Lecture - 4
INTRODUCTION TO QUANTITATIVE OPTIMIZATION

Reference: Book Chapter 5

GENERIC OPTIMIZATION PROBLEM FORMULATION

$$\min_{x} \quad J(x)$$

Vector of **objective functions**

subject to

$$g(x) \leq 0$$

Vector of **inequality constraints**

$$h(x) = 0$$

Vector of **equality constraints**

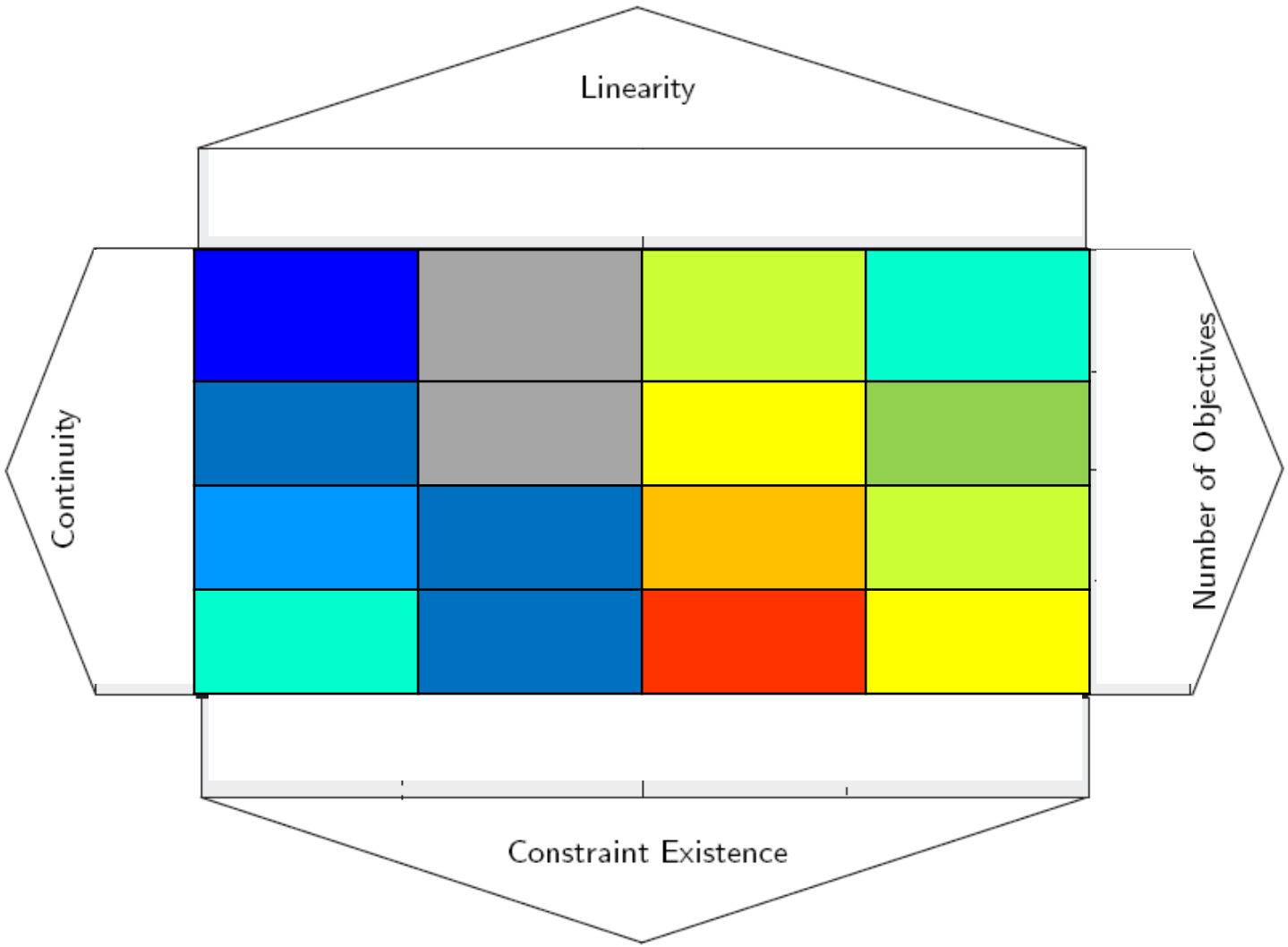
Behavioral constraints

$$x_l \leq x \leq x_u$$

Side constraints

Any set of design variables that fully satisfies all constraints is called a **feasible solution** (even if it does not minimize the objective function).

