```
<html>

<head>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=windows-1252">
<title>Nucleoid Model</title>
<meta name="keywords" content="mtDNA, mitochondria">
</head>

<script language="JavaScript">

<!-- Global variables
     var mut_num = nsize_num = ncount_num = nmflag = 0;
     var mcount_num = rep_num = run_num = gen_num = resflag = homflag = 0;

<!-- Set initial values
function fnInitial() {
     mut_num = 1000;
     nsize_num = 10;
     ncount_num = 10;
     nmflag = 0;

     mcount_num = 200;
     rep_num = 5;
     run_num = 10;
     gen_num = 20;
     resflag = 1;
     homflag = 0;
}

<!-- Handle selections
function fnMnum(what) {
     mut_num = what.options[what.selectedIndex].value - 0;
     return ""
}

function fnNsize(what) {
     nsize_num = what.options[what.selectedIndex].value - 0;
     return ""
}

function fnNcount(what) {
     ncount_num = what.options[what.selectedIndex].value - 0;
     return ""
}

function fnN_or_m(what) {
     nmflag = what.options[what.selectedIndex].value - 0;
     return ""
}

function fnMcount(what) {
     mcount_num = what.options[what.selectedIndex].value - 0;
     return ""
}

function fnRep(what) {
     rep_num = what.options[what.selectedIndex].value - 0;
     return ""
}

function fnRuns(what) {
     run_num = what.options[what.selectedIndex].value - 0;
     return ""
}

function fnGen(what) {
     gen_num = what.options[what.selectedIndex].value - 0;
```

```
     return ""
}

function fnRes(what) {
     resflag = what.options[what.selectedIndex].value - 0;
     return ""
}

<!-- Clear a text area
function fnClear(){
     return ""
}

<!--
<!-- Phase A: Starts here

function fnStart(){
     var istr = "";
     var r;
     var eachr;

     istr =        "      Number of Mutations  = " + mut_num + "\n";
     istr = istr + "      Nucleoid Size        = " + nsize_num + "\n";


<!-- Nucleoid selected
     if (nmflag == 0)
      {

<!-- For each initial event in turn use function Eachmut to find a Nucleoid capture
       for (r=1; r<=mut_num; r++)
        {

         eachr=fnEachmut(nsize_num);

<!-- End conditions
         if (eachr != 0)
          {istr = istr + "\n" + "      Mutation " + r + " captured a nucleoid.\n";
           return istr}
        }
       istr = istr + "\n" + "      No nucleoid captured.\n";
       return istr
      }

<!-- Mitochondrion selected
     if (nmflag == 1)
      {
       istr = istr + "      Nucleoid Number      = " + ncount_num + "\n\n";
       for (r=1; r<=mut_num; r++)
        {

<!-- For each initial event in turn use function Eachmut to find a Nucleoid capture
         eachr=fnEachmut(nsize_num);
         if (eachr != 0)
          {

<!-- For a Nucleoid capture use function Eachmut to look for a Mitochondrion capture
           eachr=fnEachmut(ncount_num);
           if (eachr !=0)

<!-- End conditions
           {istr = istr + "      Mutation " + r + " captured a mitochondrion.\n";
            return istr }
           }
          }
         istr = istr + "      No mitochondrion captured.\n";
         return istr
```

```
      }
}

<!-- For each event, randomly add a mutation or remove a mutation

function fnEachmut(size_num){
     var i;
     var muts;
     muts = 1;
     for (i=1; i<=200; i++){
      if (Math.floor(1+(Math.random()* size_num)) <= muts) muts = muts + 1;
      else muts = muts - 1;

<!-- end conditions
      if (muts == 0) return muts;
      if (muts == size_num) return muts;
     }
     return muts
}

<!--
<!-- Phase B: Starts here

function fnStart2(){
     istr = "";
     var tstr;
     var r;
     var p_num;

     istr = "     Initial Level of Heteroplasmy = " + 100/mcount_num + "% \n";
     istr = istr + "     Replications per Generation   = " + rep_num + "\n";
     istr = istr + "     Level of Heteroplasmy over "+ gen_num + " generations"+ "\n";

<!-- Handle each run in turn

     for (r=1; r<=run_num; r++){
      tstr = fnEachrun();
      if (resflag == 1) istr = istr + r + ". " + tstr + "\n";
      else if (homflag==1) istr = istr + r + ". " + tstr + "\n";
     }
     istr =istr + "End of runs";
     return istr
}


<!-- Single run handler
<!-- For each run, randomly add or remove 'mutated mitochondria'
<!-- global variable rep_num is number of replications per generation
<!-- global variable mcount_num is number of mitochondria in a cell

function fnEachrun(){
       var p_num;
       var k;
       var i;
       var j;
       var pstr = " ";
       var m_mt = 1;
       var t_mt;

<!-- Handle each generation,

       for (j=1; j<=gen_num; j++){

<!-- Increase level of mutated mitochondria at random with each replication
        for (i=1; i<=rep_num; i++){
         t_mt=0;
         for (k=1; k<=(m_mt*2); k++){
```

```
          if (Math.random()<=0.5) t_mt = t_mt + 1 ;
         }
         m_mt = t_mt
        }
        p_num =  (m_mt*100)/mcount_num;

<!-- End conditions
        homflag = 0;
        if (p_num <= 0) return pstr + " 0% ....";
        if (p_num >= 100){
         homflag = 1;
