Evolutionary Practical Optimization

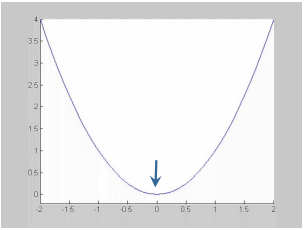
Kalyanmoy Deb
Professor of Mechanical Engineering
Indian Institute of Technology Kanpur
Kanpur, PIN 208016, India
Email: deb@iitk.ac.in
http://www.iitk.ac.in/kangal/deb.htm

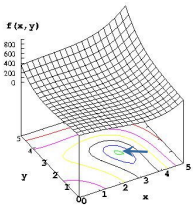
Outline of Tutorial

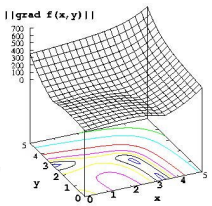
- Optimization fundamentals
- Scope of *optimization* in practice
- Classical *point-by-point* approaches
- Advantages of evolutionary *population-based* approaches
- Scope of evolutionary approaches in different problem solving tasks
  - Having one algorithm for various practical optimizations is difficult -> Customization is must
- Final words

Fundamentals of Optimization

- A generic name for minimization and maximization of a function *f(**x**)*
- Everyone knows: *df/dx=0* or $\nabla f(x)=0$
- Curse of dimensionality, multiple optima



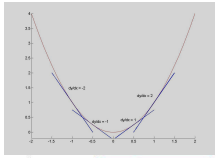


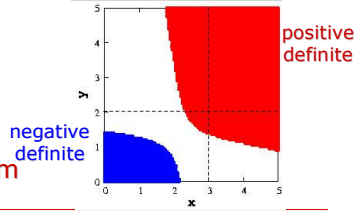


Fundamentals (cont.)

- Concept relates to mathematics
  - Second and higher-order derivatives
    $d^2f/dx^2>0$, minimum
    $d^2f/dx^2<0$, maximum

  if $\nabla^2 f$ is positive definite at x*, it is a minimum
  - Convex: One optimum





GECCO'07, July 7–11, 2007, London, England, United Kingdom.
ACM 978-1-59593-698-1/07/0007.

Constrained Optimization Basics

- Decision variables: $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$
- Constraints restrict some solutions to be feasible

$$\begin{aligned} &\text{Min. } f(\boldsymbol{x}) \\ &\text{s.t. } g_j(\boldsymbol{x}) \geq 0 \qquad j = 1,2,\ldots,J \\ &\qquad h_k(\boldsymbol{x})=0 \qquad k = 1,2,\ldots,K \\ &\qquad x_i^L \leq x_i \leq x_i^U \quad i = 1,2,\ldots,n \end{aligned}$$

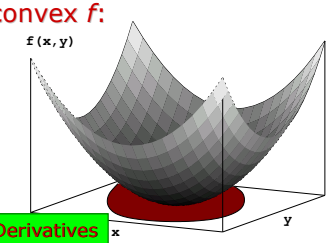
- Equality and inequality constraints
- Minimum of $f(\boldsymbol{x})$ need not be constrained minimum
- Constraints can be non-linear

Constrained Optimization Basics (cont.)

- Kuhn-Tucker (K-T) conditions for optimality
  - First-order necessary conditions
  - Convex search space, convex f: K-T point is minimum

$$\nabla f(x) - \sum_{j=1}^{J} u_j \nabla g_j(x) - \sum_{k=1}^{K} v_k \nabla h_k(x) = 0$$

$$\begin{aligned} &g_j(\boldsymbol{x}) \geq 0 && j = 1,2,\ldots,J \\ &h_k(\boldsymbol{x})=0 && k = 1,2,\ldots,K \\ &u_j g_j(\boldsymbol{x})=0 && j = 1,2,\ldots,J \\ &u_j \geq 0 && j = 1,2,\ldots,J \end{aligned}$$



Involve Derivatives and solve for roots

Duality Theory in Optimization

- By using constraints, a primal problem has an equivalent dual problem
- Dual problem is always concave
  - Comparatively easier to solve
- Theoretical results:
  - Convex problems and in some special cases:
    - Optimal primal and dual function values are same
  - Generic cases:
    - Optimal dual function value underestimates optimal primal function value
- Beneficial to use duality theory in many problems

Theory is not practical, but prudent

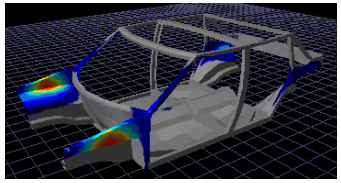
- provides support for customization Theory often not applicable in practice
  - Gradients do not always exist
  - Theory does not exist for generic problems
- But good to know
  - Know limitation of theory
  - Know extent of theory
  - Often may lead to better algorithm development
- `No Free Lunch' theorem

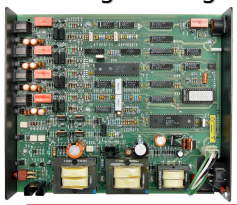
No Free Lunch (NFL) Theorem

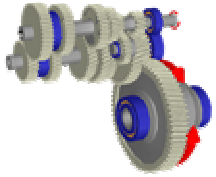
- No one gives other a free lunch
- In the context of *optimization*
  - Wolpert and McCardy (1997)
  - Algorithms A1 and A2
  - All possible problems **F**
  - Performances P1 and P2 using A1 and A2 for a fixed number of evaluations
  - P1 = P2
- NFL breaks down for a narrow class of problems or algorithms
- Research effort: Find the best algorithm for a class of problems
  - Unimodal, multi-modal, quadratic etc.

Scope of Optimization in Practice

- *Optimal design & manufacturing* for desired goals
- Major application in engineering

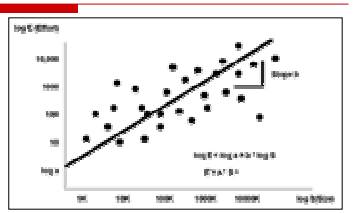






Scope of Optimization (cont.)

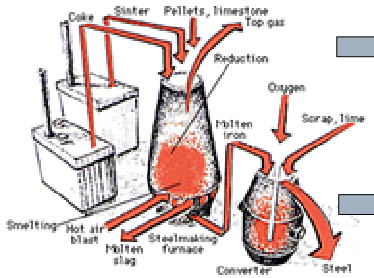
- *Parameter optimization* for optimal performance



- Scientific experiments, computer experiments

Scope of Optimization (cont.)

- *Modeling* (system, process)



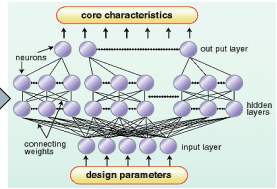
$$(1)\ \frac{\partial P}{\partial t} = -\gamma P + \beta \mathcal{U}_A - \kappa P S_A$$

$$(2)\ \frac{\partial I_A}{\partial t} = r I_A - g_A I_A + \kappa P S_A - \lambda I_A + m b_{B\to A} I_B - m b_{A\to B} I_A$$

$$(3)\ \frac{\partial S_A}{\partial t} = r S_A - g_A S_A - \kappa P S_A + m b_{B\to A} S_B - m b_{A\to B} S_A$$

$$(4)\ \frac{\partial I_B}{\partial t} = -g_B I_B - m b_{B\to A} I_B + m b_{A\to B} I_A$$

$$(5)\ \frac{\partial S_B}{\partial t} = -g_B S_B - m b_{B\to A} S_B + m b_{A\to B} S_A$$

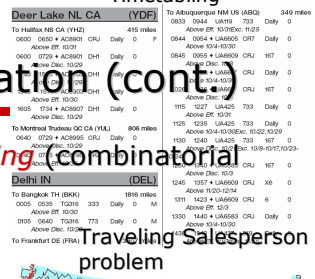


Scope of Optimization (cont.)

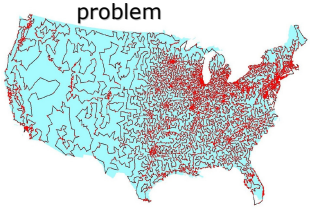
- *Scheduling and planning* (combinatorial optimization)

Routing & Scheduling

Timetabling

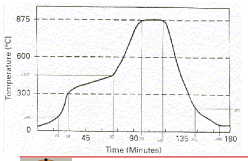


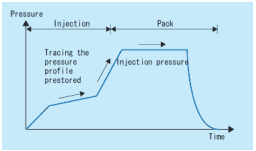
Traveling Salesperson problem



Scope of Optimization (cont.)

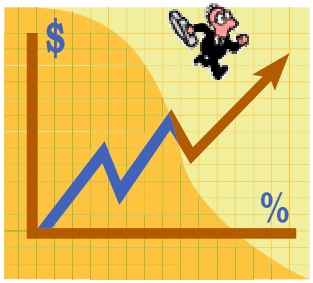
- *Optimal control*
- Time-variant profiles are to be found
  - How to lower load?
  - How to control temp, pressure?





Scope of Optimization (cont.)

- *Forecasting and prediction*



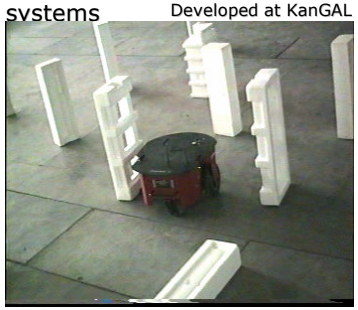
Scope of Optimization (cont.)

- *Data mining* (classification, clustering, pattern recognition)

Scope of Optimization (cont.)

- *Machine learning*
  - Designing intelligent systems

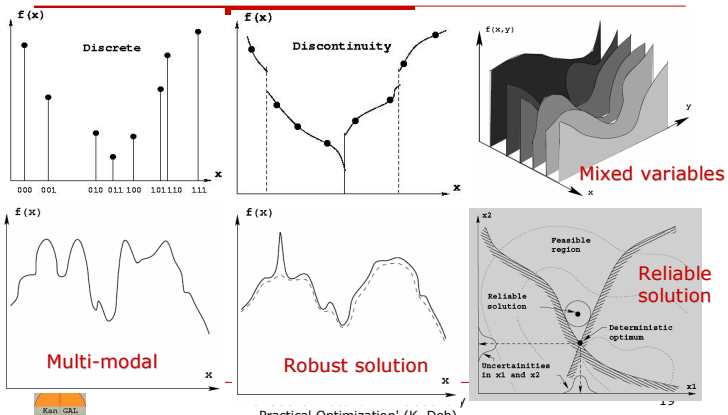
Developed at KanGAL



Properties of Practical Optimization Problems

- Non-differentiable functions and constraints
- Discontinuous search space
- Discrete search space
- Mixed variables (discrete, continuous, permutation)
- Large dimension (variables, constraints, objectives)
- Non-linear constraints
- Multi-modalities
- Multi-objectivity
- Uncertainties in variables
- Computationally expensive problems
- Multi-disciplinary optimization

Different Problem Complexities



Classical Optimization Methods and Past

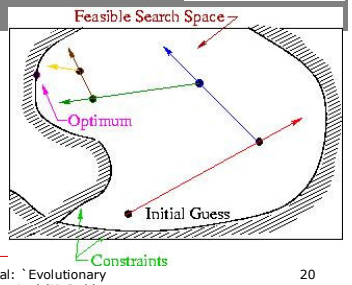
- Exact differentiation & root-finding method
  - Intractable and not sufficient for practical problems
- Numerical algorithms
  - Iterative and deterministic
  - Directions based on gradients (mostly)
- Point-by-point approaches

Linear regression:

y=mx+b

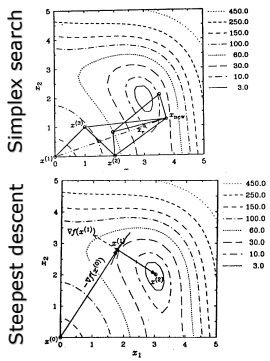
$$m = \frac{n\sum(xy) - \sum x \sum y}{n\sum(x^2) - (\sum x)^2}$$

$$b = \frac{\sum y - m\sum x}{n}$$



Classical Methods (cont.)

