

# 4M17 Exercise III : Optimizing the bird Function

eng-216er

January 12, 2015

`eng-216er.github.io`

**Summary:** Summary goes here.

## Contents

<b>1</b>	<b>Running the Code</b>	<b>1</b>
<b>2</b>	<b>Rationale behind the use of JavaScript</b>	<b>1</b>
<b>3</b>	<b>Genetic Algorithms</b>	<b>1</b>
<b>4</b>	<b>Tabu Search</b>	<b>1</b>
<b>A</b>	<b>Listings</b>	<b>2</b>
A.1	PlotImage.m . . . . .	2
A.2	Bird.png . . . . .	2
A.3	index.html . . . . .	2
A.4	minmax.js . . . . .	6
A.5	ex3.js . . . . .	6
A.6	tabu.js . . . . .	8
A.7	genetic_algorithm.js . . . . .	13

## List of Figures

## List of Tables

## 1 Running the Code

The code in this report runs as web app. It can be found in the listings, but is also hosted at [eng-216er.github.io](http://eng-216er.github.io) and can be accessed by launching this URL in a web browser. The code has been tested in the latest versions of the Firefox and Chrome browsers.

## 2 Rationale behind the use of JavaScript

JavaScript is unique in that programs written in it can be embedded in a html document, and executed in a web browser. No other language can be used for client side web programming without either using a browser extension (Java, Flash) or compiling into JavaScript (CoffeeScript).

This provided the motivation for me to implement the optimisation algorithms in JavaScript. In small part, this was because of the possibility of creating a simple html based UI for controlling the optimization parameters and inspecting the results.

Largely however, I was drawn to using JavaScript because being able to solve optimization problems in a browser could potentially be useful within several web programming contexts. For instance, the development of WebGL allows for hardware accelerated graphics problems to be developed for the web. Optimization can be used to solve useful problems in graphics programming. An example is computing the best possible conformal mapping between texture co-ordinates, and coordinates that make up a mesh of a surface. This can be used to apply a texture to a 3D surface, while minimising the effect of distortion on the surface.

There are currently very few JavaScript optimization libraries. Although the software provided in this report does very little to rectify that, it does provide a starting point for more complex software.

## 3 Genetic Algorithms

A gen

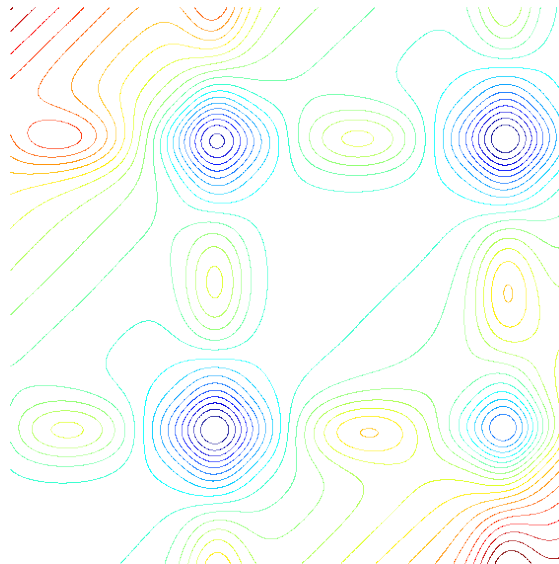
## 4 Tabu Search

## A Listings

### A.1 PlotImage.m

```
1 function [ ] = PlotImage( )
2 % Plot a contour map of the Bird Function
3 % This is to be used as the background of the canvas
4 n = 1000;
5 range = linspace(-6,6,n);
6 y = Bird( ones( n,1 ) * range , range' * ones( 1,n ) );
7
8 contour( range, range, y, 23);
9 axis equal
10 axis off
11 end
```

