Pacific Northwest
NATIONAL LABORATORY

Mathematical Optimization in Python

Dan A. Rosa de Jesús

Electrical and Computer Engineering Ph.D. Student
University of Texas at El Paso

Applied Statistics and Computational Modeling Group
National Security Directorate

GEM Fellowship

August 14, 2019.

U.S. DEPARTMENT OF ENERGY BATTELLE

PNNL is operated by Battelle for the U.S. Department of Energy

Agenda

- Introduction
- Optimization Problem
- Types of Optimization Problems
  - Linear Programming
  - Nonlinear Programming
  - Evolutionary Computing
- Conclusion

Introduction

- **Mathematical optimization** or **mathematical programming** is the selection of the best element, with regard to some criterion, from some set of available alternatives.
- Optimization problems arise in many quantitative disciplines including:
  - Computer science
  - Engineering
  - Operations research
  - Economics

Optimization Problem

- Mathematically speaking, an **optimization problem can be defined** as:
  - Given a function ***f*** from a set ***A***, look for an element $\boldsymbol{x_0}$ in ***A*** such that $f(\boldsymbol{x_0}) \leq f(\mathbf{x})$ or such that $f(\boldsymbol{x_0}) \geq f(\mathbf{x})$, for all **x** in *A*.
    - The first case is called **minimization** and the second **maximization**.
    - Note: $f(\boldsymbol{x_0}) \geq f(\mathbf{x}) \Leftrightarrow -f(\boldsymbol{x_0}) \leq -f(\mathbf{x})$
- Optimization problems can have many objective functions and many constraints.

$arg\ min\ x^2 + 1$	Objective function ***f***
$subject\ to{:}\ x > 1$	Constraint on ***x***

Optimization Problem (Cont.)

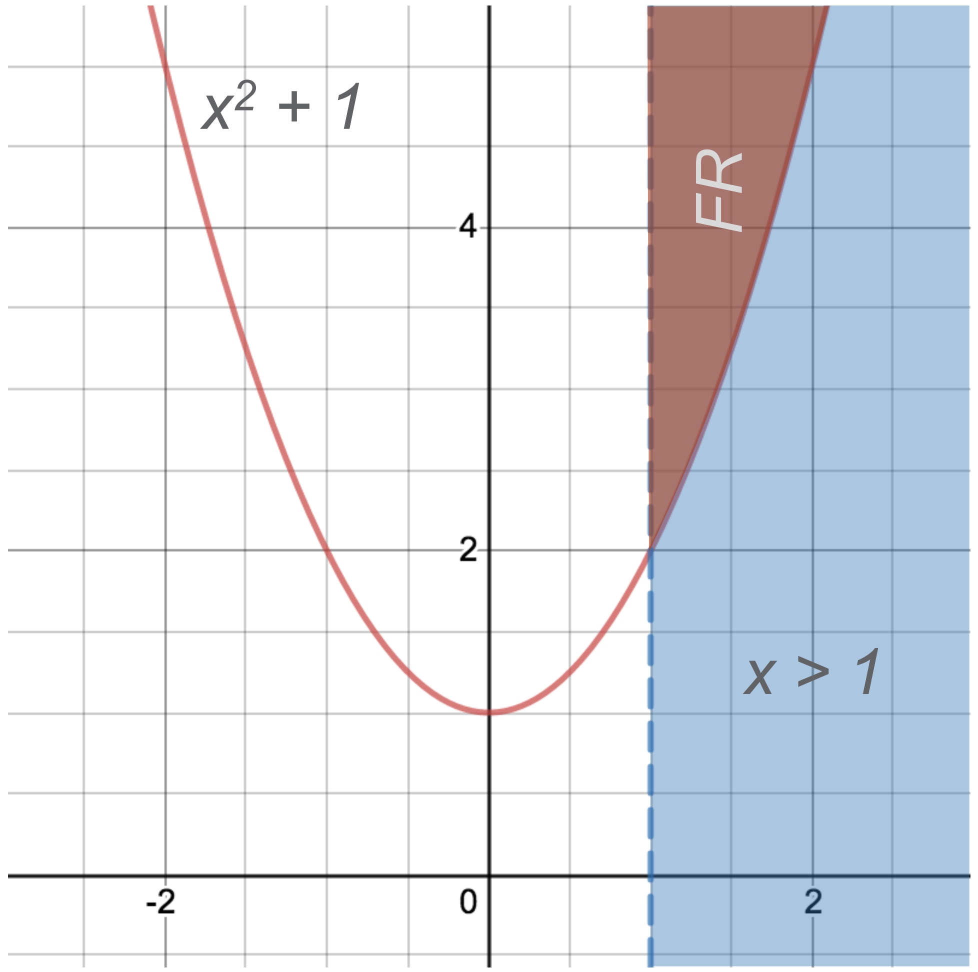
$$\underbrace{\begin{array}{l} \arg\min x^2 + 1 \\ \text{subject to: } x > 1 \end{array}}$$

