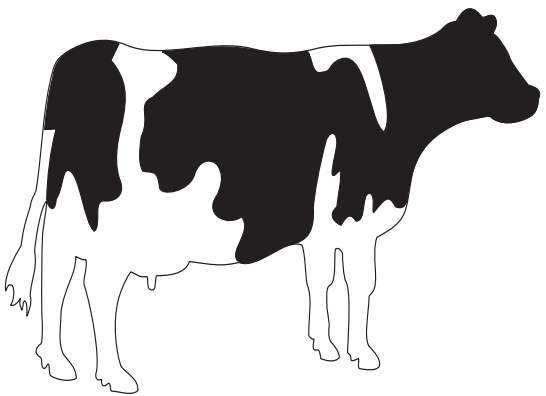
Plotting data with MetaFun/METAPOST

Alan BRASLAU
Service de physique de l'état condensé, CEA/Saclay, France

Brejlov – 15/09/2010

A Dutch cow



A French cow (race Aubrac)

More Aubrac cows

Lozère in winter

Why?

- "A picture is worth a thousand words";
- Data can often be better understood through a graphical presentation;
- A coherent presentation can be made by processing data plots within TEX.

Why?

- "A picture is worth a thousand words";
- Data can often be better understood through a graphical presentation;
- A coherent presentation can be made by processing data plots within TEX.

How?

- METAPOST/graph
- pgf/TiKz pgfgraph
- python/matplotlib
- custom macros
- ...

In MetaFun under ConTEXt, the graph macros are loaded through \usemodule[graph] and the METAPOST code is include within one or multiple instances of \startMPcode \stopMPcode or groups of \startuseMPgraphics \stopuseMPgraphics, etc.

In MetaFun under ConTEXt, the graph macros are loaded through \usemodule[graph] and the METAPOST code is include within one or multiple instances of \startMPcode \stopMPcode or groups of \startuseMPgraphics \stopuseMPgraphics, etc.

A minimal example might be:

```
\usemodule[graph]

\startMPcode
draw begingraph(8cm,6cm) ;
   gdraw (0,0)–(1,1) ;
   endgraph ;
\stopMPcode
```

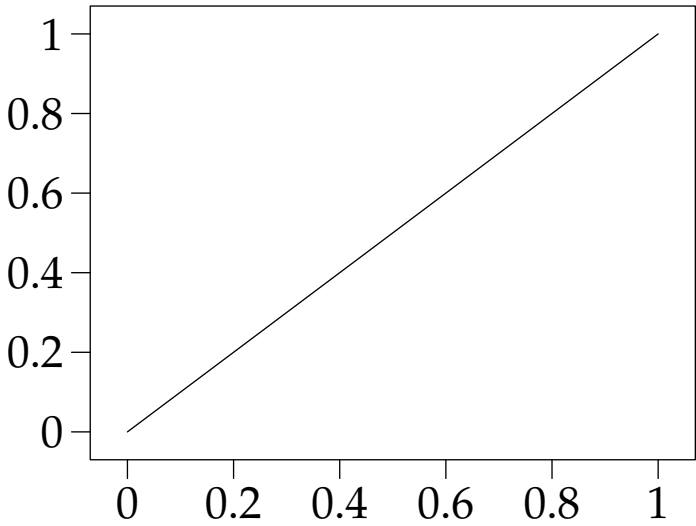


Figure 1 A simple graph

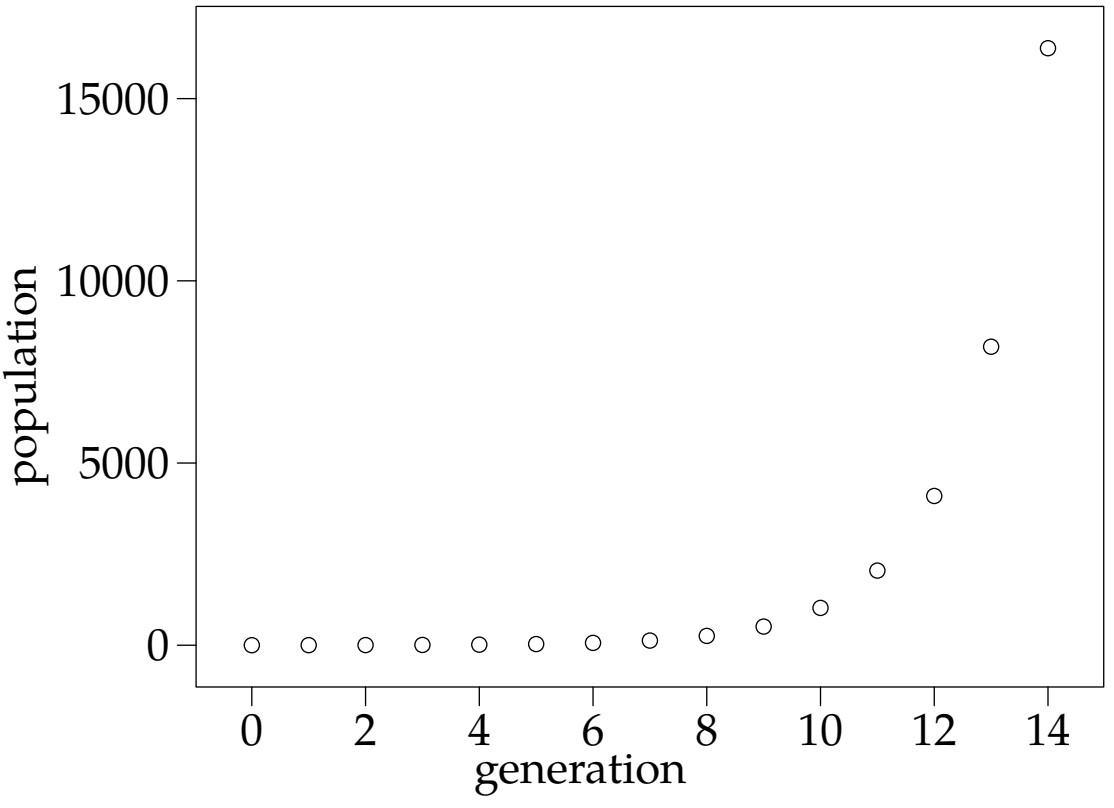
Here, we have drawn the path (0,0)–(1,1) within a frame that is 8 cm by 6 cm ($H \times V$). The range and numbering of the axes are determined automatically.

