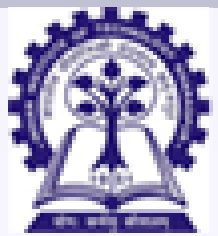


GLOBAL INITIATIVE OF ACADEMIC NETWORKS
(GIAN)

Solving linear systems and computing generalized inverses using recurrent neural networks

9th June to 19th June 2025
IIT Indore.

Gradient-based algorithms for solving nonlinear optimization

Predrag S. Stanimirović

University of Niš, Faculty of Sciences and Mathematics,
Department of Computer Sciences,
Niš, Serbia

June 9, 2025

Lecture 2:

Lecture 2:
- Definitions of gradient and Hessian, matrix and vector norms.

- Basic principles and methods in nonlinear unconstrained optimization.

- Overview of line search methods.

- Overview of gradient-descent methods, Newton method and quasi-Newton methods, conjugate gradient nonlinear optimization methods.

Introduction

The nonlinear unconstrained optimization problem is given in the general form

$$\min f(\mathbf{x}),\ \mathbf{x} \in \mathbb{R}^n, \tag{1}$$

where $f(\mathbf{x}) = f(x_1, \ldots, x_n)$ is given multivariable real and nonlinear objective function.

A local minimum $\mathbf{x}^*$ is defined as an element for which there exists some $\delta > 0$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ holds in the δ-neighborhood around $\mathbf{x}^*$:

$$\forall \mathbf{x} \in A \text{ where } \|\mathbf{x} - \mathbf{x}^*\| \leq \delta.$$

A global minimum is a point where the function value is smaller than or equal to values at all other feasible points.

The basic intuition

The basic intuition behind descent direction can be illustrated by a hypothetical scenario.
A person is trying to get down (i.e., trying to find a minimum).
There is heavy fog such, and visibility is extremely low.
Thus, he must rely on local information to determine the descent direction in order to locate the minimum.

Figure 1: Intuition behind descent direction algorithms.

- The most frequently used general iterative scheme aimed to solve the multivariable unconstrained minimization problem (1) is **the line search method**:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k,$$

where $\mathbf{x}_{k+1}$ is a new iterative point, $\mathbf{x}_k$ is the previous iterative point, $t_k > 0$ is a step length, and $\mathbf{d}_k$ is a search direction.

The line search algorithm is based on an appropriate descent direction $\mathbf{d}_k$ along which the objective function f will be reduced and then computes a step size that determines how far $\mathbf{x}_{k+1}$ should move along that direction.

Algorithm 1 Global line search algorithm

Require: Objective $f(\mathbf{x})$, initial point $\mathbf{x}_0 \in \mathbb{R}^n$ and the tolerance $0 < \varepsilon \ll 1$.

1: $k := 0$.
2: **while** stopping criterion is not satisfied **do**
3: Determine the vector of the search direction $\mathbf{d}_k$.
4: Compute the step length t_k such that

$$f(\mathbf{x}_k + t_k\,\mathbf{d}_k) < f(\mathbf{x}_k).$$

5: Compute the new approximation $\mathbf{x}_{k+1} := \mathbf{x}_k + t_k\mathbf{d}_k$.
6: $k := k + 1$
7: **end while**
8: **return** $\mathbf{x}_{k+1}, f(\mathbf{x}_{k+1})$.

Definitions of gradient and Hessian, matrix and vector norms

In further, the following notation is used:

$$\mathbf{g}(\mathbf{x}) = \bigtriangledown f(\mathbf{x}), \quad G(\mathbf{x}) = \bigtriangledown^2 f(\mathbf{x}),$$
$$\mathbf{g}_k = \bigtriangledown f(\mathbf{x}_k), \quad G_k = \bigtriangledown^2 f(\mathbf{x}_k),$$

where $\bigtriangledown f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_n}\right]^T$ denotes the gradient of f

and $\bigtriangledown^2 f(\mathbf{x}) = H(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x}_1 \partial x_n} \\ \vdots & \ddots & \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_n} \end{bmatrix}$ denotes the Hessian of f.

The gradient vector $\nabla f(\mathbf{x})$ at every point $\mathbf{x} \in \mathbb{R}^n$ is normal to the level surface with constant value $f(\mathbf{x})$ and passes through the given point $\mathbf{x}$.

The gradient vector $\mathbf{g}$ at the point $\mathbf{x}$ gives the direction of fastest increase of f at $\mathbf{x}$. In this way, negative gradient points in the direction of the steepest descent.