Objective function ***f***

Constraint on ***x***

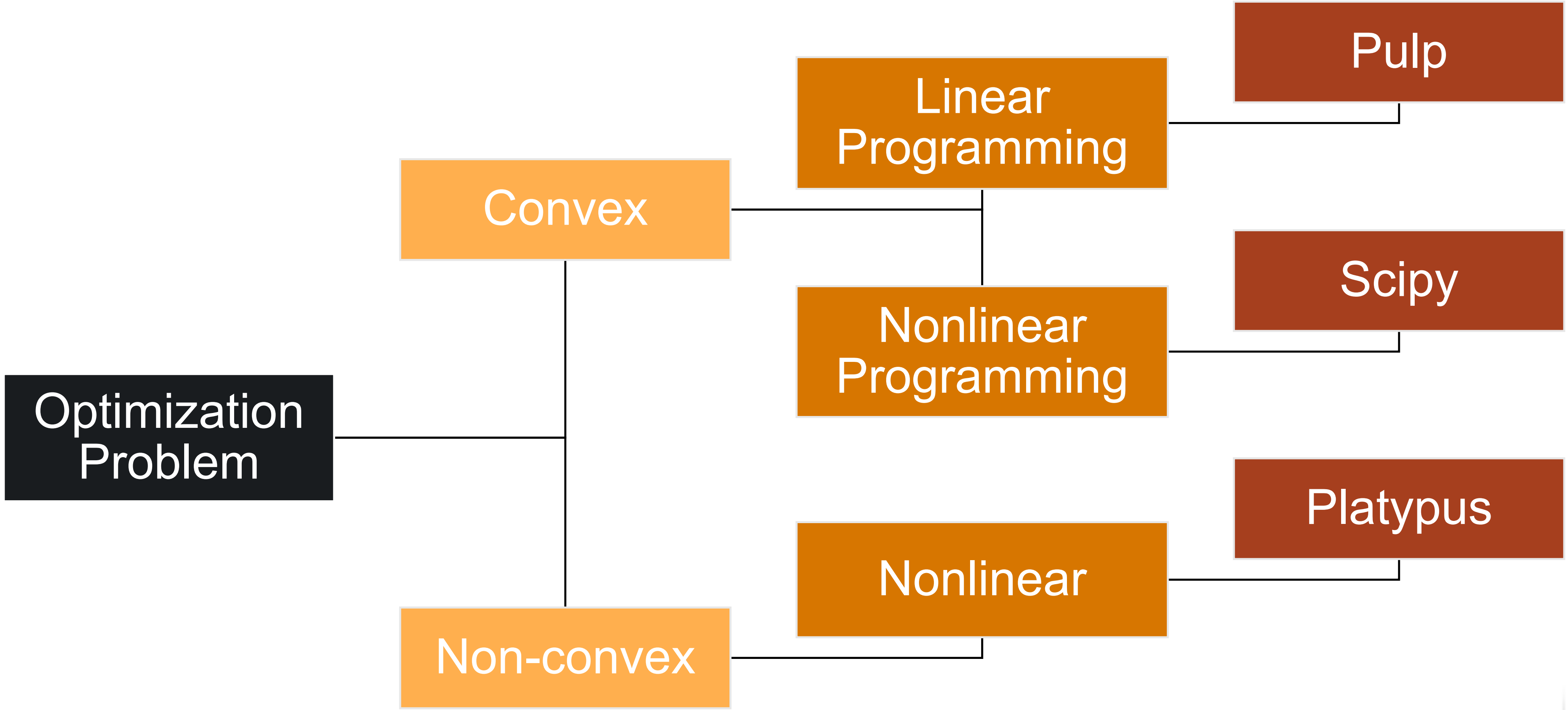
Mathematically

These define the **feasible region**.