- Direct and gradient based methods
- Convexity assumption
  - No guarantee otherwise
- Local perspectives
- Discreteness cause problems
- Non-linear constraints
- Large-scale application time-consuming
- Serial in nature



Practical Optimization

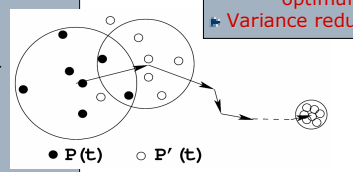
With a point in each iteration, scope is limited

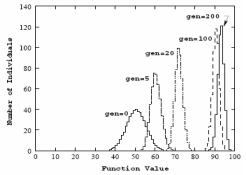
A need for non-classical, population-based optimization methods exists

Evolutionary Algorithms as Optimizers

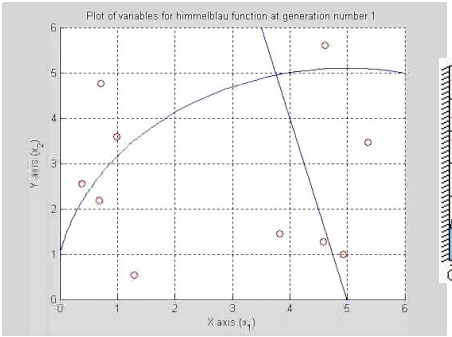
- Mean approaches optimum
- Variance reduces

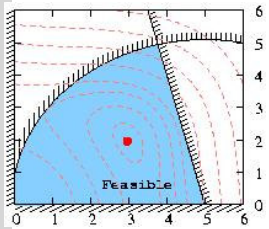
begin
Solution Representation
t := 0; // generation counter
Initialization P(t);
Evaluation P(t);
while not Termination
do
P'(t) := Selection (P(t));
P''(t) := Variation (P'(t));
Evaluation P''(t);
P(t+1):= Survivor (P(t),P''(t));
t := t+1;
od
end





Simulation of a GA

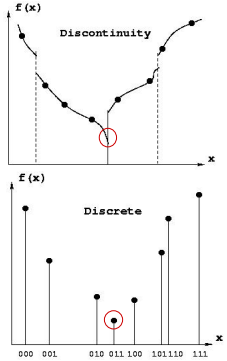




Non-smooth Problems

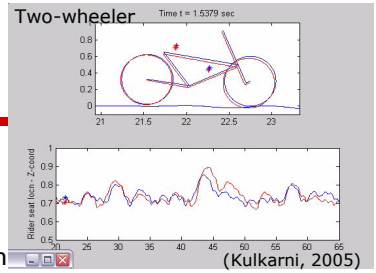
Handling discrete, discontinuous variables

- Non-differentiability, discontinuity, discreteness, non-linearity not a difficulty
- GAs work with a discrete search space anyway
  - So, GAs are natural choice for discrete problem solving

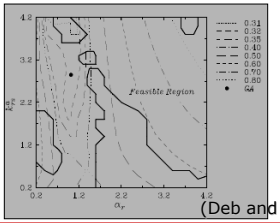


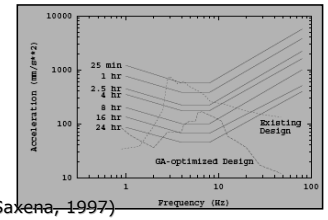
Car Suspension Design

- Practice is full of non-linearities
- MATLAB gets stuck
- An order of magnitude better than existing design



(Kulkarni, 2005)





(Deb and Saxena, 1997)

EAs for Continuous Variables

- Decision variables are coded directly, instead of using binary strings
- Recombination and mutation need structural changes
- Selection operator remains the same

Recombination

$$(x_1 x_2 \ldots\ldots\ldots\ldots x_n)$$
$$(y_1 y_2 \ldots\ldots\ldots\ldots y_n)$$
$$\Rightarrow ?$$

Mutation

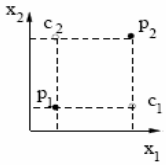
$$(x_1 x_2 \ldots\ldots\ldots\ldots x_n) \Rightarrow ?$$

- Simple exchanges are not adequate

Naive Recombination

- Crossing at boundaries do not constitute adequate search
  - Least significant digits are taken too seriously

15.345	3.569	→	11.142	3.587
21.142	5.687		25.345	5.669



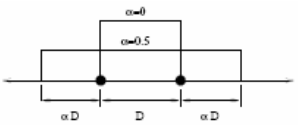
- Two Remedies:
  - Parent values (variable-wise) need to be blended to each other
  - Vector-wise recombination

Variable-wise Blending of Parents

- Use a probability distribution to create child
- Different implementations since 1991:
  - Blend crossover (BLX-α), 1991
  - Simulated binary crossover (SBX-β), 1995
  - Fuzzy recombination (FR-d), 1995
  - Self-adaptive evolution strategy (ES-τ), 1987
  - Differential evolution (DE-CR-F), 1996
  - Particle swarm optimization (PSO-param.)
- Main feature: Difference between parents used to create children
  - Provides a self-adaptive property

Blend Crossover (BLX-α) (extend)

- Uniform probability distribution within a bound controlled by α (Eshelman and Schaffer, 1991)



- Diversity in children proportional to that in parents
- Too wide a search, if parents are distant

Motivation for the SBX Operator

- Simulate processing in a binary crossover, say single-point crossover
- *Gedanken* experiment:
  - Two parent solutions in real number
  - Code them in l-bit strings
  - Use single-point crossover in all (l-1) places and find children strings
  - In each case map strings back to real numbers as children
  - Observe the relationship between parents and children
  - Use this relationship to directly recombine parents to form children

Properties of Binary Crossover

- To make crossovers independent of parents, define a spread factor, β

$$\beta = \left|\frac{c_1 - c_2}{p_1 - p_2}\right|$$

- Define probability distribution as a function of β
- Two observations:
  - Mean decoded value of parents is same as that of children
  - Child strings crossed at the same place produce the same parent strings

B_1				A_1			DV
1	0	1	0	1	0	1	85
0	1	1	0	0	1	1	51
B_2				A_2			68

$\Longrightarrow$

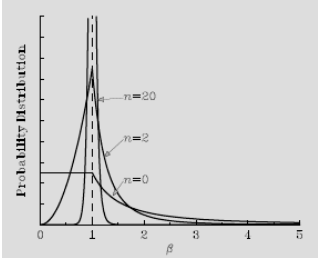
B_1				A_2			DV
1	0	1	0	0	1	1	83
0	1	1	0	1	0	1	53
B_2				A_1			68

Simulated Binary Crossover (SBX-β)

- Probability is large near the parents
- Extent is controlled by β
- Diversity in children proportional to that in parents
- Obeys *interval-schema* processing

$$\mathcal{C}(\beta) = 0.5(n+1)\beta^{\eta}$$

$$\mathcal{E}(\beta) = 0.5(n+1)\frac{1}{\beta^{n+2}}$$



SBX Procedure

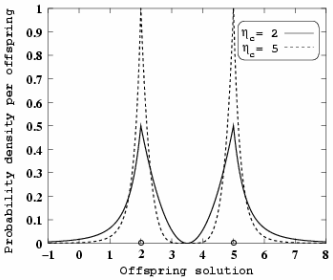
- Step 1: Choose a random number $u \in [0,1]$.
- Step 2: Calculate β_q:

$$\beta_q = \begin{cases} (2u)^{\frac{1}{\eta_c+1}}, & \text{if } u \le 0.5 \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{\eta_c+1}}, & \text{otherwise} \end{cases}$$

- Step 3: Compute two offspring:

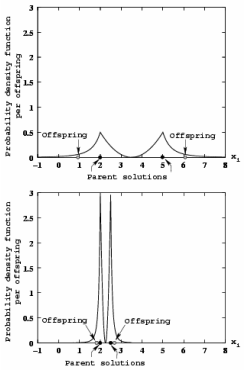
$$c_1 = 0.5\left((1+\beta_q)p_1 + (1-\beta_q)p_2\right)$$

$$c_2 = 0.5\left((1-\beta_q)p_1 + (1+\beta_q)p_2\right)$$



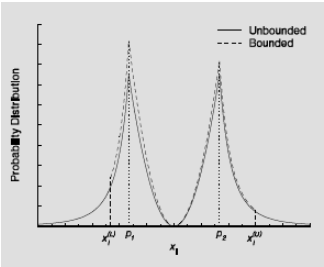
Properties of SBX Operator

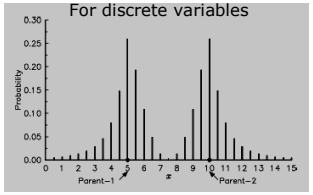
- If parents are distant, distant offspring are likely
- If parents are close, offspring are close to parents
- Self-adaptive property



Variations of SBX

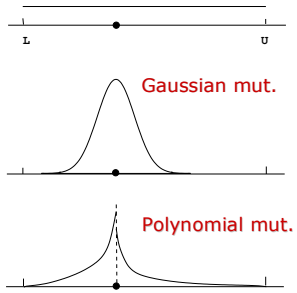
- For bounded and discrete variables





Real-Parameter Mutation Operators

- Idea: Create a neighboring solution
  - Random mutation
  - Normally distributed mutation
  - Non-uniform mutation
  - Extensions to bounded and discrete cases exist

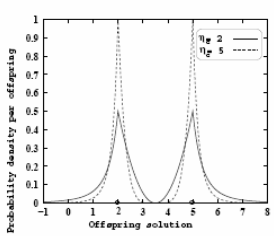


Vector-Wise Recombination Operators

- Variable-wise recombination cannot capture nonlinear interactions
- Recombine parents as vectors
  - Parent-centric recombination (PCX)
  - Unimodal normally-distributed crossover (UNDX)
  - Simplex crossover (SPX)
- Difference between parents is used to create offspring solutions
- DE, PSO, CMA-ES

Parent-Centric Recombination

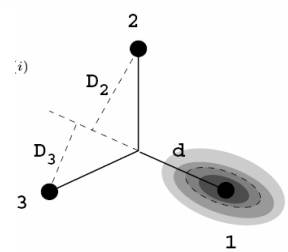
- SBX (Deb and Agrawal, 1995): variable-wise
- Vector-wise parent-centric recombination (PCX): O(μ) per offspring (Deb, Anand & Joshi, 2002)



PCX Operator

- μ parents create a offspring
- Each parent has its turn
- $e^{(i)}$ orthonormal bases spanning the subspace perpendicular to $d^{(p)}$
- w_ς and w_η are user-defined parameters, controlling extent of search

$$\vec{y} = \vec{x}_p + w_\varsigma |\vec{d}^{(p)}| + \sum_{i=1,\ i\neq p}^{\mu} w_\eta \bar{D}\bar{e}^{(i)}$$



Generalized Generation Gap (G3) Model (Steady-state approach)

1. Select the best parent and μ-1 other parents randomly
2. Generate λ offspring using a recombination scheme
3. Choose two parents at random from the population
4. Form a combination of two parents and λ offspring, choose best two solutions and replace the chosen two parents

- Parametric studies with λ and N
- Desired accuracy in F is 10^{-20}

Quasi-Newton Method

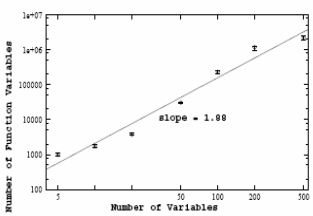
- Accuracy obtained by G3+PCX is 10^{-20}

Func.	FE	Best	Median	Worst
F_{elp}	6,000	$8.819(10^{-24})$	$9.718(10^{-24})$	$2.226(10^{-23})$
F_{sch}	15,000	$4.118(10^{-12})$	$1.021(10^{-10})$	$7.422(10^{-9})$
F_{ros}	15,000	$6.077(10^{-17})$	$4.046(10^{-10})$	3.987
F_{elp}	8,000	$5.994(10^{-24})$	$1.038(10^{-23})$	$2.226(10^{-23})$
F_{sch}	18,000	$4.118(10^{-12})$	$4.132(10^{-11})$	$7.422(10^{-9})$
F_{ros}	26,000	$6.077(10^{-17})$	$4.046(10^{-10})$	3.987

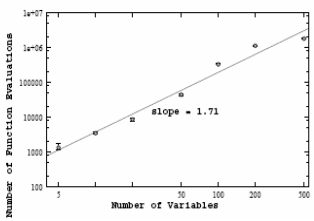
Scalability Study

- Accuracy 10^{-10} is set

Ellipsoidal

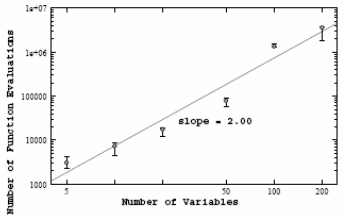


Schwefel



Scalability Study (cont.)