The basic intuition

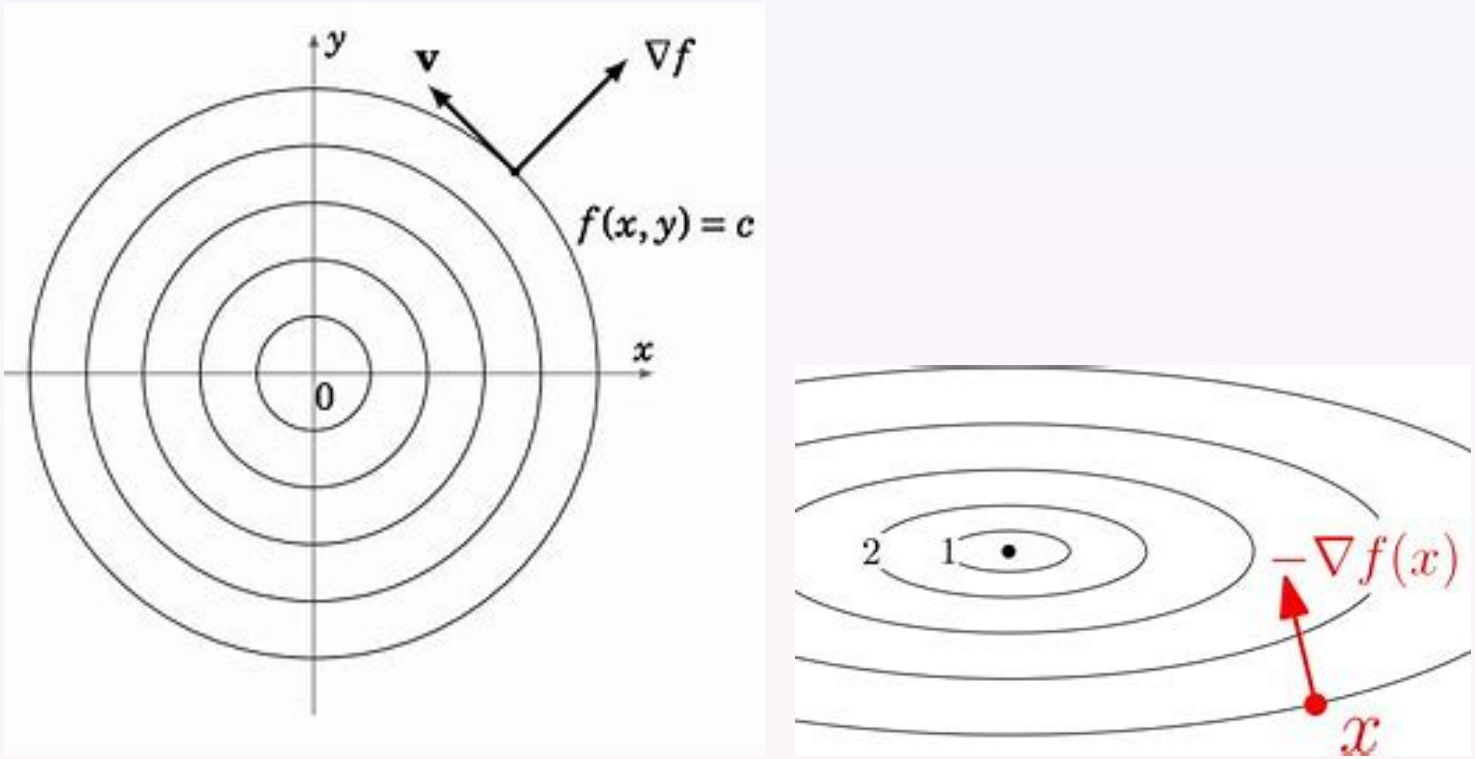


Figure 2: Gradient and anti-gradient directions.

gradient descent (GD) method

A general rule for defining appropriate search directions arises from Taylor expansion of the function f at the point $\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k$:

$$f(\mathbf{x}_{k+1}) := f(\mathbf{x}_k + t_k \mathbf{d}_k) \approx f(\mathbf{x}_k) + t_k \mathbf{g}_k^{\mathrm{T}} \mathbf{d}_k. \tag{2}$$

According to (2), the descent search direction $\mathbf{d}_k$ must satisfy the descent condition

$$\mathbf{g}_k^{\mathrm{T}} \mathbf{d}_k < 0.$$

The descent direction can be computed using various methods.

Each descent direction leads to a new class of optimization methods.

gradient descent (GD) method

The step size t_k can be determined either exactly or inexactly.

The **exact line search** (ELS), which assumes the unidimensional function with respect to the step size

$$\Phi(t) := f(\mathbf{x}_k + t\mathbf{d}_k) \tag{3}$$

and the step-size is defined after the unidimensional optimization

$$f(\mathbf{x}_k + t_k\mathbf{d}_k) = \min_{t>0} \Phi(t). \tag{4}$$

The ELS rule may give smallest value $f(\mathbf{x}_{k+1})$. However, ELS is too expensive in practice, especially in situations when $\mathbf{x}_k$ is far from the exact solution.

Conversely, an **inexact line search** offers an efficient method for choosing a step length that sufficiently decreases the objective function, such that

$$f(\mathbf{x}_k + t_k\,\mathbf{d}_k) < f(\mathbf{x}_k).$$

Algorithm 2 is one of algorithms exploited to determine the step length t_k.

Algorithm 2 The backtracking inexact line search.

Require: Goal function $f(\mathbf{x})$, a vector $\mathbf{d}_k$ at $\mathbf{x}_k$ and real quantities $0 < \sigma < 0.5$, $\beta \in (0, 1)$.

1: $t = 1$.
2: While $f(\mathbf{x}_k + t\mathbf{d}_k) > f(\mathbf{x}_k) + \sigma t \mathbf{g}_k^{\mathrm{T}} \mathbf{d}_k$, perform $t := t\beta$.
3: Output: $t_k = t$.

gradient descent (GD) method

- In the **gradient descent** (GD) method, the direction $\mathbf{d}_k$ is defined by $\mathbf{d}_k = -\mathbf{g}_k$.

Such a choice simplifies the general line search iterations $\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k$ into the *gradient descent* (GD) iterative scheme

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \mathbf{g}_k. \tag{5}$$

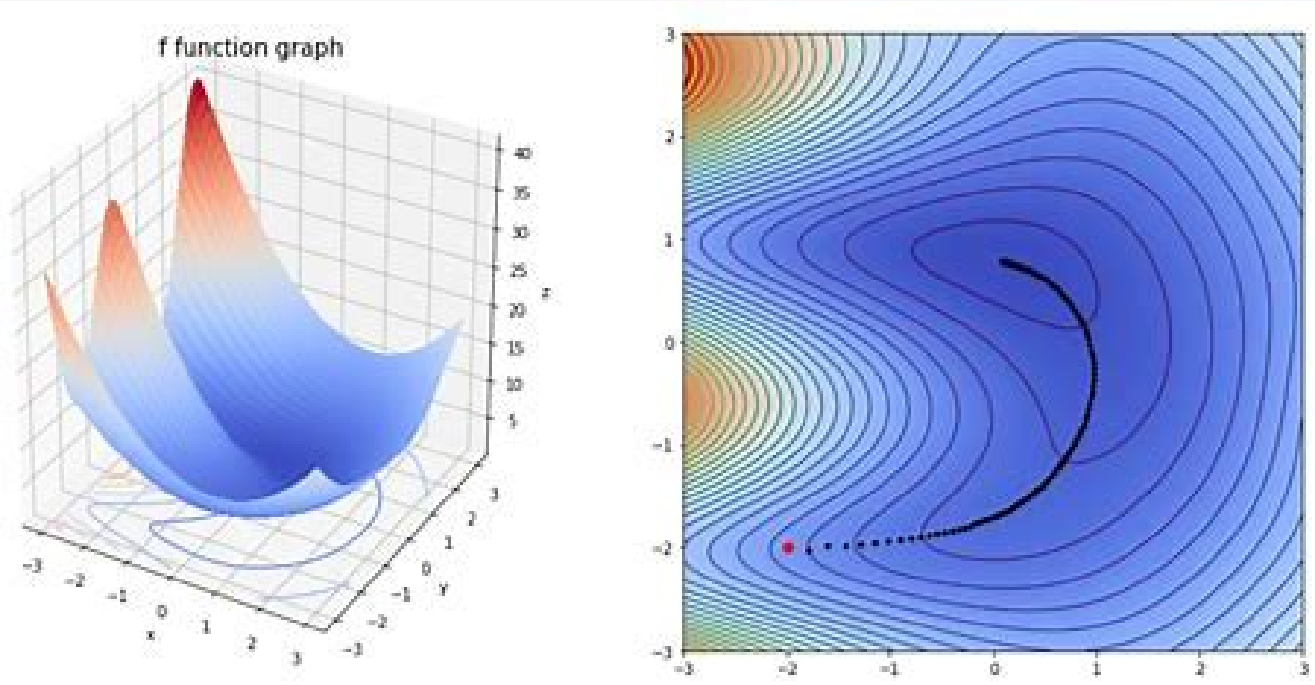


Figure 3: Illustration of the gradient descent.