A more meaningful example

A cell, for example a unicellular organism such as a bacteria, multiplies by a process of binary fission or *mitosis*. The population of bacteria in a culture, under conditions of unlimited growth, doubles with each succeeding generation. This is known as the exponential phase and can be plotted as:

```
\startMPcode
draw begingraph(12cm,9cm) ;
   path p ;
   for g=0 upto 14:
      augment.p(g, 2**g) ;
   endfor
   gdraw p plot image(draw fullcircle scaled 2mm ;) ;
   glabel.bot(btex generation etex, OUT) ;
   glabel.lft(btex population etex rotated 90, OUT) ;
   endgraph ;
\stopMPcode
```

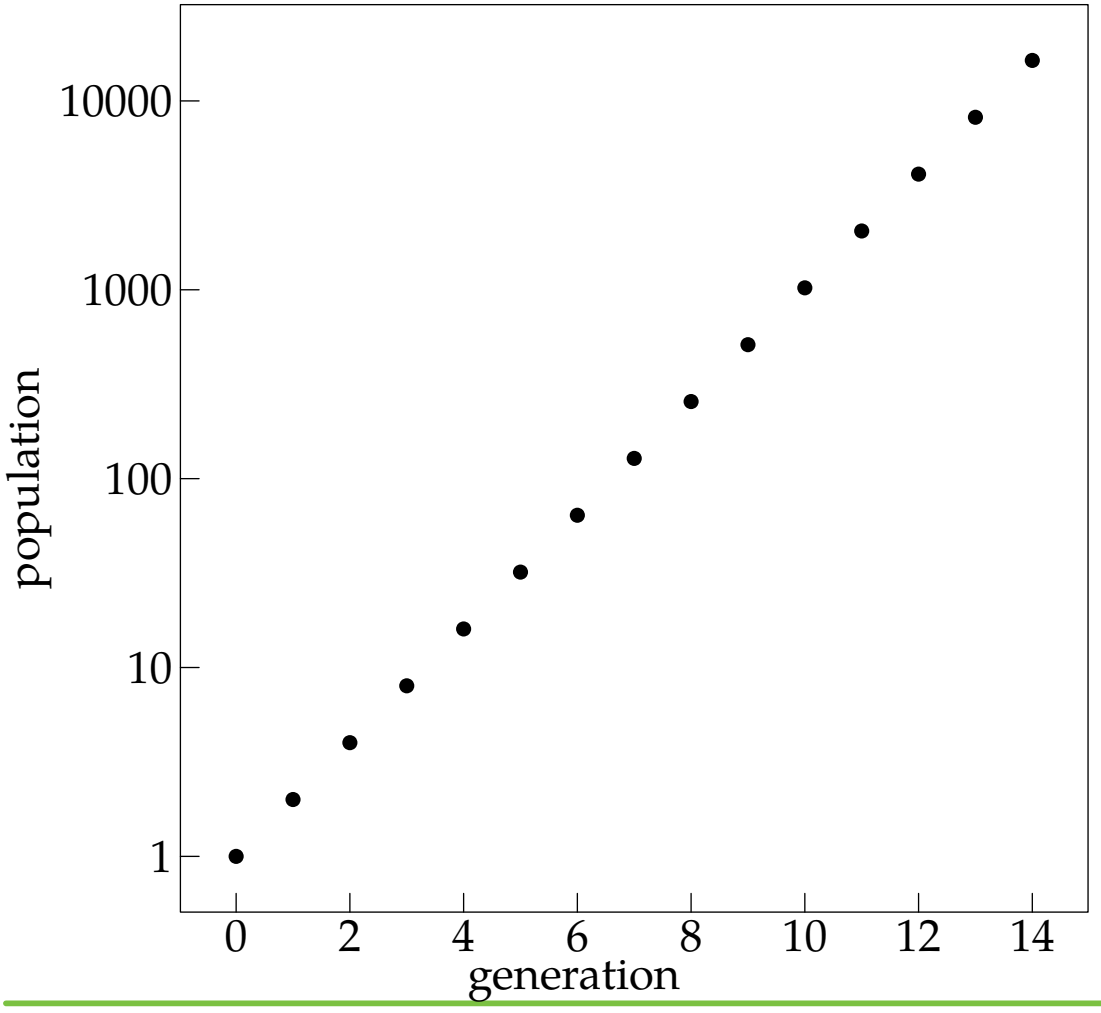
Exponential phase



The exponential growth of a bacterial colony is better represented through a semi-logarithmic scale. This can be selected through the function setcoords():

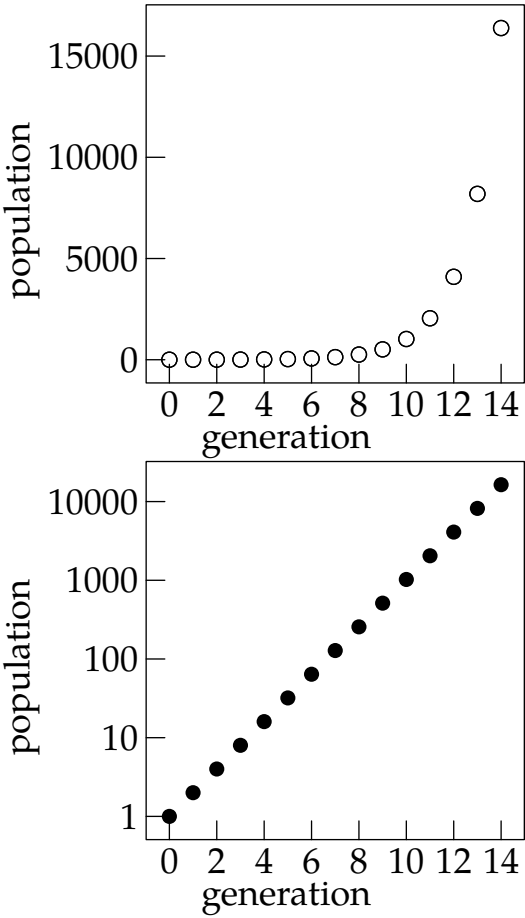
```
\startMPcode
draw begingraph(12cm,12cm) ;
   setcoords(linear,log) ;
   glabel.bot(btex generation etex, OUT) ;
   glabel.lft(btex population etex rotated 90, OUT) ;
   path p;
   for g=0 upto 14:
      augment.p(g, 2**g) ;
   endfor
   gdraw p plot image(fill fullcircle scaled 2mm ;) ;
   autogrid(itick.bot,itick.lft) ;
   endgraph ;
\stopMPcode
```

logarithmic scale



linear vs. logarithmic scale

```
\startMPcode
boolean dolog ; dolog := false ;
draw begingraph(5cm,5cm) ;
    setcoords(linear,if dolog : log
                          else : linear
                                    fi);
    glabel.bot(btex generation etex, OUT);
    glabel.lft(btex population etex
rotated 90, OUT) ;
    path p;
    for g=0 upto 14:
        augment.p(g, 2**g) ;
    endfor
    gdraw p plot image(if dolog : fill
                           else : draw fi
            fullcircle scaled 2mm ;) ;
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```



