Gernot Hoffmann

Visually Balanced Random Points in a Square

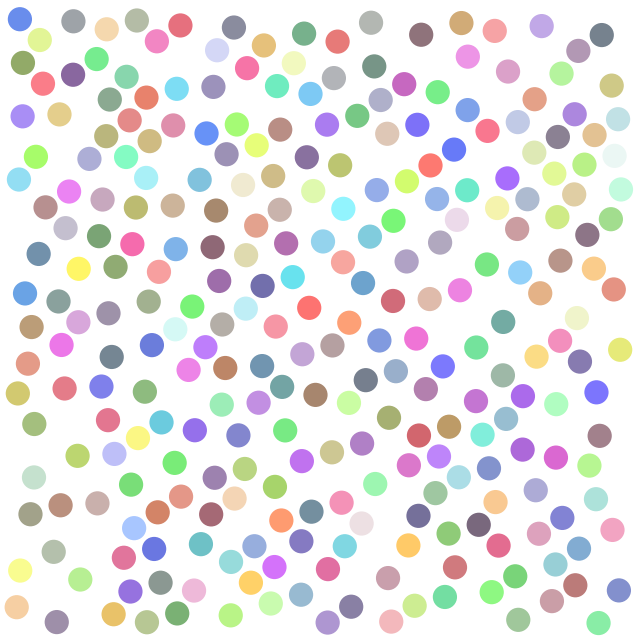


Table of Contents

1. Introduction

For many geometrical tests one needs a small number of points with random coordinates. The distribution in the histogram is not so relevant.
More interesting is the visually balanced distribution of the points in an area, here in a unit square and on the boundary. There should be no clusters and no holes.

The random generator creates pseudo random integer numbers in the range 1..32768 with the period 8192, which is fairly enough for the intended applications. Division by 32768 scales for the range 0.0 to 1.0.

A new point is checked for a minimal Euclidian distance relative to all other already existing points. If the distance is too small then the next random coordinate pair is checked, up to 4096.

It is quite clear that this method is rather slow. Therefore it cannot be used for 'real time' applications. Actual PostScript interpreters need 0.1 to 1 seconds for each pattern. The speed could be improved by sorting the points in one direction. The distance check would be faster.

Another reason for this investigation was an unpleasant feature of PostScript random generators: all interpreters deliver different sequences.

For general applications the random generator can be used without distance optimization.

Settings for Acrobat (relevant only for page 4)

Edit / Preferences / General / Page Display (since version 6)
Custom Resolution 72 dpi

Edit / Preferences / General / Color Management (full version only)
sRGB
EuroscaleCoated or ISOCoated or SWOP
GrayGamma 2.2

2. Mathematics

Integers in PostScript are generally handled by 32 bit. The parameter b below exceeds already the 16 bit number space. Higher numbers b can be used but the parameter a has to be optimized (starting with the squareroot of b). Intermediate float accuracy is not expected to be more accurate than by four byte.
The random number generator recursion algorithm was found in [1]. Details are in the PostScript code.
The input seed is automatically converted into the optimized seed (as found by trials).

