16.888
ESD.77

Multidisciplinary System Design Optimization (MSDO)

Multiobjective Optimization

Recitation 9

Andrew March

Today's Topics

- Multiobjective Optimization
  - Gradient-based methods
    - Find directly
    - Weighted sum
    - NBI
    - AWS
  - Heuristic Methods
    - General comments
    - MOGA
      - Fitness functions
      - Selection algorithms

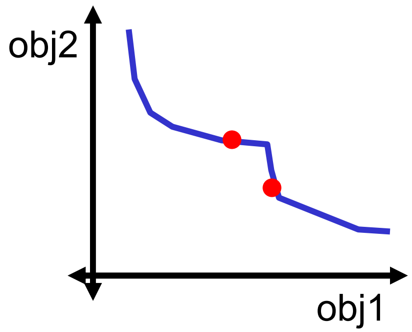
Gradient Method 1

- Choose n value of objective 1
- For each value of objective 1
  - Optimize objective 2

- Pros:
  - Fast
  - No convexity issues
- Cons:
  - Need to be able to fix an objective

Gradient Method 2

- Choose a set of n λ's $\in [0,1]$
- For each value of objective λ
  - Optimize $f=\lambda*obj1+(1-\lambda)obj2$



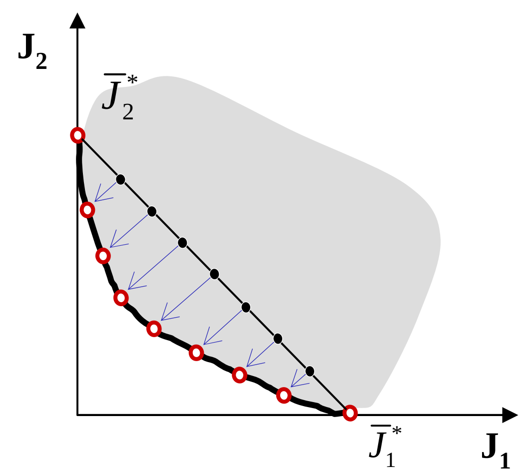
- Pros:
  - Fast
  - Can handle arbitrary number of objectives
- Cons:
  - Requires pareto front is convex

NBI

- Normal-Boundary Intersection
  - Das, I et al. 1998

- Perform single objective optimizations
- Choose n divisions between single objective optima
- Use goal programming along directions normal to current pareto front to find a feasible point.

- Pros:
  - Good distribution of points on pareto front
  - No issues of convexity
- Cons:
  - Computationally complex
  - Formulation is complex
  - Requires pareto filter

