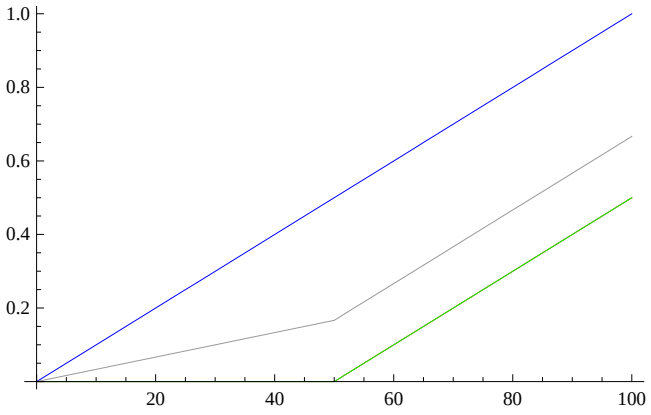
Radio-Sky Spectrograph Color File Generator

Dave Typinski, AJ4CO

August, 2010

In[1]:=
```
(* Define domain for low end fade up from black *)
lo1 = 0;
hi1 = 100;
width1 = hi1 - lo1;

(* Define low end domain functions for red, green, blue *)
fnred1[x_] := x * 1 / width1 - 0.5 /; width1 / 2 ≤ x ≤ width1;
fnred1[x_] := 0 /; 0 ≤ x < width1 / 2;
fngrn1[x_] := x * 1 / width1 - 0.5 /; width1 / 2 ≤ x ≤ width1;
fngrn1[x_] := 0 /; 0 ≤ x < width1 / 2;
fnblu1[x_] := x * 1 / width1;

(* Define a low end domain saturation function for reference use *)
fnsat1[x_] := (fnred1[x] + fngrn1[x] + fnblu1[x]) / 3;

(* Plot low end domain red, green and blue functions along with saturation *)
Plot[{fnred1[x], fngrn1[x], fnblu1[x], fnsat1[x]}, {x, 0, width1},
 PlotStyle → {{RGBColor[1, 0, 0], Thin}, {RGBColor[0, 1, 0], Thin},
   {RGBColor[0, 0, 1], Thin}, {RGBColor[.6, .6, .6], Thin}}]

(* create a table of "width1" number of RGB value triplets,
each value ranging from 0 to 1 *)
Colors1 = Table[{fnred1[i], fngrn1[i], fnblu1[i]}, {i, lo1, hi1}];

(* plot the color table ceated above *)
Graphics[Raster[{Colors1}], AspectRatio → .2]
```

