Project Food Sense

Problem Statement

- To use Nature Inspired Algorithms to determine the quality of food items using various gas sensors and camera

MFO

- Fitness measure : distance of moth from corresponding flame
- Moth position updation: using logarithmic spiral
- Adaptive decrease of no of flames: for convergence
- Convergence

GSA

- Mass updation of agents : proportional to fitness
- Role of G Updation in convergence
- Convergence

GSA + MFO Hybrid

- Assign mass concept to Moths in MFO
- Communication between moths because of both MFO and GSA
- Increase the mass of the fitter moth - GSA
- Moths spiral around fit flames - MFO

Hybrid Algorithm (Contd.)

- Moth motion is guided by not just by flames but by masses of other moths
- Ideally, faster convergence

Pseudo Code

```
initialize moth_position, moth_velocity, moth_acceleration
calculate initial fitness for each moth
while(iteration < max_iteration)
update flame_number
if (iteration == 1)
        F = sort(M)
        OF = sort(OM)
else
        F = sort(Mt-1, Mt)
        OF=sort(Mt−1, Mt);
end
for i=1: n
        for j=1: d
                update r and t
                D(i) = |F(i) – M(i)|
                M(i) = D(i) . ebt . cos(2∏t) + F(j)
        end
end
```

```
G = G0(1/iteration)^b
for i=1:n
        mass(i) = 1/fitness(i)
        for j=1:n
                r = moth_position(i) – moth_position(j)
                F = G*mass(j)/r
        end
        moth_velocity(i) = moth_velocity(i)*rand(0,1) + F
        moth_position(i) = moth_position(i)*rand(0,1) + moth_velocity(i)
end
end
```

Mass Fitness Relationship

- GSA - Measure of Fitness is Mass
- MFO - Measure of Fitness is distance to fittest moth
- GSA works on the premise that fitter agent attracts others towards itself

Mass Fitness Relationship

- Hybrid algorithm, exploits this behaviour
- Hybrid - Measure of Fitness is distance to fittest moth
- Fittest moth has highest mass

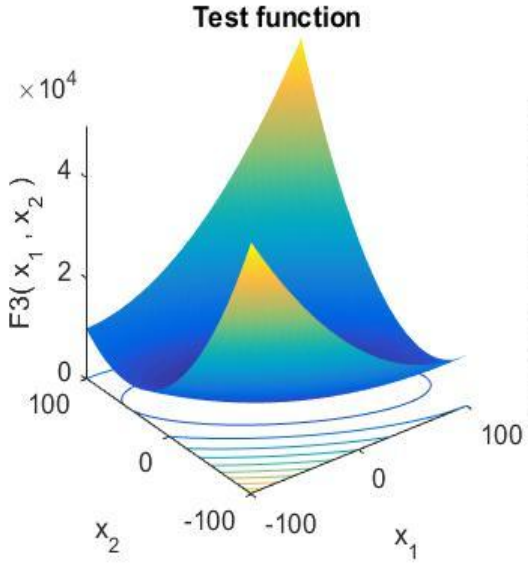
Problems Faced - G Updation

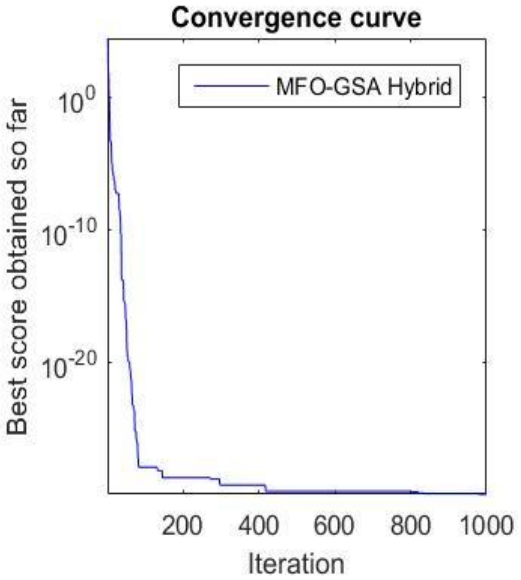
- Stagnation of algorithm for values in the order of G0
- Reason for stagnation - Slow decay of G
- $1/t$ vs $1/t^2$ vs b^t where $b<1$ and t = iteration no
- Slow decay of G: results in deviation of Moths from optimum value
- Updated decay of G to facilitate faster convergence without stagnation
- As iterations progress, decrease the GSA motion