### A.2 Bird.png



### A.3 index.html

```
1 <!DOCTYPE html>
2
3 <html>
4 <head>
5   <title>Optimization Ex3</title>
6   <meta charset="utf-8">
7   <script type="text/javascript" src="minmax.js">
8 </script>
9   <script type="text/javascript" src="ex3.js">
10 </script>
11   <script type="text/javascript" src="genetic_algorithm.js">
12 </script>
13   <script type="text/javascript" src="tabu.js">
```

```
14 </script>
15 <style type="text/css">
16 body{
17     font-family: monospace;
18 }
19 h1 {
20     text-align: center;
21     font-size: 16pt;
22 }
23 .parent {
24     max-width: 500px;
25     max-height: 500px;
26     margin: 1em auto;
27     border: 1px solid black;
28 }
29 .buttons {
30     margin: 1em auto;
31     text-align: center;
32 }
33 .animParent{
34     margin: 0.5em auto;
35     padding: 0;
36     height: 4em;
37     overflow: hidden;
38     text-align: center;
39 }
40 #loadingAnim{
41     margin: 0 auto;
42     display: none;
43     background-color: black;
44     border: 1px solid black;
45     -webkit-animation: loader 2s ease infinite;
46     animation: loader 2s ease infinite;
47 }
48 @-webkit-keyframes loader {
49     from {
50         opacity: 1.0;
51         margin-top: 1em;
52         width: 0em;
53         height: 0em;
54         border-bottom-right-radius: 0em;
55         border-bottom-left-radius: 0em;
56         border-top-right-radius: 0em;
57         border-top-left-radius: 0em;
58     }
59     to {
60         opacity: 0;
61         margin-top: 0em;
62         width: 4em;
63         height: 4em;
64         border-bottom-right-radius: 2em;
65         border-bottom-left-radius: 2em;
66         border-top-right-radius: 2em;
67         border-top-left-radius: 2em;
68     }
69 }
70 @keyframes loader {
71     from {
72         opacity: 1.0;
73         margin-top: 2em;
74         width: 0em;
75         height: 0em;
```

```

76     border-bottom-right-radius: 0em;
77     border-bottom-left-radius: 0em;
78     border-top-right-radius: 0em;
79     border-top-left-radius: 0em;
80 }
81 to {
82     opacity: 0;
83     margin-top: 0em;
84     width: 4em;
85     height: 4em;
86     border-bottom-right-radius: 2em;
87     border-bottom-left-radius: 2em;
88     border-top-right-radius: 2em;
89     border-top-left-radius: 2em;
90 }
91 }
92 #controls {
93     text-align: left;
94     margin: 0 auto;
95     width: 30em;
96 }
97 </style>
98 </head>
99
100 <body>
101     <h1>Bird Function Optimizer</h1>
102
103     <div class="parent">
104         <canvas id="c" width="500" height="500"></canvas>
105     </div>
106
107     <div class="buttons">
108         <button class="algorithm" onclick=
109             "genetic_algorithm(_bird_function_)">Genetic Algorithm</button>
110         <button class="algorithm" onclick=
111             "tabu_search(_bird_function_)">Tabu Search</button>
112
113         <div class="animParent">
114             <div id="loadingAnim">
115                 &nbsp;
116             </div>
117         </div>
118
119         <div id="urlsection"></div>
120
121     </div>
122
123     <div id="controls">
124         <h3>Settings:</h3>
125
126         <span>Pause Between Iterations</span><br>
127         <input type="range" class="range" min="0" max="1000" step="1"
128             value="500" id="pause" onchange="updateSlider('pause')">
129         <span id="pausespan"></span> ms<br>
130
131         <h3>GA Specific Settings:</h3>
132
133         <span>Population Size</span><br>
134         <input type="range" class="range" min="10" max="50" step="5"
135             value="25" id="gapopulation" onchange=
136             "updateSlider('gapopulation')"> <span id=
137             "gapopulationspan"></span><br>