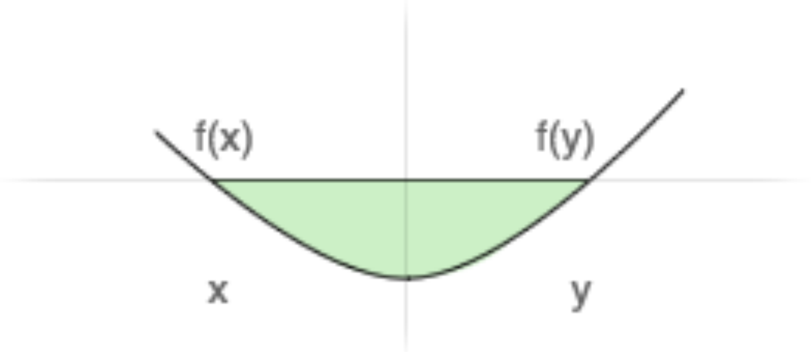


It is easy to verify that the solution to this "trivial" optimization problem is $x = 1^+$.

Some Types of Optimization Problems



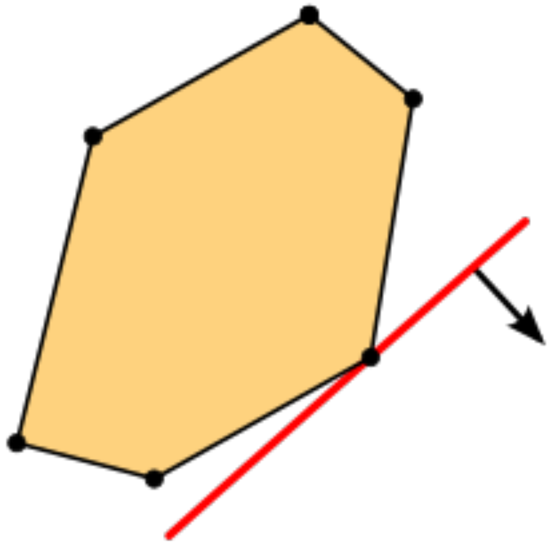
A **function is convex** if a line segment drawn from any point *(x, f(x))* to another point *(y, f(y))* lies on or above the graph of *f*.

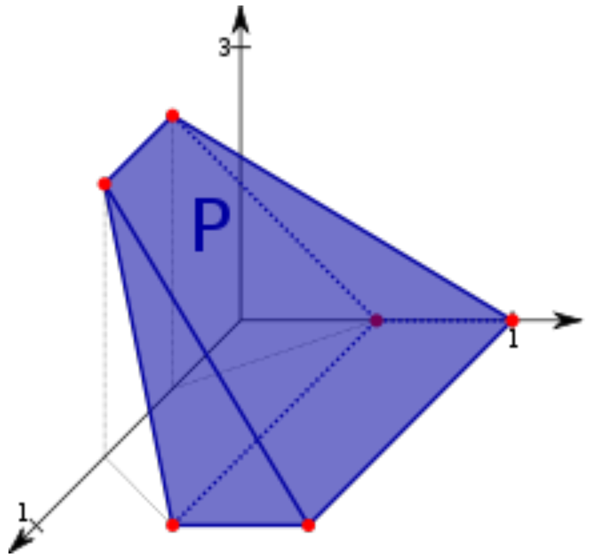


Lineal Programming

- Linear programming is a method to achieve the best outcome in a mathematical model **with objectives and constraints represented by linear relationships**.

$$\begin{aligned} &\arg\max\ \mathbf{c}^T\mathbf{x} \\ &\text{subject to: } A\mathbf{x} \le \mathbf{b} \\ &\qquad\qquad\quad \mathbf{x} \ge 0 \end{aligned}$$





Use case 1: LP Optimization Problem

Suppose you are a mechanic and your goal is to make cars run faster. The more cars you are able make faster, the better. Your secret behind making cars fasters is 2 gas additives, each of which uses special compounds. To create one unit of additive 1, you need 3.1 units of compound A and 2.4 units of compound B. Similarly, to create one unit of additive 2, you need 4.8 and 1 units of compound A and B, respectively. Now additive 1 can make a car faster by 7.5 speed units and additive 2 by 5.2. Furthermore, you only have 15 and 5 units of compound A and B at your disposal.