Rosenbrock



- All polynomial complexity $O(n^{1.7})$ to $O(n^2)$ similar to those reported by CMA-ES approach (Hansen and Ostermeier, 2001)

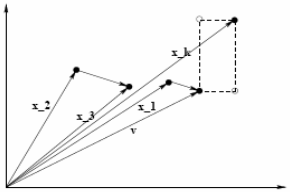
Differential Evolution (DE) (extend)

1. Start with a pool of random solutions
2. Create a child v
3. x_k and v are recombined with *p*

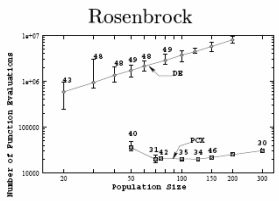
- Difference of parents in creating a child is important
- A number of modifications exist

– $\mathbf{v} = \mathbf{x}^{(1)} + \lambda(\mathbf{x}^{(2)} - \mathbf{x}^{(3)})$

– $y_i = \begin{cases} v_i, & \text{with a prob. } p \\ \mathbf{x}_i^{(k)}, & \text{else} \end{cases}$



DE Results



	F_{elp}		
	Best	Median	Worst
DE	9,660	12,033	20,881
G3	5,826	6,800	7,728
	F_{sch}		
DE	102,000	119,170	185,590
G3	13,988	15,602	17,188
	F_{ros}		
DE	243,800	587,920	942,040
G3	16,508	21,452	25,520

Particle Swarm Optimization (PSO) (Delete)

- Kennedy and Eberhart, 1995
- Particles fly through the search space
- Velocity dynamically adjusted
- $x_i = x_i + v_i$
- $v_i = v_i + c_1 rnd()(p_{i,best} - x_i) + c_2 rnd()(p_g - x_i)$
- p_i: best position of i-th particle
- p_g: position of best particle so far
  - 1st term: momentum part (history)
  - 2nd term: cognitive part (private thinking)
  - 3rd term: social part (collaboration)

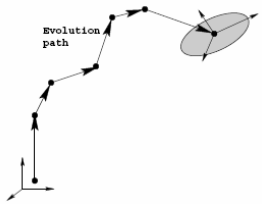
PSO Algorithm

1. Initialize a random population
2. For each particle
   - Evaluate and update p_{best}
3. Identify p_g
4. Compute velocity and position for each particle
5. Loop till termination

- Modifications exist
- Constrained PSO, Hybrid PSO, multi-objective PSO, etc.

CMA-ES (Hansen & Ostermeier, 1996)

- Selecto-mutation ES is run for n iterations
- Sucessful steps are recorded
- They are analyzed to find uncorrelated basis directions and strengths
- Required $O(n^3)$ computations to solve an eigenvalue problem
- Rotation invariant



CMA-ES On Three Test Problems

EA	F_{elp} Best	F_{elp} Median	F_{elp} Worst	F_{sch} Best	F_{sch} Median	F_{sch} Worst
CMA-ES	8,064	8,472	8,868	15,096	15,672	16,464
DE	9,660	12,033	20,881	102,000	119,170	185,590
G3+PCX	5,826	6,800	7,728	13,988	15,602	17,188

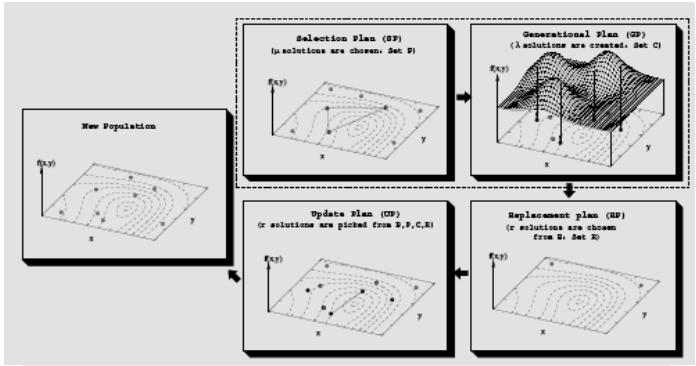
	F_{ros} Best	F_{ros} Median	F_{ros} Worst
CMA-ES	29,208	33,048	41,076
DE	243,800	587,920	942,040
G3+PCX	16,508	21,452	25,520

Accuracy 1×10^{-20}

Population-Based Optimization Algorithm-Generator

- Four functionally different Plans:
  - Selection plan (SP): choose μ solutions from B to create P
  - Generation plan (GP): create λ solutions (C) using P
  - Replacement plan (RP): choose r solutions (R) from B
  - Update plan (UP): update B by replacing R (r solutions) from (P,C,R)

A Sketch of an Iteration



Designing an Optimization Procedure

1. Choose SP, GP, RP and UP
2. Perform a parametric study on a set of test problems
   - N, μ, λ, r

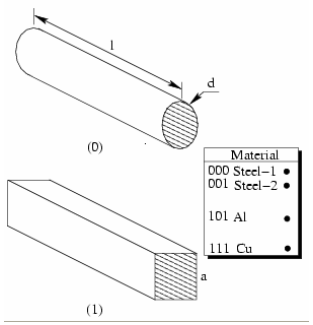
- Most classical and non-classical optimization algorithms explained
- Interestingly, PSO, DE, ES, EP can also be explained
- Possibility for a unified optimization approach

Mixed-Variable Optimization:
Handling mixed type of variables

- Treat type of cross-sections, materials, etc. as decision variables
- A mixed representation:

(1) 14 23.457 (101)

  - (1): circular or square cross-section
  - 14: diameter/side
  - 23.457: length
  - (101): material
- Permutation + real + cont.
- Complete optimization
- Deb and Goel, ASME-JMD, 1997



Constraint Handling:
Handling non-linear constraints

- Inequality constraints $(g_j(x) \geq 0)$ penalized for violation:

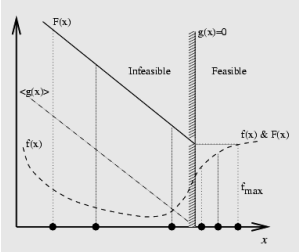
$$F(x) = f(x) + \sum_{j=1}^{J} R_j \langle g_j(x) \rangle^2$$

  - ‹a› = a if a is –ve, 0 otherwise
- Performance sensitive to penalty parameters

R	≤ 50%	Infeasible	Best	Median	Worst
10^0	12	13	2.41324	7.62465	483.50177
10^1	12	0	3.14206	4.33457	7.45453
10^3	1	0	3.38227	5.97060	10.65891
10^6	0	0	3.72929	5.87715	9.42353

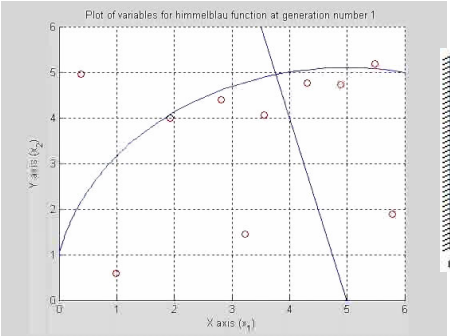
A Penalty-Parameter-less Population-based Approach

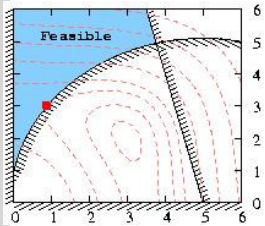
- Modify tournament sel.:
  - A feasible is better than an infeasible
  - For two feasibles, choose the one with better *f*
  - For two infeasibles, choose the one with smaller constraint violation $\left(\sum_j \langle g_j(x) \rangle\right)$
  - (Deb, 2000)



$$F(x) = \begin{cases} f(x), & \text{if } g_j(x) \geq 0, \forall j \in J \\ f_{\max} + \sum_{j=1}^{J} \langle g_j(x) \rangle, & \text{otherwise} \end{cases}$$

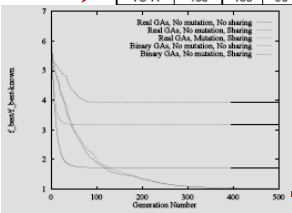
A Computer Simulation





Welded-Beam Design Problem

Method	Mutation	sharing	≤ 1%	≤ 2%	≤ 5%	≤ 10%	≤ 20%	≤ 50%	> 50%
Maximum generations = 500									
TS-B	No	No	0	0	0	0	0	0	50
TS-B	No	Yes	0	0	0	0	0	0	50
TS-R	No	No	0	0	1	4	8	16	34
TS-R	No	Yes	28	36	44	48	50	50	0
Maximum generations = 4,000									
TS-R	No	Yes	28	37	44	48	50	50	0
TS-R	Yes	Yes	50	50	50	50	50	50	0



Case	Best	Median	Worst
1	4.939027	7.183082	12.084255
2	3.820984	8.899963	14.298933
3	2.442714	3.834121	7.444246
4	2.381191	2.392892	2.645833
5	2.381187	2.392025	2.645833
6	2.381447	2.382634	2.383554

Large-Scale Optimization:
Handling large number of variables

- Large n, large pop-size, large computation
- Knowledge-augmented EAs
  - Representation
  - Operators
- EA's flexibility shows promise
- A case study involving millions of variables (Deb and Reddy, 2001)

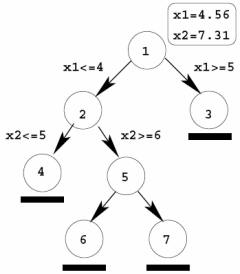
Casting Scheduling

Casting Scheduling Problem (cont.)

- Maximum metal utilization:

$$\frac{1}{H}\sum_{i=1}^{H}\frac{100\sum_{k=1}^{K} w_k x_{ki}}{W_i}$$

- Constraints:
  - Demand satisfaction: $\sum_{i=1}^{H} x_{ki} = r_k$ for $k = 1, \ldots, K$
  - Capacity constraint: $\sum_{k=1}^{K} w_k x_{ki} \le W_i$ for $i = 1, \ldots, H$



- An integer linear program (ILP)
- Branch-and-bound – exponential algorithm

Performance of LINGO

- Works up to n=500 on a Pentium IV (7 hrs.)

LINGO MILP Solver												
Heat No.	Order Number										Utilization/ Cruc. Size	Efficiency (%)
	1	2	3	4	5	6	7	8	9	10		
1	0	1	1	0	0	0	2	1	0	0	623/650	95.85
2	2	0	0	0	1	0	0	0	2	0	615/650	94.62
3	1	0	0	1	3	1	0	0	0	0	611/650	94.00
4	2	0	0	0	1	0	0	1	0	0	645/650	99.23
5	0	0	0	1	0	2	0	0	1	6	612/650	94.15
6	1	1	0	0	2	1	0	0	0	0	591/650	90.92
7	0	0	2	2	1	0	0	0	2	0	585/650	90.00
8	0	3	0	0	0	1	0	0	1	0	611/650	94.00
9	0	2	3	0	1	0	0	0	0	0	650/650	100.00
10	1	0	0	5	0	0	0	0	1	0	635/650	97.69
	7	7	6	9	9	5	2	2	7	6	Average	95.05

Off-The-Shelf GA Results

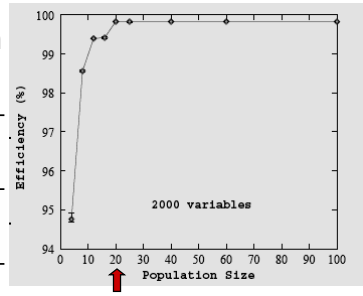
Number of Variables	Binary-coded GAs: Population Size	Efficiency	Function Eval.	Real-coded GAs: Population Size	Efficiency	Function Eval.
100	100	96.15	13,600	100	95.94	23,740
200	300	95.01	1,42,200	200	92.81	1,21,760
300	1,000	90.11	14,12,400	700	95.14	5,84,220

- Exponential function evaluations
- Random initialization, standard crossover and mutations are not enough
- Need a customized EA

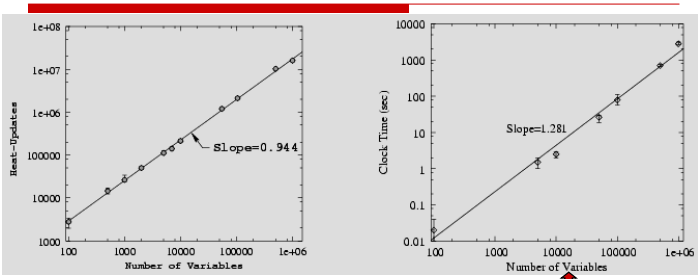
A Customized GA: Optimal Population Size

