Scuola Dottorale di Ingegneria
Sezione di Ingegneria dell'Elettronica Biomedica,
dell'Elettromagnetismo e delle Telecomunicazioni

XXV CICLO DEL CORSO DI DOTTORATO

Innovative Meta-heuristics for Solving Non Linear Inverse Problems in Industrial and Information Engineering

(Metaeuristiche innovative per la risoluzione di problemi inversi non lineari nell'ambito dell'ingegneria industriale e dell'informazione.)

Titolo della tesi

Dottorando: Giuseppe Pulcini ______________________ (firma)

Docente Guida: Prof. Alessandro Salvini ______________________ (firma)

Coordinatore: Prof. Lucio Vegni ______________________ (firma)

Innovative Meta-heuristics for Solving Non Linear Inverse Problems in Industrial and Information Engineering

Giuseppe Pulcini

A DISSERTATION SUBMITTED
THE COMMITTEE OF DOCTORAL SCHOOL OF ENGINEERING
OF THE UNIVERSITY OF ROMA TRE
SECTION OF BIOMEDICAL ELECTRONICS, ELECTROMAGNETISM AND TELECOMMUNICATION
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

April 2013

This page intentionally left blank

Abstract

In the world of technology, many industrial operations such as design of efficient devices, or planning production in a big factory, require optimization approach and the solution of inverse problems[chapter 1].

In this contest, in the last 20 years, the heuristic methods had a primary role considering their capabilities to find out solutions in all those cases in which a lot of computation time is requested. The present thesis work is mainly based on swarm algorithms, and on their capabilities to achieve global optima without remain trapped into local minima. In particular, in this work we treat high hybridization and integration among the different capabilities in exploitation and exploration, expressed by 3 optimization algorithms which are: Particle Swarm Optimization (PSO), Flock of Starlings Optimization (FSO), Bacterial Chemotaxis Optimization (BCA).

The research of high hybridization among different heuristics led to implement a new metaheuristic which has been called MeTEO (Metric Topological Evolutionary Optimization). MeTEO exploits the synergy among the three algorithms already mentioned above. Moreover, in MeTEO a further method called Fitness Modification (FM) has been used. As will be shown, the FM enhance the exploration properties of MeTEO together with benefits in the parallelization.

The first problem encountered making a metaheuristics composed of three complex algorithms is the computation time required. For this reason, the thesis work has been focused also in the analysis and synthesis of a parallel structures for supporting calculus. In this context, two different approaches have been studied: 1)the algorithm-based and 2) the fitness-based. Moreover, in order to extend the exploration capability of FSO problems with discrete variable as well, a binary version of FSO has been implemented [chapter 2].

MeTEO has been validated on benchmarks and on inverse problems. The benchmarks used are called hard benchmarks, because they show a structure without preferential tendency towards a particular point, and local minima with depth value, some monomodal, with one global minimum, and multimodal, with many equivalent minima. Afterwards a list of real inverse and optimization problems are proposed: the parameters identifications of Jiles-Atherton parameters, the efficiency improvement of Travelling Wave Tube (TWT) device, taking in account the geometry, the magnetic focusing field, and the voltage of a Multistage Compressed Collector, the harmonic detection in distorted waveforms. The simulation has been made with MATLAB, and with the help of a FEM simulator called COLLGUN. All results have been compared with those from other algorithms such as random walk, and the also from the use of a single heuristics which MeTEO exploits [chapter 3].

In the Chapter 4 of this thesis the point of view changes toward the hardware, whereas all the discussion done in the previous three chapters were focused on the improvement of the optimization process preformed by numerical algorithms (Software). In fact, we present a method for translating a numerical swarm based algorithm into an electric circuit, that is able to reproduce by mean of voltages and currents the same trajectories shown by the numerical swarm-based algorithms. A circuit, called *swarm circuit*, has been implemented with Simulink

after to have deduced the mathematical relations between the numerical algorithms and their translation into a dynamic system. The swarm circuit has been tested with hard benchmarks and with two inverse problems. The swarm circuit open the road towards a real time optimization, argument that is difficult to be addressed with software approaches.

Acknowledgements

A special thank to my Advisor Prof. A. Salvini and at his research group of optimization and soft computing, for their useful suggestions.

Thank to all my students that I helped to make their thesis. I would like to remember all of them because in front of a coffee machine, in lunch room and in laboratory, we have spent a lot of time getting one to each other not only a technical contribution, but also a human experience, that too many times is put away in work contest but that is essential for achieving the best performance from each of us.

A special thank to Angela because she guessed that my personal constant time response was less than the Ph.D course one, with all thinks that it involves.

Thank to my mother and my father because in a lot of critical moments they have shown a remarkable wisdom attitude.

A thank to dott. Valerio Ielapi and dott. Rosalva Sciammetta for their correction and feedback.

I would remember that the work made, (whatever it is), is always the contribution of all people that a person has met in his life, so this acknowledgments go to everyone that in my life in a way or another has contributed to shape my mind so as to made this work exactly as you can see it, in a uniquely way.

Contents

Italian Digest

Introduzione

Nell'ambito dei problemi di ottimizzazione, e più in generale nella risoluzione di problemi inversi, è forte l'esigenza di possedere algoritmi di ottimizzazione che siano capaci di produrre una esplorazione dello spazio delle soluzioni il più possibile esaustiva. Infatti nelle funzioni obiettivo di problemi reali si possono incontrare moltissimi minimi locali, nei quali solitamente gli algoritmi con scarsa capacità di indagine rimangono intrappolati, vanificando così l'intero processo di ottimizzazione.

D'altro canto però, esiste l'esigenza di poter raffinare la soluzione nel momento in cui ci si trova davanti a un candidato minimo globale. Quindi un algoritmo di ottimizzazione valido e performante dovrebbe possedere alte capacità di esplorazione e allo stesso tempo un'ottima capacità di convergenza.

Ovviamente è impensabile arrivare a ottenere simili performance solo con un singolo algoritmo, la via da intraprendere è quella della ibridizzazione. Pensando a queste esigenze è stato implementato un nuovo algoritmo meta-euristico chiamato MeTEO (Metric Topological Evolutionary Optimization).

Gli algoritmi utilizzati in MeTEO sono algoritmi metaeuristici, ovvero sono algoritmi ispirati al comportamento dei viventi, in particolare MetEO è composto da: Flock of Starlings Optimization (FSO), Particle Swarm Optimization (PSO) e Bacterial Chemotaxis Algorithm (BCA). I primi due possiedono un comportamento collettivo, ovvero sono in grado di condividere l'informazione riguardante l'analisi della funzione di costo tra i vari individui facenti parte dello stesso stormo. Il BCA invece non possiede un comportamento collettivo.

Con questo tipo di approccio si tenta di ricreare un ambiente naturale reale, in cui sono presenti stormi di uccelli (FSO), sciami di insetti (PSO) e popolazioni di batteri (BCA).

Nel prossimo paragrafo verrà descritto in modo dettagliato il funzionamento di MeTEO e dei suoi singoli componenti. Un altro importante metodo che completa il quadro descrittivo di questo algoritmo è la "Fitness Modification" (FM). La fitness modification consiste nel sommare una funzione gaussiana nei punti in cui l'FSO ha trovato dei minimi locali. Così facendo l'FSO avvertirà un repentino aumento della Fitness e verrà dirottato in una zona diversa da quella in cui è presente il minimo locale.

MeTEO è stato progettato per essere utilizzato su una architettura parallela. In particolare nel lavoro svolto in questa tesi, il parallelismo è stato implementato su un cluster di 20 computer, connessi attraverso una rete LAN. Gli approcci implementati sono stati due: algorithm-based e fitness-based. Nell'approccio algorithm-based è presente un nodo master sul quale l'FSO gira con il compito di esplorare il dominio delle soluzioni in modo tale da individuare delle possibili regioni in cui può giacere un minimo globale. Quando l'FSO ha individuato tale regione chiamata "suspected region", viene lanciata su un nodo slave del cluster la serie PSO+BCA. Gli individui del PSO vengono inizializzati nel punto che viene fornito dal master e quindi dall'FSO. Parimenti il PSO alla fine della sua ispezione fornirà il minimo trovato nel suo sotto-dominio di competenza al BCA che avrà il compito di rifinire la soluzione. Nell'approccio fitness-based invece, il cluster viene utilizzato per servire un algoritmo per volta in serie. In questo approccio il master distribuisce ai nodi slave il task di calcolare la fitness per ogni individuo che forma il gruppo. Sarà compito del master, una volta raccolti tutti i dati delle fitness dai nodi slave, aggiornare tutti gli altri parametri e calcolare le nuove posizioni degli individui. L'approccio fitness-based viene utilizzato e consigliato quando si ha a che fare con funzioni di costo che impiegano un tempo di esecuzione che è comparabile con l'esecuzione del codice di una iterazione di MeTEO. Nel lavoro di tesi sono stati implementati anche altri due algoritmi, in particolare il MultiFlock (MFSO), il Binary Flock of Starlings Optimization (BFSO), e una ibdridizzazione tra FSO e PSO indicata con l'acronimo FPSO. Nell'MFSO il gruppo viene diviso in sotto gruppi, tutti gli individui di uno stesso sotto-gruppo possiedono un comportamento pari a quello degli individui di un FSO. I vari sotto gruppi condividono però lo stesso global best. Tale algoritmo è risultato essere ancora più performante sotto il punto di vista dell'esplorazione dei singoli algoritmi con i quali è stato confrontato. Successivamente si è esteso l'FSO anche a problemi di ottimizzazione aventi variabili discrete implementando il BFSO, nel quale ogni individuo viene codificato attraverso una stringa binaria. La regola di aggiornamento della velocità del singolo individuo adesso rappresenta la probabilità che una determinata cifra nella stringa binaria ha di cambiare il proprio stato.

Viene inoltre fornita anche una ibridizzazione tra FSO e PSO in cui per cicli alterni vengono fatti convivere nello stesso spazio delle soluzioni il e l'FSO. In particolare, il processo di ricerca viene iniziato dall'FSO, ogni volta che viene individuato un minimo locale il gruppo di si divide in due parti: un primo sottogruppo continua a indagare lo spazio delle soluzioni a funzionando quindi come l'FSO, invece il secondo gruppo viene momentaneamente convertito in PSO e ha il compito di rifinire la soluzione. Dopo alcuni cicli in cui viene applicato l'interscambio appena descritto, il miglior punto individuato viene indagato con il solo PSO. MeTEO è stato validato su vari benchmark e su alcuni problemi inversi.

Richiamo sull'algoritmo MeTEO

METEO è un algoritmo basato su tre metaeuristiche diverse: FSO, PSO e BCA. L'FSO e il PSO appartengono alla classe di algoritmi appartenenti alla swarm intelligence. Dedichiamo i seguenti paragrafi a richiamare il tratti fondamentali di ogni algoritmo.

Flock of Starling Optimization e Particle Swarm Optimization.

Il Particle Swarm Optimization (PSO) è uno degli algoritmi più studiati ed utilizzati tra gli algoritmi di ottimizzazione. E' stato ideato da James Kennedy e Russell Eberhart nel 1995 [7]. Il PSO sulla base di una regola metrica è in grado di riprodurre il comportamento di uno sciame di insetti.

L'introduzione di una regola topologica nel PSO è il fulcro della algoritmo chiamato Flock of Starlings Optimization (FSO). L'FSO, [3], [4] adotta un approccio basato su recenti osservazioni naturalistiche [5], sul comportamento collettivo degli storni europei (Sturnus vulgaris). Gli autori del lavoro [5] hanno scoperto una interazione tra membri della stesso gruppo che possiede natura topologica: la proprietà principale dell'interazione topologica è che ogni individuo interagisce con un numero fisso di vicini, quindi la loro distanza metrica risulta non essere cruciale. L'approccio topologico è in grado di descrivere le variazioni di densità che sono tipici di stormi di uccelli, cosa che la regola metrica non è in grado di fare. In stormi reali un individuo controlla la velocità di suoi 7 individui scelti in modo casuale all'interno del gruppo di appartenenza. In figura 2 vengono riportati gli pseudo-codici dell'implementazione dell'PSO e dell' FSO di una funzione generica, $f(x_1.....x_D)$, della quale deve essere trovato il minimo, e lo spazio delle soluzioni $\mathbb{R}^D$.

```
Define
1   $\mathbb{R}^D \equiv (x_1 \ldots x_D):\ x_k^{\min} \le x_k \le x_k^{\max}$ with $k = 1 \ldots D$
2   $p_j \equiv \left(x_1^j \ldots x_D^j\right)$, with $j = 1 \ldots n_{particles}$
3   $T_{\max}$
4   $f(x_1 \ldots x_D)$
5   $v^j{}_k(t=0) = random(0,1) \cdot V_{\max}$
6   $f_{p_j}(0) = \infty$
7   $g(0) = \infty$
8   $p_j \equiv \left(x_1^j(0) \ldots x_D^j(0)\right)$ of each j-th particle : $x_k^j(0) = random(0,1) \cdot \left(x_k^{\max} - x_k^{\min}\right) + x_k^{\min}$
9   $\omega_{\max}$
10  $\lambda_{\max}$
11  $\gamma_{\max}$
12  $goal\_fitness = arbitrary\ small$
13  $\delta_{\max}$                                          } Defined just for FSO
14  $N_{ctrl\_birds}$                                        }
15  $Mccb_k^j = \frac{1}{N_{crl\_birds}} \sum_{h=1}^{N_{ctrl\_birds}} v_k^{h,j}$   }
```

Figura 1: principali parametri dell'FSO e del PSO.

```
For each j-th particle, for each step t, with $t = 0 \ldots T_{\max}$
    16  $f_j(t) = f(x_1^j(t) \ldots x_D^j(t))$
    17  If $f_j(t)$ is better than the personal best fitness of the j-th particle $f_{p_j}(t)$
            $p\_best_k^j = x_k^j(t) \quad \forall k$
            $f_{p_j}(t) = f_j(t)$
    18  If $f_j(t)$ is better than global best fitness
            $g\_best_k = x_k^j(t) \quad \forall k$
            $g(t) = f_j(t)$
    19  $\omega^j = \omega_{\max}$ , $\lambda^j = \lambda_{\max} \cdot random(0,1)$ , $\gamma^j = \gamma_{\max} \cdot random(0,1)$
    20  $v_k^j(t+1) = \omega^j v_k^j(t) + \lambda^j(p\_best_k^j - x_k^j(t)) + \gamma^j(g\_best_k - x_k^j(t))$
        ( just for PSO )
    21  $v_k^j(t+1) = \omega^j v_k^j(t) + \lambda^j(p\_best_k^j - x_k^j(t)) + \gamma^j(g\_best_k - x_k^j(t)) + \delta^j \cdot Mccb_k^j$
        ( just for FSO )
    22  $x_k^j(t+1) = x_k^j(t) + v_k^j(t+1)$
End For
```

Figura 2: pseudo codice degli algoritmi FSO e PSO.

I parametri principali di ogni individuo *j-th* che forma l'algoritmo sono: la dimensione dello spazio delle soluzioni $\mathbb{R}^D \equiv (x_1.....x_D)$, il numero massimo di iterazioni $T_{\max}$, il valore massimo consentita ad ogni individuo $V_{\max}$. Inoltre le velocità vengono inizializzate in modo randomico nel dominio[0 1]. Altri parametri sono: la fitness iniziale di ogni singolo individuo $f_{p_j}(0) = \infty$, dove il simbolo di ∞ indica un valore arbitrariamente grande; il valore del global best $g(0) = \infty$, i valori delle coordinate dei personal best indicati con $p_j = \left(x_1^j(0)...x_D^j(0)\right)$, il valore del coefficiente di inerzia ω , il valore della coefficiente cognitivo $\lambda_{\max}$, e il valore del coefficiente sociale $\gamma_{\max}$; ed infine il valore il valore di soglia sotto il quale si può considerare La minimizzazione del funzionale effettuata pari $goal\_fitness = arbitrary\ small$, impostato dall'utente.

Inoltre nella figura 1 sono stati riportati alcuni parametri che vengono utilizzati solo nell'FSO, indicati dai numeri alla sinistra da #13 a #15: il coefficiente topologico $\delta_{\max}$ scelto nel dominio [0 1]; la quantità $Mccb_k^j = \frac{1}{N_{crl\_birds}} \sum_{h=1}^{N_{crl\_birds}} v_k^{h,j}$; il numero di individui che vengono seguiti da un individuo all'interno del gruppo.

Estensione dell'FSO al MultiFlock

Con lo scopo di migliorare ulteriormente le capacità esplorative dell'FSO, è stata ideata un'ulteriore variante chiamata MultiFlock of Starling Optimization (MFSO). Essa consiste nell'implementazione di più di una serie di strmi che lavorano in contemporanea sulla stessa funzione di costo. Nell'MFSO la matrice di interconnessione viene divisa in più sub matrici di dimensioni pari a $N_{ind\_sub} \times N_{birds\_followed}$, dove N_{ind_sub} indica il numero di individui che appartengo a ogni sub-flock. Ogni individuo che appartiene a un sub-flock sarà costretto a seguire individui del suo sub-flock. In questo modo ogni sub-flock avrà delle traiettorie non correlate con gli altri sub-flock; l'unico scambio di informazione che c'è tra i vari sub-flock è rappresentato dell global best.

L'algoritmo della Chemotassi batterica

Il BCA è stata proposto in [6] ed è un algoritmo basato sulla emulazione del moto di un vero batterio mentre questo è in cerca di cibo. Una descrizione matematica del movimento del batterio 2D può essere sviluppata assumendo una velocità *v* assegnata e dalla determinazione di opportune distribuzioni probabilistiche di durata del movimento τ, e la direzione indicata da ciascun individuo φ. La descrizione 2D può essere facilmente estesa per iperspazi *n-dimensionali* che definiscono, per il percorso del *batterio virtuale*, un vettore composto da *n* posizioni x_i, con $i = 1, \ldots, n$, e un vettore fatto da $n-1$ direzioni, con $k = 1, \ldots, n-1$. Riportiamo nelle figure 3 e 4 sono elencati i principali parametri dell'algoritmo e le regole proposte implementate nello pseudo codice.

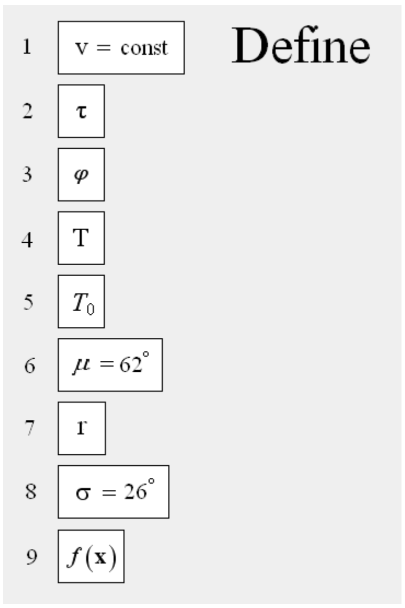
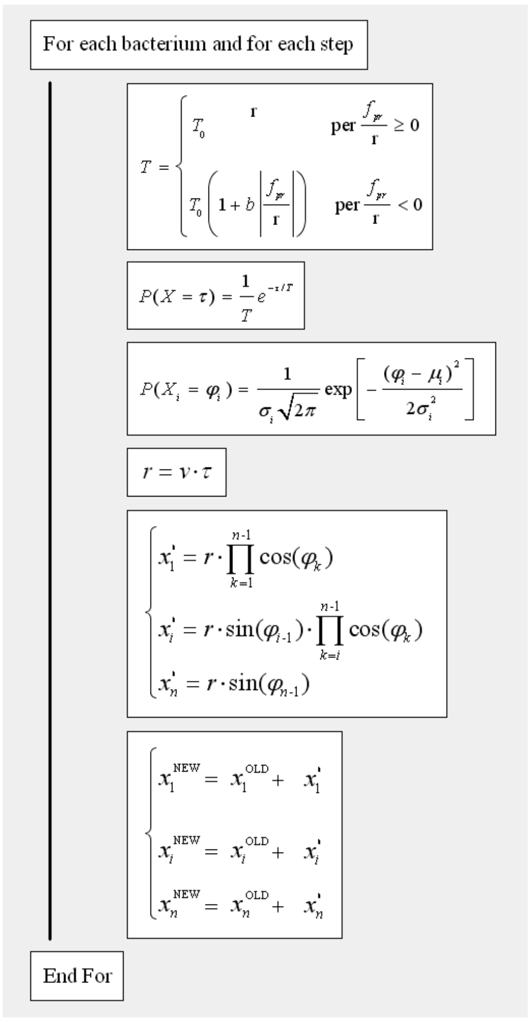


Figura 3: principali parametri presenti nell'algoritmo BCA.

Nella figura 3 sono stati indicati il valore estratto da una funzione di densità di probabilità esponenziale, pari a T, avente valore atteso e varianza pari a and μ, e σ, il tempo minimo medio di uno spostamento T_0, il modulo della differenza tra la nuova posizione del vettore, $\mathbf{x}^{NEW}$, e la vecchia posizione, $\mathbf{x}^{OLD}$, di un individuo $r = \left|\mathbf{x}^{NEW} - \mathbf{x}^{OLD}\right|$, e infine la funzione di costo da minimizzare $f(\mathbf{x})$. Riguardo alla direzione la funzione di densità di probabilità descrive l'angolo di inclinazione tra due traiettorie consecutive. La Figura 4 mostra lo pseudo code del BCA. Il BCA è fortemente influenzato dalla scelta dei parametri T_0 τ and b. Tali parametri vengono scelti in modo empirico.



```
For each bacterium and for each step
    $T = \begin{cases} T_0 \quad \mathrm{r} & \text{per } \frac{f_{pr}}{\mathrm{r}} \geq 0 \\ T_0\left(1 + b\left|\frac{f_{pr}}{\mathrm{r}}\right|\right) & \text{per } \frac{f_{pr}}{\mathrm{r}} < 0 \end{cases}$
    $P(X = \tau) = \frac{1}{T} e^{-\tau/T}$
    $P(X_i = \varphi_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left[ -\frac{(\varphi_i - \mu_i)^2}{2\sigma_i^2} \right]$
    $r = v \cdot \tau$
    $\begin{cases} x_1' = r \cdot \prod_{k=1}^{n-1} \cos(\varphi_k) \\ x_i' = r \cdot \sin(\varphi_{i-1}) \cdot \prod_{k=i}^{n-1} \cos(\varphi_k) \\ x_n' = r \cdot \sin(\varphi_{n-1}) \end{cases}$
    $\begin{cases} x_1^{\mathrm{NEW}} = x_1^{\mathrm{OLD}} + x_1' \\ x_i^{\mathrm{NEW}} = x_i^{\mathrm{OLD}} + x_i' \\ x_n^{\mathrm{NEW}} = x_n^{\mathrm{OLD}} + x_n' \end{cases}$
End For
```

Figua 4: pseudo codice dell'algoritmo della Chemotassi Batterica.

Questi tre algoritmi simulano un ambiente naturale completo, in cui ogni elemento svolge una particolare funzione. Infatti il ruolo dell'FSO è quello di esplorare lo spazio delle soluzioni trovando i punti candidati ad essere considerati minimo globale. Un punto è considerato un minimo locale se il suo valore rimane costante dopo un fissato numero d'iterazioni impostate dall'utente. Se questo avviene, l'FSO lancia una Fitness Modification (FM) che consiste nell'aggiungere una funzione gaussiana alla funzione di costo centrata nel global best attuale. Quindi, una FM opera secondo la seguente equazione:

$$f_{fitness_k}(\mathbf{x}) = f_{fitness_{k-1}}(\mathbf{x}) + A \exp\left(-\frac{\sum_{j=1}^{n} \left(x_j - x_{\min_j} \right)^2}{2\sigma^2} \right) \qquad (1)$$

dove A è una costante adatta, e σ è la deviazione standard che definisce la dimensione della *k-esima* FM lanciata dall'FSO. In questo modo, l'FSO rileva un alto valore di fitness, che lo induce ad allontanarsi da quel particolare punto, accelerando il processo di esplorazione. Ad ogni FM corrisponde il lancio su un nodo del cluster della serie PSO-BCA. Prima di terminare, l'FSO attende i risultati dai singoli processi paralleli, li raccoglie e li ritorna all'operatore. In dettaglio, quando viene lanciata una FM, in un nodo del cluster è avviata la serie PSO e BCA per esplorare nel dettaglio l'area identificata dall'FSO. Il cluster utilizzato è composto da 20 macchine. E' stato implementato un codice Matlab© che gestisce tutti gli scambi di file tra Master e Slave, nel nodo principale vi è una cartella condivisa in cui l'FSO scrive un file per ogni minimo locale, che contiene tutte le informazioni per lanciare e inizializzare la serie PSO-BCA. Ogni volta che il processo PSO-BCA ritorna i valori trovati, scrive i risultati sempre nella cartella condivisa. In seguito, l'FSO raccoglie i risultati, e gestisce la lettura e l'ordinamento dei file contenuti nella cartella condivisa. Nel nodo slave in cui viene lanciato il processo serie PSO-BCA, viene prima lanciato il PSO, che alla fine delle iterazioni ad esso assegnate fornisce la migliore posizione nel sub dominio di pertinenza, in seguito una colonia di batteri viene inizializzata nel punto fornito dal PSO in precedenza. Per il dettaglio della realizzazione del singolo algoritmo si fa riferimento a [1]. Ricordiamo che MeTEO racchiude in se un'alta proprietà di esplorazione conferita dall'utilizzo dell'FSO, una buona capacità di convergenza e media capacità di esplorazione data dal PSO, e un'ottima proprietà di convergenza dovuta all'utilizzo del BCA. Inoltre tramite la FM è capace di non rimanere "intrappolato" in qualche minimo locale, situazione che accade spesso quando si trattano problemi di ottimizzazione.

Binary Flock of Starlings Optimization

L'algoritmo di ottimizzazione FSO si basa su un costrutto che si presta alla risoluzione di problemi di ottimizzazione che abbiano delle variabili continue. Un risultato della ricerca triennale riassunto in questa tesi è stato quello di estendere l'algoritmo anche a problemi con variabili discrete. Per fare questo si è agito modificando la formula dell'aggiornamento della velocità del singolo individuo. Questa nuova versione dell'FSO è stata chiamata: Binary Flock of Starlings Optimization (BFSO).

Il BFSO è un'estensione del Discrete Particle Swarm Optimization. Le traiettorie nel nuovo modello diventano probabilistiche, e la velocità del singolo individuo rappresenta la probabilità che quella dimensione ha di cambiare stato da 0 a 1, o viceversa. Ovviamente il problema che si vuole trattare deve essere convertito in un problema a variabili binarie. Quindi le possibili posizioni per ogni dimensione possono essere 0 o 1; anche il personal best e il global best appartengono al dominio $\{0,1\}$. Essendo v_k^j una probabilità essa può avere valori compresi tra 0 e 1. Una trasformazione logistica viene introdotta utilizzando una funzione sigmoidale in modo tale da "costringere" il valore della velocità tra 0 e 1:

$$S(v_k^j) = \frac{1}{1+e^{-v_k^j}} \tag{2}$$

Successivamente, la variazione della posizione viene decisa in accordo con la:

$$if\left(rand < S\left(v_k^j\right)\right)\ then\ x_k^j\left(t\right)=1; else\ x_k^j\left(t\right)=0 \tag{3}$$

Partendo da queste assunzioni siamo in grado di definire il BFSO. L'equazione che regola la velocità del singolo individuo diventa:

$$v_k^j(t+1)=[\omega^j v_k^j(t)+\lambda^j(p\_best_k^j-x_k^j(t))+\gamma^j(g\_best_k-x_k^j(t))]\cdot Mccb_k^j \tag{4}$$

dove $Mccb_k^j$ è la probabilità media che la cifra cambia da 0 a 1.

Flock-Swarm Starling Optimization

L'FPSO è una nuovo algoritmo ibrido che incorpora i benefici dei già menzionati FSO e PSO. All'inizio del processo di ottimizzazione un algoritmo di ottimizzazione dovrebbe essere in grado di riuscire a trovare il valore del minimo globale nel minior tempo possibile. Quando il candidato ottimo viene individuato, lo stesso algoritmo dovrebbe essere in grado in questa seconda fase di rifinire la soluzione individuata. In molti casi queste due opposte capacità di esplorazione e convergenza, richiedono l'utilizzo di due algoritmi distinti lanciati in serie per essere espletate. Se nel funzionale invece sono presenti molti minimi locali, c'è la necessità di alternare esplorazione e convergenza. Nell'algoritmo proposto, l'FSO inizia il processo di esplorazione, se dopo un determinato numero di iterazioni N_{switch_off}, il valore del global best non cambia, il gruppo viene diviso due sottogruppi, modificano in modo opportuno la matrice di interconnessione $Mc_{[N\_birds,N\_followed]}$.

Identifichiamo con il simbolo F_{sub} il sub-swarm che possiede la proprietà esplorative dell'FSO, e con P_{sub} il sub-swarm che muta la propria caratteristica in quella di un PSO classico che possiede una migliore convergenza. Quindi il gruppo viene diviso in due parti ognuna delle quali svolge un differente task, una continua l'esplorazione, e l'altra rifinisce il minimo in esame.

La matrice delle interconnessioni viene modificata nella configurazione ibrida nel seguente modo:

$$Mc^{switch}_{[N\_birds,N\_followed]}=\left[\frac{Mc_{[P\cdot N\_bird,N\_followed]}}{O_{[N\_birds\_PSO,N\_followed]}}\right] \tag{5}$$

dove $O_{[N\_birds\_PSO,N\_followed]}$ è una matrice identicamente nulla.

L'algoritmo è formato quindi da tre configurazioni: 1) l'FSO esplora lo spazio delle soluzioni, 2) il gruppo viene diviso in due sottogruppi uno avente un comportamento come l'FSO e l'altro come PSO, che convivono contemporaneamente espletando comportamenti diverso, 3) infine tutto il gruppo viene mutato in PSO e viene rifinita la soluzione che è stata individuata come ottimo globale. Il comportamento 1) e 2) vengono alternato per un numero prestabilito di volte scelto dall'utente, infine si passa alla modalità 3). Le velocità massime consentite in ogni configurazione sono indicate in tabella 1.

Table 1: velocità massime consentite dai sottogruppi.

F_{sub}	P_{sub}
$V_{MAX\_FSO} = V_{MAX}$	(no)
$V_{MAX\_FSO} = V_{MAX}$	$V_{MAX\_PSO} = V_{MAX}/4$
(no)	$V_{MAX\_PSO} = V_{MAX}/10$

In figura 5 invece viene riportato il flow chart esaustivo del funzionamento dell'algoritmo FPSO.

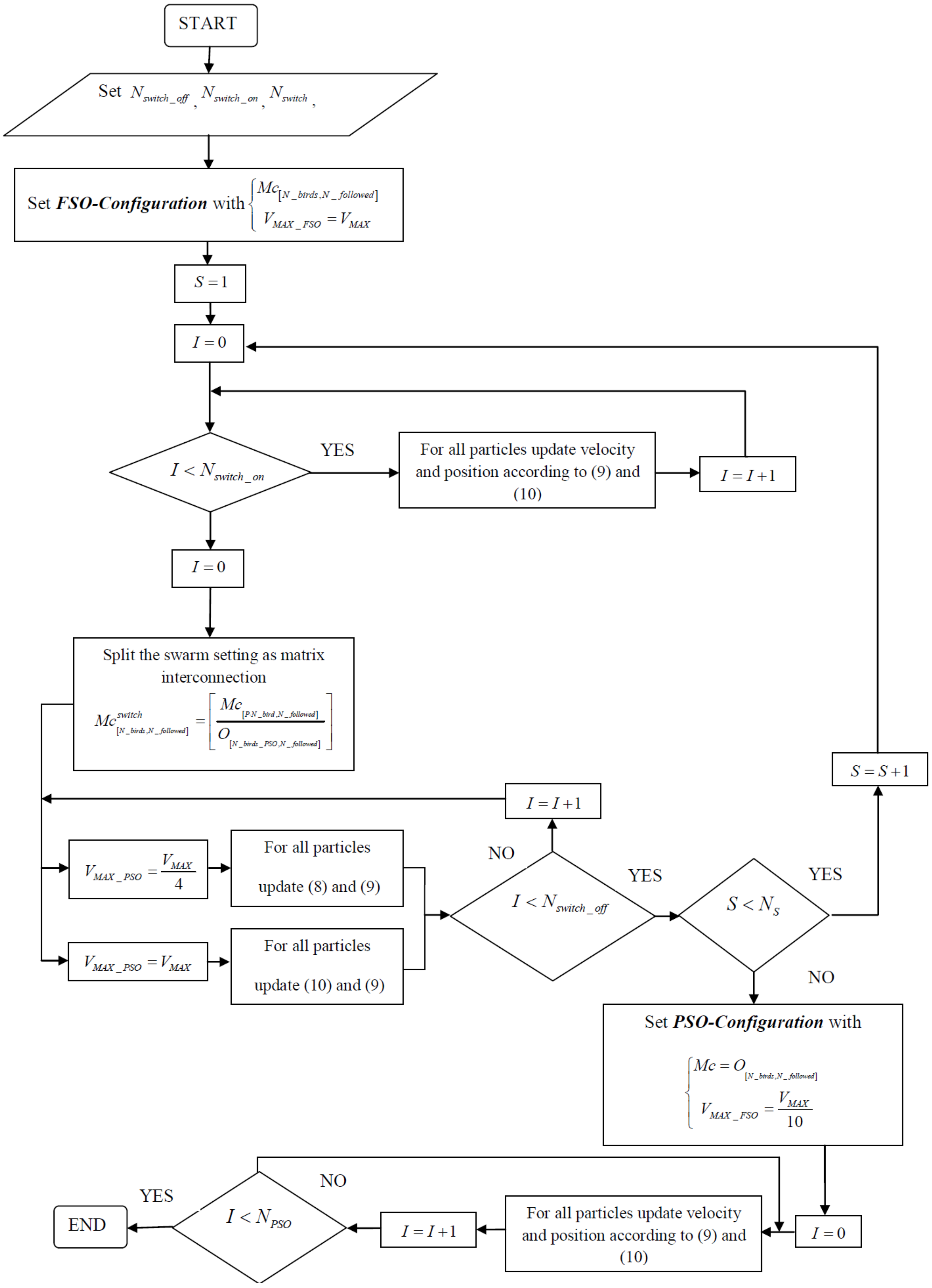

START
Set N_{switch_off}, N_{switch_on}, N_{switch},
Set ***FSO-Configuration*** with $\begin{cases} Mc_{[N\_birds, N\_followed]} \\ V_{MAX\_FSO} = V_{MAX} \end{cases}$
$S = 1$
$I = 0$
$I < N_{switch\_on}$
YES
For all particles update velocity and position according to (9) and (10)
$I = I + 1$
$I = 0$
Split the swarm setting as matrix interconnection
$Mc^{switch}_{[N\_birds, N\_followed]} = \left[\frac{Mc_{[P \cdot N\_bird, N\_followed]}}{O_{[N\_birds\_PSO, N\_followed]}} \right]$
$I = I + 1$
$V_{MAX\_PSO} = \frac{V_{MAX}}{4}$
For all particles update (8) and (9)
$V_{MAX\_PSO} = V_{MAX}$
For all particles update (10) and (9)
NO
$I < N_{switch\_off}$
YES
$S < N_S$
YES
$S = S + 1$
NO
Set ***PSO-Configuration*** with
$\begin{cases} Mc = O_{[N\_birds, N\_followed]} \\ V_{MAX\_FSO} = \frac{V_{MAX}}{10} \end{cases}$
$I = 0$
For all particles update velocity and position according to (9) and (10)
$I = I + 1$
NO
$I < N_{PSO}$
YES
END

Figura 5:flow chart dell'algoritmo FPSO.

Architetture Parallele: Algorithm-Based e Fitness-Based.

MeTEO trova una sua naturale applicazione all'interno di un sistema di calcolo parallelo. Infatti, come descritto nel funzionamento la struttura è formata da un nodo master e dei nodi slave appartenenti allo stesso cluster di computer. Nel lavoro di tesi sono state implementate due diverse architetture. La prima è stata chiamata algortihm-based, in essa all'interno del master è presente l'FSO che esplora lo spazio delle soluzioni, e ogniqualvolta incontra una "regione sospetta" in cui è presente un minimo locale, lancia su un nodo del cluster la serie PSO-BCA. In questa configurazione il parallelismo avviene su base algoritmica, ovvero ogni nodo serve la serie di algoritmi PSO-BCA, quindi sul singolo nodo non c'è una successiva parallelizzazione dei singoli algoritmi PSO e BCA; la potremo definire quindi una parallelizzazione ad alto livello. Tale architettura ha lo svantaggio che se il calcolo della fitness richiede un tempo comparabile con il tempo di aggiornamento dei parametri e delle nuove posizioni di tutti gli individui dell'algoritmo, i nodi che vengono identificati per lanciare la serie PSO-BCA rilasceranno la risorsa dopo un tempo eccessivamente lungo, vale a dire come se avessimo lanciato il PSO e il BCA su una singola macchina. Per ovviare a questo problema è stata implementata una struttura che parallelizza gli algoritmi più a basso livello, chiamata fitness-based. Se dall'analisi del profiling dell'algoritmo risulta che per il calcolo della fitness del singolo individuo vengono utilizzate la maggior parte delle risorse messe a servizio dal cluster, si può pensare di rivedere l'allocazione delle risorse di calcolo sfruttando il parallelismo per asservire il calcolo della fitness. In questa nuova architettura il cluster viene utilizzato interamente per asservire un algoritmo alla volta. In particolare inizialmente su di esso viene lanciato l'FSO, sul nodo master vengono fatte tutte le operazioni descritte tranne il calcolo delle fitness; questo viene delegato al singolo nodo, in particolare ad ogni nodo viene assegnato il calcolo della fitness di un individuo. Precisiamo quindi che l'associazione che viene fatta non è individuo-nodo ma fitness-nodo, è sempre compito poi del nodo master raccogliere tutti i dati delle fitness e aggiornare le posizioni successive. Sempre sul nodo master si opera anche, nel caso in cui fosse richiesto, il lancio della FM, che come intuibile non richiede particolare sforzo computazionale. In questo caso ogni singola FM non viene indagata ulteriormente con la serie PSO-BCA, ma viene utilizzata solamente per individuare il minimo globale. Successivamente quindi viene centrato un sub-dominio nel quale viene lanciato il PSO. Quindi nella seconda fase tutto il cluster serve il PSO con identico funzionamento tranne che per la FM. Parimenti nella terza fase per quanto riguarda il BCA.

Algoritmi di ottimizzazione come campionatori intelligenti

In un processo di ottimizzazione, prima che l'algoritmo intercetti il punto candidato ad essere minimo globale, questo ha indagato solitamente una grande quantità di punti, e di essi ne ha quindi estratto il valore del funzionale. Tali valori di solito non vengono registrati, ma a ben vedere, rappresentano informazioni estratte dal modello che risulterebbero utili qualora si volesse ricostruire ad esempio il funzionale. Lo scopo di questo paragrafo è proprio quello di considerare l'algoritmo di ottimizzazione come un campionatore intelligente in grado di fornire la mappa del funzionale. Per valutare questa capacità è stata valutata la capacità di ricostruzione di alcuni algoritmi con e senza FM, in particolare , MFSO, FSO, PSO, GA; quest'ultimo non descritto nel seguente lavoro di tesi, ma del quale il lettore può reperire informazioni vastamente presenti in letteratura visto il suo tenore storico. Gli algoritmi vengono lanciati in due configurazioni diverse in serie. Nella prima l'obiettivo è quello di individuare i minimi della funzione, invece nella seconda è quello di individuare il massimo. Ovviamente per ricostruire il funzionale l'algoritmo deve essere in grado esplorare lo spazio delle soluzioni in modo esaustivo. Per questo motivo la quantità di dominio indagato dall'algoritmo, e quindi la capacità di ricostruire il funzionale in esame, può essere considerato un metodo per misurarne la capacità esplorativa.

Alla fine del processo effettuato nelle due configurazioni descritte, la nuvola di punti campionati viene fornita a un interpolatore. Sono stati condotti alcuni test per validare la proprietà esplorative degli algoritmi con il metodo citato. In particolare ogni algoritmo è stato lanciato 30 volte su vari benchmark. In tabella 2 sono riportati i risultati ottenuti utilizzando come funzione di costo la Bird Function:

$$f(x,y)=\sin(x)\exp\left[\left(1-\cos(y)\right)^2\right]+\cos(y)\exp\left[\left(1-\sin(x)\right)^2\right]+\left(x-y\right)^2. \qquad (6)$$

Nella simulazione sono stati utilizzati lo stesso numero di individui (10) e di iterazioni (2000).