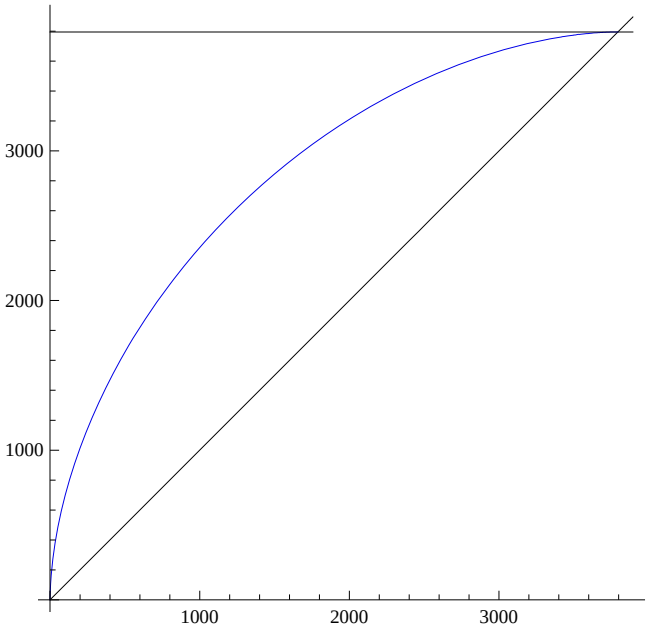
Out[10]=



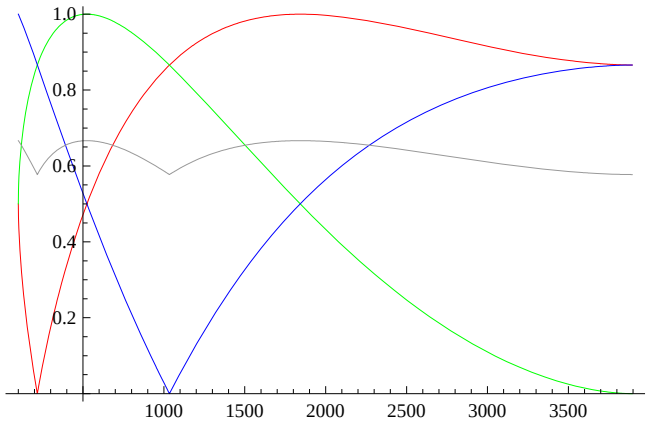
Out[12]=

In[13]:=
```
(* Define domain for bulk of color scheme *)
lo2 = 100;
hi2 = 3895;
width2 = hi2 - lo2;

(* Define squishing function; color changes rapidly at low end, slowly at high end *)
squishfactor = 1.8;
squish[x_] := (width2^squishfactor - (width2 - x)^squishfactor)^(1/squishfactor);
(* plot shishing function for reference *)
Plot[{x, squish[x], width2}, {x, 0, hi2},
 PlotStyle → {{RGBColor[0, 0, 0], Thin}, {RGBColor[0, 0, .9], Thin}}, AspectRatio → 1]


(* Define middle domain functions for red, green, blue *)
fnred2[x_] := N[Abs[Sin[(squish[x - lo2] * 5 / 6 - width2 / 6) π/width2]]];
fngrn2[x_] := N[Abs[Sin[(squish[x - lo2] * 5 / 6 + width2 / 3 - width2 / 6) π/width2]]];
fnblu2[x_] := N[Abs[Sin[(squish[x - lo2] * 5 / 6 - width2 / 3 - width2 / 6) π/width2]]];


(* Define a middle domain saturation function for reference use *)
fnsat2[x_] := (fnred2[x] + fngrn2[x] + fnblu2[x]) / 3;

(* Plot middle domain red, green and blue functions along with saturation *)
Plot[{fnred2[x], fngrn2[x], fnblu2[x], fnsat2[x]}, {x, lo2, hi2},
 PlotStyle → {{RGBColor[1, 0, 0], Thin}, {RGBColor[0, 1, 0], Thin},
   {RGBColor[0, 0, 1], Thin}, {RGBColor[.6, .6, .6], Thin}}]

(* create a table of "width2" number of RGB value triplets,
each value ranging from 0 to 1 *)
Colors2 = Table[{fnred2[i], fngrn2[i], fnblu2[i]}, {i, lo2, hi2}];

(* plot the color table ceated above *)
Graphics[Raster[{Colors2}], AspectRatio → .2]

Print["Red(", lo2, ") = ", fnred2[lo2], "     Red(", hi2, ") = ", fnred2[hi2]]
Print["Grn(", lo2, ") = ", fngrn2[lo2], "     Grn(", hi2, ") = ", fngrn2[hi2]]
Print["Blu(", lo2, ") = ", fnblu2[lo2], "     Blu(", hi2, ") = ", fnblu2[hi2]]
```