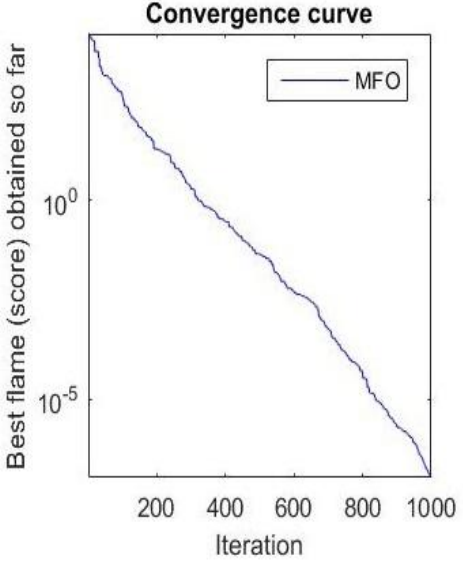
Problems Faced - Mass Updation

- Mass Updation according to previous mass
- Mass(i) = Mass(i) + Fitness(i) / Sum(Fitness(i))
- Effective increase in mass of moth was negligible in this case
- Replaced with Mass(i) = 1 / fitness(i)

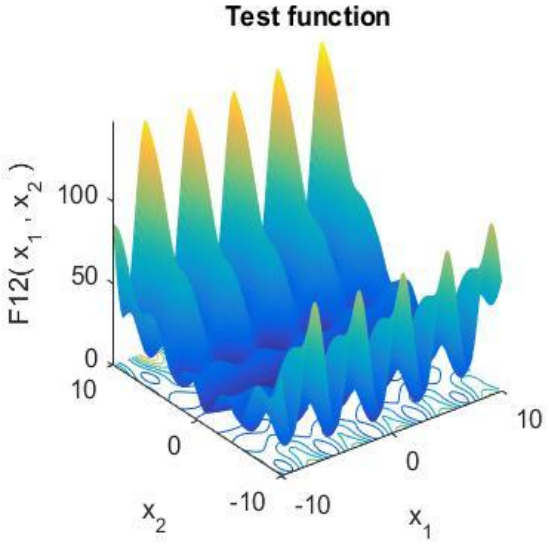
Results

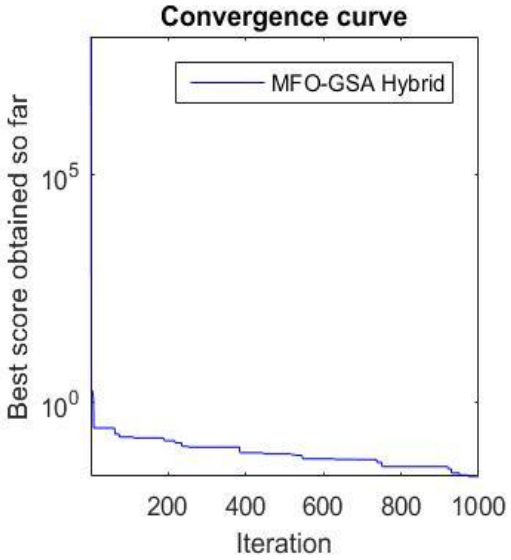

Test function
×10^4
4
2
0
F3(x1 , x2)
100
0
-100
x2
-100
0
100
x1

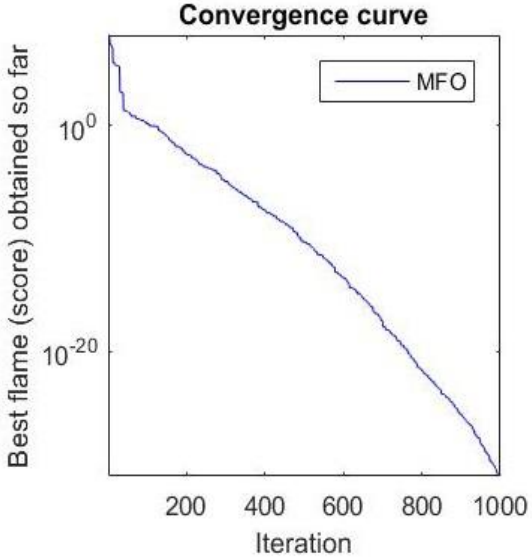

Convergence curve
MFO-GSA Hybrid
Best score obtained so far
10^0
10^-10
10^-20
200
400
600
800
1000
Iteration


Convergence curve
MFO
Best flame (score) obtained so far
10^0
10^-5
200
400
600
800
1000
Iteration