La percentuale di area investigata viene calcolata dividendo il dominio in sub domini aventi il 5% della dimensione del dominio totale. Una partizione viene considerata investigata quando almeno un punto è stato campionato in essa dall'algoritmo.

Inoltre vengono forniti l'MPE (Mean Percentage Error) e la sua varianza per tutti i test effettuati, individuati nella ricostruzione della funzione. Nell'ultima colonna sono presenti il numero di test falliti, ovvero quando l'algoritmo non ha investigato il 100% del dominio. L'MPE viene calcolato solo nella regione investigata, anche se questa non rappresenta il 100%.

Tabella 2: risultati dei test effettuati sulla bird function. Ogni algoritmo viene [-9 -9].

Algorithm	MPE	Variance	Mean Area[%]	Variance Area[%]	Failed over 30 tests
MFSO+FM	0.0034538	1.299e-005	99.2645	2.9887	0
MFSO	0.70111	0.28314	54.6482	8.8994	30
FSO+FM	0.062557	0.016329	98.3054	11.3105	20
FSO	0.75573	0.63839	58.4974	9.2509	30
PSO+FM	0.8148	0.5132	86.7807	4.1258	29
PSO	0.5493	0.35416	22.2916	9.2859	30
AG+FM	0.32664	0.15338	17.0897	2.7381	30
AG	0.11969	0.016434	7.569	0.40311	30

E’ stata fatta poi un’ulteriore analisi per stimare la sensitività che gli algoritmi hanno rispetto all’aumento delle iterazione e all’utilizzo della FM. Tali test sono stati effettuati sulla Giunta function:

$$f\left(x_1,x_2\right)=0.6+\sum_{i=1}^{2}\left[\sin\left(\frac{16}{15}x_i-1\right)+\sin^2\left(\frac{16}{15}x_i-1\right)+\frac{1}{50}\left(4\left(\frac{16}{15}x_i-1\right)\right)\right] \qquad (7)$$

Dai grafici (che non vengono riportati in questa sezione solo per non appesantire la lettura, ma che possono essere consultati nel capitolo 2 della versione inglese che segue in codesto documento) è evidente che l’utilizzo della FM opera una sorta di boost al processo di esplorazione mantenendo il ranking originale degli algoritmi. Quindi la FM può essere considerata una tecnica a sé stante per l’aumento delle capacità esplorative di un algortimo, qualsiasi esso sia.

Validazione

Ottimizzazione di un TWT con collettore bistadio

In questa sezione viene presentata l'applicazione di MeTEO per l'ottimizzazione della tensione applicata agli elettrodi di un Collettore Multistadio di un TWT al fine di aumentarne l'efficienza. L’analisi elettromagnetica delle traiettorie all'interno del collettore è stata realizzata con il risolutore di elementi finiti COLLGUN, che calcola l'efficienza del collettore a ogni lancio, che è il parametro da ottimizzare. Nei test proposti, relativi a dei collettori a due stadi, la geometria e la mesh sono assegnati e non variano durante la fase di ottimizzazione. È stato sviluppato un *wrapper* per far dialogare direttamente MATLAB, in cui gira MeTEO, con il simulatore COLLGUN.

Ottimizzazione di un Collettore Bistadio [6].

Per quanto riguarda i parametri della mesh utilizzati e quelli relativi al simulatore, questi sono riportati nella tabella 3. Il numero complessivo di tetraedri è prossimo a 20000 elementi, mentre i punti relativi a questa mesh e quindi le incognite del sistema FEM risolvente sono quasi 6000. Il numero di particelle usate per la rappresentazione del fascio sono 500. La tolleranza di fine iterazioni per il ciclo del COLLGUN è stata fissata a 0.05%. Per tali valori il tempo di calcolo di ciascun run del solver di COLLGUN si aggira sui 90 secondi, e il numero di iterazioni è circa 4 .

Tabella 3: parametri di COLLGUN utilizzati per la simulazione del collettore bi-stadio.

Number of tetrahedral	About 20000
Number of node	About 600
Number of macro-particles	500
End Tolerance	0,05%
Number of iterations for each simulation	4-6
Computing time of each simulation	About 1'30"

METEO è stato lanciato 30 volte, qui riportiamo i dati relativi al miglior valore trovato. E' stata fatta un'inizializzazione casuale per l'FSO, mentre per il PSO e il BCA l'inizializzazione dipende dal punto fornito dall'algoritmo precedente. Il numero di individui utilizzato per ogni singola meta-euristica è di 10. Un numero maggiore di individui non migliora in modo significativo le prestazioni di METEO perché con più individui è possibile campionare più punti, ma dobbiamo ricordare che ogni punto campionato rappresenta il calcolo del funzionale. Nel nostro problema il tempo speso per il calcolo della funzione di costo è pari a lanciare il simulatore COLLGUN che richiede molto tempo per ogni lancio, circa 90 secondi. Per questo motivo non abbiamo superato il valore di 10 individui. Per mezzo di METEO, è possibile ottimizzare l'efficienza di questo dispositivo e ottenere un valore massimo del 88,81%. Nel report (Tabella 4), si evince la caratteristica principale di METEO, infatti il valore dell'efficienza trovato dall'FSO diminuisce gradualmente passando dopo l'applicazione rispettivamente del PSO e del BCA. Il principio della ibridazione fatta in METEO sta proprio in questa caratteristica. Nella migliore prova l'efficienza migliora dopo l'applicazione di ogni componente di METEO come indicato nella Tabella 4.

Tabella 4: efficienze raggiunte da ogni componente di MeTEO.

	FSO	PSO	BCA
Efficiency	88.17%	88.64%	88.81
Number of iterations	40	20	20
FM	3		

Nella tabella 5 invece sono riportati i valori e i parametri relativi alla miglior efficienza trovata.

Tabella 5: miglior configurazione fornita da MeTEO

Spent beam power	192W
Power recovered	170W
Collector's efficiency	88.81%
Stage 1	
Potential	-3.05kV
Current	38.73 mA
Stage 2	
Potential	-4.22 kV
Current	12.34 mA

Ottimizzazione della efficienza di dispositivi TWT

Nell'ottimizzazione della geometria sono stati scelti un numero di parametri fissi che sono: la lunghezza del diametro interno ed esterno del primo stage e il diametro del secondo stage. E' utile notare che ogni stage viene ottenuto utilizzando tre solidi primitivi dal COLLGUN: cilindri, coni, e coni troncati. Per esempio le due configurazioni mostrate in figura 6, sono ottenute utilizzando il valore minimo e massimo per la lunghezza(10 mm) e il raggio (5mm). Il fascio elettronico ha una tensione di riferimento di 4.8 kV, un raggio di 0.63mm, e una corrente di 53.2 mA, con potenza di 190 W. Le tensioni assegnate agli elettrodi sono di 2.4 kV and 3.6kV.

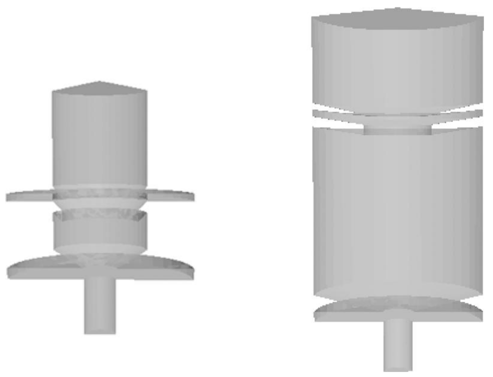


Figura 6: sezione delle geometrie dei collettori dei casi limite.

Partendo da inizializzazioni random con efficienza minore del 75%, dopo 100 iterazioni (10FSO,30PSO,60BCA), si ottiene la configurazione mostrata in figura 7, avente efficienza del 84.8%. Il tempo richiesto per di calcolo è di circa 24h. In aggiunta si sono comparati i risultati con un algoritmo random walk fatto girare per 300 iterazioni per 5 volte, e il miglior risultato trovato è un'efficienza dell'82%.

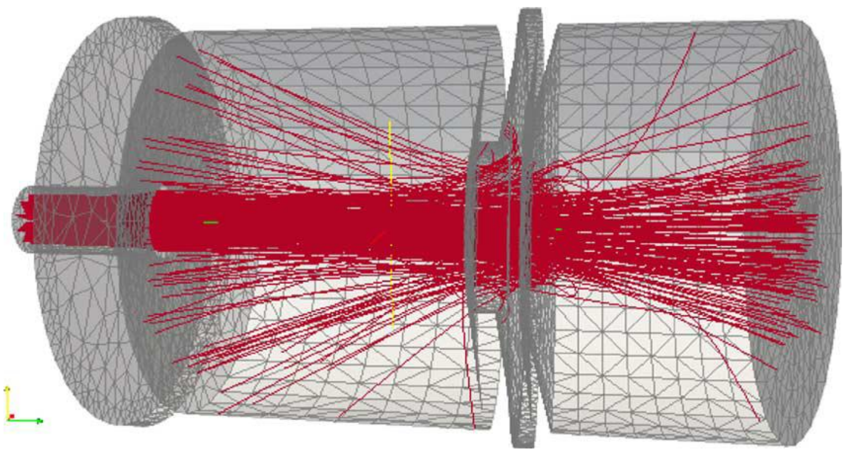


Figura 7: geometria e traiettorie elettroniche trovate con l'ottimizzatore MeTEO.

Si è inoltre ottimizzato il campo magnetico focalizzante applicato a un TWT con un collettore bistadio. Nella figura 8 è possibile osservare le traiettorie elettroniche e il profilo di campo magnetico. I valori dell'electron beam utilizzati sono presi in letteratura, riferimento di 4.8 kV, raggio di 0.63 mm, e una corrente di 53.2mA, in grado di generare una potenza di 190W. Le tensioni dei due elettrodi sono fissate a 2.4 e 3.8 kV.

Un tetraedro irregolare di circa 20000 elementi è stato utilizzato per la mesh. Come fitness, è stata utilizzata una combinazione tra l'efficienza del collettore η, la corrente di backstreaming in mA, I_{back}. COLLGUN fornisce un risultato in 3min per calcolare la singola fitness. Partendo da una configurazione con efficienza pari all'81% dopo 300 interazioni (100 FSO, 100 PSO, 100 BCA) si raggiunge un'efficienza dell'83.1%.

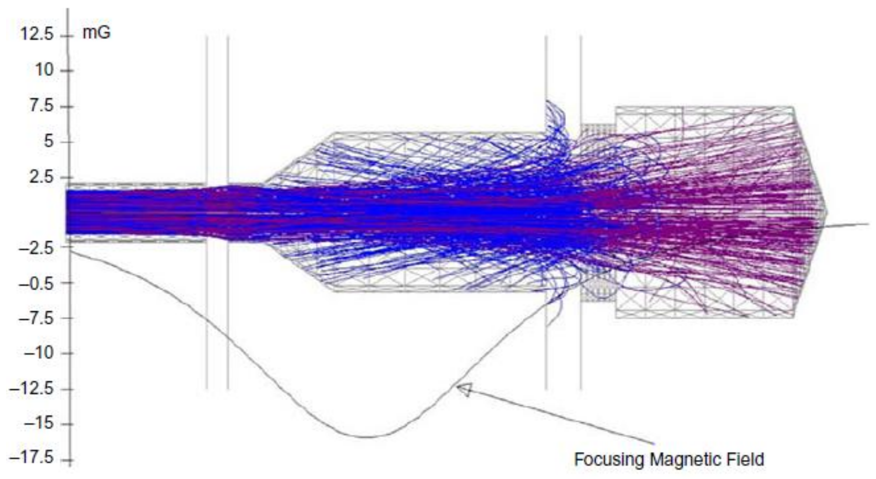


Figura 8: sezione 2D della sezione del collettore bistadio, insieme con il campo magnetico focalizzate e le traiettorie simulate con COLLGUN.

Risoluzione del problema inverso Amid_Pro proposto da Schittkowski.

MeTEO è stato utilizzato con successo anche per la risoluzione di uno dei tanti problemi inversi proposti da Schittkowski. Infatti il problema dell'ottimizzazione dei TWT si può considerare come un problema di scelta ottima e non di vero e proprio problema inverso. Nel problema Amid_Pro, viene presa in considerazione la proteina RNase T1(S54G/P55N-RNaseT1). RNase T1 è una piccola proteina di 104 residui, che viene utilizzata come "proteina modello" in laboratorio per investigare il limite di folding della prolina, un aminoacido apolare avente una molecola chirale. Nello schema 1 è riportato il modello cinetico per l'Unfolding e l'Isomerizzazione dell'RNase T1a.

$$\begin{tikzcd}
N \arrow[r, shift left, "k_{43}"] & U^{39c}_{55c} \arrow[l, shift left, "k_{34}"] \arrow[r, shift left, "k_{32}"] \arrow[d, shift left, "k_{21}"] & U^{39c}_{55t} \arrow[l, shift left, "k_{23}"] \arrow[d, shift left, "k_{21}"] \\
 & U^{39t}_{55c} \arrow[u, shift left, "k_{12}"] \arrow[r, shift left, "k_{32}"] & U^{39t}_{55t} \arrow[l, shift left, "k_{23}"] \arrow[u, shift left, "k_{12}"]
\end{tikzcd}$$

Schema 1: Modello cinetico per l'Unfolding e l'Isomerizzazione dell'RNase T1a.

Nel modello $U^{39c}_{55c}, U^{39t}_{55c}, U^{39c}_{55t}, U^{39t}_{55t}$ rappresentano le forme unfolded dell'RNase T1a con Pro39 e Pro55 nelle conformazioni cis (c) o trans (t), invece N è la proteina nativa. Il modello proposto da Schittkowski, generalizza la cinetica concentrandosi sulla variazione delle 4 specie presenti. Avremo quindi un sistema di equazioni differenziali con le 4 specie e le relative concentrazioni come riportato nel gruppo di equazioni 8:

$$\begin{cases} \dfrac{dy_1}{dt} = -k_f y_1 \\ \dfrac{dy_2}{dt} = k_f \left(c - y_2\right) \\ \dfrac{dy_3}{dt} = k_f y_4 - 0.1 k_n \left(y_2 - y_3\right) - 0.9 k_n y_3 \\ \dfrac{dy_4}{dt} = -k_f y_4 - 0.9 k_i y_4 + 0.1 k_i y_3 \end{cases} \tag{8}$$

Nel problema inverso si hanno dei dati sperimentali che indicano l'andamento nel tempo delle concentrazioni delle varie specie, e si vuole trovare la *n-pla* di 4 parametri che inseriti nel modello espresso dalle equazioni differenziali riportate in (8) restituiscono proprio gli stessi andamenti. Per i dati presi in esame i valori ottimi da attribuire ai parametri sono:

$$\left[k_f,\ k_i,\ k_n,\ c\right]=\left[10^{-4},\ 10^{-3},\ 1.18\cdot 10^{-4},\ 1.2\cdot 10^{5}\right] \tag{9}$$

Come mostrato nella (Tabella 6) METEO è in grado di trovare la soluzione migliore rispetto all'impiego delle singole euristiche. In questo test le prestazioni sono state calcolate su 50 lanci.

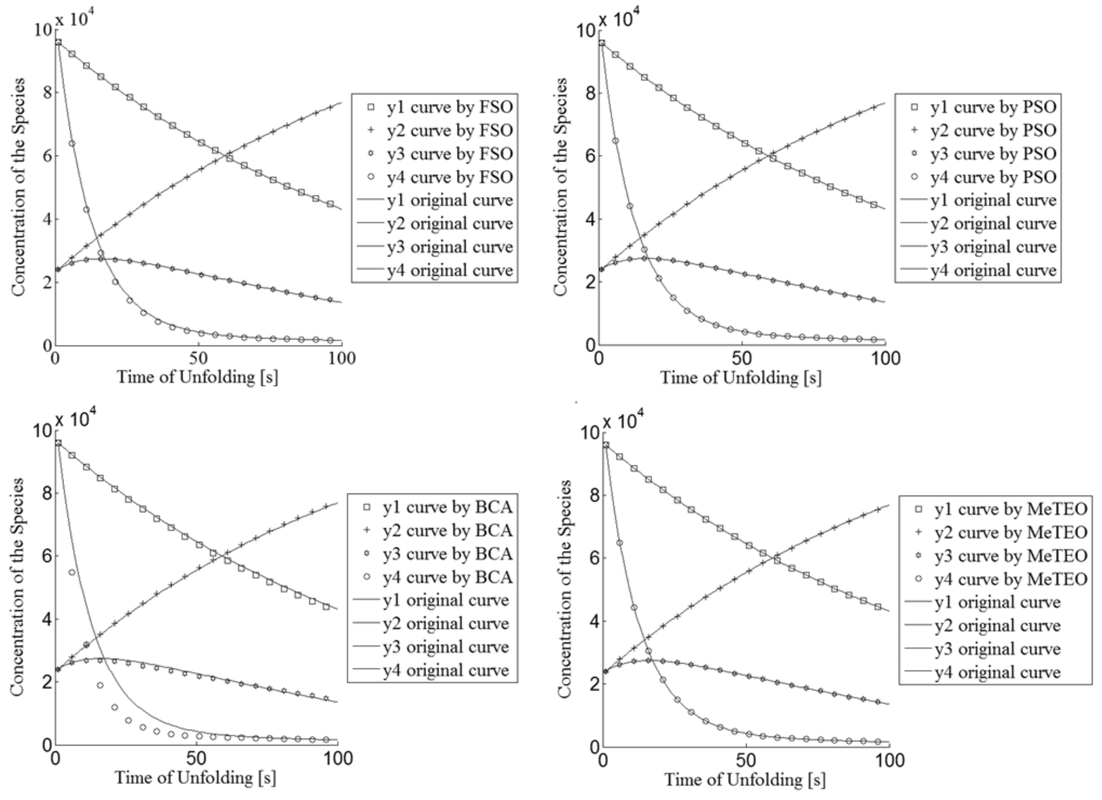


Figure 9: Comparazione tra le curve riprodotte dalle single euristiche (FSO, PSO, BCA) e da MeTEO.

In figura 9 sono rappresentate le curve per il modello amid_pro riprodotte dalla singola euristica e da METEO.

Tabella 6:Risultati relativi alla risoluzione del problema inverso Amid_Pro attraverso MeTEO e i suoi singoli componenti.

Algorithm	Parameter estimated $\left[10^{-4},\ 10^{-3},\ 1.18\cdot 10^{-4},\ 1.2\cdot 10^{5}\right]$	Percentage error on the 4-dim Amid_pro
MeTEO	$9.9975\cdot 10^{-5}$,$1.0004\cdot 10^{-3}$,1.179510^{-4},120039,57	0.0060%
FSO	$9.9607\cdot 10^{-5}$,$1.0427\cdot 10^{-3}$,$1.1505\cdot 10^{-4}$,120076,66	0.7709%
PSO	$1.0005\cdot 10^{-4}$,$1.004\cdot 10^{-3}$,$1.1793\cdot 10^{-4}$,119985,80	0.1628%
BCA	$1.0285\cdot 10^{-4}$,1.011710^{-3},1.268610^{-4},116088,48	2.0764%

E’ stata poi effettuata una analisi statistica minima in cui sono state misurate varianza, media e l’indice R^2. I risultati della statistica sono riportati in Tabella 7.

Tabella 7: Valori della statistica effettuata per il singolo componente e per MeTEO nella risoluzione del problema inverso Amid_Pro.

Amid_pro[0.3]	*MPE*		
Algorithm	*Mean*	σ^2	R^2
BCA	10.8683	5.948e-2	0.3476
PSO	2.9633	7.9884e-3	0.9968
FSO	1.0674	0.4048e-4	0.9987
METEO	0.42562	0.1292e-4	0.9999

Risoluzione del problema inverso sull’identificazione dei parametri del modello di Jiles-Atherton di un ciclo di isteresi magnetica.

MeTEO è stato testato anche nel problema dell’identificazione dei parametri del modello di Jiles-Atherton di isteresi magnetica assegnato un ciclo sperimentale di dati. Nell’equazione 10 ne viene fatto un richiamo.

$$\frac{dM}{dH}=\frac{(1-c)\frac{dM_{irr}}{dH_e}+c\frac{dM_{an}}{dH_e}}{1-\alpha c\frac{dM_{an}}{dH_e}-\alpha(1-c)\frac{dM_{irr}}{dH_e}} \qquad (10)$$

M_{an} è la magnetizzazione anisteretica fornita dall’equazione di Langevin:

$$M_{an}(H_e)=M_s\left(\coth\left(\frac{H_e}{a}\right)-\frac{a}{H_e}\right) \qquad (10.a)$$

Nella quale $H_e=H+\alpha M$. Nella (10) M_{irr} è la magnetizzazione irreversibile definita come:

$$\frac{dM_{irr}}{dH_e}=\frac{M_{an}-M_{irr}}{k\delta} \qquad (10.b)$$

dove $\delta=sign\left(\frac{dH}{dt}\right)$.

I parametri che devono essere identificati nel modello sono α,a,c,k,M_s e il loro significato fisico il seguente: a è il fattore di forma, c il coefficiente di reversibilità dei domini, M_s la magnetizzazione di saturazione, e infine α e k che rappresentano le perdite e l’iterazione tra i domini.

Nell'eseguire i test si è fatto uso di inizializzazioni randomiche. I valori corretti dei parametri da trovare sono: $[a,\ k,\ \alpha,\ c,\ M_s] = [24.70,\ \ 60,\ \ 6.9010^{-5},\ \ 0.053,\ \ 1.16 \cdot 10^5]$.

Tabella 8: Risultati relativi al problema inverso di Jiles-Atherton delle single euristiche e di MeTEO.

Algorithm	Parameter estimated $[a,\ k,\ \alpha,\ c,\ M_s]$	Percentage error on the experimental loop points
MeTEO	$[24.34\ \ 59.84\ \ 6.26 \cdot 10^{-5}\ \ 0.050\ \ 1.158409 \cdot 10^{-5}]$	0.05368%
FSO	$[14.36\ \ 64.12\ \ 3.57 \cdot 10^{-6}\ \ 0.046\ \ 1.072892 \cdot 10^{5}]$	4.71548%
PSO	$[17.27\ \ 55.73\ \ 6.22 \cdot 10^{-5}\ \ 0.004\ \ 1.047480 \cdot 10^{5}]$	7.07373%
BCA	$[22.20\ \ 53.88\ \ 1.080195758008603 \cdot 10^{-4}\ \ 0.006\ 1.044000 \cdot 10^{5}]$	8.1803%

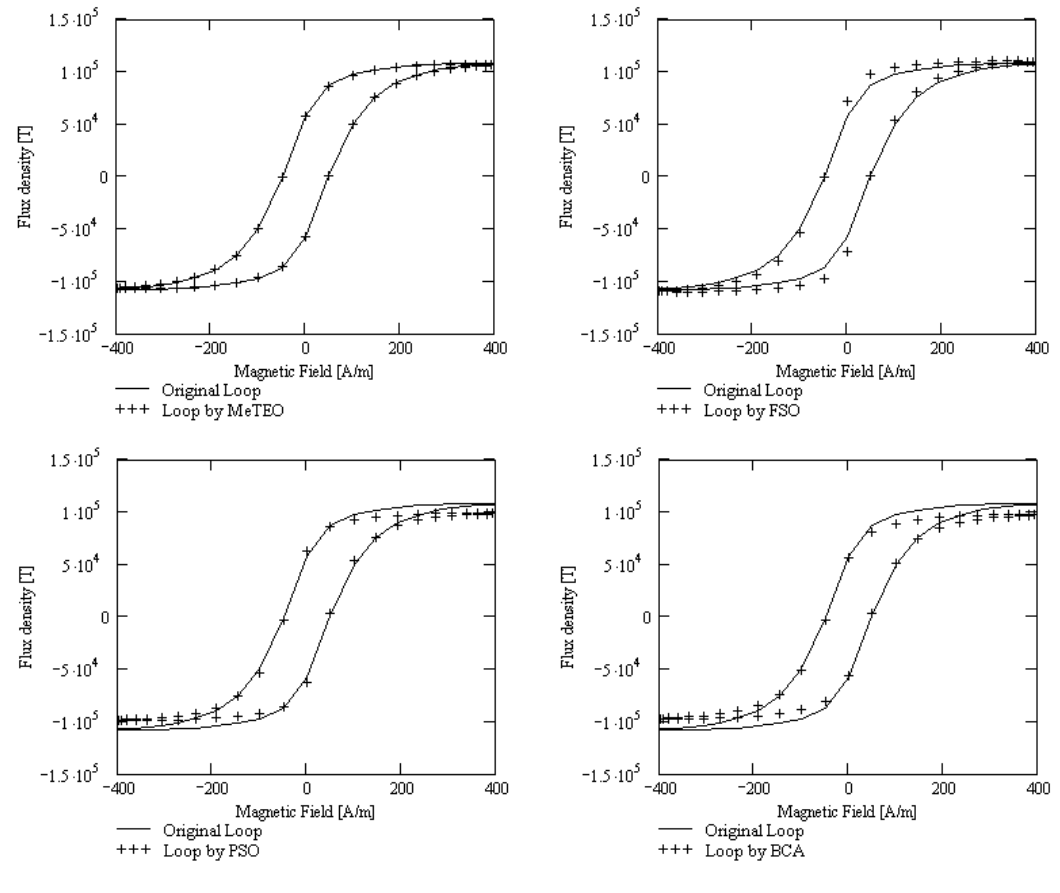


Figura 10: Comparazione tra i cicli ricostruiti attraverso l'utilizzo delle singole euristiche e di MeTEO. La linea continua rappresenta il ciclo vero.

E' evidente che il valore dell'MPE è molto più basso nel caso dell'utilizzo di MeTEO. Anche in questo caso è stata fatta un'analisi statistica di cui riportiamo i risultati in Tabella 9.

Quindi in conclusione, l'utilizzo di MeTEO nei tre problemi inversi si è dimostrato molto fruttuoso. I tre problemi sono completamente diversi l'uno dall'altro, il primo sui TWT è quello di trovare il valore ottimo di un funzionale sconosciuto (nel caso specifico un massimo), il secondo (Amid_Pro) è un verso e proprio problema inverso a 4 dimensioni, e infine il terzo dell'identificazione dei parametri di Jiles-Atherton, sempre un problema inverso ma di natura isteretica. L'eterogeneità dei problemi presi in considerazione ci permettere di generalizzare la potenza di MeTEO, premesso il fatto che per il teorema di Wolpert non esiste una sola euristica (anche ibridizzata) che sia capace di risolvere ed affrontare in modo efficace ogni problema di ottimizzazione.

Tabella 9: Analisi statistica nella risoluzione del problema inverso di Jiles-Atherton applicando le singole euristiche e MeTEO.

Jiles-Atherton[0.3]	*MPE*		
Algorithm	*Mean*	σ^2	R^2
BCA	2.4268	2.9507e-2	0.5870
PSO	1.6388	1.6527e-3	0.9968
FSO	1.294	0.9030e-3	0.9993
METEO	0.53552	1.4576e-5	0.9996

Validazione del BFSO: distribuzione ottima di cavi di potenza sotterranei.

Per validare il BFSO è stato utilizzato un problema relativo alla minimizzazione del flusso magnetico generato in superficie da cavi sotterranei di potenza. I circuiti utilizzati per la validazione sono stati estratti da [15], i numeri di circuiti impiegati sono rispettivamente 4 e 8.

Le combinazioni che si possono avere per ogni cavo sono 6, variando ovviamente la terna trifase: 123;132; 213; 231; 312; 321, per esempio se abbiamo la combinazione 6345, questo significa che i cavi del primo circuito sono disposti nella combinazione 321, i cavi del secondo fascio come 213 e così via. Altri autori scelgono le RST per identificare la terna trifase, nel nostro caso abbiamo scelto 123 solo per una più agile implementazione software. Poniamo N_C come il numero di circuiti N_B pari al numero di fasci, allora un individuo del BFSO viene codificato con un array di 0 e 1 come segue:

- Se $N_C = 4$ viene utilizzata un array binario con $(3 \cdot N_B) + (2 \cdot N_C)$ posizioni;
- Se $N_C = 8$ viene utilizzato un array binario con $2 \cdot (3 \cdot N_B)$ posizioni;

I risultati mettono a confronto il BFSO con il DPSO. Dai dati si può constatare come la media dei risultati è minore sia per il displacement a 4 che a 8 circuiti per il BFSO. Il BFSO continua a presentare lo stesso comportamento che ha nel continuo ovvero di esploratore dello spazio delle soluzioni, infatti come si può notare dalla tabella, risulta essere lacunoso nella convergenza, infatti nel caso a 8 circuiti il valore minore viene individuato con il PSO.

Tabella 10: risultati relativi ai circuiti a 4 e 8 circuiti

	4 BUNDLES-4 CIRCUITS				8 BUNDLES-8 CIRCUITS			
Algorithm	Mean	Variance	Best Configuration	Best Fitness Value	Mean	Variance	Best Configuration	Best Fitness Value
FSO	6.3148e-007	1.7522e-016	56533124	6.2566e-007	5.9456e-007	1.0058e-014	[34645531 73854126]	3.8209e-007
PSO	6.4005e-007	3.262e-016	56533124	6.2566e-007	6.1731e-007	1.1908e-014	[11411164 47812365]	3.7983e-007

Swarm Circuits in grado di risolvere problemi inversi e di ottimizzazione.

In letteratura sono presenti vari tentativi di rappresentare gli algoritmi della Swarm Intelligence come sistemi dinamici continui. Per esempio, si possono trovare degli importanti contributi in [11], [12] e [13]. I lavori proposti riguardano lo studio della dinamica del sistema, ma nessuno ha indirizzato tale ricerca a una possibile implementazione fisica del modello. Infatti, le componenti hardware, almeno nel settore dell'elettronica, che riguardano l'implementazione degli algoritmi Swarm Inspired sono solo delle implementazioni embedded con microcontrollori e FPGA. Nella ricerca svolta invece, si è introdotto il concetto di swarm-circuit, ovvero un circuito nel quale le caratteristiche cinematiche delle traiettorie seguite dai membri dello swarm, sono riprodotte in termini di forme d'onda rappresentate dalle tensioni misurate su dei capacitori e delle correnti che attraversano degli induttori. Tale approccio quindi porta a un duplice risvolto, per prima cosa possiamo adottare tutti i metodi della Teoria dei Circuiti per analizzare la cinematica dei membri dello swarm, (Trasformate di Laplace, Metodo del Tableau etc.), e in seconda istanza, tracciare le basi teoriche per una soluzione hardware innovativa, da poter sfruttare per le ottimizzazione real-time.

Algoritmi della Swarm Intelligence tradotti in sistemi continui.

L'intento di questa sezione è quello di mostrare come sia possibile convertire l'algoritmo numerico FSO in un sistema continuo nel dominio del tempo. Consideriamo prima di tutto l'equazione (11), che rappresenta l'aggiornamento della velocità di un individuo dello swarm:

$$u_k^{<j>}(t+1)=\omega v_k^{<j>}(t)+\lambda[p_{bestk}^{<j>}(t)-x_k^{<j>}(t)]+\gamma[g_{bestk}^{<j>}(t)-x_k^{<j>}(t)]+\dots$$
$$\dots\sum_{m=1}^{N}h_{km}u_m^{<j>}(t)\quad \forall j-th\ dimension \tag{11}$$

Riscriviamo poi la (11), introducendo un valore Δt:

$$u_k(t+\Delta t)=\omega\ u_k(t)+\lambda\ [pbest_k(t)-x_k(t)]+\gamma\ [gbest(t)-x_k(t)]+\dots$$
$$\dots+\sum_{m=1}^{N}h_{km}u_m(t) \tag{12}$$

L'equazione (12) può essere interpretata come lo formula di Eulero per l'integrazione di un sistema di equazioni differenziali rappresentante un sistema fisico continuo, se si assume che $\Delta t \in \mathrm{R}$ e $\Delta t \to 0$ come segue:

$$\frac{u_k(t+\Delta t)-u_k(t)}{\Delta t}=\frac{(\omega-1)u_k(t)+\lambda[pbest_k(t)-x_k(t)]+\gamma[gbest(t)-x_k(t)]+\sum_{m=1}^{N}h_{km}u_m(t)}{\Delta t} \tag{13}$$

In (13) la variabile indipendente è ora considerata come variabile $t \in \mathrm{R}$, mentre in (11) veniva considerata semplicemente come un numero naturale per contare il numero di iterazioni effettuate.

Anche la formula di aggiornamento della posizione del singolo individuo può essere riscritta come segue:

$$x_k(t+\Delta t)=u_k(t)\Delta t+x_k(t) \tag{14}$$

E' importante inoltre notare che nelle (12) e (13) il valore della variabile Δt gioca un ruolo "dimensionale". Avendo riconosciuto che la (13) rappresenta la formula di Eulero per l'integrazione di sistemi continui, il valore Δt deve essere impostato il più piccolo possibile per ottenere una integrazione ideale. Infatti, un'errata scelta del passo di integrazione implicherebbe una "cattiva" integrazione del sistema, con delle traiettorie che non risponderebbero alla reale evoluzione del sistema.

Quindi l'algoritmo in cui il passo è pari ad 1, rappresenta solo una delle possibili integrazioni del sistema reale continuo. Se noi consideriamo $\Delta t \to 0$ è immediato riscrivere la (13) nel dominio continuo come segue:

$$\frac{d}{dt}u_k(t)=\tilde{\omega}u_k(t)+\tilde{\lambda}[pbest_k(t)-x_k(t)]+\tilde{\gamma}[gbest(t)-x_k(t)]+\sum_{m=1}^{N}\tilde{h}_{km}u_m(t) \tag{15}$$

Nella (16) vengono introdotti dei nuovi parametri normalizzati, essi sono definiti come segue:

$$\begin{cases} \tilde{\omega} = \dfrac{(\omega-1)}{\Delta t} \\ \tilde{\lambda} = \dfrac{\lambda}{\Delta t} \\ \tilde{\gamma} = \dfrac{\gamma}{\Delta t} \\ \tilde{h}_{km} = \dfrac{h_{km}}{\Delta t} \end{cases} \tag{16}$$

dove $\tilde{h}_{km} = \tilde{h} = h / \Delta t$ è pari a uno se il k-esimo individuo controlla l' m-esimo, altrimenti $\tilde{h}_{km} = 0$. E' evidente dalla (16) che Δt è da intendersi come un vero e proprio nuovo parametro. In aggiunta è utile capire come vengono modellati il personal e il global best di ogni individuo. Consideriamo come eccitazione del sistema continuo la seguente quantità:

$$\Im_k(t) = \tilde{\lambda} \cdot pbest_k(t) + \tilde{\gamma} \cdot gbest(t) \tag{17}$$

e poniamo poi:

$$\tilde{\mu} = \tilde{\lambda} + \tilde{\gamma} \tag{18}$$

L'equazione di stato (15) può essere riscritta per ogni k-esima particella come segue:

$$\frac{d}{dt} u_k(t) = \tilde{\omega} u_k(t) - \tilde{\mu} x_k(t) + \sum_{m=1}^{N} \tilde{h}_{km} u_m(t) + \Im_k(t) \tag{19}$$

L'equazione di stato (19) deve essere accoppiata a una seconda equazione di stato relativa alla posizione $x_k(t)$. Ovviamente, per $\Delta t \to 0$, noi semplicemente otteniamo la definizione cinematica della velocità:

$$\frac{d}{dt} x_k(t) = u_k(t) \tag{20}$$

Le equazioni (19) e (20) descrivono il sistema dinamico e la sua evoluzione nel dominio del tempo t, con delle forzanti espresse in (17) che costituiscono le equazioni di stato del modello continuo di un individuo dell'algoritmo swarm-based FSO. Se poi consideriamo lo stormo nella sua totalità formato quindi da N uccelli, otteniamo un sistema di equazioni di stato di dimensioni $2N \times 2N$ che riportiamo scritto in forma matriciale compatta di seguito:

$$\frac{d}{dt}\begin{bmatrix} \mathbf{u} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{A_{1,1}} & \mathbf{A_{1,2}} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{x} \end{bmatrix} + \begin{bmatrix} \mathbf{F} \\ \mathbf{0} \end{bmatrix} \tag{21}$$

La sub-matrice $\mathbf{I}$ (21) è la matrice identità avente dimensioni $N \times N$. La sub-matrice $\mathbf{A_{1,1}}$ è una matrice quadrata con dimensioni $N \times N$ definita come:

$$\mathbf{A_{1,1}} = \mathbf{M} + \mathbf{H} \tag{22}$$

Dove $\mathbf{M} = \tilde{\omega} \cdot \mathbf{I}$, mentre $\mathbf{H}$ tiene conto della componente legata alle velocità, che esprime il collective behavior dell'algoritmo. Gli elementi che costituiscono $\mathbf{H}$ sono e $h_{km} = \tilde{h}$

se il $k-th$ individuo controlla la velocità dell' $m-th$ individuo, altrimenti zero. E' evidente che la matrice $\mathbf{H}$ ha elementi diversi da zero solo nel caso dell'FSO, mentre per il PSO tale matrice diventa pari a una matrice nulla. L'altra matrice che appare nella (21) ha dimensioni $N \times N$ ed è definita come:

$$\mathbf{A}_{1,2} = -\tilde{\mu} \cdot \mathbf{I} \tag{23}$$

Infine, il vettore $\mathbf{F}$ ha ogni k-esime riga definita come $\Im_k(t)$ nell'equazione (17).

The Swarm-Circuit

Una volta scritte le equazioni di stato che rappresentano un algoritmo swarm-based nel continuo, queste possono essere implementate considerandole come equazioni che governano un circuito elettrico. Le equazioni (19) e (20) descrivono lo stato di un *k-th* lato del circuito che rappresenta l'algoritmo swarm-based. Tale lato viene raffigurato in figura 11, imponendo una corrispondenza 1 a 1 tra la corrente che fluisce il lato serie e la velocità del *k*-th membro del gruppo cioè $i_k(t) \propto u_k(t)$, mentre la tensione misurata ai terminali del capacitore $v_{C,k}$ può essere assunta proporzionale alla posizione dell'individuo del gruppo nell'algoritmo. Avremo quindi $x_k(t) \propto v_{C,k}$. La forzante invece viene rappresentata con due generatori di tensione indipendenti $\Im_k(t) = e_k(t) + g(t)$, che tengono in conto il personal best e il global best. Abbiamo in particolare: $e_k(t) \propto \lambda \cdot pbest_k(t)$ e $g(t) \propto \tilde{\gamma} \cdot gbest(t)$.

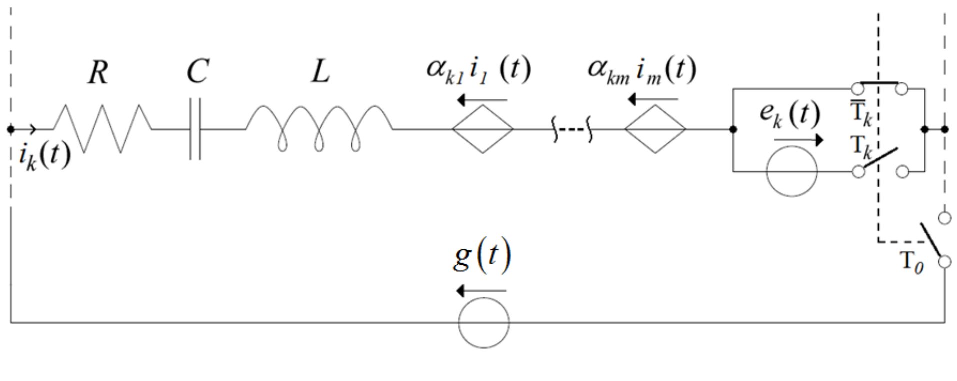


Figura 11: k-th lato di uno Swarm-Circuit.