CLASSIFICATION OF OPTIMIZATION METHODS



LINEAR vs. NONLINEAR

- When all the objective functions and constraint functions are linear, we call the problem a **linear programming** (**LP**) problem.
- When at least one of the objective functions or constraint functions is a nonlinear function of the design variables, we call the problem a **nonlinear programming** (**NP**) problem.
- Generally, LP problems are much easier to solve than NP problems.

Linear Programming problem

$$\min_{x} \quad c^T x$$

$$\text{subject to}$$

$$Ax \leq b$$

$$A_{eq}x = b_{eq}$$

$$x_l \leq x \leq x_u$$

CONSTRAINED vs. UNCONSTRAINED

- When the optimization problem has at least one constraint, we call it a **constrained** optimization problem.
- When the optimization problem does **not** have any constraints, we call it an **unconstrained** optimization problem.
- Most practical problems involve constraints.

Unconstrained optimization problem

$$\min_{x} \quad J(x)$$

$$\text{subject to}$$

$$x_l \leq x \leq x_u$$

Variable bounds may or may not exist (depends on definition)

DISCRETE vs. CONTINUOUS

- If all design variables are continuous, we call it a **continuous optimization** problem.
- If any design variable is **discrete**, we no longer have a continuous optimization problem. We can have the following typical cases:
  - **Binary Programming**: The design variables are allowed to take only the values of 0 or 1, i.e., $X \in [0,1]$.
  - **Integer Programming**: The design variables are allowed to take only integer values, i.e., $X \in Z$.
  - **Discrete Optimization**: The design variables are allowed to take only a given prescribed set of real values, i.e., $X \in S$, where S is the prescribed set of feasible real values.
  - **Combinatorial Optimization**: Any candidate design variable vector is allowed to take only certain combinations of discrete values (real, integer).
- Often, we have a mixture of the above cases, sometimes also including continuous variables, e.g., mixed integer problems.

SINGLE vs. MULTI-OBJECTIVE

- If there is only a single objective function to minimized or maximized, we call it a **single-objective optimization** problem.
- If there are 2 or more objectives to be simultaneously minimized or maximized, we call it a **multi-objective optimization** problem.
- Well-posed multi-objective optimization problems generally involve conflicting objectives, and hence demand searching for the best trade-offs among the multiple objectives.
- Multi-objective optimization problems can be simplified into multiple single objective optimization problems, e.g.,:
  - By aggregating the objective functions.
  - By converting some of the objectives into constraints.

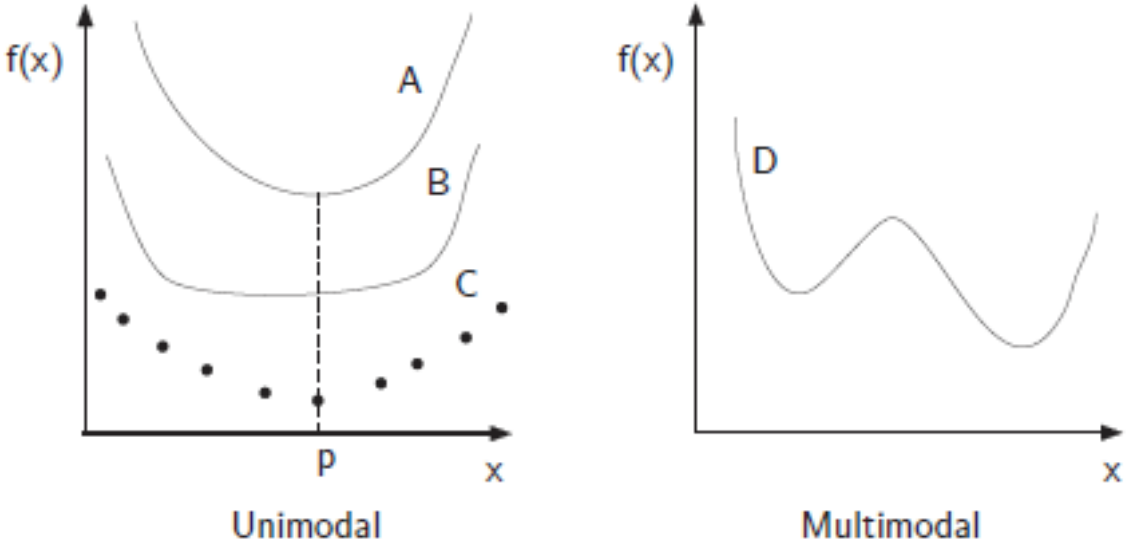
OTHER CLASSIFICATIONS OF OPTIMIZATION

In addition to the four major ways of classifying optimization problems, there exist other classifications based on the mathematical or practical nature of the problem, e.g.,