```
autogrid(itick.bot,itick.lft) ;
```

The function autogrid(x,y) is used to specify that the axes be drawn using inner tic-marks (itick), rather than outer tic-marks (otick) or gridlines (grid).

Since the automatic determination of the graph range does not occur until the data has been plotted, the function autogrid() must be placed *after* the gdraw commands
[unless the range is explicitly set *before* though the function setrange()].

METAPOST uses fixed-point arithmetic based on 32-bit integers, with the decimal point varying between the 16^{th} bit (for numerical values), the 20^{th} bit (for angles), and the 28^{th} bit (for Bézier fractions). This system is quite efficient and sufficient for graphical purposes.
As a consequence, numbers in METAPOST are limited to having an absolute value less than $2^{15} = 32768$ (corresponding to about 4.55 feet or 1.387 m at 600 dpi).

- This limitation causes problems when plotting data that easily exceed this value.
  In the above example, I limited the plot to 14 generations to avoid a "huge number" error.
- Another limitation is that path indices are limited to 12 bits; As any one path can thus contain at most 4096 points (0, . . . , 4095), some care must be taken when plotting a really large set of data.

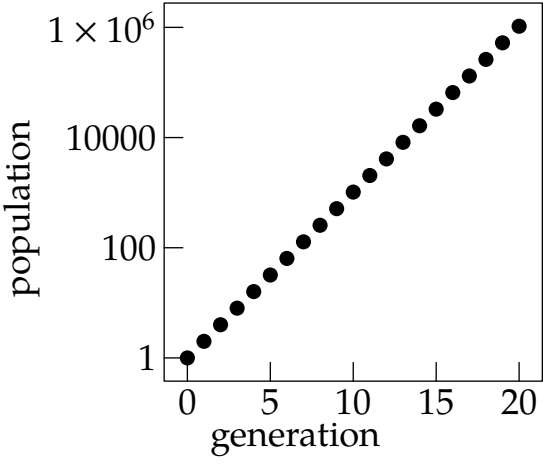
Internally (using the marith macro definitions), the graph macros can represent numbers in "Mlog" form, that is as $\mu^{x \cdot 2^{16}}$ where $\mu = -e^{2^{-24}}$:

> "A number x in Mlog form represents $\exp(x)$ if x is an even multiple of ϵ and $-\exp(x)$ if x is an odd multiple of ϵ, where $\epsilon = 2^{-16}$ [≈1.53e-5] is the basic unit for METAPOST's fixpoint numbers. Such numbers can represent values large as 3.8877e+55 or as small as 1.604e-28 (anything less than that is treated as zero)."

Mlog arithmetic

Internally (using the marith macro definitions), the graph macros can represent numbers in "Mlog" form, that is as $\mu^{x \cdot 2^{16}}$ where $\mu = -e^{2^{-24}}$. This is specified by setting the internal variable Gpaths and using the operator Mlog:

```
\startMPcode
draw begingraph(5cm,5cm) ;
    interim Gpaths := log ;
    setcoords(linear,log) ;
    glabel.bot(btex generation etex,
OUT);
    glabel.lft(btex population etex
rotated 90, OUT) ;
    path p;
    for g=0 upto 20:
        augment.p(Mlog g, g * Mlog 2) ;
    endfor
    gdraw p plot image(fill fullcircle scaled 2mm ;) ;
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```



Another scheme of handling floating-point numbers is through the use of “string” arithmatic macros

input sarith

Calculations using these macros are less efficient than the primative integer arithmatic (in linear or Mlog form), but may be successfully used to reduce large numbers to a smaller range suitable for graphical representation.

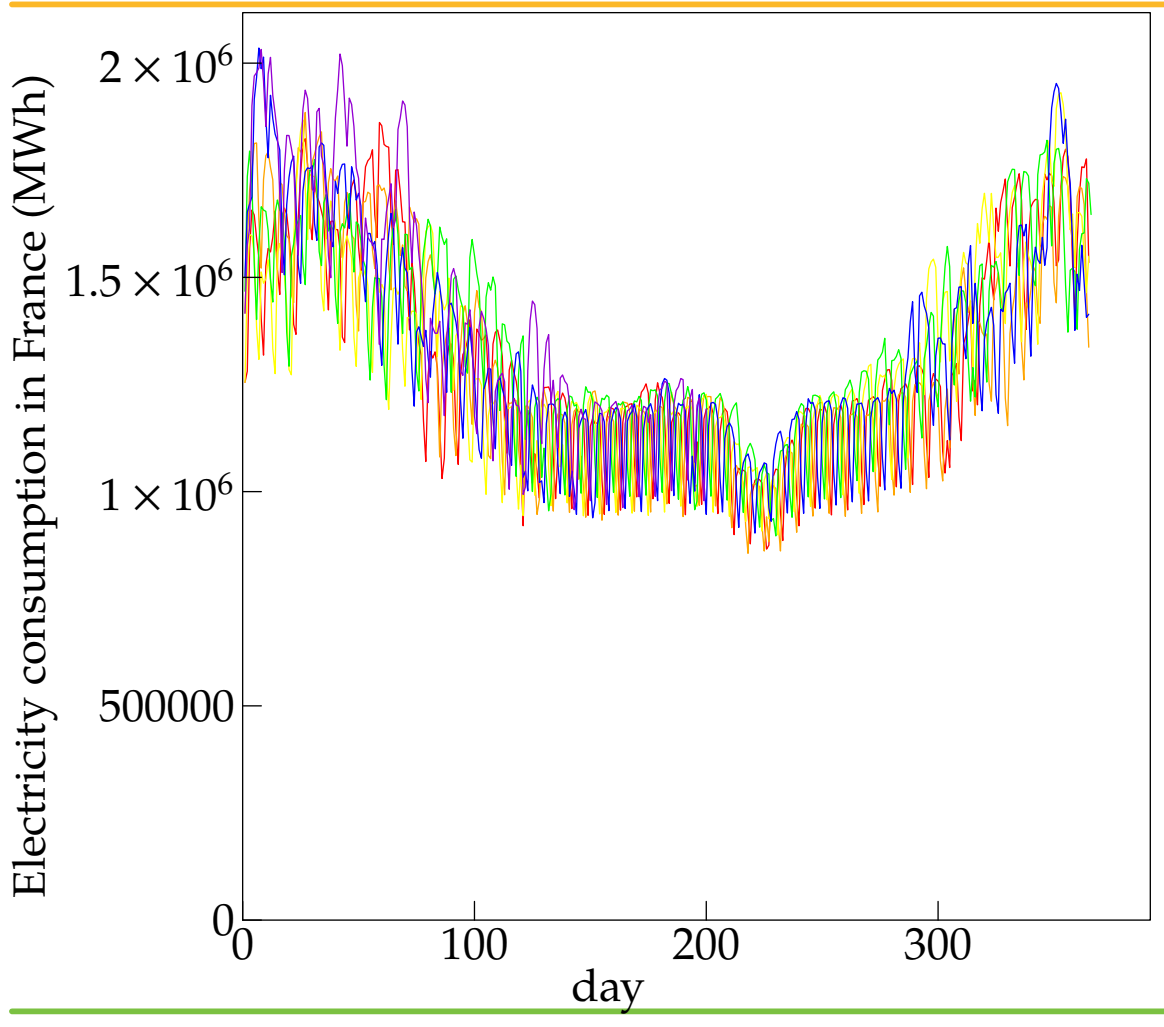
Plotting real data

```
\startMPcode
% standard resistance color code: rainbow sequence
color co[] ;                    string cn[] ;
co0 = \MPcolor{black} ;         cn0 = "black" ;
co1 = \MPcolor{brown} ;         cn1 = "brown" ;
co2 = \MPcolor{red} ;           cn2 = "red" ;
co3 = \MPcolor{orange} ;        cn3 = "orange" ;
co4 = \MPcolor{yellow} ;        cn4 = "yellow" ;
co5 = \MPcolor{green} ;         cn5 = "green" ;
co6 = \MPcolor{blue} ;          cn6 = "blue" ;
co7 = \MPcolor{darkviolet} ; cn7 = "darkviolet" ;
co8 = \MPcolor{gray} ;          cn8 = "gray" ;
co9 = \MPcolor{white} ;         cn9 = "white" ;
\stopMPcode
```