Le equazioni che governano il k-th ($\forall k = 1 \ldots N$) lato in Figura 11, sono:

$$\frac{d}{dt} i_k = -\frac{1}{L}\left(R i_k + v_{C,k} + \sum_{j=1,\, j\neq k}^{N} \alpha_{k,j} i_k - e_k(t) - g(t) \right) \tag{24}$$

$$\frac{d}{dt} Q_{C,k} = i_k \tag{25}$$

In (25) abbiamo introdotto una nuova variabile di stato, $Q_{C,k} = C \cdot v_{C,k}$ per ottenere una perfetta corrispondenza tra (19)-(20) e (24)-(25). Così derivando la (19) e operando con la (21) otteniamo la corrispondenza tra la formula dell'algoritmo continuo e quella del lato del circuito:

(26)

$$\begin{cases} \frac{d^2}{dt^2} u_k(t) = \tilde{\omega} \frac{d}{dt} u_k(t) - \tilde{\mu} u_k(t) + \sum_{m=1}^{N} \tilde{h}_{km} \frac{d}{dt} u_m(t) + \frac{d}{dt}[\tilde{\lambda} \cdot p_{best\,k}(t) + \tilde{\gamma} \cdot g_{best}(t)] \\ \frac{d^2}{dt^2} i_k = -\frac{R}{L}\frac{d}{dt} i_k - \frac{1}{LC} i_k - \sum_{m=1, m\neq k}^{N} \frac{\alpha_{k,m}}{L} \frac{d}{dt} i_m + \frac{1}{L}\frac{d}{dt}[e_k(t) + g(t)] \end{cases} \quad \forall k = 1 \dots N$$

Da questa formula otteniamo la corrispondenza tra i parametri dell'FSO e i parametri del circuito:

$$\tilde{\omega} = -\frac{R}{L}, \tilde{\mu} = \frac{1}{LC}, \tilde{h}_{km} = -\frac{\alpha_{km}}{L}, \ \tilde{\lambda} \cdot p_{best\,k}(t) = \frac{1}{L} \cdot e_k(t) \text{ and } \tilde{\gamma} \cdot g_{best}(t) = \frac{1}{L} \cdot g(t). \qquad (27)$$

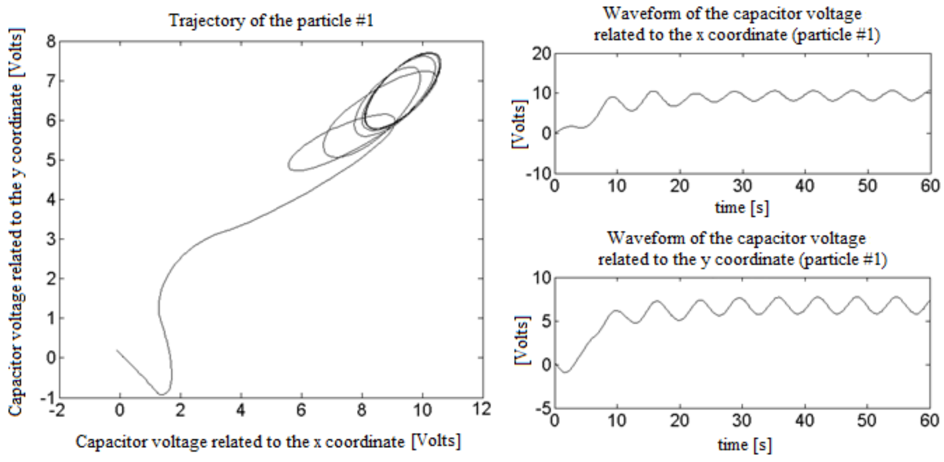
Il modello è stato poi trattato nel dominio di Laplace e sono stati individuati i valori che permettono al circuito di generare dinamiche convergenti, divergenti o di oscillazione. Riportiamo di seguito le tre condizioni che devono essere rispettate, per ottenere i comportamenti appena citati, in particolare otteniamo una convergenza per :

$$-\frac{R}{N-1} < \alpha < R \qquad (28)$$

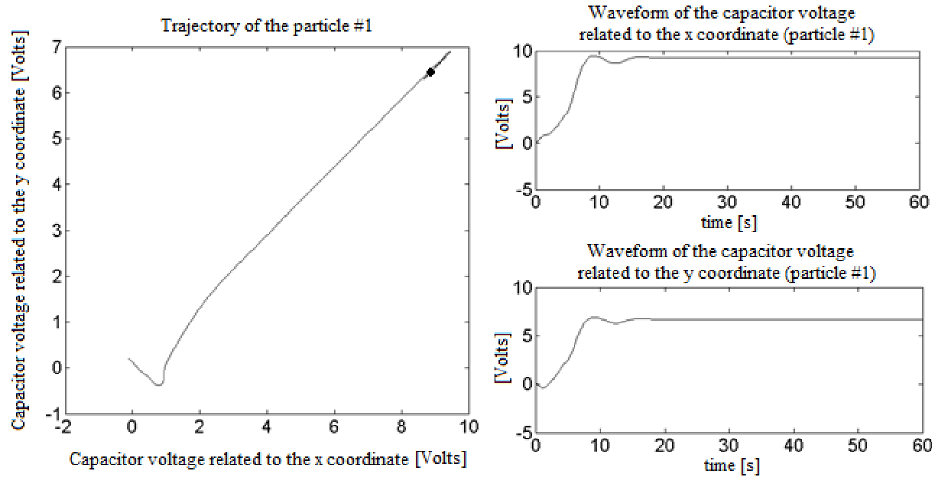
E un comportamento oscillatorio (con poli quindi complessi e coniugati) per:

$$-2\sqrt{\frac{L}{C}} + R < \alpha < 2\sqrt{\frac{L}{C}} + R \quad \text{or} \quad \alpha > \frac{2\sqrt{\frac{L}{C}} + R}{N-1} \qquad (29)$$

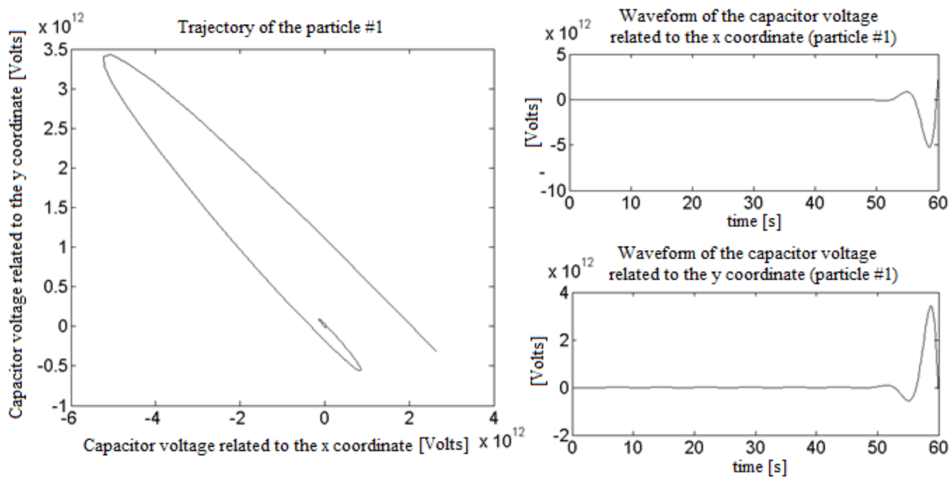
Nella figura 12 invece vengono riportate le traiettorie di un individuo per i tre comportamenti appena citati, nella caption della figura sono riportati i parametri che sono stati utilizzati.



(a)



(b)



(c)

Figura 12: (La funzione utilizzata è stata la Bird Function).(a) $\alpha = 1\ \Omega$**, (b)** $\alpha = 0.25\ \Omega$**, (c)** $\alpha = 2\ \Omega$.

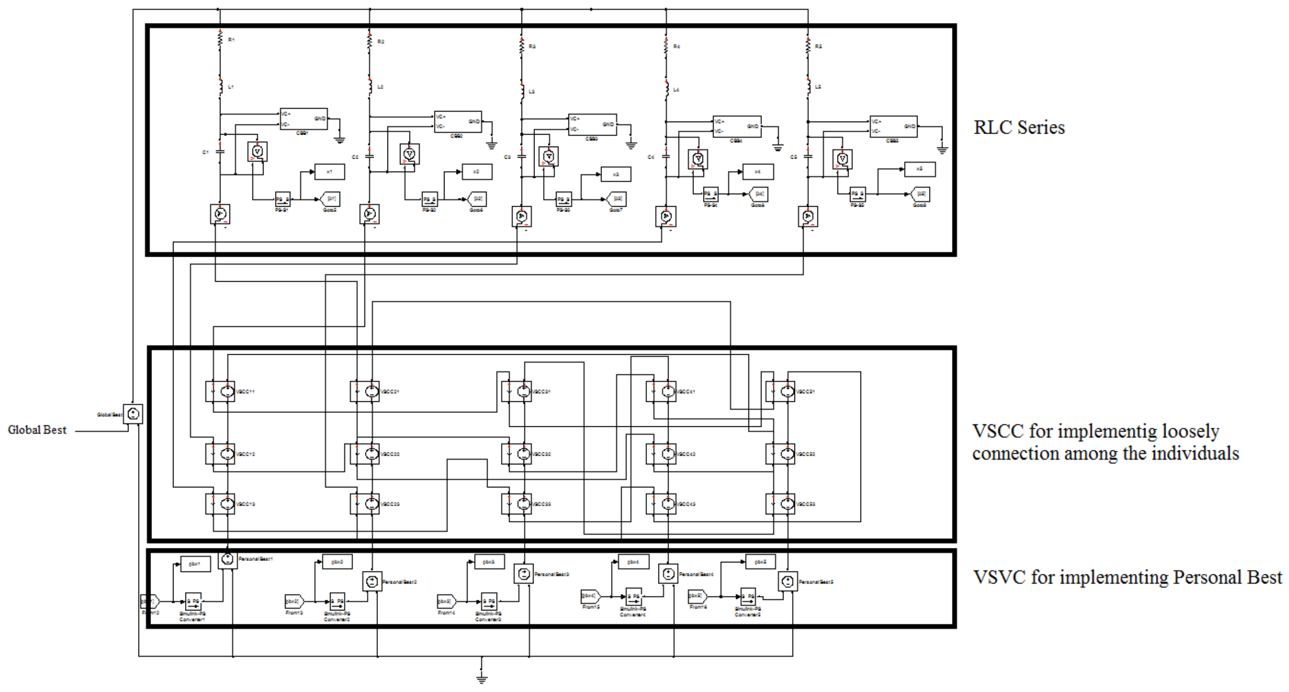

RLC Series
VSCC for implementig loosely connection among the individuals
VSVC for implementing Personal Best
Global Best

Figure 13: Implementazione 1D dell'FSO circuit in cui ogni individuo segue tre individui del gruppo.

Validazione su funzioni benchmarks di ottimizzazione e su problemi inversi

In questa sezione vengono brevemente illustrati i risultati della validazione del funzionamento dello Swarm circuit. Il circuito e tutte le traiettorie illustrate alla fine della sezione precedente sono state ottenute implementando il modello circuitale che viene descritto dalle equazioni trattate tramite Simulink (Figura 13). Per validare il circuito come ottimizzatore, sono stati svolti dei test su dei benchmark noti. In Tabella 11 vengono riportati i risultati che sono stati ottenuti. Il valore dei parametri utilizzati sono: $L=1\,H$, $R=1\,\Omega$, $C=1\,F$ mentre α è stato scelto in modo randomico: $1\,\Omega$, $0.25\,\Omega$ e $1.1\,\Omega$. Questa politica permette di cambiare il comportamento dell'algoritmo in modo dinamico durante il processo di ottimizzazione e di alternare i comportamenti di convergenza ed esplorazione. Chiaramente questa è solo una delle possibili politiche di ricerca che si possono adottare. I risultati ottenuti sono stati ottimi e sono mostrati in tabella 11.

Tabella 11: risultati sui benchmark relativi allo swarm circuit.

Name	Minimum value	Minimum coordinates circuit	Minimum value circuit
Styblinski-Tang	-78.332331	(-2.902435;-2.904137)	-78.332304
Bird	-106.764537	(4.741049;3.199581) (-1.599922;-3.117082)	-106.250957 -106.696935
Branins	0.397887	(-3.226006;12.603203) (3.145650;2.257012) (9.385460;2.523816)	0.398186 0.447557 0.411996
Six-hump camel back	–1.0316	(-0.0870;0.7354) (0.0877;-0.711)	-1.0271 -1.0316
Michalewics Function	-1.8013	(2.2436;1.5654)	-1.7731
Goldstein-Price function	3	(-0.0053; -1.0042)	3.0099

Lo swarm-circuit è stato anche testato su problem inverse ed in particolare sull'identificazione dei parametric di due sistemi dinamici: il Brussellatore e il Diodo Tunnel [14]. I risultati presentati fanno riferimento a 30 lanci con inizializzazioni randomiche delle posizioni e delle velocità.

Le equazioni differenziali che governano il sistema dinamico detto Brussellatore sono le seguenti:

$$\begin{cases} \dfrac{dy_1}{dt} = A + y_1^2 \cdot y_2 - \left(B+1\right) y_1 \\ \dfrac{dy_2}{dt} = B \cdot y_1 - y_1^2 \cdot y_2 \end{cases} \tag{30}$$

Mentre quelle relative al Diodo Tunnel:

$$\begin{cases} \dfrac{dy_1}{dt} = y_2 \\ \dfrac{dy_2}{dt} = -y_1 + y_2\left(1.4 - p \cdot y_2^2\right) + 4 \cdot a \cdot \left(-0.1649 y_1 + 0.4606\right) \end{cases} \tag{31}$$

Il numero di iterazioni è stato fissato a 2500 per entrambi i casi. I risultati ottenuti sono riportati in tabella 12 dove viene riportato il valore del MAPE (Mean Percentage Absolute Error). Lo swarm-circuit mostra lo stesso comportamento dell'algoritmo numerico. Nella figura 14 invece sono riportate le curve della soluzione trovata dallo swarm-circuit e quelle originali.

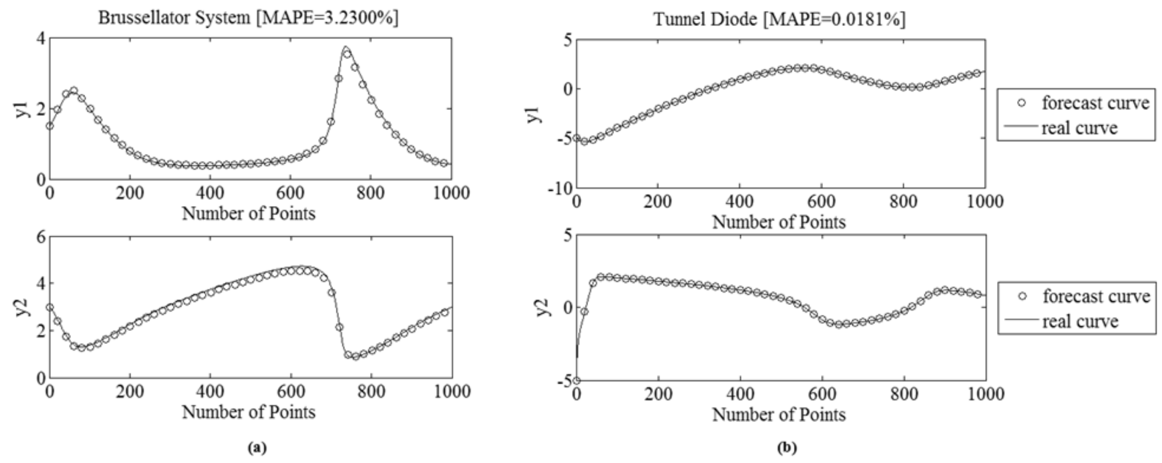


Figura 14: (a) migliore curva ottenuta dallo swarm-circuit per il sistema Brussellatore (20 s di simulazione); (b) migliore curva ottenuta dallo swarm-circuit per il sistema Tunnel Diode (4.5s di simulazione)

Table 12. Risultati della validazione dello Swarm-Circuit sui problemi inversi Brussellatore e Diodo Tunnel

	MAPE					
	Mean	σ	**Initial Value**	**Real parameters**	**Best forecast parameters**	**Mape best**
Brussellator	12.3417	13.6110	[1.5 3]	$[A,B]=[1,3]$	[0.9902,2.9016]	3.3003%
Tunnel	1.0414	1.5094	[-5 -5]	$[p,a]=[1.4,1]$	[1.0097,1.4111]	0.0181%

Bibliografia

[1]F. Riganti Fulginei, A. Salvini, G. Pulcini. “Metric-topological-evolutionary optimization.” Inverse problems in science & engineering (IPSE), ISSN: 1741-5977

[2] M. Clerc, J. Kennedy “The particle swarm-explosion, stability, and convergence in a multidimensional complex space”, IEEE Transactions on Evolutionary computation, Vol.6(1), 2002, pp.58–73.

[3] F.R. Fulginei and A. Salvini, Hysteresis model identification by the flock-of-starlings optimization,Int. J. Appl. Electromagnet. Mech. 30(3–4) (2009), pp. 321–331.

[4] F.R. Fulginei and A. Salvini, Influence of topological rules into the collective behavior of swarm intelligence: The flock of starling optimization, Stud. Comput. Intell. 327/2011 (2011), pp. 129–145.

[5] M. Ballerini, N. Cabibbo, R. Candelier, A. Cavagna, E. Cisbani, I. Giardina, V. Lecomte, A.Orlandi, G. Parisi, A. Procaccini, M. Viale, and V. Zdravkovic, Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study, Proc. Natl Acad. Sci. 105 (2008), pp. 1232–1237.

[6] S. Coco et al., “Optimization of multistage depressed collectors by using FEM and METEO,” presented at the IV Italian Workshop The Finite Element Method Applied to Electrical and Information Engineering,Rome, Dec. 13–15, 2010.

[7] Salvatore Coco, Antonino Laudani, Giuseppe Pulcini, Francesco Riganti Fulginei, and Alessandro Salvini “Shape Optimization of Multistage Depressed Collectors by Parallel Evolutionary Algorithm” IEEE Transaction on Magnetics, Vol. 48, No. 2, February 2012 435

[8] Salvatore Coco, Antonino Laudani, Giuseppe Pollicino, Giuseppe Pulcini, Francesco Riganti Fulginei, Alessandro Salvini “TWT magnetic focusing structure optimization by parallel evolutionary algorithm”, COMPEL Emerald Vol. 31 Iss: 5 pp. 1338 – 1346, 2012

[9]Al Salameh, M. S. H. and Hassouna M. A. S. (2010), "Arranging overhead power transmission line conductors using swarm intelligence technique to minimize electromagnetic fields", Progress In Electromagnetics Research B, Vol. 26, 213-236

[10]Canova, A. and Giaccone, L. (2009), "Magnetic field mitigation of power cable by high magnetic coupling passive loop", 20th International Conference and Exhibition on Electricity Distribution, 1-4, Prague, Czech Republic, Jun. 2009.

[11] M. Clerc, J. Kennedy “The particle swarm-explosion, stability, and convergence in a multidimensional complex space”, IEEE Transactions on Evolutionary computation, Vol.6(1), 2002, pp.58–73.

[12] J.L. Fernández-Martínez, E. García-Gonzalo and J.P. Fernández-Alvarez, “Theoretical analysis of particle swarm trajectories through a mechanical analogy[J]”, International Journal of Computational Intelligence Research, vol.4(2), 2008, pp.93

[13] S. M. Mikki and A. Kishk, “Physical theory for particle swarm optimization,” Progr. in Electromagn. Res., vol. 75, pp. 171–207, 2007.

[14]K. Schittkowski, Report, Department of Computer Science, University of Bayreuth (2004): http://www.ai7.uni-bayreuth.de/mc_ode.htm.

[15] Gordon G. Lai, Chien-Feng Yang, Hunter M. Huang, and Ching-Tzong Su, "Optimal Connection of Power Transmission Lines With Underground Power Cables to Minimize Magnetic Flux Density Using Genetic Algorithms" IEEE Transaction on Power Delivery, vol. 23, no. 3, JULY 2008

Chapter 1

Introduction to the Optimization and the Inverse Problems.

The basics of the arguments treated in this thesis are introduced.
An overview of optimizations and inverse problems together with the with a description of well and ill posed problems. Then, Meta Heuristic Optimization Algorithms will be discussed and in particular, those bio-inspired such as Particle Swarm Optimization and its variants; handling the emergence of social behaviour among the individuals which compose the swarm.

1.1 Optimization Problems

In the real world and in practical industrial applications, the most important task is to reach a desired goal with the maximum efficiency and not just simply to reach it. This way of thinking is strongly connected with the optimization practice. Many examples could arise in the reader's mind about all those activities in which the economic profit has to be increased. But a optimization process is tested every day by each of us, when you decide the shortest (we may say "optimal") path to go to work, or for instance when you try to combine your appointment in your agenda in order to match your travel needs etc. The first step is to model a system, identifying some characteristics of it, and assigning them variables; then identify an index of the system's performance usually called the objective that can be represented with a function. The modeling process plays a central role, because the optimization problem will depend on it. Often it is very difficult to identify which physical quantity is important or which is only noise. However, once the model has been formulated the goal is to find the optimal values of the variables that identifying the model in order to optimize the objective. Afterwards, an optimization algorithm can be employed for solving problems. Over the years several optimization algorithms have been designed. A quick classification of them can be made as listed in [1](Figure 1.1). The first distinction is between deterministic and probabilistic algorithm. An algorithm can be defined deterministic if it always yields the same results (outputs) for the same inputs. They represent a good tool for solving optimization problems, but whether there are many non linear relations among the model's variables and the objective, find the optimal solution, or just a candidate solution, could became very difficult especially if the search space is wide. The probabilistic algorithm is an algorithm where the result and/or the way by which the result is obtained depends on chance. They do not ensure the correctness of the solution, but in some cases a solution, even though it is not the best solution, is better than none. A mathematical formulation of an optimization problem can be given in its standard form as listed follow:

$$\begin{aligned} \text{minimize} \quad & f_O(x) \\ \text{subject to} \quad & f_i(x) \le 0, \ \ i = 1, \dots, m \\ & \mathrm{h}_i(x) = 0 \ \ \ i = 1, \dots, p \end{aligned} \tag{1.1}$$

Where the symbols have the following meaning:

- $x \in \mathbb{R}^n$ is s the optimization variable;

- $f_o : \mathbb{R}^n \rightarrow \mathbb{R}$ is the objective or cost function;
- $f_i : \mathbb{R}^n \rightarrow \mathbb{R}, i = 1, \ldots, m$ are the inequality constraint functions
- $h_i : \mathbb{R}^n \rightarrow \mathbb{R}$ are the equality constraint functions.

In some applications the variables have to take only integer value, this kind of problems are usually called *Discrete Optimization Problem* (DOP). In DOP there is an intrinsic constrain, because $x_i \in \mathbb{Z}$; or else they can be represented by mean of binary formulation. When the variables are uncountable, for example defined in $\mathbb{R}$, then the problem is defined as *Continuous Optimization Problems*.

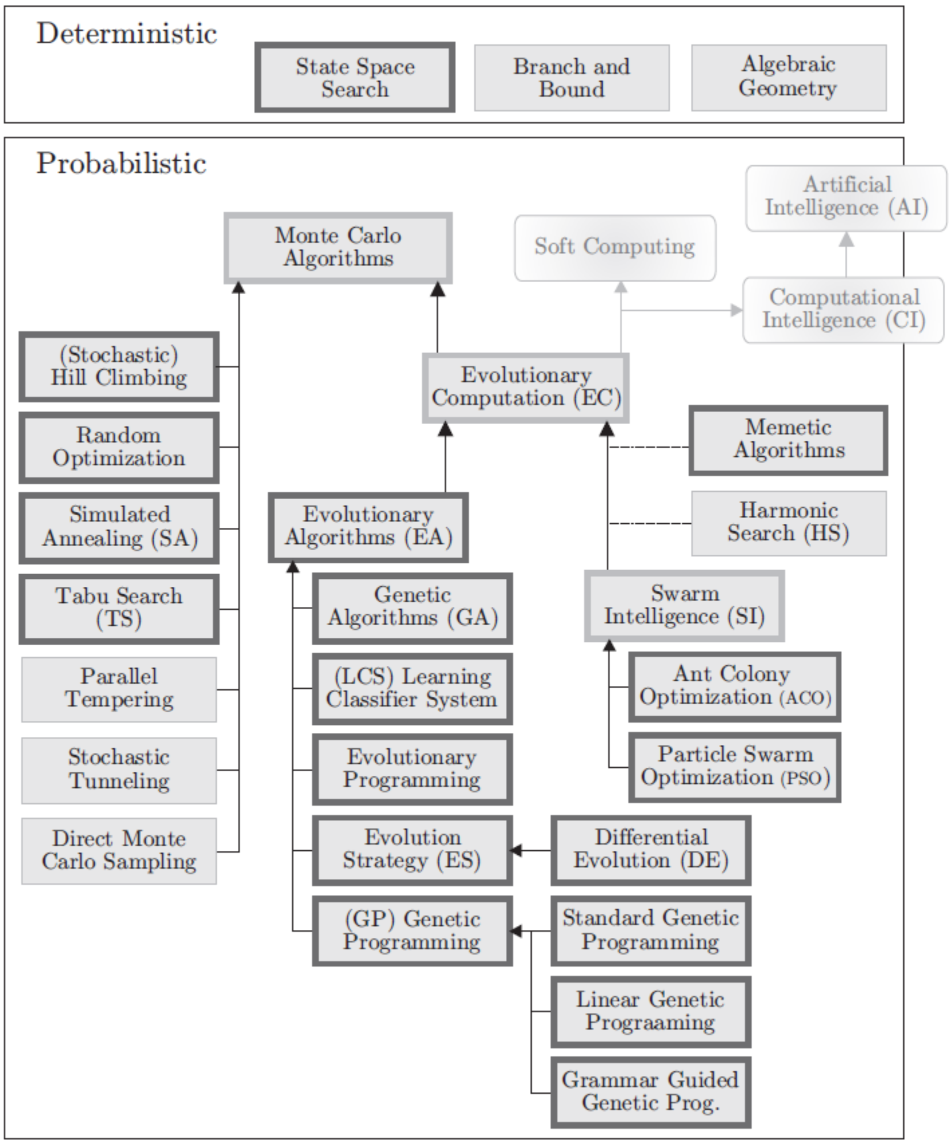

Deterministic
State Space Search
Branch and Bound
Algebraic Geometry
Probabilistic
Artificial Intelligence (AI)
Monte Carlo Algorithms
Soft Computing
Computational Intelligence (CI)
(Stochastic) Hill Climbing
Evolutionary Computation (EC)
Random Optimization
Memetic Algorithms
Simulated Annealing (SA)
Evolutionary Algorithms (EA)
Harmonic Search (HS)
Tabu Search (TS)
Genetic Algorithms (GA)
Swarm Intelligence (SI)
Parallel Tempering
(LCS) Learning Classifier System
Ant Colony Optimization (ACO)
Stochastic Tunneling
Evolutionary Programming
Particle Swarm Optimization (PSO)
Direct Monte Carlo Sampling
Evolution Strategy (ES)
Differential Evolution (DE)
(GP) Genetic Programming
Standard Genetic Programming
Linear Genetic Prograaming
Grammar Guided Genetic Prog.

Figure 1.1: The taxonomy of the optimization algorithm (image showed in the document [1]).

A third classification can be provided in the case in which there both discrete and continuous variable are present, such type of problems are called *Mixed-Integer Problem*. However, different approaches have been developed over the past years, and each of them owns peculiar characteristics depending on whether the problem addressed is DOP, COP or Mixed. Perhaps, the most famous discrete problems is "the Salesman Problem", and searching through the scientific works it is possible to find a huge quantities of algorithms for addressing the Salesman Problem. It is important to point out that in the DOP the constrain information, the objective, also called fitness, may change in a significantly. In the COP, the function's smoothness gives crucial information to find the optimal solution. A function f having a single object is defined with the term "single-object problem", otherwise if there is more than one object to reach, the optimization problem becomes a "multi-object problem". In the rest of this thesis, we will refer to some common concepts concerning optimization practice. The first concept is the word "optimal". What does an optimal solution mean about an optimization problem? For a function that represents a model of a real application, it is the maximum or the minimum value that it has. If we want to maximize a function f we simply consider $-f$. To explain the other concepts we will refer to an example of function showed in Figure 1.2, defined in the two-dimensional space.

Definition 3: A local Maximum $\bar{x} \in X$ of one (objective) function $f: X \to \mathbb{R}$ is an input element with $f(\bar{x}) \geq f(x) \forall x \in X, \varepsilon > 0, |x - \bar{x}| < \varepsilon$.

Definition 4: A local Minimum $\bar{x} \in X$ of one (objective) function $f: X \to \mathbb{R}$ is an input element with $f(\bar{x}) \leq f(x) \forall x \in X, \varepsilon > 0, |x - \bar{x}| < \varepsilon$.

Definition 5: A Global Maximum $\bar{x} \in X$ of one (objective) function $f: X \to \mathbb{R}$ is an input element with $f(\bar{x}) \geq f(x) \forall x \in X$.

Definition 6: A Global Minimum $\bar{x} \in X$ of one (objective) function $f: X \to \mathbb{R}$ is an input element with $f(\bar{x}) \leq f(x) \forall x \in X$.

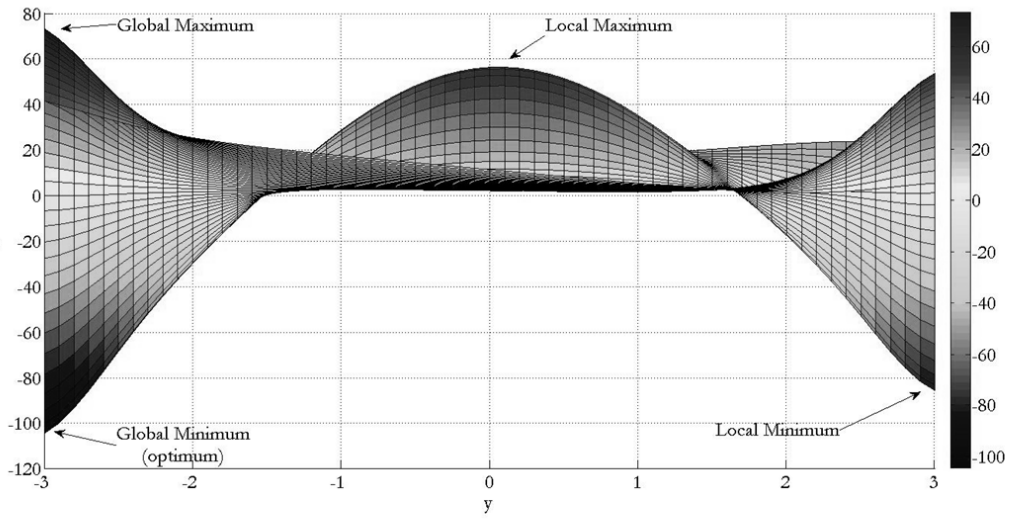


Figure 1.2: yz-plane of a two-dimension bird function.

In the optimization contest we may be faced with problems that can be solved with two different set of variables that produce the same measure, or else the same value. This problem is called multi-modal problems; they present two or more equivalent optimal solutions. By contrast, a problem that has one optimal solution is defined mono-modal problem.

1.2 Non Linear Inverse Problems

Inverse problems arise whenever one searches for causes of observed or desired effects. Usually there are two problems separated into direct and inverse; they are respectively inverse of each other if the solution of one involves the solution of the other one. We can refer to an example that I like to call the Lemonade example. Let me explain it. Let's say I wanted to get lemonade, I should prepare the ingredients: sugar, water and lemon juice. Well, I then shake them until I obtain a lemonade. This is the direct problem. Vice versa, the inverse problem consists, starting from the taste, of quantifying the three ingredients in order to obtain exactly that taste. In a scientific perspective we are usually faced with an inverse problem when we encounter partial differential equations. The direct problem is to solve the differential equations system so as to find the solution that predicts the system behavior, starting from the initial conditions and the system's parameters. The inverse problem consists of estimating the parameters starting from the observation set of data. In many engineering processes the physical law that governs them is often known and also the parameters involved in the model, whilst the actual parameters, for any particular cases, is unknown. Thus, solving an inverse problem means finding out just those values. The inverse problems are closer related with the definition of ill-posed problem. Hence, it will be useful to recall some important concepts that start from some studies made by Hadamard at the beginning of the last century. Hadamard was a famous French mathematician who founded out the definition of both well and ill posed problem. A problem of mathematical physics or a boundary value problem for a partial differential equation is called well-posed if the following conditions are satisfied: 1) a solution of the problem exists; 2) the solution of the problem is unique; 3) the solution of the problem depends continuously on the data of the problem. If one of these properties is violated, the problem is called ill-posed. Neither existence nor uniqueness of a solution to an inverse problem is guaranteed. Many different regularization methods have been developed for solving both linear and no linear inverse problems[3][4]. In this work it will be exposed the utilization of metaheuristic algorithms that in the last fifteen years have been applied in such mathematical field with optimal results. In the next paragraph we will introduce metaheuristic algorithms pointing out their usefulness and even their main shortcomings. Notice, that an inverse problem can be viewed as an optimization problem, because we can build an error function to minimize it, and thus, generally speaking, optimize it. The objective of such optimization is the minimization of the error function that describes how far we are from the good representation of the system's state and from the optimal estimation of parameters composed by it.

1.3 Metaheuristic Algorithms

Metaheuristics are innovative algorithms designed for addressing complex optimization problems, in which other optimization algorithm, often deterministic algorithms, have already failed. Normally, for addressing an optimization problem, a custom heuristic is made so as to adapt the resolution strategies of the current problem. This required a new approach to every problem and lessons learned from one problem did not always generalize well to a different class of problems. Instead, Metaheuristics requires less work to develop and adapts the algorithm to the specific problem. Those benefits allow the extension of the field of applicability of metaheuristics. In general, many metaeuristics start by an initial solution or an initial set of solutions, and then begin an improving search, guided by certain principles. It is important to point out in this contest that the prefix *meta* indicates that the detail of the search are not specified, but only a general strategy is described by a metaheuristic algorithm. For mentioned reasons they do not guarantee to find the optimal solution in bounded time. Despite the fact that they do not give any guarantee, they are applied when problems have large size, and when the goal is to quickly find a (near-)optimal solution. The main standards in the search and optimization process are exploration and exploitation, also considered as diversification and intensification; in particular, when we say intensification, we mean concentrating the search in a confined, small search space area, whilst diversification is when we explore the search space.

These two points of view are contrary and complementary; we are going to dedicate many pages to these peculiar aspects of the metaheuristic algorithms in the rest of the work.
A metaheuristic is usually non-deterministic and allow us to address a optimization problem with an high abstract level of description of it. For instance, obviously, one does not always know the close mathematic form of the model, and hence, you can consider it like a black-box. In this way the operator avoids to be obligated to define the exact model of the problem. He only has to be able to extract the value of the performance from the model. This is the only prerequisite for utilize a metaheuristic for solving a optimization problem. When, for example, we are solving an identification parameter problem (so an inverse problem), we know the model of the physical system, but we do not know the model of the "error function" that after the minimization allow us to get the optimal parameter for that problem.
There are many classifications in literature, but in the author's opinion the most simple and fast classification is in two wide categories: Trajectory method and Population-based method. In the Trajectory methods there are a set of individuals that do a trajectories in the search space following a determined rule. For example, the Particle Swarm Optimization and the Flock of Starlings optimization belong to this category, which will be exhaustively treated in this work. The second category is the Population-based methods, in which we can enumerate Evolutionary Algorithms such as Genetic Algorithm, Differential Evolutions Algorithm, Ant Colony Optimization etc.; they can be considered as a set of solutions that evolves over various epochs. Recently, the tendency is to make hybridization among the metaeuristics in order to improve the search process and overcoming the shortcomings of one with the benefits of another one. In the next few chapters this aspect will be treated exhaustively.

References

[1] Thomas Weise "Global Optimization Algorithms – Theory and Application –", Ebook http://www.it-weise.de/

[2] Melnikov, B.; "Discrete optimization problems some new heuristic approaches",High-Performance Computing in Asia-Pacific Region, 2005. Proceedings. Eighth International Conference on Digital Object Identifier: 10.1109/HPCASIA.2005.34 Publication Year: 2005 , Page(s): 8 pp. – 82

[3] Inverse Problems: Computational Methods and Emerging Applications September 8 - December 12, 2003

[4] Ill-Posed Problems of Mathematical Physics and Analysis M. M. Lavrent'ev, V. G. Romanov, and S. P. Shishatskiĭ1986; 290 pp; thrid printing 2000 ISBN-10: 0-8218-0896-6

Chapter 2

MeTEO Metric & Topological Evolutionary Optimization.

In this chapter we will discuss a new hybrid metaheuristic algorithm called MeTEO [14], that represents a solid instruments for hard optimization problems. The algorithm exploits three heuristics working in synergy on a parallel architecture in order to reach the best global minimum, or in a multimodal cases the best set solution. The idea of a new algorithm starting from 3 famous algorithms is an interesting experience since it is possible to observe how simple algorithmic rules can show the emergence of complexity when they are used in a co-operative scenario; as simple bricks are able to produce cathedrals.

2.1 Global framework of MeTEO

Exploration and convergence are two important requirements for the algorithms which are devoted to inverse-problems and/or optimization. In many applications it is usual to prefer those algorithms showing better capabilities of convergence to the global optimum. On the other hand, we have to consider that: a) many optimization problems require finding more than one single global optimum; b) in many cases the global optimum must be found within a solution space having very high dimensions [1]. Frequently, the previous points a) and b) are simultaneously present. Thus, the designer of the algorithms must decide if it is better to expand the search or to privilege the convergence on a subspace. This is very hard to be decided a-priori [2]. In fact, the risk to be entrapped into a local minimum is higher if an algorithm does not explore a suitably wide space. On the other hand, an algorithm could make an over sampling of the space and spend a long time before achieving convergence to a good solution [2]. This chapter shows a novel approach based on Evolutionary Computation able to enhance exploration preserving convergence. The proposed algorithm has been called MeTEO to highlight its Metric-Topological and Evolutionary inspiration. In fact, it is based on a hybridization of two heuristics coming from swarm intelligence: the Flock-of-Starlings Optimization (FSO) (topological swarm), the standard Particle Swarm Optimization (metric swarm) and a third evolutionary heuristic: the Bacterial Chemotaxis Algorithm (BCA) that has non collective behavior. In particular, the Flock-of-Starlings Optimization (FSO) was first described and applied in [3] and [4] as a modification of the well-known Particle Swarm Optimization (PSO) [5] by adding topological rules to the metric rules that are typical of the PSO. The FSO is inspired by recent naturalistic observation about the real starling flight [5]. As it is shown in [3] and [4], the FSO is particularly suitable for exploration and multimodal analysis. The BCA [6] is based on the emulation of the motion of a real bacterium looking for food (i.e. fitness function). It is a heuristic that shows its better performances in local search [1]. The present approach uses the FSO to explore the solution space, the PSO to investigate subspaces in which the global optimum could be present whereas the BCA is at the end used for refining solutions. A parallel strategy is implemented: the FSO is permanently running. Any time it finds a possible solution, the PSO is launched. After several iterations, the PSO is finally substituted by BCA, and so on. A final important computational strategy completes the present approach: the fitness function is made worse solely in those suitable narrow regions in which the FSO has decided to launch PSO-BCA. This Fitness Modification (FM) has the aim to prevent FSO from going back to an already explored subspace. MeTEO has been tested on the main optimization benchmarks currently used in literature but also novel harder benchmarks are presented. However, the following sections will be dedicated to the complete explanation of these MeTEO features.

2.2 Swarm Algorithms, Particle Swarm Optimization and Flock-of-Starlings Optimization.

The Particle Swarm Optimization (PSO) is one of the most used and studied algorithms among the optimization methods. It was proposed by James Kennedy and Russell Eberhart in 1995 [7], starting from the works of Reynolds [8] and Heppner and Grenander [9]. It is an algorithm based on some metric rules applied for simulating the collective behaviour of swarms. The metric approach consists in describing the behaviour of a flying bird which must keep its distance from its neighbours that is forced to stay into a fixed interaction range, i.e., all birds are keeping alignments of velocity among all flock members. Although the metric rule allowed the simulation of a collective movement of animals, significant differences still remained compared to the behaviour of a real flock. However, on the basis of this paradigm, Kennedy and Eberhart [1] first proposed their approach by establishing a one-to-one relationship between the motion of a flock governed by metric rules during its food searching and the iterative steps of a numerical algorithm searching the solution for optimization problems. As a second step, they found that some metric rules of the paradigm were an obstacle for multi-objective optimization tasks. Thus, they modified the algorithm by removing some parts of it. For example, matching the velocity of the nearest neighbour was removed, and so on. These variations have altered the virtual collective movement of the swarm and the final algorithm simulates a behaviour more similar to a swarm of insects than to a flock of birds. Thus, we can say that the original PSO has been created starting with the simulation of a flock of birds and arriving at the emulation of a swarm of generic "particles". The introduction of topological rules into PSO is the hub of the algorithm called Flock-of-starlings Optimization (FSO). The FSO, [3],[4] adopts an approach based on recent naturalistic observations [5], on the collective behaviour of the European Starlings (Sturnus Vulgaris). The authors of the paper [5] have discovered an interaction among members of the same flock which has topological nature: it is relevant how many intermediate birds separate two starlings, not how far apart they are the one from the other. This means that the main property of the topological interaction is that each starling interacts with a fixed number of neighbours, i.e. their metric distance is not crucial. Thus, real flocks of starlings have a behaviour that can be numerically simulated by using topological rules rather than metric rules. In fact, the topological approach is able to describe the density changes that are typical of flocks of birds, whereas the metric approach is not able to do it. In real flocks each generic k-th bird controls and follows the flights of a generic number of other members of the flock, no matter which are their positions inside the flock.