```

```

138
139 <span>Parent Count</span><br>
140 <input type="range" class="range" min="5" max="25" step="1"
141 value="10" id="gaparents" onchange="updateSlider('gaparents')">
142 <span id="gaparentsspan"></span><br>
143
144 <span>Selection Strategy</span><br>
145 <select id="gastrategy">
146 <option value="tournament">
147 Tournament Select
148 </option>
149 <option value="frequency">
150 Frequency Dependant Select
151 </option>
152 </select><br>
153
154 <span>Mutation Rate</span><br>
155 <input type="range" class="range" min="0" max="0.5" step="0.02"
156 value="0.02" id="gamutate" onchange="updateSlider('gamutate')">
157 <span id="gamutatespan"></span><br>
158
159 <h3>Tabu Specific Settings</h3>
160
161 <span>Short Term Memory</span><br>
162 <input type="range" class="range" min="1" max="20" step="1"
163 value="7" id="tabushort" onchange="updateSlider('tabushort')">
164 <span id="tabushortspan"></span><br>
165
166 <span>Medium Term Memory</span><br>
167 <input type="range" class="range" min="1" max="20" step="1"
168 value="4" id="tabumedium" onchange=
169 "updateSlider('tabumedium')"> <span id=
170 "tabumediumspan"></span><br>
171
172 <span>Long Term Memory Grid Length</span><br>
173 <input type="range" class="range" min="1" max="20" step="1"
174 value="3" id="tabulong" onchange="updateSlider('tabulong')">
175 <span id="tabulongspan"></span> per side<br>
176
177 <span>Intensification Step</span><br>
178 <input type="range" class="range" min="1" max="40" step="1"
179 value="10" id="tabuintensify" onchange=
180 "updateSlider('tabuintensify')"> <span id=
181 "tabuintensifyspan"></span><br>
182
183 <span>Diversification Step</span><br>
184 <input type="range" class="range" min="1" max="40" step="1"
185 value="15" id="tabudiversify" onchange=
186 "updateSlider('tabudiversify')"> <span id=
187 "tabudiversifyspan"></span><br>
188
189 <span>Step Size Reduction Step</span><br>
190 <input type="range" class="range" min="1" max="40" step="1"
191 value="25" id="tabustepreduce" onchange=
192 "updateSlider('tabustepreduce')"> <span id=
193 "tabustepreducespan"></span><br>
194
195 </div>
197 </body>
198 </html>

```

## A.4 minmax.js

```

1 Array.prototype.min = function(comparer) {
2
3     if (this.length === 0) return null;
4     if (this.length === 1) return this[0];
5
6     comparer = (comparer || Math.min);
7
8     var v = this[0];
9     for (var i = 1; i < this.length; i++) {
10         v = comparer(this[i], v);
11     }
12
13     return v;
14 }
15
16 Array.prototype.max = function(comparer) {
17
18     if (this.length === 0) return null;
19     if (this.length === 1) return this[0];
20
21     comparer = (comparer || Math.max);
22
23     var v = this[0];
24     for (var i = 1; i < this.length; i++) {
25         v = comparer(this[i], v);
26     }
27
28     return v;
29 }

```

## A.5 ex3.js

```

1 "use_strict";
2
3 function draw_background( canvas, ctx ){
4     var backgroundElem = document.getElementById( "birdFunctionContour" );
5     ctx.clearRect ( 0 , 0 , canvas.width, canvas.height );
6     ctx.drawImage( backgroundElem, 0, 0, canvas.width, canvas.height );
7 }
8
9 function clear_screen(){
10     var canvas = getCanvasAndContext()[0];
11     var context = getCanvasAndContext()[1];
12     draw_background( canvas, context );
13 }
14
15 var evalCount = 0;
16 var resetEvalCount = function(){
17     evalCount = 0;
18 };
19 var getEvalCount = function(){
20     return evalCount;
21 }
22
23 function bird_function( x1, x2 ){
24     evalCount = evalCount + 1;
25     var y = Math.sin(x1) * Math.exp( Math.pow(1 - Math.cos(x2),2 ) ) +
26         Math.cos(x2) * Math.exp ( Math.pow(1 - Math.sin(x1), 2 ) )+