The question is, how much of each additive will you create to maximize the speed of the next car?

Use case 1: LP Optimization Problem (Cont.)

arg max $7.5\,a_1 + 5.2\,a_2$

subject to: $3.1\,a_1 + 4.8\,a_2 \leq 15$

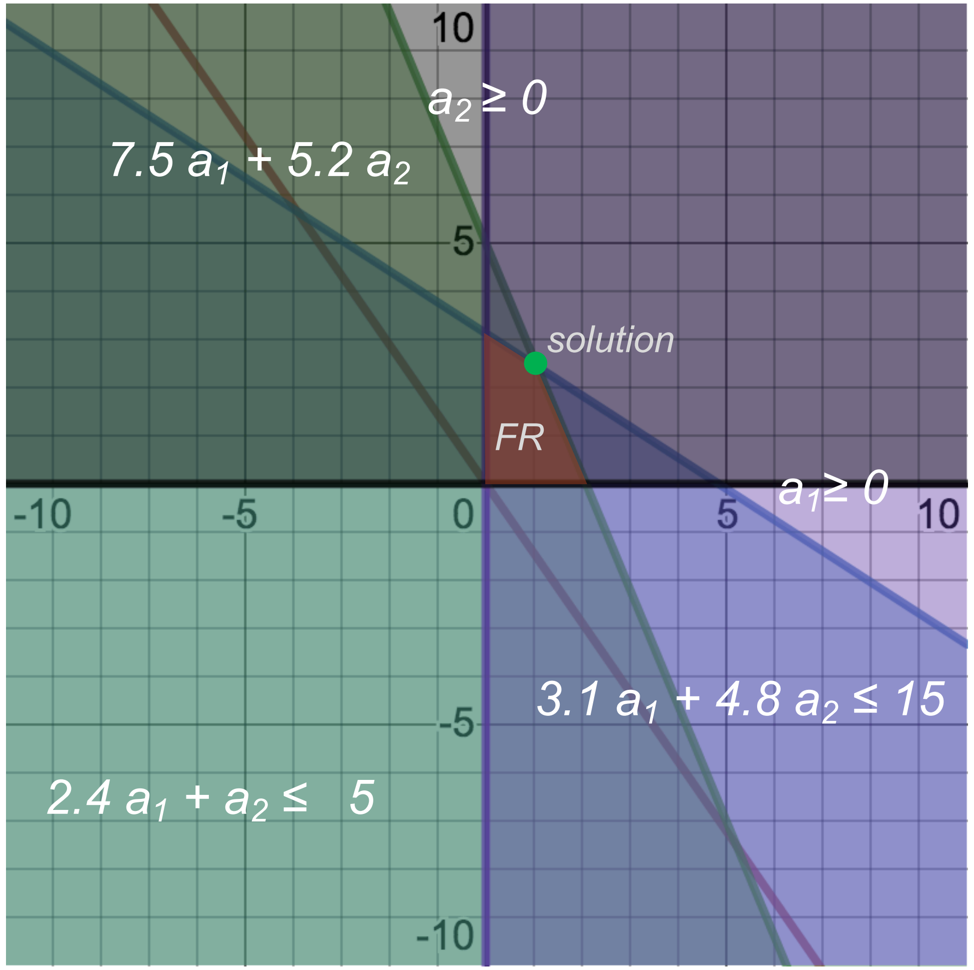
$2.4\,a_1 + a_2 \leq 5$

$a_1 \geq 0$

$a_2 \geq 0$

Solution: $a_1 = 1.07,\ a_2 = 2.43$

Speed can be maximized by 20.68 units!



Use case 1: LP Optimization Problem (Cont.)