stopping criteria

Common stopping criteria include
a) Small gradient norm: $\|\mathbf{g}_k\| \leq \varepsilon$
b) Small change in parameters:

$$\frac{\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|}{\|\mathbf{x}^{(k)}\|} \leq \varepsilon_1.$$

c) Small change in loss function:

$$\left| \frac{f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)}{f(\mathbf{x}_k)} \right| \leq \varepsilon_2.$$

d) Maximum number of iterations:
e) Early stopping:
Monitoring performance on a validation set and stopping training when the performance starts to degrade (e.g., the loss increases or the accuracy decreases). It can help avoid overfitting.

gradient descent (GD) method

Advantages and disadvantages of GD methods can be summarized as follows.

1. GD methods are globally convergent to a local minimizer, regardless of the starting point.
2. Many optimization methods switch to GD rule in the cases when they do not make sufficient progress to the convergence.
3. The convergence is linear and usually very slow.
   The steepest descent algorithm sometimes leads to orthogonal consecutive directions and possibly zig-zag trajectory.

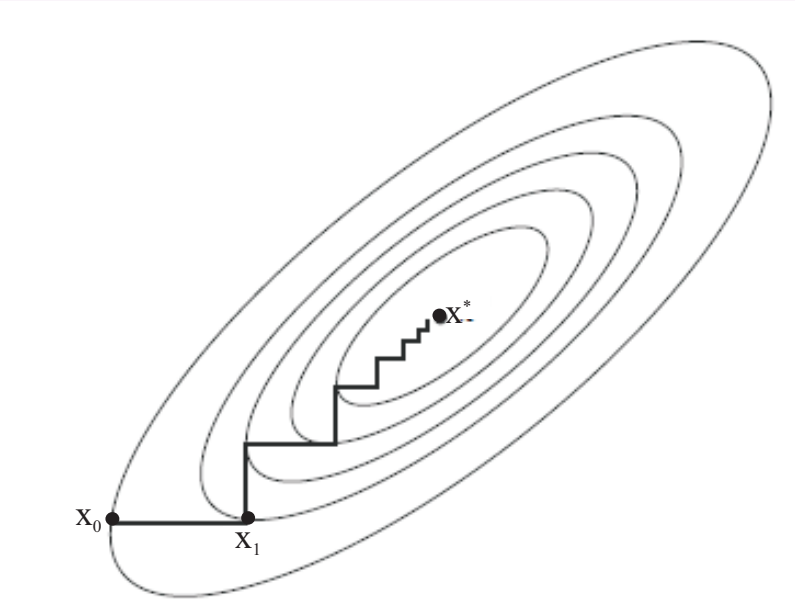


Figure 4: Zigzagging in the steepest descent method

Newton method and modifications

- The **pure Newton method** (without line search) for minimization of a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined using a quadratic approximation of $f(\mathbf{x}_{k+1})$:

$$\Phi(\mathbf{d}) := f(\mathbf{x}_k + \mathbf{d}) \approx f(\mathbf{x}_k) + \mathbf{g}_k^{\mathrm{T}}\mathbf{d} + \frac{1}{2}\mathbf{d}^{\mathrm{T}} G_k \mathbf{d}. \tag{6}$$

The solution $\mathbf{d}_k = \min_{\mathbf{d}}(\Phi(\mathbf{d}))$ is obtained using the matrix calculus

$$\frac{\partial \mathbf{d}^{\mathrm{T}} G_k \mathbf{d}}{\partial \mathbf{d}} = (G_k + G_k^{\mathrm{T}})\mathbf{d} = 2G_k\mathbf{d}$$
$$\frac{\partial \mathbf{g}_k^{\mathrm{T}}\mathbf{d}}{\partial \mathbf{d}} = \mathbf{g}_k,$$

which implies $\frac{\partial \Phi(\mathbf{d})}{\partial \mathbf{d}} = \mathbf{g}_k + G_k\mathbf{d}$ and further

$$\frac{\partial \Phi(\mathbf{d})}{\partial \mathbf{d}} = 0 \Longleftrightarrow \mathbf{g}_k + G_k\mathbf{d} = 0 \Longleftrightarrow \mathbf{d} = -G_k^{-1}\mathbf{g}_k.$$

So, the pure Newton method is defined by

$$\mathbf{x}_{k+1} = \mathbf{x}_k - G_k^{-1}\mathbf{g}_k. \tag{7}$$

More efficient approach is to generate $\mathbf{d}_k := G_k^{-1}\mathbf{g}_k$ by solving the system of linear equations $G_k\mathbf{d} = -\mathbf{g}_k$ with respect to $\mathbf{d}$.

Newton method and modifications

- The **Newton method with line search** uses an appropriate step-size t_k in the pure Newton method (7):$\mathbf{x}_{k+1} = \mathbf{x}_k - G_k^{-1}\mathbf{g}_k$ with the aim to ensure global stability. The resulting iterations are of the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k G_k^{-1}\mathbf{g}_k, \tag{8}$$

  wherein the step-size t_k is computed performing a line search.

The **Newton method** is fast, with the second order convergence rate.
But, it exhibits three major drawbacks in practical applications.

1. The descent (and convergence) may not be achieved if the Newton iterations (7) are started far away from the local minimizer.
2. Another drawback is numerically expensive requirement to compute the second derivative matrix (Hessian) and its inverse in every iteration.Moreover, the second derivatives may be sometimes unavailable.
3. The main disadvantages of the Newton method are the possibility that the Hessian G_k is not positive definite.

