called an *NLP problem*. This is the most general programming problem and all other problems can be considered as special cases of the *NLP problem*.

IV. Question 04

What is the difference between design variables and preassigned parameters?

Answer:

- ***Preassigned parameters:*** fixed values or quantities of the problem;
- ***Design Variables:*** other values or quantities of the problem, usually called *design* or *decision* variables. As the names says, obviously they can vary to accomodate the optimal result for the problem.

From the book *"Engineering Optimization - Theory and Practice"* (Rao, 2019):

> Any engineering system or component is defined by a set of quantities, some of which are viewed as variables during the design process. In general, certain quantities are usually fixed at the outset and these are called preassigned parameters. All the other quantities are treated as variables in the design process and are called design or decision variables $\mathbf{x}_i$, $i = 1, 2, \ldots, n$. The design variables are collectively represented as a design vector $\vec{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}^T$. As an example, consider the design of the gear pair, characterized by its face width b, number of teeth T_1 and T_2 , center distance d, pressure angle ϕ, tooth profile, and material. If center distance d, pressure angle ϕ, tooth profile, and material of the gears are fixed in advance, these quantities can be called preassigned parameters. The remaining quantities can be collectively represented by a design vector $X = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}^T = \{b, T_1, T_2\}^T$. If there are no restrictions on the choice of b, T_1 , and T_2 , any set of three numbers will constitute a design for the gear pair. If an n-dimensional Cartesian space with each coordinate axis representing a design variable $\mathbf{x}_i$ $(i = 1, 2, \ldots, n)$ is considered, the space is called the design variable space or simply design space. Each point in the n-dimensional design space is called a design point and represents either a possible or an impossible solution to the design problem.

V. Question 05

What is the difference between a bound point and a free point in the design space?

Answer: a bounding point is a design point that lies on one or more constraints, a free point is a design point that does not lie on any constraint.

From the book *"Engineering Optimization - Theory and Practice"* (Rao, 2019):

> A design point that lies on one or more than one constraint surface is called a *bound point*, and the associated constraint is called an *active constraint*. Design points that do not lie on any constraint surface are known as *free points*.

VI. Question 06

What are the differences between zero, first and second order optimization methods?

Answer: the terms zero, first and second order refers to the nth-derivative used by the method.

- ***Zero order methods:*** employs the objective function f directly. Examples of zero order methods are Nelder-Mead simplex (direct search), Powell conjugate directions and Optimization metaheuristics;

  From the Book *"Practical Mathematical Optimization"* (Synman and Wilke, 2018):

  > These methods are called such because they neither use first order nor second order derivative information, but only function values, i.e. only zero order derivative information.
  >
  > Zero order methods are of the earliest methods and many of them are based on rough and ready ideas with few theoretical background. Although these ad hoc methods are, as one may expect, much slower and computationally much more expensive than the higher order methods, they are usually reliable and easy to program. One of the most successful of these methods is the *downhill simplex method* of Nelder and Mead (1965), also known as Nelder-Mead.

- ***First order methods:*** employs at least the first derivative f (first order information). It approximates f as a plane. Examples of first order methods are Steepest descent methods, Nonlinear conjugate gradients methods and Broyden-Fletcher-Goldfarb-Shanno algorithms (BFGS).

  From the Book *"Practical Mathematical Optimization"* (Synman and Wilke, 2018):

  > Line search descent methods (see Section 2.1.1), that use the gradient vector $\nabla f(\mathbf{x})$ to determine the search direction for each iteration, are called first order methods because they employ first order partial derivatives of $f(\mathbf{x})$ to compute the search direction at the current iterate. The simplest and most famous

of these methods is the method of steepest descent, first proposed by Cauchy in 1847.

- ***Second order methods:*** employs at least the second derivative of f (second order information). It approximates f as a quadratic form. Examples of second order method are Newton's method, Newton conjugate gradient and Quasi-Newton method.

  From the Book *"Practical Mathematical Optimization"* (Synman and Wilke, 2018):

  These methods are based on Newton's method (see Section 1.5.4.1) for solving $\nabla f(\mathbf{x}) = 0$ iteratively: Given $\mathbf{x}_0$, then $\mathbf{x}_i = \mathbf{x}_{i-1} - H^{-1}(\mathbf{x}_{i-1})\nabla f(\mathbf{x}_{i-1})$, $i = 1,2,\ldots$ The main characteristics of this method are:

  - In the neighbourhood of the solution it may converge very fast. In fact, it has the very desirous property of being quadratically convergent if it converges. Unfortunately convergence is not guaranteed and it may sometimes diverge, even from close to the solution.
  - The implementation of the method requires that $H(\mathbf{x})$ be evaluated at each step.
  - To obtain the Newton step, $\Delta = \mathbf{x}_i - \mathbf{x}_{i-1}$ it is also necessary to solve a $n \times n$ linear system $H(\mathbf{x})\Delta = -\nabla f(\mathbf{x})$ at each iteration. This is computationally very expensive for large n, since an order n^3 multiplication operations are required to solve the system numerically.

  $H(\mathbf{x})$ is the *Hessian* matrix or the second derivative of $f(\mathbf{x})$.

VII. Question 07

Considering an optimization global problem given by min$\{f(\mathbf{x})\}$, define global minimum, strong local minimum, weak local minimum, weak global minimum, strong global minimum, and saddle points.