What would happen if we change the speed coefficients of the objective?

arg max $1.2\,a_1 + 2.1\,a_2$

subject to: $3.1\,a_1 + 4.8\,a_2 \leq 15$

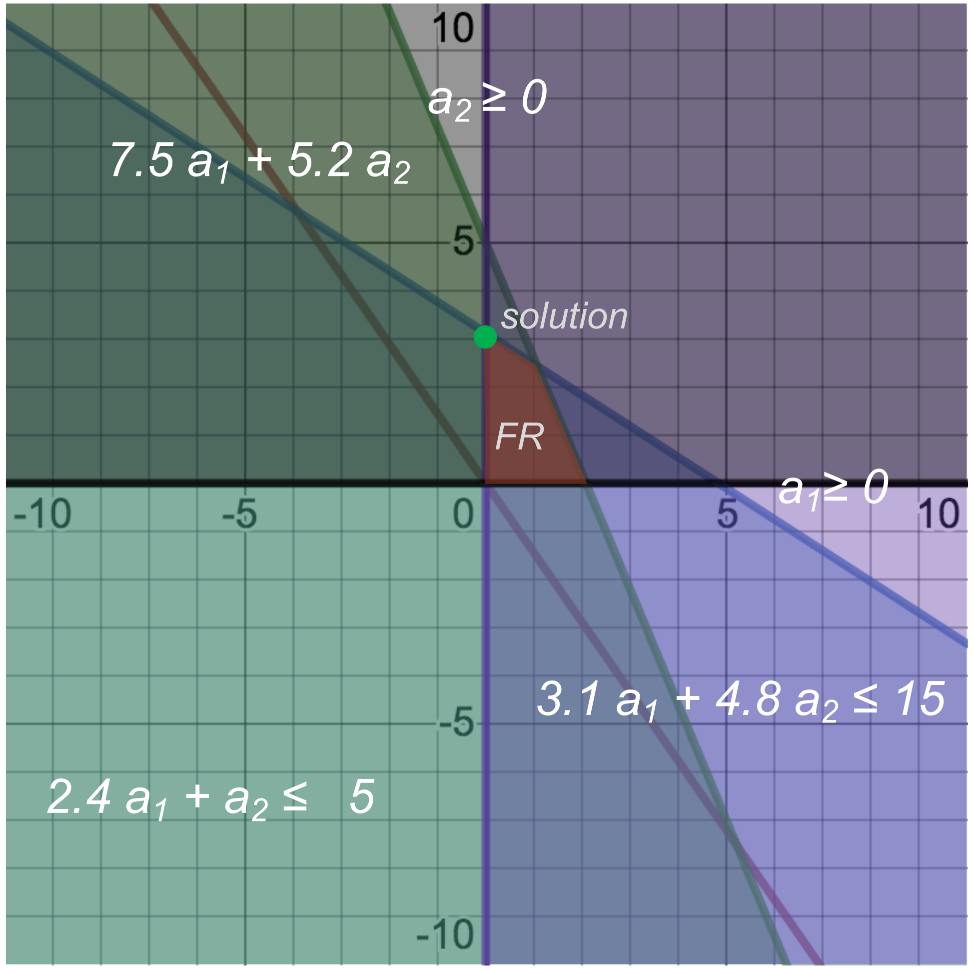
$2.4\,a_1 + a_2 \leq 5$

$a_1 \geq 0$

$a_2 \geq 0$

Solution: $a_1 = 0.00$, $a_2 = 3.12$

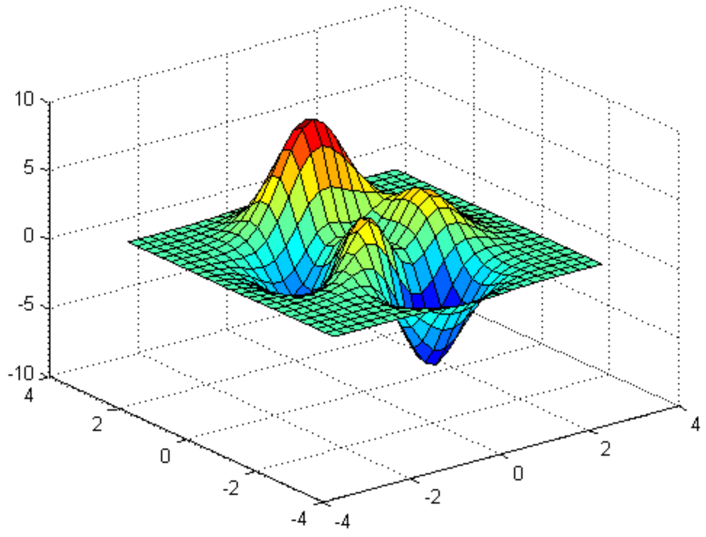
Speed can be maximized by 6.56 units!