The pseudo-codes of the implemented PSO and FSO referred to a generic function, $f(x_1.....x_D)$, to be minimized in the search space having dimension $\mathbb{R}^D$ are recalled in the following (Figure 2.1) and (Figure 2.2).

In particular in (Figure 2.1), we have indicated into the boxes numbered from 1 to 11 the definition of the main common parameters to be used both by the PSO and FSO algorithms.

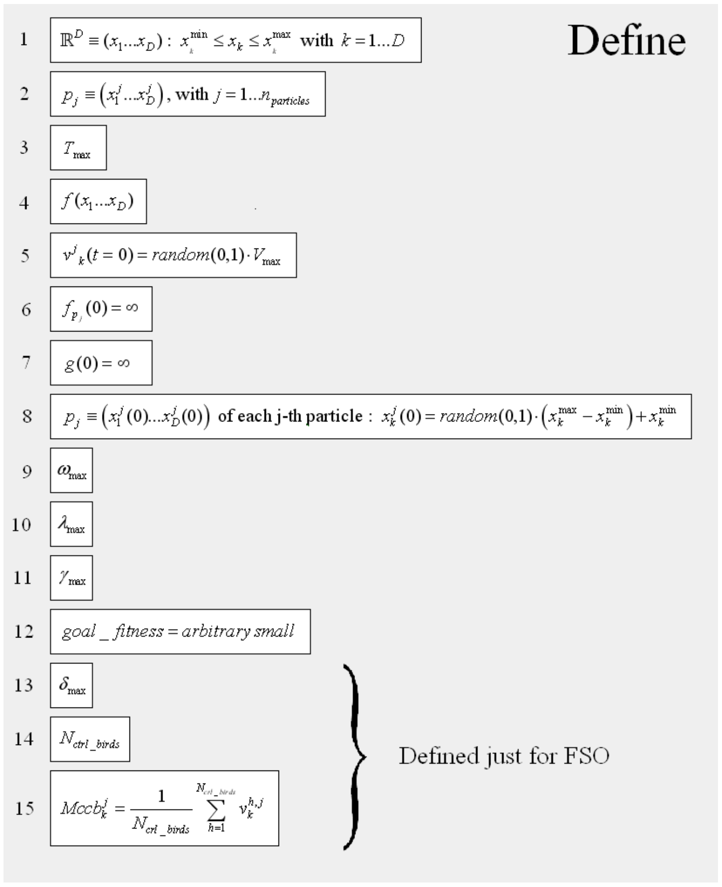
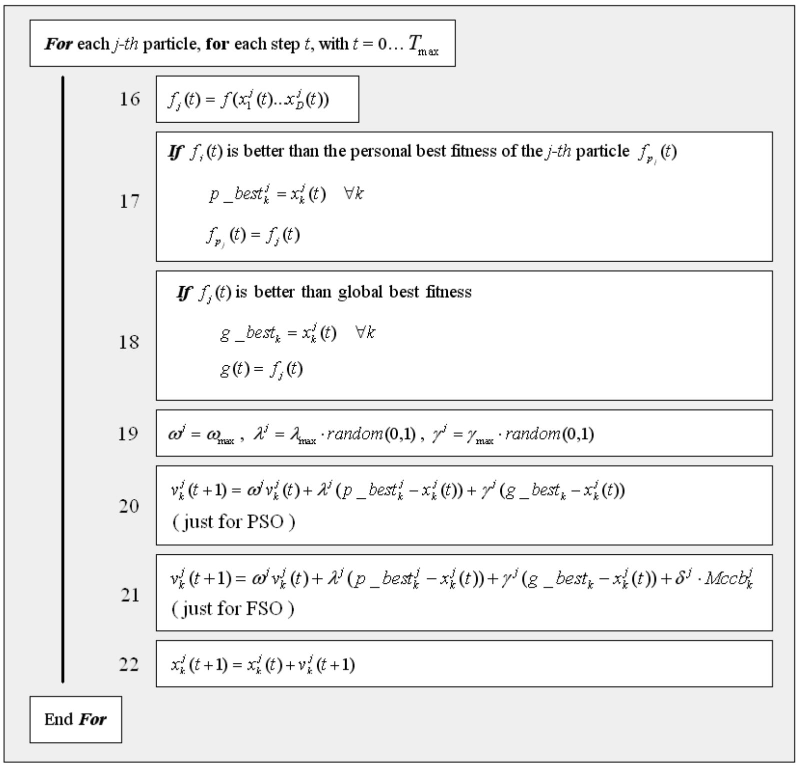

Define
1 $\mathbb{R}^D \equiv (x_1...x_D) : x_k^{\min} \le x_k \le x_k^{\max}$ with $k = 1...D$
2 $p_j \equiv \left(x_1^j...x_D^j\right)$, with $j = 1...n_{particles}$
3 $T_{\max}$
4 $f(x_1...x_D)$
5 $v^j{}_k(t=0) = random(0,1) \cdot V_{\max}$
6 $f_{p_j}(0) = \infty$
7 $g(0) = \infty$
8 $p_j \equiv \left(x_1^j(0)...x_D^j(0)\right)$ of each j-th particle : $x_k^j(0) = random(0,1) \cdot \left(x_k^{\max} - x_k^{\min}\right) + x_k^{\min}$
9 $\omega_{\max}$
10 $\lambda_{\max}$
11 $\gamma_{\max}$
12 $goal\_fitness = arbitrary\ small$
13 $\delta_{\max}$
14 N_{ctrl_birds}
15 $Mccb_k^j = \frac{1}{N_{crl\_birds}} \sum_{h=1}^{N_{crl\_birds}} v_k^{h,j}$
Defined just for FSO

Figure 2.1: Main parameters managed by PSO and FSO.

They are: the dimension of the search space, $\mathbb{R}^D \equiv (x_1.....x_D)$, the values of the main parameters for each *j-th* particle $p_j \equiv \left(x_1^j...x_D^j\right)$ $p_j = (x_1^j.....x_D^j)$ with $j = 1...n_{particles}$; the maximum number of iterations, $T_{\max}$; the fitness function $f(x_1.....x_D)$; the maximum value of each velocity component $V_{\max}$; the initialization of velocities $v_k^j(t=0) = random(0,1) \cdot V_{\max}$ (where $random(0,1)$ is a function returning a random value between 0 and 1); the personal fitness

$f_{p_j}(0)=\infty$ (∞ indicates an arbitrarily large value); the global fitness $g(0)=\infty$; the position $p_j=\left(x_1^j(0)...x_D^j(0)\right)$ of each *j-th* particle with $x_k^j(0)=random(0,1)\cdot\left(x_k^{\max}-x_k^{\min}\right)+x_k^{\min}$; the value of the inertial coefficient ω^j;

the maximum value of cognize coefficient, $\lambda_{\max}$; the maximum value of social coefficient, $\gamma_{\max}$; the fitness threshold $goal\_fitness = arbitrary\ small$. Moreover, in Figure 2.1, we have indicated the parameters used just for FSO,(i.e. they do not appear in the PSO algorithm) by the blocks numbered from #13 to #15: the maximum value of the topological coefficient, $\delta_{\max}$ among all topological coefficients $\delta^j=\delta_{\max}\cdot random(0,1)$; the quantity $Mccb_k^j=\frac{1}{N_{crl\_birds}}\sum_{h=1}^{N_{crl\_birds}} v_k^{h,j}$; the number N_{crl_birds} of the birds that are controlled by each other single bird within the flock. In (Figure 2.2), is reported the pseudo code valid both for PSO and for PSO.



```
For each j-th particle, for each step $t$, with $t = 0\ldots T_{\max}$
    16  $f_j(t)=f(x_1^j(t)..x_D^j(t))$
    17  If $f_i(t)$ is better than the personal best fitness of the j-th particle $f_{p_j}(t)$
            $p\_best_k^j=x_k^j(t)\quad\forall k$
            $f_{p_j}(t)=f_j(t)$
    18  If $f_j(t)$ is better than global best fitness
            $g\_best_k=x_k^j(t)\quad\forall k$
            $g(t)=f_j(t)$
    19  $\omega^j=\omega_{\max}$ , $\lambda^j=\lambda_{\max}\cdot random(0,1)$ , $\gamma^j=\gamma_{\max}\cdot random(0,1)$
    20  $v_k^j(t+1)=\omega^j v_k^j(t)+\lambda^j(p\_best_k^j-x_k^j(t))+\gamma^j(g\_best_k-x_k^j(t))$
        ( just for PSO )
    21  $v_k^j(t+1)=\omega^j v_k^j(t)+\lambda^j(p\_best_k^j-x_k^j(t))+\gamma^j(g\_best_k-x_k^j(t))+\delta^j\cdot Mccb_k^j$
        ( just for FSO )
    22  $x_k^j(t+1)=x_k^j(t)+v_k^j(t+1)$
End For
```

Figure 2.2: Pseudo code for PSO and FSO.

How it is evident from (Figure 2.2), the PSO and the FSO differ just for the expressions written in block #20 and #21 respectively. This apparently small difference produces huge differences in the behaviour of the two heuristics. In fact, even if the PSO shows quite a good exploration capability as well as a good degree of convergence, it has been noted that, especially when the searching space becomes larger and larger, it can be trapped into a local minimum without the possibility of escaping [1]. Furthermore, for multi-modal functions, the algorithm is not comprehensive [1]. In fact, it is important to note that the standard PSO has undergone many changes. Many authors have derived new PSO versions and published theoretical studies on the effects produced on the algorithm by changing the values of the various parameters (e.g., ω^j, λ^j, γ^j, and so on [10]). Other authors have investigated the effects produced by changing some aspects of the algorithm (see, for example, [11][12] and the references therein). It is important to note, however, that the cited papers focus their attention more on convergence rather than on exploration. On the other hand, in [3] and [4] it has been shown that FSO has high exploration capability, avoids entrapments into local minima and is particularly suitable for multimodal optimizations. Unfortunately FSO shows a lack of convergence. This is the cost to be paid for obtaining a high exploration capability for the algorithm. Practically the FSO does not stop running without user intervention. But this also is the reason that allows the FSO to operate a complete mapping of the solution space. This makes the FSO immune to the worst initializations, i.e. it can return a good solution regardless of how large the dimension of the solution space is. In fact, the FSO has the potential to find each subspace in which a minimum (local or global) lies. Although the FSO can or cannot find better solutions depending on the values assigned to the parameters appearing in (Figure 2.1), the PSO can never achieve the exploration capability of the FSO simply by means of a modification of its parameters. We can conclude these notes by saying that FSO is a very good explorer but it is not convergent, whereas PSO makes a balance between exploration and convergence, favouring (perhaps too much) convergence.

2.3 The Bacterial Chemotaxis Algorithm

The BCA has been proposed in [6] and is an algorithm based on the emulation of the motion of a real bacterium looking for food (i.e. fitness function). A mathematical description of a 2D bacterium motion can be developed by assuming an assigned speed v and by determination of suitable probabilistic distributions of the motion duration τ and of the direction φ shown by each individual. The 2D description can be easily extended to n-dimensional hyperspaces defining, for the virtual-bacterium path, a vector made of n positions x_i, with $i = 1,...,n$, and a vector made of $n-1$ directions φ_k, with $k = 1,...,n-1$. Thus, the virtual bacterium motion follows the rules proposed in [6] and here summarized in (Figure 2.3) and (Figure 2.4):

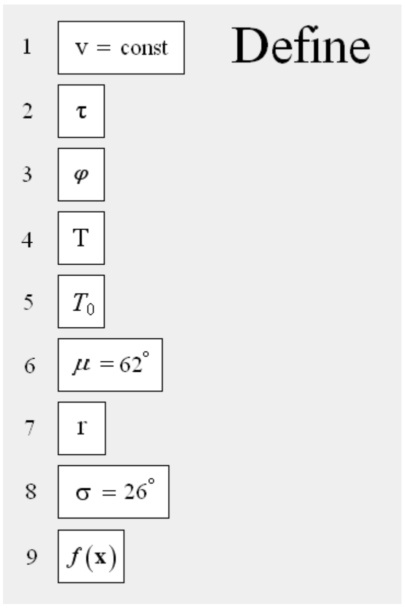


Figure 2.3. Main parameters managed by BCA.

In (Figure 2.3), we have indicated the two expectation values of a suitable exponential probability density functions with T and μ, the standard deviation with σ, the minimum mean time with T_0, the module of the difference between the new position vector, $\mathbf{x}^{NEW}$, and the old position vector, $\mathbf{x}^{OLD}$, of individuals with $r = \left|\mathbf{x}^{NEW} - \mathbf{x}^{OLD}\right|$ and the function cost to be minimized with $f(\mathbf{x})$; the parameters T_0, τ are usually called strategy parameters. Regarding directions, the probability density function describing the turning angle between two consecutive trajectories is Gaussian. (Figure 2.4) shows the flow-chart of BCA pseudo code. The path of each bacterium is composed by adding the vector $\mathbf{x}'$ to the old position of virtual bacteria, step by step, by the equation $\mathbf{x}^{NEW} = \mathbf{x}^{NEW} + \mathbf{x}'$, with $|\mathbf{x}'| = r$. Moreover, the b parameter is another strategy parameter and is introduced to modify the dynamic of individuals' movement. Finally, for each bacterium, $f_{pr} = f^{NEW} - f^{OLD}$ is the difference between the current fitness function value (f^{NEW}), referred to the current position (current algorithm step), and the value of that (f^{OLD}) referred to previous position (previous algorithm step). The BCA effectiveness is strongly influenced by the choice of the parameters: T_0, τ and b. They are usually obtained empirically depending on the typology of the optimization problems. In particular, the BCA convergence becomes difficult if the T_0 value is excessively large.

```
For each bacterium and for each step
    $T = \begin{cases} T_0 \quad \mathbf{r} & \text{per } \frac{f_{pr}}{\mathbf{r}} \geq 0 \\ T_0\left(1 + b\left|\frac{f_{pr}}{\mathbf{r}}\right|\right) & \text{per } \frac{f_{pr}}{\mathbf{r}} < 0 \end{cases}$
    $P(X = \tau) = \frac{1}{T} e^{-\tau/T}$
    $P(X_i = \varphi_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left[ -\frac{(\varphi_i - \mu_i)^2}{2\sigma_i^2} \right]$
    $r = v \cdot \tau$
    $\begin{cases} x'_1 = r \cdot \prod_{k=1}^{n-1} \cos(\varphi_k) \\ x'_i = r \cdot \sin(\varphi_{i-1}) \cdot \prod_{k=i}^{n-1} \cos(\varphi_k) \\ x'_n = r \cdot \sin(\varphi_{n-1}) \end{cases}$
    $\begin{cases} x_1^{\text{NEW}} = x_1^{\text{OLD}} + x'_1 \\ x_i^{\text{NEW}} = x_i^{\text{OLD}} + x'_i \\ x_n^{\text{NEW}} = x_n^{\text{OLD}} + x'_n \end{cases}$
End For
```

Figure 2.4: Pseudo code of BCA.

This is because T_0 is the shortest time interval of a single position change of the bacterium. This shortest movement should be small enough to allow the BCA to achieve the requested accuracy. On the other hand, if the elementary bacterium movement were too small, the BCA running-time would be excessive. Furthermore, in presence of a large gradient (i.e. the motion excitation for the bacterium), b should be chosen small enough to avoid the risk of a removal from the attraction zone.

2.4 Extension of the FSO to a MultiFlocks.

With the aim to improve the exploration capability of the FSO, a further variant has been ideated that has been called MultiFlock of Starling Optimization (MFSO). It consists of implementing more than one flock working at the same time following the rules next explained. In the MFSO, the interconnection matrix is divided into several sub matrixes having dimension equal to $N_{ind\_sub} \times N_{birds\_followed}$, where N_{ind_sub} is the number of individuals belonging each sub-flock. Each individual will belong to a sub-matrix (chunk) and it will be constrained to follow just the individuals with are members of the same chunk. In this way, any N_{sub} will have no trajectories related to those made by the other sub-flocks, i.e. they are fully uncoupled and the only exchange of information consists of the sharing of the global best. The capability of exploration of the MFSO, will be proved in the next sections.

2.5 Parallel architecture, Fitness Modification and the activation strategy of the different components of MeTEO.

2.5.1 Parallel architecture

MeTEO shows best performances on a distributed architecture, in fact, it has been fully designed for parallel computation based on a Master-Slaves configuration. According to the peculiarities of each previously described single heuristic (PSO, FSO and BCA), MeTEO uses FSO just on the Master node whereas PSO and BCA on Slave ones. The FSO performs the exploration of the whole space of solutions and whenever it finds a sub-region in which there is a high probability of discovering a global minimum (briefly called "suspected region"), two simultaneous operations are made: 1) the current fitness is modified with the aim to preventing FSO from exploring a found "suspected region" again; 2) MeTEO launches the PSO algorithm on a Slave node of the cluster, being the PSO population initialized by means of the best result found by FSO at the current iteration. Let us explain these points in more detail in the next few paragraphs.

2.5.2 Fitness Modification and Switching strategy among MeTEO components.

The Fitness modification (FM) is based on metric rules similar to those used by the famous Tabu-search algorithm [13], and in particular to the Tabu list. Since the global optimum is coincident with the smallest value achievable for a fitness function, the FM has to ensure that the algorithm does not perform investigations on those "suspected regions" already detected by the FSO. In fact, the FM consists of adding a positive Gaussian function, centered into the best co-ordinates found by FSO, to the past fitness function at the current iteration. A further control is made by MeTEO on the number of iterations for which the fitness does not improve its value. In other words, if the fitness does not improve for a number of FSO iterations equal to a fixed threashold, MeTEO assumes the coordinates of the last found global best as the centre of a "suspected region" and activates a Slave node in which PSO and BCA acting on. It is important to remark that the effect of the FM acts just on the PC-Master, i.e. it is valid just for FSO whereas the fitness function holds the original starting expression both for PSO and BCA working on the PC-Slave. In more detail, let us show how the FM acts when the FSO finds a generic k-th "suspected region". Before finding the generic $k-th$ a "suspected region", FSO was managing a fitness function that was fixed at the moment in which the FSO found the $(k-1)-th$ "suspected region": $f_{fitness_{k-1}}(\mathbf{x}): D \subset \mathbb{R}^n$ with $\mathbf{x}=(x_1 ... x_D)$, (it is clear that the starting original fitness will be indicated with $f_{fitness_0}(\mathbf{x})$). Let us assume $\mathbf{x}_{\min_k}$ to be a vector collecting the co-ordinates minimizing the value of $f_{fitness_{k-1}}(\mathbf{x})$. Then, FM operates according to the following equation:

$$f_{fitness_k}(\mathbf{x}) = f_{fitness_{k-1}}(\mathbf{x}) + A\exp\left(-\frac{\sum_{j=1}^{n}\left(x_j - x_{\min_j}\right)^2}{2\sigma^2}\right) \tag{2.1}$$

where A is a suitable constant and σ is the standard deviation that defines the size of the k-th "suspected region" detected by FSO. In this way, the new fitness $f_{fitness_k}(\mathbf{x})$, that will be managed by the FSO, will find a further "suspected region", and will no longer show a minimum in $\mathbf{x}_{\min_k}$, i.e. the FSO will no longer be attracted by that region. Obviously, in the Slave node no FM is made. Let us see now what happens in the Slave node. When a k-th "suspected region" is detected, a Slave node is activated. On this node the PSO is initialized by $\mathbf{x}_{\min_k}$ whereas the fitness function is always set equal to the starting one: $f_{fitness_0}(\mathbf{x})$. The PSO will be left to run for a prefixed number of iterations $N_{PSO-\max}$. When the PSO iterations are equal to $N_{PSO-\max}$ the PSO ends its task and it is substituted, on the same Slave node, by the BCA. The BCA population will be initialized by means of the best result achieved by PSO so far. The BCA plays the role of the local search. It searches the minimum until its own number of iterations is equal to a maximum number $N_{BCA-\max}$ that has been set before running MeTEO.

Meanwhile the process on a Slave node is running, FSO still continues to explore the space on the Master node. Any time FSO finds a further "suspected region", MeTEO launches a further process again on a different Slave node and so on. Any final result coming from Slave nodes is stored in a dedicated file that is located in a shared folder in the Master node. After to have delivered to the Master node the found solution, a Slave is ready to be used again for a further exploration of a new "suspected region". Finally, when all processes are finished and all PC-slaves have completed the PSO+BCA procedures, as previous described, a list of all detected minima is available in the file in which all results, coming from Slaves, have been stored. At the end, from this stored list the best minimum is trivially identified. Practically, this kind of parallelization can be defined *asynchronous.*

Finally let us to do some remarks on FSO stopping criterion. A simply criterion consists of

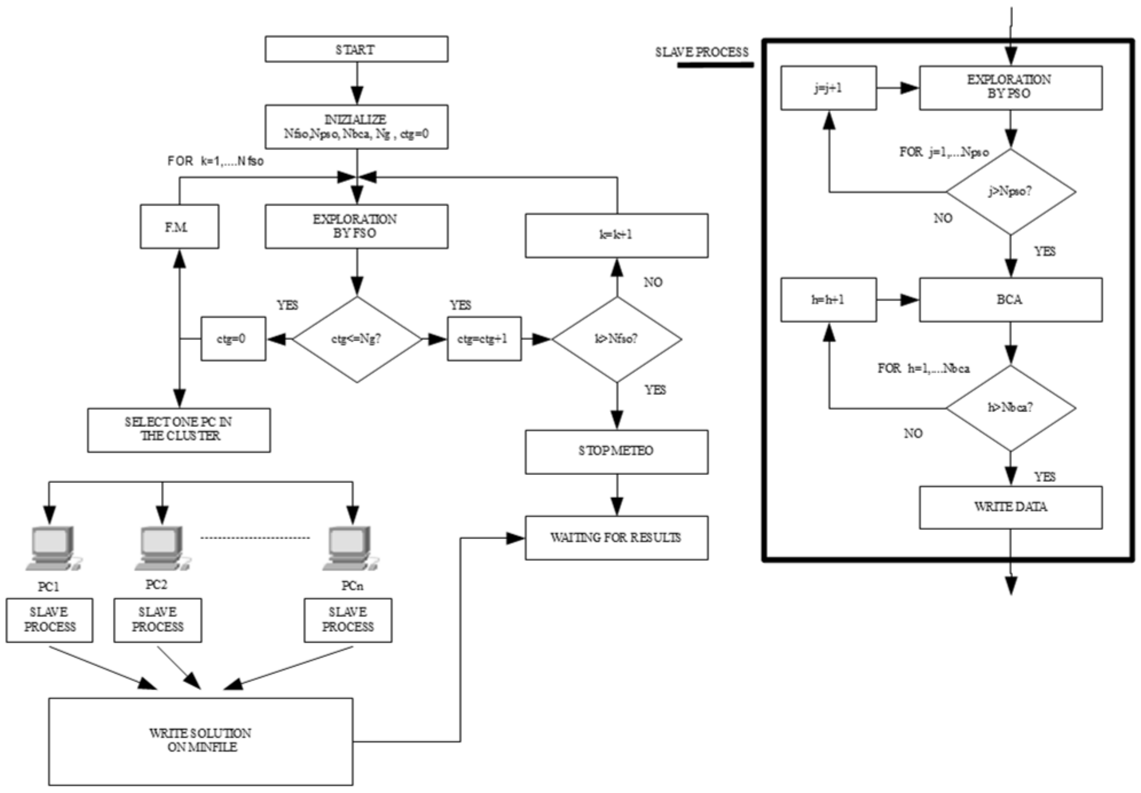


Figure 2.5: flow chart of the optimization process provided by MeTEO.

stopping METEO whenever the FSO performs a number of iterations $N_{FSO-\max}$ that have been set by the user. But this strategy may not be effective in all cases. In fact, some hard multi-modal optimizations, i.e. in those cases in which the FSO needs of a higher number of iterations for detecting all the global minima, it could be convenient to use a different stopping criterion based on a maximum number of “suspected regions”. In other cases it will be recommendable to use a Boolean OR logic gate based on both the two proposed criteria.

Obviously, the choice of a specific MeTEO stopping criterion depends on the user experience on the optimization problem or inverse problem to be solved. This approach is useful when the fitness function requires short computational time to be calculated. Indeed, if the fitness function requires a high computational effort, it is best to use a different parallel architecture, in which each slave-node computes the operation of one single individual, which is involved in the fitness calculus. The master refreshes only the global and personal bests after the calculation of all fitness for all individuals. In this approach the asynchronous parallelism between FSO and PSO present in the approach previously described now is substituted by a synchronous one. Now, the whole machines of the cluster are used first for the FSO and afterwards for PSO and BCA population in succession. For implementing the same behavior a cluster of cluster should be implemented. In the next third chapter, both the implementations will be described for facing different kinds of inverse problems.

Let us now to return to the fitness function modification. The three images depicted in Figure 2.6, refer respectively to the Giunta function [14]. Through these pictures we can better see the various steps of the FM application. The first image refers to the simple cost function without the FM. In the second one there is a first FM indicated by the black arrow. In the third figure the local minima becomes a local maximum, forcing FSO to escape from it. By choosing different weights for the FM, one can improve the performance in exploration. However, a parsimonious utilization of FM is recommendable, because if one choose a weight that is too high the function may be too much modified, in such way that the FM could mask also important regions in which the global, or one of the global bests, could lie. Then, a pre-process has to be carried out in order to choose the more suitable FM weight. An empirical method could be the iterative increasing of the weight, until the operator notices a detectable change in the fitness value while the optimization is running.

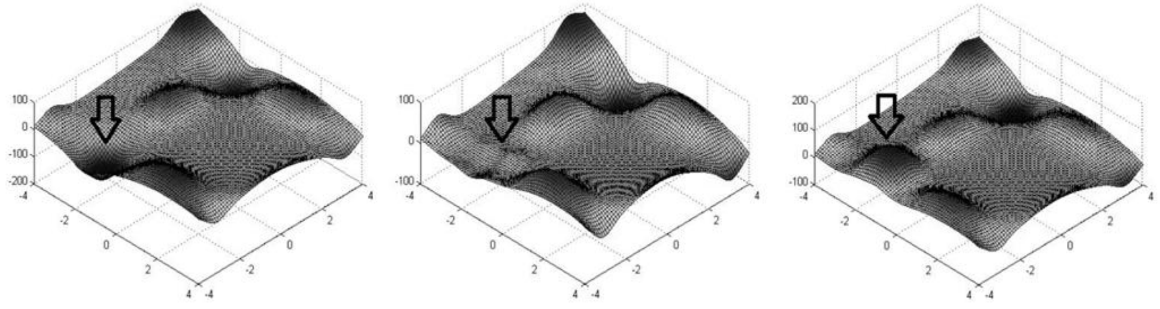


Figure 2.6: Changing in the fitness function Giunta after two launches of FM.

2.5.3 Parallel Architecture: the Algorithm-Based and the Fitness-Based.

All the methods presented above are oriented to a parallel computation that we could call "algorithm based", in this kind of parallel approach, each slave node is used for running one series PSO-BCA, hence, the task is calculated in a single machine without other parallelization, for instance working with the cores of each machine. The same philosophy is adopted in the Parallel computing Toolbox of Matlab®, that we have used in some of our tests. The exposed method, could generate some problems if the fitness computation time were too high. In fact, since the process is computed in the single node of the cluster without parallelization, there is the risk to run into a worst situation because of too many processes to be computed in series. In any case, even if one could think to overcome this problem by a further parallelization of the code related to any single node, the time response could be still too long. For these reasons, a new framework called "fitness-based" has been proposed. It consists of using any single node to compute a fitness function linked to a particle. Instead, the master node has the tasks of computing the new position and new velocities of FSO, starting from the knowledge of the old ones, provided by the slaves. Whenever the master node has to compute the fitness for a particle position it assigns a node for this task. The master will wait the return of all requested, and then it will collect them and will launch a new iteration. When all the iterations will be done, the cluster will launch the PSO in that sub-region where FSO found the best minimum. Similarly, after the PSO will be launched the BCA. In this framework the cluster serves the algorithms one-by-one, and the FM is used only with FSO. In conclusion, the main difference between and the fitness-based approach is:

1) in the algorithm-based one, the master plays the role of FSO and uses the slaves as simply finishers having the role to detect the deeper minimum in a suspected region;

2) in the fitness-based one, the master has simply the role to coordinate any slave that has the task to compute the fitness.

Obviously, the number of nodes is a crucial point. In the algorithm-based framework, it restricts the number of FM that can be launched; instead in the fitness-based framework it constrains the number of individuals in the algorithm. The fitness-based is useful when the fitness computation requires more time than all the other operations that are made in the algorithm. In this work, we have adopted this method for solving the problems related to the optimization of TWT in term of geometry, collector potentials, etc. The utilization of fitness-based for the TWT problem is justified if you think that each fitness requires about 1'30" (Intel dual-core 1.6 GHz). We can suggest this procedure to anyone who wants to use an external simulator for the fitness function. In fact, many optimization problems are treated with the help of a simulator software toolkit, especially those problems which involve finite element method. One can introduce his software in the node and recall it via web by using command shell. The figure 2.7 and 2.8 show algorithm and fitness based parallel approach.

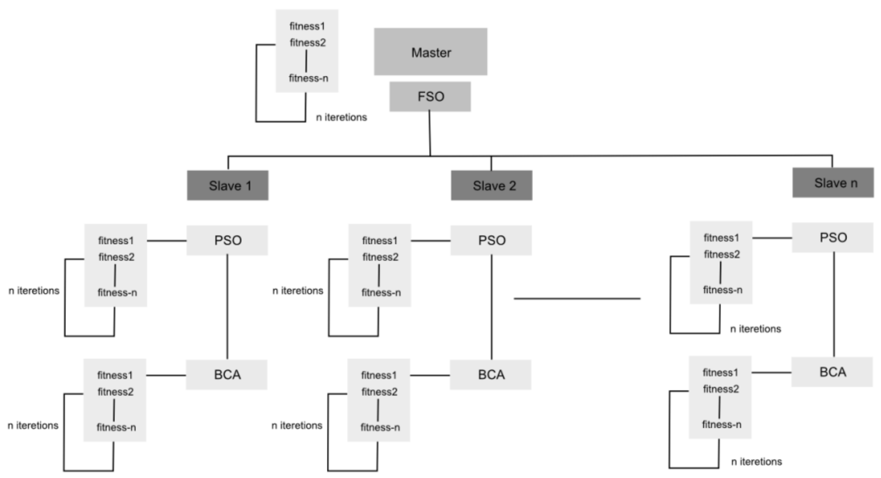


Figure 2.7: block scheme of algorithm based parallel framework.

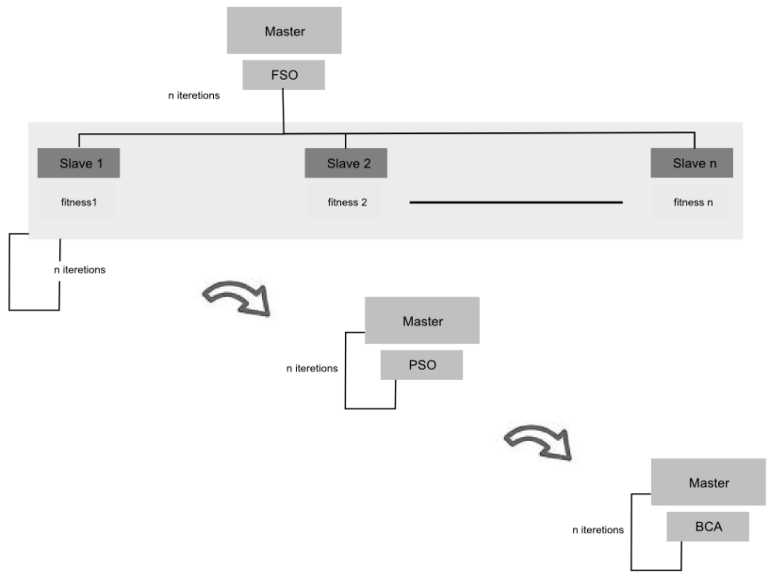


Figure 2.8: block scheme of fitness-based parallel framework.

2.6 Optimization algorithms like intelligent sampler.

In the optimization process, the exploration capability of an algorithm plays an important role in the multimodal problems. For this reason the employment of an algorithm with a good exploration capability can allow us to detect all local minima in the cost function. To proof the high exploration capability of an algorithm, could be interesting to show its capability to rebuilt the whole map of the cost function: i.e. the idea of this section is to use the algorithms as intelligent sampler in such way the cost function can be rebuilt by interpolation. By measuring the dimension of the space that an algorithm is able to explore, will be possible to estimate how good is its exploration capability. The algorithms used in this estimation of exploration capability, have already been mentioned in the abstract: PSO, FSO, MFSO. With the aim to compare the performances of the previous algorithms that are the main subject of this thesis work, the analysis has also extended to the classical Genetic Algorithm whom description is omitted (please, refer to the bibliography). Moreover, each algorithm has been tested in presence as well as in absence of the FM. In order to rebuild the cost function, each algorithm is launched in two different configurations: one oriented find all minima of the cost function, and the other oriented to find the minima of the inverted cost function. These two different configurations assure the investigation of the whole cost function, since the algorithms are designed to find minima.

In this way at the end of the evaluation, any algorithm returns a set sample data.

For rebuilding the cost function it will be necessary to invert the sign of those data coming from the sampling of the inverted cost function. Finally, it will be sufficient to use a suitable interpolator.

Several tests have been done in order to validate the exploration capability of each algorithm by following the approach based on the cost function rebuilding. In particular each algorithm has been launched 30 times on several different benchmarks.

In Table 1 are shown the obtained result related to simulations made by using the Bird function as cost function:

$$f(x,y) = \sin(x)\exp\left[\left(1-\cos(y)\right)^2\right] + \cos(y)\exp\left[\left(1-\sin(x)\right)^2\right] + (x-y)^2 . \qquad (2.2)$$

and the same number of iterations and number of individuals has been imposed for all the different algorithms (2000 iterations and 10 individuals).

The percentage of the investigated domain compared with the whole domain of the cost function is used as a parameter able to measure the capability of exploration of each algorithm. In particular the measurement of the investigated domain has been done by using the following method: the whole domain has been partitioned into several elementary sub-domains having a dimension equal to the 5% of the whole space of interest. Then we will consider as an *investigated sub-domain* that elementary portion in which the algorithm sample the cost function, or its inverted value, one time at least. Moreover, both MPE (Mean Percentage Error) and its variance over all tests are provided. MPE is calculated between the rebuilt function and the true function. As it is possible to see, the results show that MFSO with the used of the FM, obtains the best performance among the algorithms tested. In the last column the number of failures are listed. It represents the number of times when the algorithms has not explored 100% of area. In this case, MPE in calculated only over the zone rebuilt.In all tests, the cubic spline interpolation has been used. Obviously, the final quality of the interpolation depend on the specific interpolator that is used for rebuilding the cost function after the "intelligent sampling". For example, if the cloud of sampled points belong to a concave curve, the interpolator could fall in error by crossing areas that has not been investigated by the optimization algorithm. Since we make a validation starting of the domain explored this procedure could insert an overvaluation of the exploration capability. This justify the reasons from which the domain explored has been estimated by the method consisting of counting the sub-domain sampled as previously described.

Table 1. Results of the test made for the Bird Function. Each algorithm is initialized in the point [-9 -9].

Algorithm	MPE	Variance	Mean Area[%]	Variance Area[%]	Failed over 30 tests
MFSO+FM	0.0034538	1.299e-005	99.2645	2.9887	0
MFSO	0.70111	0.28314	54.6482	8.8994	30
FSO+FM	0.062557	0.016329	98.3054	11.3105	20
FSO	0.75573	0.63839	58.4974	9.2509	30
PSO+FM	0.8148	0.5132	86.7807	4.1258	29
PSO	0.5493	0.35416	22.2916	9.2859	30
AG+FM	0.32664	0.15338	17.0897	2.7381	30
AG	0.11969	0.016434	7.569	0.40311	30

A further analysis has been made with the aim to estimating the sensitivity of the exploration capability with the number of iterations of the various algorithms. In fact, increasing the number of iterations (NoI) the exploration obviously improves, because the algorithm simply has more time to attempt to find the minima. The results of this analysis will be next show with reference to the Giunta function:

$$f\left(x_1,x_2\right)=0.6+\sum_{i=1}^{2}\left[\sin\left(\frac{16}{15}x_i-1\right)+\sin^2\left(\frac{16}{15}x_i-1\right)+\frac{1}{50}\left(4\left(\frac{16}{15}x_i-1\right)\right)\right] \qquad (2.3)$$

From observing the results it is evident that the FM utilization in some way, operates a sort of boost for exploration capabilities of each algorithm, maintaining the original ranking among the algorithms. Hence, the FM can be considered as a stand-alone technique and not always connected with the MeTEO algorithm utilization. The comparison of behavior without FM shows a behavior that is deeply dependent on the probabilistic terms existing in the algorithm. In this scenario, when you double the NoI the inspected area could not double. When the algorithm is trapped in a local minima if the random terms are not high enough, it will remain in the local minima, the FM provide a way to escape from it without makes changes in the kernel formula of swarm algorithm, but simply remarking its attitude of sensing "the hills and the valley". In the figures the bar graph representing the percentage of inspected area with and without the FM are depicted. FM plays a role in a field previous the sake of the right way for finding a minimum. Before it, there is the concept of exploration that excludes at the beginning the possibility of reaching a minimum whether a particular area of the solution space has not been inspected. On the other hand, the other edge is the brute force computation of all possible solutions without any kind of intelligence. Between these two boundaries lie the utilization of FM.