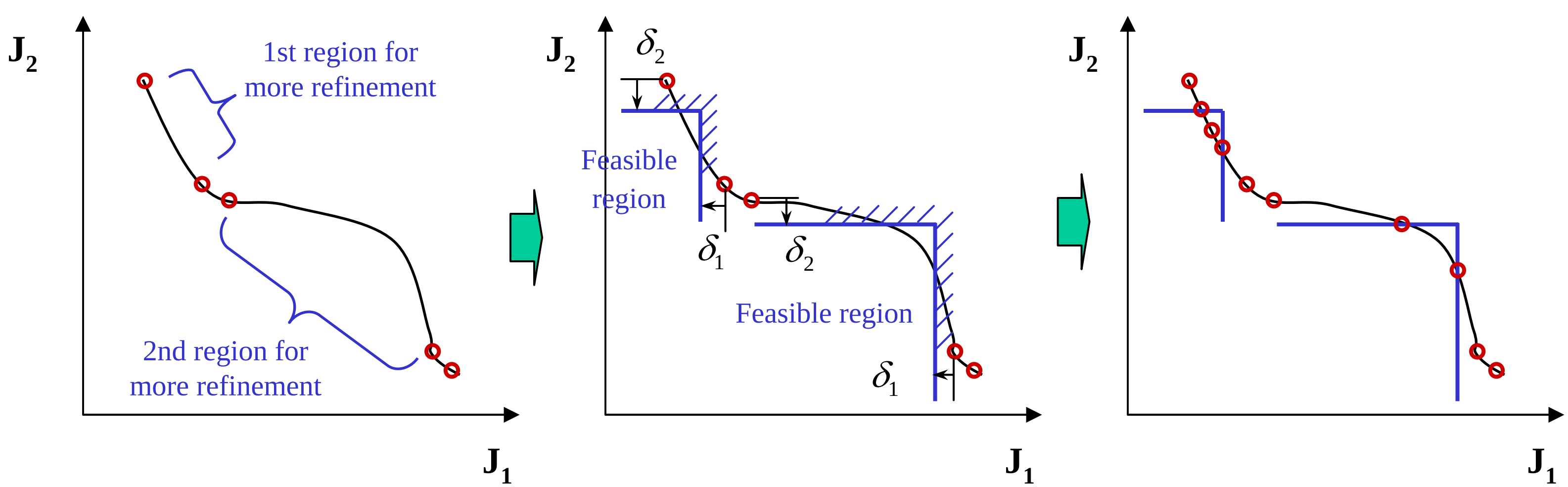


AWS

- Adaptive Weighted-Sum
  - Kim, I. Y., de Weck O. L, 2005

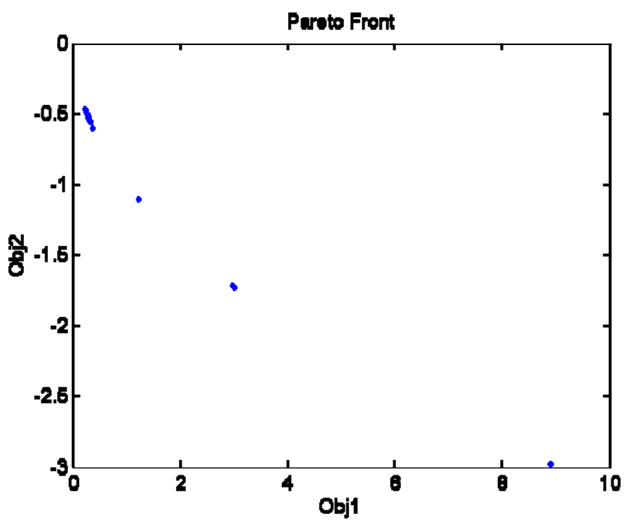
- Perform normal weighted sum optimization
- Select areas for refinements
- Add constraints and adapt objective function ratios

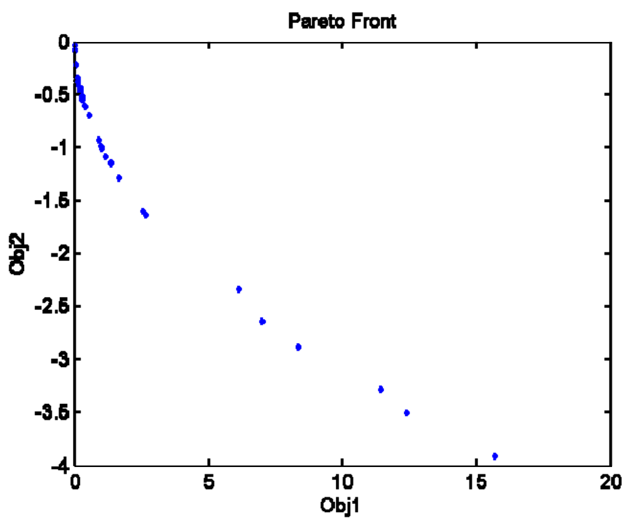
- Pros:
  - All solutions pareto optimal
  - Finds solutions evenly distributed on pareto front
- Cons:
  - Computationally expensive



General Comments

- You can be really creative with your fitness functions and selection
  - Can mitigate convexity issues
  - May not have to worry about scaling between objectives
- Can get even distribution on pareto front using a random process
  - May not have to force it, like AWS/NBI
  - High mutation rate (random variation) may be good:





- Computational expense gets LARGE.