Electricity consumption in France 2005-2010

```
\startMPcode
draw begingraph(12cm,12cm) ;
    interim Gpaths := linear ; % reset to linear
    setcoords(linear,linear) ;
    setrange(0,0,whatever,whatever) ;
    glabel.bot(btex day etex, OUT) ;
    glabel.lft(btex Electricity consumption in France (MWh) etex
        rotated 90, OUT) ;
    for y = 2005 upto 2010 : % year
        gdraw "Historique_consommation_JOUR_" & decimal y & ".d"
            withcolor co[y-2005+2] ;
    endfor
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```

The large numbers contained in the data file are handled internally by the graph macros in a way that is transparent to the user.


2005, 2006, 2007, 2008, 2009 and 2010
Electricity consumption in France (MWh)
2 × 10^6
1.5 × 10^6
1 × 10^6
500000
0
0
100
200
300
day

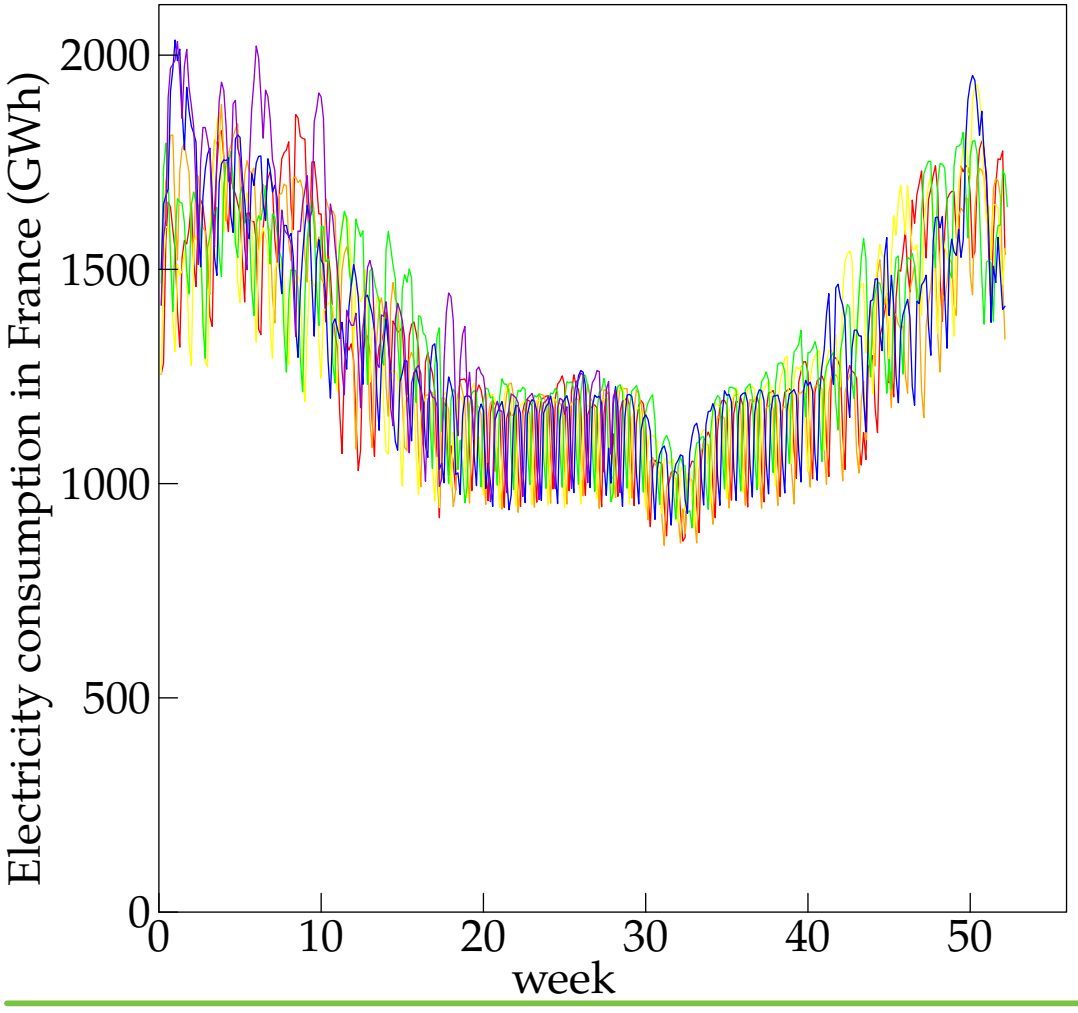
Let's rescale the data, but be careful with large numbers

```
\startMPcode
draw begingraph(12cm,12cm) ;
    setcoords(linear,linear) ;
    setrange(0,0,whatever,whatever) ;
    glabel.bot(btex week etex, OUT) ;
    glabel.lft(btex Electricity consumption in France (GWh) etex
        rotated 90, OUT) ;
    for y = 2005 upto 2010 : % year
        gdraw "Historique_consommation_JOUR_" & decimal y & ".d"
            scaled (1/7, 1/1000) withcolor co[y-2005+2] ;
    endfor
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```

