Solving non linear optimization problems

Optimisation problem

$$\min_{X_{variable}} \quad f(X_{variable}, \pi_{parameters})$$ Objective function

$$h(X_{variable}, \pi_{parameters}) = 0$$

$$X_{variable} - \pi_{specification} = 0$$

m : Model
+
Specifications

n : Decision Variables

$$g(X_{variable}) \leq 0$$

Constraints

$$X_{variable,min} \leq X_{variable} \leq X_{variable,max}$$

Degrees of freedom : n-m

Solving method

The goal of the optimisation is to fix the value of the n-m degrees of freedom

Non linear Optimisation methods : iterative

- **Direct search**
  - Exploring the search space from a location to find the lowest point
    - Easy to use and robust
    - High computing time
- **Indirect search**
  - Mathematical condition of the optimality
    - More efficient
    - More complex (derivatives)
- **Heuristic methods**
  - Explore the search space based on a property of the system
    - e.g. genetic algorithms
    - Global optimum
    - Very high computing time
    - Highly dependent on the quality of the model

Unconstrained problems
Black Box Approach

$$\min_{X_{decision}} f(X_{decision})$$

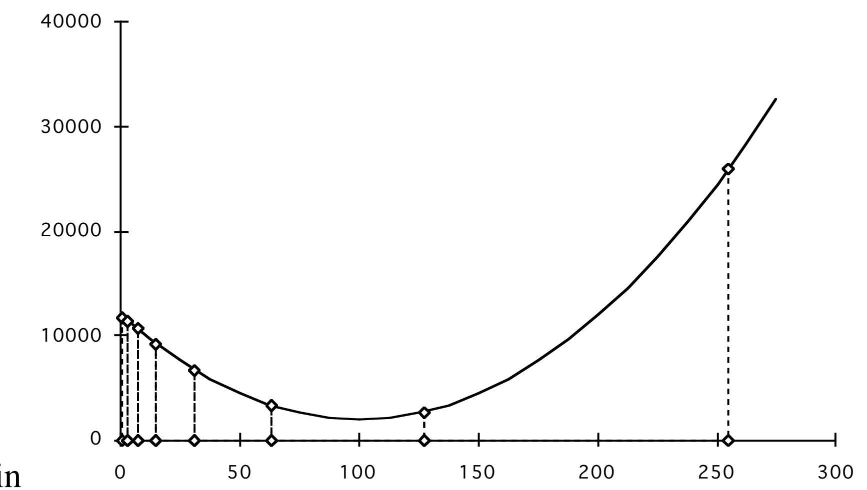
$$X_{decision,min} \leq X_{decision} \leq X_{decision,max}$$

1 DOF method

From the best points : generate a new point and eliminate the worst one

- Direct search
  - Interval reduction

$$x_{\min} < x_1 < x_2 < x_{\max}$$



  - Gold number

$$x_{\max} - x_2 = x_1 - x_{\min}$$

$$et \frac{x_1 - x_{\min}}{x_{\max} - x_1} = \frac{x_{\max} - x_1}{x_{\max} - x_{\min}} = \frac{\sqrt{5}-1}{2} \approx 0{,}618$$

  - Polynomial approximation

$$f(x) = a + bx + cx^2 \Rightarrow \left.\frac{\delta f}{\delta x}\right]^* = 0 \Rightarrow x^* = \frac{-b}{2c}$$

With 3 points

$$x^* = \frac{1}{2}\frac{\left(x_2^2 - x_3^2\right)\cdot f_1 + \left(x_3^2 - x_1^2\right)\cdot f_2 + \left(x_1^2 - x_2^2\right)\cdot f_3}{\left(x_2 - x_3\right)\cdot f_1 + \left(x_3 - x_1\right)\cdot f_2 + \left(x_1 - x_2\right)\cdot f_3}$$

Unconstrainted , multi variables

- 1rst order, follow the slope
  + choose the distance (1 dimension problem)

$$x^{k+1} = x^k + \overbrace{\lambda^k}^{\text{distance}} \overbrace{s^k}^{\text{direction}} = x^k - \lambda^k \underbrace{\nabla f}_{\text{Slope calculation}}\left(x^k\right)$$

Slope calculation

Linear approximation from previous steps
Numerical calculation of derivatives

distance : λ^k

- fixed size
- 1 dimension optimisation $\quad \min\limits_{\lambda^k} f(x^{k+1})$