Newton method and modifications

Due to the mentioned drawbacks, numerous modifications of the Newton method were created, which can be globally divided into two large groups: modified Newton's methods and **quasi-Newton** (QN) methods.
Main principle in QN methods is to use a symmetric $n \times n$ approximation B_k of the Hessian G_k and $H_k = B_k^{-1}$ as an approximation of the inverse Hessian G_k^{-1}.

- Updates of the matrix H_k are defined based on the *secant equation*:

$$H_{k+1}\mathbf{y}_k = \mathbf{s}_k \Longleftrightarrow B_{k+1}\mathbf{s}_k = \mathbf{y}_k, \tag{9}$$

where $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$.

The next iteration is defined by

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k H_k \mathbf{g}_k. \tag{10}$$

Newton method and modifications

Main Quasi-Newton updates.

1. The symmetric rank-one update (SR1 update)

$$H_{k+1} = H_k + \frac{(\mathbf{s}_k - H_k\mathbf{y}_k)(\mathbf{s}_k - H_k\mathbf{y}_k)^{\mathrm{T}}}{(\mathbf{s}_k - H_k\mathbf{y}_k)^{\mathrm{T}}\mathbf{y}_k}.$$

2. DFP update:

$$H_{k+1} = H_k - \frac{H_k\mathbf{y}_k\mathbf{y}_k^T H_k}{\mathbf{y}_k^{\mathrm{T}} H_k\mathbf{y}_k} + \frac{\mathbf{s}_k\mathbf{s}_k^{\mathrm{T}}}{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k}.$$

3. BFGS update:

$$H_{k+1} = H_k + \frac{(\mathbf{s}_k - H_k\mathbf{y}_k)\mathbf{s}_k^{\mathrm{T}} + \mathbf{s}_k(\mathbf{s}_k - H_k\mathbf{y}_k)^{\mathrm{T}}}{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k}.$$

Algorithm 3 A general quasi-Newton algorithm

Require: Objective $f(\mathbf{x})$, initial point $\mathbf{x}_0 \in \mathbb{R}^n$ and the tolerance $0 < \varepsilon \ll 1$.

1: $k := 0$.
2: **while** stopping criteria are not satisfied **do**
3: Compute $\mathbf{d}_k = H_k \mathbf{g}_k$.
4: Compute the step length t_k such that

$$f(x_k + t_k\, d_k) < f(x_k).$$

5: Compute the new approximation $\mathbf{x}_{k+1} := \mathbf{x}_k + t_k \mathbf{d}_k$
6: $k := k + 1$
7: Update H_k into H_{k+1} such that the quasi-Newton equation $H_{k+1}\mathbf{y}_k = \mathbf{s}_k$, ($\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$), holds
8: **end while**

Ensure: $\mathbf{x}_{k+1}, f(\mathbf{x}_{k+1})$

Improved Gradient Descent (IGD) methods

Our interest is a class of **accelerated line search methods** for solving (1) which is based on the iterative principle

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \theta_k t_k \mathbf{d}_k, \tag{11}$$

where t_k is the primary step length real parameter, $\mathbf{d}_k$ is a descent direction and $\theta_k > 0$ is an acceleration parameter.

- An idea of constructing the acceleration (scaling) parameter in GD methods is presented in
  [SM] [P.S. Stanimirović, M.B. Miladinović, *Accelerated gradient descent methods with line search*, Numer. Algor. **54** (2010), 503–520].

  The proposed method is called *SM method*.

Main idea used in the *SM* algorithm construction is approximation of the inverse Hessian in Quasi-Newton methods by a constant diagonal matrix:

$$H_k = \gamma_k^{-1} I, \quad \gamma_k \in \mathbb{R}, \tag{12}$$

which reduces the general QN scheme $\mathbf{x}_{k+1} = \mathbf{x}_k - t_k H_k \mathbf{g}_k$ into the **improved gradient descent** (IGD) method with the line search, so called *SM* method:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \gamma_k^{-1} \mathbf{g}_k \tag{13}$$

SM method

The steplength $\gamma?k$ is computed using the Taylor development and t_k is computed using the backtracking line search procedure.

Computing γ_k.

Taylor's approximation of the function f at the point $\mathbf{x}_{k+1}$, computed by means of the SM method (13):

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \gamma_k^{-1} \mathbf{g}_k$$

is given by

$$f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k) - t_k \mathbf{g}_k^T \gamma_k^{-1} \mathbf{g}_k + \frac{1}{2} t_k^2 (\gamma_k^{-1} \mathbf{g}_k)^T \nabla^2 f(\xi) \gamma_k^{-1} g_k, \tag{14}$$

where $\xi \in [\mathbf{x}_k, \mathbf{x}_{k+1}]$ is defined by

$$\xi = \mathbf{x}_k + \alpha(\mathbf{x}_{k+1} - \mathbf{x}_k) = \mathbf{x}_k - \alpha t_k \gamma_k^{-1} \mathbf{g}_k, \ 0 \leq \alpha \leq 1. \tag{15}$$

Having in mind that the distance between $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ is small enough (using the local character of searching) we can take $\alpha = 1$ in (15) and get the approximation $\xi = \mathbf{x}_{k+1}$. Thus we obtain

$$\nabla^2 f(\xi) = \gamma_{k+1} I. \tag{16}$$

SM method

Now, from (14) and (16), the Taylor expansion of $f(\mathbf{x}_{k+1})$ becomes

$$f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k) - t_k \gamma_k^{-1} \|\mathbf{g}_k\|^2 + \frac{1}{2} t_k^2 \gamma_{k+1} \gamma_k^{-2} \|\mathbf{g}_k\|^2, \tag{17}$$

and later

$$\gamma_{k+1} = 2\gamma_k \frac{\gamma_k \left[f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)\right] + t_k \|\mathbf{g}_k\|^2}{t_k^2 \|\mathbf{g}_k\|^2}. \tag{18}$$

The condition $\gamma_{k+1} > 0$ is ultimate. In the case $\gamma_{k+1} < 0$ we take $\gamma_{k+1} = 1$.