Solution 1: Mlog format

```
\startMPcode
draw begingraph(12cm,12cm) ;
    interim Gpaths := log ;
    setcoords(linear,linear) ;
    setrange(0,0,whatever,whatever) ;
    glabel.bot(btex week etex, OUT) ;
    glabel.lft(btex Electricity consumption in France (GWh) etex
        rotated 90, OUT) ;
    for y = 2005 upto 2010 : % year
        gdraw Mreadpath("Historique_consommation_JOUR_" & decimal y &
".d")
            shifted (-Mlog 7, -3*Mten) withcolor co[y-2005+2] ;
    endfor
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```

Solution 1: Mlog format



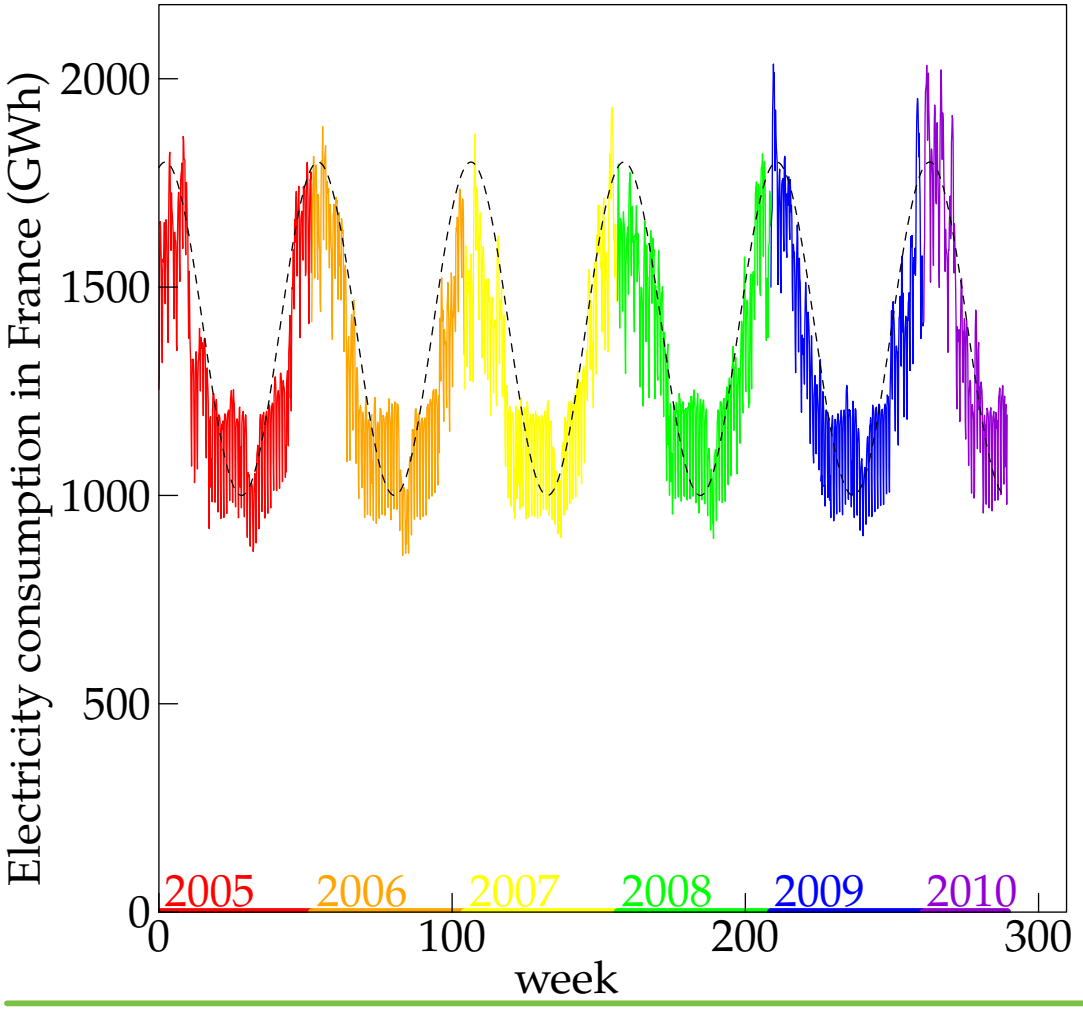
The European commission recently introduced a regulation promoting the use of "energy saving" light bulbs, in particular, prohibiting the sale of certain incandenscent bulbs. This measure is, in fact, anti-ecological, as can be seen in the data of electricity consumption: Every Watt "wasted" in lighting is lost as heat. Therefore, every Watt "economized" when lighting has to be compensated by additional heating, with no net gain. Of course, Europeans heat less in the summer months (or not at all depending on the climate) but the hours of daylight are greater and they use lighting much less in the summer as well. The effect of this measure will be negligable on the yearly consomation of electricity, and appears to be more motivated by the aim of stimulating industry in the production and distribution of a new product, with a probable net ecological *cost* to society.

Solution 2: string arithmetic

```
\startMPcode
input sarith ;
draw begingraph(12cm,12cm) ;
    interim Gpaths := linear ; % reset the data mode.
    setcoords(linear,linear) ;
    setrange(0,0,whatever,whatever) ;
    glabel.bot(btex week etex, OUT) ;
    glabel.lft(btex Electricity consumption in France (GWh) etex
        rotated 90, OUT) ;
    path p[] ;
    d := 0 ; % day
    for y = 2005 upto 2010 : % year
        picture s[] ;
        gdata("Historique_consommation_JOUR_" & decimal y & ".csv",s,
            augment.p[y](d/7, s2 Sdiv "1e3") ;
            augment.p[0](d/7, 400*cosd((d-15)/365*360)+1400) ;
            d := d + 1 ;
        )
        gdraw p[y] withcolor co[y-2005+2] ;
```

```
        % draw a colored year legend:
        gdraw (xpart (point 0 of p[y]), 0)--
            (xpart (point infinity of p[y]), 0)
            withpen pencircle scaled 3pt withcolor co[y-2005+2] ;
        glabel.urt(textext("\color[" & cn[y-2005+2] & "] " & decimal y),
            (y-2005)*52, 0) ;
    endfor
    gdraw p[0] dashed evenly ;
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```

Solution 2: string arithmetic



Weekly electricity usage

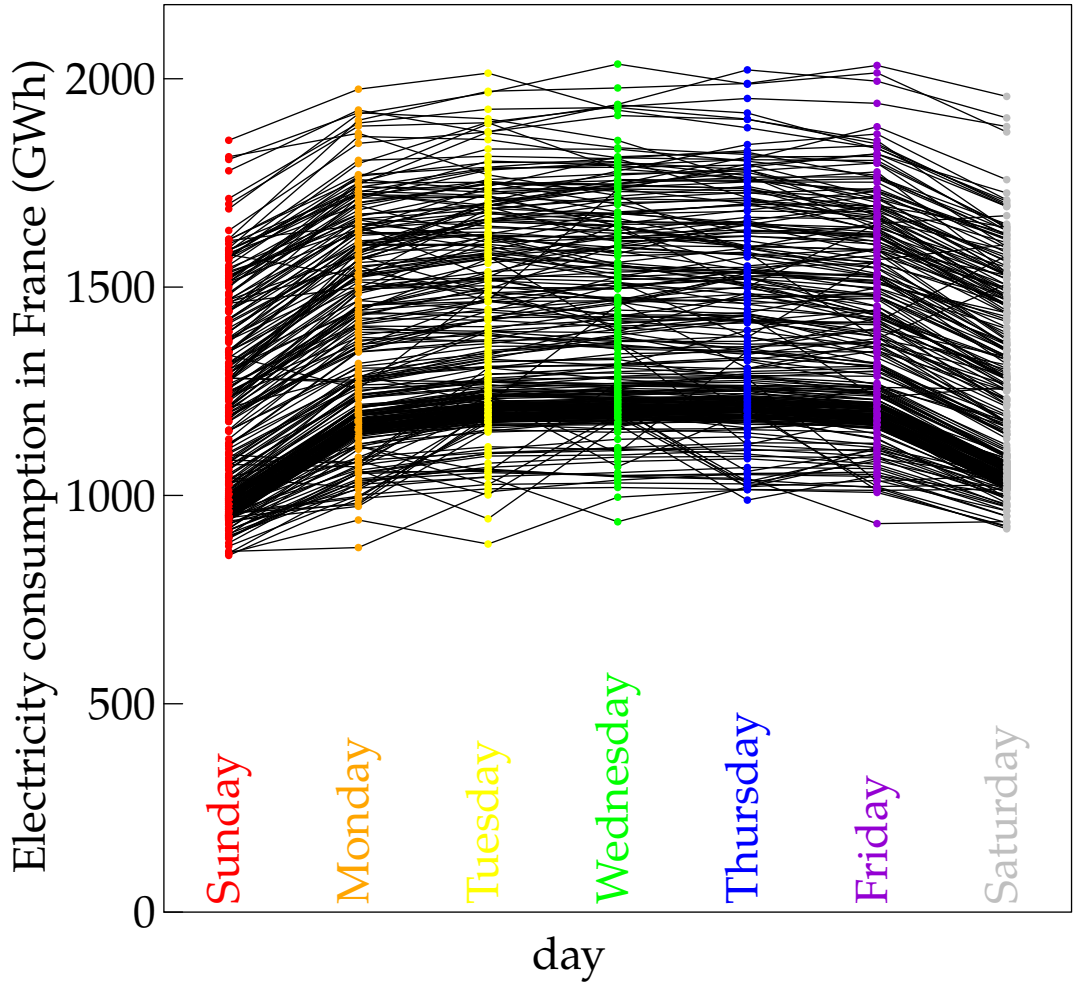
```
\startMPcode
string day[] ;
day0 := "Sunday" ;
day1 := "Monday" ;
day2 := "Tuesday" ;
day3 := "Wednesday" ;
day4 := "Thursday" ;
day5 := "Friday" ;
day6 := "Saturday" ;
\stopMPcode
```

