Optimization Project in Energy
ENT306

Elise Grosjean
Ensta-Paris

Ensta-Paris
Institut Polytechnique de Paris

Supervised classification

Training a machine learning model to assign labels (or classes) to data points based on their features.

- $(x_i)_{i=1,\ldots,d}$ features
- $y_i \in \{0, 1\}$ associated labels

Objective

Learn a relationship between x and y that allows the model to predict the label y for new unseen data points x.

Supervised classification applications

1. **Predictive maintenance of energy equipment**
   Application: Predicting whether a piece of equipment (e.g. a wind turbine) is likely to break ($y = 1$) or not ($y = 0$) based on measured parameters such as temperature, pressure...

   Interpretation:
   x: Feature vector representing measurements from the equipment (e.g., temperature, pressure, etc.).
   w: Weights indicating the relative importance of each feature in predicting equipment failure.
   y: Binary indicator of failure (1 for failure, 0 for normal operation).

2. **Classification of buildings by energy performance**
   Application: Classifying buildings based on their energy efficiency (e.g., low-energy consumption buildings $y = 0$ versus energy-intensive buildings $y = 1$). Interpretation:

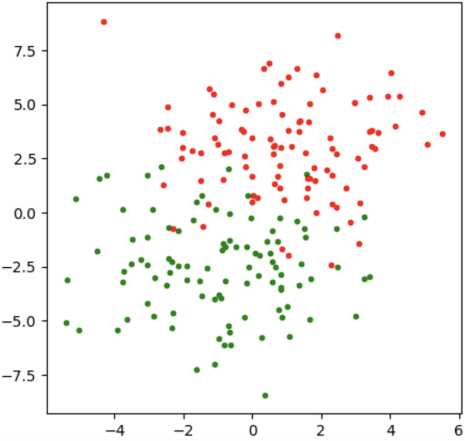
   x: Features describing the building (e.g., thermal insulation, heating type, surface area, etc.).
   w: Contributions of each feature to the likelihood of a building being energy-intensive.
   y: Energy performance classification (0 for efficient, 1 for energy-intensive).

Application example

1. **Predictive maintenance of energy equipment**
   Application: Predicting whether a piece of equipment (e.g. a wind turbine) is likely to break ($y = 1$) or not ($y = 0$) based on measured parameters such as temperature, pressure...

   Interpretation:
   x: Feature vector representing measurements from the equipment (e.g., temperature, pressure, etc.).
   w: Weights indicating the relative importance of each feature in predicting equipment failure.
   y: Binary indicator of failure (1 for failure, 0 for normal operation).

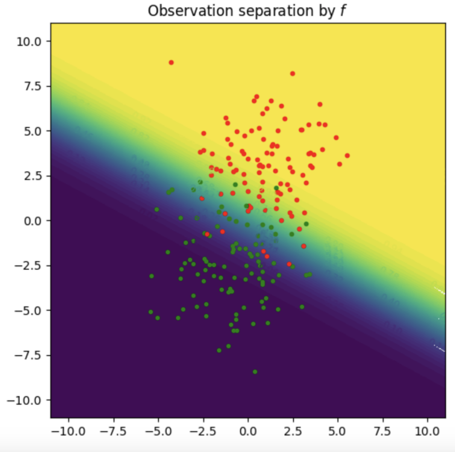
Data representation.
x_i is a green point if $y_i = 0$, red point if $y_i = 1$.



Application example

1. **Predictive maintenance of energy equipment**
   Application: Predicting whether a piece of equipment (e.g. a wind turbine) is likely to break ($y = 1$) or not ($y = 0$) based on measured parameters such as temperature, pressure...

   Interpretation:
   x: Feature vector representing measurements from the equipment (e.g., temperature, pressure, etc.).
   w: Weights indicating the relative importance of each feature in predicting equipment failure.
   y: Binary indicator of failure (1 for failure, 0 for normal operation).