1. Single vs. Multiple Optima Problem.
2. Deterministic vs. Non-deterministic Problem.
3. Easy vs. Difficult Problem.

SINGLE vs. MULTIPLE OPTIMA

- If an optimization problem has a single local optimum (which is thus also the global optimum), it is called a **unimodal** problem.
- If an optimization problem involves multiple local optimum (one or more of which could be the global optimum), then it is called a **multimodal** problem.
- Multimodal problems are generally more challenging to solve, and require global optimization approaches.



DETERMINISTIC vs. NONDETERMINISTIC

- If there is no uncertainty in the variables of the criteria functions of an optimization problem, then it is called a **deterministic** optimization problem.
- If there is uncertainty associated with any of the variables or criteria functions, then it is called a **non-deterministic** optimization problems.
- A common source of uncertainty is manufacturing tolerances – where the actual tolerance of physical system parts cannot meet the tight tolerances estimated by a quantitative optimization process.

EASY VERSUS DIFFICULT PROBLEM

Easy Problem

1. The model of the system is provided or readily created;
2. Only involves continuous variables;
3. Not strongly nonlinear;
4. Criteria functions are known to be unimodal and/or convex;
5. The computational model of the system behavior can run on a computer in seconds or minutes (not hours);
6. The number of design variables is small (e.g., <10);
7. All the models needed to describe the system behavior can run on a single computer; or
8. All the design variables are deterministic.

Difficult Problem

The opposite of these items leads to a difficult problem.

SOLUTION APPROACHES

1. Analytical Optimization
2. Experimental Optimization
3. Graphical Optimization
4. Numerical (or Algorithmic) Optimization

ANALYTICAL OPTIMIZATION

- An analytical objective function represents the performance of any candidate design or system.
- Differentiate this function, and identify the point(s) where the derivative is zero.
- Estimate the second derivative:
  - If it is positive, we have a minimum.
  - If it is negative, we have a maximum.

EXPERIMENTAL OPTIMIZATION

1. We build a version of the physical system.
2. We evaluate its performance.
3. If we are satisfied with the current performance, we STOP; if not, we think of what changes might be helpful, and we go back to step 1., to hopefully come up with an improved version.
4. This approach is largely outmoded. It can be very costly and time consuming, and it might not converge easily to a good solution.