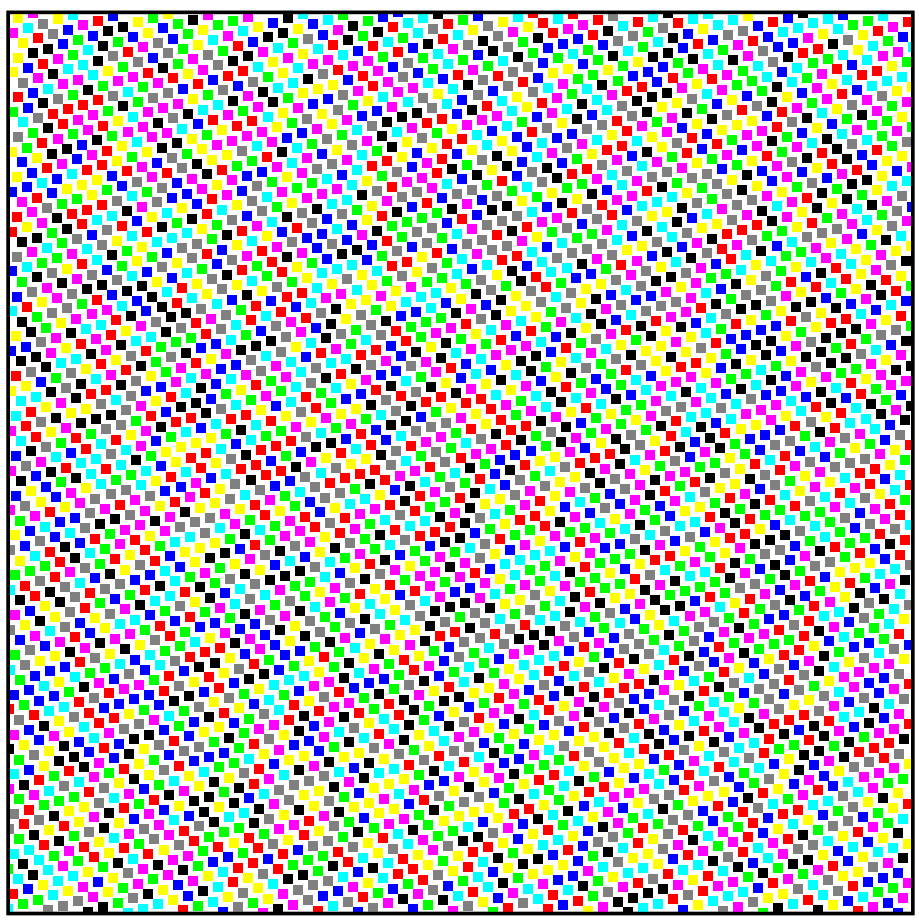
Pseudo random number generator:

$b = 32768$
$a = 181$

Input seed
$x = 0...16383$

Optimized seed
$x = 2x + 1$

Recursion
$x = a\,x - b\,\mathrm{INT}(ax/b)$



The graphic on this page shows 8·512=4096 randomly placed squares with 8 different colors. The area is covered completely and all colors appear. Each square consumes two coordinates. This leads to the conclusion that the period is 8192. It can be confirmed by counting until the first number appears again.

Explanation for chapter 4:

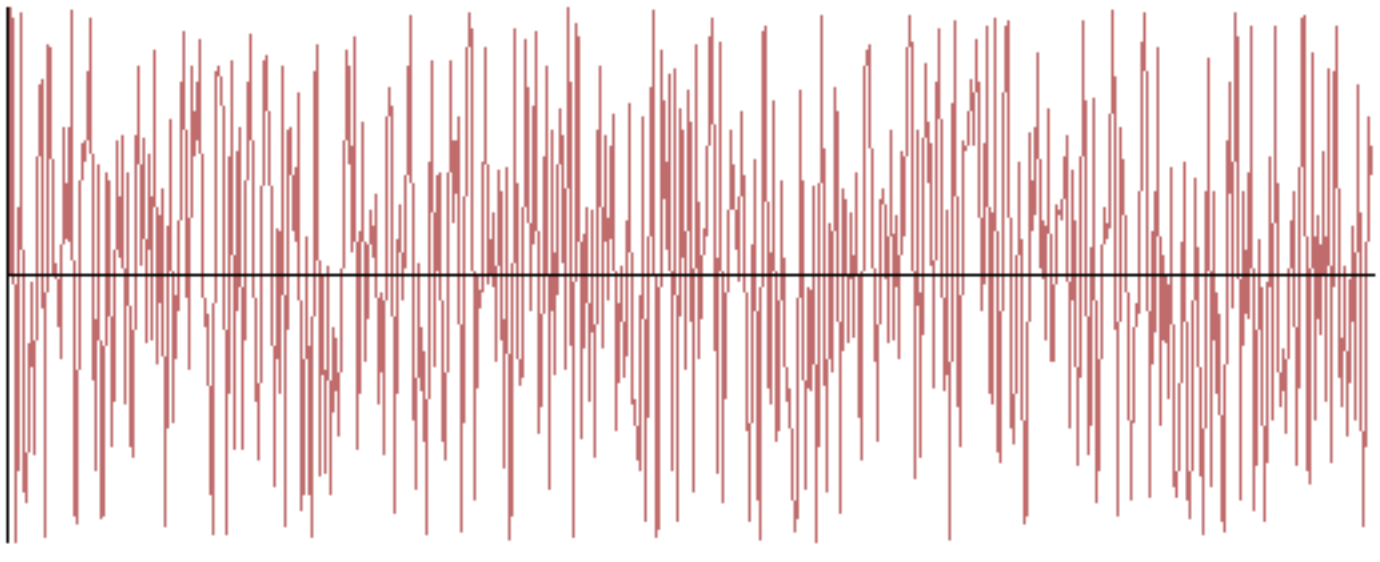
Each of N regularly distributed samples in a unit square occupies a small square with edge length 1/(Sqrt(n), at least for large numbers N.