```



```

27     Math.pow(x1 - x2, 2);
28     return y;
29 }
30
31 var getCanvasAndContext;
32
33 var drawPoint = function( x1, x2, colour ){
34
35     var canvas = getCanvasAndContext()[0];
36     var context = getCanvasAndContext()[1];
37
38     var centerX = canvas.width * ( x1 + 6 )/12;
39     var centerY = canvas.height * ( -x2 + 6 ) /12;
40     var radius = 2;
41     //window.console.log( "X1: " + x1 + " X2: " + x2 );
42     context.beginPath();
43     context.arc(centerX, centerY, radius, 0, 2 * Math.PI, false);
44     context.fillStyle = colour || 'green';
45     context.fill();
46     context.lineWidth = 0.5;
47     context.strokeStyle = 'black';
48     //context.stroke();
49 }
50 var connectPoints = function( a1, a2, b1, b2 ){
51
52     var canvas = getCanvasAndContext()[0];
53     var context = getCanvasAndContext()[1];
54
55     var aX = canvas.width * ( a1 + 6 )/12;
56     var aY = canvas.height * ( -a2 + 6 ) /12;
57     var bX = canvas.width * ( b1 + 6 )/12;
58     var bY = canvas.height * ( -b2 + 6 ) /12;
59
60     context.beginPath();
61     context.moveTo(aX, aY);
62     context.lineTo(bX, bY);
63     context.stroke();
64 }
65
66 function clearChildren( node ){
67     while (node.firstChild) {
68         node.removeChild(node.firstChild);
69     }
70 }
71
72 function displayCsvStringAsURL( string ){
73     var d = document.getElementById("urlsection");
74     var a = document.createElement("a");
75     a.href = "data:text/csv," + encodeURIComponent( string );
76     a.textContent = "data";
77     clearChildren( d );
78     d.appendChild(a);
79 }
80
81 function logMinimumHistory( minimumHistory ){
82     var csvString = "Evaluations,\u00x1,\u00x2,\u00y\n";
83     minimumHistory.forEach( function( h ){
84         csvString += h.evaluations + ",\u00" + h.x1 + ",\u00" + h.x2 + ",\u00" + h.y + "\n";
85     } );
86     displayCsvStringAsURL( csvString );
87 }
88

```

```

89 function getIterationPause(){
90     return document.getElementById( "pause" ).value;
91 }
92
93
94 function setRunning(){
95     var d = document.getElementById("urlsection");
96     clearChildren(d);
97     d.textContent = "Running";
98     document.getElementById("loadingAnim").style.display = "block";
99
100     var buttons = document.getElementsByClassName( "algorithm" );
101     [].forEach.call( buttons, function(b){
102         b.disabled = true;
103     } );
104 }
105
106 function finishRunning(){
107     document.getElementById("loadingAnim").style.display = "";
108     var buttons = document.getElementsByClassName( "algorithm" );
109     [].forEach.call( buttons, function(b){
110         b.disabled = false;
111     } );
112 }
113
114 function updateSlider(id){
115     var slider = document.getElementById( id );
116     var label = document.getElementById( id + "span" );
117     label.textContent = slider.value;
118 }
119
120 window.onload = function(){
121     var canvas = document.getElementById( "c" );
122     var ctx=canvas.getContext("2d");
123     draw_background( canvas, ctx );
124
125     var ranges = document.getElementsByClassName( "range" );
126     [].forEach.call( ranges, function(r){
127         updateSlider( r.id );
128     } );
129
130     getCanvasAndContext = function(){
131         return [canvas, ctx];
132     }
133     finishRunning();
134 }

```

## A.6 tabu.js

```

1  "use_strict";
2
3  function TabuPoint( f, x1, x2 ){
4      this.x1 = x1 || 0;
5      this.x2 = x2 || 0;
6
7      this.getValue = function(){
8          return [this.x1, this.x2];
9      }
10
11     var lastCalled;
12     this.getFValue = function(){