Indirect search methods

- Unconstrained problem
- Express the conditions of optimality

$$\min_{X_{decision}} f(X_{decision})$$

$$\text{is equivalent to} \quad \nabla f(X_{decision}) = 0$$

- Transform the optimisation problem into solving a set of equations
  - $\nabla F(X) = 0 \quad \forall X$: N variables => N derivatives => NxN problem (0 DOF)
- Inequalities = safe guards => constrained problems

$$X_{decision,min} \leq X_{decision} \leq X_{decision,max}$$

2^{nd} order method : quadratic approximation

Newton method

$$f(x) \approx q(x) = f(x^k) + \nabla^T f(x^k)\, \Delta x^k + \frac{1}{2}(\Delta x^k)^T\, H(x^k)\, (\Delta x^k)$$

$$\nabla q(x) = 0 \Rightarrow \nabla f(x^k) + H(x^k)\, (\Delta \mathrm{x}^{\mathrm{k}}) = 0$$

$$x^{k+1} - x^k = \Delta \mathrm{x}^{\mathrm{k}} = -\, [H(x^k)]^{-1}\, \nabla f(x^k)$$

with $H(x^k) = \dfrac{\delta^2 f(x)}{\delta x_i \delta x_j}$ Hessian Matrix 2^{nd} derivative !

when H is not definite positive $\Rightarrow \mathrm{H} = \mathrm{H} + \beta \mathrm{I}$

$\beta >>$ steepest descent

$\beta << Newton$

$$\nabla f(x^k) = \frac{\delta f(x)}{\delta x_i} : gradient$$

Indirect search methods

Constrained problems

Simultaneous approach
Two level approach

$$\min_{X_{decision}} \quad f(X_{decision})$$

$$h(X_{decision}) = 0$$

$$g(X_{decision}) \leq 0$$

$$X_{decision,min} \leq X_{decision} \leq X_{decision,max}$$

- Canonical formulation :

$$\min f(\underline{x}) \qquad \underline{x}=[x_1,x_2,\ldots,x_n]$$

Subject to $\quad h_j(\underline{x}) = 0 \quad j=1,\ldots,n_h$

$\quad g_k(\underline{x}) \geq 0 \quad k=1,\ldots,n_g$

Add a slag variable $s_k \geq 0$ to each inequality constraint :

$$\mathbf{g_k(\underline{x}) - s_k^2 = 0} \qquad \mathbf{k=1,\ldots,n_g}$$

$$s_k \geq 0$$

- **New problem**

$$\min f(\underline{x}) \qquad \underline{x}=[x_1,x_2,\ldots,x_n, s_g]$$

Subject to $\quad h_j(\underline{x}) = 0 \quad j=1,\ldots,n_h$

$\quad g_k(\underline{x}) - \mathbf{s_k^2} = 0 \quad k=1,\ldots,n_g$

$x_{v,min} \leq x_v \leq x_{v,max} \quad v=1,\ldots,n_v$

$s_k \geq 0 \quad k=1,\ldots,n_g$

Constrained programming

New problem

$$\min f(\underline{x}) \qquad \underline{x}=[x_1, x_2, \ldots, x_n, s_g]$$

Subject to

$h_j(\underline{x}) = 0 \quad j=1,\ldots,n_h$

$g_k(\underline{x}) - s_k^2 = 0 \quad k=1,\ldots,n_g$

$x_{v,min} \leq x_v \leq x_{v,max} \quad v=1,\ldots,n_v$

$s_k \geq 0 \quad k=1,\ldots,n_g$

Reduced gradient

- Variables are divided into two groups :
  - **Basic** Variables are calculated by solving the set of equations for fixed values of the **Non Basic** variables

$$X_{base} = X(X_{nbase}) : h(X_{base}, X_{nbase}) = 0$$

solve
$$h(X_{base}, \chi_{nbase}) = 0$$
$$\chi_{nbase} - X_{Nbase} = 0$$

For given values of X_{Nbase}

New unconstrained problem

$$min_{X_{nbase}} F(X(X_{nbase}), X_{nbase})$$
$$\text{subject to } X_{min} \leq X_{base}(X_{nbase}) \leq X_{max}$$