Nonlinear Programming

- Nonlinear programming is the process of solving an optimization problem **where some of the constraints or the objective function are nonlinear**.

$$\begin{aligned} &\arg\min f(\mathbf{x}) \\ &\text{subject to: } g(\mathbf{x}) \leq 0 \\ &\qquad\qquad\quad h(\mathbf{x}) = 0 \end{aligned}$$



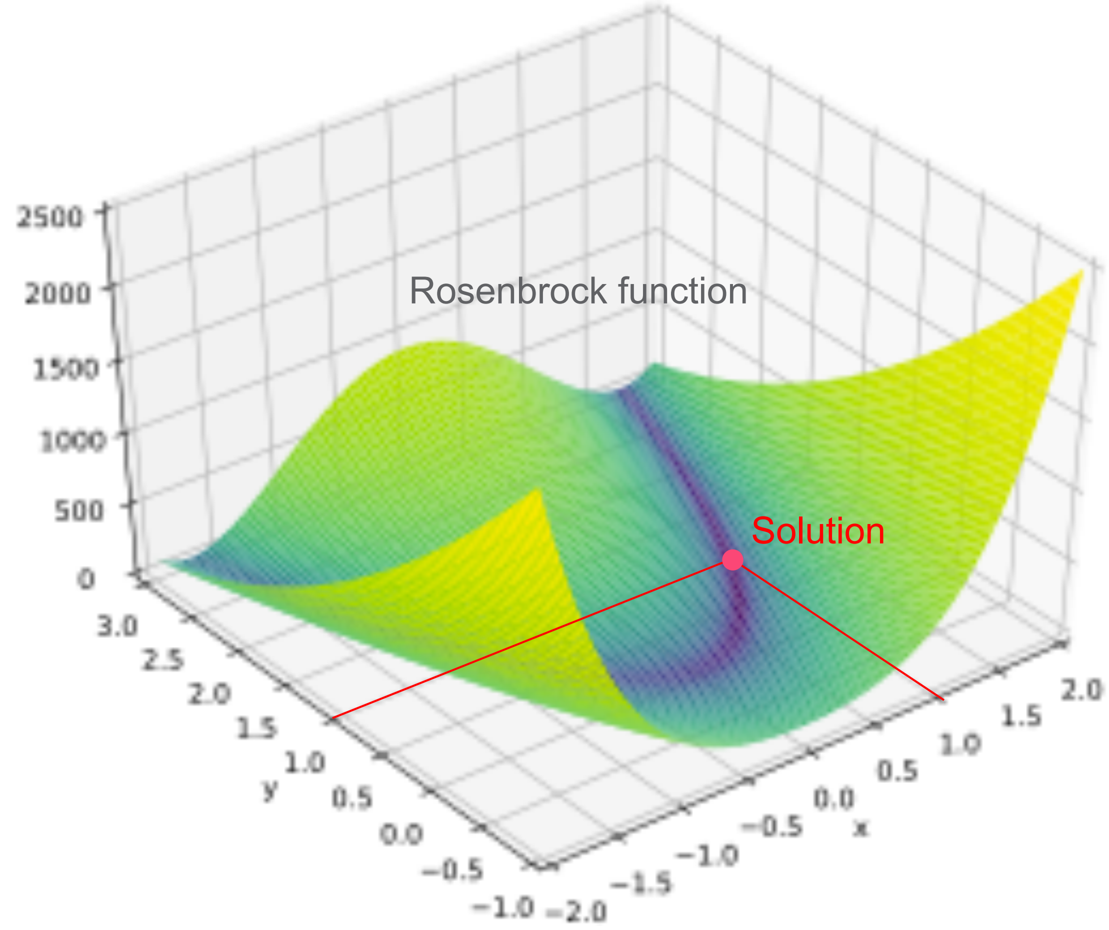
Use case 2: NLP Optimization Problem

- Let us consider the problem of minimizing the Rosenbrock function of variables:

$$\text{arg min} \sum_{i=2}^{N} 100\,(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$$

- The Rosenbrock function is used as a **performance test problem for optimization algorithms**.