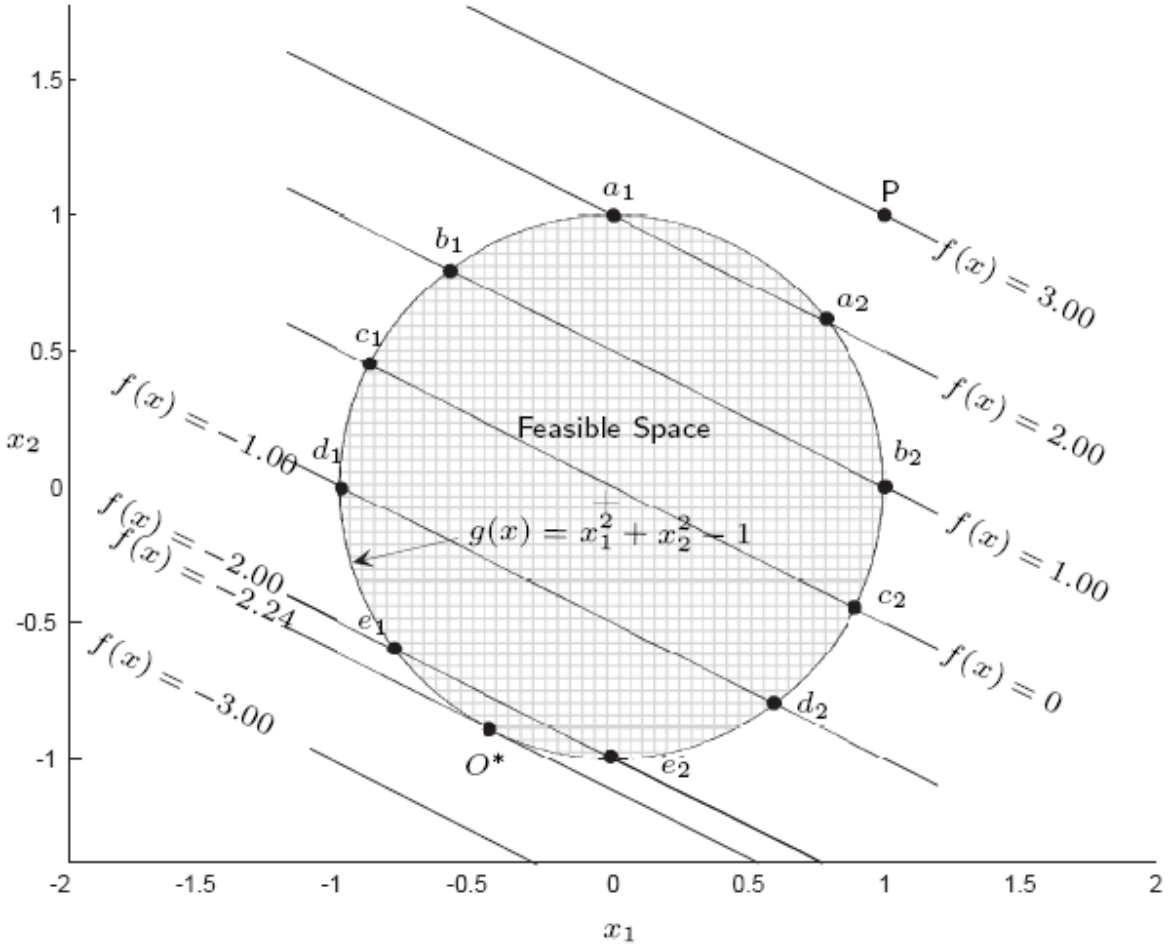
GRAPHICAL OPTIMIZATION

Example

$$\min_{x} \quad f(x) = x_1 + 2x_2$$

subject to

$$x_1^2 + x_2^2 - 1 \leq 0$$
$$-1 \leq x_1, x_2 \leq 1$$



NUMERICAL (OR ALGORITHMIC) OPTIMIZATION

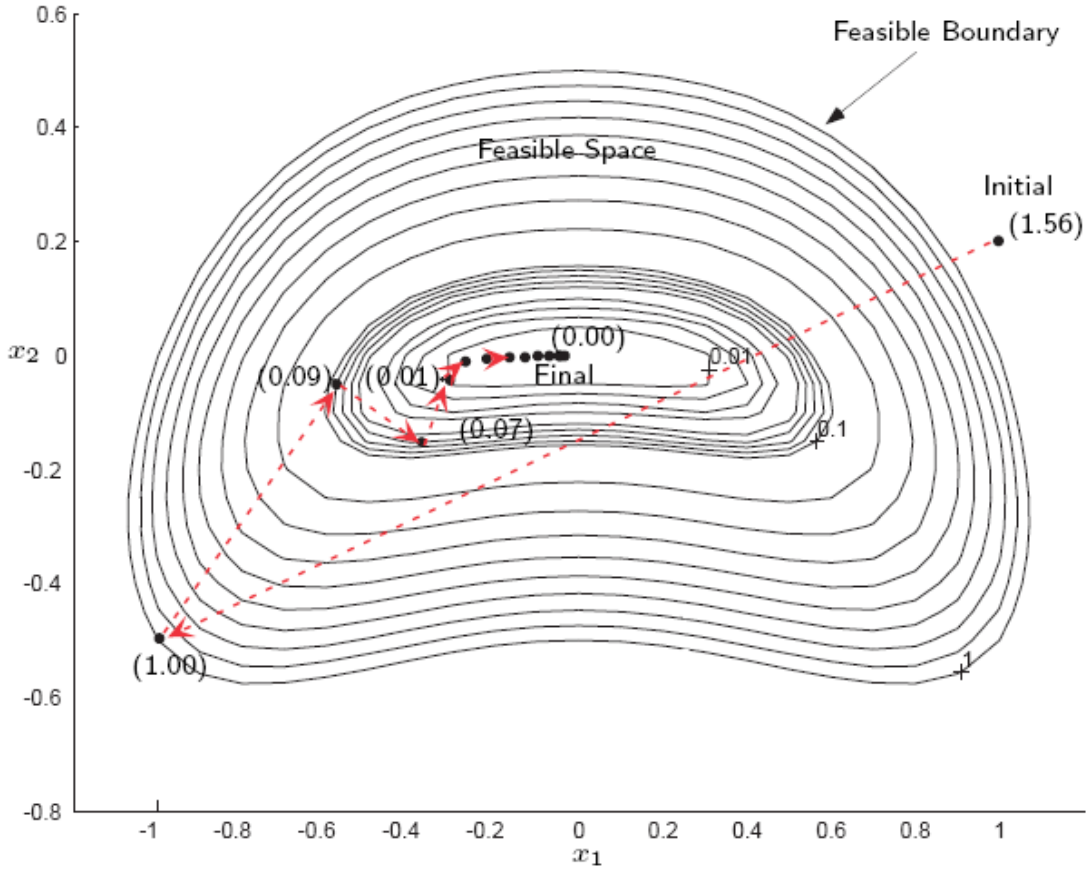
- Numerical optimization is generally performed using a collection of logical steps, called an algorithm.
- Most algorithms perform optimization through a series of iteration.
- Through iterations, the logic of the algorithm seeks to improve the functional values of a single or multiple candidate designs, and the iterations stop when a certain termination criteria is met.
- Numerical optimization algorithms leverage the number crunching power of a computer.