- N=2000 variables with max. gen.=1000/N
- A critical population size is needed

N:	4	8	12	16
# Heats:	211	204	201	201
N:	20	25	40	100
# Heats:	200	200	200	200



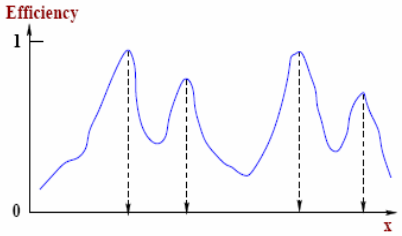
Scale-Up Results



- Knowledge-augmented GA has sub-quadratic complexity and up to one million variables (Deb and Pal, 2003)
- Never before such a large problem was solved using EAs

Multi-Modal Optimization:
Handling multiple optimal solutions

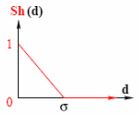
- To solve problems with multiple local/global optimum
- Classical methods can find only one optimum at a time
- EAs can, in principle, find multiple optima simultaneously

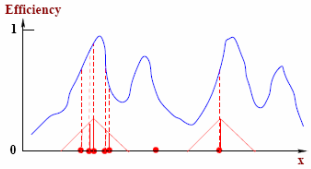


Niching Concept

Multiple niches (human and animal) coexist in nature

- By sharing resources (land, food, etc.)
- How to mimmick the concept in EAs?
- Reduce selection pressure for crowded solutions
- Use a sharing function based on two-armed bandit problem

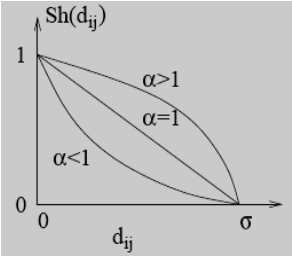




Sharing Function

- Goldberg and Richardson (1997)
- d is a distance measure between two solns.
  - Phenotypic distance: $d(x_i,x_j)$, x: variable
  - Genotypic distance: $d(s_i,s_j)$, s: string
- Calculate niche count, $nc_i=\Sigma_j Sh(d_{ij})$
- Shared fitness: $f_i'=f_i/nc_i$
- Use proportionate selection operator

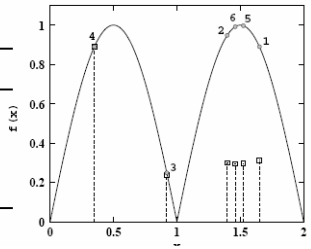
$$Sh(d_{ij}) = \begin{cases} 1-\left(\frac{d_{ij}}{\sigma}\right)^{\alpha} & \text{if } d_{ij} \le \alpha, \\ 0 & \text{otherwise.} \end{cases}$$



An Example: Phenotypic Sharing

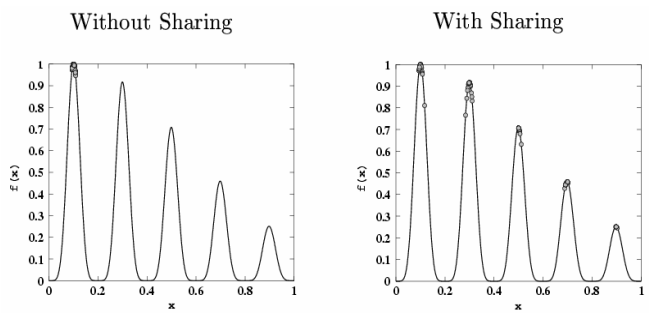
- Maximize $f(x) = |\sin(\pi x)|$, $x \in [0,2]$
- $nc_i = \sum_{j=1}^{N} Sh(d_{ij})$

Sol. i	String	$x^{(i)}$	f_i	nc_i	f_i'
1	110100	1.651	0.890	2.856	0.312
2	101100	1.397	0.948	3.160	0.300
3	011101	0.921	0.246	1.048	0.235
4	001011	0.349	0.890	1.000	0.890
5	110000	1.524	0.997	3.364	0.296
6	101110	1.460	0.992	3.364	0.295

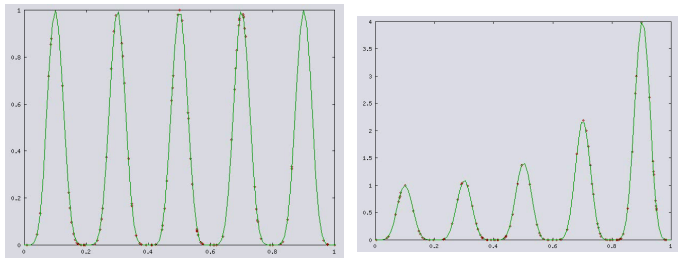


Simulation Results

- Inclusion of niche-formation strategy

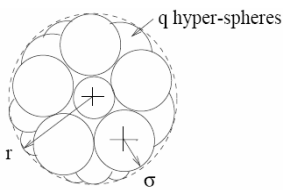


Simulation on Two Problems



Sizing Niching Parameter: Phenotypic σ_{share}

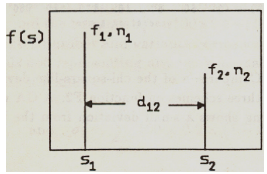
- Assume number of optima: q
- 1-D space:
  $\sigma_{share} = (x_{max}-x_{min})/2q$
- p-D space:
  - r: Euclidean length of radius of entire p-D hyper-sphere
  - Volume: $V = cr^p$
  - Divide volume into q parts
  - σ_{share} is the radius of each part
  - $V = cr^p = q\, c\sigma_{share}^p$ yields $\sigma_{share} = r/q^{1/p}$



Genotopic Sharing

- d=Hamming distance (# of bit differences)
- q niches: $\frac{1}{2^l}\prod_{i=0}^{\sigma}\binom{l}{i} = \frac{1}{q}$
- Approximate: $\sigma = \frac{1}{2}\left(l + z^*\sqrt{l}\right)$
- z* found from cum. Normal distribution chart for 1/q
- Restriction on gen. sharing:
  - Need a minimum distance between optima ($\gamma = f_1/f_2$)

$$d \geq \left(1 - \frac{1}{\gamma}\right)\sigma$$



Other Niching Approaches

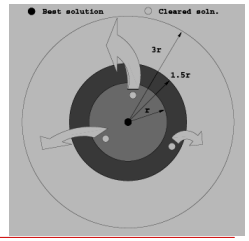
- Clearing approach
- Clustering approach
- Crowding approach
- Pre-selection approach
- Restrictive tournament selection approach
- All require at least one tunable parameter

Clearing Strategy

- Clear bad near good solutions
  - Sort population according to fitness
  - Keep the best and assign zero fitness to all others within *d* from best
  - Keep the next best and assign zero fitness to others within *d* from next best
  - Continue
  - Perform a selection with assigned fitness
- Petrowski (1996)

Modified Clearing Strategy

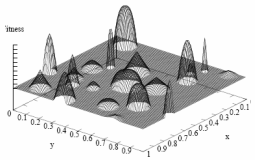
- Most members are wasted
- A better approach
  - All zero-fitness solutions are re-initialized outside 1.5*r* of best
  - Performs better
- Singh and Deb (GECCO- 2006)



Clustering Approach

- Based on phenotypic space, a clustering is performed
  - With a predefined *d*
  - With a predefined number of clusters
  - Niching within each cluster
  - Mating restriction can also be performed
- Parameter sensitive
- Yin and Germay (1993)

Comparison on a Scalable Problem (Singh and Deb, 2006)



- Optima are placed randomly
- Gaussian shape
- A8 (Mod. Clearing) performs the best
  - A1: Clearing
  - A2: Clustering
  - A4: Crowding

No. of Peaks		A1	A2	A4	A8
20	SR	15.78	19.0	19.28	**20.0**
	SD	4.18	1.44	1.20	**0.0**
	TT	112.6	23.39	120.4	2156.7
	SD	54.69	4.52	69.15	1635.4
30	SR	25. 0	27.38	28.58	**30.0**
	SD	3.02	2.82	1.56	**0.0**
	TT	155.0	36.72	144.8	2526.8
	SD	219.6	3.83	245.6	1854.5
40	SR	35.14	36.24	38.12	**40.0**
	SD	4.50	2.80	2.23	**0.0**
	TT	121.6	120.7	125.0	2967.9
	SD	15.14	68.94	9.23	1797.6
50	SR	44.86	42.2	47.6	**50.0**
	SD	2.53	4.40	2.43	**0.0**
	TT	321.8	222.8	335.6	3326.0
	SD	64.14	60.92	138.9	2142.7

Optimization with Meta-Models:

Handling computationally expensive problems

- Evaluation of most real-world problems is computationally expensive
- Optimization algorithm run into days
- To save time, use approximate models of objective function and constraints
- Different techniques
  - A fixed model
  - Updating the model with iteration

Metamodeling Techniques

- Three tasks
  - Generate data using an experimental design method or a computer
  - Choose a model to represent data
  - Fit the model to the data
- Some common techniques
  - Response surface methodology (RSM)
  - Neural networks
  - Kriging
  - Inductive learning

Response Surface Method (RSM)

- Box and Wilson (1951)
- Model: Error is independent of x
  - $y(\mathbf{x}) = g(\mathbf{x}) + \epsilon,\ \epsilon = N(0, \sigma),$
  - $\hat{y}(\mathbf{x}) = g(\mathbf{x})$
- Assume g(x), usually parametric linear or quadratic

$$\hat{y} = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \sum_{j=1}^{k} \beta_{ij} x_i x_j$$

- β_i determined by least-square regression from observed data
  - Optimize to minimize error: Find mean β_i
  - Variance of β_i determine predictive capability

RSM Tips

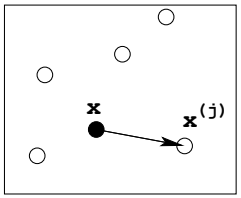
- Popular and most widely used
- Best suited in applications with random error
- However, limited handling on non-linearity
- Usually applied for k<10
- Sequencial RSM with move limits and trust region approach is better

Kriging Procedure

- D. G. Krige (a geologist): Statistical analysis of mining data
- Predict a value at a point from a given set of observations

$$f(\mathbf{x}) = \sum_{i=1}^{M} \lambda_i y(\mathbf{x}^{(j)})$$

- λ_i depends on distance of X from observed points
- Flexible, but complex
- Suited for *k*<50, deterministic problems

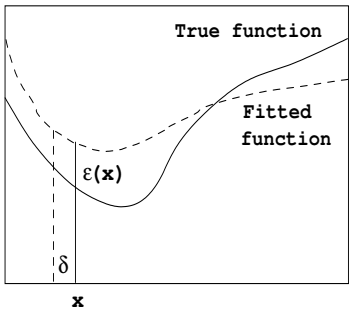


$$Y(\mathbf{x}) = \sum_{k=1}^{K} \beta_k \mathbf{f}_k(\mathbf{x}) + Z(\mathbf{x})$$

- Local variation Z(x) and β computed through spatial coorelation fun.
- Maximum likelihood fun. is optimized

Fundamentals of Kriging

- F(x) is true function
- f(x) is fitted function
- ε(x) = F(x)-f(x)
- If error at x is large, it is reasonable to believe that
  - Error at x+δ is also large

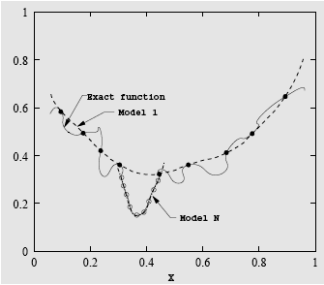


Model Verification

- Leave-one-out-cross-validation scheme
- Leave out one observation (say $x^{(J)}$ and $y^{(J)}$)
- Find the kriging model with (M-1) points and find $\hat{y}^{(J)}$
- Compute the error $s^J = \sqrt{(MSE(x^{(J)}))}$
- If the normalized error $(y^{(J)} - \hat{y}^{(J)})/s^J$ is within [-3,3], the model is acceptable

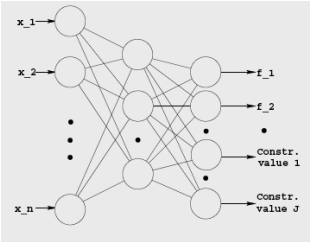
Successive Modeling Procedure

- Nain and Deb (2003)
- Successive approximations to the problem
- Initial coarse approximate model defined over the whole range of decision variables with small database
- Gradual finer approximate models localized in the search space

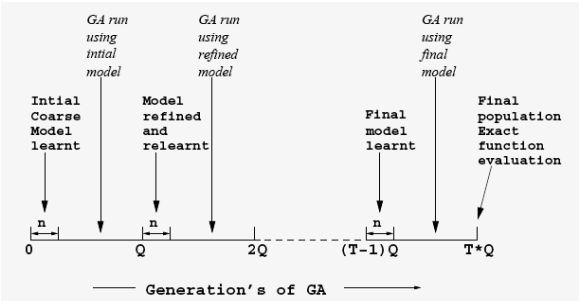


ANN Model

- A feed-forward neural network
- Input: Decision variables (size n)
- Output: Objective functions and constraint violations (size M+J)
- One hidden layer with H neurons
- Sigmoidal activation for hidden and output neurons