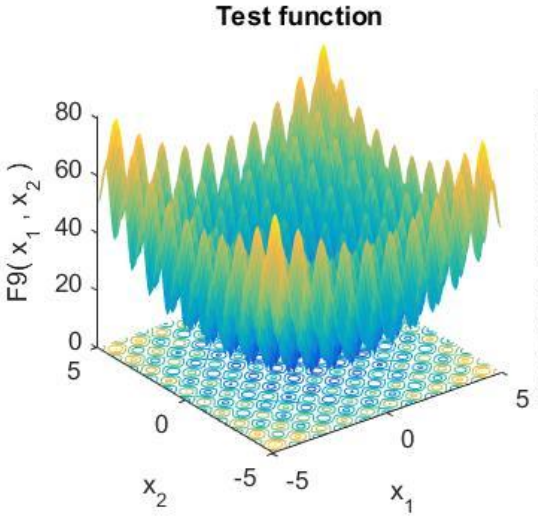
Results (Contd.)

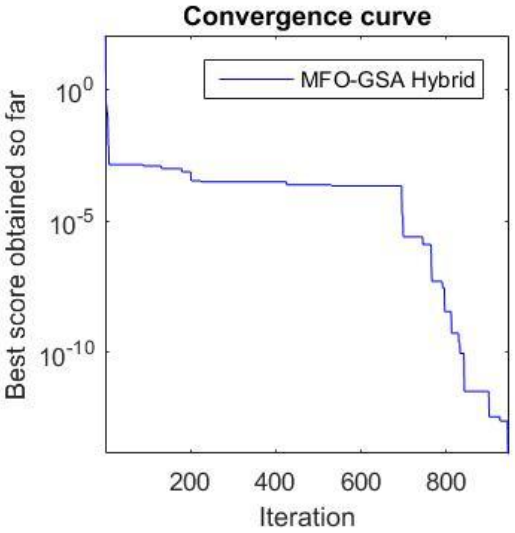


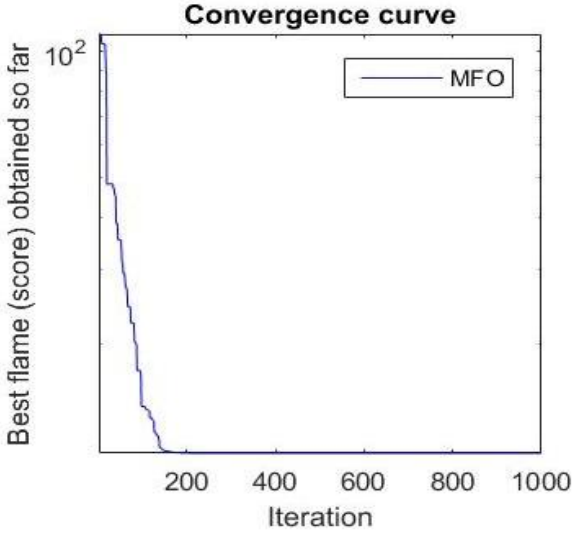




Results (Contd.)







Optimized K-Means

initialize each particle to contain k randomly selected cluster centroids
for t = 1 to tmax do
    for particle i do
        for each data vector d,
            dist(d, m_{ij}) to all cluster centroids C_{ij}
            assign d to cluster C_{ij}: dist(d, m_{ij}) = min {dist(d, m_{ij}}
            F = [F dist(d, m_{ij})]
        endfor
        C = best(F[])
    endfor
return C_{ij}

endfunction

Results of Optimized K-Means

Dataset	K-Means	Optimized K-Means
Set 1	1.11	1.11
Set 2	41.59	39.54
Iris	0.54	0.52
Cancer	29.12	27.46

Image Segmentation: K-Means

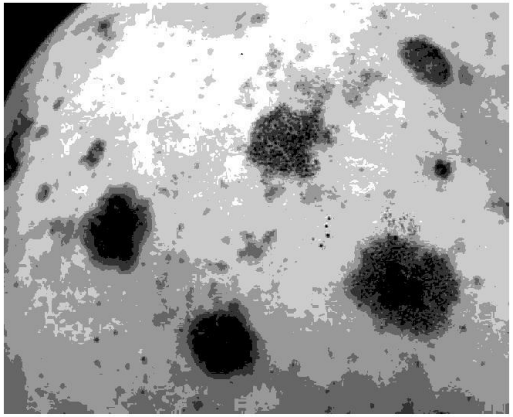
- Convert image to LAB color space
- Apply optimized K-Means on a and b

Image Segmentation: Multi-Level Thresholding

- Segment the image using thresholds
- Thresholding: manual process
- Brute force: Computationally expensive
- Optimization approach