Example

$$\min_x \quad f(x) = (x_1^2 + x_2)^2 + 3x_2^2$$

subject to

$$-1 \leq x_1 \leq 1$$
$$-0.5 \leq x_2 \leq 0.5$$



MATLAB OPTIMIZATION CODE

- A constrained nonlinear optimization problem

Matlab Problem Formulation

$$\min_{\mathtt{x}} \quad \mathtt{f(x)}$$

subject to

$$\mathtt{c(x)} \leq 0$$

$$\mathtt{ceq(x)} = 0$$

$$\mathtt{A\,x} \leq \mathtt{b}$$

$$\mathtt{Aeq\ x} = \mathtt{beq}$$

$$\mathtt{LB} \leq \mathtt{x} \leq \mathtt{UB}$$

Example

$$\min_{x} f(x) = x_1^2 + 3x_2^2 - 2x_1x_2 - 15$$

subject to

$$g_1(x) \equiv -2x_1 - 2x_2 + 8 \leq 0$$

$$-5 \leq x_1, x_2 \leq 5$$

MATLAB OPTIMIZATION: FMINCON FUNCTION

[xopt, fopt] = fmincon('fun', x0, A, b, Aeq, beq, LB, UB, 'nonlcon')

- **x0**: This is the initial guess provided to **fmincon**.
- **A, b, Aeq, and beq**: These terms need to be defined only if the problem has linear constraints. In many cases, all constraints (linear and nonlinear) can be defined in the **nonlcon.m** file, so these terms can simply be defined as empty matrices.
- **LB, UB**: These are vectors containing the lower and upper bounds on the variables x1 and x2, respectively.
- **'fun'**: is the name of the function file containing the definition of f(x).
- **'nonlcon'**: is the name of the function file containing constraints.
- The variables **xopt** and **fopt** are the outputs of **fmincon**, where **xopt** is the optimum vector of variables [x1, x2], and **fopt** is the minimum value of the objective function f.

FMINCON FUNCTION: OBJECTIVE FILE

Function file: *fun.m*

- This file should be saved in the same folder as the main m-file.
- This function returns the value of the objective function at any given point x.