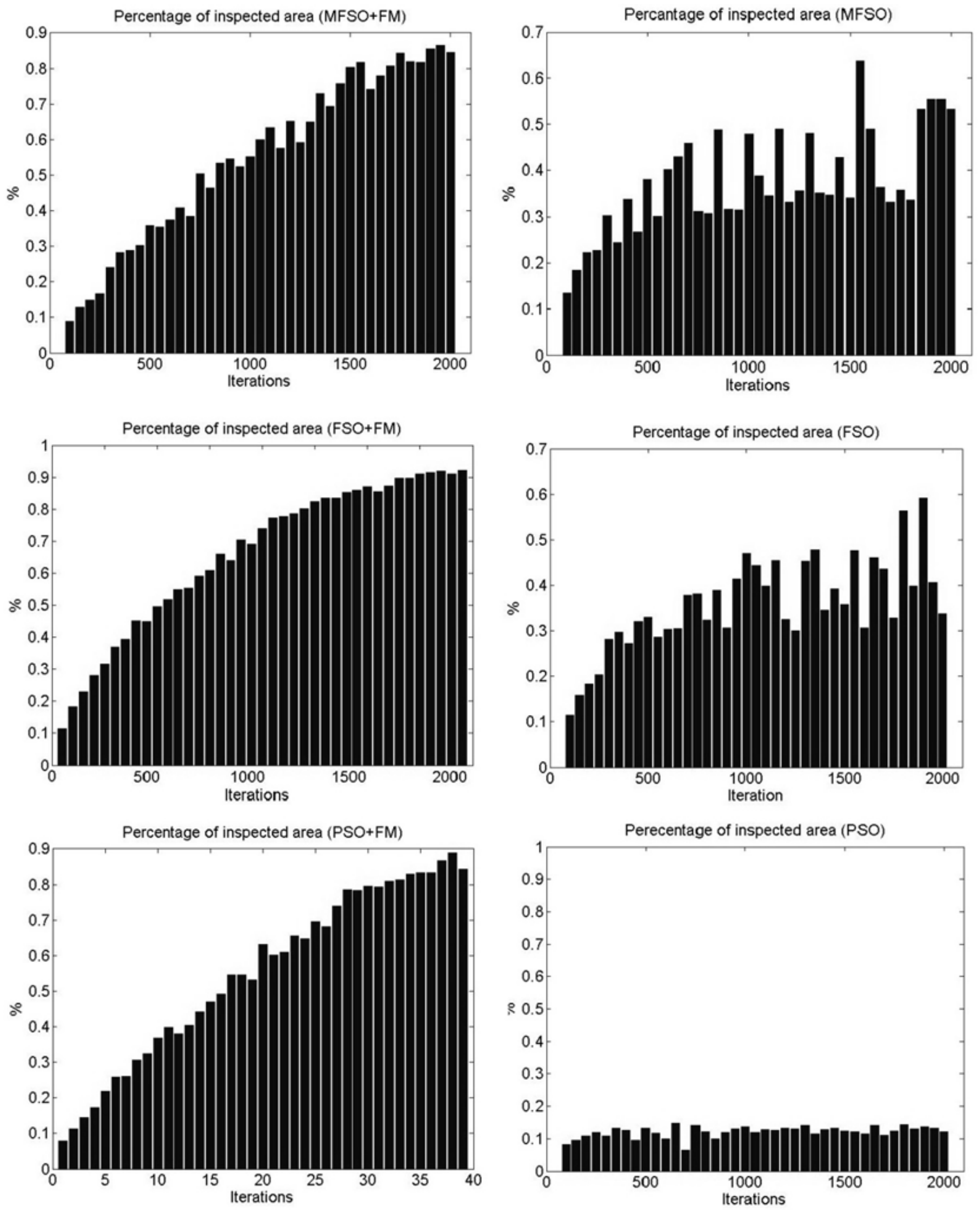


Figure 2.9: comparison of percentage of inspected area vs Number of iterations, with and without FM.

2.7 Binary Flock of Starlings Optimization BFSO.

BFSO is an extension of the Discrete Particle Swarm Optimization. The trajectory in the current model is probabilistic, and velocity on a single dimension is the probability that a bit may change. The velocity of a single particle is interpreted as the probability that the current position may change from a current state to another. In this case the algorithm becomes "binary", thus the position can be 0 or 1. In it, $p\_best_k^j, x_k^j(t), g\_best_k$ are integer in $\{0,1\}$. Being v_k^j a probability, it is constrained to 0.0 and 0.1. A logistic transformation is introduced by using a sigmoid function in order to constrain the velocity:

$$S(v_k^j) = \frac{1}{1+e^{-v_k^j}} \tag{2.3}$$

Afterwards, the resulting position is defined by the following rule:

$$\begin{aligned} &if\left(rand < S\left(v_k^j\right)\right)\, then\ x_k^j\left(t\right) = 1; \\ &else\ x_k^j\left(t\right) = 0 \end{aligned} \tag{2.4}$$

Starting from these equations we can obtain the BFSO model. In fact, in the FSO each individuals chooses the direction in accordance with the velocity of other members random chosen in the swarm. But now the velocity is the probability that an individual will change its status. Therefore, the choice of an individual is influenced by the mean probability of changing of the other member followed by it. The velocity equation for the BFSO simply becomes:

$$v_k^j(t+1) = [\omega^j v_k^j(t) + \lambda^j(p\_best_k^j - x_k^j(t)) + \gamma^j(g\_best_k - x_k^j(t))] \cdot Mccb_k^j \tag{2.5}$$

$Mccb_k^j$ is the mean probability that a figure could change from 0 to 1 affected by the members in the swarm followed by him. This is a method for linking the choice of 1 with the other members. The value of the $Mccb_k^j$ is constrained in [0.0, 1.0], so as to underestimate the influence of other members on the generic individual. It has proved a right choice because linking individuals in a strong way could produce a stagnation and saturation in 1 or 0 direction, because if all members are strongly linked, after a short period of time they will uniform the population saturating to single value, for example an array of all figures 1, or otherwise an array of only figures 0. Let us to explain the pseudo code, that follows. The information of all birds is recorded in the array called 'Starling'. In the function Initialization ( ) random initial velocities and positions are assigned to each bird. Afterwards the matrix interconnection is made to associate with each bird N_{ctrl_birds} of other members of the flock; it has dimension $[N_{birds}, N_{ctrl\_birds}]$, where N_{birds} is the number of the birds of the starling. For each bird the fitness is calculated and if it is minor than the previous, personal best values are refreshed. Indeed, global best is the best fitness among the all values of the starling. Then, the *Mccb* term is calculated taking the velocities only of the birds listed in matrix interconnection. These three terms cooperate to update the velocity and position according to eq 2.5.

```
Main()
Initialization();
Make_Mat_Interc();
For i=1:nstep
     For each bird
           Fitness();
           Gbest_perform();
           Pbest_perform();
           MCCB();
           Velocity_update (with equation 2.5);
           Position_update;
     End
End
```

References

[1] F.R. Fulginei and A. Salvini, Comparative analysis between modern heuristics and hybrid algorithms, COMPEL, 26(2) (2007), pp. 264–273.

[2] C.W. Liew and M. Labiri, Exploration or Convergence? Another Metacontrol Mechanism for GAs, Proceedings of 18th International Florida AI Research Society Conference, FLAIRS, Miami Beach, FL, USA, 2005.

[3] F.R. Fulginei and A. Salvini, Hysteresis model identification by the flock-of-starlings optimization, Int. J. Appl. Electromagnet. Mech. 30(3–4) (2009), pp. 321–331.

[4] F.R. Fulginei and A. Salvini, Influence of topological rules into the collective behavior of swarm intelligence: The flock of starling optimization, Stud. Comput. Intell. 327/2011 (2011), pp. 129–145.

[5] M. Ballerini, N. Cabibbo, R. Candelier, A. Cavagna, E. Cisbani, I. Giardina, V. Lecomte, A. Orlandi, G. Parisi, A. Procaccini, M. Viale, and V. Zdravkovic, Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study, Proc. Natl Acad. Sci. 105 (2008), pp. 1232–1237.

[6] S.D. Muller, J. Marchetto, S. Airaghi, and P. Kournoutsakos, Optimization based on bacterial chemotaxis, IEEE Trans. Evol. Comput. 6(1) (2002), pp. 16–29.

[7] J. Kennedy and R. Eberhart. Particle Swarm Optimization, Proceedings of the IEEE International Conference on Neural Networks, Vol. IV, Perth, Australia, IEEE Service Center, 1995, pp. 1942–1948.

[8] C.W. Reynolds, Flocks, herds and schools: A distributed behavioral model, Comput. Graph. 21(4) (1987), pp. 25–34

[9] F. Heppner and U. Grenander, A stochastic nonlinear model for coordinated bird flocks, in The Ubiquity of Chaos, S. Krasner, ed., AAAS Publications, Washington, DC, 1990, pp. 233–238.

[10] A.P. Engelbrecht, Computational Intelligence: An Introduction, Wiley, New York, 2002.

[11] M.M. Ali and P. Kaelo, Improved particle swarm algorithms for global optimization, Appl. Math. Comput. 196 (2008), pp. 578–593.

[12] M. Clerc and J. Kennedy, The particle swarm: Explosion stability and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. 6(1) (2002), pp. 58–73.

[13] F. Glover, Tabu search methods in artificial intelligence and operations research, ORSA Artif. Intell. 1(2) (1987), p. 6.

[14] G. Pulcini, F. Riganti Fulginei, and A. Salvini, "Metric-topological- evolutionary optimization," in Inverse Problems in Science and Engineering. London: Taylor and Francis, 2011, pp. 1–18, 10.1080/17415977.2011.624624.

Chapter 3

Validation and Application of MeTEO.

MeTEO has been tested on many benchmarks and to solve inverse problems. The aim of this chapter is to provide an exhaustive validation of MeTEO and to show its versatility and robustness. The validation process of the optimization algorithm is always a critical task, because there is not a common layer in the scientific community that allows to define a standard procedure for evaluating the various properties. Into the state-of-art, this problem is faced by providing tools that automatically generate tests combining famous benchmarks through geometric operations such as translation, mirroring, etc. In the study here presented, MeTEO has been tested not only by using of classical approaches, but even on hard benchmarks specifically ideated with the aim to proof its strong ability to return good results in comparison with the performances shown by other famous heuristics. With "hard benchmark" we mean those functions that do not present intuitive derivative direction, and that are asymmetric in relation to the hills and valleys distribution. MeTEO has been validated on typical optimization benchmarks, as well as on typical inverse problems, i.e. the identification of the J–A hysteresis model through the knowledge of an experimental hysteresis loop, and the fit problem called 'Amid_pro' which is one of the 295-ODE-test examples proposed by Schittkowski. Furthermore, some paragraphs are spent about the optimization of the TWT devices efficiency.

3.1 Validation on Optimization Benchmarks

Let us to consider an ad hoc benchmark as follows:

$$\begin{cases} f(x,y) = 50\cos(0.2\pi x) + \cos(0.05\pi) - x^2 - y \\ (-20 \le x \le +20, -20 \le y \le +20) \end{cases} \tag{3.1}$$

In Figure 3.1, the cross-sections of (3.1) in planes at constants x and y are shown. The benchmark (3.1) shows its smallest value (global minimum) in the point that is located at the border of the variable-range: $(x_{\min} = -20, y_{\min} = 20)$. Thus, if MeTEO is initialized at the opposite corner of the variable-range: $(x_{start} = -20, y_{start} = 20)$ (Figure 3.1), it has to exit to the closest 'attractive' local minimum in $(-20, 20)$, and to the other local minima that are on the 'hill' that separates the starting point from the global minimum.

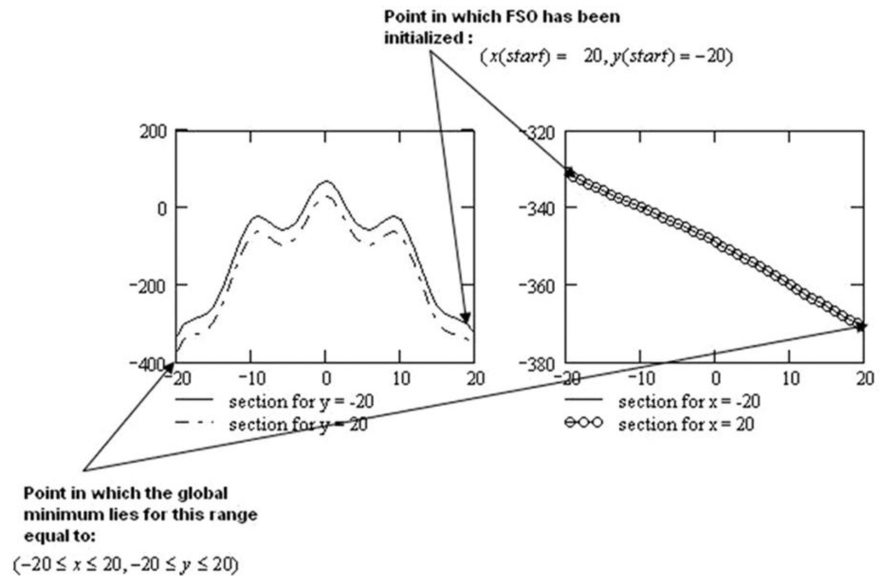


Figure 3.1: Cross-sections of (2) in planes at constants x and y.

Each single heuristic that composes MeTEO will explore a different range within the solution space according to the different exploration capabilities which are peculiar of an heuristic rather than the others. Thus, for problem (3.1), whereas the FSO explores the whole range: $-20 \le x \le 20, -20 \le y \le 20$, the PSOs and the BCAs explore smaller sub-regions, depending on the size of the 'suspected region' detected by the FSO. At the end of the whole process, MeTEO detected 14 'suspected regions' for problem (3.1). Among them, MeTEO detect the global minimum in the last detected suspected region. More in detail, in this last region, the FSO detected the smallest value equal to $f(-17.03, 17.51) = -323.04$, then the PSO performed $f(-18.03, 18.09) = -327.82$ and finally the BCA found the best fitness equal to -363.77 at the point $(-19.82, 19.58)$.

Taking into account that the true global-minimum co-ordinates are in (-20, 20), the good accuracy of the MeTEO performance is evident. The total PC-Master processing time (FSO) has been 54.8 s. After that, the processing time of PC-Slaves working in parallel has been about 67 s (the average time taken by PSOs was 63 s whereas a time of 4 s has been necessary for BCA).

This *ad hoc* example gives us the core potential of MeTEO.

The MeTEO performances has been tested also for each benchmark presented in [11].

The dimension of the search space has been intentionally made sizeable in order to make the optimization more difficult [12], and just 1000 iterations have been set both for MeTEO and for the other single heuristic.

All the simulation results are listed in Tables 2 and 3 for analysis and comparisons. In particular, Table 2 reports the number of global optima correctly detected by each algorithm. We have considered as a success the event in which the algorithm finds a minimum, showing a percentage error on the true minimum smaller than a fixed threshold. As it is possible to see in Table 2, for the unimodal optimization, Levy and Schaffer, the MeTEO always obtains a success like FSO does, whereas both PSO and BCA fail for the Schaffer function. For harder multimodal benchmarks, the power of MeTEO is more evident. In fact, whereas it finds anyway at least one global minimum, and in many cases all the global minima, FSO, PSO and BCA cannot assure the same performances.

In Table 3, the best fitness results obtained by MeTEO are listed. In particular, Table 3 shows

Table 2:number of global optima correctly detected.

Benchmark	MeTEO (number of found minima)/ (number of total minima)	FSO (number of found minima)/ (number of total minima)	PSO (number of found minima)/ (number of total minima)	BCA (number of found minima)/ (number of total minima)
Levy	1/1	1/1	1/1	0/1
Schaffer	1/1	1/1	0/1	0/1
Giunta	3/3	1/3	0/3	0/3
Bird	4/4	4/4	2/4	0/4
Test tube older	3/6	2/6	1/6	0/6
Cross in Tray	4/4	2/4	1/4	0/4
Bukin	2/5	1/5	1/5	0/5

the best performance achieved by each single MeTEO component in the best 'suspected region'. For multimodal functions, the best results obtained for one of the global minima are indicated. Finally, in the last column of Table 3, the number of 'suspected regions' detected by MeTEO during the elaboration is finally reported.

Table 3: MeTEO performances.

Benchmark	FSO	PSO	BCA	Number of detected 'suspected regions'
Levy	0.001128100069129	$1.112549902700000 \times 10^{-5}$	$1.626793947000000 \times 10^{-6}$	138
Schaffer	0.004245907810385	$6.985129917416143 \times 10^{-4}$	$-7.216739485161167 \times 10^{-5}$	30
Giunta	0.064660746325632	0.064471520076312	0.064470421318009	18
Bird	$-1.066907387112337 \times 10^{2}$	$-1.067645235721507 \times 10^{2}$	$-1.067645142301463 \times 10^{2}$	169
Test tube older	−10.808077079559949	−10.852437039909185	−10.852491680226905	131
Cross in Tray	−2.061780609712529	2.062611702638596	−2.062611853281188	155
Bukin	1.297356414422297	0.597342511668211	0.161506034691595	165

3.2 Validation on Inverse Problems

MeTEO has been tested also on inverse problems as follows.

3.2.1 Identification of the J–A model

MeTEO has been tested also on an inverse problem: the Jiles Atherton hysteresis model identification. Let us remind Jiles Atherton model [5]:

$$\frac{dM}{dH} = \frac{(1-c)\frac{dM_{irr}}{dH_e} + c\frac{dM_{an}}{dH_e}}{1-\alpha c\frac{dM_{an}}{dH_e} - \alpha(1-c)\frac{dM_{irr}}{dH_e}} \tag{3.2}$$

where M_{an} is the anhysteretic magnetization provided by the Langevin equation:

$$M_{an}(H_e) = M_s\left(\coth\left(\frac{H_e}{a}\right) - \frac{a}{H_e}\right) \tag{3.3}$$

in which H_e is the effective magnetic field $H_e = H + \alpha M$. In (3.2) M_{irr} is the irreversible magnetization component defined by:

$$\frac{dM_{irr}}{dH_e} = \frac{M_{an} - M_{irr}}{k\delta} \tag{3.4}$$

whereas $\delta = sign\left(\frac{dH}{dt}\right)$.

The parameters identified of the J-A model are α, a, c, k, M_s and their physical meaning are: a is a form factor, c is the coefficient of the walls movement reversibility, M_s

is the saturation magnetization, and finally α and k represent the hysteresis losses and the interaction between the domains respectively.

The performed test is based on the use of a given set of parameters [5] $\left[a,\ k,\ \alpha,\ c,\ M_s\right]=\left[24.70,\ \ 60,\ \ 6.9010^{-5},\ \ 0.053,\ \ 1.16\cdot 10^{5}\right]$ that has been inserted into equations (3.2)-(3.4) to obtain a pseudo-experimental loop by integration. The sampled points of this pseudo-experimental loop have been used to estimate the error between that loop and that returned by MeTEO and by the other single heuristics. All the algorithms have been initialized randomly; the maximum number of "suspected regions" has been set to 80.

Table 4: Jiles-Atherton inverse problem MeTEO and its components results

Algorithm	**Parameter estimated** $\left[a,\ k,\ \alpha,\ c,\ M_s\right]$	**Percentage error on the experimental loop points**
MeTEO	$\left[24.34\ \ 59.84\ \ 6.26\cdot 10^{-5}\ \ 0.050\ \ 1.158409\cdot 10^{-5}\right]$	0.05368%
FSO	$\left[14.36\ \ 64.12\ \ 3.57\cdot 10^{-6}\ \ 0.046\ \ 1.072892\cdot 10^{5}\right]$	4.71548%
PSO	$\left[17.27\ \ 55.73\ \ 6.22\cdot 10^{-5}\ \ 0.004\ \ 1.047480\cdot 10^{5}\right]$	7.07373%
BCA	$\left[22.20\ \ 53.88\ \ 1.080195758008603\cdot 10^{-4}\ \ 0.006\ 1.044000\cdot 10^{5}\right]$	8.1803%

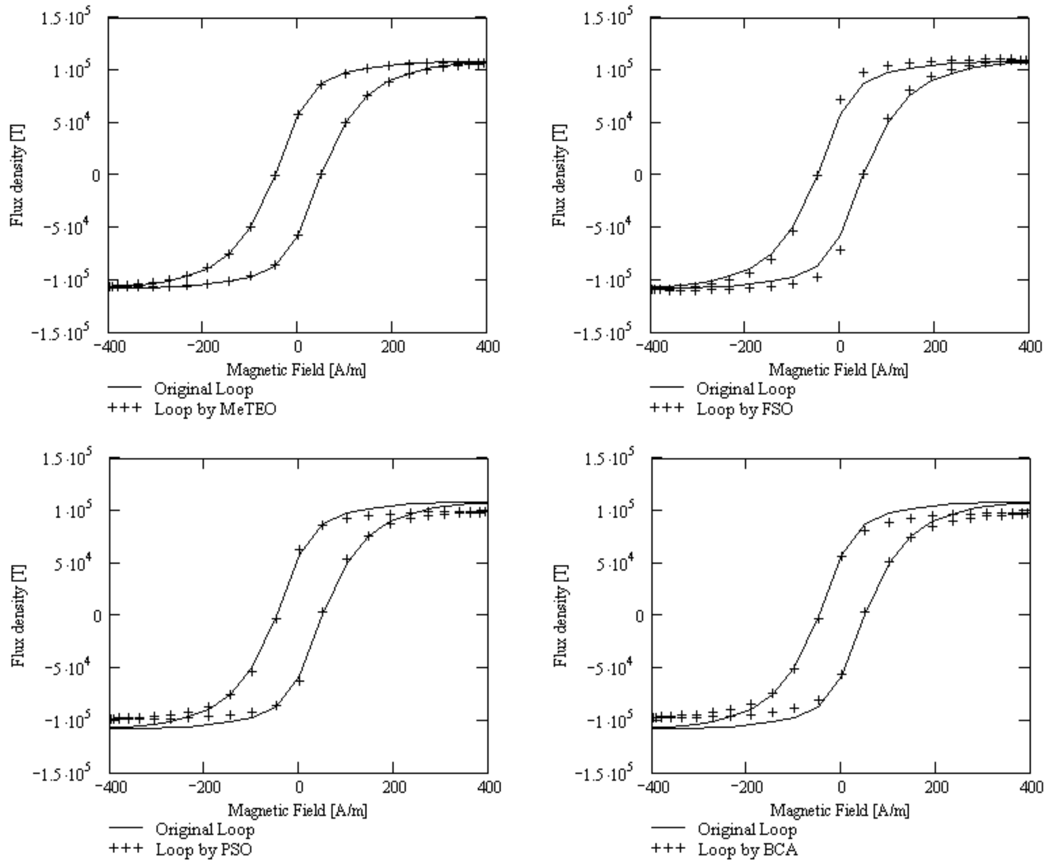


Figure 3.2: Comparison of the simulation results obtained by MeTEO and FSO, PSO and BCA when each one works alone.

In (Table 4) are shown the results obtained by MeTEO and its components working alone. At the end of the simulation the parameters identified by MeTEO show a low average percent Mean Square Error (MSE). This MSE is quite lower than that obtained by FSO, or PSO, or BCA, working alone. In (Figure 3.2) it is possible to see how MeTEO returns a hysteresis loop practically coincident with the experimental one, compared to the loops returned by FSO, PSO and BCA when each one works alone. A further statistical analysis has been made, using two indicators: MPE (Mean Percentage Error) and R^2 ; the results are listed in (Table 5).

Table 5: Jiles-Atherton inverse problem MeTEO and its components statistic analysis.

Jiles-Atherton[0.3]	*MPE*		
Algorithm	*Mean*	σ^2	R^2
BCA	2.4268	2.9507e-2	0.5870
PSO	1.6388	1.6527e-3	0.9968
FSO	1.294	0.9030e-3	0.9993
METEO	0.53552	1.4576e-5	0.9996

3.2.2 Amid proton replacement inverse problem [4].

To validate MeTEO a further test has been done, considering the Amid_pro inverse problem. It is described by the differential system equation reported in eq. (3.5):

$$\begin{cases} \dfrac{dy_1}{dt} = -k_f y_1 \\ \dfrac{dy_2}{dt} = k_f \left(c - y_2\right) \\ \dfrac{dy_3}{dt} = k_f y_4 - 0.1 k_n \left(y_2 - y_3\right) - 0.9 k_n y_3 \\ \dfrac{dy_4}{dt} = -k_f y_4 - 0.9 k_i y_4 + 0.1 k_i y_3 \end{cases} \tag{3.5}$$

The performed test is based on the usage of a given set of parameters

$\left[k_f,\ k_i,\ k_n,\ c\right] = \left[10^{-4},\ 10^{-3},\ 1.18\cdot10^{-4},\ 1.2\cdot10^{5}\right]$. As shown in (Table 6) MeTEO is able to perform the better solution than the employing of singular heuristics. Also in this tests the performance have been computed over 50 launches and the suspected regions are equal to 80. In figure 3.3 are depicted the curves for the amid_pro model, for the single heuristic, and for MeTEO.

Table 6: Amid_pro inverse problem MeTEO and its components results

Algorithm	Parameter estimated $\left[10^{-4},\ 10^{-3},\ 1.18\cdot10^{-4},\ 1.2\cdot10^{5}\right]$	Percentage error on the 4-dim Amid_pro
MeTEO	$9.9975\cdot10^{-5}, 1.0004\cdot10^{-3}, 1.179510^{-4}, 120039{,}57$	0.0060%
FSO	$9.9607\cdot10^{-5}, 1.0427\cdot10^{-3}, 1.1505\cdot10^{-4}, 120076{,}66$	0.7709%
PSO	$1.0005\cdot10^{-4}, 1.004\cdot10^{-3}, 1.1793\cdot10^{-4}, 119985{,}80$	0.1628%
BCA	$1.0285\cdot10^{-4}, 1.011710^{-3}, 1.268610^{-4}, 116088{,}48$	2.0764%

Also for the Amid_pro inverse problem has been made a statistical analysis (Table 7). The algorithms have been initialized far from the optimal parameter array written above and in particular in a point equal to $\left[k_f,\ k_i,\ k_n,\ c\right] = 0.3\cdot\left[10^{-4},\ 10^{-3},\ 1.18\cdot10^{-4},\ 1.2\cdot10^{5}\right]$.

Table 7: Amid_pro inverse problem MeTEO and its components statistic analysis.

Amid_pro[0.3]	*MPE*		
Algorithm	*Mean*	σ^2	R^2
BCA	10.8683	5.948e-2	0.3476
PSO	2.9633	7.9884e-3	0.9968
FSO	1.0674	0.4048e-4	0.9987
METEO	0.42562	0.1292e-4	0.9999

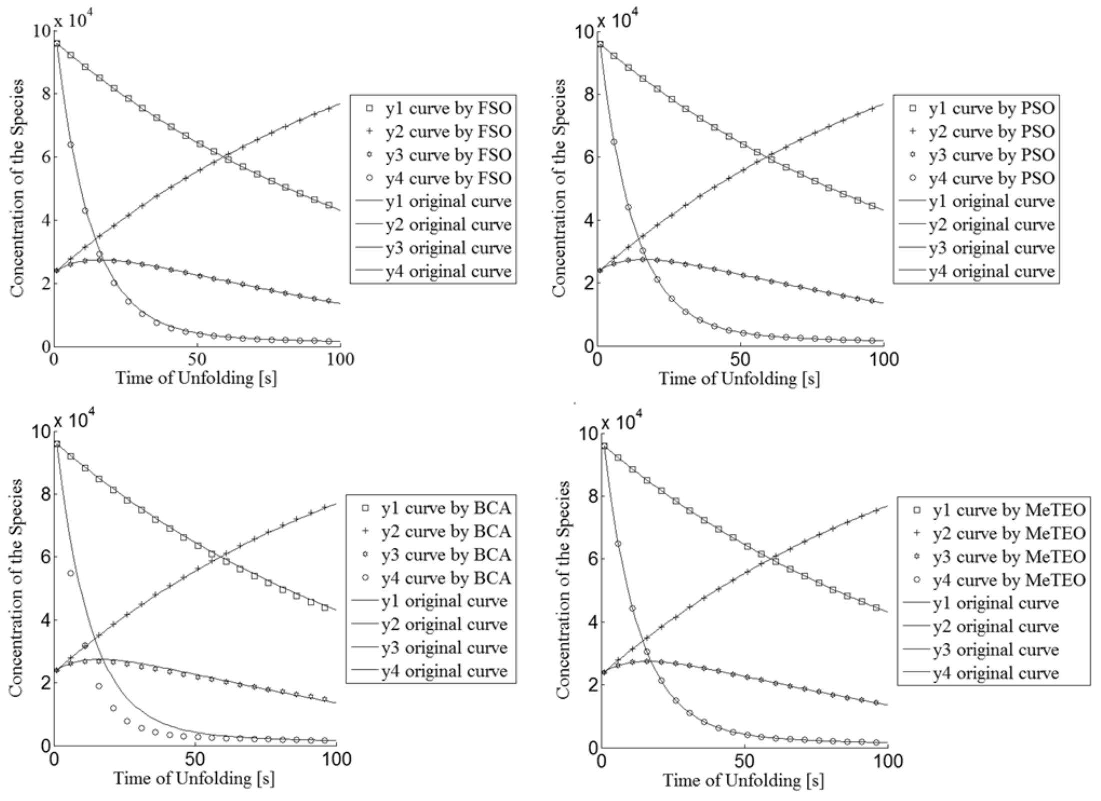


Figure 3.3: Comparison of the simulation results obtained by MeTEO and FSO, PSO and BCA when each one works alone on the amid_pro inverse problem.

3.3 Optimal Underground Power Cable Displacement [27].

In this section will be treated a particular kind of problems that has been used to validate the Binary Flock of starling Optimization introduced in the previous chapter: the optimal underground power cable displacement problem, which belongs to the discrete problems class.

In the last years the electric companies have revisited the design method of the underground power cables displacement in order to face the problem of optimal displacement. In fact, when a significant number of circuits is placed closer, they can produce a maximum magnetic flux density over the tunnel.

A substation is affected by EM pollution, but fortunately consolidated shield techniques already exist, then the field outside the substation is negligible. The problem moves on the cables that leave the substation. There is necessity of optimization techniques at design step, concerning both the geometrical and the circuit assignation of each bundle. Other authors have faced this problem [1] with evolutionary algorithm as well [2], and with good results. From the evolutionary computation perspective this kind of problems belongs to the Mixed-Integer and Constrain Programming (MICP), in which discrete variables appear. In such kind of problems, the algorithms usually employed operates as a string generator, where the string is the individual that codifies a possible solution. Being the solution a string of number the first inconvenient is that some solutions are incompatible with the physical problem. The aim of this section is to analyze the performance of Binary Flock of Starlings Optimization over MICP problems. Penalty techniques are widely employed in MICP [3], in our work we use a death penalty, which consists in assigning a huge value (for example 10^9 or the value Inf, if you are using Matlab as in our case) to the "bad" solution that does not meet the constrains.

The problem taken into exam to test and validate BFSO, is the minimization of the magnetic flux generated by underground power cables in a substation, in order to decrease its value over the ground. Following, is given the procedure to calculate of the magnetic flux using Bio-Savart law. For a cable routed by the current I, the magnetic flux generated in a generic point locates in the space is:

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0 I}{2\pi} \frac{\mathbf{l} \times \mathbf{r}}{|\mathbf{r}|^2} \tag{3.6}$$

A current parallel to z axis is considered; the direction in which it flows is identified by the vector $\mathbf{I}$, indeed, $\mathbf{r}$ is used to indicate direction that links the axis where the current is flowing, with the generic point, in the xy plane as shown in Figure(3.4).

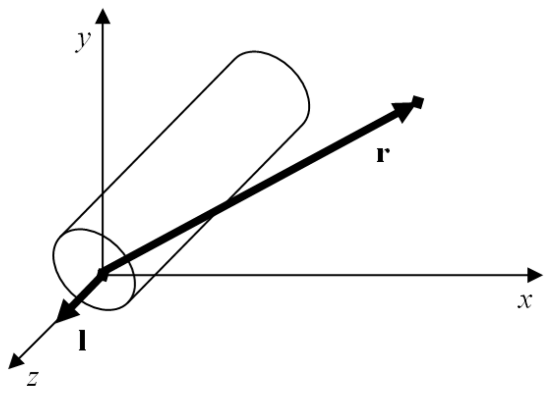


Fig. 3.4: 3D representation of flux magnetic calculation for a cable routed by a current of value I.

Although the current vector is laid upon the z axis, it is sufficient the analysis of xy plane (Figure 3.5). Each sample point is indicated with $P_S = (X_S, Y_S)$, all samples point are taken one meter over the ground (Figure. 3.5), for each of them it is calculated the magnetic flux considering a constant uniform distribution current of value $\dot{I}_{k,i}$. The passages are shown below:

$$\begin{aligned}
\mathbf{B}_{k,i} &= \frac{\mu_0}{2\pi}\frac{\dot{I}_{k,i}\hat{z}\times(a\hat{x}+b\hat{y})}{|\mathbf{r}|^2} = \\
&= \frac{\mu_0}{2\pi}\frac{-\dot{I}_{k,i}a\hat{y}-\dot{I}_{k,i}b\hat{x}}{(X_S-X_i)^2+(d+|h|)^2} = \\
&= \begin{cases} -\dfrac{\mu_0}{2\pi}\dfrac{\dot{I}_{k,i}b\hat{x}}{(X_S-X_i)^2+(d+|h|)^2} \\ -\dfrac{\mu_0}{2\pi}\dfrac{\dot{I}_{k,i}a\hat{y}}{(X_S-X_i)^2+(d+|h|)^2} \end{cases}
\end{aligned} \tag{3.7}$$

where:

- $\dot{I}_{k,i}$: complex quantity representing the current that flows in the i-th cable of the k-th bundle;
- μ_0: magnetic permeability ;
- X_i: x coordinate of the i-th cable ;
- X_S : x coordinate of the generic sample point;
- h: distance of the cable i-th from $x = 0$ plane;
- $d = 1$: is the sample line's height over the ground;
- $b = (h + d)\hat{y}$;
- $a = |X_i - X_S|\hat{x}$

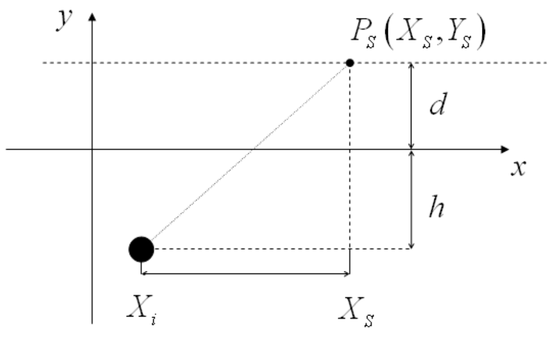


Fig. 3.5: Showing of the xy plane, in the picture there are the distance quote of the sample line over the ground and the position of the generic I-th cable.

The **r** module is simply calculated using the Pitagora Theorem. The above passages represent the standard application of the Ampere law for a cable routed by a constant and uniform current distribution. In the fitness calculation this procedure are performed querying four table in which have been record respectively:

- the value of coordinate x for each cable;
- the value of coordinate y for each cable;
- the complex value of current I for each cable;
- the values of current provided by the available circuit;
- all the possible permutations for a single bundle (for three phase bundle it is 3!=6).

The aim of the optimization is to minimize the maximum value of the magnetic field in the sample line, hence in the fitness function the magnetic field values are first computed for all sample points and then the max value is chosen.

3.3.1 Simulation and experimental results

The actual design for cables displacement in a tunnel is a trefoil configuration in order to minimize the effects of capacitive and inductive currents and support by racks, as suggested in [1]. The circuits used have been reproduced from the [1] and for clarity shown below in Table 8 and 9, changing the number of circuit employed, that is 4 and 8. In the (Figure 3.6) is depicted an example of the rack for displacement of cable with the relative position of the bundles. The combinations of the cables are 6 and they can be explicitly expressed, 123;132; 213; 231; 312; 321, for instance if there is an individual such as 6345, it means that the first circuit is arranged as 321, the second as 213 and etc. Other authors have chosen the letters RST to identify each of the three cables; in this work it is used the numbers 123 just for a more simple software implementation. Given N_C as the number of circuits, and being N_B the number of bundles, then an individual is codified as an array of 0 and 1 as follow:

- If $N_C = 4$ we use a binary array of $(3 \cdot N_B) + (2 \cdot N_C)$ positions;
- If $N_C = 8$ we use a binary array of $2 \cdot (3 \cdot N_B)$ positions;

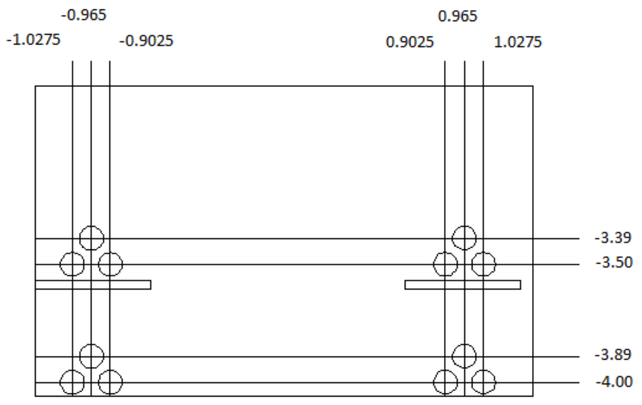


Fig.3.6: Displacement of bundles in the underground rack, all values are expressed in meter.

All the tests have been conducted with a starting random initialization; the number of individuals for each algorithm is 10; we have implemented the algorithm with MATLAB© software.

Table 9:Power circuit employed test 4

N.circuit	P(Mw)	Q(MVAr)	Im (A)	θ (deg)
A	180	60	680	18
B	155	43	577	16
C	-100	-25	370	194
D	-125	-30	461	193

The first part of the array that represents an individual has to be composed from integer value in the [1,2….,6] range, in this portion the figures can be equals, whereas in the second portion of array, which represents the connection among the individuals, all the

Table 8:Power circuit employed test 8

N. circuit	P(Mw)	Q(MVAr	Im (A)	θ (deg)
A	160	40	591	14
B	155	43	577	16
C	-105	-15	380	188
D	-100	-10	360	186
E	160	35	587	12
F	135	28	494	12
G	-90	-10	325	186
H	-95	-15	245	189

numbers have to be different from each other because the number of bundles is always equal to the number of connection circuits, and two different bundles cannot be connected to the same circuit. For these reasons has been employed a penalty process; besides, 50 launches for each test and each algorithm have been made and a statistical analysis has been performed.

As shown in the table A, for the 4 bundles and 4 circuits problem, both algorithms achieve the same result, although the BFSO with a smaller variance, basically they have the same response. In the 8 bundles and 8 circuits problem, the BFSO reaches a mean value better than that of the DPSO, this because it inherits the exploration characteristics of the continuous algorithm FSO, then it is able to find many different candidate solutions.

Nevertheless, DPSO provides the better configuration. In fact DPSO owns a good capability of convergence and over 50 launches, being the initialization random, it might start near a good solution, and refine that in a better way than the DFSO does.

Table A:result for 4 bundles test.

	Mean	Variance	Best Configuration	Best Fitness Value
BFSO	6.31e-7	1.75e-16	56533124	6.25e-7
DPSO	6.40e-7	3.26e-16	56533124	6.25e-7

Table B:result for 4 bundles test.

BFSO	5.94e-007	1.00e-014	[34645531 73854126]	3.82e-007
DPSO	6.17e-007	1.19e-014	[11411164 47812365]	3.79e-007

3.4 Optimization of TWT parameters.

This section describes the performances of MeTEO related to the optimization of Traveling Wave Tubes (TWTs) devices in order to increment the efficiency of the whole system. The TWT is a microwave device known as linear-beam tubes. TWTs are used for frequencies ranging from just below 1GHz to over 100GHz; power generation capabilities range from watts to megawatts. The two most important RF circuits devised to use in TWTs are:

- Helix, for broadband applications
- Coupled cavity, for high power applications.

The TWT addressed in this thesis is about helix-type. The efficiency of the TWT is dependent by the geometry structure, the magnetic focusing field, and utilization of the multiple depressed collectors. Travelling wave tubes are used for frequencies ranging just below 1GHz to over 100GHz; power generation capabilities range from watts to megawatts. For helix TWTs, bandwidths may be as high as two octaves or more. For coupled cavity TWTs, bandwidths in the 10-20% range are common.

TWTs have many applications. They are used as the final amplifiers in nearly all communications satellites (both up link and down link). In the coherent radar systems, one TWT (or more) is used as the high power amplifier that generates the transmitted RF pulse. In other radar systems, a TWT may be used as the driver for some other high power RF amplifier such as a crossed field amplifier. In Figure 3.7 are depicted the main components of a basic helix TWT.

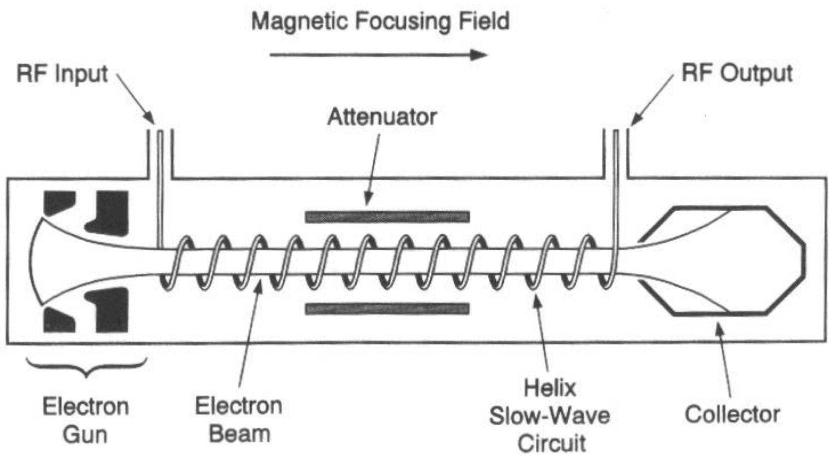


Figure 3.7: basic helix TWT components.