In conclusion, the *SM* method originated in [SM] was defined by

$$x_{k+1} = x_k - t_k (\gamma_k^{SM})^{-1} g_k, \tag{19}$$

where

$$\gamma_{k+1}^{SM} = \upsilon \left(2\gamma_k^{SM} \frac{\gamma_k^{SM} \Delta_k + t_k \|g_k\|^2}{t_k^2 \|g_k\|^2} \right),$$

such that $\Delta_k := f_{k+1} - f_k$ and

$$\upsilon(x) = \begin{cases} x, & x > 0 \\ 1, & x \leq 0. \end{cases}$$

Computing t_k.
In order to derive an upper bound for the backtracking we analyze the function derived from the Taylor expansion of $f(\mathbf{x}_{k+2})$ by replacing t_k by a variable t:

$$\Phi_{k+1}(t) = f(\mathbf{x}_{k+1}) - t\gamma_{k+1}^{-1}\|\mathbf{g}_k\|^2 + \frac{1}{2}t^2\gamma_{k+1}^{-1}\|\mathbf{g}_k\|^2.$$

The function $\Phi_{k+1}(t)$ is convex in the case $\gamma_{k+1} > 0$.
Also, it is obvious that $\Phi_{k+1}(0) = f(\mathbf{x}_{k+1})$ as well as $\Phi'_{k+1}(t) = (t-1)\gamma_{k+1}^{-1}\|\mathbf{g}_{k+1}\|^2$.

Therefore, the function $\Phi_{k+1}(t)$ decreases in the case $\Phi'_{k+1}(t) < 0$ which is true when $t \in (0, 1)$.

Finally, since

$$\Phi'_{k+1}(t) = 0 \Leftrightarrow \bar{t}_{k+1} = 1, \tag{20}$$

the minimum of $\Phi_{k+1}(t)$ is achieved for $t = 1$.
Accordingly, the step size t_{k+1} was determined using the backtracking line search procedure under the initial value $t = 1$.
Once the parameters $\gamma_{k+1} > 0$ and t_{k+1} are found, the next iterative point can be calculated as

$$\mathbf{x}_{k+2} = \mathbf{x}_{k+1} - t_{k+1}\gamma_{k+1}^{-1}\mathbf{g}_{k+1}.$$

Modified SM method

We propose two modifications of the *IGD* methods

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \gamma_k^{-1} t_k \mathbf{g}_k.$$

The first modification produces the class of iterations is of the general form

$$\mathbf{x}_{k+1} = \mathcal{M}(IGD)(\mathbf{x}_k) = \mathbf{x}_k - \gamma_k^{-1} \left(t_k + t_k^2 - t_k^3\right) \mathbf{g}_k, \tag{21}$$

which leads to the **modified IGD** (MIGD) class of methods.
The main idea used in defining the iterations (21) is the replacement of the basic step size t_k in *IGD* methods by the new basic stepsize $t_k + t_k^2 - t_k^3$.
It is assumed that t_k is defined by the backtracking procedure, which implies $t_k \in (0, 1)$. As a consequence, the justification for this modification lies in the inequalities

$$t_k \leq t_k + t_k^2 - t_k^3 \leq t_k + t_k^2.$$

As a conclusion, (21) is based on a relatively smaller increase in the step length t_k by $t_k + t_k^2 - t_k^3 \in [t_k, t_k + t_k^2]$.

Modified SM method

Since

$$\gamma_k^{-1}\left(t_k + t_k^2 - t_k^3\right) \geq \gamma_k^{-1} t_k,$$

it follows that the MIGD iterations (21) define an appropriate increase of the step length in the IGD class.

Obtained results are published in
[243] [B. Ivanov, P.S. Stanimirović, G.V. Milovanović, S. Djordjević, I. Brajević, Accelerated multiple step-size methods for solving unconstrained optimization problems, Optimization Methods and Software 36(5) (2021), 998-1029. doi: 10.1080/10556788.2019.1653868.]

Hybrid multiple step size methods

In the second modification, we will exploit the Picard-Mann-Ishikawa hybrid iterative process which was defined by the next three relations based on the mapping T on a normed space

$$\begin{cases} \mathbf{x}_1 = \mathbf{x} \in \mathbb{R}, \\ \mathbf{x}_{k+1} = T\mathbf{y}_k, \\ \mathbf{y}_k = (1-\alpha_k)\mathbf{x}_k + \alpha_k T\mathbf{x}_k, \quad k \in \mathbb{N}. \end{cases} \tag{22}$$

The real number $\alpha_k \in (0,1)$ is denoted as the correction parameter.
Innovative class of iterations is defined by the hybrid correction of the *IGD* iterations, which is defined by

$$\mathbf{x}_{k+1} = \mathcal{H}(IGD)(\mathbf{x}_k) = \mathbf{x}_k - (\alpha_k + 1)\gamma_k^{-1} t_k \mathbf{g}_k, \tag{23}$$

where $\alpha_k \geq 0$ is reused from the Picard-Mann-Ishikawa hybrid iterative process.

Accelerated multiple step size methods

Moreover, we consider the acceleration (21) and hybridization (23) incorporated in the hybrid *MIGD* class (shortly *HMIGD* class)

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathcal{H}(MIGD)(\mathbf{x}_k) = \mathcal{H}(\mathcal{M}(IGD, \mathbf{x}_k), \mathbf{x}_k) \\ &= \mathbf{x}_k - (\alpha_k + 1)\gamma_k^{-1}\left(t_k + t_k^2 - t_k^3\right)\mathbf{g}_k. \end{aligned} \tag{24}$$

Since $t_k \in (0, 1)$ and $\alpha_k + 1 \geq 1$, it follows that

$$(\alpha_k + 1)\gamma_k^{-1}\left(t_k + t_k^2 - t_k^3\right) \geq (\alpha_k + 1)\gamma_k^{-1} t_k \geq \gamma_k^{-1} t_k,$$

which means that the *HIGD* class (23) defines another increase of the step length in the *IGD* class and *HMIGD* class (24) is an acceleration of *HIGD* iterations (23).

IGD with fuzzy Parameters

Application of **Neutrosophication Logic System** (NLS) and **Fuzzy Logic System** (FLS) aims to solve the indeterminacy and selectivity of the step size parameter.

An FLS utilizes a *membership function* (MF) $\mathrm{T}(\lambda) \in [0,1], \lambda \in \Lambda$ in the universe Λ, which defines the degree of membership of λ.

An intuitionistic fuzzy set (IFS) is based on *membership function* and *non-membership functions* $T(\lambda), F(\lambda) \in [0,1], \lambda \in \Lambda$, which satisfy $\mathrm{T}(\lambda), \mathrm{F}(\lambda) : \Lambda \to [0,1]$ and are jointly correlated by inequalities $0 \leq \mathrm{T}(\lambda) + \mathrm{F}(\lambda) \leq 1$.The IFS theory was extended by

the neutrosophic theory. An element $\lambda \in \Lambda$ in a neutrosophic set is characterized by three individualistic MFs:
the truth-MF $\mathrm{T}(\lambda)$, the indeterminacy-MF $\mathrm{I}(\lambda)$, and the falsity-MF $\mathrm{F}(\lambda)$.