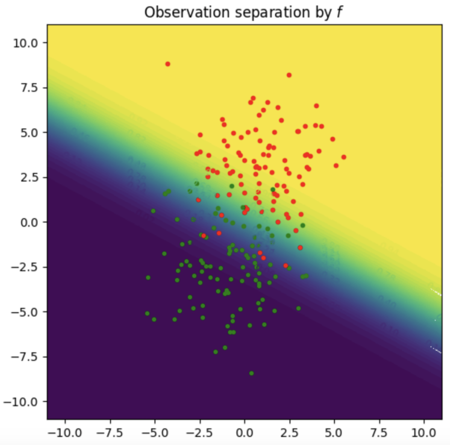
Binary supervised classification

Goal

Find the separation line thanks to the training data.
In other words, find the optimal weights $w = (w_1, w_2) \in \mathbb{R}^2$ and $b \in \mathbb{R}$ such that

$$\langle w, x \rangle + b = w^T x + b = 0.$$

Interpretation:
x: Feature vector representing measurements from the equipment (e.g., temperature, pressure, etc.).
w: Weights indicating the relative importance of each feature in predicting equipment failure.
y: Binary indicator of failure (1 for failure, 0 for normal operation).



Binary supervised classification

How does it work?

Binary supervised classification

Binary supervised classification $\rightarrow$ logistic regression

regression $=$ find a correlation between a binary variable and some observations thanks to an optimization problem

- Decision Trees
- K-Nearest Neighbors (k-NN)
- Probabilistic Models
- Neural Networks

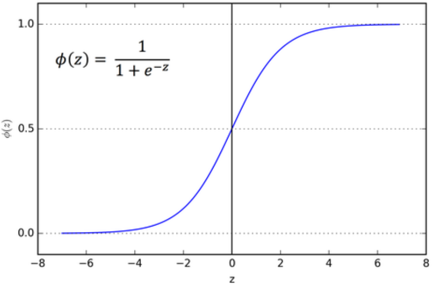
Binary supervised classification

Logistic regression

$$\begin{array}{ccc} x_1 & \rightarrow & f(\langle w_1, x_1 \rangle) \\ \dots & \rightarrow & \dots \\ x_n & \rightarrow & f(\langle w_n, x_n \rangle) \end{array}$$

The sigmoid function σ is often used (for f) in logistic regression:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



Decision rule:

- if $\sigma(\langle w, x \rangle) > 0.5, y = 1$
- if $\sigma(\langle w, x \rangle) < 0.5, y = 0$
- if $\langle w, x \rangle >> 0, P(y = 1|x) \simeq 1$
- if $\langle w, x \rangle << 0, P(y = 1|x) \simeq 0$

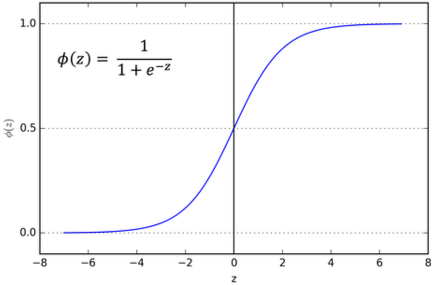
Binary supervised classification

Logistic regression

$$\begin{array}{ccc} x_1 & \to & f(\langle w_1, x_1 \rangle) \\ \dots & \to & \dots \\ x_n & \to & f(\langle w_n, x_n \rangle) \end{array}$$

The sigmoid function σ is often used (for f) in logistic regression:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



- Likelihood function: $log\left(\frac{P(y=1|x)}{P(y=0|x)}\right) = \langle w, x \rangle$
- $P(y = 1|x) = \sigma(\langle w, x \rangle)$

Likelihood function

$$P(y=1|x) = \frac{1}{1+e^{-\langle w,x\rangle}} := \sigma(\langle w,x\rangle)$$
$$P(y=0|x) = 1 - \frac{1}{1+e^{-\langle w,x\rangle}} := 1-\sigma(\langle w,x\rangle)$$

Log-loss function:

$$f(w) = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\sigma(\langle w,x_i\rangle)) + (1-y_i)log(1-\sigma(\langle w,x_i\rangle)) + \lambda\frac{1}{2}\|w\|^2$$

- y_i : true label (0 or 1),
- $\sigma(\langle w,x_i\rangle)$: probablity predicted by the model to get $y_i = 1$.

Likelihood optimization

MINIMIZE the log-loss function:

$$f(w) = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\sigma(\langle w, x_i\rangle)) + (1-y_i) log(1-\sigma(\langle w, x_i\rangle)) + \lambda\frac{1}{2}\|w\|^2$$

$\rightarrow$ Gradient descent algorithm !!!