- ***Global Minimum (Synman and Wilke, 2018):*** $\hat{\mathbf{x}}$ is a *global minimum* over the set X if $f(\mathbf{x}) \geq f(\hat{\mathbf{x}})$ for all $x \in X \subset \mathbb{R}^n$.
- ***Strong and Weak Global Minima (Arora, 2016):*** A function $f(\mathbf{x})$ of n variables has a *global (absolute) minimum* at $\hat{\mathbf{x}}$ if

$$f(\mathbf{x}) \geq f(\hat{\mathbf{x}}) \tag{1}$$

  for all $\mathbf{x}$ in the feasible design space S. If strict inequality holds for all $\mathbf{x}$ other than $\hat{\mathbf{x}}$ in Inequality 1, then $\hat{\mathbf{x}}$ is called a *strong (strict) global minimum*; otherwise it is called a *weak global minimum.*
- ***Strong and Weak Local Minima (Arora, 2016):*** A function $f(\mathbf{x})$ of n variables has a *local (relative) minimum* at $\hat{\mathbf{x}}$ if Inequality 1 holds for all $\mathbf{x}$ in a small neighborhood N of $\hat{\mathbf{x}}$ in the feasible design space S. If strict inequality holds, then $\hat{\mathbf{x}}$ is called a *strong (strict) local minimum*; otherwise it is called a *weak local minimum.*
- ***Saddle Points (Rao, 2019):*** In the case of a function of two variables, $f(x,y)$, the *Hessian* matrix may be neither positive nor negative definite at a point $(\hat{x},\hat{y})$ at which

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} = 0 \tag{2}$$

  In such a case, the point $(\hat{x},\hat{y})$ is called a *saddle point*. The characteristic of a saddle point is that it corresponds to a relative minimum or maximum of $f(x,y)$ with respect to one variable, say, x (the other variable being fixed at $y = \hat{y}$) and a relative maximum or minimum of $f(x,y)$ with respect to the second variable y (the other variable being fixed at $x = \hat{x}$).

VIII. Question 08

Considering a constrained optimization problem, define Lagrange multiplier and Karush-Kuhn-Tucker (KKT) conditions.

- ***Lagrange Multiplier (Figments of my deranged mind about my Calculus Course, 2017 - Prof. Viviana Cocco Mariani; Rao, 2019; Synman and Wilke, 2018; Arora, 2016):*** An optimization problem is given in the form:

$$\begin{aligned} &\text{minimize } f(\hat{\mathbf{x}}) \\ &\text{such that } g(\hat{\mathbf{x}}) = 0 \end{aligned} \tag{3}$$

  It is possible to write the problem as:

$$\mathscr{L}(\hat{\mathbf{x}},\hat{\lambda}) = f(\hat{\mathbf{x}}) - \hat{\lambda}^T g(\hat{\mathbf{x}}) \tag{4}$$

  Where $\mathscr{L}(\hat{\mathbf{x}},\hat{\lambda})$ is called the *Lagrangian Function*, a linear combination of $f(\hat{\mathbf{x}})$ and $g(\hat{\mathbf{x}})$. The $\hat{\lambda}$ factor is called the *Lagrange Multiplier*. If there is a $\hat{\mathbf{x}}$ which is the optimization problem solution, then there exists a unique Lagrange multiplier $\hat{\lambda}$ such that $\nabla f(\hat{\mathbf{x}}) = \hat{\lambda}^T \nabla g(\hat{\mathbf{x}})$.
  The condition $\nabla f(\hat{\mathbf{x}}) = \hat{\lambda}^T \nabla g(\hat{\mathbf{x}})$ implies that any direction perpendicular to all gradients of the constraints is also perpendicular to the gradient of the function. In other words, the directional derivative of the function is 0 in every feasible direction.
- ***Karush-Kuhn-Tucker conditions (Synman and Wilke, 2018):*** considering the optimization problem in the form:

$$\begin{aligned} &\text{minimize } f(\hat{\mathbf{x}}) \\ &\text{such that } g_j(\hat{\mathbf{x}}) \leq 0,\ \ j = 1,2,\ldots,m \end{aligned} \tag{5}$$

  and the Lagrangian function defined as:

$$\mathscr{L}(\hat{\mathbf{x}},\hat{\lambda}) = f(\hat{\mathbf{x}}) - \sum_{j=1}^{m} \hat{\lambda}_j g_j(\hat{\mathbf{x}}) \tag{6}$$

  and assuming the existence of Lagrange multiplier $\hat{\lambda}$, then at the point $\hat{x}$, corresponding to the solution of the optimization problem, the following conditions must be satisfied:

$$\begin{aligned}
\frac{\partial f}{\partial x_i}(\hat{\mathbf{x}}) - \sum_{j=1}^{m} \hat{\lambda}_j \frac{\partial g_j}{\partial x_i}(\hat{\mathbf{x}}) = 0 \;\; i = 1,2,\ldots,n \\
g_j(\hat{\mathbf{x}}) \leq 0,\;\; i = 1,2,\ldots,m \\
\hat{\lambda}_j g_j(\hat{\mathbf{x}}) = 0,\;\; i = 1,2,\ldots,m \\
\hat{\lambda}_j \geq 0,\;\; i = 1,2,\ldots,m
\end{aligned} \tag{7}$$

Which are the *Karush-Kuhn-Tucker conditions*. It can be shown, that the KKT conditions also constitute sufficient conditions (those that imply that) for $\hat{x}$ to be a constrained minimum, if $f(\mathbf{x})$ and the $g_j(\mathbf{x})$ are all convex functions.