Due to the independence between the three MFs, the neutrosophic logic is established on the symmetry involved in the ordered triple $(\mathrm{T}, \mathrm{I}, \mathrm{F})$ and the inequality $0 \leq \mathrm{T} + \mathrm{I} + \mathrm{F} \leq 3$.

IGD with fuzzy Parameters

Class of **fuzzy descent direction** (FDD) iterations are defined in the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \nu_k t_k \mathbf{d}_k, \tag{25}$$

where ν_k is appropriately defined fuzzy parameter and the step size t_k is computed using an inexact line search.

The main idea is to improve the line search iterations $\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k$ using an additional fuzzy parameter ν_k.

Results are restated from
[333] P.S. Stanimirović, B. Ivanov, D. Stanujkić, V.N. Katsikis, S.D. Mourtas, L.A. Kazakovtsev, S.A. Edalatpanah, Improvement of unconstrained optimization methods based on symmetry involved in neutrosophy, Symmetry, 2023, 15,250, doi: 10.3390/sym15010250.

IGD with fuzzy Parameters

- The general fuzzy QN (FQN) iterative scheme with the line search is defined as

$$\mathbf{x}_{k+1} = \Phi(QN)(\mathbf{x}_k)\Phi(\mathbf{x}_k - H_k\,\mathbf{g}_k) = \mathbf{x}_k - \nu_k H_k\,g_k, \tag{26}$$

- Fuzzy GD method (FGD) is defined by

$$\mathbf{x}_{k+1} = \Phi(GD)(\mathbf{x}_k) = \Phi(\mathbf{x}_k - t_k\,\mathbf{g}_k) = \mathbf{x}_k - \nu_k t_k \mathbf{g}_k. \tag{27}$$

- Fuzzy SM method (FSM) is defined as

$$\mathbf{x}_{k+1} = \Phi(SM)(\mathbf{x}_k) = \mathbf{x}_k - \nu_k t_k (\gamma_k^{FSM})^{-1}\mathbf{g}_k, \tag{28}$$

where

$$\gamma_{k+1}^{FSM} = \upsilon\left(2\gamma_k^{FSM}\,\frac{\gamma_k^{FSM}\Delta_k + \nu_k t_k \|\mathbf{g}_k\|^2}{(\nu_k t_k)^2\,\|\mathbf{g}_k\|^2}\right). \tag{29}$$

- Fuzzy MSM method (FMSM) is defined by

$$x_{k+1} = \Phi(MSM)(x_k) = x_k - \nu_k \tau_k (\gamma_k^{FMSM})^{-1} g_k, \tag{30}$$

where

$$\gamma_{k+1}^{FMSM} = \upsilon\left(2\gamma_k^{FMSM}\,\frac{\gamma_k^{FMSM}\Delta_k + \nu_k \tau_k \|g_k\|^2}{(\nu_k \tau_k)^2\,\|g_k\|^2}\right). \tag{31}$$

IGD with fuzzy Parameters

The value ν_k is obtained using properly defined NLS.
In general, the parameter ν_k should satisfy

$$\nu_k \begin{cases} < 1, & \text{if } f(x_{k+1}) > f(x_k), \\ = 1, & \text{if } f(x_{k+1}) = f(x_k), \\ > 1, & \text{if } f(x_{k+1}) < f(x_k). \end{cases} \tag{32}$$

Neutrosophication. NLS maps the input $\vartheta := f(x_k) - f(x_{k+1}) \in \mathbb{R}$ into $\langle \vartheta : T(\vartheta), I(\vartheta), F(\vartheta) \rangle$.

Algorithm 4 Framework of FDD methods.

Require: Objective $f(\mathbf{x})$ and an initial point $\mathbf{x}_0 \in \mathrm{dom}(f)$.

1: Put $k = 0$, $\nu_0 = 1$, calculate $f(\mathbf{x}_0)$, $g_0 = \nabla f(\mathbf{x}_0)$, and generate a descent direction $\mathbf{d}_0$.
2: If stopping indicators are fulfilled, then stop; otherwise, go to the subsequent step.
3: (Backtracking) Determine $t_k \in (0, 1]$ applying Algorithm 2.
4: Compute $\mathbf{x}_{k+1}$ using (25): $\mathbf{x}_{k+1} = \mathbf{x}_k + \nu_k t_k \mathbf{d}_k$.
5: Compute $f(\mathbf{x}_{k+1})$ and generate descent vector $\mathbf{d}_{k+1}$.
6: (Score function) Compute $\Delta_k := f_{k+1} - f_k$.
7: (Neutrosophistication) Compute $\mathrm{T}(\Delta_k), \mathrm{I}(\Delta_k), \mathrm{F}(\Delta_k)$ using appropriate membership functions.
8: Define neutrosophic inference engine.
9: (De-neutrosophistication) Compute ν_k using de-neutrosophication rule.
10: $k := k + 1$ and go to step 2.
11: Output: $\{\mathbf{x}_{k+1}, f(\mathbf{x}_{k+1})\}$.

IGD with fuzzy Parameters

The truth-membership function is defined as the sigmoid membership function:

$$\mathrm{T}(\vartheta) = 1/(1 + \mathrm{e}^{-c_1(\vartheta - c_2)}). \tag{33}$$

The parameter c_1 is responsible for its slope at the crossover point $\vartheta = c_2$.

The falsity-membership function is the sigmoid membership function:

$$\mathrm{F}(\vartheta) = 1/(1 + \mathrm{e}^{c_1(\vartheta - c_2)}). \tag{34}$$

The indeterminacy-membership function is the Gaussian membership function:

$$\mathrm{I}(\vartheta) = \mathrm{e}^{-\frac{(\vartheta - c_2)^2}{2c_1^2}}. \tag{35}$$

De-neutrosophication. This step assumes conversion $\langle \vartheta_k : T(\vartheta_k), I(\vartheta_k), F(\vartheta_k) \rangle \to \nu_k(\vartheta_k) \in \mathbb{R}$ resulting into a single (crisp) value $\nu_k(\vartheta_k)$. The following *de-neutrosophication rule* is proposed to obtain the parameter $\nu_k(\vartheta_k)$:

$$\nu_k(\vartheta_k) = \begin{cases} 3 - \left(T(\vartheta_k) + I(\vartheta_k) + F(\vartheta_k)\right), & f(x_{k+1}) < f(x_k) \\ 1, & f(x_{k+1}) = f(x_k) \\ 1 - \left(T(\vartheta_k) + I(\vartheta_k) + F(\vartheta_k)\right)/c_1, & f(x_{k+1}) > f(x_k). \end{cases}$$

where $c_1 \geq 3$ is defined based on the rule $\nu_k < 1$ if $f(x_{k+1}) \geq f(x_k)$.