Weekly electricity usage

```
\startMPcode
input sarith ;
draw begingraph(12cm,12cm) ;
    setcoords(linear,linear) ;
    setrange(-.5,0,6.5,whatever) ;
    glabel.bot(btex day etex, OUT) ;
    glabel.lft(btex Electricity consumption in France (GWh) etex
        rotated 90, OUT) ;
    for d=0 upto 6 :
        glabel.top(textext("\color[" & cn[d+2] & "]" & day[d])
            rotated 90,d,0) ;
    endfor
    path p ;
    d := 0 ; % day
    for y = 2005 upto 2010 : % year
        picture s[] ;
        gdata("Historique_consommation_JOUR_" & decimal y & ".csv",s,
            augment.p((d+6) mod 7, s2 Sdiv "1e3") ;
            % 01/01/2005 = Saturday = 6
            d := d + 1 ;
        )
```

```
    endfor
    gdraw subpath (0, 5) of p ;
    for d = 0 upto length(p) :
        if (xpart(point d of p) = 0) : % Sunday
            if (d+6 < length(p)) :
                m := d+6 ;
            else :
                m := length(p) ;
            fi
            gdraw subpath (d, m) of p ;
        fi
        glabel(image(fill fullcircle scaled 1mm
            withcolor co[((d+6) mod 7) + 2] ;), point d of p) ;
    endfor
    autogrid(,itick.lft) ;
    endgraph ;
\stopMPcode
```