The squared Euclidian distance along the edge is
$ds^2 = 1/N$

Because of the random variations the distance limit is smaller. We use
$ds^2 = 0.6/N$

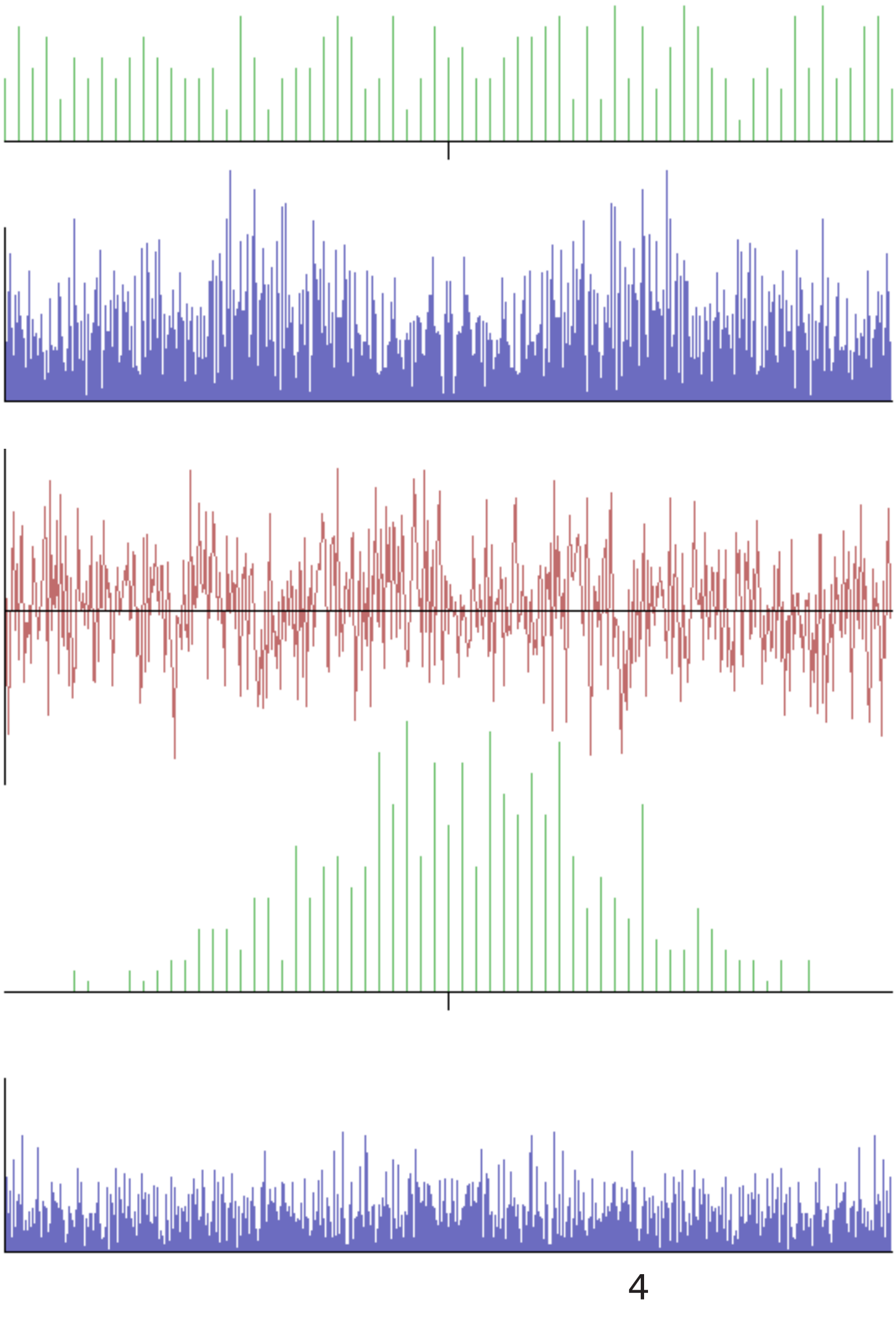
3. Features

The two groups of graphs show 512 sample points (red), the amplitude distribution (green) and the spectrum (blue, mirrored).