IGD with fuzzy Parameters

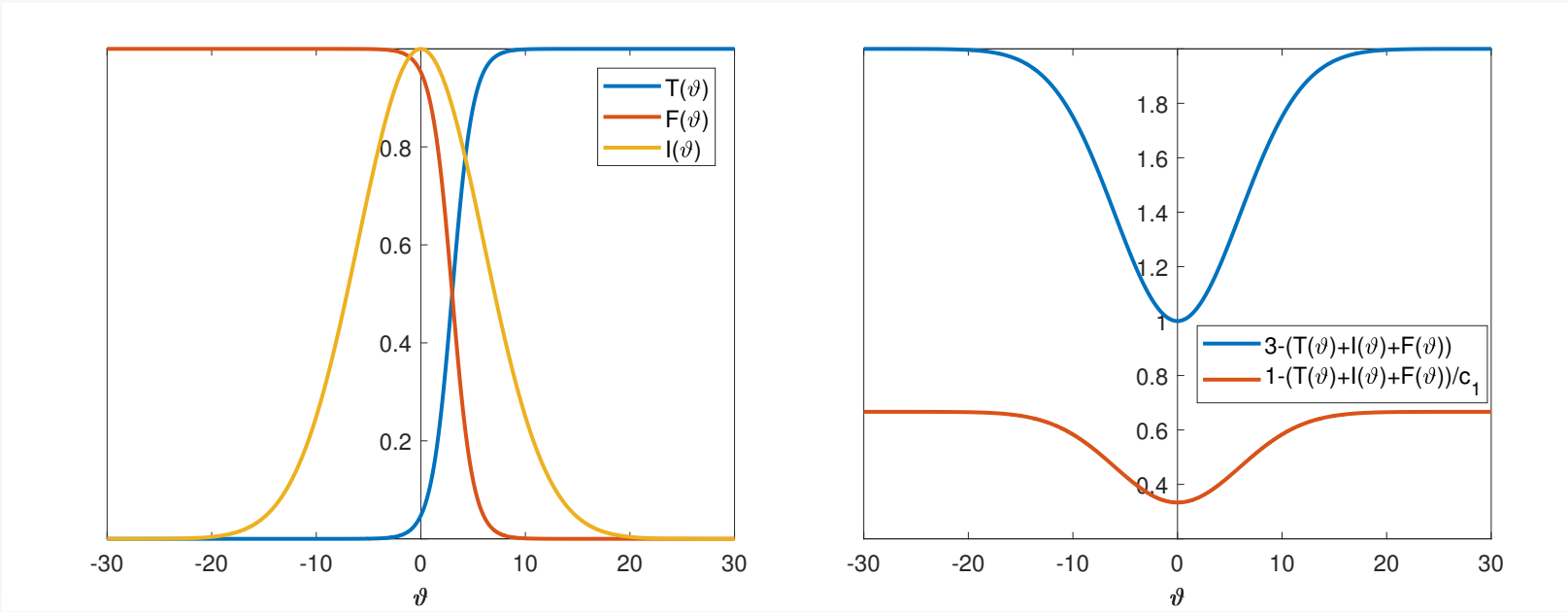
The settings in the NLC employed in numerical testing are arranged in Table 1.

Table 1: Recommended parameters in NLC.

Membership Function	c_1	c_2
Truth Sigmoid function (33)	1	3
Falsity Sigmoid function (34)	1	3
Indeterminacy Gaussian function (35)	6	0
Output function (30)	3	-

IGD with fuzzy Parameters

The membership functions values of $\mathrm{T}(\Delta_k), \mathrm{F}(\Delta_k), \mathrm{I}(\Delta_k)$ during the neutrosophication process, and presented in Figure 5(a).
The NLC output value, $\nu_k(\Delta_k)$, during the de-neutrosophication process is presented in Fig. 5(b).



(a) Neutrosophication.

(b) De-neutrosophication.

Figure 5: Neutrosophication (33), (34), (35), and de-neutrosophication (30) with parameters in Table 1.

Conjugate gradient methods

Nonlinear conjugate gradient (CG) methods form a class of methods for solving unconstrained nonlinear optimization and solving system of nonlinear equations.
CG class is defined by the line search iterates $\mathbf{x}_{k+1} := \mathbf{x}_k + t_k \mathbf{d}_k$, where the search direction $\mathbf{d}_k$ uses a conjugate direction instead of the steepest descent direction at each step. More precisely,

$$\mathbf{d}_k := \mathfrak{d}(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}) = \begin{cases} -\mathbf{g}_0, & k=0, \\ -\mathbf{g}_k + \beta_k \mathbf{d}_{k-1}, & k \geq 1, \end{cases} \tag{36}$$

where β_k is the real value which is known as the conjugate gradient update parameter (CGUP).
There exist a number of formulas for β_k with different computational properties.

Conjugate gradient methods

Table 2 surveys several classical CG methods, where $\mathbf{y}_{k-1}=\mathbf{g}_k-\mathbf{g}_{k-1}$, $\mathbf{s}_{k-1}=\mathbf{x}_k-\mathbf{x}_{k-1}$, $\mathbf{G}_{k-1}=\Delta^2 f(\mathbf{x}_{k-1})$ and $\|\cdot\|$ stands for the Euclidean vector norm.

Table 2: Some basic CGUP parameters.