Example

```
function f = fun(x)
f = x(1)^2 + 3*x(2)^2 - 2*x(1)*x(2) - 15;
```

FMINCON FUNCTION: CONSTRAINT FILE

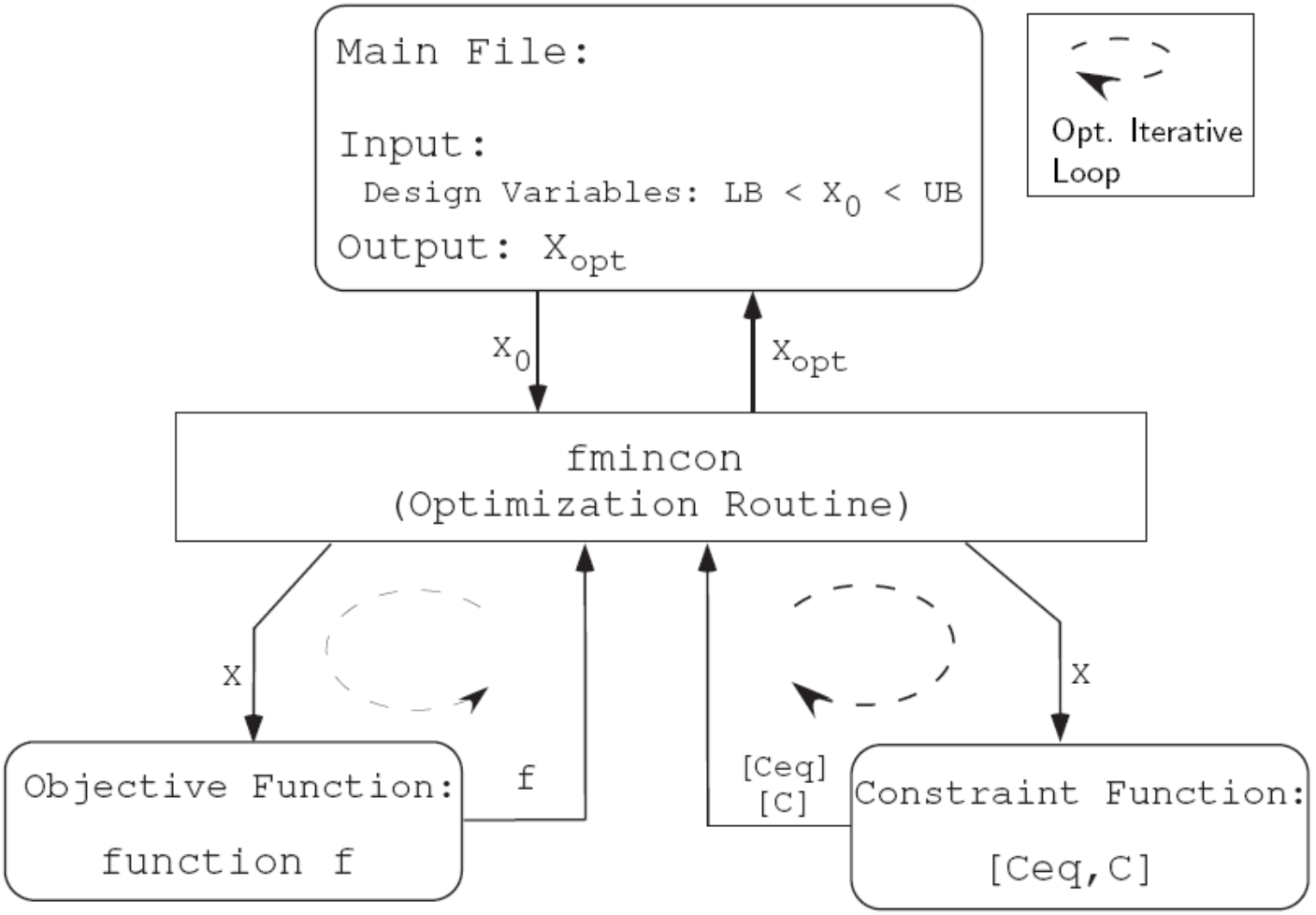
Nonlinear constraints file: *nonlcon.m*

- This file should be saved in the same folder as the main and the function files.
- It returns the values of the inequality and the equality constraints at any given point x.
- Note that all inequality constraints should first be written in the form $g(x) \leq 0$.

Example

```
function [C,Ceq] = nonlcon(x)
C(1) = -2*x(1) - 2*x(2) + 8;
Ceq = [ ];
```