Original samples

Zoom 100%



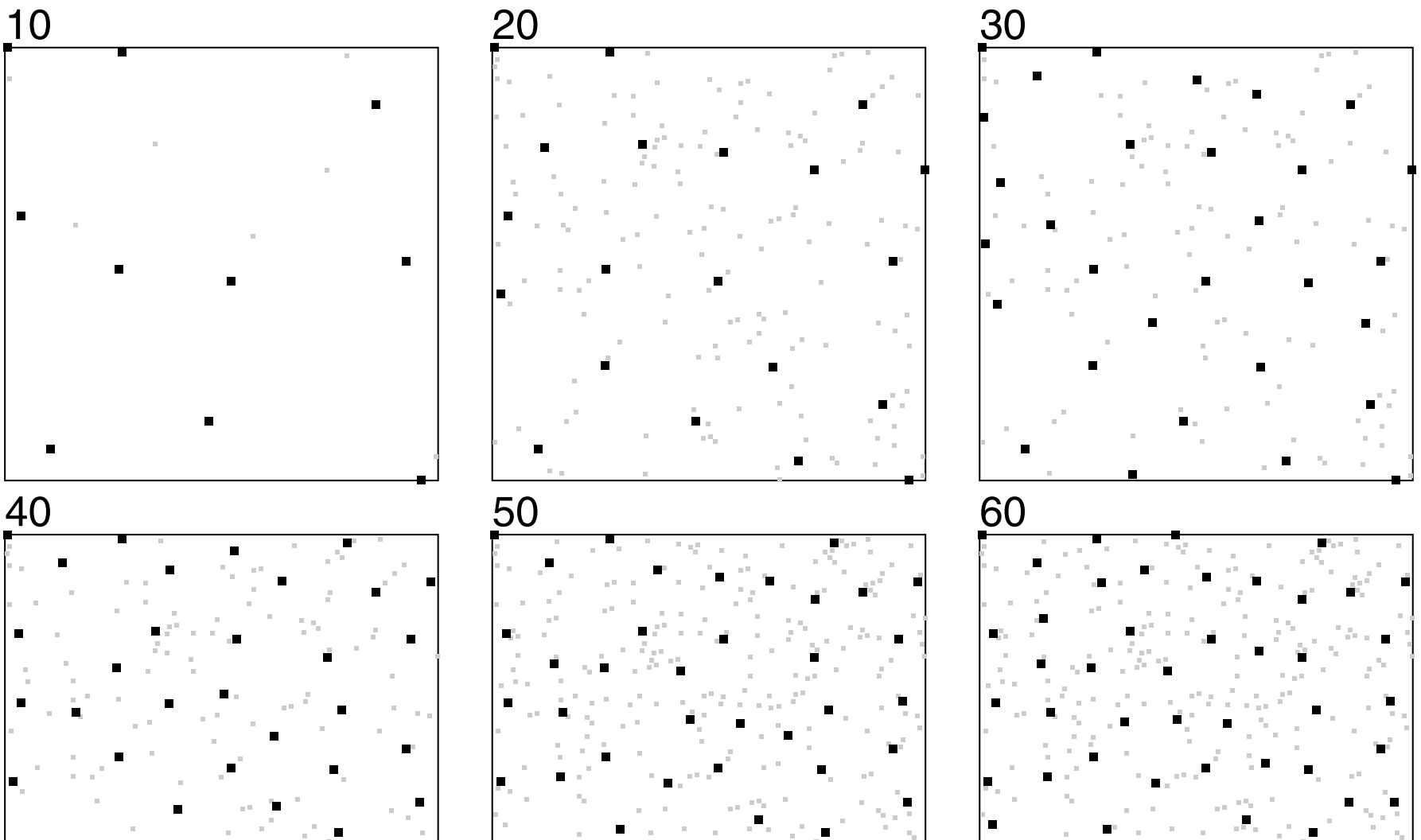
Approximation for a Gaussian distribution by averaging three samples

Zoom 100%

Note:
Statistical parameters for such a small number of samples are never perfect.

4. Examples

The examples were programmed by PostScript. The final points are drawn black. Intermediate points are gray (ignored tested points).



