Global search algorithms

• Local algorithms zoom in on optima based on known information
• Global algorithms must also have a component of exploring new regions in design space
• The key to global optimization is therefore the balance between exploration and exploitation
• Many accomplish that based on population
Plan of algorithms

• Population based exploitation – Nelder and Meads, sequential simplex method
• Deterministic balance between exploration and exploitation – the DIRECT algorithm
• Deterministic balance based on statistical model: EGO
• Stochastic population based global optimization
  – Particle Swarm Optimization
  – Genetic algorithms
  – Ant colony algorithms
Sequential Simplex Method  
(section 4.2.1)

- In $n$ dimensional space start with $n+1$ particles at vertices of a regular (e.g., equilateral) simplex.

$$x_j = x_0 + pe_j + \sum_{k=1, k \neq j}^{n} qe_k, \quad j = 1, \ldots, n,$$  

(4.2.1)

with

$$p = \frac{a}{n\sqrt{2}}(\sqrt{n+1} + n - 1), \quad \text{and} \quad q = \frac{a}{n\sqrt{2}}(\sqrt{n+1} - 1),$$

(4.2.2)

- Reflect worst point about c.g.

$$\bar{x} = \frac{1}{n} \sum_{i=0}^{n} x_i, \quad i \neq h.$$

- Read about expansion and contraction
Rozenbrock Banana function

\[ F(X) = 10X_1^4 - 20X_1^2X_2 + 10X_2^2 + X_1^2 - 2X_1 + 5 \]

Vanderplaats’s version

![Image of contour plot for the Rozenbrock function](image-url)
Matlab commands

function [y]=banana(x)
global z1
global z2
global yg
global count
y=100*(x(2)-x(1)^2)^2+(1-x(1))^2;
z1(count)=x(1);
z2(count)=x(2);
yg(count)=y;
count=count+1;

>> mat=[z1;z2;yg]

mat =

Columns 1 through 8
-1.200 -1.260 -1.200 -1.140 -1.080 -1.080 -1.020 -0.960
1.000  1.000  1.050  1.050  1.075  1.125  1.1875  1.150
24.20  39.64  20.05  10.81  5.16  4.498  6.244  9.058

Columns 9 through 16
-1.020 -1.020 -1.020 -1.125 -1.046 -1.031 -1.007 -1.013
1.125  1.175  1.100  1.100  1.119  1.094  1.078  1.113
4.796  5.892  4.381  7.259  4.245  4.218  4.441  4.813
\[ f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \]
Next iteration
DIRECT Algorithm


Lipschitzian Optimization

- Optimizer divides space into boxes and samples the vertices of each.
- One box is further divided based on a predicted maximum rate of change of the function, $K$. 
DIRECT

• The function value at the middle of each box and it’s largest dimension are used to determine potentially optimal boxes
• Each potentially optimal box is divided
• No need to predict the Lipchitz constant
DIRECT Box Division

S.E. COX ET AL.
Exploration vs. Exploitation

- DIRECT uses convex hull of box sizes to balance exploitation vs. exploration
- With enough function evaluations every region in design space will be explored
- This is clearly not feasible for high dimensional spaces
- Cox’s paper compares DIRECT to repeated local optimization with random start
Results

Minimum Weight

<table>
<thead>
<tr>
<th>Case</th>
<th>DOT</th>
<th>LFOPCV3</th>
<th>DIRECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 DV Case</td>
<td>590000</td>
<td>595000</td>
<td>580000</td>
</tr>
<tr>
<td>10 DV Case</td>
<td>570000</td>
<td>575000</td>
<td>565000</td>
</tr>
<tr>
<td>26 DV Case</td>
<td>560000</td>
<td>565000</td>
<td>550000</td>
</tr>
</tbody>
</table>
Results

Function Calls

<table>
<thead>
<tr>
<th>Case</th>
<th>DOT</th>
<th>LFOPCV3</th>
<th>DIRECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 DV Case</td>
<td>100000</td>
<td>150000</td>
<td>200000</td>
</tr>
<tr>
<td>10 DV Case</td>
<td>50000</td>
<td>50000</td>
<td>50000</td>
</tr>
<tr>
<td>26 DV Case</td>
<td>26 DV Case</td>
<td>26 DV Case</td>
<td>26 DV Case</td>
</tr>
</tbody>
</table>

Legend:
- DOT
- LFOPCV3
- DIRECT
Results

DOT Optimum Values

![Graph showing weight variation over optimization numbers](image-url)
Results

![Diagram showing DOT and DIRECT results for 26 DV case]