A Brief Introduction to Genetic Optimization

S.D. Sudhoff

Fall 2005
Traditional Optimization Methods

- Newton’s Method
Traditional Optimization Methods

• Steepest Decent
• Conjugate Direction Algorithm
• Conjugate Gradient Algorithm
• Davidon, Fletcher, Powell, Algorithm (DFP)
• Broyden, Fletcher, Goldfarb, and Shanno Algorithm (BFGS)
• Nelder-Mead Simplex Method
• Take ECE580
Traditional Optimization Methods

• Properties of all of these methods
  ➢ Need an initial guess
  ➢ Very susceptible to finding local minimum

• Properties of most of these methods (not Nelder Meed Simplex)
  ➢ Take derivatives
An Alternate Approach
Genetic Algorithms in a Nutshell

• Probabilistic Optimization Technique
• Loosely Based in Principals of Genetics
• First Developed By Holland, Late 60’s – Early 70’s
• Does Not Require Gradients or Hessians
• Does Not Require Initial Guess
• Operates on a Population
A Gene As A Parameter: Biological Systems

- Each Gene Has Value From Alphabet \{AT,GC,CG,AT\} from Nitrogenous Adenine, Guanine, Thymine, Cytosine
- Each Gene is Located on One of a Number of Chromosomes
- Chromosomes May Be Haploid, Diploid, Polyploid
A Gene As A Parameter: Canonical Genetic Algorithm

- Each Gene Has a Value From Alphabet (Normally Binary \{0,1\})
- Each Gene is Located on a Chromosome (Normally 1)
- Chromosomes Generally Haploid
Evolution in a Canonical GA

START

Initialization

Fitness evaluation

Selection

Crossover

Mutation

STOP?

END
Example Operator: Single Point Crossover
Example Operator: Mutation

Original chromosome

| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |

Mutated chromosome

| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |

Mutation point
A Gene As A Parameter: Non-Encoded Genetic Algorithms

• Each Gene Has A Range (Minimum Value, Maximum Value)
• Each Gene Has A Type (Mapping)
  ➢ Integer
  ➢ Linearly Mapped Real Number
  ➢ Logarithmically Mapped Real Number
• Each Gene is Located on One of a Number of Chromosomes
• Chromosomes Are Haploid
Genetic Optimization System
Engineering Tool (GOSET)

- Matlab based toolbox
- Usable with minimum knowledge
- Allows, if desired, high degree of algorithm control
An Individual in GOSET

- A Collection of Genes on Chromosomes
- Fitness (Measures of Goodness)
  \[ \mathbf{f} = [f_1 \ f_2 \ \cdots \ f_N]^T \]
- Region

<table>
<thead>
<tr>
<th>Gene 1</th>
<th>Gene 2</th>
<th>Gene 3</th>
<th>Gene 4</th>
<th>Gene 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene 6</td>
<td>Gene 7</td>
<td>Gene 8</td>
<td>Gene 9</td>
<td>Gene 10</td>
</tr>
<tr>
<td>Gene 11</td>
<td>Gene 12</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Chromosome 1
Chromosome 2
Chromosome 3
A Population In GOSET

Region 1

Region 2

Region 3
Evolution in GOSET
Example: Powell’s Problem

• Minimize

\[ f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4 \]

• Fitness

\[ \text{fitness}(x) = 5 - \log_{10}(f(x) + 10^{-5}) \]
% Optimization of Powell's Problem

% Initialize the parameters
GAP=gapdefault;

% gene x1 x2 x3 x4
GAP.gd_min =[-2.1 -2.1 -2.1 -2.1];
GAP.gd_max =[ 2.0 2.0 2.0 2.0];
GAP.gd_type=[ 2 2 2 2 ];
GAP.gd_cid =[ 1 1 1 1 ];

[P,GAS]= gaoptimize(@powell_fit,GAP,[],[],[],[]);

parameters=GAS.bestgenes(:,GAS.cg);
function f = powell(x)

% Powell's function taken from Chong & Zak, p. 140.
% The global miniizer is powell(0,0,0,0) = 0

x1=x(1);
x2=x(2);
x3=x(3);
x4=x(4);
f1 = (x1 + 10*x2)^2 + 5*(x3 - x4)^2 + (x2 - 2*x3)^4 + 10*(x1-x4)^4;
f=5-log10(f1+0.00001);
### Powell’s Problem: Conclusion

```plaintext
<table>
<thead>
<tr>
<th>Parameter Number</th>
<th>Normalized Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
</tr>
<tr>
<td>4</td>
<td>0.5</td>
</tr>
</tbody>
</table>

**Parameters**

- $f(B(b))$  
- $M(g)$  
- $M(n(r))$

![Graph](image)

**Parameters =**

-0.0098
0.0014
-0.0097
-0.0105
IGBT Conduction Loss

• Fit IGBT Conduction Loss Data To:

\[ v = ai + (bi)^c \]
% load experimental data
% voltage is first column
% current is second column
load igbt_data.mat
Data.v=v_data;
Data.i=i_data;

% gafit
GAP=gapdefault;
GAP.rp_lvl=1;
GAP.mg_nreg=4;
GAP.mg_tmig=20;
GAP.mg_pmsg=0.05;
GAP.dp_fp=0.5;
GAP.dth_alg=2;
% declare the types of all the parameters
GAP.gd_min = [1e-8 1.0e-6 1.0e-3];
GAP.gd_max = [1e+2 1.0e+3 1.0e+0];
GAP.gd_type= [ 3 3 3 ];
GAP.gd_cid = [ 1 1 1 ];

% do the optimization
[fP,GAS]= gaoptimize(@igbt_fit,GAP,Data,[],[],[],[]);

% plot the best fit
bestparameters=GAS.bestgenes(:,GAS.cg,1)
igbt_fit(bestparameters,Data,3);
% this routine returns the fitness of the IGBT conduction
% loss parameters based on the mean percentage error
function fitness = IGBTfitness(parameters, data, fignum)

% assign genes to parameters
a = parameters(1);
b = parameters(2);
c = parameters(3);

vpred = a * data.i + (b * data.i).^c;
error = abs(1 - vpred./data.v);
fitness = 1.0/(1.0e-6 + mean(error));
if nargin>2

fitness

figure(fignum);
Npoints=200;
ip=linspace(0,max(data.i),Npoints);
vp=a*ip+(b*ip).^c;
plot(data.i,data.v,'bx',ip,vp,'r')
title('Voltage Versus Current');
xlabel('Current, A');
ylabel('Voltage, V');
legend({'Measured','Fit'});

figure(fignum+1);
Npoints=200;
plot(data.i,data.v.*data.i,'bx',ip,vp.*ip,'r')
title('Power Versus Current');
xlabel('Current, A');
ylabel('Power, V');
legend({'Measured','Fit'});

end
IGBT Results

bestparameters =
0.0000
4.5288
0.2588
IGBT Results (Cont.)
IGBT Results (Cont.)
Your Problem

• Consider a machine for which

$$\lambda_m = 0.3 \quad VS$$

$$L_q = 15 \quad mH$$

$$L_d = 10 \quad mH$$

$$P = 4$$
Your Problem (Cont)

• Develop an approximate expression for the optimal q-axis current command (in MTPA sense) as a function of desired torque

• Deliverables: short write up with your expression, all *.m files etc used.

• NOTE: GOSET2.2 is on the web. See software distribution.
Comments on Approach