Weekly electricity usage



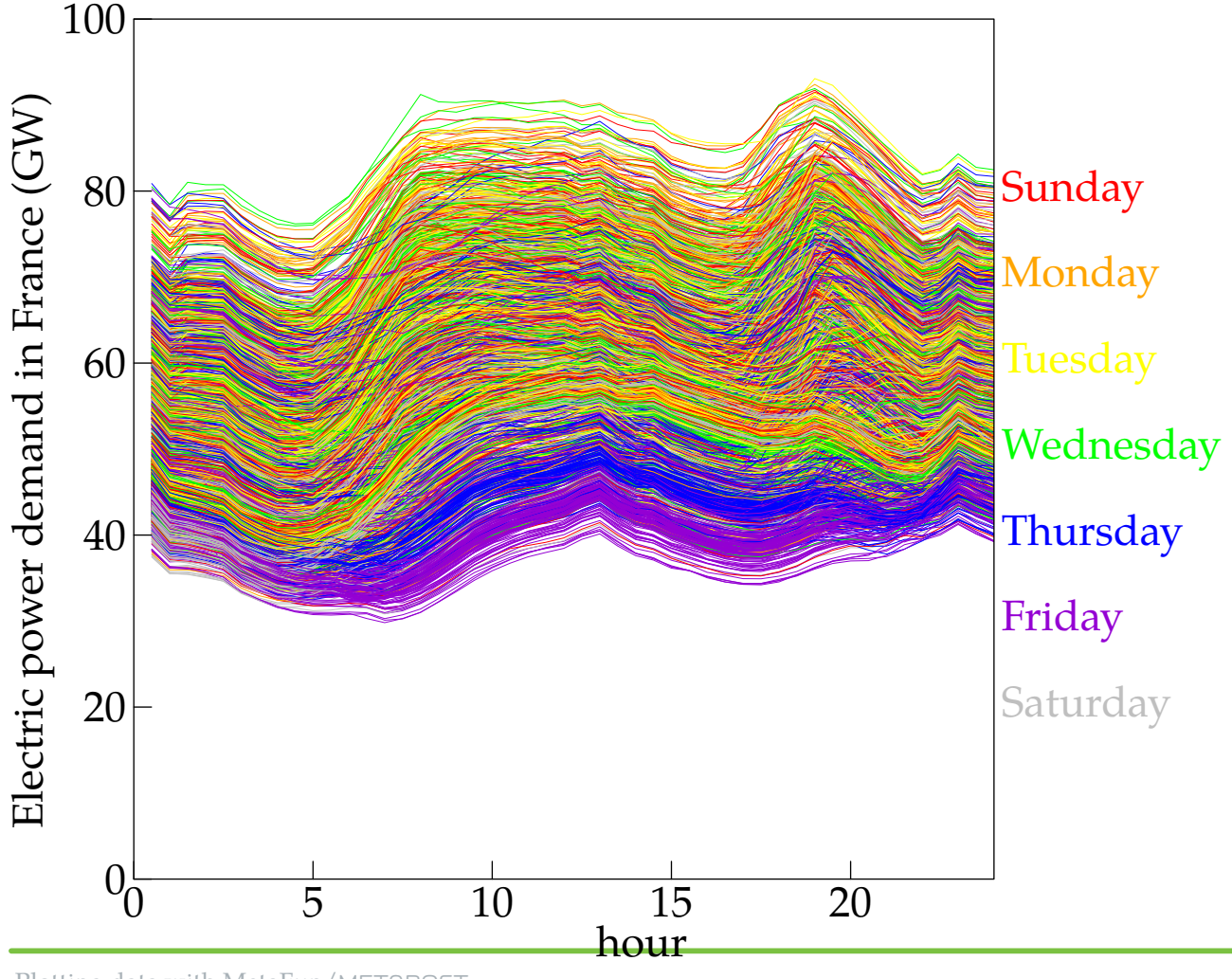
Daily power demand

This data set contains over 150 000 data points!

```
\startMPcode
input sarith ;
draw begingraph(12cm,12cm) ;
    setcoords(linear,linear) ;
    setrange(0,0,24,100) ;
    glabel.bot(btex hour etex, OUT) ;
    glabel.lft(btex Electric power demand in France (GW) etex
        rotated 90, OUT) ;
    for d=0 upto 6 :
        glabel.rt(textext("\color[" & cn[d+2] & "]" & day[d]),24,(8-d)*1
;
    endfor
    path p[] ;
    d := 0 ; % day
    for y = 2005 upto 2010 : % year
        picture s[] ;
        gdata("Historique_consommation_INST_" & decimal y & ".csv",s,
            if (s1 <> "Date") : % 00:30 - 24:00
                for h = 0.5 step 0.5 until 24 :
                    augment.p[d](h, s[2h+1] Sdiv "1e3") ;
```

```
                endfor
                gdraw p[d] withpen pencircle scaled .1pt
                    withcolor co[((d+6) mod 7) + 2] ;
                d := d + 1 ;
            fi
        )
    endfor
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```

Daily power demand



Solution 1: awk

```
\immediate\write18{awk -f INST.awk Historique_consommation_INST_20??.csv
> INST.d}
```

Solution 1: awk INST.awk:

```
{
    if ($1 != "Date") {
        MONTH = 0+substr($1,4,2)
        if (MONTH != LAST) {
            if (NUM > 0) {
                MEAN = SUM/NUM
                print ++CUMUL, MEAN, sqrt(SUMSQ/NUM-MEAN*MEAN)
            }
            LAST = MONTH
            TOTN += NUM ; NUM = 0
            TOTS += SUM ; SUM = 0
            TOTSQ += SUMSQ ; SUMSQ = 0
        }
        for (COLUMN = 2; COLUMN <= NF; COLUMN++) {
            NUM++
            SUM += $COLUMN/1e3
            SUMSQ += $COLUMN * $COLUMN / 1e6
        }
    }
}
```

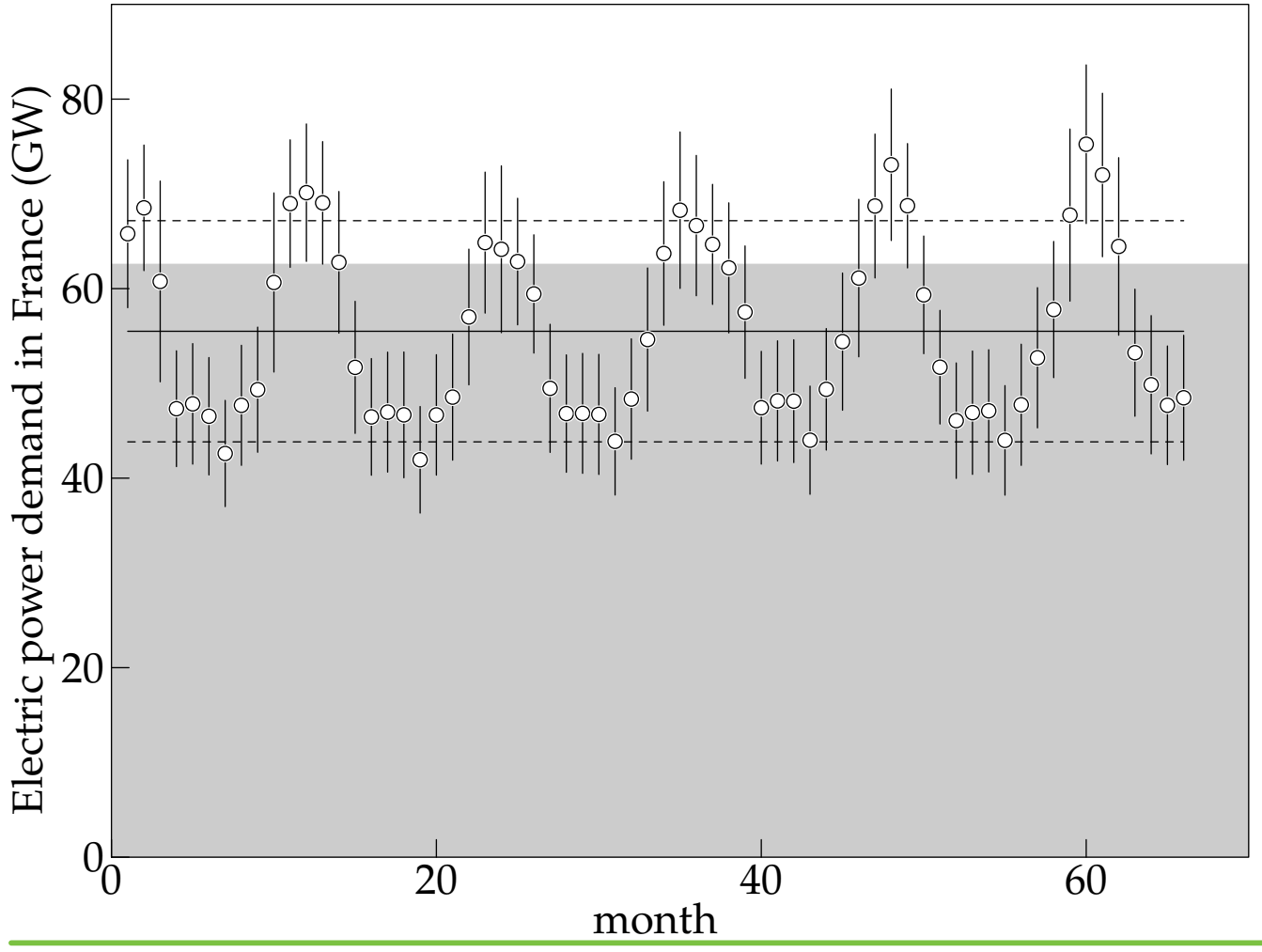
```
END {
    if (NUM > 0) {
        MEAN = SUM/NUM
        print ++CUMUL, MEAN, sqrt(SUMSQ/NUM-MEAN*MEAN)
    }
    if (TOTN > 0) {
        MEAN = TOTS/TOTN
        print "mean", MEAN, sqrt(TOTSQ/TOTN-MEAN*MEAN)
    }
}
```

Monthly average power demand

```
\startMPcode
draw begingraph(16cm,12cm) ;
    setcoords(linear,linear) ;
    setrange(0,0,70,90) ;
    glabel.bot(btex month etex, OUT) ;
    glabel.lft(btex Electric power demand in France (GW) etex
        rotated 90, OUT) ;
    gfill (0,0)--(70,0)--(70,62.58)--(0,62.58)--cycle withcolor .8white
;
    picture sym ; sym := image(unfill fullcircle scaled 2.5mm ;
                               draw fullcircle scaled 2mm ;) ;
    path p ;
    picture s[] ;
    gdata("INST.d",s,
        y1 := scantokens(s2) ;
        e  := scantokens(s3) ;
        if (s1 <> "mean") :
            x1 := scantokens(s1) ;
            augment.p(z1) ;
            gdraw (x1, y1-e)--(x1, y1+e) ;
        fi
```