The efficiency in converting dc input power to RF output power at the desired frequency is an important characteristic for a TWT. One very important part of overall efficiency is electronic efficiency, which is the conversion efficiency of power in the electron beam to RF power. The efficiency of the device can be improved with a suitable design of the collector. All the inverse and optimization problems of TWT is about the increasing of the efficiency by mean of MeTEO. Hence, let us to make a briefly recall of the efficiency calculation.

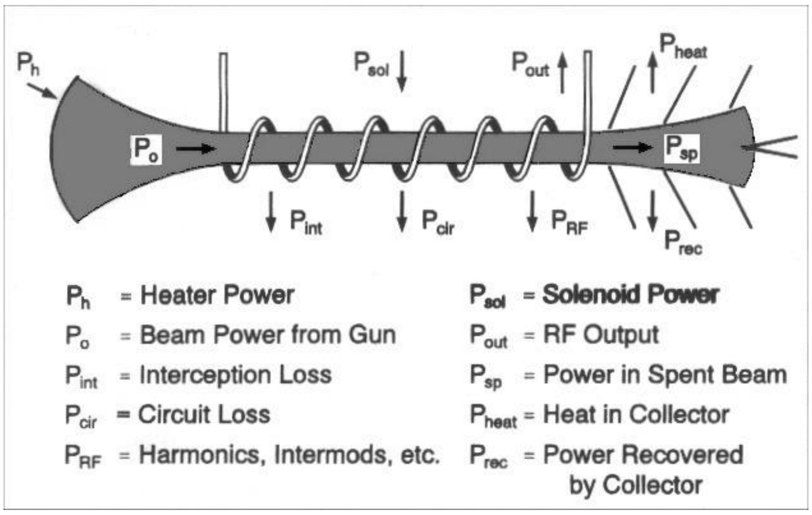


Figure 3.8: power flow overview in the linear beam tube.

Using the various elements of power identified in Figure 3.8, it is straightforward to derive the relationships between power flow and efficiency. The overall tube efficiency, η_{ov} ηov , is simply the ratio of the RF output power to the total input power, Pin, that is

$$\eta_{ov} = \frac{P_{out}}{P_{in}} \tag{3.8}$$

After several calculations, the final formula for the overall tube efficiency is:

$$\eta_{ov} = \frac{\eta_{cir}\eta_e}{\frac{P_{ot}}{P_o} - \eta_{coll}\left[1 - \eta_e - \frac{P_{\text{int}} + P_{RF}}{P_o}\right]} \tag{3.9}$$

where:

- η_{cir} is the circuit efficiency, which is the efficiency of the RF circuit in delivering the generated RF power at the desired frequency to the output connector of the tube.
- η_e is the electronic efficiency, which is the efficiency of conversion of beam power to RF power at the desired frequency.
- P_{ot} is given by the following formula $P_{ot} = P_o + P_h + P_{sol}$.

- η_{coll} is the collector efficiency, given by $P_{rec} = \eta_{coll} P_{sp}$.

3.4.1 TWT geometry structure optimization [25]

Specialized 3-D simulators are required designing TWT multistage-depressed-collector (MDC) especially in those cases which new and innovative geometries are investigated for improving the efficiency of these devices. To perform such a task a Finite Element (FE) approach can be pursued, since it allows a very flexible meshing by using irregular meshes to properly fit the MDC's geometry. In this way, complex geometries can be accurately simulated [6]. Unfortunately, the use of optimization techniques in the design process of these devices is rarely used or even it is limited to the evaluation of the optimal electrodes' voltages [7]–[10]. This fact is due to the high computational cost requested by the evaluation of the device efficiency (the fitness function) which requires both electromagnetic simulation and the tracing of electron trajectories inside the device. A further obstacle resides in the high number of parameters from which the fitness function depends. In fact, the performance of MDC is mainly related to the geometry of the electrodes, to their voltages and to applied focusing magnetic field. Whereas the voltages and the applied fields can be modified even after the construction of the device during the calibration phase, the choice of a functional geometry is a much more complicated task.

To optimize the geometrical parameters of multistage collectors, simulated by means of an FE collector and of an electron gun simulator COLLGUN [6], a package, that, in its last release, includes a parametric geometric descriptor, an unstructured mesh generator and a 3D FE-based Vlasov solver.

MDC Geometry Optimization and FE Analysis for Shape Optimization

The algorithms devoted to the shape optimization requires the combination and the interaction of several modules: the preprocessor for the geometry description and the mesh generation, the FE coupled problem solver, the postprocessor for the collector efficiency evaluation (the fitness function), and the optimizer, which is here performed by MeTEO. The optimization acts on geometric characteristics and thus it must be coupled with a geometric descriptor and a mesh generator. In order to ensure the proper optimization process, that involve functional and feasible geometries, we must consider both the geometric constraints and the procedures of geometry description that reduces the dimensionality of the problem, theoretically infinite. To address this problem we make use of simple geometric solids for the description of each stage, which constitute the entire device.

In particular we employ the principles of the Constructive Solid Geometry (CSG), which describe objects in terms of primitive (limited primitive solids) correlated through Boolean operators. Each primitive can be represented by a complex object consisting of a set of vertices, segments, and faces which define polygonal regions (piecewise linear complex, PLC) and consequently all the operations among primitives (fusion, intersection, addition, subtraction, etc.) can be easily performed by using these PLC representations. Consequently CSG allows us to describe complex geometry usually adopted for MDCs by using a small number of primitives. For example the two-stage TWT collector, shown in Fig. 3.9 can be represented by combining just 13 primitives. The primitives currently implemented in our geometric descriptor are: Cone, Sphere, Truncated Cone, Cylinder, Helix, Grids, Calottes, Ellipsoids, etc. Once the formal description is complete, the specified primitives are automatically assembled according to the CSG, and from the PLC representation of the geometry a Delaunay 3D mesh is generated. In order to be used in the shape optimizer, the primitives (defining each stage) and the Boolean operations are listed in a file, which is the input of the mesh generator. In this file some entries are used as optimization parameters and are modified by the routines of the MeTEO code. The coupled electromagnetic-motional problem inside the collector region is governed by the Vlasov equation, coupled with Maxwell equations. This system of coupled equations firstly considers the distribution of charged particles, solution of the motion equation (Vlasov equation) governed by the electromagnetic field. Next, we take into account the self-consistent electromagnetic field, i.e. the solution of the Maxwell equations in which all field sources are assumed to be the charged particles present into the tube. A variety of methods can be used for solving this coupled problem, but only few of these are suitable for an optimization framework. Among these the Particle-in-Cell (PIC)[26] steady-state approach is surely the most appropriate for this purpose. In PIC the electron beam is represented by a reasonable number of macro-particles, subject to the dynamic equations and also sources of electromagnetic field.

This algorithm consists of a main loop starting with the solve and update steps. Once the field is evaluated, the particle tracking algorithm starts. In the present steady-state model, the particles are launched according to an emission rule or injection rule. They move forward until each of them encounters one geometric boundary. The trajectories are also used in the distribution step for the evaluation of charge and current density used as source terms of the new field problem, and so on. These steps are repeated until the "distance" between two consecutive solutions becomes lower than a user-specified end-iteration tolerance. In this situation, a fixed point for the solution is approached and electromagnetic field distributions can be assumed self-consistent [10]. After the solution of the self-consistent problem the FE simulator evaluates the current recovered by each electrode and also the back-streaming current (due to secondary electrons emitted by electrodes' surface) in order to evaluate the total power recovered and the collector efficiency. In particular the latter is used as fitness function and is estimated from the ratio between the total power recovered by all the stages and the spent beam power (the power of the electron beam entering the collector).

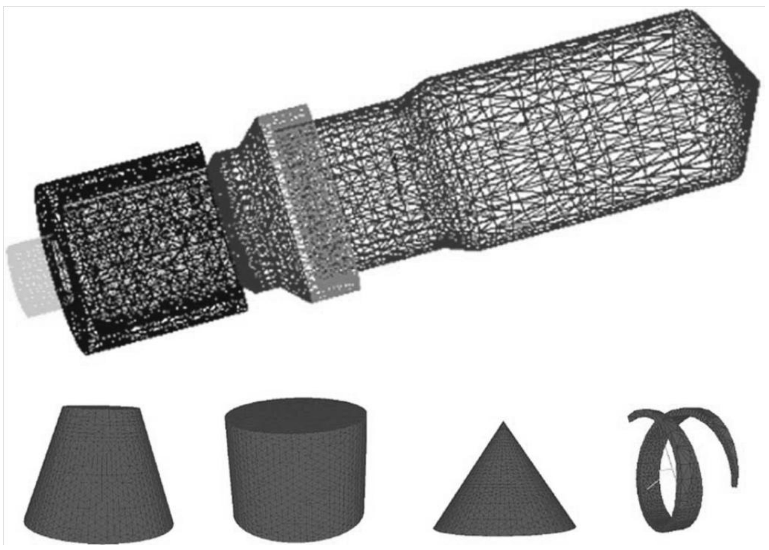


Fig. 3.9: Some primitives used for geometrical description and a two stages collector obtained by combining cylinders, cone and truncated cones.

It is worth noticing that in this optimization we have used data related to a TWT, which uses a tilted electric field (TEF) collector and whose characteristics are reported in literature [17]. The goal of the optimization is the increasing of the efficiency of the exiting device, which is about 73%. For this reason we chose in our tests an initial geometry for MDC different from the TEF collector. In particular for this shape optimization problem a limited set of parameters has been used: the length and the diameters (inner/outer) of the first stage and the outer diameters of the second stage. Each stage is easily obtained by using the CSG starting from 3 primitives: cylinder, truncated cone and cone. For example, the two configurations shown in Fig. 3.10 are obtained by using the minimum and maximum admissible values for the length (10 mm) and the radius (5 mm) and represent the borderline cases. The other data used to carry out the optimization are related to the electron beam entering in MDC, also available in literature [17]. This beam has reference voltage of 4.8 kV, a radius of 0.63mm, a current of 53.2 mA and carries out a power of about 190 W. The voltages of the two electrodes are assigned to 1/2 and ¾ of the reference voltage of the beam, i.e. 2.4 kV and 3.6 kV respectively. In the COLLGUN simulator this beam is represented by means of a ballistic model, according to which the total current is assumed uniformly distributed and the cross section of electron beam is modeled by using 25 rings, each of them divided in 10 macro-electrons. For each fitness function evaluation (collector efficiency) an irregular mesh of first order tetrahedra is generated, using a more refined mesh in the inter-electrode regions, where a more intense non-uniform electric field is expected. The typical FE simulation data for the steady-state coupled problem solved by COLLGUN are summarized in Table 10. Starting from random initial geometries (with an efficiency values lower than 75%), after 100 MeTEO iterations (10 FSO, 30 PSO, 60 BCA) leads to the optimized geometry, shown in Fig. 5 together with computed trajectories, having an efficiency of 84.8%. Very similar results were achieved taking the same test starting from different initial geometries. The computational time employed to perform this task is about 24 hours. It is worth noticing that optimizing the electrodes' voltages this geometry can reach efficiency value over 88%. In addition, in order to compare MeTEO performance with another optimizer, usually adopted in collector optimization [12], several runs of a random walk optimizer has been executed. After over 1500 FE simulations (we launch 5 times the random walk optimizer, running for 300 steps) the best result obtained is collector efficiency lower than 82%. This result confirms that the MeTEO algorithm, thanks to the combination of the three heuristics (FSO, PSO and BCA), having different capabilities in exploration and in local search, allows to obtain good results even after a moderate number of iterations.

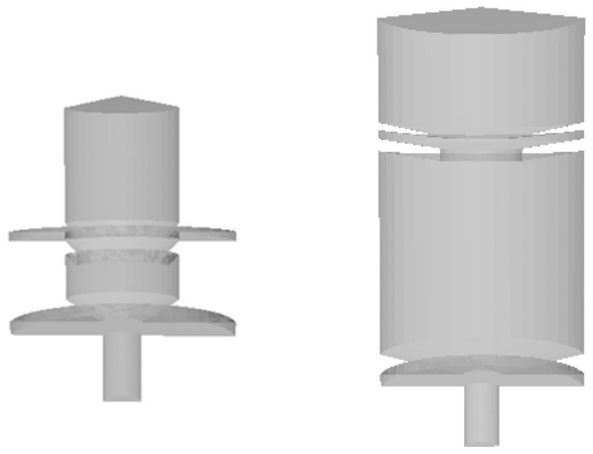


Fig. 3.10: Sections of the two collector geometries, obtained by using the minimum and maximum admissible values for the parameters.

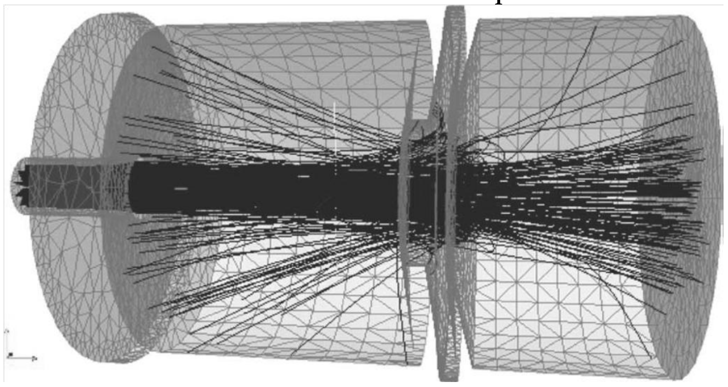


Fig. 3.11: Plot of trajectories in MeTEO optimized MDC geometry.

3.4.2 TWT magnetic focusing structure optimization [28].

The focusing magnetic field of aTWT is usually obtained by using a periodic permanent magnet (PPM) structure, which consists of a quasi-periodic sequence of toroidal permanent magnets with some differences in their inner and outer radii and in their length in such a way to obtain the desired intensity and profile for the on-axis magnetic field [18]. Usually in TWT simulator, when numerical solutions for this kind of structure are unavailable or too complicated, ESMs are employed in order to obtain approximate 3D field distributions by minimizing the error with respect to experimental on-axis values[19][20][21]. In general in an ESM, the magnetic field B is expressed as a function of some geometrical parameters (inner and outer radius, length, etc.), and of some other physical parameters strictly connected with the source intensity. Assuming a cylindrical coordinate frame (r, z, θ) , such that the z-axis is the axi-symmetry axis, the only not zero magnetic field component for $r = 0$ is B_Z depending only on the z coordinate:

$$B_z(z) = \sum_k F_k(z, z_k, R_{ik}, R_{ok}, ..., L_k, M_k) \tag{3.10}$$

where $z_k, R_{ik}, R_{ok}, L_k, M_k$ are respectively the z-position, the inner radius, the outer radius, the length and the strength of the k-th magnetic element of the periodic structure, whereas the F_k is a generic function depending on the adopted model. Other parameters, such as the width, the thickness, etc. can be considered in equation (3.10) according to the model complexity. In literature the ideal loop, the thin solenoid, the full coil, both in the single and pair configurations have been successfully used for building complex representations. As it has been shown in [19] any arbitrary focusing field profile can be represented by using ESM and from the knowledge of ESM the PPM structure can be synthesized. Following this approach we can use an ESM to represent the focusing magnetic field and modify its parameter in order to optimize the device performance.

Hereafter, an example of application is presented, regarding the optimal design of the focusing magnetic field applied to a typical TWT two stage collector, shown in a 2D view together with trajectories and optimized focusing magnetic field in Figure 3.12. The electron beam data are available in literature: it has reference energy of 4.8 kV, a radius of 0.63 mm, a current of 53.2mA and carries out a power of about 190W [22]. The voltages of the two electrodes are fixed to 2.4 and 3.8 kV. This beam represented is by means of a ballistic model, according to which the current is radially distributed in 50 rings, each of them divided in ten macro-electrons, for a total number of 500 macro-particles. An irregular tetrahedral mesh of about 20,000 first order elements is used by the COLLGUN simulator for the electromagnetic analysis. For the representation of the magnetic focusing structure an ESM based on ideal loops is adopted in the optimization and three parameters of a single loop (radius, position and current carried) are used. As fitness function f, we use a combination of the collector efficiency, h, and of the backstreaming current in mA, I_{back} , and exactly:

$$f\left(N,[Z],[R_i],[R_o],[L],[M,...]\right)=1.0-\eta+I_{back} \tag{3.11}$$

where N is the number of ESM, and $[Z],[R_i],[R_o],[L]$ and M are respectively the vectors of z-axis positions, inner radii, outer radii, lengths and intensities of ESMs. COLLGUN takes about 3 min in these conditions to evaluate the fitness function. Starting from a configuration without focusing magnetic field, which gives a collector efficiency equal to 81 percent and a backstreaming current over 1 mA, after 300 MeTEO iterations (100 FSO, 100 PSO, 100 BCA, that is about 24 h of computing time) we obtain a focusing magnetic field which gives an efficiency equal to 83.1 percent and a backstreaming current smaller than 0.2 mA.

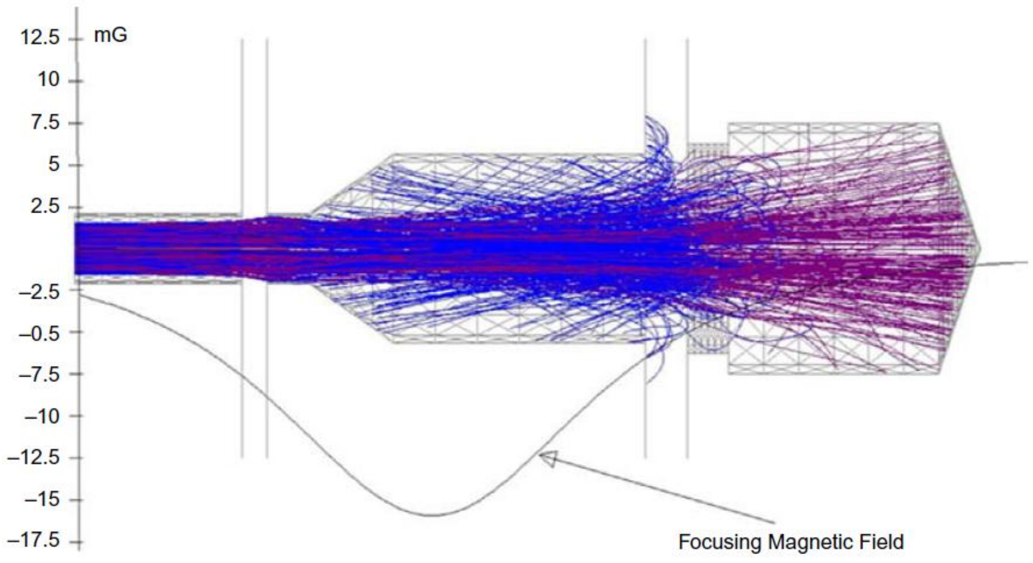


Figure 3.12: 2D view of the two stages collector, together with optimized focusing magnetic field and the resulting trajectories, blue trajectories represent electrons collected by the first stage, whereas the violet one represent those trajectories collected by the second stage.

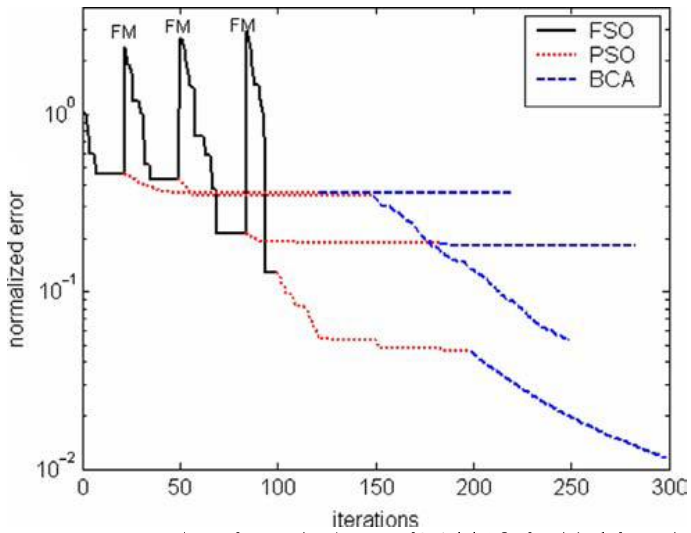


Figure 3.13: The convergence plot of a typical run of MeTEO for bird function, showing the different roles played by each heuristic: the FM labels indicate the fitness modification for the FSO heuristic.

3.4.3 Optimization of Multistage Depressed Collectors [10]

This section presents the application of MeTEO to the optimization of the electrodes' voltages of a typical two stages MDC for Traveling Wave Tubes (TWT) in order to increase

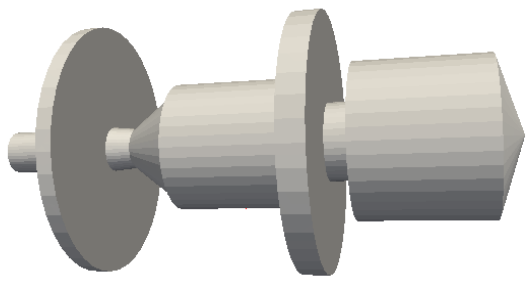


Figure 3.14: 3-D view of the two-stages depressed collector geometry used in the test performed.

its efficiency. As above discussed, the electromagnetic analysis of the trajectories inside the collector is performed by the FE code COLLGUN, which evaluates at every launch the collector efficiency, that is the fitness function of our optimization problem. We have used in our tests a geometry for MDC, shown in Figure 3.14.

Table 10: COLLGUN parameters used for the simulation of the two-stages collector.

Number of tetrahedral	About 20000
Number of node	About 600
Number of macro-particles	500
End Tolerance	0,05%
Number of iterations for each simulation	4-6
Computing time of each simulation	About 1'30"

The other data used to carry out the optimization are related to the electron beam entering in MDC, also available in literature [22]. This beam has reference voltage of 4.8 kV, a radius of 0.63mm, a current of 53.2 mA and carries out a power of about 190 W. For each fitness function evaluation (collector efficiency) an irregular mesh (shown in Figure 3.13) of first order tetrahedra is generated, using a more refined mesh in the inter-electrode regions, where a more intense non-uniform electric field is expected. The typical FE simulation data for the steady state coupled problem solved by COLLGUN are summarized in table 10.

The tTotal number of tetrahedra is close to 20000 elements, while the nodes are almost 6000. For these values, the computation time of each run of the solver COLLGUN is about 90 seconds, and the number of iterations is between 4 and 6. Being the MeTEO code written in MATLAB, a batch code has been developed on purpose to directly link the MATLAB with COLLGUN simulator. MeTEO has been launched 30 times and hereafter we report the information of the best value found. A random initialization has been done for FSO, whereas for the PSO and the BCA the initialization depends from the point returned by the previous algorithm. The number of individuals used for FSO/PSO is 10, whereas a single bacterium is used for BCA. Performed tests have shown that a greater number of individuals does not improve significantly the performance of MeTEO for this problem since more individuals require more evaluations of the fitness function, and this has a remarkable computational cost. In fact, in our problem, the time spent to compute the cost function is that one coming out from the launch of the simulator COLLGUN, that is about 90 seconds. The best results obtained by MeTEO are summarized in Table2. The high value of efficiency found for this two-stages collector was 88.81%. It is worth noticing, as clearly shown in the table 12, that the efficiency found by FSO gradually increases passing through the PSO and after to the BCA, confirming the functioning of the MeTEO. In fact, the foundation of the hybridization done in MeTEO lies exactly in this characteristic.

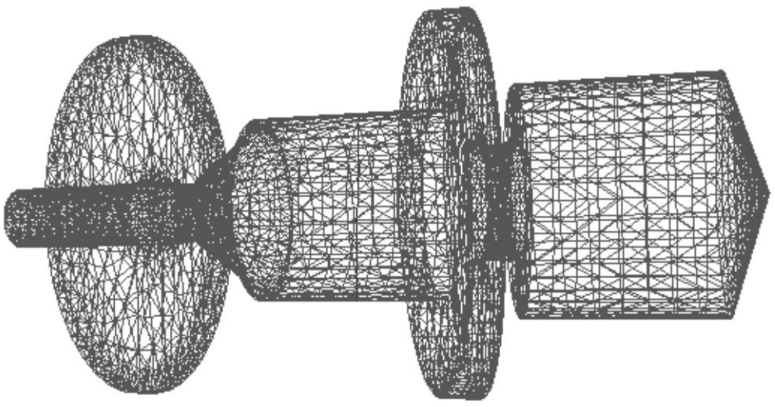


Figure 3.13: 3-D view of the boundary of the mesh used

The PSO performs an improvement of the efficiency (about 0.3%), because it is launched in a sub domain space. The application of the BCA refines the solution found improving it of 0.12%. Table 13 reports the voltage and the resulting current for each stage and efficiency found for the two-stages collector.

Table 12: Best values of efficiency of the two-stages collector obtained over all the tests by the various components of MeTEO.

	FSO	PSO	BCA
Efficiency	88.17%	88.64%	88.81
Number of iterations	40	20	20
FM	3		

Table 13: Best configuration values obtained for the two-stage collector considered.

Spent beam power	192W
Power recovered	170W
Collector's efficiency	88.81%
Stage 1	
Potential	-3.05kV
Current	38.73 mA
Stage 2	
Potential	-4.22 kV
Current	12.34 mA

Comparison between MeTEO and Random Walking algorithm

Furthermore, in order to compare MeTEO performance with another optimizer, usually adopted in collector optimization [23], several runs of a Random Walk (RW) optimizer have been executed. The RW, introduced by Karl Pearson in 1905 [24], is a mathematical formalism used to describe a path built by successive steps in random directions. For example, the path traced by a molecule that travels in a fluid or the price of a fluctuating stock can be modeled in this way. The RW algorithm applied to the optimization of the two-stages collector provides over 30 launches the best value listed in Table 14, which is worse than the one found by MeTEO.

Table 14. Best value (over 30 launches) of efficiency of the two-stages collector obtained by using Random Walking Algorithm.

Iterations	Efficiency
200	87,42%

For the comparison between MeTEO and RW an important remark regards how many times the fitness function is computed in the whole optimization process. Assuming 40 steps for FSO, 20 for PSO and 20 for BCA (all with 10 individuals) we have in total 600 fitness evaluations for each run of MeTEO, whereas each run of RW (10 individuals) employs 2000 fitness function evaluation. Therefore the hybridization proposed in MeTEO, with FM, is able to improve the optimization process. This result confirms that the MeTEO algorithm, thanks to the combination of the three heuristics (FSO, PSO and BCA), having different capabilities in exploration and in local search, allows to obtain good results even after a moderate number of iterations.

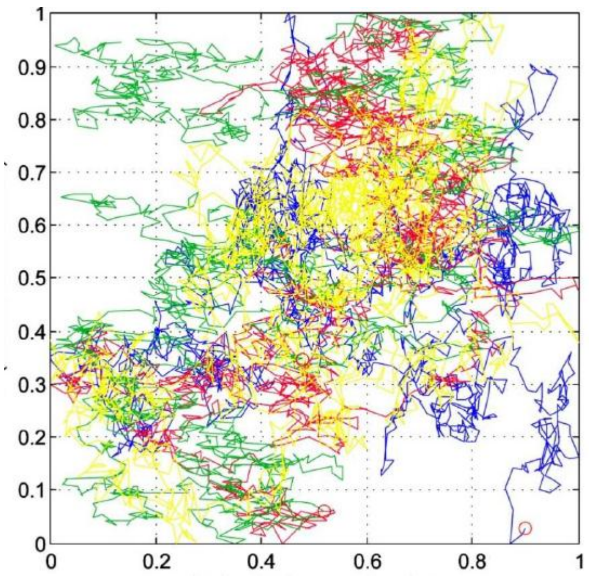


Figure 3.14: Trajectories of Random Walk individuals, axis x and y are normalized to 1.

3.5 Validation of FPSO

In this section is presented an inverse problem for validating the FPSO: the harmonic estimation problem. The harmonic estimation in a power system is becoming important to guarantee power quality, especially in the actual electrical system in which there are many different nonlinear types of equipment. Therefore different approaches and algorithms have been employed in order to identify and estimate the harmonic components due to the distortion introduced by such nonlinear loads. The Harmonic Estimation Problem consists of extrapolating amplitudes and phases from a distorted signal composed by a fundamental frequency and a determinate number of harmonics. The DFT and the Kalman filter approach have been widely used but they require the knowledge of both the state matrix and the statistics of the electrical signal. The evolutionary techniques have been recently employed to solve the harmonic estimation problem with success due to their capability to exploit exploration. In [29] and [30] are presented the well known Particle Swarm

Optimization (PSO) and Genetic Algorithm (GA). In the both works [29][30], hybrid approach with an evolutionary algorithm and the Least Square Algorithm (LS) is implemented. In particular, the evolutionary algorithm (GA or PSO) is employed to perform the identification of the phases, and the LS for the amplitude of all harmonics carried in the distorted signal.

To validate the proposed algorithm it has been employed a test signal already used in literature by [29] and [30]. In particular it is a distorted voltage signal taken from a two-bus three phase system with a full-wave six-pulse bridge rectifier at the load bus. Three tests are presented, respectively under no noise, and under the addition of noise with SNR equal to 10dB and 20dB. The total number of individuals is 10 and it remains constant for all the estimation process. Setting N the number of harmonics that must be computed, the number of parameter that must be estimated will be equal to $N+1$. The first parameter is the fundamental frequency, and the other N parameters are the phases of each harmonic. There is no need to add other parameters for the frequencies of each harmonic because they derive from the fundamental frequency. As mentioned before, the other N parameters are always the phases of the harmonic. The value of proportion P has been set to 0.5, thus the number of individuals for P_{sub} and F_{sub} in the mixed-configuration mode is simply 5. All parameters have been normalized between $[0;1]$, and the parameters used for the algorithm are: $[\omega=1; c_1=0.005; c_2=0.01; c_2=0.3]$. The percentage error has been employed to evaluate the performance, following the formula:

$$PE=\frac{\sum_{k=1}^{K}\left[Z(k)-\hat{Z}(k)\right]^2}{\sum_{k=1}^{K}Z(k)^2}\times 100 \tag{3.12}$$

Table 15:Best Results over 50 launches no noise

Test Signal $Z_0(t)$			FPSO (best $PE=1.738286\times10^{-7}$)		PSO (best $PE=1.491185\times10^{-6}$)	
Harmonic Order	Amplitude (p.u.)	Phase	Estimated Amplitude	Estimated Phase[°]	Estimated Amplitude	Estimated Phase[°]
Fundamental (50 Hz)	0.95	-2.02	-0.950004	-2.002	0.949896	-1.979
5th (250Hz)	0.09	82.1	0.090000	82.235	0.089946	81.957
7th (350 Hz)	0.043	7.9	0.043007	8.116	0.043006	7.929
11th	0.03	-147.1	0.029994	146.	0.029982	1.474
13th	0.033	161.6	0.030165	160.	0.064514	-1.712

Since the benefit of using evolutionary and swarm algorithms has already been verified, we concentrate the comparison between the improvements that FPSO approach can produce versus standard PSO. Finally, a statistical analysis consisting of 50 launches for each benchmark proposed is presented. All code has been developed in Matlab© code.

For this test the signal reported in [30] has been used and the relative amplitude and phases are listed in table (16). In particular it consists of 5 harmonics: 5th, 7th, 11th and 13th. In table(16) are reported the best results, for the no noise case; are also listed the amplitude and the phases for all harmonic components estimated. As it is possible to see, the percentage error reached by FPSO is lower than PSO. In the table (17)(18)(19) are listed the best results, all relative to over 50 launches when a Gaussian noise is added to the original waveform. For concerning the best result over 50 launches, FPSO shows good results with no noise and with SNR=20dB, whilst in the case of SNR=10dB PSO is able to reach a better percentage error. In literature has always been showed the best results obtained after applying of a particular algorithm, but it was never done a statistical analysis. In fact, to understand the data reported in the tables (17)(18)(19) we need to cross them with the mean and variance values collected in each test and arranged in the table (19). With this new information we can say that the FPSO performs on average better than PSO. Nevertheless, even if the PSO could reach best results, but over many launches this possibility decreases; instead the FPSO behavior allows us to reach always a good solution. For these reasons the best practice is to implement a hard hybridization of FSO and PSO, restructuring the logical framework, modifying matrix interconnection and leaving unchanged the number of individuals.

Table 16:Best Results over 50 launches 10dB noise

Test Signal $Z_0(t)$		FPSO (best $PE = 1.307165\times10^{-4}$)	PSO (best $PE = 2.384338\times10^{-5}$)
Harmonic Order	Amplitude (p.u.)	Estimated Amplitude	Estimated Amplitude
Fundamental (50 Hz)	0.95	0.803157	0.802465
5th (250Hz)	0.09	0.162620	0.166800
7th (350 Hz)	0.043	0.205393	0.209827
11th	0.03	0.212233	0.218075
13th	0.033	0.293665	0.292187

Table 17:
Statistical analysis
(mean and variance relative to the Percentage error calculated over 50 launches)

	FPSO		PSO	
	Mean	σ^2	Mean	σ^2
No noise	0.05706	0.02940	0.60988	0.25422
SNR=10dB	0.53598	0.05374	1.94380	0.14269
SNR=20dB	0.04752	0.06919	0.28319	0.87133

Table 18:Best Results over 50 launches 20dB noise

Test Signal $Z_0(t)$		FPSO (best $PE = 3.361186 \times 10^{-4}$)	PSO (best $PE = 3.390877 \times 10^{-4}$)
Harmonic Order	Amplitude (p.u.)	Estimated Amplitude	Estimated Amplitude
Fundamental (50 Hz)	0.95	0.944501	0.944771
5th (250Hz)	0.09	0.081102	0.081046
7th (350 Hz)	0.043	0.028676	0.028796
11th	0.03	0.027595	0.027454
13th	0.033	0.051525	0.058900

References

[1]Gordon G. Lai, Chien-Feng Yang, Hunter M. Huang, and Ching-Tzong Su, "Optimal Connection of Power Transmission Lines With Underground Power Cables to Minimize Magnetic Flux Density Using Genetic Algorithms" IEEE Transaction on Power Delivery, vol. 23, no. 3, JULY 2008

[2]Canova, A. and Giaccone, L. (2009), "Magnetic field mitigation of power cable by high magnetic coupling passive loop", 20th International Conference and Exhibition on Electricity Distribution, 1-4, Prague, Czech Republic, Jun. 2009.

[3]Clerc, M. (2004), "Discrete particle swarm optimization, illustrated by the traveling salesman problem", B. V. Babu & G.

[4] K. Schittkowski, Report, Department of Computer Science, University of Bayreuth, 2004. Available at http://www.ai7.uni-bayreuth.de/mc_ode.htm

[5] F.R. Fulginei and A. Salvini, Soft computing for identification of Jiles-Atherton model parameters, IEEE Trans. Magn. 41(3) (2005), pp. 1100–1108.

[6] S. Coco et al., "COLLGUN: A 3D FE simulator for the design of TWT's electron guns and multistage collectors," in Scientific Computing in Electrical Engineering, Mathematics in Industry. Berlin, Germany: Springer-Verlag, 2006, vol. 9, pp. 175–180.

[7] T. K. Ghosh and R. G. Carter, "Optimization of multistage depressed collectors," IEEE Trans. Electron Devices, vol. 54, pp. 2031–2039, 2007.

[8] J. Petillo et al., "Design and optimization electron guns and depressed collectors using theMICHELLE code in the ANALYST environment," in Proc. IEEE Int. Vacuum Electronics Conf. (IVEC2007), 2007, pp. 1–2.

[9] K. R. Vaden, J. D. Wilson, and B. A. Bulson, "A simulated annealing algorithm for the optimization of multistage depressed collector efficiency,"in Proc. 3rd IEEE Int. Vacuum Electronics Conf. (IVEC 2002),2002, pp. 164–164.

[10] S. Coco et al., "Optimization of multistage depressed collectors by using FEM and METEO," presented at the IV Italian Workshop The Finite Element Method Applied to Electrical and Information Engineering,Rome, Dec. 13–15, 2010.

[11] F. Riganti Fulginei and A. Salvini, "Comparative analysis between modern heuristics and hybrid algorithms," Compel, vol. 26, pp. 264–273, 2007.

[12] G. Pulcini, F. Riganti Fulginei, and A. Salvini, "Metric-topological- evolutionary optimization," in Inverse Problems in Science and Engineering. London: Taylor and Francis, 2011, pp. 1–18, 10.1080/17415977.2011.624624.

[13] F. R. Fulginei and A. Salvini, "Hysteresis model identification by the flock-of-starlings optimization," Int. J. Appl. Electromagn. Mechan., no. 3–4, pp. 321–331, 2009.

[14] H. J. Bremermann, "Chemotaxis and optimization," J. Franklin Inst., Special Issue: Math. Models Biol. Syst., no. 297, pp. 397–404, 1974.

[15] S. Coco, A. Laudani, G. Pollicino, and P. Tirrò, "A new self-consistent 3D unboundedmagnetic field FE computation for electron guns," IEEE Trans. Magn., vol. 46, no. 8, pp. 3425–3428, 2010.

[16] F. Glover, "Tabu search methods in artificial intelligence and operations research," ORSA Artif. Intell., vol. 1, no. 2, p. 6, 1987.

[17] L. Kumar, P. Spatke, R. G. Carter, and D. Perring, "Three-dimensional simulation of multistage depressed collectors on microcomputers," IEEE Transactions on Electron Devices, Vol. ED-42, pp. 1663-73.

[18] Sterrett, J.E. and Heffner, H. (1958), "The design of periodic magnetic focusing structures", IRE Transactions on Electron Devices, Vol. ED-5, pp. 35-42.

[19]Coco, S. and Laudani, A. (2007), "An iterative procedure for equivalent source representation of focusing magnetic fields in TWT", COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering, Vol. 26 No. 2, pp. 317-26.

[20]Vaughan, J.R.M. (1972), "Representation of axisymmetric magnetic fields in computer programs", IEEE Transactions on Electron Devices, Vol. 19 No. 2, pp. 144-51.

[21]Jackson, R.H. (1999), "Off-axis expansion solution of Laplace's equation: application to accurate and rapid calculation of coil magnetic fields", IEEE Transactions on Electron Devices, Vol. 46, pp. 1050-62.

[22] Kumar, L., Spatke, P., Carter, R.G. and Perring, D. (1995), "Three-dimensional simulation of multistage depressed collectors on microcomputers", IEEE Transactions on Electron Devices, Vol. ED-42, pp. 1663-73.

[23] Ghosh, T. K. and Carter, R.G. (2007) "Optimization of Multistage Depressed Collectors", IEEE trans. on electron devices, vol. 54, pp. 2031-2039.

[24] Pearson, K. (1905). The problem of the Random Walk. Nature. 72, 294.

[25] Salvatore Coco , Antonino Laudani , Giuseppe Pulcini , Francesco Riganti Fulginei , and Alessandro Salvini "Shape Optimization of Multistage Depressed Collectors by Parallel Evolutionary Algorithm" IEEE Transaction on Magnetics, Vol. 48, No. 2, February 2012 435

[26] S. Coco, A. Laudani, G. Pollicino, "A 3-D FE Particle-in-Cell Parallel code with adaptive load balancing", COMPUMAG 2009, Florianopolis, Brasile, 22-16 Novembre 2009.

[27] D. Altomonte, A. Laudani, G. Pulcini, A. Salvini, F. Riganti Fulginei "Swarm-based algorithms for the Minimization of the Magnetic Field of Underground Power Cables" SCEE2012 Scientific Computing in Electrical Engineering Zurich, September 11-14, 2012

[28] Salvatore Coco, Antonino Laudani, Giuseppe Pollicino, Giuseppe Pulcini, Francesco Riganti Fulginei, Alessandro Salvini "TWT magnetic focusing structure optimization by parallel evolutionary algorithm", COMPEL Emerald Vol. 31 Iss: 5 pp. 1338 – 1346, 2012

[29]Maamar Bettayeb and Uvais, A Hybrid Least Squares-GA-Based Algorithm for Harmonic Estimation, IEEE Transactions on Power Delivery, VOL. 18, NO. 2, APRIL 2003.

[30]Q. H. Wu,W.-K. Chen, Optimal Harmonic Estimation Using A Particle Swarm Optimizer. IEEE Transaction on Power Delivery, VOL. 23, NO. 2, APRIL 2008

[31]Haili Ma, Adly A. Girgis, Identification and Tracking of Harmonic Sources in a Power System using Kalman Filter". IEEE Transactions on Power Delivery, Vol. 11, No. 3, July 1996

Chapter 4

Towards Swarm Circuit.