```

```

13     if( lastCalled === undefined || !(this.isEqual( lastCalled )) ){
14         this.fValue = f( this.x1, this.x2 );
15         lastCalled = this.clone();
16     }
17     return( this.fValue );
18 }
19 this.isEqual = function( p ){
20     if( p.x1 === this.x1 && p.x2 === this.x2 ){
21         return true;
22     }
23     return false;
24 }
25 this.clone = function(){
26     var o = new TabuPoint( f, this.x1, this.x2 );
27     o.lastCalled = this.lastCalled;
28     o.fValue = this.fValue;
29     return o;
30 }
31 this.valid = function(){
32     if( Math.abs( this.x1 ) <= 6.0 && Math.abs( this.x2 ) < 6.0 ){
33         return true;
34     }
35     return false;
36 }
37 }
38
39 function tabuMin( a, b ){
40     return a.getFValue() < b.getFValue() ? a : b;
41 }
42
43 function tabuMax( a, b ){
44     return a.getFValue() < b.getFValue() ? a : b;
45 }
46
47 function considerForMediumTermMemory( memory, point, size ){
48     if( memory.length < size ){
49         memory.push( point );
50     } else {
51         var max = memory.max( tabuMax );
52         if( max.getFValue() > point.getFValue() ){
53             // add the point
54             var rIndex = memory.indexOf( max );
55             memory.splice( rIndex, 1, point );
56         }
57     }
58 }
59
60
61 function addToMemory( memory, value, memSize ){
62     memory.push( value );
63     if( memory.length > memSize ){
64         memory.splice( 0, memory.length - memSize );
65     }
66 }
67
68 function getAveragePoint( memory, f ){
69     var x1 = 0, x2 = 0;
70     memory.forEach( function(m){
71         x1 += m.x1/memory.length;
72         x2 += m.x2/memory.length;
73     } );
74     return new TabuPoint( f, x1, x2 );

```

```

75 }
76
77 function setupLongTermMemory(size){
78   var m = Array(size);
79   for( var i = 0; i< size; i++ ){
80     m[i] = Array( size );
81     for( var j = 0; j< size; j++ ){
82       m[i][j] = 0;
83     }
84   }
85   return m;
86 }
87
88 function addToLongTermMemory(memory, point){
89   var i = Math.floor( memory.length*( point.x1 + 6 )/12.01 );
90   var j = Math.floor( memory.length*( point.x2 + 6 )/12.01 );
91   memory[i][j] +=1;
92 }
93
94 function getDiversePointFromLongTermMemory(memory, f){
95   for( var i = 0; i< memory.length; i++ ){
96     for( var j = 0; j< memory[i].length; j++ ){
97       if( memory[i][j] === 0 ){
98         memory[i][j] = 1;
99         window.console.log( "i:␣" + i + ",␣j:␣" + j + ",␣m:" + memory[i][j]);
100         return new TabuPoint( f,
101           12 * ((i+0.5)/memory.length) - 6,
102           12 * ((j+0.5)/memory.length) - 6
103         );
104       }
105     }
106   }
107   return new TabuPoint( f, 0, 0 );
108 }
109
110 function updateTabuMinimumHistory( minimumHist, minimum ){
111   var o = {
112     evaluations: getEvalCount(),
113     x1: minimum.x1,
114     x2: minimum.x2,
115     y: minimum.getFValue()
116   };
117   minimumHist.push( o );
118 }
119
120 function getShortSize(){
121   return document.getElementById( "tabushort" ).value;
122 }
123 function getMediumSize(){
124   return document.getElementById( "tabumedium" ).value;
125 }
126 function getLongSize(){
127   return document.getElementById( "tabulong" ).value;
128 }
129 function getIntensifyStep(){
130   return document.getElementById( "tabuintensify" ).value;
131 }
132 function getDiversifyStep(){
133   return document.getElementById( "tabudiversify" ).value;
134 }
135 function getReduceStep(){
136   return document.getElementById( "tabustepreduce" ).value;