         return pstr + " 100%  HOMOPLASMY ACHIEVED";
         }
        pstr = pstr + " " + p_num + "%";
       }
       return pstr
}

//-->
</script>

<body onLoad = "fnInitial()">
<br>
<center><h1>On the Propagation of Mitochondrial Mutations</h1>
<h3>Click to see the <a href = "#instruc">Instructions</a>
 or a brief explanation of the <a href = "#tech">Modelling Technique</a>
 used</h3>
<table border="3" width="550" bgcolor="#bbbbbb"
style = "font-family: Arial; color: blue; font-size: 16">
<tr><td>
<center>
<form name = "model">
<center>
<br>
ver. 1.00
        
<b>Computer Model</b>
        
&#169;2009 Ian Logan
<br>
<br>
PHASE A:    From Initial Mutational event - to - Mitochondrial Capture
</center>
<br>
<br>
    
Initial event
        
<select name="mutnumber" onChange="fnMnum(this)">
<option value = "1">1</option>
<option value = "100">100</option>
<option value = "1000" selected>1K</option>
<option value = "10000">10K</option>
<option value = "100000">100K</option>
<option value = "1000000">1M</option>
<option value = "10000000">10M</option>
</select>
        
Nucleoid Size
        
<select name="nuclsize" onChange="fnNsize(this)">
<option value = "5">5</option>
<option value = "10"selected>10</option>
<option value = "15">15</option>
<option value = "20">20</option>
<option value = "30">30</option>
</select>
```

```
        
Nucleoid No.
    
<select name="nuclcount" onChange="fnNcount(this)">
<option value = "5">5</option>
<option value = "10" selected>10</option>
<option value = "15">15</option>
<option value = "20">20</option>
</select>
<br>
<br>
<center>
Capture a
<select name="n_or_m_choice" onChange="fnN_or_m(this)">
<option value = "0" selected >Nucleoid</option>
<option value = "1" >Mitochondrion</option>
</select>
            
<input type="button" value="CLEAR"  onClick="
          document.model.results.value = fnClear()">
            
<input type="button" value="A: START"
    onClick="document.model.results.value = fnStart()">
</select>
<br>
<br>
<textarea readonly name = "results"
  rows = "5"  cols = "102" wrap="off"
  style = "font-size: 12">
</textarea>
<br>
<br>
<center>
PHASE B:    From Mitochondrial Capture - to - Heteroplasmy - to -
Homoplasmy
<br>
<br>Runs
        
<select name="Number of Runs" onChange="fnRuns(this)">
<option value = "10" selected>10</option>
<option value = "100">100</option>
<option value = "1000">1K</option>
</select>
        
Mitochondrial No.
        
<select name="Mitocount" onChange="fnMcount(this)">
<option value = "10">10</option>
<option value = "25">25</option>
<option value = "50">50</option>
<option value = "100">100</option>
<option value = "200" selected>200</option>
<option value = "400">400</option>
<option value = "800">800</option>
</select>
        
Replications
        
<select name="replications" onChange="fnRep(this)">
<option value = "2">2</option>
<option value = "5" selected>5</option>
<option value = "10">10</option>
<option value = "15">15</option>
<option value = "20">20</option>
</select>
<br>
<br>
```

```
Generations
        
<select name="generations" onChange="fnGen(this)">
<option value = "10">10</option>
<option value = "20" selected>20</option>
<option value = "50">50</option>
<option value = "100">100</option>
<option value = "200">200</option>
<option value = "300">300</option>
</select>
        
Full Results
        
<select name="full" onChange="fnRes(this)">
<option value = "1" selected>YES</option>
<option value = "0">NO</option>
</select>
        
<input type="button" value="CLEAR"  onClick="
          document.model.results2.value = fnClear()">
        
<input type="button" value="B: START"
    onClick="document.model.results2.value = fnStart2()">
</select>
<br>
<br>
<textarea readonly name = "results2"
  rows = "14"  cols = "102" wrap="off"
  style = "font-size: 12">
</textarea>
</form>
</td></tr>
</table>
<br>
<br>
</center>
<a name = "instruc">
<p>
<h3>Instructions</h3>
<p>
This computer model ilustrates the emergence of mitochondrial mutations by considering the
process as occurring in 2 distinct phases:<br>
<br>
<p>
<b>PHASE A: From Initial Mutational event - to - Mitochondrial Capture</b><br>

<p>
To use the <b>Computer Model</b>, click on <b>A: START</b>.<br>
The number of <b>Initial Events</b> and the <b>Nucleoid Size</b> can be varied.<br>
After any alteration, press <b>CLEAR</b> and click on <b>A: START</b> again.<br>

<p>
The <b>Initial Events</b> are to be considered as single mutations occurring in individual
mtDNA molecules.</br>
The <b>Nucleoid Size</b> is the count of the number of mtDNA molecules in each
mitochondrial nucleoid.<br>

<p>
The <b>Computer Model</b> shows that a mutation occurring in a small nucleoid can become
established.<br>
However, as the <b>Nucleoid Size</b> is increased, there is only a small chance of a
mutation successfully 'capturing' the nucleoid.<br>

<p>
Next, change <b>Nucleoid</b> to <b>Mitochondrion</b> and select the number of nucleoids
found in each mitochondrion<br>
Now click on <b>A: START</b> again.<br>
```