MOGA-1

- P(selection)≈% population that a member dominates

- Pseudo code:
- Create probability of selection vector
- While(next gen size<current size)
  - i=1
  - while(i≤current size & next gen size<desired size)
    - x=rand(0,1)
    - If(x<P(selection))
      - Add member i to next gen
    - End if
    - i=i+1
  - End while
- End while

MOGA-2

- Multiobjective roulette wheel selection
- P(selection)=

  (number of members dominated by member i)/
  (sum of all dominations)

- Pseudo code:
- Create selection bins
  - Bin 1: LB_1=0, UB_1=P(member1)
  - Bin 2: LB_2=P(member1), UB_2=P(member1)+P(member2)…
- For i=1:n
  - x=rand(0,1)
  - Select member i with $x \in$ bin i
- End for

Demo

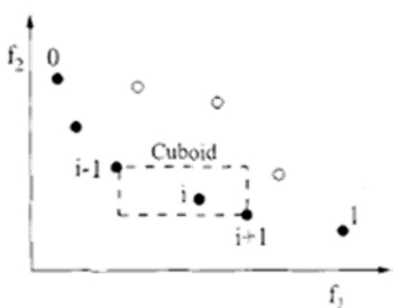
MOGA
using roulette wheel selection
(on stellar)

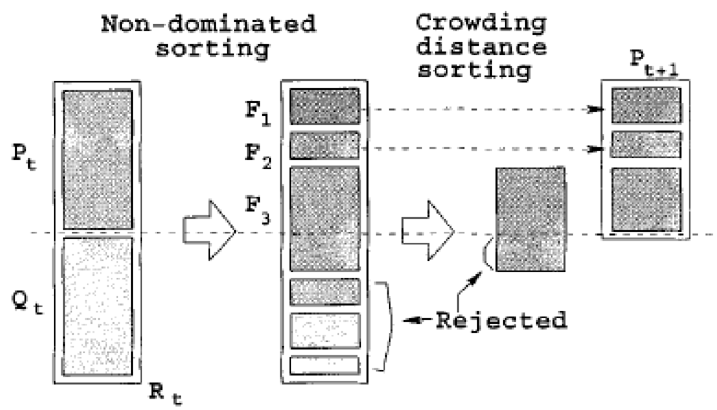
MOGA

- Formulations presented cover the pareto front well if:
  - Domination is a good fitness function
  - GA actually works well on this problem
  - Randomness alone is sufficient for spread
- Can we force spread on the pareto front with a GA?

NSGA-II

- Very commonly used Multiobjective GA
  - Deb, K. et al. 2002
- Pros:
  - No convexity issues, good spacing on pareto front
- Cons:
  - COMPUTATIONAL EFFORT!





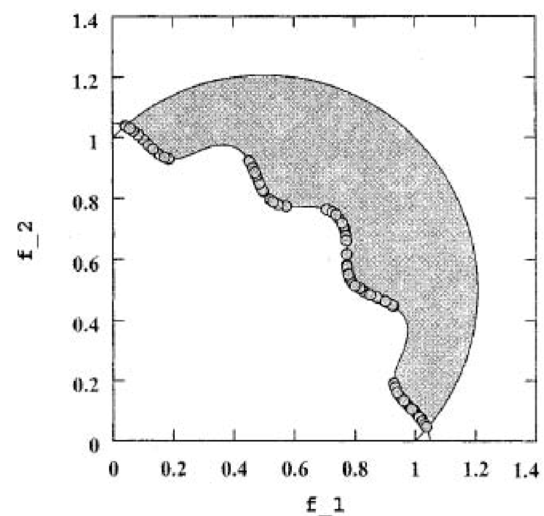


Fig. 20. Obtained nondominated solutions with Fonseca–Fleming's constraint-handling strategy with NSGA-II on the constrained problem TNK.

 Source: Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II." IEEE Transactions on Evolutionary Computation 6, no. 2 (2002): 182-97.

Summary

- You have many multiobjective optimization methods available to you.
  - And most already available as toolboxes!
- A5
  - The pareto front only requires continuous variables
  - Can use many of the methods discussed in here

ESD.77 / 16.888 Multidisciplinary System Design Optimization
Spring 2010