```

```

137 }
138
139
140 function tabu_search( f ){
141
142     var shortTerm = [];
143     var mediumTerm = [];
144     var longTerm = setupLongTermMemory( getLongSize() );
145
146     var minimumHist = [];
147
148     var point = new TabuPoint( f, 0, 0 );
149     var minimum = point;
150
151     var initialInterval = 1;
152     var interval = initialInterval;
153     minimum.interval = initialInterval;
154
155     resetEvalCount();
156     updateTabuMinimumHistory( minimumHist, minimum );
157
158     setRunning();
159
160     var isInShortTermMem = function( p ){
161         var found = false;
162         shortTerm.forEach( function( s ){
163             if( s.isEqual( p ) ){
164                 found = true;
165             }
166         });
167         return found;
168     };
169
170     var improvementCounter = 0;
171
172     clear_screen();
173
174     var step = function(){
175         var nextSteps = [];
176         ["x1", "x2"].forEach( function( param ){
177             var inc = point.clone();
178             var dec = point.clone();
179             inc[param] += interval;
180             dec[param] -= interval;
181             if( !isInShortTermMem( inc ) && inc.valid() ){
182                 nextSteps.push( inc );
183             }
184             if( !isInShortTermMem( dec ) && dec.valid() ){
185                 nextSteps.push( dec );
186             }
187         });
188         var best = nextSteps.min( tabuMin );
189         if( nextSteps.length === 0 ){
190             window.console.log( "All points are Tabu" );
191             best = point;
192         }
193
194         if( best.getFValue() < point.getFValue() ){
195             //pattern move
196             var change = { x1: best.x1 - point.x1, x2: best.x2 - point.x2 };
197             var pattern = point.clone();
198             pattern.x1 += 2.0 * change.x1;

```

```

199     pattern.x2 += 2.0 * change.x2;
200     if( pattern.getFValue() < best.getFValue() && pattern.valid() ){
201         best = pattern;
202     }
203 }
204
205 // store old point for Line Drawing purposes
206 var oldPoint = point;
207
208 point = best;
209
210 addToMemory( shortTerm, best, getShortSize() );
211 considerForMediumTermMemory( mediumTerm, best, getMediumSize() );
212 addToLongTermMemory( longTerm, best );
213
214 improvementCounter += 1;
215 if( point.getFValue() < minimum.getFValue() ){
216     improvementCounter = 0;
217     minimum = point;
218     minimum.interval = interval;
219     updateTabuMinimumHistory( minimumHist, minimum );
220 }
221
222 var colour = "green";
223 window.console.log
224 if( improvementCounter == getIntensifyStep() ){
225     //Intensify
226     window.console.log( "intensifying" );
227     point = getAveragePoint( mediumTerm, f );
228     colour = "red";
229 } else if( improvementCounter == getDiversifyStep() ) {
230     //Diversify
231     window.console.log( "diversifying" );
232     point = getDiversePointFromLongTermMemory(longTerm, f);
233     window.console.log( "point:" + point.x1 + "," + point.x2 );
234     // clear the minimum term memory
235     minimumTerm = [];
236     //reset the step Size
237     interval = initialInterval;
238     colour = "cyan";
239 } else if( improvementCounter == getReduceStep() ){
240     //Step Size Reduction
241     window.console.log( "StepSizeReduce" );
242     point = minimum;
243     interval = 0.5 * minimum.interval;
244     colour = "yellow"
245     improvementCounter = 0;
246 } else {
247     connectPoints( oldPoint.x1, oldPoint.x2, point.x1, point.x2 );
248 }
249 drawPoint( point.x1, point.x2, colour );
250
251 if( point.getFValue() < minimum.getFValue() ){
252     improvementCounter = 0;
253     minimum = point;
254     minimum.interval = interval;
255     updateTabuMinimumHistory( minimumHist, minimum );
256 }
257
258 if( getEvalCount() < 1000 - 10 ){
259     window.setTimeout( step, getIterationPause() );
260 } else {

```

```

261     logMinimumHistory( minimumHist );
262     finishRunning();
263   }
264 }
265 step();
266 }