Image Segmentation: Multi-Level Thresholding II

- Search for threshold values
- Fitness function: $S_{inter\_cluster} / S_{intra\_cluster}$



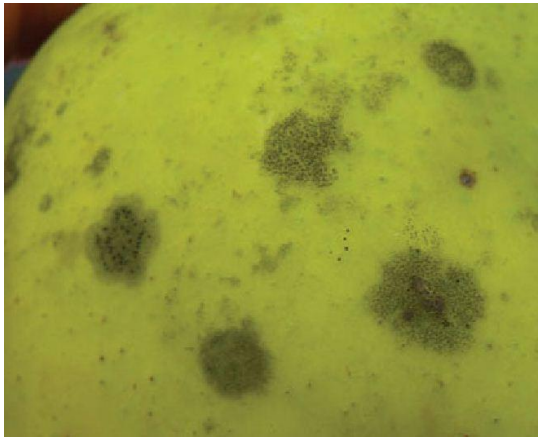


Image Segmentation: Multi-Level Thresholding III

Read Image

Add noise to the image

Generate image histogram

Initialize k moths with random positions and masses

for i = 1 to max iterations:

    while size(population) > size (new population):

        MFO+GSA hybrid locomotion

        $F_i = S_{inter\_cluster} / S_{intra\_cluster}$

        sort(F[])

    endwhile

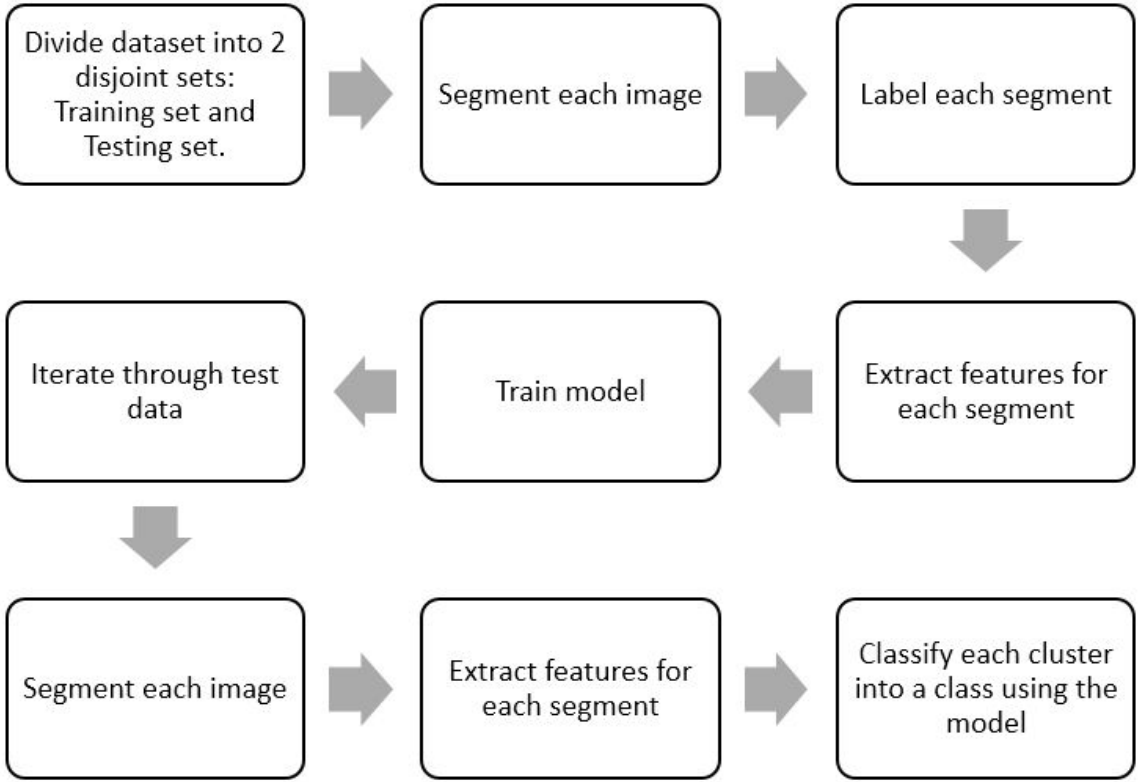
endfor

endfunction

Feature Extraction

- Local Binary Pattern
- Grey Level Co-occurrence Matrix
- Haralick Features

Classification



Future Work

- Collect gas sensor data
- Extend to other food items
- Explore other texture analysis for other food items