FMINCON FUNCTION: OPERATIONAL STRUCTURE



FMINCON FUNCTION: OPTIONS

Options

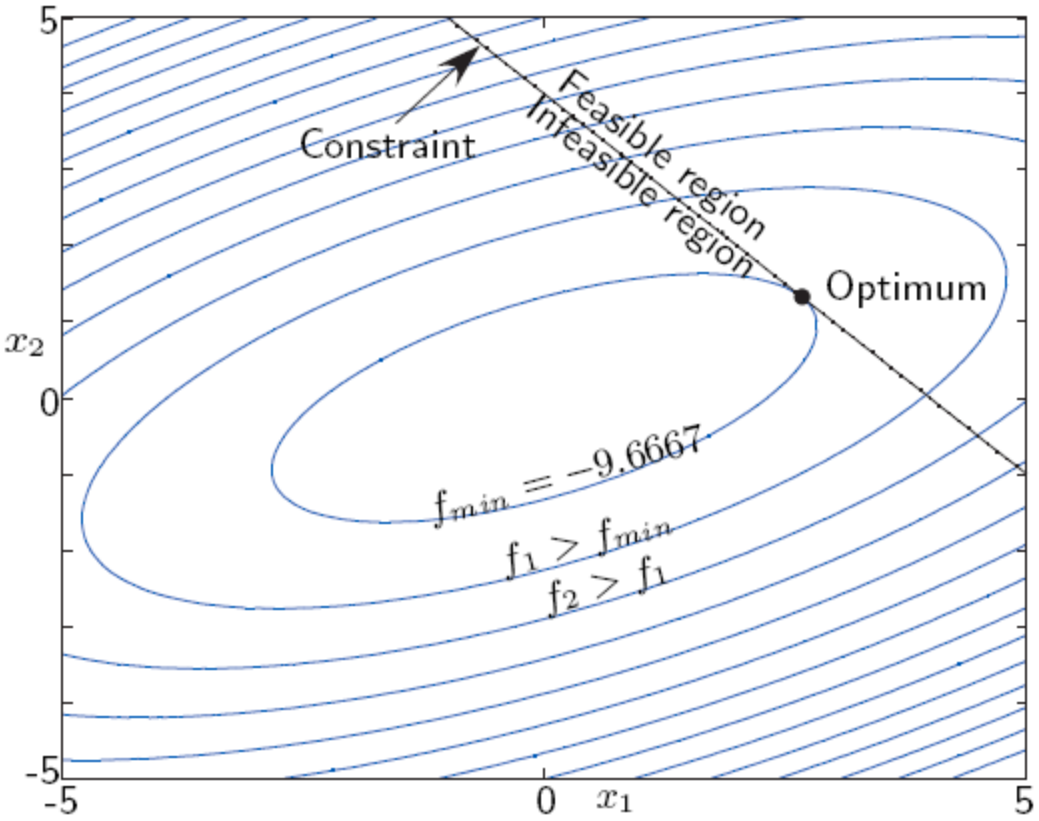
- Optimization **options** can be set for **fmincon** using the command **optimset**.
- Some options apply to all algorithms, while others are relevant to particular algorithms.
- You can use **optimset** to set or change the values of the options arguments.
- The options arguments include settings such as:
  1. algorithms selection,
  2. information display settings,
  3. gradient estimation in series or parallel.

FMINCON FUNCTION OPTIONS: DISPLAY

Display Settings

- Depending on the option selected for display, MATLAB can show different outputs.
- The information includes output at each iteration and shows a technical exit message.
- The exit messages can include hyperlinks. These hyperlinks bring up a window containing further information about the terms used in the exit messages.

Example


5
0
-5
-5
0
5
x_2
x_1
Constraint
Feasible region
Infeasible region
Optimum
$f_{min} = -9.6667$
$f_1 > f_{min}$
$f_2 > f_1$

Linear Optimization/Programming

$$\min_{\mathtt{x}} \quad \mathtt{f}^T\mathtt{x}$$

subject to

$$\mathtt{A}\,\mathtt{x} \leq \mathtt{b}$$

$$\mathtt{Aeq}\,\mathtt{x} = \mathtt{beq}$$

$$\mathtt{LB} \leq \mathtt{x} \leq \mathtt{UB}$$

f: cost coefficient vector

A and Aeq: constant matrices

b and beq: constant vectors

LB and UB: the lower and upper bounds

Example

$$\min_{x} \; 4x_1 + 4x_2$$

subject to

$$-5x_1 - 3x_2 \leq -15$$

$$-3x_1 - 5x_2 \leq -15$$

$$0 \leq x_1,\, x_2 \leq 3$$

LINPROG FUNCTION

[xopt,fopt] = linprog(f, A, b)

- **xopt** and **fopt** are the optimum values of the design variables and objective function, respectively.
- Before solving the linear program, we need to ensure that the objective function is of minimization type, and that all the inequality constraints are written in the form of $a_1x_1+a_2x_2 \leq b_1$

LINPROG FUNCTION: PARTS

- **f**: This is a row vector corresponding to the coefficients of the design variables in the objective function.
- **A**: This is a matrix in which every row corresponds to the coefficients of the design variables in each "≤" constraint. A will have as many rows as we have inequality constraints, and as many columns as we have design variables.
- **Aeq and beq**: Matrix and Vector for linear equality constraints
- **b**: This is a vector in which each element is the number that appears on the right hand side of each "≤" constraint.