- The minimum value of this function is 0 which is achieved when $x_i = 1$.



Evolutionary Computing

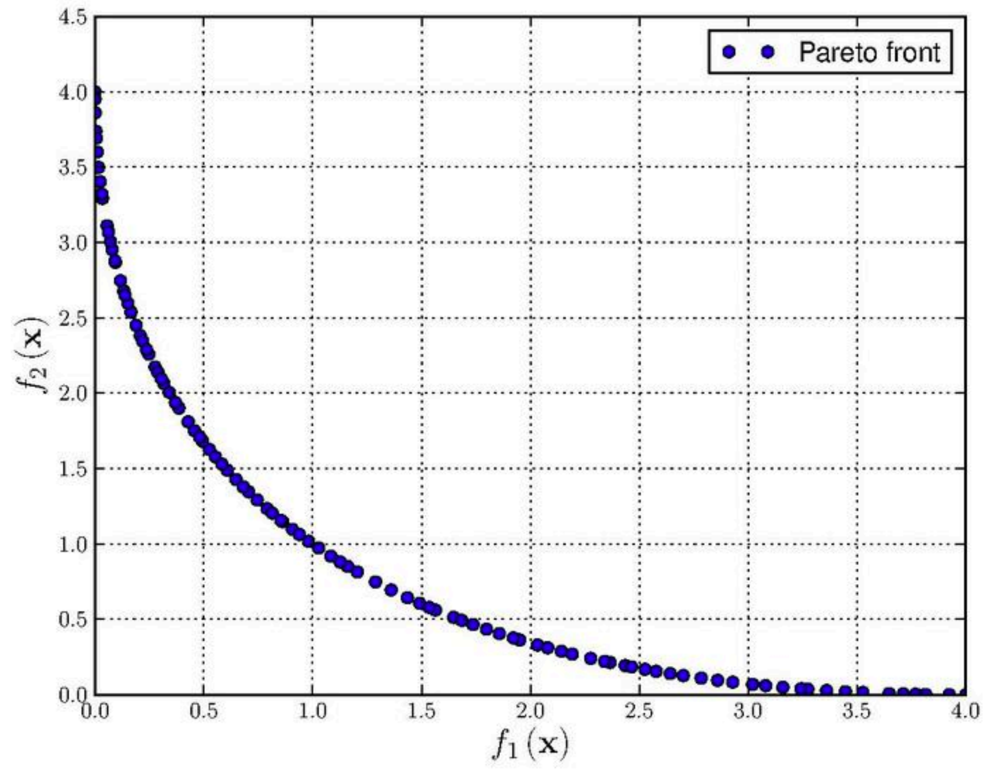
- Evolutionary computation is a family of algorithms for global optimization **inspired by biological evolution.**
  - Genetic Algorithms
    - Initialization: the population size and position is set within the FR.
    - Selection: a portion of the existing population is selected to breed a new generation
    - Genetic operators: i.e. crossover and mutation
    - Termination: if a solution satisfies a minimum criteria, the number of generations is reached, or the combinations of these two.

Use case 3: Evolutionary Computing

- Let us consider the problem of minimizing the Schaffer function:

$$\arg\min \begin{cases} f_1(x) = x^2 \\ f_2(x) = (x-2)^2 \end{cases}$$

- The Schaffer function is also used as a **performance test problem for optimization algorithms**.



Conclusion

- We discussed how to formulate and model optimization problems:
    - Convex and nonconvex
        - Linear
            - Single objective constrained
            - Multi-objective unconstrained
        - Nonlinear
            - Single objective unconstrained
- There are several Python optimization packages:
    - PuLP: A linear programming Python API
    - Scipy: A Python open-source software for mathematics
    - Platypus: A framework for evolutionary computing in Python
- The use of these packages depends on the nature of the problem at hand.

Links

- https://www.coin-or.org/PuLP
- https://docs.scipy.org/doc/scipy/reference/optimize.html
- https://platypus.readthedocs.io/en/latest/