Generation-wise Sketch of Proposed Approach



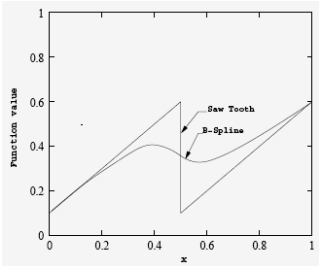
n/Q fraction of exact evaluations, although the ratio can be made smaller later

Flip Side: Parameters Involved

- Frequency of ANN modeling
- Number of training points (can reduce with iteration)
- Learning rate in training
- Final RMS error in ANN training
- Learning models (incremental or batch)
- ANN architecture (connectivity and hidden layers

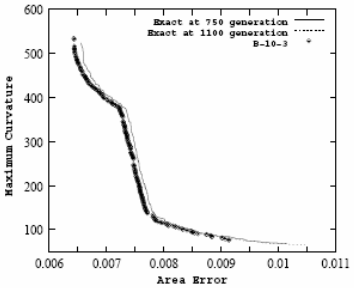
A Case Study from Curve-Fitting

- Two-objective problem:
  - Minimize error from the curve
  - Minimize the maximize curvature
- 41 control points forming a B-spline curve (39 varied)



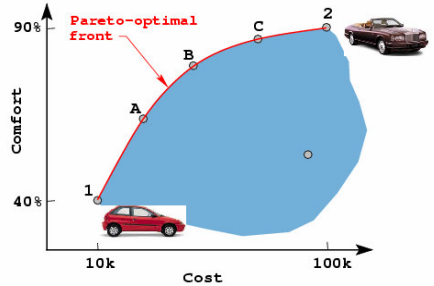
Savings in Function Evaluations

- B-10-3 finds a front in (750x200) evaluations similar to NSGA-II in (1100x200) evaluations
- A saving of 32% evaluations



Multi-Objective Optimization:
Handling multiple conflicting objectives

- We often face them

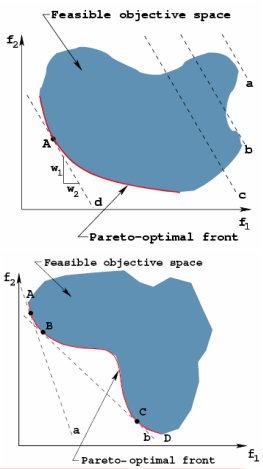


Classical Approach:
Weighted-Sum Method

- Construct a weighted sum of objectives and optimize

$$F(x) = \sum_{i=1}^{M} w_i f_i(x)$$

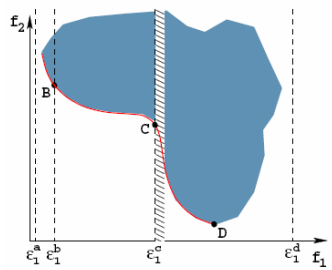
- User supplies weight vector *w*
- Non-convexity a problem



Classical Approach:
ε-Constraint Method

- Constrain all but one objective
- Need to know relevant **ε** vectors
- Non-uniformity in Pareto-optimal solutions
- Any Pareto-optimal solutions can be found with this approach

$$\text{Minimize} \quad f_\mu(\mathbf{x}),$$
$$\text{subject to} \quad f_m(\mathbf{x}) \le \epsilon_m, \; m \ne \mu;$$



Evolutionary Multi-Objective Optimization (EMO)

Step 1 :

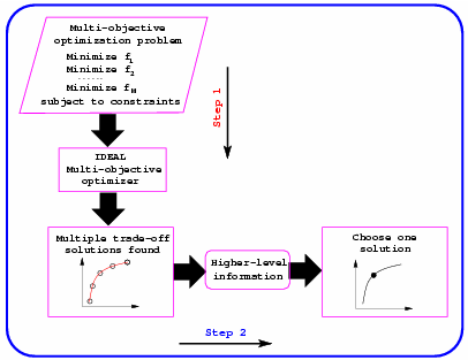
Find a set of Pareto-optimal solutions

Step 2 :

Choose one from the set

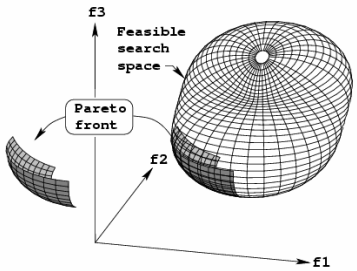
(Deb, 2001)

- Ideal for an EA



Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II)

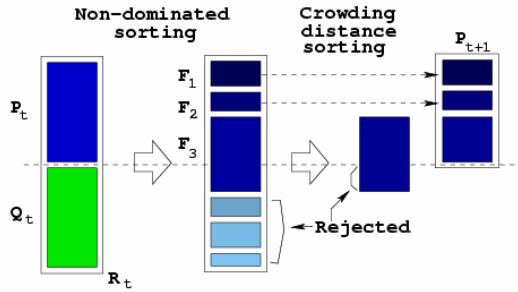
- NSGA-II can extract Pareto-optimal frontier
- Also find a well-distributed set of solutions
- iSIGHT and modeFrontier adopted NSGA-II



Fast-Breaking Paper in Engineering by ISI Web of Science (Feb'04)

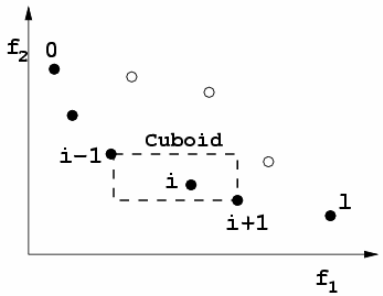
NSGA-II Procedure

Elites are preserved
Non-dominated solutions are emphasized



NSGA-II (cont.)

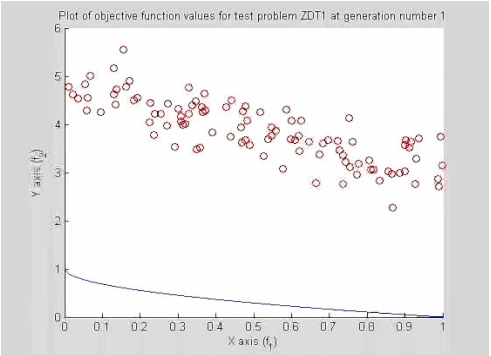
Diversity is maintained



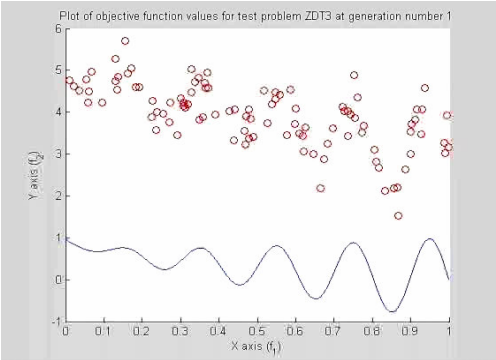
Overall Complexity
$O(N \log^{M-1} N)$

Improve diversity by
- k-mean clustering
- Euclidean distance measure
- Other techniques

Simulation on ZDT1

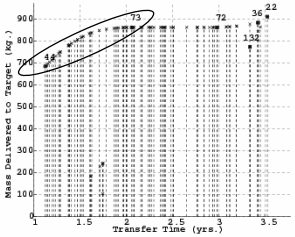


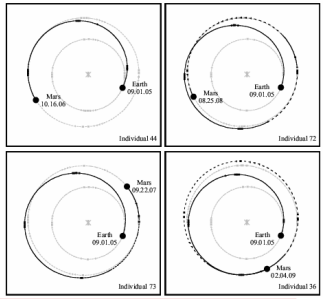
Simulation on ZDT3



EMO Applications

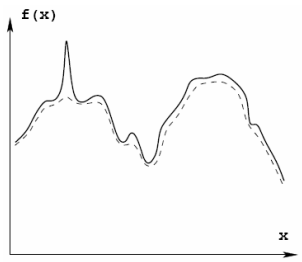
- Identify different trade-off solutions for choosing one (Better decision-making)
- Inter-planetary trajectory
  (Coverstone-Caroll et al., 2000)





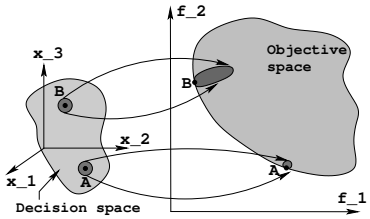
Robust Optimization
Handling uncertainties in variables

- Parameters are uncertain and sensitive to implementation
  - Tolerances in manufacturing
  - Material properties are uncertain
  - Loading is uncertain
- Who wants a sensitive optimum solution?
- Single-objective robust EAs (Branke and others)



Multi-Objective Robust Solutions

- Solutions are averaged in δ-neighborhood
- Not all Pareto-optimal points may be robust
- A is robust, but B is not
- Decision-makers will be interested in knowing robust part of the front



Multi-Objective Robust Solutions of Type I and II

- Similar to single-objective robust solution of type I

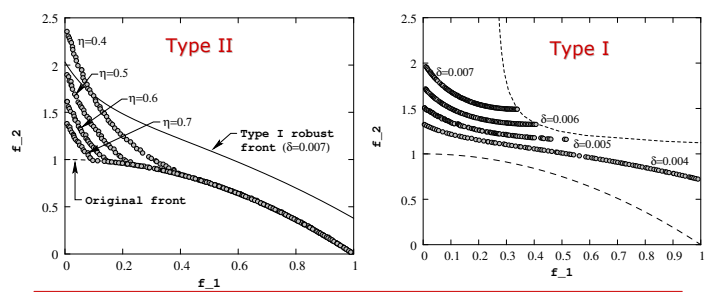
$$\begin{array}{ll} \textit{Minimize} & (f_1^{\text{eff}}(\mathbf{x}), f_2^{\text{eff}}(\mathbf{x}), \ldots, f_M^{\text{eff}}(\mathbf{x})), \\ \textit{subject to} & \mathbf{x} \in \mathcal{S}, \end{array}$$

- Type II

$$\begin{array}{ll} \textit{Minimize} & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})), \\ \textit{subject to} & \frac{\|\mathbf{f}^{\text{p}}(\mathbf{x}) - \mathbf{f}(\mathbf{x})\|}{\|\mathbf{f}(\mathbf{x})\|} \leq \eta, \\ & \mathbf{x} \in \mathcal{S}. \end{array}$$

Robust Frontier for Two Objectives

- Identify robust region
- Allows a control on desired robustness



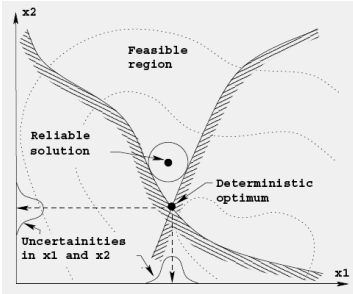
Faster Computation for Robustness

- Use of an archive to store already computed solutions
- Later, select solutions from the archive within δ-neighborhood
- If H points not found, create a solution
- Although reported quicker single-objective runs, multi-objective runs are not quicker
- A diverse set is needed, requiring archive to be large

Reliability-Based Optimization:
Making designs safe against failures

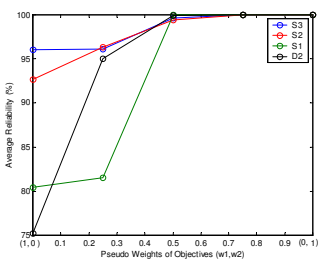
- Deterministic optimum is not usually reliable
- Reliable solution is an interior point
- Chance constraints with a given reliability

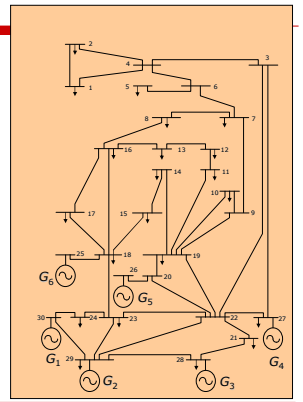
Minimize $\mu_f + k\sigma_f$
Subject to $Pr(g_j(x) \geq 0) \geq \beta_j$
β_j is user-supplied



30-Bus IEEE Power Distribution Problem

- Monte-Carlo simulation
- 6 generators with a total system load PD = 283.4 MW
- Reliability = 0.954