Likelihood optimization

MINIMIZE the log-loss function:

$$f(w) = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\sigma(\langle w, x_i\rangle)) + (1-y_i) log(1-\sigma(\langle w, x_i\rangle)) + \lambda\frac{1}{2}\|w\|^2$$

Remarks:

$\|w\|$ too high is not good:

- if $\langle w, x\rangle >> 0, P(y=1|x) \simeq 1$
- if $\langle w, x\rangle << 0, P(y=1|x) \simeq 0$

Indeed, if $\|w\|$ is high:

- then $\sigma(\langle w, x\rangle)$ becomes to close to 0 or 1, which entails numerical instability problems (NaN).
- then ill-conditioning (problems with the hessian)
- If $\sigma(\langle w, x\rangle) \simeq 0$ or 1: increase too much confidence in our model.

Likelihood optimization

$$\min_w -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\sigma(\langle w, x_i\rangle)) + (1-y_i)log(1-\sigma(\langle w, x_i\rangle))$$

such that $0 \leq \sigma(\langle w, x_i\rangle) \leq 1$ with a penalization for values where $\sigma(\langle w, x_i\rangle) = 0$ or 1. Instead of the L^2 regularization, we can use the

Interior Point Method (IPM), often used in energy application and find the minimum of:

$$f(w) = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\sigma(\langle w, x_i\rangle)) + (1-y_i)log(1-\sigma(\langle w, x_i\rangle)) + C\sum_{j=1}^{d} log(1+|w_j|^2),$$

where C is the penalization parameter.
$\rightarrow$ Gradient descent algorithm !!!

1. Interior point method
   - Nonnegative variables

Interior Point Method (IPM)

IPM is a nonlinear optimization algorithm that is particularly effective for solving **large-scale constrained problems**.
It is widely used in energy system optimization with numerous constraints, e.g.:

- Generator production limits
- Electricity transmission constraints (physical laws of the grid)
- Environmental constraints (e.g., CO_2 emissions)

IPM naturally handles these constraints by preventing them from becoming active too early (thanks to the logarithmic barrier function).

Nonnegative variables

Optimization problem. Consider

$$\inf_{x \in \mathbb{R}^n} f(x), \quad \text{subject to: } x_i \geq 0, \ \forall i = 1, \ldots, n.$$

Barrier function: given $c > 0$, consider the function B_c defined by

$$B_c(x) = f(x) - c \sum_{i=1}^{n} \log(x_i),$$

for all $x \in \mathbb{R}^n_{>0} := \{y \in \mathbb{R}^n \mid y_i > 0, \ \forall i = 1, \ldots, n\}$.

Main idea: approximate (P) by

$$\inf_{x \in \mathbb{R}^n_{>0}} B_c(x). \qquad (P_c)$$

Nonnegative variables

General comments.

- We have: $-\log(x_i) \to \infty$ as $x_i \to 0$.
  $\to$ Feasible points close to the boundary of the feasible set are **penalized** (whatever the value of c).

- A strong modification of the cost function on the feasible set is undesirable.
  $\to$ The barrier parameter c should be ideally **very small**.

- Problem (P_c) can be solved with methods for **unconstrained optimization**.
  The standard stepsize rules (Armijo,...) prevents us from getting to close to the boundary.
  Ill-conditioning for small values of c.

Nonnegative variables

Example 1.

Consider

$$\inf_{x\in\mathbb{R}} x, \quad \text{subject to: } x \geq 0.$$

- Solution: $\bar{x} = 0$.
- Barrier function: $B_c(x) = x - c\log(x)$.

$$\nabla B_c(x) = 1 - \frac{c}{x} = 0 \iff x = c.$$

  Since B_c is convex, $x_c := c$ is the global solution to (P_c).
- We have $x_c \xrightarrow[c\to 0]{} 0$.