5.1 PostScript Code

```
%!PS-Adobe-3.0   EPSF-3.0
%%BoundingBox:   0 0 842 1191
%%Creator:       Gernot Hoffmann
%%Title:         Random10-120
%%CreationDate:  Febr.12 / 2007

% Random number generator for equal geometrical distribution
% x = a*x - b*Int(a*x/b)

%—Variables

/mm {2.834646 mul} def % points per mm

/rb 2 15 exp def      % 15 bit
/ra 181 def           % optimized for period 8192
/xran 0 def           % seed

/X 8192 array def
/Y 8192 array def

/i  0 def /j  0 def  /k 0 def
/x  0 def /y  0 def
/dx 0 def /dy 0 def
/rx 0 def /ry 0 def
/d1 0 def /r2 0 def
/ns 0 def /nk 0 def
/rand 0 def /em2  0 def /dm2 0 def /d2 0 def

%—Procedures

/Box
{ 0.3 mm sca div setlinewidth
  0 setgray
  newpath  0 0 moveto 1 0 rlineto 0 1 rlineto -1 0 rlineto closepath stroke
} def

/Dot1
{/d1 0.01 def
 /d2 d1 0.5 mul def
  newpath rx d2 sub  ry d2 sub moveto d1 0 rlineto 0 d1 rlineto d1 neg 0 rlineto
  closepath fill
} bind def

/Dot2
{/d1 0.02 def
 /d2 d1 0.5 mul def
  newpath rx d2 sub  ry d2 sub moveto d1 0 rlineto 0 d1 rlineto d1 neg 0 rlineto
  closepath fill
} bind def

/Xrand % random integer rand
{/rand ra rand mul dup rb div cvi rb mul sub round cvi def
  rand
} bind def

/Srand
{/rand exch def
 /rand rand dup add 1 add def
} bind def
```

5.2 PostScript Code

```
/Pattern
{0.80 setgray
/em2 0.6 ns div def % squared distance limit
/rx Xrand rb div def
/ry Xrand rb div def
 X 0 rx put
 Y 0 ry put
 1 1 ns 1 sub
{/nk exch def
 1 1 4096
 { pop
  /rx Xrand rb div def
  /ry Xrand rb div def
   Dot1
  /dm2 999 def
  0 1 nk 1 sub
  {/k exch def
   /ds2 X k get rx sub dup mul Y k get ry sub dup mul add def
    ds2 dm2 lt {/dm2 ds2 def } if
  } for
   dm2 em2 gt {exit} if
 } for
   X nk rx put
   Y nk ry put
} for
0 setgray
0 1 ns 1 sub
{/k exch def
 /rx X k get def
 /ry Y k get def
  Dot2
} for
} bind def

/Oneset
{gsave
 xa ya translate
 sca sca scale
 Box
 xran Srand
 Pattern
/x 0.00 def
/y 1.02 def
 0 setgray
 x y moveto ns buf cvs show
 grestore
} bind def

/sca  80 mm def
/fh 22 sca div def
/Helvetica findfont fh scalefont setfont
/buf 20 string def

/dx 90 mm def
/dy 90 mm def

/xa 20 mm def
/ya 290 mm def
```

5.3 PostScript Code

```
/ns 10 def
 Oneset

/xa xa dx add def
/ns 20 def
 Oneset

/xa xa dx add def
/ns 30 def
 Oneset

/xa 20 mm def
/ya ya dy sub def
/ns 40 def
 Oneset

/xa xa dx add def
/ns 50 def
 Oneset

/xa xa dx add def
/ns 60 def
 Oneset

/xa 20 mm def
/ya ya dy sub def
/ns 70 def
 Oneset

/xa xa dx add def
/ns 80 def
 Oneset

/xa xa dx add def
/ns 90 def
 Oneset

/xa 20 mm def
/ya ya dy sub def
/ns 100 def
 Oneset

/xa xa dx add def
/ns 110 def
 Oneset

/xa xa dx add def
/ns 120 def
 Oneset

showpage
```

6. References

[1] Elmar Schrüfer
Signalverarbeitung
Carl Hanser Verlag München Wien 1990

[2] G.Hoffmann
Speed test by drawing random polygons
http://docs-hoffmann.de/ranspeed07042004.pdf

This doc
http://docs-hoffmann.de/ranpoint14032004.pdf

Gernot Hoffmann
March 14 / 2004 — February 19 / 2013
Website
Load browser / click here