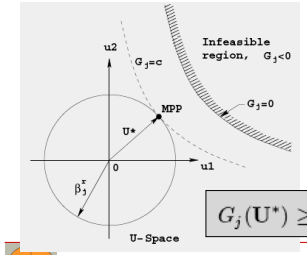




Statistical Procedure:
Check if a solution is reliable

- PMA approach

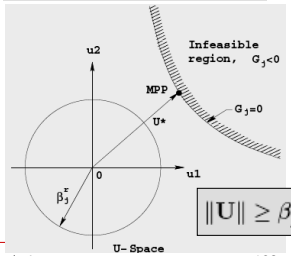
Minimize $G_j(\mathbf{U})$,
Subject to $\|U\| = \beta_j^T$,



$G_j(\mathbf{U}^*) \geq 0.$

- RIA approach

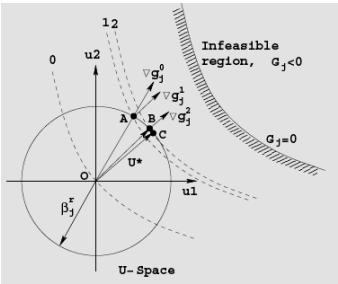
Minimize $\|\mathbf{U}\|$,
Subject to $G_j(\mathbf{U}) = 0.$



$\|\mathbf{U}\| \geq \beta_j^T.$

Proposed Faster Procedure

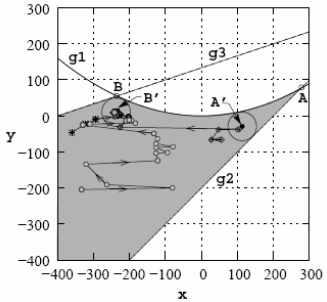
- Single-loop method
- Faster computation of U*
  - PMA: MPP as computed
  - RIA: Apply Newton's search to find MPP



Finding Global Reliable Solution: *Single-objective reliable solution*

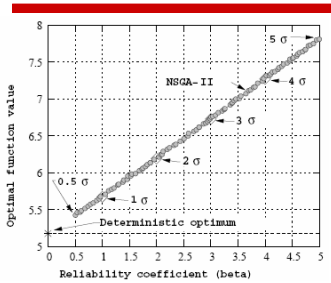
- Multiple deterministic optima
- Reliable solution for local is better
- Classical methods starts from deterministic global optimum
- GAs solve the overall problem and are better

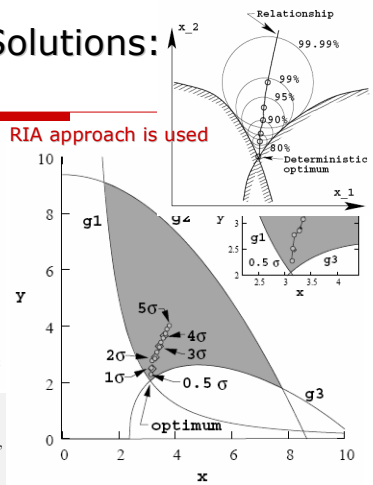
PMA approach is used



Multiple Reliability Solutions: *Get a better insight*

RIA approach is used





Maximize $x+y$,

Subject to $g_1(x,y) \equiv \frac{1}{20}x^2y - 1 \geq 0$,

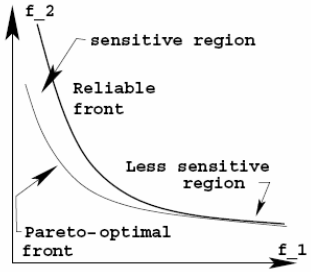
$g_2(x,y) \equiv \frac{1}{30}(x+y-5)^2 + \frac{1}{120}(x-y-12)^2 - 1 \geq 0$,

$g_3(x,y) \equiv \frac{80}{x^2+8y+5} - 1 \geq 0$,

$0 \leq x, y \leq 10$.

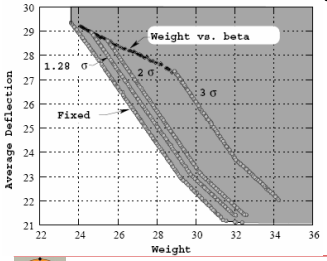
Multi-Objective Reliable Frontier

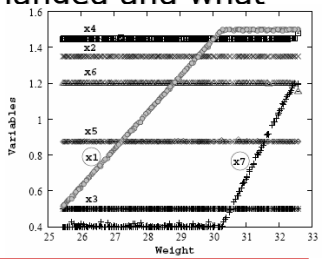
- Instead of finding deterministic Pareto-optimal front, find reliable front
- Chance constraints
- Objectives as they are
- PMA approach is used



Multi-Objective Reliability-Based Optimization

- Reliable fronts show rate of movement
- What remains unchanged and what

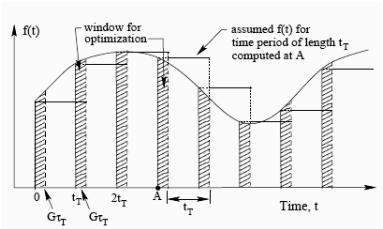


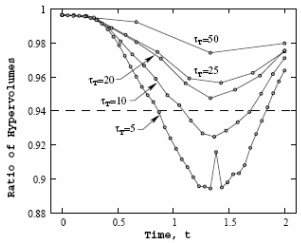


Operator Advantage

Dynamic Optimization

- Assume a statis in problem for a time step
- Find a critical frequency of change
- FDA2 test problem





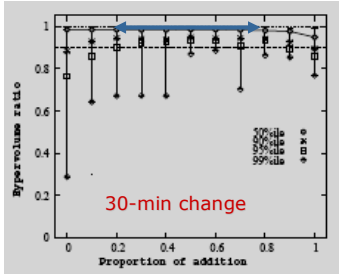
Hydro-Thermal Power Dispatch

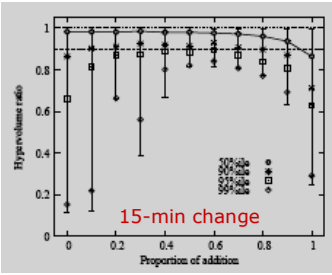
$$
\begin{aligned}
\text{Minimize} \quad & f_1(\mathbf{x}) = \sum_{m=1}^{M}\sum_{s=1}^{N_s} t_m[a_s + b_s P^s{}_{sm} + c_s(P^s{}_{sm})^2 + |d_s \sin(e_s(P^s{}_{s,\min} - P^s{}_{sm}))|], \\
\text{Minimize} \quad & f_2(\mathbf{x}) = \sum_{m=1}^{M}\sum_{s=1}^{N_s} t_m[\alpha_s + \beta_s P^s{}_{sm} + \gamma_s(P^s{}_{sm})^2 + \eta_s \exp(\delta_s P^s{}_{sm})], \\
\text{subject to} \quad & \sum_{s=1}^{N_s} P^s{}_{sm} + \sum_{h=1}^{N_h} P^h{}_{hm} - P_{Dm} - P_{Lm} = 0, \quad m = 1,2\ldots,M, \\
& \sum_{m=1}^{M} t_m(a_{0h} + a_{1h}P^h{}_{hm} + a_{2h}(P^h{}_{hm})^2) - W_h = 0, \quad h = 1,2,\ldots,N_h. \\
& P^h{}_{h,\min} \le P_{hm} \le P^h{}_{h,\max}, \quad h = 1,2,\ldots,N_h, m = 1,2,\ldots,M, \\
& P^s{}_{s,\min} \le P_{sm} \le P^s{}_{s,\max}, \quad s = 1,2,\ldots,N_s, m = 1,2,\ldots,M.
\end{aligned}
$$

- Minimize Cost and NOx emission
- Power balance and water head limits
- Dynamic due to change in power demand with time

Dynamic Hydro-Thermal Power Scheduling

- Addition of random or mutated points at changes
- 30-min change found satisfactory

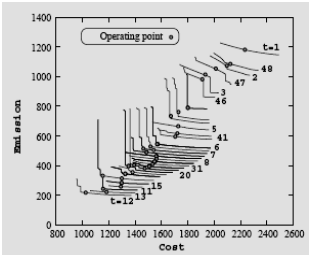


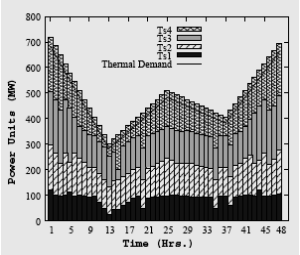


Dynamic EMO with Decision-Making

- Needs a fast decision-making
- Use an automatic procedure
  - Utility function, pseudo-weight etc.

Case	Cost	Emission
50-50%	74239.07	25314.44
100-0%	69354.73	27689.08
0-100%	87196.50	23916.09





Handling Many Objectives

- Often redundant objectives
- Run NSGA-II for a while
- Perform a PCA to identify important objectives
- Continue NSGA-II with reduced objectives
- Non-linear PCA and kernal-based approaches tried
- 50 objectives reduced to three objectives tried successfully

PCA-NSGA-II: Demonstration on DTLZ5(2,10)

Iter.1

Iter. 1 : PCA-1 (58.83 % variance)	f_7 f_{10}
PCA-2 (28.26 % variance)	f_1
PCA-3 (06.53 % variance)	f_8
PCA-4 (03.27 % variance)	f_8

	f_1	f_7	f_8	f_{10}
f_1	+	-	+	-
f_7	-	+	+	-
f_8	+	+	+	-
f_{10}	-	-	-	+

No reduction possible

4 Prin. Comp: giving 4 imp obj. of 10

f7, f8 correlated

f7 better than f8

Iter.2

Iter. 2 : PCA-1 (94.58 % variance)	f_7 f_{10}
PCA-2 (4.28 % variance)	f_8

	f_7	f_8	f_{10}
f_7	+	+	-
f_8	+	+	-
f_{10}	-	-	+

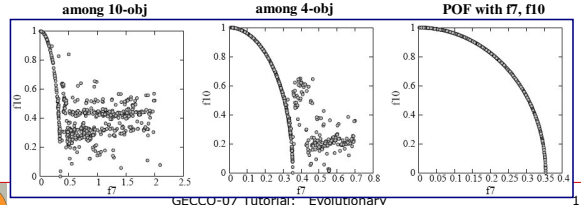
	e1:0.9458	e2:0.0428	
f_7	+0.543	-0.275	c7=0.5253
f_8	+0.457	+0.672	c8=0.4610
	PCA1	PCA2	

2 Prin. Comp: giving 3 imp obj of 4

Iter.1: f7, f10 among 10-obj

Iter.2: f7, f10 among 4-obj

Iter.3: Desired POF with f7, f10

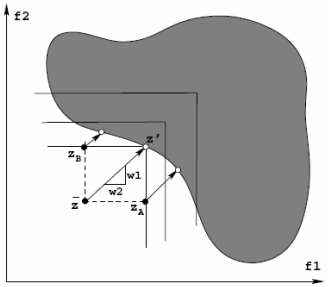


EMO for Decision-Making

- Use where multiple, repetitive applications are sought
- Use where, instead of a point, a trade-off region is sought
- Use for finding points with specific properties (nadir point, knee point, etc.)
- Use for robust, reliable or other fronts
- Use EMO for an idea of the front, then decision-making (I-MODE)

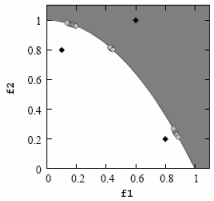
Reference Point Based EMO

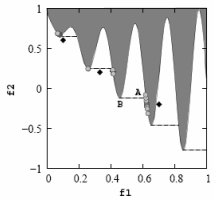
- Wierzbicki, 1980
- A P-O solution closer to a reference point
  - Multiple runs
  - Too structured
- Extend for EMO
  - Multiple reference points in one run
  - A distribution of solutions around each reference point

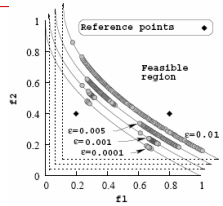


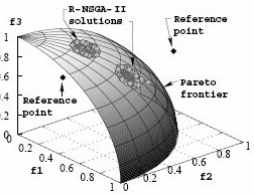
Making Decisions:
Reference Point Based EMO

- Ranking based on closeness to each reference point
- Clearing within each niche with ε



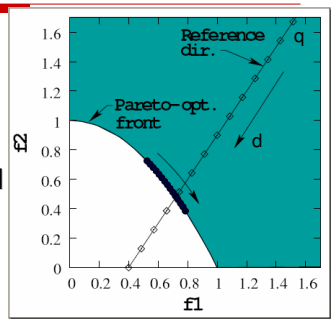






EMO for Making Decisions:
Reference Direction based EMO
(Korhonen and students, 1996-)

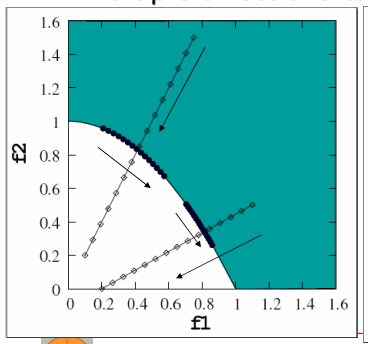
- Choose a direction d
- Solve for different t:
  min max(z_i-q_i)/w_i
  s.t. z=q+t*d
- Choose most preferred solution
- If different from previous, continue

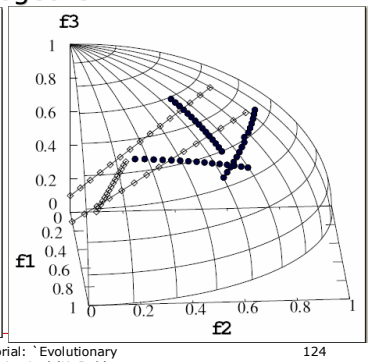


- Instead of solving several opt. problems, use EMO once

Reference Direction based EMO (cont.)