Nonnegative variables

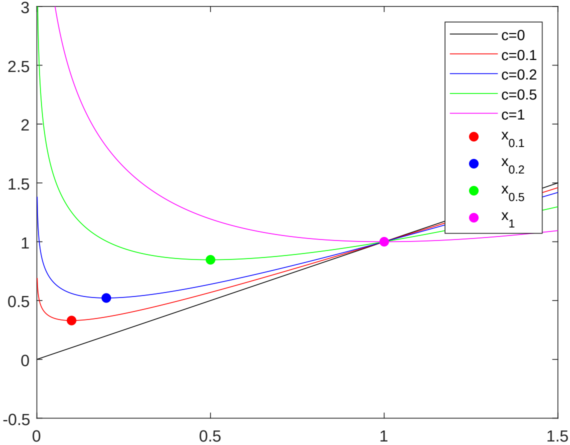


Figure: Level-sets $B_c(\cdot)$, for various values of c

Nonnegative variables

Example 2.
Consider

$$\inf_{(x,y)\in\mathbb{R}^2} \frac{1}{2}(y-1)^2 + x, \quad \text{subject to: } \begin{cases} x \geq 0 \\ y \geq 0. \end{cases}$$

- Solution: $(\bar{x}, \bar{y}) = (0, 1)$.
- Solution to the barrier problem: $(x_c, y_c) = \left(c, \dfrac{1+\sqrt{1+4c}}{2}\right)$.