$S(X_{nbase})? \geq 0$!!! non linear inequality

Generalised Reduced Gradient (GRG)

- Variables are divided into two groups : Basic and Non Basic variables
- Use slack variables for inequalities
- Search by conjugate direction => unconstrained problem within bounds
  - Optimisation search is realised in the nbase domain, the basic variables are calculated by solving :

$$h\left(x_{base}, x_{nbase}\right) = 0 \Rightarrow \frac{\partial h}{\partial x} = \left(\frac{\partial h}{\partial x_{base}}, \frac{\partial h}{\partial x_{nbase}}\right) = \left(B_{base}, B_{nbase}\right)$$

With $\dfrac{\partial h}{\partial x_{base}} : n_h \times n_h \Rightarrow x_{base} = \varphi(x_{nbase})$

and $f(x_{base}, x_{nbase})$ becomes $F(x_{base}(x_{nbase}), x_{nbase})$

Reduced Gradient $\dfrac{\partial F}{\partial x_{nbase}} = \dfrac{\partial f}{\partial x_{nbase}} - z^T B_{nbase}$

Where $z^T = \left(B_{base}\right)^{-1} \dfrac{\partial f}{\partial x_{base}}$

Constrained programming : Lagrange equivalence

- Problem equivalence
  - Original : decision variable : $n_v + n_g$

$$\min f(\underline{x}) \qquad \underline{x}=[x_1,x_2,\ldots,x_v], \underline{s}=[s_1,s_2,\ldots,s_g]$$

Submitted to	$h_j(\underline{x}) = 0$	$j=1,\ldots,n_h$
	$g_k(\underline{x}) - \underline{s}_k^2 = 0$	$k=1,\ldots,n_g$

  - Lagrange equivalent : decision variables : $n_v + n_h + 2\, n_g$

$$\min \quad L(x,\lambda) = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x) + \sum_{k=1}^{n_g} \mu_k [g_k(x) - s_k^{\,2}]$$

decision variables	$x_{v,min} \le x_v \le x_{v,max}$	$v=1,\ldots,n_v$
n_g slag variables	$s_k \ge 0$	$k=1,\ldots,n_g$
$n_h + n_g$ new variables : Langrangian multipliers	$\lambda_j \le 0$	$j=1,\ldots,n_h$
	$\mu_k \le 0$	$k=1,\ldots,n_g$

Lagrange formulation

- the goal is to transform a constrained problem into a non constrained problem but with a higher problem size :

$$L(x,\lambda) = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x) + \sum_{k=1}^{n_g} \mu_k [g_k(x) - s_k{}^2]$$

  - to be minimised with respect to x, s and λ , μ
- There are therefore (nx+ nh +2 ng variables)
- Necessary optimality conditions

$$\lambda_j \leq 0 \quad j = 1,\ldots,n_h$$

$$\mu_k \leq 0 \quad k = 1,\ldots,n_g$$

$$\frac{\delta L(x^*, s^*, \lambda^*, \mu^*)}{\delta x_i} = 0 \qquad i = 1, ..., n_x$$

$$\frac{\delta L(x^*, s^*, \lambda^*, \mu^*)}{\delta s_g} = 0 \qquad g = 1, ..., n_g$$

$$\frac{\delta L(x^*, s^*, \lambda^*, \mu^*)}{\delta \lambda_h} = 0 \qquad h = 1, ..., n_h$$

$$\frac{\delta L(x^*, s^*, \lambda^*, \mu^*)}{\delta \mu_g} = 0 \qquad g = 1, ..., n_g$$

Conditions of optimality

$$\nabla_{x,s,\lambda,\mu} L(x, s, \lambda, \mu) = 0$$

Square system : $n_x + n_h + 2\ n_g$

$$\lambda_j \leq 0 \quad j = 1,\ldots,n_h$$

$$\mu_k \leq 0 \quad k = 1,\ldots,n_g$$

$$\frac{\partial L(x^*)}{\partial x_i} = \frac{\partial f(x^*)}{\partial x_i} + \sum_{j=1}^{n_h} \lambda_j \frac{\partial h_j(x^*)}{\partial x_i} + \sum_{k=1}^{n_g} \mu_k [\frac{\partial g_k(x^*)}{\partial x_i}] = 0 \qquad i = 1,\ldots,n_x$$

$$\frac{\partial L(x^*)}{\partial \lambda_j} = h_j(x) = 0 \qquad j = 1,\ldots,n_h$$

$$\frac{\partial L(x^*)}{\partial \mu_k} = g_k(x) - s_k^2 = 0 \qquad k = 1,\ldots,n_g$$

$$\frac{\partial L(x^*)}{\partial s_k} = 2\ \mu_k\ s_k = 0 \quad k = 1,\ldots,n_g$$

We calculate the variables, the slags and the Lagrange multipliers at the same time !