```

## A.7 genetic\_algorithm.js

```

1  "use_strict";
2
3  function GAPoint(){
4    var precision = 16;
5    this.x1 = Array(precision);
6    this.x2 = Array(precision);
7
8    var getSinglePoint = function( xval ){
9      var val = -6;
10     for( var i = 0; i< precision; i++ ){
11       if( xval[i] ){
12         val += 6.0 * Math.pow(0.5, i);
13       }
14     }
15     return val;
16   }
17
18   this.getValue = function(){
19     return [ getSinglePoint( this.x1 ), getSinglePoint( this.x2 ) ];
20   }
21   this.getFunctionValue = function( f ){
22     if( this.fvalue === undefined ){
23       var x = this.getValue();
24       this.fvalue = f( x[0], x[1] );
25     }
26     return this.fvalue;
27   }
28 }
29
30
31 function getRandomGAPoint(){
32   var point = new GAPoint();
33   var randomizeBool = function(){
34     var a;
35     if(Math.random()<.5){
36       a = true;
37     } else {
38       a = false;
39     }
40     return a;
41   };
42   for( var i = 0; i< point.x1.length; i++ ){
43     point.x1[i] = randomizeBool();
44     point.x2[i] = randomizeBool();
45   }
46   return point;
47 }
48
49 function getSeveralRandomGAPoints( N ){
50   var points = Array(N);
51   for( var i = 0; i< N; i++ ){
52     points[i] = getRandomGAPoint();

```

```

53     }
54     return points;
55 }
56
57 function drawGAPoints( points, colour ){
58
59     points.forEach( function(p) {
60         var x = p.getValue();
61         drawPoint( x[0], x[1], colour );
62     } );
63
64 }
65
66
67 function swapPoints( points, i, j ){
68     var tmp = points[i];
69     points[i] = points[j];
70     points[j] = tmp;
71 }
72
73 function getRandomInt(min, max) {
74     return Math.floor(Math.random() * (max - min + 1)) + min;
75 }
76
77 function fisherYatesShuffle( list ){
78     for( var i = 0; i < list.length; i++ ){
79         var j = getRandomInt( i, list.length - 1 );
80         swapPoints( list, i, j );
81     }
82 }
83
84
85 function split(a, n) {
86     var len = a.length;
87     var out = [];
88     var i = 0;
89     while (i < len) {
90         var size = Math.ceil((len - i) / n--);
91         out.push(a.slice(i, i += size));
92     }
93     return out;
94 }
95
96 function tournamentSelect( points, f, groups ){
97
98     var size = points.length / groups;
99     fisherYatesShuffle( points );
100    var splitGroups = split( points, groups );
101    var comparer = function( a, b ){
102        return a.getFunctionValue(f) < b.getFunctionValue(f) ? a : b;
103    }
104
105    var selected = [];
106
107    splitGroups.forEach( function(g){
108        selected.push( g.min(comparer) );
109    } );
110    return selected;
111 }
112
113 function fitnessProportionateSelect( points, f, N ){
114     var sum = 0;

```



```

115 //the function has range that's roughly -100 to 100
116 // it goes a little lower than this, but we can still use 100 - function as a
117 // fitness score
118 points.forEach( function(p){
119     sum += 100 - p.getFunctionValue(f);
120 } );
121 var selected = [];
122 for( var i = 0; i < N; i++ ){
123     var r = Math.random();
124     var j = 0;
125     while( r > 0 && j < points.length ){
126         r -= (100-points[j].getFunctionValue(f))/sum;
127         j++;
128     }
129     if( points[j-1] === undefined ) alert("scary");
130     selected.push( points[j-1] );
131 }
132 return selected;
133 }
134
135 function doSingleValCrossover( p1, p2, c1, c2, val ){
136     var crossoverPoint1 = getRandomInt( 1,
137         Math.floor( ( p1[val].length -1) * 0.75 ) );
138     var crossoverPoint2 = getRandomInt( crossoverPoint1, p1[val].length );
139     for( var i = 0; i < p1.x1.length; i++ ){
140         if( i < crossoverPoint1 || i > crossoverPoint2 ){
141             c1[val][i] = p1[val][i];
142             c2[val][i] = p2[val][i];
143         } else {
144             c2[val][i] = p1[val][i];
145             c1[val][i] = p2[val][i];
146         }
147     }
148 }
149
150 function crossTwoPoints( p1, p2 ){
151     var c1 = new GPoint();
152     var c2 = new GPoint();
153     doSingleValCrossover( p1, p2, c1, c2, "x1" );
154     doSingleValCrossover( p1, p2, c1, c2, "x2" );
155     return [c1, c2];
156 }
157
158
159 function breedPoints( selectedPoints, nextGenSize ){
160     var newPoints = [];
161     for( var i = 0; i < nextGenSize/2; i += 1){
162         // pair each point in turn with a random point, and breed them
163         newPoints = newPoints.concat(
164             crossTwoPoints(
165                 selectedPoints[i%selectedPoints.length],
166                 selectedPoints[getRandomInt(0, selectedPoints.length-1)]
167             )
168         );
169     }
170     return newPoints;
171 }
172
173
174 function mutatePoint( point, value ){
175     var bit = getRandomInt( 0, point[value].length -1 );
176     if( point[value][bit] ){