β_k	Title	Year
$\beta_k^{\mathrm{HS}}=\dfrac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Hestenses–Stiefel	1952
$\beta_k^{\mathrm{FR}}=\dfrac{\|\mathbf{g}_k\|^2}{\|\mathbf{g}_{k-1}\|^2}$	Fletcher–Reeves	1964
$\beta_k^{\mathrm{D}}=\dfrac{\mathbf{g}_k^{\mathrm{T}}\mathbf{G}_{k-1}\mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{G}_{k-1}\mathbf{d}_{k-1}}$		1967
$\beta_k^{\mathrm{PRP}}=\dfrac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\|\mathbf{g}_{k-1}\|^2}$	Polak–Ribiere–Polyak	1969
$\beta_k^{\mathrm{CD}}=-\dfrac{\|\mathbf{g}_k\|^2}{\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Conjugate Descent	1987
$\beta_k^{\mathrm{LS}}=-\dfrac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Liu–Storey	1991
$\beta_k^{\mathrm{DY}}=\dfrac{\|\mathbf{g}_k\|^2}{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Dai–Yuan	1999

Conjugate gradient methods

Dai and Liao in 2001 suggested the conjugate gradient (CG) iterations

$$\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k, \tag{37}$$

in which t_k is a positive step size parameter defined as the output of a proper inexact line search, and $\mathbf{d}_k$ is a descent direction generated by the recurrent rule

$$\mathbf{d}_k = \begin{cases} -\mathbf{g}_0, & k = 0, \\ -\mathbf{g}_k + \beta_k^{\mathrm{DL}} \mathbf{d}_{k-1}, & k \geq 1, \end{cases} \tag{38}$$

where β_k^{DL} is the CG coefficient that describes the type of CG method according to the general rule

$$\beta_k^{\mathrm{DL}} = \frac{\mathbf{g}_k^{\mathrm{T}} \mathbf{y}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}} \mathbf{y}_{k-1}} - \mathrm{t} \frac{\mathbf{g}_k^{\mathrm{T}} \mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}} \mathbf{y}_{k-1}}, \tag{39}$$

wherein $\mathrm{t} > 0$ is an appropriate scalar.

Fuzzy Dai-Liao method

[352] P.S. Stanimirović, B. Ivanov, D. Stanujkić, L.A. Kazakovtsev, V,N, Krutikov, D. Karabašević, Fuzzy adaptive parameter in the Dai–Liao optimization method based on neutrosophy, Symmetry, 2023, 15, 1217. DOI: 10.3390/sym15061217.

The fuzzy neutrosophic Dai–Liao CG method is established as a modification of the Dai–Liao CG method (37): $\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k$, where the search direction $\{\mathbf{d}_k\}$ is calculated by the standard definition (38):

$$\mathbf{d}_k = \begin{cases} -\mathbf{g}_0, & k = 0, \\ -\mathbf{g}_k + \beta_k^{\mathrm{FDL}} \mathbf{d}_{k-1}, & k \geq 1, \end{cases}$$

and the CG coefficient β_k^{FDL} is defined by

$$\beta_k^{\mathrm{FDL}} = \frac{\mathbf{g}_k^{\mathrm{T}} \mathbf{y}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}} \mathbf{y}_{k-1}} - \nu_k \frac{\mathbf{g}_k^{\mathrm{T}} \mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}} \mathbf{y}_{k-1}}, \tag{40}$$

where ν_k is a proper fuzzy neutrosophic parameter.

Our intention is to define $\nu_k := \nu_k(\Delta_k)$ as a function of $\Delta_k := f(\mathbf{x}_k) - f(\mathbf{x}_{k+1})$. In addition, $\nu_k(\Delta_k)$ is defined subject to the constraints

$$0 \leq \nu_k(\Delta_k) \leq 1. \tag{41}$$

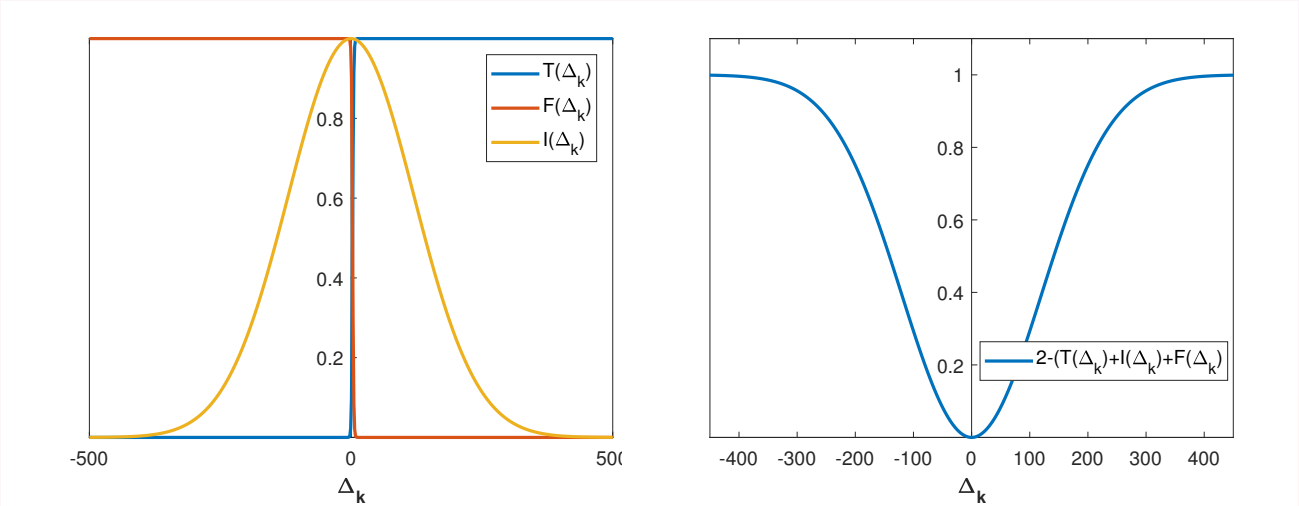
Fuzzy Dai-Liao method

(1) *Neutrosophication* maps the input $\Delta_k := f(\mathbf{x}_k) - f(\mathbf{x}_{k+1})$ into neutrosophic ordered triplets $(\mathrm{T}(\Delta_k), \mathrm{I}(\Delta_k), \mathrm{F}(\Delta_k))$, as in the FDD methods.
(2) *De-neutrosophication* is the transformation $\langle \mathrm{T}(\Delta_k), \mathrm{I}(\Delta_k), \mathrm{F}(\Delta_k) \rangle \to \nu_k \in \mathbb{R}$, resulting in a crisp value ν_k, proposed as

$$\nu_k = 2 - (\mathrm{T}(\Delta_k) + \mathrm{I}(\Delta_k) + \mathrm{F}(\Delta_k)) \,. \tag{42}$$

Table 3: Recommended parameters in NLC.

Membership Function	c_1	c_2
Truth Sigmoid function (33)	1	3
Falsity Sigmoid function (34)	1	3
Indeterminacy Gaussian function (35)	120	0
Output function (42)	-	-



(a) MFs in Neutrosophication. (b) Graph of De-neutrosophication.