`<- Difficult to solve`

Lagrange multipliers

$$\lambda^*_j = \frac{\delta f^*}{\delta h_j} \qquad \mu^*_k = \frac{\delta f^*}{\delta g_k}$$

- Explanation λ_j
  - What is the penalty of a constraint ?
  - Without the constraint the objective function is better, the multipliers measures of how much
  - Important for inequality constraints

Example :

Minimise the time for transporting goods with a maximum of 3 trucks
Constraint : maximum 3 trucks

λ = what is the time benefit when adding 1 truck

Solving the equations step by Newton step

at iteration r

$$\nabla L \approx \nabla L^r + \mathbb{H}^r \Delta x_i^{r+1} = 0$$

$$\frac{\partial f(x^*)}{\partial x_i} + \sum_{j=1}^{n_h} \lambda_j \frac{\partial h_j(x^*)}{\partial x_i} + \sum_{k=1}^{n_g} \mu_k [\frac{\partial g_k(x^*)}{\partial x_i}] = 0$$

$$h_j(x) = 0$$

$$g_k(x) - s_k^2 = 0$$

$$2\ \mu_k\ s_k = 0$$

$$\begin{bmatrix} H^r & A^r & B^r & 0 \\ A^{rT} & 0 & 0 & 0 \\ B^{rT} & 0 & 0 & -2Is^r \\ 0 & 0 & Is^r & I\mu^r \end{bmatrix} \begin{bmatrix} \Delta X^{r+1} \\ \Delta \lambda^{r+1} \\ \Delta \mu^{r+1} \\ \Delta s^{r+1} \end{bmatrix} = -\nabla L^r$$

with

$$H^r = \left.\frac{\partial^2 f(x^r)}{\partial x_i \partial x_j}\right]^r + \sum_{j=1}^{n_h} \lambda_j \left.\frac{\partial^2 h_j(x^r)}{\partial x_i \partial x_j}\right]^r + \sum_{k=1}^{n_g} \mu_k \left.\frac{\partial^2 g_k(x^r)}{\partial x_i \partial x_j}\right]^r$$

$$A^r = \frac{\partial f(x^r)}{\partial x_i}, B^r = \frac{\partial g(x^r)}{\partial x_i}$$

Solving the equations set by Newton step

$$\nabla L \approx \nabla L^r + \mathbb{H}^r \Delta x_i^{r+1} = 0$$

$$\Delta x_i^{r+1} = -\mathbb{H}^{r\,-1} \nabla L^r$$

$$x_i^{r+1} = x_i^r + (1-q)\Delta x_i^{r+1}$$

Note : for linear problems

$\mathbb{H}^r$ is constant

q=0 (no relaxation)

Problem : find which s or μ= 0

$$\begin{bmatrix} H^r & A^r & B^r & 0 \\ A^{rT} & 0 & 0 & 0 \\ B^{rT} & 0 & 0 & -2Is^r \\ 0 & 0 & Is^r & I\mu^r \end{bmatrix} \begin{bmatrix} \Delta X^{r+1} \\ \Delta\lambda^{r+1} \\ \Delta\mu^{r+1} \\ \Delta s^{r+1} \end{bmatrix} = \nabla L^r$$

with

$$H^r = \left.\frac{\partial^2 f(x^r)}{\partial x_i \partial x_j}\right]^r + \sum_{j=1}^{n_h} \lambda_j \left.\frac{\partial^2 h_j(x^r)}{\partial x_i \partial x_j}\right]^r + \sum_{k=1}^{n_g} \mu_k \left.\frac{\partial^2 g_k(x^r)}{\partial x_i \partial x_j}\right]^r$$

$$A^r = \frac{\partial f(x^r)}{\partial x_i}, B^r = \frac{\partial g(x^r)}{\partial x_i}$$