Out[18]=



Out[23]=



Out[25]=

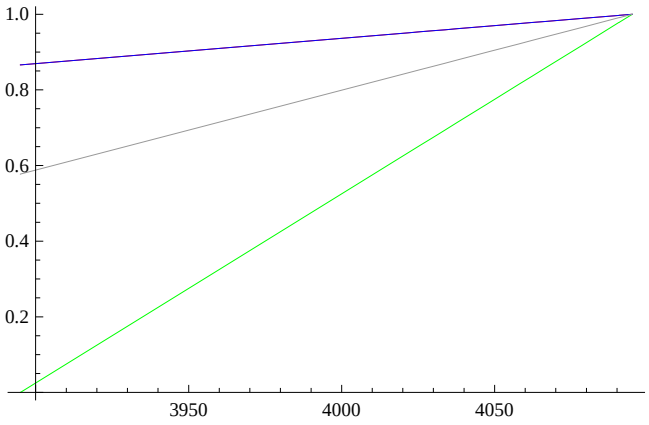
Red(100) = 0.5 Red(3895) = 0.866025

Grn(100) = 0.5 Grn(3895) = 1.2098×10^{-15}

Blu(100) = 1. Blu(3895) = 0.866025

In[29]:=
```
(* Define domain for high end fade to white *)
lo3 = 3895;
hi3 = 4095;
width3 = hi3 - lo3;

(* Define high end domain functions for red, green, blue *)
fnred3[x_] := (x - lo3) * (1 - 0.866025) / width3 + 0.866025;
fngrn3[x_] := (x - lo3) * 1 / width3;
fnblu3[x_] := (x - lo3) * (1 - 0.866025) / width3 + 0.866025;

(* Define a high end domain saturation function for reference use *)
fnsat3[x_] := (fnred3[x] + fngrn3[x] + fnblu3[x]) / 3;

(* Plot high end domain red, green and blue functions along with saturation *)
Plot[{fnred3[x], fngrn3[x], fnblu3[x], fnsat3[x]}, {x, lo3, hi3},
 PlotStyle → {{RGBColor[1, 0, 0], Thin}, {RGBColor[0, 1, 0], Thin},
   {RGBColor[0, 0, 1], Thin}, {RGBColor[.6, .6, .6], Thin}}]

(* create a table of "width3" number of RGB value triplets,
each value ranging from 0 to 1 *)
Colors3 = Table[{fnred3[i], fngrn3[i], fnblu3[i]}, {i, lo3, hi3}];

(* plot the color table ceated above *)
Graphics[Raster[{Colors3}], AspectRatio → .2]
```

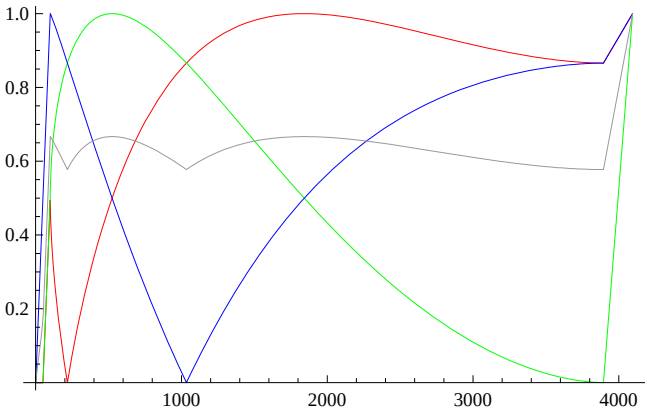
Out[36]=



Out[38]=

In[56]:=
```
(* Define domain of entire color scheme *)
lo = 0;
hi = 4095;
width = hi - lo;

(* Define piecewise domain functions for red, green, blue *)
fnred[x_] := fnred1[x] /; lo1 ≤ x ≤ hi1;
fnred[x_] := fnred2[x] /; lo2 < x ≤ hi2;
fnred[x_] := fnred3[x] /; lo3 < x ≤ hi3;

fngrn[x_] := fngrn1[x] /; lo1 ≤ x ≤ hi1;
fngrn[x_] := fngrn2[x] /; lo2 < x ≤ hi2;
fngrn[x_] := fngrn3[x] /; lo3 < x ≤ hi3;

fnblu[x_] := fnblu1[x] /; lo1 ≤ x ≤ hi1;
fnblu[x_] := fnblu2[x] /; lo2 < x ≤ hi2;
fnblu[x_] := fnblu3[x] /; lo3 < x ≤ hi3;

(* Define a saturation function for reference use *)
fnsat[x_] := (fnred[x] + fngrn[x] + fnblu[x]) / 3;

(* Plot red, green and blue functions along with saturation *)
Plot[{fnred[x], fngrn[x], fnblu[x], fnsat[x]}, {x, lo, hi},
 PlotStyle → {{RGBColor[1, 0, 0], Thin}, {RGBColor[0, 1, 0], Thin},
   {RGBColor[0, 0, 1], Thin}, {RGBColor[.6, .6, .6], Thin}}]

(* create a table of "width" number of RGB value triplets,
each value ranging from 0 to 1 *)
Colors = Table[{fnred[i], fngrn[i], fnblu[i]}, {i, lo, hi}];

(* plot the color table ceated above *)
Graphics[Raster[{Colors}], AspectRatio → .2]
```

Out[69]=



Out[71]=

In[72]:=
```
(* Create Radio-Sky Spectrograph color definitions file *)
ColorDefs = Table[65536 * Round[255 * fnblu[i]] +
    256 * Round[255 * fngrn[i]] + Round[255 * fnred[i]], {i, lo, hi}];
Export["C:\\Program Files\\Spectrograph\\AJ4CO.txt", ColorDefs]
```

Out[73]=
```
C:\Program Files\Spectrograph\AJ4CO.txt
```