```

```

177     point[value][bit] = false;
178   } else {
179     point[value][bit] = true;
180   }
181 }
182
183 function getMutationRate(){
184   return document.getElementById( "gamutate" ).value;
185 }
186
187 function mutatePoints(points){
188   var mutationProbability = getMutationRate();
189
190   points.forEach( function(p){
191     if( Math.random() < mutationProbability ){
192       var value = "x1";
193       if(Math.random() < 0.5 ){
194         value = "x2";
195       }
196       mutatePoint( p, value );
197     }
198   } );
199 }
200
201 function updateGAMinimumHistory( points, f, minimumHistory ){
202   var comparer = function( a, b ){
203     return a.getFunctionValue(f) < b.getFunctionValue(f) ? a : b;
204   }
205
206   var min = points.min(comparer)
207   var x = min.getValue();
208   var y = min.getFunctionValue( f );
209
210   if( minimumHistory.length === 0 ||
211     minimumHistory[minimumHistory.length-1].y > y ){
212     var o = {};
213     o.x1 = x[0];
214     o.x2 = x[1];
215     o.y = y;
216     o.evaluations = getEvalCount();
217     minimumHistory.push( o );
218   }
219 }
220
221
222
223 function getGAPopulation(){
224   return document.getElementById( "gapopulation" ).value;
225 }
226
227 function getGAParentsCount(){
228   return document.getElementById( "gaparents" ).value;
229 }
230
231 function getGAParents(points, f){
232   var strategy = document.getElementById( "gastrategy" ).value;
233   if( strategy === "tournament" ){
234     return tournamentSelect( points, f, getGAParentsCount() );
235   } else if( strategy === "frequency" ){
236     return fitnessProportionateSelect( points, f, getGAParentsCount() );
237   }
238 }

```

```
239
240 function GAIteration( points, f, minimumHistory ){
241   clear_screen();
242
243   drawGAPoints( points, "green" );
244
245   var reproducing = getGAParents(points, f );
246
247   drawGAPoints( reproducing, "red" );
248
249   var nextGen = breedPoints(reproducing, points.length);
250
251   updateGAMinimumHistory( points, f, minimumHistory);
252
253   var last = minimumHistory[minimumHistory.length-1];
254
255   drawPoint( last.x1, last.x2, "blue" );
256
257   mutatePoints( nextGen );
258
259   if( getEvalCount() < 1000 - nextGen.length){
260     window.setTimeout( function(){
261       GAIteration( nextGen, f, minimumHistory );
262     }, getIterationPause() );
263   } else {
264     logMinimumHistory( minimumHistory );
265     finishRunning();
266   }
267 }
268
269 function genetic_algorithm( f ){
270   resetEvalCount();
271   var nPoints = getGAPopulation();
272   var points = getSeveralRandomGAPoints(nPoints);
273   setRunning();
274   GAIteration(points, f, []);
275 }
```