Successive Quadratic Programming

- Quadratic approximation of the Langrangian
  - Quadratic Approximation of the objective function

$$\left[\nabla f\left(x^0\right)\right]^T \Delta x + (\Delta x)^T Q \Delta x$$

Q is the quadratic approximation of

$$\nabla^2 L = \nabla^2 (f - \lambda^T h - \mu^T g)$$

or other method to update Q

  - Linearised constraints
  - Defines
    - the direction
    - step lenght
  - Penality when the points is infeasible
  - Iterative procedure

Mixed Integer Problems

- MILP
  - Mixed Integer Linear Programming problems
    - Linear constraints & objective
    - continuous and integer variables
- MINLP
  - Mixed Integer Non Linear Programming problems
    - Non linear constraints & objective
    - continuous and integer variables

Mixed Integer Problems

- MILP
  - Mixed Integer Linear Programming problems
    - Linear constraints & objective
    - continuous and integer variables

$$0 \leq X_{variable} \leq 1 \quad \rightarrow \quad X_{variable} = 0 \quad or \quad X_{variable} = 1$$

2 inequality contraints

2 new equality contraints => 2 new problems
=> Objective is same or worst !

when $n_{variable} \geq 1$, there is the need to test all the combinations of "integerification" constraints in a systematic way

MILP : Branch & Bound Method

$$\min \quad F = 2x_1 - 3z_1 - 2z_2 - 3z_3$$
$$\text{s.c.} \quad x_1 + z_1 + z_2 + z_3 \geq 2$$
$$10x_1 + 5z_1 + 3z_2 + 4z_3 \leq 10$$
$$x_1 \geq 0$$
$$z_1, z_2, z_3 = 0,1$$

- Solve LP at each node
  - start with integer variables = continuous

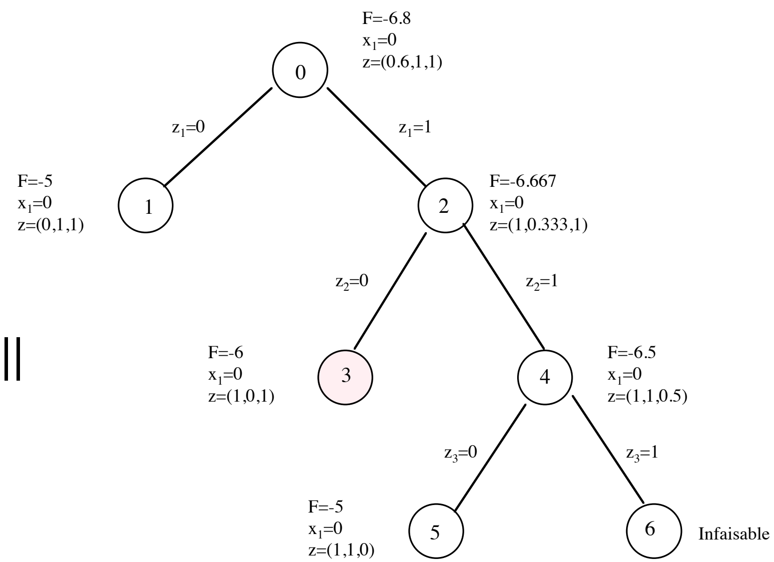
$$0 \leq z_i \leq 1 \quad \forall i = 1, ..., 3$$

- Progressively "integerify" by systematically adding constraints (Branch)