In addition to all the previously described efforts to improve numerical swarm-based algorithms, other authors have proposed in literature several different attempts to treat the swarm-based numerical algorithms as continuous dynamical systems. For example, important contributions are in [27], [31]-[33]. Although the idea to treat swarm-algorithms as continuous systems it is already in the state-of-the-art, the way that is proposed this chapter is quite different with respect to literature since it provides an equivalent analog swarm-circuit able to perform the optimization and the inverse problems by using collective behaviour.

We call swarm circuits those electrical circuits in which the cinematic characteristics of the trajectories followed by the swarm members are reproduced in terms of waveforms related to voltages measured at the terminal of capacitors and currents measured at the terminals of inductors. Thus, this approach is twofold: firstly, we can easily adopt all the methods shown by the Theory of Circuits for analyze the cinematic of the swarm members, i.e. the Laplace Transform, Tableau method and so on; and we can lay the theoretical foundations for designing of innovative hardware for real-time optimizations.

Using of Swarm Based Algorithms for solving optimization problems represents a consolidated computational approach. Since 1995, when J. Kennedy and R. Eberhart introduced the Particle Swarm Optimization (PSO) [1], this kind of algorithms have been employed for solving many different problems in many subjects such as: economic dispatch, design of arrays antenna, training neural networks. The real engineering problems are often non linear, in this scenario swarm-based algorithms are able to recognize the better solution, even in those problems in which the cost function is multimodal, but, despite of that, they suffers of long execution time. For these reasons, some authors have implemented a hardware implementation to take advantage from the proprieties of swarm-based algorithm in a real-time optimization scenario (RTO problems). Literature offers many different approaches to implementing heuristics in a hardware environment, many of them regard genetic algorithm such illustrated in [39][40]. Just a few works treat the PSO implementation [41][42][43][44]. All the listed approaches suggest, in general, a hardware implementation through DSP or microcontroller, which perform the algorithm process and manage all the interfaces with possible sensors. Many of them employ a parallel architecture and FPGA hardware. In this chapter, a new point of view of the problem is proposed; in fact, recently, a new continuous PSO algorithm has been proposed, in which the discrete equation have been modified by the Euler's formula found out a differential equation of velocity for each particle. This way has been followed by Brandstätter [46].

An analogy with a mass-spring mechanical system has been made; but in this contest we will address the system as RLC series circuit, that shows the similar mathematical form. Hence, starting from the numerical velocity rule of a swarm-based algorithm, the continuous equation can be carried out; it represents the state equation of the algorithm. Afterwards, comparing the state equation with the state equation that describes an RLC series, the parameters correspondences between the numerical algorithm and the values of the RLC components are obtained. Furthermore, in this procedure the global best and the personal best become voltage controlled generators. The fitness evaluation of each particle, and the calculus of global best and personal best, is delegated to an external microcontroller, that is not treated in this thesis. In the next few paragraph, the mathematical basis of a hardware implementation of a generic swarm based algorithm has been shown. Furthermore, a complete validation by means of both inverse problems and classical optimization benchmark have been made.

4.1 Swarm-Based Numerical Algorithms

The used Flock-of-starling optimization algorithm can be described by the following pseudo-code that can be seen as an extension of the PSO algorithm. Let us to summarize as said before in the chapter 2, by the following pseudo-code valid for a generic fitness function, $f(x_1 ... x_D)$, to be minimized in the search space R^D having dimension D:

1. Define:

- Dimension of the search space, $\mathrm{R}^{\mathrm{D}} \equiv (x^{<1>}...x^{<D>})$: $x^{<j>\min} \le x^{<j>} \le x^{<j>\max}$ with $j = 1...D$;
- the values of the main parameters for each k-th particle:
- Inertial coefficient ω, and its maximum value $\omega_{\max}$;
- Cognitive coefficient λ, and its maximum values, $\lambda_{\max}$;
- Social coefficient γ, and its maximum value γ_{MAX} ;
- Maximum number of iterations, $T_{\max}$;
- Fitness function $f(x_1...x_D)$;
- Maximum value of each velocity component $V_{\max}$;
- Iteration counter t ;
- Initialization of velocities $v^j{}_k(t=0) = random(0,1) \cdot V_{\max}$;
- Initial position $\left(x_k^{<1>}(0)...x_k^{<D>}(0)\right)$ of each k-th particle
- Initial personal fitness $f_{p_j}(t=0)$;
- Initial global fitness $g(t=0)$;
- Fitness threshold $goal\_fitness = arbitrary\, small$;
- n_{birds} (from now on we use the term birds instead of particles) is the total number of birds into the flock;
- The topological coefficient, h;
- Number of birds into the flock controlled by one single bird, N_{ctrl_birds} (topology rule). At each k-th bird is associated a group composed of randomly chosen N_{ctrl_birds} of other members of the flock;

2. For each k-th birds, for each step (integer) t, with $t = 0,1...T_{MAX}$:

- evaluate the fitness $f_k(t) = f(x_k^{<1>}(t)...x_k^{<D>}(t))$;
- If $f_k(t)$ is better than the personal best fitness of the k-th particle $f_{p_k}(t)$, then assign current position as personal best position and update the personal best fitness:

$$p_{bestk}^{<j>} = x_k^{<j>}(t) \quad \forall j-th\ dimension \tag{4.1}$$

$$f_{p_k}(t) = f_k(t) \tag{4.2}$$

- If $f_k(t)$ is better than global best fitness, then assign current position as global best position and update the global best fitness:

$$g_{best}^{<j>}(t) = x_k^{<j>}(t) \quad \forall j-th\ dimension \tag{4.3}$$

$$g(t) = f_k(t) \tag{4.4}$$

▪ Update, for each k-th particle, the vector velocity components:

$$u_k^{<j>}(t+1)=\omega v_k^{<j>}(t)+\lambda[p_{bestk}^{<j>}(t)-x_k^{<j>}(t)]+\gamma[g_{bestk}^{<j>}(t)-x_k^{<j>}(t)]+\sum_{m=1}^{N}h_{km}u_m^{<j>}(t)\quad \forall j-th\ dimension \tag{4.5}$$

where $h_{km}=h$ if the *k-th* bird is controlling the *m-th* one, $h_{km}=0$ otherwise. Thus, we can say that (5) describes a swarm-algorithm.

Equation (5.5) differs from standard PSO just for the presence of the term: $\sum_{m=1}^{N}h_{km}u_m^{<j>}(t)$ taking into account the topological neighbor-velocity-matching from which each single bird controls the velocities of a fixed number of other members of the flock. The velocity-matching term strongly modifies the collective behaviour of the swarm/flock, as it will be shown in next sections.

▪ *Update*, for each k-th particle, the position:

$$x_k^{<j>}(t+1)=x_k^{<j>}(t)+u_k^{<j>}(t+1) \tag{4.6}$$

▪ *Update*, for each *k-th* particle, all parameters:

$$\omega=\omega_{\max}\cdot random(0,1),$$

$$\lambda=\lambda_{\max}\cdot random(0,1),$$

$$\gamma=\gamma_{\max}\cdot random(0,1).$$

Then, pose $t=t+1$, and repeat the procedure.

4.2 Swarm algorithms translated into dynamic systems

Let us consider equation (4.5), in particular, let us rewrite it, by introducing a value Δt as proposed in [31] for the standard PSO:

$$u_k(t+\Delta t)=\omega\ u_k(t)+\lambda\ [pbest_k(t)-x_k(t)]+\gamma\ [gbest(t)-x_k(t)]+\sum_{m=1}^{N}h_{km}u_m(t) \tag{4.7}$$

Equation (4.7) can be interpreted as the Euler forward numerical integration of a physical system of differential equations if we assume $\Delta t\in\mathrm{R}$ and $\Delta t\to 0$ as follows:

$$\frac{u_k(t+\Delta t)-u_k(t)}{\Delta t}=\frac{(\omega-1)u_k(t)+\lambda[pbest_k(t)-x_k(t)]+\gamma[gbest(t)-x_k(t)]+\sum_{m=1}^{N}h_{km}u_m(t)}{\Delta t} \tag{4.8}$$

In (4.8) the independent-variable is now considered as the variable time $t\in\mathrm{R}$, whereas in (4.5) t was a simple natural number counting the number of iterations. In the same way equation (4.6) can be rewritten as follows:

$$x_k(t+\Delta t)=u_k(t)\Delta t+x_k(t) \tag{4.9}$$

It is important to note that in (4.8) and (4.9) the value Δt plays also a dimensional role. In fact, in a physical system we have to multiply the velocity for a time to compute a displacement. Since we have recognized equation (4.8) as Euler's basic formula suitable for numerical integration, the value Δt must be set small enough for performing a proper numerical integration. Thus, an inappropriate choice of Δt (a value too large, e.g. $\Delta t = 1$) involves a "under-integration" of (4.8) and (4.9). It means that (4.7), (4.5), where $\Delta t = 1$, cannot be seen in one-to-one correspondence with a continuous physical system, but they can however inspire further developments that pass throw the use of (4.8) and (4.9) as follows. In fact, if we consider a small enough value $\Delta t \to 0$ it is immediate to rewrite (4.8) in the continuous as follows:

$$\frac{d}{dt}u_k(t) = \tilde{\omega}u_k(t) + \tilde{\lambda}[pbest_k(t) - x_k(t)] + \tilde{\gamma}[gbest(t) - x_k(t)] + \sum_{m=1}^{N}\tilde{h}_{km}u_m(t) \qquad (4.10)$$

In (4.10) new normalized parameters appear. They are defined as follows:

$$\begin{cases} \tilde{\omega} = \dfrac{(\omega - 1)}{\Delta t} \\ \tilde{\lambda} = \dfrac{\lambda}{\Delta t} \\ \tilde{\gamma} = \dfrac{\gamma}{\Delta t} \\ \tilde{h}_{km} = \dfrac{h_{km}}{\Delta t} \end{cases} \qquad (4.11)$$

where $\tilde{h}_{km} = \tilde{h} = h / \Delta t$ if the k-th bird is controlling the m-th one, $\tilde{h}_{km} = 0$ otherwise. It is evident from (4.11) that Δt is practically a new parameter to take into account for the correct translation from numerical algorithms (4.8) and (4.9) to continuous ones. Starting from equation (4.10) is useful also for understanding how both global and personal bests can be seen as excitation of the continuous system. In fact, let us assume as excitation of the dynamic system the quantity:

$$\Im_k(t) = \tilde{\lambda} \cdot pbest_k(t) + \tilde{\gamma} \cdot gbest(t) \qquad (4.12)$$

and finally, by posing

$$\tilde{\mu} = \tilde{\lambda} + \tilde{\gamma} \qquad (4.13)$$

the state equation (4.10) can be written for each k-th particle/bird as follows:

$$\frac{d}{dt}u_k(t) = \tilde{\omega}u_k(t) - \tilde{\mu}x_k(t) + \sum_{m=1}^{N}\tilde{h}_{km}u_m(t) + \Im_k(t) \qquad (4.14)$$

The state equation (4.14) must be coupled with a second state equation referred to the positions $x_k(t)$ appearing in (4.6) for the swarm-algorithm and translate in (4.9) by using the parameter Δt. Obviously, for $\Delta t \to 0$, we simply obtain the cinematic definition of velocity:

$$\frac{d}{dt} x_k(t) = u_k(t) \tag{4.15}$$

Equations (4.14) and (4.15) describe a dynamic system, evolving in the time domain, t, being forced by (4.12). Equations (4.14) and (4.15) are the state equations describing the continuous swarm-based system. Then, if we consider a flock made of N birds, the state-equation system has $2N \times 2N$ dimension and it can be written in compact form, for each j-th co-ordinate of the space solution as follows:

$$\frac{d}{dt}\begin{bmatrix} \mathbf{u} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{A_{1,1}} & \mathbf{A_{1,2}} \\ \mathbf{I} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \mathbf{u} \\ \mathbf{x} \end{bmatrix} + \begin{bmatrix} \mathbf{F} \\ \mathbf{0} \end{bmatrix} \tag{4.16}$$

The sub-matrix $\mathbf{I}$ appearing in (4.16) is the identity matrix having $N \times N$ dimension. The sub-matrix $\mathbf{A_{1,1}}$ is a square matrix having $N \times N$ dimension defined as:

$$\mathbf{A_{1,1}} = \mathbf{M} + \mathbf{H} \tag{4.17}$$

where the matrix $\mathbf{M} = \tilde{\omega} \cdot \mathbf{I}$, whereas $\mathbf{H}$ takes into account the neighbor-velocity-matching. The $\mathbf{H}$ entries are $h_{km} = \tilde{h}$ if the $k-th$ bird control the velocity of the $m-th$ one, zero otherwise. It is evident that the matrix $\mathbf{H}$ has not all zero entries just for continuous FSO, whereas it is the null matrix for continuous PSO since $\tilde{h}_{km} = \tilde{h} = 0$ (i.e., we can simply commutate from PSO to FSO, just considering the absence or the presence of $\mathbf{H}$ in equation (4.17)).

The last $N \times N$ sub- matrix appearing in (4.16) is:

$$\mathbf{A_{1,2}} = -\tilde{\mu} \cdot \mathbf{I} \tag{4.18}$$

Finally, the vector $\mathbf{F}$ has each k-th row-entry defined by $\Im_k(t)$ of equation (4.12).

4.3 Time-windowing

An emerging problem is how to consider the force (4.12) that takes into account both of the global and of the personal bests whose values are a priori unknown. Indeed, they should be evaluated dynamically according to the several fitness values which are monitored during the motion of the swarm/flock. On the other hand, for a correct translation of the numerical swarm-algorithm into the continuum we have to find a way for modeling these functions of the time. This apparent paradox can be solved if we assume to make a sampling of the time axis by means of a succession of time-windows, each of which having duration equal to a fixed value, τ. For each i-th time-window, we assume that the force (4.12) still keeps the value that has been evaluated at the previous time-window (guess values are assumed for the first time-window). Thus, taking into consideration a time-window that begins at the generic time t_{in} and ends at the time $t_{in}+\tau$, we will assume that the force is equal to a constant value $\Im_{k,i}(t_{in})$ for any time within that i-th window. Globally, we consider the complete solution of the continuous dynamic system as the union of all single solutions obtained from each single time-window.

4.4 The Swarm Circuit

Once one has written the state equations of the continuous swarm (4.14) and (4.15), they can easily be seen as equations governing an electrical circuit.

Equations (4.14) and (4.15) describe the state equation of a k-th circuital branch as shown in Figure 1 by posing the one-to-one correspondence between the current flowing in the k-th branch of the circuit and the velocity of the k-th member of the swarm-flock, i.e. $i_k(t) \propto u_k(t)$, whereas the voltage measured at the terminals of a suitable capacitor, $v_{C,k}$, can be assumed proportional to the displacement: $x_k(t) \propto v_{C,k}$. Obviously, by following this approach, the force (4.12) can be made proportional to the summation of the voltage values of two voltage-independent-sorces: $\Im_k(t) = e_k(t) + g(t)$, i.e. one taking into account personal best and the other one the global best. We have: $e_k(t) \propto \lambda \cdot pbest_k(t)$ and $g(t) \propto \tilde{\gamma} \cdot gbest(t)$.

Thus, let us assume the circuit-branches in Figure 4.1, where the ideal switches, play the role to make the time-windowing for time depending global-voltage source as well for personal voltage source. The values of $e_k(t)$ and $g(t)$ are evaluated TW-by-TW by the values computed on the cost-function to be minimized.

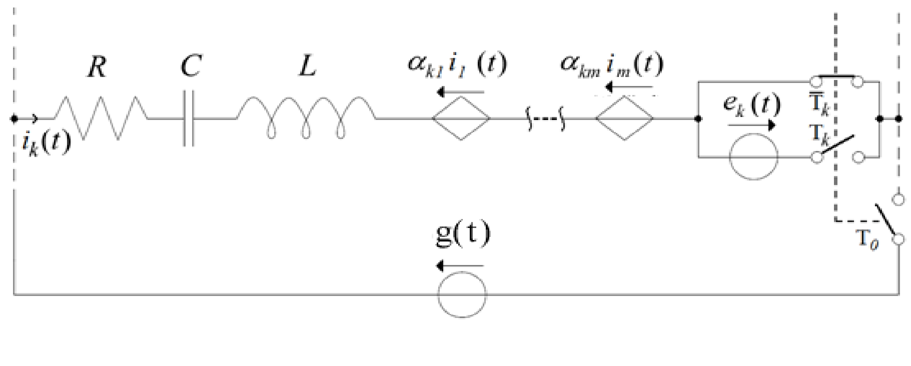


Figure 4.1: k-th circuital branch of the Swarm-Circuit and the global-voltage generator branch feeding all circuit branches

The equations governing the k-th ($\forall k = 1 \ldots N$) electric branch in Figure 4.1, are:

$$\frac{d}{dt} i_k = -\frac{1}{L}\left(R i_k + v_{C,k} + \sum_{j=1, j\neq k}^{N} \alpha_{k,j} i_k - e_k(t) - g(t) \right) \tag{4.19}$$

$$\frac{d}{dt} Q_{C,k} = i_k \tag{4.20}$$

In (4.20) we have introduced a new state variable electrical charge, $Q_{C,k} = C \cdot v_{C,k}$ just with the aim to have a perfect one-to-one correspondence among (4.14) - (4.15) and (4.19)-(4.20). As a consequence, the inductor current and the electric charge on the capacitor are the velocity and the position of the swarm members, whereas the voltages at the terminals of capacitors are still the positions but scaled by a factor C . Thus, by deriving (4.14) and inserting (4.15) and similarly by deriving the (4.19) and inserting (4.20) it is immediate to obtain the following comparison:

$$\begin{cases} \frac{d^2}{dt^2} u_k(t) = \tilde{\omega} \frac{d}{dt} u_k(t) - \tilde{\mu} u_k(t) + \sum_{m=1}^{N} \tilde{h}_{km} \frac{d}{dt} u_m(t) + \frac{d}{dt}[\tilde{\lambda} \cdot p_{best\,k}(t) + \tilde{\gamma} \cdot g_{best}(t)] \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \forall k = 1 \ldots N \\ \frac{d^2}{dt^2} i_k = -\frac{R}{L}\frac{d}{dt} i_k - \frac{1}{LC} i_k - \sum_{m=1, m\neq k}^{N} \frac{\alpha_{k,m}}{L} \frac{d}{dt} i_m + \frac{1}{L}\frac{d}{dt}[e_k(t) + g(t)] \end{cases} \tag{4.21}$$

From which the one-to-one correspondences related to FSO parameters and the circuital parameters are:

$$\tilde{\omega} = -\frac{R}{L}, \tilde{\mu} = \frac{1}{LC}, \tilde{h}_{km} = -\frac{\alpha_{km}}{L}, \ \tilde{\lambda} \cdot p_{best\,k}(t) = \frac{1}{L} \cdot e_k(t) \text{ and } \tilde{\gamma} \cdot g_{best}(t) = \frac{1}{L} \cdot g(t). \tag{4.22}$$

Under these statements we can now analyze the circuital behavior by using the Laplace Transform, as the Theory of Circuits establishes. In the next sections we will focus our analyses just for the case in which $\underset{k\neq m}{h_{k,m}} = \tilde{h}$. We will call this case: fully-equal-connected. Although this is a simplification it does not involve a loss of generality and allows a easier writing of the equations governing the swarm-circuit.

4.5 Dynamic Analysis of Swarm Circuits By Laplace Transform.

Since the continuous FSO is a time-invariant systems within each time-window, we can adopt the Laplace Transform of (4.14) and (4.15) (i.e., of (4.16)), window by window. Then, as it has been made in each time-window for the force (4.12), we will assume that all parameters keep constant the values $\tilde{\lambda}_i, \tilde{\gamma}_i, \tilde{h}_i$ for each i-th time-window. It is important to note that more complex approaches than the basic Laplace Transform (methods based on 2D Laplace-Carson Transform) can be however applied if the previous assumptions are not valid. Under the previous assumptions, and by assuming that a generic time-window begins at the time indicating by t_{in} and ends at the time $t_{in}+\tau$, the Laplace Transform, $£\{\cdot\}$, in the new variable $s \in \mathrm{C}$, can be easily applied to (4.16) as follows:

$$(s\begin{bmatrix}\mathbf{U}_i(s)\\ \mathbf{X}_i(s)\end{bmatrix} = \begin{bmatrix}\mathbf{A}_{1,1} & \mathbf{A}_{1,2}\\ \mathbf{I} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{U}_i(s)\\ \mathbf{X}_i(s)\end{bmatrix} + \begin{bmatrix}\mathbf{F}_i(s)\\ \mathbf{0}\end{bmatrix} + \begin{bmatrix}\mathbf{u}_i(t_{in})\\ \mathbf{x}_i(t_{in})\end{bmatrix} \tag{4.23}$$

In (4.23), both $\mathbf{u}_i(t_{in}) = \mathbf{i}_{L,i}(t_{in})$ and $\mathbf{x}_i(t_{in}) = C\cdot\mathbf{v}_{C,i}(t_{in})$ are column vectors containing the time-window initial conditions evaluated at the time t_{in}, i.e. all the currents flowing into the inductors, that corresponds to the swarm member velocities, and all the voltages at the terminals of capacitors, that corresponds to the swarm member coordinates (positions), respectively. Similarly the unknown vectors contains the Laplace transforms of the inductor currents: $\mathbf{U}_i(s) = \mathbf{I}_L(s)$, capacitor voltages $\mathbf{X}_i(s) = C\cdot\mathbf{V}_C(s)$ and independent generators voltages $\mathbf{F}_i(s) = \frac{1}{L}\cdot(\mathbf{E}(s) + G(s)\cdot\mathbf{1})$, where with $\mathbf{1}$ we indicate a column vector of 1's. More in details, each single $£\{\cdot\}$ is then:

$$\mathbf{U}_i(s) = £\{\mathbf{u}_i(t)\} = £\{\mathbf{i}_{L,i}(t)\} \int_0^{+\infty} \mathbf{i}_{L,i}(t)\rho(t,t_{in},\tau)\cdot e^{-st}dt \tag{4.24}$$

For the velocities (inductor currents) at the i-th time window

$$\mathbf{X}_i(s) = £\{\mathbf{x}_i(t)\} = £\{C\cdot\mathbf{v}_{C,i}(t)\} = \int_0^{+\infty} C\cdot\mathbf{v}_{C,i}(t)\rho(t,t_{in},\tau)\cdot e^{-st}dt \tag{4.25}$$

For the positions (capacitor voltages) at the i-th time window

$$\mathbf{F}_i(s) = £\{\mathbf{F}_i(t)\} = \int_0^{+\infty} \frac{1}{L}\cdot(\mathbf{E}(s) + G(s)\cdot\mathbf{1})\rho(t,t_{in},\tau)\cdot e^{-st}dt \tag{4.26}$$

where

$$\rho(t,t_{in},\tau)=\begin{cases}1 & for \quad t_{in}<t\le t_{in}+\tau\\ 0 & otherwise\end{cases} \tag{4.27}$$

Moreover the sub-matrices in (4.23) are now, by using (4.22) and (4.17) and in the simpler case in which $\tilde{h}_{km}=\tilde{h}$, i.e. $-\alpha_{km}=\alpha$, $\forall k,m$:

$$\mathbf{A_{1,1}}=\mathbf{M}+\mathbf{H}=-\frac{R}{L}\cdot\mathbf{I}-\frac{\alpha}{L}\cdot\mathbf{J} \tag{4.28}$$

Where $\mathbf{J}$ is the symmetric matrix having *0's on the main diagonal* and 1's *elsewhere.* It is evident that the matrix $\mathbf{H}$ has not all zero entries just for continuous "full connected" FSO, whereas it is the null matrix for continuous PSO since $\tilde{h}_{km}=\tilde{h}=0$ (i.e., we can simply commutate from PSO to FSO, just considering the absence or the presence of $\mathbf{H}$ in equation (4.17)).

The last $N\times N$ sub- matrix appearing in (4.16) is:

$$\mathbf{A_{1,2}}=-\tilde{\mu}\cdot\mathbf{I}=-\frac{1}{LC}\cdot\mathbf{I} \tag{4.29}$$

To avoid a cumbersome formalism in the Laplace Transform, let us consider always $t_{in}=0$, whereas the proper solution in the time domain will be provided by a suitable a posteriori time-shifting, since the whole trajectories are obtained simply by the union of every result valid for each time-window.

Thus, by assuming that the force (4.12) is constant for all the duration of each i-th time-window, we have that the Laplace-transform of (4.12) for each k-th particle/bird is:

$$F_{k,i}(s)=\frac{\tilde{\gamma}\cdot gbest_i+\tilde{\lambda}\cdot pbest_{k,i}}{s}=\frac{1}{L}\left[E_{k,i}(s)+G(s)\right] \tag{4.30}$$

Where $E_k(s)=\pounds\{e_k(t)\}=\dfrac{\tilde{\lambda}\cdot pbest_{k,i}}{s}$ and $G(s)=\pounds\{g(t)\}=\dfrac{\tilde{\gamma}\cdot gbest_i}{s}$

Finally, by compacting (23) and solving for $\mathbf{X}_i(s)$, we have:

$$\mathbf{X}_i(s)=\left(s^2\mathbf{I}-\mathbf{A_{1,1}}s-\mathbf{A_{1,2}}\right)^{-1}\cdot\left\{\left(s^2\mathbf{I}-\mathbf{A_{1,1}}s\right)\cdot\mathbf{v}_{C,i}(t_{in})+\mathbf{i}_{L,i}(t_{in})+\mathbf{F}_i(s)\right\} \tag{4.31}$$

where,

$$\left(s^2\mathbf{1}-\mathbf{A}_{1,1}s-\mathbf{A}_{1,2}\right)=\begin{pmatrix} s^2-\tilde{\omega}s+\tilde{\mu} & -h_{12}s & \cdots & -h_{1N}s \\ -h_{21}s & s^2-\tilde{\omega}s+\tilde{\mu} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ -h_{N1}s & -h_{N2}s & \cdots & s^2-\tilde{\omega}s+\tilde{\mu} \end{pmatrix}=$$

$$=\begin{pmatrix} s^2+\frac{R}{L}s+\frac{1}{LC} & \frac{\alpha}{L}s & \cdots & \frac{\alpha}{L}s \\ \frac{\alpha}{L}s & s^2+\frac{R}{L}s+\frac{1}{LC} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\alpha}{L}s & \frac{\alpha}{L}s & \cdots & s^2+\frac{R}{L}s+\frac{1}{LC} \end{pmatrix} \quad (4.32)$$

and

$$\left(s^2\mathbf{1}-\mathbf{A}_{1,1}s\right)=\begin{pmatrix} s-\tilde{\omega} & -h_{12} & \cdots & -h_{1N} \\ -h_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -h_{N-1,N} \\ -h_{N,1} & \cdots & -h_{N,N-1} & s-\tilde{\omega} \end{pmatrix}=\begin{pmatrix} s+\frac{R}{L} & \frac{\alpha}{L} & \cdots & \frac{\alpha}{L} \\ \frac{\alpha}{L} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \frac{\alpha}{L} \\ \frac{\alpha}{L} & \cdots & \frac{\alpha}{L} & s+\frac{R}{L} \end{pmatrix} \quad (4.33)$$

How it is usual, we can decompose the response (solution) as:

$$\mathbf{X}_i(s)=\mathbf{X}_i^{free}(s)+\mathbf{X}_i^{forced}(s) \quad (4.34)$$

where:

$$\mathbf{X}_i^{free}(s)=\left(s^2\mathbf{I}-\mathbf{A}_{1,1}s-\mathbf{A}_{1,2}\right)^{-1}\cdot\left\{\left(s^2\mathbf{I}-\mathbf{A}_{1,1}s\right)\cdot\mathbf{x}_i(t_{in})+\mathbf{u}_i(t_{in})\right\} \quad (4.35)$$

is the L-Transform of the free response of the dynamic system and

$$\mathbf{X}_i^{forced}(s)=\left(s^2\mathbf{I}-\mathbf{A}_{1,1}s-\mathbf{A}_{1,2}\right)^{-1}\cdot\mathbf{F}_i(s) \quad (4.36)$$

is the forced response.

In this way, we can collect the contributions of initial conditions in the free solution solution (4.33), whereas the global and personal bests play their role just in the (4.34). For studying the collective behavior of the swarm, we will just analyze the free response (4.33), because it is not influenced from habitat (fitness), but just from the initial conditions of the swarm members.

With the aim to consider the contribution of each i-th time-window, the whole $x_k^{tot}(t)$ solution on t domain for each k-th particle/bird can be written as follows, taking into account time-shifting due to the union of all the responses coming from different time-windows shifted in time:

$$x_k^{tot}(t)=\sum_{i=0}^{N_{TW}-1} x_{k,i}(t-i\tau)\cdot\left[\Phi(t-i\tau)-\Phi(t-(i+1)\tau)\right] \tag{4.37}$$

where N_{TW} is the number of time-windows taken into account and the function $\Phi(t)=\begin{cases}1 & t\geq 0\\ 0 & t<0\end{cases}$ is the Heaviside unit step-function.

As it is known from a famous Laplace Transform rule, we can write that:

$$£\left\{x_k(t)\cdot\left[\Phi(t)-\Phi(t-\tau)\right]\right\}=X_k(s)\left[1-e^{-\tau s}\right] \tag{4.38}$$

In this way the whole Laplace Transform, $X_k^{tot}(s)=£\left\{x_k^{tot}(t)\right\}$, can be evaluated as follows:

$$X_k^{tot}(s)=\sum_{i=0}^{N_{TW}-1} X'_{k,i}(s)e^{-i\tau s}=\sum_{i=0}^{N_{TW}-1} X_{k,i}(s)\left[1-e^{-\tau s}\right]e^{-i\tau s} \tag{4.39}$$

From (4.37) it is possible to note that just the generic functions $X_{k,i}(s)$ characterize the swarm dynamics. In fact, $X_{k,i}(s)$ is just the k-th row-entries of (4.29) valid for a generic i-th time-window.

The evaluation of the inverse of the symmetric matrix $\left(s^2\mathbf{I}-\mathbf{A}_{1,1}s-\mathbf{A}_{1,2}\right)^{-1}$ returns that all its entries on the main diagonal are equal to:

$$\frac{s^2-\tilde{\omega}s+\tilde{\mu}-(N-2)\tilde{h}s}{\left(s^2-\left(\tilde{\omega}+(N-1)\tilde{h}\right)s+\tilde{\mu}\right)\left(s^2+\left(\tilde{h}-\tilde{\omega}\right)s+\tilde{\mu}\right)} \tag{4.40}$$

whereas all other entries are equal to:

$$\frac{\tilde{h}s}{\left(s^2-\left(\tilde{\omega}+(N-1)\tilde{h}\right)s+\tilde{\mu}\right)\left(s^2+\left(\tilde{h}-\tilde{\omega}\right)s+\tilde{\mu}\right)} \tag{4.41}$$

Thus, after to have defined the following quantity:

$$a_k(t_{in})=-\tilde{\omega}x_k(t_{in})-\sum_{m=1,m\neq k}^{N}\tilde{h}x_m(t_{in})+u_k(t_{in})=-\tilde{\omega}x_k(t_{in})+\tilde{h}x_k(t_{in})-\tilde{h}\sum_{m=1}^{N}x_m(t_{in}) \tag{4.42}$$

and by a suitable rearranging and re-ordering of all terms referred to the solution expressed by (4.32), (4.33) and (4.34), we obtain the following Laplace-Transform written in explicit form for each k-th member of the flock:

$$X_k(s)=\frac{sx_k(t_{in})+a_k(t_{in})}{s^2+(\tilde{h}-\tilde{\omega})s+\tilde{\mu}}+\frac{\tilde{h}s^2\sum_{j=1}^{N}x_j(t_{in})}{\left(s^2-\left(\tilde{\omega}+(N-1)\tilde{h}\right)s+\tilde{\mu}\right)\left(s^2+(\tilde{h}-\tilde{\omega})s+\tilde{\mu}\right)}+$$
$$+\frac{\tilde{h}s\sum_{j=1}^{N}a_j(t_{in})}{\left(s^2-\left(\tilde{\omega}+(N-1)\tilde{h}\right)s+\tilde{\mu}\right)\left(s^2+(\tilde{h}-\tilde{\omega})s+\tilde{\mu}\right)}+\frac{\tilde{h}\tilde{\lambda}\sum_{j=1}^{N}pbest_j}{\left(s^2-\left(\tilde{\omega}+(N-1)\tilde{h}\right)s+\tilde{\mu}\right)\left(s^2+(\tilde{h}-\tilde{\omega})s+\tilde{\mu}\right)}+$$
$$+\frac{\tilde{\lambda}\cdot pbest_k}{s\left(s^2-(\tilde{\omega}-\tilde{h})s+\tilde{\mu}\right)}+\frac{\tilde{\gamma}\cdot gbest}{s\left(s^2-\left(\tilde{\omega}+(N-1)\tilde{h}\right)s+\tilde{\mu}\right)}$$

(4.43)

Equation (4.41) can be regarded in terms of the Superposition Principle being the system linear. Moreover, reminding that this system is also time-invariant within each time-window, it is possible to adopt the concept of transfer (or network) function. A network function is the ratio between an output (effect) and the input (stimulus) which generated it. More in general, the network function is the Laplace-Transform of the response (effect) generated by the unit-impulse (that is a stimulus also said delta function or Dirac function). Thus, it is immediate to identify seven network-functions in (4.41) which can be compacted as follows:

$$X_k(s)=\sum_{j=1}^{7}H_j(s)\Im_j(s) \qquad (4.44)$$

In which the Laplace Transforms of the excitations due to the initial conditions at the time window-beginning, t_{in} , are:

$$\Im_1(s)=x_k(t_{in})\ ,\ \Im_2(s)=a_k(t_{in}),\ \Im_4(s)=\sum_{j=1}^{N}a_j(t_{in}) \qquad (4.45)$$

Whereas the Laplace Transforms of the excitations due to personal and global bests are:

$\Im_5(s)=\sum_{j=1}^{N}pbest_j/s$, $\Im_6(s)=pbest_k/s$ and $\Im_6(s)=gbest/s$.

This way, the $H_j(s)$ in (4.42) are the seven transfer functions that are explicitly written as follows for initial conditions:

$$H_1(s)=\frac{s}{s^2-(\tilde{\omega}-\tilde{h})s+\tilde{\mu}}$$
$$H_2(s)=\frac{1}{s^2-(\tilde{\omega}-\tilde{h})s+\tilde{\mu}} \qquad (4.46)$$
$$H_3(s)=\frac{\tilde{h}s^2}{\left(s^2-(\tilde{\omega}+(N-1)\tilde{h})s+\tilde{\mu}\right)\left(s^2-(\tilde{\omega}-\tilde{h})s+\tilde{\mu}\right)}$$

And those due to personal and global bests:

$$H_4(s)=\frac{\tilde{h}s}{\left(s^2-(\tilde{\omega}+\left(N-1\right)\tilde{h})s+\tilde{\mu}\right)\left(s^2-(\tilde{\omega}-\tilde{h})s+\tilde{\mu}\right)}$$
$$H_5(s)=\frac{\tilde{h}\tilde{\lambda}s}{\left(s^2-(\tilde{\omega}+\left(N-1\right)\tilde{h})s+\tilde{\mu}\right)\left(s^2-(\tilde{\omega}-\tilde{h})s+\tilde{\mu}\right)} \tag{4.47}$$
$$H_6(s)=\tilde{\lambda}H_2(s)$$
$$H_7(s)=\frac{\tilde{\gamma}}{\left(s^2-(\tilde{\omega}+\left(N-1\right)\tilde{h})s+\tilde{\mu}\right)}$$

It is interesting to note that for a PSO-circuit, i.e. based on standard PSO, the $\mathbf{H}$ matrix has all its entries null: $\tilde{h}=0$. Then, the corresponding transfer functions do not exist and the related inputs have no effects on the system dynamics.

Now, let us evaluate the Inverse Laplace Transform referred to (4.42). Firstly, we have to compute the poles of (4.42). These poles referred to the several different transfer functions are:

$$s_{1,2}=\frac{1}{2}\left\{\tilde{\omega}-\tilde{h}\pm\sqrt{(\tilde{\omega}-\tilde{h})^2-4\tilde{\mu}}\right\} \tag{4.48}$$

$$s_{3,4}=\frac{1}{2}\left\{\tilde{\omega}+\left(N-1\right)\tilde{h}\pm\sqrt{(\tilde{\omega}+\left(N-1\right)\tilde{h})^2-4\tilde{\mu}}\right\} \tag{4.49}$$

In the following mathematical developments, without loss of generality, we will consider parameters having values that always return $(\tilde{\omega}-\tilde{h})^2-4\tilde{\mu}\neq 0$ and $(\tilde{\omega}+\left(N-1\right)\tilde{h})^2-4\tilde{\mu}\neq 0$, i.e. all poles are simples with multiplicity equal to 1. In fact, the case for which the poles has multiplicity equal to 2, do not produce a significant changing in the waveform of the Inverse Laplace Transform when it is compared with the waveform referred to the case in which the poles are real and with multiplicity equal to 1. Obviously the poles can be real or complex. Under this last assumption, we can apply the classic residual method for the evaluation of the Inverse Laplace Transforms, $\pounds^{-1}\left\{\cdot\right\}$, by obtaining the responses to each input of the system:

$$\pounds^{-1}\left\{H_1(s)\Im_1(s)\right\}=x_k(t_{in})\sum_{j=1}^{2}\frac{(-1)^j s_j}{s_2-s_1}\cdot e^{s_j t} \tag{4.50}$$

$$\pounds^{-1}\left\{H_2(s)\Im_2(s)\right\}=a_k(t_{in})\sum_{j=1}^{2}\frac{(-1)^j}{s_2-s_1}\cdot e^{s_j t} \tag{4.51}$$

$$\pounds^{-1}\left\{H_3(s)\Im_3(s)\right\}=\sum_{j=1}^{4}\tilde{h}\left(\sum_{n=1}^{N}x_n(t_{in})\right)\frac{s_j^2}{\prod\limits_{n=1,n\neq j}^{4}(s_j-s_n)}\cdot e^{s_j t} \tag{4.52}$$

$$£^{-1}\left\{H_4(s)\Im_4(s)\right\}=\sum_{j=1}^{4}\tilde{h}\left(\sum_{n=1}^{N}a_n(t_{in})\right)\frac{s_j}{\prod_{n=1,n\neq j}^{4}(s_j-s_n)}\cdot e^{s_j t} \tag{4.53}$$

$$£^{-1}\left\{H_5(s)\Im_5(s)\right\}=\sum_{j=1}^{4}\tilde{h}\tilde{\lambda}\left(\sum_{n=1}^{N}pbest_n\right)\frac{1}{\prod_{n=1,i\neq j}^{4}(s_j-s_n)}\cdot e^{s_j t} \tag{4.54}$$

$$£^{-1}\left\{H_6(s)\Im_6(s)\right\}=\tilde{\lambda}\cdot pbest_k\sum_{j=1}^{2}\frac{(-1)^j}{s_2-s_1}\cdot e^{s_j t} \tag{4.55}$$

$$£^{-1}\left\{H_7(s)\Im_7(s)\right\}=gbest\sum_{j=1}^{2}\tilde{\gamma}\frac{(-1)^j}{s_{j+2}(s_4-s_3)}\cdot e^{s_{j+2}t} \tag{4.56}$$

Finally the wanted closed form in the time domain is obtained by Inverse Laplace Transform of (4.43) by means of superposition:

$$x_k(t)=v_{C,k}(t)=\sum_{m=1}^{7}£^{-1}\left\{H_m(s)\Im_m(s)\right\} \tag{4.57}$$

Equation (4.55) is valid for a generic k-th particle/bird just within a single time-window having width τ and starting from $t=t_{in}$ (i.e. $t_{in}\le t\le t_{in}+\tau$). The equation (4.57) describes the portion of the trajectory along the *j-th* dimension followed by the *k-th* member of the flock within that time-window (remind that we chose to omit the superscript <j> for avoid as more as possible cumbersome formalisms). The corresponding closed form of the time-varying component of the inductor currents, i.e. velocity, along the *j-th* dimension is trivially obtainable by Inverse Laplace Transform, as follows:

$$u_k(t)=\sum_{m=1}^{7}£^{-1}\left\{sH_m(s)\Im_m(s)\right\} \tag{4.58}$$

Whereas on the basis of (4.20), the current flowing into the inductors is simply:

$$i_{L,k}(t)=C\frac{d}{dt}v_{C,k}(t)=C\cdot\sum_{m=1}^{7}£^{-1}\left\{sH_m(s)\Im_m(s)\right\} \tag{4.59}$$

Finally, it is enough to apply the superposition shifted in time (4.32) for obtaining the whole trajectory along the j-th dimension of the space.