<p>
The Computer Model shows that for the successful capture of a mitochondrion it is necessary for the number of initial events
to be high, and both the size of the nucleoid and the number of mtDNA molecules in each mitochondrion to be fairly low.

<p>
PHASE B: From Mitochondrial Capture - to - Heteroplasmy - to - Homoplasmy

<p>
This Computer Model assumes that with the 'capture' of a mitochondrion, it is possible to show how
the level of heteroplasmy varies over a number of generations; and the progression to homoplasmy might be achieved.

<p>
Clicking on B: START shows the heteroplasmy levels over a number of generation, for a specified number of trial runs.
<p>
In this phase, it is possible to alter the number of trial runs, the number of mitochondria per cell, the number
of replications that place between generations and, finally, the number of generations for which the results are calculated.
<p>
The selection of Full Results YES permits viewing of all the results, even when heteroplasmy levels fall to zero.
Whereas, selecting of Full Results NO restricts the viewing to only the trail runs that lead to homoplasmy.

<p>
The Computer Model suggests that a single 'captured' mitochondrion can lead to homoplasmy
when the number of mitochondria per cell is reasonably low.
The model also suggests that low levels of heteroplasmy can exist in a population over many generations.

<a name = "tech">
<p>
<p>
<h3>Modelling Technique</h3>
This Computer Model considers the propagation of a new mutation in three separate stages.

- from an initial mutational event to the capture of a nucleoid.
- from a captured nucleoid to the the capture of a mitochondrion.
- from a captured mitochondrion to heteroplasmy and homoplasmy.
<p>
The first two stages use the same Javascript function to randomly increase or decrease the number of mutations each time the function is called.

The core of the function used is:
<pre>
     for (i=1; i<=200; i++){
      if (Math.floor(1+(Math.random()* size_num)) <= muts) muts = muts + 1;
      else muts = muts - 1;
     }
</pre>
I.e.
If a randomly selected number is less than, or equal to, the present number of mutations

then the number of mutations present is increased by 1, otherwise it is decreased by 1.

And, this operation is repeated until either the mutation dies out, or captures the whole nucleoid, or mitochondrion.
<p>

This routine models the natural world where a particular mtDNA molecule, or nucleoid, can become duplicated at RANDOM
and the mutated mtDNA molecules, or nucleoid, again at RANDOM, can become the dominant form.
<p>
The Javascript Math.random function is only a pseudo-random number generator, but the random number sequence produced
does appear to be sufficiently varied for the present model.
<p>
The third stage of the model uses a similar routine which at RANDOM increases or decreases the level
of heteroplasmy each time the function is called.
<p>
The present model assumes that the maturity of a nucleoid, a mitochondrion, or a cell, makes no dfference
to the propagation of a new mutation. But in any future model it may be appropriate to consider the
how these factors alter the rate at which mew mutations might appear.
<p>
Back

<hr>

2009

</body>
</html>