Nonnegative variables

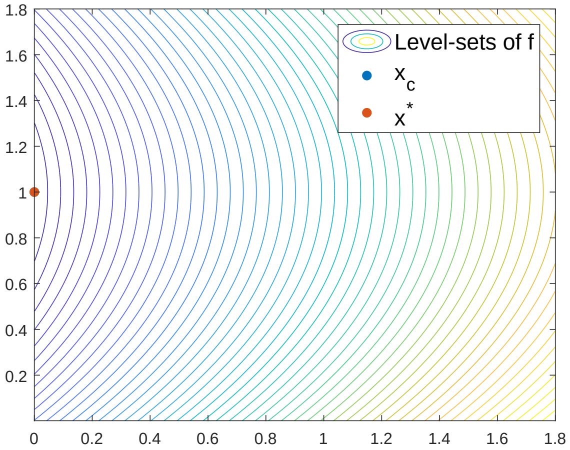


Figure: Level-sets of $f(\cdot)$

Nonnegative variables

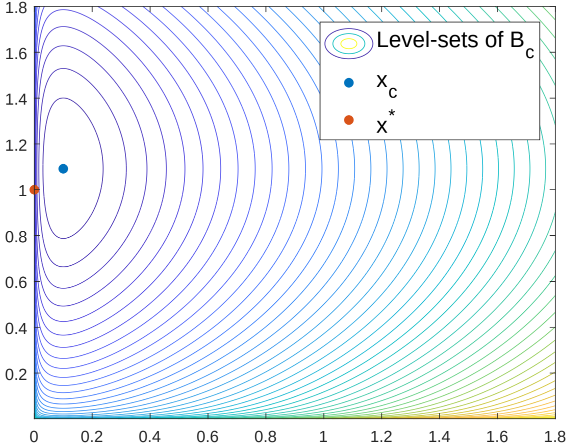


Figure: Level-sets of f, for $c = 0.1$

Nonnegative variables

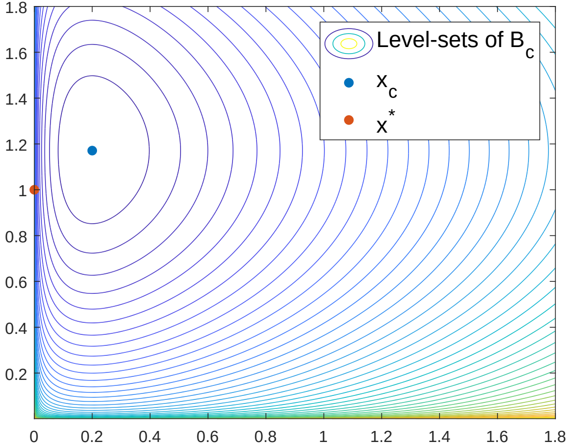


Figure: Level-sets of $B_c(\cdot)$, for $c = 0.2$

Nonnegative variables

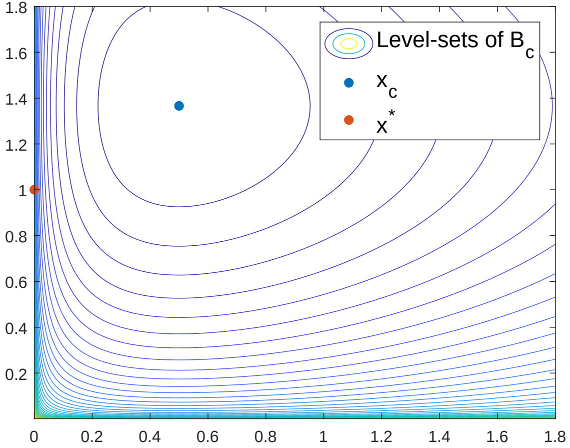


Figure: Level-sets of $B_c(\cdot)$, for $c = 0.5$

Nonnegative variables

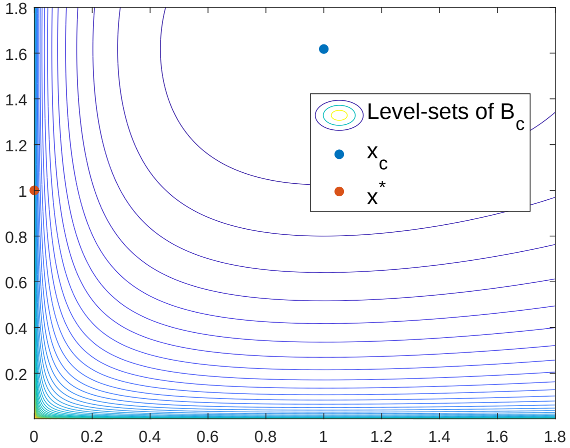


Figure: Level-sets of $B_c(\cdot)$, for $c = 1$

Nonnegative variable

Interpretation with the KKT conditions.

Let $\bar{x}$ be a solution to (P). Let $\bar{\lambda} \in \mathbb{R}^n$ be the associated Lagrange multiplier.

- Lagrangian: $L(x, \lambda) = f(x) - \langle \lambda, x \rangle$. Stationarity condition:

$$\nabla_x L(\bar{x}, \bar{\lambda}) = \nabla f(\bar{x}) - \bar{\lambda} = 0.$$

- Sign condition: $\bar{\lambda}_i \geq 0$.
- Complementarity condition: $\bar{x}_i > 0 \Longrightarrow \bar{\lambda}_i = 0$.
  Equivalently: $\bar{x}_i \bar{\lambda}_i = 0$.

Nonnegative variable

Optimality conditions for the barrier problem.

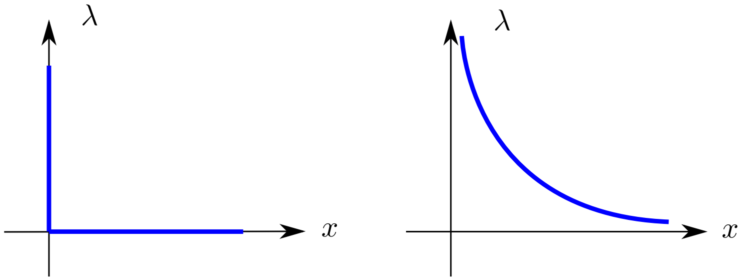
- For any $x \in \mathbb{R}^n_{>0}$ we denote $\frac{1}{x} = \left(\frac{1}{x_1}, \ldots, \frac{1}{x_n}\right)$.
- Let x_c be a solution to (P_c). We have

$$\frac{\partial B_c}{\partial x_i}(x_c) = \frac{\partial f}{\partial x_i}(x_c) - \frac{c}{x_i} = 0.$$

  Therefore $\nabla B_c(x_c) = \nabla f(x_c) - \dfrac{c}{x_c} = \nabla_x L(x_c, \dfrac{c}{x_c})$.
- Define $\lambda_c = \frac{c}{x_c} \in \mathbb{R}^n_{>0}$.
  The pair (x_c, λ_c) satisfies the KKT conditions approximately:

$$\nabla L(x_c, \lambda_c) = 0, \quad x_{c,i}\lambda_{c,i} = c, \ \forall i \in \{1, \ldots, n\}.$$