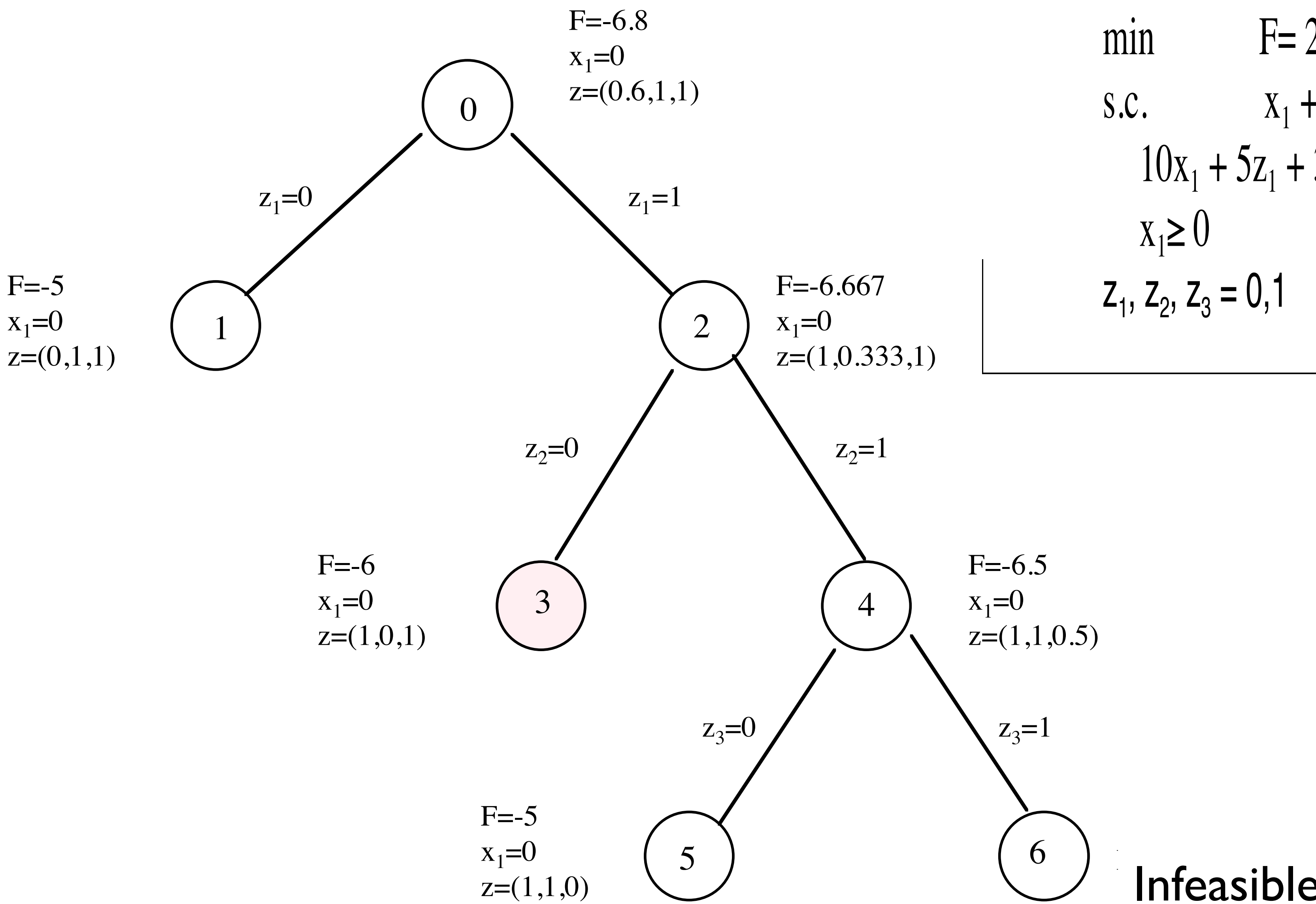
$$z_i = 1 \text{ or } 0 \text{ for i in Nodes} \quad 0 \leq z_j \leq 1 \quad \forall j \neq i$$

➡the objective is worsening

- when a set where all zi is integer
  - define the bound (the objective function will never be worse then this value
- re-explore the branches
  - cut the three exploration when the objective function reaches the bound



MILP : Branch & Bound Method



$$
\begin{aligned}
\min \quad & F = 2x_1 - 3z_1 - 2z_2 - 3z_3 \\
\text{s.c.} \quad & x_1 + z_1 + z_2 + z_3 \geq 2 \\
& 10x_1 + 5z_1 + 3z_2 + 4z_3 \leq 10 \\
& x_1 \geq 0 \\
& z_1, z_2, z_3 = 0,1
\end{aligned}
$$

MINLP : outer approximation

- Decomposition theorem
  - Partition variables in 2 sets
    - complicating set (integer)
    - continuous variables
  - Solve 2 problems
    - NLP with fixed integer (lower bound)
    - MILP : outer-approximate the objective function (upper bound)
  - Lower = upper => convergence
    - integer cut to avoid looping
- Problems :
  - NLP converge ?
  - Calculation time ?
  - Initial set feasible ?
  - Derivatives for MILP
    - outer approximation