Stability Analysis of the continuous swarm-circuit

The stability analysis becomes trivial when the full knowledge of poles and zeros of the transfer-functions is available. The full asymptotical convergence is achieved if the continuous swarm parameters are set to be:

$$\tilde{\omega}<\tilde{h}<-\frac{\tilde{\omega}}{N-1} \tag{4.60}$$

For the swarm-circuit parameters (4.22), equation (4.58) becomes:

$$-\frac{R}{N-1} < \alpha < R \tag{4.61}$$

oscillations (i.e., complex conjugate poles) will be observed if

$$-2\sqrt{\tilde{\mu}}+\tilde{\omega} < \tilde{h} < 2\sqrt{\tilde{\mu}}+\tilde{\omega} \qquad \text{or} \qquad \tilde{h} < \frac{2\sqrt{\tilde{\mu}}-\tilde{\omega}}{N-1} \tag{4.62}$$

For the swarm-circuit parameters (22), equations (60) becomes:

$$-2\sqrt{\frac{L}{C}}+R < \alpha < 2\sqrt{\frac{L}{C}}+R \quad \text{or} \qquad \alpha > \frac{2\sqrt{\frac{L}{C}}+R}{N-1} \tag{4.63}$$

A deeper discussion on the tuning of parameters and the effect of stability will be made in a next section.

4.6 Deterministic tuning of parameters

Another important feature of the continuous dynamic system, i.e. the swarm-circuit, is the possibility to provide the behaviour of particles/birds simply studying the poles of the network functions for any different Δt value. In fact, by varying these poles we can force the trajectories of the swarm members to converge towards a stable point, to escape from a space zone, or to stay into a limited zone (closed loops) of the solution space. In this way we can improve the capabilities of exploration or exploitation simply by modifying the nature of the poles at each time windows. Thus, by extending the update of poles to all time windows, it is possible to control the whole behaviour of the swarm-members. On the contrary, the numerical algorithms employ random updating of parameters to avoid local minima. By using the continuous dynamic system, the randomness of parameters can be eliminated, because the user knows how he has to manage the parameters to obtain convergence or divergence, i.e. how exalting the exploitation or the exploration capabilities. The best way for the tuning of parameters for the continuous algorithm is an open problem and will be object of next studied. The simplest idea is to fix the maximal amplitude that is admissible for a time-interval in which there are not improvements of the fitness value. Thus, we can use parameters that assure negative values of real poles or complex poles with negative (or zero) real parts, until the monitoring of the fitness returns improved values. Then, if the monitoring time exceeds the fixed threshold, the parameters can be switched to a new set of values that assures positive values of real pole or positive real part of complex poles. An example of a simple control of parameter working in this way is used and described in a next section.

4.7 Comparison between the numerical algorithms and the closed forms.

The aim of this section is to compare the trajectories made by capacitor voltages and inductor currents of the swarm circuit versus the behavior of the corresponding numerical algorithm. The comparative analysis proposed in this section has not the aim to compare the performances in optimization (see next section) but consists of verifying that the trajectories of the numerical algorithms can become fully coincident with those obtained by using the closed forms in the time domain when the amplitude of the time-windows Δt is small enough to produce the correct integration of Euler's formulas. First of all, some preliminary considerations about the way in which the comparative analysis is designed must be done: both the numerical algorithms and the dynamic state equations have been identically initialized. The random updating of parameters, which is present in the numerical codes, has been removed. The parameters inserted in the equations of the swarm circuit have been evaluated according to (4.22) after to have chosen a suitable integration-step Δt that fixes the values of parameters by (4.11).

Consequently we pose $\tau = \Delta t$ for each new i-th time-window ($i = 0,1 \ldots N_{TW} - 1$, where N_{TW} is both the number of the iterations of the numerical algorithm and the number of time-windows). The validations are referred to a functional:

$$\begin{aligned} f_1(x,y) = {} & peaks(x,y) + 0.01x^2 + 0.01y^2 + peaks(x+5,y+5) + 2peaks(x-5,y-5) + \\ & + 5peaks(x-5,y+5) + 3peaks(x+5,y-5) \end{aligned} \tag{4.64}$$

in which shifted MATLAB™'s "peaks" functions are used (see Fig.4.2). In Figure 4.3, the (4.64) is shown.

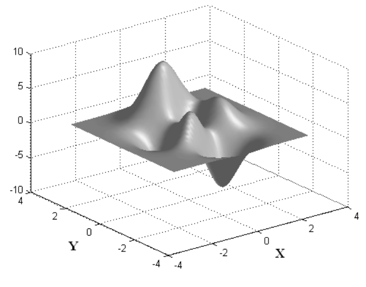


Fig. 4.2. MATLAB™ "peaks" function.

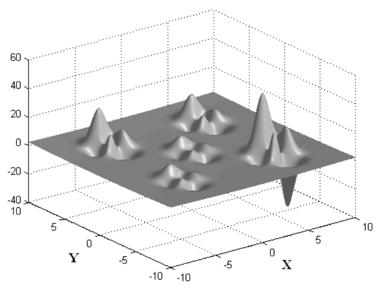


Fig. 4.3. Functional $f_1(x,y)$ obtained by (4.62).

A swarm composed by 10 particles has been used for all validations. The parameters $\tilde{\omega}$, $\tilde{\lambda}$, $\tilde{\gamma}$ and $\tilde{h}_{km}$ have been a priori fixed, whereas the parameters inserted into the numerical algorithm have been fixed by inverting the relations 4.11) as follows:

$$\begin{cases} \omega = \tilde{\omega}\Delta t + 1 \\ \lambda = \tilde{\lambda}\Delta t \\ \gamma = \tilde{\gamma}\Delta t \\ h_{km} = \tilde{h}_{km}\Delta t \end{cases} \tag{4.65}$$

The trajectories of the continuous fully connected swarm-circuit described by (4.48)-(4.54) have been compared with those performed by the corresponding numerical algorithm described by the pseudo-code in the first section (the code has been implemented by MATLAB™). With the aim to allow a full reproducibility of the present validations, we furnish in Table I all the used initial positions (x_0, y_0) related to each bird and in Table II all the corresponding initial velocities $(u_{0,x}, u_{0,y})$. Each particle has been numbered to be identified, and the number associated to each particle is shown in the first column of the tables.

The used fixed values of continuous swarm parameters are $\tilde{\omega} = -1$, $\tilde{\lambda} = 1$, $\tilde{\gamma} = 1$ and $\tilde{h}_{km} = 1/9$ and, according with (4.22), the swarm circuit parameters have been fixed equal to: $L = 0.5\ H$, $R = 0.5\ \Omega$ and $C = 1\ F$. As a consequence of the value of C, the capacitor voltages are exactly equal to the value of the positions. In Figures 4.3, 4.4 and 4.5, the trajectories of particle #1 both for numerical algorithm and continuous system are shown for different values of Δt.

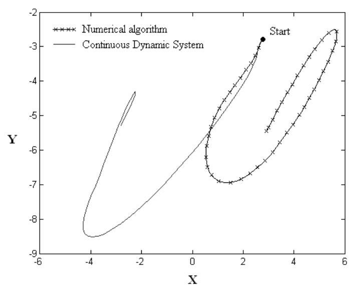
Table 20
Initial Positions (capacitor voltages)

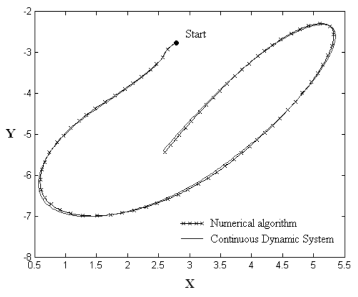
Particle #	x_0	y_0
1	2.7757	-2.7839
2	-0.8044	2.0467
3	0.3901	6.6219
4	-5.8568	-6.9263
5	2.5187	-4.4633
6	5.2352	-1.1346
7	2.7788	-5.6013
8	-7.0789	0.2979
9	-6.8048	3.8978
10	5.1059	-4.3699

Table 21
Initial Velocities (inductor currents)

Particle #	$u_{0,x}$	$u_{0,y}$
1	-0.7578	-0.5640
2	-0.8951	-0.4093
3	-0.9526	0.3173
4	-0.2132	-0.1345
5	0.3105	-0.2495
6	1.9085	0.1222
7	-0.1625	0.6901
8	-1.5327	-1.0979
9	-0.4113	-0.3680
10	0.4394	-0.0896

Obviously, for different values of Δt we have different values of the numerical algorithm parameters according to (4.63). The used values of Δt are shown in each figure caption.





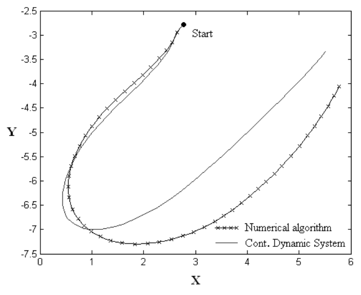


Fig. 4.3. Comparison between the trajectories referred to the numerical algorithm and to the continuous system made by Bird #1. The time-window duration is $\tau = \Delta t = 0.1$ and total number of iterations $N_{TW} = 53$.

Fig. 4.4. Comparison between the trajectories referred to the numerical algorithm and to the continuous system made by Bird #1. The time-window duration is $\tau = \Delta t = 0.05$ and total number of iterations $N_{TW} = 70$.

Fig. 4.5. Comparison between the trajectories referred to the numerical algorithm and to the continuous system made by Bird #1. The time-window duration is $\tau = \Delta t = 0.01$ and total number of iterations $N_{TW} = 550$.

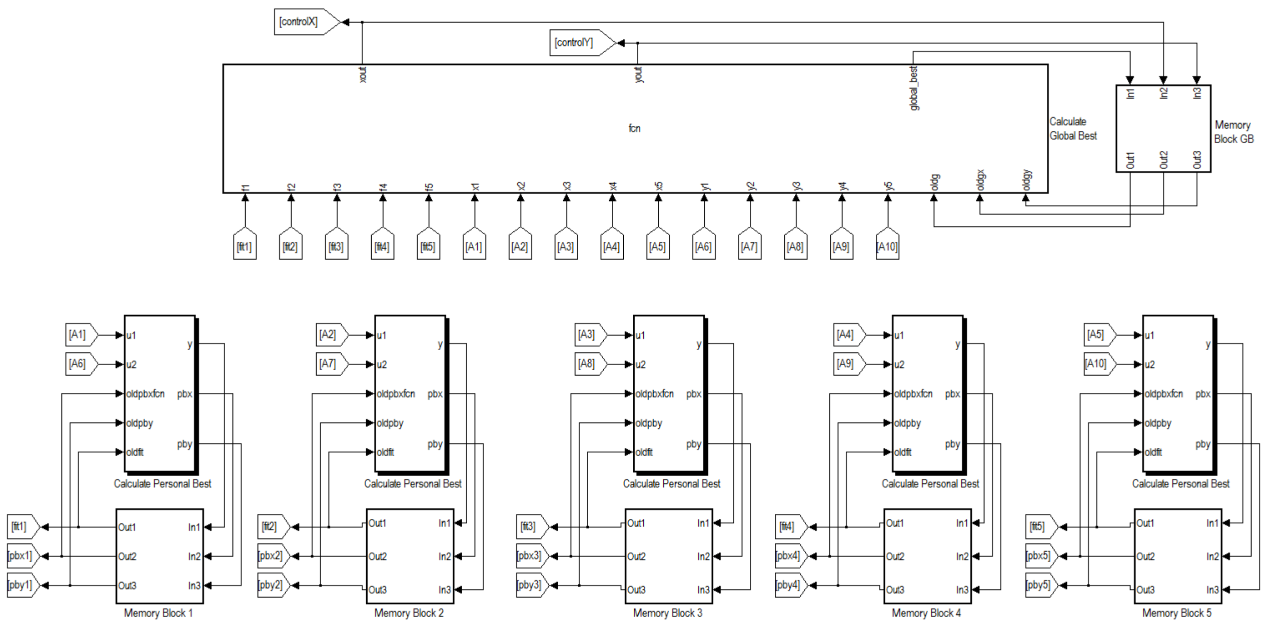


Figure 4.6:Simulinks block for calculating the global and personal best.

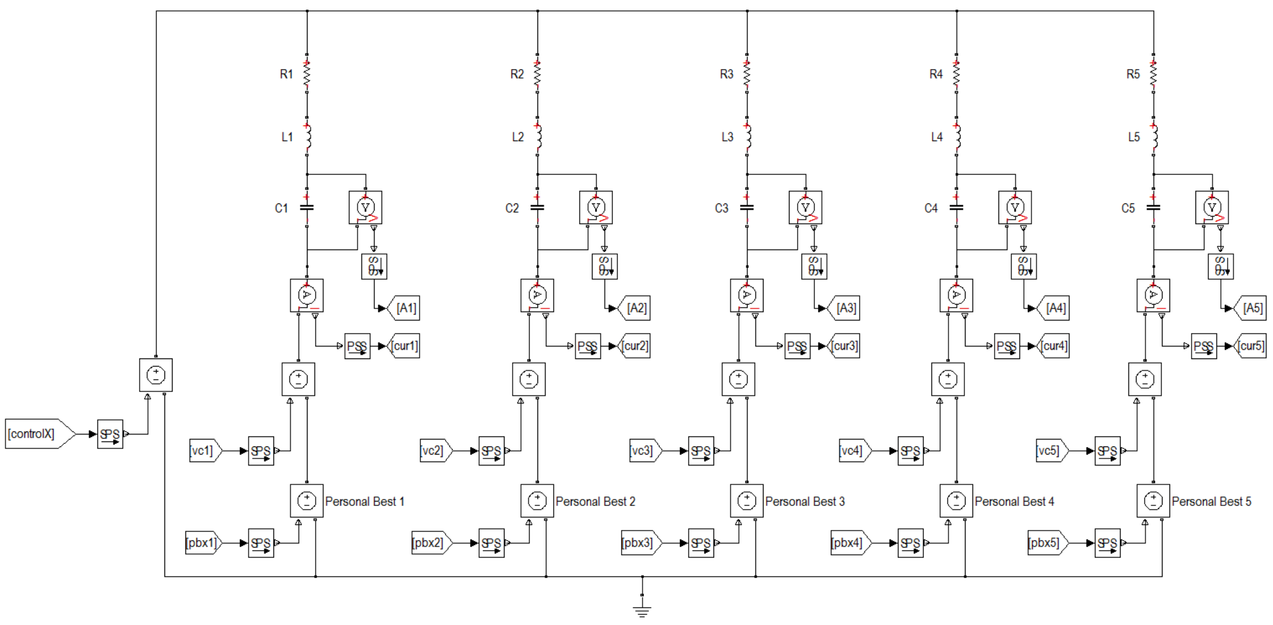


Figure 4.7: One dimension of the circuit.

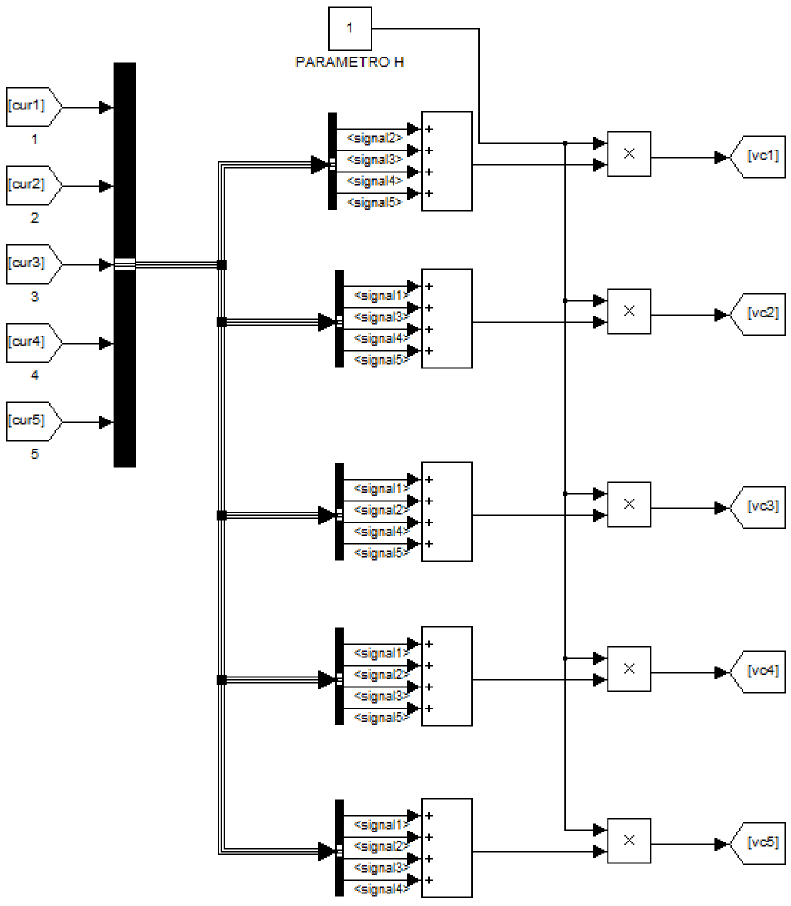
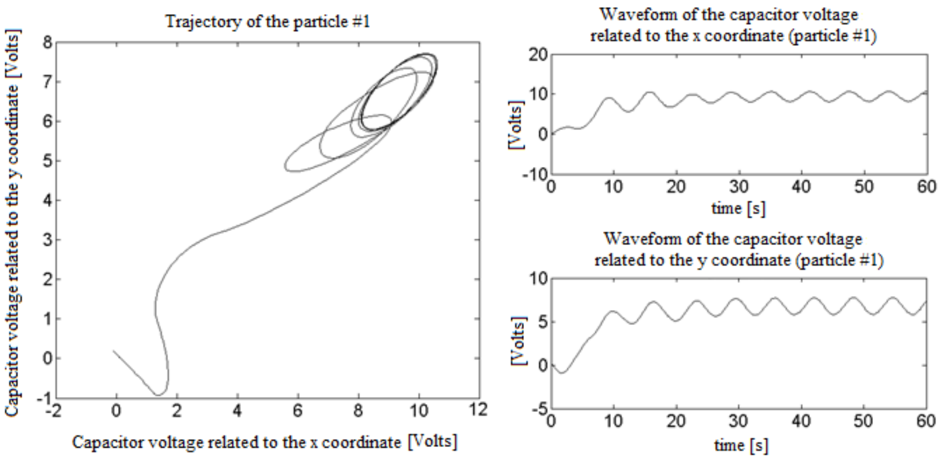


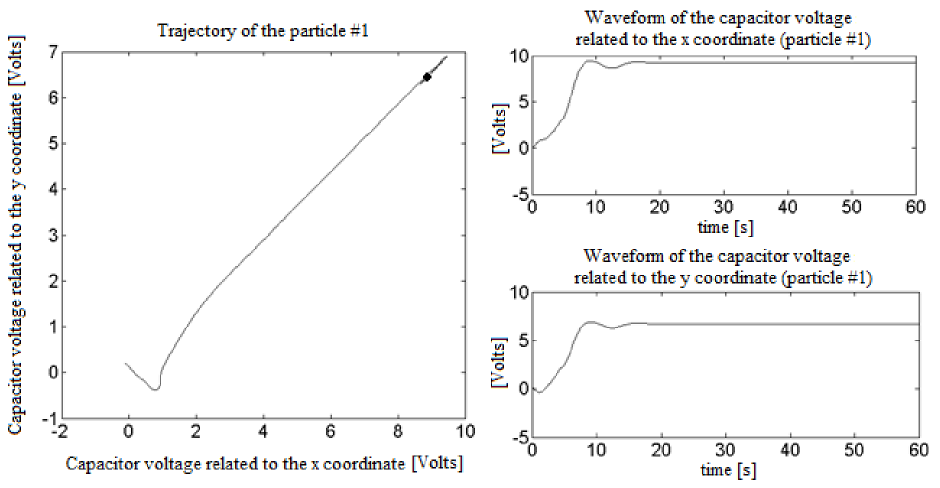
Figure 4.8:Simulink block for calculating the first component's velocity.

4.8 Validation on Benchmarks and Inverse Problems.

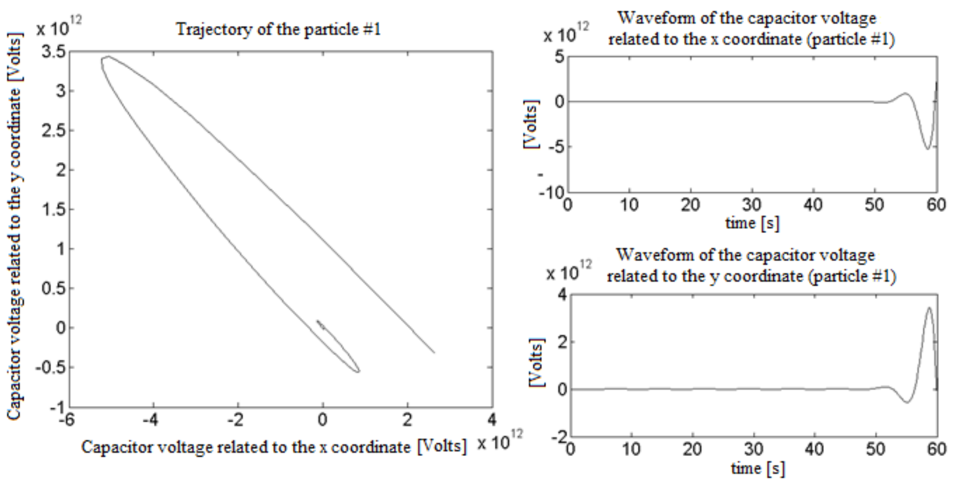
The Swarm-Circuit has been implemented by Simulink. The implementation of the Swarm-Circuit starts by assembling a number of N_B circuital branches each one equal to that shown in Figure 4.1. In particular $N_B = D \cdot N$ where D is the dimension of the solution space and N is the number of particles. More in detail, we have D independent circuits as in Figure 4.7 each of them dedicated to one dimension of the solution space. As previously said, the circuit is simulated by a time-windowing. Within each time-window the values of the independent voltage generators that take into account personal bests $e_k(t)$, and global best, $g(t)$, are updated by the values that were found in previous time-windows. These values are trivially carried out by using standard electronic devices that compare the voltage levels measured on each capacitor. Obviously, the use of commercial circuital simulators makes automatic the correct evaluation of the initial conditions measured on inductors and capacitors. Finally, the switches (see Figure. 4.1) that play the role to generate the time-windowing are assumed ideal, i.e. they commute instantaneously. Regarding to the Current-Controlled-Voltage-Sources (CCVC), they are used to govern convergence or divergence of the trajectories according to (4.59) and (4.61) for a fixed value of R,L and C parameters. With the aim to provides some examples on the different behaviors obtainable by tuning the α parameter, the next figure 4.9 is provided.



(a)



(b)



(c)

Fig. 4.9: (a) $\alpha = 1\ \Omega$, **(b)** $\alpha = 0.25\ \Omega$, **(c)** $\alpha = 2\ \Omega$

In the Table 22 validation results performed by the Swarm-Circuit are shown. The used parameter values are: $L = 1\ H$, $R = 1\ \Omega$, $C = 1\ F$ whereas α is random chosen, time-window by time-window, among one of these three values: $1\ \Omega$, $0.25\ \Omega$ and $\alpha = 1.1\ \Omega$. This last strategic choice allows to change the behavior of particles from oscillations to convergence or divergence and vice versa to perform exploration and exploitation. While the circuit is working, all the global bests and the relative coordinates found are stored in a suitable memory. Thus, at the end of the process it is enough to select the best solutions.

Table 22. Swarm-Circuit Validation Results

Name	Minimum value	Minimum coordinates circuit	Minimum value circuit
Styblinski-Tang	-78.332331	(-2.902435;-2.904137)	-78.332304
Bird	-106.764537	(4.741049;3.199581) (-1.599922;-3.117082)	-106.250957 -106.696935
Branins	0.397887	(-3.226006;12.603203) (3.145650;2.257012) (9.385460;2.523816)	0.398186 0.447557 0.411996
Six-hump camel back	–1.0316	(-0.0870;0.7354) (0.0877;-0.711)	-1.0271 -1.0316
Michalewics Function	-1.8013	(2.2436;1.5654)	-1.7731
Goldstein-Price function	3	(-0.0053; -1.0042)	3.0099

4.8.1 Validation on Inverse Problems.

The swarm-circuit has been also tested for two inverse problems related to the identification of two dynamic systems: Brussellator and Tunnel Diode [38]. The presented results refer to 30 launches with different random initializations of all positions and velocities.

The differential equations system that governs the Brussellator system are:

$$\begin{cases} \dfrac{dy_1}{dt} = A + y_1^2 \cdot y_2 - (B+1)\, y_1 \\ \dfrac{dy_2}{dt} = B \cdot y_1 - y_1^2 \cdot y_2 \end{cases} \tag{4.66}$$

Whereas the Tunnel Diode is governed by the following law:

$$\begin{cases} \dfrac{dy_1}{dt} = y_2 \\ \dfrac{dy_2}{dt} = -y_1 + y_2\left(1.4 - p \cdot y_2^2\right) + 4 \cdot a \cdot \left(-0.1649 y_1 + 0.4606\right) \end{cases} \tag{4.67}$$

The number of iterations is fixed to 2500 for each case. The obtained results are listed in table 23 in which appears the Mean Absolute Percentage Error (MAPE). The swarm-circuit results show the same behavior performed by the FSO numerical algorithm. However it is important to note that the utilization of hybrid numerical techniques allow reaching a better performance. In fact, the novel hybrid algorithm MeTEO by using three different numerical algorithms and the technique called "fitness modification" is able to reach an error closer to zero. But it is evident that the aim of the test is to validate that a swarm-circuit is just able to solve inverse problems like a numerical algorithm; whereas future developing could be made designing hybrid-circuits. A further detail, instead, is worth to notice: even if the Brussellator system is monomodal, the global minimum is hidden in such way that for an algorithm able to perform just exploration this is one of the the worst situations. In fact, the exploration algorithm is just able to detect the region in which the global minimum lies but fails in founding where actual it is. The error function of the Tunnel Diode, instead, refers to a smoother function, thus, it is possible to reach a good refinement of the parameters. In figure 5.11 the curves of the best results found over all 30 launches are shown.

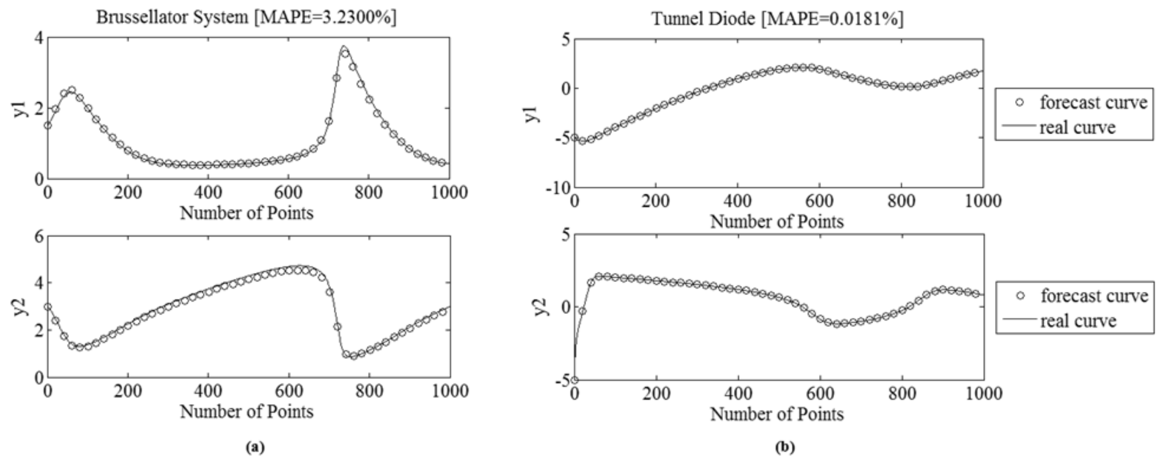


Fig. 4.11. (a) best curve forecast found by Swarm Circuit of the Brussellator (20 second of simulation); (b) best curve forecast found by Swarm Circuit of the Tunnel Diode (4.5 seconds of simulation)

Table 23. Swarm-Circuit Inverse Problems Validation Results

	MAPE					
	Mean	σ	**Initial Value**	**Real parmeters**	**Best forecast parameters**	**Mape best**
Brussellator	12.3417	13.6110	[1.5 3]	$[A,B]=[1,3]$	[0.9902,2.9016]	3.3003%
Tunnel	1.0414	1.5094	[-5 -5]	$[p,a]=[1.4,1]$	[1.0097,1.4111]	0.0181%

REFERENCES

[1] J. Kennedy, and R. Eberhart, “Particle swarm optimization”, Proceedings of the IEEE International Conference on Neural Networks, Perth, (vol. IV, pp. 1942-1948) Australia, 1995.

[2] C. W. Reynolds, “Flocks, herds and schools: a distributed behavioral model”, Computer Graphics, 21(4), 25-34, 1987.

[3] F. Heppner, U. Grenander, A stochastic nonlinear model for coordinated bird flocks. In S. Krasner, Ed., The Ubiquity of Chaos. AAAS Publications, Washington, DC., 1990.

[4] A. Engelbrecht, Fundamentals of Computational Swarm Intelligence, New York: Wiley, 2005.

[5] M. Clerc, Particle Swarm Optimization, London, ISTE Ltd., 2006.

[6] M. R. AlRashidi and M. E. El-Hawary, "A Survey of Particle Swarm Optimization Applications in Electric Power Systems", IEEE Transactions On Evolutionary Computation, Vol. 13, No. 4, August 2009, pp. 913-918

[7] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” Proc. IEEE Int. Conf. Evol. Comput., pp. 69–73, 1998.

[8] J.H. Seo, C.H. Im, S.Y. Kwak, C.G. Lee and H.K. Jung ,"An Improved Particle Swarm Optimization Algorithm Mimicking Territorial Dispute Between Groups for Multimodal Function Optimization Problems", IEEE Transactions On Magnetics, Vol. 44, No. 6, 2008, pp. 1046-1049.

[9] K. Parsopoulos and M. Vrahatis, “Modification of the particle swarm optimizer for locating all the global minima,” Artificial Neural Networks and Genetic Algorithms, Springer, 2001, pp. 324–327.

[10] D. Bratton, J. Kennedy, “Defining a standard for particle swarm optimization”, IEEE Swarm Intelligence Symposium, (SIS 2007), pages 120–127. IEEE Press

[11] Xin Chen and Yangmin Li, Senior Member, “A Modified PSO Structure Resulting in High Exploration Ability With Convergence Guaranteed”, IEEE Trans. on systems, man, and cybernetics—part b: cybernetics, vol. 37, no. 5, oct. 2007, pp. 1271- 1289.

[12] Z.H. Zhan, J. Zhang, Y. Li, and H.S.H Chung, "Adaptive Particle Swarm Optimization", IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, Vol. 39, No. 6, December 2009, pp.1362 – 1381.

[13] M.M. Ali, P. Kaelo, “Improved particle swarm algorithms for global optimization”, Applied Mathematics and Computation, - Elsevier, 2008, pp. 578-593.

[14] M. A. Montes de Oca, T. Stützle, M. Birattari and M. Dorigo, “Frankenstein’s PSO: A Composite Particle Swarm Optimization Algorithm”, IEEE Transactions On Evolutionary Computation, Vol. 13, No. 5, October 2009, pp. 1120 – 1132.

[15] Z. Xinchao, “A perturbed particle swarn algorithm for numerical optimization”, Applied Soft Computing, vol. 10, no. I, pp. 119-124, 2010.

[16] L. Liu, S. Yang, and D. Wang, “Particle Swarm Optimization with Composite Particles in Dynamic Environments”, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 40(6) pp. 1634-1648, 2010.

[17]R. Mendes, "Population Toplogies and Their Influence in Particle Swarm Performance", PhD thesis, Universidade do Minho, Braga, Portugal, 2004.

[18]R. Mendes, J. Kennedy and J. Neves, "The Fully Informed Particle Swarm: Simpler, Maybe Better", IEEE Transactions On Evolutionary Computation, Vol.8,No.3, June 2004. pp. 204-210.

[19]J. Kennedy and R. Mendes, "Neighborhood Topologies in Fully Informed and Best-of-Neighborhood Particle Swarms", IEEE Transactions On Systems, Man, and Cybernetics - Part C: Applications and Reviews, Vol. 36, No. 4, July 2006, pp. 515 – 519.

[20]A. S. Mohais , R. Mendes , C. Ward and C. Posthoff "Neighborhood re-structuring in particle swarm optimization", Proc. 18th Australian Joint Conf. Artificial Intell., vol. LNCS 3809, p.776 , 2005.

[21]H. Liu, E. Howely, and J. Duggan, "Particle swarm optimisation with gradually increasing directed neighbourhoods", Proceedings of the 13th annual conference on Genetic and evolutionary computation (GECCO '11), 2011, pp. 29-36.

[22]M. A. Montes de Oca, T. Stutzle, K. Van den Enden, and M. Dorigo, "Incremental social learning in particle swarms" IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics, 2011, vol. 41(2), pp.368-384.

[23]X. Li, "Niching Without Niching Parameters: Particle Swarm Optimization Using a Ring Topology", IEEE Transactions on Evolutionary Computation, Vol. 14, No. 1, pp. 150-169, February 2010.

[24]F. Riganti Fulginei, A. Salvini "Hysteresis model identification by the Flock-of-Starlings Optimization", International Journal of Applied Electromagnetics And Mechanics, vol. Volume 30, Number 3-4; p. 321-331, 2009.

[25]F. Riganti Fulginei, A. Salvini, "The Flock-of Starlings Optimization: influence of topological rules on the collective behaviour of Swarm Intelligence" Computational Methods for the Innovative Design of Electrical Devices - Series: "Studies in Computational Intelligence" Springer. pp. 139-157, 2010.

[26]M. Ballerini, N. Cabibbo, R. Candelier, A. Cavagna, E. Cisbani, I. Giardina, V. Lecomte, A. Orlandi, G. Parisi, M. Procaccini, M. Viale, and V. Zdravkovic., "Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study", Proceedings of the National Academy of Science, pp.1232-1237, 2008.

[27]M. Clerc, J. Kennedy "The particle swarm-explosion, stability, and convergence in a multidimensional complex space", IEEE Transactions on Evolutionary computation, Vol.6(1), 2002, pp.58–73.

[28]W. Li, "Stability Analysis of Swarms with General Topology". IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, Vol. 38, No. 4, pp. 1084-1097, August 2008

[29]V. Kadirkamanathan, K. Selvarajah, and P. J. Fleming, "Stability Analysis of the Particle Dynamics in Particle Swarm Optimizer", IEEE Transactions on Evolutionary Computation, Vol. 10, No. 3, pp. 245-255, June 2006.

[30]S. Kiranyaz, T. Ince, A. Yildirim, and M. Gabbouj, “Fractional Particle Swarm Optimization in Multidimensional Search Space”, IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, Vol. 40, No. 2, April 2010, pp. 298 – 319.

[31]J.L. Fernández-Martínez, E. García-Gonzalo and J.P. Fernández-Alvarez, “Theoretical analysis of particle swarm trajectories through a mechanical analogy”, International Journal of Computational Intelligence Research, vol.4(2), 2008, pp.93

[32]H. M. Emara and H. A. Abdelfatah, “Continuous swarm optimization technique with a stability analysis,” in Proc. IEEE-ACC. Conf., 2004, pp. 2811–2817.

[33]R. M. Gray, “Toeplitz and Circulant Matrices: A review” Foundations and Trends in Communications and Information Theory, Vol 2, Issue 3, pp 155-239, 2006.

[34]J. Baker et al., “A recursive algorithm to invert multiblock circulant matrices”, Kyungpook Math. J. 28 (1988), pp. 45–50.

[35]C. MacNish, “Towards unbiased benchmarking of evolutionary and hybrid algorithms for real-valued optimisation,” Connection Science, vol. 19, no. 4, pp. 361–385, 2007.

[36]K. Tang, X. Yao, P. N. Suganthan, C. MacNish, Y. P. Chen, C. M. Chen, and Z. Yang, “Benchmark functions for the CEC’2008 special session and competition on large scale global optimization,” Nature Inspired Computation and Applications Laboratory, USTC, China, Tech.Rep.2007.[Online].Available: http://nical.ustc.edu.cn/cec08ss.php.

[37]K. Schittkowski, Report, Department of Computer Science, University of Bayreuth (2004): http://www.ai7.uni-bayreuth.de/mc_ode.htm.

[38]F. Riganti Fulginei, A. Salvini and G. Pulcini, "Metric-topological-evolutionary optimization". Inverse Problems in Science & Engineering (IPSE), vol. 20, p. 41-58, 2012

[39] Konfrst, Z.,2004. “Parallel genetic algorithms: advances, computing trends, applications and perspectives.” Proceedings of the IEEE Parallel and Distributed Processing Symposium, Santa Fe,NM,USA,pp.162–169

[40] Hidalgo,J.I.,Prieto,M.,Lanchares,J.,Baraglia,R.,Tirado,F.,Garnica,O., 2003. “Hybrid parallelization of a compact genetic algorithm”. Proceedings of the Euromicro Conference on Parallel, Distributed and Network-Based Processing, Genova, Italy, February, pp.449–455.

[41] Amin Farmahini - Farahani, Shervin Vakili, Saeed Mehdi Fakhraie, Saeed Safari,Caro Lucas. “Parallel scalable hardware implementation of asynchronous discrete particle swarm optimization”, Elsevier, Engineering Applications of Artificial Intelligence 23 (2010) 177–187.

[42] Kokai, G.,Christ,T.,Frhauf,H.H.,2006. “Using hardware-based particle swarm method for dynamic optimization of adaptive array antennas.Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems,Istanbul, Turkey, June15–18,pp.51–58.

[43] Reynolds, P.D.,Duren,R.W.,Trumbo,M.L.,Marks,R.J.,2005.FPGA implementation of particle swarm optimization for inversion of large neural networks.In:Proceedings of the IEEE Swarm Intelligence Symposium,Pasadena,TX,USA,June 8–10,pp.389–392.

[44] Pena, J.,Upegui,A.,Sanchez,E.,2006. “Particle swarm optimization with discrete recombination: an on line optimizer for evolvable hardware”. Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems,Istanbul,Turkey, June15–18,pp.163–170.

[45] Juan Luis Fernàndez-Martìnez and Esperanza Garcìa-Gonzalo. "Stochastic Stability Analysis of the Linear Continuous and Discrete PSO Models". IEEE Transactions on Evolutionary Computation, Volume: 15, Issue: 3, 2011 , Page(s): 405 - 423.

[46] Bernhard Brandstätter and Ulrike Baumgartner. "Particle Swarm Optimization—Mass-Spring System Analogon." IEEE Transactions on MAgnetics, VOL. 38, NO. 2, March 2002

Conclusion

The main arguments treated in this thesis have been: 1) the implementation of a new algorithm which exploit topology and metric characteristics showed by some meta-heuristics (especially those swarm-based); 2) the design of a new circuit, called *swarm circuit*, able to perform social behavior in terms of voltages and currents. In particular a new meta heuristic algorithm called MeTEO (Metric and Topological Evolutionary Optimization) has been presented. It consists of coordination of three different metaheuristic: Flock of starlings Optimization, Particle Swarm Optimization and Bacterial Chemotaxis Optimization. MeTEO imports from these three algorithms their main properties that are:

- a good exploration get by the FSO;
- a standard capability of refinement of minima (PSO);
- a good convergence inherited by BCA.

In addition, MeTEO has been completed by using a new method from escaping from local minima called Fitness Modification.

The innovation of MeTEO is in the robustness and in capability to perform exploitation and exploration. Thus MeTEO can also be easily used by non-expert users in optimization tasks.

As a further contribution of this thesis work, new techniques (fitness-based and swarm-based parallelization) have been proposed for designing a suitable parallel framework able to exalt the characteristics of the swarm based algorithms.

MeTEO has been validate by mean of many benchmarks and inverse problems.

A further step has been made, introducing the interpretation of numerical swarm-based algorithm as dynamical system in the continuum, and the relative design for hardware implementation.

This hardware approach represents the first step towards real time optimization without utilization of microcontrollers or CPUs or ALUs, since the approach is completely “analog-oriented”.