```
    ) ;
    z0 = point 0 of p ;
    gdraw (x0, y1)--(z1) ; % mean
    gdraw (x0, y1+e)--(x1, y1+e) dashed evenly ; % upper standard error
    gdraw (x0, y1-e)--(x1, y1-e) dashed evenly ; % lower standard error
    gdraw p plot sym ;
    autogrid(itick.bot,itick.lft) ;
    endgraph ;
\stopMPcode
```

Monthly average power demand



80% of electricity in France is of nuclear origin

- 58 nuclear reactors
- 19 different sites
- capacities: 900 MW, 1300 MW or 1495 MW
- total production: 62.58 GW

« chemin d'Ablis »

- 52 windmills, 2 MW/turbine
- 17 km along the A10 highway
- The French electric power demand would need 33037 windmills.
- They would have to be installed along a distance of 10800 km, or along all of the highways in France!
- If installed along the coastline to take advantage of the effect of a thermal breeze, one would need to install 2 continuous rows of windmills to meet this demand!

Windmills

Solution 2: METAPOST 2.0

...

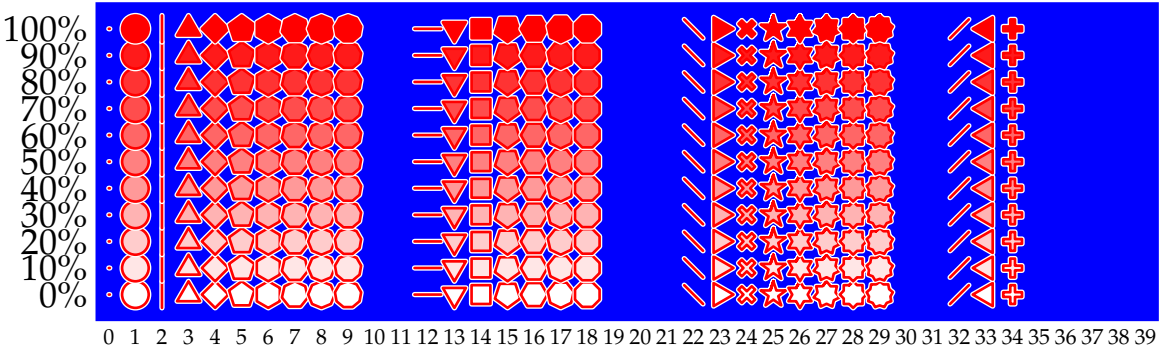
Solution 3: lua

?

What else is missing?

- multi-dimensional data: 3D plots, surface plots, contour plots, image plots, . . .
- curve fitting
- bar plots, pie charts, (and various other *awful* but often used formats of data presentation).

Bonus: plot symbols



Quality plotting symbols are outlined and filled so that overlapping data points remain readable. Here, I define a series of *n*-sided polygons of varying rotations and fill color and shadings:

Bonus: plot symbols (2)

```
\startMPcode
sym_size := fontsize defaultfont ; % can be redefined

pickup pencircle scaled 1pt ;
u := sym_size ;
% canvas
fill unitsquare xscaled 40u yscaled 12u withcolor blue ;
for j = 0 upto 10 :
      for i = 0 upto 39 :
        draw plotsymbol(i,red,j/10) shifted ((i+.5,j+1) scaled u) ;
      endfor
      label.lft(textext("\tfx " & decimal 10j & "\%"), (0, j+1) scaled u) ;
endfor
for i = 0 upto 39 :
      label.bot(textext("\tfxx " & decimal i), (i+.5, 0) scaled u) ;
endfor
\stopMPcode
```

Bonus: plot symbols (3)

```
\startMPcode
def plotsymbol(expr n,c,f) = % (number,color,color|number)
    if known sym_[n] :
        image(
            path p ; p := sym_[n] scaled sym_size ;
            undraw p withpen currentpen scaled 2 ;
            if cycle p : fill p withcolor
                if color f and known f :
                    f
                elseif numeric f and known f and color c and known c :
                    f[background,c]
                elseif numeric f and known f :
                    f[background,black]
                else :
                    background
                fi ;
            fi
            draw p if color c and known c : withcolor c fi ;
        )
    else :
        nullpicture
```

```
    fi
enddef ;
\stopMPcode
```

Bonus: plot symbols (4)

```
\startMPcode
path sym_[] ; % (internal) symbol path
sym_[0] := (0,0) ;                    % point
sym_[1] := fullcircle ;               % circle
sym_[2] := (up -- down) scaled .5 ; % vertical bar
for i = 3 upto 9 :                    % polygons
    sym_[i] := for j = 0 upto i-1 :
        (up scaled .5) rotated (j*360/i) -- endfor cycle ;
endfor
sym_[12] := sym_[2] rotated +90 ;   % horizontal line
sym_[22] := sym_[2] rotated +45 ;   % backslash
sym_[32] := sym_[2] rotated -45 ;   % slash
sym_[13] := sym_[3] rotated 180 ;   % down  triangle
sym_[23] := sym_[3] rotated -90 ;   % right triangle
sym_[33] := sym_[3] rotated +90 ;   % left  triangle
sym_[14] := sym_[4] rotated +45 ;   % square
sym_[15] := sym_[5] rotated 180 ;   % down pentagon
sym_[16] := sym_[6] rotated +90 ;   % turned hexagon
sym_[17] := sym_[7] rotated 180 ;
sym_[18] := sym_[8] rotated +22.5 ;
for j = 5 upto 9 :
```

```
    l := length(sym_[j]) ;
    pair p[] ;
    for i = 0 upto l :
        p[i] = whatever [point i             of sym_[j],
                         point (i+2   mod l) of sym_[j]] ;
        p[i] = whatever [point (i+1   mod l) of sym_[j],
                         point (i+l-1 mod l) of sym_[j]] ;
    endfor
    sym_[20+j] := for i = 0 upto l : point i of sym_[j]--p[i]--endfor
cycle ;
endfor
path s, q ; s := sym_[4] ; q := s scaled .25 ;
l := length(s) ;
pair p[] ;
sym_[24] := for i = 0 upto l-1 :
     hide(
         p[i]   = whatever [point i   of s, point (i+1   mod l) of s] ;
         p[i]   = whatever [point i   of q, point (i-1+l mod l) of q] ;
         p[i+l] = whatever [point i   of s, point (i+1   mod l) of s] ;
         p[i+l] = whatever [point i+1 of q, point (i+2   mod l) of q] ;
     )
```

```
      point i of q -- p[i] -- p[i+1] --
endfor cycle ;
sym_[34] := sym_[24] rotated +45 ;
\stopMPcode
```

Bonus+: error bars