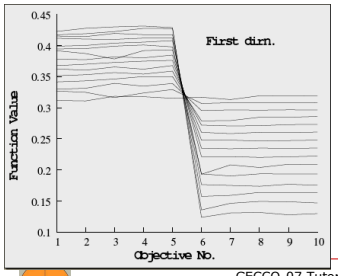
- Multiple directions together

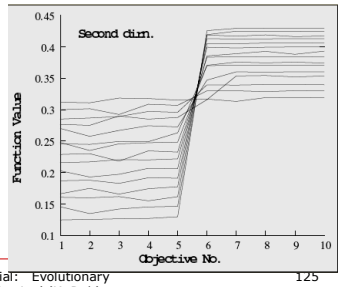




Reference Direction Based EMO (cont.)

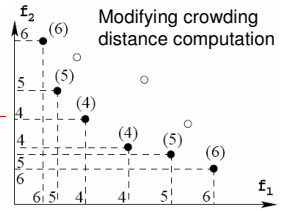
- 10-objective problem with two directions
- (0.8,0.8,0.8,0.8,0.8,0.2,0.2,0.2,0.2,0.2)
- (0.2,0.2,0.2,0.2,0.2,0.8,0.8,0.8,0.8,0.8)

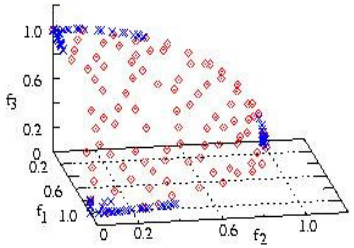


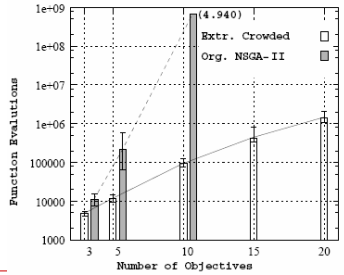


EMO for Finding Nadir Point

- DTLZ problems extended up to 20 objectives

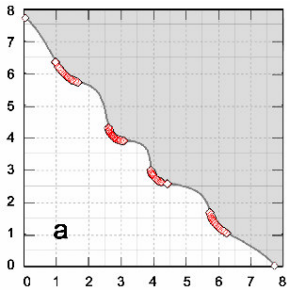


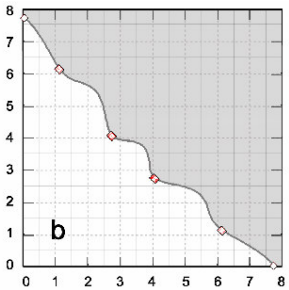




Finding Knee Solutions
(Branke et al., 2004)

- Find only the knee or near-knee solutions

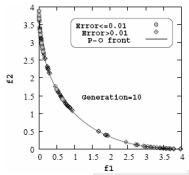


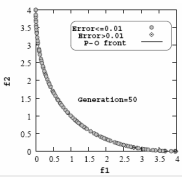


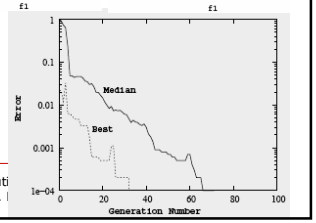
EAs with Theoretical Confidence

$$\frac{\lambda_2}{\lambda_1}(-\nabla f_2) + \sum_j \frac{u_j}{\lambda_1}\nabla g_j = \nabla f_1$$

- EA solution(s) improved with local search (classical or hill-climbing)
- Multi-objective problems: Verify with existing methods (e-constraint etc.)
- If derivative exists, verify the solution to be a KKT point





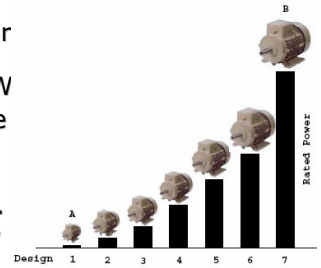


Norm can be used as termination crit.

Innovization:

Discovery of Innovative design principles through optimization

- Understand important design principles in a routine design scenario
  - Example: Electric motor design with varying ratings, say 1 to 10 kW
    - Each will vary in size and power
    - Armature size, number of turns etc.
  - How do solutions vary?
    - Any common principles!



Innovization Procedure

- Choose two or more conflicting objectives (e.g., size and power)
  - Usually, a small sized solution is less powered
- Obtain Pareto-optimal solutions using an EMO
- Investigate for any common properties manually or automatically
- Why would there be common properties?
  - Recall, Pareto-optimal solutions are all optimal!

In Search of Common Optimality Properties

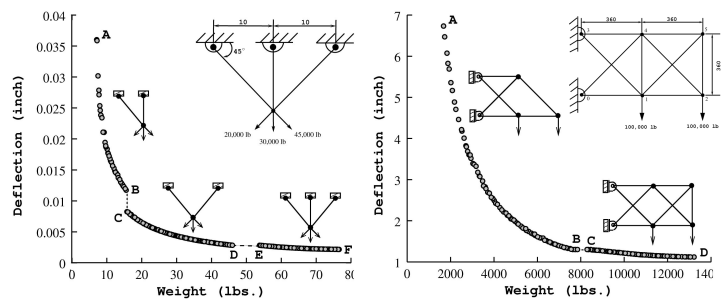
Fritz-John Necessary Condition:

Solution x^* satisfy

1. $\sum_{m=1}^{M} \lambda_m \nabla f_m(x^*) - \sum_{j=1}^{J} u_j \nabla g_j(x^*) = 0,$ and
2. $u_j g_j(x^*) = 0$ for all $j = 1, 2, 3, \ldots\ldots, J$
3. $u_j \geq 0$, $\lambda_j \geq 0$, for all j and $\lambda_j > 0$ for at least one j

- To use above conditions requires differentiable objectives and constraints
- Yet, it lurks existence of some properties among Pareto-optimal solutions

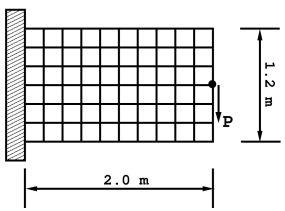
Revealing Salient Insights: Truss Structure Design



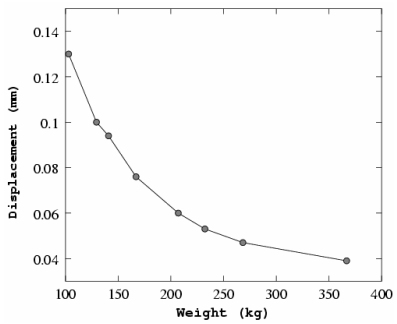
(Deb, Khan and Jindal, 2000)

Revealing Salient Insights: A Cantilever Plate Design

Base Plate



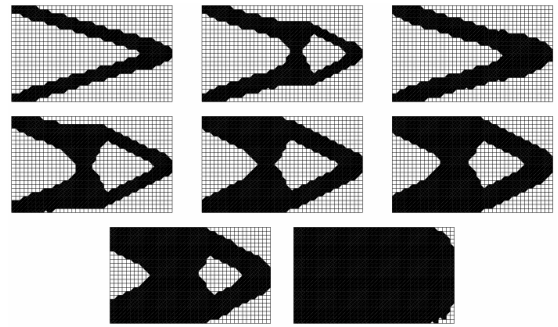
Eight trade-off solutions are chosen



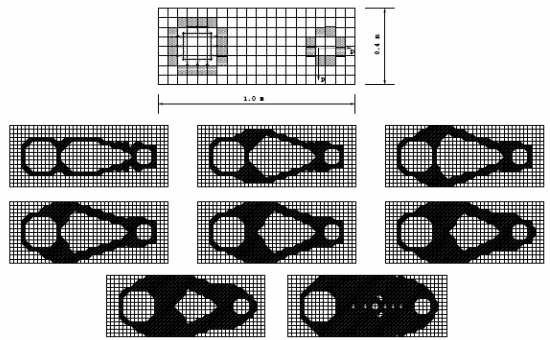
(Deb and Chaudhuri, 2003)

Trade-Off Solutions

- Symmetry in solutions about mid-plane, discovery of stiffener



A Connecting Rod

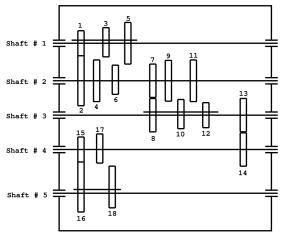


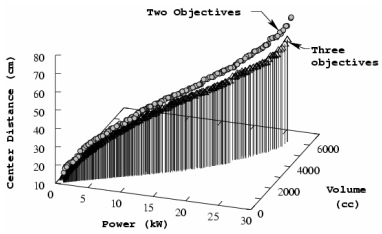
Innovized Principles

- Mid-line symmetry
- Straight arms to reach load is minimum-weight strategy
- Two ways to increase stiffness
  - Thickening of arms
  - Use of a stiffener
  - Additional stiffening by a combination
- Chamfering of corners helpful

Gear-box Design

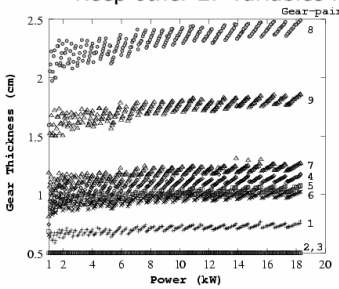
- A multi-spindle gear-box design (Deb and Jain, 2003)
- 28 variables (integer, discrete, real-valued)
- 101 non-linear constraints
- Important insights obtained (larger module for more power)

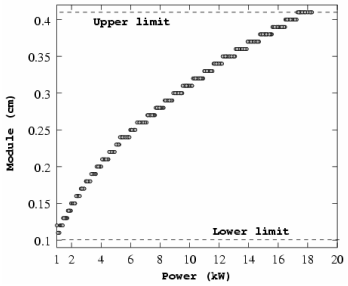




Innovized Principles

- Module varies proportional to square-root of power
- Keep other 27 variables more or less the same



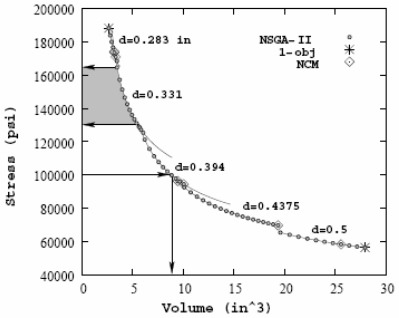


Mechanical Spring Design

- Minimize material volume
- Minimize developed stress
- Three variables: (d, D, N): discrete, real, integer
- Eight non-linear constraints
  - Solid length restriction
  - Maximum allowable deflection ($P/k \leq 6$in)
  - Dynamic deflection $(P_m - P)/k \geq 1.25$in
  - Volume and stress limitations