Nonnegative variables



$\{(x,\lambda)\,|\,x \geq 0,\ \lambda \geq 0,\ x\lambda = 0\}$ $\qquad$ $\{(x,\lambda)\,|\,x \geq 0,\ \lambda \geq 0,\ x\lambda = c\}$

Figure: Regularization of the complementarity condition

IPM

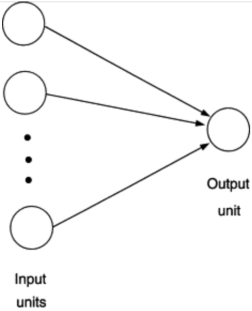
$$f(w) = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\sigma(\langle w, x_i\rangle)) + (1-y_i)log(1-\sigma(\langle w, x_i\rangle)) + C\sum_{j=1}^{d} log(1+|w_j|^2),$$

Exercise 5

Code the IPM fonction and compare the level-set obtained with the functions f without penalization and with the L^2 penalization.

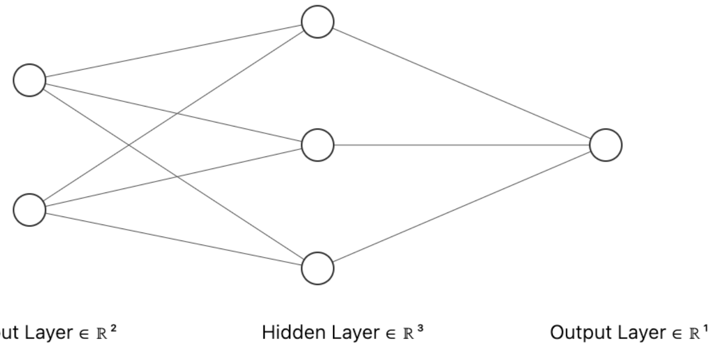
Multilayer perceptron

Perceptron

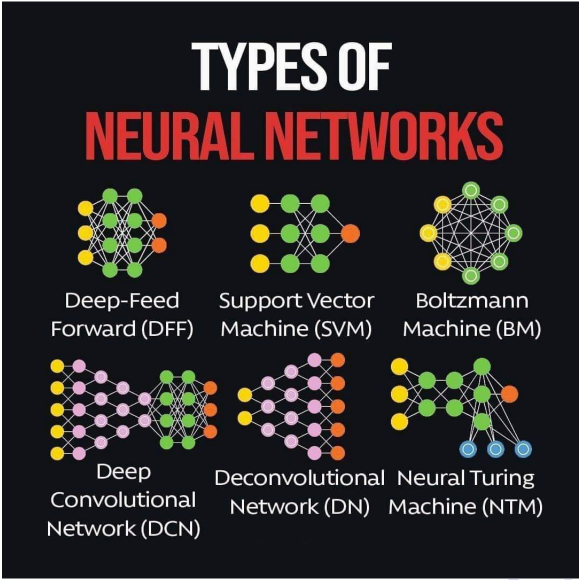


Multilayer perceptron

One hidden layer



Neural networks



BFGS

Quasi-Newton

$$H_k = H_{k-1} + \frac{(y_{k-1} - H_{k_1} d_{k-1}) d_{k-1}^T}{d_{k-1}^T d_{k-1}}$$

with $d_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = \nabla f(x_k) - \nabla f(x_{k-1})$

BFGS

Quasi-Newton

$$H_k^{-1} = \left(I - \frac{\bar{d}_{k-1} y_{k-1}^T}{\bar{d}_{k-1}^T y_{k-1}}\right) H_{k-1}^{-1} \left(I - \frac{\bar{d}_{k-1} y_{k-1}^T}{\bar{d}_{k-1}^T y_{k-1}}\right) + \frac{\bar{d}_{k-1} \bar{d}_{k-1}^T}{\bar{d}_{k-1}^T y_{k-1}}$$

with $\overline{d}_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = \nabla f(x_k) - \nabla f(x_{k-1})$
$x_k = x_{k-1} + \alpha d$
$\overline{d} = \alpha d$

- Update d
- Update α thanks to Armijo/ Wolfe
- Update x
- Update $y, \overline{d}$
- Find H^{-1}