Example

```
f = [4;4]
A = [-5 -3; -3 -5]
Aeq = [ ]
beq = [ ]
b = [-15; -15]
```

LINPROG FUNCTION: OPERATIONAL ILLUSTRATION

Example

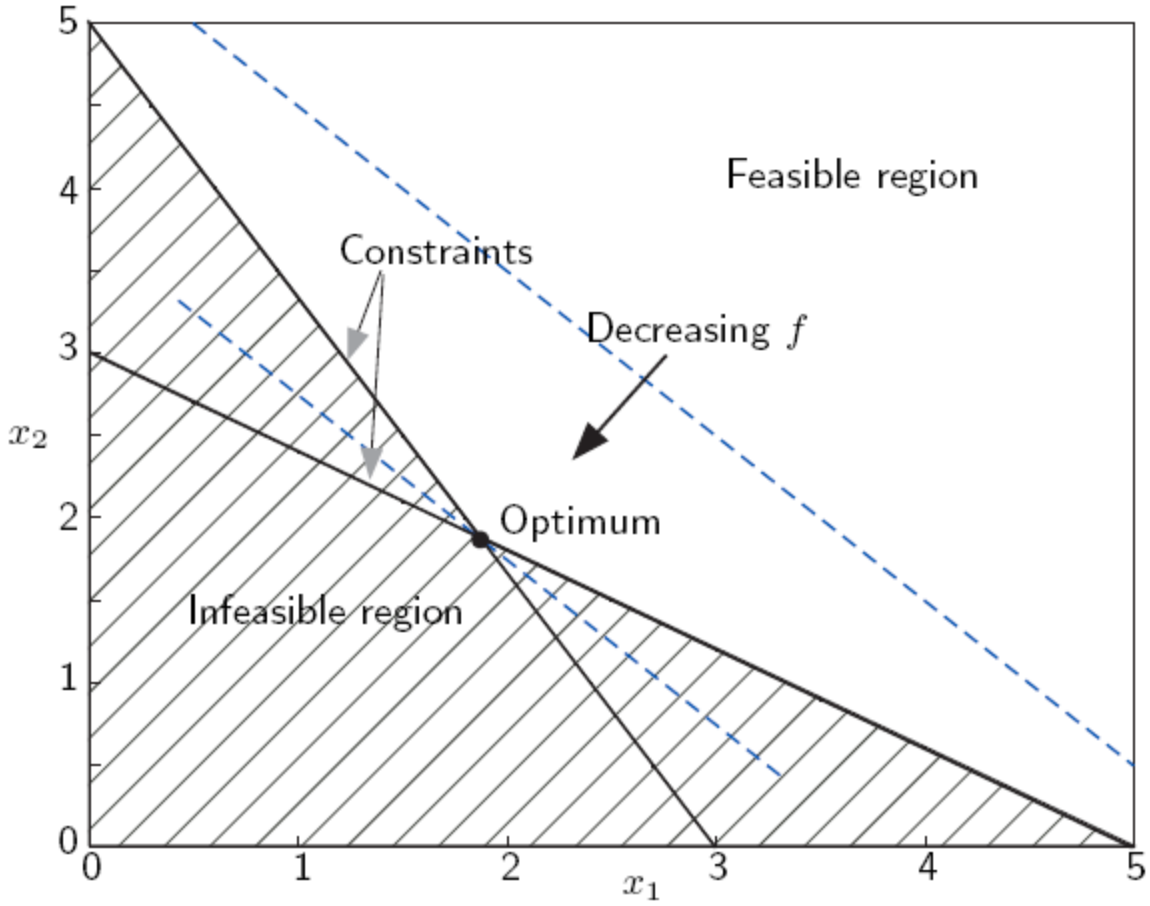
xopt =

1.8750

1.8750

fopt =

15.0000



SOFTWARE OPTIONS

- **A stand-alone optimization code**. The primary focus is to solve various types of optimization problems.
- **An integration framework** where analysis codes from different engineering disciplines can be conveniently integrated and designs can be optimized.
- **Large scale analysis codes** that have optimization as one of the features.

STAND-ALONE OPTIMIZATION CODES

Multipurpose Packages

1. Matlab optimization toolboxes
2. NEOS server
3. GENESIS, DOT and DOC
4. NAG library

Packages for Specific Classes of Problems

- ***Nonlinear constrained problems***
  1. NPSOL
  2. GRG2
- ***Linear programming problems***
  1. LSSOL
  2. CPLEX
- ***Multiobjective problems***
  1. NIMBUS
  2. PhysPro

INTEGRATION FRAMEWORKS WITH OPTIMIZATION

1. iSIGHT
2. Phoenix Integration
3. Boss Quattro
4. modeFRONTIER
5. HEEDS

COMMERCIAL SOFTWARE WITH OPTIMIZATION

1. Altair Optistruct
2. NASTRAN:
3. ANSYS
4. ABAQUS
5. COMSOL
6. Microsoft Excel Add-ins for optimization
   a. What'sBEST! Add in
   b. RISKOptimizer
   c. Business spreadsheets: Portfolio optimization