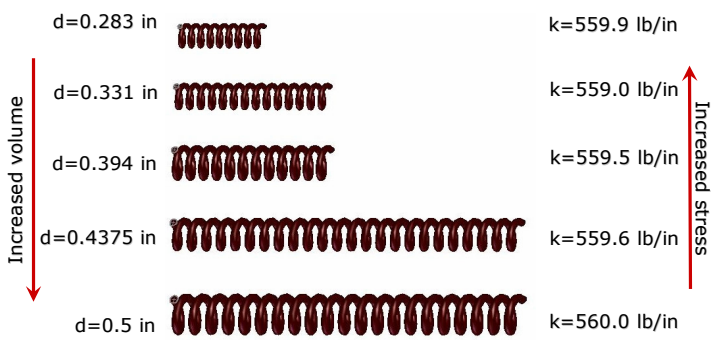
Innovized Principles

- Pareto-optimal front have niches with d
- Only 5 (out of 42) values of d (large ones) are optimal
- A blue-print for optimal design



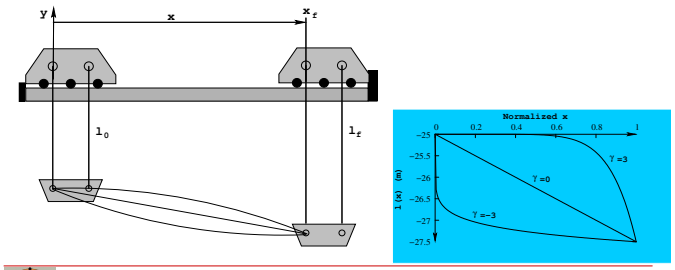
(Deb and Srinivasan, 2005)

Optimal Springs, Optimal Recipe



Overhead Crane Maneuvering

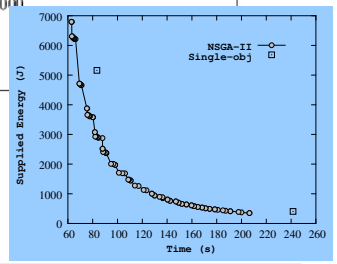
- Minimize time of operation
- Minimize operating energy



Simulation Results

γ	Force (N)	Pattern
2.98	149.26	1111111100100010000000000000000000000000000000000000000
3.00	187.76	111111110010001000000000000000000000000000000
3.00	238.56	11111111001000100000000000000000000
3.00	327.86	1111101100100110000000000
2.98	427.93	11111011001001100000000
3.00	544.94	1111111100100010 0
2.87	629.62	11111111001000 1
2.88	818.99	110111010010001

- NSGA-II finds trade-off and interesting properties



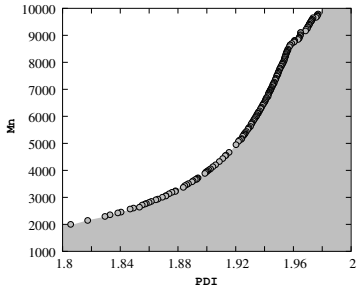
Innovative Principles

- For optimum operation:
  - Lower load suddenly at the end
  - Spend energy only at the beginning
- Fast unloading demands more energy
  - For fixed θ, E ∞ l
  - Delay lowering for saving energy

$$E = mg(1 - \cos\theta)l$$

Epoxy Polymerization

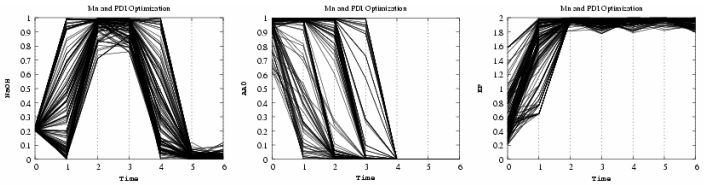
- Three ingredients added hourly
- 54 ODEs solved for a 7-hour simulation
- Maximize chain length (Mn)
- Minimize polydispersity index (PDI)
- Total 3x7 or 21 variables
- (Deb et al., 2004)



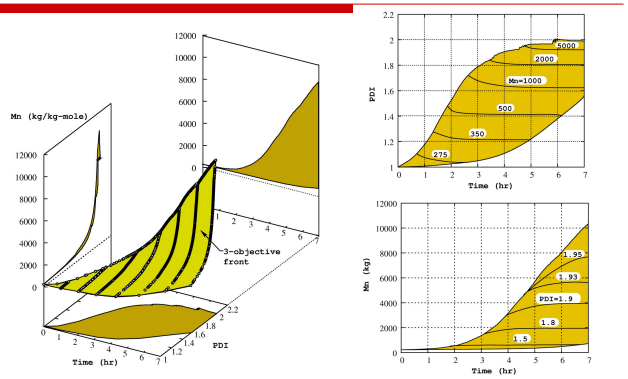
A non-convex frontier

Epoxy Polymerization (cont.)

- Some patterns emerge among obtained solutions
- Chemical significance unveiled

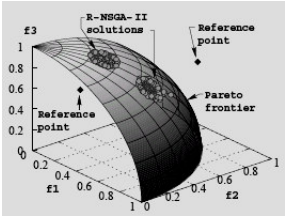


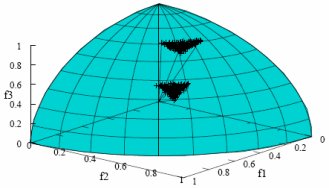
Innovized Principles: *An Optimal Operating Chart*



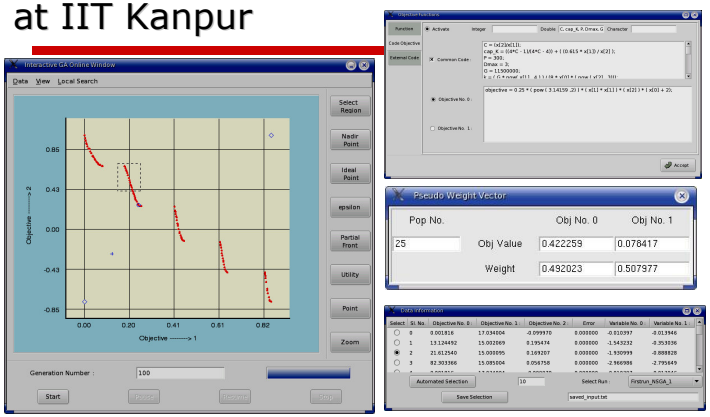
Interactive EMO with MCDM

- Optimize and choose a single solution
- EMO coupled with MCDM techniques
- EMO for repetitive optimization
- Reference point, reference direction, light beam search
- Put different ideas together in a software

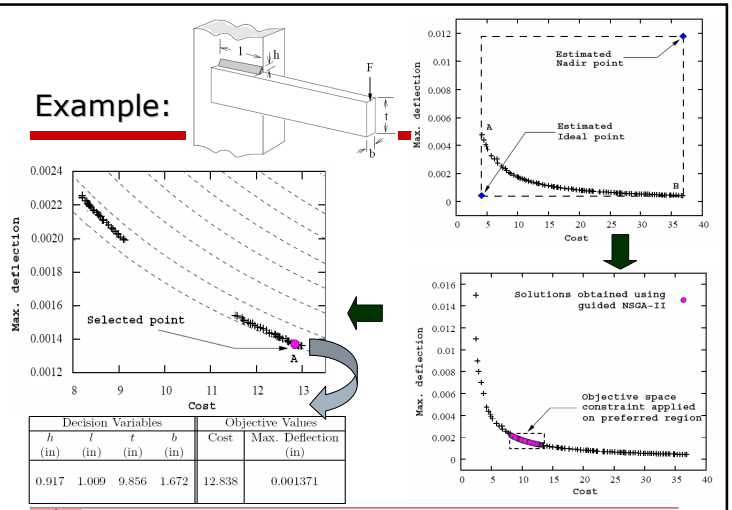




I-MODE Software Developed at IIT Kanpur

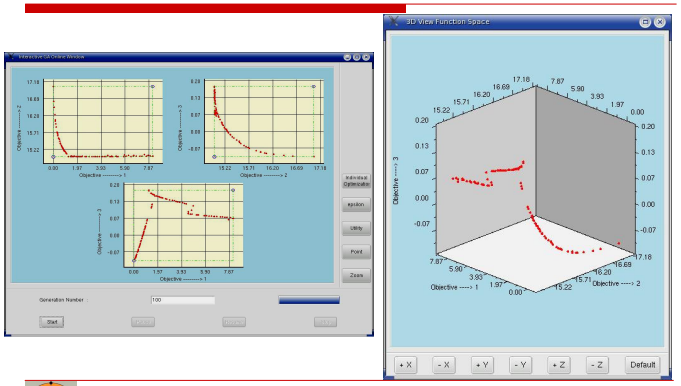


Example:



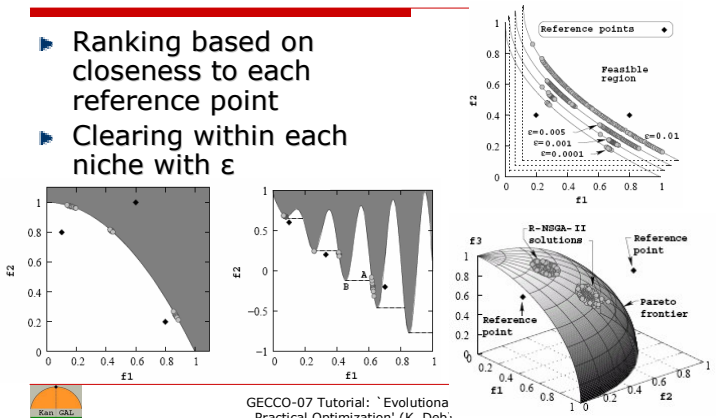
Decision Variables				Objective Values	
h (in)	l (in)	t (in)	b (in)	Cost	Max. Deflection (in)
0.917	1.009	9.856	1.672	12.838	0.001371

I-MODE for Three Objectives



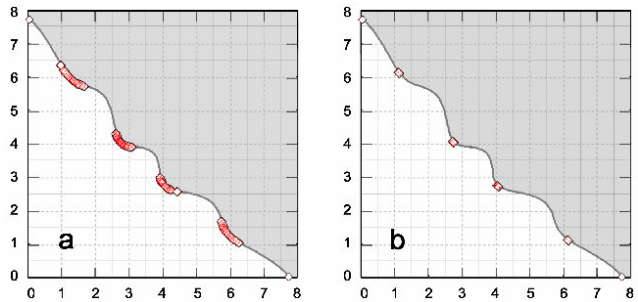
Making Decisions:
Reference Point Based EMO

- Ranking based on closeness to each reference point
- Clearing within each niche with ε



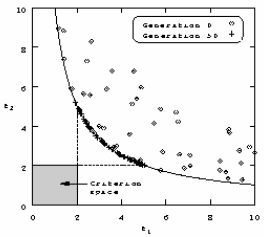
Finding Knee Solutions

Branke et al. (2004) for more details



Goal Programming:
Not to find Optimum

- Target function values are specified
- Convert them to objectives and perform domination check with them



Type	Goal	Objective function
$\leq$	$f_j(x) \leq t_j$	Minimize $\langle f_j(x) - t_j \rangle$
$\geq$	$f_j(x) \geq t_j$	Minimize $\langle t_j - f_j(x) \rangle$
$=$	$f_j(x) = t_j$	Minimize $\lvert f_j(x) - t_j \rvert$
Range	$f_j(x) \in [t_j^l, t_j^u]$	Minimize $\max(\langle t_j^l - f_j(x) \rangle, \langle f_j(x) - t_j^u \rangle)$

Conclusions

- Most application activities require optimization routinely
- Classical methods provide foundation
  - If applicable, good accuracy is achievable
- Evolutionary methods enable applicability to near-optimality
  - Try when classical methods fail
  - Parallel search ability
- A good optimization task through EAs and local search

How are EAs Unique?

- Broader applicability
  - Mixed variables, non-linearity, non…
  - Constraint handling
- Easy to use for distributed computing
- Beyond optimization
  - Learn your problem better
  - What makes a solution optimum, robust, or reliable?

Practical Optimization
Final words

- Seems impossible to have one algorithm for many practical problems
- Needs a customized optimization
  - Calls for collaborations
- An application requires
  - Domain-specific knowledge
  - Thorough knowledge in optimization basics
  - Good knowledge in optimization algorithms
  - Good computing background
- Have successful show-cases, make a data-base, choose one in an application
  - Calls for collaborations

What We Have Not discussed?

- Combinatorial optimization
  - Non-linearity, large dimension
- Problem Formulation aspects
  - Objectives, constraints, etc.
- Very large computational overhead
  - One evaluation takes a day or more
- Parallel EAs
- Termination criteria

Thank You for Your Attention

- Acknowledgement:
  - KanGAL students, staff and collaborators
  - GM, GE, Honda R&D, STMicroelectronics
  - Governmental Research Labs

For further information:

http://www.iitk.ac.in/kangal
Email: deb@iitk.ac.in

Wishing you to have a productive GECCO-2007